Preface

We were pleased to present this CEUR-WS volume, the Proceedings of the 8th Bayesian Modeling Applications Workshop (BMAW-11), held in Barcelona, Spain on July 14, 2011, as a workshop of the 27th Conference on Uncertainty in Artificial Intelligence (UAI 2011).

Bayesian networks are now a powerful, well-established technology for reasoning under uncertainty, supported by a wide range of mature academic and commercial software tools. They are now being applied in many domains, including environmental and ecological modelling, bioinformatics, medical decision support, many types of engineering, robotics, military, financial and economic modelling, education, forensics, emergency response, surveillance, and so on. This workshop, the eighth in the series of workshops focusing on real-world applications of Bayesian networks, provides a focused, informal forum for discussion and interchange between researchers, practioners and tool developers.

This year we encouraged the submission of papers addressing the workshop theme *Knowledge Engineering*, which we use as a general term that includes expert elicitation, learning from data, taking existing models from the literature, and any hybrids of these. Authors have been encouraged to describe the knowledge engineering process used to build their application, along with the pitfalls encountered and lessons learned.

Papers in the workshop also address the practical issues involved in developing real-world applications, such as knowledge engineering methodologies, elicitation techniques, evaluation, and integration methods, with some describing software tools developed to these support these activities. Some papers describe stand-alone Bayesian networks, while others describe applications where the Bayesian neworks are embedded in a larger software system.

This year all submissions were full length papers peer-reviewed by at least two reviewers. We were very pleased with the quality and number of the 25 submission received, with 16 papers accepted and appearing in these proceeding (12 accepted for oral presentation, 4 for poster presentation). Authors have been encouraged to present demonstrations during the poster and demo session. In addition, this year the workshop includes a panel session on the topic "The challenges for developing and deploying applications in the real world".

We are grateful to all the program committee members for their outstanding work in refereeing the papers within a very tight schedule, plus their assistance with organisational matters. We also appreciate the generous financial and organizational support of the main UAI conference. In particular, we thank the UAI 2011 conference general chair, Peter Grünwald, the program co-chairs Avi Pfeffer and Fabio Cozman, and the local arrangements chair Lluis Godo.

Ann Nicholson
Workshop Chair
July 2011

Program Committee

John Mark Agosta	Impermium.com, USA (Emeritus Chair)
Russell Almond	Florida State University, USA
John Bromley	Oxford University Centre for the Environment (OUCE), UK
Dennis Buede	Innovative Decisions, Inc., USA
Luis de Campos	University of Granada, Spain
Marek Druzdzel	University of Pittsburgh, USA, Bialystok University of Technology, Poland
Julia Flores	Universidad de Castilla - La Mancha, Spain
Judy Goldsmith	University of Kentucky, USA
James Jones	Science Applications International Corporation (SAIC), USA
Lionel Jouffe	Bayesia, France
Oscar Kipersztok	Boeing Research & Technology, USA
Kevin Korb	Monash University, Australia
Helge Langseth	Norwegian University of Science and Technology, Norway
Kathryn Laskey	George Mason University, USA
Anders Madsen	HUGIN EXPERT, Denmark
Suzanne Mahoney	Innovative Decisions Inc, USA
Jose-Luis Molina	Polytechnic University of Valencia, Spain
Ann Nicholson	Monash University, Australia
Thomas Dyhre Nielsen	Aalborg University, Denmark
Agnieszka Onisko	Bialystok University of Technology, Poland, University of Pittsburgh, USA
Olivier Pourret	Electricité de France, France
Silja Renooij	Utrecht University, The Netherlands
Carl Smith	University of Queensland, Australia
Jim Smith	Warwick University, UK
L. Enrique Sucar	INAOE, Mexico
Charles Twardy	George Mason University, USA

Additional Reviewers

We would like to thank the following additional reviewers:

Mihail, Paul
Singliar, Tomas

Table of Contents

Real-time event recognition from video via a “bag-of-activities”

Rolf H. Baxter **Neil M. Robertson** **David M. Lane**

Heriot-Watt University
Edinburgh, Scotland, EH14 4AS

Abstract

In this paper we present a new method for high-level event recognition, demonstrated in real-time on video. Human behaviours have underlying activities that can be used as salient features. We do not assume that the exact temporal ordering of such features is necessary, so can represent behaviours using an unordered “bag-of-activities”. A weak temporal ordering is imposed during inference, so fewer training exemplars are necessary compared to other methods. Our three-tier architecture comprises low-level tracking, event analysis and high-level recognition. High-level inference is performed using a new extension of the Rao-Blackwellised Particle Filter. We validate our approach using the PETS 2006 video surveillance dataset and our own sequences. Further, we simulate temporal disruption and increased levels of sensor noise.

1 INTRODUCTION

Considerable attention has been given to the detection of events in video. These can be considered low-level events and include agents entering and exiting areas (Fusier et al., 2007), and object abandonment (Grabner et al., 2006). High-level goals have been recognised from none-visual data sources with reasonable success (Liao et al., 2007). However, there has been far less progress towards recognising high level goals from low-level video.

Detecting events from surveillance video is particularly challenging due to occlusions and lighting changes. False detections are frequent, leading to a high degree of noise for high-level inference. Although complex events can be specified using semantic models, they are largely deterministic and treat events as facts (e.g. (Robertson et al., 2008)). Mechanisms for dealing with observation uncertainty are unavailable in these models (Lavee et al., 2009). On the other hand, probabilistic models are very successful in noisy environments, and are at the core of our approach.

Plan recognition researchers such as (Bui and Venkatesh, 2002; Nguyen et al., 2005) used hierarchical structures to model human behaviour. By decomposing a goal into states at different levels of abstraction (e.g. sub-goals, actions), a training corpus can be used to learn the probability of transitioning between the states. Although this work does consider video, a major shortfall is the necessity for training data, which is often unavailable in surveillance domains.

A common way to avoid this issue is to model “normal” behaviours for which training data is easier to obtain (Boiman and Irani, 2007; Xiang and Gong, 2008). Activities with a low probability can then be identified as abnormal. Because semantic meanings cannot be attached to the abnormal activities, they cannot be automatically reasoned about at a higher level, nor explained to an operator.

Another alternative to learning temporal structure is to have it defined by an expert. For simple events this is trivial, but increases at least proportionally with the complexity of the event. In (Laxton et al., 2007) the Dynamic Belief Network for making French Toast was manually specified. Their approach only considers a single goal.

Dee and Hogg showed that interesting behaviour can be identified using motion trajectories (Dee and Hogg, 2004). Their model identified regions of the scene that were visible or obstructed from the agent’s location, and produced a set of goal locations that were consistent with the agent’s direction of travel. Goal transitions were penalised so irregular behaviours were identified via their high-cost.

In (Baxter et al., 2010) a simulated proof of concept suggested behaviours could be identified using temporally unordered features. This has the advantage that training exemplars are not required. Our work furthers the idea that complex behaviour can be semantically recognised using a feature-based approach. We present methods for representing behaviours, performing efficient inference, and demonstrate validity and scalability on real, multi-person video.

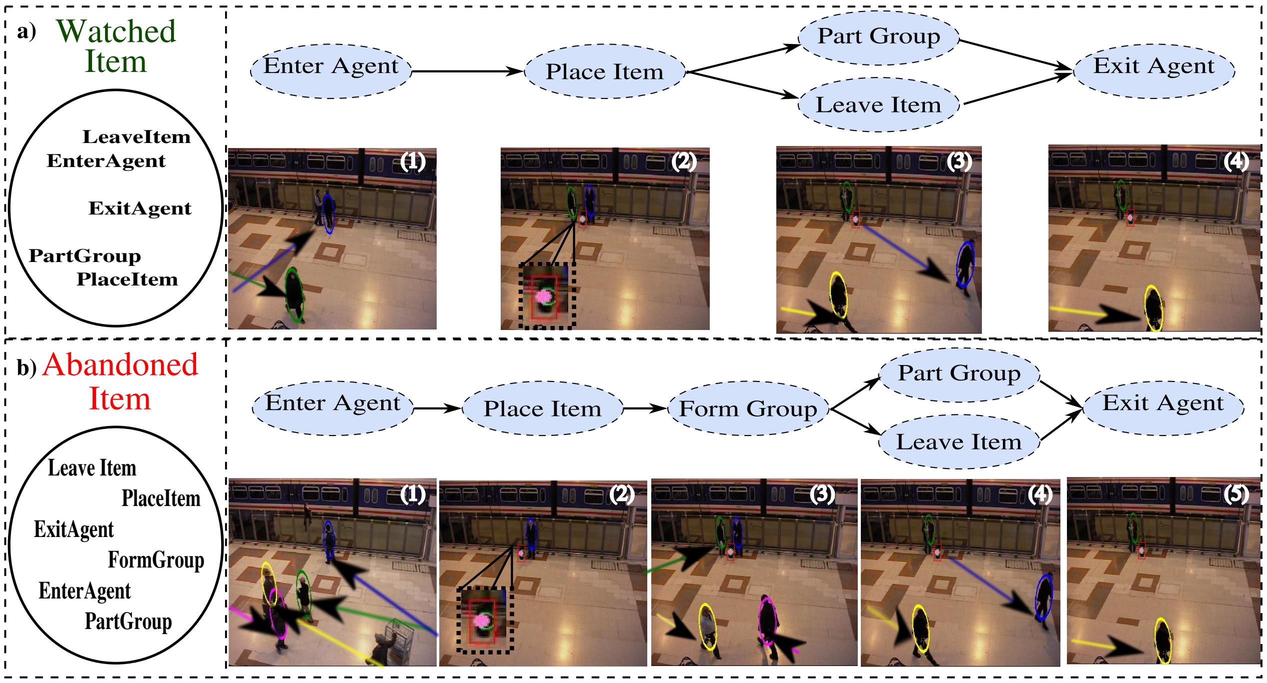


Figure 1: (a) When two agents enter together, an item left by one agent is not a threat when the second agent remains close. (b) When two agents enter separately, it cannot be assumed that the item is the responsibility of the remaining agent

This paper presents a framework with three major components : (1) low-level object detection and tracking from video; (2) detecting and labelling simple visual events (e.g. object placed on floor), and (3) detecting and labelling high-level, complex events, typically including multiple people/objects and lasting several minutes in duration. Our high-level inference algorithm is based upon the Rao-Blackwellised Particle Filter (Doucet et al., 2000a), and can recognise both concatenated and switched behaviour. Our entire framework is capable of real-time inference.

We validate our approach chiefly on real, benchmarked surveillance data: the PETS 2006 video surveillance dataset. We report classification accuracy and speed on four of the original scenarios, and one additional scenario. The fifth scenario was acquired by merging frames from different videos to provide a complex, yet commonly observed behaviour. Further evaluation is conducted by simulating sensor noise and temporal disruption, and on additional video recorded in our own vision laboratory.

Throughout this paper the term activity is used to refer to a specific short-term behaviour that achieves a purpose. An activity is comprised of any number of atomic actions. Activities are recognised as simple events. These terms are interchanged depending upon context. Similarly, collections of activities construct goals, and will be referred to as features of that goal. Goals are detected as complex events.

2 RECOGNITION FRAMEWORK

Figure 1 illustrates two complex behaviours: *Watched Item* and *Abandoned Item*. *Watched Item* involves two persons who enter the scene together. One person places an item of luggage on the floor and leaves, while the other person remains in close proximity to the luggage. This scenario is representative of a person being helped with their bags. *Abandoned Item* is subtly different: the two people do not enter the scene together (Frames 1 and 3 in Figure 1b).

Traditionally, the proximity of people to their luggage is used to detect abandonment. This would generate an alert for both of the above scenarios. To distinguish between them, we integrate low-level image processing with high-level reasoning (Figure 2). We use a hierarchical, modular framework to provide an extendible system that can be easily updated with new techniques. Video data is provided as the source of observations and is processed at three different levels: Object Detection and Tracking, Simple Event Recognition, and Complex Event Recognition. Image processing techniques provide information about objects in the scene, allowing simple semantic events to be detected. These then form observations for high-level recognition.

2.1 OBJECT DETECTION AND TRACKING

Static cameras allow foreground pixels to be identified using background subtraction. This technique compares the current frame with a known background frame. Pixels that

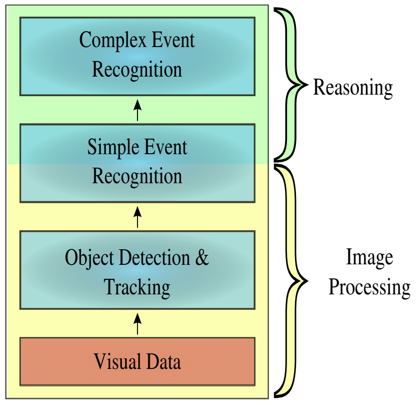


Figure 2: Our architecture for complex event recognition

are different are classed as the foreground. Connected foreground pixels give foreground blobs, and are collectively referred to as B_t. The size/location of each blob can be projected onto real-world coordinates using the camera calibration information. Two trackers operate on B_t.

Person Tracker: Our person tracker consists of a set of SIR filters (Gordon et al., 1993). SIR filters are similar to Hidden Markov Models (HMMs) in that they determine the probability of a set of latent variables given a sequence of observations (Rabiner, 1989). However, when latent variables are continuous, exact approaches to inference become intractable. The SIR filter is an approximation technique that uses random sampling to reduce the state space.

Our filters consist of one hundred particles representing the person's position on the ground plane, velocity, and direction of travel (Limprasert, 2010). For each video frame, the blobs (groups of foreground pixels) that contain people are quickly identified from B_t using ellipsoid detection. We denote these blobs E_t. For each ellipsoid that cannot be explained by an existing filter, a new filter is instantiated to track the person.

In order to address the temporary occlusion of a person (e.g. people crossing paths), particles also contain a visibility variable (0/1) to indicate the person's disappearance. This variable applies to all particles in the filter. By combing this variable with a time limit, the filter continues to predict the person's location for short occlusions, while longer occlusions will cause the track to be terminated.

Object Tracker: Our second tracking component consists of an object detector. In the video sequences this detects luggage and is similarly heuristic to other successful approaches (Lv et al., 2006). To remove person blobs and counteract the effect of lighting changes, which spuriously create small foreground blobs, the tracker eliminates blobs that are not within the heuristically defined range: $0.3 \leq width/height \leq 1m$. Each remaining blob is classified as a stationary luggage item if the blob centroid remains within 30cm of its original position, and is present for at least 2 continuous seconds. The red rectangle identifies a tracked luggage item in Figures 1a&b, frame 2. Inversely, if the blob matching a tracked luggage object cannot be identified for 1 second, the luggage is classed as "removed". To prevent incorrect object removal (e.g. when a person is occluding the object), the maximum object size constraint is suspended once an object is recognised.

2.2 SIMPLE EVENT RECOGNITION

Simple events can be generated by combining foreground detection/tracking with basic rules. Table 1 specifies the set of heuristic modules used in our architecture to encode these rules. It should be highlighted that the *GroupTracker* only uses proximity rules to determine group membership (we suggest improvements in Future Work). *Group Formed* events are trigged when two people approach, and remain within close proximity of each other. Inversely, *GroupSplit* events are triggered when two previously "grouped" people cease being in close proximity.

Although these naive modules achieve reasonable accuracy on the PETS dataset, it is important to acknowledge that they would be insufficient for more complex video. The focus of our work is high-level inference and thus state-of-the-art video processing techniques may not have been used. The modularity of our framework allows any component to be swapped, and thus readily supports the adoption of improved video processing algorithms. Furthermore, we demonstrate via simulation that high-level inference remains robust to increased noise.

2.3 COMPLEX EVENT RECOGNITION

Human behaviour involves sequential activities, so it is natural to model them using directed graphs as in Figure 1. Dynamic Bayesian Networks (Figure 3) are frequently chosen for this task, where nodes represent an agent's state, solid edges denote dependence, and dashed edges denote state transitions between time steps (Murphy, 2002). Each edge has an associated probability which can be used to model the inherent variability of human behaviour [1].

Like many others, (Bui and Venkatesh, 2002) learnt model probabilities from a large dataset. However, annotated libraries of video surveillance do not exist for many interesting behaviours, making there no clear path for training high-level probabilistic models. Similar problems occur when dealing with military or counter-terrorism applications, where data is restricted by operational factors. Alternative approaches include manually specifying the probabilities, and using a distribution that determines when transitions are likely to occur (Laxton et al., 2007).

We hypothesise that many human behaviours can be recog-

[1] Figure 3 will be fully explained in section 3

nised without modelling the exact temporal order of activities. This means that model parameters do not need to be defined by either an expert, or training exemplar. We consider activities as salient features that characterise a behaviour. Goals can be recognised by combining a collection (bag) of activities with a weak temporal ordering.

Feature based recognition algorithms have primarily been developed for object detection applications. To identify features that are invariant to scale and rotation, object images are often transformed into the frequency or scale domains, where invariant salient features can be more readily identified (Lowe, 1999). The similarities between recognising objects and human behaviours has previously been noted (Baxter et al., 2010; Patron et al., 2008), and it is this similarity upon which we draw our inspiration.

Figure 1 helps visualise a behaviour as a set of features. Each ellipse represents a complex event as a bag of activities (cardinality: one). We formally denote a bag by T, the **T**arget event, where each element is drawn from the set of detectable simple events α. Each simple-event is a feature.

The agents progress towards a target event can be monitored by tracking the simple events generated. Fundamentally, the simple events should be consistent with T if T correctly represents the agent's behaviour. For instance, if simple event α^i is observed but $\alpha^i \ni T$, then α^i must be a false detection, or T is not the agent's true behaviour.

As time increases more events from T should be generated. If we make the assumption that each element of a behaviour is only performed once, then the set of expected simple events reduces to the elements of T not yet observed. If $T = \langle \gamma, \delta, \epsilon \rangle$ and γ has already been observed, then the set of expected events is $\langle \delta, \epsilon \rangle$. In this way a weak temporal ordering can be applied to the elements of T without learning their absolute ordering from exemplar.

If C is defined as the set of currently observed simple events, $T \backslash C$ is the set of expected events. At each time step, events in $T \backslash C$ have equal probability, while all other events have 0 probability. This probability distribution encapsulates the assumption that each simple event is only truthfully generated once per behaviour, and is consistent with other work in the field (Laxton et al., 2007). We discuss the implications and limitations of this assumption in section 5.

Worked Example: Using Figure 1's *Watched Item* behaviour as an example, at time step t=0 each of the 5 events (LeaveItem, EnterAgent, ExitAgent, PartGroup, PlaceItem) has equal probability. In frame 1 ($t = 1$), $p(EnterAgent) = 0.2$. At $t = 2$, $p(EnterAgent) = 0$, while $\forall i \in T \backslash C : p(i) = 0.25$. Note that $p(FormGroup|T = WatchedItem) = 0$ at all time steps, because *FormGroup* $\ni$ *WatchedItem.*

Table 1: The simple event modules used by our architecture

Module	Description
Agent Tracker	Detects the entry/departure of people from the scene.
Object Tracker	Upon luggage detection, associates that luggage with the closest person.
Group Tracker	Identifies when people are in close proximity, and split from a single location.
Abandoned Object Detector	Detects when luggage is ≥ 3 metres from its owner.

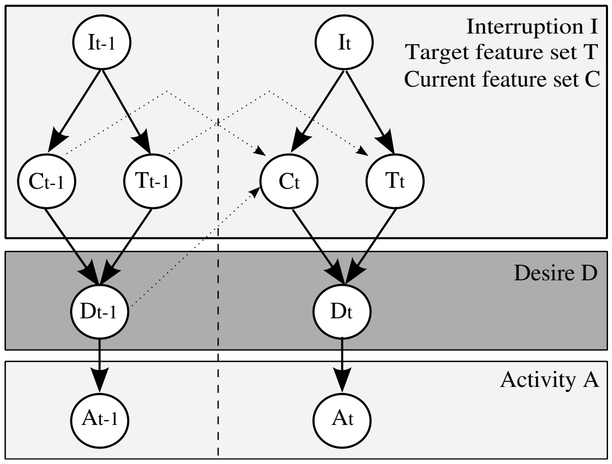


Figure 3: The top two layers of the Dynamic Bayesian Network predict low-level events for a complex event

3 DYNAMIC BAYESIAN NETWORK

This approach can be captured by the Dynamic Bayesian Network (DBN) in Figure 3. Nodes within the top two layers represent elements of the person's state and can be collectively referred to as x. The bottom layer represents the simple event that is observed. The vertical dashed line distinguishes the boundary between time slices, $t-1$ and t.

Activity observations: Recognition commences at the bottom of the DBN using the simple-event detection modules. Ours are described in section 3. Each detection must be attributed to a tracked object or person.

Desire: Moving up the DBN hierarchy the middle layer represents the agent's current desire. A desire is instantiated with a simple-event (activity) that supports the complex-event (goal). Given the previous definitions of T and C the conditional probability for D (desire) is:

$$p\left(d^i\right) = p\left(d^j\right) \forall_{i,j} : d^i, d^j \in C \backslash T \tag{1}$$

$$p\left(d^k\right) = 0 \, \forall_k : d^k \ni C \backslash T \tag{2}$$

Define $TP\left(\alpha^i\right)$ as the true positive detection probability of simple event α^i. Having now defined A and D the the emission probabilities can also be defined by the function $E\left(A_t, D_t\right)$:

$$\begin{aligned} E\left(A_t, D_t\right) &= p\left(A_t = \alpha^i | D_t = \alpha^j\right) && (3) \\ &= TP\left(\alpha^i\right) \qquad : \text{i = j} && (4) \\ &= 1 - TP\left(\alpha^i\right) \quad : \text{i} \neq \text{j} && (5) \end{aligned}$$

Goal Representation: The top layer in the DBN represents the agent's top-level goal and tracks the features that have been observed. The final node; I, removes an important limitation in (Baxter et al., 2010). I represents behaviour interruption, which indicates that observation A_t cannot be explained by the state x_t (the top two layers of the DBN). It implies one of two conditions. 1) A person has switched their complex behaviour (e.g. goal) and thus $T_{t-1} \neq T_t$. Although humans frequently switch between behaviours, this condition breaks the assumptions made by (Baxter et al., 2010), causing catastrophic failure. 2) A_t is a false detection. In this case, the elements of *T* and *C* are temporarily ignored.

3.1 MODEL PARAMETERS

Given the model description above, the DBN parameters can be summarised as follows.

Variables: α is the set of detectable simple events. T represents a single behaviour (complex event) and $\forall t \in T : t \in \alpha$. C represents the elements of T that have been observed and thus $\forall c \in C : c \in T$. D is a prediction of the next simple-event and is drawn from $T \backslash C$. Finally, A is the observed simple event and is drawn from α.

Probabilities: Define $Beh\left(\beta\right)$ as the target feature set for behaviour β, and $Pr\left(\beta\right)$ as the prior probability of β. The transition probabilities for latent variables C and T can then be defined as per Table 2.

The distribution on values of D is defined by equations 1 and 2, and the emission probabilities by equations 3 to 5.

It should be noted that of all these parameters, only functions $Beh\left(\beta\right)$ and $E\left(A_t, D_t\right)$ need to be defined by the user. It is expected that $Beh\left(\beta\right)$ (the set of features representing behaviour β) can be easily defined by an expert, while $E\left(A_t, D_t\right)$ may be readily obtained by evaluating the simple-event detectors on a sample dataset. All other parameters are calculated at run-time, eliminating learning.

3.2 RAO-BLACKWELLISED INFERENCE

The DBN in Figure 3 is a finite state Markov chain and could be computed analytically. However, given our target application of visual surveillance, which has the requirement of near real-time processing, we adopt a particle filtering approach to reduce the execution time. In Particle Filtering the aim is to recursively estimate $p(x_{0:t}|y_{0:t})$, in which a state sequence $\{x_0, ..., x_t\}$ is assumed to be a hidden Markov process and each element in the observation sequence $\{y_0, ..., y_t\}$ is assumed to be independent given the state (i.e. $p(y_t|x_t)$) (Doucet et al., 2000b).

We utilise a Rao-Blackwellised Particle Filter (RBPF) so that the inherent structure of a DBN can be utilised. We wish to recursively estimate $p(x_t|y_{1:t-1})$, for which the RBPF partitions x_t into two components $x_t \; : \; (x_t^1, \; x_t^2)$ Doucet et al. (2000a). This paper will denote the sampled component by the variable r_t, and the marginalised component as z_t. In the DBN in Figure 3, $r_t \; : \; \langle \; C_t, \; T_t, \; I_t \; \rangle$ and $z_t \; : \; D_t$. This leads to the following factorisations:

$$p(x_t|y_{1:t-1}) = p(z_t|r_t, y_{1:t-1})p(r_t|y_{1:t-1}) \qquad (6)$$

$$= p(D_t|C_t, T_t, I_t, y_{1:t-1})p(C_t, T_t, I_t|y_{1:t-1}) \qquad (7)$$

The factorisation in 7 utilises the inherent structure of the Bayesian network to perform exact inference on D, which can be efficiently performed once $\langle \; C_t, \; T_t, \; I_t \; \rangle$ has been sampled. Each particle i in the RBPF represents a posterior estimate (hypothesis) of the form h_t^i : $\langle \; C_t^i, \; T_t^i, \; I_t^i, \; D_t^i, \; W_t^i \rangle$, where W_t is the weight of the particle calculated as $p(y_t^i|x_t^i)$.

For brevity we will focus on the application of the RBPF to our work, but refer the interested reader to (Bui and Venkatesh, 2002; Doucet et al., 2000a) for a generic introduction to the approach.

3.2.1 Algorithm

At time-step 0, T is sampled from the prior and $C = \emptyset$ for all N particles. For all other time steps, N particles are sampled from the weighted distribution from $t-1$ and each particle predicts the new state $\langle C_t^i, T_t^i, I_t^i \rangle$ using the transition probabilities in Table 2.

After sampling is complete, the particle set is partitioned into those where $p(y_t|C_t, T_t, I_t)$ is non-zero, and zero. The first partition is termed the *Eligible* set because the particle states are consistent with the new observation, while the second partition is termed the *Rebirth* set. Particles in the *Rebirth* set represent those where an interruption has occurred. For each particle in this set, T and C are re-initialised according to the prior distribution with a probability of $p(TP)$, indicating the true positive rate of the observation. With a probability of $1 - p(TP)$, particles are flagged as "FP" (False Positive), and are not re-initialised.

At the next step, the *Eligible* and *Rebirth* sets are recombined and the Rao-Blackwellised posterior is calculated: $p(z_t^i|r_t^i, y_{1:t-1}) \; = \; p(D_t^i|C_t^i, T_t^i, I_t^i, y_{1:t-1})$. The value of D_t^i (the agent's next desire) is then predicted according to the Rao-Blackwellised posterior. At this point each particle has a complete state estimate x_t^i, and can be weighted according to equation 8. It is important to note that particles

Table 2: DBN transition probabilities between time steps $t-1$ and t

$p\left(C_t = C_{t-1} \cup \{D_{t-1}\} \vert I_t = 0\right)$	$= TP\left(A_{t-1}\right)$	when $D_{t-1} = A_{t-1}$
$p\left(C_t = C_{t-1} \cup \{D_{t-1}\} \vert I_t = 0\right)$	$= 0$	when $D_{t-1} \neq A_{t-1}$
$p\left(C_t = \emptyset \vert I_t = 1\right)$	$= 1$	
$p\left(T_t \neq T_{t-1} \vert I_t = 0\right)$	$= 0$	
$p\left(T_t = Beh\left(\beta\right) \vert I_t = 1\right)$	$= pr\left(\beta\right)$	if A_{t-1} not assumed false positive
$p\left(T_t = T_{t-1} \vert I_t = 1\right)$	$= 1$	if A_{t-1} assumed false positive

flagged as "FP" are weighted with $1 - p(TP)$.

$$p(y_t|x_t^i) = p(A_t|C_t^i, T_t^i, I_t^i, D_t^i) \tag{8}$$

The final step in the algorithm is to calculate the transition probabilities. This step ensures that the algorithm is robust to activity recognition errors. The transition probability encapsulates the probability that the agent really has performed the predicted feature D_t^i, observed via A_t.

4 RESULTS AND DISCUSSION

Two datasets were used to evaluate our framework. Five complex behaviours were extracted from four PETS 2006 scenarios, and our own video dataset contains the same behaviours but encompasses more variability than PETS in terms of luggage items and the ordering of events. Experiments were run on a Dual Core 2.4Ghz PC with 4GB RAM.

Figure 4 shows the average F-Scores for the low-level detectors (trackers, event modules). An F-score is a weighted average of a classifiers accuracy and recall with range [0:1], where 1 is optimal. Our person tracker performs well (F-Score ≥ 0.92), but occasionally misclassified non-persons (e.g. trolley), instantiates multiple trackers for a single person, or does not detect all persons entering in close proximity. The object tracker has an F-Score ≥ 0.83, and is limited by partial obstructions from the body and shadows.

The naivety of our simple event modules makes them reliant on good tracker performance. Although the average score is 0.83, the "Group Formed" module is particularly unreliable (F-Score: 0.6).

4.1 COMPLEX EVENT RECOGNITION

The five complex behaviours used in our evaluation are: *Passing Through 1 (PT-1):* Person enters and leaves, *Passing Through 2 (PT-2):* Person enters, places luggage, picks it up and leaves, *Abandoned Object 1 (AO-1):* Person meets with a second person, places luggage and leaves, *Abandoned Object 2 (AO-2):* Person enters, places luggage and leaves, and *Watched Item (WI):* Two people enter together,

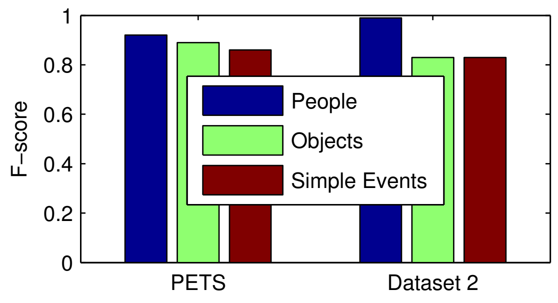


Figure 4: The Low-level F-scores for objects and people tracking, and simple events

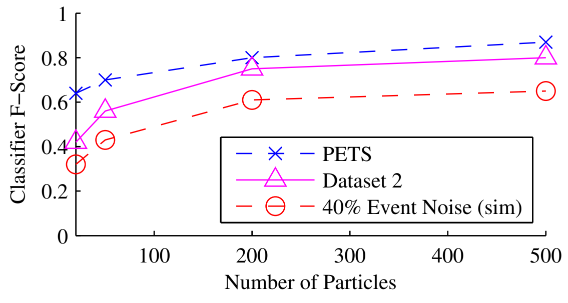


Figure 5: Classifier F-Score as the number of particles is increased (reducing speed).

one places luggage and leaves, one remains. This last behaviour was synthesized for the PETS dataset by merging track data from scenarios six and four.

Figure 5 compares the average classifier F-Scores as the number of particles is increased. Classifications are made after all simple events have been observed by selecting the most likely complex event. A minimum likelihood of 0.3 was imposed to remove extremely weak classifications. As the number of particles increases accuracy/recall is improved. The algorithm remains very efficient with 500 particles, and is capable of processing in excess of 38,000 simple events per second. The classifiers achieve 0.8 F-Score on Dataset 2, and 0.87 on PETS.

In section 3 we highlighted that our naive simple-event

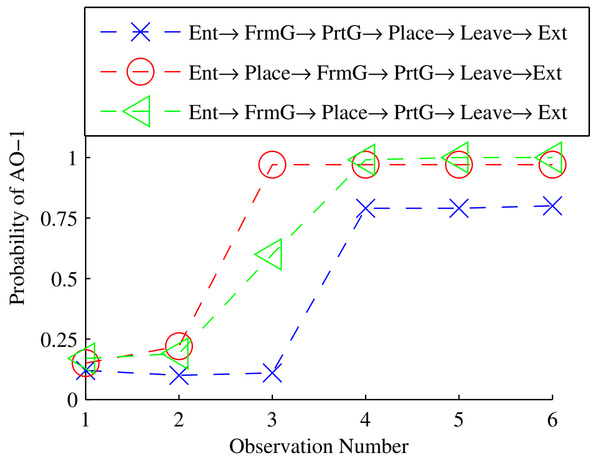


Figure 6: Observations arriving in different orders still match the correct goal (AO-1). Nomenclature: Enter (Ent), Form Group (FrmG), Part Group (PrtG), Place Item (Place), Leave Item (Leave), Exit (Ext)

modules would perform poorly on more complex video. To simulate these conditions, we artificially inserted noise into the observation stream to lower the true positive rate to 60%. Figure 5 shows that even with this high degree of noise, complex events can be detected with 0.65 F-Score.

Table 3 shows classifier confusion across both datasets. Missing object detections cause confusion between *PT-1* and *PT-2*. Behaviours *AO-1* and *WI* differ by only one event (Form Group), and thus absent group detections lead to confusion here.

Table 3: Confusion Matrix for the combined video datasets

	Scenario				
	PT-1	PT-2	WI	AO-1	AO-2
PT-1	0.92	0	0	0.08	0
PT-2	0.33	0.58	0	0	0.08
WI	0	0	0.9	0.1	0
AO-1	0	0	0.2	0.8	0
AO-2	0	0	0	0	1

4.2 TEMPORAL ORDER

We proposed that the exact temporal order of observations does not need to be modelled to recognise human behaviour. Figure 6 supports this thesis by showing complex-event likelihood for three different activity permutations of the *AO-1* behaviour. In all three cases *AO1* is highly probable, although there are differences in probability. These differences are because some activity subsequences are shared between multiple behaviours. For instance, $\langle PlaceItem, LeaveItem, Exit\rangle$ matches both

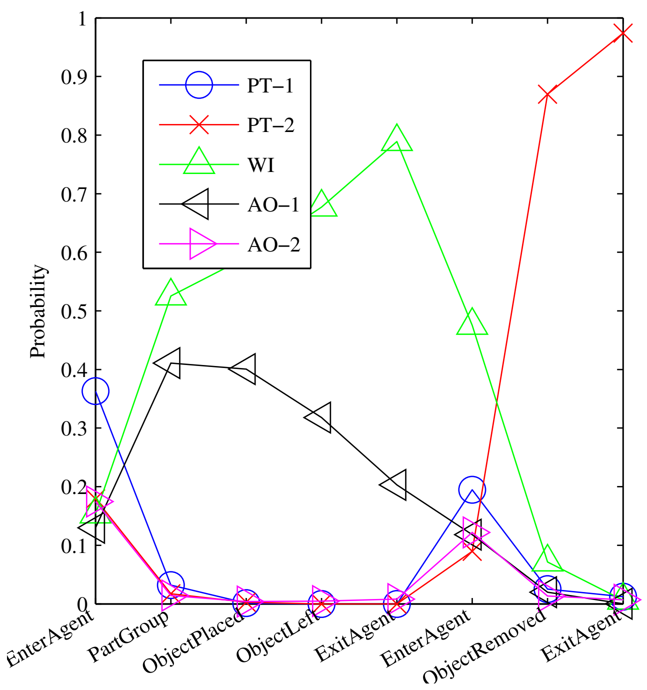


Figure 7: Probability of each behaviour as observations are made. Behaviour switches from WI to PT-2 at observation 6, causing a similar shift in behaviour probability.

AO-1 and *AO-2*. There is a low probability that observations $\langle FormGroup, PartGroup\rangle$ were false detections, and thus some probability is removed from AO1 in support of AO2, which can also explain the subsequence $\langle PlaceItem, LeaveItem, Exit\rangle$. Although observation order can have an impact on goal probability, it is clear that our thesis holds for these behaviours.

4.3 BEHAVIOUR SWITCHING

Our inference algorithm contains components to detect behaviour switching, which occurs when an agent concatenates or otherwise changes their behaviour (see Section 2.3). To demonstrate the effectiveness of these components Figure 7 plots the probability of each behaviour as observations are received from two concatenated behaviours. The behaviours are *WI*, followed by *PT-2*.

In observation 1 the agent enters. The distributions on the features within each behaviour causes *PT-1* to be most probable because it has the least features. The second observation can only be explained by two behaviours and is reflected in the figure. At observation six "EnterAgent" cannot be explained by any of the behaviours, triggering behaviour interruption. Observation seven can only be explained by *PT-2* and this is reflected in the figure. As a result, the behaviours that best explain the observations are *WI* and *PT-2*, which matches the ground truth.

5 CONCLUSION AND FUTURE WORK

This paper has argued that data scarcity prevents the advancement of high-level automated visual surveillance using probabilistic techniques, and that anomaly detection side-steps the issue for low-level events. We proposed that simple visual events can be considered as salient features and used to recognise more complex events by imposing a weak temporal ordering. We developed a framework for end-to-end recognition of complex events from surveillance video, and demonstrated that our "bag-of-activities" approach is robust and scalable.

Section 2.3 made the assumption that for a set of features defining a behaviour, each feature is only performed once. This assumption limits our approach but is not as strong as it may at first appear. An agent who enters and exits the scene can still re-enter, as this is simply the concatenation of two behaviours. Each individual behaviour has only involved one 'EnterAgent' event so the assumption is not in conflict. Furthermore, it is also possible to consider actions that are opposites. For instance, placing and removing a bag, or entering and exiting the scene, can both be considered action pairs that 'roll-back' the state. Although not implemented in this paper, further work has shown that this is an effective means of allowing some action repetition. The only behaviours prevented by the assumption are those that require performing action A twice (e.g. placing two individual bags).

Clearly, improving the sophistication of the simple event detection modules is a priority in extending our approach to more complicated data. The *Group Tracker* module could be improved by estimating each person's velocity and direction using a Kalman filter. These attributes could then be merged with the proximity based approach to more accurately detect the forming and splitting of groups.

Acknowledgements

This work was partially funded by the UK Ministry of Defence under the Competition of Ideas initiative.

References

Rolf H. Baxter, Neil M. Robertson, and David M. Lane. Probabilistic behaviour signatures: Feature-based behaviour recognition in data-scarce domains. In *Proceedings of the 13th International Conference on Information Fusion*, 2010.

Oren Boiman and Michal Irani. Detecting irregularities in images and in video. *International Journal of Computer Vision*, 74(1): 17–31, 2007.

Hung H. Bui and Svetha Venkatesh. Policy recognition in the abstract hidden markov model. *Journal of Artificial Intelligence Research*, 17:451–499, 2002.

Hannah Dee and David Hogg. Detecting inexplicable behaviour. In *British Machine Vision Conference*, volume 477, page 486, 2004.

Arnaud Doucet, Nando de Freitas, Kevin Murphy, and Stuart Russell. Rao-blackwellised particle filtering for dynamic bayesian networks. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pages 176–183, 2000a.

Arnaud Doucet, Simon J. Godsill, and Christophe Andrieu. On sequential monte carlo sampling methods for bayesian filtering. *Statistics and computing*, 10(3):197–208, 2000b.

Florent Fusier, Valry Valentin, Franois Brmond, Monique Thonnat, Mark Borg, David Thirde, and James Ferryman. Video understanding for complex activity recognition. *Machine Vision and Applications*, 18:167–188, 2007. ISSN 0932-8092.

Neil J. Gordon, David J. Salmond, and A.F.M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *Radar and Signal Processing, IEE Proceedings F*, 140(2):107 –113, April 1993. ISSN 0956-375X.

Helmut Grabner, Peter M. Roth, Michael Grabner, and Horst Bischof. Autonomous learning of a robust background model for change detection. In *Workshop on Performance Evaluation of Tracking and Surveillance*, pages 39–46, 2006.

Gal Lavee, Ehud Rivlin, and Michael Rudzsky. Understanding video events: a survey of methods for automatic interpretation of semantic occurrences in video. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 39(5):489–504, 2009. ISSN 1094-6977.

Benjamin Laxton, Jongwoo Lim, and David Kriegman. Leveraging temporal, contextual and ordering constraints for recognizing complex activities in video. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1 –8, 2007.

Lin Liao, Donald J. Patterson, Dieter Fox, and Henry Kautz. Learning and inferring transportation routines. *Artificial Intelligence*, 171(5-6):311–331, 2007. ISSN 0004-3702.

Wasit Limprasert. People detection and tracking with a static camera. Technical report, School of Mathematical and Computer Sciences, Heriot-Watt University, 2010.

David G. Lowe. Object recognition from local scale-invariant features. In *International Conference on Computer Vision*, volume 2, pages 1150–1157, 1999.

Fengjun Lv, Xuefeng Song, Bo Wu, Vivek Kumar, and Singh Ramakant Nevatia. Left luggage detection using bayesian inference. In *Proceedings of PETS*, 2006.

Kevin P. Murphy. *Dynamic Bayesian networks: representation, inference and learning*. PhD thesis, 2002.

Nam T. Nguyen, Dinh Q. Phung, Svetha Venkatesh, and Hung Bui. Learning and detecting activities from movement trajectories using the hierarchical hidden markov models. In *Computer Vision and Pattern Recognition*, volume 2, pages 955–960, 2005. ISBN 0-7695-2372-2.

Alonso Patron, Eric Sommerlade, and Ian Reid. Action recognition using shared motion parts. In *Proceedings of the 8th International Workshop on Visual Surveillance*, October 2008.

Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, volume 77, pages 257–286, San Francisco, CA, USA, 1989.

Neil Robertson, Ian Reid, and Michael Brady. Automatic human behaviour recognition and explanation for CCTV video surveillance. *Security Journal*, 21(3):173–188, 2008.

Tao Xiang and Shaogang Gong. Video behavior profiling for anomaly detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(5):893, 2008.

Ontology-based generation of Object Oriented Bayesian Networks

Mouna Ben Ishak
LARODEC Laboratory
ISG, University of Tunis
Tunisia, 2000
mouna.benishak@gmail.com

Philippe Leray
Knowledge and Decision Team
LINA Laboratory UMR 6241,
Polytech'Nantes, France
philippe.leray@univ-nantes.fr

Nahla Ben Amor
LARODEC Laboratory
ISG, University of Tunis
Tunisia, 2000
nahla.benamor@gmx.fr

Abstract

Probabilistic Graphical Models (PGMs) are powerful tools for representing and reasoning under uncertainty. Although useful in several domains, PGMs suffer from their building phase known to be mostly an NP-hard problem which can limit in some extent their application, especially in real world applications. Ontologies, from their side, provide a body of structured knowledge characterized by its semantic richness. This paper proposes to harness ontologies representation capabilities in order to enrich the process of PGMs building. We are in particular interested in object oriented Bayesian networks (OOBNs) which are an extension of standard Bayesian networks (BNs) using the object paradigm. We show how the semantical richness of ontologies might be a potential solution to address the challenging field of structural learning of OOBNs while minimizing experts involvement which is not always obvious to obtain. More precisely, we propose to set up a set of mapping rules allowing us to generate a prior OOBN structure by morphing an ontology related to the problem under study to be used as a starting point to the global OOBN building algorithm.

1 Introduction

Knowledge representation (KR) is one of the principal areas of Artificial Intelligence which was studied by different techniques coming from various disciplines. In this work we will focus on probabilistic graphical models and ontologies which are considered within the most efficient frameworks in KR.

Probabilistic graphical models (PGMs) provide an efficient framework for knowledge representation and reasoning under uncertainty. Ontologies allow logical reasoning about concepts linked by semantic relations within a knowledge domain. Even though they represent two different paradigms, PGMs and ontologies share several similarities which has led to some research directions aiming to combine them. Concern for the majority of them was to extend ontologies in order to support uncertainty. This is either by adding additional markups to represent probabilistic information or by mapping the ontology into a PGM in order to enrich ontology reasoning with probabilistic queries. Few works intend to construct PGMs using ontologies. In this area, Bayesian networks (BNs) (Pearl, 1988) are the most commonly used. Typically, concepts are associated to nodes, ontology relations are used to link these nodes, and for some proposals, axioms are involved to express nodes or edges or to define the states of variables. However, given the restrictive expressiveness of Bayesian networks, these methods focus on a restrained range of ontologies and neglect some of their components such as representing concepts properties, non taxonomic relations, etc. To overcome this weakness, we propose to explore other PGMs, significantly more expressive than standard BNs, in order to address an extended range of ontologies.

We are in particular interested in *object oriented Bayesian networks* (Bangsø and Wuillemin, 2000) (OOBN), which are an extension of standard BNs. In fact, OOBNs share several similarities with ontologies and they are suitable to represent hierarchical systems as they introduce several aspects of object oriented modeling, such as inheritance. Our idea is to benefit from ontologies in order to address the challenging problem of OOBN structure learning known to be an NP-hard process. To this end, we first establish the correspondence between OOBNs and ontologies. Then, we describe how to generate a prior OOBN structure by morphing an ontology related to the problem under study and then to use it as a starting point to the global building OOBN algorithm. This latter will take advantages from both semantical data, derived from ontology which will ensure its good start-up and observational data.

The remainder of this paper is organized as follows: In sections 2 and 3 we provide a brief representation of our working tools. In section 4, we show how to benefit from knowledge provided by an ontology to define the structure of an OOBN. In section 5 we represent a survey on existing approaches trying to find a combination between PGMs and ontologies. The final section summarizes conclusions reached in this work and outlines directions for future research.

2 Object Oriented Bayesian Networks

Probabilistic graphical models (PGMs) provide an efficient framework for knowledge representation and reasoning under uncertainty. In the literature, we distinguish a panoply of PGMs sharing two common components: a graphical one (i.e. a set of nodes and links) and a numerical one allowing the quantification of different links defined in the graphical component via probability distributions. Among the most used PGMs we can mention Bayesian networks (BNs) (Pearl, 1988) which have been largely developed and used in several real world applications. Despite their great success, BNs are limited when dealing with large-scale systems. Thus, several extensions have been proposed in order to broaden their range of application, such as *object oriented Bayesian networks* (OOBNs) (Bangsø and Wuillemin, 2000), (Koller and Pfeffer, 1997) which introduce the object oriented paradigm into the framework of BNs. Object Oriented Bayesian Networks (OOBNs) are a convenient representation of knowledge containing repetitive structures. So they are a suitable tool to represent dynamic Bayesian networks as well as some special relations which are not obvious to represent using standard BNs (e.g., examine a hereditary character of a person given those of his parents). Thus an OOBN models the domain using fragments of a Bayesian network known as classes. Each class can be instantiated several times within the specification of another class. Formally, a class T is a DAG over three, pairwise disjoint sets of nodes $(\mathcal{I}_T, \mathcal{H}_T, \mathcal{O}_T)$, such that for each instantiation t of T:

- $\mathcal{I}_T$ is the set of input nodes. All input nodes are references to nodes defined in other classes (called referenced nodes). Each input node have at most one referenced node, it has no parents in t and no children outside t.

- $\mathcal{H}_T$ is the set of internal nodes including instantiations of classes which do not contain instantiations of $\mathcal{T}$. They are protected nodes that can't have parents or children outside t.

- $\mathcal{O}_T$ is the set of output nodes. They are nodes from the class usable outside the instantiations of the class and they can not have parents outside t. An output node of an instantiation can be a reference node if it is used as an output node of the class containing it.

Internal nodes, which are not instantiations of classes, and output nodes (except those that are reference nodes) are considered as real nodes and they represent variables. In an OOBN, nodes are linked using either directed links (i.e., links as in standard BNs) or reference links. The former are used to link reference or real nodes to real nodes, the latter are used to link reference or real nodes to reference nodes. Each node in the OOBN has its potential, i.e. a probability distribution over its states given its parents. To express the fact that two nodes (or instantiations) are linked in some manner we can use construction links $(- - -)$ which only represent a help to the specification.

When some classes in the OOBN are similar (i.e. share some nodes and potentials), their specification can be simplified by creating a class hierarchy among them. Formally, a class S over $(\mathcal{I}_S, \mathcal{O}_S, \mathcal{H}_S)$ is a subclass of a class T over $(\mathcal{I}_T, \mathcal{O}_T, \mathcal{H}_T)$, if $\mathcal{I}_T \subseteq \mathcal{I}_S$, $\mathcal{O}_T \subseteq \mathcal{O}_S$ and $\mathcal{H}_T \subseteq \mathcal{H}_S$.

Example 1. *Figure 1 represents the insurance network adapted to the OOBN framework (Langseth and Nielsen, 2003). This network contains six classes (Insurance, Theft, Accident, Car, CarOwner and Driver). In this figure only the interfaces of the encapsulated instantiations are shown, dashed ellipses represent input nodes, while shaded ellipses represent output nodes.*
For instance, the class CarOwner describes properties of a car owner. It has no input nodes, the nodes Age, SocioEcon, HomeBase, AntiTheft, VehicleYear and MakeModel operate as output nodes of this class. Moreover, Driven characteristics are a part of the notion of a car owner. Thus, an instantiation of the class Driver is then encapsulated in the class CarOwner.
Note that the output node DrivQuality of the class Driver is used as output reference node of the class CarOwner as it is referenced in the Accident class.

In the extreme case where the OOBN consists of a class having neither instantiations of other classes nor input and output nodes we collapse to standard BNs.

As all PGMs, OOBNs have two fundamental cornerstones: construction and reasoning. The construction of an OOBN concerns both learning the graph structure and parameters estimation. Few works have been proposed in the literature to learn the structure (Bangsø et al., 2001), (Langseth and Nielsen, 2003) and the parameters (Langseth and Bangsø, 2003) of such a model from data. Given an OOBN, reasoning

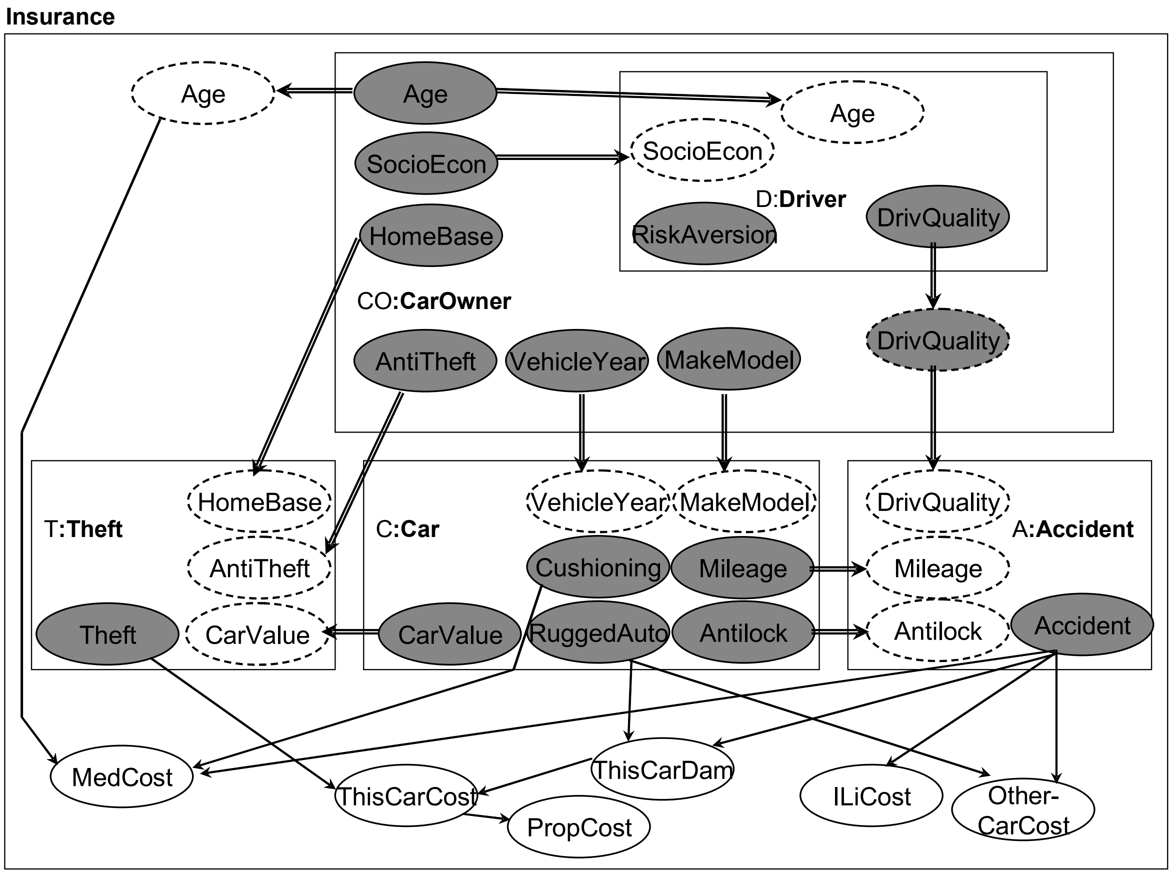


Figure 1: The insurance network represented using the OOBN framework

stands for probabilistic inference and this requires to translate the OOBN into a BN or a multiply sectioned Bayesian network (MSBN) (Bangsø and Wuillemin, 2000).

In this paper we are interested in the learning process. The standard approach proposed by (Langseth and Bangsø, 2003) is the OO-SEM algorithm. This algorithm is based on an **Object Oriented assumption** which states that *all instances of a class are assumed to be identical w.r.t. both parameters and structure.* This algorithm is based on a prior expert knowledge about a partial specification of the OOBN by grouping nodes into instantiations and instantiations into classes. Then, on the basis of this prior, the learning process adapts the SEM algorithm (Friedman, 1998) in order to learn the OOBN structure that fits best to the data. Learning in object oriented domains allows to reduce the search space, however it remains an NP-hard problem. In fact, the main computational phase in the OO-SEM algorithm consists in finding the interfaces of instantiations, which is exponential in the number of instantiations. So, this information may be also elicited from domain experts. However, human expertise, required to initiate the learning process, is not always obvious to obtain. To overcome this limitation we propose to use ontologies richness. Before introducing our method, we give basic notions on ontologies.

3 Ontologies

Over the last few years, there has been an increasing interest in the application of ontologies in various domains (e.g., linguistics, semantic web, bioinformatics). They represent not only a fixed structure but also the basis for deductive reasoning. For the AI community, an ontology is *an explicit specification of a conceptualization* (Gruber, 1995). That is, an ontology is a description of a set of representational primitives with which to model an abstract model of a knowledge domain. Formally, we define an ontology $\mathcal{O} = \langle \mathcal{C}p, \mathcal{R}, \mathcal{I}, \mathcal{A} \rangle$ as follows:

- $\mathcal{C}p = \{cp_1, \dots cp_n\}$ is the set of n concepts (classes) such that each cp_i has a set of k properties (attributes) $\mathcal{P}_i = \{p_1, \dots p_k\}$.
- $\mathcal{R}$ is the set of binary relations between elements of $\mathcal{C}p$ which consists of two subsets:

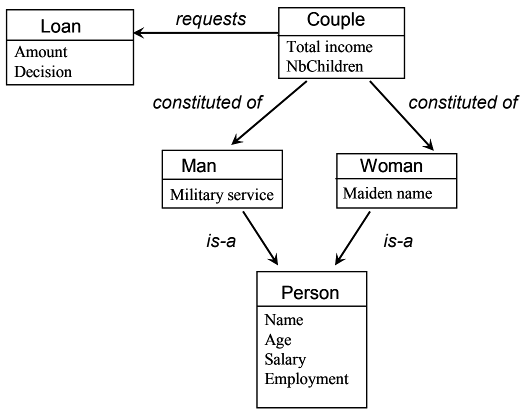


Figure 2: The joint credit ontology

- $\mathcal{H}_\mathcal{R}$ which describes the inheritance relations among concepts.
- $\mathcal{S}_\mathcal{R}$ which describes semantic relations between concepts. That is, each relation $cp_i s_R cp_j \in \mathcal{S}_\mathcal{R}$ has cp_i as a domain and cp_j as a range.

- $\mathcal{I}$ is the set of instances, representing the knowledge base.
- $\mathcal{A}$ is the set of the axioms of the ontology. $\mathcal{A}$ consists of constraints on the domain of the ontology that involve $\mathcal{C}p$, $\mathcal{R}$ and $\mathcal{I}$.

Axioms are of the form $A \equiv B$ (A and B are equivalent), $R1 \subseteq R2$ ($R1$ is a subproperty of $R2$), $R1(x, y)$ (x is related to y by the relation $R1$), $A(x)$ (x is of type A), etc. Where A and B are concepts, $R1$ and $R2$ are relations and x and y are instances.

Example 2. *Figure 2 is an example of a joint credit ontology.*
$\mathcal{C}p = \{Loan, Couple, Man, Woman, Person\}$.
For instance $\mathcal{P}_{Loan} = \{Amount, Decision\}$.
$\mathcal{S}_\mathcal{R}$ = $\{$*requests*$(Couple \times Loan)$, *constituted of*$(Couple \times Man)$, *constituted of*$(Couple \times Woman)\}$.
Is-a relations represent $\mathcal{H}_\mathcal{R}$ *and are equivalent to subsumption axioms, e.g.,* $Man \subseteq Person$. *For instance, we can have an instance p of the concept Man* $p\,\{Paul, 30, 2000\$, teacher, T\}$, *p.Name = Paul, p.Age = 30, p.Salary = 2000\$, p.Employment = teacher and p.Military service = T.*

4 A new approach for OOBNs building based on ontologies

Clearly PGMs and ontologies share several similarities even they are derived from different frameworks. Thus our idea is to use the ontological knowledge in the OOBN learning process by morphing the ontology in hand into the a prior OOBN structure. To this end, we first define the common points and similarities between these two paradigms, then we describe the main steps of our proposal.

4.1 OOBNs vs ontologies

In this part, we highlight the common points and similarities between ontologies and object oriented Bayesian networks. The main components of an ontology (i.e., concepts and relations) may be viewed as a start-up to define the main components of an OOBN (i.e., classes and relations among them).

- **Concepts vs classes**

  Ontology concepts are translated into classes of the OOBN framework. Hence, for each class so defined, concept properties will constitute the set of its random variables (real nodes). It is clear that the set of the concept properties does not cover the three sets of nodes of a class. Let:

  - cp_i be the concept of the ontology translated to the class c_i in the underlying OOBN, where c_i is a DAG over $\mathcal{I}_c, \mathcal{H}_c$ and $\mathcal{O}_c$.
  - $P_i = \{p_1 \dots p_k\}$ be the set of properties of cp_i.
  - $\mathcal{H}'_c = \mathcal{H}_c \setminus \mathcal{H}^{inst}_c$, where $\mathcal{H}^{inst}_c$ is the set of internal nodes which are instantiations of other classes.
  - $\mathcal{O}'_c = \mathcal{O}_c \setminus \mathcal{O}^{ref}_c$, where $\mathcal{O}^{ref}_c$ is the set of output nodes which are reference nodes.

  P_i allows us to generate $\mathcal{H}'_c \cup \mathcal{O}'_c$. Reference nodes, namely, $\mathcal{I}_c \cup \mathcal{O}^{ref}_c$, are pointers to nodes defined in other classes. Consequently their set of states as well as parameters are copied from the referenced nodes. These latter are properties of other concepts in the ontology side. Reference nodes as well as $\mathcal{H}^{inst}_c$ will be derived from the semantic relations.

- **Inheritance relations vs class hierarchy**

  As ontological inheritance relations already model a hierarchical feature, then all concepts connected by an *is-a* relation in the ontology will be represented by a class hierarchy in the OOBN framework.

- **Semantic relations vs links**

  Having two concepts $\{cp_i, cp_j\} \in Cp^2$ related by a semantic relation means that there is at least one property of one of them that affects at least one property of the other, which means that the definition of one of them depends on the existence of the other. In the underlying OOBN, this allows to set up dependencies among nodes from different classes. Suppose that the property p_k of concept cp_i affects the property $p_{k'}$ of concept cp_j. Then, the node that represents p_k in the class c_i will be either an output node connected directly using directed links to the internal node representing $p_{k'}$ in the class c_j, in this case c_j could only be an encapsulating class of the instance of c_i, or an output node referenced, using reference links, by a reference node in the class c_j, this reference node is either an input node, parent of the node that represents $p_{k'}$ in c_j or an output reference node of the class containing an instance of c_i and communicates with c_j.

Semantic relations might provide an information about classes interfaces and instantiations organization in the OOBN. However, the link direction of the semantic relation can not provide a good informer about dependence relations among the variables of the OOBN, which variable depends on the other? So, it is required that the semantic relations be designed from the beginning of a causal or an anti-causal orientations. The choice of a fixed orientation is a determining factor to specify which instantiation I_i could be referenced from an instantiation I_j. Suppose that all semantic relations are of causal orientation, the cause is then conceived as the direct explanation of the fact and it is involved in its production. consequently, the definition of the concept range depends on the existence of the concept domain. In the OOBN side, this means that the definition of the class representing the concept domain is part of the class representing the concept range. This can be translated in the OOBN by instantiating the class representing the concept domain within the specification of the class representing the concept range.

In what follows, we assume that all semantic relations have a causal orientation. Thus, $\forall \{cp_i, cp_j\} \in Cp^2$ related by a semantic relation, where cp_i is the domain and cp_j is the range, cp_i is considered as the cause of cp_j and this latter is the effect.

In fact, the ontology conceptual graph is simply the result of the ontology components definition. Thus, we require that semantic relations definition to be, from the beginning, done following a causal reasoning that is considered as an intuitive reflexion of the ontologist. Then, if we require to have all semantic relations to be anti-causal, we just have to reverse their definitions (i.e., define the domain as range and vice versa).

4.2 The morphing process

To ensure the morphing process, we need to traverse the whole ontology. To provide this, we assume that the ontology is a directed graph whose nodes are the concepts and relations (semantic and hierarchical ones) are the edges. Our target is to accomplish the mapping of the ontology graphical representation into an OOBN while browsing each node once and only once. To this end, we propose to adapt the generic Depth-First Search (DFS) algorithm for graph traversing. The idea over the Depth-First Search algorithm is to traverse a graph by exploring all the vertices reachable from a source vertex: If all its neighbors have already been visited (in general, color markers are used to keep track), or there are no ones, then the algorithm backtracks to the last vertex that had unvisited neighbors. Once all reachable vertices have been visited, the algorithm selects one of the remaining unvisited vertices and continues the traversal. It finishes when all vertices have been visited. The DFS traversal allows us to classify edges into four classes:

- Tree edges: are edges in the DFS search tree.
- Back edges: join a vertex to an ancestor already visited.
- Forward edges: are non-tree edges connecting a vertex to a descendant in a DFS search tree.
- Cross edges: all other edges.

We use these classes of edges to determine actions to do on each encountered concept. Tree edges allow to define actions on concepts encountered for the first time, while forward and cross edges allow to define actions to do on concepts that are already visited crossing another path and so having more than one parent. According to their definition, back edges allow cycle detection, in our case these edges will never be encountered. As our edges respect a causal orientation having a cycle of the form $X_1 \rightarrow X_2 \rightarrow X_3 \rightarrow X_1$ means that X_1 is the cause of X_2 which is the cause of X_3 so this latter can't be the cause of X_1 at the same instant t but rather at an instant $t + \epsilon$. We are limited to ontologies that do not contain cycles, because such relationships invoke the dynamic aspect which is not considered in this work.

A deep study of the similarities discussed above shows that the morphing process can be done in three main steps, namely initialization, discovery and closing. At each step, we define a set of actions that might be done:

i. **Initialization step:** All concepts are undiscovered, we generate the OOBN class and a class to

Algorithm 1: Generate_OOBN

```
Input: An ontology $\mathcal{O}$
$\mathcal{O}$ is of an anti-causal orientation.
For all concepts, the color must be initialized to "white" before running the algorithm.
begin
    CREATE_OOBN_GLOBAL ;
    for each concept $cp \in \mathcal{C}p$ do
        RECORD_PREDECESSOR[$cp$]=NULL;
        CREATE_CLASS(cp)
    for each concept $cp \in \mathcal{C}p$ do
        if color[cp]= white then
            Handling_Process($\mathcal{O}$, $cp$)
```

each concept:

CREATE_OOBN_GLOBAL : creates the OOBN class.

CREATE_CLASS(Concept cp): transforms a concept cp to a class c_{cp}.

ii. **Discovery step:** The classes of edges are used to determine actions to do on each encountered concept. These actions allow us to define input, internal and output sets for each class of the OOBN.

ADD_INPUT_NODE(Node n, Class c) : adds an input node n to a class c. this action is invoked on all properties of a concept which is related by an out edge to another one. Its properties are considered as candidate input nodes of the class representing the second concept.

ADD_INTERNAL_NODE(Node n, Class c): adds an internal node n to a class c. The set of internal nodes of a class consists of instantiations of other classes representing concepts that are related by an out edge to the corresponding concept of the class c in the ontology and this edge is in the same DFS search tree.

ADD_OUTPUT_NODE(Node n, Class c): adds an output node n to a class c. all properties of a concept are transformed into variables of its corresponding class in the OOBN. These nodes are considered as candidate output nodes of the class.

ADD_OUTREF_NODE(Class $c1$, Class $c2$): adds output reference nodes to classes containing $c1$ until reaching $c2$. In fact, some concepts might have parents coming from more than one DFS search tree or from different paths. Let cp_i be a concept having two parents $cp1_i$ and $cp2_i$ coming from two different branches. Then, c_{cp_i} would to be instantiated within the specification of only one of them. However, c_{cp_i} has its output nodes

Algorithm 2: Handling_Process

```
Input: An ontology $\mathcal{O}$, A concept $S$.
We use color markers to keep track of which vertices have been discovered: white marks vertices that have yet to be discovered, gray marks a vertex that is discovered but still has vertices adjacent to it that are undiscovered and black marks discovered vertex that is not adjacent to any white vertices.
begin
    color[$S$]:= gray;
    for each property $p$ of $S$ do
        ADD_OUTPUT_NODE($p, c_S$)
    for each $V \in adjacent[S]$ do
        if color[V]=white then
            RECORD_PREDECESSOR[$V$]=$S$;
            Handling_Process($\mathcal{O}$, $V$);
            ADD_INTERNAL_NODE
            (INSTANCE_OF($c_V$), $c_S$);
            if $(S, V)$ is an inheritance relation then
                ADD_CONSTRUCT_LINK
                INSTANCE_OF($c_V$),INSTANCE_OF($c_S$))
            if $(S, V)$ is a semantic relation then
                for each node $n \in$ GET_OUTPUT($c_V$) do
                    ADD_REFERENCE_LINK
                    ($n$,ADD_INPUT_NODE($n$,$c_S$))
        if color[V]=black then
            if $(S, V)$ is an inheritance relation then
                ADD_CONSTRUCT_LINK
                INSTANCE_OF($c_S$),INSTANCE_OF($c_V$))
            if $(S, V)$ is a semantic relation then
                for each node $n \in$ GET_OUTPUT($c_V$)
                do
                    ADD_INPUT_NODE($n$, $c_S$)
                ADD_OUTREF_NODE
                (INSTANCE_OF($c_S$),INSTANCE_OF($c_V$)
    color[$S$]:= black;
    if $RECORD\_PREDECESSOR[S] = Null$ then
        ADD_INTERNAL_NODE
        (INSTANCE_OF($c_S$),GLOBAL_OOBN_CLASS)
```

to be referenced by output reference nodes of the class containing it until reaching its second parent (see figure 3).

ADD_CONSTRUCT_LINK(Class $c1$, Class $c2$): a construct link appears between instantiations of superclasses and instantiations of their subclasses (see figure 4). All properties of the super-concept are considered as properties of its subconcepts.

ADD_REFERENCE_LINK(Node $n1$, Node $n2$): allows the communication between classes interfaces.

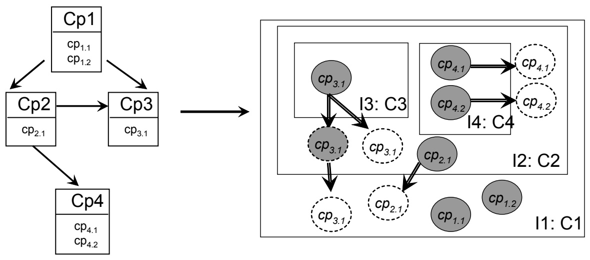


Figure 3: An example of more than one parent

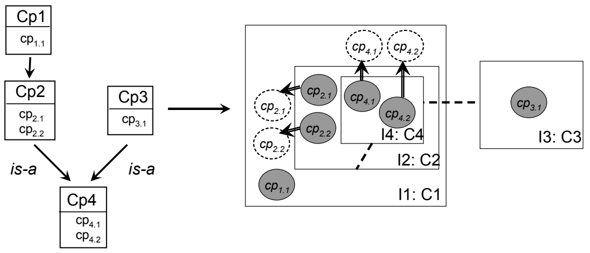


Figure 4: An example of inheritance relation

iii. **Closing step:** We check whether the vertex is a root (having no predecessor), if it is, we add an instance of its class to the global OOBN class using the ADD_INTERNAL_NODE action.

We also define the INSTANCE_OF (Class c) which allow to instantiate a class c and the GET_OUTPUT(Class c) which returns all output nodes of a class c.

All these actions are used in Algorithms 1 and 2. The *Handling_Process* function (see algorithm 2) provides actions to do at each vertex.

Example 3. *We assume that all random variables are modeled in the corresponding ontology 5(b) as concepts properties and that all semantic relations present in the ontology are of an anti-causal orientation.*

We will follow the steps of the Generate_OOBN algorithm (see algorithm 1)to generate our prior OOBN structure.

First of all, we start by generating the Global OOBN class Prior_OOBN. Then we create a class to each concept of the ontology, that is, we create 11 classes $c_{cp_i}, i = \{1, \dots 11\}$ which are initially empty. their sets of nodes will be discovered during the generation process.

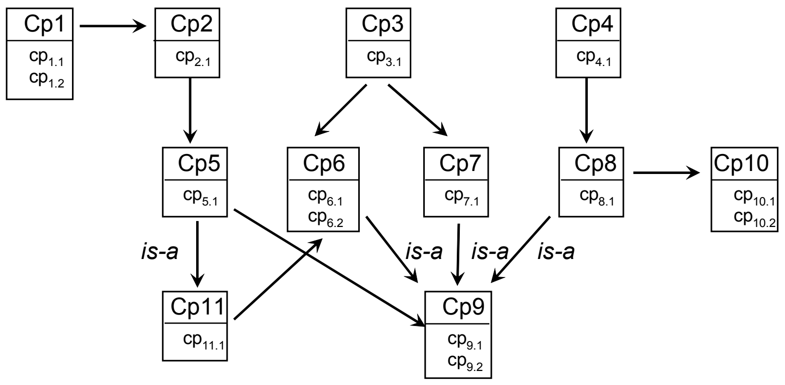
Initially, all concepts are white. cp_1 is the source concept, it is grayed and all its properties are declared as output nodes of the class representing it in the prior OOBN. Then, each concept adjacent to cp_1 is recursively visited if it is white and its properties are treated in the same way. cp_1 has cp_2 as adjacent concept, it is painted gray and it has cp_5 as adjacent concept, so on until reaching cp_9. cp_9 is grayed and all its properties are declared as output nodes of the class representing it in the prior OOBN (c_{cp_9}). As it has no adjacent, it is instantiated within its ancestor c_{cp_6}. As cp_6 and cp_9 are related by an inheritance relation then, we add a construction link between c_{cp_6} and c_{cp_9} and all properties of the super-class c_{cp_9} are considered as properties of its subclass c_{cp_6}. The concept cp_9 is finished and blackened. We backtrack to the cp_6 concept, it is gray and it has finished his adjacent concepts so, it is instantiated within its ancestor $c_{cp_{11}}$. As cp_{11} and cp_6 are related by a semantic relation then, all its output nodes are considered as input nodes of the $c_{cp_{11}}$ class linked by reference links. cp_6 is blackened and we go back to cp_{11} it is gray and it has finished his adjacent concepts so, it is instantiated within its ancestor c_{cp_5}. As $c_{cp_{11}}$ and c_{cp_5} are related by an inheritance relation then, we add a construction link between them. cp_{11} is blackened and we go back to the second adjacent of the cp_5 concept. cp_5 is gray and cp_9 is black, so (cp_5,cp_9) is a cross/forward edge, means that cp_9 has already been instantiated so, we add output reference nodes from c_{cp_6} until reaching the c_{cp_5} class. cp_5 is gray and it has finished his adjacent concepts so, it is instantiated within its ancestor and so on until backtracking to the concept c_{cp_1}, it is gray and it has not ancestors, so it is blackened and instantiated within the specification of the Prior_OOBN class. The first DFS tree is finished, so we choose an undiscovered node from the remainder nodes and we apply the algorithm until discovering all the concepts. The result of this process is shown in figure 5.(b)) shows the final result.

5 Related work

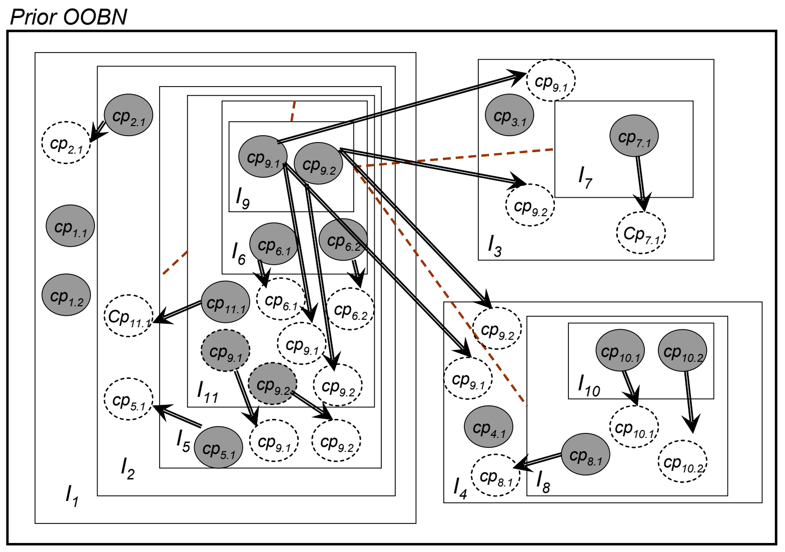
In recent years, a panoply of works have been proposed in order to combine PGMs and ontologies so that one can enrich the other. We can outline two main directions for these proposed approaches. The first aims to enhance ontologies capabilities to support probabilistic inference. While the second aims to enhance PGMs construction by integrating ontologies.

Ontologies provide a support for logical reasoning, but they do not support uncertainty. Hence, several extensions have been proposed to overcome this limitation. One line of research aims to extend existing ontology languages, such as OWL[1], to be able to catch uncertainty in the knowledge domain. Proposed methods, such as BayesOWL (Ding and Peng, 2004), OntoBayes (Yang and Calmet, 2005) use additional markups to represent probabilistic information attached to indi-

[1]ttp://www.w3.org/TR/2004/REC-owl-features-20040210/



(a) The ontology to morph



(b) The resulting OOBN

Figure 5: An example of ontology morphing to a prior OOBN structure

vidual concepts and properties in OWL ontologies. The result is then a probabilistic annotated ontology that can be translated into a BN to perform probabilistic inference. Other works define transition actions in order to generate a PGM given an ontology with the intention of extending ontology querying to handle uncertainty while keeping the ontology formalism intact (Bellandi and Turini, 2009).

On the other hand, some solutions proposed the use of ontologies to help PGMs construction. Some of them are designed for specific applications (Helsper and Van der Gaag,2002), (Zheng et al., 2008), while some others give various solutions to handle this issue. We can mention the semi-automatic approach provided in (Fenz et al., 2009) to create BNs and the SemCaDo (Semantical Causal DiscOvery) algorithm (Ben Messaoud et al., 2009) (Ben Messaoud et al., 2011) which ensure the integration of ontological knowledge, more precisely, subsumption relationships, to learn the structure of causal Bayesian networks (i.e. BNs with causal relations) (Pearl, 2000) and improve the causal discovery.

However, all these solutions are limited to a restrained range of PGMs, usually BNs. So, they neglect some ontology important aspects such as representing concepts having more than one property, non taxonomic relations, etc. In our approach we used OOBNs which are much richer graphical model than standard BNs. They allowed us to address an extended range of ontologies, we focused on concepts, their properties, hierarchical as well as semantic relations and we showed how these elements would be useful to automatically generate a prior OOBN. Our proposal concerns exclusively the OOBN structure definition through the use of ontologies.

6 Conclusion and future work

The crossing-over of PGMs and ontologies can allow us to improve relevant tasks related to each of them. In this paper, we showed how we take advantage of the semantic richness provided by ontologies to generate a prior OOBN structure and this is by exploring similarities between these two paradigms. The use of the OOBN framework has enabled us to handle an extended range of ontologies unlike works which were limited to the use of standard Bayesian networks, which brings us to say that this work is an initiative aiming to set up new bridges between these two paradigms.

The final structure resulting from the learning process may also be useful to make the initial ontology evolve, and this is by trying to find how the new relations discovered by the learning process can affect the (semi) automatic ontology enrichment process. Thus, as an ongoing work, we aim to analyze the elements that are common to both tasks and provide a two-way approach that uses ontology power in representing knowledge to help the hard process of OOBN structure learning by proposing new metrics, based on ontological knowledge, allowing to assess better the choice of the best structure. Then, uses novel relations discovered by the learning process in order to improve the hard activity of ontology enrichment.

References

O. Bangsø H. Langseth and T. D. Nielsen (2001). Structural learning in object oriented domains. *Proceedings of the Fourteenth Florida Artificial Intelligence Research Society Conference*, 340-344. Key West, Florida, USA:AAAI Press.

O. Bangsø and P-H. Wuillemin (2000). Object Oriented Bayesian networks: a framework for top-down specification of large Bayesian networks with repetitive structures. Aalborg University, Denmark.

A. Bellandi and F. Turini (2009). Extending Ontology Queries with Bayesian Network Reasoning. *Proceedings of the 13th International Conference on Intelligent Engineering Systems*, 165–170. Barbados.

M. Ben Messaoud, Ph. Leray and N. Ben Amor (2011). SemCaDo: a serendipitous strategy for learning causal bayesian networks using ontologies. *To appear in Proceedings of the 11th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, ?–?. Belfast, Northern Ireland.

M. Ben Messaoud, Ph. Leray and N. Ben Amor (2009). Integrating ontological knowledge for iterative causal discovery. *In Proceedings of the 10th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, 168–179. Verona, Italy.

Z. Ding and Y. Peng (2004). A probabilistic extension to ontology language OWL. *Proceedings of the 37th Hawaii International Conference On System Sciences.* Big Island, HI, USA.

S. Fenz, M. Tjoa and M. Hudec (2009). Ontology-Based Generation of Bayesian Networks. *Proceedings of the Third International Conference on Complex, Intelligent and Software Intensive Systems*, 712–717. Fukuoka, Japan.

N. Friedman (1998). The Bayesian structural EM algorithm. *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, 129–138. University of Wisconsin Business School, Madison, Wisconsin, USA:Morgan Kaufmann.

T. Gruber (1995). Toward Principles for the Design of Ontologies Used for Knowledge Sharing. *International Journal Human-Computer Studies* **43** (5–6): 907–928.

E. M. Helsper and L. C. van der Gaag (2002). Building bayesian networks through ontologies. *Proceedings of the 15th European Conference on Artificial Intelligence*, 680-684. Amsterdam.

J. Pearl (2000). *Causality: Models, reasoning and inference.* Cambridge.: MIT Press.

J. Pearl (1988). *Probabilistic reasoning in intelligent systems*, San Franciscos: Morgan Kaufmann.

D. Koller and A. Pfeffer (1997). Object-oriented Bayesian networks. *Proceedings of the thirteenth conference on Uncertainty in Artificial Intelligence*, 302–313. Providence, Rhode Island, USA: Morgan Kaufmann.

H. Langseth and O. Bangsø (2001). Parameter learning in object oriented Bayesian networks. *Annals of Mathematics and Artificial Intelligencen*, **32**:221-243.

H. Langseth and T. D. Nielsen (2003). Fusion of domain knowledge with data for structural learning in object oriented domains. *Journal of Machine Learning Research*, **4**:339-368.

Y. Yang and J. Calmet (2005). OntoBayes: An Ontology-Driven Uncertainty Model. *International Conference on Intelligent Agents, Web Technologies and Internet Commerce.* Vienna, Austria.

H-t. Zheng, Bo-Y. Kang and H-G. Kim (2008). An Ontology-Based Bayesian Network Approach for Representing Uncertainty in Clinical Practice Guidelines. *Uncertainty Reasoning for the Semantic Web I: ISWC International Workshops, URSW 2005-2007, Revised Selected and Invited Papers*, 161–173.

Building a Real-Time System for High-Level Person Recognition

Denver Dash
Intel Embedded Systems Science and Technology Center
Carnegie Mellon University
Pittsburgh, PA 15213
denver.h.dash@intel.com

Long Quoc Tran
Georgia Institute of Technology

Abstract

We describe a system that uses graphical models to perform real-time high-level perception. Our system uses Markov Logic Networks to relate entities in images via first-order logical sentences to perform real-time semi-supervised person recognition. The system is a collection of "commodity-level" vision algorithms such as the Viola-Jones face detector, histogram matching and even low-level pixel comparisons, together with logical relationships such as mutual exclusivity and entity confusion combined with a small number of labeled examples into a Markov random field which can be solved to provide labels for faces in the images. We describe the methodology for constructing the logical relations used for the system, and the (many) pitfalls we encountered despite the small number of relations used. We also discuss several future approaches to achieve interactive speeds for such a system, including bounding the size of the graph using temporal weighting of instances, approximating the structure of the graphical model, parallelizing graphical model inference, and low-level hardware acceleration.

Introduction

In this paper, we describe a real-time system[1] (Figure 1) for recognition using a small labeled dataset plus first-order logic relations. The system assumes a constrained environment, i.e., one in which the same people generally occur and that the instances that we want to classify are localized in time.

[1]A batch version of this system along with empirical results was described by Chechetka et al. (2010). Here we focus more on the implementation details as well as the knowledge management of the system.

In recent years there has been an explosion of work on exploiting in-frame context for entity classification in images (Torralba, 2003, Kumar and Hebert, 2005, Torralba et al., 2005, Heitz and Koller, 2008, Heitz et al., 2008, Gupta and Davis, 2008, Rabinovich and Belongie, 2009, Gould et al., 2009). The work typically involves finding some useful relations for the specific domain at hand, e.g., "the sky is usually above the ground"(Gupta and Davis, 2008), building a customized conditional random field model over the entities in a frame and jointly classifying each entity in an image given the observed pixel values. Despite these successes, at present few if any practical real-time systems exist that attempt to do high-level reasoning by integrating context at a high-level. In this paper, we discuss our attempts at building such a system using Markov Logic Networks (MLNs) and by constructing a database of logical relationships that are useful for relating entities to be identified.

This paper also makes the point that MLNs provide a uniform, intuitive and modular interface for performing high-level perception. More importantly, we show that MLNs can provide a newer more global sense of context that allows them to jointly classify an entire dataset of images (entities), using meaningful relations between these entities, in a manner similar to the collective classification of citation entries done by Singla and Domingos (2006). The image representation provides a wealth of relations that can be brought to bear on the problem, such as mutual exclusivity of multiple faces in an image, temporal and spatial stratification, personal traits that may relate people to various objects or distinctive clothes, etc. We thus expect that this application is even more suited for the use of a powerful tool like MLNs than the case of citation matching.

This use of MLNs for collective classification resembles graph-based semi-supervised learning (SSL) approaches (c.f., Fergus et al., 2009), which relate entities across a corpus via a distance or similarity measure. However, compared to SSL approaches, MLNs provides a much richer way of connecting labeled/unlabeled instances, allowing one to combine multiple similarity metrics at the same time

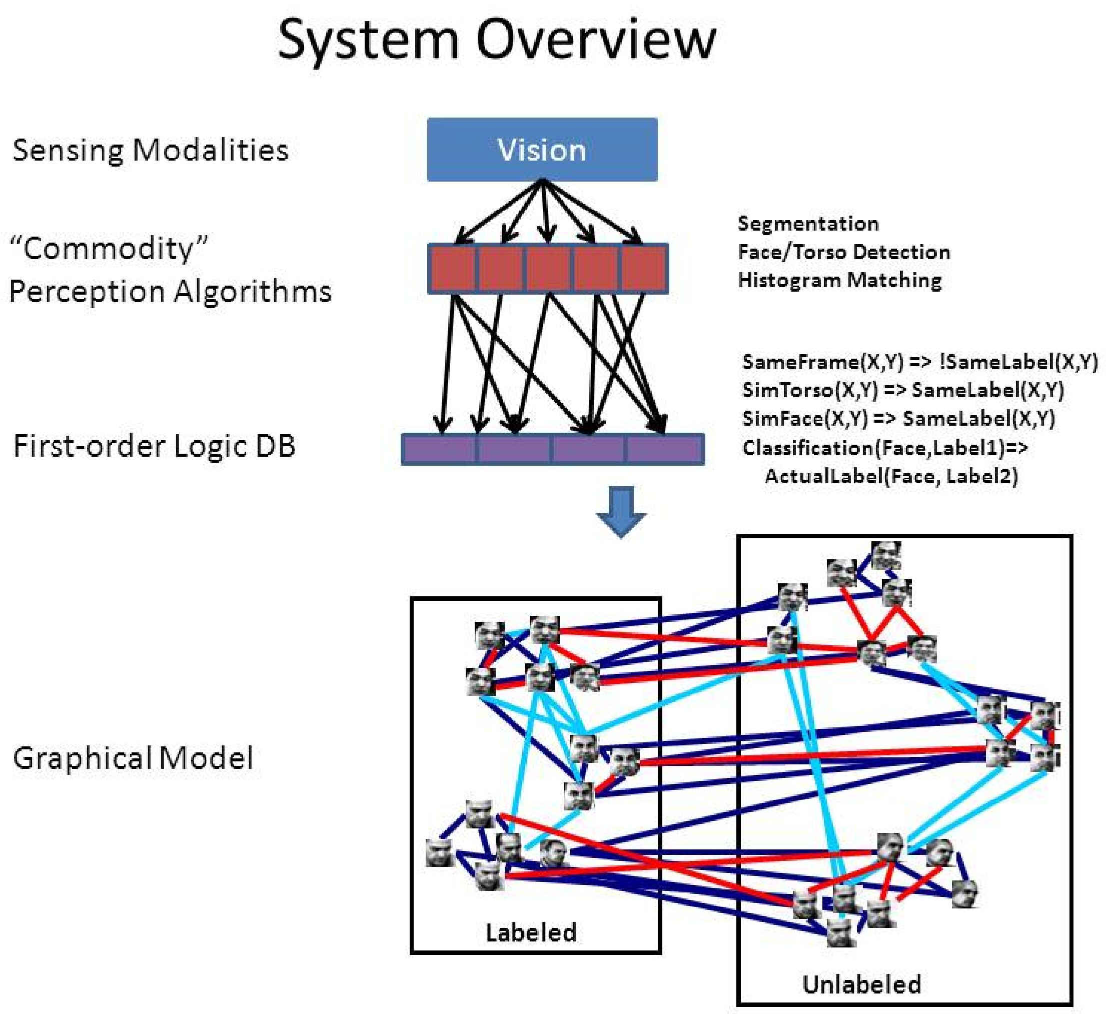

System Overview
Sensing Modalities
Vision
"Commodity" Perception Algorithms
Segmentation
Face/Torso Detection
Histogram Matching
SameFrame(X,Y) => !SameLabel(X,Y)
SimTorso(X,Y) => SameLabel(X,Y)
SimFace(X,Y) => SameLabel(X,Y)
Classification(Face,Label1)=> ActualLabel(Face, Label2)
First-order Logic DB
Graphical Model
Labeled
Unlabeled

Figure 1: System Overveiw

as well as incorporate arbitrary logical relationships. In fact we argue that MLNs can provide an approximate generalization to some of the standard SSL approaches by discretizing a distance/similarity measure and incorporating them into the MLN. In addition one can continue to exploit other relations that would not fit well within the SSL framework, such as contextual information that relates entities within frames. We show empirically that this approach yields favorable results for face recognition in images of three datasets collected by us, and that the use of the additional logical relations, which would be difficult in standard SSL, is crucial for the best classification accuracy.

MLN Background

Markov logic (Richardson and Domingos, 2006) is a probabilistic generalization of finite first-order logic. A Markov logic network (MLN) consists of a set of weighted first-order clauses. Given a set of constants, an MLN defines a Markov network with one binary variable for every ground atom and one potential for every possible grounding of every first-order clause. The joint probability distribution over the ground atom variables is defined as

$$P(x) = \frac{1}{Z} \exp\left\{ \sum_f \sum_{x_i} w_f f(x_i) \right\}, \qquad (1)$$

where f is an indicator function corresponding to a first-order clause (1 if that clause is true and 0 otherwise), w_f is a weight of that clause, and x_i is the set of ground atom variables in a particular grounding of that clause. The inner summation in (1) is over all possible groundings. Therefore, for every grounding of every first-order clause, the higher the weight for that clause, the more favored are assignments to x where that grounding is true.

Two fundamental problems in Markov logic that apply to our application are those of *learning optimal weights* for the known set of first-order clauses given the knowledge base of known ground atoms, and *inference*, or *finding the most likely assignment* to unknown ground atoms given the knowledge base. Even though both problems are intractable in general, well-performing approximate algorithms are available. For weight learning, we used preconditioned conjugate gradient with MC-SAT sampling implemented in the Alchemy package (Kok et al., 2009). For inference, we used a high-performance implementation of residual belief propagation (Gonzalez et al., 2009) along with a lazy instantiation of MLN structure as recommended by Poon et al. (2008).

Model Description

In the existing literature, many types of very different features have been shown to be useful for face recognition (and object recognition more generally). In particular, SSL approaches (Fergus et al., 2009) exploit similarity in object appearances in different images to propagate label information from labeled to unlabeled blobs, and between unlabeled blobs. In a supervised setting, typically a low-dimensional representation of blob appearances is extracted (e.g., Turk and Pentland, 1991) and a standard technique such as a support vector machine (Vapnik, 1995) is then applied. Besides the blob appearance information, it has been shown that taking context in which the blob appears, such as blob location within the frame or labels of other objects in the scene, is crucial to accurate object recognition. In this section, we show that all the above sources of information can be combined efficiently using a Markov logic network. Our approach thus combines the advantages of the diverse existing approaches to improve face recognition accuracy. In the MLN described below, we will use the query predicate $\texttt{Label}(\texttt{b},\texttt{o})$, which is true if and only if blob $\texttt{b}$ has label $\texttt{o}$. The evidence predicates will be introduced gradually, as they are needed for the MLN rules.

We assume that face *detection* has already been performed by some standard approach, such as that of Viola and Jones (2001). The input to our system thus consists of a set of images, and for each image, a set of bounding boxes for the detected faces, some of which are labeled with people's names. The goal is to assign labels to the remaining unlabeled face blobs.

Label propagation: semi-supervised component

A key idea of the SSL approaches is to classify all the objects of the test set *simultaneously* by rewarding the cases of similar-looking objects having the same label (equivalently, penalizing labels mismatches for similarly looking objects). Let x_i and x_j be the appearances of blobs $\texttt{b}_\texttt{i}$ and $\texttt{b}_\texttt{j}$ respectively. Denote $\|x_i - x_j\|$ to be the distance between x_i and x_j. We define the evidence predicate $\texttt{SimilarFace}(\texttt{b}_\texttt{i},\texttt{b}_\texttt{j})$ that is true if and only if $\|x_i - x_j\| < \Delta_f$, where Δ_f is a threshold. Then the rule to favor matching labels for similar faces is simply

$$\texttt{SimilarFace}(\texttt{b}_\texttt{i},\texttt{b}_\texttt{j}) \wedge \texttt{Label}(\texttt{b}_\texttt{i},\texttt{o}) \Rightarrow \texttt{Label}(\texttt{b}_\texttt{j},\texttt{o}) \qquad (2)$$

We selected threshold Δ_f so as to get precision 0.9 on the training data: $\frac{\sum_{\texttt{i},\texttt{j}} I(\texttt{SimilarFace}(\texttt{b}_\texttt{i},\texttt{b}_\texttt{j})=\texttt{true})}{\sum_{\texttt{i},\texttt{j},\texttt{o}} I(\texttt{Label}(\texttt{b}_\texttt{i},\texttt{o})=\texttt{Label}(\texttt{b}_\texttt{j},\texttt{o}))} = 0.9$, where $I(\cdot)$ is the indicator function. For simplicity of implementation, we used 16-bin color histograms as representations for x_i and χ^2 distance $\|x_i - x_j\|_{\chi^2} \equiv \sum_{k=1}^{\#bins} \frac{(x_i(k)-x_j(k))^2}{x_i(k)+x_j(k)}$. Naturally, any other choice of representation and distance can be used instead.

Observe that similar face appearance is not the only possible clue that two image fragments actually depict the same person. For example, similar clothing appearance is another useful channel of information, as was demonstrated by Sivic et al. (2006). In our approach, information about clothing appearance similarity is used in the same way to the face similarity: for every face blob $\mathtt{b_i}$, we define the corresponding torso blob $\mathtt{t_i}$ to be a rectangle right under $\mathtt{b_i}$; the scale of the rectangle is determined by the size of $\mathtt{b_i}$. Let y_i be the appearance representation of $\mathtt{t_i}$. We define the evidence predicate $\mathtt{SimilarTorso}(\mathtt{b_i}, \mathtt{b_j})$ which is true if and only if $\|y_i - y_j\| < \Delta_t$ and introduce the corresponding label smoothing rule

$$\mathtt{SimilarTorso}(\mathtt{b_i}, \mathtt{b_j}) \wedge \mathtt{Label}(\mathtt{b_i}, \mathtt{o}) \Rightarrow \mathtt{Label}(\mathtt{b_j}, \mathtt{o}) \tag{3}$$

into the MLN. One can see that we have two versions of essentially the same rule exploiting different channels of information for label propagation. Even though it is possible in principle to achieve the same effect in standard graph Laplacian-based SSL approaches (Fergus et al., 2009), one would need to use costly cross-validation to find a good way to combine the two separate distance metrics into one (alternatively, find the relative importance of the torso distance and face distance metrics). In contrast, standard algorithms for MLN weight learning provide our approach with the relative importance of the two rules automatically.

More fine-grained label smoothing. One advantage of the graph Laplacian-based unsupervised methods over our approach is that the former naturally support real-valued blob similarity values, while our approach requires thresholding. However, our approach can also be adapted to handle varying degrees of similarity: instead of a single similarity threshold, one can use multiple different similarity thresholds and introduce corresponding similarity predicates. For example, suppose we want to use two different thresholds, $\Delta_f^{(1)} < \Delta_f^{(2)}$, for face blob similarity. Then we would introduce two similarity predicates, $\mathtt{SimilarFace}^{(1)}(\mathtt{b_i}, \mathtt{b_j})$, which is true if and only if $\|x_i - x_j\| < \Delta_f^{(1)}$, and analogously $\mathtt{SimilarFace}^{(2)}(\mathtt{b_i}, \mathtt{b_j})$, for $\Delta_f^{(2)}$. Then for highly similar blobs, those with $\|x_i - x_j\| < \Delta_f^{(1)}$, both versions of the formula in Eq. 2 for $\mathtt{SimilarFace}^{(1)}$ and $\mathtt{SimilarFace}^{(2)}$ will have the left-hand side to be true, providing a higher reward for matching the labels. On the other hand, for weakly similar blobs, those with $\Delta_f^{(1)} < \|x_i - x_j\| < \Delta_f^{(2)}$, only the version of Eq. 2 corresponding to $\mathtt{SimilarFace}^{(2)}$ will have the LHS to be true, providing a weaker reward for matching labels.

Exploiting single-image context

In addition to the appearance of the blob of interest itself and the labels of similar blobs in other images, powerful contextual cues often exist in the image containing the blob. In the broader context of object recognition, spacial context (e.g. sky is usually in the top part of an image), co-occurrence (computer keyboards tend to occur together

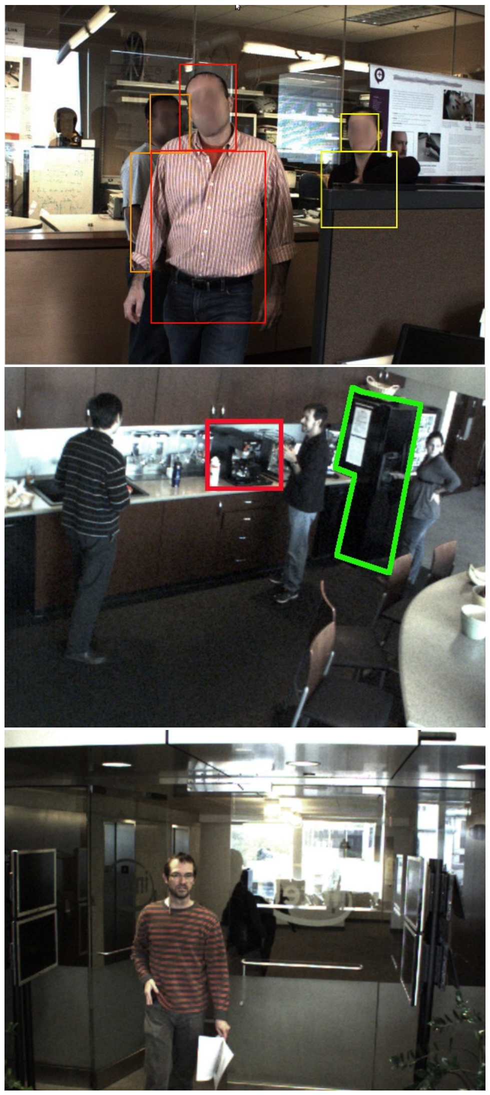


Figure 2: Example security images for datasets 1–3 (top to bottom). The top image shows an example of torso extraction (faces on this set have been blurred by subjects' request). The middle image shows a view of a kitchen area where a coffee machine (red) is in the middle of the frame, while the refrigerator (green) is on the right; thus coffee drinkers might be more likely to appear in the middle.

with monitors) and broad scene context (fridges usually occur in kitchen scenes) have all been shown to enable dramatic improvements in recognition accuracy. Here, we describe the MLN rules used by our system to take single-image context into account.

A person only occurs once in an image. In the absence of mirrors, for every person at most one occurrence of their respective face is possible in a single image. Therefore, if two faces are present in the same image, they necessarily have to either have different labels, or be both labeled as unknown. Hence we introduce an evidence predicate

$\texttt{SameImage}(\texttt{b}_\texttt{i}, \texttt{b}_\texttt{j})$ which is true if and only if $\texttt{b}_\texttt{i}$ and $\texttt{b}_\texttt{j}$ are in the same image, and the following MLN rule:

$$\texttt{SameImage}(\texttt{b}_\texttt{i}, \texttt{b}_\texttt{j}) \Rightarrow !\texttt{Label}(\texttt{b}_\texttt{j}, \texttt{o}_1) \vee \\ !\texttt{Label}(\texttt{b}_\texttt{j}, \texttt{o}_2) \vee (\texttt{o}_1 != \texttt{o}_2) \vee (\texttt{o}_1 == \texttt{Unknown}) \quad (4)$$

Face location. For multiple images taken with the same camera pose, such as images from a security camera, often different people will tend to occupy different parts of the frame. For example, in the middle image of Fig. 2 the refrigerator is in the right part of the frame, and the coffee machine is in the middle. Therefore, faces of coffee drinkers may be more likely to appear in the middle of the frame, while those preferring soft drinks may spend more time in the right part. In addition, false-positive face detections (which are given the label "junk") will appear randomly whereas actual faces appear in more constrained locations. Using the spacial prior in such settings will benefit the recognition accuracy. In our approach, we subdivide every image into 9 tiles of the same size, arranged in a 3×3 grid and introduce an evidence predicate $\texttt{InTile}(\texttt{b}, \texttt{tile})$ and an MLN rule capturing the spacial prior:

$$\texttt{InTile}(\texttt{b}, +\texttt{tile}) \Rightarrow \texttt{Label}(\texttt{b}, +\texttt{o})$$

Notice we use the Alchemy convention $+\texttt{tile}$ and $+\texttt{o}$, meaning that for every combination of the tile and label a separate formula weight will be learned, yielding different priors over the face labels for different regions of the image.

Time of the day. Similar to face location, a time-dependent label prior is also useful when processing images from security cameras: "early birds" will be more likely to occur in images taken earlier in the day and vice versa. We subdivide the duration of the day into 3 intervals: morning (before 11AM), noon (11AM to 2PM) and evening (after 2PM), introduce an evidence predicate $\texttt{TimeOfDay}(\texttt{b}, \texttt{time})$ and the corresponding MLN rule:

$$\texttt{TimeOfDay}(\texttt{b}, +\texttt{time}) \Rightarrow \texttt{Label}(\texttt{b}, +\texttt{o})$$

Again, to obtain a time-dependent label prior we force the system to learn a separate weight for every combination of the time interval and face label.

One can see that extracting the relations introduced in this section requires little preprocessing, and it is possible to come up with similar common-sense relations to improve accuracy for settings other than security camera image sequences.

Plugging in existing face recognizers

The relations and predicates described so far only use simple representations and similarity metrics. However, there is a large amount of existing literature and expert knowledge dealing with design of representations, distance metrics and integrated face recognition systems that improve accuracy significantly over simpler baselines in a supervised setting. If such a recognition system is available, it is desirable to be able to leverage its results in our framework instead of completely discarding the existing system and replacing it with the MLN model. Fortunately, it is easy to combine any existing face recognition system with our approach by using the face labels produced by the existing system as observations in our model. Formally, we use an evidence predicate $\texttt{ObservedLabel}(\texttt{b}, \texttt{observedLabel})$, which is true if and only if the external face recognition system assigned $\texttt{observedLabel}$ as the label for blob b. The MLN rule

$$\texttt{ObservedLabel}(\texttt{b}, +\texttt{observedLabel}) \Rightarrow \texttt{Label}(\texttt{b}, +\texttt{o}) \quad (5)$$

then provides the observation model. Observe that several different external classifiers can be used as observations simultaneously, by mapping the labels produced by different classifiers to disjoint sets of atoms. For example, if there are two different classifiers, $\texttt{clf}_1$ and $\texttt{clf}_2$, and both label blob $\texttt{b}_1$ as $\texttt{John}$, then one would set two ground predicates to true: $\texttt{ObservedLabel}(\texttt{b}_1, \texttt{John\_clf}_1)$ and $\texttt{ObservedLabel}(\texttt{b}_1, \texttt{John\_clf}_2)$. Again, as in the case of multiple measures of blob similarity, MLN weight learning would automatically determine the relative importance and reliability of the two classifiers by assigning corresponding weights to the groundings of the observation model.

We used a boosted cascade of Haar features as given by Viola and Jones (2001) for face detection, and face recognizer of Kveton et al. (2010) as observations for the MLN rule in Eq. 5. This classifier is based on calculating the L_2 distance in pixel space for down-sampled (92×92 resolution) and normalized images. This method was shown by Sim et al. (2000) to be generally superior to the more common method based on PCA for face classification in single images. For evaluating torso similarity for $\texttt{SameTorso}$ evidence predicate, simple torso occlusion handling was performed by assuming that larger faces were in the foreground. Thus, larger-faced torsos were assumed to lie in front of smaller-faced torsos, and the resulting torso bounding boxes did not intersect (see Fig. 2 for an example).

Results

Quantitative results for a batch version of this model were presented by Chechetka et al. (2010). Here, for some added context, we just present some of the qualitative lessons learned from those experiments.

Exploiting additional information channels dramatically improves accuracy. Classification error is reduced by our approach by a factor from 1.35 to 5.2 compared to the baseline of Kveton et al. (2010). Such an improvement confirms the long-standing observation that using the context, such as time of the day, is crucial for achieving high recognition accuracy. It also shows that the framework of

Markov logic is an efficient way to combine the multiple sources of information, both within a single image, and multiple types of relations between different images, for the goal of face recognition.

No single relation accounts for the majority of the improvement. Over all the dataset, the most extreme single-relation accuracy improvement over the baseline of Kveton et al. (2010) (`InTile` predicate and the corresponding location prior is less than 40% of the total performance improvement of the full model over the baseline. Therefore, the multiple relations of our full model are not redundant and represent information channels that complement each other. It is the interaction of multiple relations that enables significant accuracy improvements.

Relation importance is not uniform across datasets. One can see that the effect of the same relation can be dramatically different for different datasets, depending on those datasets' properties. Only label propagation via the `SimilarTorso` relations provides a consistently significant performance improvement, the effect of other relations is much more varied. The varying degree of relation importance for different datasets makes it important for a face recognition approach to be easily adjustable to emphasize important relations and ignore the unimportant ones. Fortunately, the Markov logic framework makes such adjustability extremely easy on two levels. First, learning the weights of the formulas automatically assigns large weights to important formulas and close to zero weight to irrelevant ones. Second, any relation or formula can be easily taken out of the model or put back in, enabling the search for the optimal set of relations using cross-validation.

Building a Real-time system

In this section, we explore how the ideas in this paper can be augmented into a real-time system. There are two broad objectives that need to be addressed:

1. Updating the model as new instances come in (online learning).

2. Performing graphical model inference at interactive speeds (online inference).

To perform these two tasks simultaneously, we chose an asynchronous architecture (Figure 3) where learning and inference are performed in separate processes. This provided a natural parallelism for the whole system. Even with this parallelized approach, both learning and inference components required special enhancements to enable real-time operation. Online learning was necessary whenever a new labeled instance was observed by the system. This could happen whenever Incorporating new instances caused the graph structure to change.

Online inference was triggered whenever a new unlabeled instance was observed. In this case, the structure of the graph was altered. Specifically, new nodes corresponding to new instantiations of all propositions involving the new observed faces will be added to the network. At this point, since the structure of the network has changed, the beliefs of the network are necessarily invalidated. Thus, an exact algorithm would run belief propagation over the entire graph after such events occurred. In our system, we avoided this with the following heuristic: we maintained the current beliefs of the network (as of the last iteration), and we pushed the beliefs of the new nodes on the top of the priority queue in the Residual BP calculation. This had the effect of focusing the next round of computation on the new nodes until convergence was reached.

Related Work

There exists quite a lot of work now on incorporating relations into image classification. Rabinovich and Belongie (2009) provides a good overall review of this work, and contrasts "scene-based" and "object-based" context. The former methods are represented by (Torralba, 2003, Kumar and Hebert, 2005, Heitz and Koller, 2008, Heitz et al., 2008), which all attempt to understand the scene ("the gist") before trying to recognize objects. Gould et al. (2009) and Torralba et al. (2005) use MRFs to do joint segmentation and object recognition by exploiting physical relations between entities. Gupta and Davis (2008) uses prepositions present in annotated images to help determine relative positions of objects in images. For example, if an image is annotated with "car on the street", one might infer that a car is above a street in the image. Many of these efforts have a different aim from our work. Namely, they attempt to do object class detection, i.e., detect all the objects of some given classes in an image; whereas in our face recognition application, we are doing object-instance recognition: given the presence of objects of a given type, find specific labels for those objects. On the other hand, these methods have in common with us the intent to exploit physical relations between objects and abstract relations between a set of objects and the gist of a scene to improve their results. The difference between their application of this principle and ours is that they all attempt to relate entities across a single image; whereas we use cross-image relationships. Second, by using the framework of Markov Logic, we have a unified, automated mechanism to add arbitrary relations and automatically generate the CRF.

Fergus et al. (2009) and Kveton et al. (2010) present approximations to the graph Laplacian-based semi-supervised learning solution for classifying images. These methods in general have the advantage over our method that they allow continuous similarity measures rather than our discretized version, and they can be solved efficiently. However, these approaches are typically restricted

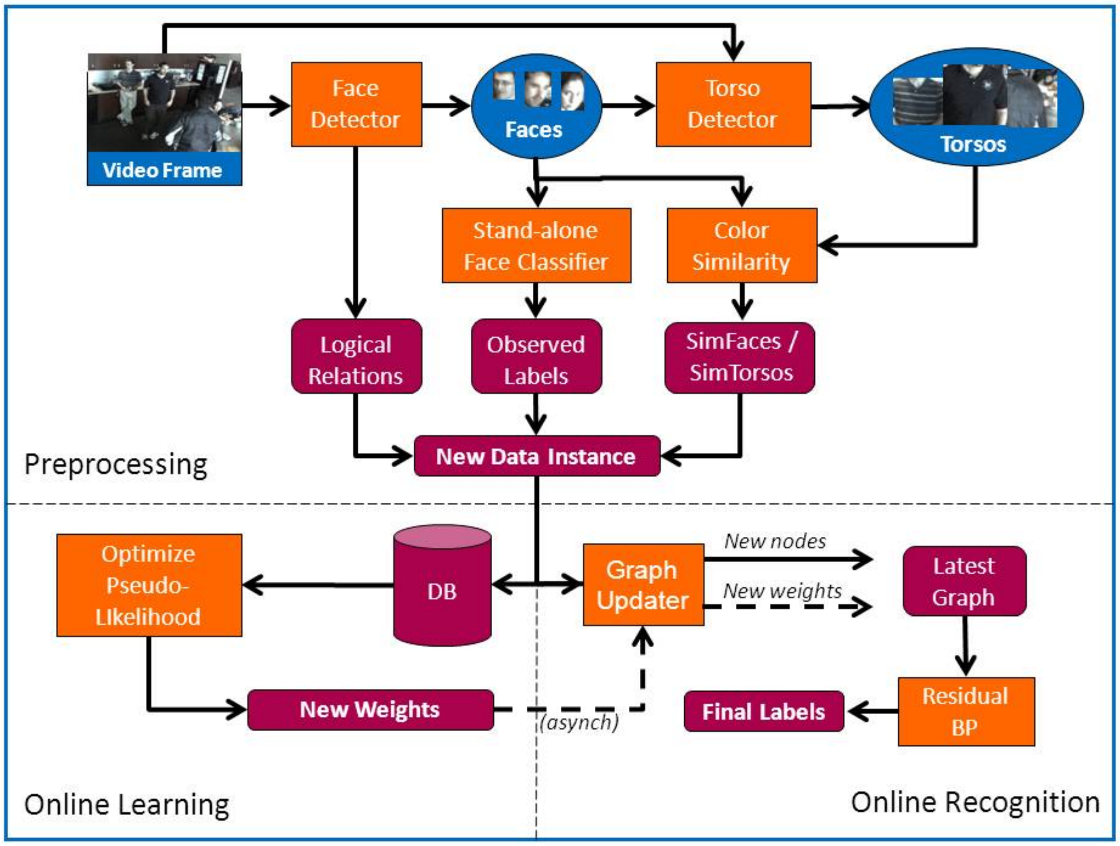


Figure 3: Real time system architecture

to similarity-based classification; whereas we can incorporate much more general relations such as our mutual exclusivity. Furthermore, our approach can easily incorporate any of these classifiers (as we do in this paper by taking the classifier of Kveton et al. (2010)) and use them as core face recognizers in an object model. Finally, our approach can approximate these approaches (albeit much less efficiently) by using a discretized version of a similarity-measure, as we do using face and torso histograms in this work.

Conclusions

Our contributions in this paper are as follows: First, we present a real-time perception system that incorporates Markov Logic for multilabel classification in images. Whereas there has been much existing research showing the benefits of exploiting local and global in-frame context, they all have involved custom-made graphical models and therefore are less accessible as a general modeling tool for specific domains. Second, we show that Markov Logic can also provide a powerful new type of context for collective classification across frames, especially when the database is expected to have many repeated shots of the same entity in different circumstances. We have argued that this type of context generalizes graph-based SSL approaches, and adds much to these approaches in the expressibility of the relations across frames that can guide the collective classification of entities. Thus, we show that Markov Logic can provide a beneficial unification of two quite dissimilar cutting-edge techniques for entity classification in images. Finally, for the specific case of person identification, we have shown empirically that relations such as clothing preferences, mutual exclusivity, spatial and temporal stratification as well as multiple similarity channels can dramatically improve face recognition over the state-of-the-art. Although much work remains to be done, we present some of the specific modeling issues involved with this system, as well as some of the obstacles to making the system operate at interactive speeds.

References

A. Chechetka, D. Dash, and M. Philipose. Relational learning for collective classification of entities in images. In *Workshop on Statistical Relational AI in conjunction with the Twenty-Fourth Conference on Artificial Intelligence (AAAI-10)*, Atlanta, Georgia, 2010.

R. Fergus, Y. Weiss, and A. Torralba. Semi-supervised learning in gigantic image collections. In *NIPS*. 2009.

J. Gonzalez, Y. Low, and C. Guestrin. Residual splash for optimally parallelizing belief propagation. In *AISTATS*, 2009.

S. Gould, T. Gao, and D. Koller. Region-based segmentation and object detection. In *NIPS*. 2009.

A. Gupta and L. S. Davis. Beyond nouns: Exploiting prepositions and comparative adjectives for learning visual classifiers. In *ECCV*, 2008.

G. Heitz and D. Koller. Learning spatial context: Using stuff to find things. In *ECCV*, 2008.

G. Heitz, S. Gould, A. Saxena, and D. Koller. Cascaded classification models: Combining models for holistic scene understanding. In *NIPS*. 2008.

S. Kok, M. Sumner, M. Richardson, P. Singla, H. Poon, D. Lowd, and P. Domingos. The alchemy system for statistical relational AI. Technical report, Department of Computer Science and Engineering, University of Washington, Seattle, WA., 2009. URL `http://alchemy.cs.washington.edu/`.

S. Kumar and M. Hebert. A hierarchical field framework for unified context-based classification. In *ICCV*, 2005.

B. Kveton, M. Valko, A. Rahimi, and L. Huang. Semi-supervised learning with max-margin graph cuts. In *to appear, AISTATS*, 2010.

H. Poon, P. Domingos, and M. Sumner. A general method for reducing the complexity of relational inference and its application to mcmc. In *AAAI*. AAAI Press, 2008.

A. Rabinovich and S. Belongie. Scenes vs. objects: a comparative study of two approaches to context based recognition. In *International Workshop on Visual Scene Understanding (ViSU)*, Miami, FL, 2009.

M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62(1–2):107–136, Feb 2006.

T. Sim, R. Sukthankar, M. Mullin, and S. Baluja. Memory-based face recognition for visitor identification. In *Proceedings of International Conference on Automatic Face and Gesture Recognition*, 2000.

P. Singla and P. Domingos. Entity resolution with markov logic. In *ICDM*, 2006.

J. Sivic, C. L. Zitnick, and R. Szeliski. Finding people in repeated shots of the same scene. In *Proceedings of the British Machine Vision Conference*, 2006.

A. Torralba. Contextual priming for object detection. *International Journal of Computer Vision*, 53(2):169–191, July 2003.

A. Torralba, K. P. Murphy, and W. T. Freeman. Contextual models for object detection using boosted random fields. In *NIPS*. 2005.

M. A. Turk and A. P. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

V. N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.

P. Viola and M. Jones. Robust real-time object detection. In *International Journal of Computer Vision*, 2001.

Constructing Bayesian Networks for Linear Dynamic Systems

Sander Evers
Radboud University Nijmegen
Inst. for Computing and Information Sciences
Nijmegen, the Netherlands

Peter J.F. Lucas
Radboud University Nijmegen
Inst. for Computing and Information Sciences
Nijmegen, the Netherlands

Abstract

Building probabilistic models for industrial applications cannot be done effectively without making use of knowledge engineering methods that are geared to the industrial setting. In this paper, we build on well-known modelling methods from linear dynamic system theory as commonly used by the engineering community to facilitate the systematic creation of probabilistic graphical models. In particular, we explore a direction of research where the parameters of a linear dynamic system are assumed to be uncertain. As a case study, the heating process of paper in a printer is modelled. Different options for the representation, inference and learning of such a system are discussed, and experimental results obtained by this approach are presented. We conclude that the methods developed in this paper offer an attractive foundation for a methodology for building industrial, probabilistic graphical models.

1 Introduction

As part of the manual construction process of a Bayesian network for an actual problem one needs to somehow translate knowledge that is available in the problem domain to an appropriate graphical structure. Problem characteristics also play a major role in the choice of the type of the associated local probability distributions. The whole process can be looked on as a form of *knowledge engineering*, where the actual construction of the Bayesian network is only one of the many needed activities in the development process. The acquisition of knowledge from domain experts is traditionally seen as one of the most important bottlenecks in knowledge engineering. It is well known that this can be greatly alleviated if there is an easy mapping from the informal ways domain knowledge is described, in documents or verbally by experts, to the Bayesian network formalism [7]. A typical example of such a mapping is the exploitation of available causal knowledge in a particular domain; often, causal knowledge can be easily translated to an initial graph structure of a Bayesian network, which can be refined later, for example by examining the conditional independence relationship represented by the resulting network. Fields where causal knowledge has been successfully used in knowledge engineering for Bayesian networks include medicine and biology.

However, for industrial applications the situation is somewhat different. This is mainly because in time, engineers have developed their own notational conventions and formalisms to get a grip on the domain of concern in the system-development process. In addition, industrial artifacts are designed by humans, and already early in the design process models are available that also can be used for other purposes: model-based design and development is here the central paradigm. Thus, rather than replacing methods from engineering by some new and unrelated methods, a better option seems to be to deploy as many of the engineering principles, methods, and assumptions as possible. Linearity is one of the assumptions frequently used, as it facilitates the development of complex models as industrial applications often are. Another commonly used method is the use of diagrams that act as abstractions of the system being developed. Diagrams act as important means for communication. Ideally, one would like to use similar diagrams as a starting point for building Bayesian networks or related probabilistic graphical models.

In this paper we explore these ideas by taking *Linear Dynamic Systems*, LDS for short, as a start for the construction process. LDS models enjoy a well-developed theory and practice, as they are widely used throughout many engineering disciplines for *tracking* system behaviour (cf. the well-known Kalman filter [4]) as well as for *controlling* this behaviour. Like Bayesian networks, an LDS can often be represented by a graphical diagram, which facilitates documentation and communication of the model among experts and non-experts.

Although an LDS is deterministic in nature, it is often used in situations that involve uncertainty. The Kalman filter is the canonical example here: given some noisy observations, it determines the expected current state of the system (mean and variance). However, the Kalman filter only accounts for one specific type of uncertainty: additive linear Gaussian noise on the state and output variables.

In this article, we explore a different direction of augmenting an LDS with probabilities: we regard the *parameters* as unknown. This allows us for example to model a printer that heats different types of paper, in which it is uncertain what the current type is. Previous Bayesian networks modelling this situation were developed in a laborious ad hoc manner by close cooperation of domain experts and probabilistic modelling experts [3]; in this article we aim for a more systematic approach.

2 LDS models and their role in the engineering process

2.1 Basic definitions

In its most basic form, a *Linear Dynamic System* or LDS describes a system's behaviour by the differential equation

$$\tfrac{d}{dt}\mathbf{x}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)$$

known as the *state-space representation.* Here, vector $\mathbf{x}(t)$ represents the system's state at time t, $\mathbf{u}(t)$ its input at time t, and matrices $\mathbf{A}$ and $\mathbf{B}$ describe how the current *state change* depends linearly on the current state and input.

This is a continuous-time representation; in order to calculate with LDS models, time is usually discretized. In this article, we therefore use the simple discretized model

$$\mathbf{x}_{t+1} - \mathbf{x}_t = \mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{u}_t$$

in which the size of the discretization steps conveniently equals the unit of the time domain in order to simplify the exposition (in practice one can use discretization steps of size Δt and scale the $\mathbf{A}$ and $\mathbf{B}$ matrices with this factor). The equation can then be rewritten to:

$$\begin{aligned} \mathbf{x}_{t+1} &= \mathbf{A}_d\mathbf{x}_t + \mathbf{B}_d\mathbf{u}_t \\ \mathbf{A}_d &= \mathbf{A} + \mathbf{I} \\ \mathbf{B}_d &= \mathbf{B} \end{aligned} \tag{1}$$

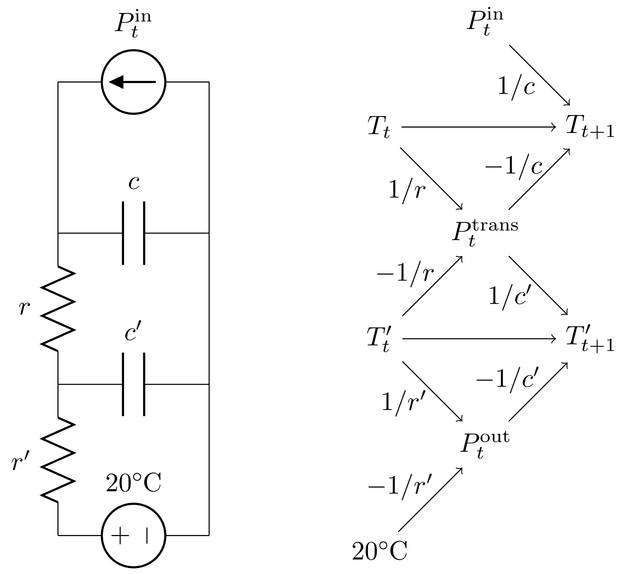


Figure 1: LDS model of paper heater. **Left:** as an electrical diagram, where voltage represents temperature and current represents heat flow. From top to bottom, the diagram consists of a (1) time-variable heat source, (2) heat mass with capacity c, (3) heat resistance r, (4) heat mass with capacity c', (5) heat resistance r', (6) heat sink with temperature 20°C modelling the constant flow of paper. **Right:** as a graphical representation of the state-space equations for two discrete time points $t, t{+}1$. The state of the system is described by the T variables, which represent the temperatures of the two heat masses. Its input is P_t^{in}, the power entering the system. Auxiliary variables represent the power flowing from the first heat mass to the second (P_t^{trans}), and from the second to the heat sink (P_t^{out}). **Note:** The *parameters* of the system are r, c, r', c' (instantiated with concrete values in a real system).

2.2 Role in engineering

In the engineering process, LDS models of systems are often represented by means of diagrams. We exemplify the role of these models and diagrams using a case study which remains our running example thoughout the paper. The case study originates from a manufacturer of industrial printers. To ensure print quality, paper needs to be heated to a certain temperature, which is accomplished by passing the paper along a metal heater plate. It is quite important that the paper reaches the right temperature. If it is too cold, print quality suffers; if it is too hot, energy (and money) is wasted or worse: the printer might malfunction. Therefore, engineers have put a lot of effort in the accurate modelling of the heating process. This results in models such as Fig. 1, in which the heater is modelled as two distinct heat masses: when the heater is powered, the first mass is directly heated, thereby indirectly heating the second mass, which transfers the heat to the paper.[1] In the diagram, the heating dynamics are represented as an *electrical circuit*, where temperature plays the part of voltage and heat flow that of current. A diagram like this has important advantages for the engineering process:

- It is very well suited for *documentation and communication.* A trained engineer can read the system's basic dynamic behaviour off this diagram in a blink of an eye; for a non-expert with a science background it is relatively easy to gain some intuition.
- It has a *formal meaning* as an LDS; it translates into the state-space equations in the form of Eq. (1), connecting it to a vast body of theoretical and practical knowledge.
- It *separates qualitative and quantitative aspects* of the model; the former are determined by the diagram structure, the latter by the parameters.
- It is *composable*: other models like this can be developed independently and joined into a larger system.
- It is *supported by software*: drawing and managing modules of electrical circuits (and also other graphical forms like bond graphs [5] and schematic diagrams) can be done by tools like 20-sim [11], which can also perform the translation to the state-space representation. This representation can be used for simulation, e.g. in MATLAB.

However, it is confined to modelling deterministic behaviour. In the realm of probabilistic modelling, the formalism of *Bayesian networks* shares the above attractive properties. A natural question is therefore: how can we combine these well-known LDS models with Bayesian networks?

Specifically, this paper will explore the situation where the *parameters* of the system (in this case: r, c, r', c') involve uncertainty. This direction is induced by the following use case: the paper heater modelled above is used with different paper types $\{\mathsf{pt}_1, \mathsf{pt}_2, \mathsf{pt}_3\}$ (for example: 80 g/m^2 A4, 200 g/m^2 A4, 80 g/m^2 Letter). We have no direct knowledge about which type is in the heater, and would therefore like to model it as a probabilistic variable PT. Each paper type leads to a different value for the system's r' parameter (the heat resistance between plate and paper). The question we ask ourselves is: *How can we join the paper type variable to the LDS model, so we can infer probabilistically which paper type is in the heater, by observing the T' values of the system?*

[1]This is known as a *lumped element model*; in contrast, the heat distribution could also be modelled as a 3-dimensional temperature field over the plate.

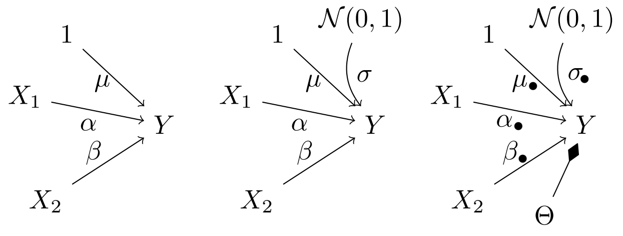


Figure 2: The three basic types of Bayesian network nodes we will use for LDS models. **Left:** Linear deterministic node $\mathsf{P}(y|x_1, x_2) = 1$ if $y = \mu + \alpha x_1 + \beta x_2$. **Middle:** Linear Gaussian node $\mathsf{P}(Y|x_1, x_2) = \mathcal{N}(\mu + \alpha x_1 + \beta x_2, \sigma^2)$. **Right:** Conditional linear Gaussian node $\mathsf{P}(Y|x_1, x_2, \theta) = \mathcal{N}(\mu_\theta + \alpha_\theta x_1 + \beta_\theta x_2, \sigma_\theta^2)$ (for discrete Θ). The notation is ours and is introduced in section 3.3.

3 Augmenting LDS models with uncertainty

3.1 Bayesian network representation

A *Bayesian network* $\mathcal{B} = (G, \Phi)$ is an acyclic directed graph G, consisting of *nodes* and *arcs*, that is faithful to a joint probability distribution factored as Φ, which contains for each node Y (with parents $X_1, X_2, \ldots$) a family of local conditional probability density or mass functions $\mathsf{P}(Y|X_1, X_2, \ldots)$. A *dynamic Bayesian network* [1] is a special case of this, where the nodes are partitioned in time slices all consisting of the same structure and distributions. Furthermore, arcs are only allowed between nodes in the same or adjacent time slice.

For modelling linear dynamic systems as (dynamic) Bayesian networks, only the following types of nodes are needed:

Deterministic nodes: a node Y with parents $X_1, X_2, \ldots$ is called *deterministic* if its conditional probability distribution is

$$\mathsf{P}(y|x_1, x_2, \ldots) = \begin{cases} 1 & \text{if } y = f(x_1, x_2, \ldots) \\ 0 & \text{if } y \neq f(x_1, x_2, \ldots) \end{cases}$$

for a certain function f; in this article, these functions are mostly *linear*, i.e.

$$y = \mu + \alpha x_1 + \beta x_2 + \ldots$$

We use a special notation for these *linear deterministic* nodes shown in Fig. 2 (left).

Linear Gaussians: a node Y with parents $X_1, X_2, \ldots$ is known in Bayesian network literature as a *linear Gaussian* if

$$\mathsf{P}(Y|x_1, x_2, \ldots) = \mathcal{N}(\mu + \alpha x_1 + \beta x_2 + \ldots, \sigma^2)$$

Networks that consist only of linear Gaussians (with $\sigma > 0$) have theoretical significance: their joint distribution is multivariate Gaussian, and exact inference is easy and efficient (e.g. see [6]). A linear Gaussian without parents $\mathcal{N}(\mu, \sigma^2)$ is simply called Gaussian; the Gaussian $\mathcal{N}(0, 1)$ is called a *standard* Gaussian. A linear Gaussian can be written as a linear deterministic node with two extra parents; see Fig. 2 (middle).

Conditional linear Gaussians: a node Y with parents $X_1, X_2, \ldots$ and discrete parent Θ is *conditional linear Gaussian* if

$$\mathsf{P}(Y|x_1, x_2, \ldots, \theta) = \mathcal{N}(\mu_\theta + \alpha_\theta x_1 + \beta_\theta x_2 + \ldots, \sigma_\theta^2)$$

i.e. it is linear Gaussian for each value θ. If $X_1, X_2, \ldots$ are Gaussian, the marginal distribution over Y is a mixture of Gaussians:

$$\mathsf{P}(Y) = \sum_{\theta \in \Theta} \mathsf{P}(\theta) \mathcal{N}(\hat{\mu}_\theta, \hat{\sigma}_\theta^2)$$
$$\hat{\mu}_\theta = \mu_\theta + \alpha_\theta \mu_{X_1} + \beta_\theta \mu_{X_2} + \ldots$$
$$\hat{\sigma}_\theta^2 = \sigma_\theta^2 + \alpha_\theta^2 \sigma_{X_1}^2 + \beta_\theta^2 \sigma_{X_2}^2 + \ldots$$

Again, this also holds for complete networks: if all nodes are (conditional) linear Gaussian, the joint distribution is a mixture of multivariate Gaussians. However, this number of components in this mixture is exponential in the number of Θ variables, which can make inference hard. A conditional linear Gaussian can also be written as a deterministic node with extra parents. For this we use a special notation shown in Fig. 2 (right), to which we will return later.

As these three node types can all be written as deterministic nodes, we will henceforth use the convention that *all non-root nodes in our networks are deterministic.*

3.2 LDS models as Bayesian networks

The paper heater model in Fig. 1 translates to the following discrete-time state-space equations in the form of Eq. (1):

$$\begin{bmatrix} T_{t+1} \\ T'_{t+1} \end{bmatrix} = \mathbf{A}_d \begin{bmatrix} T_t \\ T'_t \end{bmatrix} + \mathbf{B}_d \begin{bmatrix} P_t^{\text{in}} \\ 20°\text{C} \end{bmatrix}$$
$$\mathbf{A}_d = \begin{bmatrix} 1 - 1/rc & 1/rc \\ 1/rc' & 1 - 1/rc' - 1/r'c' \end{bmatrix} \tag{2}$$
$$\mathbf{B}_d = \begin{bmatrix} 1/c & 0 \\ 0 & 1/r'c' \end{bmatrix}$$

The state of the system consists of the temperatures T_t and T'_t of the two heat masses. In fact, translating the electrical diagram by tools such as 20-sim first leads to a more elaborate form in which auxiliary power variables are present. It is instructive to represent this form as a Bayesian network consisting only of linear deterministic nodes; this is shown at the right side of Fig. 1. As the network is completely deterministic, it might also be read as a system of equations over ordinary variables:

- Each *node* represents the left-hand side of an equation, consisting of one variable.
- The incoming *arcs* represent the right-hand side: a linear combination of the parent variables, with coefficients as specified on the arcs. Note: we follow the convention that empty arcs carry a coefficient of 1.

For example, the figure shows that

$$P_t^{\text{trans}} = \frac{1}{r} T_t + \frac{-1}{r} T'_t = \frac{T_t - T'_t}{r}$$
$$T_{t+1} = \frac{1}{c} P_t^{\text{in}} + T_t + \frac{-1}{c} P_t^{\text{trans}} = T_t + \frac{P_t^{\text{in}} - P_t^{\text{trans}}}{c}$$

The state-space equations (2) are obtained from this system by substituting the P^{trans} and P^{out} variables for their right-hand sides. Interpreted as a Bayesian network, this corresponds to marginalization.

We will now start to add uncertainty to the LDS. First, as is often done (e.g. in the Kalman filter), we augment the state variables with additive zero-mean Gaussian noise:

$$\begin{bmatrix} T_{t+1} \\ T'_{t+1} \end{bmatrix} = \mathbf{A}_d \begin{bmatrix} T_t \\ T'_t \end{bmatrix} + \mathbf{B}_d \begin{bmatrix} P_t^{\text{in}} \\ 20°\text{C} \end{bmatrix} + \begin{bmatrix} W_t \\ W'_t \end{bmatrix}$$
$$W_t \sim \mathcal{N}(0, \sigma^2) \tag{3}$$
$$W'_t \sim \mathcal{N}(0, \sigma'^2)$$

The noise is represented by two independent variables W_t and W'_t. A graphical representation of this system is shown in Fig. 3; we have only replaced W_t and W'_t by two anonymous standard Gaussian variables $\mathcal{N}(0, 1)$ whose value is multiplied by σ and σ'. As a result, the T_t variables now have the linear Gaussian form from Fig. 2 (middle).

In fact, this makes the whole system Gaussian: although the P_t^{trans} and P^{out} nodes are not linear Gaussian, we have already seen that they can be marginalized out, making the T_{t+1}, T'_{t+1} nodes directly depend on T_t, T'_t. As for the P_t^{in} node: as it is always used with concrete evidence p_t^{in}, it never represents a probability distribution, and can be reformulated to take

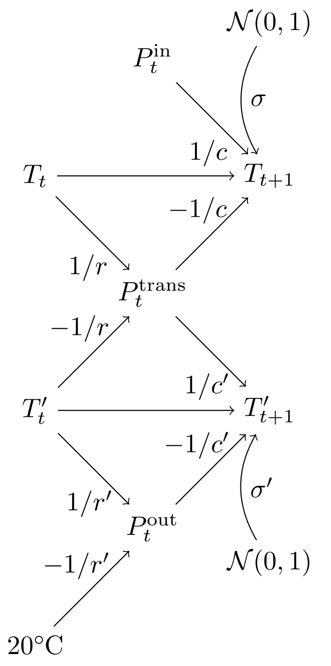


Figure 3: LDS of Eq. (3), containing additive zero-mean Gaussian noise on the state variables.

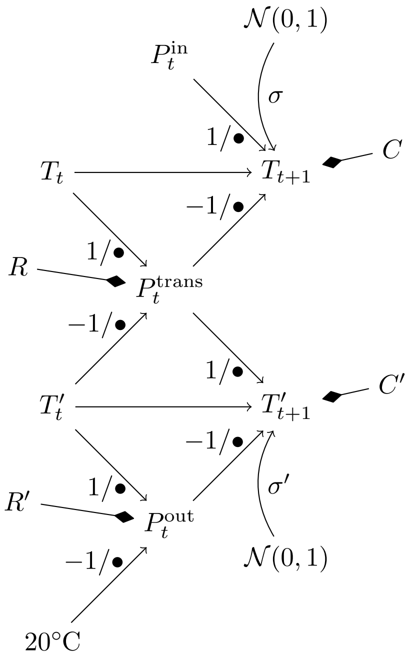
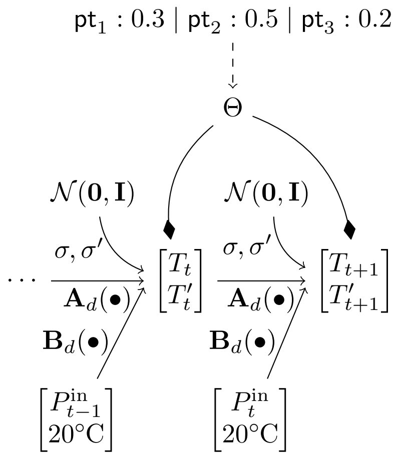


Figure 4: LDS of Eq. (3) augmented with uncertain parameters. These are modelled by *conditional linear deterministic* nodes.



$\mathsf{pt}_1 : 0.3 \mid \mathsf{pt}_2 : 0.5 \mid \mathsf{pt}_3 : 0.2$
Θ
$\mathcal{N}(\mathbf{0},\mathbf{I})$
$\mathcal{N}(\mathbf{0},\mathbf{I})$
σ, σ'
σ, σ'
$\cdots$
$\mathbf{A}_d(\bullet)$
$\mathbf{A}_d(\bullet)$
$\begin{bmatrix} T_t \\ T'_t \end{bmatrix}$
$\begin{bmatrix} T_{t+1} \\ T'_{t+1} \end{bmatrix}$
$\mathbf{B}_d(\bullet)$
$\mathbf{B}_d(\bullet)$
$\begin{bmatrix} P_{t-1}^{\text{in}} \\ 20°\text{C} \end{bmatrix}$
$\begin{bmatrix} P_t^{\text{in}} \\ 20°\text{C} \end{bmatrix}$

Figure 5: The model from Fig. 4, summarized using vector variables (where Θ represents R, C, R', C') and augmented with a paper type variable with a discrete distribution. The relation between time slices $t-1$ and t is also shown. There are several options for the relation between paper type and Θ (the dashed arc here is not a linear influence).

the place of μ in the T_{t+1} distribution. Thus, Fig. 3 is a dynamic Bayesian network with a Gaussian joint distribution. This means that we can use a variant of the standard Kalman filter to do exact inference on this network; we discuss this in detail in Sect. 4.

3.3 LDS models with uncertain parameters

Above, we have shown how to represent an LDS with additive zero-mean Gaussian noise as a Bayesian network; while it may be instructive, thus far it is only a convenient reformulation of known theory. However, to model the relation with the paper type variable, we need uncertainty in the *parameters*, i.e. the coefficients of the linear relation. Our solution is to transform these coefficients into probabilistic variables. We accomplish this by introducing a new node type:

Conditional linear deterministic node: a linear deterministic node extended with a special parent, which is distinguished from the rest because its arc ends in a diamond (and carries no coefficient); we call this parent the *conditioning parent.* The coefficients on the other arcs *can depend on the value of the conditioning parent.* This dependence is shown by putting a bullet in the place where this value is to be filled in.

We have already given an example of such a node: the conditional linear Gaussian node in Fig. 2. Just like the linear Gaussian node is an instance of a linear deterministic node, viz. having specific parents 1 and $\mathcal{N}(0,1)$, a conditional linear Gaussian node is a specific instance of conditional linear deterministic node.

By using conditional linear deterministic nodes, we can extend our already noisy paper heater model with uncertain parameters: we replace parameter r by variable R—which becomes the conditioning parent of the node that depended on r—and do the same for the other parameters. The result is shown in Fig. 4.

We can now proceed to connect a discrete paper type variable to the model, with an example distribution assigning to $\mathsf{pt}_1, \mathsf{pt}_2, \mathsf{pt}_3$ the probabilities 0.3, 0.5 and 0.2. Like we mentioned, the paper type determines the R' parameter, but for generalization's sake we will assume that it influences all the parameters. The resulting model, in Fig. 5, also shows that the notation for (conditional) linear deterministic nodes extends very naturally to vector-valued variables: coefficients become matrices. These are the matrices from Eq. (2), written as functions $\mathbf{A}_d(\theta)$ and $\mathbf{B}_d(\theta)$ of $\theta = (r, c, r', c')$. Thus, we have made a *graphical summary* of the model which is linked very clearly to the state-space equations. Although this hides some of the model's internal structure, it is useful for keeping an overview.

Regarding the probability distribution of the Θ variable, we give two examples:

Discrete Θ: The paper types $\mathsf{pt}_1, \mathsf{pt}_2, \mathsf{pt}_3$ deterministically set a value $\theta_1, \theta_2, \theta_3$ (resp.) for Θ. In fact, this turns T_t and T_{t+1} into conditional linear Gaussians (conditioned by the paper type), so the joint distribution is a mixture of 3 Gaussians.

Continuous Θ: The paper types $\mathsf{pt}_1, \mathsf{pt}_2, \mathsf{pt}_3$ determine the parameters $(\mu_{\theta 1}, \sigma_{\theta 1})$, $(\mu_{\theta 2}, \sigma_{\theta 2})$, $(\mu_{\theta 2}, \sigma_{\theta 2})$ for a Gaussian-distributed Θ. This model is no longer linear.

These options have an influence on inference and learning in the model, which we discuss in the next sections.

4 Inference

In this section, we shortly discuss inference in the uncertain parameter model of Fig. 4, for both the discrete and continuous Θ given above. Assume we observe the system's T'_t variable responding to the P^{in}_t input for a while, resulting in data $\mathcal{D} = \{p^{\text{in}}_0, t'_0, \ldots, p^{\text{in}}_{m-1}, t'_{m-1}, t'_m\}$, and the goal is to find out the paper type.

This can be done by a forward pass over the model as known from dynamic Bayesian network literature. We start with the prior distribution

$$\mathsf{P}(t_0, \theta, t'_0) = \mathsf{P}(t_0)\mathsf{P}(\theta)\mathsf{P}(t'_0)$$

and perform a recursive forward pass from $t = 0$ to $t + 1 = m$:

$$\mathsf{P}(t_{t+1}, \theta, p^{\text{in}}_{0..t}, t'_{0..t+1}) = \int \mathsf{P}(t_t, \theta, p^{\text{in}}_{0..t-1}, t'_{0..t})\mathsf{P}(t_{t+1}|t_t, t'_t, p^{\text{in}}_t, \theta)\mathsf{P}(t'_{t+1}|t_t, t'_t, \theta)\, dt_t$$

Finally, we marginalize out t_m:

$$\mathsf{P}(\theta, \mathcal{D}) = \int \mathsf{P}(t_m, \theta, \mathcal{D})\, dt_m$$

The details of the inference algorithm depend on the model used. For the discrete Θ, all the distributions above are linear Gaussian, so we can multiply and integrate exactly. To be precise, $\mathsf{P}(t_{t+1}|t_t, t'_t, p^{\text{in}}_t, \theta)$ and $\mathsf{P}(t'_{t+1}|t_t, t'_t, \theta)$ are linear Gaussian for each of the 3 individual θ_i values; the algorithm thus independently works on 3 Gaussians.

For the continuous Θ, the conditional distributions of T_{t+1} and T'_{t+1} are not linear Gaussian; we can do *approximate* inference by linearizing these distributions at each timeslice around the means of T_t, Θ (given the data up to t), in analogy to the Extended Kalman Filter (see e.g. [6, 10]).

A second type of inference that we do with the model is *smoothing*. In particular, we want to calculate $\mathsf{P}(\theta, t_t, t_{t+1}|\mathcal{D})$ for each timeslice, in order to do EM learning (see the next section). We have used a *Rauch-Tung-Striebel*-type smoother [9]. This uses a forward pass like discussed above, with the adjustment that it stores the distributions over two time slices. The last of these, i.e. $\mathsf{P}(t_{m-1}, t_m, \theta, \mathcal{D})$, is used as the input for a recursive backward pass defined as follows:

$$\mathsf{P}(t_{t-1}, t_t, \theta, \mathcal{D}) = \int \mathsf{P}(t_t, t_{t+1}, \theta, \mathcal{D})\mathsf{P}(t_{t-1}|t_t, \theta, \mathcal{D})\, dt_{t+1}$$

where the first factor in the integral is the recursive one, and the second is calculated from the distribution over t_{t-1}, t_t stored in the forward pass:

$$\mathsf{P}(t_{t-1}|t_t, \theta, \mathcal{D}) = \mathsf{P}(t_{t-1}|t_t, p^{\text{in}}_{0..t-1}, t'_{0..t}) = \frac{\mathsf{P}(t_{t-1}, t_t, p^{\text{in}}_{0..t-1}, t'_{0..t})}{\int \mathsf{P}(t_{t-1}, t_t, p^{\text{in}}_{0..t-1}, t'_{0..t})\, dt_{t-1}}$$

The advantage of such a smoother over an independent backward pass is that it does not linearize the distribution over T_{t-1}, T in two different ways (the backward pass uses the linearization of the forward pass).

5 Learning

For *learning* the model, we discuss the situation where we *know the paper type* (assume it is pt_1) and observe the system like before, i.e. $\mathcal{D} = \{p^{\text{in}}_0, t'_0, \ldots, p^{\text{in}}_{m-1}, t'_{m-1}, t'_m\}$. The goal is to learn the parameter set ρ that maximizes the likelihood $\mathsf{P}(\mathcal{D}|\mathsf{pt}_1; \rho)$. The situation is a little different depending on the model for Θ.

5.1 EM for continuous Θ

For the continuous Θ, the restriction to pt_1 means that we are learning the parameters for one multivariate Gaussian variable Θ (actually consisting of four independent variables R, C, R', C') and the σ, σ' process noise parameters. Thus, $\rho = (\mu_{\theta 1}, \sigma_{\theta 1}, \sigma, \sigma')$. This can be done by a standard EM algorithm [2] for Bayesian networks: given a set of initial parameters ρ_i, the approximate smoother infers the distributions $\mathsf{P}(t_t, t_{t+1}, \theta|\mathcal{D}, \mathsf{pt}_1; \rho_i)$. From these, the expected sufficient statistics are gathered for maximizing

$$\mathsf{P}(\mathcal{D}, T_{0..m}, \Theta|\mathsf{pt}_1; \rho_{i+1})$$

expected under the old parameters ρ_i; this is repeated until convergence.

5.2 EM for discrete Θ

For the discrete parameter space model, we are looking for the parameter set $(\theta, \sigma, \sigma')$ that maximizes the likelihood

$$\mathsf{P}(\mathcal{D}|\mathsf{pt}_1, \Theta = \theta; \sigma, \sigma')$$

Note that the role of Θ is different here; we are not learning the optimal parameters for a distribution over Θ, but the optimal single value. This requires some adjustments to the smoother: it should store distributions over (T_t, T_{t+1}) instead of over (T_t, T_{t+1}, Θ), and should not use a prior distribution over Θ either. Because all the probability distributions are linear Gaussian again, smoothing is exact now.

However, maximizing the expected likelihood is not so trivial now: we are looking for the optimal linear Gaussian distribution $\mathsf{P}(t_{t+1}, t'_{t+1}|t_t, t'_t, p^{\text{in}}_t, \theta)$ *constrained to a certain form* prescribed by $\mathbf{A}$ and $\mathbf{B}$. Specifically, the log likelihood for an individual time slice is:

$$\log \mathsf{P}(t_{t+1}, t'_{t+1}|t_t, t'_t, p^{\text{in}}_t, \theta) = -\frac{1}{2}\delta^T\Sigma^{-1}\delta - \frac{1}{2}\log|2\pi\Sigma|$$

where $\Sigma = \begin{bmatrix} \sigma^2 & 0 \\ 0 & \sigma'^2 \end{bmatrix}$ and δ is an abbreviation for

$$\delta = \begin{bmatrix} t_{t+1} \\ t'_{t+1} \end{bmatrix} - \mathbf{A}_d(\theta)\begin{bmatrix} t_t \\ t'_t \end{bmatrix} - \mathbf{B}_d(\theta)\begin{bmatrix} p^{\text{in}}_t \\ 20°\text{C} \end{bmatrix}$$

Separating variables and parameters, we can write:

$$\begin{aligned} \delta &= \mathbf{D}(\theta)\mathbf{x}_t \\ \mathbf{D}(\theta) &= \begin{bmatrix} \mathbf{I} & -\mathbf{A}_d(\theta) & -\mathbf{B}_d(\theta) \end{bmatrix} \\ \mathbf{x}_t &= \begin{bmatrix} t_{t+1} & t'_{t+1} & t_t & t'_t & p^{\text{in}}_t & 20°\text{C} \end{bmatrix}^T \end{aligned}$$

The log likelihood for all the time slices (ignoring the term $-\frac{1}{2}\log|2\pi\Sigma|$ for now) is then:

$$\log \mathsf{P}(\mathcal{D}, t_{0..m}|\theta) = -\frac{1}{2}\sum_{t=0..m}(\mathbf{D}(\theta)\mathbf{x}_t)^T\Sigma^{-1}\mathbf{D}(\theta)\mathbf{x}_t$$

$$= -\frac{1}{2}\sum_{t=0..m}\frac{(\mathbf{D}_1(\theta)\mathbf{x}_t)^T\mathbf{D}_1(\theta)\mathbf{x}_t}{\sigma^2} + \frac{(\mathbf{D}_2(\theta)\mathbf{x}_t)^T\mathbf{D}_2(\theta)\mathbf{x}_t}{\sigma'^2}$$

where $\mathbf{D}_i$ denotes the ith row of $\mathbf{D}$. The goal is to maximize the expected value of this expression. At first sight, it seems that the two terms are dependent through θ, but on closer inspection

$$\mathbf{D}(\theta) = \begin{bmatrix} 1 & 0 & -1+1/rc & -1/rc & -1/c & 0 \\ 0 & 1 & 1/rc' & -1+1/rc'+1/r'c' & 0 & -1/r'c' \end{bmatrix}$$

we see that the values in the first row do not constrain those in the second, or vice versa. We can therefore minimize the expected value of the two terms independently. We can also see that there *are* linear constraints for the values *within* a row, e.g. $\mathbf{D}_{1,3}(\theta) + \mathbf{D}_{1,4}(\theta) = -1$. We record these constraints in a matrix $\mathbf{C}$ and vector $\mathbf{c}$ such that $\mathbf{C}\mathbf{D}_1^T(\theta) = \mathbf{c}$. Substituting $\mathbf{d} = \mathbf{D}_1^T(\theta)$, for the first term we are looking for the $\mathbf{d}$ that minimizes

$$\sum_{t=0..m} E(\mathbf{x}_t^T\mathbf{d}\mathbf{d}^T\mathbf{x}_t) = \mathbf{d}^T\left[\sum_{t=0..m} E(\mathbf{x}_t\mathbf{x}_t^T)\right]\mathbf{d}$$

under the constraint $\mathbf{C}\mathbf{d} = \mathbf{c}$. This is a linearly constrained quadratic optimization problem that can be solved by the method of Lagrange multipliers. The second term can be minimized in the same way.

In conclusion, we have derived the M-phase for the discrete Θ model; in the E-phase, we therefore have to collect the expected sufficient statistics $E(\mathbf{x}_t\mathbf{x}_t^T)$.

5.3 Comparison

It is interesting to compare learning for continuous and discrete Θ. In order to do this, we have simulated the system in Fig. 1 for 150 time slices, with a sine wave as P^{in} input and random Gaussian disturbance. We provide the EM algorithms discussed above with the P^{in} and the generated T'_t data. Typical results of a 60-iterations run are shown in Fig. 6.

The most interesting fact to observe is that the two approaches converge to different values (but the same log likelihood). This probably means that the system is not *identifiable*: several choices for Θ lead to the same likelihood of the observed behaviour T'_t. To test this hypothesis, we also generated synthetic T_t, T'_t data (without disturbance) for systems with the learned parameters. The results are plotted in Fig. 7. These results indeed show that both methods arrive at the same approximation (green, red) of the original (blue) T'_t data; however, the different values for the parameters lead to a different behavior of T_t.

A second observation from Fig. 6 is that learning continuous Θ converges faster than learning discrete Θ. The explanation for this is as follows: the EM algorithm for the continuous parameter space uses inference to compute a posterior distribution over the variable Θ. In this algorithm, the posterior distribution is updated for each time slice. However, we can also regard the algorithm as doing full Bayesian learning where Θ is viewed as a parameter; the algorithm is then performing *incremental EM* [8], which is known to converge faster.

6 Conclusion

The central scientific hypothesis which initiated the research described in this paper was that knowledge engineering methods for industrial applications of probabilistic graphical models should be based as much as

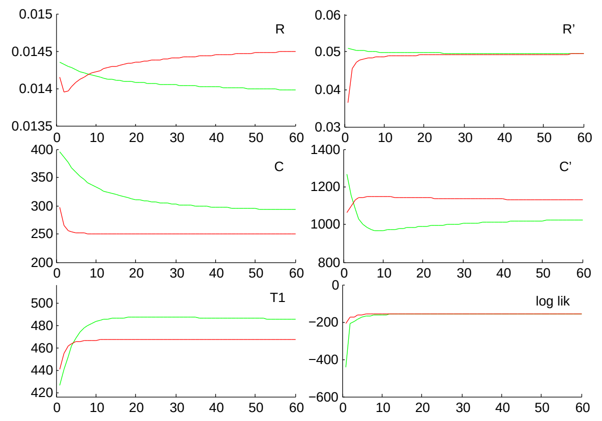


Figure 6: EM learning of the Θ parameters: comparison of discrete parameter space (parameters are single values; shown in green) and continuous parameter space (parameters are Gaussians; μ values shown in red). The horizontal axis represents 60 EM iterations. Also shown are the learned distribution over T_1 (Gaussian, μ value) and the log likelihood.

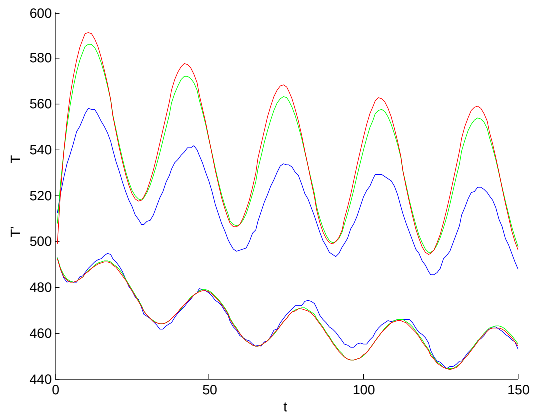


Figure 7: Blue: synthetic data used for learning (T_t and T'_t are shown, but only the latter is given to the learning algorithms). As input P_t^{in}, we used a sine wave. We disturbed the system with additive zero-mean Gaussian noise. **Green, red**: response of an undisturbed deterministic system using the learned parameters (with discrete and continuous parameter space, resp.) to the same P_t^{in} sine wave.

possible on existing methods from engineering. We have developed a systematic framework, where we start with linear system theory and associated diagrams and notations as the first step in the knowledge engineering process. The advantage of this approach is that engineers have already dealt with the unavoidable complexity issues in representing realistic models. Subsequently, it was shown how linear dynamic system models can be augmented with probabilistic variables for uncertain parameters, transforming them into dynamic Bayesian networks with *conditionally linear* nodes. We introduced a concise notation that combines LDS and Bayesian network concepts in a natural way and demonstrated methods for inference and learning from data in these models. The practical usefulness of the framework was illustrated by a case study from the domain of large production printers.

Acknowledgements

This work has been carried out as part of the OCTOPUS project under the responsibility of the Embedded Systems Institute. This project is partially supported by the Netherlands Ministry of Economic Affairs under the Embedded Systems Institute program.

References

[1] Thomas Dean and Keiji Kanazawa. A model for reasoning about persistence and causation. *Computational Intelligence*, 5(3):142–150, 1989.

[2] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[3] A.J. Hommersom and P.J.F. Lucas. Using Bayesian networks in an industrial setting: Making printing systems adaptive. In *19th European Conference on Artificial Intelligence*, pages 401–406, 2010.

[4] R.E. Kalman et al. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.

[5] D.C. Karnopp, D.L. Margolis, and R.C. Rosenberg. *System dynamics: modeling and simulation of mechatronic systems*. John Wiley & Sons, 2006.

[6] D. Koller and N. Friedman. *Probabilistic graphical models: Principles and techniques*. The MIT Press, 2009.

[7] K.B. Korb and A.E. Nicholson. *Bayesian Artificial Intelligence*. Chapman & Hall/CRC, 2004.

[8] Radford Neal and Geoffrey E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In M.I. Jordan, editor, *Learning in Graphical Models*, pages 355–368. Kluwer Academic Publishers, 1998.

[9] H.E. Rauch, F. Tung, and CT Striebel. Maximum likelihood estimates of linear dynamic systems. *AIAA journal*, 3(8):1445–1450, 1965.

[10] H.W. Sorenson. *Kalman filtering: theory and application*. IEEE, 1985.

[11] `http://www.20sim.com/`.

Exploratory network analysis of large social science questionnaires

Robert J. B. Goudie
Department of Statistics
University of Warwick
Coventry, UK

Sach Mukherjee
Department of Statistics &
Centre for Complexity Science
University of Warwick
Coventry, UK

Frances Griffiths
Health Sciences Research Institute
University of Warwick
Coventry, UK

Abstract

There are now many large surveys of individuals that include questions covering a wide range of behaviours. We investigate longitudinal data from the Add Health survey of adolescents in the US. We describe how structural inference for (dynamic) Bayesian networks can be used to explore relationships between variables in such data and present this information in an interpretable format for subject-matter practitioners. Surveys such as this often have a large sample-size, which, whilst increasing the precision of inference, may mean that the posterior distribution over Bayesian networks (or graphs) is concentrated on disparate graphs. In such situations, the standard MC^3 sampler converges very slowly to the posterior distribution. Instead, we use a Gibbs sampler (1), which moves more freely through graph space. We present and discuss the resulting Bayesian network, focusing on depression, and provide estimates of how different variables affect the probability of depression via the overall probabilistic structure given by the Bayesian network.

1 INTRODUCTION

Hypotheses of multifactorial causes of symptoms and outcomes play an important role in the social sciences and in public health. Regression-based approaches are widely-used in these fields to explore such hypotheses. A great deal of insight can be gained through such approaches, but it is sometimes overly constraining to fix a particular quantity as the dependent variable, especially if the goal is to explore the possibility of unexpected relationships between the data. Instead, we can consider a number of variables on an equal footing, and study the possibility of unexpected relationships in the data.

Graphical models provide a statistical framework within which the relationship between variables can be studied. These models enable complex multivariate distributions to be decomposed into simpler local distributions. This can reveal a great deal about the relationships between the variables, as well as provide a statistical and computationally tractable description of their (often large) joint distribution. The decomposition is formed by the conditional independence structure, which can be represented by a graph. The use of graphs helps to make the interpretation of the model simpler. In this paper, we focus on the structure of the model, as given by the graph. We aim to make inference about this using statistical model selection. The structure of the model suggests how the different components of the system interact, which may be helpful in understanding the system as a whole. These methods have been widely adopted in molecular biology (2, 3), and have been used in some areas of medical sciences (4).

Consideration of unexpected relationships between factors requires datasets that incorporate a wide range of topics. Such data is now widely available for representative samples of populations in many countries, and for many sub-groups of interest. Many of these datasets are derived from surveys that are general in scope, and are not collected to study any one particular question. For example, in the US, the health of the whole population is representatively sampled annually for the Behavioral Risk Factor Surveillance System (BRFSS) survey, and the Add Health study, which we use here, followed a cohort of young people from 1994 until 2008. Data from both of these have been used in scores of studies, but these commonly focus on one specific aspect, often using the data to evaluate existing hypotheses. Given the wide scope inherent in the design of these studies and the large samples available in many cases, it is possible to broaden the scope of the analysis by considering richer structures. In this paper, we discuss the potential that such a more explorative approach yields. We do not seek to make conclusive causal claims, but instead suggest that a broader approach may uncover important aspects that have been neglected.

Our focus will be on depression among adolescents in the US, drawing on data from the National Longitudinal Study of Adolescent Health (Add Health). It is estimated that around 1–6% of adolescents each year are affected by depression (5, 6). The effects of depression in this age-group are wide-ranging (7), and include the stigma associated with poor mental health more generally (8). There is considerable evidence that there are a wide range of causal factors for depression amongst adolescents, spanning biological, psychological and social domains. Understanding these causal factors and separating them from the consequences of depression has been recognised as an important aim (9). Some of the relevant causal factors may interact and the approach taken here accounts for this.

The remainder of this paper is organised as follows. We first introduce the AddHealth dataset and describe the Bayesian network framework. Inference for Bayesian networks is performed using Markov Chain Monte Carlo (MCMC), but the large sample size of the dataset we consider makes achieving convergence difficult because the posterior distribution may be concentrated on disparate graphs, and so we describe an alternative sampler that has superior properties in this situation. Whilst the PC-algorithm (10, 11) has properties that often make it attractive in such contexts, we found that the results in this situation were not robust (see Discussion). We then present and discuss the results for the Add Health dataset.

2 MATERIALS AND METHODS

2.1 Add Health

The data that we use are drawn from the National Longitudinal Study of Adolescent Health (Add Health) that explores health-related behavior of adolescents (12) in the US. The questionnaire contains over 2000 questions that cover many aspects of adolescent behaviours and attitudes. We consider the representative sample of adolescents from Waves I and II of the in-home section, and the parental questionnaire from Wave I of the study. The analysis we perform is not feasible when the data is not complete (see Discussion), and so individuals with missing data were removed from the study. Removing incomplete samples leaves 5975 individuals in the study.

Our measure of depression is a self-assessed scale based upon the Centre for Epidemiologic Studies Depression Scale (CES-D) (13). Two questions from the 20-item scale are omitted from AddHealth, and two are modified, and so we scale the score given by the available questions (14). A Receiver Operating Characteristic (ROC) analysis showed that thresholds of 24 for females and 22 for males provided the best agreement with clinical assessments of depression (15). We use this threshold to create a binary indicator of depression status.

Many of the remainder of the variables that we consider (Table 1) are drawn from the risk factors described in the depression literature, and the mental health literature more generally. A recent review (8) described a wide range of factors that are associated with poor mental health in young people, including gender, poverty, violence and the absence of social networks in the local neighbourhood. The quality of relationships with parents is also thought to be important, especially with the mother (16), as are parental alcohol problems (17) and parental discord (16). The individual's use of alcohol, drugs, smoking and HIV/AIDS are all also associated with depression (18, 19). Physical exercise has been proposed in some studies as a useful intervention for the management of depression, but many of these studies have been deemed to be poor quality (20).

2.2 Bayesian Networks

Our study uses Bayesian networks to explore the relationships between variables in the Add Health study. Bayesian networks are a particular type of graphical model that enable classes of probability distributions to be specified using a directed acyclic graph (DAG). A Bayesian network G is represented using a DAG with vertices $V = (V_1, \ldots, V_p)$, and directed edges $E \subset V \times V$. The vertices correspond to the components of a random vector $\mathbf{X} = [X_1, \ldots, X_p]^T$, subsets of which will be denoted by X_A for sets $A \subseteq \{1, \ldots, p\}$. For $1 \leq i, j \leq p$, we define the parents G_j of each node V_j to be the subset of vertices V such that $V_i \in G_j \Leftrightarrow (V_i, V_j) \in E$. Specifying the parents of the vertices determines the edges E of the graph G. We denote by $\mathcal{G}$ the space of all possible directed acyclic graphs with p vertices. We will use X_{G_i} to refer to the random variables that are parents of X_i in the graph G.

The graph specifies that the joint distribution for $\mathbf{X}$, with parameters $\theta = (\theta_1, \ldots, \theta_p)$, can be written as a product of conditional distributions $p(X_i \mid X_{G_i}, \theta_i)$, given the variables X_{G_i} corresponding to the parents of X_i in the graph.

$$p(\mathbf{X} \mid G, \theta) = \prod_{i=1}^{p} p(X_i \mid X_{G_i}, \theta_i)$$

We will need to be able to evaluate the marginal likelihood $p(\mathbf{X} \mid G)$ easily, and so we consider only a conjugate analysis in which the conditional distributions $p(X_i \mid X_{G_i}, \theta_i)$ are multinomial, with Dirichlet priors $p(\theta_i)$ for each θ_i. In this case, the marginal likelihood can be evaluated analytically. Suppose each X_i takes one of r_i values, and define q_i as the number of levels of the sample space of X_{G_i}, each element of which we call a configuration. For each configuration j of X_{G_i}, let N_{ijk} be the number of observations in which X_i takes value k. We assume the Dirichlet priors for each θ_i, each with hyperparameters N'_{ijk}, are independent. We define $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$ and $N'_{ij} = \sum_{k=1}^{r_i} N'_{ijk}$, and the local score $p(X_i \mid X_{G_i})$ to be

$$p(X_i \mid X_{G_i}) = \prod_{j=1}^{q_i} \frac{\Gamma(N'_{ij})}{\Gamma(N_{ij} + N'_{ij})} \prod_{k=1}^{r_i} \frac{\Gamma(N_{ijk} + N'_{ijk})}{\Gamma(N'_{ijk})}.$$

The marginal likelihood can be shown to equal the product $p(\mathbf{X} \mid G) = \prod_{i=1}^{p} p(X_i \mid X_{G_i})$ of these local scores (21).

2.3 Structural inference for Bayesian Networks

We aim to make inference about the DAG G, given data $\mathbf{X}$ and so our interest focuses on the posterior distribution $\Pr(G \mid \mathbf{X})$ on Bayesian networks. Under the assumptions we have made, this can be written in terms of the marginal likelihood $p(\mathbf{X} \mid G)$, and a prior $\pi(G)$ for the Bayesian network structure.

$$\Pr(G \mid \mathbf{X}) \propto \pi(G) \prod_{i=1}^{p} p(X_i \mid X_{G_i})$$

The priors $\pi(G)$ can be chosen to encode domain information (3). For the analyses in this paper, we choose an improper prior $\pi(G) \propto 1$ that is flat across the space of graphs.

The posterior distribution $\Pr(G \mid \mathbf{X})$ is difficult to evaluate, because cardinality of $\mathcal{G}$ grows super-exponentially in p. This motivates the use of approximations to $\Pr(G \mid \mathbf{X})$, which are usually based on Markov chain Monte Carlo (MCMC).

2.4 Approximate inference for Bayesian Networks

The standard form of MCMC that is used for structural inference for Bayesian networks is MC^3 (22). This is a Metropolis-Hastings sampler that explores $\mathcal{G}$ by proposing to add or remove a single edge from the current graph G. This sampler works surprisingly well in many situations, but if the posterior distribution is not unimodal, the local moves may fail to explore the space fully because the sampler may become ‘trapped’ in one mode. This issue becomes more severe as the sample size increases because the posterior distribution becomes more concentrated. A natural approach in such situations is to use the PC-algorithm (10, 11), which has been shown to be asymptotically consistent (23), but we found in this case that the results were not robust (see Discussion).

Our analyses in this paper were performed using a Gibbs sampler (1), which we found to converge rapidly to its equilibrium state. A naïve Gibbs sampler for structural inference that proposes single-edge additions and removals can easily be constructed, but this sampler offers no advantages over the analogous MC^3. This naïve scheme, however, can be improved by ‘blocking’ together a number of components, and sampling from their joint conditional distribution. In theory, any group of components can be taken as a block, but sampling from their joint conditional distribution needs to be possible and, ideally, computationally quick.

For Bayesian networks, the most natural blocks are those consisting of parent sets $G_1, \ldots, G_p$. This is natural because the marginal likelihood $p(\mathbf{X} \mid G)$ for a graph G factorises across vertices into conditionals $p(X_j \mid X_{G_j})$ and these conditionals depend on the parent set of the vertex. Therefore, since any graph $G \in \mathcal{G}$ can be specified by a vector $G = (G_1, \ldots, G_p)$ of parent sets, the posterior distribution on Bayesian networks $G \in \mathcal{G}$ can be written as functions of $G_1, \ldots G_p$ in the following way.

$$\Pr(G_1, \ldots, G_p \mid \mathbf{X}) \propto \pi(G_1, \ldots, G_p) \prod_{i=1}^{p} p(X_i \mid X_{G_i})$$

In the following, we will denote subsets of the vector $G = (G_1, \ldots, G_p)$ by $G_A = \{G_k : k \in A\}$, and the subset given by the complement $A^C = \{1, \ldots, p\} \setminus A$ of a set A will be denoted by $G_{-A} = \{G_k : k \in A^C\}$. In particular, the complete graph can be specified by $G = (G_1, \ldots G_p) = (G_i, G_{-i})$ for any $i \in \{1, \ldots, p\}$.

To be able to construct a Gibbs sampler using parent sets, we need to find their conditional distribution, given the other parent sets $G_{-j} = \{G_1, \ldots, G_{j-1}, G_{j+1}, \ldots, G_p\}$. Parent sets G_j for which $G = (G_j, G_{-j})$ is cyclic will have no probability mass in the conditional distribution. Let $K_j^\star$ be the set of parent sets G_j such that $G = (G_j, G_{-j})$ is acyclic. The conditional posterior distribution of G_j is multinomial, with weights given by the posterior distribution of $G = (G_j, G_{-j})$. When the cardinality of $K_j^\star$ is constrained (for example, by restricting the maximum number of parents of each node) the conditional posterior distribution for $G_j \in K_j^\star$ can be evaluated

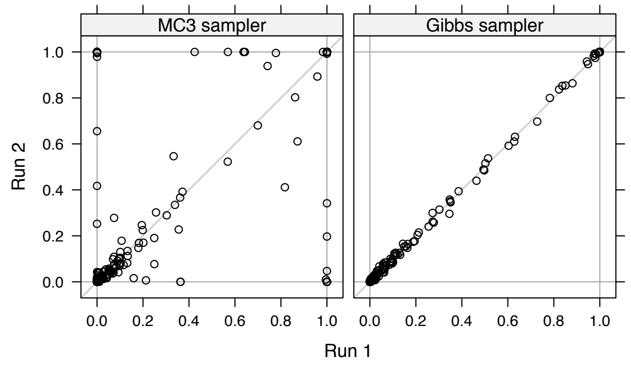


Figure 1: Diagnostic runs for MC^3 (left) and the Gibbs sampler (right). The posterior edge probabilities given by two independent runs are plotted against each other. When the two runs give the same estimates of the posterior edge probabilities, all of the points appear on the line $y = x$. We observe that the two Gibbs runs gives similar posterior edge probabilities, but the MC^3 runs do not. (5 runs of 750,000 samples (MC^3) or 100,000 samples (Gibbs) of each sampler were performed; the first half of the samples were discarded as burn-in; mean Pearson correlation between runs was 0.9999 ± 0.0002 (standard deviation) for Gibbs and 0.6322 ± 0.0477 for MC^3.)

exactly.

$$\begin{aligned}\Pr(G_j \mid G_{-j}, \mathbf{X}) &= \frac{\Pr(G_j, G_{-j} \mid \mathbf{X})}{\Pr(G_{-j} \mid \mathbf{X})} \\ &= \frac{\Pr(G_j, G_{-j} \mid \mathbf{X})}{\sum_{G_j \in K_j^\star} \Pr(G_j, G_{-j} \mid \mathbf{X})} \qquad (1)\end{aligned}$$

We can improve the speed of convergence of this sampler by allowing pairs of parent sets to be sampled together. At each step of the Gibbs sampler we conditionally sample pairs of parent sets (G_{j_1}, G_{j_2}), given the remainder of the graph $G_{-\{j_1,j_2\}}$. Parent sets $G_{-\{j_1,j_2\}}$ such that $G = (G_{j_1}, G_{j_2}, G_{-\{j_1,j_2\}})$ is cyclic have no probability mass in the conditional distribution. Let $K^\star_{j_1,j_2}$ be the set of pairs of parent sets (G_{j_1}, G_{j_2}) such that $G = (G_{j_1}, G_{j_2}, G_{-\{j_1,j_2\}})$ is acyclic. For $(G_{j_1}, G_{j_2}) \in K^\star_{j_1,j_2}$, the conditional posterior distribution is multinomial, by analogy with (1), with weights given by posterior distribution of $G = (G_{j_1}, G_{j_2}, G_{-\{j_1,j_2\}})$.

$$\begin{aligned}&\Pr(G_{j_1}, G_{j_2} \mid G_{-\{j_1,j_2\}}, \mathbf{X}) \\ &= \frac{\Pr(G_{j_1}, G_{j_2}, G_{-\{j_1,j_2\}} \mid \mathbf{X})}{\sum_{(G_{j_1},G_{j_2}) \in K^\star_{j_1,j_2}} \Pr(G_{j_1}, G_{j_2}, G_{-\{j_1,j_2\}} \mid \mathbf{X})}\end{aligned}$$

Similarly, sets of three parent sets can be conditionally sampled. Full technical details are presented in (1).

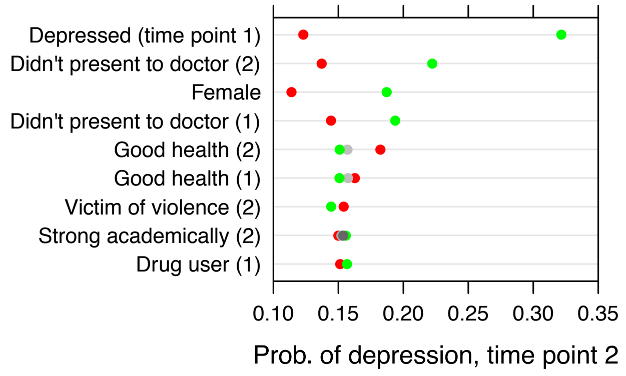


Figure 3: Conditional probability of depression. The conditional probability of being depressed at Wave II given the variable indicated is changed to the level indicated by the colours, conditional on the DAG shown in Figure 2. For binary variables, ● is true, and ● is false; shades of grey indicate intermediate levels. Wave number (time point) is indicated in parentheses. Only variables for which the conditional probability differed between levels by at least 0.005 are displayed.

3 RESULTS

The variables that we consider are detailed in Table 1. As is common when using graphical models (24), all of these variables were grouped, initially into 'Background', 'Wave I' and 'Wave II', and then refined into whether the question asked about the long- or short-term, as shown in Table 2. These groups define constraints on the Bayesian networks that are considered. Specifically, no edges can be directed backwards through the groups. Edges, however, are allowed within groups. For example, no edge is allowed to be directed into 'Gender', and no edge can pass backwards in time, for example, from Depression at Wave II to Depression at Wave I. Additionally, no edge can pass from a short-term variable to a long-term variable, for example, from Depressed at Wave I to Have HIV/AIDS at Wave I.

We precomputed the local scores, and then drew 100,000 samples (the first half of which were discarded as burn-in) using the Gibbs sampler (Section 2.3), which took 30 minutes (on a single core of a cluster computer). The graph space was constrained such that no node had more than 3 parents, to ensure Equation 1 could be evaluated.

We ran 5 independent samplers, with disparate initial states. This enables a simple test of convergence to be performed that compares the posterior edge probabilities obtained from each of the independent runs (25). The agreement between runs can be examined graphically by plotting the edge probabilities against

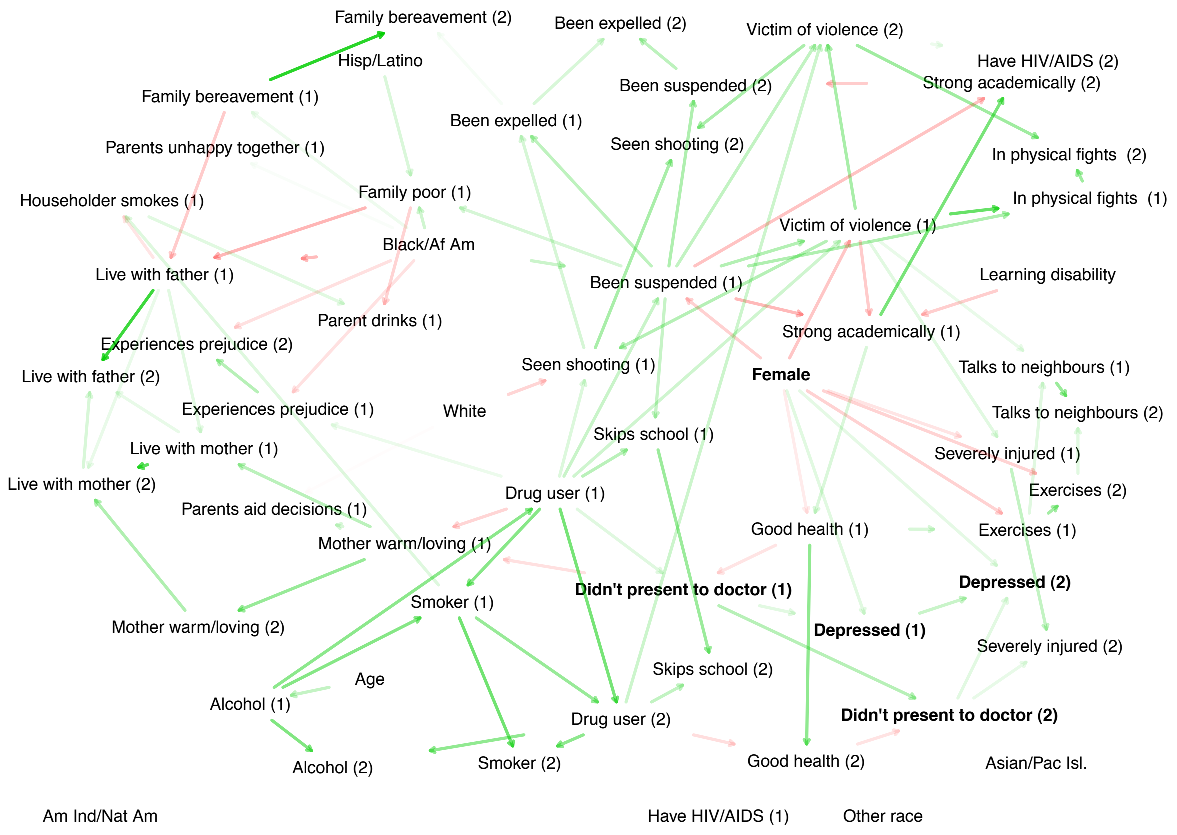


Figure 2: Summary network for the AddHealth variables considered. The edge colors are given by the Kendall correlation coefficents between the two variables, with green edges corresponding to positive correlation, and red edges to negative correlation. The strength of the correlation is indicated by the transparency of the line, with greater transparency indicating weaker correlation. The variables 'Depressed (1)', 'Depressed (2)' and their parents are shown in bold.

each other (Figure 1). Mean Pearson correlation coefficients between edge probabilities from pairs of runs were 0.9999 ± 0.0002 (standard deviation) for the Gibbs sampler and 0.6322 ± 0.0477 for MC^3. The agreement between the independent runs of the Gibbs sampler gave us confidence in our results, in contrast to the large disagreements between MC^3 runs. In addition, cumulative edge probability plots for each edge showed regular excursions around the mean (26), and a numerical diagnostic (27) monitoring the number edges in the sampled graph also clearly suggested that sufficient samples had been drawn ($\hat{R} \approx 1.0$).

The samples drawn using MCMC allow the posterior distribution of Bayesian networks to be approximated. In particular, the samples can be used to estimate the posterior edge probability $P(e|\mathbf{X})$ with $e \in E$. Figure 2 displays all edges with posterior probability of at least 0.5.

Our focus is on depression, the parents of which in Figure 2 we observe are "Didn't present to doctor" and "Gender". It important, however, to note that the model does not say that these are the only factors that are important. For example, "Drug user" at Wave I is related to depression through "Didn't present to doctor" at Wave I and II (Figure 2).

This is shown in Figure 3, which gives the conditional probability of being depressed at Wave 2 when a particular variable is set to a specific value. We see that general health, violence, academic performance and drug use all affect the conditional probability of depression at Wave II. Note that to compute this probability, links from the parents of the variable in which we 'intervene' are removed; this is equivalent to the 'do-operator' in the terminology of Pearl (28).

The analysis reveals the interaction between the many aspects of life that have an impact on depression. The connection between the depression and its two parents in Figure 2 have been previously discussed in the literature. The importance of gender in depression is particularly extensively documented in the literature (8). The connection to a failure in seeking medical care even when the individual thinks they should has also been discussed in the literature, often in terms of poor accessibility of health care services for young people (29, 8). Several decades of research have revealed the complex causation of depression in young people, as suggested by this study (8).

4 DISCUSSION

There is a large amount of information held in large social science questionnaires. In this paper we have examined a graphical model approach to inferring structure amongst the variables in such questionnaires. In contrast to the standard regression-based approaches, a graphical model approach forgoes the need to specify a particular variable as the response. Instead, a more comprehensive estimate of the entire structure of the underlying system can be obtained. Regression approaches posit a particular conditional-independence structure, while graphical approaches allow consideration of more general structures.

The limitations of this study include those of all similar studies using observational data that are collected for multiple audiences. These forms of data, including the longitudinal data used here, do not permit strong causal conclusions to be drawn. In particular there may be important variables that we have not included in the analysis. However, the results are consistent with studies that have used other research approaches including experimental designs. The connection between an individual not seeking medical care when they think they should and depression supports current practice guidance in the UK (30) where there is an emphasis on providing access to health care through the school system rather than expecting young people to seek health care themselves. Not seeking medical care despite believing it should be sought is a complex factor because it captures both barriers to getting medical care within the individual, such as lacking motivation to seek care, and barriers within the individual's environment, such as poor access to care. This may mean that the variable encapsulates a number of different characteristics related to depression, and thus may form a 'marker' for depression. However, the use of a form of the question "Has there been any time over the past year when you thought you should get medical care, but you did not?" as a screening question in different contexts needs further consideration.

This method of analysis clarifies the complexity of depression and suggests why when using traditional methods of analysis it can be difficult to clarify whether or not factors, such as experiences in the family, in the wider community and at school, impact on the experience of depression for young people. It may also suggest why interventions for prevention of depression have not yet been demonstrated to be cost effective (31).

We performed structural inference for the Bayesian network using a Gibbs sampler (1), because MC^3 did not mix in a reasonable time. We have also found (1) this algorithm to be superior to the REV sampler (32), and it has the advantage of avoiding the need to consider an order prior as required by order MCMC methods (33, 34), which induces a bias that can only be corrected exactly by NP-hard computation of a correction factor.

An alternative to the MCMC method used here is the PC-algorithm (10, 11). This method is computationally efficient and is asymptotically consistent. However, to test whether the sample size available here is sufficient to reach the asymptotic regime, we applied the PC-algorithm (without constraints) to 10 different subsamples, each containing 90% of the data. We found that these results differed significantly, with a mean 84 in structural Hamming distance between the pairs of completed partially directed acyclic graphs (CPDAGs) given for the subsamples.

We used a Multinomial-Dirichlet model for the local conditional distributions, which yields a closed-form marginal likelihood. This model posits an entirely general discrete distribution, allowing its form to be guided by the data. However, the number of parameters in the local distributions for this model increases exponentially with the number of parents, which may mean that overly-sparse models are preferred. This is problematic when the sample size of the available data is small, because models with many parameters cannot be assessed adequately without a large dataset. The large sample size of the dataset used here minimises this issue, but it would nonetheless be worthwhile to consider more compact parameterisations. However, estimating such models (35) significantly increases the complexity of the model space, which makes such an approach computationally challenging in this setting.

For this paper, we removed samples with missing data. It is possible to handle missing data formally, for example by using structural EM (36), and similarly consider latent variables (e.g. shared genetics driving both child and parent behaviour). However, at present, doing so whilst robustly exploring large model spaces remains an open challenge. Tackling these computational and inferential issues is a key area for future research.

References

[1] Goudie, R. J. B. and Mukherjee, S. M. (2011). An Efficient Gibbs Sampler for Structural Inference in Bayesian Networks. CRiSM Working Paper 11-21 (Dept. of Statistics, University of Warwick).

[2] Friedman, N. (2004) *Science*, **303**, 5659, 799–805.

[3] Mukherjee, S. and Speed, T. P. (2008) *Proc Natl Acad Sci USA*, **105**, 38, 14313–14318.

[4] Acid, S., de Campos, L. M., Fernández-Luna, J. M., Rodríguez, S., Rodríguez, J. M. and Salcedo, J. L. (2004) *Artif Intell in Med*, **30**, 3, 215–232.

[5] Costello, E. J., Mustillo, S., Erkanli, A., Keeler, G. and Angold, A. (2003) *Arch Gen Psych*, **60**, 8, 837–844.

[6] Costello, E. J., Erkanli, A. and Angold, A. (2006) *J Child Psychol Psych*, **47**, 12, 1263–1271.

[7] Thapar, A., Collishaw, S., Potter, R. and Thapar, A. K. (2010) *Br Med J*, **340**, c209.

[8] Patel, V., Flisher, A. J., Hetrick, S. and McGorry, P. (2007) *Lancet*, **369**, 9569, 1302–1313.

[9] Barnett, P. A. and Gotlib, I. H. (1988) *Psych Bull*, **104**, 1, 97–126.

[10] Spirtes, P., Glymour, C. and Scheines, R. (2000) *Causation, Prediction, and Search* (The MIT Press, Cambridge, MA).

[11] Korb, K. B. and Nicholson, A. E. (2011) *Bayesian Artificial Intelligence* (CRC Press, Boca Raton, FL).

[12] Harris, K. M., Halpern, C. T., Whitsel, E. A., Hussey, J., Tabor, J., Entzel, P. and Udry, J. R. (2009) The National Longitudinal Study of Adolescent Health: Research Design.

[13] Radloff, L. (1977) *App Psych Meas*, **1**, 3, 385–401.

[14] Goodman, E. (1999) *Am J Pub Health*, **89**, 10, 1522–1528.

[15] Roberts, R. E., Lewinsohn, P. M. and Seeley, J. R. (1991) *J Am Acad Child Adolesc Psych*, **30**, 1, 58–66.

[16] Holt, S., Buckley, H. and Whelan, S. (2008) *Child Abuse & Neglect*, **32**, 8, 797–810.

[17] Obot, I. S. and Anthony, J. C. (2004) *J Child Adolesc Subst Abuse*, **13**, 4, 83–96.

[18] Brown, R. A., Lewinsohn, P. M., Seeley, J. R. and Wagner, E. F. (1996) *J Am Acad Child Adolesc Psych*, **35**, 12, 1602–1610.

[19] Battles, H. B. and Wiener, L. S. (2002) *J Adoles Health*, **30**, 3, 161–168.

[20] Larun, L., Nordheim, L. V., Ekeland, E., Hagen, K. B. and Heian, F. (2006) *Cochrane Database Syst Rev*, 3, CD004691.

[21] Heckerman, D., Geiger, D. and Chickering, D. M. (1995) *Mach Learn*, **20**, 197–243.

[22] Madigan, D. and York, J. C. (1995) *Int Stat Rev*, **63**, 2, 215–232.

[23] Kalisch, M. and Bühlmann, P. (2007) *J Mach Learn Res*, **8**, 613–636.

[24] Cox, D. and Wermuth, N. (1996) *Multivariate Dependencies Models, Analysis and Interpretation* (Chapman & Hall, London).

[25] Robert, C. P. and Casella, G. (2004) *Monte Carlo Statistical Methods* (Springer, New York).

[26] Yu, B. and Mykland, P. (1998) *Statistics and Computing*, **8**, 3, 275–286.

[27] Gelman, A. and Rubin, D. B. (1992) *Statistical Science*, **7**, 4, 457–472.

[28] Pearl, J. (2000) *Causality: Models, Reasoning, and Inference* (Cambridge University Press, New York).

[29] Rickwood, D. J., Deane, F. P. and Wilson, C. J. (2007) *Med J Aus*, **187**, 7 Suppl, S35–S39.

[30] National Institute for Health and Clinical Excellence (2005) *Depression in Children and Young People* (NICE, London).

[31] Merry, S. N. (2007) *Curr Opin Psych*, **20**, 4, 325–329.

[32] Grzegorczyk, M. and Husmeier, D. (2008) *Mach Learn*, **71**, 2-3, 265–305.

[33] Ellis, B. and Wong, W. H. (2008) *J Am Stat Assoc*, **103**, 482, 778–789.

[34] Friedman, N. and Koller, D. (2003) *Mach Learn*, **50**, 1-2, 95–125.

[35] Friedman, N. and Goldszmidt, M. (1996) In *Proc. Twelfth Conference on Uncertainty in Artificial Intelligence (UAI-96)* (Morgan Kaufmann Publishers Inc.), 252–260.

[36] Friedman, N. (1998) In *Proc. Fourteenth Conference on Uncertainty in Artificial Intelligence (UAI-98)* (Morgan Kaufmann Publishers Inc.), 129–138.

Table 1: The table shows the label used in the plots above, the number of levels (r), and the exact wording of the question. The ID(s) of the relevant variables in the Add Health dataset are in parentheses. See www.cpc.unc.edu/projects/addhealth for full details of all of these questions.

Label	**r**	**Question**
Female	2	Interviewer, please confirm that R's sex is (male) female. (BIO_SEX)
Hisp/Latino	2	Are you of Hispanic or Latino origin? (H1GI4)
White	2	What is your race? [White] You may give more than one answer (H1GI6A)
Black/Af Am	2	What is your race? [Black or African American] You may give more than one answer (H1GI6B)
Am Ind/Nat Am	2	What is your race? [American Indian or Native American] You may give more than one answer (H1GI6C)
Asian/Pac Isl.	2	What is your race? [Asian or Pacific Islander] You may give more than one answer (H1GI6D)
Other race	2	What is your race? [Other] You may give more than one answer (H1GI6E)
Skips school	4	[If SCHOOL YEAR:] During this school year [If SUMMER:] During the 1994-1995 school year how many times HAVE YOU SKIPPED/DID YOU SKIP school for a full day without an excuse? (H1ED2; H2ED2)
Experiences prejudice	3	[If SCHOOL YEAR:] Students at your school are prejudiced [If SUMMER:] Last year, the students at your school were prejudiced. (H1ED21; H2ED17)
In physical fights	4	In the past 12 months, how often did you get into a serious physical fight? (H1DS5; H2FV16)
Didn't present to doctor	2	Has there been any time over the past year when you thought you should get medical care, but you did not? (H1GH26; H2GH28)
Severely injured	3	Which of these best describes your worst injury during the past year? (H1GH54; H2GH47)
Have HIV/AIDS	2	Have you ever been told by a doctor or a nurse that you had... HIV/AIDS (H1CO16D; H2CO19D)
Seen shooting	3	During the past 12 months, how often did each of the following things happen? You saw someone shoot or stab another person. (H1FV1; H2FV1)
Mother warm/loving	4	Most of the time, your mother is warm and loving toward you. (H1PF1; H2PF1)
Been suspended	2	Have you ever received an out-of-school suspension from school? (H1ED7; H2ED3)
Been expelled	2	Have you ever been expelled from school? (H1ED9; H2ED5)
Good health	3	In general, how is your health? Would you say... (H1GH1; H2GH1)
Talks to neighbours	2	In the past month, you have stopped on the street to talk with someone who lives in your neighborhood? (H1NB2; H2NB2)
Age	5	Age at interview, computed from date of birth, and date of interview (Constructed from IYEAR, IMONTH, IDAY, H1GI1Y, H1GI1M)
Live with mother	2	Indicator variable (Constructed from H1HR3A-T; H2HR4A-Q)
Live with father	2	Indicator variable (Constructed from H1HR3A-T; H2HR4A-Q)
Smoker	4	Frequency of smoking (Constructed from H1TO1/2/5; H2TO1/5)
Drinks alcohol	4	Frequency and amount of drinking alcohol (Constructed from H1TO12/15/18; H2TO15/19/22)
Exercises	3	Amount of exercise (Constructed from H1DA4/5/6; H2DA4-6)
Depressed	2	Rescaled CES-D, following (14) (Constructed from H1FS1-18; H2FS1-18)

Victim of violence	2	Indicator variable (Constructed from H1FV2-6; (H2FV2-5)
Family bereavement	3	Number of bereavements (Constructed from H1NM2/F2, H1FP24A1-5; H2NM4/F4, H2FP28A1-3)
Strong academically	4	Quartiles (Constructed from H1ED11-4; H2ED7-10)
Drug user	2	Indicator variable (Constructed from H1TO30/34/37/41; H2TO44/50/54/58)
Family poor	5	Census Bureau measure of poverty (Constructed from H1HR2/3/7/8, PA55)
Parents unhappy together	4	(Parent asked.) Do you and your partner argue/talk of separating? (Constructed from PB19/20)
Parent drinks	4	(Parent asked.) Number/frequency of drinks (Constructed from PA61/2)
Householder smokes	3	(Parent asked.) Either parent or others in household smokes (Constructed from PA63/4)
Has learning disability	2	(Parent asked.) Does (he/ she) have a specific learning disability, such as difficulties with attention, dyslexia, or some other reading, spelling, writing, or math disability? (PC38)
Parents aid decisions	5	(Parent asked.) How often would it be true for you to make each of the following statements about {child's name}? {Child's name} and you make decisions about (his/ her) life together. (PC34B)

Table 2: The groupings of the variables that were used to determine constraints on the Bayesian networks. Each variable in the analysis is either a Background variable, or from Wave I or Wave II of the Add Health study. Within each wave of the study, variables were further classified into whether they asked about the short- or long-term.

Background	Wave I Long-term	Wave I Short-term	Wave II Long-term	Wave II Short-term
Female	Skips school	Househol. smokes	Seen shooting	Smoker
Age	Experiences prejudice	Smoker	Alcohol	Live with mother
Hisp/Latino	In physical fights	Live with mother	Drug user	Live with father
White	Didn't pres. to doctor	Live with father	Mother warm/loving	Talks neighbours
Black/Af Am	Severely injured	Parent drinks	Have HIV/AIDS	Exercises
Am Ind/Nat Am	Have HIV/AIDS	Talks neighbours	Family bereavement	Depressed
Asian/Pac Isl.	Seen shooting	Exercises	Experiences prejudice	
Other race	Mother warm/loving	Depressed	Been expelled	
Has learning dis.	Been suspended		Been suspended	
	Been expelled		Victim of violence	
	Good health		In physical fights	
	Alcohol		Strong academically	
	Victim of violence		Didn't pres. to doctor	
	Family bereavement		Skips school	
	Strong academically		Severely injured	
	Drug user		Good health	
	Family poor			
	Parents unhappy togth.			
	Parents aid decisions			

Constructing a Dynamic Bayes Net Model of Academic Advising

Joshua T. Guerin and Judy Goldsmith[*]
Department of Computer Science
University of Kentucky
Lexington, KY 40506-0046
jtguer2@uky.edu, goldsmit@cs.uky.edu

Abstract

In this paper we apply ideas from collaborative filtering to the problem of building dynamic Bayesian network (DBN) models for planning. We demonstrate that item-based collaborative filtering can be used to construct dynamic Bayesian networks for use in large, factored domains with sparse data. Such Bayesian networks can model the transition function for decision-theoretic planning. We demonstrate the feasibility and effectiveness of this technique on an academic advising domain, based on student grades in computer science and related courses at the University of Kentucky.

1 Introduction

In this paper we examine the use of memory-based CF algorithms for constructing static models of data. This work is grounded in the real-world domain of academic advising. We use an item-based collaborative filtering algorithm to generate dynamic Bayesian network models of an advising domain from sparse grade data.

Collaborative filtering (CF) algorithms are designed to aggregate the opinions or preferences of a large number of users to extrapolate information about unnamed preferences for new and existing users. Recommendation systems are constructed using CF techniques to locate items in a database which a target user is likely to prefer. Preferences are typically defined by grades that the user provides either explicitly (by the user providing grades for items that have already seen) or implicitly (often indicated by patterns of behavior such as browsing habits). These grades can be represented in a number of ways, but are often numerical in nature; most recommender systems ask for a numerical grade (1–5) or a grade based on letters or "stars" which is easily mapped to numerical grade (for instance 1–5 stars, or a letter grade of A–E).

CF algorithms can be roughly divided into model-based and memory-based algorithms. Model-based algorithms involve generating a predictive model based on the data and using it to make preference-related predictions. One formalism that has seen success in model-based CF is the Bayesian network [1].

Memory-based CF operate over the database of items to make predictions, leveraging a measure of similarity between users or (more commonly) between items to determine grades for unseen items. This class of algorithms provides us with several notable features which are useful for making predictions. Namely, these algorithms are designed to operate over very large datasets (common examples include the Netflix dataset, the MovieLens datasets, or the Amazon.com recommendation system). Such datasets typically contain tens of thousands of items and grades from hundreds of thousands of users, however since most users only provide grades for a small percentage of items these datasets are very sparse. Because of this, modern recommendation systems must scale well and must work well with very sparse data.

2 A Predictive Model for Academic Advising

Reasoning in the domain of undergraduate academic advising is often approached as a deterministic process. Short and long-term decision making is based on the assumption that a student's actions (i.e., taking one or more courses) will succeed. This doesn't capture the nuances and complexity of the real world. The outcome of taking a course can not always be predicted with certainty; even a student who makes consistent A's may perform poorly at some point.

Given the stochastic nature of grade prediction, it may

[*]This material is based upon work supported by the National Science Foundation under Grant No. 1049360.

be desirable to construct statistics-based models of student performance from real world data. Students leave behind tangible evidence of progress in the form of transcript data. Universities amass a wealth of data with which to make predictions about grades. From this we can construct probabilistic predictive models. The Dynamic Bayesian Network (DBN) formalism has a number of features which make it ideal for this sort of modeling.

A DBN model consists of a directed acyclic graph with links representing temporal, probabilistic relationships between variables and conditional probability tables (CPTs) that specify those relationships quantitatively [2] (a discussion of DBNs will follow in Section 3.2). We are interested in a class of DBNs which model only a single time-step known as 2-slice DBNs. This imposes restrictions on the underlying graphical structure. Specifically, variable values at one time-step are conditioned only on the values of parent variables at the previous time-step.

The structural restrictions imposed on 2-slice DBNs make them a potentially compact representation for decision theoretic planning. For this reason we limit our attention to 2-slice DBNs.

In the case of discrete-valued variables, each child node in the DBN has an associated conditional probability table (CPT) which gives a probability distribution over possible values for every possible assignment to parent variables (incoming edges in the graph) at a previous time-step. Because all possible assignments to parent variables may need to be enumerated explicitly, CPT size is exponential in the number of parent variables. For example, a CPT for a single course with 5 parents, each of which has 6 possible values (A–D, Failure, and NOT_TAKEN) will have $6^5 = 7,776$ rows, each containing a probability distribution over the 6 possible outcomes.

For modern computers, tables of this size are unlikely to cause representational issues. However the need for enough data to populate a table's $6^6 = 46,656$ probabilities makes seemingly abundant data seem rather sparse. Popular or required courses may be taken by hundreds or even thousands of students within the span of several years, but even this is insufficient to derive realistic probability distributions from straight statistical analysis. This problem is worse for most courses (and for smaller colleges and universities) where enrollment over several years may reach only hundreds of students or fewer.

In order to deal with the problem of prediction when data is sparse, we turn to techniques from collaborative filtering to aggregate the data that is available. Collaborative filtering algorithms are commonly used to narrow down choices based on a user's preferences and the preferences of current and past users. A common example application is predicting preferences over unseen items (movies, music, groceries) based on grades given for other items [1].

The problem of grade prediction very closely resembles the problem of grade prediction in collaborative filtering: make predictions about a student's grades in untaken courses, given their past grades and the transcript data from many past students. Letter "grades" can map directly to integers where A=1 and Failure=5.

In this paper we present a simple collaborative filtering algorithm, and demonstrate how it is used to generate a valid DBN model of state transitions in the advising domain. We use real-world data from the Computer Science Department at our university as a testbed for our model generation techniques.

3 Background

3.1 Bayesian Networks

A *Bayesian network* is a directed acyclic graph $G = \langle V, E \rangle$, where each vertex $v \in V$ is a variable with domain $dom(v)$. Each $v \in V$ has an associated probability distribution over values in $dom(v)$, conditioned on the values of $Pa_v \subset V$, the parents of v. These conditional probability distributions are usually enumerated in tabular form as conditional probability tables (CPTs) for each variable.

Learning of Bayesian networks is often divided into structure learning and parameter learning. Structure learning is the problem of learning the graphical structure E by discovering predictive or causal dependencies between variables. Parameter learning is the problem of learning the conditional probability distributions for a given network structure.

Because the space of all possible networks is very large, structure learning is usually approached as a heuristic search problem or an exact search of a constrained version of the search space (see [4,6,11] for examples). Search for an an optimal (or near optimal) network structure is guided by some scoring function (one example is the log-likelihood scoring function).

Once structure is known, CPT parameters (probability distributions over outcomes) are generally learned from the data. Examples of parameter learning for DBNs include maximum likelihood estimation (one example being the expectation maximization [3] algorithm), or Bayesian estimation.

Unlike most Bayesian network learning algorithms, our

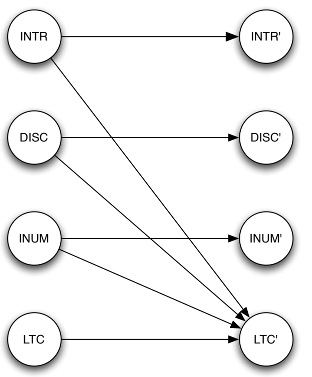


Figure 1: An example DBN structure.

validation is based on the quality of *predictions* rather than of *inference*. In other words, our work looks forward in time rather than backward. We conjecture that good learned probabilistic planning models may actually differ from probabilistic inference models learned from the same data.

3.2 2-Slice Dynamic Bayesian Networks

Bayesian networks have been demonstrated to be useful for inference in a number of domains, however the standard framework does not have an explicit notion of time. A *dynamic Bayesian network* builds upon the Bayesian network idea, incorporating temporal or sequential aspects of data into its structure. Variables at one time-step may influence the value of variables at future time-steps (or at the same time-step).

We are interested in a special case of dynamic Bayesian networks, the 2-slice dynamic Bayesian network. A 2-slice dynamic Bayesian network is a Bayesian network with $V = V, V'$, representing variables at time t and $t+1$, and edges from V to V' (and sometimes between vertices in V'). In DBNs of this form, V and V' may be visualized as two separate columns representing, respectively, the variables at time t and $t+1$.

This structural formulation implies two theoretical assumptions under which we operate. These are a *stationary assumption* where models are not time-dependant and a *Markov assumption* where there is no memory of past states; future values are conditioned *only* over the current system state.

Figure 1 gives the structure of an example of a 2-slice DBN which could be used for planning in an academic domain. This DBN structure shows that the expected grade in Logic and Theory of Computing (LTC) is conditioned over the grades obtained in Introduction to Programming (INTR), Discrete Mathematics (DISC), and Introduction to Numerical Methods (INUM).

Rather than selecting a single ideal structural size we choose to make structure size a parameter of our algorithm. Since we are considering models for the purpose of planning, we must consider the tradeoff between accuracy of the representation and tractability of planning. Our goal is to be able to generate DBNs of different sizes for different purposes. We examine how our algorithm fares as a function of structure size in Section 5. At this point we are left with the question of how to select n parent nodes for each node.

Goldenberg et al. approached a similar problem of learning Bayesian network structures from sparse data using frequent set mining [5]. Frequent sets are widely used in data mining for learning common co-occurrence between sets of items. The idea of applying frequent set mining to academic advising may be useful in other capacities (learning combinations of courses which should or should not be taken together), however co-occurrence of actions is less applicable to building predictive models of advising; courses which are frequently taken together are unlikely to make good predictors for each other. Parent courses should be taken before child courses, otherwise they provide little information.

Rather than using co-occurance we make the assumption that similar variables make better predictors than dissimilar variables. We examine the use of pairwise *item similarity* in selecting parent nodes. Item similarity is commonly used in collaborative filtering and other data mining applications to determine which items hold the most predictive power for a target item, allowing for better predictions to be made.

One of the most common approaches for collaborative filtering is to use the database of user grades to determine item-item similarity. For each pair of items in the database a vector of grades is created (retaining only grades where users voted for both items) [7]. To these vectors a number of distance metrics can be applied. In our implementation we tested two common vector similarity metrics: Pearson's correlation coefficient and cosine similarity.

3.3 Collaborative Filtering

Collaborative filtering recommendation algorithms typically fall into one of two general categories: model-based algorithms and memory-based algorithms [1]. Model-based algorithms involve generating a model based on data, and using the model to make predictions. We are interested in memory-based algorithms which use the entire data set to make predictions. This class of algorithms is described in Section 3.4.

Collaborative filtering algorithms also rely heavily on the notion of similarity. That is, similar users are likely to assign similar grades to items. Likewise, similar items may also be given similar grades. Collaborative filtering systems often employ one of these assumptions. These are known as *user-based* and *item-based* collaborative filtering. In this paper we focus on the use of item-based collaborative filtering because of the performance demonstrated by these algorithms and because of their user-independent nature.

3.4 Item-Based Collaborative Filtering

The collaborative filtering algorithm that we used in this paper is an item-based algorithm presented by Sarwar et al. [7]. First, item-item similarity is calculated over all items in the database. For item-item similarity we are using Pearson's correlation coefficient and cosine similarity. For a user u and an item i, predictions are made using the weighted sum of u's grades for all items which are similar to i. This can be expressed as:

$$p_{u,i} = \frac{\sum_{all\ similar\ items,N}(s_{i,N} * R_{u,N})}{\sum_{all\ similar\ items,N}(|s_{i,N}|)}. \qquad (1)$$

Here, $p_{u,i}$ is the predicted grade that user u might give item i, $s_{i,N}$ is the similarity between items i and N, and $R_{u,N}$ is the grade that u provided for item N.

Equation 1 produces a single, most likely grade for the given user and item. Because a DBN requires a probability distribution over *all* possible grades, we are not yet ready to encode our DBNs.

4 Algorithm Details

The CF algorithm based on the function $p_{u,i}$ described in Section 3.4 defines a deterministic version of the DBN CPTs that we want to generate. We use these predictions and the data from past students' transcripts to generate probability distributions over possible grades to produce full CPTs. Algorithm 1 describes the process of turning deterministic predictions from $p_{u,i}$ into CPTs.

In this algorithm we make the assumption that deviation from predictions in past data will produce a distribution which is a reasonable approximation of the probability distribution.

Given the predictions from the CF function described in 4, we build a distribution table, *grade_distribution*, for the set of items with rows and columns indexed by predicted and actual grades. If G_1 and G_2 are possible grades, then the *grade_distribution*$[G_1][G_2]$ entry in the table is the number of transcripts for which the CF algorithm predicted G_1 and the student received G_2 for the class in question.

After we construct *grade_distribution* we normalize each row of the table to form probability distributions. For a grade g, row *grade_distribution*$[g]$ is now a probability distribution over actual grades when R predicts g.

```
Input: Past_Users - a database of past user grades.
Output: CPT - A set of CPTs for each course
foreach user in Past_Users do
    foreach item in user's graded items do
        p = $p_{user,item}$;
        actual = actual grade for item;
        grade_distribution[p][actual]++;
    end
end
normalize rows of grade_distribution;
foreach item do
    T = create prediction table for item;
    foreach row in T do
        u* = temporary user using grade assignments
        in row;
        p = $p_{user,item}$;
        add distribution from grade_distribution[p] to
        current row of T;
    end
    CPT(course) = T;
end
```

Algorithm 1: Generate DBNs from CF predictions

The second half of our algorithm constructs a set of prediction tables for each course. A prediction table T for a course c reflects the overall structure of a final CPT for c; each row of T contains a set of values for parent variables (defined by δ and our distance metric). For each row of T, we fill in the probability distribution over grades using the appropriate row of *grade_distribution*.

Each row of the prediction table T implies a hypothetical user transcript as an assignment over past grades. Using R we can make a prediction p for each row. We select a probability distribution from *grade_distribution*$[p]$, adding probability distributions over grades to each row of T.

After completion, CPT is a set of CPTs for each course, where $CPT(c)$ is the CPT for course c.

5 Results

In this section we describe the tests we run on the academic advising data. We evaluate the two variants of the item-based collaborative filtering algorithm on this dataset. We also generate two baseline DBN models and two collaborative filtering based models, and analyze their performance on this dataset.

5.1 Data and Experimental Setup

Models are generated from the transcript data for approximately 4760 undergraduate students who enrolled during the 2000–2003 academic years. These anonymized data are a time-stamped (semester and year) series of transactions labeled with course and instructor information and grade outcomes. Because we have meta-data from computer science courses, we restricted our attention to students who took computer science courses during their academic careers.

Our analysis is broken down into two steps: collaborative filtering evaluation and DBN evaluation. We chose to evaluate the item-based collaborative filtering algorithm first to give a measurement of the algorithm's performance on an academic dataset. Testing of both collaborative filtering and DBNs is performed using 10-fold cross validation (partitioned randomly).

We are looking at two methods for evaluating the item-based collaborative filtering algorithm on this dataset: mean absolute error and the percent of misclassified predictions. Together, these statistics give us an indication of how far predictions are from actual grades and how often predictions are misclassified, respectively. We selected these statistics because they are fairly straightforward to interpret, and because mean absolute error has been used in the past for collaborative filtering evaluation, allowing comparison to performance on other datasets.

As a baseline for comparison of our 2-slice DBNs we generated baseline DBNs using more standard techniques. Baseline DBN structures were found through exhaustive search of the network structure space, using Bayesian information criterion (BIC) [8] as a scoring function. The highest scoring network was selected for a specified neighborhood size, and parameters were estimated using both maximum likelihood (ML) and Bayesian parameter estimation. Baseline DBNs were generated using the bnlearn software package [9].

As a means of evaluating the performance of the DBNs we calculated the log-likelihood loss of the models, and the percent of misclassified predictions. Log-likelihood loss is the negation of the log-likelihood, which we wish to minimize. "Predictions" in this case are similar to the deterministic predictions made by a collaborative filtering algorithm. We select the most likely outcome as a deterministic prediction and count the number that were classified correctly/incorrectly. This also gives us a basis for comparing our DBNs to the item-based collaborative filtering algorithm.

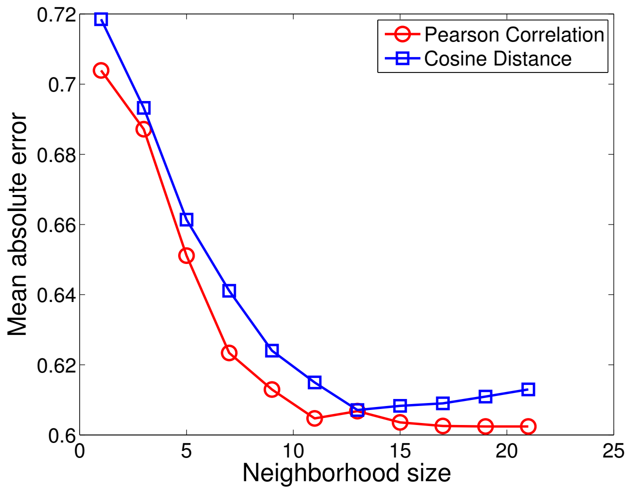


Figure 2: CF prediction mean absolute error

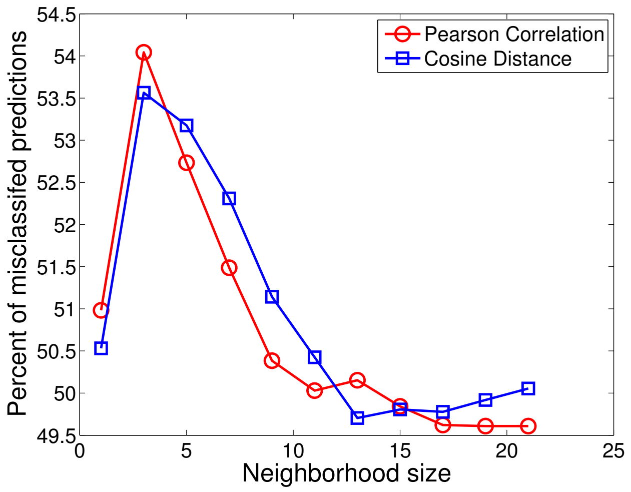


Figure 3: Percent of misclassified predictions.

5.2 Collaborative filtering evaluation

Figures 2 and 3 show that the two different distance metrics, Pearson's correlation and cosine similarity, yield DBNs that perform very similarly. These figures display the mean absolute error and percent of misclassified predictions for both Pearson correlation and cosine distance similarity metrics.

Figure 2 shows that mean absolute error decreases swiftly as the neighborhood size increases. After about 11 neighbors this decrease slows and little change is observed as the number of predictors continues to increase. This curve is similar to tests conducted on the MovieLens dataset [7].

Figure 3 shows how the percent of misclassified predictions changes as neighborhood size increases. At first there is an abrupt jump in this percent, however afterward this curve resembles the curve for mean absolute error, with an apparent ideal neighborhood size of about 15 neighbors.

5.3 DBN evaluation

Figures 4 and 5 show that the DBNs learned using collaborative filtering (Pearson and cosine) outperform the baseline DBNs. Figure 4 shows the log-likelihood loss averaged over all models for a given neighborhood size. Figure 5 shows the percent of misclassified predictions for each model. Baseline DBNs are labeled as "Bayes" and "ML" for their parameter estimation methods. Collaborative filtering inspired DBNs are labeled "Pearson" and "Cosine" for the distance metric used in the collaborative filtering algorithm.

In terms of minimizing loss (Figure 4), the maximum-likelihood, Pearson, and cosine models show similar performance. At a neighborhood size of 2, these models have a log-likelihood loss tightly clustered around 1.14–1.16. Loss shows a steady decrease as the neighborhood size increases. However, the Bayesian model appears unable to cope with increasing neighborhood size, showing an increasing loss. This is likely due to the sparsity of data, and the increase in the possible number of configurations that corresponds with an increased neighborhood size.

Classification accuracy (Figure 4) shows steady improvement as neighborhood size increases across all models, with collaborative filtering models showing much better accuracy than Bayes and maximum-likelihood models at all neighborhood sizes. At a neighborhood size of only one the Pearson and cosine models show comparable accuracy (48.74-49.45% misclassified respectively) to the ML model Bayesian model at a neighborhood size of 5 (49.52% misclassified) and 7 (49.47% misclassified), respectively. At a neighborhood size of 10, the Pearson model shows the lowest misclassification rate at approximately 42.18%.

Comparing Figures 3 and 5, we find that at 6-7 parent variables, our baseline DBNs outperformed the item-based collaborative filtering algorithm. This is consistent with other experiments that demonstrated that Bayesian methods of classification showed better results than the standard item-based algorithm [10].

However, in terms of the percent of misclassified observations the CF-based DBNs outperformed both our benchmarks *and* the item-based collaborative filtering algorithm that they were based on at all neighborhood sizes. At 17 neighbors the CF algorithm hit a misclassification rate of approximately 49.6%, however at a neighborhood size of only 10 the CF-based DBNs had a misclassification rate of approximately 42.18%. This indicates that by observing the way that predicted grades deviated from actual grades on a per-item basis (as we did with CPT generation in algorithm 1) one may be able to construct a better collaborative filtering algorithm.

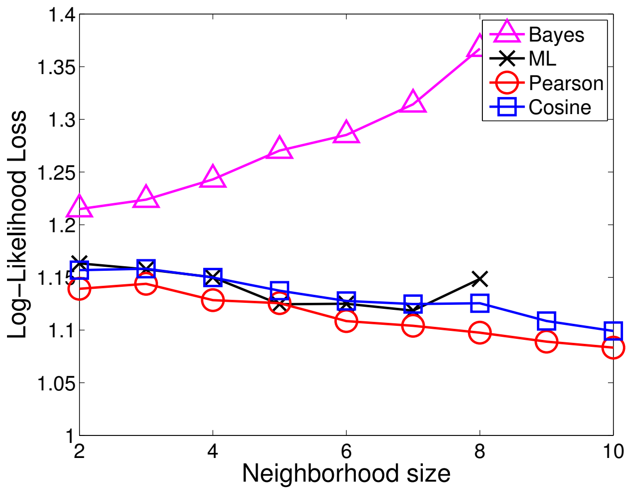


Figure 4: Average Log-Likelihood Loss.

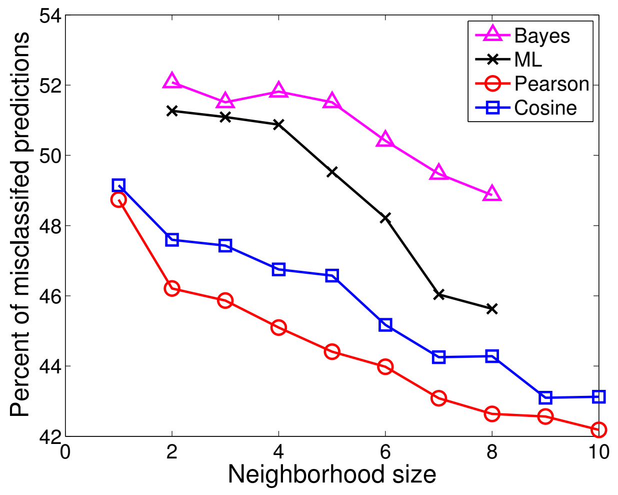


Figure 5: Observations misclassified by the DBN.

Across all tests the Pearson model showed a slight advantage over the cosine model. This indicates that improvements in the item-based collaborative filtering used to generate DBNs may lead to improvements in resulting DBN models.

6 Conclusions and future directions

Our goal is to develop DBN transition models for the purpose of decision-theoretic planning. In this paper we have presented a novel approach for generating DBN planning models from sparse data. We used academic advising data to show the validity of our method. One of the benefits of this method is the flexibility regarding the use of collaborative filtering recommendation algorithms. Our models were constructed using a generic item-based collaborative filtering algorithm. Any similar item-based collaborative filtering algorithm can be used in its place, giving us a wide variety of algorithms which can be employed using off-the-shelf software packages.

We are also investigating methods for modeling utility in this and similar domains, as well as decision-theoretic planning algorithms that can run on domains of the size and complexity presented here, and larger.

Acknowledgments

We thank the University's SSTARS lab consultants for assistance with statistical analysis. We also thank Nick Mattei, Robert Crawford, Daniel Michler and other members of our lab for their contributions to this and related projects.

References

[1] John Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence (UAI-98)*, pages 43–52. Morgan Kaufmann, 1998.

[2] Thomas Dean and Keiji Kanazawa. A model for reasoning about persistence and causation. *Computational Intelligence*, 5(3):142–150, 1990.

[3] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[4] Nir Friedman, Iftach Nachman, and Dana Pe'er. Learning Bayesian network structure from massive datasets: The "sparse candidate" algorithm. In *The 15th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 206–215, 1999.

[5] Anna Goldenberg and Andrew Moore. Tractable learning of large Bayes net structures from sparse data. In *ICML '04: Proceedings of the 21st International Conference on Machine learning*, New York, NY, USA, 2004. ACM.

[6] Kaname Kojima, Eric Perrier, Seiya Imoto, and Satoru Miyano. Optimal search on clustered structural constraint for learning Bayesian network structure. *Journal of Machine Learning Research*, 11:285–310, 2010.

[7] Badrul Sarwar, George Karypis, Joseph Konstan, and John Reidl. Item-based collaborative filtering recommendation algorithms. In *WWW '01: Proceedings of the 10th International Conference on World Wide Web*, pages 285–295, New York, NY, USA, 2001. ACM.

[8] Gideon Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6(2):461–464, 1978.

[9] Marco Scutari. Learning Bayesian networks with the bnlearn R package. *Journal of Statistical Software*, 35(3):1–22, 2010.

[10] Xiaoyuan Su and Taghi M. Khoshgoftaar. Collaborative filtering for multi-class data using belief nets algorithms. In *Proceedings of the 18th IEEE International Conference on Tools with Artificial Intelligence*, ICTAI '06, pages 497–504, 2006.

[11] Ioannis Tsamardinos, Laura E. Brown, and Constantin F. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65(1):31–78, 2006.

Context-dependent Incremental Intention Recognition through Bayesian Network Model Construction

Han The Anh and **Luís Moniz Pereira**
Centro de Inteligência Artificial (CENTRIA)
Departamento de Informática, Faculdade de Ciências e Tecnologia
Universidade Nova de Lisboa, 2829-516 Caparica, Portugal
$h.anh@fct.unl.pt, lmp@di.fct.unl.pt$

Abstract

We present a method for context-dependent and incremental intention recognition by means of incrementally constructing a Bayesian Network (BN) model as more actions are observed. It is achieved with the support of a knowledge base of readily maintained and constructed fragments of BNs. The simple structure of the fragments enables to easily and efficiently acquire the knowledge base, either from domain experts or automatically from a plan corpus. We exhibit experimental results improvement for the Linux Plan corpus. For additional experimentation, new plan corpora for the iterated Prisoner's Dilemma are created. We show that taking into account contextual information considerably increases intention recognition performance.

1 INTRODUCTION

We propose a method for intention recognition in a dynamic, real-world environment. An important aspect of intentions is *future-directedness*, i.e. if we intend something now, we mean to execute a course of actions to achieve something in the future [3]. Most actions may be executed only at a far distance in time. During that period, the world is changing, and the initial intention may be changed to a more appropriate one or even abandoned [3, 6]. An intention recognition method should take into account these changes, and may need to reevaluate the intention recognition model depending on some time limit; in addition, as new actions are observed, the model should be reconfigurable to incorporate them.

Generally, *intention recognition* (also called *goal recognition*) is defined as the process of becoming aware of the intention of another agent and, more technically, as the problem of inferring an agent's intention through its actions and their effects on the environment [10]. *Plan recognition* is closely related to intention recognition, extending it to also recognize the plan the observed agent is following in order to achieve his intention [20]. Intention recognition is performed in domains in which it is better to have a fast detection of just the user's goal/intention rather than a more precise but time consuming detection of the complete user's plan, e.g. in the interface agents domain [12].

In this work, we use Bayesian Networks (BN) as the intention recognition model. The flexibility of BNs for representing probabilistic dependencies and the efficiency of inference methods for BN have made them an extremely powerful and natural tool for problem solving under uncertainty [16, 17]. We present a knowledge representation method to support incremental BN construction for performing intention recognition during runtime, from an initially given domain knowledge base. As more actions are observed, a new BN is constructed reinforcing some intentions whilst ruling out others (Section 3). This incremental method allows domain experts to specify knowledge in terms of small and simple BN fragments, which can be easily maintained and changed. Alternatively, these fragments can be learned from data. Our intention recognition method is evaluated on the Linux Plan corpus [2] (Section 5) and on our new, so-called IPD plan corpora (Section 6). We also propose a method to represent relationship among intentions when considering the case that agents may pursue multiple intentions simultaneously (Section 4). It is an indispensable aspect, but mostly omitted in prior works, which also allows us to sometimes significantly decrease the complexity of the probability inference [7].

It is inspired in that knowledge experts often consider a related set of variables together, and organize domain knowledge in larger chunks. An ability to represent conceptually meaningful groupings of variables and their interrelationships facilitates both knowledge elicitation and knowledge base maintenance [14, 13]. To this end, there have been several methods proposed for BN construction from small and easily maintained network fragments [16, 19, 14, 15, 13]. In essence, a combination of BNs is a graph that includes all nodes and links of the networks, where nodes with the same name are combined into a common node. The main issue for a combination method is how the influence of different parents of the common node can be combined in the new network, given the partial influence of each parent in the corresponding fragment. The most popular method is Noisy-Or, firstly proposed by [16] for BNs of Boolean variables, and generalized by [22] for the general case of arbitrary domains.

2 BAYESIAN NETWORKS

Definition 1 *A BN is a pair consisting of a directed acyclic graph (DAG) whose nodes represent variables and missing edges encode conditional independencies between the variables, and an associated probability distribution satisfying the Markov assumption of conditional independence, saying that variables are independent of non-descendants given their parents in the graph [16, 17].*

In a BN, associated with each node of its DAG is a specification of the distribution of its variable, say A, conditioned on its parents in the graph (denoted by $pa(A)$)—i.e., $P(A|pa(A))$ is specified. If $pa(A) = \emptyset$ (A is called *root* node), its unconditional probability distribution, $P(A)$, is specified. These distributions are called Conditional Probability Distribution (CPD) of the BN.

The joint distribution of all node values can be determined as the product of conditional probabilities of the value of each node on its parents $P(X_1, ..., X_N) = \prod_{i=1}^{N} P(X_i|pa(X_i))$, where $V = \{X_i | 1 \leq i \leq N\}$ is the set of nodes of the DAG.

Suppose there is a set of evidence nodes (i.e. their values are observed) in the DAG, say $O = \{O_1, ..., O_m\} \subset V$. We can determine the conditional probability distribution of a variable X given the observed value of evidence nodes by using the conditional probability formula

$$P(X|O) = \frac{P(X,O)}{P(O)} = \frac{P(X, O_1, ..., O_m)}{P(O_1, ..., O_m)} \quad (1)$$

where the numerator and denominator are computed by summing up the joint probabilities over all absent variables with respect to V.

3 INCREMENTAL INTENTION RECOGNITION

In [18], a general BN model for intention recognition is presented and justified based on Heinze's intentional model [10]. Basically, the BN consists of three layers: cause/reason nodes in the first layer (called *pre-intentional*), connecting to intention nodes in the second one (called *intentional*), in turn connecting to action nodes in the third (called *activity*) (Figure 1). In this work, we

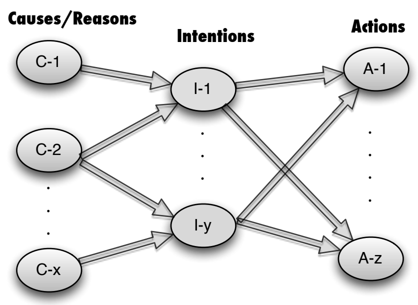


Figure 1: Bayesian Network for Intention Recognition.

present a method for incrementally constructing such BN model for performing *incremental* intention recognition.

Definition 2 (Intention Recognition BN – IRBN) *A BN for intention recognition (IRBN) W is a triple $\langle\{Cs, Is, As\}, pa, P_W\rangle$ where*

- Cs, Is *and* As *are the sets of cause/reason nodes, intention nodes and action nodes, respectively. They stand for binary random variables (i.e. their value is either* true (T) *or* false (F)*).*
- pa *is a mapping which maps a node to the set of its parent nodes such that: $pa(C) = \emptyset \; \forall C \in Cs$; $pa(I) \subseteq Cs \; \forall I \in Is$; and $\emptyset \neq pa(A) \subseteq Is \; \forall A \in As$.*
- *CPD tables are given by the probability distribution P_W, i.e. $P_W(X|pa(X))$ defines the probability of X conditional on $pa(X)$ in W, $\forall X \in Cs \cup Is \cup As$.*

The intention recognition method will be performed by incrementally constructing an IRBN as more actions are observed. The construction is based on a prior knowledge base consisting of Unit BN Fragments.

Definition 3 (Unit Fragments) *There are two types of unit fragments used for IRBN model construction:*

1. *A unit fragment* for an action A *consists of an intention I connecting to (i.e. causally affecting) A, and is denoted by $UF_{\mathfrak{A}}(I, A)$.*
2. *A unit fragment* for an intention I *consists of a context-independent and fixed over time set of causes/reasons* Cs *connecting to (i.e. causally affecting) I, and is denoted by $UF_{\mathfrak{I}}(Cs, I)$.*

Definition 4 (Knowledge Base) *The domain knowledge base* KB *consists of a set of actions Δ, a set of intentions Υ, a set of unit fragments for each action in Δ and a single unit fragment for each intention in Υ, satisfying that*

- *An intention I has a unique unit fragment in KB. The set of its parents (causes) and the CPD table associated with it are fixed. Let $\mathfrak{C}(I)$ denote the set of parents of I and $P_{KB}(I|\mathfrak{C}(I))$ define its CPD table.*

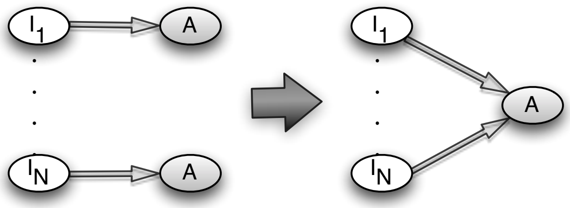


Figure 2: Noisy-OR Combination Method

- *A cause C has the same prior probability distribution in all the unit fragments (for intentions) that it belongs to, denoted by $P_{KB}(C)$.*

The simple structures of unit fragments enable domain experts to easily construct and maintain the knowledge base. The fragments also can be learnt from appropriate datasets, as we shall see later with the Linux and IPD corpora. Before presenting the intention recognition algorithm, let us define some (original) operators for handling CPD tables and IRBNs.

3.1 OPERATORS FOR CONSTRUCTING IRBN

As a new action A is observed, we need to incorporate it into the current IRBN. First, appropriate unit fragments for A are selected from KB. Let *select(A)* denote the set of *all* unit fragments for A from KB [1]. They are then combined using the Noisy-OR method [16, 22], thereby obtaining a BN with a single action A (Figure 2).

Definition 5 (Unit IRBN via Noisy-OR) *The* Unit IRBN *for action* A *is an IRBN with a single action, denoted by $irBN(A) = \langle\{Cs, Is, \{A\}\}, pa, P_W\rangle$. It is obtained via* Noisy-OR *method as follows. Let $select(A) = \{UF_{\mathfrak{A}}(I_1, A), ..., UF_{\mathfrak{A}}(I_N, A)\}$ and for $1 \leq i \leq N$, $P(A = T|I_i = T) = q_i$ (defined in fragment $UF_{\mathfrak{A}}(I_i, A)$). Then,*

- $Is = \{I_1, ..., I_N\}$*;* $Cs = \bigcup_{I \in Is} \mathfrak{C}(I)$*;*
- $pa(I) = \mathfrak{C}(I)\ \forall I \in Is$*;* $pa(A) = Is$*;*
- $P_W(C) = P_{KB}(C)\ \forall C \in Cs$*;* $P_W(I) = P_{KB}(I)\ \forall I \in Is$*; and, according to the* Noisy-OR *method,* $P_W(A = T|pa(A)) = 1 - \prod_{i:I_i=T}(1 - q_i)$.

The rationale and appropriateness of the application of the Noisy-OR method here for combining unit fragments is based on the intuition that each intention can be interpreted as a "*cause*" of action A; and action A occurs when one or more of the intentions are active. Detailed arguments for this can be found in [5, 16].

[1]The selection can be done in a context-dependent manner, but it is beyond the scope of this paper.

Definition 6 (Project of CPD Table) *Let Tb be a CPD table defining $P(X|V)$, the probability of a random variable X conditional on a set of random* binary *variables V, and $V' \subsetneq V$. The project of Tb on V', denoted by* $\texttt{proj}(\texttt{Tb}, \texttt{V}')$*, is the part of Tb corresponding to all variables in $V \setminus V'$ being false.*

Now we need to combine the obtained unit IRBN, $\texttt{irBN}(\texttt{A})$, with the current IRBN. For that, in the sequel we define how to combine two IRBNs. Intuitively, we simply add up all the new nodes and links of the new IRBN to the current IRBN, keeping the CPD tables from the original IRBNs.

Definition 7 (Combination of IRBNs) *Let $W_1 = \langle\{Cs_1, Is_1, As_1\}, pa_1, P_1\rangle$ and $W_2 = \langle\{Cs_2, Is_2, As_2\}, pa_2, P_2\rangle$ be two IRBNs, such that $As_1 \cap As_2 = \emptyset$ (the actions in As_2 which are already in As_1 are renamed). The combination of these two IRBNs is an IRBN, denoted by* $\texttt{comb}(\texttt{W}_1, \texttt{W}_2) = \langle\{Cs, Is, As\}, pa, P_W\rangle$*, where*

- $As = As_1 \cup As_2$*;* $Is = Is_1 \cup Is_2$*;* $Cs = Cs_1 \cup Cs_2$*;*
- $pa(I) = \mathfrak{C}(I)\ \forall I \in Is$*;* $pa(A) = pa_1(A) \cup pa_2(A)$*;*
- $P_W(C) = P_{KB}(C)\ \forall C \in Cs$*;* $P_W(I|pa(I)) = P_{KB}(I|\mathfrak{C}(I))\ \forall I \in Is$*;* $P_W(A|pa(A)) = P_{W_k}(A|pa_k(A))$ *if* $A \in As_k$ *(with* $k = 1, 2$*).*

Note that here it is allowed the possibility that the observed agent follows multiple intentions simultaneously. When some intentions are found irrelevant—e.g. because they are much unlikely[2]—those intentions should be removed from the IRBN. This is enacted by considering them as completely false and employing the *project* operator.

Definition 8 (Remove Intentions from IRBN) *Let $W = \langle\{Cs, Is, As\}, pa, P_W\rangle$ be an IRBN and $R \subset Is$ a strict subset of* Is*. The result of removing the set of intentions R from W is an IRBN, denoted by* $\texttt{remove}(\texttt{W}, \texttt{R}) = \langle\{Cs_R, Is_R, As_R\}, pa_R, P_R\rangle$*, where*

- $As_R = As$*;* $Is_R = Is \setminus R$*;* $Cs_R = \bigcup_{I \in Is_R} \mathfrak{C}(I)$*;*
- $pa_R(I) = \mathfrak{C}(I)\ \forall I \in Is_R$*;* $pa_R(A) = pa(A) \setminus R\ \forall A \in As_R$*;*
- $P_R(C) = P_{KB}(C)\ \forall C \in Cs_R$*;* $P_R(I|pa_R(I)) = P_{KB}(I|\mathfrak{C}(I))\ \forall I \in Is_R$*; and for each* $A \in As_R$*,* $P_R(A|pa_R(A))$ *is defined by the CPD table* $\texttt{proj}(\texttt{Tb}, \texttt{pa}_\texttt{R}(\texttt{A}))$ *where Tb is the CPD table for A in W, i.e. defined by* $P_W(A|pa(A))$*.*

Based on these operators, we now describe an algorithm for incremental intention recognition in a real-time manner.

Incremental Intention Recognition Algorithm. Repeat the following steps until some given time limit is reached; the most likely intention in previous cycle is the final result.

[2]One intention is much less likely than the other if the fraction of its likelihood and that of the most likely intention is less than some small threshold. It is up to the KB designer to provide it.

- Let A be a new observed action. Combine the current IRBN W with $\texttt{irBN(A)}$ to obtain $\texttt{W}' = \texttt{comb}(\texttt{W}, \texttt{irBN(A)})$. If A is the initially observed action, let $\texttt{W}' = \texttt{irBN(A)}$.
- Compute the probability of each intention in W', conditional on the set of current observations in W'. Remove the intentions which are much less likely than the others (following Definition 8).

4 RELATION AMONG INTENTIONS

When considering the case in which the observed agent may pursue multiple intentions simultaneously, it is undoubtedly indispensable to take into account and express the relations amongst the intentions in the model. Pursuing one intention may exclude some other intention to be pursued. We introduce a so-called *exclusive relation e*—a binary relation on the set of intention nodes—representing that if one intention is pursued, then the other intention cannot be pursued. It is usually, although perhaps not always, the case that intentions exclusiveness is symmetric. Here we assume that e is symmetric; it can be renamed *mutually exclusive relation*.

Intentions I_1 and I_2 are mutually exclusive iff they cannot be pursued simultaneously, i.e. $P(I_1 = T, I_2 = T) = 0$. Thus, for any action A, if $I_1, I_2 \in pa(A)$ then the CPD table for A is undefined. Hence, the BN needs to be restructured. The mutually exclusive intentions must be combined into a single node since they cannot co-exist as parents of a node. Each intention represents a possible value of the new combined node. Namely, let $I_1, ..., I_t$ be such that $e(I_i, I_j),\ \forall i, j : 1 \leq i < j \leq t$. The new combined node, I, stands for a random variable whose possible outcomes are either $I_i,\ 1 \leq i \leq t$, or $\tilde{I}$—the outcome corresponding to the state that none of $I_i = T$. Note that if the intentions are exhaustive, $\tilde{I}$ can be omitted. Next, I is linked to all the action nodes that has a link from one of $I_i,\ 1 \leq i \leq t$.

There remains to re-define CPD tables in the new BN. They are kept the same for action A where $I \notin pa(A)$. For A such that $I \in pa(A)$, the new CPD table at $I = I_i$ corresponds to the CPD table in the original BN at $I_i = T$ and $I_j = F\ \forall j \neq i$, i.e. $P(A|I = I_i, ...) = P(A|I_0 = F, ..., I_{i-1} = F, \mathbf{I_i} = \mathbf{T}, I_{i+1} = F, ..., I_t = F, ...)$. Note that the left hand side is defined in the new BN, and the right hand side is defined in the original BN. Similarly, the new CPD table at $I = \tilde{I}$ corresponds to $I_i = F\ \forall 1 \leq i \leq t$. We now specify the CPD table of I. In the new BN, the causes/reasons of each intention are connected to the combined node, i.e. $pa(I) = \bigcup_{i=1}^{t} \mathfrak{C}(I_i)$. Applying the Markov assumption (Def.1) we have $P(I = I_i|pa(I)) = P_i(I_i = T|\mathfrak{C}(I_i))$ and $P(I = \tilde{I}|pa(I)) = \prod_{i=1}^{t} P_i(I_i = F|\mathfrak{C}(I_i))$, where P_i is the probability distribution of the unit fragment for I_i.

In the next section we focus on the single intention recognition case, showing how the approach to representing relationships amongst intentions can significantly decrease the complexity of the probability inference therein. We then present experimental results on the Linux Plan corpus. After that, in Section 6, we provide further experimentation on our novel, so-called IPD plan corpora.

5 SINGLE INTENTION RECOGNITION

5.1 THE MODEL

Suppose the observed agent pursues a single intention. In this case, all intentions are mutually exclusive, and they can be combined into a single node. The IRBN then has a single intention node, linking to all action nodes. All cause/reason nodes are connected to the intention node.

Let $I_1, ..., I_n$ be the intentions in the original IRBN. As usual, they are assumed to be exhaustive, i.e. the observed agent is assigned an intention from them. The combined node I thus has n possible outcomes $I_i,\ 1 \leq i \leq n$. Let $As = \{A_1, ..., A_m\}$ be the set of current observed actions. The set of all cause/reason nodes are $Cs = \cup_{i=1}^{n} \mathfrak{C}(I_i)$. Suppose $C_e \subseteq Cs$ is the set of cause/reason nodes which are observed (evidence nodes). Let $C_{ne} = Cs \setminus C_e$.

Applying Eq. 1, we obtain the probability of each intention conditional on the current observations, for $1 \leq j \leq n$,

$$P(I = I_j|C_e, As) = \frac{P(I_j, C_e, As)}{\sum_{i=1}^{n} P(I_i, C_e, As)}, \quad \text{where}$$

$$P(I_j, C_e, As) = \prod_{i=1}^{m} P(A_i|I_j) \left(\sum_{C_{ne}} P(I_j|Cs) \prod_{C \in Cs} P(C) \right)$$

This implies that, when not including causes/reasons of intentions ($Cs = \emptyset$) as in case of Linux Plan corpus below, our intention recognizer has a linear complexity $O(|n|)$.

If all the cause/reason nodes are not observable, i.e. $C_{ne} = Cs$ (as in the case of the Linux Plan we examine in the next subsection), it is easily seen that: $P(I_j, C_e, As) = P(I_j) \prod_{i=1}^{m} P(A_i|I_j)$. If all of them are observed ($C_{ne} = \emptyset$) (as we shall see in the IPD Plan corpora), the term $\prod_{C \in Cs} P(C)$ is simplified in the fraction. Thus, in these two cases, we do not need to define prior probabilities distribution of the root nodes in Cs. Note that in the latter case we still need to compute the conditional probabilities $P(I_j|Cs)$.

5.2 EXPERIMENTAL EVALUATION

The Linux Plan Corpus. Plan corpus is the term used to describe a set of plan sessions and consists of a list of goals/intentions and the actions a user executed to achieve them [1]. Although there are many corpora available for testing machine learning algorithms in other domains, just a few are available for training and testing plan/intention recognizers; furthermore, each of the recognizers using plan corpora usually has its own datasets, leading to a difficult comparison among them. For that important reason, we chose Linux Plan corpus [2]—one of the rare regularly used plan corpora—which was kindly made publicly available by Nate Blaylock—to test our system. It enables a better comparison with other systems using this corpus [2, 1].

The Linux plan corpus was gathered from 56 human users. The users have different levels of expertise in the use of Linux, and they were allowed to perform as many times as they wished, in order to contribute more plan sessions. The sessions, consisting in sequences of commands performed by the users to achieve a given goal/intention, were automatically recorded. At the end of each session, the users were asked to indicate whether they succeeded in achieving their goal. In total, there are 547 sessions, 457 of which were indicated as successfully completing the goal, 19 goals and 43 actions.

The Linux Plan corpus is an important (especially in the interface-agents domain [12]) and hard benchmark for intention/goal recognition. First, data is collected from real humans and thus noisy. Second, involved humans expertise is varied, and they sometimes used wrong commands due to limited knowledge about the domain [2]. Furthermore, we observe that plan sessions' lengths in the corpus are quite varied. The minimum, maximum, and mean number of actions of a plan session are 1, 60, and 6.124, respectively.

Learning Unit Fragments from Data. For unit fragment $UF_{\mathfrak{A}}(I, A)$, the conditional probability of A given I is defined by the frequency of A in a plan session for achieving the goal/intention I divided by the frequency of any action for achieving I: $P(A = T|I = T) = \frac{freq(A_I)}{freq(I)}$. For better understanding, in the plan corpus each action is marked with the intention which the action is aiming at. Then, $freq(A_I)$ is the frequency of A being marked by I, and $freq(I)$ is the frequency of seeing the mark I.

Prior probabilities of all the intentions in the corpus are given initially, and used for generating tasks for users [2].

Making Predictions. Similar to [2], instead of letting the recognizer make a prediction after each observed action, we set a *confidence* threshold τ $(0 \leq \tau \leq 1)$, which allows the recognizer to decide whether or not it is confident enough to make a prediction; the recognizer only makes a prediction if the likelihood of the most likely intention in the model is greater than τ. Otherwise, it predicts "don't know". In addition, instead of only predicting the most likely intention, the recognizer provides a set of N most likely ones (*N-best prediction*).

Evaluation Metrics. For evaluating our system and comparing with the previous ones [2, 1], we use three different metrics. *Precision* and *recall* report the number of correct predictions divided by total predictions and total prediction opportunities, respectively. More formally (also see [1]), let $Seq = a_1, ..., a_n$ be a sequence of actions (plan session) achieving intention I. Considering N-best prediction case, let $correct(A) = 1$ if I is one of N most likely intentions, and 0 otherwise. Then, precision and recall for *Seq* are defined as: $precision(Seq) = (\sum_{i=1}^{n} correct(a_i))/z$; $recall(Seq) = (\sum_{i=1}^{n} correct(a_i))/Z$, where z and Z are the number of predictions made (when the recognizer is confident enough) and the total number of prediction opportunities (i.e. when $\tau = 0$), respectively.

On the other hand, *convergence* is a metric that indicates how much time the recognizer took to converge on what the current user goal/intention was. Let t be such that $correct_i = 0$ for $0 \leq i \leq t-1$ and 1 for $t \leq i \leq n$ (i.e. t is the first time point which from there on the system always correctly predicts), *convergence* for *Seq* is defined as: $convergence(Seq) = (z - t + 1)/z$.

Finally, the *overall* precision, recall and convergence are obtained by taking averages over all testing sessions.

Table 1: Intention Recognition Results on the Linux Plan Corpus

N-best	1-best	2-best	3-best	4-best
τ	0.95	0.5	0.45	0.42
Precision	0.786	0.847	0.870	0.883
Recall	0.308	0.469	0.518	0.612
Converg.	0.722	0.799	0.822	0.824

Experiments and Results. Because of the small size of the Linux corpus, similar to previous works, we ran experiments using the one-out cross validation method [1].

Table 1 shows the results for different values of N (and the corresponding value of τ). Similar to the previous works [2, 1], we keep the best results for each value of N w.r.t. τ. For example, we obtained a precision of 78.6% for 1-best that is increased to 87.0% for 3-best prediction and 88.3% for 4-best one. Convergence is increased from 72.2% for 1-best to 82.2% for 3-best and 82.4% 4-best prediction.

The best performance on the Linux corpus (namely, in terms of precision and convergence) so far was reported in [1], where the authors use variable Markov model with exponential moving average. Here we got an increment of 14% better precision and 13.3% better convergence for 1-best prediction, 8.2% better precision and 9.3% better convergence for 2-best prediction, and 7.5% better precision and 7.7% better convergence for 3-best prediction. We also obtained better recalls comparing with [2] in all cases.

The Linux corpus allows an appropriate comparison with existent works. However, it does not include contextual information (reasons/causes of intentions), and there is no intention change/abandonment occurrences (users follow a single intention throughout entire plan sessions). To evaluate the context-dependent aspect as well as the capability of dealing with intention change/abandonment, we next present new plan corpora.

6 IPD PLAN CORPORA

We present new plan corpora in the context of iterated Prisoner's Dilemma (IPD) [21] and provide experimental results for them. The intentions/goals to be recognized are the (known) strategies in IPD (see below). Plan sessions are sequences of moves played by such strategies.

6.1 ITERATED PRISONER'S DILEMMA

Prisoner's Dilemma (PD) is a symmetric two-player non-zero game defined by the payoff matrix

$$\begin{array}{c c} & \begin{array}{cc} C & D \end{array} \\ \begin{array}{c} C \\ D \end{array} & \begin{pmatrix} R, R & S, T \\ T, S & P, P \end{pmatrix} \end{array}$$

Each player has two options in each round, cooperates (C) or defects (D). A player who chooses to cooperate with someone who defects receives the sucker's payoff S, whereas the defecting player gains the temptation to defect, T. Mutual cooperation (resp., defection) yields the reward R (resp., punishment P) for both players. PD is characterized by the payoff ranking $T > R > P > S$ (and, in addition, $2R > S + T$ for IPD). Thus, in a single round, it is always best to defect, but cooperation may be rewarded if the game is iterated. Let r denote the (average) number of rounds the game is iterated.

IPD is usually known as a story of tit-for-tat (TFT), which won both Axelrod's tournaments [21]. *TFT* starts by cooperating, and does whatever the opponent did in the previous round. It will cooperate if the opponent cooperated, and will defect if the opponent defected. But if there are erroneous moves due to noise (i.e. an intended move is wrongly performed with a given execution error), the performance of *TFT* declines: it cannot correct errors or mistakes. Tit-for-tat is then replaced by generous tit-for-tat (GTFT), a strategy that cooperates if the opponent cooperated in the previous round, but sometimes cooperates even if the opponent defected (with a fixed "forgiveness" probability $p > 0$) [21]. *GTFT* can correct mistakes. Subsequently, *TFT* and *GTFT* were replaced by win-stay-lose-shift (WSLS) as the winning strategy chosen by evolution [21]. *WSLS* repeats the previous move whenever it did well, but changes otherwise. Some other less famous strategies (which we are going to use later) are *GRIM* – a grim version of *TFT*, prescribing to defect except after a round of mutual cooperation, and Firm-But-Fair (FBF) – known as a tolerant brother of *TFT*, prescribing to defect only if getting a sucker's payoff S in previous round. Details of all strategies described above can be found in [21] (Chapter 3).

Next, we describe how training and testing plan corpora are created employing these strategies.

6.2 CORPUS DESCRIPTION

We made an assumption that all strategies to be recognized have the memory size bounded-up by M ($M \geq 0$)—i.e. their decision at the current round is independent of the past rounds that are at a time distance greater than M. The strategies described above have memory $M = 1$. Abusing notations, R, S, T and P are referred to as game states (in a single round or interaction). We too use E (standing for *empty*) to refer to the game state having had no interaction.

An action in the corpus is of the form $s_1 ... s_M \xi$, where $s_i \in \{E, R, T, S, P\}$, $1 \leq i \leq M$, are the states of the M last interactions, and $\xi \in \{C, D\}$ is the current move. We denote by Σ_M the set of all possible types of action. E.g, $\Sigma_1 = \{EC, RC, TC, SC, PC, ED, RD, TD, SD, PD\}$. This encoding method enables to save the game states without having to save the co-player's moves, thus simplifying the corpus representation, described below.

Suppose we have a set of strategies to be recognized. The plan corpus for this set consists of a set of plan sessions generated for each strategy in the set. A plan session of a strategy is a sequence of actions played by that strategy (more precisely, a player using that strategy) against an arbitrary player. As an example, let us consider *TFT* and the following sequence of its interactions with some other player (denoted by X), in the presence of noise

round :	0	1	2	3	4	5
TFT :	–	C	C	D	D	D
X :	–	C	D	D	C	D
TFT-states :	E	R	S	P	T	P

The corresponding plan session for *TFT* is $[EC, RC, SD, PD, TD]$. At 0-th round, there is no interaction, thus the state is E. *TFT* starts by cooperating (1-st round), hence the first action of the plan session is EC. Since player X also cooperates in the 1-st round, the game state at this round is R. *TFT* reciprocates in the 2-nd round by cooperating, hence the second action of the plan session is RC. Similarly for the third and the fourth actions. Now, at the 5-th round, *TFT* should cooperate since X cooperated in 4-th round, but because of noise, it makes an error to defect. Therefore, the 5-th action is TD.

6.3 PLAN CORPORA GENERATION

Let us start by generating a plan corpus for seven most popular strategies within the IPD framework: *AllC* (always cooperate), $AllD$ (always defect), *TFT*, *GTFT* (probability of forgiving a defect is $p = 0.5$), *WSLS*, *GRIM* and *FBF*.

We collect plan sessions of each strategy by playing a random move (C or D) in each round with it. To be more thorough, we can also play all possible combinations for each given number of rounds r. E.g, if $r = 5$, there are 2^5 combinations: C or D in each round. When noise is present, each combination is played repeatedly several times.

The training corpus to be used here is generated by playing with each strategy all the possible combinations 10 times, for each number of rounds r from 5 to 10. The testing dataset is generated by playing a random move with each strategy in each round, also for r from 5 to 10. We continue until obtaining the same number of plan sessions as of the training dataset (corpus). Both datasets are generated in the presence of noise (namely, an intended move is wrongly performed with probability 0.05).

In this testing dataset, intention (strategy) changes/abandonment are not taken into account. The players use the same strategy in all the rounds. We refer to this testing dataset as `Testset-IRFIX`. For testing the context-dependent aspect of our intention recognizer, as well as taking into account intention changes/abandonment, we next introduce the concept of *social learning* within the framework of evolutionary game theory [11].

6.4 SOCIAL LEARNING

In social learning, individuals in a population can observe the behavior of others and the outcomes of those behaviors. They copy the behavior of others whenever these appear to be more successful [21]. The accumulated payoff from all interactions emulates the individual *fitness* or social *success* and the most successful individuals will tend to be imitated by others. There are many ways to model social learning [11, 21]. The most popular one is implemented using the so-called pairwise comparison rule [21]: an individual **A** with fitness f_A will adopt the strategy of a randomly chosen individual **B** with fitness f_B with a probability given by the Fermi function (from statistical physics): $p(f_A, f_B) = \left(1 + e^{-\beta[f_B - f_A]}\right)^{-1}$, where the quantity β controls the "imitation strength", i.e. how strongly the players are basing the decision to imitate on payoff comparisons. Henceforth, **A** and **B** are referred to as imitating and imitated individuals, respectively. For simplicity, we use $\beta = 1$ for the rest of this paper: the imitation depends on comparing the exact payoffs.

It is now allowed the possibility that a player can change his/her strategy (intention) by imitating the randomly met player's strategy (intention), depending on how the latter player is more successful. The two players' ongoing success difference (SD) causally affects the imitating player's current intention. In addition, this intention is causally affected by the so-called imitation event (IE), stating whether the player is meeting some other player for learning/imitating. Now we have an IRBN with two cause/reason nodes, a single intention node, and observed action nodes (Figure 3).

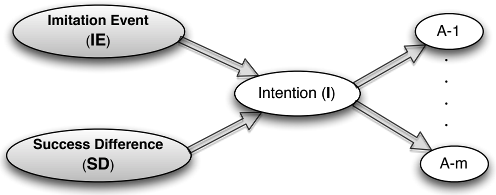


Figure 3: IRBN in IPD Context

We define the conditional probability distribution $P(I_i|IE, SD)$. If the player does not meet any other player for imitation (i.e. $IE = F$), I_i is independent of the success difference *SD*: $P(I_i|IE = F, SD) = P(I_i|IE = F)$. Now, let us consider the case $IE = T$. If the successes are also observable (thus, *SD* is observed, say, equal χ)[3], but the strategy of the imitated player is not, we have

$$P(I_i|IE = T, SD = \chi) = (1-u)p_i + \frac{u}{S-1}\sum_{j\neq i} p_j \quad (2)$$

where $u = (1+e^{-\chi})^{-1}$; p_i is the probability that I_i was the player's intention in the last prediction; and S is the number of strategies in the corpus. The formula is explained as follows. With probability $(1-u)p_i$ the imitating player's strategy remains I_i. Moreover, not being observed, the probability that I_i was the imitated player's strategy is (assumed) equal $1/(S-1)$. The second term expresses the probability that the player adopts the new strategy I_i by imitation.

Now, in case the imitated player's strategy is also observable, denoted by $I_{i^\star}$, similarly we have

$$\begin{aligned} P(I_{i^\star}|IE = T, SD = \chi) &= (1-u)p_i + u\sum_{j\neq i^\star} p_j \\ P(I_i|IE = T, SD = \chi) &= (1-u)p_i \quad \forall\, i \neq i^\star \end{aligned} \quad (3)$$

Testing Dataset. The testing dataset in this setting is generated by using a simplified evolutionary simulation as follows. We play a random choice with each of the seven above mentioned strategies for 10 rounds. The payoff of each strategy is accumulated over all the rounds. Then, for each strategy, another strategy is randomly chosen from the other six for imitation using the pairwise comparison rule. After all the seven strategies are given the chance to change their strategy (imitate another), the interactions are repeated for 10 more rounds. At the 10-th round, we save the accumulated payoff values of the imitating and imitated strategies. We experiment until obtaining the same number of plan sessions as in the training dataset. The PD payoff

[3]There may be noise in the evaluation of the successes. The observed value χ of SD is randomly taken in the range $((1-\epsilon)\chi_1, (1+\epsilon)\chi_1)$, where ϵ is a small positive number (here we use $\epsilon = 0.01$) and χ_1 is the exact value of the difference.

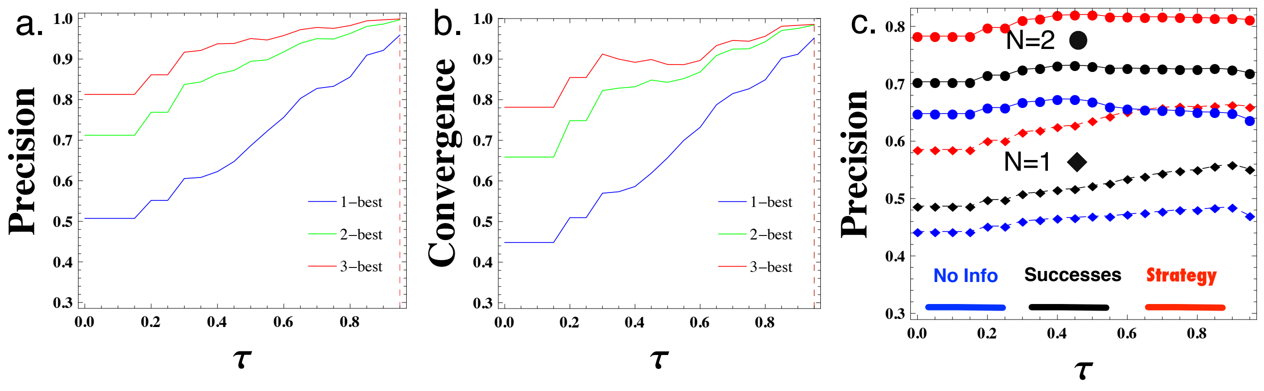


Figure 4: Panels (a) and (b): Precision and convergence for $\tau \in [0,1]$ and for different values of N (N = 1,2,3) with respect to `Testset-IRFIX` dataset. Panel (c): Precision for different levels of contextual information, for $\tau \in [0,1]$, with respect to `Testset-IRCHANGE` dataset. We consider $N = 1$ (dashed diamond) and $N = 2$ (circle).

matrix being used: $T = 20$, $R = 15$, $P = 10$, $S = 5$; and $noise = 0.05$. This testing dataset is referred to as `Testset-IRCHANGE`.

6.5 RESULTS

The intention recognition model is acquired using the training corpus. Figures 4a and 4b show the precision and convergence of the model with respect to the `Testset-IRFIX`. Given that the training as well as the testing datasets are generated in presence of noise, the achieved performance is quite good. Namely, for big enough τ, both precision and convergence scores are greater than 0.9, even for the 1-best case.

In Figure 4c we show the effects of having different levels of contextual information on the intention recognition performance, using `Testset-IRCHANGE` dataset. Namely, in the first setting (blue curves), there is no information about the imitation event (IE) – it is not known if the recognized player may imitate and adopt another strategy. In the second setting (black curves), IE and the successes are observable. In the third setting (red curves), the strategy of the imitated player is also observable. It is clearly shown that the performance is considerably increased as more contextual information is available. Namely, comparing with the first setting where no contextual information is taken into account, an increase of about 5% and 15% precision is achieved in the second and third settings, respectively.

7 RELATED WORK

Bayesian Networks have been one of the most successful models applied for the intention/plan recognition problem, e.g. in [4, 6]. Depending on the structure of plan libraries, a knowledge-based model construction is employed to build BNs from the library—which is then used to infer the posterior probability of explanations (for the set of observed actions). These works address a number of important issues in intention/plan recognition (see [6] for details), but they made several assumptions for the sake of computational efficiency. First, prior probabilities of intentions are assumed to be fixed. This assumption is not reasonable because those prior probabilities should depend on the situation at hand [3], and can be captured by causes/reasons of the intentions as in our work. Second, intentions are assumed to be independent of each other. This is not generally the case since the intentions may support or exclude one another. Those works hence do not appropriately address multiple intention recognition. This latter assumption must always, explicitly or implicitly, be made by the approaches based on (Hidden) Markov Models, e.g. [1], or statistical corpus-based machine learning [2]. Generally, in those approaches, a separate model is built for each intention; thus no relations amongst the intentions are expressed or can be expressed. These works were restricted to the single intention case.

Different from all above mentioned works, our model is *context-dependent*, which is achieved by including in it causes/reasons of intentions. This way, our model can appropriately deal with the abandonment/changes of intentions—when the causes/reasons do not support or force the intending agent to hold those intentions anymore.

8 CONCLUDING REMARKS AND FUTURE WORKS

We have presented a novel method for incremental and context-dependent intention recognition. The method is performed by dynamically constructing a BN model for intention recognition from a prior knowledge base consisting of easily maintained fragments of BN. We have evaluated the method on the Linux Plan corpus and compared with previous works. In general, our performance is better than all existent ones that make use of the corpus.

For further experimentation, we have created the so-called IPD plan corpora for the famous strategies in the context of the iterated Prisoner's Dilemma. We employed the famous model of (human) behaviors by means of social learning and evolutionary game theory to simulate intention changes/abandonment—enabling us to evaluate the context-dependent aspect of our intention recognizer and as well as its capability of dealing with intention changes/abandonment. Our experimental results show that taking into account contextual information is crucial, enabling to achieve significant recognition improvements.

The good performance of our method with respect to the Linux corpus shows its applicability to the important interface-agents domain [12]. In addition, given that PD and other social dilemmas [21] are regularly found in real life [11, 21], its good performance for the IPD corpora makes it highly applicable for a wide range of application domains, as diverse as Economics (e.g. recognizing companies policies), Psychology and Biology (e.g. the role of intention recognition in the evolution of cooperation, as our recent works exhibit in [8, 9], using the intention recognition methods described in this paper).

In Section 4 we made an implicit assumption that the intentions to be combined are perfectly mutually exclusive. This assumption can be relaxed by utilizing a latent variable for any subset of perfectly mutually exclusive intention nodes. We are exploring this direction to provide a more general method for representing relationships amongst intention nodes.

9 Acknowledgments

We thank the reviewers for useful comments. HTA acknowledges the support from FCT-Portugal, grant SFRH/BD/62373/2009.

References

[1] M.G. Armentano and A. Amandi. Goal recognition with variable-order markov models. In *IJCAI'09*.

[2] N. Blaylock and J. Allen. Statistical goal parameter recognition. In *ICAPS04*, pages 297–304.

[3] M. E. Bratman. *Intention, Plans, and Practical Reason*. The David Hume Series, CSLI, 1987.

[4] E. Charniak and R.P. Goldman. A Bayesian model of plan recognition. *Artificial Intelligence*, 64(1), 1993.

[5] F. G. Cozman. Axiomatizing noisy-or. In *ECAI'04*.

[6] C. W. Geib and R. P. Goldman. A probabilistic plan recognition algorithm based on plan tree grammars. *Artificial Intelligence*, 173:1101–1132, 2009.

[7] V. Gogate and R. Dechter. SampleSearch: Importance sampling in presence of determinism. *Artificial Intelligence*, 175(2):694–729, 2011.

[8] T. A. Han, L. M. Pereira, and F. C. Santos. The role of intention recognition in the evolution of cooperative behavior. In *IJCAI'2011*.

[9] T. A. Han, L. M. Pereira, and F. C. Santos. Intention recognition promotes the emergence of cooperation. *Adaptive Behavior*, 2011.

[10] C. Heinze. *Modeling Intention Recognition for Intelligent Agent Systems*. PhD thesis, The University of Melbourne, Australia, 2003.

[11] J. Hofbauer and K. Sigmund. *Evolutionary Games and Population Dynamics*. Cambridge U. P., 1998.

[12] E. Horvitz, J. Breese, D. Heckerman, D. Hovel, and K. Rommelse. The lumière project: Bayesian user modeling for inferring the goals and needs of software users. In *UAI'98*, pages 256–265, 1998.

[13] K. B. Laskey. MEBN: A language for first-order Bayesian knowledge bases. *Artificial Intelligence*, 172(2-3):140 – 178, 2008.

[14] K.B. Laskey and S.M. Mahoney. Network fragments: Representing knowledge for constructing probabilistic models. In *UAI'97*, 1997.

[15] S. Natarajan, P. Tadepalli, T. G. Dietterich, and A. Fern. Learning first-order probabilistic models with combining rules. *Annals of Mathematics and Artificial Intelligence*, 54:223–256, 2008.

[16] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufman, 1988.

[17] J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge U.P, 2000.

[18] L. M. Pereira and T. A. Han. Intention recognition with evolution prospection and causal Bayesian networks. In *Computational Intelligence for Engineering Systems*, pages 1–33. Springer, 2011.

[19] A. Pfeffer, D. Koller, B. Milch, and Ken T. Takusagawa. Pook: A system for probabilistic object-oriented knowledge representation. In *UAI'99*, 1999.

[20] F. Sadri. Logic-based approaches to intention recognition. In *Handbook of Research on Ambient Intelligence: Trends and Perspectives*. 2010.

[21] Karl Sigmund. *The Calculus of Selfishness*. Princeton U. Press, 2010.

[22] S. Srinivas. A generalization of the noisy-or model. In *UAI'93*, 1993.

Using Bayesian Belief Networks for Modeling of Communication Service Provider Businesses

Pekka Kekolahti
Department of Communications and Networking
Aalto University
Espoo, Finland

Abstract

This paper analyses the usage of Bayesian Belief Networks (BBNs) for Communication Service Provider (CSP) business modeling and simulation. Large and complex BBNs have been created to describe the causal relationships in CSP business domains. As a part of the study, a novel method to collect knowledge from a large number of independent experts living in different countries has been introduced. A BBN from each expert result was created (referred to here as a sub-BBN). Business model ontology was utilized to combine sub-BBNs together into a comprehensive model. The resulting BBN represents typical business circumstances in the European telecommunications domain. The experts participating in the study represented expertise in different business related categories such as technology, processes, customer experience, regulation, organization and products. Experts were asked to list causality triplets for business categories including causal connection strengths, in order to assess the belief part as well. The triplets were manually converted to a graphical causal map and conditional probability tables constructed. The benefit of the method is the capability to introduce rapidly a high number of variables and causal relationships. A challenge is that experts use different terms with the same underlying meaning.

1. INTRODUCTION

The Communication Service Provider (CSP) business is facing major restructuring due to several disruptive factors. These include new business players like Google and Facebook, technologies like the Internet, cloud computing and smart-phones, as well as a growing number and size of applications. It is clear that the CSP value chain structure has to be re-evaluated. To respond to these changes and customer requirements, and to adapt successfully to new business challenges, CSP top management needs reliable methods to model and to analyze the essential factors driving the change, and to understand the impact of these factors on their current and future business. In addition, trusted and unified information is needed for strategy planning processes and day-to-day management. Today the strategic decisions are often made by a small group and they are based on insufficient knowledge due to lack of data or expert knowledge and under time constraints.

Bayesian Belief Networks (BBNs), also called belief networks, Bayes Nets and causal probabilistic networks are increasingly popular methods for modeling uncertain and complex domains (Uusitalo, 2007). In this paper we examine how the BBN methodology can be utilized to help CSP management in their day-to-day work, strategy planning and to better control the business.

A BBN is a probabilistic model which represents a set of random variables and their conditional dependencies via a directed acyclic graph. Two basic approaches are used to construct Bayes networks: data-based and knowledge-based approaches. Data-based methods use conditional independence semantics of Bayes networks to infer models from data whereas the knowledge-based approach utilizes causal knowledge from domain experts to construct BBNs. The benefits of BBNs in data analysis are, according to Nadkarni, 2004; Uusitalo, 2007; Jensen, 2001; Lee, 2009:

1) Possibility to combine prior knowledge and data,
2) Managing situations where some data is missing,

3) Modeling of causal relationships,
4) Structural learning possibilities,
5) Support for different kind of analyses, such as making inferences about probabilities of different causes given the consequences and
6) Fast response to queries from the model.

Known challenges in BBNs are

1) Difficulty to obtain prior knowledge in a form that can be converted into probability distributions. However, for example a weighted sum algorithm utilizing compatible parent configurations has been developed to ease the calculation of conditional probability tables in complex environments (Das 2004).
2) Handling of continuous variables only in a limited manner (Uusitalo 2007) and
3) Lack of support of feedback loops due to acyclic nature of a BBN. Feedback loops are useful when analyzing phenomena like new disruptive CSP technologies as a function of time (Casey et al. 2010).

According to our knowledge, BBNs have in the past not been used to model the CSP industry in a large scale. The utilization of causality itself is wide spread in business management due to widely used performance measuring and management tools such as the Balanced Scorecard (BSC) and Tableau de Board. 66% of enterprises used BSC in 2007 (Rigby, 2007). Both the BSC and the Tableau de Board rely on causal assumptions (Kasperskaya, 2006). Causal mapping tools like fishbone diagrams, cause-and effect diagrams, impact wheels, issue trees, strategy maps, and risk-assessment mapping are tools to help managers to understand and improve complex systems in the areas of quality, strategy, and information systems. (Scavarda et al., 2006). The causalities in the performance measuring and strategy creation have been normally deduced by using human interaction techniques such as brainstorming or interviews. These methods rely on person-to-person or group interaction in eliciting the knowledge and are fraught with biases associated with inter-person dynamics. Methods to elicit a non-biased knowledge in large scale have been developed (Nadkarni et al., 2004; Scavarda et al., 2006). Scavarda introduces a formal Collective Causal Mapping Methodology (CCMM), which collects information asynchronously from an expert group which is dispersed and diverse. Person-to-person interaction possibility is eliminated and a large amount of experts can be utilized in a controlled way. Nadkarni introduced a procedure for constructing BBNs from domain knowledge experts, where through four steps of a text analysis process the first round interview results can be converted into causal relationships. Once the causal map is available, the states of the variables can be defined and validated with experts through subsequent interviews and finally the probability assessment done either manually or by using noisy-OR method or weighted sum algorithm utilizing compatible parent configurations (DAS, 2004) to reduce the number of probability assessments.

This study focuses on BBNs as a methodology for modeling and analysis of CSP business. As part of the study, both multiple sub-BBNs (one per expert) and a comprehensive CSP BBN combining sub-BBNs have been created. The experts were asked to list and categorize the variables they considered to have an effect on CSP business and also how strong this effect would be. The used seven categories are the same as in typical Balanced Scorecards and business models (Kasperskaya, 2006; Osterwalder, 2002 and 2005; Faber, 2003) namely financial variables, customer-related variables, product and service innovations, staff and internal processes, technology and architecture, strategy and competition, local and global economy and legislation.

The following types of information can be derived from the comprehensive model and sub models:

- Financial variables: Effect of variables like customer experience on revenue, OPEX (operating expense) and CAPEX (capital expenditure).
- Customers: The causes and consequences related to customer satisfaction.
- R&D organization: How do organization agility, managerial structures, salary and incentives affect on efficiency, productivity, OPEX and customer experience.
- Technologies: How do new technologies like rapid growth of smart-phones affect on CAPEX, revenue and data traffic.

BBNs that include all the seven categories are very complex. The number of variables and arcs, and especially the size of conditional probability tables play great effect on the practical usability of the BBN for CSB business analysis purposes. Optimization between practical usability and model granularity and accuracy is examined through creating the comprehensive BBN from sub-BBNs.

The remainder of this article is structured as follows:

Chapter 2 introduces a novel method for the collection of the expert knowledge, and describes how the expert knowledge is converted into BBNs.

Chapter 3 describes the constructed sub-BBNs and comprehensive BBN and elaborates on key variables and their analysis states. Also some result examples are given.

Chapter 4 discusses challenges in eliciting and conversion of prior knowledge into BBN and how well these models truly represent different aspects of CSP businesses. Also future research topics for this line of study are identified.

2. METHODS

2.1 THE KNOWLEDGE COLLECTION METHOD

Five targets were set for the developed method: 1) to combine different expert knowledge from various business categories with the help of a broad expert team and 2) give the experts freedom to focus on those causalities they feel, by their expertise, to be important in order to make sure that new innovative cause-consequence –relationships would arise, 3) to discover as much as possible variable candidates from CSP business domains, 4) to ensure that the experts acted as individuals and no group –thinking possibility existed and 5) to facilitate also disruptive proposals. Thus a pre-defined variable list was not introduced but instead experts had freedom to also name the variables. The financial category was seen more a deterministic than a probabilistic cause- consequences structure and thus it was decided that only a few experts need to be dedicated to financial topics.

An email was sent to 100 expert candidates working in 12, mostly European countries, for CSPs, universities, CSP infrastructure vendors and software and consulting companies which offer services to CSPs. The email included extensive background information about the study targets, introductions of BBN and causality, example variables and an excel template based on the seven CSP business categories. With the help of the template, experts were asked to list variables they considered to have effect on CSP businesses and to categorize the variables to the correct category. Basically experts were asked to list causality triplets of “variable X has some cause on variable Y, which has some effect on variable Z”, see *table 1*. It was supposed, that with this method, an expert can easily just start to write the triplets without need to first have a big picture in mind. In addition, experts were asked to estimate the strength of effect by using numbers:

Strong effect=3,
Moderate effect = 2,
Weak effect = 1

These values were used for measuring the expert’s degree-of-belief value for causal connections. The plan was to use a simplistic method, where both weight and belief parts originate from this strength of effect.

Triplets are in fact mini causal maps (see *Figure 1*) and constructing of one full BBN required combining these triplets together. This was done with a BBN tool called BayesiaLab (www.bayesia.com) by hand. The plan was to review the achieved model with each expert.

Table 1: Part of given example triplets.

Causing - variable(s)	List of variables	Effected variable(s)
Number of staff 2	Marketing effort	2 Market share, 1 aver. service usage, 2 OPEX
Network equip. need 1, current network equipment capability 1	Number of staff	3 OPEX, 2 marketing effort

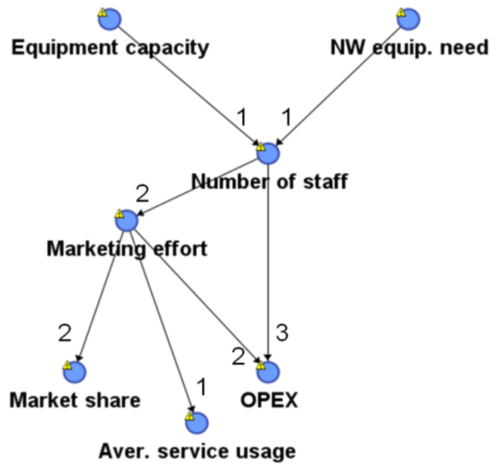


Figure 1: A causal map of two triplets from Table 1 including strengths.

2.2 SUCCESS OF KNOWLEDGE COLLECTION

Out of 100 expert candidates, 48 answered with survey results. The resulting causal models were reviewed with 60% of these 48 experts. The distribution of expertise was:

- Product and service innovations 21%
- Technology and architecture 20%
- Staff and internal processes 20%
- Strategy and competition 19%
- Customers-related 11%
- Local & global economy and legislation 5%
- Financial 4%

Experts used between 1 and 5 hours for the survey, with the average being 2,5 hours. More than 2200 variables and 3400 arcs and 40 sub- BBNs were created from the survey results. Text analysis (www.textanalyser.net) was used in order to understand word frequencies used in variable names. Out from about 5000 used words, 40% were unique. The top 12 used words for variable names were “product and service” 80 times, “customers” 60 times, “costs” 56 times, “market” 50 times, “product” 36 times, “brand 28” times, “new” 22 times, “revenue” 20 times, “price”, “marketing”, “personnel”, “network” 16 times.

From the text analysis it was clear that:

- The process to create a comprehensive BBN is challenging because of the high number of different variable names that have closely the same meaning. The plan was to give full freedom to experts in order to make sure that there were innovative approaches, but this study demonstrated clearly the need of business dictionary if Bayes Belief Networks are to be widely used in CSP business modeling and simulation.
- The competition for customers and tight cost control in European CSP markets might explain the top 12 used words, as the majority of experts were from European countries.

2.3 CONSTRUCTION OF A SUB-BBN

It was quickly concluded that the creation of a comprehensive CSP business model directly from triplets was a too complicated task. It was decided that individual BBNs, called sub-BBNs would be first created. One sub-BNN was created per expert and then the comprehensive BBN was merged from these sub-BBNs. This approach has two benefits: 1) it filters out excess of variables with the same meaning in the sub-BBN review –process with the expert and 2) innovative sub-BBNs will be documented individually.

The creation of a sub-BBN is straightforward: Variables and their causal connection were created manually from triplets by using BayesiaLab-tool (www.bayesia.com). A model review was organized whenever possible with the expert including the states. Each variable has typically only two states which describe best the variable in question like true/false, big/small, high/low, positive/negative, fast/slow.

2.4 PROBABILITY CALCULATIONS FOR A SUB-BBN

The conditional probability tables were calculated with weighted sum –algorithm utilizing compatible parent configurations defined by Das (Das, 2004). This algorithm allows for simplification of the calculation through the utilization of compatible parent configurations for the evaluations performed by the expert, limiting the need of individual probability state combinations needed to be evaluated.

For this study a simplistic method was used in calculation: The weights 3, 2, 1, -1, -2, -3 were used as relative weights and the same weight as probability after

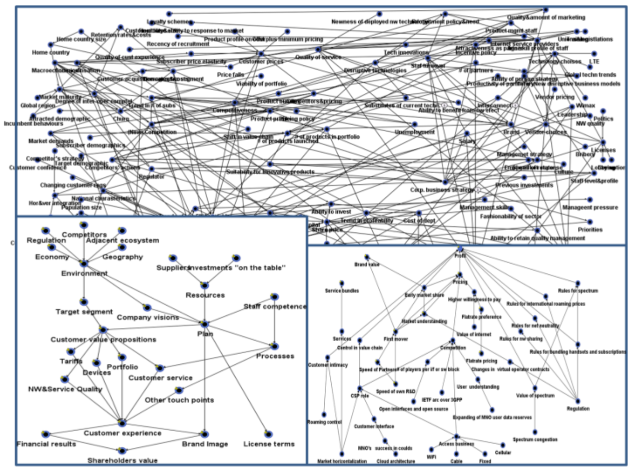


Figure 2: Sub-BBNs derived from expert surveys covered in 70% of cases all seven categories of CSP businesses but granularity varied greatly depending on expert.

converted them in the following way: 3=> 90%, 2=> 75%, 1=> 60%, -3=>10%, -2=>25% and 1=>40%.

In further studies, when the model(s) will be tested in CSP environment, the dual review method with experts will be used, namely first a causal model review with states alone, and after it second review with weights and confidence values.

2.5 CONSTRUCTING OF COMPREHENSIVE BBN

The 40 sub-BBNs varied in granularity and coverage (*Figure 2*) because experts were not asked to focus solely on their own expertise topic. Merging the sub-BBNs to a comprehensive BBN became challenging without a standard “kernel”. The Osterwalder business model ontology (Osterwalder, 2002) is used as a standardized causal kernel (*Figure 3*) to which sub-BBNs was merged.

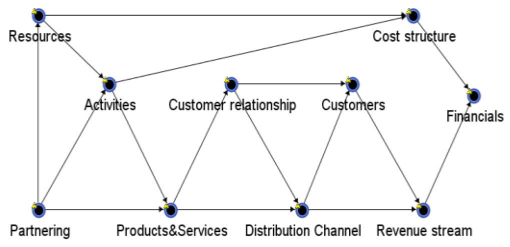


Figure 3: Osterwalder business model blocks, which are used as “a kernel” for comprehensive BBN.

The comprehensive BBN can be seen as an onion-like structure, where the kernel is from the business model ontology and surrounding layers represent experts’ sub-BBNs (*Figure 4*).

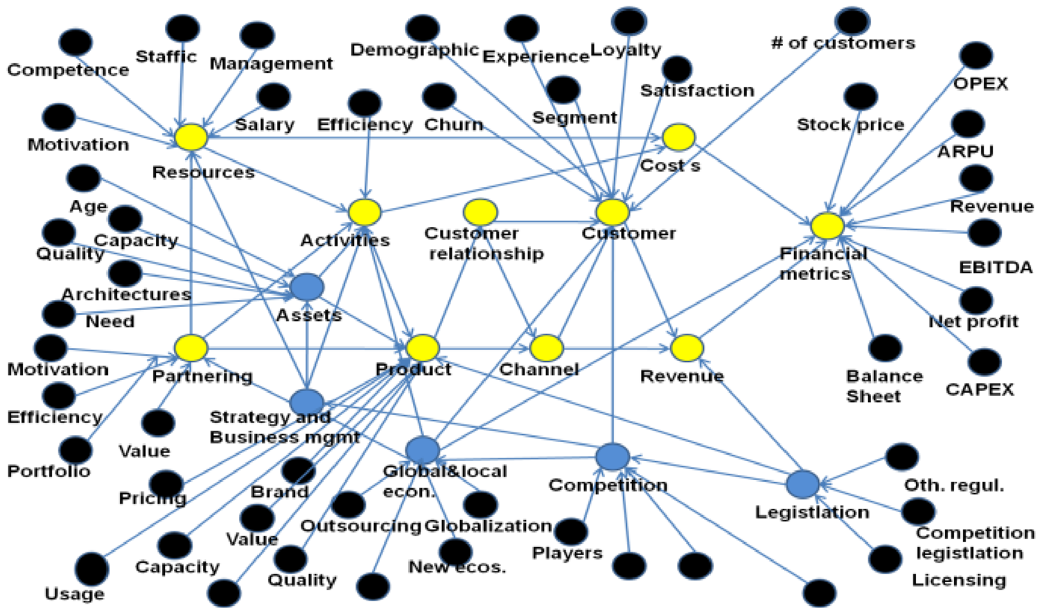


Figure 4: The comprehensive BBN constructs onion type of layers (black and blue circles) around the kernel model (yellow circles)

Comprehensive BBN with different granularities (number of variables and arcs) were created to test the tool and computer environment constraints. When the number of variables exceeds 100, and at the same time the relationship between number of arcs divided by number of variables is on the average greater than three and if a few of variables have five to ten common effects, the practical utilization of the comprehensive BBN for different kind analysis decreases due to slowness of the PC-environment. The objective of this study is not to focus on the tool usability nor model complexity topics but to discover a Bayes Belief Network which can be utilized in practice, contains all the seven business categories and which reflects the expert's common view about CSP variables effecting on business.

The merge process was performed manually, with variables and arcs being combined from each sub-BBN to the comprehensive network around it's kernel. If certain variable and causal connection existed in many sub-BBNs, the weights (used in sub-BBNs) were summed together. Thus, if 10 sub-BBNs have a variable "customer satisfaction" affecting with weight 3 "customer loyalty", then the combined weight is 30. The conditional probability tables have been calculated with the same method as described in sub-BBN-case. However, a dual review method is planned to be used when the model will be tested in real life.

3. RESULTS

This chapter presents both sub-BBNs, created based on individual expert's survey and the comprehensive BBN, merged from individual BBNs. Chapter 3.1 gives three examples of innovative sub-BBNs, which can be used, not only as an input to the comprehensive BBN but also independently. Chapter 3.2 presents results on the comprehensive BBN.

3.1 SUB-BBN EXAMPLES

Example 1: A generic purpose financial causal map with 32 variables and their relationships (*Figure 5*). Many of the variables and causalities are more deterministic than probabilistic and values are results of mathematical equations like calculation of EBITDA (Earnings before Interest, Taxes, Depreciation and Amortization). This map can be used to analyze the effect of non financial variables analyzed in other sub-BBNs connected to a comprehensive set of financial variables in this model.

Example 2: The variable "new business opportunities" is a parent variable for many new business opportunities for CSPs in a electric-car ecosystem (*Figure 6*), The business opportunities vary from traditional bit-pipe services to content service opportunities. The model contains variables such as the effect of regulator actions, environmental circumstances, renewal energy portion, new technology, price of electricity, price of a electric car, number of electric cars and emergence of new business opportunities. The model offers ways to analyze the effect of different ecosystem variables on potential new services. The states and probabilities of key variables in the model are shown *in Figure 6*.

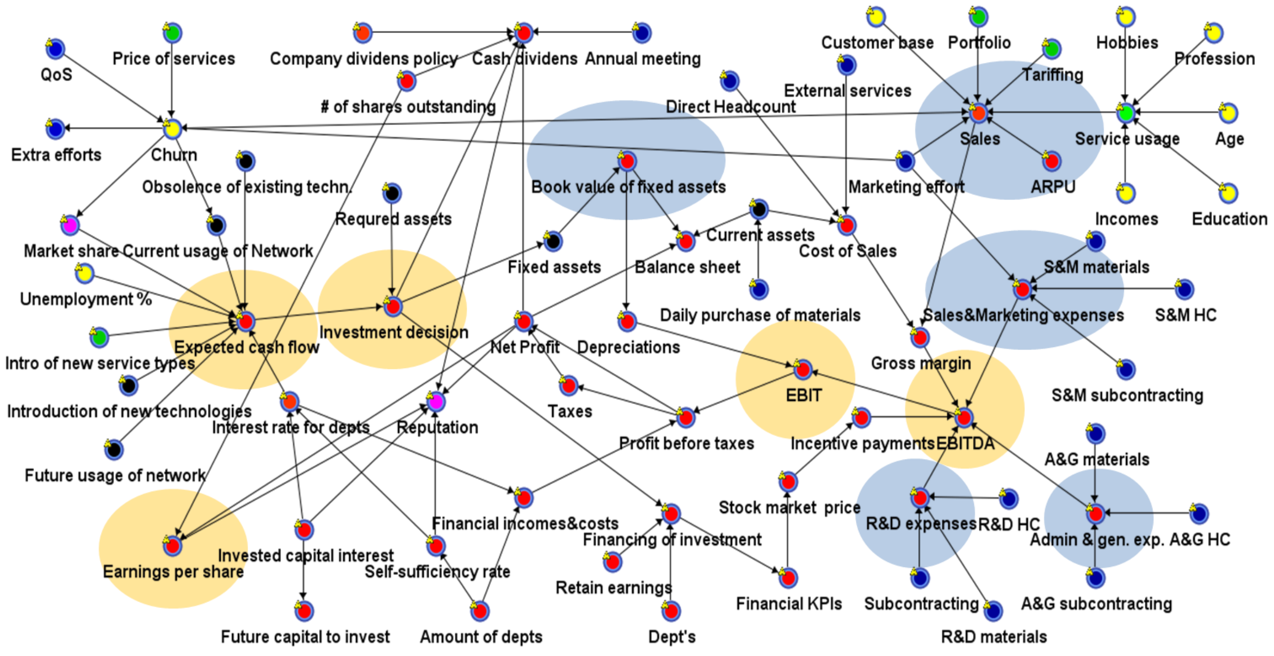


Figure 5: A generic purpose financial –related causal map. The red circles represent financial, yellow customers, blue staff and processes, greed product, black technical and pink competition-strategy –related variables. This model in its many parts is deterministic in nature and don't contain probabilities in this study. Sub-BBN's end up often to Sales, ARPU, R&D etc (blue disks) and with this map the financial analysis towards EBITDA , Earnings per share, expected cash flow and investment decisions (orange disks) can be extended.

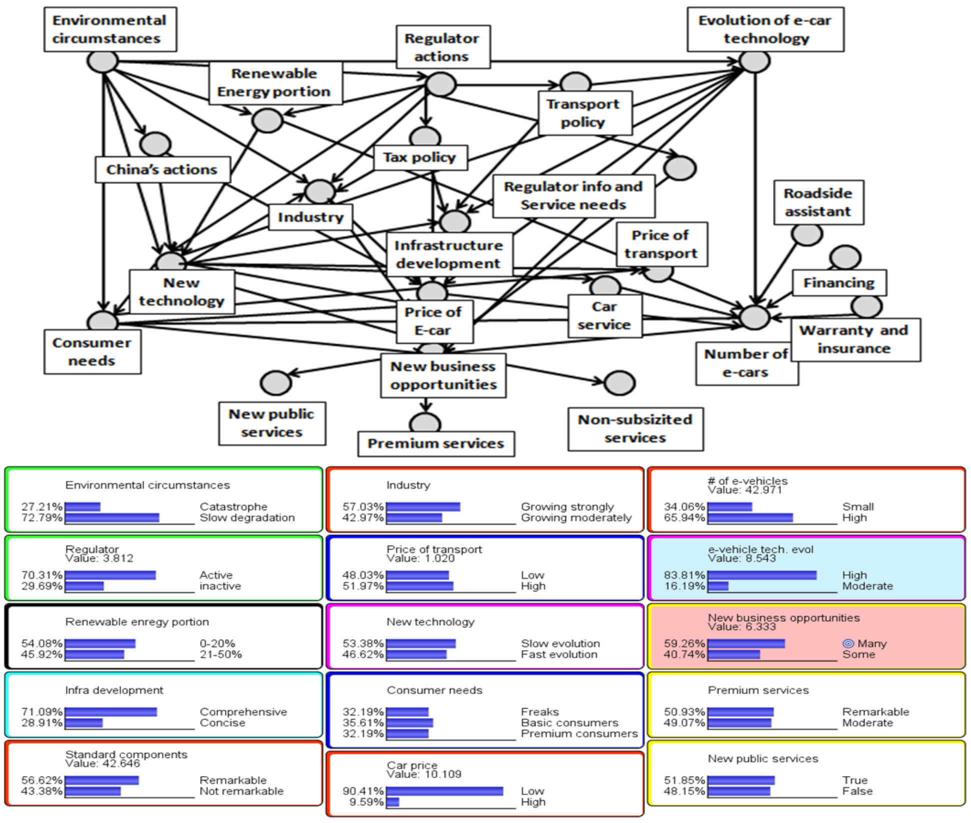
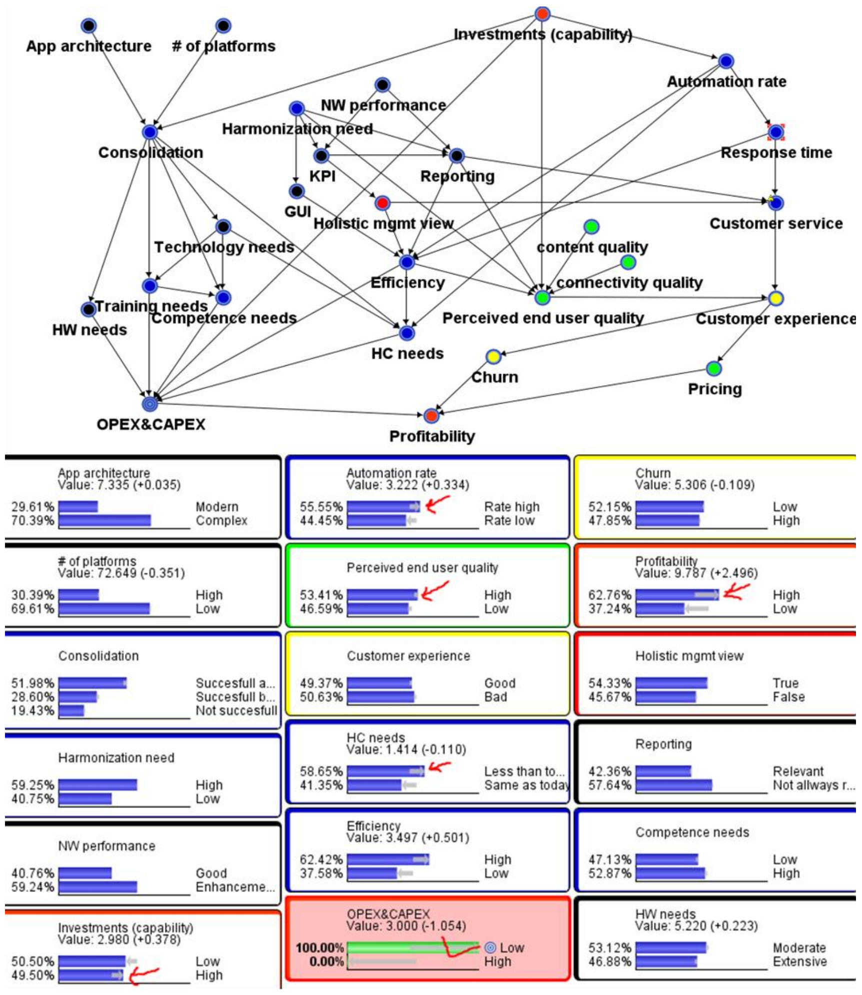


Figure 6: Key causal structure (upper part), states and conditional probabilities (lower part) of some variables in electric car ecosystem model. The variable "New business opportunities" represents potential new business for CSPs.



```
App architecture
Value: 7.335 (+0.035)
29.61%  Modern
70.39%  Complex

# of platforms
Value: 72.649 (-0.351)
30.39%  High
69.61%  Low

Consolidation
51.98%  Succesfull a...
28.60%  Succesfull b...
19.43%  Not succesfull

Harmonization need
59.25%  High
40.75%  Low

NW performance
40.76%  Good
59.24%  Enhanceme...

Investments (capability)
Value: 2.980 (+0.378)
50.50%  Low
49.50%  High

Automation rate
Value: 3.222 (+0.334)
55.55%  Rate high
44.45%  Rate low

Perceived end user quality
53.41%  High
46.59%  Low

Customer experience
49.37%  Good
50.63%  Bad

HC needs
Value: 1.414 (-0.110)
58.65%  Less than to...
41.35%  Same as today

Efficiency
Value: 3.497 (+0.501)
62.42%  High
37.58%  Low

OPEX&CAPEX
Value: 3.000 (-1.054)
100.00%  Low
  0.00%  High

Churn
Value: 5.306 (-0.109)
52.15%  Low
47.85%  High

Profitability
Value: 9.787 (+2.496)
62.76%  High
37.24%  Low

Holistic mgmt view
54.33%  True
45.67%  False

Reporting
42.36%  Relevant
57.64%  Not allways r...

Competence needs
47.13%  Low
52.87%  High

HW needs
Value: 5.220 (+0.223)
53.12%  Moderate
46.88%  Extensive
```

Figure 7: High level sub-BBN for typical European CSP Operations Support System (OSS) based on one expert's views. OPEX and CAPEX targets have been set to 100% in order to test the consequences: Automation rate needs to be enhanced, similarly more investment, head count reduction and activities to enhance perceived user quality are needed (red arrows).

Example 3: The task of Operations Support Systems (OSS) is to take care of day-to-day infrastructure management so that the network and related services work properly with high quality and in an optimized way.

OSS BBN parent variables are the number of today's management platforms (rather low), investment capability of the company (often restricted), current OSS architecture (often complex), network performance (often not enough) and harmonization need (typically high). The target variables in the model are OPEX and revenue/profitability. The model covers variables like training needs, technology, head count, perceived quality seen by customer, customer experience and automation need (*Figure 7*). The model, even though it is on a rather high level, demonstrates the great potential of Bayes Belief Network as a methodology for business reasoning and what-if analysis.

Also other innovative sub-BBNs were created, such as IPTV model, customer experience & satisfaction model, regulator causalities model.

3.2 THE COMPREHENSIVE BBN

The comprehensive Bayesian Belief Network was created from sub-BBNs as described in chapter 2. The BBN contains the kernel shown in *Figure 3*. The model (*Figures 8 and 9*) contains the 32 most used variables and their 93 causal connections. It is remarkable that three variables are very central in the model: 12 variables have variable called "Customer experience & satisfaction", 9

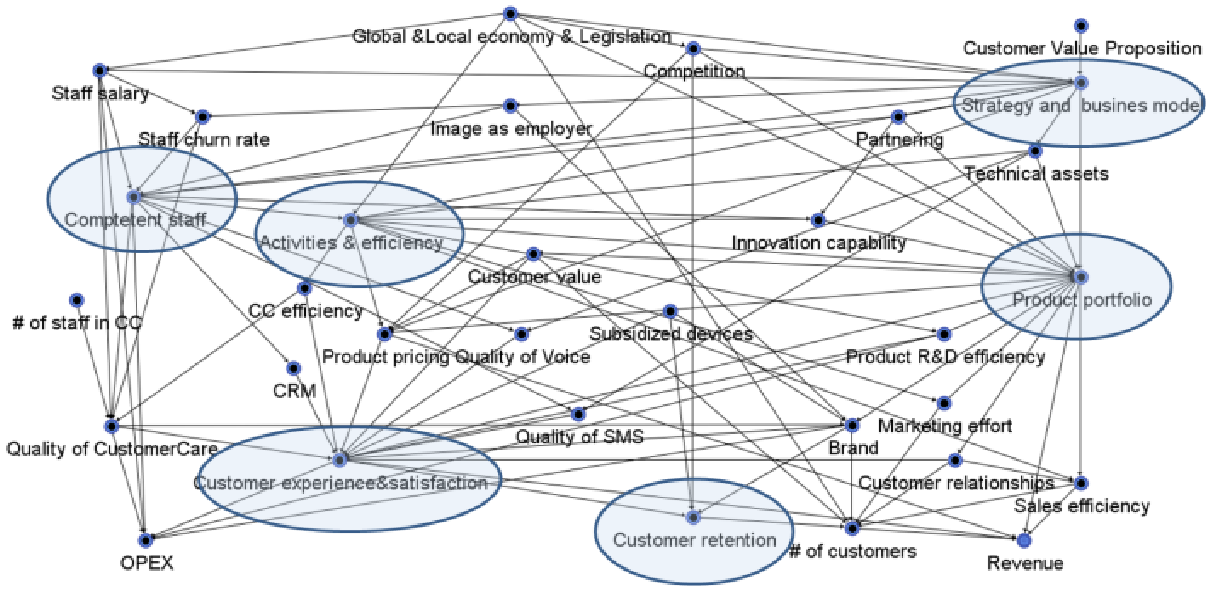
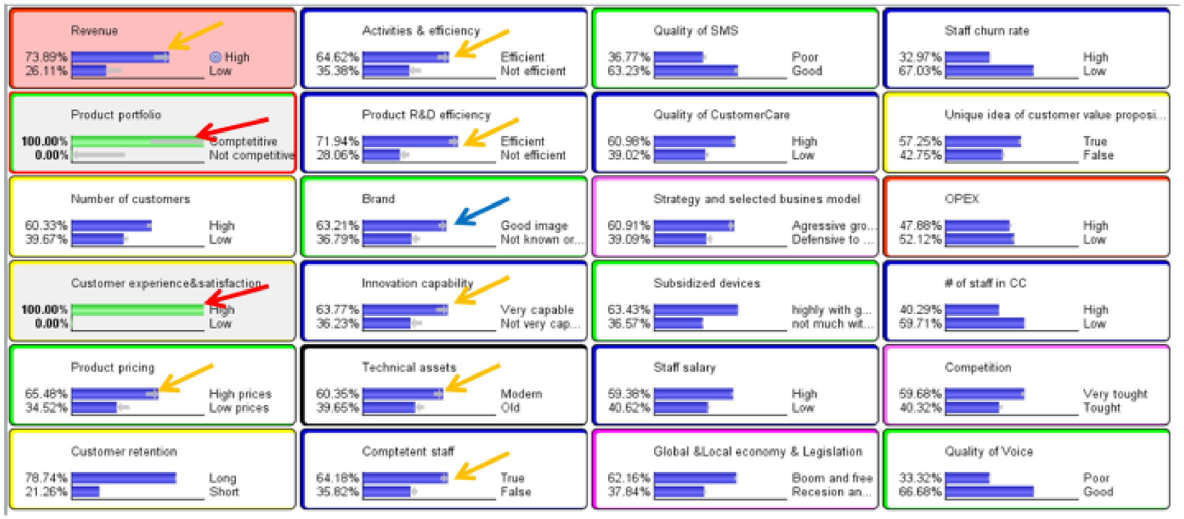


Figure 8: The comprehensive Bayesian Belief Network representing the overall feedback of the survey with granularity of 32 variables and 93 causal connections. The blue disks are the six variables, which have highest node forces as a sum of entering and outing arcs forces. Three from them, namely Customer experience & satisfaction, Product portfolio and Activity & efficiency are the central variables in the model.



Revenue
73.89% High
26.11% Low

Activities & efficiency
64.62% Efficient
35.38% Not efficient

Quality of SMS
36.77% Poor
63.23% Good

Staff churn rate
32.97% High
67.03% Low

Product portfolio
100.00% Competitive
0.00% Not competitive

Product R&D efficiency
71.94% Efficient
28.06% Not efficient

Quality of CustomerCare
60.98% High
39.02% Low

Unique idea of customer value proposi...
57.25% True
42.75% False

Number of customers
60.33% High
39.67% Low

Brand
63.21% Good image
36.79% Not known or...

Strategy and selected busines model
60.91% Agressive gro...
39.09% Defensive to ...

OPEX
47.88% High
52.12% Low

Customer experience&satisfaction
100.00% High
0.00% Low

Innovation capability
63.77% Very capable
36.23% Not very cap...

Subsidized devices
63.43% highly with g...
36.57% not much wit...

# of staff in CC
40.29% High
59.71% Low

Product pricing
65.48% High prices
34.52% Low prices

Technical assets
60.35% Modern
39.65% Old

Staff salary
59.38% High
40.62% Low

Competition
59.68% Very tought
40.32% Tought

Customer retention
78.74% Long
21.26% Short

Comptetent staff
64.18% True
35.82% False

Global &Local economy & Legislation
62.16% Boom and free
37.84% Recesion an...

Quality of Voice
33.32% Poor
66.68% Good

Figure 9: The states and probabilities of comprehensive BBN. The probability of the first state of variable “Product portfolio”, “Customer experience and satisfaction” has been set to 100% (green bars).

variables “Product portfolio” and 5 variables “Efficiency” as a common variable. On the other hand there is only one purely technical variable even though 20% of experts had technical expertise. The reason for the lack of technical variables might be the fact that most of the experts were from Europe and the model represents thus mostly a mature European mobile and convergent operator’s environment where customer experience, efficiency and portfolio play important role. The 32 variables and 93 arcs in the model were selected as a compromise between model granularity and usability and based on response times in analysis.

A light validation has been done for the model to verify whether it gives logical results especially because a simplistic conditional probability calculation method, where both weights for arcs and probabilities originate from same strength of effect value given by experts, has been used. *Figure 9* gives an example of the tests: It shows that when the probability of “Product portfolio” state “competent” is set to 100% and “Customer experience and satisfaction” state “high” is also set to 100%, the consequence will be that the revenue will increase clearly, when it is assumed that product pricing can be higher, efficiency of internal processes will be better, innovation capacity will increase, technical assets are modern and competent staff will be in place. These validations demonstrated that the model yield logical results.

4. SUMMARY AND DISCUSSIONS

An extensive study to model communication service provider businesses was performed. A novel method was used to elicit this prior information. Especially the way to create the so called causal triplets in the survey and to construct the initial BBN based on the triplets was found to be a fast, innovative and effective method that can be used to create different kind of causal maps. All together 40 Bayes Belief Networks were created, each representing an individual expert’s view. These networks were then merged to one comprehensive Bayes Belief Network.

Models such as an IPTV model, OSS management model, customer experience & satisfaction model, regulator causalities model, the electric car ecosystem, financial model are examples of innovative sub-models. The biggest challenge in the knowledge collection was the excessive freedom in variable naming. Creation and usage of a dictionary would be, from work amount and quality points of view, a clear improvement for the eliciting of prior knowledge and this is highly recommended for future studies.

Calculations of conditional probability tables were based on a weighted sum algorithm utilizing compatible parent configurations for the states. The process used in this study was simplified but yielded promising results which motivate the further testing of the models in a real life environment. However, a dual review method with experts is needed in order to achieve more reliable results.

The results showed that the BBN is a potential method for what-if analysis and predictions in the strategy creation process but also in daily management and decision tasks.

Future research topics are to enhance and benchmark some of models in mobile operator environment in focused use cases as well as synthesis of expert knowledge with data in order to enhance the dynamicity of the model.

Acknowledgements

Colleagues in Department of Communications and Networking of Aalto University as well as all the nearly 50 experts, who spent hours by documenting their opinions, deserve a big thank. Special thanks to Gareth Smith and Matti Aksela, who created the excel and R – based macros for calculation of conditional probability tables and finally big thanks to Antti Myllykangas, who has altruistic supported me in many aspects of this study.

References

T. Casey (2009). Analysis of Radio Spectrum Market Evolution Possibilities. *Communications and Strategies, No. 75, 109-130, September 2009*

B. Das (2004). Generating Conditional Probabilities for Bayesian Networks: Easing the Knowledge Acquisition Problem. *Command and Control Division, DSTO, Edinburgh, SA 5111, Australia*

E. Faber, P. Ballon, H. Bouwman, T. Haaker, O. Rietkerk, M. Steen (2003). Design business models for mobile ICT services. *16th Bled Electronic Commerce Conference Bled, Slovenia June 9-11, 2003*

F.V. Jensen (2001). Bayesian Networks and Decision Graphs.*Springer-Verlag, New York, ISBN 0-387-95259-4.*

Y. Kasperskaya (2006). Essays on Causal Performance Measurements Models. *Universitat Pompeu Fabra, Department of Economics and Business, PhD Dissertation December 2006*

E. Lee, Y.Park and J.G. Shin (2009). Large engineering project risk management using a Bayesian belief network. *Expert Systems with Applications 36 (2009) 5880-5887*

S. Nadkarni and P. Shenoy (2004). A causal mapping approach to constructing Bayesian networks. *Decision Support Systems 38 (2004) 259-281*

A. Osterwalder (2003). An e-Business Model Ontology for Modeling e-Business. *15th Bled Electronic Commerce Conference.Bled, Slovenia, June 17-19, 2002*

A. Osterwalder (2005). Clarifying Business Models: Origins, Present, and Future of the Concept. *Communications of the Association for Information Systems Volume 15, May 2005*

J. Peppard, A. Rylander (2006). From Value Chain to Value Networks: Insights for Mobile Operators. *European Management Journal Vol. 24, Nos. 2-3, pp. 128-141, 2006*

D. Rigby and B. Bilodeau (2007). Management tools and trends 2007. *Bain & Company, Management Tools 2007*

A.J. Scavarda, T. Bouzdine-Shameeva, S. Meyer Goldstein, J.M. Hayas, A.V. Hill (2006). A Methodology for Constructing Collective Causal Maps. *Decision Science Volume 37, Number 2, May 2006*

L. Uusitalo (2007). Advantages and challenges of Bayesian networks in environmental modelling, *Ecological modelling 203 (2007) 312-38*

Requirements for Developing Strategic Decision Facilitation and Event Prediction Tools

Oscar Kipersztok
Boeing Research & Technology
P.O.Box 3707, MC: 7L-44
Seattle, WA 98124
oscar.kipersztok@boeing.com

Abstract

This paper describes the requirements of a strategic decision facilitation tool that relies on forecasting to support critical decisions. A hypothesis-driven (data supported) system rather than a purely data-driven methodology. It further describes the importance of simple and natural human-computer interactions that simplify the creation of complex domain models in a system that uses probabilistic reasoning methods to facilitate high-quality decision making under uncertainty. Such a system helps users create complex models, query them for predictions, formulate hypotheses and validate their prediction with evidence retrieved from a corpus of text documents. The system must have a technology to automatically assemble and explain the forecasts so that users--who should not be required to understand the mathematics behind the forecast--will be able to understand why certain predictions are being made.

1 INTRODUCTION

A principal goal in any forecast of future events is to help decision-makers deal with uncertainty. Our pictures of the present and the past are always incomplete and noisy. So uncertainty is ubiquitous. A deeper concern is not just the future, but even our *theories and models* of how events will unfold that are underdetermined by our data. Thus, forecasting from data alone is not sufficient and it can be improved by embedding the forecast technology within a larger framework for decision support. Such a framework can supplement our raw data with information about the appropriate context in which to interpret the forecast, will be able to better focus computational resources and minimize (as far as possible) the quantified uncertainty over the most relevant aspects of the forecast.

The goal of this paper is to clarify the requirements of a strategic decision facilitation tool that relies on forecasting to support critical decisions.

Such system will offer users maximum flexibility and provide quick turn-around through a decision facilitation process that allows: a) easy capture and organization of knowledge, b) building complex models that can be readily queried about future events, c) applying advanced algorithms, made transparent to users, to forecast predictions, d) searching and piecing together relevant and coherent argumentation in favor (or against) courses of action; and e) making actionable recommendations to facilitate significant strategic decisions.

2 KEY COMPONENTS

There are four key components to a forecasting system that will be discussed to facilitate high-quality decision-making: 1. the forecasting algorithms should have access to the context of decisions under consideration, not simply the raw data--that is, they should be *hypothesis-driven*; 2. the system should enable *simple and natural human-computer interactions* to allow forecasting directly over concepts of relevance and importance to the decision makers; 3. the simplicity of user interaction should not prevent the use of advanced *probabilistic reasoning methods* to quantify and minimize uncertainty over forecasts; and, lastly, 4. the system should be capable of *automatically constructing explanations* of forecasts which can be understood without requiring users to master the details of the forecasting algorithms. Together, these components yield a complete decision-support framework that allows for ongoing critical evaluation and validation of high-quality forecasts created.

3 HYPOTHESIS DRIVEN METHODOLOGY

3.1 HYPOTHESIS VERSUS DATA DRIVEN

There is a body of evidence in experimental psychology suggesting different modalities in the way people make decisions; some modalities result in more accurate decisions than others (Heuer, 1999). In general, there is the distinction between "data-driven" and "hypothesis-driven" decision making. In the former, the emphasis is on initial search and gathering of as much information as possible before raising a hypothesis leading to an informed decision. In the latter, the emphasis is on a more selective and guided information search driven by a prior hypothesis. An iterative process follows where the search is aimed at specific information enabling validation or rejection of the hypothesis. The hypothesis is either accepted with sufficient evidence or re- formulated based on insufficient evidence. Validated hypotheses with sufficient evidence, in general, lead to more accurate decisions. Our decision support system is architected to direct users to follow a process that in practice has been shown to result in more accurate decisions.

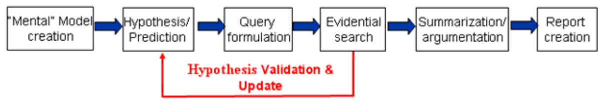


Figure 1: Sequential steps and feedback in hypothesis-driven decision making

4 SIMPLE AND NATURAL HUMAN COMPUTER INTERACTION

This section describes basic human computer interaction principles used to facilitate the creation of complex domain models while making transparent the complexity of analytic methods.

Figure 2 shows the three types of analytic methods required by the decision facilitation tool: 2a) knowledge representation and capture, 2b) reasoning inference and 2c) text processing and search methods.

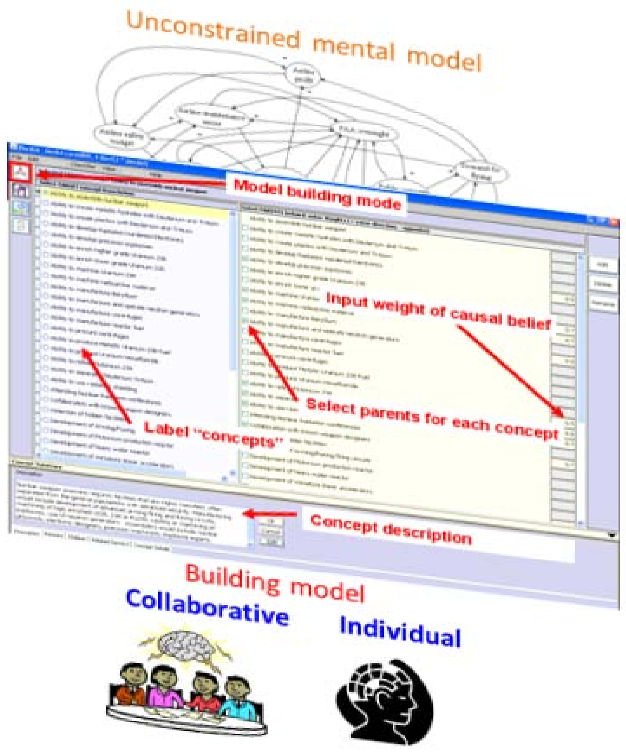


Figure 2a: Knowledge capture and representation

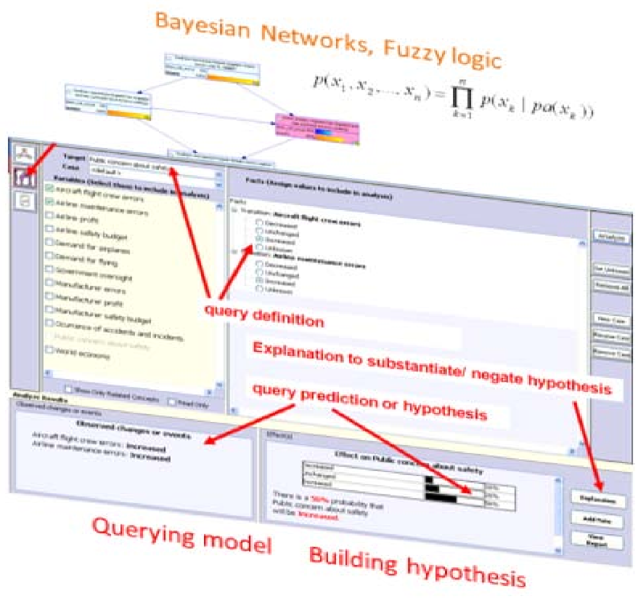

Bayesian Networks, Fuzzy logic
$p(x_1, x_2, \ldots, x_n) = \prod_{k=1}^{n} p(x_k \mid pa(x_k))$
query definition
Explanation to substantiate/ negate hypothesis
query prediction or hypothesis
Querying model
Building hypothesis

Figure 2b: Reasoning methods

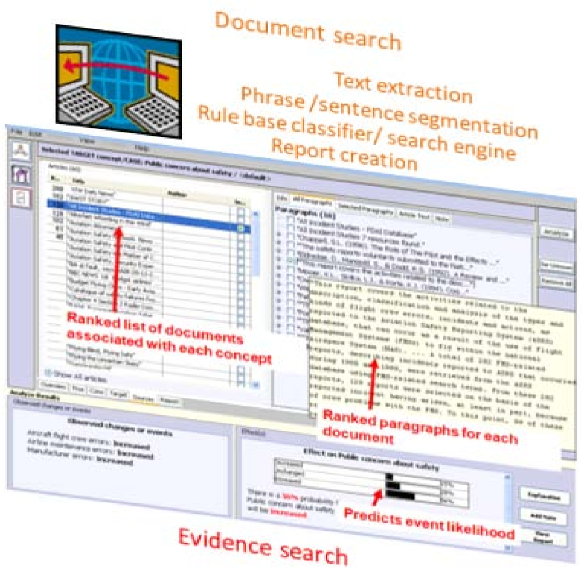

Document search
Text extraction
Phrase /sentence segmentation
Rule base classifier/ search engine
Report creation
Ranked list of documents associated with each concept
Ranked paragraphs for each document
Predicts event likelihood
Evidence search

Figure 2c: NLP & Text Processing

Three corresponding screens have been designed as user interfaces to allow users to build models by defining concepts and their associations, allowing to query the model and make predictions by making assumptions based on existing facts or beliefs, and searching for information through a corpus of documents in order to validate the assumptions and predictions.

These methods are used to provide maximum flexibility and *ease of use* for *rapid* model creation, immediate query response and prediction, and fast document retrieval for forecast validation.

4.1 TRANSPARENCY OF ANALYTIC METHODOLOGY AND TERMINOLOGY

Advanced analytic methods often require familiarity with complex methodology. In our approach to modeling complex domains, it is not necessary for users to learn and familiarize with analytic methods.

By making the analytic methods transparent, users interact with the system by only using the familiar language of their domain. Domain concepts are defined using free language and users can add to those definitions to make their meaning more precise. This eliminates the need for knowledge engineers to acquire and convert user knowledge and expertise into computational models.

4.2 KNOWLEDGE AND RATIONALE CAPTURE

The model creation process is the most critical step. Users create a “mental model” of their domain, which consists of concept definitions and causal relations between them. Most concepts affect or are affected by other concepts. In most realistic domains there is feedback where a particular concept starts a causal chain feeding back to itself. Feedback loops can induce complex reinforcing or inhibiting dynamic behavior. The “mental model” is critical because it is used to make predictions and to process and interpret outside information.

4.3 FREE LANGUAGE, ASSOCIATION AND BRAINSTORMING

The use of free language in model creation enables more flexibility in building models. Concepts are defined and labeled with short phrases or using a few words. To reduce ambiguity, users further expand concept definition by providing added descriptions for more precise meaning. Concepts are defined based on specific assumptions that also need to be captured. Additional documents and information (e.g. names, locations, specific dates, events, etc.) are also associated with each concept for further clarification. The use of free language serves a dual purpose. Firstly, the words and phrases used to define the concepts are also used in the creation of a rule- based search engine to improve the recall and precision of retrieved content needed to substantiate the decisions. Secondly, the concept definitions and attached descriptions are also used to create chains of rationale that will provide explanations to subsequent predictions.

4.4 COMPLEXITY MANAGEMENT AND SCALABILITY

Automated knowledge capture should be made easy for the user. It should be just as easy to build models of high complexity as it is very simple models. The user should not be concerned with how the knowledge is being captured, represented and organized. Users should be able to add or subtract information to and from the model with ease and at will. Quantity of information should not be of concern to users. The information should be easily accessible at any time during the model building process, or later during the analysis phase. Providing flexibility to users during model creation in a free-associative, brainstorming fashion is important since it enables: a) adequate coverage of the domain, leaving no stone unturned; b) seamless scalability to large, complex domains; c) collaborative multiple-user participation with access to second opinions and feedback; d) ease of model refinement and evolution at any future time; and e) speed - quick addition and deletion of ideas without concern about performance or limits of scalability. Building models fast, with ease and with transparent complexity management, enables users to build unconstrained models of any size. The knowledge is represented using constructs that are readily mapped into graphical probabilistic networks for subsequent forecasting and predictive analysis.

5 PROBABILISTIC REASONING METHODS

This section describes the creation behind-the-scenes, following the construction of the unconstrained mental model, of a Bayesian network for making probabilistic forecasts of events and trends.

5.1 FORECASTING PROBABILITY, IMPACT AND TIMING OF EVENTS

Analysis and prediction methods must be compatible with knowledge representation and acquisition constructs used in the knowledge capture phase. In addition to defining concepts and relations, analysis methods require additional quantitative input parameters that need to be obtained from the user during model creation. In the spirit of devising easy ways to capture knowledge directly from users, the number of requested variables is kept to a minimum. Methods were developed to map those variables to fit the requirements of the analysis and inference algorithms. Quantitative inputs should be acquired from the user in an intuitive manner within the context of the familiar user's domain.

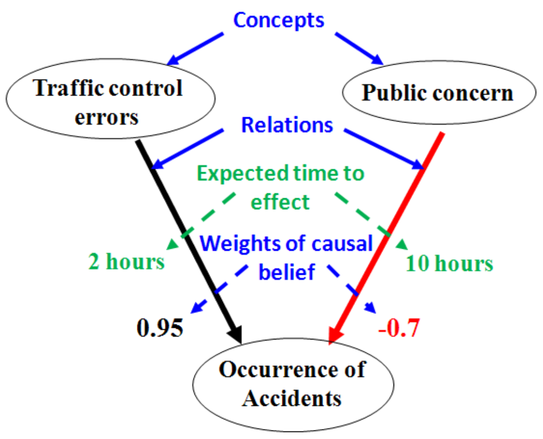


Figure 3: Request minimal number of parameters

The qualitative and quantitative inputs variables are shown in Figure 3. These are used to populate the parameters of graphical probabilistic (Bayesian) networks. The numbers represent the *weight of causal belief* that the user associates with each relation between pairs of concepts in the model. The relations and the belief numbers are used to build the structure and the conditional probability tables of the Bayesian networks by combining the weights of causal belief of incoming parent nodes (*Expected time to effect* will be discussed in section 5.2).

Alternative methods for building the networks and their conditional probability tables require obtaining probability estimates directly from domain experts for each combination of child and parent nodes' states. This tedious process can be aided by methods developed for probability elicitation (Wang, 2004). A major goal of our approach, however, is to circumvent this difficulty and make the process of building models readily accessible to users without need of expertise in graphical probabilistic methods. In either case, once the models are built, their performance and robustness can be validated using sensitivity analysis which can help identify the parameters that are most influential for any given query and prediction (Kipersztok and Wang, 2003).

Analysts and decision makers require probability estimates to guide strategic decisions. In order to maintain the simple and natural human-computer interface, our approach limits the decision space to predictions of *event occurrences* and *trends*. Decision makers need to know: a) how probable occurrences of events or emerging trends are; b) the magnitude of their *impact*; and c) the *time* when such events are expected to occur. Probabilistic models, in particular graphical models, provide capability to handle problems where data and information may be sparse, noisy or incomplete. In addition to rapid knowledge capture during model building, our methods also provide quick turn-around forecasting during the prediction phase.

As part of our on-going effort, we have built prototypes of the system. (Kipersztok, 2007) describes in more detail the implementation of various features of the system. The system has been applied to several specific domain areas. In (Seidler, et. al., 2010), the authors describe the DecAid system which was used to model and predict the readiness of a country to possess nuclear weapons capability. The paper reviews the domain associations used to build an unconstrained model. The model predictions were used to retrieve textual documents with information on Iran's nuclear program and to compile the risk assessment against the hypothesis that they are building a nuclear weapon.

5.2 REASONING ABOUT EVENT TIME

New methods are being developed that allow for probabilistic reasoning over systems evolving in continuous time (Nodelman, et. al., 2002). These techniques allow direct computation of distributions over when events of interest may occur. Moreover, they allow for automatic focusing of computational resources on those portions of the domain that may undergo rapid change. Larger, unified models over domains which include variables with widely divergent rates of change can thus be made computationally tractable. Machine learning algorithms can be used to help discover underlying structure in context where connections within the data are poorly understood (Nodelman, et. al., 2003).

There are already, in the literature, reviews of the advantages of these methods over traditional discrete-time probabilistic models--for instance, showing that the discrete-time models are subject to artifacts from the fixed time granularity and are less efficient to learn than the continuous time models (Nodelman, 2007). Adoption of these methods has been slow due to lack of exposure, limited software support, and ongoing research and development.

We define the *expected-time-to-effect* as one additional quantity to be provided by the user for each concept-pair relation defined in the model (Figure 3). Just as the *weight of causal belief* is used to create conditional probability tables in Bayesian networks; the addition of *expected-time-to-effect* is used for the construction of transition matrices in underlying continuous-time Bayesian networks. In this manner the system also can forecast the timing of events.

5.3 IDENTIFICATION OF CONCEPTS RELEVANT TO A QUERY

Experimental psychology experiments show that a critical number of variables is needed to make predictions at a fixed level of accuracy. Adding more variables to the decision increases the expert's confidence, without necessarily improving the accuracy of the prediction (Oskamp, 1965)(Shepard , 1964). This emphasizes the importance of being able to determine the critical, relevant concepts associated with a specific query. This important feature of our decision facilitation system is based on research done on relevance and feature selection learning algorithms (Druzdzel and Suermondt., 1994)(Fu and Desmarais, 2008b).

Querying the model triggers a prediction. A query is a request for predicting the future state of a 'target' concept given the assumptions about the current state of a set of 'source' or 'trigger' concepts. The model can contain any number of concepts and associations, but for each query the 'source', the 'target' and the set of 'relevant' concepts are the critical set of concepts that matter.

Once the model is complete the system is ready for inference and prediction, based on a specific query. At the time when the query is made, the system identifies the variables and relations relevant to that query and that subset of the unconstrained model is converted into a predictive model (Figure 4).

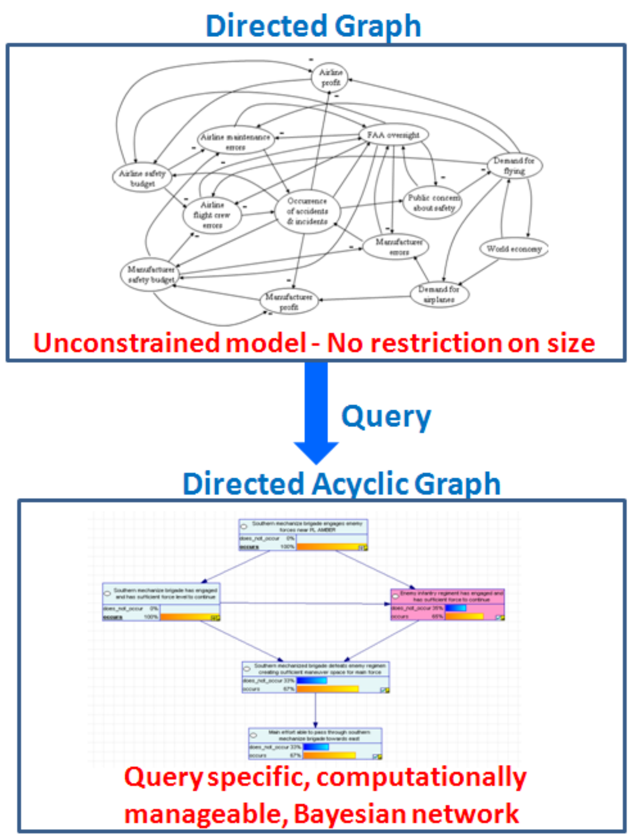

Directed Graph
Airline profit
Airline maintenance errors
FAA oversight
Demand for flying
Airline safety budget
Airline flight crew errors
Occurrence of accidents & incidents
Public concern about safety
Manufacturer errors
World economy
Manufacturer safety budget
Manufacturer profit
Demand for airplanes
Unconstrained model - No restriction on size
Query
Directed Acyclic Graph
Query specific, computationally manageable, Bayesian network

Figure 4: Creation of a predictive model (DAG)

5.4 INDIVIDUALS VS. COLLECTIVE JUDGMENT – CONSENSUS VS. DISENTING OPINIONS

A system that offers such model building flexibility and quick turn-around in decision-facilitation and forecasts can be equally effective for use by a single analyst as well as by a collective group of decision makers. Difficult and significant decisions are often arrived at by consensus in a group setting. Collective consensus is often built around a particular set of assumptions, a hypothesis, and a prediction. Once this occurs, it becomes increasingly difficult to deviate from the consensus opinion. Consensus can often be dominated by a vocal minority within a group at the risk of ignoring dissenting but equally, or more, valid alternative opinions.

In our system, various parallel hypotheses can be formulated with ease and subject to different sets of assumptions. With our proposed approach, a single team member may be capable of quickly making predictions and forecasting scenarios based on a dissenting hypothesis, while at the same time compiling evidence that can be used to steer the consensus opinion in a different direction that may, convincingly, lead to a different and possibly better decision.

6 FORECAST VALIDATION AND EXPLANATION

In (Lacave and Díez, 2002), the authors review various explanation methods for Bayesian networks and argue, on the one hand, that the *normative approach* for building expert systems, based on probabilistic reasoning, leads to more robust and accurate results. On the other hand, they also require more explanation capability because the methods are more foreign to human beings than in the *heuristic approach*.

Here, we are suggesting that much of the information to be presented as explanation during inference and prediction can be captured upfront, during the model building phase. It is part of the contextual knowledge imparted by the user as concepts and relations are defined. A very specific context and specific assumptions are made for every concept and causal relation defined in the unconstrained model. The *weight-of-causal- belief* and the *expected-time-to-effect* are quantities also defined subject to very specific assumptions. The system allows for user to systematically add such context during the model creation phase. The information is organized so that it is readily available for retrieval at the time that the explanation is needed.

6.1 EXPLANATION AND CHAIN OF REASONING

Predictions of highly probable events, of high impact and possibly occurring in the near future will be of most interest to users. Before courses of action are decided, decision makers require *explanations* that support a particular prediction and its assumptions. In addition, they also require convincing *evidence* that can back the predictions with plausible and believable facts. Qualitative explanations are provided by showing the causal chains of reasoning from trigger assumptions to predicted target outcomes where the entire relevant context that was captured during model building is organized and presented in every step along various paths of reasoning. Rationale captured and documented by the users, together with evidence retrieved from document search, constitute the basis for the explanation given to decision makers associated with forecasts and predictions by the system.

6.2 INFORMATION RETRIEVAL-VALIDATION THROUGH SEARCH

In political, cultural and socio- economic domains, validation often comes from validated evidential facts. Having a hypothesis makes the search more efficient because it narrows the search for specific information as evidence for clearly stated assumptions; thus, lending credibility and validity to the predictions. Precise concept definitions and rationale that explain concept relations together with gathered evidence from search make it possible to support a hypothesis-driven prediction. In arriving at critical decision, the facilitation methods discussed can help users step through a process that helps capture knowledge and data, organize them, invoke analysis methods to forecast predictions, piece together evidence, and rationale for or against courses of action, and make actionable recommendations. The final choice of action must be ultimately made by humans. The system will compute and present the necessary trade-offs between risk and cost for each recommended course of action.

Retroactive historical analysis constitutes another validation approach. It entails making predictions of past events and comparing the model forecast to actual outcomes. Predictions can also be compared among different methods.

6.3 RAPID PROTOTYPING - 'WHAT IF" SCENARIOS

Being able to quickly build complex mental models, and having the underlying machinery to automatically convert the created entities and relations into analytic models to make immediate predictions, provide single or multiple users with great flexibility. A single user can in one sitting use their knowledge to build a complex model, define concepts and relations, document their rationale, query the model to make predictions, and search for evidence to validate a new hypothesis. A quick turn-around decision facilitation method like ours enables users to postulate various 'what-if' scenarios and test parallel hypotheses side by side.

6.4 AUTOMATED DOCUMENTATION AND SUMMARIZATION

Our system automatically compiles and packages all the information needed for a strategic decision by summarizing the hypothesis and its assumptions, together with the associated evidence, the forecasts and the chain of reasoning explaining the prediction in the context of the specific concept definitions and the assumptions made when causal relations were defined. This capability to provide a summary documentation of the prediction, the assumptions and its explanation can be made available to second parties for critique and revision before actions of significant consequence are taken.

7 SUMMARY

Requirements for a decision facilitation system are presented that describe human- machine interface concepts that simplify for users the creation of complex domain models, while making transparent the analytic methodology that requires additional, specialized expertise. Those simplifying features are built into the user-interface to help users step through the creation of a model, query the model to make predictions, formulate hypotheses and validate the prediction from searched evidence (for or against) retrieved from a large corpus of documents. Explanation to predictions combines the rationale captured from the user during model development and the evidence gathered in support of a hypothesis; and it is presented to decision makers in context along the various paths of causal inference.

Acknowledgments

Very special thanks to Uri Nodelman for his insightful comments and feedback; and for his valuable contribution in the area of temporal reasoning.

References

Druzdzel, M., Suermondt, H. (1994). Relevance in Probabilistic Models: “Backyards” in a “Small World”. In *Working Notes of the AAAI 1994 Fall Symposium Series: Relevance*, pp 60-63, New Orleans, Louisiana, Nov 4-6, 1994.

Fu, S. K. and Desmarais, M. C. (2008b). Tradeoff Analysis of Different Markov Blanket Local Learning Approaches. *In Proceedings of Pacific Asian Conference on KDD (PAKDD)*, 562-571, 2008b.

Heuer, R. J. Jr. (1999). Psychology of Intelligence Analysis. *Center for the Study of Intelligence, Central Intelligence Agenc*y, 1999.

Kipersztok, O. (2007). “A Tool that Uses Human Factors to Simplify Model Building and Facilitate More Accurate Strategic Decisions”. *Fourth Bayesian Modeling Applications Workshop at the Uncertainty in AI Conference*, Vancouver BC, Canada, July, 2007.

Kipersztok, O. and Wang, H. (2003). Validation of Diagnostic Models Using Graphical Belief Networks. *In Intelligent Systems for Information Processing: From Representation to Applications.* B. Bouchon-Meunier, L. Foulloy, R.R. Yager. Elsevier, 2003.

Koller, D., and Sahami, M. (2006). Toward Optimal Feature Selection. In *Proceedings of International Conference in Machine Learning (ICML)*, 284-292, 2006.

Lacave, C. and Díez, F. J. (2002). "A review of explanation methods for Bayesian networks", *Knowl. Eng. Rev.*, vol. 17, p.107 , 2002.

Nodelman, U., Shelton, C. R., and Koller, D. (2002). Continuous Time Bayesian Networks. *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence*, pp. 378-387, 2002.

Nodelman, U., Shelton, C.R., and Koller, D. (2003). Learning Continuous Time Bayesian Networks. *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence,* pp. 451-458, 2003.

Nodelman, U. (2007). Continuous Time Bayesian Networks. *Ph.D. Dissertation, Stanford University*. 2007.

Oskamp, S., (1965). Overconfidence in Case-Study Judgments, *Journal of Consulting Psychology,* 29, 1965, pp. 261-265, 1965.

Shepard, R. N. (1964). On Subjectively Optimum Selection Among Multi-attribute Alternatives, in M. W. Shelly, II and G. L. Bryan, eds., *Human Judgments and Optimality*, New York: Wiley, 1964, p. 166, 1964.

Seidler, W., Kipersztok, O., and Wright, K. (2010). "Tools to Identify Nuclear Terrorist," *Journal of Radiation Effects, Research and Engineering*, Vol 28, No.1, pp 122 - 133, February 2010.

Wang H. (2004). Building Bayesian Networks: Elicitation, Evaluation, and Learning. *Ph.D. Dissertation, University of Pittsburgh*, 2004.

Modeling Dynamic Systems with Memory: What Is the Right Time-Order?

Anna Łupińska-Dubicka[1] and **Marek J. Druzdzel**[1,2]
a.lupinska@pb.edu.pl, marek@sis.pitt.edu
[1] Faculty of Computer Science, Białystok University of Technology, Wiejska 45A, 15-351 Białystok, Poland
[2] Decision Systems Laboratory, School of Information Sciences and Intelligent Systems Program, University of Pittsburgh, Pittsburgh, PA 15260, USA

Abstract

Most practical uses of Dynamic Bayesian Networks (DBNs) involve temporal influences of the first order, i.e., influences between neighboring time steps. This choice is a convenient approximation influenced by the existence of efficient algorithms for first order models and limitations of available tools. We focus on the question whether constructing higher time-order models is worth the effort when the underlying system's memory goes beyond the current state. We present the results of an experiment with a series of DBN models monitoring woman's monthly cycle. We show that higher order models are significantly more accurate. However, we have also observed overfitting and a resulting decrease in accuracy when the time order chosen is too high.

1 Introduction

All real world systems change over time. Modeling their equilibrium states or ignoring change altogether, when it is sufficiently slow, is sufficient for solving a wide spectrum of practical problems. In some cases, however, it is necessary to follow the change that the system is undergoing and introduce time as one of the model variables.

We concentrate in this paper on models that belong to the class of probabilistic graphical models, with their two prominent members, Bayesian networks (BNs) (Pearl, 1988) and dynamic Bayesian networks (DBNs) (Dean & Kanazawa, 1989). BNs are widely used practical tools for knowledge representation and reasoning under uncertainty in equilibrium systems. DBNs extend them to time-dependent domains by introducing an explicit notion of time and influences that span over time. Most practical uses of DBNs involve temporal influences of the first order, i.e., influences between neighboring time steps. This choice is a convenient approximation influenced by existence of efficient algorithms for first order models and limitations of available tools. After all, introducing higher order temporal influences may be costly in terms of the resulting computational complexity of inference, which is NP-hard even for static models. Limiting temporal influences to influences between neighboring states is equivalent to assuming that the only thing that matters in the future trajectory of the system is its current state. Many real world systems, however, have memory that spans beyond their current state.

The question that we pose in the paper is whether introducing higher order influences, i.e., influences that span over multiple steps, is worth the effort in the sense of improving the accuracy of the model. The idea of increasing modeling accuracy by means of increasing the time order of the model was beautifully illustrated by Shannon (1948). In his seminal paper, he shows sentences in the English language, generated by a series of Markov chain models of increasing time order, trained by means of the same corpus of text. The following sentence was generated by a first order model:

```
OCRO HLI RGWR NMIELWIS EU LL NBNESEBYA TH
EEI ALHENHTTPA OOBTTVA NAH BRL.
```

Compare this with the following sentence generated by a sixth order model:

```
THE HEAD AND IN FRONTAL ATTACK ON AN ENGLISH
WRITER THAT THE CHARACTER OF THIS POINT IS
THEREFORE ANOTHER METHOD FOR THE LETTERS
THAT THE TIME OF WHO EVER TOLD THE PROBLEM
FOR AN UNEXPECTED.
```

The resemblance of the latter sentence to ordinary English text, an informal measure of the model's accuracy, has increased dramatically between the first and the sixth orders. A first order model was essentially impotent in its ability to model the problem.

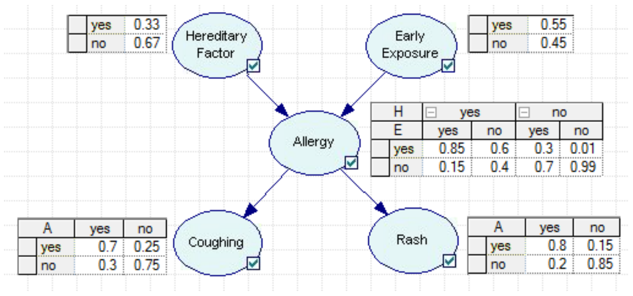
While generation of English sentences may be too hard of a problem, the vehicle for our experiments is the problem of monitoring the woman's monthly cycle, a problem central to family planning. Every couple seeking help in a fertility clinic is asked to monitor the monthly cycle before any intervention is undertaken. An accurate monitoring model can be a great aid in natural family planning, indicating optimal days for sexual intercourse. What is important from the perspective of the question posed in this paper is that woman's monthly cycle is a system with memory going most certainly beyond one day and probably spanning over a period of roughly a month.

We report the results of an experiment in which we successively introduce higher order DBNs modeling the monthly cycle and measure the accuracy of these models in predicting the day of ovulation. We train our models on real time series data obtained from a longitudinal study of fecundability conducted in several European centers (Colombo & Masarotto, 2000). We show that increasing the time order of the model greatly improves its accuracy. However, we also observe that when the time order is too high, the model can overfit the data and the quality of its predictions may decrease.

2 BNs and DBNs

Consider the simple BN shown in Figure 1, illustrating various causes and effects of allergy in children. All variables in this example are Boolean. The tendency to develop allergies has a hereditary component: Allergic parents are more likely to have allergic children, whose allergies are likely to be more severe than those from non-allergic parents. Exposure to allergens, especially in early life, is also an important risk factor for allergy. When an allergen enters the body of an allergic child, the child can cough or develop a rash. Figure 1 shows the dependency structure among the variables and the conditional probability distributions for each of the variables.

DBNs (Dean & Kanazawa, 1989) are an extension of BNs for modeling dynamic systems. The term *dynamic* means that we model the system's development over time and not that the model structure and its parameters change over time, even though the latter is theoretically possible. In a DBN, the state of a system at time t is represented by a set of random variables $\mathbf{X}^t = (X_1^t, \ldots, X_n^t)$. The state at time t generally dependents on the states at previous k time steps. There is nothing in the theory that prevents k from being any number between 1 and $t-1$.



yes	0.33
no	0.67

yes	0.55
no	0.45

H	yes		no	
E	yes	no	yes	no
yes	0.85	0.6	0.3	0.01
no	0.15	0.4	0.7	0.99

A	yes	no
yes	0.7	0.25
no	0.3	0.75

A	yes	no
yes	0.8	0.15
no	0.2	0.85

Figure 1: A simple BN illustrating selected causes and effects of allergy in children

When each state of the model depends only on the immediately preceding state (i.e., $k = 1$, the system is first-order Markov, often assumed in practice), we represent the transition distribution $P(X^t|X^{t-1})$. This can be done using a two-slice BN fragment (2TBN) $\mathcal{B}^t$, which contains variables from $\mathbf{X}^t$ whose parents are variables from $\mathbf{X}^{t-1}$ and/or $\mathbf{X}^t$, and variables from $\mathbf{X}^{t-1}$ without their parents. A first order DBN is often defined as a pair of BNs $(\mathcal{B}^0, \mathcal{B}^{\rightarrow})$, where $\mathcal{B}^0$ represents the initial distribution $P(\mathbf{X}^0)$, and $\mathcal{B}^{\rightarrow}$ is a two time slice BN, that defines the transition distribution $P(\mathbf{X}^t|\mathbf{X}^{t-1})$ as follows:

$$P(\mathbf{X}^t|\mathbf{X}^{t-1}) = \prod_{i=1}^{n} P(X_i^t|Pa(X_i^t)) \ .$$

Consider a two years old child whose parents suffer from allergy and who has been exposed to allergens. We know that this child has not developed any symptoms of allergy in the previous year. Suppose that we want to know the probability that allergy appears in the third year. If we use the BN pictured in Figure 1, we omit all historical information except that for the current year. Figure 2a shows a DBN of first temporal order, which allows us to predict the probability of the child developing allergy in this and in the future years. Number of slices is the number of steps for which we perform the inference. In this example, one step means one year. Temporal plate is the part of a DBN that contains nodes changing over time. *Hereditary Factor* is outside of the temporal plate and, hence, is time invariant.

Figure 2b shows a second time-order DBN, i.e., a model in which there are two temporal arcs from node *Allergy*, the first order takes the information from one step before, the second from two steps before. Typically, the older the child, the lower the probability of allergy appearing. And, generally, a child who has not developed allergy two years in a row has a lower chance of developing allergy in the third year. A reasonable expectation is that modeling higher order dependencies should increase the accuracy of the model.

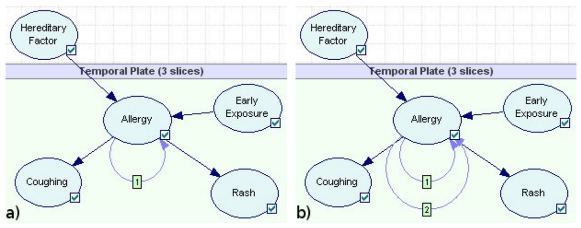


Figure 2: DBNs modeling causes and effects an allergy in children: first order (a) and second order (b) DBN

3 Woman's monthly cycle

Woman's monthly cycle is driven by a highly complex interaction among hormones produced by three organs of the body: the hypothalamus, the pituitary gland, and the ovaries. There are five main hormones involved in the menstrual cycle process: estrogen, progesterone, gonadotropin releasing hormone (GnRH), follicle stimulating hormone (FSH), and lutenizing hormone (LH).

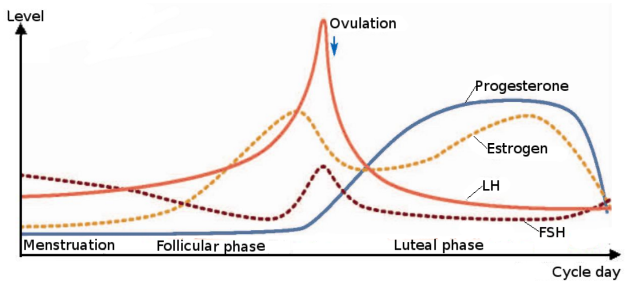


Figure 3: Levels of hormones during the phases of the woman's monthly cycle (Barron & Fehring, 2005)

The woman's monthly cycle consists of four phases (Figure 3 shows these four phases along with the associated hormone levels): (1) menstruation, (2) the follicular phase, (3) ovulation, and (4) the luteal phase. Counting from the first day of the menstrual flow, the length of each phase may vary from woman to woman and then cycle to cycle.

In addition to measurable blood hormone levels, there are several easily accessible indicators of the phase of the cycle, two of which we will use in our models. Basal body temperature (BBT) is defined as the body temperature measured immediately after awakening and before any physical activity has been undertaken. It should be measured every day at the same time. Before ovulation, BBT is relatively low. Following the ovulation, as a result of an increased level of progesterone in the body, women typically experience an increase in the basal body temperature (BBT) of at least 0.2°C. This shift indicates that ovulation has occurred. The BBT charting may provide valuable information about woman's monthly cycle, such as duration of the cycle, length of the follicular and luteal phases, and the pattern of the timing of ovulation. Sometimes BBT can rise due to causes other than ovulation. This atypical rise is treated as disturbance and can be caused by a change in conditions around the measurement, such as later measurement time, lack of sleep, different thermometer, high stress, travel, or illness.

As the cycle progresses, due to hormonal fluctuations, the cervical mucus increases in volume and changes texture. When there is no mucus or the mucus discharge is small, the day is considered infertile. There can be also a feeling of dryness around the vulva. Around the ovulation, mucus is the thinnest, clearest, and most abundant, resembling egg white. In the luteal phase, it returns to the sticky stage.

It seems that the menstrual cycle is a temporal process with memory spanning over the entire cycle. This means that the current state is not only influenced by the previous state but also by prior days, going back to the beginning of the phase.

4 The Model

Accurate prediction of the fertile phase of the menstrual cycle is critical for couples who want to conceive or couples who want to avoid pregnancy using natural methods. The fertile phase of the menstrual cycle is defined as the time when an intercourse has a non-zero probability of resulting in conception.

The number of fertile days during the menstrual cycle is difficult to specify, as it depends on the life span of the ovum and sperm, which varies from person to person and from cycle to cycle. It is generally believed that an ovum can be fertilized only within the first 24 hours after ovulation. Many authors agree that the start of the fertile interval is strictly connected with changes in vaginal discharge and, in particular, estrogenic-type cervical mucus secretions. However, they differ in their estimates of the length of the fertile window. Potter (1961) calculated that there are only two days during the menstrual cycle when a woman can become pregnant. Wilcox *et al.* (1995) found that the maximum sperm life span equals approximately five days (in presence of sufficient level of estrogenic-type mucus), which comes down to a fertile period of six days, including the day of the ovulation. The results of a multi-center study conducted by the World Health Organization (WHO, 1983) estimate the fertile period to be 10-days before ovulation. Natural family planning methods assume this interval to be as long as 13 days.

It is useful and important to be able to predict ovulation. Because the fertile period starts roughly five days before ovulation, prediction has to be made in advance and, hence, asks for models that include an explicit notion of time.

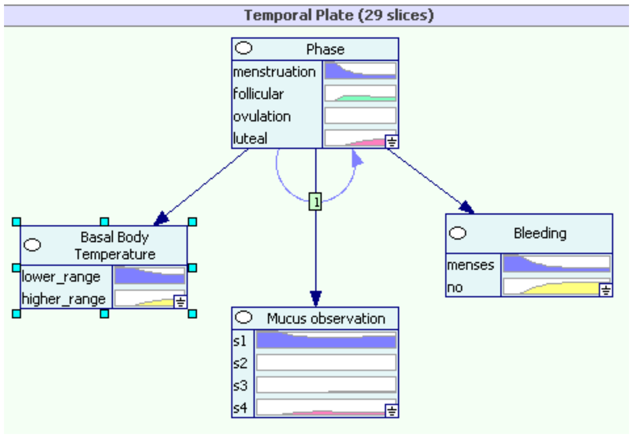


Figure 4: A first-order DBN model of woman's monthly cycle

Our model (Figure 4), combines information retrieved from BBT charting with observations of the cervical mucus secretions. It contains a variable *Phase* with four states: menstruation, follicular, ovulation, and luteal. We included three observation variables: *Basal Body Temperature* (BBT), *Bleeding* and *Mucus observation*. All variables are discrete. BBT has two possible values: lower range and higher range, representing temperature before and after the BBT shift respectively. *Bleeding* describes whether on a particular day the woman had menses or not. *Cervical observation* can be in one of four states (s1 through s4), described in detail in (Dunson, Sinai, & Colombo, 2001). We modeled time explicitly as n time steps, where n is the number of days of the longest monthly cycle of the particular woman.

Admittedly, this is a simple model. However, we would like to point out that it reasonably models the causal interactions among the variables in the data available to us.

5 The Training Data

Our training data are drawn from an Italian study of daily fecundability (Colombo & Masarotto, 2000), which enrolled women from seven European centers (Milan, Verona, Lugano, Düsseldorf, Paris, London and Brussels). To our knowledge, this is one of the most comprehensive data sets describing woman's monthly cycle. Between the years 1992 and 1996, 782 women recorded a total of over six thousand monthly cycles. Women participating in the study satisfied the following five entry criteria: (1) experienced in use of a Natural Family Planning method, (2) married or in a stable relationship, (3) between 18th and 40th birthday at admission, (4) had at least one menses after cessation of breastfeeding or after delivery, (5) not taking hormonal medication or drugs affecting fertility. In addition, neither partner could be permanently infertile and both had to be free from any illness that may affect fertility.

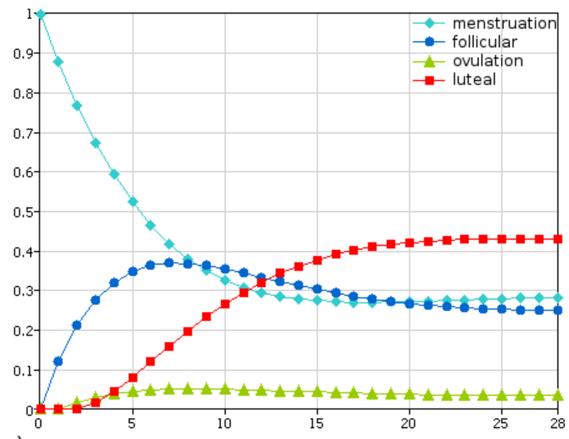
In each menstrual cycle, the subject was asked to record the days of her period, her basal body temperature and any disturbances such as illness, disruption of sleep or travel. She was also asked to observe and chart her cervical mucus symptoms daily during the cycle and to record every episode of coitus, with specification whether the couple used contraceptives or not.

Typically, a menstrual cycle is defined as the interval in days between the first day of menstrual bleeding in two neighboring cycles, where day 1 was the first day of fresh red bleeding, excluding any preceding days with spotting. The day of ovulation was identified in each cycle from records of basal body temperature and mucus symptoms. The daily mucus observations were classified into four classes; ranging from a score of 1 (no discharge and dry) to 4 (transparent, stretchy, slippery). The cervical mucus peak day was defined as the last day with best quality mucus, in a specific cycle of the woman. If there were different mucus observations on one day, the most fertile characteristic of the mucus observed determined the classification. To determine the BBT shift, the "three over six" rule was used: The first time in the menstrual cycle when three consecutive temperatures were registered, all of which were above the average temperature of the last six proceeding days.

6 Experiments

We tested our model on two different women taken from the Italian study. For each woman, we created a BN and nine DBNs of temporal orders ranging from 1 to 9, training them (i.e., learning their parameters) on the available monthly charts, using the leave-one-out method, i.e., training the network on all but one chart and testing it on the remaining chart. Because of computational limitations (with 30 time slices, ninth order models become fairly complex), we had to find women with a not too long average duration of the follicular phase. We were able to find two women with over 30 monthly charts each, whose follicular phase lasted typically around 9 days. We set the number of slices of the DBNs to the length of the longest cycle.

Just to give an idea of the capability of such models to reproduce the monthly cycle, we present the prob-

a)


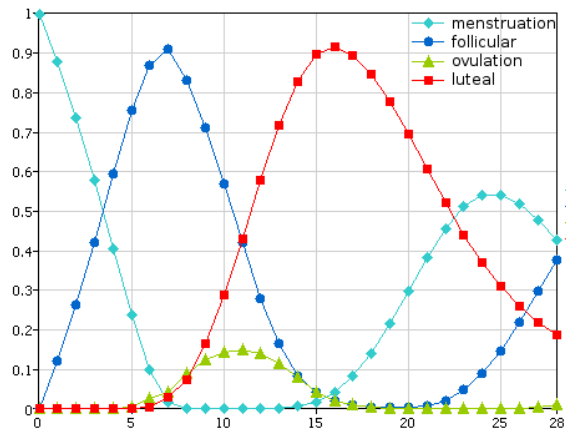
b)


Figure 5: Probabilities of each phase during the monthly cycle: order 1 (a) and order 7 (b) DBNs

abilities of the four phases of the monthly cycle as a function of time in Figure 5. These probabilities were generated by models of the first (a) and the seventh (b) order DBNs, trained on monthly charts of one of the women in the data set. We entered no observation into the models, except for anchoring the first time step to the first day of menses, i.e., first day of the monthly cycle. Please note the increased similarity of the shape of the curves to that of the hormone levels in Figure 3, which are direct indications of phases of the monthly cycle. Memory of the order 7 model is such that the model is capable of predicting roughly the day of ovulation on the first day of menses.

To compare the accuracy of different models, we used two measures: the true positive rate (TPR) and the false positive rate (FPR). These are defined as $\mathrm{TPR} = \mathrm{TP}/(\mathrm{TP} + \mathrm{FN})$ and $\mathrm{FPR} = \mathrm{FP}/(\mathrm{FP} + \mathrm{TN})$ respectively. In our model, TP is the number of true positives, i.e., the number of days of the cycle classified as ovulation that in fact were ovulation. FP is the number of false positives, i.e., the number of days of the cycle classified as ovulation that in fact belong to menstruation, follicular, or lutheal phase. TN is the number of true negatives, i.e., the number of days of the cycle not classified as ovulation that in fact belong to menstruation, follicular, or lutheal phase. FN is the number of false negatives, i.e., the number of days of the cycle not classified as ovulation that are ovulation.

From the practical perspective, for a model of a monthly cycle to be useful, it has to predict the day of ovulation at least five days in advance. Please note that because of a possible application of a model like this in family planning, false negatives may be very costly, so the model should minimize its false negative rate to zero. This is essentially the case with all natural family planning methods.

For each network, we created ROC graphs (Fawcett, 2003) by plotting sensitivity (TPR) vs. complement of specificity ($1-\mathrm{FPR}$). For each model, we had as many curves as there were cycles of the particular woman available. To plot the ROC curves, we used vertical averaging, i.e., for each FPR we took the averaged TPRs of the ROC curves over all cycles. For each curve, we also calculated the area under the ROC curve (AUC), which is a measure of model's ability to predict ovulation five days in advance. A useless model would have the AUC of 0.5. A model with perfect ability to predict would have the AUC of 1.0. If the 95% confidence interval of the model's AUC would include 0.5, the model would be not likely to predict accurately. We used ROCR (Sing, Sander, Beerenwinkel, & Lengauer, 1975), an R package for evaluating and visualizing classifier performance.

Figure 6 shows selected ROC curves created for the two selected women: static BN, first order DBN, DBNs with temporal orders from first to fourth, from first to sixth, from first to seventh, and with temporal orders from first to ninth. We did not picture every curve in order to avoid clattering the graphs but instead showed the ranges (vertical lines on the plot). Figure 7 presents the average AUCs for these women along with their ranges (vertical bars).

As we can see, in both women, a BN is not much better than a random classifier. From all DBNs, the networks with first temporal order and with first and second temporal orders give the worst results. In case of woman ID 20050265, the higher order of the network, the higher sensitivity at the same point of specificity (Figure 6a).

Figure 6b shows that for woman ID 20380003 the curve for DBN with orders higher than 6 does not achieve value $\mathrm{TPR} = 1.0$ until $1 - \mathrm{FPR} = 0.42$. Starting at the $1-\mathrm{FPR} = 0.28$, these high order DBNs give worse results than DBNs with lower temporal orders. Fig-

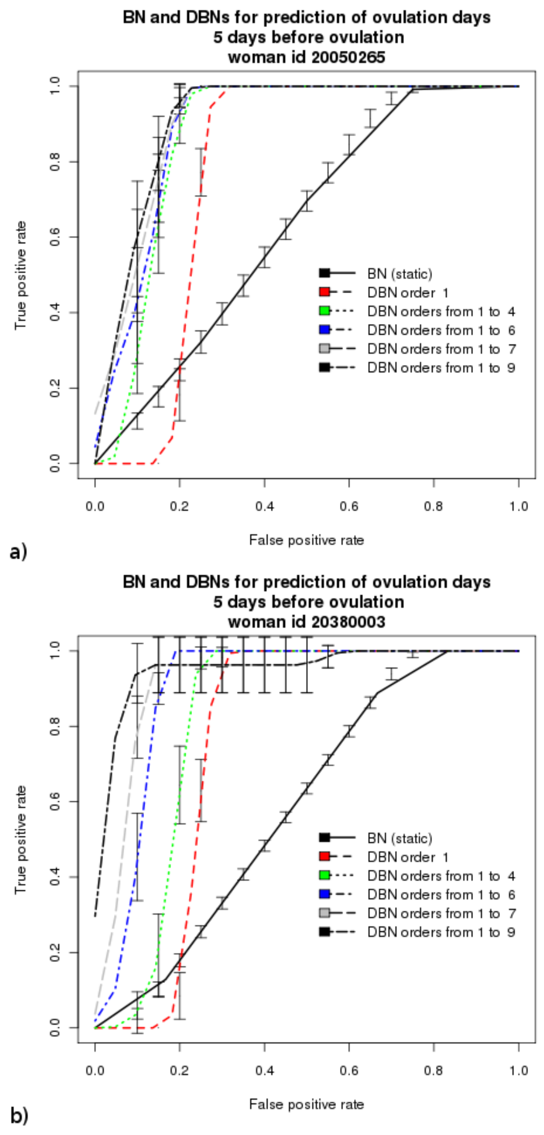


Figure 6: ROC curves with vertical averaging of BN and DBNs for prediction of the ovulation day

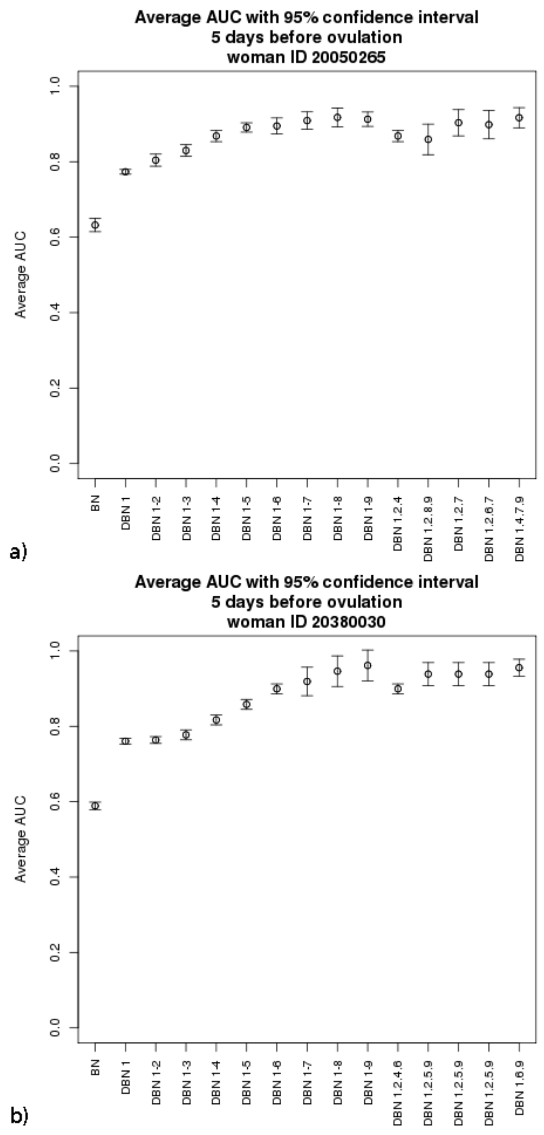


Figure 7: AUC ROC curves of BN and DBNs for prediction of the ovulation day

ure 8 shows this for each cycle separately. As we can see, there is one curve, whose AUC is smaller than 0.5. In this cycle, the follicular phase lasted only six days, while in all previous cycles its most common length was nine days. The model, learned on the basis of previous cycles, predicted ovulation day for the 15th day, while in reality it took place on the 12th day. Figure 9 is an equivalent of Figure 6b but with this anomalous cycle omitted. In this case, the higher order of the network, the higher sensitivity at the same point of specificity.

Clearly, too high of an order can reduce accuracy of the model. What is the optimal order of a model? We performed a number of additional experiments with monthly cycles of other women, varying the model order, and came to the conclusion that the optimal order of the DBN model depends directly on the nature of the system and the task that we set to perform. This number should be derived from the domain knowledge. If anomalies are to be expected, it does not make any sense to go beyond the order equal to the smallest of the following three numbers: (1) the length of the system's memory, which could be argued in our case to be the length of the woman's monthly cycle, (2) the length of the prediction horizon, which is the number of slices that we want to predict ahead (6 in our case), and (3) the maximum order that is still computable comfortably, which was in case of SMILE around 9.

Furthermore, while any DBN model should contain at least one first order influence (if that were not the case,

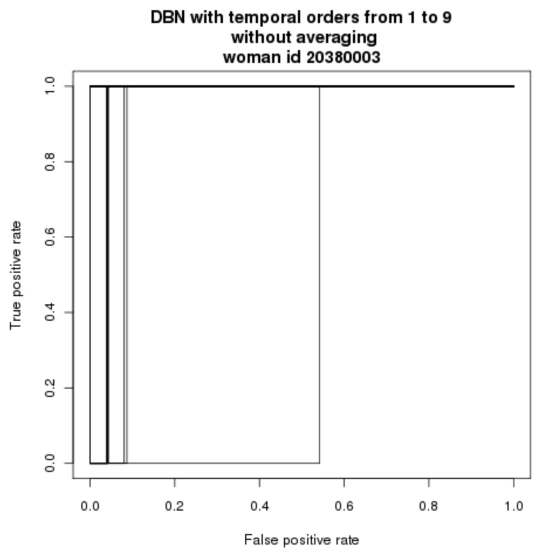


Figure 8: ROC curves for individual cycles of a DBN of order 1 through 9

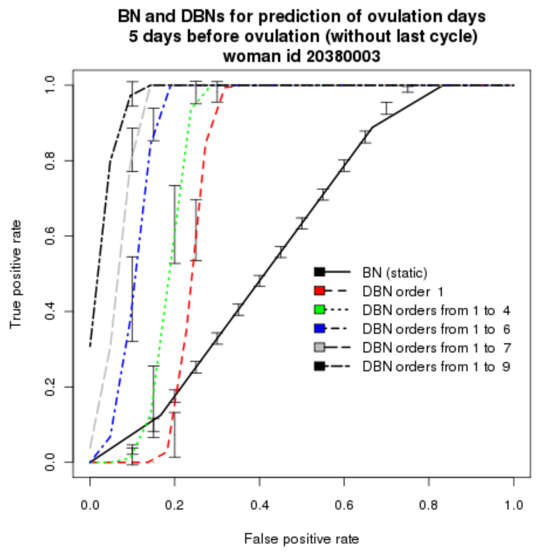


Figure 9: ROC curve for the DBN with temporal orders from 1 to 9 with anomalous cycle removed

some slices would be disconnected from the model!), a model of order k does not need to include influences of all orders between 1 and $k-1$. In our experiments with the monthly cycle, we focused on those influences that seemed critical for phase transitions and used orders that were equal to the lengths of the menstruation and the follicular phases, as given usually a clear indication the end of the menses, these influences could fairly precisely pinpoint the expected day of ovulation, even without additional observations.

Figure 10 shows ROC curves generated by DBNs with

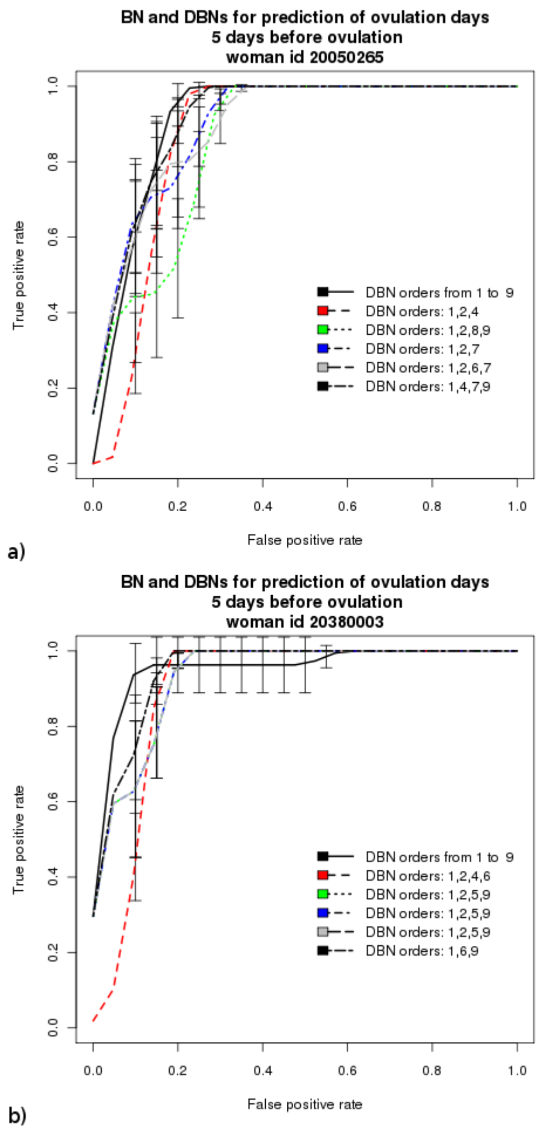


Figure 10: ROC curves with vertical averaging of additional DBNs for prediction of the ovulation day

temporal orders including the shortest, the longest, the most common, and the average length of the menstruation and the follicular phases. The last pictured networks have temporal orders connected with the minimum, maximum, mode, and average length of the follicular phase of the particular woman, whose charts were used to train the model. Figure 11 shows networks with selected orders for woman 20380003 with the anomalous cycle removed.

7 Discussion

We have presented the results of an experiment with a series of DBN models monitoring woman's monthly cycle. We have shown that higher order models are

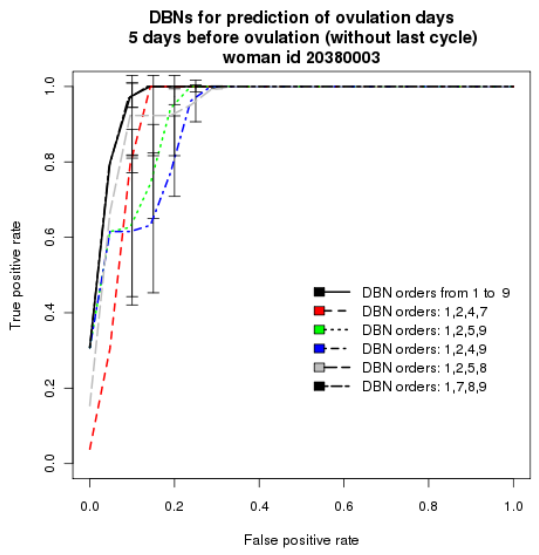


Figure 11: ROC curves with vertical averaging of additional DBNs for prediction of the ovulation day with the anomalous cycle removed

significantly more accurate than first order models, as summarized by the AUC graph in Figure 7. The ROC curves for higher order models were clearly closer to the upper left corner of an ROC graph, which indicates a better ability of the model to predict ovulation.

However, we also observed overfitting and a resulting decrease in accuracy when the time order chosen was too high. Having learned the lengths of the phases, which were shorter than the model's memory, the model seemed to lose its ability to predict accurately, when the cycle happened to be anomalous in terms of its length. Model's memory seemed to have a stronger influence on prediction than observations collected during the cycle. DBNs of lower orders reached sensitivity of 1.0 for lower values of specificity (Figures 6b and 10).

Thorough understanding of the underlying system and the task at hand is required to select the optimal order of the model. In addition to computational issues and issues related to a negative influence of model complexity on the quality of parameters learned from data, one should avoid choosing orders that are higher than system's memory and the task horizon.

Acknowledgements

Our work was supported in part by the National Institute of Health under grant number U01HL101066-01. We thank Bernardo Colombo, Guido Masarotto, Fausta Ongaro, Petra Frank-Herrmann, and other investigators of the European Study of Daily Fecundability for sharing their data with us. The empirical part of the paper was performed using SMILE, an inference engine, and GeNIe, a development environment for reasoning in graphical probabilistic models, both developed at the Decision Systems Laboratory, University of Pittsburgh, and available at `http://genie.sis.pitt.edu/`.

References

Barron, M. L., & Fehring, R. J. (2005). Basal body temperature assessment: Is it useful to couples seeking pregnancy? *American Journal of Maternal Child Nursing*, *30*(5), 290–296.

Colombo, B., & Masarotto, G. (2000). Daily fecundability: First results from a new data base. *Demographic Research*, *3*(5).

Dean, T., & Kanazawa, K. (1989). A model for reasoning about persistence and causation. *Computational Intelligence*, *5*(2), 142–150.

Dunson, D. B., Sinai, I., & Colombo, B. (2001). The relationship between cervical secretions and the daily probabilities of pregnancy effectiveness of the TwoDay Algorithm. *Human Reproduction*, *16*(11), 2278–2282.

Fawcett, T. (2003). *ROC graphs: Notes and practical considerations for researchers* (Technical Report No. HPL-2003-4). Hewlett Packard Laboratories.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: Networks of plausible inference.* San Mateo, CA: Morgan Kaufmann Publishers, Inc.

Potter, J., R.G. (1961). Length of the fertile period. *Milbank Quarterly*, *39*, 132–162.

Shannon, C. E. (1948, July, October). A mathematical theory of communication. *The Bell System Technical Journal*, *27*, 379–423, 623–656.

Sing, T., Sander, O., Beerenwinkel, N., & Lengauer, T. (1975). ROCR: Visualizing classifier performance in R. *Bioinformatics (Oxford, England)*, *55*(4), 699–706.

WHO. (1983). A prospective multicentre trial of the ovulation method of natural family planning. III. Characteristics of the menstrual cycle and of the fertile phase. *Fertility and Sterility*, *40*(6), 773–778.

Wilcox, A., Weinberg, C., & Baird, D. (1995). Timing of sexual intercourse in relation to ovulation. Effects on the probability of conception, survival of the pregnancy, and sex of the baby. *New England Journal of Medicine*, *333*(23), 1517–1521.

A decision support-system for the mediastinal staging of non-small cell lung cancer

Manuel Luque and **Francisco J. Díez**
Dept. Artificial Intelligence, UNED
Juan del Rosal, 16
28040 Madrid, Spain
{mluque,fjdiez}@dia.uned.es

Carlos Disdier
Pneumology Section
University Hospital
47005 Valladolid, Spain
cdisdier@separ.es

Abstract

Lung cancer is a very frequent tumor in the developed world and the leading cause of cancer death, with non-small cell lung cancer being the most prevalent type and with most difficult prognosis. In this paper we present a decision support system built for finding the optimal selection of tests and therapy for each patient. The system basically consists of an influence diagram with super value nodes. The parameter λ, which in cost-effectiveness analyses represents the amount of money that the decision maker is willing to pay to obtain a unit of effectiveness, has been included in the influence diagram, and has allowed us to find a trade-off between cost and effectiveness. Finally, given the uncertainty on the values of the parameters, we have assigned, with the expert's help, a probability distribution to each parameter of the model and have performed a probabilistic sensitivity analysis.

1 INTRODUCTION

Lung cancer is a very frequent tumor in the developed world and the leading cause of cancer death. Lung cancer can be classified into two major types: small-cell lung cancer (SCLC) and non-small cell lung cancer (NSCLC). The first one appears in 20% of cases, is usually inoperable and only treatable with chemotherapy or chemo-radiotherapy. In contrast, when limited to the lung, certain adjacent structures, and lymph nodes proximal to the lung, surgery resection remains the optimal treatment for NSCLC. However, more than 80% of NSCLC patients can not be treated with surgery because the disease is out of control due to an advanced local extension of the tumor or spreading to other parts of the body (metastasis). A disappointing fact is that a high percentage of patients that may benefit from surgery die of lung cancer. A correct assessment at an early stage of the disease and an accurate selection of patients (staging phase) is very important to apply surgery in good prognosis patients, and, in turn, to avoid dangerous, painful, and unnecessary surgery in bad prognosis patients.

When there are no distant metastases, mediastinal staging, i.e., determining whether malignant mediastinal lymph nodes are present or absent, is the most important prognostic factor in patients with NSCLC and, consequently, determines the therapeutic strategy. Different techniques are available to study the mediastinum. There are non-invasive imaging techniques, such as CT scan and PET, with high sensitivity but low specificity; there are also minimally invasive endoscopic techniques (TBNA, EBUS, EUS)[1], with low risk, high specificity and varying degrees of sensitivity, as well as more invasive surgical techniques, such as mediastinoscopy, which is considered as the gold standard.

The main treatment options for lung cancer include surgery, chemotherapy, radiation therapy, radio-chemotherapy, and palliative and supportive care. The applicability of each treatment depends on the stage of the tumor.

Because of this variety of available tests and treatments, each one having pros and cons, there is a vivid debate among specialists about which technologies should be used (Fritscher-Ravens et al., 2003; Schimmer et al., 2006). Nease and Owens (1997) proposed an influence diagram for the mediastinal staging of NSCLC, which provides a strategy for a simplified version of the problem. We propose here a new ID, with important improvements. We also describe how we have searched for a tradeoff between cost and effec-

[1]CT scan stands for computer tomography, PET for position emission tomography, TBNA for transbronchial needle aspiration, EBUS for endobronchial ultrasound, and EUS for endoscopic ultrasound.

tiveness by including in the influence diagram the parameter λ, which represents the amount of money that the decision maker is willing to pay to obtain a unit of effectiveness. Finally, we present the probabilistic sensitivity analysis (PSA) that we have performed given the uncertainty on the values of the parameters.

2 CONSTRUCTION OF MEDIASTINET

In this section, we describe the construction of Mediastinet, an influence diagram (ID) for the mediastinal staging of non-small cell lung cancer (NSCLC).

2.1 STRUCTURE OF THE GRAPH

Influence diagrams (Howard and Matheson, 1984) are a framework for representing and solving decision problems. An ID consists of an acyclic directed graph having three kinds of nodes: decision (graphically represented by squares or rectangles), chance (circles or ovals), and utilities (diamonds). Each decision node represents to actions under the direct control of the decision maker. Each chance node represents a random variable. In medical IDs, utility nodes represent medical outcomes and costs (morbidity, mortality, economic cost...). We will use the terms node and variable indifferently.

We next describe how the ID has been built by exploiting expert knowledge.

2.1.1 Identification of variables

Chance variables Given that our objective is the mediastinal staging of NSCLC, we have included a variable representing the value of N factor in the TNM classification[2] (Lloyd and Silvestri, 2001). Even though the N factor takes on four possible values, from N0 to N3, we have modeled it as a binary variable because the cancer is operable for groups N0 and N1, but it is inoperable for N2 and N3. The variable has been named *N2_N3* (see Figure 1).

The laboratory tests that can be performed are represented by the binary variables *CT_scan*, *TBNA*, *PET*, *EBUS*, *EUS,* and *MED* (the result of the mediastinoscopy). We have also created the variable *MED_Sv*, which represents whether the patient has survived mediastinoscopy.

[2]The TNM classification is a cancer staging system using three factors (T, N and M) to describe the extent of cancer in a patient's body. N factor describes regional lymph nodes that are involved.

Decision variables The set of possible treatments is represented by the variable *Treatment.* Its states are *thoracotomy*, *radio-chemotherapy*, and *palliative.*[3]

The decisions about whether to perform the different laboratory tests have been represented by the variables with the prefix *Decision_* on the name.[4] These decisions forced us to add a new state *no_result* to the variables *TBNA*, *PET*, *EBUS*, *EUS*, and *MED*, to reflect that when we do not perform a medical test its result is not available.

Ordinary utility nodes The quality-adjusted life expectancy (QALE) (Weinstein and Statson, 1977) of the survivors to the medical tests (except the mediastinoscopy) and the treatment is represented by the node *Survivors_QALE.*

The morbidities due to TBNA, EBUS, EUS, and mediastinoscopy, are depicted by *TBNA_Morbidity*, *EBUS_Morbidity*, *EUS_Morbidity*, and *Med_Morbidity* respectively, and measured in QALYs.

Med_Survival indicates whether the patient has survived to the mediastinoscopy.

The probability of survival to the treatment is represented by *Immediate_Survival.*

Super value nodes The ordinary utility nodes presented above have been combined by using super-value nodes (SVNs), as proposed by Tatman and Shachter (1990). SVNs are either of type sum or product. The type of each SVN has been represented by attaching the corresponding sign of sum or product, as shown in Figures 1 and 2.

Nodes *Survivors_QALE* (QALE of the survivors to the medical tests and the treatments) and *Med_Survival* (probability of survival to the mediastinoscopy), have been combined into the product node *Net_QALE.*

Nodes *TBNA_Morbidity*, *EBUS_Morbidity*, *EUS_Morbidity*, and *Med_ Morbidity* have been combined with *Net_QALE* into the sum node *Total_QALE.*

2.1.2 Arcs of the graph

The influence diagram contains four kinds of arcs:

1. **Arcs into chance nodes.** They represent probabilistic dependencies. In our influence diagram,

[3]Other possible treatments are irrelevant from a medical point of view in the scenarios considered in this diagram.

[4]We do not include a node *Decision_CT_scan* because CT scan is always performed to a patient.

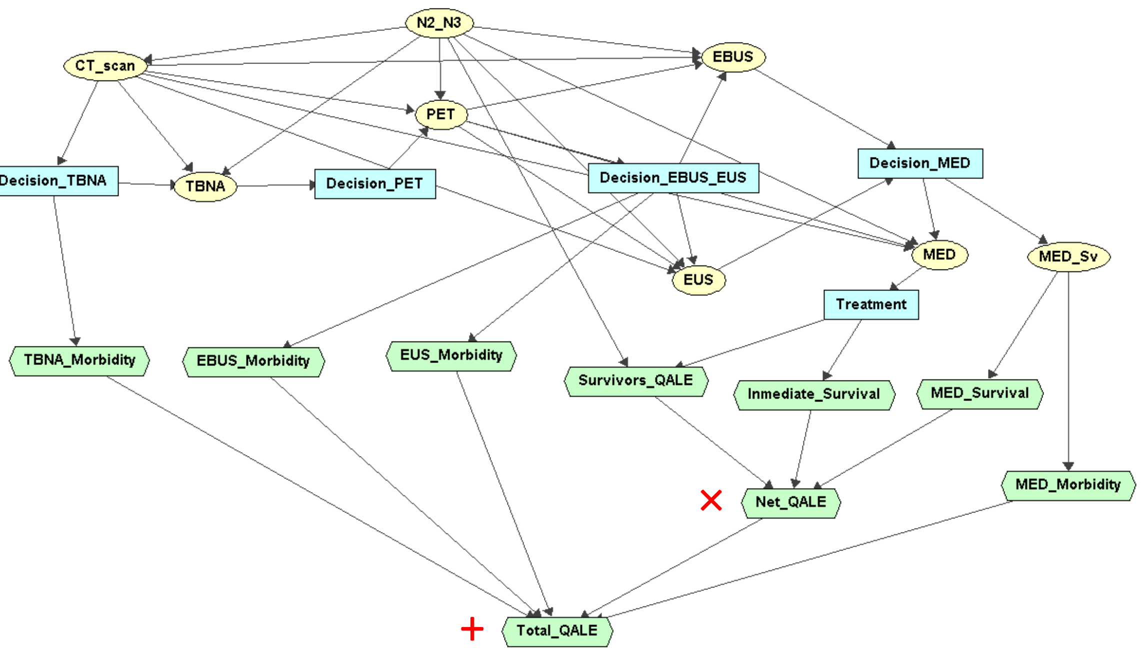


Figure 1: Influence diagram of MEDIASTINET.

an arc from a node representing the decision of a test, such as the arc *Decision_TBNA→TBNA*, indicates that the result (in this case *TBNA*) is only available when we perform the test (*Decision_TBNA=yes*)

2. **Arcs into decision nodes.** They imply informational precedence. Based on the "non-forgetting" assumption (Nielsen and Jensen, 1999), we have not drawn non-forgetting links, to make the influence diagram more clear. For example, the arc *CT_scan→Decision_PET* is not necessary due to the no-forgetting assumption.

3. **Arcs into ordinary utility nodes.** They represent functional dependencies. For example, the arcs into the node *Immediate_Survival* means that the domain of its utility function consists of nodes *N2_N3* and *Treatment*.

4. **Arcs into SVNs.** They indicate the set of utility nodes that are combined into the SVN. For instance, the arcs pointing at the node *Net_QALE* mean that is the combination of *Survivors_QALE*, *MED_Survival* and *Immediate_Survival*.

2.2 PROBABILITIES AND UTILITIES

The quantitative part of the ID consists of a set of probability and utility potentials. For example, for each chance node C we must give a conditional probability potential $p(C|pa(C))$ for each configuration of its parents, $pa(C)$. Then, the table for $p(C|pa(C))$ requires $|dom(C)| \cdot \prod_{X \in pa(C)} |dom(X)|$ numbers, but given the restriction that $\sum_{c} P(c|pa(C)) = 1$, only some of them are independent.

Given that the parameters of MEDIASTINET are not known with precision we attached a probability distribution to each parameter. We identified the type of distribution of each parameter with the expert's help. For the probabilities (prevalence of the disease, the sensitivities and the specificities of tests) we assigned beta distributions. For the utilities (QALE of the survivors to the treatments) we assigned normal distributions.

In spite of the uncertainty of the parameters, the analysis of the optimal strategies requires to focus on a particular model, called *reference case*, in which all the parameters are assumed to be known with certainty. We have assumed that the reference case of MEDIASTINET takes the mean of each numerical parameter as the value in the reference model.

2.3 COST-EFFECTIVENESS AND NET HEALTH BENEFIT

The version of MEDIASTINET presented above does not include the economic cost of the diagnostic tests and the treatments. However, in medical decision making, costs cannot be ignored. Including the economic cost turns the above problem into a multiobjective problem with two attributes: the effectiveness, measured in clinical unit, which we want to maximize, and the economic cost, measured in monetary units, which we want to minimize.

One approach to solve the above problem is based on the concept of *net health benefit* (Stinnett and Mullahy, 1998), defined as follows:

$$NHB = E - C/\lambda = E - \lambda^* C, \tag{1}$$

where E is the effectiveness, C is the cost, λ, sometimes called *willingness to pay*, is used here to convert the effectiveness into a monetary scale, and $\lambda^* = 1/\lambda$. The value of λ depends on each decision maker, it is assumed to be positive, but it is usually unknown.

Other possible solution in the framework of IDs would be to use multi-currency IDs (Nielsen et al., 2007). However, this approach would require to specify two parameters, α_1 and α_2, which act as weights of the efectiveness and the economic cost. We have instead preferred to use the approach based on the NHB*,* besides other reasons (see Section 5), because it only requires one parameter, λ, which has been included explicitly in the ID.

In our model, we identified the effectiveness with the QALE, whose unit is the *quality-adjusted life year* (QALY) (Weinstein and Statson, 1977).

Nevertheless, instead of performing the analysis based on the incremental cost-effectiveness ratios (ICERs) (Gold et al., 1996), which is the standard method, we will apply an equivalent approach: the maximization of the net health benefit, defined in Equation 1. Its integration in Mediastinet is as follows (see Figure 2):

- The cost, C, is represented by the sum node *Total_Economic_Cost*, whose parents represent the economic costs of tests and treatments.
- The effectiveness, E, is depicted by *Total_QALE*, explained in Section 2.1.
- The parameter λ^*, the inverse of λ, is represented by *C2E* (cost to effectiveness).
- *Weighted_Economic_Cost* is a product node standing for $\lambda^* C$.
- *Net_Health_Benefit* represents the NHB (Equation 1).

With regards to the utilities, the economic costs have been attached to normal distributions, and parameter λ was characterized by a log-normal distribution.

If we make $\lambda^* = 0$, the evaluation of the ID returns the strategy that maximizes the effectiveness, without taking into account the economic costs. The medical doctor participating in this study was very interested in knowing this strategy, which turns out to be different from the one obtained with the value of $\lambda = 30,000$ €/QALY, used as a reference point for the Spanish public health system (Sacristán et al., 2002).

This justifies why in our ID we have used λ^* as a parameter in Equation 1 instead of λ: because when looking for the maximum-effectiveness strategy (without caring about the costs), it suffices to make $\lambda^* = 0$, In contrast, making $\lambda = +\infty$ would present computational problems.

3 OPTIMAL STRATEGIES FOR THE REFERENCE CASE

3.1 COMPUTATION AND REPRESENTATION OF THE STRATEGIES

The object of decision analysis on a probabilistic decision problem, represented for example in a decision tree or an ID, is twofold: to determine an optimal strategy, and to compute the maximum expected utility (MEU).

We have computed two strategies for Mediastinet with two different criteria: the maximization of the effectiveness (disregarding costs) and the maximization of the net health benefit. They have been obtained by solving Mediastinet twice: one with $\lambda^* = 0$ (see Eq. 1), and the other one making $\lambda^* = 1/\lambda = 1/30,000$. Changing λ^* in Mediastinet only implies setting the utility node *C2E* to the value of λ^*.

The optimal strategy of an ID contains a policy for each decision. Policies are usually presented in the form of a *policy table*, containing a column for each configuration of informational predecessors of the decision. For example, Figure 3 displays optimal policy for *Decision_PET* of Mediastinet.

However, given that the size of the policy tables grows exponentially with the number of informational predecessors, we felt the need of presenting the optimal policy of each decision in the form of a *policy tree* (see Figure 4). A policy tree (PT) is similar to a decision tree (DT) (Raiffa and Schlaifer, 1961): it consists of chance and decision nodes, and arcs labeled with the states of the nodes. The ancestors of a decision node in the PT are informational predecessors in the ID. Leaves indicate the optimal decision in the corresponding scenario. In contrast with DTs, a PT only represent scenarios that are possible by following the optimal strategy. This reduces enormously the size of the representation and makes it more understand-

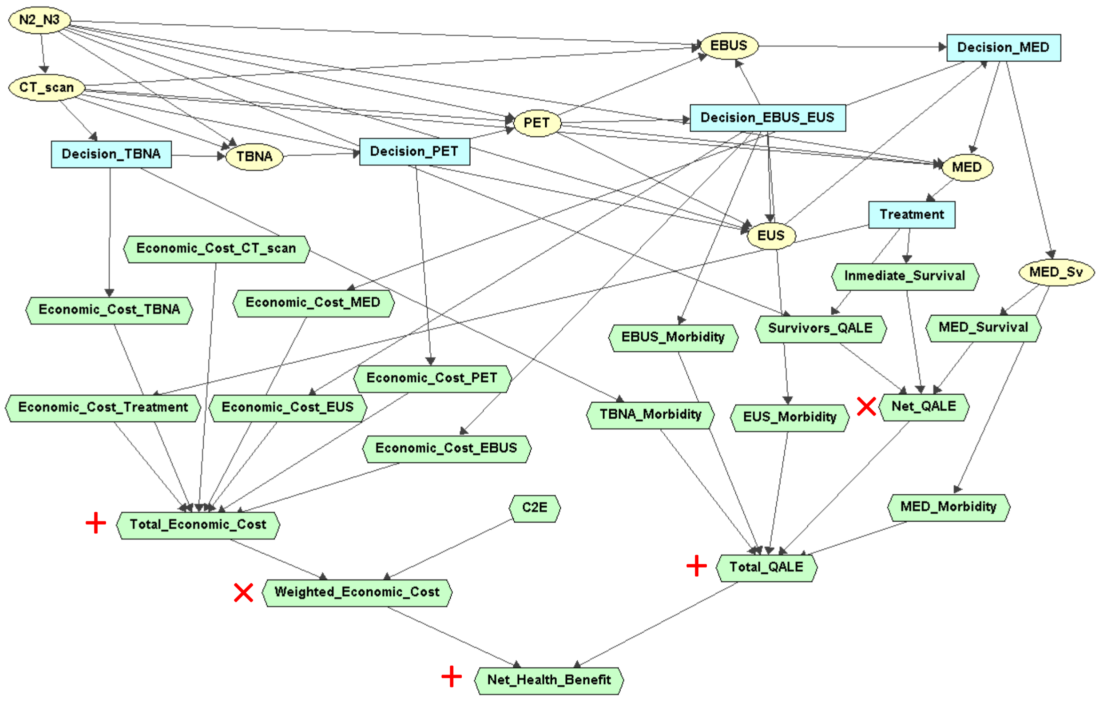
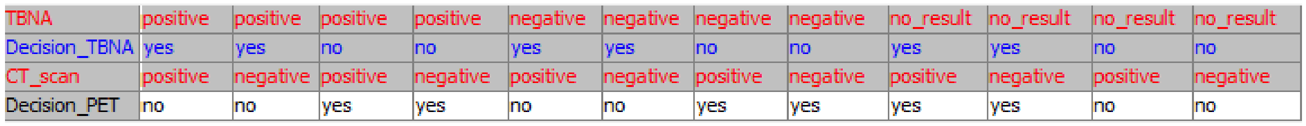


Figure 2: A new version of MEDIASTINET, including economic costs.



TBNA	positive	positive	positive	positive	negative	negative	negative	negative	no_result	no_result	no_result	no_result
Decision_TBNA	yes	yes	no	no	yes	yes	no	no	yes	yes	no	no
CT_scan	positive	negative	positive	negative	positive	negative	positive	negative	positive	negative	positive	negative
Decision_PET	no	no	yes	yes	no	no	yes	yes	yes	yes	no	no

Figure 3: Policy table for *Decision_PET*

able for the medical expert. For example, the policy table for decision *Treatment* of MEDIASTINET has 15,552 columns. In contrast, the PT of *Treatment* in MEDIASTINET when considering costs has 5 leaves (see Figure 4). That PT also represents the entire optimal strategy of the ID.

3.2 SUBJECTIVE EVALUATION OF MEDIASTINET'S STRATEGIES

After obtaining the two optimal strategies we have presented it to the expert to know his opinion about the policies obtained. He said that he would apply a slightly different strategy but he is not sure whether his decisions are better than those recommended by MEDIASTINET. However, the expert's recommendation and MEDIASTINET's agree in that the treatment must be selected depending on the result of the last test performed: If it is positive then apply chemotherapy, otherwise apply thoracotomy.

The expert concluded that the optimal strategies yielded by MEDIASTINET were very reasonable and "logic", and that the system was "quite intelligent."

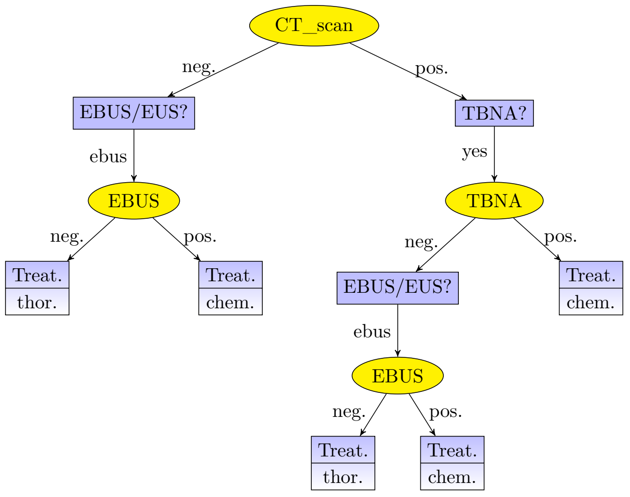


Figure 4: Optimal strategy for MEDIASTINET with economic costs.

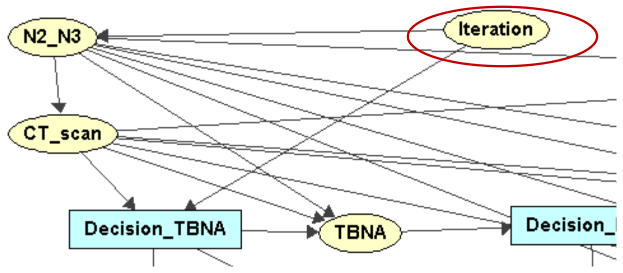


Figure 5: Part of the augmented ID of MEDIASTINET for performing SA of the prevalence of node *N2_N3*.

4 PROBABILISTIC SENSITIVITY ANALYSIS IN MEDIASTINET

After computing the optimal policies and the MEU for the reference case, we investigated whether the results depend on (are sensitive to) the uncertainty in the model. This post-hoc investigation is called *sensitivity analysis* (SA).

4.1 UNCERTAINTY ON THE NUMERICAL PARAMETERS OF MEDIASTINET

We have performed a SA that can be characterized as *quantitative*, *probabilistic*, *multi-one-way*.[5]

It is *quantitative* because we only consider variations in the numerical parameters and do not vary the structure of the ID. It is *probabilistic* because we have a probability distribution for each parameter.

It is *multi-one-way* because we consider the individual variations of a set of parameters, as for example in a tornado diagram.

Depending on the effects analyzed, *value SA* measures variations in the EU, and *decision SA* explores the changes in the optimal strategy.

For the SA we have built an *augmented ID* for each parameter. For example, Figure 5 shows the augmented ID for performing SA of the prevalence of *N2_N3*, identical to MEDIASTINET except that we have added the node *Iteration* and two arcs: one to the node affected by the parameter, *N2_N3*, and another to the first decision of the ID. Because of the non-forgetting hypothesis, this link implies that we will obtain the optimal strategy for each value of the parameter under study, and we can determine whether it is the same as the optimal strategy for the reference case. It also allows us to calculate the expected value of perfect information (Felli and Hazen, 1998).

[5]A complete definition of the characterizations of SA in IDs can be found in (Nielsen and Jensen, 2003).

All the chance and decision variables in MEDIASTINET are discrete. Each continuous distribution has been discretized by taking 100 points of an interval of the domain of the parameter. The intervals partitioned for normal and log-normal distributions of parameters μ and σ^2 have been $[\mu - k \cdot \sigma, \mu + k \cdot \sigma]$ and $[e^{\mu - k \cdot \sigma}, e^{\mu + k \cdot \sigma}]$ respectively, by using $k = 3.5$, which accumulates 99.953 % of the probability mass. We have taken the entire interval $[0, 1]$ when discretizing beta distributions.

4.2 RESULTS OF THE SA

We recorded three metrics of analysis:

- the thresholds of policy change, which define a set of intervals, contained in the domain of the parameter, where the optimal strategy is identical to the reference case,
- the expected value of perfect information (EVPI), very well-known in SA literature (Felli and Hazen, 1998), and
- the sensitivity of each decision to each parameter, which analyzes the probability of change in the optimal policies when the parameter varies.

4.2.1 Thresholds of policy change

Our analysis has shown that most of the parameters have a wide range of variation where the optimal policies do not change, and thus the optimal strategy is very robust. However, there are some exceptions, such as the sensitivity of CT scan. Its value in the reference model is 0.51, but its policy change thresholds are given by the interval $[0.41, 0.574]$. It means the value of sensitivity of CT scan in the reference model is not very far from the thresholds, and some policy might change if the value of the parameter varies.

4.2.2 EVPI

Most of the parameters present very small values of EVPI. The parameter with the highest EVPI is λ, as we expected. Thus, knowing its value with certainty would have a high impact on the MEU of the ID.

4.2.3 Sensitivity of each decision to each parameter

We have also observed that decisions are not sensitive to the variations of the parameters in most cases, which indicates that the optimal strategy is very robust.

The three parameters with highest probability of changing the optimal policies are: (1) the QALE of the survivors to the thoracotomy when there is no metastasis; (2) the sensitivity of the TBNA when the result of CT scan is negative; and (3) λ. The only parameter that affects the policies of all the decisions is λ.

Finally, the decisions more affected by the variations on the parameters are *Decision_TBNA* and *Treatment.*

The main conclusion of the SA is that there are only two parameters that can have significant impact on the strategy: the QALE of the survivors to the thoracotomy when there is no metastasis and λ. Even though the former is the parameter with the highest impact on a decision (*Dec_TBNA*), the parameter that reflects to have more overall impact in the strategy is λ.[6]

5 RELATED WORK

Our model Mediastinet has several differences with the ID for the mediastinal stating of NSCLC built by Nease and Owens (1997):

1. Mediastinet assumes that a CT scan is always performed.
2. Four new laboratory tests have been included, namely *TBNA*, *PET*, *EBUS*, and *EUS*, as well as the decisions about whether to perform them.
3. Mediastinet considers the morbidities of the tests.
4. In Mediastinet the results of CT scan and PET influence the sensitivity and specificity of the other tests.
5. Palliative care is a possible treatment.
6. The economic costs of tests and treatments and λ (willingness to pay) are represented in Mediastinet.

As a result, Mediastinet is much bigger and more complex than the model of Nease and Owens (1997). For example, the decision table of the treatment has a domain of 72 columns in their model, while it contains 15,552 scenarios in Mediastinet.

Nease and Owens (1997) also built an ID that analyzes any arbitrary sequencing of CT scan and mediastinoscopy. In contrast, the order of decisions has been set in Mediastinet because the dependence relations of the test results in the problem are quite difficult to analyze in an ID with partial order and would need additional expert help. This would require a hard work of elicitation because the result of a test in our model can influence the sensitivity and specifity of future tests. For example, if the result of CT scan is positive then it also gives valuable information about where the doctor has to stick in the needle during the TBNA. However, that information is not available if the TBNA is performed before the CT scan.

We discarded the use of multi-currency IDs (Nielsen et al., 2007) for representing and solving the problem because that approach is a bit more difficult to be understood by a medical expert and there are no software tools with explanation capabilities for multi-currency IDs. In contrast, explanation capabilities of Elvira system for IDs (Lacave et al., 2007) have been quite useful while building and debugging the model with the expert's help.

Although quantitative SA has also been studied by Nielsen and Jensen (2003), the main preliminary steps in PSA in IDs can be found in (Felli and Hazen, 1998) and (Bielza et al., 2000). However, they do not consider the computation of the thresholds of policy change and the sensitivity of each decision to each parameter.

6 CONCLUSIONS AND FUTURE WORK

We have built an ID, Mediastinet, for the mediastinal staging of NSCLC.

From a medical point of view, there is a vivid debate among specialists about which technologies should be used for the mediastinal staging of NSCLC, and it is not possible to arrive at a consensus (Fritscher-Ravens et al., 2003; Schimmer et al., 2006). For this reason, Mediastinet is very useful as a decision analysis tool that combines objective data and subjective estimates and may show whether the discrepancies are due to differences in the numerical parameters used by each expert or to a wrong estimation of the consequences of each policy.

From the perspective of IDs, we have proposed a method for finding a tradeoff between cost and effectiveness, based on λ, the willingness to pay, which is also used in cost-effectiveness analyses. This parameter has been included explicitly in our model, which allows us to modify its value easily.

Additionally, we have performed a probabilistic sensitivity analysis that has studied three metrics, one of them is very well known (EVPI), and the others are new: the probability of change in the optimal strategy, and the intervals of the parameters where the optimal policies do not change. We have used for each param-

[6]The overall impact in the strategy is calculated as the average impact on each of the decisions of the ID.

eter the most appropriate distribution: beta, normal, or log-normal. We have proposed efficient methods for recording the three metrics when analyzing the variations of all the parameters on an ID of considerable size such as Mediastinet.

Finally, due to the interest of the expert in considering the possibility of having partial orderings of the decisions, unconstrained IDs (Jensen and Vomlelová, 2002) are a future research topic line. Decisions were totally ordered in Mediastinet because tests are not independent given that sensitivities and specificities can be influenced by other tests, as we explained above. A partial order would require a hard work of elicitation for every possible ordering.

The expert would also like to include in the model the possibility of repeating some decisions, which is known as *restaging*.

Acknowledgements

This work has been supported by the Department of Education of Madrid and the European Social Fund under a doctoral grant, and by the Spanish Ministry of Education of Science under grants TIN2006-11152 and TIN2009-09158.

References

Bielza, C., Ríos-Insua, S., Gómez, M., and del Pozo, J. A. F. (2000). Sensitivity analysis in Ictneo. *Lecture Notes in Statistics*, 152:317–334.

Felli, J. C. and Hazen, G. B. (1998). Sensitivity analysis and the expected value of perfect information. *Medical Decision Making*, 18(1):95–109.

Fritscher-Ravens, A., Bohuslavizki, K. H., Brandt, L., Bobrowski, C., Lund, C., Knofel, W. T., and Pforte, A. (2003). Mediastinal lymph node involvement in potentially resectable lung cancer. *Chest*, 123(2):442–451.

Gold, M. R., Siegel, J. E., Russell, L. B., and Weinstein, M. C. (1996). *Cost-Effectiveness in Health and Medicine.* Oxford University Press, New York.

Howard, R. A. and Matheson, J. E. (1984). Influence diagrams. In Howard, R. A. and Matheson, J. E., editors, *Readings on the Principles and Applications of Decision Analysis*, pages 719–762. Strategic Decisions Group, Menlo Park, CA.

Jensen, F. V. and Vomlelová, M. (2002). Unconstrained influence diagrams. In *Proceedings of the Eighteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI'02)*, pages 234–241, San Francisco, CA. Morgan Kaufmann.

Lacave, C., Luque, M., and Díez, F. J. (2007). Explanation of Bayesian networks and influence diagrams in Elvira. *IEEE Transactions on Systems, Man and Cybernetics—Part B: Cybernetics*, 37:952–965.

Lloyd, C. and Silvestri, G. A. (2001). Mediastinal Staging of Non-Small-Cell Lung Cancer. *Cancer Control*, 8(4):311–317.

Nease, R. F. and Owens, K. D. K. (1997). Use of influence diagrams to structure medical decisions. *Medical Decision Making*, 17(3):263–275.

Nielsen, S. H., Nielsen, T. D., and Jensen, F. V. (2007). Multi-currency influence diagrams. In Salmerón, A. and Gámez, J. A., editors, *Advances in Probabilistic Graphical Models*, pages 275–294. Springer, Berlin, Germany.

Nielsen, T. D. and Jensen, F. V. (1999). Welldefined decision scenarios. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI'99)*, pages 502–511, San Francisco, CA. Morgan Kaufmann.

Nielsen, T. D. and Jensen, F. V. (2003). Sensitivity analysis in influence diagrams. *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, 33:223–234.

Raiffa, H. and Schlaifer, R. (1961). *Applied Statistical Decision Theory.* MIT press, Cambridge.

Sacristán, J. A., Oliva, J., del Llano, J., Prieto, L., and Pinto, J. (2002). ¿Qué es una tecnología sanitaria eficiente en España? *Gaceta Sanitaria*, 16:334–343. In Spanish.

Schimmer, C., Neukam, K., and Elert, O. (2006). Staging of non-small cell lung cancer: clinical value of positron emission tomography and mediastinoscopy. *Interact CardioVasc Thorac Surg*, 5(4):418–423.

Stinnett, A. A. and Mullahy, J. (1998). Net health benefits: A new framework for the analysis of uncertainty in cost-effectiveness analysis. *Medical Decision Making*, 18:S68–S80.

Tatman, J. A. and Shachter, R. D. (1990). Dynamic programming and influence diagrams. *IEEE Transactions on Systems, Man, and Cybernetics*, 20:365–379.

Weinstein, M. and Statson, W. (1977). Foundations of cost-effectiveness analysis for health and medical practices. *New England Journal of Medicine*, 296:716–721.

Using Tree Augmented Naïve Bayes Classifiers to Improve Engine Fault Models

Daniel L.C. Mack
EECS Dept.
Vanderbilt University
Nashville, TN 37212

Gautam Biswas
EECS Dept.
Vanderbilt University
Nashville, TN 37212

Xenofon D. Koutsoukos
EECS Dept.
Vanderbilt University
Nashville, TN 37212

Dinkar Mylaraswamy
Honeywell Aerospace
1985 Douglas Drive N
Golden Valley, MN 55422

Abstract

Online fault diagnosis is critical for detecting and mitigating adverse events that arise in complex systems such as aircraft, automobiles, and industrial processes. A typical fault diagnosis system consists of a reference model that mathematically links diagnostic monitors providing partial evidence to potential fault hypotheses. A reasoning algorithm operated on this model uses a set-covering scheme to establish likely fault candidates and their rankings. However, incompleteness in the reference model and simplifying assumptions affect the accuracy of the reasoning algorithms. In this paper, we describe a Tree Augmented Naïve Bayes Classifier (TAN) approach to systematically extend a reference model structure using data from system operations. We compare the performance of the TAN models against a typical reference model, and demonstrate that the TAN improves classification accuracy by finding new causal links among the system monitors.

1 Introduction

Aircraft are complex systems containing several interacting components and subsystems such as propulsion, electrical, flight management, avionics, and bleed subsystems. Smooth and integrated operation of these subsystems is essential to keep the aircraft operating safely. However, any operating system degrades over time and monitoring the system online for detecting the onset of unfavorable conditions and intrinsic faults is essential for increasing aviation safety.

The current state of online fault diagnosis is focused on installing a variety of sensors onboard an aircraft along with reasoning software to automatically interpret the evidence generated to explore the presence of faults. One such state-of-the-art system is the Aircraft Diagnostic and Maintenance System (ADMS) (Spitzer, 2007) that is used on the Boeing B777. The ADMS uses an expert-derived fault propagation model, called the *system reference model* that captures the interactions between aircraft components under various operating modes. Generation of this reference model is a manual process and often the step results in incompleteness and inaccuracies in the development and deployment of an ADMS.

Some of the incompleteness and inaccuracies can be overcome as the engineering teams acquire additional knowledge from an operating fleet, and generate heuristics rather than a systematic upgrade to the original reference model. In other words, a gap exists for systematic upgrades and increments to the reference model even though vast amount of operational data is collected by operating airlines. Closing this gap using advances in data mining methods is the focus of this paper. We describe a specific data mining approach for augmenting an existing aircraft engine reference model as an alternative to ad hoc approaches. We demonstrate the effectiveness of our work on data generated from a realistic aircraft engine simulator.

Statistical analysis and designing classifiers for discovering knowledge from real-world data has been studied extensively. For example, Witten (Witten & Frank, 1999) describes several data mining approaches for producing black box models. Unfortunately, such models are very difficult to verify, making them almost impossible to certify for airworthiness. Further, the lack of transparency in these models makes it difficult to *append* this new knowledge to existing ADMS reference models. For practical purposes, data mining approaches for aircraft reference models have to "build upon" existing model structures rather than create something new, which will incur considerable engineering overhead cost.

The proposed approach to combing data mining with fault models is somewhat unique. The data mining does not start from a clean slate, but builds up from an existing ADMS reference model structure. In section 2, we describe a typical reference model structure along with the reasoning algorithm (called the W-algorithm). Next, we systematically enumerate the missing or partially correct information in this state-of-the-art reference model. These gaps formalize the data mining problem described in Section 3. We discuss the use of Tree-Augmented Naïve Bayes Networks (TANs) as a data driven modeling structure for diagnosis with causal probabilistic models in section 4. The data mining approach is illustrated using data from a high fidelity simulator. Section 5 discusses the CMAPS-S simulator and the data selection task for our experiments. Section 6 describes the experimental results using the CMAPS-S data set, with a comparison of a Naïve Bayes classifier that replicates a system reference model against a TAN classifier model derived from a learning algorithm. Metrics are defined for evaluating classifier performance, and a number of different experiments are run to examine the addition of evidence to these models. Section 7 presents a summary of our approach, and outlines our directions for future work for diagnostic and prognostic reasoning using the data mining algorithms.

2 Reference Models and Reasoning

Model-based strategies for diagnosing large, complex, real-world systems rely on domain experts to craft the *reference models* used for monitoring and isolating faults. The complexity of the system makes it almost impossible to create complete physics-based models with reasonable resources. A more pragmatic solution is to rely on expert-generated cause-effect models. In simple terms, the reference model of the system being monitored can be represented as a bipartite graph consisting of two types of nodes: failure modes and evidence. The set F defines all *distinct* failure modes defined for the system under consideration. A failure mode $fm_i \in F$ may be present or absent in the system. This is defined as the state of the failure mode. In the primary model, we allow only binary (occurring or not-occurring) states for the failure mode. We use the following shorthand notations regarding these assertions.

$$\begin{aligned} fm_i = 0 &\Leftrightarrow \text{The failure mode is not present} \\ fm_i = 1 &\Leftrightarrow \text{The failure mode is present} \end{aligned} \tag{1}$$

Every failure mode has an a priori probability of occurring in the system. This probability is denoted by $P(fm_i = 1)$. A failure mode fm_k can occur (or not occur) independently of another failure mode fm_j occurring, that is, $P(fm_k = 1|fm_j = 1) = P(fm_k = 1)$.

To isolate and disambiguate failure modes, the model also defines an entity called "evidence". The jth evidence is denoted by e_j and the set E denotes all distinct monitors defined for the system under consideration. The diagnostic monitor associated with the ith evidence can either *indict* or *exonerate* a subset of failure modes called its *ambiguity group*. The monitor m_i can take three mutually exclusive values allowing a monitor to express indicting or exonerating or unknown support for the failure modes in its ambiguity group. The notations are described in equation (2).

$$\begin{aligned} m_i = 0 &\Leftrightarrow \text{Exonerating evidence} \\ m_i = 1 &\Leftrightarrow \text{Indicting evidence} \\ m_i = -1 &\Leftrightarrow \text{Unknown evidence} \end{aligned} \tag{2}$$

Ideally, we want a monitor associated with evidence e_i to fire only when the failure modes in its ambiguity group are occurring. Given the fact that the ith failure mode is occurring in the system, d_{ji} denotes the probability that there will be a monitor providing an indicting evidence under this condition.

$$d_{ji} = P(m_j = 1|fm_i = 1), \tag{3}$$

d_{ji} is called the detection probability of failure mode monitor fm_j with respect to the ith evidence. A monitor may fire when there is no failure mode present in the system. False alarm probability is the probability that an indicting monitor is present when there are no failure modes occurring in the system. That is,

$$\epsilon_i = P(m_i = 1|fm_j = 0, \forall fm_j \in F) \tag{4}$$

A reference model describes the relation between failure modes and monitors. The reference model is a 6-tuple defined as: $[\,E, F, D, Pr, \epsilon\,]$,where: E is evidence set, F is failure mode set, D is detection probabilities, Pr is a priori probability of failure modes, ϵ is false alarm rate for monitors.

Figure 1 illustrates an example reference model graphically, with fault modes (hypotheses) as nodes on the left, and diagnostic monitors (DM) on the right. Each link has an associated detection probability, i.e., conditional probability $P(m_j = 1|fm_i = 1)$. In addition, fault nodes on the right contain the a priori probability of fault occurrence, i.e., $P(fm_i)$. Probabilities on the DM nodes indicate the likelihood that a particular monitor would indicate a fault in a nominal system, which as defined above is ϵ_i. Bayesian methods are employed to combine the evidence provided by multiple monitors to estimate the most likely fault candidates.

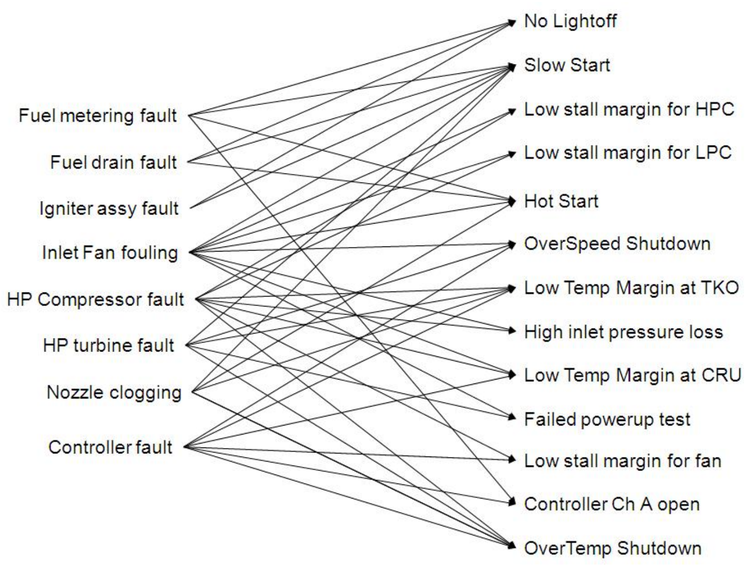


Figure 1: Example Reference Model

The reasoner algorithm (called the W-algorithm) combines an abductive reasoning algorithm with a forward propagation algorithm to generate and rank possible failure modes. This algorithm operates in two steps: (1) *Abductive reasoning step*: Associated with each DM is an *ambiguity set*, $AG = \{fm_1, fm_2, \cdots fm_k\}$. This step assumes that the firing of the DM implies at least one of the faults in the ambiguity set has occurred; and (2) *Forward reasoning step*: For each fm_i belonging to the AG, we extract other DMs that support fm_i. We call this set the supporting DMs, or the monitors of interest, i.e., $S - DM_i$ for fm_i. As these additional monitors fire, fm_i without that monitor in $S - DM_i$ are removed from the AG. Over time as the monitors fire, AG reduces in size, and ideally, to a single fm_i. Additional details about the reasoning algorithm is described in (Honeywell, 2010).

The reasoning algorithm generates multiple single fault hypotheses, each hypothesis asserting the occurrence of exactly one failure mode in the system. The basic probability update rules assume independence of monitor firing events. In other words, $P(m_j, m_k|fm_i) = P(m_j|fm_i)\ P(m_k|fm_i)$ for all monitors m_j and m_k. The two independence assumptions on: (1) Fault modes, and (2) monitors implies that the reasoning algorithm treats the reference model as a set of Naïve Bayes classifiers. The direct correspondence between the reference model for diagnosis and the simple Bayesian structure provides opportunities to use a class of generative Bayesian model algorithms to build these model structures from data and enhance the existing structures produced by a domain expert.

This reasoning algorithm assumes the DMs used in the reference model are strictly binary. The DMs are often derived by applying a threshold to other real valued features known as condition indicators(CIs). These CIs are built as functions of sensors to provide more information about the health of the system. The thresholds applied to create DMs are selected by a domain expert. Data collected from these systems more often contain raw sensors and the CIs rather than the DMs. This creates an issue when trying to examine structures built with data and comparing them to the expert crafted models. Our approach utilizes the idea of the abstracted CIs when constructing models from data. Models built with data and containing CIs or other select sensors are only missing the thresholding, and as such, when the the probabilities are calculated, a Naïve Bayesian model is in essence approximating the reasoning algorithm above. No fault modes are removed from consideration, but the probabilistic ranking of all failure modes will render many with a probability at or near 0. The inference used in Bayesian networks is calculated in the context of discretized values (Conditional Probability Tables). Any necessary discretization of these values is providing a thresholding that acts similar to the reasoning algorithm on an expert model. We believe these similarities are enough to warrant comparisons in the analysis of our results. We utilize this similarity in computation of learned models and their metrics for evaluation.

3 The Data Mining Problem

The reasoning algorithm may not reduce the ambiguity group to a single fault element. For example, all of the evidence (i.e., DMs) required to isolate the single fault may not fire, leaving the size of the ambiguity set to be greater than 1. In this case, the reference model is incomplete. This gap can be addressed by employing heuristic rules or systematically discovering new diagnostic monitors from vast amount of historical data.

A second source of error arises from the "independence assumptions". The assumption of independence between (1) different pieces of evidence and (2) different fault modes may lead to certain hypotheses being assigned higher likelihood than the evidence truly implies. This assumption is made primarily because, causality (or correlation) between evidence in the system is not easily discernible while the system is being designed and assembled. Furthermore, deriving conditional probability tables with joint probabilities such as when nodes have multiple parents is a difficult task for human experts, and can be derived from data. Therefore, the knowledge required to overcome the simplifying (but erroneous) assumptions of independence are best derived by analyzing data from an operating fleet.

As implied above, the reference model that does not make the simplifying independence assumptions can be interpreted as a Noisy-OR classifier, which is a simplified form of a standard Bayes Network. A number of Machine Learning techniques for building Bayes networks from data have been reported in the literature (Friedman, Geiger, & Goldszmidt, 1997) We have studied a number of these approaches in the framework of diagnostic and prognostic reasoning. Among the important considerations have been the notion of independence among the monitors that support the diagnostic reasoning. Our choice for a Bayesian model and for the data mining algorithms that build these models has been guided by:

1. The data mining algorithms should be designed to provide information that supplements existing expert-generated reference models. It is very important that the experts be able to interpret the results of the data mining algorithms, and characterize them as:
   (a) new relations between monitors and fault hypotheses that will improve the reference model;
   (b) additional monitors (both simple and advanced) that help differentiate and provide support for specific diagnostic hypotheses;
   (c) refinements to the conditional probability values between hypotheses and monitors.
2. The computational complexity of the data mining algorithms should be manageable, so that they can be used as exploratory analysis tools by the domain experts. We envision a successive refinement process, where the expert requests a sequence of experimental runs, each built from their observations and interpretations from previous results generated by the algorithms. They can interpret the causal relations between faults and monitors, and discover the dependence among the monitors for different fault situations. The expert may also consider different analysis scenarios to estimate methods for increasing the accuracy (while reducing false positives) in the diagnostic reasoner.

After considering these factors and staying within the Bayes net paradigm, we selected Tree Augmented Naïve Bayes(TAN), a model that could address the factors in a reasonable fashion, as well as challenge the independence assumption in limited ways.

4 Data Mining with Tree Augmented Naïve Bayes Networks

The choice of the data driven techniques to apply to a particular class of problems is very much a function of the nature of the data and the problem(s) to be solved using the data. For example, using data we can systematically test and relax the independence assumptions employed in the reference model, especially if it is useful for diagnosis. There are several interesting alternatives, but one that fits well with our reference model structure is the Tree Augmented Naïve Bayes (TAN) Method (Friedman et al., 1997). The TAN structure is a simple extension of the Naïve Bayes network. Like Naïve Bayes, the root node is the class node, corresponding to one or more fault modes, is causally connected to every evidence (monitor) node. In addition, the TAN structure relaxes the assumption of independence between the evidence nodes, and allows most evidence nodes to have a second parent, which can be a related evidence node. This maintains the directed acyclic graph requirements and produces a tree that captures relationships among the monitors. Generation of this structure is not as computationally expensive as a general Bayesian network.

An example TAN structure is illustrated in Figure 2. The class node is the fault hypothesis under consideration. The other nodes represent supporting evidence for the particular fault hypotheses. In this structure, the only node connected to the class node, is the root observational node. Dependencies among the monitors are captured as additional causal links in the TAN structure.

The TAN Structure can be generated in several dif-

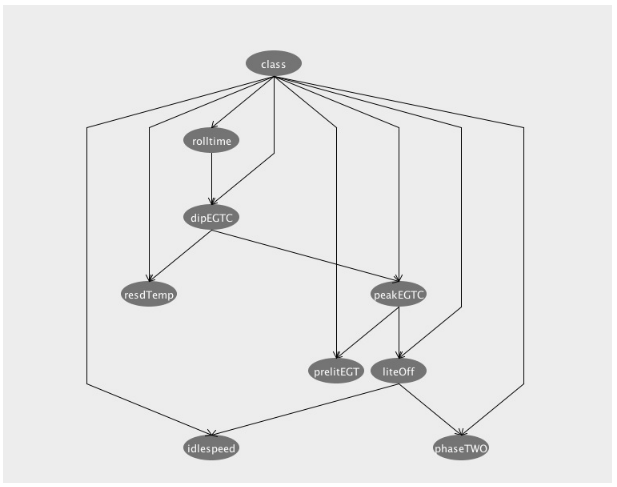


Figure 2: Example TAN Structure

ferent ways that includes (1) a *greedy search* with the constraint that *illegal edges* (i.e., a node having more than one parent from the evidence nodes) are disallowed (Cohen, Goldszmidt, Kelly, Symons, & Chase, 2004); and (2) a *Minimum Weighted Spanning Tree* (MWST) approach that builds a minimum spanning tree to capture the dependencies among monitors, and then connects the class (fault mode) to all of the monitor nodes (Friedman et al., 1997). In either case, a decision has to be made about the monitor node to use as the observational root node in the derived tree structure. The derived TAN structure is static, i.e., it does not include explicit temporal information through causality.

A standard algorithm (e.g., Kruskal's algorithm (Kruskal, 1956)) is applied to generate the MWST. The edge weights of the MWST structure are a log likelihood function, e.g., Bayesian value (Chickering, Heckerman, & Meek, 1997) or the Bayesian Information Criterion (BIC) (Schwarz, 1978). The Bayesian likelihood metric is preferred for discrete data, wheras the BIC measure works better for continuous distributions. The algorithm we use calculates the BIC value for every pair of evidence nodes (note that directionality matters, therefore, for nodes A and B, two BIC values are computed from A to B and B to A). The values are stored in a matrix, which facilitates the application of Kruskal's algorithm to generate the MWST.

The MWST version of this algorithm is implemented in the data mining toolkit called Weka (Hall, Eibe, Holmes, Reutemann, & Witten, 2009) It does not handle continuous features, and instead uses a discretization algorithm to bin each of the features into sets that best discriminate among classes. This produces better classifiers, but it may create very fine splits for features that result in excessive binning(thus building very large conditional probability tables).

5 The CMPAS-S Data

The CMAPS-S data set is generated from a simulator developed at NASA's Glenn Space Center (Frederick, DeCastro, & Litt, 2007). The engine simulator takes into account the wear and tear on a turbine engine over multiple flights, and it can produce data for a number of sensors for climb, cruise, and descent modes of operation. The simulator parameters can be set to run in nominal and faulty modes of operation.

As a first step, we select appropriate sensor measurements as features and transform them into a sequence of values for the data mining task. Since the reference model structure and the reasoner do not directly include temporal information, the data is separated into the different modes of operation. For this study, all of the data for fault analysis was extracted from the cruise mode of operation. In this mode, most sensor values remain steady, except for measurement noise. Therefore, for this study each flight was represented as a datapoint consisting of a vector of sensor values, and the entire dataset was made up of n data points corresponding to n flights.

Table 1 shows the different features in the CMAPS-S data set. Some features are marked as a "condition indicator"(CI), which is a term for complex features that can be used to indicate when an engine is experiencing abnormal behavior. A threshold on these values would produce the *health indicator* (also called a diagnostic monitor, DM) that a reference model would relate to a fault mode.

The reference model as defined above is in terms of DMs which in this data would be HIs. Since the data contains only the CIs for the engine and an expert crafted reference model was unavailable, we used a Naïve Bayes structure based on CIs as the "base" reference model. This represents an approximation, but the approximation is a good one. As mentioned, experts avoid complex relationships in these models (such as between monitors and faults) they often implicitly assume independence. We find a close approximation of this as a Naive Bayes classifier.

The rest of the features extracted from the data represent the sensors, and thus, features that would most likely be available in data from other complex systems of this nature. These features are selectively added to determine if the reasoner can generate more accurate results with the added information and the refined structures that the learning algorithm generates.

The CMAPS-S data was generated in a way that the

Sensor	Notes
Altitude	$\mathbb{R}$, unit is feet
Mach Number	$\mathbb{R}$, the unit is Mach
Throttle Angle	$\mathbb{R}$, measured in degrees
Fuel Flow	$\mathbb{R}$, measure in percent
Stall Margin of HPC	CI
Stall Margin of LPC	CI
Stall Margin of Fan	CI
Temp. of High Pressure Turbine	$\mathbb{R}$, measured in Centigrade
Temp. of the Fan Inlet	$\mathbb{R}$, measured in centigrade
Temp. of the Low Pressure Turbine	$\mathbb{R}$, measured in centigrade
Pressure of Fan Inlet	$\mathbb{R}$, measured in PSI
Phys. Fan Speed	$\mathbb{R}$, measured in RPM
Phys. Core Speed	$\mathbb{R}$, measured in RPM

Table 1: Sensor values and Monitors (Conditional Indicators) for the CMAPS-S Engine Data

fault(s) and their time of introduction was known, so it was easy to assign *nominal* and *faulty* labels for each data stream. The CMAP-S data models three faults: (1) a fan fault (Fan), (2) a High Pressure Compressor fault (HPC), and (3) a High Pressure Turbine fault (HPT). The reference model for the three faults could be constructed in different ways. For example, one could construct three different models – each model defining a classifier that differentiated a fault condition from nominal behavior. Another possibility was to treat the model building as a multi class learning problem. The result would be a single classifier structure that distinguished between four hypotheses that included the three faults and nominal operations. This structure as the model would likely produce insights on how to differentiate between several faults hypothesis. Given that we were adopting an exploratory framework to study the effectiveness of different classifier models, it made sense to compare between different classifier structures and analyze the discriminating evidence provided by each model. Furthermore, the availability of the CMAPS-S simulator facilitated this approach, since in real situations it may be hard to collect sufficient amounts of fault data to build robust classifiers that include multiple fault hypotheses.

6 Experiments

To initially evaluate the ability of the data mining techniques to improve over the Naïve Bayes based reference models, we have conducted and evaluated a set of experiments using the data from the CMAPS-S engine system to establish whether the TAN-based model produces a better diagnostic classifier than a reference model that is implemented as a Naïve Bayes Classifier. Our experiments compare the performance results of the Naïve Bayes versus the TAN models.

In the CMAPS-S data, we utilize two feature sets. The first experiment uses the feature set defined as the baseline reference model(only CIs), and extracts a classifier structure by running our machine learning algorithms. The next experiment adds additional sensors to the baseline that are not conditional indicators, to see if using these sensors can improve diagnostic accuracy while reducing false alarms.

A systematic study of the performance of the algorithms requires running of n-Fold Cross Validation experiments. Dividing the data into n equally sized and distinct sets of samples, each with the balance of classes maintained as in the original set allows for the creation of $n-1$ training sets with the last set being held out as the test set. This is done n times, and the metrics generated are then averaged over each of the n runs. This experimental style helps test the robustness of the classifier and keeps the metrics from being overly optimistic or pessimistic depending on the random construction of one hold out set. The experiments include: (1) derivation of models for the individual faults, and (2) derivation of a model for the multi-fault case. The metrics reported in Tables 2 and 3 are the average of 10-Fold Cross Validation runs.

6.1 Experimental Results

The data generated for the experimental study included the three faults discussed previously, and the analysis was conducted in the cruise mode with the aircraft flying at an altitude of 35,000 feet. The data mining algorithms were run to derive individual models for the three single fault modes, as well as a combined model with all three faults. Tables 2 and 3, summarize our experimental results in terms of the accuracy metrics, i.e., overall accuracy (Acc), false positives (FP), and false negatives (FN).

6.1.1 Experiment 1

The Naïve Bayes model with only the CIs represents the reference model for analysis of core engine anomalies. The TAN structure with additional causal relations results in a model with better accuracy. The results in Tables 2 and 3 demonstrate that the TAN Structure for the FAN Fault and the multi-fault classifier have higher accuracy. Their superior performance shows that even with a small number of fea-

	Fan			HPC			HPT			All Three		
	Acc	FP	FN	Acc	FP	FN	Acc	FP	FN	Acc	FP	FN
Naïve Bayes Network	67.9	15.4	36.7	71.4	0	35.3	94.2	0	9.3	82.1	15.5	19.6
TAN	99.4	0.4	0.7	80.8	36.7	0	94.7	8.9	2.9	97.4	1.1	3.8

Table 2: Cruise Mode: Model with Only Conditional Indicators

	Fan			HPC			HPT			All Three		
	Acc	FP	FN	Acc	FP	FN	Acc	FP	FN	Acc	FP	FN
Naïve Bayes Network	68.8	12.5	49.5	72.9	0	56.7	93.8	3.6	9.9	84.9	1.1	23.2
TAN	99.8	0	0.4	87.96	23.0	0	96.6	5.4	0.5	98.0	0.8	0.7

Table 3: Cruise Model: Model with Conditional Indicators + Sensor Measurements

tures(3), introduction of two new causal links, the results improved considerably(67.9% to 99.4% for the Fan and 82.1% to 97.4% for multi-fault). Figure 3 shows the representative TAN used in the multi-fault scenario(the NB Model on the right is for comparison). The CI corresponding to stall margin for the Low Pressure Compressor provided the best discriminating evidence between different faults when only conditioned by the class variable. For the single fault classifiers, the Fan and HPC TANs outperformed the Naïve Bayes, but the HPT classifier provided minimal improvement. The HPT Classifier seems to require a simple classifier and both models achieved over 90% accuracy. The classifiers for the HPC fault were the lowest performing set. Although the TAN did better than the NB classifier by over 8%, this would indicate that the reference model for the engine may not be able to detect and isolate this fault, particularly from cruise data.

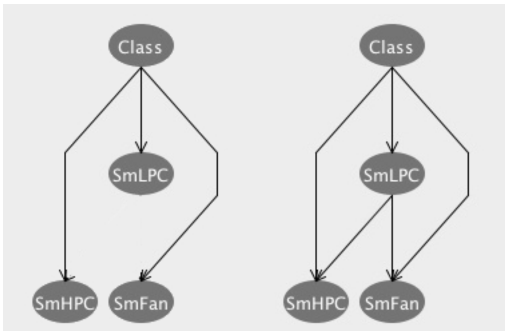


Figure 3: NB Model on the left and the TAN Model on the right for the Multi-Fault Scenario with Only CI

6.1.2 Experiment 2

For the second set of experiments, we consider the additional sensors. From Table 3, there is an improvement in the accuracy numbers for all of the TAN models. This is highlighted by the HPC fault scenario, which was problematic in first experiment, but the accuracy increased significantly. This improved the False Positive rate, while not increasing the corresponding false negative metric. This additional information improved it significantly over its Naïve Bayes counterpart as well as the models in the first experiment. This improvement without a negative cost to the error rates is true for the TAN models across all scenarios. As interesting observation is that the additional information seems to have had a small negative impact in a few cases of the Naïve Bayes models. In summary, the additional information provided an advantage to the TANs , which were able to generate additional causal relations and information to improve diagnostic accuracy.

Figure 4 displays the TAN model structure generated for the HPC scenario. This TAN model with additional features has an accuracy metric of 88% as compared to the original TAN model that produced an accuracy of 80.8%. The Naïve Bayes Model using the additional sensors improved to 72.9%, from the original Naïve Bayes model at 71.4%. The accuracy results clearly indicate: (1) additional sensor information increases diagnostic accuracy and (2) Switching from a Naïve Bayes to a TAN model improves diagnostic accuracy.

This improvement can be examined visually in Figure 4, where in place of the three CIs, the Mach Number sensor becomes the observational root node. The new causal structure, captured in Figure 4 shows the Fuel Flow sensor as a parent to two of the CIs. Network structures such as the one for the HPC fault explicitly illustrate how additional sensor information can be included to enhance the accuracy of the reference model. In general, the new causal relations suggested can be examined by a domain expert who in turn can construct new and improved indicators to use in a reference model. The results generated by these data driven models can provide numbers on how the new in-

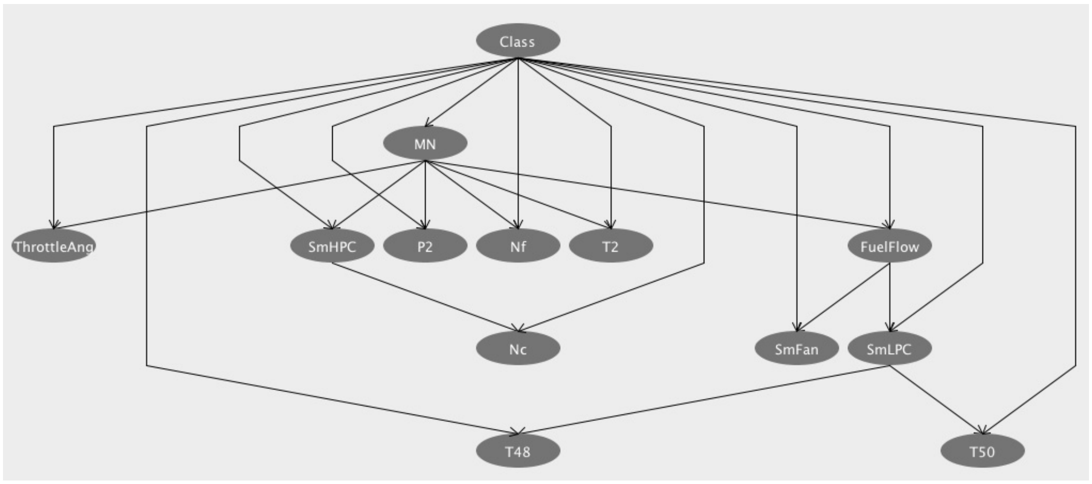


Figure 4: TAN Model for HPC Scenario with Conditional Indicators and Extra Sensors

formation can improve the accuracy of the diagnoser, and how it may impact the error rates.

7 Conclusions and Future Work

The results on experiments conducted with the CMAPS-S data illustrate the promise of the methodology and process we have been developing. To further validate our work, we have identified a number of directions and tasks we need to pursue as we move forward in this project.

- The Naïve Bayes Classifier is an approximation to the expert built reference models. We would like to perform a more thorough experiment and use actual models constructed by domain experts.
- Simulation systems, such as CMAPS-S study particular systems, like the core engine functions in greater detail than any information that can be derived from sensors and monitors in current aircraft configurations. We are looking to develop methods by which detailed simulation data may be combined with actual aircraft flight data to carry on extensive analyses of diagnostic and prognostic events and their propagation through the aircraft system.

Acknowledgements

This project has been supported by NASA NRA NNL09AD44T.

References

Chickering, D. M., Heckerman, D., & Meek, C. (1997). A Bayesian approach to learning Bayesian networks with local structure. In *In Proceedings of Thirteenth Conference on Uncertainty in Artificial Intelligence.* Morgan Kaufmann.

Cohen, I., Goldszmidt, M., Kelly, T., Symons, J., & Chase, J. S. (2004). Correlating instrumentation data to system states: a building block for automated diagnosis and control. In *Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation - Volume 6* (pp. 16–16). Berkeley, CA, USA: USENIX Association.

Frederick, D., DeCastro, J., & Litt, J. (2007). *Users Guide For the Commercial Modular Aero-Propulsion System Simulator* (Tech. Rep.). NASA.

Friedman, N., Geiger, D., & Goldszmidt, M. (1997). Bayesian Network Classifiers. *Machine Learning*, *29*, 131–163.

Hall, M., Eibe, F., Holmes, B., Geoffrey amd Pfahringer, Reutemann, P., & Witten, I. H. (2009). The WEKA Data Mining Software: An Update. *SIGKDD Explorations*, *11*(1), pp. 10-18.

Honeywell. (2010). Vehicle Integrated Prognostic Reasoner. *NASA Contractor Report to appear*, *NNL09AD44T*.

Kruskal, J., Joseph B. (1956). On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem. *Proceedings of the American Mathematical Society*, *7*(1), pp. 48-50.

Schwarz, G. (1978). Estimating the Dimension of a Model,. *Annals of Statistics*, *6*.

Spitzer, C. (2007). Honeywell Primus Epic Aircraft Diagnostic and Maintenance System. *Digital Avionics Handbook*(2), pp. 22-23.

Witten, I., & Frank, E. (1999). *Data Mining : Practical Machine Learning Tools and Techniques with Java Implementations.*

Anomaly Detection in Vessel Tracks using Bayesian networks

Steven Mascaro
Bayesian Intelligence Pty Ltd,
www.bayesian-intelligence.com

Ann E. Nicholson
Clayton School of IT,
Monash University, Australia

Kevin B. Korb
Clayton School of IT,
Monash University, Australia

Abstract

In recent years, electronic tracking has provided voluminous data on vessel movements, leading researchers to try various data mining techniques to find patterns and, especially, deviations from patterns, i.e., for anomaly detection. Here we tackle anomaly detection with Bayesian Networks, learning them from real world Automated Identification System (AIS) data, and from supplementary data, producing both dynamic and static Bayesian network models. We find that the learned networks are quite easy to examine and verify despite incorporating a large number of variables. Combining the mined models improves performance in a variety of cases, demonstrating that learning Bayesian Networks from track data is a promising approach to anomaly detection.

1 INTRODUCTION

A wealth of information on vessel movements has become available to authorities through the use of the Automated Identification System (AIS). Much of this data has even filtered through to the general public via the Internet. Surveillance authorities are interested in using this data to uncover threats to security, illegal trafficking or other risks. While in the past, surveillance has suffered from a lack of solid data, electronic tracking has transformed the problem into one of overabundance, leading to a need for automated analysis.

The main goal of vessel behaviour analysis is to identify anomalies. This requires the development of a model representing normal behaviour, with anomalous behaviour being then identified by the extent of a vessel's deviation from normality. A common approach is to cluster the data around a set of points in a multi-dimensional feature space, where the features of the track are items such as longitude and latitude, speed and course (Laxhammar, 2008). Tracks that are within or near one of these clusters may be considered normal, while the remainder are flagged as potential anomalies. Researchers use many different machine learning techniques to generate normality models from vessel movement data (typically AIS data), and the models are commonly specified in the language of Gaussian mixture models (Laxhammar, 2008), support vector machines (Li et al., 2006), neural networks and others. A disadvantage of these approaches is that they do not provide a causal model that a human user, such as a surveillance officer, can understand, interact with and explore.

Here, we explore the use of Bayesian Networks (BNs) (Pearl, 1988; Korb and Nicholson, 2010) for analysing vessel behaviour and detecting anomalies. While BNs have been widely applied for surveillance and anomaly detection (e.g., Wong et al., 2003; Cansado and Soto, 2008; Wang et al., 2008; Loy et al., 2010), to date there have been only a few preliminary applications of BNs to maritime anomaly detection. As noted by Johansson and Falkman (2007), however, BNs potentially have two substantial advantages in this domain over other types of models: 1) BN models are easily understood by non-specialists and 2) they allow for the straightforward incorporation of expert knowledge. They can also represent causal relations directly and, in that case, have the advantage of being more easily verified and validated, as we show in Section 3.

Johansson and Falkman (2007) used the constraint-based PC algorithm (Spirtes et al., 1993) to learn BNs from simulated data representing normal vessel behaviour. While they claimed their approach identifies a "reasonable proportion" of anomalous tracks, while missing others, no specifics such as false (or true) positive rates were given, nor did they examine how their parameters affect anomaly detection. Helldin and Riveiro (2009) also looked at the use of BNs in anomaly detection with AIS data, but focused specifically on how the reasoning capabilities of a BN can assist surveillance system operators, such as by flagging potential anomalies, but they did not look at learning BNs from the data.

Outside of the maritime domain, Wong et al. (2003)

use BNs to detect disease outbreaks by detecting anomalous patterns in health care data, such as an upswing in the number of people with flu or an unusual decrease in the number of people buying decongestants. Wong et al. use a method called WSARE (What's Strange About Recent Events) to detect when a temporal stream of such data begins deviating from its own baseline profile. This differs from our approach here in that we are concerned with tracks as a whole, rather than trying to identify if and when a track has begun deviating from a normal baseline.

In our study here we data mined AIS data supplied by the Australian Defence Science and Technology Organisation (DSTO). Since many factors can contribute to the (ab)normality of a vessel's behaviour, in this study we also enhanced that data set by adding information such as weather and time, as well as vessel interactions. We used a metric BN learner, CaMML (Wallace and Korb, 1999), that flexibly allows various kinds of structural priors (e.g., directed arcs and tiers), aiding the learning of sensible models.

We investigated two approaches to model learning. First, we trained a model on the track data in its original time series form. For variables related to motion, we added new variables to represent the motion at both step k and $k+1$, effectively making the data represent a dynamic Bayesian network (DBN), which have been used successfully for other kinds of anomaly detection (e.g., Loy et al., 2010). Second, we also created single summary records of each track and learned static models from them. Summary data included average speed and course, number of stops, major stopping points and percentage of time travelling straight.

To assess the value of the networks in anomaly detection we took the common approach of using a measure for how probable a given track is according to the learned models of normality. This measure was applied to data sets representing both normal and anomalous tracks. In addition, we also mutated the presumed normal tracks to help us see how the network's probability estimates change. This led to a very interesting understanding of both the network's behaviour and the nature of the normal data set.

Next we describe our approach to the main components of this study, including details of the time series and track summary methods, the variables used by the BNs and the learning algorithms. We analyse interesting aspects of the learned BNs, then present experimental results in Section 3, which demonstrate the value of Bayesian networks for anomaly detection.

2 APPROACH

While our basic approach is well known — applying a BN learner to produce normality models to be used in assessing degrees of anomaly — in practice the experimental workflow was complex, as shown in Figure 1.

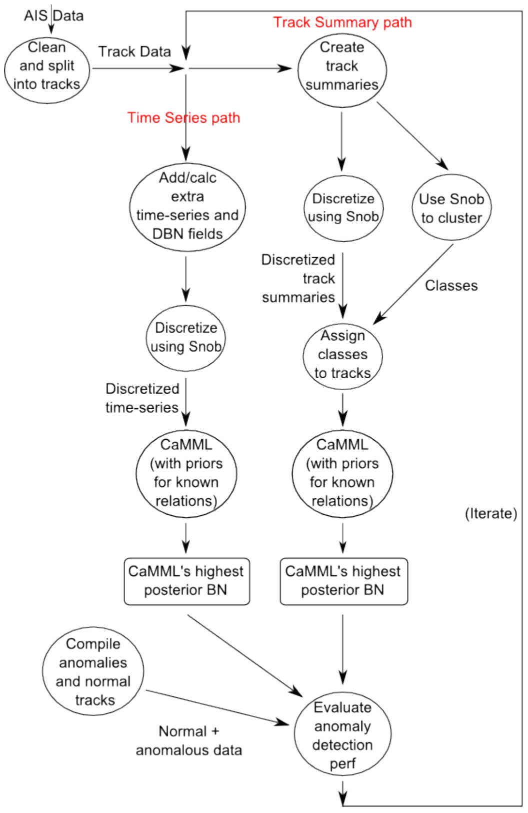

AIS Data
Clean and split into tracks
Track Data
Track Summary path
Create track summaries
Time Series path
Add/calc extra time-series and DBN fields
Discretize using Snob
Use Snob to cluster
Discretized track summaries
Classes
Discretize using Snob
Assign classes to tracks
Discretized time-series
CaMML (with priors for known relations)
CaMML (with priors for known relations)
(Iterate)
CaMML's highest posterior BN
CaMML's highest posterior BN
Compile anomalies and normal tracks
Normal + anomalous data
Evaluate anomaly detection perf

Figure 1: Workflow for the experiments.

2.1 THE DATA

We used AIS data from May 1st to July 31st, 2009 for a section of the NSW coast framing Sydney harbour (see Figure 2). The raw data initially contained just under 9.2 million rows and each row consisted of seven fields: the vessel's MMSI (a nine digit numerical vessel identifier), a timestamp, the latitude and longitude of the vessel, and its reported speed, course and heading (see Table 1). We did not use the MMSI directly in learning, but did use it in pre-processing and to locate additional information about the vessel.

The AIS data was cleaned and separated into 'tracks', first by assigning each record to a separate track based on the MMSI. We then cleaned the data in each track by rounding (and interpolating) each row to the nearest 10 second interval and eliminating duplicate data. However, since the raw data contained many cases in which a single vessel transmits for much of the three month period of the data, further track splitting was required. We split a track record into multiple records when the vessel was stopped or not transmitting for 6 hours or more.[1] This yielded 2,473 tracks across 544 unique MMSIs averaging 1,995 rows each.

Vessel track anomaly detection models have been lim-

[1]We note, however, that since such stops may themselves indicate an anomaly, deciding what constitutes a track warrants future investigation.

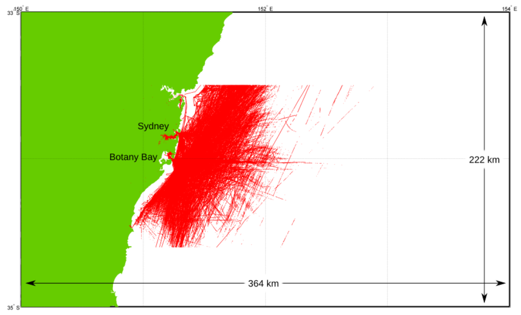


Figure 2: Example AIS tracks.

Table 1: An example of five consecutive rows from the original AIS data, with information removed to preserve anonymity. Each row has been produced by a different ship.

MMSI	Timestamp	Lat	Lon	Speed	Course	Hdng
X	200905X	-33.X	151.X	18.7	49.9	46
X	200905X	-34.X	151.X	2.1	218	80
X	200905X	-33.X	151.X	0	0	511
X	200905X	-34.X	151.X	17.5	183	179
X	200905X	-33.X	151.X	1.2	28	64

ited to kinematic variables, such as location, speed and course, coupled with the type of the vessel (e.g., Johansson and Falkman, 2007). One aim of our study was to investigate the possible advantages of considering additional factors. We added variables related to the ship itself (including type, dimensions and weight), the weather (such as temperature, cloud cover and wind speed), natural temporal factors (including hour of day and time since dawn or dusk), kinematic DBN nodes and elementary information on vessel interactions for both the time series and track summary models. Information about each ship was obtained from three locations: the public websites `marinetraffic.com` and `digital-seas.com` and also from the DSTO. Coverage was generally excellent; for example, only 13 of the 544 vessels lacked ship type information. On the few occasions in which data was missing, we used a "missing" value. Weather information for the period was retrieved from the Australian Bureau of Meteorology website, based on observation stations around Sydney harbour (Bureau of Meteorology, 2010).

2.2 THE MODELS

We investigated two kinds of model based on two different forms of the training data. The first, the **time series model**, uses the data in its original time series form. Each timestep in a track was associated with a set of variables, such as latitude, longitude, speed and so on, that have corresponding nodes in the BN. This approach, of course, has the advantage that learned models can be used in online analysis, but it may miss patterns at a broader time scale.

The second model, the **track summary model**, was based on summaries of each track — e.g., identifying for a given track the number of times the vessel stops, the main stopping locations, etc. While track summaries cannot be used as easily in real-time surveillance, they can capture patterns that occur at the time scale of the track as a whole. For example, if a vessel heads straight out to sea, turns around at a constant rate, then returns directly home, each timestep in the track may appear perfectly normal to any time series-based normality model. However, the behaviour embodied by the track as a whole may be anomalous and worthy of attention. The variables for each type of model are in Figure 3 (see Mascaro et al., 2010).

2.3 CLASSIFICATION AND DISCRETIZATION

We were interested in whether the pre-processing summarization might help us directly to identify types of tracks and anomalies. To test this, we classified the summary tracks using Snob (Wallace and Freeman, 1992), an unsupervised clustering tool comparable to AutoClass (Cheeseman et al., 1988), producing a class variable for each track (see 'Class' node in Figure 3(b)).

Discretization of variables in the data set was needed for technical reasons: (1) the version of CaMML that allows structural priors requires discrete data and (2) we used Netica, which also requires discrete variables. To perform discretization, we again used Snob to classify each continuous variable in one dimension, with each discovered class becoming a state. Using Snob in this way allowed us to recover any hidden regularities and is similar to the attribute clustering approach taken by Li et al. (2006). This can often lead to nodes with uneven distributions. For example, the 'Speed' node in Figure 3a contains lower probability states wedged amongst higher probability states. One might expect to see a more even distribution, however Snob has identified 12 underlying classes corresponding to these 12 states — some of which are much more frequent than their neighbours.

2.4 THE CaMML BN LEARNER

In this work, we make use of the CaMML BN learner (Wallace and Korb, 1999). CaMML (Causal discovery via MML) learns causal BNs from data using a stochastic search (MCMC) and score approach. After learning the structure, we parameterized the model with a standard counting-based procedure (Heckerman, 1998), as did Johansson and Falkman (2007).

CaMML allows one to specify different types of expert priors (ODonnell et al., 2006). These can be hard priors (e.g., an arc *must* be present or absent) or soft priors that specify the probability of certain arcs connecting pairs of variables; other soft priors for more indirect dependencies can also be specified. Here, we used some simple hard priors in the time series model

Table 2: Causal tiers for the variables in the time series model, given as hard priors to CaMML.

1st Tier	ShipType, ShipSize, Rainfall, MaxTemp, EstWindSpeed, EstOktas
2nd Tier	Lat, Lon, Speed, Course, Heading, Acceleration, DayOfWeek, HourOfDay, CourseChangeRate, HeadingChangeRate, NumCloseInteractions, NumLocalInteractions, ClosestType, ClosestSpeed, ClosestCourse, ClosestDistance, SinceDawn, SinceDusk
3rd Tier	Lat-t2, Lon-t2, Course-t2, Heading-t2, Speed-t2, Acceleration-t2

to guarantee that the right DBN relationships held across time steps. We also specified priors in the form of "temporal tiers", putting a temporal partial order over variables and so indicating which variables could *not* be ancestors of which others (Table 2).

2.5 EXPERIMENTAL METHODOLOGY

After pre-processing the data, we ran experiments using CaMML. We divided the data randomly (both time series and track summaries) into 80% (or 1,978 tracks) for training and 20% for testing. As is common with anomaly detection models (e.g. Das and Schneider, 2007; Johansson and Falkman, 2007), the training data consisted of unfiltered real or 'normal' data in order to produce a model of normality against which we could assess deviations. We did a set of 10 runs of CaMML, using different seeds, taking CaMML's reported "best" (highest posterior) network each time, from which we derived the reported results.

3 EVALUATION

3.1 INTERPRETING THE LEARNED MODELS

Figure 3(a) shows an example BN produced by CaMML from the time series data, while Figure 3(b) shows an example learned from the track summary data. It is clear that few arcs in the learned networks represent intuitive *direct* causal relations, other than the DBN arcs (given as hard priors) and the weather variables. Many of the other variables are simultaneous properties of the vessel, which will be correlated by hidden common ancestors. For example, while we would expect a ship's speed, size and course to be related, it isn't obvious what the underlying causes might be. They may be such things as the business the vessel belongs to, the purpose of its trip or the nature of its crew and contents. Some of these hidden causes will be partly captured by the ShipType, e.g., the purpose of a trip employing a cargo ship is almost always transport. This explains why that variable is the common cause of so many others in the time series models. In the track summary network this common cause role is assumed by the 'Class' variable instead.

Causal discovery relying on joint sample data very often gets arc directions wrong, in the anti-causal direction, because it is dependent upon sparse information about any uncovered collisions (where two parents of a child node are not themselves directly connected) to infer all arc directions. For example, Figure 3(a) shows ShipType→Weather, for a variety of weather variables. Of course, ship type cannot affect the weather. A plausible interpretation of this result is that weather conditions *do* affect which types of ship put to sea, so, if anything, the arc directions here are reversed. The simplest and very effective method of dealing with this problem is to introduce extra prior constraints, such as putting some weather variables into a zeroeth Tier.

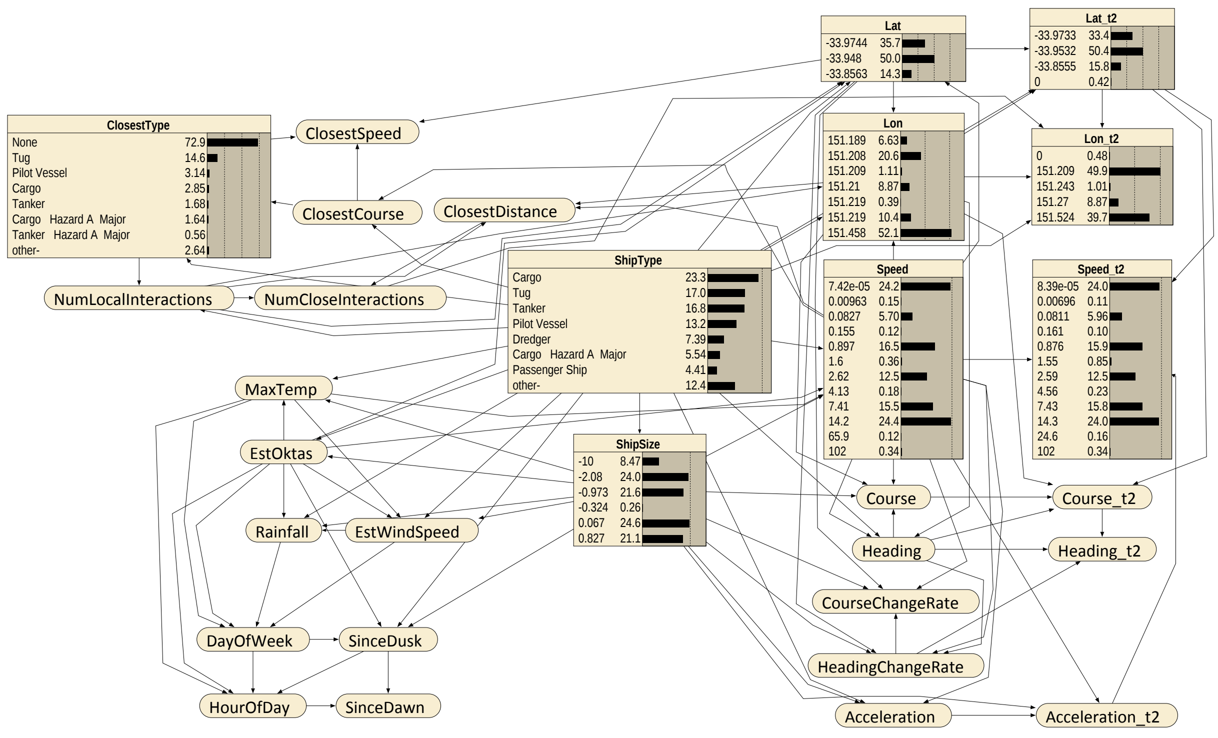
Exploring Bayesian networks is very easy and natural and here turned up many points of interest. In confirming the reasonableness of the time series model, we found that entering 'Tug' or 'Pilot Vessel' into the 'ShipType' variable significantly increases the chance of another vessel being nearby. Cargo ships, on the other hand, travel mostly solo and tankers almost exclusively so. Ship sizes (i.e., the 'ShipSize' variable) are also highly correlated with position (the 'Lat' and 'Lon' variables) via the 'ShipType' variable, with larger vessels tending to appear in a restricted set of locations. The track summary model shows that cargo ships and tankers spend most of their time travelling straight, while tug directions are much more variable. Tugs also tend to stop in different locations from cargo ships, and they tend to be stopped for longer periods than cargo ships.

3.2 ANOMALY SCORES

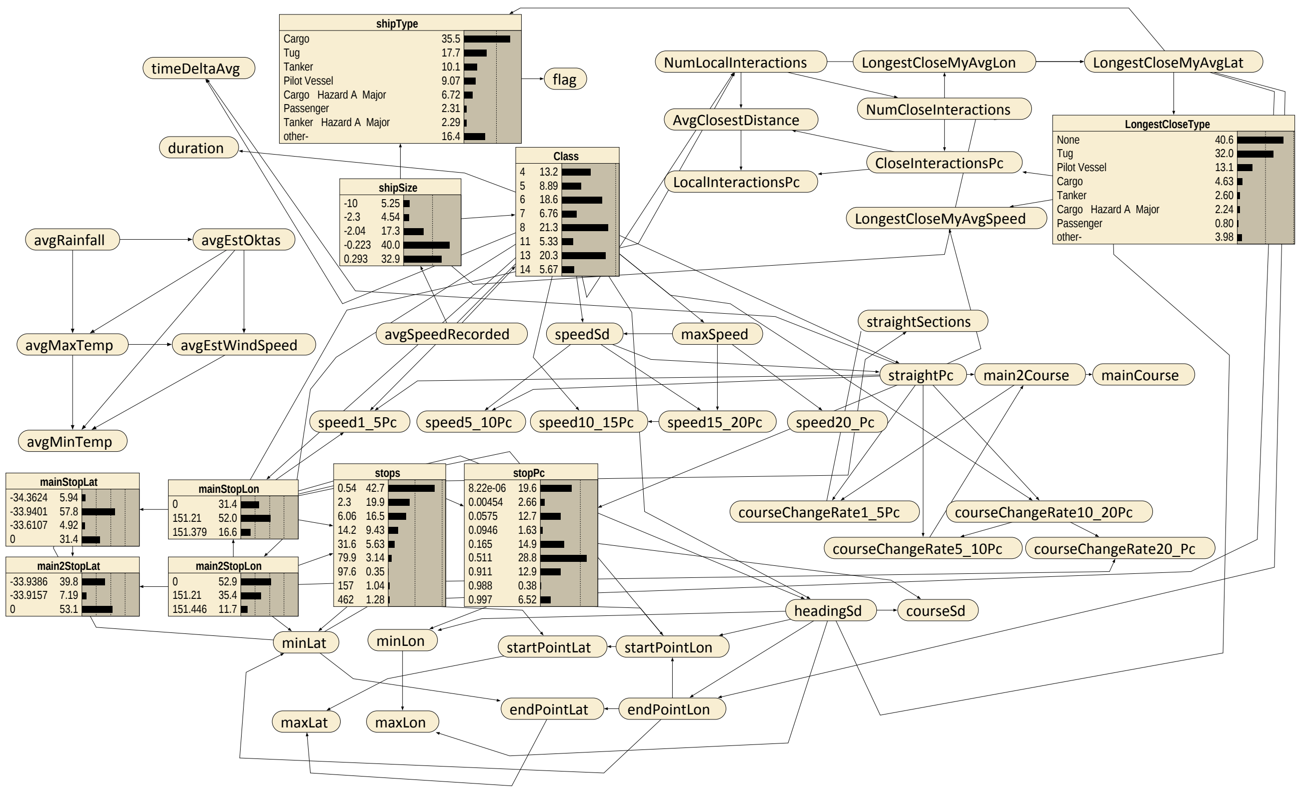
There is no generally accepted method for detecting anomalies from BN models. Jensen and Nielsen (2007) proposed a "conflict measure" to detect possible incoherence in evidence $\mathbf{E} = \{E_1 = e_1, \ldots, E_m = e_m\}$:

$$C(\mathbf{E}) = \log \frac{P(E_1 = e_1) \times \ldots \times P(E_m = e_m)}{P(\mathbf{E})}$$

Jensen and Nielsen use this to identify when a power plant begins behaving abnormally. Unfortunately, this will only catch cases where each attribute is independently common but jointly uncommon. Here, we're interested in any kind of joint uncommonness, even when variables are independently uncommon, which simply comes down to a difference in requirements. In other approaches Loy et al. (2010) used learned DBNs to calculate log-likelihoods and compare them against thresholds selected to maximize the accuracy, extended to detecting abnormal correlations between multiple objects. Cansado and Soto (2008) simply assumed that records with low probabilities given the learned BN are anomalies.



(a)



(b)

Figure 3: Example BNs produced by CaMML for the (a) time series data and (b) track summary data.

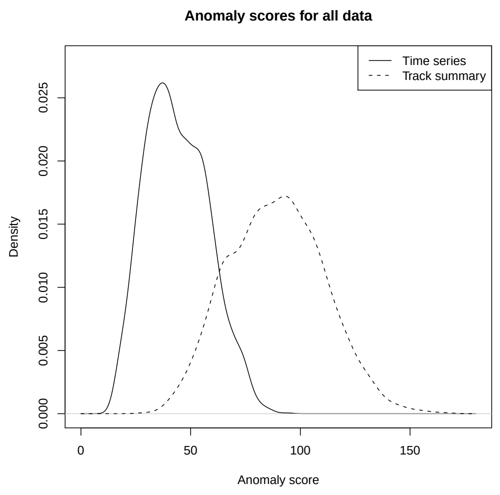


Figure 4: The KDE distributions of anomaly scores for all tracks in the data set according to the (a) time series and (b) track summary networks.

We started from the same assumption as Cansado and Soto, however we think choosing any particular threshold for deciding when tracks are anomalous would be arbitrary. In real applications a specific threshold may present itself as most suitable, but in general we feel it is better to present the probability itself to surveillance operators, albeit in a more convenient form.

Thus, for track summary data, we first computed each track's prior probability given the normality model. Since these probabilities are usually very low (around the order of 1^{-10}) we took the negative log (base 2) to produce an "anomaly score" (i.e., the number of bits required to describe the data, given the model). Put simply, the higher the anomaly score, the less probable the track.

For time series networks we took a similar approach, but instead fed each timestep of the track into the network to yield a probability estimate for that timestep. We then took the average probability over all timesteps to generate a negative log anomaly score. For time series data it is possible, of course, to base anomaly criteria upon *changes* in the track over time. Johansson and Falkman (2007), for example, used sliding windows across a track, looking for any anomalous *windows*. For this study, however, we focused on criteria for assessing the tracks as wholes, leaving this kind of alternative for future investigation.

Calculating anomaly scores for all the tracks in our data set and plotting the distribution of the results (using a Gaussian Kernel Density Estimator [KDE]), we obtained Figure 4. These show a fair amount of diversity among anomaly scores, i.e. they do not simply clump around the lowest possible score. Note that the scores produced by the time series model are quite distinct from those of the track summary model. One likely reason is that the track summary scores are simply based on more variables, making each instance more specific and less probable. There is a surprisingly small correlation between the two sets of scores (r =0.159; $p < 0.001$).[2] The two models look at different aspects of each track, and, as we see below, reinforce each other when performing anomaly detection.

3.3 RESULTS ON ANOMALOUS DATA

Unfortunately, we did not have any access to known anomalous tracks nor are there any standardised or publicly available vessel track data sets containing anomalies (or otherwise). Nevertheless, there are many ways to create anomalous data. Cansado and Soto (2008) generated anomalies by modifying selected attributes to random values within their ranges. Johansson and Falkman (2007) generated anomalous data using anomalous models.[3] Here, we tried three approaches, partly inspired by these previous methods: modifying instances by swapping incorrect ship type information, splicing tracks together, and drawing anomalous tracks.

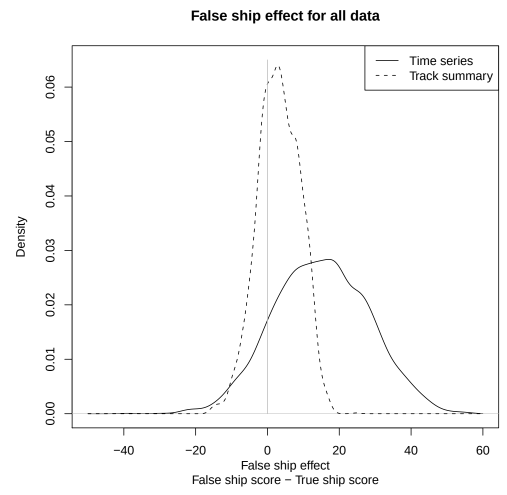
3.3.1 The False Ship Effect

For each track in the training set, the ship type information was swapped with that of another randomly selected ship of a different type, leaving the track data alone. Figure 5(a) shows how this affected the anomaly score. In most cases this false ship effect is positive, increasing the anomaly score. The false ship effect for the time series model is positive in around 87.2% of the cases as opposed to 69.4% of cases for the track summary model. Sometimes, however, tracks have become *more* probable given incorrect ship information, which itself seems anomalous! To be sure, many of the ship types are in fact quite similar (e.g., there are several sub-categories of cargo ship) so switching between these may randomly produce a more likely track. However, this does not account for all the cases. A closer look at these showed that many are highly improbable (i.e., have high anomaly scores), suggesting that either they have been mislabelled or, more intriguingly, that they do indeed behave anomalously according to their type. This suggests a new criterion for anomalousness based not merely upon the probability of the given track but on what alterations might explain the track better. This has some of the spirit of Jensen and Nielsen's conflict measure, though is clearly quite different; we leave this possibility for future exploration.

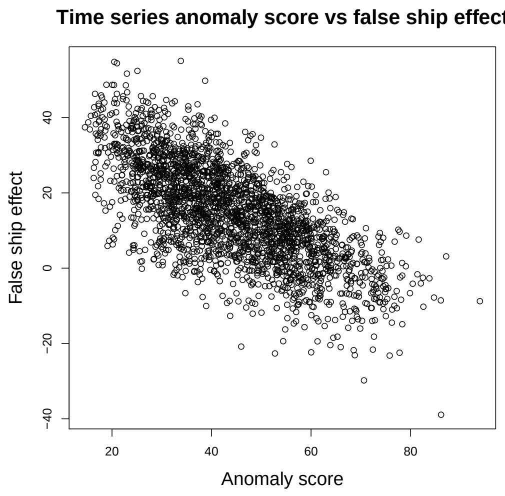
Figures 5(b) and 5(c) show scatter plots of the anomaly score versus the false ship effect. With the time series model, we can see that as the anomaly score grows, the

[2]Earlier iterations with cruder discretizations and more variables in common showed a stronger correlation — however, as models grew more detailed, the correlation shrank.

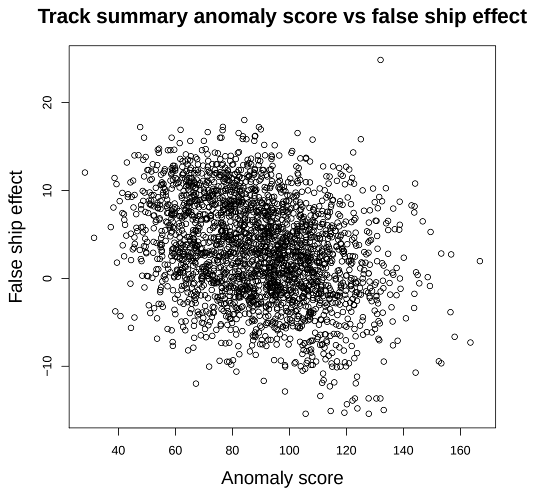
[3]Wang et al. (2008), without known anomalous data, simply weakened their threshold to find "anomalies", whether they were there are not!



(a)



(b)



(c)

Figure 5: False ship effect: (a) anomaly score differences for false ship information versus correct ship information, sorted by score; and scatter plots for (b) time series and (c) track summary networks.

false ship effect falls (r =-0.70, $p \ll 0.01$). This also occurs with the track summary model, to a smaller extent (r =-0.31, $p \ll 0.01$).

3.3.2 Track Splices

We also created anomalous tracks by splicing random tracks together. This allowed us to test our models for their ability to detect discontinuities as well as major changes in behaviour. Specifically, we selected 140 tracks at random and replaced their tails with those of other tracks (retaining the times and types of the original track). We spliced half of the tracks with those created by ships of a different type and we spliced the other half with tracks created by ships of the same type. When assessing these tracks using the track summary model, tracks forged from different types yield an average anomaly score of 121.3 while those forged from the same type yield an anomaly score of 115.4 ($p \ll 0.01$). Both scores are significantly different from the average anomaly score for all data of 89.0.

With the spliced tracks, as we expected the track summary model performed slightly better than the time series approach, because the time series model is not able to detect unusual behaviour across the whole track. Tracks put together from ships of different types produced an average score of 48.9 while those spliced from same types had a score of 45.6; while a small difference, this was statistically significant ($p < 0.01$). In addition, while the higher score was significantly different ($p < 0.01$) from the average of the full data set (43.8), the lower score was not ($p \gg 0.01$). Here we can see the advantage of the higher level view of the track summaries.

3.3.3 Manually drawn anomalies

Finally, we tested models using anomalous tracks drawn with a mouse over a map, where the mouse location and speed generated the vessel location and speed respectively. Other factors were created randomly, including the time and duration, noise in the data, vessel details and maximum speed. This allowed us to compare the performance of both models across several different categories of anomalous behaviour, thereby shedding light on the strengths and weaknesses of each model. Anomalous behaviour in these tracks included very noisy data, close interactions with many other vessels, vessels that circle in unusual patterns, vessels travelling over land, overly short tracks in the middle of the sea and vessels behaving against their type. In all, 107 such tracks were created.

When combined with the normal track test data, and scored using the two models both independently and combined, the ROC (receiver operating characteristic) curves of Figure 6 are the result. The ROC curves demonstrate the tradeoffs that can be made (if we were to settle on specific thresholds for anomalies) between false positives and true positives; the greater

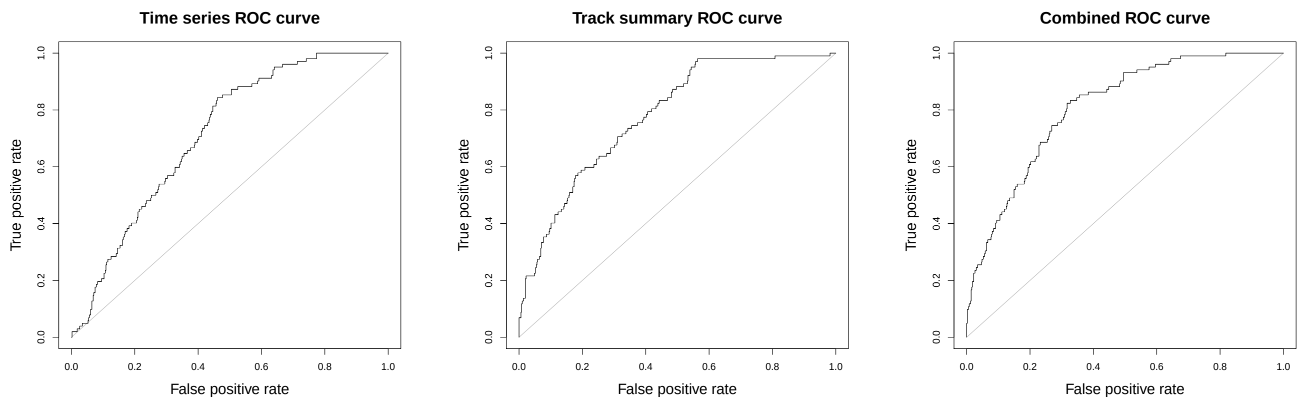


Figure 6: ROC curves for test data, containing both normal tracks and manually created anomalous tracks, given the (left) time series, (middle) track summary and (right) combined models.

the area under the curve (AUC), the less severe the tradeoff needs to be. We can see that the track summary model (with an AUC of 0.780) performs better than the time series model (AUC 0.712). Adding the anomaly scores from the two models together (in effect, creating a combined model with equal weight given to each individual model) performs better again (AUC 0.809). Table 3 shows the average scores each model yielded for various kinds of anomalous tracks. We can see that both models easily detected the tracks containing too many close interactions (average scores of 139.9 and 75.8, against the test averages of 90.8 and 45.7, giving Deltas of +49.1 and +30.1 for track summary and time series models, respectively). The time series model detected overly short tracks best (track summary: +4.7; time series: +17), while the track summary model substantially outperformed the time series model for tracks containing unusual stops, as would be expected (track summary: +28.3; time series: +2.9). In most cases, the track summary model outperformed the time series model.

3.3.4 Testing on Johansson & Falkman's simulated data

We also applied our methods to the simulated data used by Johansson and Falkman (2007), both normal and anomalous. Our models, while not well suited to the simulated data, performed reasonably well. In particular, with the track summary model, anomalous tracks received an average anomaly score of 22, while normal tracks averaged 17; while in the time series model, anomalous tracks received an average score of 29, with normal tracks averaging 25. When we calculated the ROC curves, we found that the time series model performed better with this data set with an AUC of 0.691, over the track summary AUC of 0.652. This was likely due to a lack of extended ship type information. The combined model (whose ROC curve is shown in Figure 7) again performs better than both individually, with an AUC of 0.727.

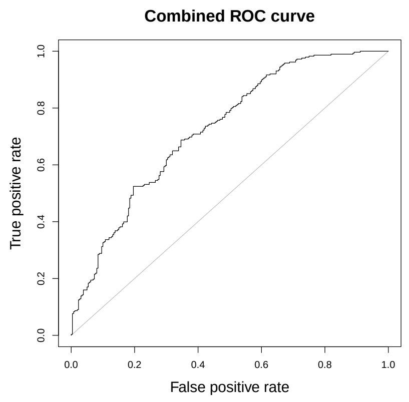


Figure 7: ROC curves for the Johansson and Falkman data using the combined models.

We also examined what happens when the ship type of the tracks is altered. Interestingly, the only cases in which this change created a notable *negative* false ship effect (i.e., increased the probability of the track) again involved high anomaly scores. These scores were 25 and above for the track summaries and 36 and above for the time series — both much higher than the respective average scores for the anomalous tracks.[4]

4 CONCLUSION

We have demonstrated Bayesian Networks are a promising tool for detecting anomalies in vessel tracks. By using a BN learner on AIS data supplemented by additional real world data, we produced both dynamic and static networks, which demonstrated distinct and complementary strengths in identifying anomalies. Thus, we were able to improve anomaly detection by combining their assessments. This sug-

[4]For further details of our comparison with Johansson & Falkman's work, see Mascaro et al. (2010).

Table 3: Average anomaly scores for various forms of anomaly. Columns headed 'Delta' indicate the difference from the average score for normal test tracks.

Type	**Track Summary Score**	**Delta**	**Time Series Score**	**Delta**
Normal test tracks	90.8	(0)	45.7	(0)
Random movement in the middle of water	102.4	+11.7	50.8	+5.1
Closed tracks in the middle of water	101.7	+10.9	53.7	+8.0
Very short tracks	95.5	+4.7	62.7	+17.0
Unusual stops	119.1	+28.3	48.6	+2.9
Tracks with many interactions	139.9	+49.1	75.8	+30.1
Tracks with many loops	126.2	+35.4	52.7	+7.0
Travel over land	122.2	+31.4	60.2	+14.5
Appearing at edges of observable area only	103.5	+12.7	54.2	+8.6
Very noisy observations	135.2	+44.4	54.6	+8.9
Tracks behaving against type	113.7	+22.9	57.8	+12.0
Multiple anomalies	126.9	+36.1	53.9	+8.2

gests that learning networks at still additional time scales, intermediate between the full track and each AIS snapshot, may improve anomaly detection even further. Such approaches may well generalize to other kinds of anomaly detection and can be extended to work with other kinds of track, such as those created by cars, planes and humans.

Acknowledgements

The authors would like to thank the DSTO for supporting this work, Göran Falkman and especially Fredrik Johansson for providing access to their data and generously answering many questions about their approach, and SAAB Microwave Systems for permitting us the use of their simulated data.

References

Bureau of Meteorology (2010). Daily weather observations, May – July 2009. `http://www.bom.gov.au/climate/dwo/IDCJDW2124.latest.shtml`.

Cansado, A. and A. Soto (2008). Unsupervised anomaly detection in large databases using Bayesian networks. *Applied Artificial Intelligence 22*(4), 309 – 330.

Cheeseman, P., J. Stutz, M. Self, J. Kelly, W. Taylor, and D. Freeman (1988, August). Bayesian classification. In *AAAI 88*, pp. 607–611.

Das, K. and J. Schneider (2007). Detecting anomalous records in categorical datasets. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 220–229. ACM.

Heckerman, D. (1998). A tutorial on learning with Bayesian networks. In M. Jordan (Ed.), *Learning in Graphical Models*, pp. 301–354. MIT.

Helldin, T. and M. Riveiro (2009). Explanation methods for Bayesian networks: Review and application to a maritime scenario. In *Proc. of the 3rd Annual Skövde Workshop on Information Fusion Topics (SWIFT 2009)*, pp. 11–16.

Jensen, F. V. and T. D. Nielsen (2007). *Bayesian networks and decision graphs* (2nd ed.). New York: Springer Verlag.

Johansson, F. and G. Falkman (2007). Detection of vessel anomalies — a Bayesian network approach. In *Int. Conf. on Intelligent Sensors, Sensor Networks and Information Processing*, pp. 395–400.

Korb, K. B. and A. E. Nicholson (2010). *Bayesian Artificial Intelligence* (2nd ed.). Chapman & Hall/CRC Press.

Laxhammar, R. (2008). Anomaly detection for sea surveillance. In *The 11th Int. Conf. on Information Fusion*, pp. 55–62.

Li, X., J. Han, and S. Kim (2006). Motion-Alert: Automatic anomaly detection in massive moving objects. In *Proc. of the 2006 IEEE Intelligence and Security Informatics Conference (ISI 2006)*, Berlin, pp. 166–177. Springer.

Loy, C., T. Xiang, and S. Gong (2010). Detecting and discriminating behavioural anomalies. *Pattern Recognition*.

Mascaro, S., K. B. Korb, and A. E. Nicholson (2010). Learning normal vessel behaviour from AIS data with Bayesian networks at two time scales. Technical Report TR 2010/, Bayesian Intelligence.

ODonnell, R., A. Nicholson, B. Han, K. Korb, M. Alam, and L. Hope (2006). Causal discovery with prior information. In A. Sattar and B.-H. Kang (Eds.), *AI 2006*, Volume 4304 of *LNCS*, pp. 1162–1167. Berlin: Springer.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems*. San Mateo, CA: Morgan Kaufmann.

Spirtes, P., C. Glymour, and R. Scheines (1993). *Causation, Prediction and Search*. Springer Verlag.

Wallace, C. S. and P. R. Freeman (1992). Single factor estimation by MML. *Journal of the Royal Statistical Society B 54*(1), 195–209.

Wallace, C. S. and K. B. Korb (1999). Learning linear causal models by MML sampling. In A. Gammerman (Ed.), *Causal Models and Intelligent Data Management*, pp. 89–111. Heidelberg: Springer-Verlag.

Wang, X., J. Lizier, O. Obst, M. Prokopenko, and P. Wang (2008). Spatiotemporal anomaly detection in gas monitoring sensor networks. *Wireless Sensor Networks*, 90–105.

Wong, W., A. Moore, G. Cooper, and M. Wagner (2003). Bayesian network anomaly pattern detection for disease outbreaks. In *Int. Conf. on Machine Learning*, Volume 20, pp. 808.

ABC-BN: A tool for building, maintaining and using Bayesian networks in an environmental management application

Ann E. Nicholson, Owen Woodberry, Steven Mascaro, Kevin Korb
Bayesian Intelligence Pty Ltd.
2/21 The Parade, Clarinda
VIC, 3169, Australia

Adrian Moorrees, Alicia Lucas
Biodiversity and Ecosystem Services
2/8 Nicholson Street
East Melbourne, VIC 3002, Australia

Abstract

The Victorian state government Department of Sustainability and Environment has a web-based application called Actions for Biodiversity Conservation (ABC) as a central resource for managing over 400 threatened species and communities. ABC maintains information for species and communities at individual locations, including lists of threats (to species and communities), actions (to mitigate threats) and population and habitat factors. Here we describe an extension to ABC, a tool for building Bayesian networks for selected populations of threatened species and occurrences of threatened communities, to model the interactions between actions, threats and population and habitat factors. The tool, called ABC-BN, also allows users to do what-if scenario reasoning, and is integrated with ongoing monitoring and reporting. Unlike most ecological BN modelling to date, which typically takes months or years to produce a single, often complex, BN, for a specific problem, the aim here was to produce a tool that would facilitate the development, maintenance and use of a large number of quite simple standardized BNs, specialized for particular instances. We describe the incremental development of ABC-BN over 3 years. A prototype version was evaluated in 2009 by building models for 100 species, and the completed ABC-BN was deployed in April, 2011.

1 INTRODUCTION

The Victorian state government Department of Sustainability and Environment (DSE) has a web-based application called Actions for Biodiversity Conservation (ABC) as a central resource for managing threatened species and communities (DSE, 2009). It facilitates the management of actions documented in Action Statements prepared under the Flora and Fauna Guarantee Act 1988 and Recovery Plans prepared under the Environment Protection and Biodiversity Conservation Act 1999. ABC currently holds information on more than 400 threatened species and communities[1] and over 8,000 management actions at approximately 2000 locations across Victoria. ABC maintains information for species and communities at individual locations, including lists of threats (to species), actions (to mitigate threats) and population and habitat factors. By bringing together information from a range of sources, ABC is intended to make significant improvements in both knowledge about threatened species and communities and the transfer of that knowledge. The recording of actions implemented for populations and communities enables the development of prioritized lists of actions to be made on an increasingly sound basis by land and water managers. Reporting on outcomes provides a basis for applying adaptive management, whereby the effectiveness of management can be improved based on the current and previous outcomes for a population. By 2004, ABC Stage 1 was operational and able to report on whether an action to ameliorate a threat had been carried out. ABC Stage 2 (deployed in 2006), incorporated the facility to prepare actions plans within the system. However ABC did not support any modelling of interactions between actions, threats and outcome factors and had no predictive capability. More importantly, it was not able to report on the status of a species or community for which there was little or no data collected, in a way that was comparable across the state.

Bayesian networks (BNs) (Pearl, 1988; Jensen and Nielsen, 2007), are becoming increasingly popular for

[1]The FFG Act 1988 defines a community as “a type of assemblage which is wholly or substantially made up of taxa of flora or fauna existing together in the wild”.

environmental and ecological monitoring and risk assessment (see § 5.2.3 in Korb and Nicholson, 2010 for a recent survey). There have been a number of modelling guidelines published (e.g., Varis and Kuikka, 1999; Borsuk et al., 2004; Renken and Mumby, 2009), while Uusitalo (2007) reviews their features and use in modelling environmental applications. In 2008, DSE decided to extend ABC with BN technology, to provide the capacity to model the interactions between actions, threats and outcomes and overall status, and allow users to do what-if predictive and diagnostic scenario reasoning. These additional capacities would be integrated with ongoing monitoring and reporting.

In most ecological BN modelling to date the knowledge engineering paradigm is to develop a single, often complex BN, for a specific problem, which typically takes months or years to complete (e.g., Pollino et al., 2007; Smith et al., 2007; Chee et al., 2005). Here, in contrast, the aim was to produce a tool that would facilitate the development, maintenance and use of BNs for specific populations and occurrences of species and communities with a high priority within ABC, rather than entire species and communities.

This aim led to some key design decisions. First, the tool would be for use by the DSE so-called monitors – DSE scientists and Biodiversity Officers – already maintaining information in ABC about the species for which they were responsible. The BN technology would be hidden from these users. Second, the tool would support the building of simple BNs using a template that enforced a certain causal structure between actions, threats and outcomes.

In this paper, we describe the incremental development of the tool, called ABC-BN, over a 3 year period, with the challenges of changes to the underlying BN template, and a number of substantial additions to the tool requirements. We give an overview of the ABC-BN's functionality, which is based on the iterative, incremental knowledge engineering of BNs Laskey and Mahoney (2000) A prototype version was evaluated in 2009 by building models for 100 species, and the completed ABC-BN was deployed in April 2011.

2 ACTIONS FOR BIODIVERSITY CONSERVATION (ABC)

The first version of ABC was developed for DSE by Spatial Vision, a software company specialising in GIS applications. Stage I became operational in 2004, with the Stage II in place from 2006. The addition of ABC-BN, to be developed by Bayesian Intelligence, became part of Stage III, along with enhancements to ABC itself, to be delivered by Spatial Vision.

ABC is a web-based interface to a database application. It is based around entities called **items**, which may be flora or fauna species, communities, or potentially threatening processes. Within ABC, there are different types of users, with different responsibilities and access levels. ABC is based on recording information about each item at each location (linked with a GIS) it is found; there is a **location monitor** responsible for recording the management and monitoring. The **item monitor** oversees the information about each item, while **system administrators** have additional responsibilities such as appointing monitors and adding actions and threats to the global lists. Finally, there are various **stakeholders** (e.g., members of other government organizations, NGOs, community groups, academics and researchers), who can be given access to the information in ABC, but may not make changes.

For each item, the ABC database contains information about the **threats** that have been identified at each location, which are ranked along two dimensions, **"'likelihood"** and **"impact"**, representing priorities. It records **actions** (from a central list of possible actions) that have been identified as potentially useful to reduce threats to the item at that location. Finally, monitors must record monitored information in actions taken, the status of threats, and various **outcomes**, which are observations relating to the species populations or community occurrences, or the habitat which can be monitored. This information facilitates the prioritizing of species and communities based on the importance of the location and contributions of actions to mitigate threats.

3 ARCHITECTURE

The key requirement for ABC-BN was that it be completely aligned with the existing ABC application, with the same web interface, the same types of users, and model the same action, threat and outcome information in the ABC database.

Figure 1 shows the system architecture for integration of ABC and ABC-BN. ABC-BN is invoked via a menu tab from ABC, with the current item as well as the user ID, passed to ABC-BN. From that point, ABC-BN interacts directly with the users via their browser, with uploads and downloads of BN models allowed via a shared DSE file system. ABC-BN is a TOMCAT application, implemented using JavaServer Pages and the NeticaJ Java API.[2] The BNs and ABC-BN specific information are stored in new tables added to the ABC Oracle database.

[2]`www.norsys.com`

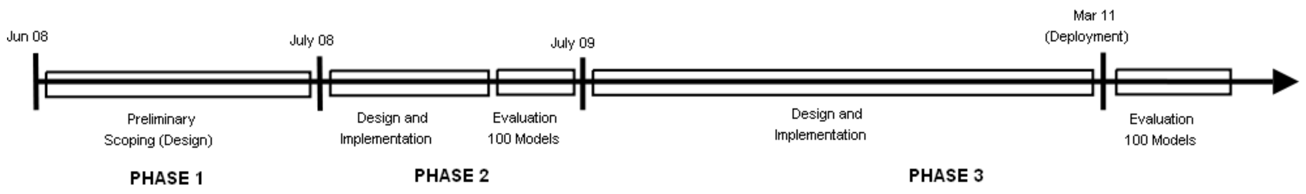
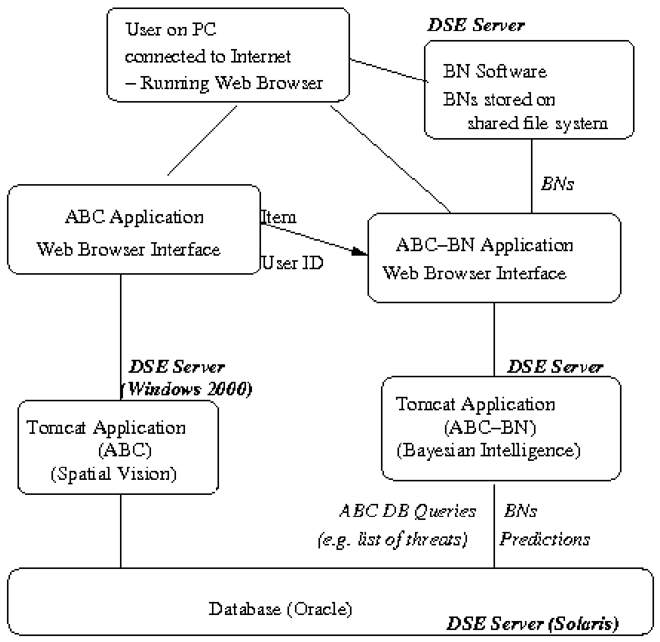
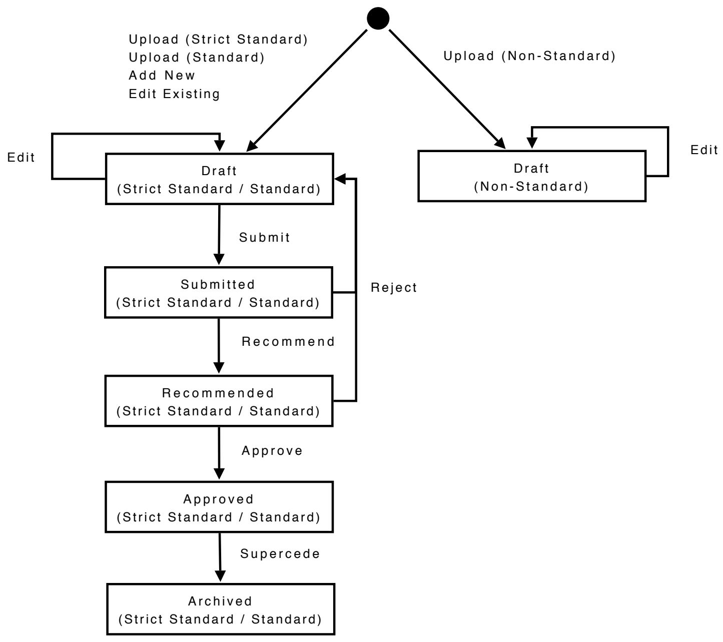


Figure 2: ABC-BN development phases



User on PC connected to Internet – Running Web Browser
DSE Server
BN Software
BNs stored on shared file system
BNs
ABC Application
Web Browser Interface
Item
User ID
ABC–BN Application
Web Browser Interface
DSE Server (Windows 2000)
DSE Server
Tomcat Application (ABC) (Spatial Vision)
Tomcat Application (ABC–BN) (Bayesian Intelligence)
ABC DB Queries (e.g. list of threats)
BNs Predictions
Database (Oracle)
DSE Server (Solaris)

Figure 1: System architecture for the integration of ABC and ABC-BN



Upload (Strict Standard)
Upload (Standard)
Add New
Edit Existing
Upload (Non-Standard)
Edit
Draft (Strict Standard / Standard)
Draft (Non-Standard)
Edit
Submit
Submitted (Strict Standard / Standard)
Reject
Recommend
Recommended (Strict Standard / Standard)
Approve
Approved (Strict Standard / Standard)
Supercede
Archived (Strict Standard / Standard)

Figure 3: Model lifecycle

3.1 Incremental and Iterative Development

ABC-BN was designed and built incrementally, following the prototype-based spiral software development cycle advocated by Brookes (1995) and Boehm (1988). There were three main phases (see Figure 2): a preliminary scoping phase, then two main design and development phases. These were due to (1) changes to the underlying BN template and (2) a number of substantial additions to the tool functionality.

4 USER ROLES AND MODEL MANAGEMENT

ABC-BN provides model management and an approval workflow, together with an audit history. The BN models are managed for each item location; there may be many draft models, but only one approved model at any one time. Models must go through a three step submission process: **submitted** → **recommended** → **approved**. When an approved model is superseded (by a new approved model) it becomes **archived**. Figure 3 shows the model lifecycle.

Two user roles were added for ABC-BN: (1) **Model administrators** have privileged access to the ABC-BN module, notably approving models and the ability to edit the ABC-BN global settings, upload Netica BNs to the system, and delete any model; (2) **Outcome monitors** are nominated by the model administrators for each item location.[3] Outcome monitors may build and submit models for approval, while item monitors are responsible for recommending models.

Table 1 summarizes the user roles within ABC-BN., while Figure 12 shows the model management page.[4]

[3]This is because they may be someone other than the location monitor.

[4]Figures showig screen shots of ABC-BN are placed together at the end of the paper.

Table 1: User roles

Role	Abbr.	Comments
Model Administrator	MA	Full access to the ABC-BN module. Able to create, submit, recommend and approve models.
Item Monitor	IM	Access all models. Able to submit and recommend models for locations under the item.
Outcome Monitor	OM	Access all models. Able to create and submit models for item location.
Any (Other) Monitor	M	Access all models.
Stakeholder	SH	Access only Approved models.

5 MODEL BUILIDNG

ABC-BN supports all stages of the BN elicitation process. The elicitation algorithm is shown in Table 2. Navigation around the elicitation process is flexible, the user may move to any module, and any sub-module within that module (using tab menus). Thus it provides the iterative and incremental BN construction process advocated by many (e.g., Laskey and Mahoney, 2000; Korb and Nicholson, 2010; Boneh, 2010). Figure 4 shows an example of a BN created within the system.[5]

5.1 Structure

ABC-BN supports building models specific to each item location, based on a pre-defined structure template, incorporating actions, threats and asset factors in the ABC database. This template evolved over ABC-BN's development phases as shown in Figure 5:

Phase 1: Template contains only threats, which are parents of a ThreatTrend node, with values {Improving, Stable, Worsening}

Phase 2: Template contains Action nodes (all root nodes) which are parents of the Threat nodes, which are in turn parents of AssetFactor nodes, as well as two Trend nodes.

Phase 3: Trend nodes modelling *change* now replaced by Status nodes representing absolute values: {Severe,Moderate,Negligible} for ThreatStatus and {Good, Fair, Poor} for AssetStatus.

The elicitation process builds only BNs that follow this so-called **strict standard** template, which limits the complexity of the networks and ensures they are easily analyzed for reporting (based on the Status nodes).[6]

[5]This BN was produced by Phil Papas and Di Crowther of DSE, Biodiversity and Ecosystem Services, Arthur Rylah Institute.

[6]ABC-BN also allows uploading of BNs (for example, constructed in Netica) that do not follow the strict stan-

Table 2: Elicitation algorithm

Module	Sub-module	Algorithm
		-create standard template model
Structure:	Name:	-name the model
	Threat:	-select threats to be included in the model
	Actions & Asset Factors:	-for each threat variable -select action variables affecting this threat -select asset factor variables affected by this threat
	States:	-for all action, threat and asset factor variables -select a state space -if optional state description -give description
	Additional Links:	-for each threat and asset factor variable -select other variables, of the same type, affected by this variable
CPTs:		-for all threat, asset factor and query variables -for each parent state combination -select verbal cues for each state
Scenarios:		-test model with scenarios tool

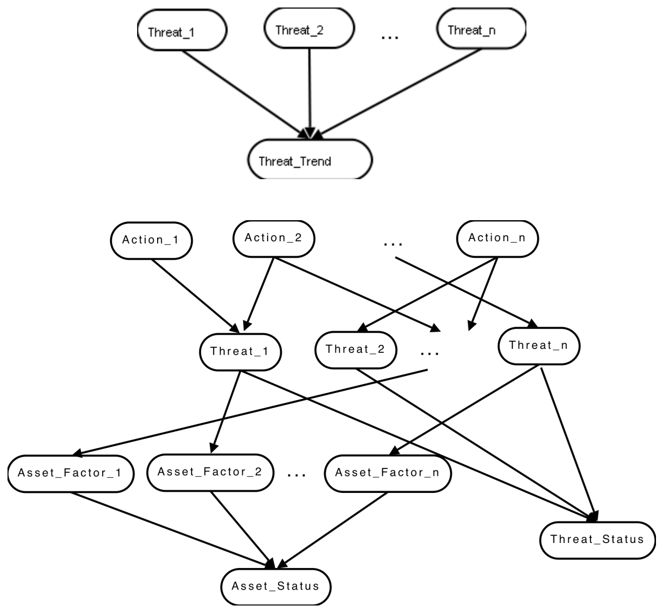


Figure 5: ABC-BN's templates from Phase 1 (above) and Phase 3 (below). The Phase 2 template was as for Phase 3, but contained Trend nodes, instead of Status nodes.

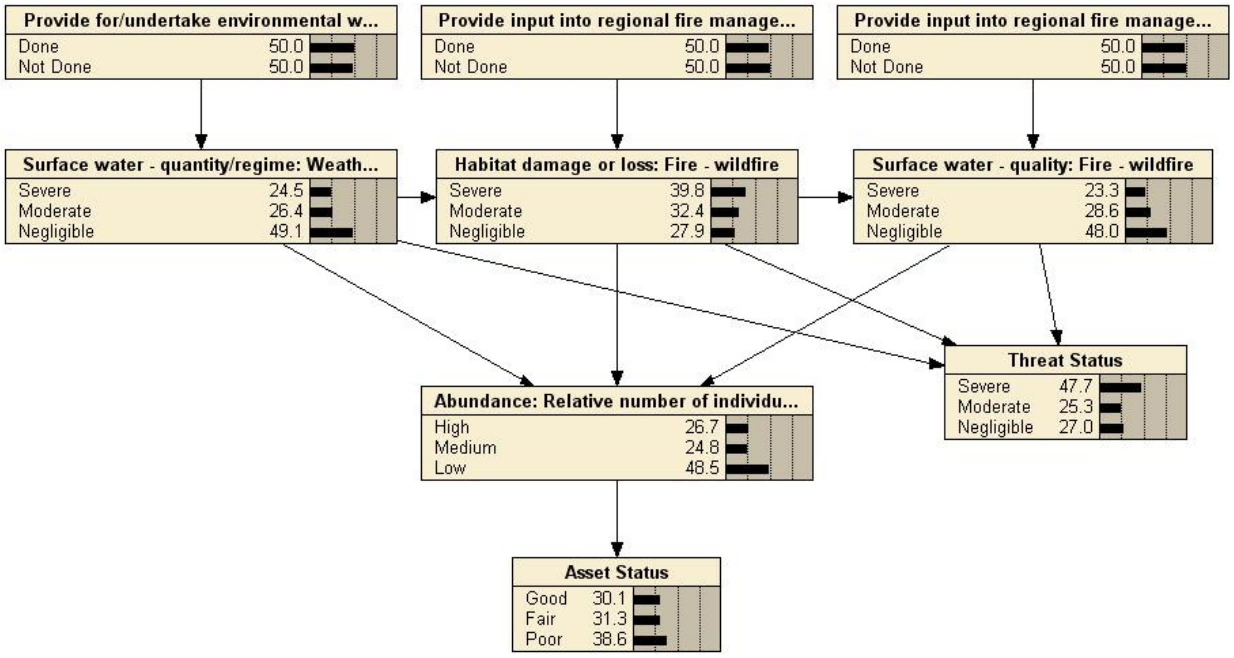
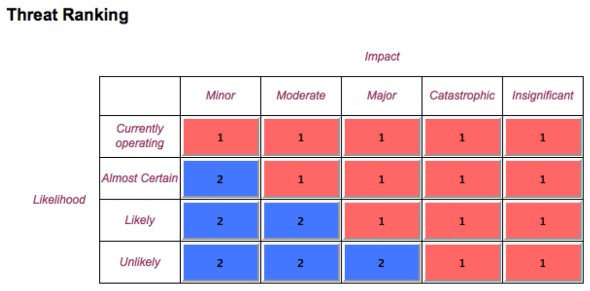


Figure 4: An example complete BN created within ABC-BN.



Threat Ranking

Likelihood \ Impact	Minor	Moderate	Major	Catastrophic	Insignificant
Currently operating	1	1	1	1	1
Almost Certain	2	1	1	1	1
Likely	2	2	1	1	1
Unlikely	2	2	2	1	1

Figure 6: Threat Ranking page.

The BN nodes are selected from the values already in the database for that item and location. The discrete state space for each node is selected from a global list (intended to provide consistency across the system, and maintained by the Model Administrator), divided into "types"; for example the "Choice" type could have alternatives {Yes,No} and {Done,NotDone}, while an example "Amount" type is {High, Medium, Low}.

The user first selects threats from a list of threats for that item. Administrators can assign a threat to one of two ranks based on the impact and likelihood of a threat, (see Figure 6), which changes the way threats are presented on the Threat Selector page (see Figure 13). Once the THREAT nodes are selected, the user then selects the associated ACTION and ASSET-FACTOR nodes for each (see Figure 14).

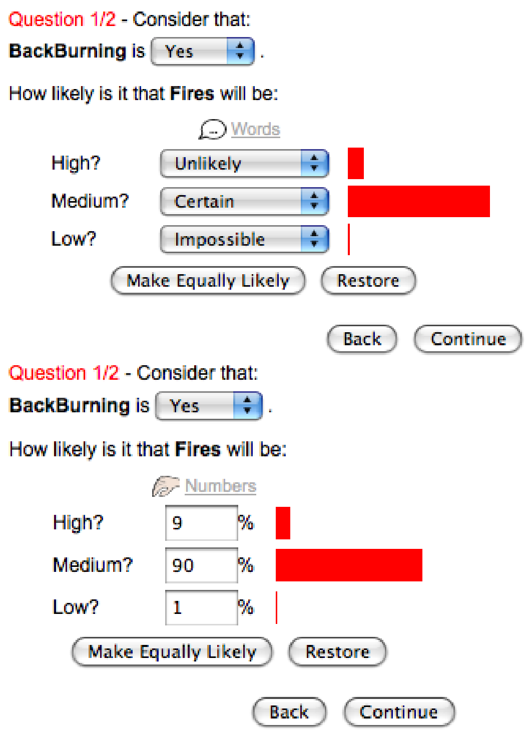
dard: (1) standard models, which are similar to the strict standard structure, but without any constraints on the links, and may be submitted for approval; (2) non-standard models can be any Netica BN, but may not be submitted for approval, as they can't be analyzed for reporting.

5.2 Parameter Elicitation

Eliciting the parameters of the BN, the conditional probabilities, is recognised as a difficult task. For each node, there is distribution for each combination of values of the parent variables; this is exponential in the number of parent variables. It can be hard for experts to express their experience/opinion in numbers, and they are often inconsistent. Hence the decision was made for ABC-BN to support the alternative qualitative assessment of probabilities using verbal anchors, developed by van der Gaag et al. (1999), and implemented in the Verbal Elicitor tool (Hope et al., 2002). The user is presented with scenario descriptions (about combination of parent values) and a selection of common chance tags, e.g., "certain", "likely" or "impossible". In this way actual probability values are not required or shown to the expert. ABC-BN contains a so-called verbal map (which is maintained by the Model Administrator) which maps verbal cues to probabilities (see Figure 15).

For Phase 2, ABC-BN provided only this qualitative elicitation, and the users could not see the probabilities stored into the CPT.[7] Feedback from the users during the Phase 2 evaluation indicated that this was

[7]That is, unless they downloaded the BN for viewing in Netica!


Question 1/2 - Consider that:

BackBurning is Yes .

How likely is it that **Fires** will be:

Words

High?	Unlikely
Medium?	Certain
Low?	Impossible

Question 1/2 - Consider that:

BackBurning is Yes .

How likely is it that **Fires** will be:

Numbers

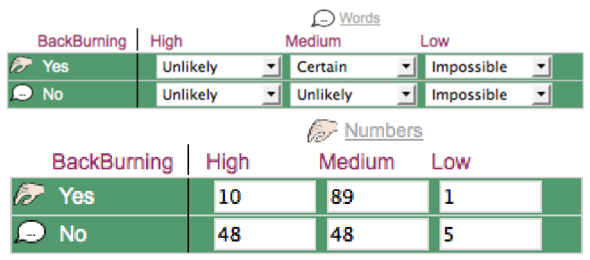
High?	9 %
Medium?	90 %
Low?	1 %

Figure 7: Example of single question parameter elicitation: (above) Verbal cue; (below) Number

quite frustrating. Also, by their very nature, the probabilities elicited using verbal tags will be inaccurate, as the cues are limited to only some probabilities, and they are nearly always changed during normalization. Hence in Phase 3 the parameter elicitation was changed to a hybrid system, combining both qualitative and quantitative elicitation, catering for an iterative stepwise refinement of the probabilities.

Basic elicitation is done via single questions, where the user can switch between "Words" and "Numbers"; an example of these alternatives, for a FIRE threat node, with a single BACKBURNING action parent node, is shown in Figure 7. In single question mode, the "Continue" option takes the user to the next combination of parent values - that is, the next row in the CPT. Note that the user is told the question is "X/Y", meaning Xth question out of Y total questions.

Feedback on the Phase 2 prototype also indicated that the users found it hard to "calibrate" their answers when answering these single questions, so Phase 3 included an alternative table interface, allowing visualisation and flexible navigation of the CPT (in either word or number form). An example is shown in Figure 8. Navigation of the table can be done by either clicking on a summary table, or by selecting a parent state combination via drop down boxes , including an


Words

BackBurning	High	Medium	Low
Yes	Unlikely	Certain	Impossible
No	Unlikely	Unlikely	Impossible

Numbers

BackBurning	High	Medium	Low
Yes	10	89	1
No	48	48	5

Figure 8: Example of table parameter elicitation: (above) Verbal cue; (below) Number

"any" state.[8] Users can switch between a single question (i.e., a parent state combination) and the full table view. ABC-BN also provides visualization of the parameter elicitation progress, incorporating a summary table (not shown) with colour coding for questions that are either complete, incomplete or in need of revision (i.e., when there have been structural changes).

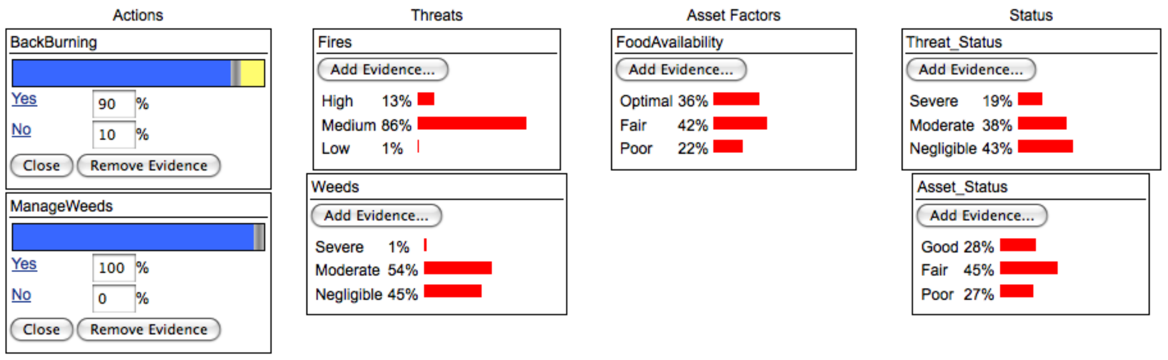
6 Reasoning in ABC-BN

ABC-BN's reasoning component (developed in Phase 2), called "Scenarios", allows the user to enter any combination of exact or likelihood evidence and provides a visualization of the posterior distributions after belief updating. The screen is layed out in four columns – Actions, Threats, Asset Factors, and Status – corresponding to the layers in the BN template, while the arcs in the underlying network are hidden. is shown in Figure 9.

To enter evidence, the user clicks on a node's "Add Evidence..." button, which changes the node view to show a slider for evidence. This rectangular slider is split into as many parts as there are states, where the length of each part represents the probability of a state. The user can either drag the dividers, or enter number values, to enter the desired likelihood evidence. Alternatively, the user can simply select a state if the evidence is exact.

In Phase 3, the reasoner was extended with a similarity analysis report, which summarizes the current scenario, that is, the model and the entered set of evidence; an example is shown in Figure 10. As well as reporting the calculated probabilities for all the nodes, column 1 shows the difference between the posteriors and the initial distribution (before evidence was entered), while column 2 shows differences between the final distribution and distribution given *only* the Action node evidence.

[8]The "any" state is useful if one parent (or a set of parents) dominate in a particular combination, rendering other parents inconsequential.



Actions

BackBurning

Yes 90 %

No 10 %

Close | Remove Evidence

ManageWeeds

Yes 100 %

No 0 %

Close | Remove Evidence

Threats

Fires

Add Evidence...

High	13%
Medium	86%
Low	1%

Weeds

Add Evidence...

Severe	1%
Moderate	54%
Negligible	45%

Asset Factors

FoodAvailability

Add Evidence...

Optimal	36%
Fair	42%
Poor	22%

Status

Threat_Status

Add Evidence...

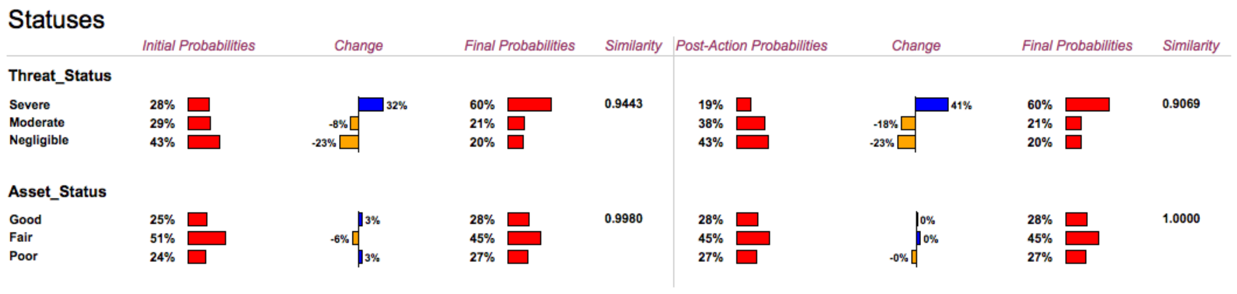
Severe	19%
Moderate	38%
Negligible	43%

Asset_Status

Add Evidence...

Good	28%
Fair	45%
Poor	27%

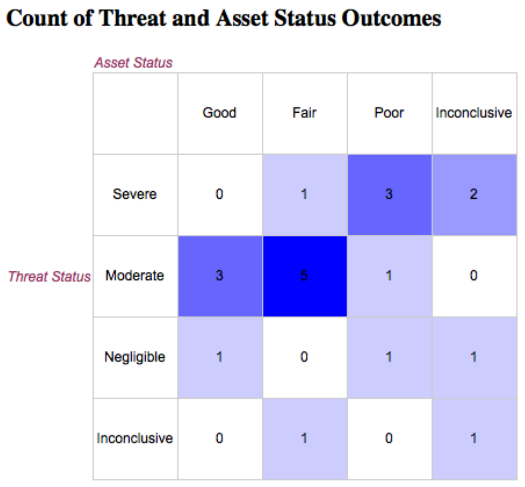
Figure 9: Scenario tester page



Statuses

	Initial Probabilities	Change	Final Probabilities	Similarity	Post-Action Probabilities	Change	Final Probabilities	Similarity
Threat_Status								
Severe	28%	32%	60%	0.9443	19%	41%	60%	0.9069
Moderate	29%	-8%	21%		38%	-18%	21%	
Negligible	43%	-23%	20%		43%	-23%	20%	
Asset_Status								
Good	25%	3%	28%	0.9980	28%	0%	28%	1.0000
Fair	51%	-6%	45%		45%	0%	45%	
Poor	24%	3%	27%		27%	-0%	27%	

Figure 10: Similarity analysis report



Count of Threat and Asset Status Outcomes

Threat Status \ Asset Status	Good	Fair	Poor	Inconclusive
Severe	0	1	3	2
Moderate	3	5	1	0
Negligible	1	0	1	1
Inconclusive	0	1	0	1

Figure 11: Outcome summary page

7 OUTCOME SUMMARY

This component was added in Phase 3, integrating the BN models with the monitoring and reporting functions of ABC. For each model, the user enters evidence for the BN, for any of the ACTION, THREAT or ASSET-FACTOR nodes, along with meta-data. This evidence is stored in the database with the BN. The model is then re-run with this evidence, and an Outcome Report is produced. This report contains a summary of the posterior probabilities of the two STATUS nodes (given the evidence), plus metadata on all included factors (actions, threats and asset factors), such as the source of the information (observed data, expert or/and literature).

As well as reports on individual items, the system also produces a summary table with categorization, across a user-specified set of items; an example is shown in Figure 11. This table is an aggregation of the classification using the highest posterior probabilities for STATUS nodes computed for each BN, given the outcomes evidence added by the monitor. Results are defined as Inconclusive if the difference between the two most probable states is less than the "inconclusive threshold", a global variable (modifiable by administrators).

8 EVALUATION

The prototype developed in Phase 2 was trialled in May-June, 2009 in 5 elicitation workshops around the state. It was used to generate BNs of 100 threatened species and communities. In these trials, the facilitators were DSE personnel[9] overseeing the development of ABC-BN, while the workshop participants were other ABC monitors. These gave very positive

[9]The facilitators were authors Moorrees and Lucas – ABC administrators, as well as item and location monitors.

feedback, as well as suggestions for improvement that were adopted for Phase 3.

Phase 3 was deployed early April, 2011, with the first round of outcomes for 100 models due at the end of June. We intend to undertake a more formal evaluation of both the tool, and of the individual models developed by it. The tool can be evaluated by measuring the rate of elicitation (e.g. number of models, number of parameters), with qualitative feedback from the experts regarding their level of comfort using the tool. The quality of the models is harder to assess, but we intend to have a selection (10%) reviewed by experts not involved in the elicitation or approval process.

9 CONCLUSION

We have described ABC-BN, a tool for building, maintaining and using Bayesian networks in an existing environmental management application. In contrast to most knowledge engineering of ecological applications to date, ABC-BN supports the construction of a large number of simple, standardized BNs over a relatively short time period. It supports iterative and incremental model construction, including hybrid qualitative and quantitative parameter elicitation. It allows users to do what-if predictive and diagnostic scenario reasoning, and the BN models are integrated with monitoring data and reporting, providing outcome status reports for both individual items as well as status summaries for sets of items. Overall, the tool allows managers to estimate the costs associated with actions to mitigate threats and objectively assess the relative importance of actions for different species populations and community occurrences.

References

Boehm, B. W. (1988). A spiral model of software development and enhancement. *IEEE Computer*, 61–72.

Boneh, T. (2010). *Ontology and Bayesian decision networks for supporting the meteorological forecasting process*. Ph. D. thesis, Clayton School of Information Technology, Monash University.

Borsuk, M., C. Stow, and K. Reckhow (2004). A Bayesian network of eutrophication models for synthesis, prediction, and uncertainty analysis. *Ecological Modelling 173*(2-3), 219–239.

Brooks, F. (1995). *The Mythical Man-Month: Essays on Software Engineering* (Second ed.). Reading, MA: Addison-Wesley.

Chee, Y., M. Burgman, and J. Carey (2005). Use of a Bayesian network decision tool to manage environmental flows in the Wimmera river, Victoria. Report No. 4, LWA/MDBC Project UMO43: Delivering Sustainability Through Risk Management, University of Melbourne, Australia.

DSE (2009). Actions for biodiversity conservation (abc): Managing our threatened species and communities. `http://www.dse.vic.gov.au/DSE/nrenpa.nsf/LinkView/487AAFDF542DDAC6CA25703F001C16FEB44D7EEB86E4BDFDCA257115001408B8`.

Hope, L., A. Nicholson, and K. Korb (2002). Knowledge engineering tools for probability elicitation. Technical report 2002/111, School of Computer Science and Software Engineering, Monash University.

Jensen, F. V. and T. D. Nielsen (2007). *Bayesian networks and decision graphs* (2nd ed.). New York: Springer Verlag.

Korb, K. B. and A. E. Nicholson (2010). *Bayesian artificial intelligence* (2nd ed.). Chapman & Hall/CRC.

Laskey, K. and S. Mahoney (2000). Network engineering for agile belief network models. *IEEE: Transactions on Knowledge and Data Engineering 12*(4), 487–498.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems*. San Mateo, CA: Morgan Kaufmann.

Pollino, C., O. Woodberry, A. Nicholson, K. Korb, and B. T. Hart (2007). Parameterisation of a Bayesian network for use in an ecological risk management case study. *Environmental Modelling and Software 22*(8), 1140–1152.

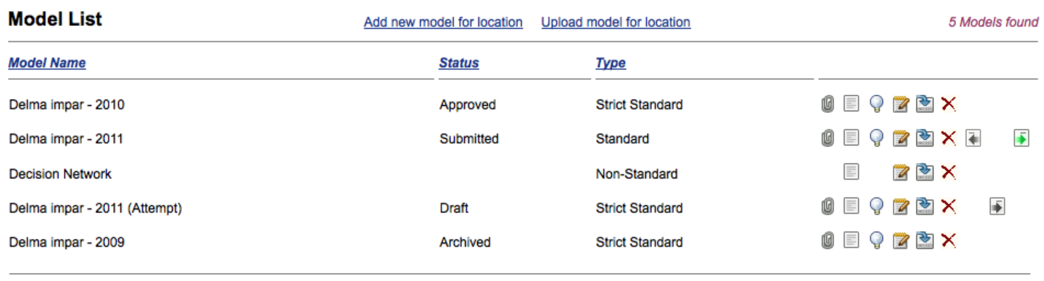
Renken, H. and P. J. Mumby (2009). Modelling the dynamics of coral reef macroalgae using a Bayesian belief network approach. *Ecological Modelling 220*(9-10), 1305 – 1314.

Smith, C., A. Howes, B. Price, and C. McAlpine (2007). Using a Bayesian belief network to predict suitable habitat of an endangered mammalThe Julia Creek dunnart (Sminthopsis douglasi). *Biological Conservation 139*, 333–347.

Uusitalo, L. (2007). Advantages and challenges of Bayesian networks in environmental modelling. *Ecological Modelling 203*(3-4), 312–318.

van der Gaag, L. C., S. Renooij, C. L. M. Witteman, B. M. P. Aleman, and B. G. Taal (1999). How to elicit many probabilities. San Francisco, CA, pp. 647–654.

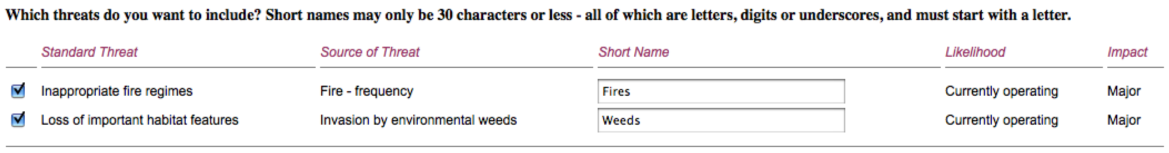
Varis, O. and S. Kuikka (1999). Learning Bayesian decision analysis by doing: lessons from environmental and natural resources management. *Ecological Modelling 119*, 177–195(19).



Model List Add new model for location Upload model for location *5 Models found*

Model Name	Status	Type
Delma impar - 2010	Approved	Strict Standard
Delma impar - 2011	Submitted	Standard
Decision Network		Non-Standard
Delma impar - 2011 (Attempt)	Draft	Strict Standard
Delma impar - 2009	Archived	Strict Standard

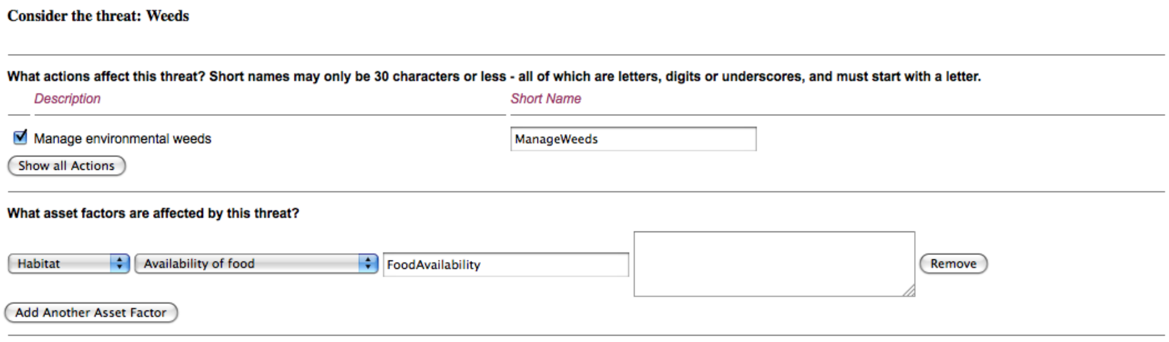
Figure 12: Model management page



Which threats do you want to include? Short names may only be 30 characters or less - all of which are letters, digits or underscores, and must start with a letter.

	Standard Threat	Source of Threat	Short Name	Likelihood	Impact
✓	Inappropriate fire regimes	Fire - frequency	Fires	Currently operating	Major
✓	Loss of important habitat features	Invasion by environmental weeds	Weeds	Currently operating	Major

Figure 13: Threat selector page



Consider the threat: Weeds

What actions affect this threat? Short names may only be 30 characters or less - all of which are letters, digits or underscores, and must start with a letter.

	Description	Short Name
✓	Manage environmental weeds	ManageWeeds

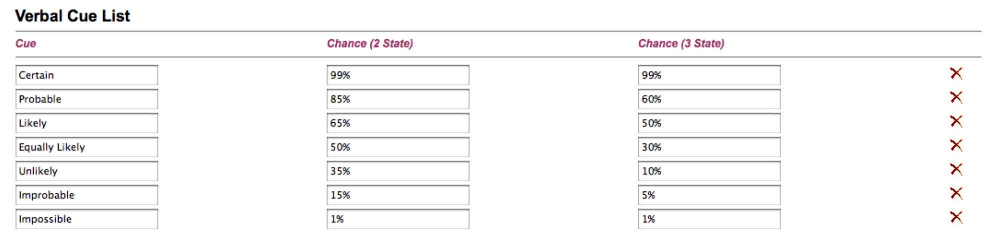
Show all Actions

What asset factors are affected by this threat?

Habitat | Availability of food | FoodAvailability | Remove

Add Another Asset Factor

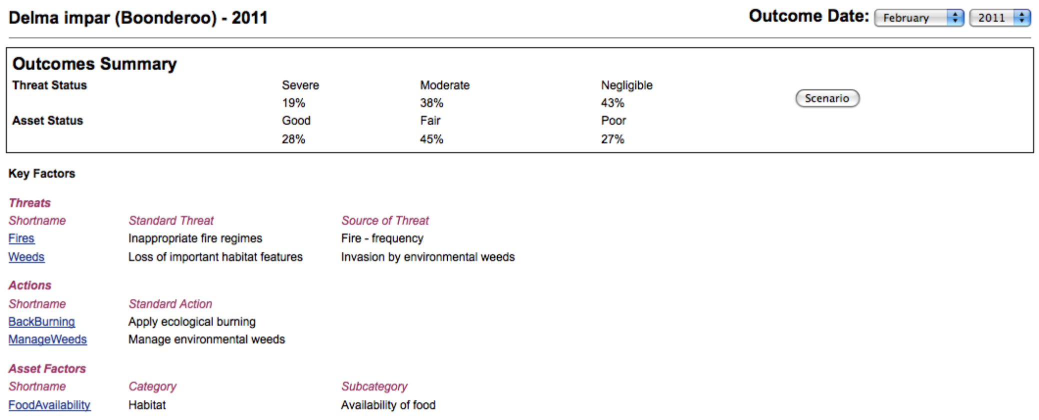
Figure 14: Action and Asset Factor selector page



Verbal Cue List

Cue	Chance (2 State)	Chance (3 State)
Certain	99%	99%
Probable	85%	60%
Likely	65%	50%
Equally Likely	50%	30%
Unlikely	35%	10%
Improbable	15%	5%
Impossible	1%	1%

Figure 15: Verbal cues page



Delma impar (Boonderoo) - 2011 **Outcome Date:** February 2011

Outcomes Summary

Threat Status	Severe	Moderate	Negligible	Scenario
	19%	38%	43%	
Asset Status	Good	Fair	Poor	
	28%	45%	27%	

Key Factors

Threats

Shortname	Standard Threat	Source of Threat
Fires	Inappropriate fire regimes	Fire - frequency
Weeds	Loss of important habitat features	Invasion by environmental weeds

Actions

Shortname	Standard Action
BackBurning	Apply ecological burning
ManageWeeds	Manage environmental weeds

Asset Factors

Shortname	Category	Subcategory
FoodAvailability	Habitat	Availability of food

Figure 16: Example outcome page

Does Query-Based Diagnostics Work?

Parot Ratnapinda[1] and **Marek J. Druzdzel**[1,2]
par34@pitt.edu, marek@sis.pitt.edu
[1] Decision Systems Laboratory, School of Information Sciences and Intelligent Systems Program,
University of Pittsburgh, Pittsburgh, PA 15260, USA
[2] Faculty of Computer Science, Białystok University of Technology, Wiejska 45A, 15-351 Białystok, Poland

Abstract

Query-based diagnostics (Agosta, Gardos, & Druzdzel, 2008) offers passive, incremental construction of diagnostic models that rest on the interaction between a diagnostician and a computer-based diagnostic system. Effectively, this approach minimizes knowledge engineering, the main bottleneck in practical application of Bayesian networks. While this idea is appealing, it has undergone only limited testing in practice. We describe a series of experiments that subject a prototype implementing passive, incremental model construction to a rigorous practical test. We show that the prototype's diagnostic accuracy reaches reasonable levels after merely tens of cases and continues to increase with the number of cases, comparing favorably to state of the art approaches based on learning.

1 Introduction

Even though Bayesian network (BN) models (Pearl, 1988) have proven useful in diagnostic domains, they are quite hard to field in practice. Interestingly, it is not computational complexity that is critical here. The main hurdle in applying Bayesian networks to complex diagnostic problems seems to be model building.

One way of addressing this problem is learning models from data accrued over past cases. Given a sufficiently large set of past cases, we can learn both the structure and the parameters of Bayesian networks (Cooper & Herskovits, 1992; Pearl & Verma, 1991; Spirtes, Glymour, & Scheines, 1993). Although model construction from data can significantly reduce knowledge engineering effort, learning faces other problems, such as small data sets, unmeasured variables, missing data, and selection bias. Collections of past cases that are large and complete enough are often hard to find. There are many complex devices that do not break too often or, at least, are not supposed to break often. When dealing with some devices, it is not uncommon to spend months on constructing models that become outdated soon after deployment. Building Bayesian networks requires such a considerable effort on the part of knowledge engineers and domain experts that it is considered the main bottleneck in this area.

There have been several lines of research outside of learning from data that focus on model building. The first approach focuses on providing more expressive building tools. The Noisy-OR model (Pearl, 1988; Henrion, 1989) and its generalizations (Díez, 1993; Srinivas, 1993) simplify the representation and elicitation of independence interactions among multiple causes. Heckerman (1990) developed the concept of *similarity networks* in order to facilitate structure building and probability elicitation. The second approach, usually referred to knowledge-based model construction (KBMC), emphasizes aiding model building by automated generation of decision models from a domain knowledge-base guided by the problem description and observed information (see a special issue at the journal IEEE Transactions on Systems, Man and Cybernetics on the topic of KBMC (Breese, Goldman, & Wellman, 1994)). The third approach is to apply system engineering and knowledge engineering techniques for aiding the process of building Bayesian networks. Laskey and Mahoney (1996; 1997) address the issues of modularization, object-orientation, knowledge-base, and evaluation in a spiral model of development cycle. Koller and Pfeffer (1997; 1999) developed Object-Oriented Bayesian Networks (OOBN) that use objects as organizational units to reduce the complexity of modeling and increase the speed of inference. Lu *et al.* (2000) propose mechanism-based model construction, in which models are constructed from a collection of mechanisms based on scientific laws or pieces of existing models. (Ibargengoytia, Vadera, & Sucar, 2006) propose to learn a Bayesian network model for a normal mode of operation, for which data are typically available, and then detect anomalies as deviations from this model.

Agosta *et al.* (2008) went further and proposed an approach that eliminates knowledge engineering altogether. In what they call query-based diagnostics, they propose embedding a diagnostic aid in existing systems for diagnostic record keeping. A diagnostician working on a case, recording symptoms and other findings along with the final diagnosis, without being aware of it, participates in constructing a simplified Bayesian network model that supports future cases. From the theoretical perspective, the idea is a combination of structure elicitation and incremental learning. The diagnostician provides the system with a basic distinction between symptoms, background information, and the final diagnosis. Past cases solved by diagnosticians can provide considerable information about the domain. Every new case acquired by the system adds useful information and, in the long run, leads to building a usable model. As cases accrue, the system refines the structure and the parameters of such model and improves its accuracy.

While this idea is appealing, it has undergone only limited testing in practice. To the best of our knowledge, there are two existing prototypes implementing this approach. An industrial prototype of the system has been implemented and fielded at Intel and tested in the domain of diagnostics and corrective maintenance of factory equipment (Agosta, Khan, & Poupart, 2010). A widely accessible prototype, called Marilyn (Pols, 2007), was tested in a limited setting of a help desk at a university computing laboratory (Ratnapinda & Druzdzel, 2009). Neither of the two prototypes and the very idea of a system that eliminated completely the knowledge engineering phase and learns successively from diagnostic cases have undergone a formal evaluation. In this paper, we attempt to evaluate one of these two prototypes (Marilyn) systematically, based on several real data sets, obtained from the Irvine Machine Learning Repository.[1] We show that the prototype's diagnostic accuracy reaches reasonable levels after merely tens of cases and continues to increase with the number of cases, comparing favorably with state of the art approaches based on learning.

2 Background

We will start with a brief review of the technology involved in a query-based diagnostic prototype like Marilyn, notably Bayesian networks, noisy-OR gates, and the EM algorithm.

[1] `http://archive.ics.uci.edu/ml/`

2.1 Bayesian Networks

Bayesian networks (Pearl, 1988) are acyclic directed graphs representing joint probability distributions over sets of variables. Every node is the graph represents a random variable. Lack of an arc between two nodes represents conditional independence between the variables that these nodes represent. Nodes are quantified by means of conditional probability tables (CPTs), representing the probability distribution of the variables that they represent conditional on their parent variables in the graph. Nodes without parents are specified by prior probability distributions. The joint probability distribution over a set of variables $\mathbf{X} = \{X_1, \ldots, X_n\}$ can be obtained by taking the product of all prior and conditional probability distributions:

$$\Pr(\mathbf{X}) = \Pr(X_1, \ldots, X_n) = \prod_{i=1}^{n} \Pr(X_i|Pa(X_i)) \ . \quad (1)$$

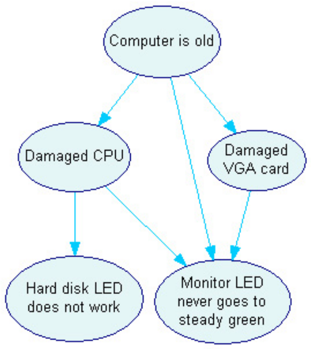

Computer is old
Damaged CPU
Damaged VGA card
Hard disk LED does not work
Monitor LED never goes to steady green

Figure 1: An example Bayesian network modeling computer hardware problems

Figure 1 shows a simple Bayesian network modeling two computer hardware problems. The variables in this model are: *Computer is old*, *Damaged CPU*, *Damaged VGA card*, *Hard disk LED does not work* and *Monitor LED never goes to steady green*. Each of the variables in the model is binary, i.e., has two outcomes: *True* and *False*.

A directed arc between *Damaged CPU* and *Hard disk LED does not work* indicates that *Damaged CPU* will affect the probability that *Hard disk LED does not work*. Similarly, an arc from *Computer is old* to *Damaged VGA card* indicates that computer age influences the likelihood of a damaged VGA card.

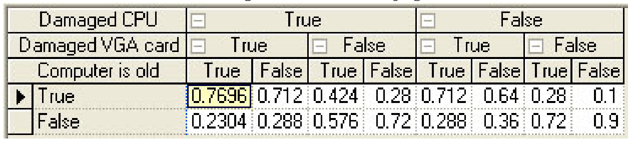
The most important type of reasoning in Bayesian networks is known as belief updating and amounts to computing the probability distribution over variables of interest given the evidence. For example, in the model of Figure 1, the variable of interest could be *Damaged*

CPU and the BN could compute the posterior probability distribution over this node given the observed values of *Computer is old*, *Hard disk LED does not work*, and *Monitor LED never goes to steady green.* Once the network has updated the probability values, these can be used to make a diagnostic decision.

2.2 The Leaky Noisy-OR Gate

Bayesian networks suffer from a practical problem: Because CPTs represent the probability distribution of a node conditional on all combinations of parent variables, their size grows exponentially with the number of parents. Table 1 shows the CPT for the node *Monitor LED never goes to steady green.* The node has three parents and the size of its CPT is $2^3 = 8$.

Table 1: Conditional probability table of the node *Monitor LED never goes to steady green*



Damaged CPU	True				False			
Damaged VGA card	True		False		True		False	
Computer is old	True	False	True	False	True	False	True	False
True	0.7696	0.712	0.424	0.28	0.712	0.64	0.28	0.1
False	0.2304	0.288	0.576	0.72	0.288	0.36	0.72	0.9

One solution to the exponential growth of CPTs is application of Independence of Causal Influences (ICI) models (Díez & Druzdzel, 2006). The ICI models assume that parent variables can cause the effect independently of each other. This assumption allows to reduce the number of parameters needed to specify an interaction from exponential to linear in the number of parents.

Marilyn is based on the ICI model called the noisy-OR gate (Pearl, 1988; Henrion, 1989). The noisy-OR gate is a probabilistic extension of the deterministic OR gate. Each variable in a noisy-OR gate is binary and has two states: *present* and *absent.* Presence of the parent variables X_i effects the presence of the child variable Y. If all the parent variables are *absent*, then the child variable is also *absent.*

In general, it is infeasible to explicitly include all possible causes of an effect. Marilyn uses an extension of the noisy-OR gate called *leaky noisy-OR* gate (Henrion, 1989; Díez, 1993). The parameter p_i of a leaky noisy-OR gate is defined as the probability that Y will be true if X_i is present and every other parent of Y, including unmodeled causes of Y (the leak), are absent.

2.3 The EM Algorithm

The Expectation-Maximization (EM) algorithm (Dempster, Laird, & Rubin, 1977) is widely used to compute maximum likelihood estimates given incomplete data. Implementations of the EM algorithm have been successfully applied to compute parameters of ICI models, including the noisy-OR model (Natarajan et al., 2005; Vomlel, 2006; Abbasi, Dailey, Afzulpurkar, & Uno, 2010). EM consists of two steps: (1) the expectation step (E-step) uses current parameters to compute the expected values of the missing data, and (2) the maximization step (M-step), in which the maximum likelihood of the parameters are estimates based on the current expected values of the data. Then, the EM process repeats until it converges to the maximum likelihood.

3 Marilyn

Marilyn is a web-based application that implements the idea of query-based diagnostics, i.e., passive construction of diagnostic decision models. It is written in C# and ASP.NET, using a Microsoft SQL database to store data. It utilizes the Bayesian reasoning engine SMILE[2] running under the Microsoft Windows Vista Server. Figure 2 shows Marilyn's architecture. Marilyn appears to the user diagnostician as a computer program for logging case data. The user interacts with it though a web browser, entering elements of the case at hand. The case data are entered in free text format, although the system performs simple text matching to suggest values entered in prior cases. Marilyn presents the user unobtrusively with a list of most likely diagnoses implied by the observations entered so far, suggests additional observations to make and tests to perform. Behind the screen, Marilyn constructs a Bayesian network from the prior cases stored in the database and, ultimately, adds the current case to the database.

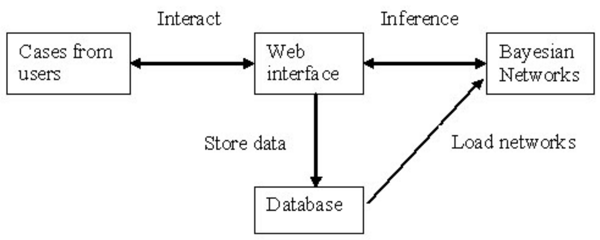


Figure 2: Marilyn's architecture

3.1 Model Structure

The Bayesian networks constructed by Marilyn use a simplified structure called the BN3M model (Kraaijeveld & Druzdzel, 2005), which distinguishes three fundamental types of variables:

[2]http://genie.sis.pitt.edu/

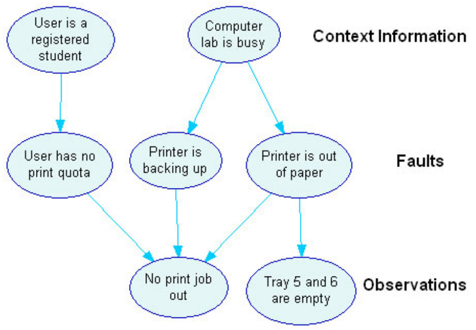


Figure 3: An example of a BN3M model

- *Fault* variables, which represent the problems that the diagnostician wants to identify (e.g., a disease or a device malfunction).
- *Observation* variables, which include observed symptoms and results of diagnostic tests.
- *Context* variables, which are the background, history, or other information known by the technician performing the diagnosis that may influence the probability of a fault and, therefore, are relevant to the diagnosis.

The structure of BN3M networks consists of three levels, with the *context information* variables on the top, the *fault* variables in the middle, and the *observation* variables at the bottom. Influences are possible only between neighboring layers. Figure 3 shows an example of this structure. The first context variable, *User is a registered student*, influences the variable *User has no print quota*. The second context variable *Computer lab is busy* influences the faults *Printer is backing up* and *Printer is out of paper*. *No print job out* is influenced by any three of the fault variables. *Trays 5 and 6 are empty* is influenced only by the fault *Printer is out of paper*.

3.2 Model Construction in Marilyn

When Marilyn starts, it constructs a Bayesian network from the existing database (in the very beginning, this database is empty). The database consists of six tables: *arcs*, *diagnosis*, *domains*, *nodes*, *lablog*, and *emlog*. The first four tables store the information about causal interactions among variables, the number of diagnostic sessions that have been stored by the system, the diagnostic domains, and variables, respectively. The last two tables store data for each session and store the diagnostic logs used in refining the model parameters.

Marilyn constructs the BN3M structure by going through all diagnostic cases entered so far and connecting all context variables and and all observation variables to the fault node observed in the case (i.e., the final diagnosis, as indicated by the diagnostician). This provides a graph skeleton that is subsequently quantified in the following way. All prior probability distributions are set to 0.1/0.9. All conditional probability distributions are set to 0.8/0.2. The EM algorithm, which Marilyn subsequently invokes, treats these initial values as a prior probability distributions and refines them by means of the records accrued during the past diagnostic cases. While the above priors are arbitrary, we found that they are capable of inducing reasonable behavior on the part of the model, even if the number of existing records is small. There is an increasing body of evidence that the precise values of the parameters are not crucial in practice (Pradhan, Henrion, Provan, del Favero, & Huang, 1996; Oniśko & Druzdzel, 2011).

The final model, i.e., model obtained after the parameter refinement stage, is used by Marilyn to generate a list of most likely diagnoses for the current diagnostic case.

4 Empirical Evaluation

4.1 The Data

We tested the accuracy of Marilyn on four different real data sets listed in Table 2. The computing lab data set was collected over the course of two semesters at a help desk of a University of Pittsburgh campus computing lab. Typical campus computing lab help desk problems involve printing problems and printer troubleshooting. Among the four hundred cases in the data set, there are a total of 16 different observations, 12 different context variables, and 21 different problems. The remaining three data sets originate from the UCI Machine Learning repository and were selected based on the following four criteria:

- The data include a known class variable.
- The attribute types of all variables are discrete. We wanted to avoid the need for discretization, which could become a factor confounding our experiment.
- The number of cases in the data file should be over 100, which we believe to be large enough for the purpose of the experiment.
- The data should have been used in the literature in the past, so that we have information about baseline performance of learning algorithms.

The three Irvine repository data sets that fulfilled the above requirements were *SPECT Heart*, *Breast Can-*

cer and *Lymphography.* Their properties are listed in Table 2. #I in the table denotes the number of data records, #A denotes the number of attributes, #CV denotes the number of class variables, and MV describes presence of missing values.

Table 2: Data sets used in our experiments.

Dataset	**#I**	**#A**	**#CV**	**MV**
Computer lab	400	49	21	No
SPECT Heart	267	23	2	No
Breast Cancer	286	10	2	Yes
Lymphography	148	19	4	No

4.2 Methodology

We wanted to test the accuracy of MARILYN as a function of the number of cases that it has seen on each of the data sets listed in Table 2. This is of interest because the idea of query-based diagnostics is meant to work especially when there are no data that can be used to learn a model. Availability of a complete data set would make MARILYN useless, as the model could be learned from data by means of any of the Bayesian network learning methods available in the literature.

We imitated MARILYN's diagnostician's work-flow, which consists of entering three types of information: context information, observations, and the final diagnosis. While, in case of the computing lab help desk data, we had full knowledge of the three types of information, we did not know which of the features in the Irvine medical data sets were context variables and which were observations. Effectively, we treated all features in these data sets as observations. This is a conservative assumption, as it is an additional handicap for MARILYN in the experiments. The effect of our treatment of the medical data was that Marilyn constructed two layer BN2O networks in these cases, similarly to the QMR-DT model (Middleton et al., 1991).

We ran MARILYN 30 times for each data set, randomizing each time the order of records in the data file. The order of the records offered to MARILYN may affect its performance and presenting different orders allows us to observe a range of behaviors. We used the simplest possible criterion in making a diagnostic decision and assumed that the most likely diagnosis is MARILYN's final diagnosis. This is, again, a conservative assumption, as the system displays the top n most likely diagnoses and this gives the user a chance to improve on the system, especially in the early stages, when the model is very crude.

4.3 The Results

4.3.1 MARILYN Results

We calculated MARILYN's cumulative accuracy after each record, so as to know how this performance develops as a function of the number of diagnostic cases that the system has seen. Figure 4 shows the average accuracy of MARILYN as a function of the number of cases for each of the four data sets with range of the curves (vertical bars) plotted for selected number of records. The plots show that while MARILYN was rather weak in the beginning (during the first thirty cases or so), it became quite accurate after roughly 70 to 100 cases (this varied per data set). Interestingly, in case of the SPECT data set, MARILYN reached the accuracy of over 60% after fewer than ten cases. In all data sets, 40 or so cases were sufficient to reach a reasonable accuracy. This accuracy not only improved over time but also improved reliably, as indicated by smaller variance in the results of different random orders of records. Interestingly, there is some similarity between the plots of MARILYN's performance, as in Figure 4, and the so called *power curve of practice* in the psychology literature (Newell & Rosenbloom, 1981).

4.3.2 MARILYN's Relative Performance

Cumulative accuracy for the last record entered is the final accuracy result of MARILYN on the data set. MARILYN's final performance on the four data sets was 90.25%, 78.75%, 77.18%, and 69.95% for the *Computer Lab*, *SPECT Heart*, *Breast Cancer* and *Lymphography* data respectively (see the extreme right cumulative performance in Figure 4). It has to be added that in achieving this result MARILYN has seen (i.e., was trained on) the average of 50% of the records. When processing the first case, Marilyn has seen zero prior cases, when processing the 10th case, it has used only 9 preceding cases, when processing the last, nth case, it has seen $n-1$ preceding cases. The average number of training records is thus $n/2$.

Table 3: Accuracy comparison results with Bayesian approaches using leave-one-out cross validation

Dataset	MARILYN	**Naive Bayes**	**GTT**
CompLab	94.50%	94.25%	91.25%
SPECT	79.40%	79.40%	78.65%
BC	68.18%	42.57%	47.97%
Lymph	81.08%	66.08%	67.83%

In order to disambiguate the specific procedure that we used to obtain MARILYN's cumulative performance from the capability of the learning function by it-

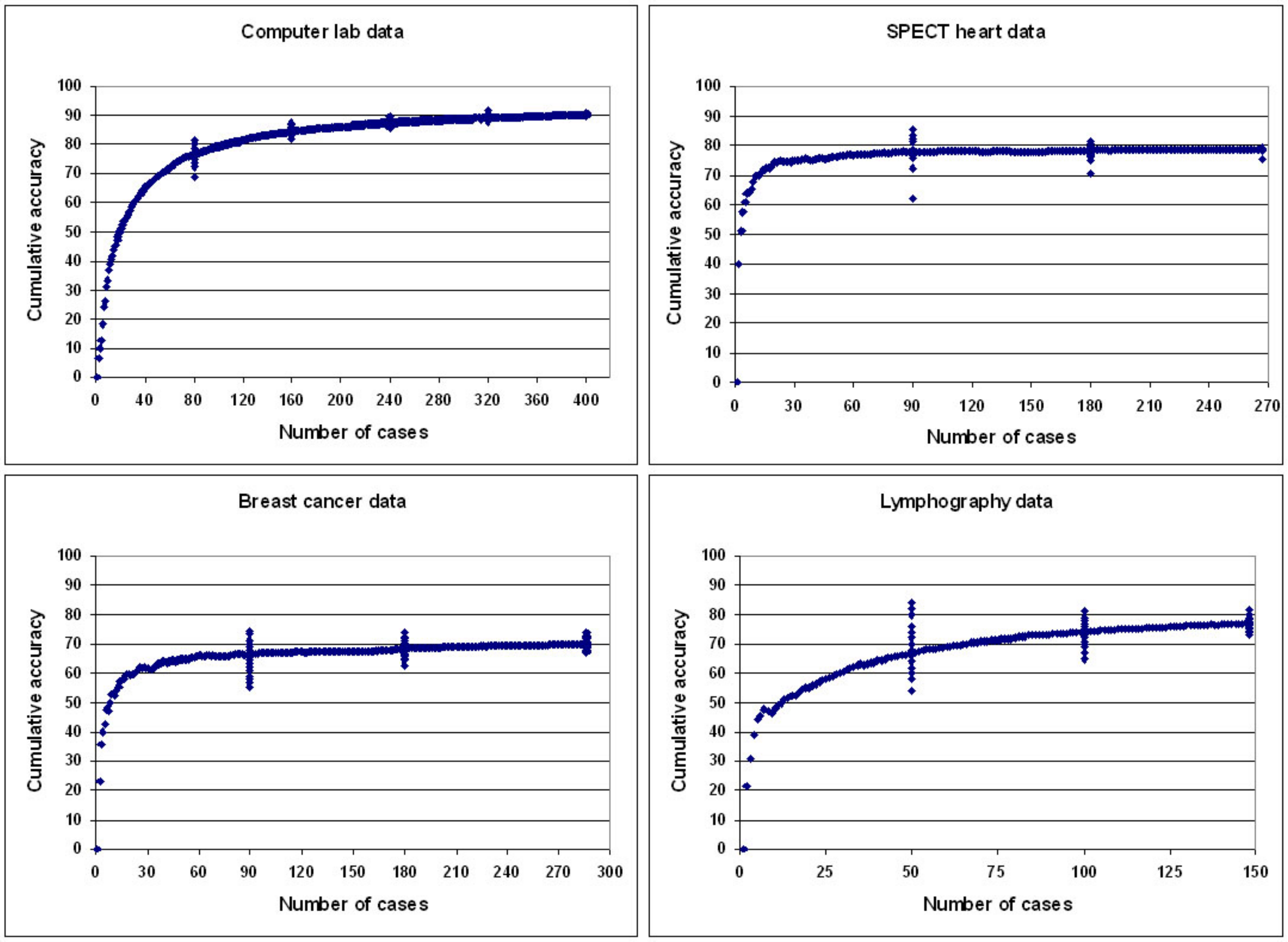


Figure 4: Marilyn's cumulative accuracy as a function of the number of cases seen

self, we performed an experiment in which we allowed Marilyn to learn from all available records alongside with two Bayesian learning algorithms: (1) Naive Bayes (Langley, Iba, & Thompson, 1992), and (2) a Bayesian search algorithm Greedy Thick Thinning (GTT) (Dash & Druzdzel, 2003). We used the leave-one-out cross validation to measure the accuracy of the three classifiers, assuming that the diagnosis is correct when the most probable class matches the correct class. We show the results of this experiment in Table 3. Marilyn performed better than Naive Bayes and GTT on all data sets. We believe that some of Marilyn's power comes from its priors and structural information extracted from the data.

The three data sets that we chose for our experiments have been subject of experiments published in the literature. The best accuracy result for SPECT heart data with CLIP3 machine learning algorithm is 84% (Kurgan, Cios, Tadeusiewicz, Ogiela, & Goodenday, 2001). The best accuracy achieved on the Breast cancer data was by means of *k*-nearest neighbor (*k*-NN) algorithm and amounted to 79.5% (Kononenko, Bratko, & Kukar, 1997). The best accuracy on the Lymphography set was achieved by means of the Tree-Augmented Naive Bayes algorithm and was 85.47% (Madden, 2002). We compared Marilyn's performance to each of these, repeating the experiment under the same conditions, i.e., with precisely the same cross-validation method as used in the experiments reported in the literature. Table 4 shows the accuracy for each of the data sets and each of the algorithms.

Table 4: Accuracy comparison results with state of the art approaches

Dataset	Marilyn	**CLIP3**	***k*-NN**	**TAN**
SPECT	93.58%	84%	N/A	N/A
BC	73.02%	N/A	79.50%	N/A
Lymph	81.92%	N/A	82.60%	85.47%

While Marilyn's accuracy is typically lower than that of the state of the art learning algorithms, it is certainly in the same ballpark. We would like to point out that the best results reported in the literature belong to different algorithms, i.e., there seems to be no algorithm that is uniformly best on all data sets. If the same algorithm were applied to all four data sets, there is a good chance that its accuracy on some of these could be worse than the accuracy of Marilyn.

5 Conclusion

Query-based diagnostic offers passive, incremental construction of diagnostic models based on the interaction between a diagnostician and a computer-based diagnostic system. Effectively, this approach eliminates knowledge engineering, the main bottleneck in practical application of Bayesian networks.

While this idea is appealing, it has undergone only limited testing in practice. In this paper, we described a series of experiments that subject a prototype implementing passive, incremental model construction to a rigorous practical test. Data obtained from the Irvine repository made the evaluation fairly realistic. The results of our experiments show that a system like Marilyn is capable of giving reasonable suggestions after a modest number of observed cases. Performance in the order of 70-90% typically occurred not later than after roughly 40 cases. Even though this experiment offers just a few data points and this type of systems need to be tested more in practice, we believe that the result is very promising and compares favorably with state of the art approaches based on learning.

Acknowledgements

This work has been supported by the National Institute of Health under grant number U01HL101066-01, the Government of the Kingdom of Thailand, the Air Force Office of Scientific Research grant FA9550–06–1–0243, and by Intel Research. Implementation of Marilyn is based on SMILE, a Bayesian inference engine developed at the Decision Systems Laboratory and available at `http://genie.sis.pitt.edu/`. We would like to express our thanks to the DSL members for their valuable suggestions. Special thanks go to Erik Pols, Mark Voortman and Adam Zagorecki.

References

Abbasi, A., Dailey, M., Afzulpurkar, N., & Uno, T. (2010, March). Student mental state inference from unintentional body gestures using dynamic Bayesian networks. *Journal on Multimodal User Interfaces*, *3*(1), 21–31.

Agosta, J. M., Gardos, T. R., & Druzdzel, M. J. (2008). Query-based diagnostics. In M. Jaeger & T. D. Nielsen (Eds.), *Proceedings of the Fourth European Workshop on Probabilistic Graphical Models (PGM–08)* (pp. 1–8). Aalborg, Denmark.

Agosta, J. M., Khan, O. Z., & Poupart, P. (2010). Evaluation results for a query-based diagnostics application. In P. Myllymaki, T. Roos, & T. Jaakkola (Eds.), *Proceedings of the Fifth European Workshop on Probabilistic Graphical Models (PGM–10)* (pp. 1–9). Helsinki, Finland.

Breese, J. S., Goldman, R. P., & Wellman, M. P. (1994). Introduction to the special section on knowledge-based construction of probabilistic and decision models. *IEEE Transactions on Systems, Man and Cybernetics*, *24*(11), 1577–1579.

Cooper, G. F., & Herskovits, E. (1992). A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, *9*(4), 309–347.

Dash, D., & Druzdzel, M. J. (2003). Robust independence testing for constraint-based learning of causal structure. In C. Meek & U. Kjærulff (Eds.), *UAI* (pp. 167–174). Morgan Kaufmann.

Dempster, A., Laird, N., & Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, *39*(1), 1–38.

Díez, F. J. (1993). Parameter adjustment in Bayes networks. The generalized Noisy-OR gate. In *Proceedings of the Ninth Annual Conference on Uncertainty in Artificial Intelligence (UAI–93)* (pp. 99–105). Washington, D.C..

Díez, F. J., & Druzdzel, M. J. (2006). *Canonical probabilistic models for knowledge engineering* (Tech. Rep.). UNED, Madrid, Spain. CISIAD-06-01.

Heckerman, D. (1990, August). Probabilistic similarity networks. *Networks*, *20*(5), 607–636.

Henrion, M. (1989). Some practical issues in constructing belief networks. In L. Kanal, T. Levitt, & J. Lemmer (Eds.), *Uncertainty in Artificial Intelligence 3* (pp. 161–173). New York, N. Y.: Elsevier Science Publishing Company, Inc.

Ibargengoytia, P. H., Vadera, S., & Sucar, L. E. (2006). A probabilistic model for information and sensor validation. *The Computer Journal*, *49*, 113–126.

Koller, D., & Pfeffer, A. (1997). Object-oriented Bayesian networks. In *Proceedings of the Thirteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI–97)* (pp. 302–313). San Francisco, CA: Morgan Kaufmann Publishers.

Kononenko, I., Bratko, I., & Kukar, M. (1997). Application of machine learning to medical diagnosis. In I. Michalski R.S. Bratko & M. Kubat (Eds.), *Machine Learning, Data Mining and Knowledge Discovery: Methods and Applications.* John Wiley & Sons.

Kraaijeveld, P., & Druzdzel, M. J. (2005, June 1–3). *GeNIeRate*: An interactive generator of diagnostic Bayesian network models. In *Working Notes of the 16th International Workshop on Principles of Diagnosis (DX–05)* (pp. 175–180). Monterey, CA.

Kurgan, L. A., Cios, K. J., Tadeusiewicz, R., Ogiela, M. R., & Goodenday, L. S. (2001). Knowledge discovery approach to automated cardiac SPECT diagnosis. *Artificial Intelligence in Medicine*, *23*, 149.

Langley, P., Iba, W., & Thompson, K. (1992). An analysis of Bayesian classifiers. In *Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI–92)* (pp. 223–228). MIT Press.

Laskey, K. B., & Mahoney, S. M. (1997). Network fragments: Representing knowledge for constructing probabilistic models. In *Proceedings of the Thirteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI–97)* (pp. 334–341). San Francisco, CA: Morgan Kaufmann Publishers.

Lu, T.-C., Druzdzel, M. J., & Leong, T.-Y. (2000). Causal mechanism-based model construction. In *Uncertainty in Artificial Intelligence: Proceedings of the Sixteenth Conference (UAI–2000)* (pp. 353–362). San Francisco, CA: Morgan Kaufmann Publishers.

Madden, M. G. (2002). Evaluation of the performance of the Markov blanket Bayesian classifier algorithm. *CoRR*, *cs.LG/0211003*.

Mahoney, S. M., & Laskey, K. B. (1996). Network engineering for complex belief networks. In *Proceedings of the Twelfth Annual Conference on Uncertainty in Artificial Intelligence (UAI–96)* (pp. 389–396). San Francisco, CA: Morgan Kaufmann Publishers.

Middleton, B., Shwe, M., Heckerman, D., Henrion, M., Horvitz, E., Lehmann, H., et al. (1991). Probabilistic diagnosis using a reformulation of the INTERNIST–1/QMR knowledge base: II. Evaluation of diagnostic performance. *Methods of Information in Medicine*, *30*(4), 256–267.

Natarajan, S., Tadepalli, P., Altendorf, E., Dietterich, T. G., Fern, A., & Restificar, A. (2005). Learning first-order probabilistic models with combining rules. In *Proceedings of the 22nd International Conference on Machine Learning* (pp. 609–616). New York, NY, USA: ACM.

Newell, A., & Rosenbloom, P. S. (1981). Mechanisms of skill acquisition and the law of practice. In J. R. Anderson (Ed.), *Cognitive Skill and Their Acquisition* (pp. 1–55). Hillsdale, NJ: Lawrence Erlbaum Associates.

Oniśko, A., & Druzdzel, M. J. (2011). Impact of quality of Bayesian network parameters on accuracy of medical diagnostic systems. In *Working Notes of the 2011 AIME'11 Workshop on Probabilistic Problem Solving in Biomedicine.* Bled, Slovenia.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: Networks of plausible inference.* San Mateo, CA: Morgan Kaufmann Publishers, Inc.

Pearl, J., & Verma, T. S. (1991). A theory of inferred causation. In J. Allen, R. Fikes, & E. Sandewall (Eds.), *Principles of Knowledge Representation and Reasoning: Proceedings of the Second International Conference (KR–91)* (pp. 441–452). Cambridge, MA: Morgan Kaufmann Publishers, Inc., San Mateo, CA.

Pfeffer, A., Koller, D., Milch, B., & Takusagawa, K. T. (1999). SPOOK: A system for probabilistic object-oriented knowledge representation. In *Proceedings of the Fifteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI–99)* (pp. 541–550). San Francisco, CA: Morgan Kaufmann Publishers.

Pols, E. (2007, April). *Marilyn: A guided maintenance system that represents direct probabilistic influences among diagnostic knowledge* (Technical Report). Delft, The Netherlands: Delft University of Technology.

Pradhan, M., Henrion, M., Provan, G., del Favero, B., & Huang, K. (1996, August). The sensitivity of belief networks to imprecise probabilities: An experimental investigation. *Artificial Intelligence*, *85*(1–2), 363–397.

Ratnapinda, P., & Druzdzel, M. J. (2009). Passive construction of diagnostic decision models: An empirical evaluation. In *International Multiconference on Computer Science and Information Technology (IMCSIT–09)* (pp. 601–607).

Spirtes, P., Glymour, C., & Scheines, R. (1993). *Causation, prediction, and search.* New York: Springer Verlag.

Srinivas, S. (1993). A generalization of the noisy-OR model. In *Proceedings of the Ninth Annual Conference on Uncertainty in Artificial Intelligence (UAI–93)* (pp. 208–215). San Francisco, CA: Morgan Kaufmann Publishers.

Vomlel, J. (2006, March). Noisy-OR classifier: Research articles. *International Journal of Intelligent Systems*, *21*, 381–398.

A Web-based Tool for Expert Elicitation in Distributed Teams

Carlo Spaccasassi
IBM Dublin Research Lab
Damastown, Dublin, D15
Ireland
spaccasa@ie.ibm.com

Lea Deleris
IBM Dublin Research Lab
Damastown, Dublin, D15
Ireland
lea.deleris@ie.ibm.com

Abstract

We present in this paper a web-based tool developed to enable expert elicitation of the probabilities associated with a Bayesian Network. The motivation behind this tool is to enable assessment of probabilities from a distributed team of experts when face-to-face elicitation is not an option, for instance because of time and budget constraints. In addition to the ability to customize surveys, the tool provides support for both quantitative and qualitative elicitation, and offers administrative features such as elicitation surveys management and probability aggregation.

1 Introduction

There is a thriving research community that studies techniques for learning the structure and parameters of a belief network from data [1]. However, when there is no relevant data available, or any literature to guide the construction of the model, the network must be elicited from the individuals whose beliefs are being captured - such a person is often referred to as the *domain expert*, or simply expert. Both the structure and the parameters of a belief network need to be elicited. Often it is easier to construct the *structure* of a belief network, as compared to eliciting the *parameters*, i.e. the conditional probabilities [2, 3, 4, 5]. We focus in this paper on the subject of parameter elicitation, assuming that the structure of the network has already been ascertained.

Best practice in terms of parameter elicitation is based on face-to-face interviews of the expert by a trained analyst (or knowledge engineer) [6, 7, 8]. However, situations arise where such an approach is not feasible, mostly because of time and budget constraints. This is especially salient in projects with a distributed team of experts, which as Bayesian modeling gains popularity are more likely to arise than in the past [9].

Consider the following real-world example. We undertook a project focused on understanding variability in the performance of a specific human resource process and elected to use a Bayesian network as our modeling framework. The domain experts were regular employees acting as experts, they were scattered across the world and spanned different domains of expertise. We did not have the possibility of undertaking face-to-face sessions and opted for replacing them with phone interviews. The structural definition of the model, identifying the variables and inter-dependence, did not yield many difficulties nor complains from the experts. By contrast, the quantitative phase proved time consuming and generated significant frustration on both sides (analysts and experts). In particular, our efforts were hampered by (i) the time difference leading to early or late at night sessions for either the expert or the analyst and (ii) the time pressure on the experts because of the analyst waiting on the phone for them to provide an answer. The main challenge however was to have experts understand the format of the conditional probability table (CPTs). Overall, we concluded that phone elicitation was not an adequate support for remote parameter elicitation and that eliciting probabilities directly in the CPT created unnecessary cognitive burden.

The risk elicitation tool that we present here aims at addressing those concerns. We opted for a web-based tool, whose asynchronous feature enables more comfortable time management of the elicitation process from experts side (albeit less control for the analyst). An advantage of the web-based set up is the ability for the analyst to centrally manage the elicitation surveys. While we recognize that web-based approaches are second-best to face-to-face elicitation, we feel that such a tool would enable wider adoption of Bayesian models in settings where face-to-face elicitation is unlikely.

The remainder of the paper is organized as follows. In Section 2, we review the literature related to probability elicitation in Bayesian networks. Section 3 provides an in-depth description of the Risk Elicitation tool. Finally Section 4 discusses related research endeavors.

2 Expert Elicitation in Belief Networks

The process of eliciting probabilities from experts is known to be affected by numerous cognitive biases, such as overconfidence and anchoring effects [10]. When eliciting probabilities in the context of a belief network, additional practical challenges must be considered [11].

One particular problem lies with the number of parameters that have to be elicited from the experts, which leads to long and tiring elicitation sessions and sometimes inconsistent and approximate answers. To alleviate such problems, the analyst often resorts to making assumptions about the conditional relationships that reduce the number of parameters to be elicited by parameterizing the network structures using NOISY-OR and NOISY-MAX models (see for instance [2, 12]). This is in fact an option that we will provide in the next version of our tool.

As we mentioned in the introduction, another challenge associated with elicitation in Bayesian networks is the problem for the expert to understand the structure of a conditional probability table. While considering scenarios is fairly intuitive, understanding which entry corresponds to which scenario can be unnecessarily confusing. Efforts have thus been made to improve the probability entry interface in probability elicitation tools [13, 14]. Our tool integrates findings from this stream of research, by asking simple text questions corresponding to each cell of the CPT and by grouping all assessments corresponding to the same scenario together (although our support does not enable us to show them all at once but simply sequentially). Indeed, previous research has shown that presenting all conditioning cases for a node together during elicitation reduces the effect of biases [5].

Finally, the need to provide precise numerical answers is considered an additional cognitive obstacle for experts. One solution to address this problem is to present the elicitation scale with verbal and numerical anchors [15, 5, 16]. We included such findings into the design of our tool, enabling analyst to ask questions in a qualitative manner. Another solution is to elicit qualitative knowledge from experts, for instance by asking them to provide a partial order of the probabilities and leveraging limited data whenever available [17].

3 Description of the Tool

3.1 Overview

The Risk Elicitation tool is a web-based application that offers both (i) an interactive web interface through which parameter elicitation surveys themselves can be answered and automatically collected, and (ii) support for survey management. The tool can be freely accessed from the Internet; any web browser with Adobe's Flash Player 10 [18] installed will be able to run it. Given its web availability, the Risk Elicitation tool is virtually always available. Moreover, interviewees can complete a survey with little external help, pause and resume the survey at a later time, thus further relaxing the need to coordinate interviewers and interviewees.

We distinguish two classes of users of the Risk Elicitation tool: *analysts* and *domain experts*. In the following sections we describe the main use cases of the tool setting up elicitation surveys (Analyst), answering a survey (Expert) and collecting and aggregating results (Analyst). We also provide at the end of this section a description of the architectural set up along with a short discussion of the technical challenges that we met.

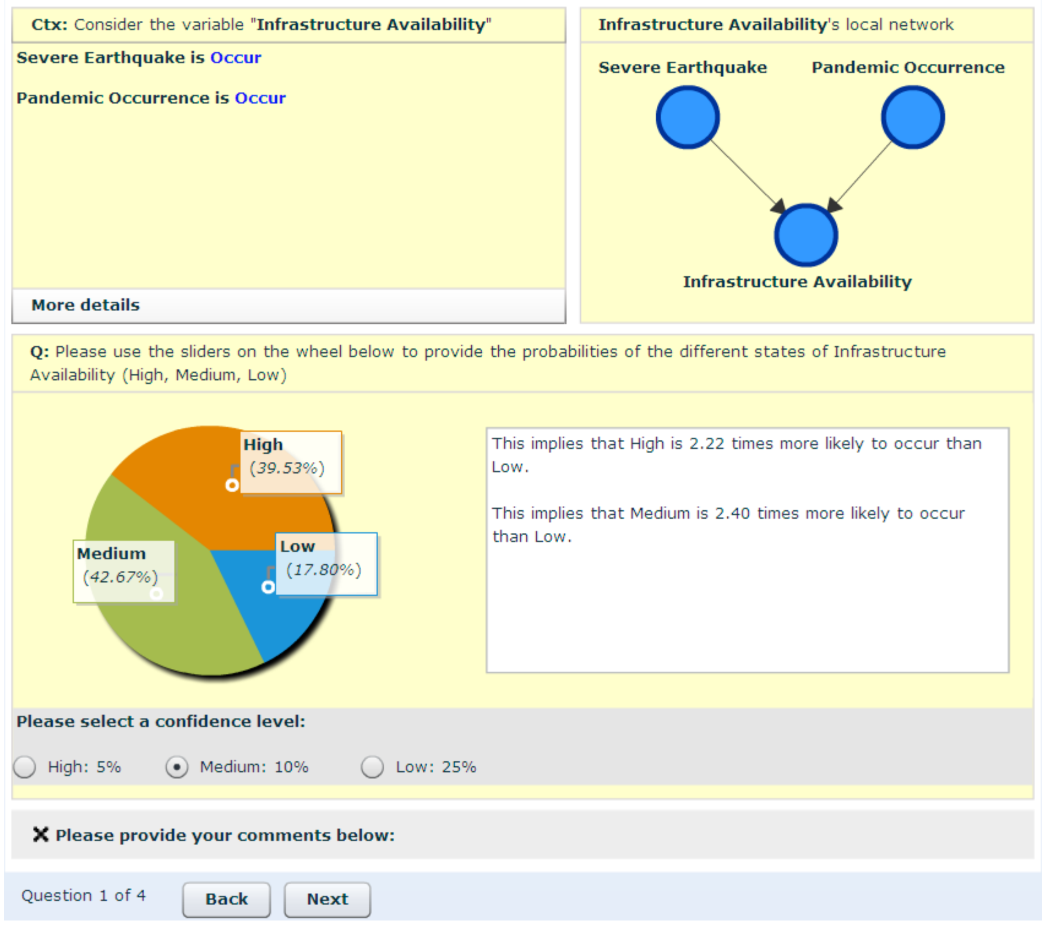
3.2 Setting up Elicitation Surveys

As mentioned earlier, we assume that the starting point of the process is a Bayesian network whose structure is fully defined, including clear description of nodes and associated states. The first step for the analyst is therefore to load his Bayesian Network file on the Risk Elicitation tool. The tool will automatically generate a sample survey, which the analyst can further customize. The second step for the analyst is to create a user account for each expert. Experts, having various domains of expertise, may not be qualified to provide information for all the nodes in the Bayesian network. To address that situation, the analyst can define roles and associate a subset of the nodes to each role. Each expert can then be associated to one or several roles and will only be asked questions on the Bayesian nodes pertaining his/her role(s)[1].

The main features of the tool that enable survey setup are:

BBN Import The Risk Elicitation tool enables analysts to create a personalized survey of the BBN

[1] In the remainder of this paper we will refer to expert and analyst as *he*.


Ctx: Consider the variable "**Infrastructure Availability**"

Severe Earthquake is Occur

Pandemic Occurrence is Occur

More details

Infrastructure Availability's local network

Q: Please use the sliders on the wheel below to provide the probabilities of the different states of Infrastructure Availability (High, Medium, Low)

This implies that High is 2.22 times more likely to occur than Low.

This implies that Medium is 2.40 times more likely to occur than Low.

Please select a confidence level:

High: 5% [✓ Medium: 10%] Low: 25%

Please provide your comments below:

Question 1 of 4 Back Next

Figure 1: Expert Elicitation page for a quantitative question.

they want to elicit. The BBN can be submitted from the Risk Elicitation tool to a server that automatically generates a survey template. The template can then be customized by the analyst, who can perform the following modifications:

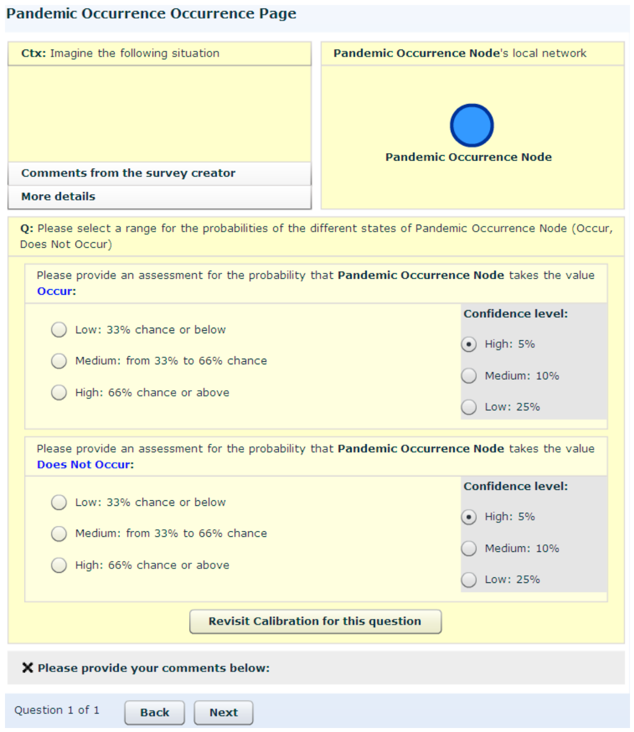
- Provide descriptive details on the Bayesian networks, its nodes and its states; add analyst notes to specific questions,
- Choose how to elicit node, whether quantitatively or qualitatively,
- Define, for the qualitative questions, the possible answers and relative numerical ranges (which we call *calibrations*),
- Customize the question texts,
- Choose whether to ask experts about their confidence level,
- Assign an order to the elicitation process (to control in which order nodes are elicited).

At the moment we only support the GeNIe file format, but our tool can be easily extended to other formats. We have developed our own format for Bayesian networks, to which the GeNIe file format is translated during the template generation.

User Management In the Administration section, analysts can register experts to the Risk Elicitation tool and assign them roles. The analyst is presented with a classic user management console, where he can add, delete and update both user accounts and the roles they play in an elicitation survey. Whenever a user account is created, the tool generates an automatic email, that the analyst can further customize and send to the expert, presenting him his credentials to access the tool and the survey he has been assigned.

3.3 Expert Elicitation

After an expert has been notified of his account credentials, he can access the Risk Elicitation tool. Upon logging in, he can select one of the surveys and roles he has been assigned to. At that point, he is offered the option of reviewing a short tutorial of the tool. Moving to the survey answering, he is presented with questions


Pandemic Occurrence Occurrence Page

Ctx: Imagine the following situation

Pandemic Occurrence Node's local network

Comments from the survey creator

More details

Q: Please select a range for the probabilities of the different states of Pandemic Occurrence Node (Occur, Does Not Occur)

Please provide an assessment for the probability that **Pandemic Occurrence Node** takes the value **Occur:**

- Low: 33% chance or below
- Medium: from 33% to 66% chance
- High: 66% chance or above

Confidence level:

- [✓ High: 5%]
- Medium: 10%
- Low: 25%

Please provide an assessment for the probability that **Pandemic Occurrence Node** takes the value **Does Not Occur:**

- Low: 33% chance or below
- Medium: from 33% to 66% chance
- High: 66% chance or above

Confidence level:

- [✓ High: 5%]
- Medium: 10%
- Low: 25%

Revisit Calibration for this question

Please provide your comments below:

Question 1 of 1 Back Next

Figure 2: An example of qualitative question.

for each relevant node of the Bayesian network. They can ask for either quantitative or qualitative answers. When the survey is complete, the expert can submit the survey on the Risk Elicitation tool and exit.

Expert elicitation is supported by the following features:

Quantitative and Qualitative Elicitation Probabilities can be elicited through either *quantitative* or *qualitative* questions. Quantitative questions ask experts to state exact probabilities, using a pie chart for discrete nodes. As shown in Figure 1, each slice of the pie represents a state of the Bayesian node with its associated probability. Users can drag the pie chart edges to provide their estimates of the node being currently evaluated, given that the scenarios defined in the context pane (parent nodes and states), at the top left corner of the question page. We also provide direct feedback about the implied odd ratios on the right side of the pie chart, as some situations may be more suited to thinking about relative chances. The map in the top right corner shows the local network topology for the node being elicited. The full Bayesian Network is also available in the *Road Map* tab on the left-hand side.

Qualitative questions do not elicit exact probabilities but ranges of probabilities. As shown in Figure 2, experts are offered a set of labeled ranges, called *calibrations*, and can select the calibration that best describes the probability of a node being in a state, given the conditions expressed in the Context pane. Calibrations are initially defined by analysts at BBN Import time, both in terms of labels and numerical range. However, experts have the ability to modify the numerical values of ranges from the tool itself if they feel they are not appropriate for the specific question.

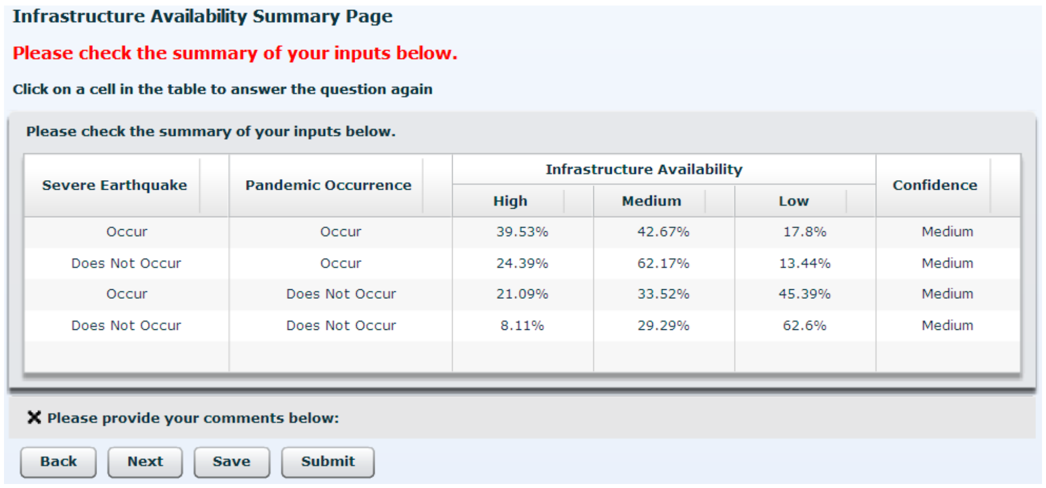
Summary Tables There are as many questions for each node as parent state configurations. After all questions for a node have been answered, the expert is shown a summary table that provides a report of all the answers they have given (see Figure 3). This is in fact the conditional probability built from the answers provided. However, at this point the expert has been actively involved in building it from the ground up and should not be as confused by the structure as if we had presented it upfront. The summary table enables to compare answers across scenarios. If the expert wants to change any of the input, he can navigate back to the associated question by clicking on the related cell in the summary table, as shown in Figure 3. When the expert is satisfied with his answers, he can save and proceed to either answer questions about another node, or submit the survey if all nodes have been answered.

Confidence For each question/node, the expert can provide an indication of his confidence in his answer (provided the analyst has enabled this feature). At this point, confidence indication is qualitative (Low/Medium/High) but could be further defined in terms of notional sample space for instance. Confidence information can be used during aggregation, to modify the weight an answer has, or to provide a threshold to filter out answers (e.g. consider only high confidence answers).

Comment For each question, the expert has the opportunity to provide a comment through an apposite collapsible text area, placed below the question itself. One use of the comment section is to provide details about understanding of a node description or state or to specify an implicit assumption that the expert has made when providing answers.

3.4 Gathering information

After setting up surveys and notifying experts, the analyst can use the Status section of the tool to check


Infrastructure Availability Summary Page

Please check the summary of your inputs below.

Click on a cell in the table to answer the question again

Please check the summary of your inputs below.

Severe Earthquake	Pandemic Occurrence	Infrastructure Availability			Confidence
		High	Medium	Low	
Occur	Occur	39.53%	42.67%	17.8%	Medium
Does Not Occur	Occur	24.39%	62.17%	13.44%	Medium
Occur	Does Not Occur	21.09%	33.52%	45.39%	Medium
Does Not Occur	Does Not Occur	8.11%	29.29%	62.6%	Medium

✕ Please provide your comments below:

Figure 3: Summary page for the questions elicited in Figure 1.

on the progress of the elicitation process. He is provided with a summary of how many surveys have been completed. From the same section, experts can be reminded to complete their survey by an automatically generated email. Once enough surveys have been completed, the analyst has the option to aggregate expert answers and export a file of the Bayesian network populated with the aggregated values.

The main mechanisms to enable gathering and aggregation of answers are:

Surveys Monitoring Analysts can monitor the advancement of survey completion from a dedicated section, called Status Tab. The Status Tab reports which experts have completed their surveys and when, which surveys have not been submitted yet and which experts have been reminded to finish the survey. To remind an expert to complete his survey, an automatic mailing system is provided to automatically generate and send email reminders to the interested parties. Generated email kindly remind experts of which surveys they have been assigned, the role they play into it, their account details in case they forgot and a link to the tool. Analysts can also customize the generated email before sending it from the tool itself, as shown in Figure 4.

Probability Aggregation After all surveys have been completed, an analyst may need to aggregate the answers provided by the experts. We currently support two methods of aggregation: *linear opinion pool* and *logarithmic opinion pool* [19]. The analyst can control the aggregation method by assigning a weight to each expert, to credit some experts more importance. The tool goes through all completed surveys, collects the probabilities elicited by experts and aggregates them using the method and weights specified by the analyst. Given that qualitative questions do not provide an exact number but a range, we take the midpoint of each range as the representative of the range (while acknowledging that this is a rather simple approach which we will refine in later versions of the tool).

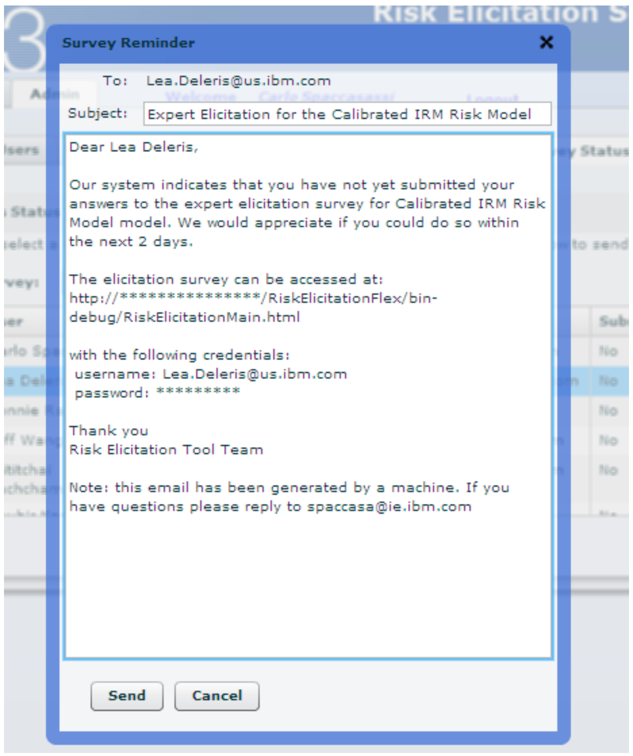
Aggregated values are used to populate the original Bayesian Network file imported in the tool. The analyst can then export the aggregated BBN on his computer.

3.5 Implementation Details

The tool employs a classic two-tiers architecture, with a web application developed on top of IBM's Websphere Application Server 6.1 [20] and a Flash client built with Adobe's Flex Builder 3 [18]. We have employed a Model-Driven Architecture approach [21] to develop the tool, following the standard Model-View-Control pattern, where the view is the Flash client, most of the controls are in the web server and the model is the survey itself, exchanged and modify by both server and clients. Communication is handled by web services using JAX-RPC [22].

Surveys data has been modeled using the Eclipse Modeling Framework (EMF) [23]. We first designed an abstract, graphical representation of the data that the survey needed to capture in EMF. The resulting representation, or model, is similar to a UML Class diagram. Code to manipulate and also persist the model is automatically generated from the model and taken care of by EMF.

EMF does not support natively Actionscript, Adobe's programming language: EMF's standard tools cannot



To: Lea.Deleris@us.ibm.com

Subject: Expert Elicitation for the Calibrated IRM Risk Model

Dear Lea Deleris,

Our system indicates that you have not yet submitted your answers to the expert elicitation survey for Calibrated IRM Risk Model model. We would appreciate if you could do so within the next 2 days.

The elicitation survey can be accessed at:
http://***************/RiskElicitationFlex/bin-debug/RiskElicitationMain.html

with the following credentials:
username: Lea.Deleris@us.ibm.com
password: *********

Thank you
Risk Elicitation Tool Team

Note: this email has been generated by a machine. If you have questions please reply to spaccasa@ie.ibm.com

Figure 4: Customizable email templates from the administration console.

generate model manipulation code automatically for it. To address this problem, we bridged EMF to a Web Service definition file (WSDL). We first exported EMF's models to an XML Schema, which we imported into the WSDL file. Adobe's Flex can then generate code from the WSDL file both to communicate with the server and to access the model.

Communication points between server and client are also generated from the WSDL file. Extensions and modifications to either the model or the communication points, on the server and client side, were handled automatically by either EMF or Flex, handling manual error-prone tasks and saving development time.

Finally, we import and export BBN files written in the SMILE/GeNIe format [24]. EMF automatically manages GeNIe file loading and saving, using the XML Schema definition which is publicly available. The GeNIe files are then converted to an internal EMF model designed to ease BBN manipulation.

4 Related Research

In this section, we briefly discuss some of the research questions that have arisen from the development of the risk elicitation tool. In particular, we have focused on the effect on the elicitation process of the order in which the nodes are presented. Because of the web-based feature of our tool, we have more freedom in determining the order than traditional face-to-face approaches.

4.1 Experiencing Different Orders

We considered the following question: *Does the parameter elicitation ordering in belief networks even matter to a domain expert?* To answer this question, we explored the relationship between node ordering and user-friendliness of the elicitation process in an experimental setting. Specifically, three different node orderings for the same belief network were considered: two 'top-down' and one 'bottom-up' ordering, with parameter elicitation performed using the risk elicitation tool described in this paper. Around seventy Stanford University graduate students were asked to elicit a belief network with six nodes on the subject of getting a job immediately after their studies; they were split into approximately three equal groups, one group for each order. The top-down orders presented questions to elicit parameters of parent nodes before children nodes, while the bottom-up order visited children before parents.

In this particular experiment, there was no drop-out - all subjects completed the elicitation process, perhaps due to the small size of the network and the incentive of extra class credit (which was only granted for complete assessments). Along with the Web-based elicitation survey, the students also responded to a short survey requesting feedback about the elicitation process and the corresponding tool. The results did indicate that the order in which the nodes are presented affects not only how comfortable experts claim to be with the process, but also the time required to elicit the parameters. In particular, there was a significant difference between the orders with regard to user-friendliness based on the survey responses. For the two top-down orders, hardly any of the subjects felt that the order was confusing, compared to 23% for the bottom-up order. Moreover, the average time to complete the elicitation was lower for the two top-down orders as compared to the bottom-up order. The two top-down orders differed in survey completion time: an average of 400 seconds with a standard deviation of 170 seconds for the first one, against an average of 500 seconds with a standard deviation of 400 seconds for the second one.

4.2 Ordering Mathematically

In a separate study, we explored the problem of determining, for a particular belief network whose structure is known, the optimal order in which the parameters of the network should be elicited. Our objective in determining the order is to maximize information acquisition. While the order of the elicitation process is irrelevant if all nodes are elicited and if experts are able to provide their true beliefs, we believe that new trends in belief network modeling make these assumptions questionable. When only a subset of the nodes may be elicited or when answers can be noisy, it is necessary to devise an ordering strategy that seeks to salvage as much information as possible.

We therefore developed an analytical method for determining the optimal order for eliciting these probabilities, where optimality is defined as shortest distance to the true distribution (on which we have a prior). We considered the case where experts may drop out of the elicitation process and modeled the problem through a myopic approach.

For the case of uniform Dirichlet priors, we show that the 'bottom up' elicitation heuristic can be optimal. For other priors, we showed that the optimal order often depends on which variables are of primary interest to the analyst (whether all the nodes in the network or a subset, as is often the case in risk analytic applications).

The orderings resulting from the methods proposed in that model are driven solely by analytical concerns, and do not consider the user-friendliness of the elicitation process. In practice, as we discussed in the previous section, different orderings can impact the perceived difficulty of the process, thereby making the elicitation of complete and accurate beliefs more difficult. These results further motivate the need to investigate the consequence of forcing a possibly unnatural ordering upon experts and to assess whether the 'information gain' from an analytical perspective is worth the 'cost' in practice, i.e. in terms of the amount of confusion, fatigue and increased imprecision. More generally, empirical research to investigate how experts actually react to different orders is an important topic, similar to the empirical work on understanding how experts actually feel about different probability elicitation tools [15, 14]. The tool presented in this paper could be a useful support for such endeavors.

4.3 Comparison with existing web-based tools

Finally, we compare our tool to two existing web-based tools for risk elicitation pointed out by reviewers: BayesiaLab [25] and the Elicitation Tool from ACERA, the Australian Center of Excellence of Risk Analysis, described in [26].

BayesiaLab, a commercial product developed by Bayesia, provides an integrated environment for working with Bayesian networks. It supports many features, such as BBN modeling, BBN learning from data, and elicitation. With respect to elicitation, analysts can create a profile for each expert, select the portions of a variables' CPT to be elicited, and send this information to a web server over the Internet. The web server generates surveys to elicit probabilities quantitatively, using a slider bar to capture expert input. Experts can also provide a level of confidence in an answer, expressed as a percentage, along with additional comments. In comparison with our tool, many features are similar: both tools provide expert profile management, on-line surveys, and survey import and export. Our tool, however, allows for both quantitative and qualitative elicitation of probabilities. The elicitation formats are different as well: our tool uses pie charts to capture quantitative probabilistic information for discrete random variables and a slider bar (expressed as a percentage difference from baseline) to capture impact of a factor on a (continuous-valued) metric under a specified scenario. Additionally, our tool allows analysts to fully customize surveys and aggregate by one of several algorithms. We also provide experts with additional contextual information, including a local and global map of the Bayesian network, a tutorial, the description of each node and state in the Bayesian network, along with analysts' comments.

The Elicitation Tool from ACERA is quite different from both our tool and BayesiaLab, in that it is an online questionnaire to directly elicit estimates of risks. Questions are open-ended. An example is: "Will DAGGRE win?". When answering a question, users need to provide four numerical estimates in an HTML form: the lowest estimate, the highest estimate, the best estimate and a confidence level. A graphical representation of the estimates is displayed and the user can submit the answers. After submission, the tool displays a selection of answers from users who have already completed the survey. The user is given the chance to review his own answers in light of this new input and submit again. In contrast, our tool allows review of only the expert's own answers, as shown in Summary Pages, and are tailored for Bayesian networks where the goal is to elicit (conditional) probabilistic information. To help experts frame the context of the question, we provide additional information, such as the Bayesian network's local map and network description. ACERA's tool does not seem to be tied to Bayesian networks, so less contextual information is required in this case.

5 Conclusion

In this paper, we describe a web-based expert elicitation tool for Bayesian network models that is especially relevant for the management of distributed teams of experts. We focus especially on facilitating the understanding of a conditional probability table by asking each entry separately and in a textual format. The tool enables the management of the survey administration cycle, from the customization of the survey and the creation of roles (associated with a subset of the network) to the monitoring of progress from experts and aggregation of results.

While we have implemented several best-practices from the elicitation literature, we also have identified various directions for further development. One simple extension will consist in allowing for NOISY-OR and NOISY-MAX parameterization. Going further, we would like to more strongly encourage for qualitative elicitation, asking for orders of magnitudes for instance, or if limited date was available following the relative order procedure suggested by [17]. In fact, for cases where partial data is available, one could also consider providing feedback to the expert directly during the elicitation session [27]. Finally, we have started providing support for utility/value nodes but so far in a coarse manner. Initially, experts are asked to identify a parent states configuration for which they are comfortable with providing an exact estimate of utility. We call this configuration *base case*. For non-base case configurations, experts only need to specify how much in percentage the utility of the node differs from the base case.

Acknowledgements

This work is partially supported by fundings from IDA Ireland (Industrial Development Agency).

References

[1] D. Heckerman. A tutorial on learning with bayesian networks. In M. Jordan, editor, *Learning in Graphical Models*. Kluwer, Netherlands, 1998.

[2] M. Henrion. Some practical issues in constructing belief networks. In L. Kanal M. Henrion, R. Shachter and J. Lemmer, editors, *Proceedings of the Fifth Conference on Uncertainty in Artificial Intelligence*, pages 161–173. Elsevier Science, New York, NY, 1989.

[3] M. Druzdzel and L. van der Gaag. Elicitation of probabilities for belief networks: Combining qualitative and quantitative information. In P. Besnard and S. Hanks, editors, *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, pages 141–148. Morgan Kauffman, San Francisco, CA, 1995.

[4] M. Druzdzel and L. van der Gaag. Building probabilistic networks: Where do the numbers come from? *IEEE Transactions on Knowledge and Data Engineering*, 12(4):481–486, 2000.

[5] L. van der Gaag, S. Renooij, C. Witteman, B. Aleman, and B. Taal. How to elicit many probabilities. In K. Laskey and H. Prade, editors, *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, pages 647–665. Morgan Kauffman, San Francisco, CA, 1999.

[6] C. Spetzler and C. von Holstein. Probability encoding in decision analysis. *Management Science*, 22(3):340–358, 1975.

[7] M. Merkhofer. Quantifying judgmental uncertainty: Methodology, experiences and insights. *IEEE Transactions on Systems, Man and Cybernetics*, 17(5):741–752, 1987.

[8] R.L. Keeney and D. von Winterfeldt. Eliciting probabilities from experts in complex technical problems. *Engineering Management, IEEE Transactions on*, 38(3):191 –201, aug 1991.

[9] Sandra Hoffmann, Paul Fishbeck, Alan Krupnick, and Michael McWilliams. Elicitation from large, heterogeneous expert panels: Using multiple uncertainty measures to characterize information quality for decision analysis. *Decision Analysis*, 4(2):91–109, 2007.

[10] D. Kahneman, P. Slovic, and A. Tversky. *Judgment under Uncertainty: Heuristics and Biases.* Cambridge University Press, 1982.

[11] S. Renooij. Probability elicitation for belief networks: Issues to consider. *The Knowledge Engineering Review*, 16(3):255–269, 2001.

[12] Adam Zagorecki and Marek Druzdzel. Knowledge engineering for bayesian networks: How common are noisy-max distributions in practice? In *Proceeding of the 2006 conference on ECAI 2006: 17th European Conference on Artificial Intelligence August 29 – September 1, 2006, Riva del Garda, Italy*, pages 482–486, Amsterdam, The Netherlands, The Netherlands, 2006. IOS Press.

[13] Hope Nicholson Korb, L. R. Hope, A. E. Nicholson, and K. B. Korb. Knowledge engineering tools for probability elicitation. Technical report, 2002.

[14] H. Wang and M. Druzdzel. User interface tools for navigation in conditional probability tables and elicitation of probabilities in bayesian networks. In C. Boutilier and M. Goldszmidt, editors, *Proceedings of the Sixteenth Conference on*

Uncertainty in Artificial Intelligence, pages 617–625. Morgan Kaufmann, San Francisco, CA, 2000.

[15] S. Renooij and C. Witteman. Talking probabilities: Communicating probabilistic information with words and numbers. *International Journal of Approximate Reasoning*, 22:169–194, 1999.

[16] F. Fooladvandi, C. Brax, P. Gustavsson, and M. Fredin. Signature-based activity detection based on bayesian networks acquired from expert knowledge. In *Information Fusion, 2009. FUSION '09. 12th International Conference on*, pages 436 –443, july 2009.

[17] E. M. Helsper, L. C. van der Gaag, A. J. Feelders, W. L. A. Loeffen, P. L. Geenen, and A. R. W. Elbers. Bringing order into bayesian-network construction. In *Proceedings of the 3rd international conference on Knowledge capture*, K-CAP '05, pages 121–128, New York, NY, USA, 2005. ACM.

[18] Jeff Tapper, Michael Labriola, Matthew Boles, and James Talbot. *Adobe Flex 3: training from the source*. Adobe Press, first edition, 2008.

[19] Robert T. Clemen and Robert L. Winkler. Combining probability distributions from experts in risk analysis. *Risk Analysis*, 19:187–203, 1999.

[20] E.N. Herness, R.H. High, and J.R. McGee. Websphere application server: a foundation for on demand computing. *IBM Syst. J.*, 43(2):213–237, April 2004.

[21] Richard Soley and the OMG Staff Strategy Group. Model-driven architecture. `http://www.omg.org/~soley/mda.html`, 2000.

[22] Roberto Chinnici. Java APIs for XML based RPC (JSR 101). `http://jcp.org/aboutJava/communityprocess/first/jsr101/`, October 28, 2003.

[23] Frank Budinsky, Stephen A. Brodsky, and Ed Merks. *Eclipse Modeling Framework*. Pearson Education, 2003.

[24] Marek J. Druzdzel. Smile: Structural modeling, inference, and learning engine and genie: a development environment for graphical decision-theoretic models. In *Proceedings of the sixteenth national conference on Artificial intelligence and the eleventh Innovative applications of artificial intelligence conference innovative applications of artificial intelligence*, AAAI '99/IAAI '99, pages 902–903, Menlo Park, CA, USA, 1999. American Association for Artificial Intelligence.

[25] Bayesia. Bayesialab. `http://www.bayesia.com/en/products/bayesialab.php`.

[26] Andrew Speirs-Bridge, Fiona Fidler, Marissa McBride, Louisa Flander, Geoff Cumming, and Mark Burgman. Reducing overconfidence in the interval judgments of experts. *Risk Analysis*, 30(3):512–23, 2010.

[27] A. H. Lau and T. Y. Leong. Probes: a framework for probability elicitation from experts. In *Proceedings of the AMIA Symposium*. American Medical Informatics Association, 1999.

Author Index