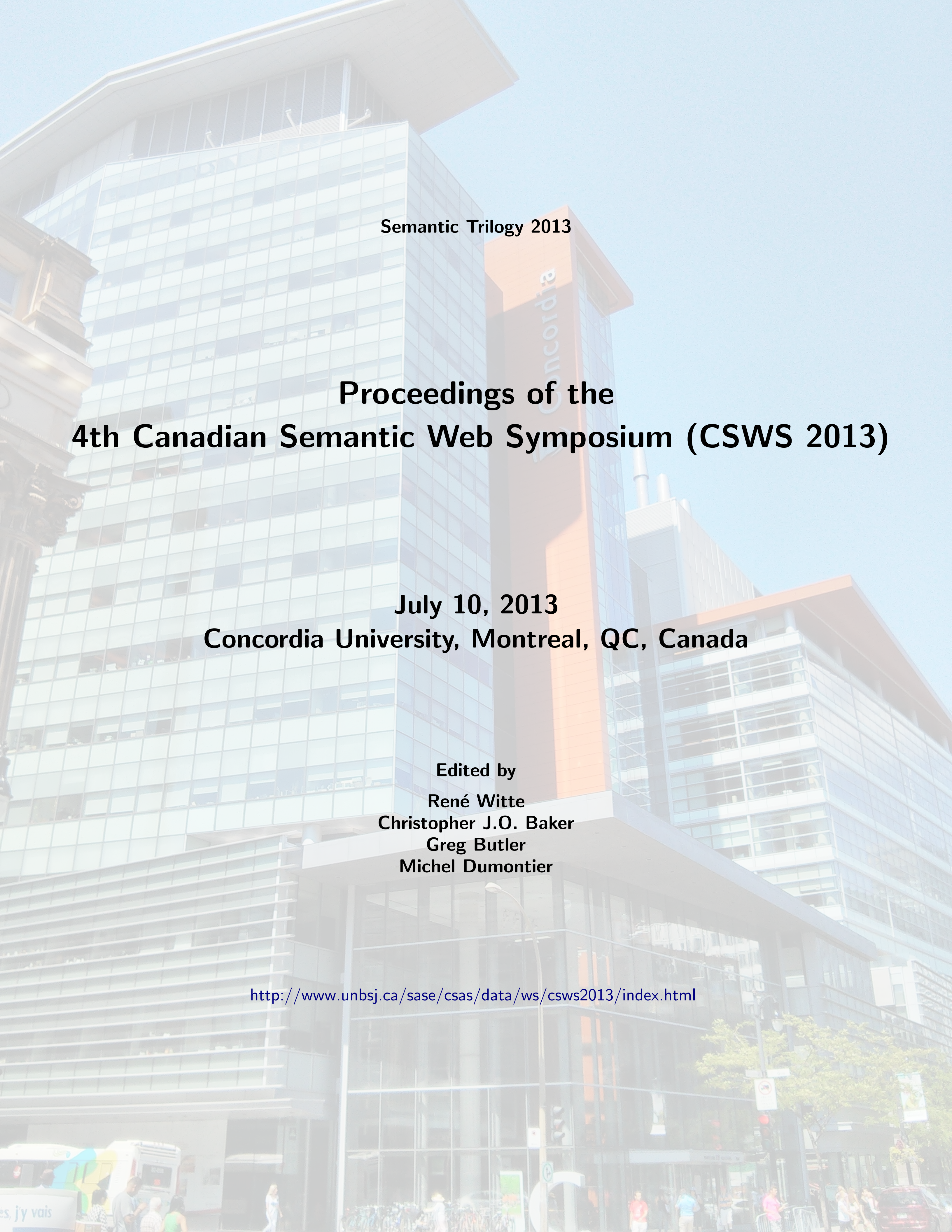


Semantic Trilogy 2013

Proceedings of the 4th Canadian Semantic Web Symposium (CSWS 2013)

July 10, 2013
Concordia University, Montreal, QC, Canada

Edited by

René Witte
Christopher J.O. Baker
Greg Butler
Michel Dumontier

http://www.unbsj.ca/sase/csas/data/ws/csws2013/index.html

Contents

Fourth Canadian Semantic Web Symposium (CSWS 2013)

July 10, 2013, Concordia University, Montréal, Canada

Program

08:45-9:15 Registration, Welcome

09:15-10:15 Keynote Speaker: Birgitta König-Ries, University of Jena, Germany
Why Biodiversity Science needs Semantics

10:15-10:45 Coffee Break

10:45-12:15 Long and Short Research Papers
Borna Jafarpour and Syed Sibte Raza Abidi
A Semantic Web Based Ontology Mapping and Instance Transformation Framework

Mahsa Kiani, Virendrakumar C. Bhavsar and Harold Boley
Combined Structure-Weight Graph Similarity and its Application in E-Health

Jocelyne Faddoul and Wendy MacCaull
Parallelizing Algebraic Reasoning for the Description Logic SHOQ

Yevgen Biletskiy
A Framework for Web-based Interoperation among Business Rules

12:15-14:00 Lunch break

14:00-14:40 Invited Speaker: Volker Haarslev, Concordia University, Canada
Speed-forming your ontology or how to improve reasoning performance for your OWL ontology

14:40-15:20 Early Career Track Papers (I)
John Cuzzola, Dragan Gasevic and Ebrahim Bagheri
Product Centric Web Page Segmentation and Localization

Mohammad Sadnan Al Manir, Alexandre Riazanov, Harold Boley and Christopher J.O. Baker
Generating Semantic Web Services from Declarative Descriptions

15:20-16:20 System Paper Demos with Coffee Break
Felicitas Löffler, Bahar Sateli, Birgitta König-Ries and René Witte
Semantic Content Processing in Web Portals

John Cuzzola, Zoran Jeremic, Ebrahim Bagheri, Dragan Gasevic, Jelena Jovanovic and Reza Bashash
Semantic Tagging with Linked Open Data

Lingkai Zhu, Kevin Quach and Helen Chen
A Semantic Framework for Data Quality Assurance in Medical Research

Ismail Akbari, Bo Yan, Junyan Zhang and Harold Boley
Visualizing SWRL Rules: From Unary/Binary Datalog and PSOA RuleML to Graphviz and Grailog

16:20-17:20 Early Career Track Papers (II)
Laleh Kazemzadeh, Helena F. Deus, Michel Dumontier and Frank Barry
Looking into Reactome through Biopax Lens

Jerry George, Fatna Belqasmi, Roch Glitho and Nadjia Kara
A Substrate Description Framework and Semantic Repository for Publication and Discovery in Cloud Based Conferencing

Altaf Hussain and Wendy MacCaull
Context aware service discovery and service enabled workflow

17:20-17:30 Symposium Closing

18:00-21:00 Symposium Reception: McKibbin's Irish Pub (2nd floor)
1426 Bishop St., www.mckibbinsirishpub.com

Committees

Symposium Organizers

Greg Butler (Concordia University, Montreal, Canada)
Christopher J.O. Baker (University of New Brunswick, Canada)
Michel Dumontier (Carleton University, Ottawa, Canada)

Program Committee Chair

René Witte (Concordia University, Montreal, Canada)

Programme Committee

Abdolreza Abhari (Ryerson University, Canada)
Alexandre Riazanov (IPSNP Computing Inc, Canada)
Arash Shaban-Nejad (McGill University, Canada)
Artjom Klein (University of New Brunswick)
Babak Esfandiari (Carleton University, Canada)
Bruce Spencer (National Research Council Canada, Canada)
Christopher Baker (University of New Brunswick, Canada)
Daniel Lemire (LICEF Research center, TELUQ, Montreal, Canada)
Ebrahim Bagheri (Ryerson University, Canada)
Faezeh Ensan (Athabasca, Canada)
Fred Popowich (Simon Fraser University, Canada)
Greg Butler (Concordia University, Montreal, Canada)
Hassan Ait-Kaci (Universite Claude Bernard Lyon 1, France)
Helen Chen (University of Waterloo, Canada)
Juergen Rilling (Concordia University, Canada)
Marek Reformat (University of Alberta, Canada)
Marina Sokolova (University of Ottawa, Canada)
Michel Dumontier (Carleton University, Canada)
Raza Abidi (Dalhousie University, Canada)
Vio Onut (IBM Canada, Canada)
Weichang Du (University of New Brunswick, Canada)
Weiming Shen (National Research Council, Canada)
Wendy MacCaull (St. Francis Xavier University, Canada)
Yevgen Biletskiy (University of New Brunswick, Canada)

Additional Reviewers

Bahar Sateli (Concordia University, Montreal, Canada)
Samir Amir (LIRIS Research Center, Lyon, France)
Neil Swainston (University of Manchester, UK)

Part I.

Invited Talks

Why Biodiversity Science needs Semantics

Birgitta König-Ries
Institute for Computer Science
Friedrich-Schiller University of Jena
Germany

***Abstract*—Biodiversity science investigates biological diversity on all levels and scales. There is strong evidence that biodiversity is declining at unprecedented rates and that this will have dire consequences for humankind. Consequently, there is a strong need for science to support political decision making in this area. Thus key questions of biodiversity science are: What is out there? Why is it there? Does it matter (to us)? How can we save it?**

In this talk, we argue that computer science in general and semantics in particular are essential to answering these questions and that biodiversity science needs more involvement by computer scientists. We will take a look at a number of projects on biodiversity and will investigate the role of computer science in them and how semantics can help.

Speed-forming your ontology or how to improve reasoning performance for your OWL ontology

Volker Haarslev
Department of Computer Science and Software Engineering
Concordia University
Montreal, Canada

***Abstract*—In this talk we describe the OWL reasoning landscape and its obstacles. We overview the three tractable OWL fragments and discuss dramatic speed improvements achieved by corresponding specialized reasoners. Furthermore, various combinations of OWL constructors and their impact on practical reasoning performance are analyzed. In the last part we give a brief overview of promising approaches to speed up reasoning for OWL ontologies that are outside of the three tractable OWL fragments.**

Part II.

Long Research Papers

A Semantic Web Based Ontology Mapping and Instance Transformation Framework

Borna Jafarpour
NICHE Research group
Computer Science Department, Dalhousie University
Halifax, Canada
borna@cs.dal.ca

Syed Sibte Raza Abidi
NICHE Research group
Computer Science Department, Dalhousie University
Halifax, Canada
Sraza@gmail.com

We present a semantic-based ontology mapping framework that offers instance transformation and discovery of new mapping using reasoning. Our framework comprises an expressive OWL-Full Mapping Representation Ontology (MRO) and a mapping translation method. Ontology mappings are represented in terms of an instantiation of the MRO. We define formal semantics for our ontology mapping representation by translating the ontology mappings in OWL-Full to OWL and SWRL in order to derive new ontology mappings and perform instance transformation using reasoning. We have evaluated the workings of our ontology mapping framework by mapping three ontologies each representing a disease specific Clinical Practice Guideline (CPG) to a general CPG representation ontology. The intent of the mapping is to provide knowledge-driven decision support for the management of patients with multiple diseases.

Keywords—Ontology; Semantic Web; Ontology mapping; Instance Transformation; SWRL, OWL

I. Introduction

Complex knowledge-centric systems demand the integration of multiple knowledge objects in order to achieve a comprehensive knowledge model. Given the open nature of semantic web, several heterogeneous knowledge models exist for representing the knowledge in any domain area. For instance, in healthcare, there exist variety of knowledge models to model and computerize clinical practice guidelines (CPG)—these models share a range of concepts but differ in the interpretation and specification of these concepts. To develop a holistic knowledge model based on multiple heterogeneous knowledge models, therefore demands the establishment of standardized interoperability specifications and criterion, at both the structural and semantic levels, to achieve the integration of multiple heterogeneous knowledge models.

Lately, ontologies have emerged as expressive knowledge representation formalisms, together with methods to reason over the knowledge. An ontology typically represents a specific aspect of knowledge with varying levels of abstraction and description. To formulate a broader and holistic knowledge model, researchers aim to integrate multiple existing ontologies that demand an interoperability solution that aligns heterogeneous ontologies in keeping with the domain-specific interpretations and constraints surrounding knowledge consistency. A semantic interoperability framework aims to establish explicit and well-defined *mapping* between two ontologies. In practice, ontology mappings methods map the ontology elements between two ontologies based on the similarity of their names, their relations and their shared instances using name-based, structure-based and instance-based approaches respectively [10].

An alternative mapping approach is called semantic-based ontology mapping. This approach has two steps [10]: (i) *anchoring step* in which a number of initial mappings or anchors are created between two ontologies using name, instance or structure based ontology mapping approaches; (ii) *reasoning step* in which a reasoner performs reasoning on the mappings and the mapped ontologies to (a) transform instances between the two ontologies; and (b) improve the existing mappings by discovering new ones based on the formal semantics of the mappings and the mapped ontologies. Typically, proprietary reasoning algorithms [1][4][14], propositional logic [11][12] and Description Logic (DL) [5][6][7][8][9] are used in the reasoning step.

The quality of ontology mapping based on a semantic-based approach is contingent on the ontology mapping representation language’s level of expressivity and formal semantics—reasoning over a more expressive ontology can yield more new mappings as opposed to reasoning over a less expressive ontology. Our review of the existing mapping representation languages [1][2][3][4][9][11][12][13][15] and an existing surveys [13] reveal that most of the current ontology mapping languages suffer from lack of expressivity and formal semantics. Lack of formal semantics stops us from using the mappings in a semantic-based ontology mapping approach.

To address the lack of expressivity and formal semantics in ontology mapping languages, in this paper we use semantic web technologies to present a semantic-based ontology mapping approach that entails: (a) a general purpose OWL-Full based *Mapping Representation Ontology* (MRO) that serves as an expressive ontology mapping language that can represent complex mappings such as predefined mapping patterns, conditions, condition satisfaction criteria, variables, structural modifications and mathematical operators. An instance of the MRO represents the mappings between a source and a target ontology; and (b) *translation algorithm* to translate the instantiations of the MRO (which are in OWL-Full and hence undecidable) to OWL-DL or OWL 2 RL + SWRL which is a decidable combination. The translated mappings and the mapped ontologies are reasoned over to achieve both instance transformation and to discover new mappings. Please note that our approach is not problem-specific and can be used for mapping any two ontologies as long as they are represented in OWL.

We chose to represent mappings in OWL-Full and then translate them to OWL+SWRL instead of using OWL+SWRL directly because of the following reason: (a) The expressivity of MRO being OWL-Full—i.e. using properties and classes as instances—makes the ontology mappings more readable and less verbose—i.e. with fewer triples compared to OWL-DL; (b) It enables us to support conditional mappings and complex condition satisfaction criteria, meta modelling, Boolean operators and converting ontology elements and creating new ones which are not directly supported by either OWL or SWRL. These aspects of ontology mapping are supported by automatic generation of several OWL axioms and SWRL rules that simulate the lacking feature during the translation process; (c) SWRL rules are difficult to write and can easily become undecidable if not written correctly. In our translation algorithm, DL-Safe SWRL rules are generated automatically thus relieving the user about decidability concerns.

In order to evaluate the efficacy of our ontology mapping framework, we instantiated MRO to map three disease-specific CPG ontologies to a general CPG ontology. We then successfully transformed instantiations of the source ontologies to instantiations of the target ontology. The problem being pursued here is to handle comorbidities by integrating two or more disease-specific CPG to manage a patient with multiple simultaneous diseases.

II. Related Work

In this section, we review the existing semantic-based ontology mapping approaches and the existing mapping representation languages.

A. *Semantic-Based Ontology Mapping Approaches*

These approaches can be categorized based on the reasoning techniques that they use. Literature reports on using proprietary reasoning algorithms [1][4][14], propositional satisfiability solvers [11][12] and description logic reasoners [5][6][7][8][9].

Methodologies that use proprietary reasoning algorithms such as [1][4][14] are not desirable because of the following disadvantages: (a) Because of their proprietary algorithms, they can't benefit from the existing reasoners and a special reasoning engine should be developed in order to perform the reasoning step; (b) Since these engines can only perform reasoning on the mappings and not the ontology representation languages they cannot exploit the internal structure (knowledge) of the ontologies to draw new mappings based on them.

There are semantic-based algorithms that use propositional logic to perform reasoning. In these approaches, a theory is built by conjunction of the axioms from the mapped ontologies. This theory can be constructed by using one of the name, instance or structure based approaches. Then, a matching formula is made for each pair of classes from the mapped ontologies. Afterwards, the validity of the formula is checked by using a propositional satisfiability solver. BerkMin [11] and GRASP [12] are two examples to name. None of these approaches goes beyond finding equivalence, subclass, and complement relationship between classes. We believe that this is due to lack of expressivity in propositional logic for the task of ontology mapping.

Description logic reasoners have also been used in the reasoning step of semantic-based ontology mapping approaches. Two approaches that use description logic to find disjointness, overlap, inclusion and equivalence relations between concepts are reported in [5][7]. Meilicke and colleagues [6] used description logic to debug the mappings by detecting inconsistencies. In a theoretical work [8] it is suggested that description logic can be used for reasoning about the mapping themselves to find containment, minimality, consistency and embedding attributes in them. Therefore, DL has been used for reasoning about the mappings, debugging them and deriving simple mappings (class equivalence, etc.) but no attempt has been made to represent more complex mappings such as value transfer mappings or mathematical computations. We believe that lack of an expressive mapping representation language that formally defines the mapping semantics in DL is limiting the capabilities of DL-based semantic-based mapping methodologies.

There are also approaches such as [1] and [3] that translate the mappings to OWL and SWRL to use OWL reasoners. These methodologies transform the mapping to either OWL or SWRL but not a combination of them. However, we believe that OWL or SWRL cannot be used separately for mapping ontologies unless we need very low levels of expressivity. Therefore, we can conclude that complex mappings are not possible to be transformed using these approaches. Moreover, no explanation or details of the translation process have been provided in this regard.

B. *Ontology mapping representation languages*

In this section, we review the expressivity levels of the mapping representation languages with formal semantics. We reviewed the literature trying to define the requirements of the mapping representation languages [13][15]. The support for mathematical, Boolean, string and structural modification operators, frequently used mapping patterns, predefined set of relations between ontology elements, variables and the ability to express conditions and condition satisfaction criteria are the most important expressivity requirements identified in these publications.

Many of the existing mapping representation languages such as MAFRA [4], C-OWL [9] and many others [1][4][5][6][8][9][11][12] are only capable of expressing simple relations such as equivalent, disjoint, subclass and super class between ontology classes. A review of 13 of these languages in [13] shows that 61% of all of them are only capable of expressing the equivalence relationship. Only C-OWL has formal semantics that can be used by reasoners in the semantic-based ontology mapping. Even though authors of MARFA claim that they have formal semantics no details are provided in that regard. OWL is more expressive than these languages as it supports a wide range of predefined relations between classes, properties and instances. It also supports a large number of class and property manipulation operators that can be used towards structural modification. The rest of the desired features described earlier are not supported by OWL. Having formal semantics makes it possible to use OWL in the reasoning step of a semantic-based ontology mapping approach.

An important requirement of these languages is the ability to support variables and to express mathematical, Boolean, date, string computation and comparison and structural modification operators. SWRL is the only language that is able to express a wide list of the necessary functions for mappings that are supported by the concept of built-ins. This language however cannot support Boolean operators, mapping patterns, conditions, qualified cardinality restrictions and some of the property relations and structure modifications operators that are expressible in OWL such as union operator. SWRL also has formal semantics and can be used in semantic-based ontology mapping approaches.

Two expressive mapping languages are discussed in [2] and [3]. The language in [2] supports a wide range of mappings patterns, conditions and variables. However, this language does not support representation of complex condition satisfaction criteria, and mathematical, Boolean, string and date operators. The language in [3] supports a large number predefined set of relations between ontology elements, mapping patterns, ability to express conditions and structural modification operators. Even though some descriptions of the formal semantics of these languages are discussed, enough details for a practical implementation of a semantic-based ontology mapping approach are not provided.

III. Our Ontology Mapping Approach

Our ontology mapping approach entails the following two components: A Mapping Representation Ontology (MRO) in OWL-Full to represent the ontology mapping; and a translation algorithm that transforms an instantiation of the MRO to OWL + SWRL. Our ontology mapping approach is pursued by performing the following three steps:

1. Anchoring (MRO Instantiation): In the first step, initial inter-ontology mappings are created by establishing semantic relations between classes, properties and instances of the mapped ontologies. These mappings can be either created using existing automatic discovery algorithms such as methods based on similarity of names or by a domain expert. Due to complexity of the mappings between ontologies of our domain area, we opted to create the initial mappings manually. Therefore, a mapping between two ontologies is an instantiation of the MRO created by the domain expert. For instance, by instantiating MRO we may indicate that classes *Person* and *Human* from source and target ontologies are equivalent classes. Source and target ontologies are represented by o1: and o2: name spaces in the rest of the paper.

2. Translation to OWL-DL + SWRL: In the next step, we transform the instantiation of MRO to a combination of OWL-DL or OWL2 RL + SWRL depending on the expressivity needs of the mappings. To avoid the possible undecidability as the result of using SWRL rules, only DL-Safe rules [17] are added in the translation process. As an example, the instantiation of MRO that expresses *Human* and *Person* classes are equivalent is translated to:

```
o1:Human owl:equivalentClass o2:Person.
```

3. Reasoning: Finally, we use OWL reasoners to perform reasoning on the translated mappings and the mapped ontologies to improve the mapping by discovering new ones and to perform instance transformation. As an example, The following SWRL rule which is the result of the translation of an instantiation of MRO to SWRL, calculates the Body Mass Index (BMI) of an instance of the class *o1:Person* and assigns it to the class *o2:ObesePerson* if the value of the BMI is greater than 30 and the condition c1 is satisfied.

```
o1:Person(?InstVar), o1:hasHeight(?InstVar,
?HeightVar), o1:hasWeight(?InstVar,
?weightVar),swrlb:divide(?BMIVarVar,
?func1SWRLVar,?HeightVarVar),swrlb:divide(?fun
c1Var,?weightVar,?HeightVar),swrlb:greaterThan
(?BMIVar, 30),SatisfiedCondition(c1)->
o2:ObesePerson(?o1InstVar),
o2:hasBMI(?o1InstVar, ?BMIVar)
```

As a result of reasoning on this rule and source and target ontologies all together, the reasoner infers that an instance of the *Person* class in the source ontology with the weight of 97kg and height of 179cm belongs to the class *o2:ObesePerson* in the target ontology and has the value 32.2 for the property *o2:hasBMI*. In this way, instances of the *o1:Person* class in the source ontology are transformed to instances of the *o2:ObesePerson* class in the target ontology. We have used Pellet as our reasoner since it supports both OWL and SWRL. Any other reasoners with support for OWL and SWRL can be used for this purpose.

IV. Mapping Representation Ontology

In this section, we describe MRO, its classes, properties and instances. In order to easily identify classes, properties and instances in the text, class names are *italicized* and their first letter are Capitalized (e.g. *ClassNameExample*), property names are *italicized* (e.g. *propertyExample*) and instance names are underlined (e.g. instanceExample).

A. Mappings and Relations

In order to represent mappings between instances, properties and classes of source and target ontologies we have created the *Mapping* class. Three types of mappings have been modeled in MRO using the following classes: *RelationalMapping*, *TransformationMapping* and *ValueTransferMapping*.

RelationalMappings express a relation between two ontology elements. *hasSource* and *hasTarget* properties with the domain of *Mapping* and range of OWL:thing are used assign the source and the target elements to a mapping. The *hasRelation* Property with the domain of *RelationalMapping* and the range of *MappingRelation* defines the relation in a relational mapping. Depending if the relation is between two instances, properties or classes, one of the instances of the *InstanceRelation, PropertyRelation* or *ClassRelation* classes is used. In the following example we have used subClassRelation an instance of the *ClassRelation* to map the *o1:Father* class as a subclass to the *o2:MalePerson* class:

```
:m1 a:RelationalMapping;
  :hasSource   o1:Father;
  :hasTarget   o2:MalePerson;
  :hasRelation :subClassRelation
```

TransformationMapping specifies how the source ontology elements need be structurally modified and transformed to elements of the target ontology. Two types of transformation mapping have been modeled: (i) Property to class mapping represented by *PropToClassTransMapping* class. As an example, a property to class transformation mapping transforms the OWL triple `o1:john o1:isMarriedTo o1:jane` to

```
o2:john_jane_marriage a o2:Marriage;
 o2:hasMalePartner o1:john;
 o2:hasFemalePartner o1:merry.
```

As you can see, an instance of the class *o2:Marriage* for each pair of instances connected by the property *o1:isMarriedTo* should be created. The following instantiation of the mapping class represents this mapping from o1 to o2.

```
:m a :PropToClassTransMapping;
   :hasSourceProperty o1:isMarriedTo
   :hasTargetClass o2:Marriage;
   :hasTargetProperty1 o2:hasMalePartner;
   :hasTargetProperty2 o2:hasFemalePartner.
```

Please note the *hasSourceProperty*, *hasTargetClass*, *hasTargetProperty1* and *hasTargetProperty2* properties in this mapping and their purposes.

(ii) Class to property mapping which is the exact opposite of the property to class mapping. This mapping is represented by the *ClassToPropertyTransMapping* class. The following instantiation of the mapping ontology performs the exact opposite transformation in the abovementioned example from o2 to o1:

```
:m a :ClassToPropertyTransMapping;
   :hasTargetProperty o1:isMarriedTo
   :hasSourceClass o2:Marriage;
   :hasSourceProperty1 o2:hasMalePartner;
   :hasSourceProperty2 o2:hasFemalePartner.
```

Please note the *hasTargetProperty*, *hasSourceClass*, *hasSourceProperty1* and *hasSourceProperty2* properties in this mapping and their purposes.

ValueTransferMappings perform mathematical, string and date computation and comparison to find the new value in the target ontology based on the value of the source ontology. An instance of this mapping would be computing the Body Mass Index in the target ontology based on the weight and the height of a person in the source ontology. No relation is assigned to this type of mapping. The *hasFunction* property with the range of *Function* is used to assign the participating functions in data transformation to a mapping.

B. *Variables*

Variables that are represented by the *Variable* class can be used to represent values or a fragment of the ontology and be used as the source or target of the mappings. Fig. 1 shows subclasses of the *Variable* class. It has two subclasses: *ClassVariable, InstanceDataVariable.*

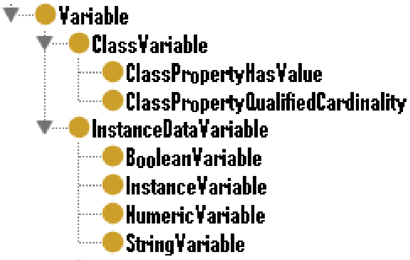


Fig. 1. Subclasses of the Variable Class

1) ClassVariable: This class and its associated properties can be used to represent a class of instances. It has two subclasses: *ClassPropertyHasValue* and *ClassPropertyQualifiedCardinality*. *ClassPropertyHasValue* can be used to create a variable which represents a class whose instances have a specific value for a specific property. For instance, the following class variable represents the students who have taken course math101 for the summer:

```
:cv1 a :ClassPropertyHasValue;
 :classVariableHasClass o1:Student;
 :classVariableHasProperty o1:hasSummerCourse;
 :classPropertyRestrictionHasValue o1:math101.
```

An instance of the *ClassPropertyQualifiedCardinality* class represents instances that have a restriction on the number and type of values that a specific property can have. For instance, we can create a class that represents students who have registered for at least two elective courses:

```
:cv2 a: ClassPropertyQualifiedCardinality;
  :classVariableHasClass o1:Student;
  :classVariableHasProperty o1:hasCourse;
  :classPropertyQCROnClass o1:ElectiveCourse
  :hasCardinalityType :min;
  :hasNumericValueForCardinality “2”^^xsd:int.
```

hasCardinalityType with the range of *Cardinality* represents the cardinality type. Instance of the *Cardinality* class are any, all, min and max.

2) InstanceDataVaraible: They have a similar purpose to data varible in programming languages. They can hold a string, numeric, boolean values or represent an instance of the ontology using sublcasses *StringVariable*, *NumericVariable*, *BooleanVariable* and *InstanceVariable* respectively. In the following example, we create an instance variable which represents all the instances of the *Student* class in the source ontology and a data variable which represents the weight of the student represented by the instance variable:

```
:studentVar a :InstanceVariable.
:weightVar a :NumericVariable.
```

```
:cv1 a :ClassPropertyHasValue;
 :hasInstanceVariable :studentVar;
 :classVariableHasClass o1:Student;
 :classVariableHasProperty o1:hasWeight;
 :classPropertyRestrictionHasValue :weightVar.
```

The value of the *weightVar* variable can be compared with a predefined number and the result can be used to make the decision whether the instance variable *studentVar* belongs to the class *o2:ThinStudent* or *o2:NormalWeightStudent* in the target ontology. In order to perform such a mapping we need to be able to define mathematical functions.

C. Functions and Operators

Expressivity of a mapping representation language is highly dependent on its support for representation of Boolean, mathematical, string, date and instance comparison and computations.

The *Function* class is the smallest entity that can be used for computation in our mapping ontology. Each function accepts an operator, a set of input variables and generates an output. A function has at most two inputs that are assigned to it by *hasInput1* and *hasInput2* properties with the domain of *Function* and range of *Variable*. Outputs of functions are assigned to them by the object property *hasOutput* with the range of *Variable* class. The operator of a function is assigned to it by the *hasOperator* property with the range of *Operator*. The *Operator* class represents all the possible operators that can be applied to ontology elements during the mappings. Fig. 2 shows the subclasses of the *Operator* class.

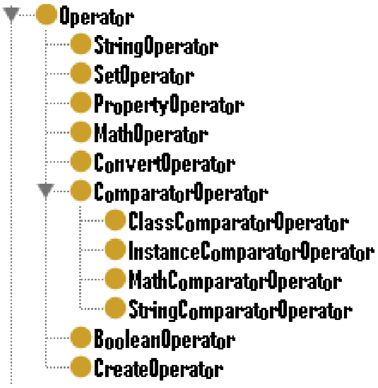


Fig. 2. Subclasses of the *Operator* class

Other than Boolean, mathematical and string operators, we have created the following operators to help with the mappings: 1. *SetOperator*: They are used to create intersection, unions and complements of classes. 2. *ConvertOperator*: Instances of the *ConvertOperator* that are convertToClass, convertToInstance and convertToProperty are used to convert any element of the source ontology to a class, instance or property respectively in the target ontology. 3. *CraeteOperator*: class is used for creating new elements in the target ontology during the mapping. 4. *ClassComparatorOperator*: Class comparators are used in the functions that compare classes to find sub-class, super-class and equivalence relations. 5. *InstanceOperator*: Instances of this class are equalInstance and notEqualInstance. The output of a function comparing two instances using equalInstnace is a Boolean variable with the value "true" if they are equivalent classes or with the value "false" otherwise. notEqualInstance works the opposite way.

D. Conditions

Mappings may be conditional. Property *hasCondition* with the domain of *Mapping* and range of *Condition* assigns conditions to a mapping. *Condition* class represents the conditions. *hasCardinalityType* with the domain of *Mapping* and the range *Cardinality* represents the cardinality type. Instance of the *Cardinality* are any, all, min and max. Data type property *hasNumericValueForCardinality* with the domain of *Mapping* shows the number of conditions that should be satisfied. A mapping whose condition satisfaction criterion is met is considered for mapping and instance transformation otherwise it is ignored. Using the abovementioned properties, one is able to express that at least three conditions of a mapping should be satisfied in order to participate in the mapping process.

V. Translation of MRO From OWL-Full to OWL-DL + SWRL

In order to use an instantiation of MRO (representing an ontology mapping) in the reasoning step of our semantic-based ontology mapping approach, we translate it to a combination of OWL-DL + SWRL or OWL2-RL + SWRL depending on the level of expressivity needed to represent the mapping. In this way, the translated mappings, the source and the target ontologies all can be regarded as a single ontology and an OWL reasoner can be used to improve the existing mappings by discovering new ones and perform instance transformation. Our translation algorithm performs the following steps on each mapping:

(1) Put all the non-output class variables in list1. Put all the output variables (Except for Boolean variables) in list2. Put all input Boolean output variables in list3.
(2) Translate the variables in list1 until no further transformation is possible.
(3) Translate the variables in list2 until no further transformation is possible.
(4) If list1 and list2 are empty, go to 5 else go to 2.
(5) Translate all the Boolean variables in list3 and process conditions.
(6) If all mappings are translated then go to 7 otherwise go to the next mapping
(7) Prepare the translated mapping for reasoning according to the translated variable.

Lists 1 and 2 are repeatedly swept for variables to be translated until both of the lists are empty. The reason is that translation of all of the output variables depends on the input variables and the translation of some of the input variables may depend on output variables. For example, an instance variable may belong to a class using property *classVariableHasClass* that is the output of a set function. In order to translate that instance variable, the class variable that it belongs to should be translated in list2 first. Steps 2, 3, 5 and 7 are further discussed in the following sub-sections.

A. *Step 2 translation of list1*

These variables are either translated to OWL constraints or SWRL axioms. If a variable has a value for one of the properties *hasInstanceVariable* or *classVariableHasValue*, it is translated to a SWRL axiom. In order to understand the translation process we go through the following example:

```
:cv1 a :ClassVariable;
 :hasInstanceVariable :personVar;
 :classVariableHasClass o1:Student;
 :classVariableHasProperty :hasWeight;
 :classVariableHasValue o1:weightVar.
```

Firstly, two SWRL variables are made with the name of the values of properties *hasInstanceVariable* and *classVariableHasValue* + “SWRLVar”:

```
:personVarSWRLVar a swrl:Variable.
:weightVarSWRLVar a swrl:Variable.
```

Then a SWRL class atom is made to represent the class to which the created instance variable belongs. This class which is represented by the *classVariableHasClass* property is *o1:Student*:

```
[a swrl:ClassAtom ;
  swrl:argument1 :personVarSWRLVar;
  swrl:classPredicate o1:Student].
```

Finally, another axiom is created to show that the created SWRL variables are connected using the property indicated by the *classVariableHasProperty* that is *hasWeight* here:

```
[a swrl:DatavaluedPropertyAtom ;
   swrl:argument1 :personVarSWRLVar;
   swrl:argument2 :weightVarSWRLVar;
   swrl:propertyPredicate o1:hasWeight;]
```

Depending if the translated class variable belongs to the source or the target of the mapping, these created SWRL axioms are added to the body or the head of SWRL rule representing this mapping respectively.

If a variable is not translated to SWRL rules, it is translated to OWL axioms. Depending on the values of the properties *classPropertyQCROnClass*, *hasCardinalityType*, and *hasNumericValueForCardinality* a class variable is translated to a cardinality restriction in OWL-DL or a qualified cardinality restriction in OWL-2. In the following example, cv1 class variable represents instances that have maximum of two different values from the *SummerCource* class for the *hasCourse* property:

```
:cv1 a :ClassVariable
 :classPropertyQCROnClass o1:SummerCourse;
 :classVariableHasProperty o1:hasCourse;
 :hasCardinalityType :max;
 :hasNumericValueForCardinality “2”^^xsd:int.
```

The above example is translated to the following OWL triples:

```
[a owl:Restriction;
  owl:onClass o1:SummerCourse;
  owl:onProperty o1:hasCourse
  owl:maxQualifiedCardinality “2”^^xsd:int]
```

B. *Step 3 translation of list2*

Output variables with different operators are translated differently. For instance, set operators are translated to OWL axioms that make use of owl:intersectionOf, owl:unionOf etc. As an example, considering the following mapping function:

```
:func1 a :Function;
  :functionHasInputVariable1 o1:Male
  :functionHasInputVariable2 o1:Parent
  :functionHasOperator :intersectionSO
  :functionHasOutputVariable :func1OutVar.
```

This example is translated to:

```
:func1OutVar :variableHasClassValue
[a owl:class;
owl:intersectionOf( :Parent :Male)].
```

Output variables of functions that make use of mathematical operators are translated to SWRL rules that make use of SWRL built-ins. For instance, in order to add up two variables a and b and put the result in the variable c, we create the following function:

```
:a a :NumericVariable. :b a :NumericVarable.
:addFunc a : Function;
  :hasInput1 :a; :hasInput2 :b; :hasOutput :c;
  :hasOperator :mathDivide.
```

This example is translated to:

```
[a swrl:BuiltinAtom ;
 swrl:arguments (:outputSWRLVar :bSWRLVar
 :aSWRLVar); swrl:builtin swrlb:divide].
```

C. *Step 5 translation of list3 and processing Conditions*

Since OWL and SWRL do not support Boolean operators, mappings are first translated into a single mapping rule without considering the Boolean functions in it. Then we iterate through all the possible combination of values of the non-output Boolean variables and compute the values of the Boolean output variables in list3. As we iterate through the values, we create a copy of the existing SWRL rule created for the current mapping and add the SWRL axioms that represent the current values of both input and output Boolean variables. In this way, each rule is copied to several rules each representing a combination of the Boolean input variables. In this way, each created SWRL rule handles a specific combination of input Boolean variables.

In order to handle conditions, we go through the created rules in the previous step and discard the SWRL rules in which the assigned Boolean variables do not meet the condition satisfaction criteria. In this way, a great number of created SWRL rules are discarded in this step.

D. *Step 7 preperation of the mappings for reasoning*

Mappings represented by SWRL rules are ready for reasoning. However, relational mappings that are represented by OWL axioms need the final translation from OWL-Full to OWL-DL. During this translation, all the variables are replaced by their translated values. For instance, consider the following translated variable and mapping:

```
:func1OutVar :variableHasClassValue
```

```
 [a owl:class;
  owl:intersectionOf(o1:Parent o1:Male)].
:m1 a:RelationalMapping;
  :hasSource   o1:func1OutVar;
  :hasTarget   o2:Father;
  :hasRelation :subClassRelation
```

This example is translated to:

```
[a owl:class;
 owl:intersectionOf(o1:Parent o1:Male)
] rdfs:subClassOf o2:father.
```

VI. Evaluation

Mapping health informatics related ontologies especially CPG ontologies is usually a challenging task due to their high levels of expressivity. In order to evaluate the efficacy of our mapping representation language, we used it to map 3 CPG ontologies with a total of 9 instantiations to a general CPG representation ontology. During the mapping process, we did not come across a mapping pattern or an operator that was not supported by our mapping ontology. We translated the mappings to OWL + SWRL and performed reasoning on them in order to discover new mappings and to perform instance transformation. We executed all the 9 transformed instantiations using the execution engine developed in [16] for executing our general CPG ontology. We also executed these CPG in their original format using their own proprietary execution engine. We compared the execution results generated by our execution engine and the original execution engines for three imaginary patient scenarios. In all 9 cases, both execution engines generated the exact same recommendations. This indicates that the mapping has been accurate and the instances are transformed successfully. In all three mappings, the translation algorithm translated the mappings to either to OWL-DL or OWL 2-RL + SWRL. This is important to ensure the decidability of the process of discovering new mappings and instance transformation.

Comparison of our mapping ontology with the existing mapping representation languages against a comprehensive set of mapping patterns surveyed in [2] shows that our mapping representation ontology supports the widest range of these mapping patters. For instance, unlike most of these languages, our mapping ontology supports variables, meta-modelling and a wide range of operators that are needed for data manipulation and structural modifications. We also introduced the possibility of conditions and complex condition satisfaction criteria.

VII. Conclusion

In this paper, we introduced a new semantic-based ontology mapping approach based on semantic web technologies. We used our approach to map three CPG ontologies to a general CPG ontology and to transform their instances. Execution results showed that our approach represents the mapping accurately and performs instance transformation correctly. Our mapping approach has three advantages over existing mapping approaches: (1) higher levels of expressivity; (2) better shareability and acceptance due to support by several semantic web tools developed for manipulation, visualization and reasoning; (3) Formal semantics in OWL and SWRL that enables us to improve the existing mappings and perform instance transformation automatically in a semantic-based ontology mapping approach. For future work, we are interested in using the functions provided by either SQWRL or SPARQLE query languages to improve the mapping representation expressivity.

Acknowledgements: This research project is sponsored by a research grant from Green Shield, Canada. The authors acknowledge the support of Green Shield, Canada.

References

[1] J. Euzenat. "An API for ontology alignment," in The Proceeding Of Semantic Web ISWC 2004, S. McIlraith, D. Plexousakis and F. Harmelen, Eds. Berlin Heidelberg: Springer, 2004, pp. 698-712.

[2] F. Scharffe, J. Bruijn, D. Foxvog. "D 4.3.2 ontology mediation patterns Library V2," Deliverable D4.3.2, EU-IST Integrated Project (IP) IST-2003-506826 SEKT, 2006.

[3] F. Scharffe, A. Zimmermann, "D 2.2.10 Expressive alignmentlanguage and implementation", Deliverable D2.2.10 EU-IST Integrated Project (IP) IST-2004-507482 SEKT, 2007.

[4] A. Maedche, B. Motik, N. Silva and R. Volz, "MAFRA - A MApping FRAmework for distributed ontologies," in Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, 2002, pp. 235-250.

[5] A. Sotnykova, C. Vangenot, N. Cullot, N. Bennacer and M. Aufaure, "Semantic mappings in description logics for spatio-teMROral database schema integration," in Journal on Data Semantics III, S. Spaccapietra and E. Zimányi, Eds. Berlin / Heidelberg: Springer, 2005, pp. 586-586.

[6] C. Meilicke, H. Stuckenschmidt and A. Tamilin, "Improving automatically created mappings using logical reasoning." in Ontology Mapping Workshop at ISWC, Athens, GA, USA, 2006, pp. 61-72.

[7] D. Calvanese, G. D. Giacomo and M. Lenzerini, "Ontology of integration and integration of ontologies," Description Logics, vol. 49, pp. 10-19, 2001.

[8] H. Stuckenschmidt, L. Serafini and H. Wache, "Reasoning about ontology mappings," ITC-IRST, Trento, 2005.

[9] P. Bouquet, F. Giunchiglia, F. van Harmelen, L. Serafini and H. Stuckenschmidt, "C-OWL: Contextualizing ontologies." in International Semantic Web Conference, 2003, pp. 164-179.

[10] J. Euzenat and P. Shvaiko, Ontology Matching. Springer-Verlag: New York Inc, 2007.

[11] E. Goldberg and Y. Novikov, "BerkMin: A fast and robust SAT-solver," in Proceedings of Design, Automation and Test in Europe Conference and Exhibition. 2002, pp. 142-149.

[12] J. P. Marques-Silva and K. A. Sakallah, "GRASP: a search algorithm for propositional satisfiability," IEEE Trans. Comput., vol. 48, pp. 506-521, 1999.

[13] H. Thomas, D. Sullivan and R. Brennan, "Ontology Mapping Representations: a Pragmatic Evaluation," Management, pp. 228-232, 2009.

[14] N. F. Noy and M. A. Musen, "Anchor-prompt: Using non-local context for semantic matching," in Proceedings of the Workshop on Ontologies and Information Sharing at the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-2001), 2001, pp. 63-70.

[15] J. d. Bruijn and A. Polleres, "Towards an ontology mapping specification language for the semantic web," DERI - DIGITAL ENTERPRISE RESEARCH INSTITUTE, Tech. Rep. DERI-2004-06-30, 2004.

[16] B. Jafarpour, S. Abidi and S. Abidi. "Exploiting OWL reasoning services to execute ontologically-modeled clinical practice guidelines," in Proceedings of the 13th conference on Artificial intelligence in Medicine, M. Peleg, N. LavraÄ and C. Combi, Eds. Berlin Heidelberg: Springer-Verlag, 2011, pp. 307-311.

[17] B. Motik, U. Sattler and R. Studer, "Query Answering for OWL-DL with rules," Web Semant. Sci. Serv. Agents World Wide Web, vol. 3, pp. 41-60, 2005.

Combined Structure-Weight Graph Similarity and its Application in E-Health

Mahsa Kiani, Virendrakumar C. Bhavsar, and Harold Boley
Faculty of Computer Science
University of New Brunswick
Fredericton, NB, Canada
{*mahsa.kiani, bhavsar, harold.boley*}*[AT]unb.ca*

***Abstract*—A combined structure-weight similarity approach for comparing directed (vertex- and edge-)labeled (edge-) weighted graphs is presented. Vertex labels (as types) and edge labels (as attributes) embody semantic information. Edge weights express assessments regarding the (percentage-)relative importance of the attributes, a kind of pragmatic information. These graphs are uniformly represented and interchanged using a weighted extension of Object Oriented RuleML. We propose semantic-pragmatic information retrieval and clustering where a combination of structure and weight similarities between a query and stored graphs is calculated. The structure and weight similarity values are used as primary and secondary criteria, respectively, to rank the retrieved graphs. The proposed weight similarity algorithm refines the ranking of retrieved graphs that have identical or nearly identical query-graph structure similarity but have different edge weights. It is shown that our approach leads to higher precision compared to earlier approaches that did not incorporate the similarity of edge weights. The proposed approach of semantic-pragmatic information retrieval and clustering can be applied, for example, in e-Learning, e-Business, social networks, and Health 3.0. In this paper, the application focus is in e-Health, specifically the retrieval of mental health records.**

***Keywords*-graph similarity; structure similarity; weight similarity; weighted Object Oriented RuleML; e-Health.**

I. Introduction

Semantic information can be represented using hierarchical structures, which express knowledge in multiple levels of detail. In the e-Business domain, vertex-labeled, edge-labeled and edge-weighted trees [1] are used in order to represent attributes of products. In [2], these weighted trees are generalized to weighted Directed Acyclic Graphs (wDAGs) in which substructures can be shared. Efficient similarity algorithms are required in many applications, such as for schema matching in databases, buyer-seller matching in e-Business, and health record retrieval in e-Health. They can also be used in social networks, e.g. to form similarity-clustered wellness or patient groups [3]. Calculating similarities between patient profiles (i.e., health records) is difficult, as the various aspects of a disease should be weighted differently, which entails that simple matching of attributes is not adequate in e-Health [4]. Weights are already used in similarity algorithms [1], [2], [4]. In [4], similar patients are identified based on similarity of symptoms and diseases. In this system, different aspects of a disease are weighted using regression estimation. Then, these calculated weights are used as coefficients in a weighted distance measure. Note that each particular user group (e.g., profiles of all patients having lung cancer) has the same values in the weight vector. This approach differs from the structural similarity algorithms [1], [2] which consider different set of weights for each profile (even if they belong to the same group). The similarity algorithms in [1], [2] compute the arithmetic mean of the two weights on corresponding edges of compared trees/wDAG in order to determine the weighted similarity. In this way, edge weights are used as scaling factors to ensure that the overall similarity value is in the real interval $[0, 1]$. We have found that this approach cannot differentiate trees nor wDAGs with different edge weights having identical or nearly identical structure similarity to the given query. Therefore, we propose modifications to the original weighted similarity algorithm to address this issue.

In this paper, a combined structure-weight similarity algorithm is proposed based on two component algorithms: a version of the structure similarity algorithm in [2] and a new weight similarity algorithm. In our approach, we perform ranked retrieval over a set of (meta)data represented as directed (vertex- and edge-)labeled (edge-)weighted graphs, each optionally associated with a data record. A special case is that the 'metadata' already are the 'data' to be retrieved, with no need for a separate data record. Similar to [2], graphs must be transformed to an internal representation before computing their similarity. Such graphs are expressed using a weighted extension of Object Oriented RuleML [5]. The XML parent-child structure reflects the hierarchical structure of the graphs, while the role element `<slot>` expresses edge labels and the attribute `weight` expresses edge weights. Also, the sharing of a rooted subgraph by multiple parents can be represented using a RuleML element with an XML `key` referred to from multiple `keyrefs`. The graphs could be expressed using other representation approaches (e.g., Turtle [6] and RDF/XML [7]) as well. We assume that, given a query graph, a ranked list of matching (meta)data graphs (and consequently corresponding records), which are stored in a dataset, is constructed. The structure similarity and the weight similarity algorithms match the query graph to each (meta)data graph and calculate their structure and edge weight similarity values, respectively. These pair values of structure and weight similarities (resulting from matching the query graph to each (meta)data graph) are considered as ranking criteria to generate the ranking list of (meta)data graphs. We demonstrate that this approach is able to differentiate the graphs having identical or nearly identical structure similarity but different edge-weight similarity to the given query.

The proposed combined structure-weight similarity approach is applied in e-Health domain. We represent (meta)data of Electronic Medical Records (EMRs) using graphs which express disorders and treatment priorities of patients. Then, our similarity approach is used to find mental health EMRs having similar (meta)data graphs to a given query. To provide patient privacy and security for health records as well as (meta)data, different technological safeguards as well as policies could be used [8]. In addition, using (meta)data could act as an extra level of privacy, as for extracting some statistics or trend, information in (meta)data itself is enough. Also, in retrieval applications, only records related to the ranked results would be retrieved not all records.

The rest of the paper is organized as follows. Section II explains our similarity approach. Section III focusses on an application of the proposed approach in the e-Health domain. Section IV concludes the paper.

II. Combined Structure-Weight Graph Similarity

In this section, graph representation and the architecture of the combined structure-weight similarity approach are presented. The theoretical basis of the proposed weight similarity is explained and the characteristics of the weight similarity are mentioned. A recursive weight similarity algorithm and the computational experiments on a synthetic dataset are presented.

A. Approach

Graph Representation: As stated earlier, we assume that we are given a set of records, with each record having an associated (meta)data represented as a graph. Note that all graphs throughout this paper are single-rooted wDAGs. All graphs are hierarchical as concepts can be represented using sub-concepts having different importance. The root vertex carries a class label, which types the main object. This object is further described by the labeled weighted edges leading to other labeled vertices of the graph, etc. Labels on outgoing edges from each given vertex are unique and appear in lexicographic (alphabetical) left-to-right order. Also, edge weights are values in the real interval $[0, 1]$ and for each graph its edge weights normalized; therefore, the sum of weights for all outgoing edges from each vertex equals 1. Further, we assume that given a query graph, a ranked list of the matching graphs is required to be constructed. Subsequently, these ranked (meta)data graphs are used to look up corresponding records. The computed weight similarity values should be comparable, therefore (similar to [1]) our graphs have to conform to the same standard schema.

Architecture: The proposed similarity approach has three modules: the structure similarity evaluation module, the weight similarity evaluation module, and the integration and ranking module (see Figure 1). We have a set of graphs $\mathbf{G} = \{G_1, G_2, G_3, \cdots, G_n\}$, which represents the (meta)data for a set of records. Both number of vertices and edges are assumed to be finite. Given a graph G', the structure similarity of G' with each member of $\mathbf{G}$ is calculated using the recursive graph similarity algorithm proposed in [2]; here G' may represent a query. The structure similarity algorithm is iterative. The given graphs are traversed from their roots to their leaves (top-down) and then their similarity is computed bottom-up. The structure similarity values and weight similarity values are in the real interval $[0, 1]$. The weight similarity evaluation module matches each member of $\mathbf{G}$ with G'; then it calculates the edge-weight similarity value. Figure 1 shows the architecture of the similarity approach where $\mathbf{G}$ and G' represent a set of graphs and a given query graph being matched, $sSim(\mathbf{G}, G')$ denotes their structure similarity values, and $wSim(\mathbf{G}, G')$ expresses their weight similarity values.

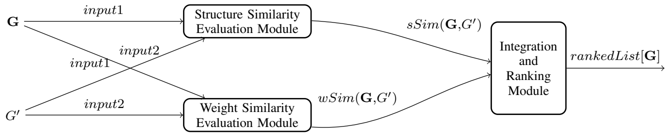

G
G'
$input1$
$input2$
$input1$
$input2$
Structure Similarity Evaluation Module
Weight Similarity Evaluation Module
$sSim(\mathbf{G},G')$
$wSim(\mathbf{G},G')$
Integration and Ranking Module
$rankedList[\mathbf{G}]$

Figure 1: Proposed Combined Structure-Weight Similarity Architecture

The structure similarity values and weight similarity values of $\mathbf{G}$ and G' are inputs to the integration and ranking module. After receiving the similarity pairs $[sSim(G_i, G'), wSim(G_i, G')]$ for all graphs $i = \{1, 2, 3, \cdots, n\}$ in set $\mathbf{G}$, the integration and ranking module ranks the graphs in $\mathbf{G}$ based on the structure similarity and weight similarity. Structure and weight similarity values could be combined with different approaches. Here, we consider weight similarity as the secondary criterion in ranking of graphs. As a result, G_1 could appear before G_2 ($G_1 \succ G_2$) in the ranked list if and only if structure similarity value of G_1 to G' (the query) is greater than the structure similarity value of G_2 to G'; or the difference between their structure similarity is less than or equal to a threshold while the weight similarity value G_1 to G' is greater than the weight similarity value of G_2 to G'. Thus,

(a) $G_1 \succ G_2$ if and only if $[sSim(G_1, G') > sSim(G_2, G')]$, or $[|sSim(G_1, G') - sSim(G_2, G')| \leq$

$Threshold$ and $wSim(G_1, G') > wSim(G_2, G')$].

(b) $G_1 \succ G_2$ or $G_2 \succ G_1$ if $[|sSim(G_1, G') - sSim(G_2, G')| \leq Threshold$ and $wSim(G_1, G') = wSim(G_2, G')]$

In this paper, we consider the threshold equal to 0. For each graph, we keep a count of the number of edges, assigning a unique integer j to each edge, starting from 1 in top-down (root to leaf) and left-to-right order. As a result, each edge is represented by e_j, $j \,\epsilon \{1, 2, 3, \cdots, z\}$, considering z as the total number of edges in a graph. As all edges are directed, the source vertex u and the destination vertex v of each edge e_j can be represented as an ordered pair (u, v). Also, the weight of edge e_j is represented as $w(e_j)$. The edge e_j in graph G and the edge $e_{j'}$ in graph G' are called *corresponding edges* if and only if they have identical edge labels as well as identical source vertex labels and destination vertex labels. The relation between corresponding edges e_j and $e_{j'}$ is denoted as $e_j \doteq e_{j'}$. Consider d_u as the depth of vertex u. In our graphs, d_u and $d_{u'}$ are equal for two corresponding edges e_j and $e_{j'}$.

B. Weight Similarity

In the proposed weight similarity approach, the similarity of weights related to two corresponding edges can be calculated based on two similarity measures [9], viz. Manhattan distance, Equation 1, or Min/Max similarity measure, Equation 2, as given below:

$$weSim1 = 1 - |w(e_j) - w(e_{j'})| \tag{1}$$

$$weSim2 = \frac{\min(w(e_j), w(e_{j'}))}{\max(w(e_j), w(e_{j'}))} \tag{2}$$

The importance of each edge can be considered to be a function of the depth of its source vertex. As stated earlier, the root vertex carries a class label, which types the main object; therefore, the outgoing edges from the root have the highest importance. This importance decreases as the depth of the source vertex of the edge increases. Similarly, contribution of the weight similarity of two corresponding edges in weight similarity of two graphs depends on the depth of the source vertex related to corresponding edges. The coefficient for adjusting the contribution of edge weight similarity needs to decreases as the depth of the source vertex of corresponding edges increases. One approach for defining this coefficient is using an exponential function with D as the fixed base and $d+1$ as the variable exponent. Therefore, in this paper, the adjustment coefficient is expressed as D^{d+1}. If p enumerates the pairs of corresponding edges in depth d and m_d $(m_d \geq 0)$ denotes the number of corresponding edges in depth d, the weight similarity value of graphs is expressed using Equation 3. In this equation, each edge similarity value is multiplied by D^{d+1}, in which D is the global depth degradation factor ($D \leq 0.5$) and d is the depth of the source vertex of the edge. $0 \leq d \leq d_{max}$, where d_{max} is the maximum possible depth of the source vertex of corresponding edges in two graphs.

$$Sim = \sum_{d=1}^{d_{max}} (\sum_{p=1}^{m_d} weSim_p \cdot D^{d+1}) \tag{3}$$

As the similarity of weights, numbers in the real interval $[0, 1]$, related to two corresponding edges is calculated using the Manhattan distance (Equation 1) or the Min/Max similarity measure (Equation 2), the similarity value of a pair of weights $weSim_p$, $p \,\epsilon \{1, 2, 3, \cdots, m_d\}$ is in interval $[0, 1]$. Also d, which is the depth of the source vertex related to an edge, could be a value larger than or equal to 0. As a result, D^{d+1} is a positive number. Thus, the summation of $(weSim_p \cdot D^{d+1})$ for all corresponding edges could result in a value larger than 1 and therefore Sim could be greater than 1. In order to express the graph similarity as a value in real interval $[0, 1]$, the combined edge weight similarity values (viz. Sim) is normalized by the sum of the D^{d+1} used in various iterations of the recursive weight similarity algorithm. Starting from the first level in graphs, each time a pair of weights is compared, the related depth factor is added and this process is repeated for all levels of graphs. The normalization factor denoted by F is expressed as,

$$F = \sum_{d=1}^{d_{max}} (\sum_{p=1}^{m_d} D^{d+1}) \tag{4}$$

Thus, the normalized weight similarity of two graphs $(wSim)$ is given as,

$$wSim = \frac{Sim}{F}, \tag{5}$$

which lies in real interval $[0, 1]$. The global depth degradation factor (D) could be equal to 1. In this case, the proposed similarity approach gives the same importance to the weight similarities of various levels of the graphs and the arithmetic mean of the weight similarity values is calculated. Therefore, the result of such a calculation is identical to considering the weight similarity of all attributes having the same effect on the weight similarity of two graphs. This approach results in a linear trend of similarity values. In Equation 6, m_{total} denotes the number of corresponding edges in total. $weSim1(w(e_j), w(e_{j'}))$ is the similarity of weights related to two corresponding edges based on the Manhattan distance, while $wSim$ is the global weight similarity of two graphs based on the Manhattan distance. The same relation holds when the weight similarity is calculated based on the Min/Max similarity measure as well.

$$wSim = (1/m_{total}) \cdot \sum_{k=1}^{m_{total}} (weSim1(w(e_j), w(e_{j'}))) \tag{6}$$

The weight similarity also has the following characteristics: (a) The similarity value generated by the weight similarity approach is a non-negative number. The minimum similarity value equals 0. (b) The weight similarity of a graph to itself is 1.0. The similarity of each pair of weights $weSim(w(e_j), w(e_{j'}))$ is 1.0. Therefore, Sim has the same value as F and as a result the weight similarity of two graphs (i.e., $wSim$) is equal to 1.0. (c) The weight similarity measure is a symmetric function, as the order of pair of graphs does not affect the result of the computation of weight similarity. (d) The weight similarity like many other similarities does not obey triangular inequality. The weight similarity measure is a partial matching approach as only the weights related to the corresponding edges are compared.

C. Algorithm

Algorithm 1, which calculates the weight similarity of two graphs based on Manhattan distance, is represented in Figure 2.

```
 1: procedure wSIMILARITY($G$, $G'$)
 2:     if $G$ or $G'$ only contains a single vertex then
 3:         return 0
 4:     end if
 5:     if $G.root.label \neq G'.root.label$ then
 6:         return 0
 7:     else
 8:         $d \leftarrow root(G).depth$
 9:         $k \leftarrow 1$
10:         $k' \leftarrow 1$
11:         while $k \leq G.root.outDegree$
                        $\wedge\ k' \leq G'.root.outDegree$ do
12:             $e_j \leftarrow G[k].root.edge$
13:             $e'_j \leftarrow G'[k'].root.edge$
14:             if $e_j \doteq e_{j'}$ then
15:                 $F \leftarrow F + D^{d+1}$
16:                 $weSim \leftarrow (1 - |w(e_j) - w(e_{j'})|)$
17:                 $Sim \leftarrow Sim + weSim \cdot D^{d+1}$
                        $+\ wSimilarity(G.subgraph(e_j),$
                                    $G'.subgraph(e_{j'}))$
18:                 $k \leftarrow k + 1$
19:                 $k' \leftarrow k' + 1$
20:             else if $e_j \succ e_{j'}$ then
21:                 $k \leftarrow k + 1$
22:             else
23:                 $k' \leftarrow k' + 1$
24:             end if
25:         end while
26:         $wSim \leftarrow Sim/F$
27:         return $wSim$
28:     end if
29: end procedure
```

Figure 2: Algorithm 1. Weight Similarity of two Graphs based on Manhattan Distance

Algorithm 1 (see Figure 2) gives the weight similarity algorithm, which traverses two input graphs G and G' in a left-right depth-first strategy. The parameter of the algorithm is D, which represents the global depth degradation factor. Here we assume that D is equal to 0.5; however, a learning component could be used to adjust the parameter. Considering graphs G and G' as the inputs of the algorithm, $G.subgraph(e_j)$ denotes the sub-graph rooted at destination vertex of e_j in graph G. $G.root.label$, $G.root.inDegree$, and $G.root.outDegree$ represent vertex label, in-degree, and out-degree of the root of graph G, respectively. Also, $e_j \succ e_{j'}$ represents that e_j could appear before $e_{j'}$ in a lexicographic ordered list. $weSim$ is the similarity of weights related to two corresponding edges. $root(G).depth$ is a function which gives the depth for root of graph G relative to the root of the original graph. The output, $wSim$, is the weight similarity value of G and G'.

The proposed weight similarity algorithm traverses two given graphs in a top-down (root-leaf) order to compute the edge-weight similarity of the graphs. If two edges being traversed are corresponding edges, their weight similarity is calculated using Equation 1 or 2. Two pointer variables, k and k', indicate the positions of two outgoing edges being matched. If $e_j \succ e_{j'}$, k is set to point to the next outgoing edge in G, while if $e_{j'} \succ e_j$, k' would be increased to point to the next outgoing edge in G'. If $e_j \doteq e_{j'}$, k and k' are set to point to the next outgoing edges in G and G', respectively. The loop is terminated as soon as any one of the following conditions is met: $k > G.root.outDegree$ or $k' > G'.root.outDegree$.

The algorithm is recursive, so the base case and recursive case should be defined. The base case is where the problem can be solved directly, while in the recursive case the problem is expressed as subproblems that are closer to the base case [10, pp. 228]. In this algorithm the base of the recursion is where G or G' only contains a single vertex (Algorithm 1, lines 2-4) or if $G.root.label \neq G'.root.label$ (Algorithm 1, lines 5-6). In both cases, their weight similarity is 0. The algorithm is tail recursive, i.e., the recursive invocation is the very last thing which is performed [10, pp. 245]. In the recursive case, the algorithm recursively invokes itself using the roots of two sub-graphs of G and G' as arguments (Algorithm 1, line 17).

As stated earlier, the labels of outgoing edges from each vertex are arranged in the lexicographic order. Also, two pointers indicate the positions of two edges being matched. Using these features, the time complexity of the algorithm is improved. If G or G' only contains a single vertex or $G.root.label \neq G'.root.label$ for the roots of two graphs, then the algorithm sets the weight similarity directly to 0 without any further computation; If $G.root.label = G'.root.label$, the algorithm uses one loop (Algorithm 1, line 11) to find the corresponding edges. For two graphs, consider t, $t \,\epsilon\, \{1, 2, 3, \cdots, r\}$, in which r equals to the total number of pairs of matched non-leaf vertices. When matching all outgoing edges of a pair of vertices, three cases should be considered: (i) If $e_j \succ e_{j'}$ or $e_j \doteq e_{j'}$, for all values of k and k', the number of iterations equals to ${I_G}^t = G.root.outDegree$, (ii) If

$e_{j'} \succ e_j$, for all values of k and k', the number of iterations is equal to ${I_{G'}}^t = G'.root.outDegree$, and (iii) If only for some values of k and k', $e_j \succ e_{j'}$ or $e_j \doteq e_{j'}$, the number of iterations to find the corresponding edges is in the interval $[\min({I_G}^t, {I_{G'}}^t), \max({I_G}^t, {I_{G'}}^t)]$. The number of iterations for finding all corresponding edges in graphs, I, equals to the summation of iterations performed for each pair of vertices; I is in interval $[\sum_{t=1}^{r} \min({I_G}^t, {I_{G'}}^t), \sum_{t=1}^{r} \max({I_G}^t, {I_{G'}}^t)]$. In the worst case, $I = \sum_{t=1}^{r} \max({I_G}^t, {I_{G'}}^t)]$, and therefore the complexity of the algorithm is $\Theta(\sum_{t=1}^{r} \max({I_G}^t, {I_{G'}}^t))$.

D. Computational Experiments

Now, we test the proposed weight similarity algorithm on a synthetic dataset, in which weights are changed systematically to understand the effects of structure and weights on the similarity. The dataset contains graphs structurally identical to the graphs given in Figure 3, but with different weights. The graphs are balanced with maximum breadth assuming branching factor of 2. The dataset contains 29 graphs.

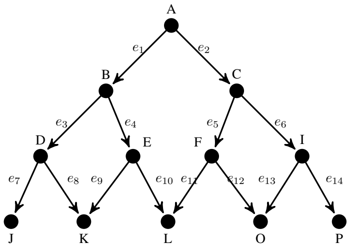


Figure 3: Graph Structure of Metadata in Dataset

In this dataset, we have five possible values for a pair of edge weights: $[0.01, 0.99]$, $[0.25, 0.25]$, $[0.5, 0.5]$, $[0.75, 0.25]$, or $[0.99, 0.01]$. In G_1 of dataset, weights of all edges having the same source vertex are $[0.01, 0.99]$. Now, we change the edge weights from right to left in a level and then bottom-up for various levels, exhausting the five possible sets of edge weight pairs. This results in 29 graphs in the dataset, of which eight graphs, G_1 to G_8, are shown in Table I, where each row represents the weights related to a graph [1]. Enabling a compact specification and description of the weights, this notation is used to illustrate different weight values for one graph structure.

Considering this systematic changes in weights, the weight similarities of G_1 in the dataset with respect to the remaining graphs are expected to decrease gradually. Therefore, the synthetic dataset provides a starting point for an evaluation of our weight similarity algorithm.

Table I: Edge Weights of a Subset (G_1 to G_8) of 29 Graphs (G_1 to G_{29}) with the Structure given in Figure 3

((a)) Weights of Edges e_1 to e_7

Graph	$w(e_1)$	$w(e_2)$	$w(e_3)$	$w(e_4)$	$w(e_5)$	$w(e_6)$	$w(e_7)$
G_1	0.01	0.99	0.01	0.99	0.01	0.99	0.01
G_2	0.01	0.99	0.01	0.99	0.01	0.99	0.01
G_3	0.01	0.99	0.01	0.99	0.01	0.99	0.01
G_4	0.01	0.99	0.01	0.99	0.01	0.99	0.01
G_5	0.01	0.99	0.01	0.99	0.01	0.99	0.25
G_6	0.01	0.99	0.01	0.99	0.25	0.75	0.25
G_7	0.01	0.99	0.25	0.75	0.25	0.75	0.25
G_8	0.25	0.75	0.25	0.75	0.25	0.75	0.25

((b)) Weights of Edges e_8 to e_{14}

Graph	$w(e_8)$	$w(e_9)$	$w(e_{10})$	$w(e_{11})$	$w(e_{12})$	$w(e_{13})$	$w(e_{14})$
G_1	0.99	0.01	0.99	0.01	0.99	0.01	0.99
G_2	0.99	0.01	0.99	0.01	0.99	0.25	0.75
G_3	0.99	0.01	0.99	0.25	0.75	0.25	0.75
G_4	0.99	0.25	0.75	0.25	0.75	0.25	0.75
G_5	0.75	0.25	0.75	0.25	0.75	0.25	0.75
G_6	0.01	0.25	0.75	0.25	0.75	0.25	0.75
G_7	0.01	0.25	0.75	0.25	0.75	0.25	0.75
G_8	0.25	0.75	0.25	0.75	0.25	0.75	0.25

Figure 4 depicts the similarity values of G_1 for the synthetic dataset with the remaining graphs using the graph similarity algorithm in [2] as well as our combined structure-weight similarity algorithm (using the similarity measure based on the Manhattan distance).

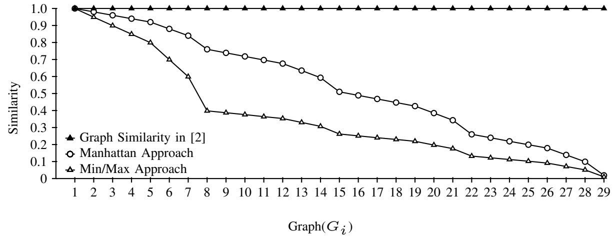


Figure 4: Similarity of G_1 to 29 Graphs in the Dataset

While similarity values based on the previous graph similarity algorithm [2] are always equal to 1, our combined structure-weight similarity approach differentiates the structurally identical graphs with different weights. Also, Figure 4 gives a comparison of the two similarity measures, the similarity measure based on the Manhattan distance and the Min/Max similarity measure. Here again we compute the similarity w.r.t. G_1. For the depth degradation factor equal to 0.5 and for the same set of weights for a dataset, both similarity measures generate similarity values with a decreasing trend. It is important to note that for Figure 4, the similarity decreases as a result of the systematic change of weights of edges (having the same source): gradual increase of the edge weight for the left vertex and gradual decrease of the edge weight for the right vertex. The bumps in the similarity plots (e.g. at G_8, G_{15}, and G_{22}) are observed as the result of level transitions, i.e., the systematic changes of weights in each level of the graph.

[1] The complete dataset is available from authors.

We have given above the computational results for the similarities of members of a dataset. We can generalize the behavior of the similarity computation to other possible graph structures such as trees [1], and generalized trees [11], and conclude that the proposed weight-similarity algorithm, with any one of the similarity measures, is effective in differentiating graphs having identical or nearly identical structure similarity values (but different weights). Weight similarity considers only weight of common subgraphs of two graphs being compared, while structure similarity takes into account common as well as uncommon subgraphs. Therefore, two graphs could be similar from weight similarity perspective, while their uncommon sub-graphs are large (i.e. small structure similarity). Note that although the numerical similarity values of the two similarity measures are different, they result in the same relative ranking of the graphs for the given query. Since there is no universal benchmark for evaluating similarity [12], it is not possible to select or recommend one of the similarity measures over the other and both similarity measures could be used for the purposes of relative ranking.

III. E-Health Application: Mental Health Electronic Medical Record

Group therapy is used as a treatment option for drug abusers [13, pp. 577-620]. Newcomers should be placed in groups with at least one or two similar members. Open group membership in which new members are allowed to enter as others leave is the norm [14, pp. 262-273]. Therefore, retrieving similar mental health EMRs to select patients for group therapy is a challenging task. This selection should be based on the gathered dynamic, behavioral, and diagnostic information in a screening interview [15, pp. 934]. Consider the scenario where the user (e.g., a psychologist) wants to find an appropriate group for a new patient in order to schedule group therapy sessions. In this case, mental health EMRs that describe similar disorders as well as treatment priorities should be found. Each (meta)data expresses the individualized treatment plan about patient's disorders and the treatment priorities based on the last psychological evaluation. Similar to [2], we represent the attributes of each (meta)data using a graph based on a standard schema. The attributes of this schema are extracted from [15], [16], and the terms representing the (meta)data are based on DSM IV [16]. The attributes express possible affective, behavioral, and cognitive problems of a patient. The edge weights in graphs represent the relative priority regarding treatment of each disorder in the group therapy session. Therefore, severe, influential, and dangerous disorders as well as the items for which treatments have the greatest benefit have higher priority (i.e., higher weight) in our treatment-oriented (meta)data. As treatment priorities change over time, edge weights could be different in each evaluation phase by the psychologist. In order to select patients for group therapy, in the proposed system the edge weights of (meta)data are always related to the last psychological evaluation of patients (available in the mental health records). Figure 5 illustrates the generic structure of (meta)data of mental health EMRs in the database as well as a query having the same structure.

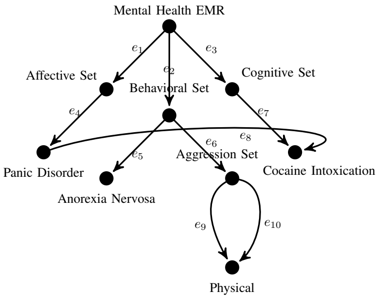


Figure 5: Graph Structure of a Query and Metadata of Mental Health EMRs

Table II represents the edge labels of the generic structure (in Figure 5), in which $l(e_j)$ denotes the label of edge e_j, $j \, \epsilon \, \{1, 2, 3, \cdots, 10\}$. The patients have panic disorder and also delirium due to cocaine intoxication. Other disorders of the patients are anorexia nervosa and physical aggression including fantasies and real acts [15, pp. 421].

Table II: Edge Labels of a Query and Metadata Graphs (having the Structure in Figure 5) for Mental Health EMRs

$l(e_1)$ Affective disorders	$l(e_6)$ Aggression
$l(e_2)$ Behavioral disorders	$l(e_7)$ Delirium
$l(e_3)$ Cognitive disorders	$l(e_8)$ Substance induced panic
$l(e_4)$ Anxiety	$l(e_9)$ Real act
$l(e_5)$ Appetite disorder	$l(e_{10})$ Fantasies

Edge weights of four EMR (meta)data, representing the diagnosis segment of a mental health EMR, and a query are illustrated in Table III. Note the different last subscripts for the two edges emanating from the Aggression vertex and terminating at the same Physical destination vertex. Further, there are three edges from the root vertex.

Table III: Edge Weights of a Query (G'_1) and four Metadata Graphs (having the Structure in Figure 5) for Mental Health EMRs

Graph	$w(e_1)$	$w(e_2)$	$w(e_3)$	$w(e_4)$	$w(e_5)$	$w(e_6)$	$w(e_7)$	$w(e_8)$	$w(e_9)$	$w(e_{10})$
G'_1	0.01	0.01	0.98	1.0	0.01	0.99	1.0	1.0	0.01	0.99
G_1	0.01	0.01	0.98	1.0	0.01	0.99	1.0	1.0	0.01	0.99
G_2	0.5	0.25	0.25	1.0	0.25	0.75	1.0	1.0	0.25	0.75
G_3	0.4	0.3	0.3	1.0	0.5	0.5	1.0	1.0	0.5	0.5
G_4	0.3	0.35	0.35	1.0	0.75	0.25	1.0	1.0	0.75	0.25

Now we compare the similarity of query with the four (meta)data graphs G_1, G_2, G_3, and G_4 of the EMRs given in Tables II and III using the combined structure-weight similarity algorithm. The computed similarity values are given in Table IV. The structure similarity values between query G' and any of four (meta)data graphs are identical; therefore, we cannot distinguish between them using the structure similarity alone. The edge weight similarity results using the proposed algorithm are also shown in Table IV.

Table IV: Computational Results for the Metadata of Mental Health EMRs and the Query in Table III

Graph	Graph	Structure Similarity	Manhattan Approach	Min/Max Approach	Rank
G'	G_1	1.0	1.0	1.0	1
G'	G_2	1.0	0.6834	0.3762	2
G'	G_3	1.0	0.6356	0.3492	3
G'	G_4	1.0	0.5878	0.3249	4

We can clearly see that the similarities are different and they can be used to rank four (meta)data graphs. Further, both similarity measures (see columns 4 and 5 in Table IV) are equally acceptable as they result in the same relative ranks. Instead of ranked graphs based on their similarity to a given query, the proposed approach could cluster the mental health EMRs based on a threshold to facilitate creation of supportive virtual communities, which is one of the main goals of Health 3.0 [17].

IV. Conclusion

Our combined structure-weight similarity approach is able to distinguish graphs having identical or nearly identical structure but different weights. By considering the weight similarity in addition to the structure similarity, preferences of user are compared with the preferences expressed as edge weights of graphs stored in dataset. The similarity of edge weights is calculated in a recursive way, giving more importance to weights of edges in higher levels of a graph. The combined structure-weight similarity algorithm has been implemented in Java and it has been applied to retrieve mental health electronic medical records (EMRs).

Acknowledgments

The authors would like to thank Mehrdad Kiani, M.D., who provided help in the health domain. This research is partially funded by a Discovery Grant from the Natural Sciences and Engineering Council of Canada and a Ph.D. Fellowship Grant from the Atlantic Computational Excellence Network (ACEnet) awarded to the second author.

References

[1] Bhavsar, V.C. and Boley, H. and Yang, L., “A Weighted-Tree Similarity Algorithm for Multi-Agent Systems in E-Business Environments,” *Computational Intelligence*, vol. 20, no. 4, pp. 584–602, 2004.

[2] Jin, J., “Similarity of Weighted Directed Acyclic Graphs,” MSc Thesis, Faculty of Computer Science, University of New Brunswick, Canada, Sep. 2006.

[3] Boley, H. and Shafiq, O., and Smith, D. and Osmun, T., “The Social Semantic Subweb of Virtual Patient Support Groups,” in *Proc. the 3rd Canadian Semantic Web Symposium (CSWS2011), Vancouver, British Columbia, Canada.* CEUR, Aug. 2011, pp. 1–18.

[4] Fritz, P. and Klenk, S. and Dippon, J. and Heidemann, G., “Determining patient similarity in medical social networks,” in *Proc. MedEx Workshop*, 2010, pp. 6–13.

[5] H. Boley, “Object-Oriented RuleML: User-Level Roles, URI-Grounded Clauses, and Order-Sorted Terms,” in *Proc. Rules and Rule Markup Languages for the Semantic Web (RuleML-2003).* LNCS 2876, Springer, Oct. 2003, pp. 1–16.

[6] D. Beckett and T. Berners-Lee. (2011) Turtle A Readable RDF Syntax. [Online]. Available: http://www.w3.org/TeamSubmission/turtle/

[7] Cyganiak, R. and Wood, D., Ed., *Resource Description Framework (RDF): Concepts and Abstract Syntax.* World Wide Web Consortium, Jan. 2013. [Online]. Available: http://www.w3.org/TR/2013/WD-rdf11-concepts-20130115/

[8] Jacques, L.B., “Electronic Health Records and Respect for Patient Privacy: A Prescription for Compatibility,” *Vand. J. Ent. & Tech. L.*, vol. 13, pp. 441–462, 2011.

[9] Boriah, S. and Chandola, V. and Kumar, V., “Similarity Measures for Categorical Data: A Comparative Evaluation,” in *Proc. the 8th SIAM International Conference on Data Mining*, 2008, pp. 243–254.

[10] Drake, P., *Data Structures and Algorithms in Java.* Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2005.

[11] Dehmer, M. and Emmert-Streib, F. and Kilian, J., “A Similarity Measure for Graphs with Low Computational Complexity,” *Applied Mathematics and Computation*, vol. 182, no. 1, pp. 447 – 459, 2006.

[12] Janowicz, K. and Raubal, M. and Schwering, A. and Kuhn, W., “Semantic Similarity Measurement and Geospatial Applications,” *T. GIS*, vol. 12, no. 6, pp. 651–659, 2008.

[13] Carr, A., *The Handbook of Child and Adolescent Clinical Psychology: A Contextual Approach.* Rouledge, New York: Taylor and Francis Group, 1999.

[14] Ruiz, P. and Strain, E.C. and Langrod, J., *The Substance Abuse Handbook*, ser. Doody’s all reviewed collection. Philadelphia: Wolters Kluwer Health/Lippincott Williams and Wilkins, 2007.

[15] Sadock, B.J. and Sadock, V.A., *Kaplan and Sadock’s Synopsis of Psychiatry: Behavioral Sciences/Clinical Psychiatry*, 10th ed. Philadelphia: Lippincott Williams and Wilkins, 2007.

[16] A. P. Association and A. P. A. T. F. on DSM-IV., *Diagnostic and Statistical Manual of Mental Disorders: DSM-IV-TR.*, 4th ed., ser. Diagnostic and Statistical Manual of Mental Disorders. Washington, DC: American Psychiatric Association, 2000.

[17] Kiani, M. and Bhavsar, V.C. and Boley, H., “Clustering Using Combined Structure-Weight Graph Similarity,” University of New Brunswick, Canada, Internal Report (In Preparation).

Part III.

Short Research Papers

Parallelizing Algebraic Reasoning for the Description Logic $\mathcal{SHOQ}$

Jocelyne Faddoul
Centre for Logic and Information
St. Francis Xavier University
Nova Scotia, Canada
Email: jfaddoul@stfx.ca

Wendy MacCaull
Centre for Logic and Information
St. Francis Xavier University
Nova Scotia, Canada
Email: wmaccaul@stfx.ca

***Abstract*—Reaching the full potential of the semantic web awaits the availability of highly scalable reasoners. Despite numerous efforts to optimize existing Description Logics reasoners, there is always the need to compromise the expressivity or the size of the used ontologies in time sensitive applications. Hybrid algebraic reasoning has been investigated in the context of optimizing reasoning with ontologies where the expressivity is rich enough to include qualified cardinality restrictions and nominals. On the other hand parallel models have been considered to allow scalable reasoning with ontologies, however, only poor Description Logic expressivity has been considered. In this work, we investigate parallelizing hybrid algebraic reasoning as a means to seek scalable solutions without the need to sacrifice expressivity.**

I. Motivation

Applications of the semantic web are numerous, wide ranging and have tremendous potential for adding value in a vast array of situations which can take advantage of intelligence, i.e., the capacity to reason over knowledge stored in a knowledge base such as an ontology. However, if the application is time sensitive, the time required for reasoning can be prohibitive.

Description logics (DL) have gained a lot of attention in the research community as they provide a logical formalism for the codification of medical knowledge, ontologies, and the semantic web. There has been a great deal of research into optimizing DL reasoning strategies and in carving out fragments over which reasoning can proceed at a reasonable pace — but reasoning using these strategies or over these fragments often does not scale to allow the use of large ontologies. Reasoning for time sensitive tasks still requires severe restrictions on the expressivity, the complexity and/or the size of the ontology which, of course, limits the knowledge that can be used.

Standard DL inference services, e.g., TBox classification, concept satisfiability checking, instance checking, etc., have been extended with query answering in order to extract information and drive applications such as web services and workflow management systems [1]. For many applications (e.g., associated with health services delivery) these services are time sensitive, but require time consuming reasoning over complex and often large ontologies. The expressivity of the domain knowledge is often sacrificed in order to meet practical reasoning performance, hence the recent popularity of lightweight ontologies, i.e., expressed using the extensions of the tractable DL $\mathcal{EL}$. Sacrificing the expressivity of the knowledge modelled is a limiting (and often unacceptable) compromise. For example, given the Foundational Model of Anatomy (FMA) ontology[1], one might add axiom (1) to express the fact that the adult human has 206 bones, and axiom (2) to express the fact that the knee joint is involved in more than 100 rheumatic diseases[2]. These axioms use the qualified cardinality restrictions (QCRs) DL operator, which is known to lead to severe performance degradation of existing state of the art DL reasoners (e.g., Fact++[3], Hermit[4], Pellet[5]). RacerPro[6] remains the only DL reasoner that can efficiently handle QCRs using algebraic reasoning, however, it does not fully support nominals.

$$Adult \sqsubseteq Person \sqcap \geq 206\, hasBone \quad (1)$$

$$KneeJoint \sqsubseteq \geq 100\, involvedInDisease.RhDisease \quad (2)$$

To the best of our knowledge, algebraic reasoning remains the most promising approach for DL reasoning with ontologies relying on the use of QCRs. This has been shown in fragments of DL using Qualified Cardinality Restrictions (QCRs)[2], [3], inverse roles [4], [5], and nominals [6], [7]. Practical implementations of such algebraic tableau algorithms requires a carefully chosen set of optimizations in order to outperform the highly optimized existing state of the art reasoners. Most algebraic tableau-based algorithms proposed so far are double exponential in the worst case; their optimized implementations have been tested on a suite of artificial or often adapted subsets of ontologies. The scalability of the algebraic approach with real world and often large ontologies remains open.

The high performance computing (HPC) paradigm would seem to offer a solution to these problems, but progress using high performance computing methodologies has been challenging and slow [8], [9], [10]. The techniques that have offered speedy solutions in other domains (e.g., for "number

[1] http://sig.biostr.washington.edu/projects/fm/index.html

[2] http://www.medicinenet.com/knee_pain/article.htm

[3] http://owl.man.ac.uk/factplusplus/

[4] http://www.hermit-reasoner.com/

[5] http://clarkparsia.com/pellet/

[6] http://www.racer-systems.com/

crunching" in the physical sciences) do not suffice to crack the time bottleneck of reasoning tasks required for effective use of ontologies. Work is needed to find techniques for this kind of computing. The increasing availability of cloud computing facilities means that we can all have access to powerful computing resources; indeed, our own laptops have multiple cores. New methods are needed if we are to take advantage of their potential.

Recently, there has been encouraging results [11], [12], [10], [13], [14]. The work considered so far, considers parallelizing the TBox classification task [12], the Abox querying task [15], or the concept satisfiability checking task [8], [11] using ontologies relying on the least expressive fragments of DLs. Parallelizing algebraic reasoning to allow the handling of large ontologies using number restrictions needs further investigation.

Our research is focused on finding ways to combine high performance computing and algebraic tableau reasoning [6] to enable scalable reasoning with ontologies handling the expressivity of the DL $\mathcal{SHOQ}$.

II. High Performance Computing and Algebraic Reasoning

In this work, we investigate combining HPC and algebraic tableau reasoning for deciding DL concept satisfiability. Every standard DL reasoning task can be reduced to a concept satisfiability check. Our goal is to parallelize the algebraic tableau reasoning algorithm presented in [6] for the DL $\mathcal{SHOQ}$, which is basic DL $\mathcal{ALC}$ extended with transitive roles, role hierarchies, nominals and qualified cardinality restrictions, and for which the satisfiability problem is ExpTime-complete.

The algebraic algorithm presented in [6] decides the satisfiability of a concept C by constructing a compressed completion graph representing a model. The algorithm is hybrid; it relies on tableau expansion rules working together with an integer programming solver (e.g., simplex solver) and comes with a double exponential worst case complexity. However, in practice and when equipped with suited optimizations, algebraic reasoning performs better than existing state of the art reasoners in handling qualified cardinality restrictions and nominals [16], [7]. In this paper, we argue that algebraic reasoning is well suited for parallel programming models offering a potential improvement over standard tableau-based DL reasoning.

A. Parallel Reasoning

Constructing completion models for DL concepts often requires non-deterministic choices, which result in separately exploring more than one completion graph expansion. In the case of the DL $\mathcal{SHOQ}$, non-deterministic tableau-rules lead to an independent construction of tableau branches since nodes belonging to different branches do not exchange information. This feature suggests that we extend the search strategy adopted to construct tableau models using parallel processing. In the following, we list and compare the main sources of non-deterministic expansions in the cases of standard tableau DL reasoning and hybrid algebraic tableau DL reasoning for the DL $\mathcal{SHOQ}$.

a) Standard Tableau: In the case of the standard tableau algorithm for the DL $\mathcal{SHOQ}$ [17], non-determinism is due to:

- handling disjunctions (The $\sqcup$-Rule): if there exists in the completion graph a node x such that $C_1 \sqcup C_2$ is in the label, $\mathcal{L}(x)$, of x then there can be two possible and distinct ways to extend the completion model: one in which C_1 is added to the label of x, and one in which C_2 is added to the label of x.
- handling qualified cardinality restrictions (*choose*-rule, $\leq$-rule): if there exists in the completion graph a node x with $\leq nR.C$ its label and there exists $m >= 1$ nodes y, $y_1 \ldots y_m$, related to x via the role R, then:
  - for each y_m there can be two possible and distinct ways to extend the completion model: one in which C is added to the label of y_m, and one in which $\neg C$ is added to the label of y_m (*choose*-rule).
  - if $m > n$ there can be $\frac{m!}{n!}$ possible and distinct ways to extend the completion model such that in each case excess role fillers (y_i and y_j, $i \neq j$) are merged until the at-most restriction is satisfied ($\leq$-rule).

b) Algebraic Tableau: In the case of the hybrid algebraic tableau algorithm for the DL $\mathcal{SHOQ}$ [6], disjunctions are handled similar to the case of standard tableau. However, handling qualified cardinality restrictions relies on the use of the atomic decomposition technique [18], which computes disjoint partitions by considering all possible interactions between domain elements. This handling of domain elements results in only one additional source of non-determinism rather than the two sources for handling qualified cardinality restrictions with the standard tableau:

- handling partitions (the ch-Rule): for each partition computed by the atomic decomposition technique there can be two possible and distinct ways to extend the completion model: one in which the partition must be empty, and one in which the partition must have at-least one element.

We argue that hybrid algebraic reasoning appears to have a better potential for parallelization than standard tableau for the following reasons:

- having less sources of non-determinism (2 instead of 3) means less overhead in managing concurrent execution of non-deterministic rules. This also means that adopting optimizations such as dependency directed backtracking becomes more fine grained and less complicated.
- the ch-rule always fires for two choices. This means that the search trees resulting from the distinct branches have similar structure which facilitates load balancing between parallel expansions of the search tree. Load balancing is a common goal in parallel computing where unequal thread workloads can easily diminish the performance gain of parallelization.
- satisfying qualified cardinality restriction is delegated to an inequation solver and can be done in isolation from tableau expansion. This means that the task of satisfying

the inequations can be delegated to the use of separate threads, or even FPGAs [14] and GPUs (Graphical Processing Units).

- the use of "compressed completion graph" consisting of proxy nodes representing sets of domain elements instead of completion graphs consisting of a node representing each domain element allows the use of a smaller data structure representing the completion model. This means that a smaller amount of data needs to shared among and communicated between parallel tasks thus reducing the communication overhead between threads.

B. The Parallel Execution Framework

We consider parallelization of the hybrid algebraic reasoning algorithm using an object-oriented framework supporting thread level parallelism (TLP). In this framework, a compressed completion graph data structure is shared among threads which concurrently apply tableau rules until termination of the satisfiability check. In this approach we choose to investigate the or-parallelism with a shared memory strategy, where non-deterministic branches of the *ch*-Rule and the $\sqcup$-Rule are explored using parallel threads.

In order to minimize the overhead of creating and destroying threads every time a non-deterministic rule is applied, we implement the *Thread Pool* design pattern. This means that a fixed number of threads is created and organized into a queue until associated with an applicable completion rule. The number of threads can be assigned depending on the number of available processors and resource thrashing can be avoided. In this framework, threads coordinate themselves using the *Leader/Followers* design pattern where a single thread from the thread pool acts as a leader and manages thread-rule assignment. Figure 1 illustrates the state transitions between threads when adopting the Leader/Followers design pattern. When a thread is in the leader state, it can immediately change state to become in executing state if a non-deterministic completion rule becomes applicable. A thread in the executing state can run concurrently with the leader thread and other executing threads. Once a thread finishes expanding a completion graph, it either changes state to become leader, if no leader thread is available, or to become follower. In the latter case, a thread is waiting, in the thread pool, to be promoted to the leader state by the current leader. Since the threads expand a shared model, the compressed completion graph can be implemented as a *Monitor Object* to ensure synchronization between threads.

Even though the order in which expansion rules are applied does not affect soundness and completeness of the satisfiability test, in practice, results have shown that performance speedup can be achieved using certain ordering. We plan to investigate our parallel model while considering different ways of enforcing an ordering in which node labels are chosen as premise for tableau rules as was done in [11] for the basic DL $\mathcal{ALC}$.

III. Discussion

The implementation and evaluation of this parallel framework is ongoing work. The HARD (Hybrid Algebraic Reasoner for Description Logics) prototype reasoner [16], implemented in java, is being redesigned to adopt the parallel execution framework described in the previous section. Given that HARD has been equipped with a suite of crucial optimization techniques such as lazy partitioning and dependency directed backtracking, one has to consider the possible effects of the TLP. One of the features that renders non-deterministic rules appealing for parallelization, is that they result in completion graph expansions which can be explored in isolation. However, dependency directed backtracking relies on information exchange between branches such that dependencies must be recorded and consulted before pruning the search space. Such required communication between branches complicates the use of dependency directed backtracking in our parallel framework. In this context, the use of the *Thread-Specific Storage* or the *Monitor Object* seems worth investigating.

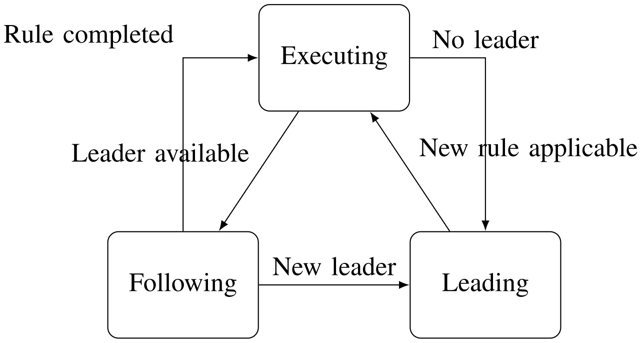


Fig. 1. Thread transitions in the Leader/Followers design pattern.

IV. Related Work and Outlook

Our work here is motivated by problems arising in the area of health services delivery. The healthcare system is composed of many different professionals operating at many sites of care offering a wide variety of services and requiring a vast amount of information both in the form of data and also in the form of clinical and other protocols. We are currently involved in a multi-year project in collaboration with our local health authority and industry partner to develop an ontology-driven Careflow Management System. Our lab has developed an ontology-driven workflow system and we have done some work in the scalability problem for querying over the OWL 2RL fragment [19], [20], [21] over large ABoxes. We are currently expanding our system with an ontology-driven service discovery engine. We believe that the high performance computing paradigm offers a great deal of hope for the scalability problem for knowledge crunching, that is, for ontological reasoning tasks, in time sensitive applications.

A parallel algorithm for description logics reasoning has been considered in 1995 [22], with limited scalability results due to hardware limitations. Further results and research activity have been reported in this area since the work presented in [8], where non-deterministic choices in core satisfiability test were explored concurrently. Parallelizing rule-based OWL inferencing has been considered in [13] by examining a data partitioning approach and a rule partitioning approach. Parallel reasoning has also been investigated in the context of distributed resolution reasoning [23] about interlinked ontologies as an alternative to centralized tableau-based reasoning (DL $\mathcal{ALCHIQ}$). Techniques using the MapReduce algorithm to classify $\mathcal{EL}^+$ ontologies [24] and fuzzy $\mathcal{EL}^+$ ontologies [25] have been proposed with no empirical evaluation. Concurrent classification of lightweight ontologies has also been considered in the context of consequence-based reasoning [26]. Tableau-based concurrent classification of more expressive ontologies has been recently reported in [9], where lock-free algorithms with limited synchronization have been used in a multi-core environment, and in [10] where specialized data structures have been proposed to optimize the use of a shared memory environment.

We plan to investigate parallel reasoning in the context of enhancing core satisfiability tests for expressive ontologies. Little work has been reported in this context. In [11], a parallel search engine (Mozart system) was used to parallelize Description Logics satisfiability check, however, the algorithm only considers basic DL $\mathcal{ALC}$. We plan to handle the expressivity of the DL $\mathcal{SHOQ}$ by designing a parallel architecture for the algebraic tableau calculus presented in [6], and which was shown to be the only one able to decide the satisfiability of complex ontologies relying on the use of nominals and qualified cardinality restrictions [16], [7]. We plan to implement and evaluate our approach in a multi-core and multi-processor environment using the Atlantic Computational Excellence Network (ACEnet) resources.

Acknowledgment

The first author is an ACEnet Postdoctoral Fellow, and gratefully acknowledges support from Atlantic Canada's HPC Consortium; the second author acknowledges support from NSERC and ACOA.

References

[1] W. MacCaull and F. Rabbi, "NOVA Workflow: A Workflow Management Tool Targeting Health Service Delivery," in *International Smposium on Foundations of Health Information Engineering and Systems (FHIES - 2011)*, ser. Lecture Notes in Computer Science, vol. 7151. Springer, 2012, pp. 75–92.

[2] V. Haarslev, M. Timmann, and R. Möller, "Combining tableaux and algebraic methods for reasoning with qualified number restrictions," in *Proceedings of the International Workshop on Description Logics (DL'2001), Aug. 1-3, Stanford, USA*, ser. CEUR Workshop Proceedings, vol. 49, 2001, pp. 152–161. [Online]. Available: citeseer.ist.psu.edu/article/haarslev01combining.html

[3] N. Farsiniamarj and V. Haarslev, "Practical reasoning with qualified number restrictions: A hybrid abox calculus for the description logic," *AI Communications*, vol. 23, no. 2-3, pp. 205–240, 2010.

[4] Y. Ding, "Tableau-based reasoning for description logics with inverse roles and number restrictions," Ph.D. dissertation, Concordia University, 2008.

[5] L. R. Pour, "Algebraïc reasoning with the description logic $\mathcal{SHIQ}$," Master's thesis, Concordia University.

[6] J. Faddoul and V. Haarslev, "Algebraic tableau reasoning for the description logic $\mathcal{SHOQ}$," *Journal of Applied Logic*, vol. 8, no. 4, pp. 334–355, 2010.

[7] ——, "Optimized algebraic tableau reasoning for the description logic $\mathcal{SHOQ}$," *Journal of Artificial Intelligence Research (JAIR) - In preparation*, 2013.

[8] T. Liebig and F. Müller, "Parallelizing tableaux-based description logic reasoning," in *Proceedings of the 2007 workshop on On the Move to Meaningful Internet Systems 2007 - OTM 2007*, 2007, pp. 1135–1144.

[9] M. Aslani and V. Haarslev, "Concurrent classification of owl ontologies - an empirical evaluation," in *Proceedings of the 2012 International Workshop on Description Logics, DL-2012*, Y. Kazakov, D. Lembo, and F. Wolter, Eds., vol. 846, Rome, Italy, June 7-10, 2012.

[10] K. Wu and V. Haarslev, "A parallel reasoner for the description logic $\mathcal{ALC}$," in *Proceedings of the 2012 International Workshop on Description Logics, DL-2012*, ser. CEUR Workshop Proceedings, Y. Kazakov, D. Lembo, and F. Wolter, Eds., vol. 846. Rome, Italy, June 7-10: CEUR-WS.org, 2012.

[11] A. Meissner, "Experimental analysis of some computation rules in a simple parallel reasoning system for the $\mathcal{ALC}$ description logic," *International Journal of Applied Mathematics and Computer Science*, vol. 21, no. 1, pp. 83–95, March 2011.

[12] M. Aslani and V. Haarslev, "Parallel tbox classification in description logics - first experimental results," in *ECAI 2010 - 19th European Conference on Artificial Intelligence*, ser. Frontiers in Artificial Intelligence and Applications, H. Coelho, R. Studer, and M. Wooldridge, Eds., vol. 215. Lisbon, Portugal, August 16-20: IOS Press, 2010, pp. 485–490.

[13] R. Soma and V. Prasanna, "Parallel inferencing for owl knowledge bases," in *37th International Conference on Parallel Processing - ICPP'08*, 2008, pp. 75–82.

[14] P. Subramanian, "A field programmable gate array based finite-domain constraint solver," Master's thesis, School of Graduate Studies, Utah State University, 2008.

[15] J. E. M. Alvarez, "Query engine for massive distributed ontologies using mapreduce," Master's thesis, Technische Universitat Hamburg-Harburg, 2010.

[16] J. Faddoul, "Reasoning algebraïcally with description logics," Ph.D. dissertation, Concordia University, Montreal, Canada, 2011.

[17] I. Horrocks and U. Sattler, "Ontology reasoning in the $\mathcal{SHOQ}$(D) description logic," in *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI 2001)*. Morgan Kaufmann, Los Altos, 2001, pp. 199–204. [Online]. Available: download/2001/ijcai01.pdf

[18] H. J. Ohlbach and J. Koehler, "Modal logics description logics and arithmetic reasoning," *Artificial Intelligence*, vol. 109, no. 1-2, pp. 1–31, 1999.

[19] F. Rabbi and W. MacCaull, "T-square: A domain specific language for rapid workflow development," in *ACM/IEEE 15th International Conference on Model Driven Engineering Languages & Systems (MODELS 2012)*, ser. Lecture Notes in Computer Science, vol. 7590, Innsbruck, Austria, September 2012, pp. 36–52.

[20] M. R. U. Faruqui and W. MacCaull, "Owlontdb: A scalable reasoning system for OWL 2 RL ontologie," in *FHIES 2012: International Symposium on Foundations of Health Information Engineering and Systems*, ser. Lecture Notes in Computer Science, vol. 7789. Springer, 2013.

[21] F. Rabbi, W. MacCaull, and M. R. U. Faruqui, "A scalable ontology reasoner via incremental materialization," in *Submitted to CBMS 2013*.

[22] F. W. Bergmann and J. J. Quantz, "Parallelizing description logics," in *19th Ann. German Conference on Artificial Intelligence*, ser. LNCS. Springer-Verlag, 1995, pp. 137–148.

[23] A. Schlicht and H. Stuckenschmidt, "Distributed resolution for expressive ontology networks," in *Web Reasoning and Rule Systems, 3rd International Conference (RR-2009)*, Chantilly, VA, USA, October 2009, pp. 87–101.

[24] R. Mutharaju, F. Maier, and P. Hitzler, "A MapReduce algorithm for el+," in *23rd International Workshop on Description Logics*, 2010, pp. 464–474.

[25] Z. Zhou, G. Qi, C. Liu, P. Hitzler, and R. Mutharaju, "Reasoning with fuzzy-$\mathcal{EL}^+$ ontologies using mapreduce," in *ECAI 2012 - 21st European Conference on Artificial Intelligence*, L. D. R. et al., Ed. IOS Press, 2012, pp. 933–934.

[26] Y. Kazakov, M. Krötzsch, and F. Simancík, "Concurrent classification of el ontologies," in *Proceedings of the 10th international conference on The semantic web*, ser. ISWC' 11. Bonn, Germany: Springer-Verlag, 2011, pp. 305–320.

A Framework for Web-based Interoperation among Business Rules

Yevgen Biletskiy
Department of Electrical and Computer Engineering
University of New Brunswick
Fredericton, Canada
biletski AT unb.ca

Abstract—**The present paper describes the approach and two technical solutions for interoperation between business rules represented in various formats. The Semantic Web techniques are used to enable this interoperation. One of the interoperation methods uses the Java Interoperation Object (JIO) described in the context of Positional-Slotted Language (POSL), which a human-friendly variant of the Rule Markup Language (RuleML), and Notation 3 (N3) representations. Details of the connections between these document representations are demonstrated with the use of query-based interoperation between POSL and N3. Another solution described in the present paper is conversion of business rules stored in Microsoft Excel as decision tables into POSL using OpenL tablets. Although the current business rules interoperation framework involves three formats (Excel, POSL, and N3), it can be extended to other document representations through appropriate conversions of data in rule bases and queries.**

Keywords— Knowledge Representation, XML, RDF, Notation 3, Positional-Slotted Language, Rule Markup Language, Semantic Web, Query

I. Introduction

Business rules are becoming ubiquitous in modern industry and are usually created, stored, and maintained by business analysts, knowledge engineers and software engineers in various formats. Some formats are technical, and some formats are non-technical and more user-friendly. Classically, business rules are logic constructs (e.g. "IF-THEN" type), and they are often represented using decision tables or decision trees. Technically, business rules can also be implemented using a programming language like C or Cobol, or by the use of a controlled English. There are many specific solutions for creation and maintenance of business rules. For instance, Microsoft Excel tables can be deployed as a user-friendly way to build documents representing decision tables. Systems like Drools provide excellent platforms to build and maintain more complex business rules. There are some standards for business rules representations. The most known is the Semantics of Business Vocabulary and Business Rules (SBVR) [1] adopted by the Object Management Group (OMG). With the wide proliferation of the Semantic Web techniques, some new languages for business rules representation appeared, for instance, Rule Markup Language (RuleML) [2], Positional-Slotted Language (POSL) [3] and Notation 3 (N3) [4]. These and other Web-based techniques can be integrated with the purpose to find better business solutions based on information stored in different rule bases accessible through Internet. The purpose of the present work is to enable semantic interoperability between business rules created in various formats.

II. The Framework

The resented approach to interoperation is based on semantic interoperability using a mediator, which can convert business rules among various knowledge representations. The software mediator can process and interpret business rules stored in various formats, as well as convert a query formulated in any of these formats to search an answer in all rule bases connected. This will assist clerks, brokers, managers, and other specialists in finding better business solutions and decision-making.

Since business rules become Web-based, the modern solutions for interoperation can be deployed. The solutions presented in this work use the Sematic Web infrastructure and related tools. The Semantic Web offers solutions allowing to semantically enriching business rules using a background ontology, which serves as a knowledge base (or vocabulary). On the other hand, the disadvantage of creating and maintaining business rules in a Semantic Web language is that rules are difficult for human understanding. Even POSL, which is more human-oriented than RuleML, is difficult for a non-specialist to understand. The focus of the present work is query-based interoperation between two Semantic Web based languages: POSL and N3, and conversion of business rules in MS Excel format into POSL. The focused interoperation framework is presented in Fig. 1.

The framework presented in Fig. 1 consists of the following main components:

1. POSL rule base, which consists of business rules and facts in the POSL format. rule base, which consists of rules and facts in the N3 format.
2. MS Excel database, which contains rules and facts in the user-friendly format.

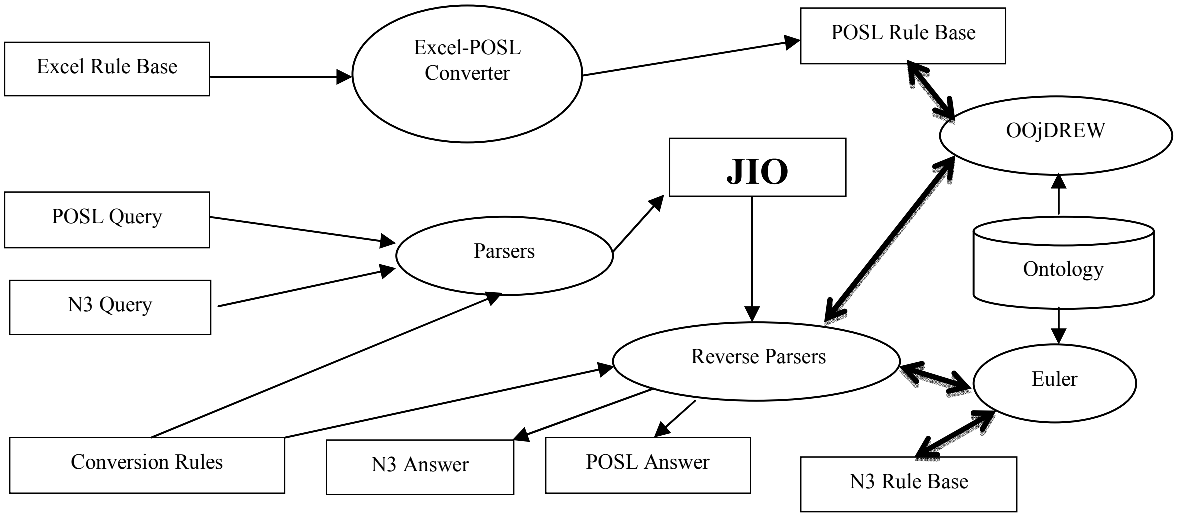


Figure 1. Focused Business Rules Interoperation Framework

3. JIO (Java Interoperation Object), which The Java Interoperation Object is the basis of interoperation methods developed for the Knowledgebase Representation Interoperation Tool (KRLIT) described technically in [5]. The objective of KLIRT is to facilitate interoperation among existing knowledge representation paradigms through a universal Java-based architecture, which used RuleML as a building block. The current KRLIT has been successfully developed and used for query-based translation between POSL and N3 knowledge bases.
4. Excel – POSL converter is presented in [6]. POSL can be generated from Excel decision tables. Before this, it is necessary to identify and extract the facts and rules contained within the tables. OpenL Tablets [7] provides an API that facilitates simple creation and processing of Java-based tables in Microsoft Excel. While OpenL itself has a rule engine capable to process these decision tables, this engine does not offer semantic enrichment with a background ontology and application-independence. As a result, OpenL is used solely for its ability to process Excel tables and externalize Java code from the application logic. The OpenL table parser uses templates to extract the relevant information (data and rules) from the decision tables into memory, where it can generate semantically rich POSL.
5. Reasoning engines: OOjDREW [8] for RuleML/POSL and Euler for N3.
6. Parsers – to syntactically analyze business rules in POSL and N3, and convert them into the JIO format.
7. Reverse parses – to convert business rules from JIO format into a format required by the user.

In the present framework, the user can query the knowledge bases using POSL or N3 formats, and receive the answer in the same format as query.

III. JIO, POSL AND N3

The technical details of JIO (Java Interoperation Object) architecture and the use of JIO within the KRLIT are explained in [5]. The Java Interoperation Object is the basis of methods for business rules interoperation. The objective of this component is to capture as many aspects of the various knowledge representation paradigms available. This concept is used to translate supported POSL and N3 rule bases. JIO uses atoms to represent chunks in a rule base (e.g. for POSL this is a relation, for N3 this is a subject).

POSL provides object-centered instance descriptions via binary properties, taxonomies over classes and properties, class-forming operations and class/property axioms, and derivation, integrity, transformation, and reaction rules [3]. POSL is a more human-readable language than the XML-based RuleML [2], but has the same language constructs. POSL has two representation paradigms which it can use, depending on what the user requires. The first of which is *positional*; this means that slots are not used to represent relation contents. The second option is *slotted*; this means that property names are associated with every element in a relation. The latter best suits our JIO framework, and so this paradigm has been chosen.

Notation 3 is a compact, rule-extended version of RDF's XML syntax [4]. In this way, RDF's complex machine understandable language becomes more readable to humans. RDF facts and rules are still written with triples (subject – property - object) and so this language is expressive in nature, but also good for human comprehension.

In order to deal with any input and fetch the answers from the available Knowledge, the system should re-present this input in order to convert it to intermediate JIO representation (RuleML building blocks) which from-and-to the system can be interoperated to the target language. The interoperation process using the JIO representation can be done by implementing *Parser* and *ReverseParser*.

The main goal of Parser is to take a query or answer of a query of a language from a File, URL or as a String in the form of *InputObjectCollection*, and then parses it (breaks down) to RuleML building blocks, which will compose a single *AtomCollection* as JIO representation in order to provide it to *ReverseParser* as input. Similarly, *ReverseParser* takes this JIO *AtomCollection* as input and reverse parses it (translates) to the target language as output with option of returning the result as an answer or query. Details of parser's implementation are presented in [5].

IV. INTEROPERATION BETWEEN POSL AND N3 BUSINESS RULES

The present work describes POSL-N3 interoperation using an example of an insurance company *Farm Insurance* and two on-line insurance brokers, which are insurance companies *Mainland Insurance* and *Healthy Life*. The companies use different knowledge representation languages, but use the same schema for their facts and rules. Assume the *Farm Insurance* has a business rules set describing automobile insurance Age-Class discounts as follows:

Age From	Age To	Customer Class	Discount Value
16	20	Economic	0.0
21	25	Economic	0.1
26	30	Economic	0.2
16	20	Gold	0.2
21	25	Gold	0.3
26	30	Gold	0.4
16	20	Platinum	0.5
21	25	Platinum	0.6
26	30	Platinum	0.7

This rule base is not accessible by individual users because it is not Web-based, but can be accessed by insurance brokers through some internal communications.

Mainland Insurance focuses primarily on *Economic*-class customers, and prefers to use an N3 knowledge base as follows (policy for providing a discount of *10%* to an *Economic* customer who is between the age of *21* and *25)*:

```
{ ?Client
                :type               :client;
                :clientID           ?ClientID;
                :age                ?Age;
                :name               [:type :fullname; :first ?FName; :last ?LName];
                :class              ?b.
      ?Age math:notLessThan      21 .
      ?Age math:notGreaterThan  25 .
      ?b    log:equalTo           :Economic. }
      =>
      {?resultDiscount
      :type           :Discount;
```

```
        :company       :MainlandInsurance;
                :clientID           ?ClientID;
                :age                ?Age;
                :class              ?b;
                :discount           0.1.}.
```

Healthy Life focuses on *Gold* and *Platinum*-class customers, and prefers to use a POSL knowledge base as follows (policy for providing a discount of *20%* to a *Gold* customer who is between the age of *16* and *20)*:

```
Discount(company->HealthyLife; clientID->?ClientID; age-> ?a:Integer;
          class-> ?b;discount-> 0.2:Real) :-
             client(clientID->?ClientID;age->?a:Integer;
                  name->fullname[
                                 first->?FName;
                                 last->?LName];
                  class-> ?b),
                    greaterThanOrEqual(?a, 16 : Integer),
                    lessThanOrEqual(?a, 20 : Integer),
                                    equal(?b, Gold).
```

Suppose the customer is familiar with POSL only, but wants to find discount policies of both insurance brokers. The query is:

```
Discount(company->?All; clientID->?clientID; age->?age;
          class-> ?b;discount->?discount).
```

If business rules interoperation is not enabled, the only *Healthy Life* database is accessible. During query processing time, this query is transformed by POSL parser into JIO, and the N3 reverse parser class accepts the transformed query as input. This provides the N3 representation of this POSL query as follows:

```
        ?subject
                   :type                :Discount;
                   :company:MainlandInsurance;
                   :clientID   ?ClientID;
                   :age                 ?Age;
                   :class               ?Class;
```

This query in POSL is given to OOjDREW, whose answers are returned in POSL. Since now the equivalent N3 query is available, it can be given to Euler as input, whose answers are given in N3. The answer is used by N3 parser and stored in JIO. It is then sent to the POSL reverse parser to generate the POSL representation of the N3 answers. This answer is combined with the OOjDREW answer resulting in the following combined POSL answer:

```
Discount(company->MainlandInsurance;
        clientID->1:Real;age->19:Real;class->Economic;discount->0.0:Real).
Discount(company->MainlandInsurance;
        clientID->6:Real;age->17:Real;class->Economic;discount->0.0:Real).
Discount(company->MainlandInsurance;
        clientID->2:Real;age->22:Real;class->Economic;discount->0.1:Real).
Discount(company->HealthyLife;
        clientID->3:Real;age->19:Real;class->Gold;discount->0.2:Real).
Discount(company->HealthyLife;
        clientID->5:Real;age->30:Real;class->Gold;discount->0.4:Real).
```

```
Discount(company->HealthyLife;
        clientID->4:Real;age->29:Real;class->Platinum;discount->0.7:Real).
```

The answer is consistent with the business rules maintained by the Farm Insurance.

V. INTEROPERATION BETWEEN MS EXCEL AND POSL USINESS RULES

Assume *Healthy Life* would like to update its rule base automatically using data from Excel sheets created by *Farm Insurance*. The business rules below need to be converted from the user-friendly Excel format into POSL:

Age From	Age To	Customer Class	Discount Value
16	20	Gold	0.2
21	25	Gold	0.3
26	30	Gold	0.4
16	20	Platinum	0.5
21	25	Platinum	0.6
26	30	Platinum	0.7

Using rule transformation templates, the table was automatically converted to POSL syntax, parsed, and loaded into the OO jDREW reasoning engine [8]. Examples POSL rules derived from the rules above are:

```
Discount(?a : Integer, ?b : Customer, 0.2 : Real) :-greaterThanOrEqual(?a, 26 :
Integer), lessThanOrEqual(?a, 30 : Integer), equal(?b, Economic : Customer).

Discount(?a : Integer, ?b : Customer, 0.2 : Real) :- greaterThanOrEqual(?a, 16 :
Integer), lessThanOrEqual(?a, 20 : Integer), equal(?b, Gold : Customer).
```

As a test, the following query was issued to OO jDREW:

```
Discount(25 : Integer, Gold : Customer, ?discount : Real).
```

The query asks "what is the discount value for a customer with age 25 and type Gold?" The results of query, issued using the OO jDREW Top-Down reasoning engine, are as follows:

```
?discount = 0.3 of type Real.
```

The solution presented allows automatically updating the rule base in POSL using rules created in Excel. A similar solution can be developed for N3. This allows business analysts to work with user friendly formats rather than to use heavily human readable Semantic Web languages.

CONCLUSION AND FUTURE WORK

The present paper has described the business rules interoperation framework as a solution to the Web-based interoperation gap issue. The work has focused on interoperation between business rules created in two different Semantic Web languages. The usage examples have been presented. The second focus of the paper is a methodology to partially automate the process of converting human-readable business rules stored in the form of MS Excel tables to machine-processible POSL, with the goal of combining the ease of use of Excel-based rule tables with the semantically-rich queries supported by reasoning engines. Although the work in current state covers Excel, POSL and N3 formats only, it can extend to other business rules representations.

ACKNOWLEDGMENT

The present work includes business rules interoperation techniques which had been implemented as part of students' projects conducted at the University of New Brunswick under supervision of the author. The students were Girish R Ranganathan (Dr. Ranganathan now), J Anthony Brown, Taylor Osmun, and Patrick Thébeau. The main funding sources for this work are NSERC Discovery Grant and ACOA AIF.

REFERENCES

[1] SBVR. Available: http://www.omg.org/spec/SBVR/1.0/

[2] H. Boley, The RuleML Family of Web Rule Languages, Invited Talk. *Proc. Fourth Workshop on Principles and Practice of Semantic Web Reasoning,* Budva, Montenegro, LNCS 4187, Springer-Verlag (2006) 1-15.

[3] H. Boley, POSL: An Integrated Positional-Slotted Language for Semantic Web Knowledge (2004). Available: http://www.ruleml.org/submission/ruleml-shortation.html.

[4] T. Berners-Lee et. al. Notation (N3), A readable RDF Syntax. Available: http://www.w3.org/TeamSubmission/n3/

[5] T. M. Osmun, P. Thébeau, Y. Biletskiy. Knowledgebase Representation Language Interoperation Tool. *In Proc RuleML America*, LNCS 7018, Springer-Verlag (2011) 58-65.

[6] Y. Biletskiy, G. R. Ranganathan, J. A. Brown. Representing User-Friendly Business Rules in a Semantic Web-Based Format. *ISAST Transactions on Computers and Software Engineering* 2(1) (2008) 8-12.

[7] OpenL Tablets, Available: http://openl-tablets.sourceforge.net/index.html

[8] Ball M., Boley H., Hirtle D., Mei J., and Spencer B. The OO jDREW Reference Implementation of RuleML. *In Proc. Rules and Rule Markup Languages for the Semantic Web (RuleML-2005)*, LNCS 3791, Springer-Verlag (2005) 218–223.

[9] Euler. Available: http://eulersharp.sourceforge.net/

.

Part IV.

Early Career Track Papers

Product Centric Web Page Segmentation and Localization

John Cuzzola
Ryerson University
350 Victoria St
Toronto, ON M5B 2K3
Canada
jcuzzola@ryerson.ca

Dragan Gašević
Athabasca University
1 University Drive
Athabasca, AB T9S 3A3
Canada
dgasevic@acm.org

Ebrahim Bagheri
Ryerson University
350 Victoria St
Toronto, ON M5B 2K3
Canada
bagheri@ryerson.ca

ABSTRACT

The Internet is home to an ever increasing array of goods and services available to the general consumer. These products are often discovered through search engines whose focus is on document retrieval rather than product procurement. The demand for details of specific products as opposed to just documents containing such information has resulted in an influx of product collection databases, deal aggregation services, mobile apps, twitter feeds and other just-in-time methods for rapid finding, indexing, and notifying shoppers to sale events. This has led to our development of intelligent Web crawler technology aimed towards this specific category of information retrieval. In this paper, we demonstrate our solution for Web page categorization, segmentation and localization for identifying Web pages with shopping deals and automatically extracting specifics from the identified Web pages. Our work is supported with empirical data of its effectiveness. A screencast demonstration is also available online at http://youtu.be/HHPme6AJuCk.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval - *Information filtering, retrieval models, search process, selection process*. I.2.7 [**Artificial Intelligence**]: Natural Language Processing - *text analysis*.

General Terms

Algorithms, Experimentation.

Keywords

Natural language processing, search, classification, segmentation, localization, deals, products, web crawling

1. INTRODUCTION

The World Wide Web has given rise to a digital marketplace where goods and services of all varieties are sold. Retailers, wholesalers, and private individuals are using this communication medium to advertise their products directly to the consumer. Conversely, consumers are looking for these products and are using the traditional search engine as the method for discovery. However, these engines are document-centric rather than product-centric; hence they are optimized for the former rather than the latter. A successful search engine relies on its web crawlers to intelligently process visited Web pages for useful information while discarding data that does not contribute to retrieval. Geared specifically to this domain of product search, we have created technology that can identify product Web pages, segment Web pages into logical regions, and discard those regions that do not contain information regarding a specific goods or service. The remainder of this paper explains our Web page classification, segmentation and deal localization technology.

2. BACKGROUND

Our work reported in this paper was inspired by the needs of our industrial partner, SideBuy Technologies, which is a daily deal aggregator; a service which collects for-purchase goods and services from various deal sites such as Groupon, PriceGrabber and others. The process of collecting and aggregating these deal information is performed manually where large numbers of staff are employed as deal seekers [5]. Deal aggregators commonly deploy web scraping tools targeted at deal sites to harvest these deals. However, the collection process usually is dependent on pre-programmed recognized patterns specific to the site being scraped, e.g., using specific sequence of HTML tags. Consequently, even small modifications in such Websites will require programming changes in scraping tools to accommodate these changes. Furthermore, this targeted pattern matching approach does not scale to the unstructured and ever-changing content of the Web where many products are being sold but remain unnoticed and out-of-reach from the scrapers. Finally, the time sensitive nature of these deals further fuels the desire to leverage a more automated solution to the deal discovery dilemma.

To this end, we have developed algorithms to allow Web crawlers to identify unstructured, previously unseen, Web pages as containing information regarding relevant online deals. Once a page is classified as containing relevant information, our algorithms can segment and localize the regions of the Web page that contain product information, while discarding those areas that are not of interest.

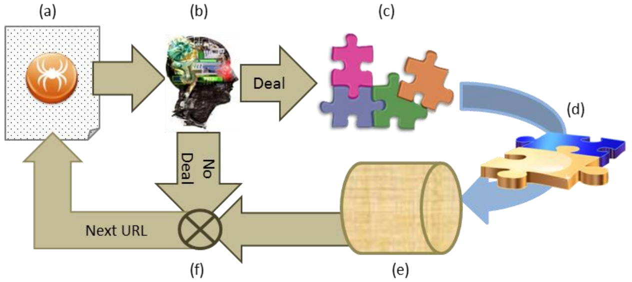


Figure 1: Technology pipeline: (a) Web crawler (b) Deal classifier (c) Page Segmentation (d) Localization (e) Storage

3. SYSTEM PIPELINE

Our process of information extraction from unstructured Web content is summarized in Figure 1. A Web crawler scrapes a given page for its HTML content (a). A binary classifier then determines whether the text of the page contains products for purchase (deal) or no such offerings exists on that page (no-deal). Those pages classified as not containing products (no-deal) are discarded (f) while those pages categorized as deal undergo segmentation resulting in several segments per page (c). Each of the extracted segments will in turn be recursively classified as either containing deal or no-deal information in their own respect in an effort to localize individual products (d). Further processing on the deal segments involve semantic annotation, pattern matching, and image recognition that would extract property/value pairs, which are ultimately stored in a central repository (e).

3.1 Binary Classifier

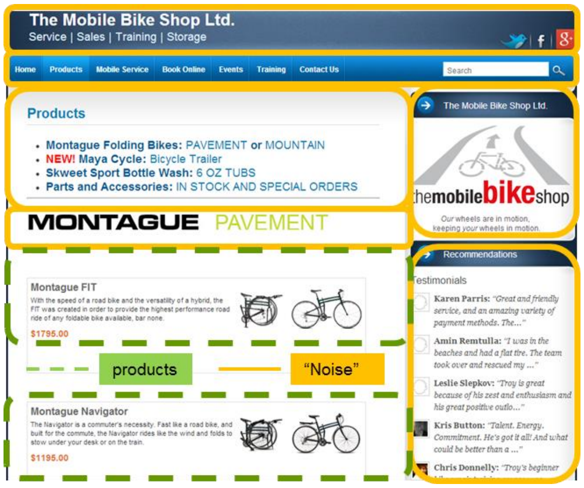
We have developed a binary classifier capable of classifying a text/html fragment as either containing relevant products (deals) information or being void of such information (no-deal). The classifier is a hybrid Naive Bayes/Expectation-Maximization model trained using the WEKA machine learning framework [4]. We use the OpenNLP toolkit to incorporate named entity recognition for dates, organizational entities, time, location, percentages, money, and people. Part of speech tagging is combined with the WordNet lexical database to disambiguate word sense forms [3]. This information is used as features within our training dataset. The classifier is trained on information already manually extracted using SideBuy Technologies' deal scrapers. The detail of our classifier is available in [1].

```
<div class>
|________<div style>
         |________<p>
                  The X7 Smartphone
                  features a/b/g/n WiFi.
<div class>
|________<div style>
         |________<blockquote>
                  The model S2 tablet
                  comes with 4-GB RAM.
```

Listing 1: A sample recurring pattern in HTML.

3.2 Segmentation

Web page segmentation is the process of partitioning a Web page into logically grouped sections either visually, structurally, or semantically to form cohesive subsets of the Web page. As already reported by various researchers [6,7,8], ecommerce Websites often use a recurring pattern to represent product information. Therefore, each of the product information sets is represented under its own Web segment within the page. Besides the product segments on the page, there may be other segments such as banners, Web page footers, and others that are not relevant to product retrieval and search and can hence be discarded for our purpose (see Figure 2). We base our work on this observation and develop a Web page partitioning algorithm that processes Web page HTML contents and extracts all possible Web segments from that page.



- Montague Folding Bikes: PAVEMENT or MOUNTAIN
- NEW! Maya Cycle: Bicycle Trailer
- Skweet Sport Bottle Wash: 6 OZ TUBS
- Parts and Accessories: IN STOCK AND SPECIAL ORDERS

Our wheels are in motion, keeping your wheels in motion.

With the speed of a road bike and the versatility of a hybrid, the FIT was created in order to provide the highest performance road ride of any foldable bike available, bar none.

The Navigator is a commuter's necessity. Fast like a road bike, and built for the commute, the Navigator rides like the wind and folds to stow under your desk or on the train.

Karen Parris: "Great and friendly service, and an amazing variety of payment methods. The..."

Amin Remtulla: "I was in the beaches and had a flat tire. The team took over and rescued my ..."

Leslie Slepkov: "Troy is great because of his zest and enthusiasm and his great positive outlo..."

Kris Button: "Talent. Energy. Commitment. He's got it all! And what could be better than a ..."

Chris Donnelly: "Troy's beginner

Figure 2: A segmented page. Product blocks in green dashed. "Noisy" blocks include header/footer, navigation bar, company logo, customer tweets in yellow solid.

Our system segments web pages based on HTML structure and textual clues obtained from natural language processing. Segmentation of a Web page is accomplished by finding the Longest Frequent Pattern (LFP) [2] of HTML tags at the topmost (outermost) block level. The identified LFP becomes the boundary of division for each partition in the Web page. For example, consider the sequence of nested HTML tags and textual content in Listing 1.

The topmost longest frequent pattern occurs twice with <div class>,<div style> resulting in two segments with fragments of "<p> *the X7 Smartphone feature a/b/g/n WiFi*" and "<blockquote> *the model S2 tablet comes with 4-GB RAM*". The result of this segmentation process is the localization of individual product offerings within each page in such a way that each individual segment will either contain individual product specifications such as name, description, and price or will represent non-product information in which case the segment is of no interest to us.

```
1. Let C be a set of candidate blocks of a web page.
    1.1 Initialize C with the outermost block.
     (Typically C⟵<HTML>…</HTML>)
2. For each block in C, classify block as either deal or no-deal using the binary classifier. Separate blocks into a deal set (η) or non-deal set.
     2.1 for each block $f \in \eta$
          2.1.1 Find the longest frequent HTML pattern
                (LFP) of sentence block $f$.
         2.1.2 If (LFP) exists:
                    2.1.2.1 Split $f$ in blocks on (LFP) $\rightarrow \beta$
                    2.1.2.2 Add split blocks to C: $C \leftarrow C + \beta$
3. Goto Step 2 if C is non-empty
```

Algorithm 1: The Segmentation-Localization algorithm

3.3 Localization

Once Web segments have been extracted from a Web page, we perform localization on each of these segments. Localization is the process of determining which of these extracted segments contain useful and relevant product information such as the green dashed boxes in Figure 2 and also identifying those segments that contain non-relevant information and can be discarded such as the solid yellow boxes in Figure 2. In order to be able to efficiently perform the location process, we employ the same classifier that was introduced in Section 3.1. The classifier will now be used to determine whether each segment on their own would be classified as containing product-specific information or not. Therefore, the difference between the first step and the localization step would be that in the first step the classifier is used to determine whether the whole page contains product information, while in the localization step an individual segment within an already positively classified page is tested for containing product-specific information. Here, rather than evaluating the text of the entire page, only the text within this candidate segment is considered. If this block is positively classified, it is split recursively into smaller segments using the segmentation approach of Section 3.2. This process repeats iteratively for each newly segmented block until either the new block is negatively labeled, or a frequent pattern of HTML tags cannot be found. This process is illustrated in Figure 3 and can be visually summarized in a *segmentation parse tree* which is constructed by our implementation shown in Figure 4. The leaves of the segmentation parse tree represent the final outcome where each leaf node is either a segment of non-interest (negatively classified) or a segment containing a single product offering (positively classified localized segment). The localization algorithm is formally defined in Algorithm 1.

4. EVALUATION

Initial testing of our segmentation and deal localization algorithm involved 42 individual Web pages each from different Web sites. This set gave us a total of 1,402 individual products. The criteria used in the determination whether the final outcome was successful were as follows.

Criteria 1: A block is correctly classified if and only if the block makes reference to exactly one product offering. If the block contains information for more than a single product then it was under-partitioned and should have undergone further segmentation in order to split its contents into individual products.

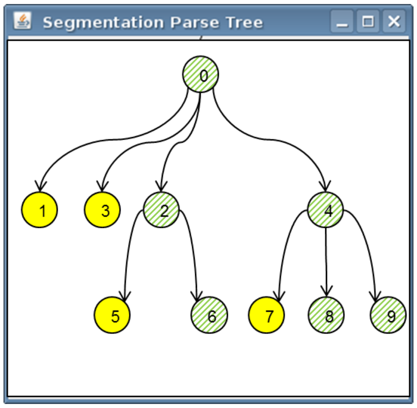


Figure 4: Tree representation of Figure 3. Segments 6, 8, and 9 contain individual product offerings (relevant).

Criteria 2: Because the descriptiveness of a product will vary significantly between websites; the minimum amount of information necessary is the name of the product and its price. Blocks that do not meet this minimum were considered to be over-partitioned.

Criteria 3: A leaf node that satisfies Criteria 1 and 2 but makes reference to the same product will only get credit for correctly classifying the product once.

With the above criteria in place, our system performed favorably with an average F-score of 0.903. The algorithm correctly identified 1,282 products with 154 misclassifications (false positives). A summary of the results is given in Table 1 sorted by best F-score. The relatively poor F-score's of the bottom 5 web pages appeared to be related to either the structure of the web page in which frequent patterns were difficult to find or the content of the page itself where the classifier mislabeled the segmented region as a non-deal area.

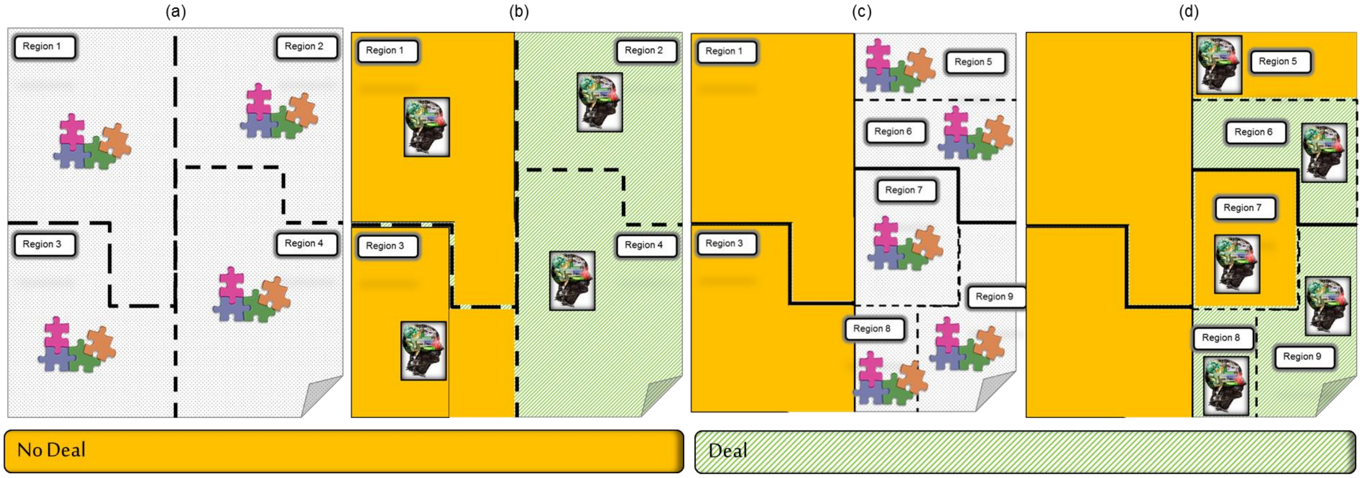


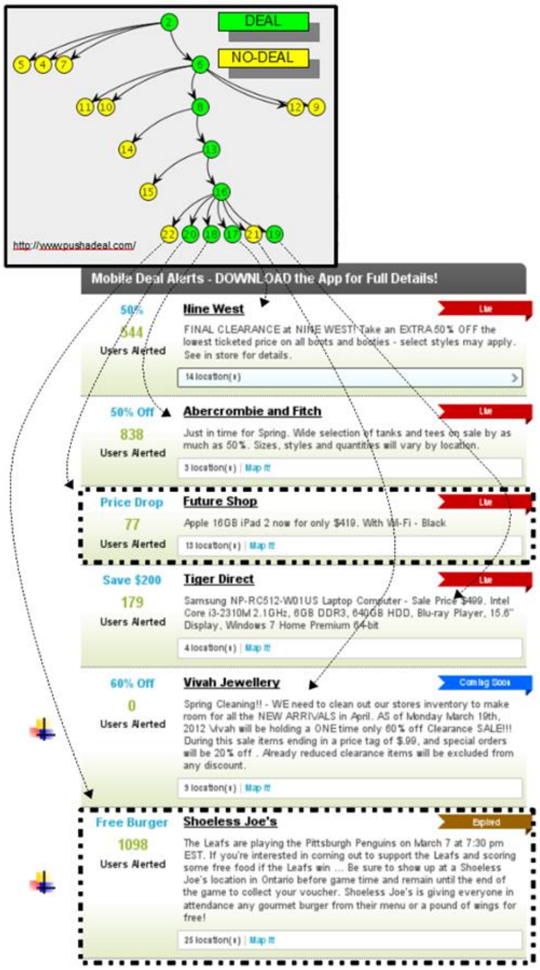
Figure 3: Segmentation/Localization illustrated. (a) Segmentation is performed on the entire web page using Longest Frequent Pattern (LFP). (b) Binary classifier labels each segment as either relevant (dashed green) or non-relevant (solid yellow). (c) Relevant segments are further partitioned using LFP. (d) The classifier labels partitions

Table 1: segmentation/localization evaluation results.

SITE	ACTUAL DEALS	FOUND DEALS	RIGHT	WRONG	PREC.	RE-CALL	F-SCORE
dailynews.com	8	8	8	0	1	1	1
trackdailydeals.com	13	13	13	0	1	1	1
deals.com	25	25	25	0	1	1	1
dealextreme.com	52	53	52	1	0.981	1	0.99
mydealbag.com	246	238	238	0	1	0.967	0.983
↖ TOP 5 ↗							
(32 sites aggregated)	950	970	869	101	0.896	0.915	0.905
↙ BOTTOM 5 ↘							
sidebuy.com	9	17	9	8	0.529	1	0.692
music123.com	20	13	11	2	0.846	0.55	0.667
dealfrenzy.com	5	10	5	5	0.5	1	0.667
elivedeals.com	53	64	39	25	0.609	0.736	0.667
rubywallet.com	21	25	13	12	0.52	0.619	0.565
TOTALS:	1402	1436	1282	154	0.893	0.914	0.903

5. DEMONSTRATION

Our segmentation/localization system was tested on a Web page from a deal aggregator's website: *pushadeal.com*. The output of the analysis is shown in Figure 5. Our intelligent crawler correctly identified the HTML pattern that encompasses individual products on this Web page.



Mobile Deal Alerts - DOWNLOAD the App for Full Details!

50% 544 Users Alerted **Nine West** Lie
FINAL CLEARANCE at NINE WEST! Take an EXTRA 50% OFF the lowest ticketed price on all boots and booties - select styles may apply. See in store for details.
14 location(s)

50% Off 838 Users Alerted **Abercrombie and Fitch** Lie
Just in time for Spring. Wide selection of tanks and tees on sale by as much as 50%. Sizes, styles and quantities will vary by location.
3 location(s) | Map it!

Price Drop 77 Users Alerted **Future Shop** Lie
Apple 16GB iPad 2 now for only $419. With Wi-Fi - Black
13 location(s) | Map it!

Save $200 179 Users Alerted **Tiger Direct** Lie
Samsung NP-RC512-W01US Laptop Computer - Sale Price $499. Intel Core i3-2310M 2.1GHz, 6GB DDR3, 640GB HDD, Blu-ray Player, 15.6" Display, Windows 7 Home Premium 64-bit
4 location(s) | Map it!

60% Off 0 Users Alerted **Vivah Jewellery** Coming Soon
Spring Cleaning!! - WE need to clean out our stores inventory to make room for all the NEW ARRIVALS in April. AS of Monday March 19th, 2012 Vivah will be holding a ONE time only 60% off Clearance SALE!!! During this sale items ending in a price tag of $.99, and special orders will be 20% off . Already reduced clearance items will be excluded from any discount.
5 location(s) | Map it!

Free Burger 1098 Users Alerted **Shoeless Joe's** Expired
The Leafs are playing the Pittsburgh Penguins on March 7 at 7:30 pm EST. If you're interested in coming out to support the Leafs and scoring some free food if the Leafs win ... Be sure to show up at a Shoeless Joe's location in Ontario before game time and remain until the end of the game to collect your voucher. Shoeless Joe's is giving everyone in attendance any gourmet burger from their menu or a pound of wings for free!
25 location(s) | Map it!

Figure 5: A segmented and deal-localized Web page.

The leaves of the generated segmentation parse tree reveal two potential product offerings that were classified as non-relevant (✚). By looking closely at the content of the page, one can see that this was correct since one product offer had "expired" while the other was "coming soon" and therefore not yet available. A further illustration of our system is available as a screencast at: http://youtu.be/HHPme6AJuCk. Also, visit the inextweb showcase section at http://inextweb.com which demonstrates how a database of localized segments are being utilized to provide an object-centered search engine over the familiar document centric engines of Google, Bing, and others.

6. CONCLUSION

This paper demonstrates our approach to Web page classification, segmentation and localization specific to the domain of goods and services procurement. We describe an intelligent Web crawler implementation that sees Web pages as containing product information. Our technology can be used to build a collection of properly annotated product objects, which can be leveraged for smarter search in the domain of e-commerce. In our demonstration we will showcase the described technology as follows 1) We will demonstrate how our machine learning and page segmentation techniques were trained and built; 2) We will introduce and provide open access to the wrapper API of our technology that is able to extract product information segments from Web pages; 3) We will show how to use our API to quickly write an application that would crawl a given website and extract product segments. An online demo is available at:

http://ls3.rnet.ryerson.ca:8086/DealExtractorSampleJavaClient/sampleform.html

7. ACKNOWLEDGMENTS

The authors would like to thank The National Science and Research Council of Canada (NSERC) and SideBuy Technologies Inc. for their funding support.

8. REFERENCES

[1] Cuzzola, J., Gašević, D., Bagheri, E., "What's the Deal? – Identifying Online Bargains," In Proceedings of the 2013 Australasian Web Conference (AWC 2013), Adelaide, Australia, 2013.

[2] J. Kang, J. Yang, J. Choi, "Repetition-based Web Page Segmentation by Detecting Tag Patterns for Small-Screen Devices", IEEE Transactions on Consumer Electronics, vol. 56, no. 2, pp. 980-986, 2010.

[3] Miller, G. "WordNet: A Lexical Database for English". Communications of the ACM 38(11): 39-41, 1995.

[4] Hall, M. Eibe, F., Holmes, G. Pfahringer, B., Reutemann, P. Witten, I. The WEKA Data Mining Software: An Update. SIGKDD Explorations, 11(1): 2009.

[5] Ghigliotty, D. "Do You Really Want a Job at Groupon?" Retrieved from http://salesjobs.fins.com/Articles/SBB0001424052970204528204577012073472414832/Do-You-ReallyWant-a-Job-at-Groupon, 2011.

[6] Chakrabarti, D.,Kumar, R., Punera,K. Page-level template detection via isotonic smoothing. In Proceedings of the 16th international conference on World Wide Web (WWW '07). ACM, New York, NY, USA, 61-70, 2007.

[7] Kao, H., Ho, J., Chen, M. WISDOM: Web Intrapage Informative Structure Mining Based on Document Object Model. IEEE TKDE 17 (5): 614-627, 2005.

[8] Chakrabarti, D., Kumar, R., Punera, K. A graph-theoretic approach to webpage segmentation, International conference on World Wide Web, pp 377-386., 2008.

Generating Semantic Web Services from Declarative Descriptions

Mohammad Sadnan Al Manir, Christopher J.O. Baker
Department of Computer Science and Applied Statistics
University of New Brunswick
Saint John, Canada
{*sadnan.almanir,bakerc*}*[at]unb.ca*

Alexandre Riazanov
IPSNP Computing Inc, Canada
alexandre.riazanov[at]ipsnp.com

Harold Boley
Faculty of Computer Science
University of New Brunswick
Fredericton, Canada
harold.boley[at]ruleml.org

***Abstract*—Semantic Web services are an effective middleware for semantic querying of relational databases. Despite the benefits of this approach, writing Web service code manually is labor-intensive and error-prone. To ameliorate this, we propose a framework to generate SADI web services from declarative service descriptions in which access to databases is achieved through semantic mappings. These mappings are scripted in the Datalog sublanguage of Positional-Slotted Object-Applicative (PSOA) RuleML. We outline a novel methodology, a system architecture, and an early stage implementation for service generation. We demonstrate the utility of this approach in a use case for querying patient data from a hospital data warehouse.**

I. Introduction

Semantic Querying (SQ) is based on the automatic application of domain knowledge formalized as ontologies and rules, which semantically capture the underlying database design. An explicit semantic correspondence between the database schema and relevant domain ontologies is established by rules. Each *domain ontology* constitutes a high-level model in the form of logical axioms using RDF(S)[1,2] and OWL[3], which allows domain experts to pose queries in a semantic context that they are familiar with.

Existing SQ systems such as D2RQ [1], MASTRO [2], Incremental Query Rewriting (IQR) [3] typically allow database programmers to define mappings between relational schemas and domain knowledge bases, in the form of logical axioms or similar declarative constructs. These systems then translate the domain-based queries into SQL queries that can be directly executed on the data.

In recent work, HAIKU [4], [5] considers a different approach based on the deployment of Semantic Web Services on top of relational databases. This approach relies on suitable mappings written by the database programmers, allowing SADI [6] framework-based Semantic Web services to extract Hospital-Acquired Infections (HAI) data from The Ottawa Hospital (TOH) Data Warehouse (DW) [7]. One limitation of this approach is that service creation becomes labor-intensive and can be error-prone, because it requires writing code for the service. This motivated us to investigate if code generation could be automated from declarative service descriptions, specifically by incorporating the necessary input and output parameters in appropriate places of generic Web service code-blocks.

In this paper, an architecture based on SQ is presented and its implementation is outlined with the goal of generating SADI Semantic Web service code automatically from their declarative input and output descriptions. The architecture enables access to relational data via the expressive rule language PSOA RuleML [8]. The automation facilitates Web service generation without human intervention and users are able to run queries over the generated services with the help of SADI query clients like Hydra and SHARE (see, e.g., [5]).

The methodology and architecture are novel: we are not aware of another system that allows the creation of Semantic Web services on top of relational data by leveraging an expressive rule language for semantic mapping and a first-order logic reasoner for query rewriting.

The paper is organized as follows: we start with a brief description of SADI in Section II. A use case for service generation is shown in Section III and the work flow of our architecture is described in Section IV. Finally, in Section V some of the implementation challenges and an evaluation of the methodology are briefly discussed.

II. Preliminaries

A. Basic SADI Ideas

SADI [6] is a framework which utilizes Semantic Web standards and allows integration and interoperability among resources on the Web. SADI uses RDF[S], OWL for data representation and modeling, and HTTP-based recommendations (GET, POST) for interacting with the services.

The main distinguishing features of SADI services is that (1) they only exchange RDF, so “they speak the same language” (2) they can be automatically discovered and (3) orchestrated with the help of query clients like Hydra and SHARE (see, e.g., [5]).

III. Early-Stage Implementation

To show the advantages of our architecture, we walk through the generation of a simple SADI service that can

[1] http://www.w3.org/TR/rdf-concepts/

[2] http://www.w3.org/TR/rdf-schema/

[3] http://www.w3.org/TR/owl-overview/

query a database and retrieve results.

Our early-stage experiment comprises of a database schema, a corresponding domain ontology, service I/O descriptions modeled according to the ontology, and an SQL template generated from inputs to the reasoner.

A schema named PatientDiseaseDB is shown in Fig. 1. A relevant ontology describing the same domain is presented next.

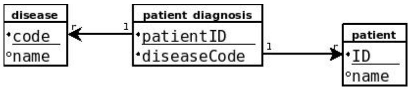


Figure 1. Database of patients and their diagnosed diseases

```
@prefix servOnt: <http://unbsj.biordf.net/servOnto.owl#>.
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

servOnt:Disease a owl:Class.
servOnt:Patient a owl:Class.

servOnt:hasDisease a owl:ObjectProperty.
servOnt:isDiseaseOf a owl:ObjectProperty;
 owl:inverseOf servOnt:hasDisease.

servOnt:name a owl:DatatypeProperty;
 rdfs:range xsd:string.

servOnt:getPatientNameByDiseaseName_Input a owl:Class;
 owl:equivalentClass [a owl:Class;
   owl:intersectionOf (servOnt:Disease [a owl:Restriction
     owl:onProperty servOnt:name;
     owl:someValuesFrom xsd:string])].

servOnt:getPatientNameByDiseaseName_Output a owl:Class;
 rdfs:subClassOf [a owl:Restriction;
  owl:onProperty servOnt:isDiseaseOf;
  owl:someValuesFrom [a owl:Class;
   owl:intersectionOf (servOnt:Patient [a owl:Restriction;
    owl:onProperty servOnt:name;
    owl:someValuesFrom xsd:string])]].
```

Here we describe a simple SADI service getPatientNameByDiseaseName which, upon receiving the name of a disease as input, provides the corresponding patients' names.

The input class is defined by a disease name with the *name* data property attached to Disease class which is expressed in Protégé syntax as Disease and *name* some string.

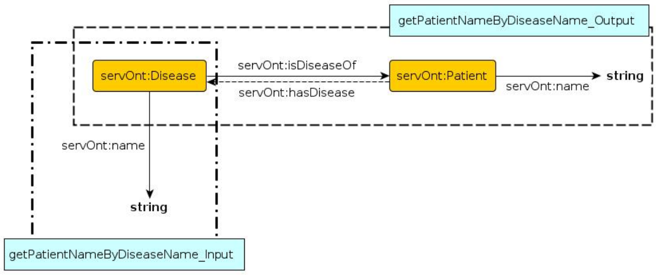


Figure 2. A Simple SADI Service getPatientNameByDiseaseName

The output class is defined by the patient names with the *name* property attached to the class Patient, which is attached to the Disease class by the *isDiseaseOf* property and is expressed as *isDiseaseOf* some (Patient and *name* some string).

Fig. 2 depicts the modeling of both the input and the output classes. The root node for both the classes is Disease. The solid arrows are labeled by the object property *isDiseaseOf* and by the single string-type data property *name*. Although the inverse property *hasDisease* is defined in the ontology, it is not part of the declarative descriptions, and denoted only by a dotted arrow.

Our reasoner, VampirePrime[4] uses TPTP [9] as its primary input syntax. Hence, a translation is necessary to transform any non-TPTP syntax for generating SQL. This is accomplished by incorporating three translators into the architecture. The semantic mappings expressed in PSOA in Section IV-B can be translated into TPTP by using open-source tools such as the PSOA RuleML API [10] and PSOA2TPTP [11] (part of PSOATransRun[5]). The declarative input and output descriptions and the ontology are translated by the OWL API [12].

For example, the input and output declarative descriptions are translated by the OWL API-based translator into a single TPTP rule below ('- -' labels conditions while '++' labels the conclusion, X, N, D are variables):

```
--p_Patient(X),--p_name(X, N),--p_isDiseaseOf(D, X)
, --p_Disease(D), --p_name(D, "?"), ++answer(N)
```

The tuple N in the unary predicate *answer* denotes the patient tuple X's names who have the disease tuple D with a name "?", which is like a formal parameter and its actual values come from actual service inputs in run time. This rule is created by merging the input and output class based on the SADI principle that both the input and output class have a common root node.

VampirePrime generates the following SQL query template from the semantic mapping, ontology and the TPTP rule. Although in this specific case no reasoning is necessary, potentially VampirePrime can do very complex reasoning to rewrite queries. The template query contains the WHERE clause with the condition disease.name = "?", where the symbol '?' is extracted from the TPTP predicate p_name(D, "?") above.

```
SELECT patient.name AS patName
FROM patient, disease, patientdiagnosis
WHERE  disease.name = "?"
AND patient.id = patientdiagnosis.patient_id
AND disease.code = patientdiagnosis.code
```

Due to space constraints, we refrain from documenting the complete Java code for the SADI Web service.

One of the most important tasks of our system is to extract the inputs from an RDF input instance and to place them

[4] http://riazanov.webs.com/software.htm, last accessed on June 14, 2013
[5] http://wiki.ruleml.org/index.php/PSOA_RuleML

precisely where they are needed. Once invoked, the Web service determines the string-type input value Arthritis, extracts it from the RDF input and replaces '?' with Arthritis in the WHERE clause, making the instantiated SQL query executable over the database:

```
SELECT patient.name AS patName
FROM patient, disease, patientdiagnosis
WHERE  disease.name = "Arthritis"
AND patient.id = patientdiagnosis.patient_id
AND disease.code = patientdiagnosis.code
```

After the call and execution, the service returns the output RDF file containing a list of patient names John Doe, Bob, Alice etc. having Arthritis, each extracted from the tables in Fig. 1.

The following figure shows a graphical representation of the above RDF input and output instances:

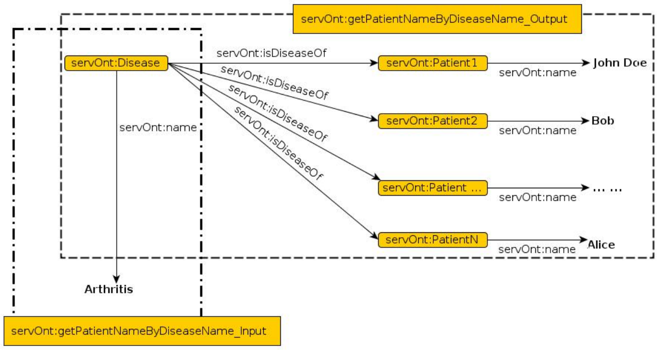


Figure 3. Service Input and Output Instances

The generated SADI service can be invoked and tested by issuing a simple SPARQL query in SADI query clients such as Hydra and SHARE. User asking *Which patients have Arthritis*?, issues the following SPARQL query:

```
PREFIX servOnt: <http://unbsj.biordf.net/servOnto.owl#>
SELECT DISTINCT ?patientName
WHERE
{ servOnt:Patient servOnt:name ?patientName.
  servOnt:Disease servOnt:isDiseaseOf servOnt:Patient.
  servOnt:Disease servOnt:name "Arthritis". }
```

IV. Architecture

The Web service generation process is best described by the main components (modules) of the architecture shown in Fig. 4.

A. Module for Declarative Descriptions of the Service

Declarative service descriptions are composed of the properties along with the class names and various logical connectives from the ontology(ies) as shown by the input and output classes in Section II using Protégé syntax.

B. Module for Semantic Mapping of Databases in PSOA

This module provides mappings between ontologies and databases using the Datalog sublanguage of the expressive Web rule language PSOA RuleML. The SQL queries and pseudo-RDF indicate how relational data is mapped.

The PSOA rule below embodies the semantic mapping of HAI-related data from TOH DW. Lines 12-18 essentially represent the SQL query while lines 1-9 and 19-23 capture the meaning of the pseudo-RDF. The relations among SQL queries and pseudo-RDF with these rules are exemplified in [4], [5] in detail.

```
1 And
2 (
3   ?diagnosis # haio:Diagnosis()
4   haio:is_performed_for(?diagnosis ?patient)
5   haio:identifies(?diagnosis ?disease)
6   ?disease # haio:Disease()
7   ?disease # ?diseaseClass()
8 )
9   :-
10 And
11 (
12  ?encounterRow #
13     dwt:Nencounter(dwa:encWID->?encounterID
14                    dwa:encPatWID->?patientID)

15  ?diagnosisRow #
16     dwt:NhrDiagnosis(dwa:hdgWID->?diagnosisID
17                        dwa:hdgHraEncWID->?encounterID
18                        dwa:hdgCd->?diseaseCode)
19  ?patient = External(modf:Patient_by_patWID(?patientID))
20  ?diagnosis = External(modf:Diagnosis_by_hdgWID(?diagnosisID))
21  ?diseaseClass
22         = External(modf:disease_class_by_ICD10(?diseaseCode))
23  ?disease = External(modf:Disease_by_diagnosis(?diagnosisID))
24 )
```

C. SQL Query Template Generation Module

The generation of SQL queries requires declarative service descriptions, semantic mapping of the database, and ontology (semantic schema) as the inputs. Our architecture will be using the IQR technique because it facilitates such SQL generation. The IQR technique takes the inputs and generates a (possibly infinite) number of SQL queries.

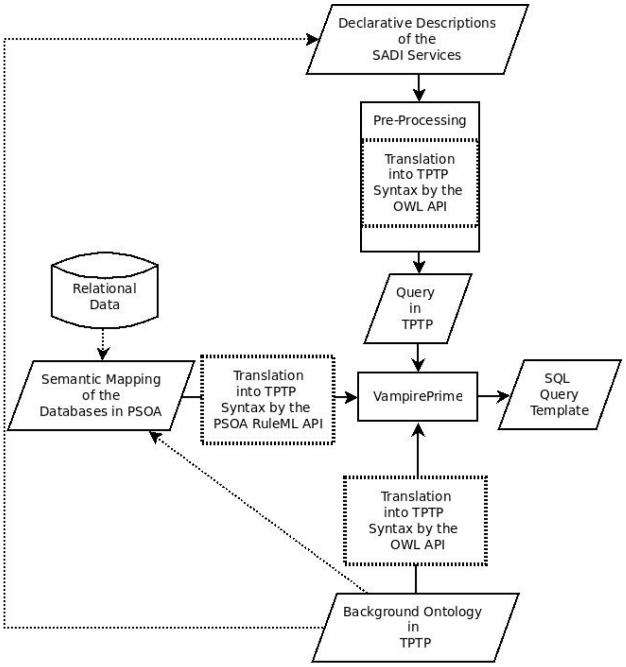

Declarative Descriptions of the SADI Services
Pre-Processing
Translation into TPTP Syntax by the OWL API
Query in TPTP
Relational Data
Semantic Mapping of the Databases in PSOA
Translation into TPTP Syntax by the PSOA RuleML API
VampirePrime
SQL Query Template
Translation into TPTP Syntax by the OWL API
Background Ontology in TPTP

Figure 4. Architecture

D. Service Generator Module

The service generator module generates Java code for the Web service. The service code consists of three parts: reading input RDF, business logic and writing RDF output. The code for reading input and writing output are generated based on the input and output class definitions, respectively. The service code when executed, reads input RDF and places all input values in appropriate places of the generated code as well as in the generated SQL template. Finally, the data drawn from the database is presented as RDF output according to the modeling of the output class. Thus, the module ensures automatic generation of a fully functional Web service code with no human intervention.

V. Discussion, Ongoing Work and Evaluation

The open-source D2RQ platform uses a declarative language and employs a tool called D2R server, which uses a customizable D2RQ mapping to map database contents into RDF and allows users to issue SPARQL queries which are rewritten into SQL queries via the mapping. MASTRO is an ontology-based data integration tool. The mapping language in MASTRO allows for expressing Global-As-View mappings, answers unions of conjunctive queries, and it provides a sound and complete query answering algorithm for a rather restricted logic fragment. For our work, we plan to adopt the IQR technique which is based on a sound and complete algorithm that works with a full first-order logic, but without a general termination guarantee and rewrites TPTP queries into SQL queries. Results from initial experiments show that simple SQL queries can be generated without problem. We plan to address complex query generation, case-by-case, in future.

Unlike D2RQ which exposes the database as a virtual RDF graph, in our approach, semantic mappings are written to map the existing ontology and the relational database. Any changes occurring in the database schema must be reflected in the mappings and such modifications are to be written by the database programmers. The mappings allow decoupling of applications from the database design. Should there be changes in design, the applications need not be changed provided that suitable mappings can be written for the new design. A detailed description of the semantic mappings is beyond the scope of this paper, we plan to address this issue in future.

For generating SQL queries by the VampirePrime engine, three inputs are required: semantic mappings, the ontology and the declarative descriptions. As VampirePrime can process only TPTP syntax, three translators are necessary for processing these inputs. We plan to reuse and modify existing tools such as the OWL API, the PSOA RuleML API, and PSOA2TPTP for the translation tasks.

In general, relational data are URI-free while any entity in a Web ontology is identified by a URI. Hence, efficient handling of URIs is important. As formulas in a Web rule language, PSOA rules can easily use entities with or without URI. We plan to use URI constructing functions for URI handling.

A list of HAI use cases has been identified in [5]. A thorough evaluation of our system can be performed by generating HAIKU SADI services that leverage these use cases and run on HAI data stored in the TOH DW.

References

[1] C. Bizer and A. Seaborne, "D2RQ - Treating Non-RDF Databases as Virtual RDF Graphs," in *ISWC2004 (posters).*

[2] D. Calvanese, G. D. Giacomo, M. Lenzerini, D. Lembo, A. Poggi, and R. Rosati, "MASTRO-I: Efficient Integration of Relational Data through DL Ontologies." in *Description Logics*.

[3] A. Riazanov and M. A. T. Arago, *Incremental Query Rewriting with Resolution*, ser. Canadian Semantic Web II. Springer US, 2010.

[4] A. Riazanov, G. W. Rose, A. Klein, A. J. Forster, C. J. Baker, A. Shaban-Nejad, and D. L. Buckeridge, "Towards clinical intelligence with SADI semantic web services: a case study with hospital-acquired infections data," in *Proceedings of the 4th International Workshop on Semantic Web Applications and Tools for the Life Sciences*, ser. SWAT4LS '11.

[5] A. Riazanov, A. Klein, A. Shaban-Nejad, G. W. Rose, A. J. Forster, D. L. Buckeridge, and C. J. O. Baker, "Semantic querying of relational data for clinical intelligence: a semantic web services-based approach," *J. Biomedical Semantics*.

[6] M. Wilkinson, B. Vandervalk, and L. McCarthy, "The Semantic Automated Discovery and Integration (SADI) Web service Design-Pattern, API and Reference Implementation," *Journal of Biomedical Semantics*, vol. 2, no. 1, p. 8, 2011.

[7] G. Rose, V. Roth, K. Suh, M. Taljaard, C. Van Walraven, and A. Forster, "Use of an electronic data warehouse to enhance cardiac surgical site surveillance at a large canadian centre." *Clin Invest Med*, vol. 31, no. 4, p. S21, 2008.

[8] H. Boley, "A RIF-style semantics for RuleML-Integrated Positional-Slotted, Object-Applicative Rules," in *Proceedings of the 5th international conference on Rule-based reasoning, programming, and applications, in RuleML'2011.*

[9] G. Sutcliffe, "The TPTP Problem Library and Associated Infrastructure: The FOF and CNF Parts, v3.5.0," *Journal of Automated Reasoning*, vol. 43, no. 4, pp. 337–362, 2009.

[10] M. S. A. Manir, A. Riazanov, H. Boley, and C. J. O. Baker, "PSOA RuleML API: A Tool for Processing Abstract and Concrete Syntaxes." in *RuleML'2012*.

[11] G. Zou, R. Peter-Paul, H. Boley, and A. Riazanov, "PSOA2TPTP: A Reference Translator for Interoperating PSOA RuleML with TPTP Reasoners." in *RuleML'2012*.

[12] M. Horridge and S. Bechhofer, "The OWL API: A Java API for OWL ontologies," *Semantic Web*, vol. 2, no. 1, pp. 11–21, 2011.

Looking into Reactome through Biopax Lens

Laleh Kazemzadeh *, Helena Deus*, Michel Dumontier† and Frank Barry‡
* Digital Enterprise Research Institute
National University of Ireland, Galway,
Email : laleh.kazemzadeh@deri.org
† Department of Biology
Ottawa Institute of Systems Biology
Ottawa, Canada
Email: michel_dumontier@carleton.ca
‡ National Centre for Biomedical Engineering Science
National University of Ireland, Galway
Email:frank.barry@nuigalway.ie

Abstract—**In order to understand cell behavior under different conditions, the computational simulation of biological pathways is of great interest. Hence, to simulate a biological pathway computationally, extensive knowledge of protein-protein interactions (PPIs) in the pathway is required, along with the information about the generic flow of the pathway components i.e. biological reactions, which comprise the concerned pathway.**

The popularity of Semantic Web technologies in tackling the integrative bioinformatics challenges has increased, with various approaches used to aggregate and correlate data from different sources. However the integration of publicly available pathway databases, to determine the different PPIs and hence effectively simulate the cell behavior, has still various obstacles. In this paper, we present a semantic approach in pathway-wise analysis of protein-protein interactions (PPIs) using Biopax standards focusing particularly on Reactome database. We have identified the PPIs involved in a given pathway by the hierarchical extraction of its components (complexes, proteins, small molecules). We have developed a visualization tool which automatically generates a visual representation of the directed graph of PPIs in any specified pathway. Our approach provides intuitive inference of the data by flattening the nested pathways in Reactome and their components instead of wrapping each layer of data in the shell of outer pathway. We have also discussed that the representation of a pathway in Biopax standard format is highly complex and even contains redundant information. Hence tools are needed in order to facilitate the navigation and analysis of pathway datasets, which have been structured in Biopax format.

I. INTRODUCTION

The functionality of the human body is tightly regulated by biological pathways. Basic building blocks of these pathways are proteins, which act in an orchestra in order to keep the regulation of pathways intact. Therefore understating the dynamic of these pathways is directly dependent on understanding how the proteins involved in a pathway interact with each other. Interaction between two proteins might be of different types e.g. activation, inhibition, and methylation. Analyzing biological data from a pathway perspective can result in valuable information about the process of disease and suggest new drug discovery methods that target mis-regulation in specific pathways, thus enabling a much more precise targeting of diseases. However, computationally representing a pathway is not a trivial exercise due to the various types of components and interactions; regulation of pathways requires a cascade of events and interactions between genes, proteins and small molecules.

In addition, there is significant cross-talk between pathways, which highlight the fact that pathways are not isolated but are made up of a network of components. As such treating them as a system as opposed to an enclosed and self-contained pathway, can support a more realistic investigation.

II. STATE OF THE ART

A large number of tools and applications, vocabularies and ontologies aimed at computationally modeling biological pathways currently exist with enough precision to enable realistic simulations of its processes and determination of mechanism of action of various molecular compounds; examples include the systems biology markup language (SBML) [1] and the Proteomics Standards Initiative-Molecular Interaction (PSI-MI)[1]. These models and data format are also devised to deepen and broaden our understanding of pathways. A few models also keep track of semantics, i.e. they attempt to precisely and unambiguously describe each compound and each interaction such that they can be interpreted by applications and thus be integrated with other models. Biological Pathway Exchange (Biopax) [2] is one such data format. Biopax is a standard format for representing pathways and molecular interactions within and between pathways which has been developed with the aim of facilitating the process of collecting, indexing and sharing data [2]. Several databases hosting pathway and protein interaction information, such as Reactome[2] and Pathway Commons [3], are already available in this format. Information retrieved from expert-curated databases like Reactome is highly valuable for scientific advancement since they are the most accurate training data sets. However, because they rely on human curation, they suffer from limited coverage in the amount of interactions available. Integrating such data

[1] http://www.psidev.info
[2] http://www.reactome.org

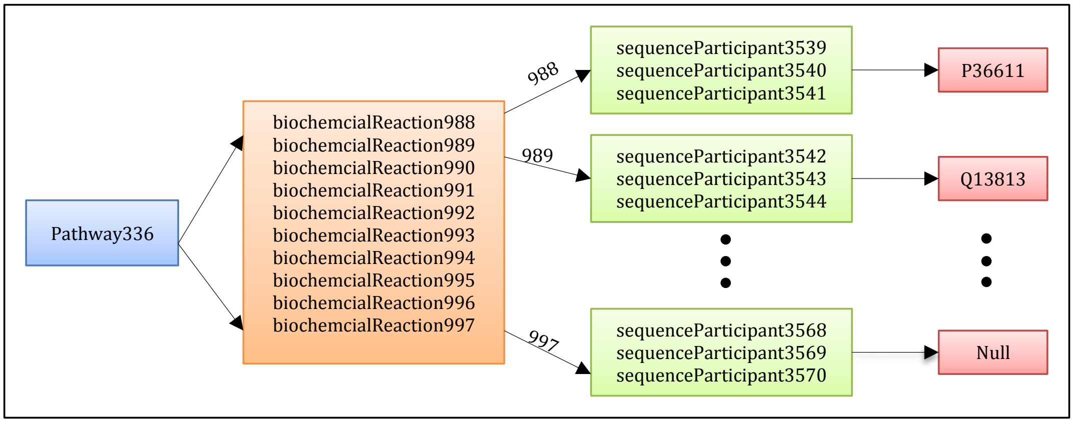


Fig. 1: Example of redundancy and incompleteness of data represented in Biopax level2 taken from caspase-mediated cleavage of cytoskeletal pathway. Blue box indicates the sample pathway, orange boxes represent list of biochemical reactions associated to this pathway, green boxes show sequence participant at left and right of each biochemical reaction, red boxes depict the unique Uniprot ID for each protein which each left and right of a biochemical reaction points to.

warehouses in one standard format will improve the coverage and highlight the role of Biopax in standardization. There is an enormous potential in using the information represented in Biopax format to realistically address biological questions, for example, the metabolic effects of a compound in the cell or how certain alterations in the metabolic network can be at the root cause of diseases or drug resistance. The discovery and confirmation of a biologically meaningful molecular interaction often requires the analysis of enormous amount of heterogeneous data which are typically deposited in local databases and isolated from each other. Therefore, considerable amount of molecular interactions are "hidden" in this data, which can only be exposed once these results are integrated and recurrence of patterns indicative of interactions analyzed. The data integration challenges in life science have motivated the researchers to adapt the new integration technologies offered by Semantic Web and Linked Data. Semantic Web technologies can provide a bridge between the datasets, enabling the discovery of links, which are often not obvious. These bridges are often standard vocabularies and ontologies developed toward improvements in knowledge discovery that lead to the next challenge: the representation, application and acceptance of these standard vocabularies by the domain experts. The motivational scenario for the work presented here is the extraction of all the molecular components that act in a particular biological process as described by Biopax in its various data sources. We have chosen Biopax firstly because it has been adapted by several databases, which provide information in signalling pathways and secondly becasue it faciliates data integration from other sources containing protein information.

Biopax has been developed to capture various aspects of signalling, regulatory and metabolic pathways. However in order to provide a descriptive solution and to cover all details in the description of pathways, some complexity needed to be introduced. In Biopax each pathway is constructed in the form of nested pathways which partially, but not fully, illustrate the overlaps between several pathways. Furthermore, each biochemical reaction is described as a function of the "left" and "right" hand side of the stoichiometric equation. Fig. 1. illustrates an example of data complexity and redundancy in representing biochemical reactions involved in pathway336 (caspase-mediated cleavage of cytoskeletal). As it is mentioned before each biochemical reaction has left and right components each of which refers to unique and separate sequence participant. However, each of these sequence participants points to the same protein ID from UniProt database. In other word, both left and right of a given biochemical reaction point to the same protein and this increases the redundancy of the data. The aim of our work is to devise a tool that aggregates information from this data e.g. the protein interactions and components of protein complexes in pathways. This will allow us to easily identify common interaction between various components (proteins, complexes, etc.) across pathways, abstracting from the complexity of pathway representation in Biopax. The data analysis tools made available by Reactome are unable to provide this inner-pathways analysis unless pathways are nested or siblings.

III. Methods

One typical way of querying a pathway or interaction between two proteins from different online databases is through browsing their webpage. As easy as it seems, it is time consuming and cumbersome to go through all the databases available manually. Instead we can query the PPIs directly from the raw data provided by the databases like Reactome and other such pathway databases. We propose an approach to overcome such problems which is explained below.

Fig. 2 shows an overall view of the steps, which were taken in our approach in order to identify the protein-protein interactions pathway-wise. We downloaded the protein-protein interaction file for Homo sapiens from Reactome webpage in Biopax format. This data was uploaded to our Sesame

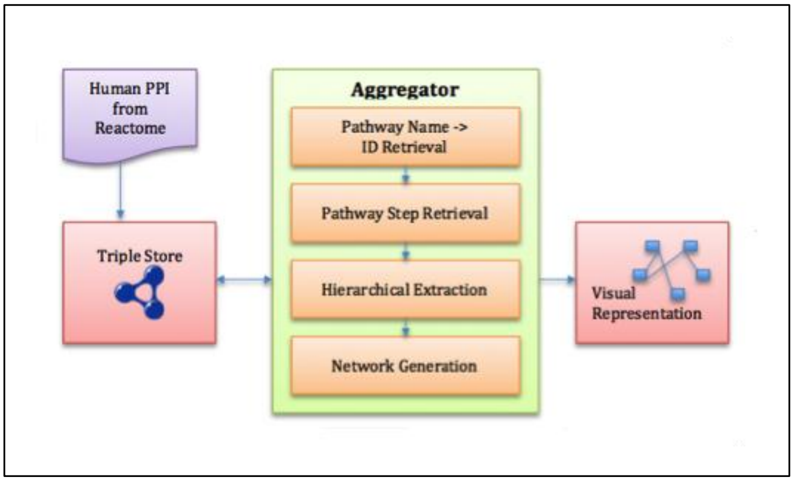


Fig. 2: Overall view of the proposed method.

server[3] in the form of triples. The Aggregator module has been developed in order to extract the components involved in a pathway and break down the pathway to the level of complexes, proteins and molecules.

The system provides a list of selectable pathways compatible with the pathways names used in Reactome. The ID of the selected pathway e.g. Apoptosis or Programed Cell Death (PCD) is retrieved from the triple store by the ID Retrieval module. The Pathway Step Retrieval retrieves the list of inner pathways (pathway-steps) forming the selected pathway. Each of these pathways is segregated hierarchically in the Extraction module.

The extracted data from Pathway Step contains bundle of relational information explaining reactions, complex blocks, proteins and small molecules forming complexes. Network Generator constructs a model in the final stage from the data extracted in the previous step. This model is then fed to the network visualizer, which renders and displays the relational graph between components of the pathway. In this model, the relation between each entity, complex, protein and molecule in the pathway is illustrated in a directed graph where nodes represent the entities, pathways, proteins and molecules and edges represent the connections between source and target nodes or the higher level and lower level components in a pathway tree.

The interaction Aggregator is written in PHP using ARC2[4] package in order to query the Reactome triples. The force-directed graph is generated by the Data Driven Documents (d3)[5], library written in Javascripts.

IV. Results

Raw material in our approach is an input .owl file, which contains the information of any pathway in Biopax. Applying our method we were able to generate a pathway wise PPIs network which is shown and discussed below.

Fig. 3 shows a small part of the network visualization generated by our tool for the Apoptosis pathway. The generated network contains 60 interactions between 40 pathways, representing nested pathways in Reactome, and 87 proteins involved in inner pathways of Apoptosis. Here we show the interaction between pathway336 and pathway335, which are caspase-mediated cleavage of cytoskeletal proteins and apoptotic cleavage of cellular proteins pathways respectively. These two pathways are part of outer pathways of Apoptotic execution phase and Apoptosis, which are not shown here.

The number of identified proteins in pathway336 is 8, while the number of reported proteins for the same pathway in Reactome database is 32. The reason for these differences is that some of the reported proteins in Reactome point to the same unique protein identifier. As an example protein P08670, Vimentin, has been mentioned 7 times. Likewise Q151149 and the rest of identified proteins have been reported 3 times. Our algorithm was not able to identify 3 proteins (caspase 3,6,7) in the list of 32 proteins reported in Reactome database due to incompleteness of the original data which was downloaded from the Reactome webpage.

Of great interest in pathway anlysis is identification of protein hubs. Protein hubs are those proteins with high degree of connectivity and more likely to be essential in the cell. Example of such a protein is shown in Fig. 4. Protein Q14790 (caspase 8), appears to be involved in the following pathways: Fasl/DD95L signaling (pathway309), TNF signaling (pathway310), Trail signaling (pathway311), Formation of caspase 8 (pathway312), Activation of pro-caspase 8 (pathway313) and Apoptotic execution (pathway 334). Knowing the protein ID or name and assuming the protein of interest is involved in different pathways we are able to retrieve the same information from Reactome search tool, however it does not give us the intuitiveness of the visualization. Querying the same protein, casapse 8, in Reactome returns more hits than the number of

[3] http://hcls.deri.org:8080/openrdf-workbench/repositories/
[4] https://github.com/semsol/arc2/wiki
[5] http://d3js.org/

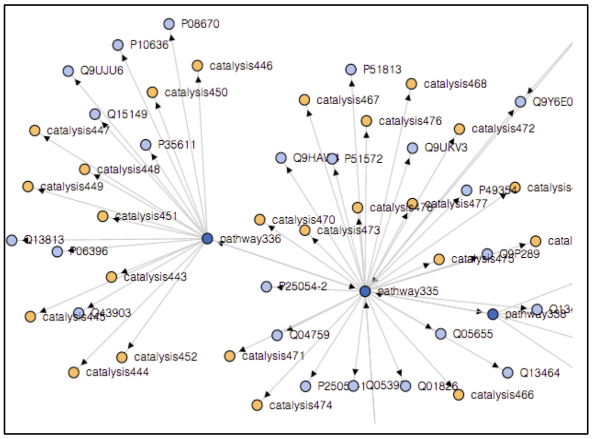


Fig. 3: Directed graph generated by the network visualizer. Graph shows the interaction between and within two pathways. Pathways and proteins are shown with their unique IDs. Each edge represents the connection between pair of source and target nodes. Dark Blue: pathways, light blue: proteins, orange: catalysis.

pathways we discussed here since we limited the search only to the Apoptosis pathway and not all the pathways exist in Reactome.

V. Conclusion

In this work we were able to extract PPI associated with any given pathway. Our visualization provides a better representation of elements involved in a pathway since it is capable of retrieving and representing data while conserving the hierarchy in which data was originally represented. Our aim was to highlight the PPIs in the pathways hence we represented only pathways and proteins in the deepest level of each pathway step of an outer pathway. However the data retrieved from the triple store by Aggregator contains more information about each pathway than only its components (e.g. pathway name) and with the current structure of our tool it is possible to add an extra layer of data to the Network Generator and create a visual representation of the extended network including e.g. protein complexes or type of interactions which, if added, the system will be more infromative. Our tool is compatible with Biopax level 2 thus it may not generate the same expected result when it is provided with a data file in Biopax level 3. Moreover, during the course of this work we have observed and analyzed Biopax format in detail. Some of the classes and properties introduced in Biopax appear unnecessary but also raise the level of complexity in the pathway representation and pathway analysis. Some of these complexity issues have been addressed and improved in later release of Biopax but pathways represented in Biopax level 2 suffers from this unnecessary complexity. In this work we tried to diminish the amount of redundant data by omitting the biochemical reaction, left and right step in each pathway step and showing only the proteins involved in a single pathway at the most inner level.

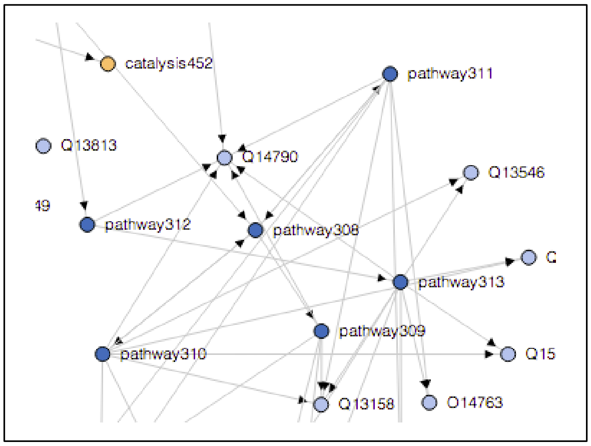


Fig. 4: Protein hub connecting six inner pathways in the Apoptosis pathway.

VI. Future Work

Future work will be the integration of pathways and interactions from other databases like BioGrid [4], MINT [5], HPRD [6] and the expansion of the query and visualization in such a way that two or more pathways from different sources can be queried and the common interactions highlighted. Furthermore, identified interactions will be ranked based on the number of occurrence in the databases and the literature.

Acknowledgment

This work has been funded by Program for Research in Third Level Institutions (PRTLI) Cycle 5, which is co-funded by the European Regional Development Fund (ERDF).

References

[1] M. Hucka, A. Finney, H. M. Sauro, H. Bolouri, J. C. Doyle, H. Kitano, A. P. Arkin, B. J. Bornstein, D. Bray, A. Cornish-Bowden, A. A. Cuellar, S. Dronov, E. D. Gilles, M. Ginkel, V. Gor, I. I. Goryanin, W. J. Hedley, T. C. Hodgman, and J. Hofmeyr, *he Systems Biology Markup Language (SBML): A medium for representation and exchange of biochemical network models, Bioinformatics, vol. 19, pp. 524–531, 2003*

[2] E. Demir, M. Cary, S. Paley, K. Fukuda, C. Lemer, I. Vastrik, G. Wu, P. D'Eustachio, C. Schaefer, J. Luciano, F. Schacherer, I. Martinez-Flores, Z. Hu, V. Jimenez-Jacinto, G. Joshi-Tope, and K. Kumaran, *The BioPAX community standard for pathway data sharing, Nature Biotechnology, vol. 28, pp. 935–942, 2010*

[3] E. G. Cerami, B. E. Gross, E. Demir, I. Rodchenkov, Ö. Babur, N. Anwar, N. Schultz, G. D. Bader, and C. Sander, *Pathway Commons, a web resource for biological pathway data, Nucl. Acids Res., 2010*

[4] C. Stark, B.J Breitkreutz, T. Reguly, L. Boucher, A. Breitkreutz, and and M. Tyers, *BioGRID: a general repository for interaction datasets, Nucleic Acid Re., no. 1, pp. 535–9, 2006*

[5] A. Ceol, A. A. Chatr, L. Licata, D. Peluso, L. Briganti, L. Perfetto, L. Castagnoli, and G. Cesareni, *MINT, the molecular interaction database: 2009 update, Nucleic Acids Res., vol. 38,Database, 2010*

[6] T. S. K. Prasad, R. Goel, K. Kandasamy, S. Keerthikumar, S. Kumar, S. Mathivanan, D. Telikicherla, R. Raju, B. Shafreen, A. Venugopal, L. Balakrishnan, A. Marimuthu, S. Banerjee, D. S. Somanathan, A. Sebastian, S. Rani, S. Ray, and C. J. H. Kishore, *Human Protein Reference Database - 2009 Update, Nucleic Acids Research., no. 37, 2009*

A Substrate Description Framework and Semantic Repository for Publication and Discovery in Cloud-Based Conferencing

Jerry George[#1], Fatna Belqasmi[#2], Roch Glitho[#3], Nadjia Kara[*4]

[#]*Concordia University, Canada*
[#1]`jerry.george@concordia.ca`
[#2]`fbelqasmi@alumni.concordia.ca`
[#3]`glitho@ece.concordia.ca`

[*]*ETS, University of Quebec, Canada*
[*4]`nadjia.kara@etsmtl.ca`

***Abstract* – Cloud computing is an emerging paradigm with three main facets: Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS). Several benefits are expected from cloud-based conferencing (e.g. efficiency in resource usage, easy introduction of new conferencing applications). This paper proposes a publication and discovery architecture for the interactions between the substrate providers, the infrastructure providers, and the broker of a cloud based conferencing business model.**

***Keywords*—cloud-based conferencing, publication, discovery, semantic repository, cloud conference ontology.**

I. Introduction

Conferencing is the conversational exchange of media between several parties. A business model has been recently proposed for cloud-based conferencing [1]. There are five roles in the proposed business model: connectivity provider, broker, conferencing substrate provider, conferencing infrastructure provider, and conferencing service provider. Conference substrates are elementary building blocks that can be virtualized and shared between conferencing applications for resource efficiency purposes. This paper proposes an architecture for realizing the interactions between the substrate provider, the infrastructure provider, and the broker. The substrates need to be described in a non-ambiguous manner for publication and discovery purposes from both technical and business perspectives. Furthermore, a repository is also required to enable the actual publication and discovery of the substrates.

Our proposed architecture is made up of a semantic-oriented description framework for substrates and a repository for publication and discovery of the substrates. The description framework is made up of a substrate description language and cloud-based conference ontology, both of which should meet ten key requirements.

First, the substrate description framework should be standards-based. Second, it should enable machine-readable substrate description. Third, the substrate description framework should hide the heterogeneity of the substrates and provide the service interfaces in a uniform manner. Fourth, the substrate description language and cloud conference ontology should accommodate both the technical and business aspects of the conference substrates. Fifth, the substrate description language should be flexible by supporting a wide range of data formats.

Sixth, the repository interface for publication and discovery should be independent of the stored substrates. Seventh, the interface should be based on existing standard protocols/APIs. Eighth, to support easy interoperability, the interface should be flexible in terms of the supported serialization formats for substrate description. Ninth, the interface should enable the specification of both technical and business aspects using standard technologies, while publishing or discovering the substrates. Tenth, the substrate repository should provide either an extensible architecture for adding support for new languages or explicit support for a chosen description language.

Work has been done on both the substrate description language [2], [3] and cloud-based conference ontology [4], [5]. However, none of this related work meets all the requirements. The next section presents the proposed architecture, followed by the implementation architecture and prototype. The final section concludes this paper.

II. Proposed Architecture

In this section, the overall architecture is presented first, followed by the substrate description framework, and then the substrate repository.

A. Overall Architecture

Figure 1 depicts the overall proposed architecture. The substrate and infrastructure providers communicate with the repository via a REST interface. The discovery requests are described using SPARQL, and are transferred as REST request content.

The substrate repository uses a semantic data store to save the substrate descriptions and the cloud conference ontology, which serves as a reference ontology for the validation of substrate description documents during publication. The repository includes a set of supporting components to access, validate, and manage the substrate description documents and cloud conference ontology. These components can be classified into three categories. The first category supports the validation and the management of the substrate descriptions and it includes the substrate document validator and substrate classifier.

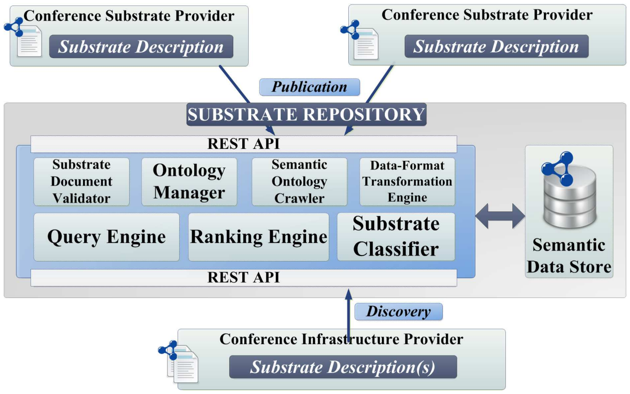


Fig. 1: Proposed architecture for publication and discovery

The second category is used for the management of the cloud conference ontology and it consists of the ontology manager and semantic ontology crawler. The last category enables efficient discovery of substrates and it contains the query and ranking engines. The data-format transformation engine is a supporting component used for both management and discovery of substrates.

B. *Substrate Description Framework*

The description framework defines a new cloud-based conference ontology and reuses OWL as the description language. The cloud conference ontology consists of three key constituent ontologies – cloud infrastructure, substrate description and conference ontologies. The reasoning support for these ontologies can be supported by OWL-DL reasoners. It reuses existing ontology (e.g. Linked-USDL) concepts, which extends to meet the conferencing specifics.

The cloud infrastructure ontology describes the business aspects of the cloud conferencing infrastructure, such as the substrate and the infrastructure providers' information, and the subscription information (i.e. which infrastructure provider is subscribed to which substrate). Figure 2 presents the main concepts and properties that constitute the cloud infrastructure ontology.

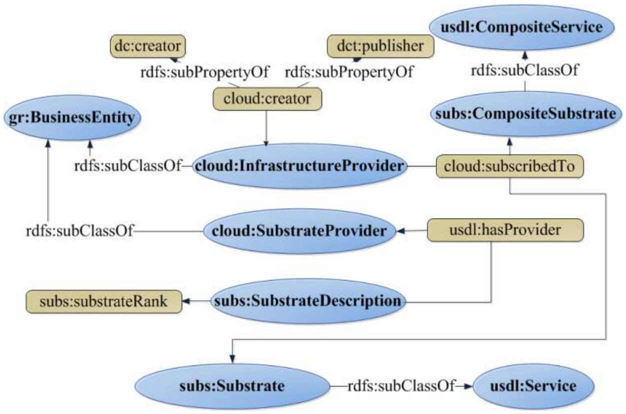


Fig. 2: Cloud Infrastructure Ontology

The conferencing substrates are modelled as Linked-USDL services, allowing the reuse of the Linked-USDL models for expressing the pricing (e.g. per user, per month, etc.) and the constraints information. Linked-USDL allows constraints specification for both atomic substrates (e.g. signalling substrate) and composite substrates (e.g. dial-out audio conference substrate).

The substrate description ontology (Figure 3) describes the technical aspects of the substrates, such as the interfaces and the substrate features. The substrate interfaces are described through the set of operations they encompass, along with the inputs and outputs of each operation. The operations are described as per the SA-REST service model, which we extend in order to support asynchronous substrate operations. We added a collection of seven properties to define an asynchronous callback end-point. The substrate features indicate the other functional features of the substrate (i.e. other than the interface ones), such as the substrate type (e.g. audio mixing, signalling). Composite substrates may have multiple features or capabilities, which are described using an RDF list. The substrate description ontology provides a classification for the common conferencing substrate features,

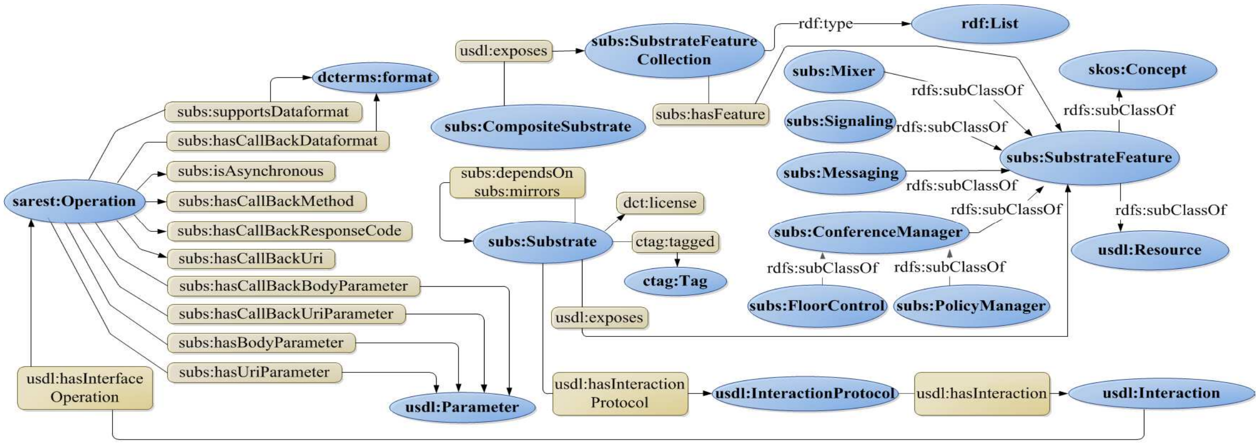


Fig. 3: Substrate Description Ontology

including signalling, mixing, and advanced conference control features such as floor control and policy management.

The conference ontology gives in-depth information about the conference and its participants (Figure 4). A conference is depicted as a composition of a set of substrates. A conference is also defined as a Linked-USDL resource, to capture the fact that it is the concrete object that implements the conferencing service. The participants are described using three important descriptors – signalling, media and preference descriptors.

C. Substrate Repository

The substrate provider may choose to publish the substrate description document in any supported RDF serialization format. Prior to storing a published document, the substrate repository converts the document into XML format using the data transformation engine. The substrate repository then checks the document validity against the cloud conference ontology and set of inference rules. This function is handled by the substrate document validator, which seeks the help of the ontology manager to retrieve the latest version of the ontology from the semantic data store. Once the validation is completed, the substrate description document is stored in the semantic data store. At regular intervals of time, the substrate classifier indexes the published documents based on the substrates' type (e.g. signalling, mixing, etc.). Indexing periodically instead of after each publication optimizes the repository resource usage and up time. For instance, the indexing may be scheduled for periods when traffic is low, and use the full capacity of the repository to answer the users' requests during the busiest period. The indexing reduces the response time for simple discovery requests (e,g. those based on substrate type), and it is performed only when needed. The infrastructure provider can look for a substrate by providing the criteria required as part of the request content. The criteria are specified using the SPARQL specification. Upon receiving the request, the substrate repository uses the query engine to parse the SPARQL query and ensures the request is coherent with the described ontologies. The query engine is then used to optimize the query using SPARQL re-writing rules for basic graph pattern (BGP) based on the index generated by the substrate type classifier. The infrastructure provider may limit the number of substrates to get in the response, in which case the ranking engine is used to prioritize the results. The ranking engine utilizes the multi-criteria decision making scheme proposed in [6] to rank the substrates based on some of their characteristics (e.g. latency, availability, and cost). The description documents of the selected substrates are then reformatted (if needed) according to the data format (e.g. XML, JSON, N3) supported by the infrastructure provider. Such a transformation is performed by the data transformation engine.

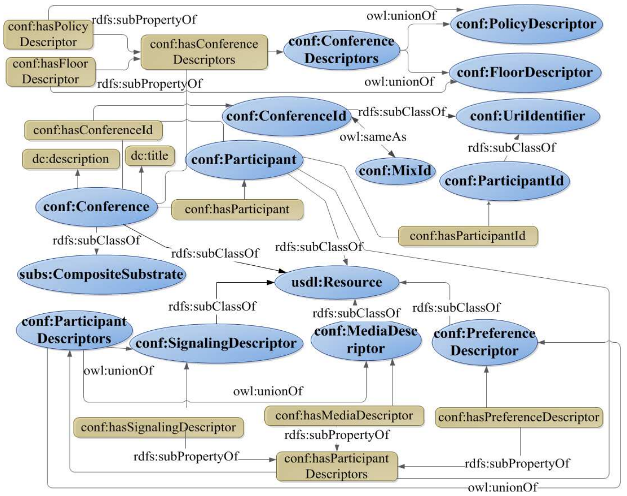


Fig. 4: Conference Ontology

III. Implementation

We first present the implemented prototype, followed by the performance measurements.

A. Prototype

The prototype consists of a substrate repository with both publication and discovery interfaces, and a set of infrastructure and conference substrate providers. The semantic data store component of the repository is based on Sesame and the other components are implemented using Sesame and RDF2Go libraries. Sesame is an open-source framework for storing and querying RDF data, and RDF2Go provides an abstraction layer for easier communication with the Sesame data store. The built-in SPARQL query optimizer of Sesame is extended to support optimizations based on BGPs related to substrate types. To support inference and validation, the Sesame framework's parser module is used along with OWLIM[1], a family of semantic-based database management systems. The data transformation engine uses Apache Any23 libraries for transformation between the RDF serialization formats. The REST interfaces are implemented using Jersey, a reference implementation of JSR 311.

To have a near-realistic view of the system execution, we needed a test bed setup with several dozens of substrates belonging to different providers, as well as random and varied constraints. We implemented a benchmarking tool including a substrate test data generator and a query generator, representing a set of substrate and infrastructure providers respectively. Both generators are implemented using Java concurrency API and can issue varying numbers of parallel requests to the substrate repository. Some of the existing benchmarking tools for RDF-based repositories such as Berlin SPARQL Benchmark[2] allow only the benchmarking of pre-defined use cases with specific sets of product templates.

Two laptops were used to run the prototype. The first one was used to run the substrate repository, while the second was used to run the benchmarking tool for publication and discovery.

[1] OWLIM - http://www.ontotext.com/owlim

[2] Berlin SPARQL Benchmark (BSBM) - http:// bit.ly/17RxHHZ

B. Performance Metrics

The performance of our prototype is assessed in terms of time delays for both publication and discovery. The publication delay measurements were taken for different numbers of substrate providers, different number of simultaneous requests, and for the cases where different numbers of substrates were published prior to the time of measurements. The discovery delays were measured for two types of queries: simple and complex. Simple queries are, for instance, those based only on the substrate type. Complex queries may include multiple relational criteria (e.g. capacity>=100 and latency<=1000ms), textual operations (e.g. textual search for a specific provider or substrate within a specific region), or ranking criteria (e.g. get an ordered list of the first 10 recommended audio mixers in Canada). We also compared the discovery delays of simple queries with and without optimization to show the added value of the optimization algorithm.

C. Performance Results

Figure 5 shows the results. Each measurement is calculated as an average of 15 experiments. Figure 5.a displays the measurements for publishing up to 32 substrates simultaneously by varying the number of existing substrates in the semantic data store. As expected, the delays increase with the number of simultaneous publications as well as the number of substrates already in the registry. Nevertheless, the delays remain acceptable considering that the publication is a one-time operation performed by the substrate providers.

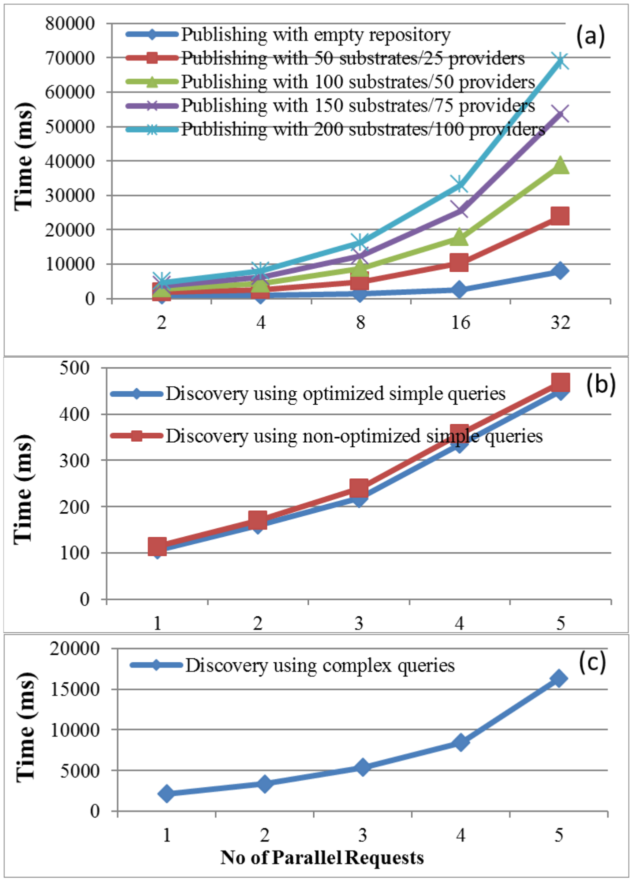


Fig. 5: Performance measurements for substrate repository: a) publication delays; b) discovery delays for simple queries; c) discovery delays for complex queries.

The discovery delay measurements were performed on a substrate repository containing 100 substrates. The discovery requests are randomly generated by the benchmarking tool, according to the chosen request complexity (i.e. simple or complex). Figure 5.b compares the discovery delays for optimised and non-optimised simple queries. The results show that optimization reduces delays by about 7%; this percentage can be further increased by creating indexes for frequently-used BGPs, such as substrate provider region. Complex discovery queries require more processing time and induce much larger delays compared to simple queries (Figure 5.c). An optimization solution for such queries is therefore worth investigation.

IV. Conclusion

We proposed a substrate description framework and semantic repository architecture for cloud-based conferencing substrates. A proof-of-concept prototype was implemented, deployed, and successfully tested. The performance results for the proposed architecture delivers satisfactory results for publication and discovery of conference substrates. However, methods for further optimization need to be investigated for complex queries. Our future work is also directed toward extending the already-implemented repository architecture to other providers of the cloud-based conferencing business model.

References

[1] R. H. Glitho, ‘Cloud-based Multimedia Conferencing: Business Model, Research Agenda, State-of-the-Art’, in *2011 IEEE 13th Conference on Commerce and Enterprise Computing (CEC)*, 2011, pp. 226 –230.

[2] R. Kanagasabai, ‘OWL-S Based Semantic Cloud Service Broker’, in *Web Services (ICWS), 2012 IEEE 19th International Conference on*, 2012, pp. 560–567.

[3] Jos de Bruijn and Dieter Fensel, ‘Web Service Modeling Language (WSML) - W3C Submission’. [Online]. Available: http://www.w3.org/Submission/WSML/. [Accessed: 18-Jan-2013].

[4] J. Li and F. Yang, ‘Resource-Oriented converged network service modeling’, in *Communications Technology and Applications, 2009. ICCTA’09. IEEE International Conference on*, 2009, pp. 895–899.

[5] N. Loutas, E. Kamateri, and K. Tarabanis, ‘A Semantic Interoperability Framework for Cloud Platform as a Service’, in *Cloud Computing Technology and Science (CloudCom), 2011 IEEE Third International Conference on*, 2011, pp. 280–287.

[6] Y. Cui, C. Chen, and Z. Zhao, ‘Web Service Selection Based on Credible User Recommended and QoS’, in *Computer and Information Science (ICIS), 2012 IEEE/ACIS 11th International Conference on*, 2012, pp. 637–642.

CONTEXT AWARE SERVICE DISCOVERY AND SERVICE ENABLED WORKFLOW

Altaf Hussain, Wendy MacCaull
Centre for Logic and Information, Dept. of Mathematics, Statistics, and Computer Science
St. Francis Xavier University
Antigonish, Nova Scotia, Canada
ahussain@stfx.ca, wmaccaul@stfx.ca

***Abstract*—We provide a conceptual model for context aware Semantic Web Service (SWS) discovery, which can utilize real-time legacy data from external systems and support user context-based service discovery and selection. This model offers advantages over current SWS technology which cannot be easily applied to different domains or be integrated with legacy systems. Using this conceptualization we propose an intelligent decision support system, which offers Service Enabled Workflow.**

Keywords—Semantic Web Service, Context Aware Service Discovery, Service Enabled Workflow, Service Metadata, Ontology

I. INTRODUCTION

A service is an entity that offers an intended value to its consumer; in today's society, people are dependent on service paradigms. A service consumer may need to pay an exchange value to consume a service but does not have to be concerned with how the service is developed or delivered. The service model design, development, and delivery are the concern of, and are handled by, the service providers: e.g., *the Postal Service*. *Web Service (WS)* is the technology that makes services available as consumable entities accessed and consumed through computers, via the Web: e.g., *the Email Service*. WS technology, backed by *Service Oriented Computing* and *Service Oriented Architecture (SOA)* has gained attention and popularity in the commercial computing sector as an enabling technology for the most enduring service planning, development, delivery and management methodology. As a result, a new spectrum of web applications has emerged supporting Business-to-Business integration, e-commerce, and industry wide collaboration. These applications are empowered by the WS technology, which provides a platform supporting independent communication and machine-to-machine interaction framework. However, WS technologies need extensive human involvement for service discovery, composition, invocation, etc.

In the recent years, a new paradigm has evolved, called the *Semantic Web (SW)*, supporting machine-readability, and automated trusted interaction between computers with minimal human intervention. The markup language of the SW is based on the Web Ontology Language (OWL), which can be used to express logical relations among entities on the web, and leads to a new class of WS called *Semantic Web Service (SWS)*. A Semantic Webservice is a standalone piece of functionality that is self-descriptive, machine-readable, and can be automatically discovered and executed via the web. The SWS, inheriting the properties of the SW and the WS has achieved many desirable properties, namely: a) machine independent communication and machine readability b) easy and widely acceptable collaboration methodologies c) exploitation of SW and reasoning techniques. Effort has been made in the areas of SWS, for example: semantic description of WS, semantic reasoning based WS discovery and SWS delivery thorough ontology based concepts and frameworks e.g., Web Ontology Language–Services (OWL-S) [1], and Web Service Modeling Ontology (WSMO) [8].

As SWS becomes more popular, users expect it to be easier to integrate with different domains and legacy systems. Existing SWS approaches do not provide any easy methodology to integrate domain data (often housed in traditional databases) in the service discovery process to

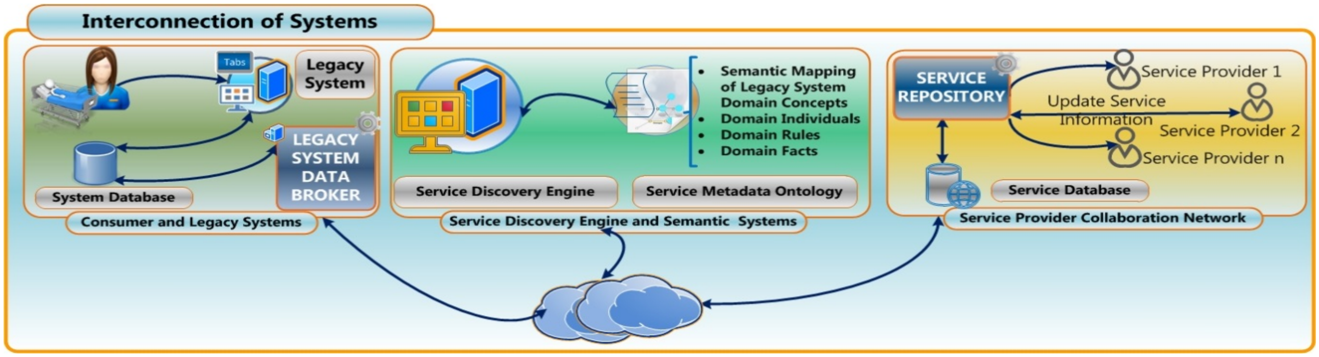

Interconnection of Systems
Tabs
Legacy System
LEGACY SYSTEM DATA BROKER
System Database
Consumer and Legacy Systems
Semantic Mapping of Legacy System
Domain Concepts
Domain Individuals
Domain Rules
Domain Facts
Service Discovery Engine
Service Metadata Ontology
Service Discovery Engine and Semantic Systems
SERVICE REPOSITORY
Service Provider 1
Update Service Information
Service Provider 2
Service Provider n
Service Database
Service Provider Collaboration Network

Figure 1: Interconnection of legacy systems and SWS system

support *context aware* service discovery. However, users frequently need to select services based on domain situations. To support automatic interoperation of the SWS discovery process with traditional systems, SWS discovery should be able to utilize real-time data from external systems and domains, providing automatic discovery and selection services based on domain situations and conditions. See figure 1.

TABLE I. RELATED SERVICES AND THEIR DESCRIPTION

Service Name	Service Quality Property: Cost, Relocation Duration	Dependent Services	Related Domain Data
Helicopter Service	$2000, 1 hour	Paramedic Service, Oxygen Supply Service, …	Patient Condition, Patient Respiratory Status, …….
Ambulance Service	$1000, 3 hours	Paramedic Service, Oxygen Supply Service, …	
Bus Service	$100, 4 hours	Paramedic Service …	
………	…………..	……………..	

We present a small example from healthcare describing problems users face to discover a service that depends on domain context, and motivating features to be supported. Suppose a patient is in a hospital in Antigonish and a medical professional determines that he should be relocated to Halifax for care that is more specialized. The user submits *Query 1* (see Textbox 1 below) to a SWS discovery engine, which will match the query with a service repository and provide a list of relocation services. However, this query does not incorporate other inputs such as patient condition, or patient disease history and the user later may need to select a service depending on such patient properties (examples of such selection strategies are *Selection Strategy 1* and *Selection Strategy 2)*. If none of the discovered services fits patient properties, the user must initiate another discovery request and lose precious time.

Query 1:"Get a Relocation Service that can relocate Patient P from Antigonish to Halifax."
Selection Strategy 1: If the Patient's Condition is Normal, Select the Low Cost Service for relocating the Patient from Antigonish to Halifax.
Selection Strategy 2: If the Patient's Condition is Critical, select the Fastest service for relocating a Patient from Antigonish to Halifax.
Fact 1: The condition of the Patient P is Critical.
Rule 1: If the patient's condition is Critical, use fastest mode of Relocation Service.
Fact 2: The Patient P has a Respiratory Problem.
Rule 2: If the patient has a Respiratory Problems, there should be an Oxygen Sservice supplied while relocating.
Rule 3: If the Patient's Condition is Critical, a Paramedic should accompany the Patient while relocating.
IQ 1: "Get the fastest Relocation Service to relocate Patient P from Antigonish to Halifax (uses Query 1, Fact 1, and Rule 1)."
IQ 2:"Get the fastest Relocation Service that supports Oxygen Supply Service while relocating Patient P from Antigonish to Halifax (uses Query 1, Fact 1, Fact 2, Rule 1, and 2)."
IQ 3: "Get the fastest Relocation Service that can support Oxygen Supply Service and Paramedic Sservice while relocating Patient P from Antigonish to Halifax (uses Query 1, Fact 1, Fact 2, Rule 1, 2 and 3)."

Textbox 1: Examples of Queries, Facts and Rules

Query 1 can be answered by state-of-the-art SWS approaches like OWL-S or WSMO. However, *Selection Strategies 1* and *2* show how a user's decision may change based on patient properties. To support strategies representing domain awareness, the user must inspect the patient medical record and then make a decision based on the quality properties of the list of services discovered. The selection strategies can be articulated using domain object properties called *facts* stating the real-time patient data properties and values, *rules* stating the action required to be taken by the user based on the facts and services, and the queries. We can model S*election* S*trategy* 2 by the listed *Fact 1, Rule 1* and *Query 1*.

In addition, the patient's condition may also force the user to select several other services that should accompany the selected service (the primary service). In such a case, the user must know which services can be provided to the patient along with a primary service. To support such features, the user has to consider the services enabled by one service and with regard to patient's medical service consumption history and current condition. For example in Table 1, an Oxygen Supply Service is enabled by the Helicopter Service which means, if a user chooses Helicopter Service, he can also choose Oxygen Supply Service. However, for the Bus Service, he cannot choose the Oxygen Supply Service. The user has to manually interface different system components, namely: the patient data system, the service dependability knowledge and SWS discovery engine. Hence, the user faces a great deal of difficulties while trying to provide more than one service at a time to the patient.

Step 1: Determine if the Patient's Condition isCritical or not. If yes, then
Step 2: Select the fastest service manually from the list of services returned by the service discovery engine for Query 1.
Step 3: Find out if the Patient has a Respiratory Problem. If yes, then
Query 2: "Get an Oxygen SupplySservice that can be provided while Patient is transferring using fastest Relocation Service selected by Query1."
Step 4: If there is an Oxygen Supply Service that can be provided with the selected Relocation Service then continue to the next fact. If there is no such Oxygen Supply Service selected from Query 1, go back, reissue Query 1, and select the next fastest service. Repeat until an Oxygen Supply Service is found.
Step 5: If the Patient's Condition is Critical then,
Query 3: "Get Paramedic Service that can be provided while the Patient is relocating with the service selected by Query 1."
Step 6: If there is a Paramedic Service returned by the service discovery engine, the user could select that one. If there is no such service, the user has to select next fastest service from Query 1.

Textbox 2: A scenerio of user interfacing different systems manually

In addition, while the user tries to select services for a patient the user might need facts and rules in relation to selection strategies (e.g., facts and rules are *Facts 1 and 2, Rules 1*, *2* and *3 in* Textbox 1*)*. This situation requires the user to check the database, and do additional steps. Also, based on the service dependencies, the user may have to restart the process from the beginning if the selected service cannot provide all of the required services. A typical scenario is given below.

The domain facts and rules lead the user to do several more queries (Query 2 and Query 3) (see Textbox 2) and manually select services that are returned by traditional SWS discovery processes. However, one can see that from Query 1, Facts 1 and 2 and Rules 1, 2, and 3, we are really interested in getting the results of the possible *inferred queries* IQ1, IQ2 or IQ3 (see Textbox1), where IQ3 is the optimal query. For time critical applications, taking such service dependencies into the discovery process makes it more efficient and user friendly.

We describe a framework for intelligent SWS description, discovery, and delivery that extends existing frameworks to: improve service discovery performance, facilitate integration of domain-based information, and interface with legacy systems such as workflow management systems. A workflow is a collection of interconnected Tasks with a specific control flow. Each Task has a specification representing the action needed to be carried out. We propose the notion of Service Enabled Workflow (SEW) which will allow us to discover services using the task specification as a query to the SWS discovery engine which will determine services that can carry out the action required by the task. SEW can provide desirable features such as: a) distributed workflow execution utilizing the standalone nature of the services; b) service collaboration among various service providers as SEW can support the choice and execution of services from different providers, using them in a single workflow; c) decentralization of the workflow design, execution, and low coupling among workflow design and execution environment.

II. PROPOSED MODEL AND ARCHITECTURE

Our framework focuses on the easy integration of SWS with a domain context and facilitates the interfacing with systems developed using traditional approaches. The basic approach of service discovery traditionally contains a Service Discovery Engine, a Service Repository, and a Domain Service Ontology; we add a data and context integration component and a service metadata ontology. We can incorporate the data and context of legacy system by facts and rules that can be utilized by SWS discovery for context and real-time data service discovery and selection. The model supports context dependent service discovery using two ontologies which provide the logic for a given service selection: 1) Service Metadata Ontology which contains the service relationships with legacy system data and context; 2) Domain Facts and Rules Ontology. The Service Metadata Ontology consists of a) Service Domain Data Dependencies and b) Inter-Service Dependencies. These ontologies allow us to do reasoning over service metadata, can be specified using OWL-2, and, can be accommodated in both the OWL-S and in WSML-DL versions of WSMO approaches. We now discuss desirable features of a hybrid SWS based decision support systems.

Domain Integration and Context Aware Service Discovery: The "Relevant Domain Data" model articulates the association of a service with the relevant domain data; in Table 1 it includes column 1 and column 4. Based on the relevant domain data stated, we fetch data from the legacy system and assert them as facts in the Domain Facts and Rule Ontology. We can then use these facts asserted based on real-time data in the SWS discovery process. Asserting a fact about a domain at runtime, such as *Fact 1,* depends on the availability of the Patient P's property "*Patient Condition*" and the availability of property value "*Critical*" which is gathered in real-time from a database. Rules depending on the system's situation and data context that express the decision strategy related to a fact are also asserted in the domain ontology. At runtime, these rules will change the result of the discovery query to that of an inferred query due to a more refined search and discovery of services. Applying the facts and rules during discovery, the answer to an inferred query can will obtained by applying reasoning. This will reduce the need of user inspection and interaction to get a service that best suits the user's need.

Service Metadata Based Reasoning and Discovery: The Inter-Service Dependency Relationships model can enable us to do on the fly service orchestration which can also save the number of queries required. The model expresses the relationships between services in the service spectrum. A list of interdependent services are provided in the service description which then can be used in the discovery process and reasoning. E.g., in Table 1, if the user selected a *Relocation Service* like *BusService*, the user cannot select *OxygenSupplyService* because it is not supported but can select *ParamedicService*. So, depending on the need of the patient and service relationship, a service selection decision can be made.

Aggregation Query Support during Discovery: It is hard to support some special queries like *"Get the fastest relocation service"* using existing SWS discovery techniques. This requires that we incorporate procedural programming capability in a service discovery query. Procedural programming operation will be used along with DL based ontology query languages e.g. SPARQL Protocol and RDF Query Language (SPARQL) [4] and Semantic Query-Enhanced Web Rule Language (SQWRL) [14]. This will allow users to express complex aggregation and procedural operations easily and intuitively in a discovery query.

Service Enabled Workflow: SEW imagines workflow as a collection of tasks with control flows where tasks are carried out as services. A workflow task has defined specifications, which can be imagined as a user query for the discovery of a service and the workflow engine can ask the service discovery engine to discover services according to these task specifications. The workflow user may select a service to execute from the discovered list of services. Continuing in this fashion, we can provide dynamic composition of services: the overall result is SEW. SEW is a desirable feature that can easily provide workflow collaboration support with minimum efforts thorough service discovery and runtime composition.

We propose a SOA based architecture shown in figure 2, which supports integration of different domains; it consists of the following components:

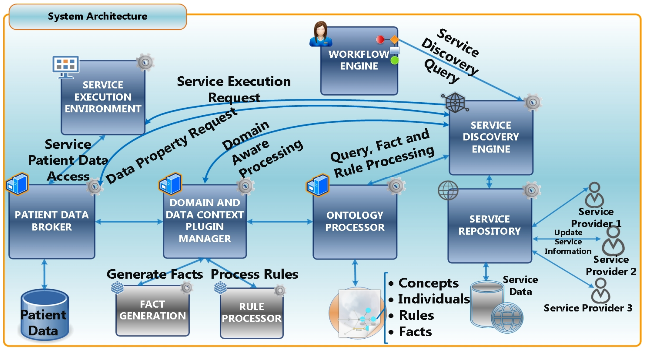


Figure 2: System Architecture

Workflow Engine: works as a user query generator and execution engine that enables Service Enabled Workflow.

Service Discovery Engine: serves as a central communication hub. It also carries out several decision-making tasks about service dependency reasoning, and carries out rules resulting in procedural steps.

Service Execution Environment: a server environment providing service runtime requirements and run services.

Patient Data Broker (Object Data Broker): works as a broker to get data from external systems.

Ontology Processor: is responsible for managing the ontologies and querying the ontologies.

Service Repository: is responsible for holding information about services provided by the service providers.

Domain and Data Context Plugin Manager: is responsible for the facts and rules related processing and domain based plugin management.

III. Related Work

The prominent conceptualizations of the SWS are OWL-S [1][11] and WSMO [3][15]. OWL-S helps software agents to discover web services that satisfy some specified quality constraints also provide a minimal set of composition templates. However, these abstract definitions can only be applied in a static service composition and can only be arranged as a predefined combination of services in the ontology. In [6], several types of inter-process dependencies are modeled using UML including Enabling, Cancelling, Triggering, and Disabling dependencies. WSMO also provides a concept vocabulary to express service description in terms of IOPEs but it currently only supports syntactical matching of a user's goal against service descriptions. OWLS-MX [9] and WSMX [7] are the SWS execution and testing environments for the SWS developed using OWL-S and WSMO approaches, respectively. OWLS-MX implemented the hybrid service discovery matchmaking using the OWL-2 reasoner Pellet. OWLS-MX and WSMX both support SW query languages SQWRL or SPARQL to perform semantic discovery of services but do not use domain data dependent facts and rules to discover services. SADI [16] provides a design pattern for publication of services, interoperability with traditional WS, and, semantic discovery and workflow generation based on service input/output transition metadata. SADI does not support domain data and context based service discovery and selction via integration and interoperation with legacy systems and data. Presently, there are a variety of approaches to improve the accuracy of a service discovery process, including collecting and integrating user feedback [2] and the addition of contextual information by defining design time semantic based user context [12]. In our approach, the service description enables us to foresee the services dependencies and reason about them to discover services that best suit the system and context conditions based on described facts and rules. In [5] a conceptual model of task-based workflow is provided that motivates our proposed Service Enabled Workflow. We extend the approach of [5] to support closer relationships with systems and contexts, and improve the state-of-the-art of such workflow systems. The Nova Workflow Workbench [10] is a task based workflow engine equipped with a high-level language, T□, [13] which is used to write task specification which include integration of data from a domain ontology. We plan to integrate our service discovery process to accept the task specification. The discovery process then can provide the selected service to the Nova Workflow engine, which executes the service to accomplish the task.

IV. References

[1] Ankolekar, A. et al., 2002. "DAML-S: Web service description for the semantic web." In The Semantic Web—ISWC 2002, Springer, p. 348–363.

[2] Averbakh, A., et al., 2009. "Exploiting user feedback to improve semantic web service discovery." In The Semantic Web-ISWC 2009, Springer, p. 33–48.

[3] Davies, J. et al., 2006. Semantic Web technologies: trends and research in ontology-based systems. Wiley.

[4] Garc'ia, J. et al., 2012. "Improving semantic web services discovery using SPARQL-based repository filtering." Web Semantics: Science, Services and Agents on the WWW.

[5] Grossmann, Georg et al., 2011. "Conceptual modelling approaches for dynamic web service composition." In The evolution of conceptual modelling, Springer, p. 180–204.

[6] Grossmann, G. et al., 2008. "Modelling inter-process dependencies with high-level business process modelling languages." In Proceedings of the fifth Asia-Pacific conference on Conceptual Modelling-Volume 79, p. 89–102. Austrian Computer Society, Inc.

[7] Harold, M. 2008. "WSMX documentation." http://maczar.deri.ie/papers/wsmx-documentation.pdf.

[8] Keller, U. et al., 2004. "Wsmo web service discovery." WSML Draft, http://www.wsmo.org/2004/d5/d5.1/v0.1/20041112.

[9] Klusch, M. et al., 2009. "OWLS-MX: A hybrid Semantic Web service matchmaker for OWL-S services." Web Semantics: Science, Services and Agents on the World Wide Web 7(2): 121–133.

[10] W. MacCaull and F. Rabbi, "NOVA Workflow: A Workflow Management Tool Targeting Health Services Delivery," in FHIES'11, ser. LNCS, vol. 7151. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 75–92.

[11] Martin, D. et al., 2004. "OWL-S: Semantic mark-up for web services." W3C member submission 22: 2007–04.

[12] Merla, C. 2010. "Context-Aware Match-Making in Semantic Web Service Discovery." Int'l Journal of Advanced Engineering Sciences and Technologies 9(2): 243–247.

[13] F. Rabbi and W. MacCaull: "T-Square: A Domain Specific Language for Rapid Workflow Development," in ACM/IEEE 15th Conf. on Model Driven Engineering Languages & Systems (MODELS 2012), Innsbruck, Austria (September, 2012). Proc, Lecture Notes in Computer Science, Volume 7590. pp. 36-52.

[14] Rodriguez, J. et al., 2010. "Improving Web Service descriptions for effective service discovery." Science of Computer Programming 75(11): 1001–1021.

[15] Wang, H. et al., 2012. "A formal model of the Semantic Web Service Ontology (WSMO)." Information Systems 37(1): 33–60.

[16] M. D. Wilkinson, et al., 2011. The semantic automated discovery and integration (sadi) web service design-pattern, api and reference implementation. *Journal of biomedical semantics*, *2*(1), 8.

Part V.

Systems Papers

Semantic Content Processing in Web Portals

Felicitas Löffler*, Bahar Sateli†, Birgitta König-Ries*, René Witte†
*Institute for Computer Science, Friedrich-Schiller University of Jena, Germany
†Department of Computer Science and Software Engineering, Concordia University, Montréal, Canada

***Abstract*—Web portals provide a standardized way of integrating multiple information sources and applications in a single web interface. However, they currently do not provide semantic support for users that need to navigate the often overwhelming amount of content. We demonstrate our open source portal architecture "hanüwa" that integrates text mining web services, based on the Semantic Assistants framework, with the Liferay portal server.**

I. Introduction

Web portals are a specific kind of web-based systems that provide for an integration of diverse information sources and applications. Deployed for a concrete scenario in an organization, they typically address the information needs of a wide range of users and their tasks through both internal and external services.

While a web portal provides convenient access to information, there is no standardized way that allows to further process the available content in order to support users in their tasks. There is also a lack of appropriate technologies for document filtering within a web portal. We envision a new generation of web portals that can provide context-sensitive support through semantic analysis services, in particular based on natural language processing (NLP). These services are deployed in shared or private servers and can be dynamically requested by users that ask for help in a specific task: e.g., finding entities in a documents, summarizing a text, answering a question, or linking content to external sources. As such, they perform the role of AI "assistants" that support their users. Furthermore, we imagine enhancing web portals with a personalization component to adapt the content to the user's needs. Sorting documents or highlighting terms according to a specific user interests would be a great advantage for the user and a step towards working against information overload.

In previous work, Bakalov et al. [1] demonstrated the feasibility and usability of a portal integration with natural language processing services. However, this implementation was tied to a specific, commercial portal engine (IBM WebSphere[1]). The work presented here is a complete re-design and re-implementation of the NLP-portal integration, taking into account future extensions and based exclusively on open source software. Similar to the solution presented in [1], we rely on the Semantic Assistants framework [2] for brokering text mining pipelines as web services, but our new architecture is based on the Liferay[2] open source portal server.

Our new *portlets* can be deployed in any existing Liferay-based portal to offer natural language processing services to its users. Here, we demonstrate the core functionality with named entity recognition in a given article, but the framework is not limited to a single domain: A clear separation of concerns allows a language engineer to make new NLP services available without requiring knowledge in portal technology, and a web engineer can easily design a new web portal that incorporates language technology.

II. Architecture

Our novel Semantic Assistants-portal integration architecture, illustrated in Fig. 1, is designed to allow various portlets to benefit from NLP techniques on their content. The core idea is to enable generic portlets to communicate with the *Semantic Assistants portlet*, specifically designed to connect to the back-end Semantic Assistants server and provide inquiry and invoking capability of NLP pipelines to portal users.

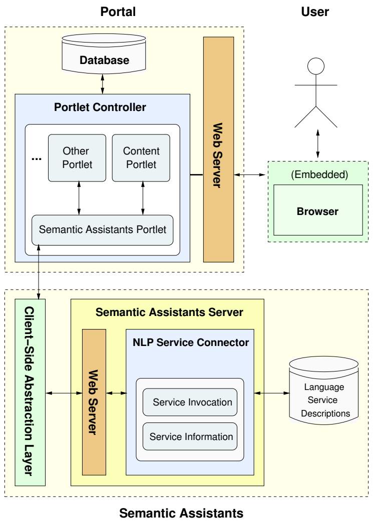


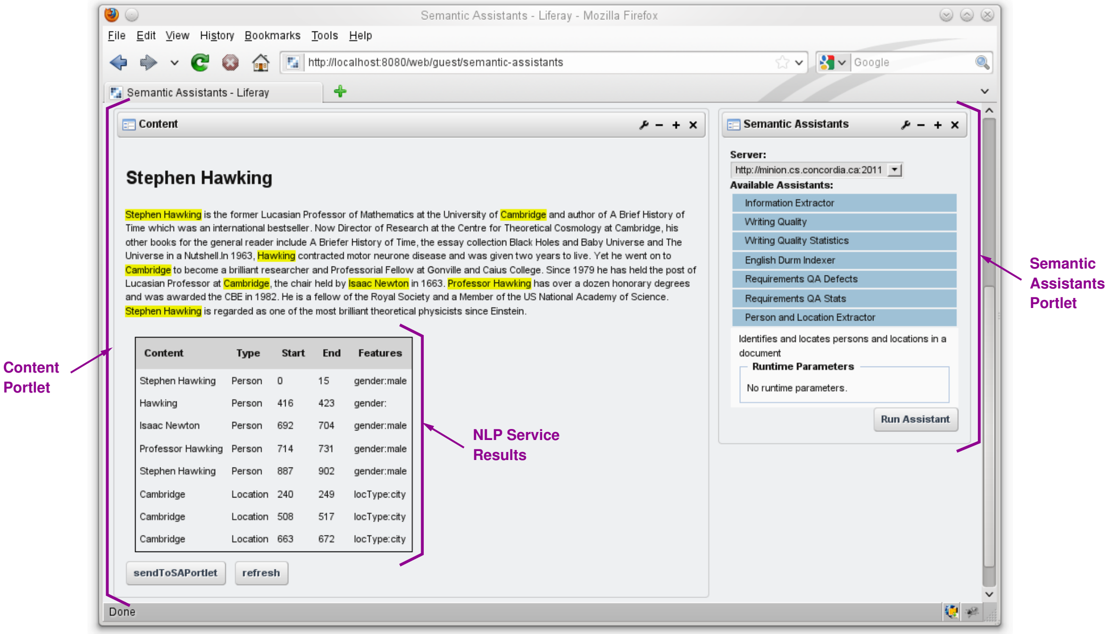
Fig. 1. The Semantic Assistants-Portal Integration architecture

In this architecture, all available portlets in a page can communicate with the Semantic Assistants portlet by sending content for analysis and receiving the results. To commence an analysis session, users interact with the portal via their web browser, for example, on their desktop computer or from a mobile device. Through this integration, users can select an NLP service to execute on a portlet's content from a dynamically-generated list of available *assistants* in the Semantic Assistants server repository. Where applicable, users can also customize the services' behaviour by setting runtime parameters. An execution request is then sent to the Semantic Assistants server from the Semantic Assistants portlet in form of a W3C[3] standard web service call that triggers the execution of the designated NLP pipeline on the provided content. The results of each

[1] IBM WebSphere, http://www.ibm.com/software/websphere

[2] Liferay, http://www.liferay.com/

[3] World Wide Web Consortium (W3C), http://www.w3.org


Semantic Assistants - Liferay - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost:8080/web/guest/semantic-assistants
Google
Semantic Assistants - Liferay
Content
Semantic Assistants
sendToSAPortlet
refresh
Run Assistant
Done
Content Portlet
NLP Service Results
Semantic Assistants Portlet

Stephen Hawking

Stephen Hawking is the former Lucasian Professor of Mathematics at the University of Cambridge and author of A Brief History of Time which was an international bestseller. Now Director of Research at the Centre for Theoretical Cosmology at Cambridge, his other books for the general reader include A Briefer History of Time, the essay collection Black Holes and Baby Universe and The Universe in a Nutshell.In 1963, Hawking contracted motor neurone disease and was given two years to live. Yet he went on to Cambridge to become a brilliant researcher and Professorial Fellow at Gonville and Caius College. Since 1979 he has held the post of Lucasian Professor at Cambridge, the chair held by Isaac Newton in 1663. Professor Hawking has over a dozen honorary degrees and was awarded the CBE in 1982. He is a fellow of the Royal Society and a Member of the US National Academy of Science. Stephen Hawking is regarded as one of the most brilliant theoretical physicists since Einstein.

Content	Type	Start	End	Features
Stephen Hawking	Person	0	15	gender:male
Hawking	Person	416	423	gender:
Isaac Newton	Person	692	704	gender:male
Professor Hawking	Person	714	731	gender:male
Stephen Hawking	Person	887	902	gender:male
Cambridge	Location	240	249	locType:city
Cambridge	Location	508	517	locType:city
Cambridge	Location	663	672	locType:city

Server: http://minion.cs.concordia.ca:2011

Available Assistants:

- Information Extractor
- Writing Quality
- Writing Quality Statistics
- English Durm Indexer
- Requirements QA Defects
- Requirements QA Stats
- Person and Location Extractor

Identifies and locates persons and locations in a document

Runtime Parameters

No runtime parameters.

Fig. 2. Semantic Assistants-Portal Integration User Interface in Liferay

successful service execution are first received by the Semantic Assistants portlet and then passed on to the portlet that requested the service execution. The NLP pipelines are described in the OWL[4] language and the Semantic Assistants server uses SPARQL[5] for a dynamic discovery of available services upon each user request. Hence, adding or removing NLP services to the integration requires no modification to the code base of the portal.

The basis of the personalization component will be an ontology-based user profile, where all user interests are recorded automatically while browsing through the portal and reading documents. A user interface, embedded into a portlet, allows a user to control interests, add new terms, delete or change concepts. The user can also enable or disable the personalization mode. When personalization is desired, the documents are re-sorted and the relevant terms of the user profile are highlighted within the text. In contrast to [1], the personalization feature will be available to various portlets in form of services, rather than a concrete implementation on a per-portlet basis.

III. Application

The integration of NLP assistants within a portal context allows for a multitude of applications. Fig. 2 shows an example scenario in which a portal user needs assistance in analyzing the textual content available in the content portlet (left). Such assistance can be offered to the user through the NLP services listed in the Semantic Assistants portlet (right). This portlet allows the user to connect to different Semantic Assistants servers and review the list of their available pipelines in order to find a suitable assistant for his task at hand. In our example, the list of assistants contains a "Person and Location Extractor" service that extracts entities of *person* and *location* types from a given text. The user then sends the text in the content portlet to the Semantic Assistants portlet for analysis and requests the service execution by clicking on the "Run Assistant" button. This interaction will request the designated Semantic Assistants server for the execution of the ANNIE pipeline, provided by GATE.[6] Subsequently, the results are returned to the content portlet in form of *annotations* in a tabular format and highlighted in the text based on their offsets. The processing time for different scenarios depends on both the length of the input text and the actual NLP pipeline. Naturally, sophisticated NLP pipelines with deep syntactic or semantic analysis require more time to process. Currently, we are working on a personalization scenario aimed at tackling the user's information overload issue, by filtering the portal's content according to a user's interest. The idea is to embed such capability directly within portlets, allowing users to be able to switch to various personalization modes.

IV. Conclusions

In this paper, we described our open source integration of natural language processing capabilities within a portal environment. We also intend to integrate a personalization feature into portals to adapt their content according to a user's needs. Furthermore, we want to provide a user interface to give the users the opportunity to have control over their recorded interests. The NLP-portal integration will be available as part of the Semantic Assistants distribution hosted on SourceForge.[7]

References

[1] F. Bakalov, B. Sateli, R. Witte, M.-J. Meurs, and B. König-Ries, "Natural Language Processing for Semantic Assistance in Web Portals," in *IEEE International Conference on Semantic Computing (ICSC 2012)*. Palermo, Italy: IEEE, September 2012.

[2] R. Witte and T. Gitzinger, "Semantic Assistants – User-Centric Natural Language Processing Services for Desktop Clients," in *3rd Asian Semantic Web Conference (ASWC 2008)*, ser. LNCS, vol. 5367. Bangkok, Thailand: Springer, Feb. 2–5, 2009 2008, pp. 360–374. [Online]. Available: http://rene-witte.net/semantic-assistants-aswc08

[4] Web Ontology Language, http://www.w3.org/2004/OWL/

[5] SPARQL Query Language, http://www.w3.org/TR/rdf-sparql-query/

[6] General Architecture for Text Engineering (GATE), http://gate.ac.uk/

[7] Semantic Assistants, http://sourceforge.net/projects/semantic-assist/

Semantic Tagging with Linked Open Data

John Cuzzola, Zoran
Jeremic, Ebrahim Bagheri
Ryerson University

Dragan Gasevic
Athabasca University

Jelena Jovanovic
University of Belgrade

Reza Bashash
SideBuy Technologies

***Abstract*—Making sense of text is a challenge for computers particularly with the ambiguity associated with language. Various annotators continue to be developed using a variety of techniques in order to provide context to text. In this paper, we describe Denote – our annotator that uses a structured ontology, machine learning, and statistical analysis to perform tagging and topic discovery. A short screencast for the curious is also available at http://youtu.be/espItTRQVzY as well as demonstration links provided in the conclusion.**

Keywords—semantic web, disambiguation, entity recognition, annotators, tagging, wikifying, linked-data, LOD

I. Introduction

The availability of structured link open data, through initiatives such as the “Linked Open Data (LOD)” project[1], has given rise to a new class of annotators for unstructured text. Annotators like TagME [1], DBPedia Spotlight [2], and Alchemy[2] all offer such capability. In this systems paper we describe Denote – our semantic tagging platform based on Linked Open Data. In section II, we outline Denote’s algorithm, describe its vocabulary, and key features. In III, we demonstrate these features and compare Denote’s output with other annotators.

II. Denote’s Design

Denote searches its ontology for similar concepts to the input text by performing keyword extraction then calculating a weighted Jaacard coefficient on resource descriptions. This provides a measure of text similarity. For each resource, its known categories (defined in the ontology) are subjected to a Bayesian filter to exclude those resources and categories that do not appear relevant. This provides a measure of semantic similarity. The surviving resources are then used for the annotations. Denote’s output is in the form of a synopsis whose lexicon is given in Table I. The output is a single sentence per annotation with a set of relevant URIs sorted in order of likelihood with confidence and available support statistics.

“Text” [Is_A {}] [[[With_Value •] Of_Units •] |
Acting_As {}] [Cat_Of {}]

Fig. 1. The output of an annotated text.

Denote uses a database of linked open data, represented in the form of n-triples (<subject><predicate><object>), to perform annotations, similarity identification, disambiguation and topic categorization. Denote's database is DBPedia [3]; an ontology derived from Wikipedia. In this respect, it resembles DBPedia Spotlight (DBPedia) and TagME (Wikipedia). However, Denote distinguishes itself in key ways. First, it attempts to assign context to the annotations by its [Acting_As] lexicon. Second, it attempts to annotate numbers [With_Value] through statistical analysis of similar concepts whose <predicate>:<object> are of the same data type [Of_Units]. Third, Denote has an extensive list of topic categories, made available through DBPedia’s <dcterms:subject> predicate, which it assigns to its annotations [Cat_Of]. These key differences were the motivation for Denote’s creation. While other annotators perform in a similar manner by first spotting word phrases and linking them to the disambiguated top-surface form;- Denote attempts to find related concepts that will be used to determine the properties of the spotted word phrases. This allows for role-based annotations [Acting_As]. We coin this process as *deep tagging* as opposed to the *shallow tagging* of Denote’s peers.

TABLE I. Denote’s annotation lexicon explained

Lexicon	Explanation
Is_A {}	"is a", "is an", "is used by". Asks: What is it?
Acting_As {}	Context/role. Asks: How is it used?
With_Value •	If number, Asks: What is the number value?
Of_Units •	If number, Asks: What is the units of measure?
Cat_Of {}	Asks: What relevant topic categories?

III. Demonstration

In this section, we describe three core functions in Denote’s toolkit: text annotation, number annotation, and category disambiguation.

A. The Text Annotator

Table II demonstrate Denote’s capabilities when compared to TagME and DBPedia Spotlight using the same input text of: “*BLT. The sub that proves great things come in threes. In this case, those three things happen to be crisp bacon, lettuce and juicy tomato. While there's no scientific way of proving it, this BLT might be the most perfect BLT sandwich in existence*. The default configuration for Denote, TagME and Spotlight were unchanged. Spotlight does not perform category analysis. TagMe gives a topic listing but this list is simply the annotated text rather than a separate categorization. Consequently, the [Cat_Of] portion of Denote’s synopsis was omitted and left for part C.

DBPedia Spotlight was the least effective with the fewest annotations and an incorrect disambiguation of BLT as a “Bizarre Love Triangle”. TagME performed well with

[1] http://linkeddata.org/
[2] http://www.alchemyapi.com/

numerous annotations with few mistakes (incorrectly tagged words “crisp” and “juicy”. Both Denote and TagME shared similar annotations but it is through Denote’s [Acting_As] vocabulary that provided context information. For example, both correctly annotated “lettuce” to its surface form, but it was Denote that identified that lettuce was *acting as* a *main ingredient*. Similarly, Denote linked the phrase “*bacon, lettuce, and juicy tomato*” as an *alias* or *alternate name*.

TABLE II. ANNOTATION OF “BLT. THE […] IN EXISTENCE.” WITH DENOTE, TAGME AND DBPEDIA SPOTLIGHT.

Annotated Word(s)	Denote (DBPedia)	TagME (Wikipedia)	DBPedia Spotlight (DBPedia/Wikipedia)
BLT	Is_A {/BLT} Acting_As {/name}		/Bizarre_Love_Triangle
BLT sandwich	Is_A {/BLT} Acting_As {/name}	/BLT	
sandwich			/Sandwich
in existence			/Existence
sub		/Submarine_sandwich	
crisp		/Potato_chip	
bacon	Is_A {/Bacon_sandwich, Bacon,Side_bacon} Acting_As {/mainIngredient, /ingredient}	/Bacon	
lettuce	Is_A {/Lettuce} Acting_As {/mainIngredient, /ingredient}	/Lettuce	
juicy		/Juice	
tomato	Is_A {/Tomato} Acting_As {/mainIngredient, /ingredient}	/Tomato	
bacon , lettuce and juicy tomato	Acting_As {/alias, /alternateName}		
scientific way		/Scientific_method	

B. *The Number Annotator*

The number annotator is unique with respect to other annotators in that Denote attempts to identify text that is normally associated with a numerical value. Using statistical analysis on the Jaacard/Bayes-discovered list of similar concepts, Denote attempts to match up number values with annotated text. Figure 2 demonstrates on the input text “*The radio shack color computer has only 16 kb of memory*”.

“memory” With_Value 16 Of_Units #int Cat_Of
{/Home_Computers, TRS-80_Color_Computer}

Fig. 2. An example of number annotation with Denote

C. *The Categorizer*

Denote has access to over 656,000 categories defined in DBPedia’s <dcterm:subject> ontology. A Bayesian filter is used on each similar concept in order to determine if the subject(s) of which the concept belongs to is contextually related to the text being annotated. DBPedia Spotlight demo does not perform topic category determination. TagME’s demo performs topic categorization by simply listing its annotated text in a cloud-tag structure rather than a defined set of category topics. Consequently, we compare Denote’s output with Alchemy. The Alchemy annotator can perform named entity extraction from a list of 200+ defined (sub)-entities. In this comparison, the “storyline” of *The Godfather* movie was retrieved from the *Internet Movie Database* (IMDb) and annotated. Table III gives the results.

TABLE III. DENOTE VERSUS ALCHEMY IN CATEGORY/TOPIC TAGGING

Annotated Word(s)	Denote with Category Determination	Alchemy Entity Extraction
Corleone Family	Is_A {/The_Family_Corleone} Cat_Of {/Italian_American_novels, /Novels_about_organized_crime_in_the_United_States,/Novels_by_Mario_Puzo, /Family_saga_novels}	
Don		TelevisionShow
Vito Corleone	Is_A {/Vito_Corleone} Cat_Of {/The_Godfather_characters}	Person
Vito	Acting_As {Person}	
New York	Acting_As {Location}	City
Micheal		Person
Don Vito		Person
Don Vito Corleone	Is_A {/Don_Vito_Corleone} Cat_Of {/The_Godfather_characters}	
Don’s		Person
Mafia	Is_A {/Mafia_Don} Cat_Of {/The_Godfather_characters}	
Drugs	Is_A {/Drugs} Cat_Of {/The_Godfather_characters}	

Alchemy results were limited to primitive named entity types of city and person with the exception of an incorrect categorization of “television show”. In contrast, Denote tagged text into rich categories that include “Italian-American novels”, “organized crime novels”, and “Godfather characters ”.

IV. CONCLUSION

In this paper we demonstrated Denote – a semantic annotator based on the DBPedia ontology and compared its features with that of same-class text taggers. Denote’s middleware engine demo is available at http://ls3.rnet.ryerson.ca/annotator while a developer-friendly demo is at http://inextweb.com/denote_demo. Denote’s annotation capabilities are wrapped around a RESTful interface allowing for 3rd-party developers to create their own semantic-aware applications. The result, we hope, is an improvement in information search and retrieval for the end user. Our future work involves parallelisation to scale the service for a large number of concurrent clients. We are also developing proof-of-concept demonstrations including a semantic movie recommender whose database will be included as a data-set to the LOD project.

REFERENCES

[1] P. Ferragina, and U. Scaiella, “TAGME: On-the-fly Annotation of Short Text Fragments (by Wikipedia Entities)∗”, In Proceedings of the 19th ACM international conference on Information and knowledge management (CIKM '10). 2010.

[2] P. Mendes, M. Jakob, A. García-Silva, and C. Bizer. “DBpedia spotlight: shedding light on the web of documents”, In Proceedings of the 7th International Conference on Semantic Systems (I-Semantics '11), 2011.

[3] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann. “DBpedia - A crystallization point for the Web of Data.” In Web Semant. 7, 3 (September 2009), 154-165. 2009.

A Semantic Framework for Data Quality Assurance in Medical Research

[1]Lingkai Zhu, [3]Helen Chen[*]
[1,3]School of Public Health and Health Systems
University of Waterloo
Waterloo, Ontario, N2L 3G1, Canada
{[1]l49zhu, [3]helen.chen}@uwaterloo.ca
[*] Corresponding author

[2]Kevin Quach
[2]Multi Organ Transplant Institute
University Health Network
Toronto, Ontario, M5G 2C4, Canada
Kevin.Quach@uhn.ca

***Abstract* — The large amount of patient data amassed in the Electronic Patient Record systems are of great value for medical research. Aggregating research-grade data from these systems is a laborious, often manual process. We present a semantic framework that incorporates a data semantic model and validation rules to accelerate the cleansing process for data in Electronic Patient Record systems. We demonstrate the advantages of this semantic approach in assuring data quality over traditional data analysis methods.**

Keywords* — *data quality assurance, data quality measurement, ontology modelling, semantic framework, semantic web standards

I. Introduction

Patient care is a highly complex process that involves multiple services and care providers in the continuum of care. Patient data collected may be incorrectly recorded or missing during busy clinical encounters. Thus, it is often very difficult to use patient data aggregated from a hospital's Electronic Patient Record (EPR) directly in health research which requires high quality data. Traditionally, data quality checking is performed by manual inspection and information processing, with the assistance of pre-defined data entry forms to impose data validation rules. The "cleaned" data are then stored in a research database. However, such activities must be customized to the registry platform, such as Microsoft Excel and Access. These proprietary rules are hardly interoperable with other systems and are limited in function. We propose a semantic framework that can explicitly describe the validation rules to govern data quality. The semantic framework can also perform complex cross-reference checks; whereas traditional error checking mechanisms would have difficulty incorporating, especially when the list of conditions changes over time, or changes with different application domains. Therefore, the use of a semantic framework can help accelerate and generate high quality research data over traditional techniques.

II. Literature Review

A. Categorizing Data Quality Problems

The quality of data is measured in multiple dimensions, which means "aspects or features of quality" [1]. We refer to three notable summaries of data quality dimensions [2][3][4]. Although there is no general agreement on classifications and definitions for dimensions, we identified three dimensions that are most suitable in our context: completeness, consistency and interoperability.

B. Improving Data Quality via a Semantic Framework

Brueggemann and Gruening presented three examples that demonstrate how a domain ontology can help improve data quality management [5]. According to the authors, applying semantic techniques brings advantages like suggesting candidate consistent values, using XML namespace to keep track of data origins and flexible annotation on results. We apply their three-phase methodology (construction, annotation and appliance) and demonstrate other benefits, e.g. rules expressed in semantic restrictions are more explicit than external algorithms.

Fürber and Hepp pursued a semantic approach of handling missing value, false value, and functional dependency data quality problems [6]. They chose SPARQL queries to implement rules detecting data deficiencies and described handling missing value sections that constraints, such as cardinality, are difficult to model in RDFS or OWL. However, OWL features such as owl:allValuesFrom and owl:oneOf are sufficient to model constraints from the database schema we use. We will express our semantic framework in OWL DL and SWRL. OWL DL provides class and property restrictions we need while remains decidable. DL-Safe SWRL rules are sufficiently expressive for our data quality rules, whilst provide ease of reusing already defined OWL classes and properties. This combination receives reasoning support from the Pellet reasoner[1].

III. Methodology

A. Architecture of Data Quality Assurance Framework

The data quality assurance framework is illustrated in Fig. 1 (rectangles and circles represent data repositories/ontologies and software modules, respectively). The whole framework revolves around a transplant EPR ontology, which is built with the openEHR reference model ontology [2] as the core framework, and refers to an ICD-10 ontology [3] for proper diagnoses definitions. The construction of EPR ontology starts with a script converting the database schema of an

[1] http://clarkparsia.com/pellet/
[2] http://trajano.us.es/~isabel/EHR/
[3] https://dkm.fbk.eu/index.php/ICD-10_Ontology

anonymized test medical database into an EPR taxonomy. The attributes in the database are captured in a class hierarchy and mapped into the OpenEHR ontology, and patients with data are imported as instances. Class restrictions and data quality validation rules are written in OWL and SWRL, respectively, and the Pellet reasoner handles reasoning for both. Through reasoning, data quality issues within the patient instances are recognized and annotated, which enables the data exporter module to clean the data, and provide the cleaned data to researchers for analysis.

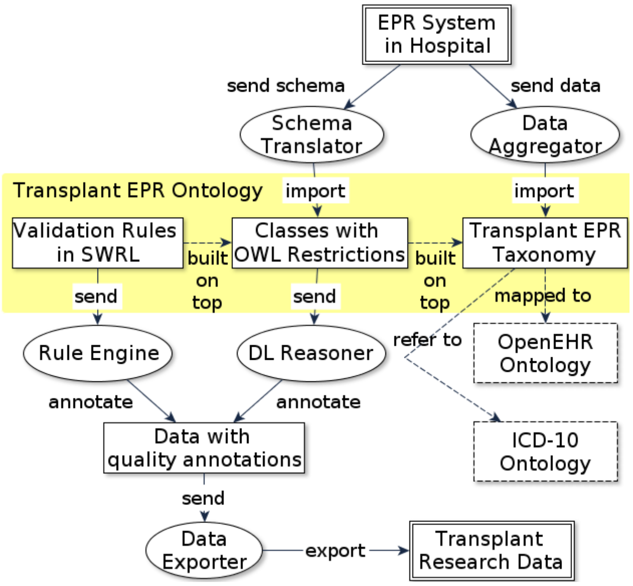


Fig. 1. Data Quality Assurance Framework Architecture

B. *Data quality assessment by dimensions*

To assess EPR data, three data quality dimensions are summarized for reference:

1. Completeness

Completeness refers to the proportion of data that is available in EPR relative to an expected complete dataset. This dimension can be used to examine the whole dataset as well as a single attribute.

Example: for all required attributes, instances that have at least one (by defining owl:someValuesFrom restrictions) valid value are annotated as complete.

2. Consistency

The consistency dimension refers to the logical coherence of relationships between data from different attributes, which frequently appear in an EPR domain. SWRL rules are employed to translate medical knowledge into logical connections properly.

Example: a post-transplant diagnosis cannot have a date earlier than transplant date; otherwise, it is a pre-transplant diagnosis and needs to be recorded as an error. A SWRL rule, using the date built-in, is able to identify such temporal inconsistencies and annotate them.

3. Interoperability

The interoperability dimension refers to the compatibility of a data element with other information systems. When importing diagnosis data, our data aggregator tries to seek each value in an external, standardized taxonomy, such as ICD-10. If the value is found, an owl:sameAs statement is made to map the value to the standard diagnosis definition, and the data element is marked interoperable.

IV. Preliminary Results

Restrictions and rules are implemented reflecting the identified data quality dimensions. Annotation sub-classes, such as "patient with complete demographic info", are created under the patient class. A reasoner is applied to classify all patient instances into these sub-classes. For each instance, we detect how many criteria it meets. For each sub-class, we know how many patients fall into it. Custom filters such as "patients who satisfy all rules" are also constructed. The results are manually reviewed and found correct.

V. Discussion and Future Work

Traditionally, data restrictions are enforced in an E-R database but its limited function could only ensure the completeness and the value range of data. Our semantic framework can perform the latter functions and can check for data consistency and interoperability, which brings greater benefit to medical research data quality.

The next step of our work is to repeat our methodology on a real and uncleaned EPR dataset. A research proposal has been submitted to a hospital based in Toronto with a transplant program for access to their dataset of 2000 patients. We will apply our semantic framework and identify any errors for review by researchers in the program. Once the framework’s robustness and accuracy is established, EPR data in production can be checked regularly to ensure the quality of health data.

References

[1] D. McGilvray, Executing Data Quality Projects: Ten Steps to Quality Data and Trusted InformationTM. Morgan Kaufmann, 2010.

[2] C. Batini, C. Cappiello, C. Francalanci, and A. Maurino, “Methodologies for data quality assessment and improvement,” ACM Computing Surveys (CSUR), vol. 41, no. 3, p. 16, 2009.

[3] C. Fürber and M. Hepp, “Towards a vocabulary for data quality management in semantic web architectures,” in Proceedings of the 1st International Workshop on Linked Web Data Management, 2011, pp. 1–8.

[4] P. Oliveira, F. Rodrigues, and P. Henriques, “A formal definition of data quality problems,” in International Conference on Information Quality, 2005.

[5] S. Brüggemann and F. Gruening, “Using domain knowledge provided by ontologies for improving data quality management,” in Proceedings of I-Know, pp. 251–258, 2008.

[6] C. Fürber and M. Hepp, “Using semantic web resources for data quality management,” in Knowledge Engineering and Management by the Masses, Springer, 2010, pp. 211–225.

Visualizing SWRL Rules: From Unary/Binary Datalog and PSOA RuleML to Graphviz and Grailog

Ismail Akbari, Bo Yan, Junyan Zhang, Harold Boley
Faculty of Computer Science
University of New Brunswick
Fredericton, NB, Canada
Email: {iakbari, b.yan, boliuy12, harold.boley} AT unb.ca

***Abstract-* SWRL rules are transformed in two steps for visualization in a subset of Grailog. A Unary/Binary Datalog rule in SWRL presentation syntax is translated to a corresponding PSOA RuleML rule in a RIF-like presentation syntax employing frame formulas. This is then translated to the Graphviz DOT language so that the Graphviz tool can render it visually as a Grailog graph with an object identifier and slots. Supported by the obtained visual graphs, users can more easily analyze the original symbolic logic rules.**

Keywords-* *Semantic Web; SWRL rules; Unary/Binary Datalog; F-logic; PSOA RuleML; Grailog; Visualization; Graphviz; Transformation

I. INTRODUCTION

The Semantic Web Rule Language (SWRL) [1] combines the sublanguages Web Ontology Language Description Logic (OWL DL) with the Unary/Binary Datalog RuleML sublanguage of the Rule Markup Language. The Graph inscribed logic (Grailog) has been introduced as a systematic graph standard for visual-logic knowledge [2]. This work uses transformations targeting the Graphviz tool [8] to visualize SWRL rules as Grailog 1.0 graphs. SWRL rules are translated to corresponding PSOA (Positional-Slotted, Object-Applicative) RuleML [6] frame rules, which are then translated to the Graphviz DOT language for rendering as Grailog graphs.

II. LANGUAGES AND TOOLS

There exist many methods and tools to visualize data and knowledge [3] in diverse areas. One of these areas is the Semantic Web, whose knowledge can be visualized via Directed Labeled Graphs (DLGs) and DLG-extending Grailog graphs.

A. OWL DL and OWL Lite

OWL achieves machine interpretability of Web ontologies by providing an XML syntax and a formal semantics [4]. SWRL's sublanguage OWL DL supports users who want high expressiveness while retaining computational completeness and decidability. OWL DL's sublanguage OWL Lite supports those users primarily needing a classification hierarchy and simple constraints.

B. Frame Logic

Frame logic (F-logic) is a frame-based language using slot-described objects typed by classes that are organized as a light-weight ontology (taxonomy) [5]. The semantics of F-logic makes the closed world assumption as opposed to the open world assumption of description logics. Also, F-logic is generally undecidable whereas OWL DL is decidable.

C. PSOA RuleML

PSOA RuleML is a rule language that deeply integrates relational (predicate-based) and object-centered (frame-based) modeling. In PSOA RuleML, the notion of a PSOA term is introduced as a generalization of: (1) the positional-slotted term in POSL [10] and (2) the frame term and the class membership term in F-logic and RIF-BLD [6].

D. Graphviz

Graph Visualization Software (Graphviz) is a package of open source tools that was introduced by AT&T Labs Research for graph drawing, e.g. via DOT language scripts [7]. Graphviz layout programs take the description of graphs in a simple text file based on the DOT language script format and generate diagrams (graphs) in the desired output format [8].

III. UNARY/BINARY FRAME DATALOG

In Grailog, we extend Unary/Binary Datalog with frames. A unary relation is a class pointing to the relation’s single argument as the node it types. A binary relation describes a relationship between two nodes.

A. Frame Formulas: Associating Slots with an Object Identifier

Slots in Grailog are drawn as special, bullet-attached arrows distinguishing a start node as playing the role of the Object IDentifier (OID). In Unary/Binary Frame Datalog, a node (an instance or a variable), acting as the OID of a frame, can be pointed to by a class-originating arrow for ('unary') typing and can have outgoing slot arrows. The same node can also act as the first or second argument of a binary relation, drawn as a regular (bullet-free) arrow. See figure 1 for an example.

IV. STRUCTURE OF THE IMPLEMENTATION

The main steps of our prototype implementation are as follows. First, the tool receives SWRL's (Unary/Binary) Datalog rules from the input and translates them into Frame Datalog. Next, it splits each rule into its components, including instances, classes and slots, written to a text file. From these components, it then generates the Graphviz DOT file. Finally, it calls Graphviz for the visual rendering of the graph output.

V. SWRL-TO-PSOA TRANSFORMATION

This section describes how to transform Datalog SWRL rules to Frame Datalog PSOA RuleML rules, used by our Grailog visualization and reusable generally. SWRL rules use a conjunctive formula as premise and as conclusion. After receiving a SWRL rule, it will be translated to a Frame Datalog rule in PSOA RuleML. As an example, consider the following SWRL rule. The "?" symbol indicates variables and "^" denotes conjunction:

```
Person(?x) ^                                   (1)
Man(?y) ^
hasAge(?x,?age1) ^
hasAge(?y,?age2) ^
hasSibling(?x,?y) ^
swrlb:greaterThan(?age2,?age1)
->
hasOlderBrother(?x,?y)
```

This is translated to the following PSOA RuleML rule, whose first two premises represent single-slot frames, where the term f(t) encodes the slot f->t:

```
hasOlderBrother(?x ?y) :-                      (2)
        And(?x#Person(hasAge(?age1))
            ?y#Man(hasAge(?age2))
            hasSibling(?x ?y)
            swrlb:greaterThan(?age2 ?age1))
```

The frame premises check that object "?x" of class "Person" has property "hasAge" with value "?age1" and object "?y" of class "Man" has property "hasAge" with value "?age2".

VI. ILLUSTRATIVE RULE RENDERING

An example is used to show the tool's operation. Consider formula (1) as the SWRL rule input. Its transformation to formula (2) and further processing described in [9] lead to the output (the graph) shown in figure 1. The red arrows show the premises of the rule. The green arrow shows its conclusion. Recall that a bullet distinguishes the OID of a slot arrow. An oval shows a class and an octagon shows a variable.

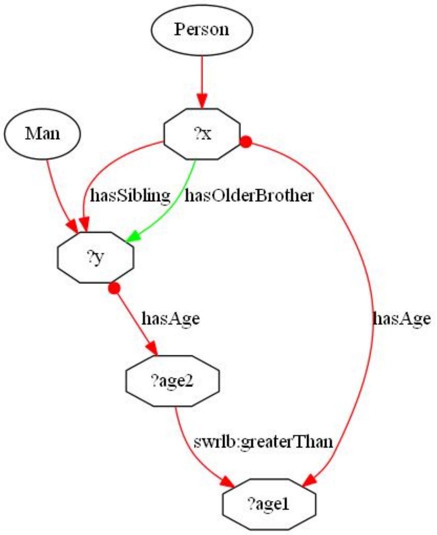


Figure 1. Graph rendered from SWRL rule in formula (1)

VII. CONCLUSION

Our tool transforms SWRL from Unary/Binary Datalog rules to Frame Datalog PSOA RuleML. The Graphviz-rendered visualization of frame rules as Grailog graphs lets people more easily analyze the logic of SWRL rules. By visualizing SWRL rules, this work is an implementation of a Grailog 1.0 subset. A demo and more details about the implemented system are online [9].

REFERENCES

[1] Horrocks, Ian, et al. SWRL: A Semantic Web rule language combining OWL and RuleML.W3C subm. 21 (2004): 79.

[2] Harold Boley. Grailog 1.0: Graph-Logic Visualization of Ontologies and Rules. Proc. RuleML 2013, Seattle, Washington, USA, July 2013, Springer LNCS 8035. Preprint: http://cs.unb.ca/~boley/papers/GrailogVisOntoRules.pdf

[3] Goldstein, Ira P. The genetic graph: a representation for the evolution of procedural knowledge. International Journal of Man-Machine Studies 11.1 (1979): 51-77.

[4] Pascal Hitzler, et al., OWL 2 Web Ontology Language Primer (Second Edition), W3C Consortium, Recommendation REC-owl2-primer-20121211, Dec 2012.

[5] Michael Kifer, et al.,. Logical foundations of object-oriented and frame-based languages. Journal of the ACM (JACM) 42.4 (1995): 741-843.

[6] Harold Boley. A RIF-Style Semantics for RuleML-Integrated Positional-Slotted,Object-Applicative Rules, Proc. 5th Int'l Symposium on Rules: Research Based and Industry Focused (RuleML-2011 Europe), Barcelona, Springer, July 2011.

[7] Koutsofios, Eleftherios, and Stephen North. Drawing graphs with dot. Technical Report 910904-59113-08TM, AT&T Bell Laboratories, Murray Hill, NJ, 1991.

[8] http://graphviz.org/

[9] http://2012team8project.weebly.com/index.html

[10] Harold Boley. Integrating Positional and Slotted Knowledge on the Semantic Web, *JETWI* 2(4):2010, 343-353.

Part VI.

Appendix

Author index