Monika Heiner, Annegret K. Wagler (Eds.)

Biological Processes & Petri Nets

6th International Workshop BioPPN 2015
Brussels, 22 June 2015
Proceedings

CEUR Workshop Proceedings
Volume 1373

Editors:

Monika Heiner
Brandenburg University of Technology Cottbus-Senftenberg
Computer Science Institute
Data Structures and Software Dependability
03013 Cottbus, Germany
monika.heiner@b-tu.de

Annegret K. Wagler
University Blaise Pascal (Clermont-Ferrand II)
Faculty of Sciences and Technology
ISIMA - LIMOS - CNRS
BP 10125, 63173 Aubiére, France
wagler@isima.fr

Online available as CEUR Workshop Proceedings (ISSN 1613-0073), Volume 1373
http://CEUR-WS.org/Vol-1373/

BibTeX entry:

```
@proceedings{bioppn2015,
  editor    = {Monika Heiner, Annegret K. Wagler},
  title     = {Proceedings of the 6th International Workshop on
               Biological Processes \& Petri Nets (BioPPN 2015),
               satellite event of Petri Nets 2015, Brussels, Belgium, June 22, 2015},
  booktitle = {Biological Processes \& Petri Nets},
  location  = {Brussels, Belgium},
  publisher = {CEUR-WS.org},
  series    = {CEUR Workshop Proceedings},
  volume    = {Vol-1373},
  year      = {2015},
  url       = {http://CEUR-WS.org/Vol-1373/}
}
```

Preface

This volume contains the peer-reviewed papers presented at BioPPN 2015 – the 6th International Workshop on *Biological Processes & Petri Nets* held on June 22, 2015 in Brussels as satellite event of PETRI NETS 2015 and ACSD 2015.

The workshop had been organised to provide a platform for researchers aiming at fundamental research and real life applications of Petri nets and other concurrency models in Systems and Synthetic Biology. Systems and Synthetic Biology are full of challenges and open issues, with adequate modelling and analysis techniques being one of them. The need for appropriate mathematical and computational modelling tools is widely acknowledged.

Petri nets offer a family of related models, which can be used as a kind of umbrella formalism – models may share the network structure, but vary in their kinetic details (quantitative information). This undoubtedly contributes to bridging the gap between different formalisms, and helps to unify diversity. Thus, Petri nets have proved their usefulness for the modelling, analysis, and simulation of a diversity of biological networks, covering qualitative, stochastic, continuous and hybrid models. The deployment of Petri nets to study biological applications has not only generated original models, but has also motivated research of formal foundations.

In this context, the invited talk on *A compact modeling approach for deterministic biological systems* given by Luis M. Torres from Centro de Modelización Matemática (ModeMat)/Escuela Politécnica Nacional Quito, Ecuador, addressed the problem of extending Petri nets in such a way to obtain a compact model for the dynamics of certain discrete deterministic systems.

In addition, there is a position paper presented by Hugues Bersini on *State-transition diagrams for biological modelling: A few cases* to initiate a discussion on advantages and drawbacks of different modeling frameworks (Petri nets versus state-transition diagrams) for biological modelling.

Each submission was reviewed by five to six program committee members assisted by one external reviewer. The list of reviewers comprises 18 professionals of the field coming from 11 different countries and writing in total 28 reviews, most of them of substantial length. The programme committees decided finally to accept five papers, involving 14 authors coming from four different countries.

In summary, the workshop proceedings enclose theoretical contributions and biological applications, demonstrating the interdisciplinary nature of the topic.

For more details see the workshop's website
http://www-dssz.informatik.tu- cottbus.de/BME/BioPPN2015.

We acknowledge substantial support by the EasyChair management system, see http://www.easychair.org, during the reviewing process and the production of these proceedings.

June 16, 2015 Monika Heiner
Cottbus Annegret K. Wagler

This page is intentionally left blank.

Table of Contents

Program Committee

Gianfranco Balbo	University of Torino, Computer Science Department, Italy
Marco Becutti	University of Torino, Computer Science Department, Italy
Rainer Breitling	University of Manchester, Manchester Institute of Biotechnology, UK
Ming Chen	Zhejiang University, College of Life Sciences, Department of Bioinformatics, China
David Gilbert	Brunel University, Centre for Systems and Synthetic Biology, UK
Simon Hardy	Université Laval, Institut universitaire en santé mentale de Québec, Canada
Monika Heiner	Brandenburg University of Technology Cottbus-Senftenberg, Computer Science Institute, Germany
Mostafa Herajy	Port Said University, Mathematics and Computer Science Department, Egypt
David R.C. Hill	Universite Blaise Pascal, Faculty of Sciences and Technology
Peter Kemper	College of William and Mary, Department of Computer Science, USA
Sriram Krishnamachari	Indraprastha Institute of Information Technology (IIIT), India
Chen Li	Zhejiang University, School of Medicine, Center for Genetic & Genomic Medicine, China
Fei Liu	Harbin Institute of Technology, Control and Simulation Center, China
Wolfgang Marwan	Otto von Guericke University Magdeburg & Magdeburg Centre for Systems Biology, Germany
Hiroshi Matsuno	Yamaguchi University, Graduate School of Science and Engineering, Japan
P.S. Thiagarajan	National University of Singapore, School of Computing, Department of Computer Science, Singapore
Annegret K. Wagler	Université Blaise Pascal (Clermont-Ferrand II), Faculty of Sciences and Technology

Additional Reviewers

C

Cordero, Francesca

The dynamics of deterministic systems – A survey

Luis M. Torres[1], Annegret K. Wagler[2]

luis.torres@epn.edu.ec, wagler@isima.fr

[1] Centro de Modelización Matemática (ModeMat)
Escuela Politécnica Nacional, Quito, Ecuador
[2] LIMOS (UMR 6158 CNRS)
University Blaise Pascal, Clermont-Ferrand, France

Abstract. We present a model for the dynamics of discrete deterministic systems, based on an extension of the Petri net framework. Our model relies on the definition of a priority relation between conflicting transitions, which is encoded in a compact manner by orienting the edges of a transition conflict graph. The benefit is that this allows the use of a successor oracle for the study of dynamic processes from a global point of view, independent from a particular initial state and the (complete) construction of the reachability graph. We provide a characterization, in terms of a local consistency condition, of those deterministic systems whose dynamic behavior can be encoded using our approach and consider the problem of recognizing when an orientation of the transition conflict graph is valid for this purpose . Finally, we address the problem of gaining the information that allows to provide an appropriate priority relation gouverning the dynamic behavior of the studied system and dicuss some further implications and generalizations of the studied approach.

1 Introduction

Petri nets constitute a well-established framework for modeling complex dynamic systems. Their broad application range includes the design of asynchronous hardware circuits [31], the analysis of production and workflow systems [2], the analysis and control of batch processes [9], the design of distributed algorithms for networks of agents [26], and the modeling and simulation of biological networks [10,11,21], to cite only some prominent examples.

Petri nets also turned out to be a flexible modeling framework that has been extended in various ways for dealing with different applications. For instance, colored and high-level Petri nets have been widely used for protocol specification in communication networks [5,14]. Stochastic Petri nets are used in cases where uncertainty is attached to input data, to describe external noise generated by the environment, as well as noise that might be intrinsic to a system [16,3]. Hybrid Petri nets allow to model systems were both continuous and discrete processes

M Heiner, AK Wagler (Eds.): BioPPN 2015, a satellite event of PETRI NETS 2015,
CEUR Workshop Proceedings Vol. 1373, 2015.

coexist [6], and the inclusion of additional features required for modeling certain biological networks has led to the definition of Hybrid Functional Petri nets [22]. In this paper we consider another possible extension that aims at representing systems whose dynamic behavior exhibits deterministic features.

More formally, a *network* $G = (P, T, A, w)$ reflects the involved components (like network elements, technical components, biological entities etc.) by places $p \in P$ and their interactions (like transformations, causal dependencies, chemical reactions etc.) by transitions $t \in T$, linked by weighted directed arcs. Each place $p \in P$ can be marked with an integral number of tokens defining a system state $x \in \mathbb{Z}_+^P$, and dynamic processes are represented by sequences of state changes, starting from an initial state x^0 and performed by consecutively switching or firing enabled transitions (see Section 2 for more details).

Usually, the dynamic behavior is the result of several conflicting as well as concurrent ongoing dynamic processes. A *Petri net* is a pair (G, x^0) consisting of a network G together with an initial state x^0, and its state space $\mathcal{X}(x^0)$ is understood as the set of all further system states which can be reached from x^0 by switching or firing sequences of transitions, see e.g. [25] for more information. In this framework, describing the dynamics of a system might be done by model animation, i.e., by simulating the flow of tokens inside the network as transitions are switched [12]. Given a network G and an initial state $x^0 \in \mathcal{X}$, some central problems that are usually studied in this context are:

- *Reachability*: Determine whether the system may eventually reach one of a set of target states after a finite sequence of transition switches.
- *Boundedness*: Determine if there are sequences of transition switches that lead to the accumulation of an unlimited number of tokens at some place of the network.
- *Existence of deadlocks*: Determine whether the system can reach a state at which no transitions are enabled.
- *Liveness*: Determine whether no sequence of transition switches can put the system into a state where some transition is permanently disabled.

The problems mentioned above are in general hard from a computational complexity point of view. For instance, *reachability* was proven to require exponential space [15] and decidability of this problem could only be established some years later [23,24].

Partial versus global point of view on a system. Dynamic processes can be encoded as directed paths in a state digraph $\mathcal{G}$ where nodes represent states and there is a directed arc between two states x, x' if x' can be obtained from x by switching a single transition. It is common practice to use the term *reachability* or *marking graph* to refer to the subgraph $\mathcal{G}(x^0)$ of $\mathcal{G}$ induced by the set $\mathcal{X}(x^0)$ of nodes corresponding to those states that can be reached from the initial state x^0 of the network.

Considering $\mathcal{G}(x^0)$ allows only a partial view on the studied system which is, e.g., suitable for a technical system performing exactly one process. However, already a failure of one or several components of such a system can change the initial state, and a more global view is required to coherently model both the normal and the mal function of the system, and to detect the faulty element(s) by an according failure analysis.

Similarly, the study of biological systems by performing experiments can be seen as putting the system in a particular initial state and observing the evolution of the system in terms of resulting sequences of state changes. Due to the intrinsic complexity of biological systems, the dynamic behavior of such systems can rarely be understood by performing a single experiment. In general, several experiments starting from different initial states are required and need to be coherently interpreted in a single model.

To study dynamic systems and processes therein from a more global point of view (independent from a particular initial state), we therefore suggest to consider the state digraph $\mathcal{G}$ on the potential state space $\mathcal{X}$ of the system, i.e., on the set of all theoretically possible states.

Non-deterministic versus deterministic systems. While studying dynamic processes, a particular situation occurs when so-called dynamic conflicts are present at states, in which two transitions are enabled, but switching one disables the other. In this case, model animation does not allow definite conclusions about any system properties, as mentioned in [11]. The reason is that the occurrence of dynamic conflicts is understood as alternative (branching) system behavior, where a decision between these alternatives is taken non-deterministically.

However, there are examples of systems that show a deterministic behavior despite the existence of dynamic conflicts. We call a dynamic system *deterministic* if any state $x \in \mathcal{X}$ has a unique successor state $\text{succ}(x)$. For technical systems, a deterministic behavior is often crucial in order to guarantee the reliability of the performed processes. In addition, also some biological systems are deterministic, as stimulating them in a certain way triggers always the same response (see e.g. the light-induced sporulation of *Physarum polycephalum plasmodia* or the phototaxis of halobacterial cells described in [18,17,19]). Petri nets, as originally defined, can be used to model such systems only in some trivial cases.

A compact encoding for deterministic systems. Our aim is to overcome the above mentioned difficulties by presenting a way to model deterministic systems from a global point of view (independent from a particular initial state), and to predict the systems behavior for all possible system states (whithout generating explicitely the state digraph). For that, we define a *successor function* $\mathsf{succ} : \mathcal{X} \to \mathcal{X}$ which returns $\text{succ}(x)$ for every $x \in \mathcal{X}$. An explicit encoding of succ, for instance pointwise, is about of the same size as the state digraph $\mathcal{G}$, and hence at least exponential in the size of G. Here we propose a more compact way of representing the dynamics of a deterministic system, by encoding the global dynamic behavior of the system through a local selection criterion that chooses among all currently enabled transitions the one which switches to the successor state $\text{succ}(x)$.

In the next section, we formally provide all definitions and concepts needed for Petri nets as models for deterministic systems. Our model relies on the definition of a priority relation between conflicting transitions, which is encoded in a compact manner by orienting the edges of a transition conflict graph. This is subject of Section 3, where the transition conflict graph is introduced together with suitable orientations of its edges in order to impose priorities among transitions as a selection criterion. This leads to a compact encoding of both a successor and a predecessor oracle. We also provide a characterization of the deterministic systems that can be encoded by our model. The issue of recognizing and characterizing such suitable orientations of the transition conflict graph is addressed in Section 4, while Section 5 is devoted to the problem of finding such orientations. We finally summarize all proposed concepts and their benefits and discuss some further implications and generalizations of the studied approach.

2 Petri nets with a deterministic behavior

In this section, we will formally provide all definitions and concepts needed for Petri nets as models for systems with a deterministic behavior.

Recall that a *network* $G = (P, T, A, w)$ is used to encode the structure of the underlying system, where the set P of places represents the system's components, the set T of transitions stands for their interactions, and the weighted directed arcs (p, t) or $(t, p) \in A$ exclusively connect places and transitions. A network $G = (P, T, A, w)$ can also be represented by its *incidence matrix* $M := (m_{pt}) \in \mathbb{Z}^{P \times T}$ where each row corresponds to a place $p \in P$ and each column to a transition $t \in T$. We have

$$m_{pt} := \begin{cases} -w_{pt} & \text{if } p \in P^-(t), \\ +w_{tp} & \text{if } p \in P^+(t), \\ 0 & \text{otherwise,} \end{cases}$$

where $P^-(t) := \{p \in P : (p, t) \in A\}$, denotes the set of *pre-places* of $t \in T$ and $P^+(t) := \{p \in P : (t, p) \in A\}$ the set of its *post-places*.

Some places $B \subseteq P$ may have bounded capacities, which are given by the positive integral vector $u \in \mathbb{Z}_+^B$. Each place $p \in P$ can be marked with an integral number x_p of (at most u_p) tokens, and any marking defines a state of the system that can be represented as an integral nonnegative vector $x \in \mathbb{Z}_+^P$. The *potential state space* of a capacitated network (G, u) is the set of all theoretically possible states $\mathcal{X} := \left\{x \in \mathbb{Z}_+^P : x_p \leq u_p,\ \forall p \in B\right\}$. It is at least exponential in $|P|$ and is finite if $B = P$ holds.

Dynamic processes are described as sequences $x^1, \ldots, x^k$ of consecutive system states[3], where state x^{i+1} is obtained from x^i by *switching* or *firing* a (single) transition $t \in T$. Thereby, t consumes w_{pt} tokens from each pre-place $p \in P^-(t)$ and produces w_{tp} new tokens on each post-place $p \in P^+(t)$. A transition $t \in T$ is *enabled* at a state $x \in \mathcal{X}$ if switching t yields a valid successor state $x + M_{\cdot t} \in \mathcal{X}$, where $M_{\cdot t}$ is the column of M associated with t, and *disabled* otherwise. Thus, $t \in T$ is enabled in $x \in \mathcal{X}$ if

[3] Throughout this paper, we will use superindices to reference different states and subindices to specify places. Thus, x_p^i is the number of tokens assigned to place p at state x^i.

E1 $x_p - w_{pt} \geq 0$ for all $p \in P^-(t)$ and
E2 $x_p + w_{tp} \leq u_p$ for all $p \in P^+(t) \cap B$.

Remark 1. An extended Petri net $(P, T, (A \cup A_R \cup A_I), w)$ is a Petri net which has, besides the (standard) arcs in A, two additional sets of so-called control-arcs: the set of read-arcs $A_R \subset P \times T$ and the set of inhibitor-arcs $A_I \subset P \times T$. In such a Petri net, switching of transitions is also controlled by read- and inhibitor-arcs: a transition $t \in T$ is enabled at $x \in \mathcal{X}$ if E1 and E2 are satisfied and it additionally holds that

E3 $x_p \geq w_{pt}$ for all p with $(p, t) \in A_R$, and
E4 $x_p < w_{tp}$ for all p with $(p, t) \in A_I$.

Note that also in an extended Petri net, switching of an enabled transition t at x yields $x + M_{\cdot t}$ as succesor state since control-arcs do not affect the marking.

In general, it is possible that more than one transition is enabled in a state $x \in \mathcal{X}$; we denote by $T(x)$ the set of all such transitions. Conversely,

$$X(t) := \left\{x \in \mathcal{X} : x_p \geq w_{pt},\, \forall p \in P^-(t);\;\; x_p \leq u_p - w_{tp},\, \forall p \in P^+(t) \cap B\right\}$$

denotes the set of states at which transition t is enabled.

Recall that the state digraph of a system is defined by $\mathcal{G} := (\mathcal{X}, \mathcal{A})$ where nodes represent potential states and a node x has an outgoing arc $(x, x') \in \mathcal{A}$ if and only if $x' = x + M_{\cdot t}$ for some $t \in T(x)$, i.e., if x' can be obtained from x by switching a single transition. We call $x \in \mathcal{X}$ a *branching state* if there are at least two transitions in $T(x)$. The existence of a branching state implies either an alternative or a concurrent behavior of the system.

An alternative behavior occurs when at least two transitions $t, t' \in T(x)$ are in *dynamic conflict*, i.e., if they cannot switch simultaneously as for some place $p \in P$, we have

$$x_p - w_{pt} - w_{pt'} < 0, \quad \text{or} \quad x_p + w_{tp} + w_{t'p} > u_p,$$

thus, if $x + M_{\cdot t} + M_{\cdot t'} \notin \mathcal{X}$ holds. While animating the model, *either t or t'* has to be selected at x. In the case of concurrency, on the contrary, no two transitions from $T(x)$ are in a dynamic conflict. All transitions in $T(x)$ could be switched simultaneously, resulting in the valid state

$$x' = x + \sum_{t \in T(x)} M_{\cdot t} \;\in \mathcal{X}.$$

In this case, $\mathcal{G}$ contains paths for all possible interleaving sequences between x and x', corresponding to all possible permutations of the transitions in $T(x)$.

Petri nets are an adequate model for simulating the dynamic behavior of deterministic systems, only if no branching states are present. Otherwise, a mechanism has to be specified that allows to decide "which is the right path to choose" in the state digraph. This includes in particular a way of resolving dynamic conflicts.

We call a Petri net *deterministic* if each state $x \in \mathcal{X}$ has a unique successor $\text{succ}(x) \in \mathcal{X}$. The dynamic behavior of a deterministic system can be specified in the form of a *successor-oracle* which returns, for every state $x \in \mathcal{X}$, the value of $\text{succ}(x)$. Our goal is to propose a compact implementation for this oracle. Hereby, we restrict our attention to systems where the successor-oracle can be expressed via a *transition selection function* $\mathsf{trans} : \mathcal{X} \to T$, which assigns to every state $x \in \mathcal{X}$ a unique transition $\mathsf{trans}(x) \in T(x)$ that must be switched in order to reach state $\text{succ}(x)$. Not all deterministic systems can be modeled in this way, see the discussion in Section 6, and even if it is possible, an explicit (i.e., pointwise) encoding of trans is extensive as it requires an amount of space proportional to the size of the state digraph $\mathcal{G}$, which, as pointed out in the previous section, is at least exponential in the number of places of the network.

In [20], it was proposed to use priorities between the transitions of the network as additional activation rules to determine which transition from $T(x)$ has to be selected as $\mathsf{trans}(x)$ in order to reach $\text{succ}(x)$. Our contribution consists in providing a compact scheme for encoding trans based on such priorities.

Remark 2. The priority relations proposed in [20] shall reflect the relative reaction rates of the (chemical) reactions represented by the transitions of the network with the idea that faster reactions have higher priorities and are taken. On the model side, priorities can be seen as a discrete extreme case of firing rates of transitions defined by probability distributions, where exactly one transition in $T(x)$ (namely, the highest-priority transition $\mathsf{trans}(x)$) has probability 1, and all other transitions in $T(x)$ have probability 0.

As we show in the next section, if a deterministic system $\mathcal{S} = (G, u, \mathsf{succ})$ satisfies a quite natural consistency condition, then there exists a digraph $\mathbb{D} := (T, \mathbb{A})$ on the set of transitions with the following properties:

– for every $x \in \mathcal{X}$, the set $T(x)$ induces a subgraph having a unique sink $t(x)$
– and $t(x) = \mathsf{trans}(x)$ is the required transition to switch from x to $\text{succ}(x)$.

Observe that this graph has a size of $O(|T|^2)$, which is polynomial in the size of G. Indeed, the arcs of $\mathbb{D}$ can be interpreted as priority relations between pairs of transitions, with $\mathsf{trans}(x)$ being the transition with highest priority among all transitions in $T(x)$.

3 Encoding valid priority relations

As observed in the previous section, one key issue for modeling the dynamic behavior of a deterministic system is the specification of a mechanism for the (unambiguous) resolution of dynamic conflicts between enabled transitions.

Definition 1. *Given a capacitated network $G = (P, T, A, w)$ with $u \in \mathbb{Z}_+^B$, its* transition conflict graph *is an undirected graph $\mathbb{K}_{(G,u)} = (T, \mathbb{E})$ having as nodes the transitions from G, where two transitions t, t' are joined by an edge if and only if there exists at least one state where both are enabled, i.e.,*

$$tt' \in \mathbb{E} \quad \Leftrightarrow \quad X(t) \cap X(t') \neq \emptyset.$$

As long as there is no risk of confusion, we will simply write $\mathbb{K}$ instead of $\mathbb{K}_{(G,u)}$. It follows straightforwardly from this definition that, for every state $x \in \mathcal{X}$, the set $T(x)$ of enabled transitions induces a clique in $\mathbb{K}$, i.e., a set of mutually adjacent nodes.

Remark 3. The converse is not necessarily true, as it is shown in [29]. However, as the sets $X(t)$ are boxes, it is straightforward to prove that at least the inclusionwise maximal cliques in $\mathbb{K}$ are associated with states of the system.

The transition conflict graph $\mathbb{K}$ can be constructed in $O(|P|\ |T|^2)$ time using a straightforward algorithm to check for box intersections $X(t) \cap X(t')$. The next lemma from [29] provides a more efficient way of its computation.

Lemma 1. *Consider a capacitated network (G, u), with $G = (P, T, A, w)$, $u \in \mathbb{Z}_+^B$ and $X(t) \neq \emptyset\ \forall t \in T$. Then $t, t' \in T$ are in conflict if and only if*

$$\begin{array}{ll} w_{pt} \leq u_p - w_{t'p} & \forall p \in P^-(t) \cap P^+(t') \cap B \textit{ and} \\ w_{pt'} \leq u_p - w_{tp} & \forall p \in P^-(t') \cap P^+(t) \cap B \textit{ holds.} \end{array}$$

Observe that, in the particular case where we have capacities $u = \mathbb{1}$, it follows as a corollary from the last lemma that two transitions $t, t' \in T$ are in conflict if and only if

$$P^-(t) \cap P^+(t') \cap B = \emptyset \quad \text{and} \quad P^-(t') \cap P^+(t) \cap B = \emptyset. \tag{1}$$

The following example illustrates the previous results.

Example 1. Consider the capacitated network $(G, \mathbb{1})$, with $G = (P \cup T, A, \mathbb{1})$, from Figure 1(a). Checking the pre- and post-places for all transitions in T yields:

t_i	$P^-(t_i)$	$P^+(t_i)$
t_1	$\{2\}$	$\{1\}$
t_2	$\{2\}$	$\{3\}$
t_3	$\{2\}$	$\{5\}$
t_4	$\{5\}$	$\{1, 4\}$
t_5	$\{2, 5\}$	$\{3\}$

According to (1), all pairs of transitions are in conflict, except (t_3, t_4) and (t_3, t_5), and we obtain the transition conflict graph shown in Figure 1(b).

As stated in the introduction, our main purpose is to encode the dynamics of a given deterministic system $\mathcal{S} = (G, u, \mathsf{trans})$ in a compact way, by introducing enough priority relations among transitions, as to be able to determine $\mathsf{trans}(x)$ for every $x \in \mathcal{X}$. We have observed that the set of enabled transitions at each state corresponds to a clique in the transition conflict graph $\mathbb{K}$, and hence $\mathbb{K}$ is a plausible candidate for a framework where these priorities could be embedded. This idea motivates the following definition.

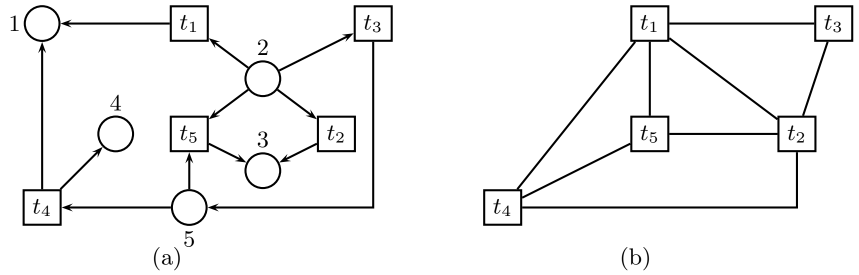


Fig. 1. (a) A capacitated network of a dynamic system, and (b) its transition conflict graph. All place capacities and arc weights are assumed to be equal to one.

Definition 2. *Let $\mathbb{D}_{(G,u)} = (T, \mathbb{A})$ be a directed graph obtained by orienting the edges of $\mathbb{K}_{(G,u)}$. $\mathbb{D}_{(G,u)}$ is* valid for the system $\mathcal{S} = (G, u, \mathsf{trans})$ *if, for every state $x \in \mathcal{X}$ with $T(x) \neq \emptyset$, the subgraph induced by the nodes from $T(x)$ has a unique sink, and this sink coincides with $\mathsf{trans}(x)$.*

As for the unoriented transition conflict graph, we will write in the following $\mathbb{D}$ instead of $\mathbb{D}_{(G,u)}$ whenever there is no risk of confusion. Observe that the existence of a valid orientation implicitly imposes a further requirement on the nature of a dynamic system. Namely, if two states $x, x^* \in \mathcal{X}$ have the same set $T(x) = T(x^*)$ of enabled transitions, then both induce the same subgraph of $\mathbb{D}$ and, therefore, $\mathsf{trans}(x) = \mathsf{trans}(x^*)$ must hold. Moreover, if a transition t is enabled at two states $x, x' \in \mathcal{X}$ and t is the highest-priority transition at x, then either t is also the highest-priority transition at x' or $\mathsf{trans}(x') \notin T(x)$. The next result from [29] shows that this condition is also sufficient for the existence of a valid orientation.

Theorem 1. *A system $\mathcal{S} = (G, u, \mathsf{trans})$ has a valid orientation if and only if the following* consistency condition *holds for any pair of states x and x': if $\mathsf{trans}(x) \in T(x) \cap T(x')$ then* either $\mathsf{trans}(x') = \mathsf{trans}(x)$ or $\mathsf{trans}(x') \in T(x') \setminus T(x)$.

As illustrated in Algorithm 1, a valid orientation completely encodes the dynamic behavior of a deterministic system, since it can be used to obtain an implicit implementation of the successor-oracle. Given a state $x \in \mathcal{X}$, all we need to do in order to find its successor is to determine the highest-priority transition $\mathsf{trans}(x)$ at x. This can be accomplished by searching for the unique sink in the subgraph of $\mathbb{D}$ induced by the set $T(x)$ of enabled transitions. (If no transitions are enabled at x, the algorithm just returns the same current state x as successor.)

A *predecessor* of x is a state $y \in \mathcal{X}$ with $\mathrm{succ}(y) = x$. For that, we must have $x = y + M_{\cdot \mathsf{trans}(y)}$. Since there are at most $|T|$ candidates for the transition $\mathsf{trans}(y)$, it follows that the set $\mathrm{pred}(x)$ of all possible predecessors of x has cardinality not larger than $|T|$ and can be constructed calling the successor oracle at most $|T|$ times, as shown in Algorithm 2. Here, first a set $\mathrm{cand}(x)$ of

```
Input: $(G, u, \mathbb{D})$, $x \in \mathcal{X}$
Output: $\mathrm{succ}(x)$
Construct the set $T(x)$ of enabled transitions
if $T(x) = \emptyset$ then
   return $x$
end if
{Compute out-degree of transitions in the subgraph of $\mathbb{D}$ induced by $T(x)$}
for $t \in T(x)$ do
   $\delta^-(t) := |\{tt' \in \mathbb{A} : t' \in T(x)\}|$
end for
{Return successor state}
$t^* \leftarrow t \in T(x)$ with $\delta^-(t) = 0$
return $x + M_{\cdot t^*}$
```

Algorithm 1: Implementing a successor-oracle by a valid orientation.

all candidate transitions $t \in T$ fulfilling $y := x - M_{\cdot t} \in \mathcal{X}$ is computed, then the condition $t = \mathsf{trans}(y)$ tested for each candidate by calling the successor oracle.

```
Input: $(G, u, \mathbb{D})$, $x \in \mathcal{X}$
Output: $\mathrm{pred}(x)$
{Construct set of transition candidates}
$\mathrm{cand}(x) \leftarrow \big\{t \in T : x_p + w_{pt} \leq u_p, \forall p \in P^-(t) \cap B;\;\; x_p - w_{tp} \geq 0, \forall p \in P^+(t)\big\}$
{Call successor-oracle to construct sets of predecessors}
$\mathrm{pred}(x) \leftarrow \emptyset$
for $t \in \mathrm{cand}(x)$ do
   $y \leftarrow x - M_{\cdot t}$
   if $\mathsf{trans}(y) = t$ then
      $\mathrm{pred}(x) \leftarrow \mathrm{pred}(x) \cup \{y\}$
   end if
end for
{Return set of possible predecessors}
return $\mathrm{pred}(x)$
```

Algorithm 2: An implementation for a predecessor-oracle.

Computing successors and sets of predecessors is a key operation within simulation algorithms to study the dynamic behavior of a system and in particular to address the different questions described in the previous section. To the best of our knowledge, currently no solution algorithm for these problems is known, which does not rely on storing explicit descriptions of the state digraph or some equivalent structure to compute $\mathrm{succ}(x)$ at every state x, which strongly limits the size of the networks that can be considered, due to the high memory requirements. Moreover, in certain application areas such as systems biology, the explicit values of $\mathrm{succ}(x)$ can only be determined by carrying out expensive and time-consuming wet lab experiments. Having, however, a valid orientation it is possible to determine or predict $\mathrm{succ}(x)$ for each state $x \in \mathcal{X}$.

4 Recognizing and characterizing valid orientations

Another issue of interest concerns the *recognition* of valid orientations: Given a capacitated network (G, u), and an orientation $\mathbb{D}$ of the edges in the transition conflict graph, determine whether $\mathbb{D}$ induces a valid dynamic behavior on G, in the sense that for every state $x \in \mathcal{X}$, the corresponding clique in $\mathbb{D}$ has a unique sink. Observe that a clique cannot have more than one sink, and that every clique without a sink contains a directed cycle. Hence, if $\mathbb{D}$ is acyclic then every clique of the graph has a unique sink, and we immediately obtain:

Observation 1 *Every acyclic orientation of* $\mathbb{K}$ *is valid.*

On the other hand, the next example shows that there are deterministic systems for which the valid orientation encoding the dynamic behavior contains directed cycles.

Example 2. Figure 2 depicts a network $(G, \mathbb{1})$ with $w = \mathbb{1}$ together with a valid orientation $\mathbb{D}$ of the transition conflict graph that contains a directed cycle $C = (t_1, t_2, t_3)$. Table 1 lists all branching states of the system and the corresponding sets of enabled transitions. It is straightforward to check that each of these sets induces a clique in $\mathbb{D}$ with a unique sink $t(x)$.

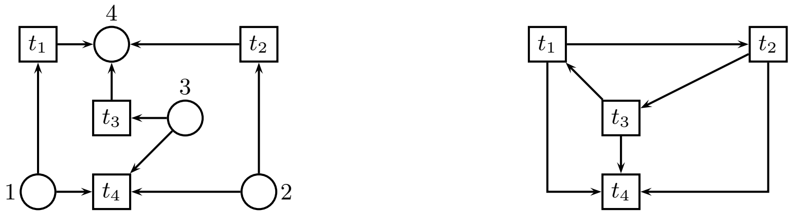


Fig. 2. A capacitated network and a valid orientation for its transition conflict graph which contains a directed cycle $C = (t_1, t_2, t_3)$. All arc weights and capacities of the places are assumed to be equal to one.

Table 1. All branching states of the system from Figure 2.

x_1	x_2	x_3	x_4	$T(x)$	$t(x)$
1	1	0	0	$\{t_1, t_2\}$	t_2
0	1	1	0	$\{t_2, t_3\}$	t_3
1	0	1	0	$\{t_1, t_3\}$	t_1
1	1	1	0	$\{t_1, t_2, t_3, t_4\}$	t_4

The fact that one has to check, for each clique Q in $\mathbb{K}$, if there is a corresponding state $x \in \mathcal{X}$ with $T(x) = Q$ makes the recognition of valid orientations hard in general, as there might be up to $2^{|T|}$ cliques. In the following we describe a special class of orientations for which this recognition problem can be solved efficiently.

In [29], it is shown that two different dynamic systems might share the same transition conflict graph. This motivated the definition of the following equivalence relation between capacitated networks

$$(G, u) \sim_{\mathbb{K}} (G', u') \quad \Leftrightarrow \quad \mathbb{K}_{(G,u)} \cong \mathbb{K}_{(G',u')},$$

where $\cong$ stands for graph isomorphism. An orientation is *strongly valid* if it is valid for all networks of some equivalence class of $\sim_{\mathbb{K}}$. Acyclic orientations are one specific example of strongly valid orientations. In general, any orientation where *every* clique has a unique sink is (trivially) strongly valid. Moreover, the converse is also true, as it follows from a result in [29] that each equivalence class contains a network with the property that every clique in $\mathbb{K}$ is associated to a state. This fact makes strongly valid orientations easy to recognize.

Theorem 2. *An orientation $\mathbb{D}$ of the transition conflict graph of a capacitated network (G, u) is strongly valid if and only if it does not contain any directed cycle of length 3.*

In this context, one is tempted to wonder whether it is possible to gain more insights on (strongly) valid orientations by exploring structural properties of $\mathbb{K}$. In [29], it is shown that for any undirected graph H, there is a network G having H as its transition conflict graph. Thus, in general, neither $\mathbb{K}$ nor $\mathbb{D}$ can admit particular graph-theoretical properties. Studying the problem however from a hypergraph-theoretical point of view as done in [30], opens the possibility of characterizing valid orientations further.

A hypergraph is a generalization of a graph in which it is possible to connect any number of nodes. Formally, a hypergraph is a pair $\mathcal{H} = (V, \mathcal{E})$ where V is a set of elements, called nodes or vertices, and $\mathcal{E}$ is a family of non-empty subsets of V, called hyperedges. We further use two other combinatorial structures related to hypergraphs. The dual $\mathcal{H}^*$ of $\mathcal{H}$ is a hypergraph whose nodes and hyperedges are interchanged, so that the nodes are given by all $E_i \in \mathcal{E}$ and there is one hyperedge $V_v = \{E_j | v \in E_j\}$ for each $v \in V$. The intersection graph $G(\mathcal{H})$ of $\mathcal{H}$ is a graph whose nodes represent the hyperedges of $\mathcal{H}$, and two nodes are joined by an edge if and only if the corresponding hyperedges intersect.

Recall that, by definition, for every state $x \in \mathcal{X}$, the set $T(x)$ of enabled transitions induces a clique in $\mathbb{K}$, whereas the converse is not necessarily true (but at least the inclusion-wise maximal cliques in $\mathbb{K}$ are associated with states of the system), see Remark 2.

The equivalence relation on $\mathcal{X}$, where $x \sim x'$ iff $T(x) = T(x')$, partitions the state space into $r \leq 2^{|T|}$ equivalence classes $\mathcal{X}_1, \ldots, \mathcal{X}_r$ of states that share the same sets of enabled transitions. Let $\tilde{x}^i \in \mathcal{X}_i$ with $1 \leq i \leq r$ be (arbitrarily chosen) representative elements for each of these classes, and define the *transition* *hypergraph* $\mathcal{H}_T = (T, \mathcal{E}_T)$ *of* G to be the hypergraph on the set T of transitions, whose family $\mathcal{E}_T$ of hyperedges is given by $\mathcal{E}_T = \left\{T(\tilde{x}^i) : 1 \leq i \leq r\right\}$. Thus, the size of $\mathcal{H}_T$ is exponential in the number of transitions of the network, but it is always finite, as there are at most $2^{|T|}$ different subsets of T.

The *state hypergraph of* G is the dual hypergraph $\mathcal{H}_{\tilde{\mathcal{X}}}$ of $\mathcal{H}_T$ and has $\tilde{\mathcal{X}} = \left\{\tilde{x}^1, \ldots, \tilde{x}^r\right\}$ as node set, where its family of hyperedges is determined by $\mathcal{E}_{\tilde{\mathcal{X}}} = \left\{X(t) \cap \tilde{\mathcal{X}} : t \in T\right\}$. Directly from the definition of $\mathcal{H}_{\tilde{\mathcal{X}}}$ we deduce:

Lemma 2. *The intersection graph* $G(\mathcal{H}_{\tilde{\mathcal{X}}})$ *of the state hypergraph* $\mathcal{H}_{\tilde{\mathcal{X}}}$ *is exactly the transition conflict graph* $\mathbb{K}$*.*

A hypergraph $\mathcal{H} = (V, \mathcal{E})$ has the *Helly property* if, for any family $\mathcal{E}' \subseteq \mathcal{E}$ of pairwise intersecting hyperedges, there exists a node $v \in V$ contained in all hyperedges from $\mathcal{E}'$. A well-known result in hypergraph theory [4] states that $\mathcal{H}$ has the Helly property if and only if the inclusion-wise maximal hyperedges of its dual hypergraph $\mathcal{H}^*$ are precisely the inclusion-wise maximal cliques of the intersection graph $G(\mathcal{H})$ of $\mathcal{H}$ [4]. This implies:

Lemma 3. $\mathcal{H}_{\tilde{\mathcal{X}}}$ *satisfies the Helly property.*

A direct consequence from Lemma 3 and the preceding observations is obtained in [30]:

Theorem 3. *The inclusion-wise maximal hyperedges from* $\mathcal{E}_T$ *are exactly the inclusion-wise maximal cliques of* $\mathbb{K}$*. Thus, for every inclusion-wise maximal clique* Q *there is some state* $\tilde{x}^i \in \tilde{\mathcal{X}}$*,* $1 \leq i \leq r$*, satisfying* $T(\tilde{x}^i) = Q$*.*

We next show that $\mathbb{K}$ has an interesting property, provided that $\mathcal{H}_T$ and $\mathcal{H}_{\tilde{\mathcal{X}}}$ belong to certain classes of hypergraphs. A *cycle* of length k in a hypergraph $\mathcal{H} = (V, \mathcal{E})$ is a sequence $(v_1, E_1, v_2, E_2, \ldots, v_k, E_k, v_{k+1})$ such that $v_i \in V$ for all $1 \leq i \leq k+1$, $E_i \in \mathcal{E}$ for all $1 \leq i \leq k$, $E_i \neq E_j$ if $i \neq j$, $v_i, v_{i+1} \in E_i$ for all $1 \leq i \leq k$, and $v_{k+1} = v_1$. Due to [1], a hypergraph $\mathcal{H}$ is *acyclic* if and only if $\mathcal{H}$ is conformal (i.e., its dual $\mathcal{H}^*$ has the Helly property) and for every cycle of length at least 3 in $\mathcal{H}$, some edge of $\mathcal{H}$ contains at least 3 nodes of the cycle. On the other hand, a hypergraph $\mathcal{H}^* = (V^*, \mathcal{E}^*)$ is called *arboreal* if there exists a tree T^* on the node set V^* such that every hyperedge of $\mathcal{H}^*$ induces a subtree in T^*. Due to structural characterizations in [7,8,27], arboreal hypergraphs are dual to acyclic hypergraphs and their intersection graphs are *chordal*, i.e. each cycle having four or more nodes contains a chord: an edge joining two nodes that are not adjacent in the cycle. Moreover, it can be shown that arboreal hypergraphs have the Helly property.

Combining all these properties, the following result is obtained in [30].

Theorem 4. *The following assertions are equivalent:*

- $\mathcal{H}_T$ *is acyclic.*
- $\mathcal{H}_{\tilde{\mathcal{X}}}$ *is arboreal.*
- $\mathbb{K}$ *is chordal.*

Note that a chord (t, t') within a directed cycle C in $\mathbb{D}$ induces a new directed cycle with a smaller number of nodes, as C contains both a directed path from t to t' and a directed path from t' to t. Hence, an orientation of a chordal graph contains a directed cycle of length 3 if and only if it contains any directed cycle. This observation was used in [30] to obtain a new characterization of acyclic transition hypergraphs:

Theorem 5. *The transition hypergraph $\mathcal{H}_T$ is acyclic if and only if, for any orientation $\mathbb{D}$ of $\mathbb{K}$, the following two statements are equivalent:*

1. *$\mathbb{D}$ is strongly valid.*
2. *$\mathbb{D}$ is acyclic.*

5 Finding valid orientations

In the previous sections, we addressed the problem of the existence of a valid orientation and how to recognize and characterize such orientations. This motivates a canonical further question, namely, once the existence of a valid orientation has been confirmed for a deterministic system, can we also find it?

As pointed out in [28], infering a valid orientation of a transition conflict graph shall be done by taking observations on dynamic processes in the underlying deterministic system into account. That is: Based on the knowledge of the values of $\mathrm{succ}(x)$ at *some* states $x \in \mathcal{X}$, we aim at finding a valid orientation that encodes the global dynamic behavior of the system since $\mathrm{succ}(x)$ can be determined with the help of this orientation for *all* states $x \in \mathcal{X}$.

In the following we consider a deterministic system $\mathcal{S} = (G, u, \mathsf{trans})$ that fulfills the consistency condition from Theorem 1 and hence trans can be encoded as a valid orientation $\mathbb{D}$ of the transition conflict graph. Given as input the capacitated network (G, u) and an *oracle* for returning the value of $\mathsf{trans}(x)$ at any state $x \in \mathcal{X}$, we want to determine $\mathbb{D}$ by calling the oracle as few times as possible since in practice, a call to the oracle stands for the execution of a (probably expensive and time-consuming) experiment.

We call a set $\mathcal{X}' \subseteq \mathcal{X}$ of states a *valid test set* if the information about the corresponding highest-priority transitions $\{\mathsf{trans}(x) : x \in \mathcal{X}'\}$ suffices for inferring the direction of all arcs in $\mathbb{D}$. Consequently, we are interested in the following problem, formulated in [28].

Definition 3 (Minimum Valid Test Set Problem (MVTP)). *Given a deterministic system $\mathcal{S}$ together with an oracle for returning highest-priority transitions, find a valid test set of minimum cardinality.*

The meaning of *inferring* an orientation deserves further explanation. Let us denote by *partial orientation* a (mixed) graph $\mathbb{D}' := (T, \mathbb{A}', \mathbb{E}')$ obtained by fixing the direction for *some* edges of $\mathbb{K}$, with $\mathbb{A}'$ being the set of the corresponding oriented arcs, and $\mathbb{E}'$ the set of the remaining unoriented edges. A partial orientation is *extendible* if it is possible to choose directions for all unoriented edges in such a way that a valid orientation is obtained. Given an extendible partial orientation $\mathbb{D}' := (T, \mathbb{A}', \mathbb{E}')$, a yet unoriented edge $tt' \in \mathbb{E}'$ is said to be *inferable as* (t, t') if the partial orientation $\hat{\mathbb{D}}' := (T, \mathbb{A}' \cup \{(t', t)\}, \mathbb{E}' \setminus \{tt'\})$ is not extendible. Moreover, $\mathbb{D}'$ is *sufficient* for inferring a valid orientation $\mathbb{D} := (T, \mathbb{A})$ if all edges in $\mathbb{E}'$ are inferable as the corresponding arcs in $\mathbb{A}$.

Example 3. Figure 3 illustrates these concepts. The partial orientation depicted in (a) is not extendible, as choosing any direction for edge t_2t_3 produces one inclusionwise maximal clique without sink. In contrast, any direction chosen for this edge in (b) leads to a valid orientation, so t_2t_3 is not inferable in the second example. A sufficient partial orientation is shown in (c): the two unoriented edges t_2t_3 and t_3t_4 *must* be oriented as (t_3, t_2) and (t_3, t_4). Orienting any of them in the opposite direction leads to a non-extendible partial orientation.

This shows that the optimal solution of MVTP may depend on the underlying valid (complete) orientation $\mathbb{D}$, and not on the transition conflict graph alone.

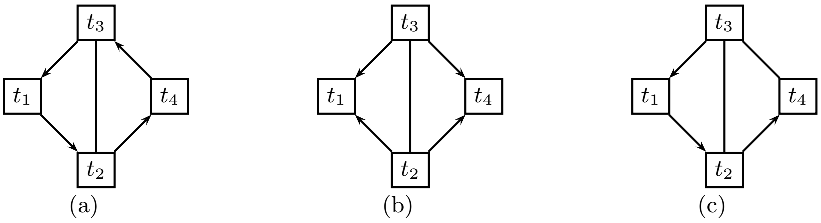


Fig. 3. Different types of partial orientations: (a) non-extendible; (b) extendible, but non-sufficient; and (c) sufficient

In general, recognizing if a partial orientation is extendible, or if an unoriented edge is inferable constitutes a difficult task, due to the lack of a characterization of valid orientations from a graph theoretical point of view. In this section we consider the particular case when $\mathbb{D}$ is acyclic. As mentioned earlier, acyclic orientations are always (strongly) valid and the *only* strongly valid orientations for deterministic dynamic systems with acyclic transition hypergraph $\mathcal{H}_T$ (see Theorem 5). Trivially, if $\mathbb{D}$ is acyclic then any partial orientation cannot contain a directed cycle. On the other hand, in [28] it is proved the following:

Lemma 4. *Any acyclic partial orientation is extendible.*

Recall from Theorem 2 that an orientation is strongly valid if and only if it does not contain a directed cycle of length three. Hence, for any extendible partial orientation $\mathbb{D}' := (T, \mathbb{A}', \mathbb{E}')$, if $(t, t'), (t', t'') \in \mathbb{A}'$ and $tt'' \in \mathbb{E}'$ then tt'' is inferable as (t, t'').

It follows from the previous lemma that an edge $tt' \in \mathbb{E}'$ in an acyclic partial orientation $\mathbb{D}' = (T, \mathbb{A}', \mathbb{E}')$ can be inferred as (t, t') if and only if the digraph $(T, \mathbb{A}')$ contains a directed path from t to t'. As a consequence, $\mathbb{D}'$ is sufficient if for every $tt' \in \mathbb{E}'$ the digraph $(T, \mathbb{A}')$ contains either a directed path from t to t', or a directed path from t' to t. We say in this case that $\mathbb{D}'$ has the *path-property.*

Conversely, given a (complete) orientation $\mathbb{D} = (T, \mathbb{A})$, an arc $a := (t, t') \in \mathbb{A}$ is called *essential* if the digraph $(T, \mathbb{A} \setminus \{a\})$ does not contain a directed path from t to t'. Observe that the direction of a cannot be inferred in any partial orientation extendible to $\mathbb{D}$. Hence, any sufficient partial orientation must contain all essential arcs from $\mathbb{D}$. The next result from [28] shows that no further arcs are required.

Theorem 6. *Let $\mathbb{D} = (T, \mathbb{A})$ be a valid orientation and $\mathbb{A}^* \subseteq \mathbb{A}$ the set of essential arcs. The partial orientation $\mathbb{D}^* := (T, \mathbb{A}^*, \mathbb{E}^*)$, where $\mathbb{E}^*$ is the set of edges from $\mathbb{K}$ corresponding to the arcs in $\mathbb{A} \setminus \mathbb{A}^*$, is sufficient for inferring $\mathbb{D}$.*

Observe that the set $\mathbb{A}^*$ is unique, and that it can be computed from $\mathbb{D}$ in $O(m^2)$ time complexity, with $m := |\mathbb{A}|$, for example by several runnings of a breadth-first search algorithm to determine if each arc of $\mathbb{A}$ is essential or not.

Determining the direction of essential arcs is specially important for reconstructing valid orientations. The following lemma from [28] implies that any valid test set has cardinality larger than or equal to the number of essential arcs in $\mathbb{D}$.

Lemma 5. *For any state $x \in \mathcal{X}$, knowledge of the highest-priority transition from $T(x)$ can be used to orient at most one essential arc.*

As a consequence, the following result is obtained in [28].

Theorem 7. *A valid test set $\mathcal{X}' \subseteq \mathcal{X}$ is optimal for MVTP if and only if $A(x) \cap \mathbb{A}^* \neq \emptyset$ for every $x \in \mathcal{X}'$, where $A(x) := \{(t, \mathsf{trans}(x)) : t \in T(x),\, t \neq \mathsf{trans}(x)\}$ is the set of arcs whose directions are revealed by testing the system at x.*

Note that such an optimal valid test set can be easily determined if $\mathbb{D}$ is known. However, in the setting of the MVTP it is not possible to devise a "winning strategy" that guarantees that each $A(x)$ contains an essential arc, as the next example shows.

Example 4. Consider the conflict graph given in Figure 5(a). Observe that $\mathbb{K}$ contains three cliques of size 3 (triangles) and seven edges, and that every edge is contained in one triangle. It can be checked that for any edge $e \in \mathbb{E}$, there exists a valid orientation where e is not essential. Hence, any winning strategy for MVTP must start by querying the oracle at a state $x^* \in \mathcal{X}$ related to one of the cliques of size 3. But for any of these cliques, there is a valid orientation where $A(x^*)$ contains no essential arc. Indeed, if $T(x^*) = \{t_1, t_2, t_3\}$ or $T(x^*) = \{t_1, t_3, t_4\}$, choose the orientation from Figure 5(c). Otherwise, if $T(x^*) = \{t_1, t_4, t_5\}$, choose the orientation from Figure 5(b).

Even if it is not possible to devise a "winning strategy" that guarantees to orient an essential arc with each oracle call in general, a valid orientation can be obtained by Algorithm 3, provided that the underlying system has one state associated with each clique of the transition conflict graph $\mathbb{K} = (T, \mathbb{E})$. Observe that in this case the obtained orientation is strongly valid.

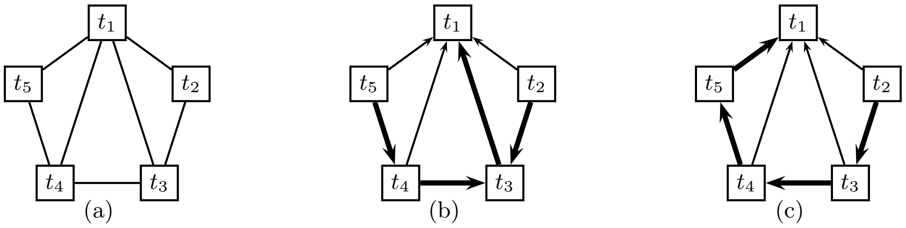


Fig. 4. A transition conflict graph (a) and two valid acyclic orientations (b)-(c). Essential arcs are marked bold.

```
Input: $\mathbb{K}$ {transition conflict graph}
$\mathcal{Q} = \{Q_1, \ldots, Q_k\}$ {partition of the edges into cliques}
Output: $\mathbb{D}$ {strongly valid orientation}
for $i \leftarrow 1, \ldots, k$ do
  while $Q_i$ contains more than one node do
    determine $x \in \mathcal{X}$ with $T(x) = Q_i$
    call oracle and determine $\mathsf{trans}(x)$
    orient arcs $\{(t, \mathsf{trans}(x)) : t \in T(x),\ t \neq \mathsf{trans}(x)\}$
    remove $\mathsf{trans}(x)$ from $Q_i$
  end while
end for
```

Algorithm 3: Inferring a strongly valid orientation.

Algorithm 3 takes as input a partition of the edges in $\mathbb{E}$ into a set of cliques $\mathcal{Q} := \{Q_1, \ldots, Q_k\}$. At each iteration of the outer loop, it processes a clique $Q_i \in \mathcal{Q}$. At first, the oracle is called to orient all edges incident to the unique sink node of Q_i. Then this node is removed from Q_i and the oracle is called again on the remaining subclique. The procedure is repeated until Q_i contains only one node, which means that all its edges have been oriented. Since the inner loop is executed $|Q_i| - 1$ times, a total of $\sum_{i=1}^{k} |Q_i| - k$ calls to the oracle are required to find the valid orientation $\mathbb{D}$. This quantity is strictly smaller than m if $|Q_i| \geq 3$ holds for at least some $Q_i \in \mathcal{Q}$, since then there is at least one iteration where two edges are oriented simultaneously, and no edge is oriented more than once.

This indeed ensures:

Theorem 8. *If the underlying system has one state associated with each clique of the transition conflict graph, then Algorithm 3 computes a strongly valid orientation.*

6 Discussion

Petri nets constitute a well-established framework for modeling complex dynamic systems. It is common practice to study dynamic processes in terms of directed paths in the reachability or marking graph $\mathcal{G}(x^0)$ starting from a designated initial state x^0. Hereby, a particular situation occurs when dynamic conflicts are present at states, in which two transitions are enabled, but switching one disables the other. In this case, model animation does not allow definite conclusions about any system properties [11]. The reason is that the occurrence of dynamic conflicts is understood as alternative (branching) system behavior, where a decision between these alternatives is taken non-deterministically. Therefore, the classical reachability analysis has the drawbacks that it

- explores the studied system only from a local point of view, starting from a particular initial state x^0;
- is only applicable for systems of a limited size since the reachability graph $\mathcal{G}(x^0)$ needs to be explicitly constructed, but grows exponentially in the size of the network;
- cannot be used to study systems that exhibit a deterministic behavior.

Our aim was to overcome these difficulties by presenting a way to compactly model deterministic systems from a global point of view (independent from a particular initial state), and to predict the systems behavior for all possible system states (whithout generating explicitly the state digraph). For that, we have examined a new approach for encoding the dynamic behavior of certain deterministic discrete systems that relies on extending the familiar framework of Petri nets. Our encoding consists in a realization of the successor-oracle as a valid orientation of the edges of the transition conflict graph $\mathbb{K}$, together with Algorithm 1. This encoding is compact in the sense that the amount of space required for its storage is polynomial in the size of the network. Therefore, it is well-suited for being integrated into simulation algorithms like [12] to study the dynamics of large complex deterministic systems, and to address issues such as reachability, boundedness, existence of deadlocks, and liveness, among others.

A possible interesting extension of our model concerns the study of dynamic systems where the concurrent switch of various transitions can occur. Throughout this paper we have assumed that a change from a state $x \in \mathcal{X}$ to its successor state succ(x) can always be explained by the switch of a *single* transition $\mathsf{trans}(x)$. However, there are deterministic systems where this does not hold, as shown in the following example.

Example 5. Consider a deterministic system $(G, \mathbb{1}, \mathsf{succ})$ with network G as depicted in Figure 5 and $w = \mathbb{1}$. It is straightforward to check that the system has the four branching states $x^0, \ldots, x^3$ shown in the figure. The following table lists the sets of enabled transitions at each of these states, as well as the corresponding successor states specified by succ:

j	branching state x^j	$T(x^j)$	succ(x^j)
0	$(1,1,0,0)^T$	$\{\mathbf{t_1}, \mathbf{t_2}\}$	$(0,0,1,1)^T$
1	$(0,1,1,0)^T$	$\{t_2, \mathbf{t_3}\}$	$(0,1,0,1)^T$
2	$(1,0,0,1)^T$	$\{t_1, \mathbf{t_4}\}$	$(0,1,0,0)^T$
3	$(1,1,1,0)^T$	$\{t_2, \mathbf{t_3}\}$	$(1,1,0,1)^T$

Observe that t_1 and t_2 have to be switched *concurrently* at state x^0 to reach $\mathrm{succ}(x^0)$, whereas at each of the other branching states a single transition is switched to reach the corresponding successor state, namely, t_3 at x^1 or x^4, and t_4 at x^2. The enabled transitions are highlighted in gray in Figure 5, while the transitions actually switched are marked in bold.

The above successor function *cannot* be realized with the help of a transition selection function trans due to the following reason. Considering x^0 as initial state, trans has to select *one* of the transitions $t_1, t_2 \in T(x^0)$. However, any choice leads to an intermediate state (x^1 for $\mathsf{trans}(x^0) = t_1$, x^2 for $\mathsf{trans}(x^0) = t_2$) where the remaining transition from $T(x^0)$ is still enabled but must *not* be selected, as both $\mathsf{trans}(x^1) = t_3$ and $\mathsf{trans}(x^2) = t_4$ are implied by the specification of succ. (See the reachability graph in Figure 5.) Thus, none of the potential interleaving sequences between x^0 and $\mathrm{succ}(x^0)$ can be expressed by means of single transition switches such that every switch is determined by the value of trans at the corresponding state.

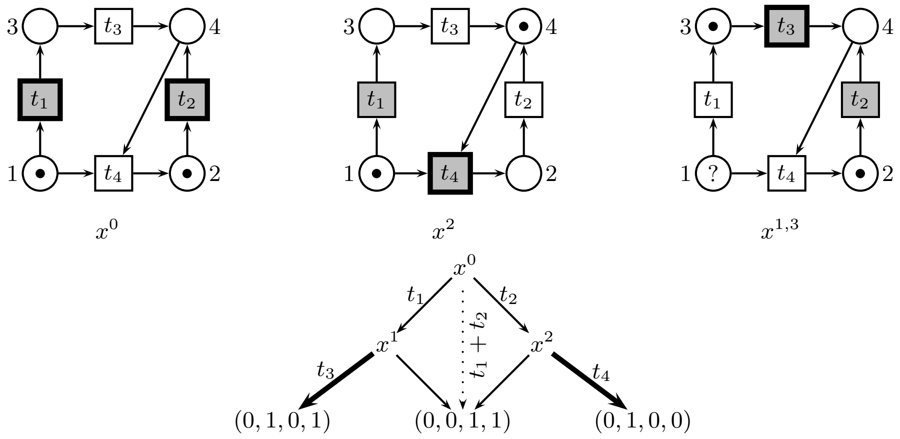


Fig. 5. A deterministic system whose dynamic behavior cannot be modeled via a transition selection function. The enabled transitions at each state are highlighted in gray, the next transitions to be switched are shown in bold. All place capacities and arc weights are assumed to be equal to one.

One way of dealing with these systems could be through the inclusion of pseudo-nodes in the transition conflict graph, to account for the simultaneous switching of various transitions. The advantage would be that all results from the previous sections directly carry over to the extended setting. The disadvantage of such a modification is, however, the dramatic increase in the size of the transition conflict graph, as (in the worst-case) the number of required pseudo-nodes may grow exponentially with respect to $|T|$.

An alternative approach consists in working on a slightly different transition conflict graph, where an edge between two transitions means that they are in dynamic conflict (i.e. when switching one transition disables the other). In this case, $T(x)$ does not longer induce a clique in $\mathbb{K}$. Yet we can now define an orientation to be valid if for any state $x \in \mathcal{X}$ the set $T(x)$ of enabled transitions induces a subgraph containing *at least one* sink. These sinks reveal an *anti-chain* of transitions with maximal priorities, which have to be switched concurrently. Theorem 1 can be generalized to this new setting in a straightforward manner, characterizing which deterministic systems admit valid orientations. The advantage of this scheme is that it is still compact with respect to the size of the network G. However, as the sets $T(x)$ do not induce cliques in the conflict graph, many of the combinatorial properties pointed out in Section 3-5 do not hold anymore. Moreover, since all transitions that *may* switch concurrently are *required* to do so, it is again possible to construct examples of deterministic systems whose behavior cannot be modeled in this way. Hence, further research is necessary to ensure a compact encoding of a successor oracle for all deterministic systems. For a large number of such systems, however, the here presented results allow us already the use of such an oracle for the study of dynamic processes from a global point of view, independent from a particular initial state and the (complete) construction of the reachability graph.

References

1. Acharya, B.D., Las Vergnas, M.: Hypergraphs with cyclomatic number zero, triangulated hypergraphs and an inequality. J. Combin. Theory (B) **33**, 52–56 (1982)
2. Adam, N.R., Atluri, V., Huang, W.K.: Modeling and analysis of workflows using Petri nets. J. Intell. Inf. Syst. **10**(2), 131–158 (1998).
3. Balbo, G.: Introducation to stochastic Petri nets. In: Lectures on formal methods and performance analysis: first EEF/Euro summer school on trends in computer science, pp. 84–155. Springer-Verlag New York, Inc., New York, NY, USA (2002)
4. Berge, C., Duchet, P.: A generalisation of Gilmore's theorem. In: M. Fiedler (ed.) Recent Advances in Graph Theory, pp. 49–55. Acad. Praha, Prague (1975)
5. Billington, J., Diaz, M., Rozenberg, G.: Application of Petri Nets to Communication Networks, Advances in Petri Nets. Springer-Verlag, London, UK (1999)
6. David, R., Alla, H.: Discrete, Continuous, and Hybrid Petri Nets. Springer-Verlag Berlin Heidelberg, Heidelberg (2005)
7. Duchet, P.: Propriété de Helly et problèmes de représentations. In: Problèmes Combin. et Théorie des Graphes, Coll. Orsay 1976, pp. 117–118. CNRS Paris (1978)
8. Flament, C.: Hypergraphes arborés. Discrete Math. **21**, 223–226 (1978)
9. Gu, T., Bahri, P.A.: A survey of Petri net applications in batch processes. Comput. Ind. **47**(1), 99–111 (2002).
10. Hardy, S., Robillard, P.N.: Modeling and simulation of molecular biology systems using Petri nets: modeling goals of various approaches. Journal of Bioinformatics and Computational Biology **2**(4), 619–637 (2004)
11. M. Heiner, D. Gilbert, R. Donaldson: Petri nets for systems and synthetic biology, In proceedings of SFM 2008, Springer, LNCS 5016:215–264, 2008.
12. M. Heiner, M. Herajy, F. Liu, C. Rohr, M. Schwarick: Snoopy – a unifying Petri net tool. In proceedings of PETRI NETS 2012, Hamburg, Springer, LNCS 7347:398–407, 2012.
13. M. Heiner, C. Rohr, M. Schwarick: MARCIE - Model checking And Reachability analysis done effiCIEntly; In proceedings of PETRI NETS 2013, Milano, Springer, LNCS 7927:389–399, 2013.

14. Jensen, K.: Coloured Petri nets: basic concepts, analysis methods and practical use, volume 3. Springer-Verlag New York, Inc., New York, NY, USA (1997)
15. Lipton, R.: The reachability problem requires exponential space. Research Report 62, Yale University, Computer Science Dept. (1976)
16. Marsan, M., Balbo, G., Donatelli, S., Franceschinis, G., Conte, G.: Modelling with generalized stochastic Petri nets. Wiley Series in Parallel Computing (1995)
17. Marwan, W.: Theory of time-resolved somatic complementation and its use for the analysis of the sporulation control network of physarum polycephalum. Genetics **164**, 105–115 (2003)
18. Marwan, W., Starostzik, C.: The sequence of regulatory events in the sporulation control network of physarum polycephalum analysed by time-resolved somatic complementation of mutants. Protist **153**, 391–400 (2002)
19. Marwan, W., Sujatha, A., Starostzik, C.: Reconstructing the regulatory network controlling commitment and sporulation in physarum polycephalum based on hierarchical Petri net modeling and simulation. J. Th. Biology **236**, 349–365 (2005)
20. Marwan, W., Wagler, A., Weismantel, R.: A mathematical approach to solve the network reconstruction problem. Math. Meth. Oper. Research **67**, 117–132 (2008)
21. W. Marwan, A. Wagler, R. Weismantel: Petri nets as a framework for the reconstruction and analysis of signal transduction pathways and regulatory networks, Natural Computing **10**, 639–654 (2011)
22. Matsuno, H., Aoshima, H., Doi, A., Tanaka, Y., Matsui, M.: Biopathways representation and simulation on hybrid functional Petri net. In Silico Biology **3**(3), 389–404 (2003)
23. Mayr, E.W.: An algorithm for the general Petri net reachability problem. In: Proceedings of the 13th Ann. ACM Symposium on Theory of Computing, pp. 238–246. ACM Press (1981)
24. Mayr, E.W.: An algorithm for the general Petri net reachability problem. SIAM J. Comput. **13**(3), 441–460 (1984)
25. Reisig, W.: Petri nets: an introduction. Springer-Verlag New York, Inc., New York, NY, USA (1985)
26. Reisig, W.: Elements of distributed algorithms: modeling and analysis with Petri nets. Springer-Verlag New York, Inc., New York, NY, USA (1998)
27. Slater, P.J.: A characterization of soft hypergraphs. Canad. Math. Bull. **21**, 335–337 (1978)
28. Torres, L.M., Wagler, A.: Model reconstruction for discrete deterministic systems. Electronic Notes of Discrete Mathematics **36**, 175–182 (2010)
29. Torres, L.M., Wagler, A.: Encoding the dynamics of deterministic systems. Mathematical Methods of Operations Research **73**(3), 281–300 (2011)
30. Torres, L.M., Wagler, A.: Analyzing the dynamics of deterministic systems from a hypergraph theoretical point of view. RAIRO Oper. Research **47**, 321–330 (2013)
31. Yakovlev, A., Koelmans, A., Semenov, A., Kinniment, D.: Modelling, analysis and synthesis of asynchronous control circuits using Petri nets. Integr., VLSI J. **21**(3), 143–170 (1996).

Petri nets for modelling and analysing trophic networks

Paolo Baldan[1], Martina Bocci[2], Daniele Brigolin[2], Nicoletta Cocco[2], and Marta Simeoni[2]

[1] Dipartimento di Matematica, Università di Padova, Italy
baldan@math.unipd.it
[2] Dipartimento di Scienze Ambientali, Informatica e Statistica,
Università Ca' Foscari di Venezia, Italy
{martina.bocci,brigo,cocco,simeoni}@unive.it

Abstract. We consider trophic networks, a kind of networks used in ecology to represent feeding interactions (what-eats-what) in an ecosystem. We observe that trophic networks can be naturally modelled as Petri nets and this suggests the possibility of exploiting Petri nets for the analysis and simulation of trophic networks. Some preliminary steps in this directions and some ideas for future development are presented.

1 Introduction

Ecosystems are very complex systems constituted by biotic communities (populations of different species), abiotic components of the environment (like air, water, soil) and interactions among these (living and non-living) elements. A branch of ecology deals with the study of feeding relationships within ecosystems and represents them as networks of interacting compartments called *trophic networks* or *food webs*. Due to the common limited availability of experimental information, a static approach (the mass balance steady state approach) to the study of such networks has been developed as alternative to the dynamic description.

Complex networks of interacting entities are widely studied in computer science: computer networks, agent systems, and, in general, all concurrent and distributed systems fall into this category. Uncountably many formalisms and practical tools have been developed for the representation and analysis of interacting systems. This suggests the possibility of reusing models and techniques from computer science for the study of trophic networks.

This idea is pursued in [17], where the authors advocate the use of process calculi for ecological modelling. Their claim is that the compositionality properties of process calculi can be fruitfully exploited for a modular representation of complex ecosystems. Moreover, process calculi provide an individual based modelling and stochastic extensions.

In this paper we explore the use of another widely used model of concurrency, namely Petri nets [19, 11]. Petri nets permit individual based modelling, they explicitly represent parallelism and dependencies among entities, they offer

M Heiner, AK Wagler (Eds.): BioPPN 2015, a satellite event of PETRI NETS 2015,
CEUR Workshop Proceedings Vol. 1373, 2015.

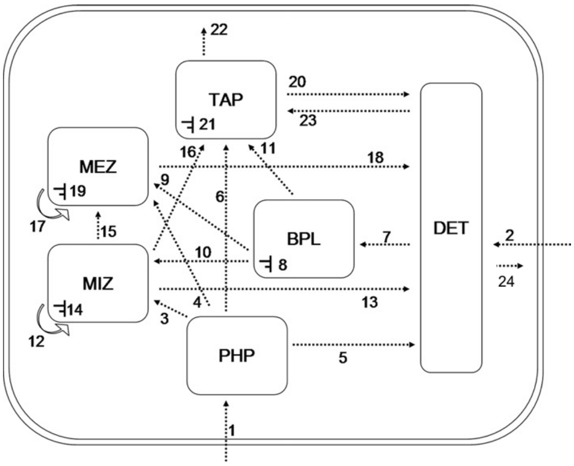
stochastic and continuous extensions and, as a major advantage, they enable a qualitative analysis of systems when dynamic information are not available. Many tools for systems visualisation, analysis and simulation are also available (see *The Petri net World site* [21]). In this paper we consider the representation and analysis techniques generally adopted for trophic networks and discuss the pros and cons of the application of Petri nets to this field.

The structure of the paper is as follows. In Section 2 trophic networks are introduced with a small case study related to the Venice lagoon. In Section 3 the main concepts in Petri nets used to model trophic networks are briefly recalled. In Section 4 we propose a simple application of Petri nets to the representation and analysis of trophic networks when dynamic information are not available. This is exemplified in the case study. Some conclusions and suggestions for further work are given in Section 5.

2 Tropic Networks

An *ecosystem* is a community of living organisms, such as plants, animals and microbes, in conjunction with the nonliving components of their environment, such as air, water and bioavailable organic matter (detritus), which interact as a system. A *trophic network* (or *food web*) is a representation of feeding interactions in an ecosystem, where the components are connected by binary links (what-eats-what). Food webs permit to represent and analyse the trophic structure and functioning of an ecosystem. This knowledge can be used to identify key species and to detect anthropogenic impacts, such as the effects of pollution, of physical disturbance, of resources exploitation, etc. Real trophic networks are very complex, hence models provide partial and abstract representations where, for instance, similar species are aggregated into groups with similar feeding behaviour. Model representation of a trophic network generally focuses on the fluxes of energy or biomass between nodes. Such fluxes are directional and generally encompass some very relevant organism-level processes, such as production, consumption, assimilation, predation, non-predatory mortality and respiration. An ecosystem is generally an open system, i.e. there are flows of material or energy between the system and the rest of the world. For this reason, when representing and analysing trophic networks, generally also the input and output flows are taken into account. Inputs can be primary production, immigration or incoming of detrital matter into the system, while outputs can be emigration, harvesting by humans and exit of detrital matter from the system. Some energy may be dissipated into heat (respiration) or some material may be degraded into its lowest energy form (detritus).

Knowledge on the species present in the studied ecosystem and on their feeding behaviour is a needed prerequisite for representing the trophic network. First of all it is necessary to single out the n living and non-living compartments to be represented. A compartment can represent a population of a given species or of some aggregation of species with comparable feeding habits. For each compartment it is necessary to determine which other taxa are included in its diet, thus



Nocost.	Flux
1	$CO_2 \rightarrow$PHP
2	input→DET
3	PHP→MIZ
4	PHP→MEZ
5	PHP→DET
6	PHP→TAP
7	DET→BPL
8	BPL→CO_2
9	BPL→MEZ
10	BPL→MIZ
11	BPL→TAP
12	MIZ→MIZ
13	MIZ→DET
14	MIZ→CO_2
15	MIZ→MEZ
16	MIZ→TAP
17	MEZ→MEZ
18	MEZ→DET
19	MEZ→CO_2
20	TAP→DET
21	TAP→CO_2
22	TAP→Harvesting
23	DET→TAP
24	DET→Export

Fig. 1. A trophic network $\mathcal{T}_V$ of the Venice Lagoon [4] (left) and its fluxes (right).

specifying the interactions among species or groups of species. These information determine the network topology, which already provides some relevant insights on the features of the ecosystem. It is normally represented as a directed graph where each node represents a compartment and each arc denotes an interaction between the source and target nodes. More precisely, an arc from node A to node B represents a flow of energy or biomass from A to B. A common convention is to depict dissipation for some node with an arc outgoing from the node and ending in the ground symbol of electrical circuits [20]. A quantity may be associated with each arc, representing the magnitude of biomass or energy flow or the relative occurrence of such a flow. The resulting graph is a directed weighted graph.

The graph of a simple planktonic trophic network of the Venice Lagoon, taken from [4], is shown in Figure 1 (left). Numbers on arrows indicate the fluxes, which are listed in Figure 1 (right). The compartments considered are phytoplankton (PHP), bacterioplankton (BPL), microzooplankton (MIZ), mesozooplankton (MEZ), *R. philippinarum* (TAP) and organic detritus (DET). The network provides a representation of the food items digested and assimilated by *R. philippinarum* (a marine bivalve mollusk), namely, green algae, cyanobacteria, diatoms, bacterioplankton, microzooplankton, and dead, dissolved, and/or particulate organic matter.

This trophic network has some peculiarities:

- dissipation (respiration) of PHP is not considered because the flow from CO_2 to PHP models the net photosynthetic production, known from experimental data, i.e. the CO_2 needed for respiration has been already subtracted;

- flow from BPL to DET (mortality of BPL) is not considered because it is known to be negligible by experimental data;
- flows from TAP, MEZ and MIZ to DET include both natural mortality and production of faeces;
- flow from PHP to DET indicates only mortality, because PHP does not produce faeces;
- in the case of MIZ and MEZ cannibalism is represented by arrows exiting and entering in the same compartment (flows 12 and 17).

From the topology of the graph, or the corresponding adjacency matrix, some information about system behaviour can be derived. Clearly the adjacency matrix does not represent the information on weights of the interactions. For this reason various other matrices have been defined and used for analysis purposes, such as the matrix of dietary coefficients, the Leontief structure matrix, the total dependency matrix and many others which express different views of the network in relation to structural and quantitative dependencies among compartments [28]. The main advantage of a matrix representation of a trophic network is that linear algebra techniques can be applied and in fact matrix methods are the most used for static analysis of trophic networks (e.g. I-O modelling techniques for economics modified for ecosystems [28]).

To move from purely topological analysis of a trophic network to quantitative analysis, ecologists need quantitative data. Estimation of biomass and knowledge of several rates (e.g. production rate, consumption rate, respiration rate, etc.) are needed to quantify flows among compartments, together with quantitative knowledge about diet composition of each living compartment. Some information on primary production, specific consumption rates and diet compositions can be gained from field and laboratory studies but it is unfeasible to determine the magnitudes of all flows in the system directly. It becomes necessary, therefore, to estimate the magnitudes of some of them by indirect means. A helpful approach for estimating unknown flows consists in assuming the balance of inputs and outputs for each compartment. If a sufficiently long time period is considered, mass balance in each node of the network is a reasonable assumption because of the conservation of mass principle. Under the mass balance assumption, the system is represented as a steady state snapshot of energy flows, averaged over time. Different techniques are used for the trophic network reconstruction, that is to infer unspecified flows by solving the balance equations and satisfying the constraints among the flows in the system. The problem is generally underdetermined and an infinite number of solutions comply with the data set and the mass balance assumption. One technique is the Inverse Model (IM), which has been firstly applied to trophic network in [31] and it has become quite common among ecologists. IM combines mass balance equations, data equations and constraints on the flows expressed as inequalities. It finds a unique solution based on some optimisation criteria, for example by minimising the sum of squared flows, which corresponds to the most parsimonious solution. The package LIM implements linear inverse models in R [29]. Another freely available popular automated balancing routine that supports representation of trophic networks, estimation of unknown flows and ecological network analysis is Ecopath [6] and its evolutions Ecopath-Ecosym-Ecospace [7, 8].

Several analyses on ecological networks have been defined in the last decades. Some of them are based only on the topology of the model, for instance determining food chain length, connectance and the presence of cycles. In a balanced model it is possible to study both qualitative and quantitative properties measured by global system status indexes such as degree of recycling [2], stability [16, 30], development [27], ascendency [28] and maturity [20]. Analysis of recycling is intended to characterise how the biomass or energy is reused in a trophic network. Such analysis requires the topology of pathways over which the medium is recycled, as well as the amounts of material cycling in each loop. In [28] the author proposes to do this into two steps: first all simple cycles in the network are identified, then cycled flows are separated from straight-through flows and a technique is proposed to identify and subtract them from the original network.

3 Petri Nets

Petri nets are a well known formalism originally introduced in computer science for modelling discrete concurrent systems. Petri nets have a sound theory and many applications which are not limited to computer science (see, e.g., [19] and [11] for surveys). A large number of tools have been developed for analysing Petri nets (see a list at the *Petri Nets World* site [21]).

We denote a basic Petri net by $N = (P, T, W, M_0)$, where $P = \{p_1, \ldots, p_n\}$ is the set of places, $T = \{t_1, \ldots, t_m\}$ is the set of transitions, $W : \big((P \times T) \cup (T \times P)\big) \to \mathbb{N}$ is the weight function and M_0 is the *initial marking* of the net, an n-dimensional integer vector assigning to each place its initial number of tokens.

We write t^- for the *pre-condition* of a transitions t, namely the n-dimensional vector $t^- = (i_1, \ldots, i_n)$, where $i_j = W(p_j, t)$ for $j \in \{1, \ldots, n\}$. Sometimes it will be confused with its support, i.e., the set of places $\{p_j \mid i_j > 0\}$. The *post-condition* $t^+ = (o_1, \ldots, o_n)$ is defined dually.

The *incidence matrix* of a Petri net N, denoted by $\mathcal{A}_N$, is the $n \times m$ matrix which has a row for each place and a column for each transition. The column associated with transition t is the vector $(t^+ - t^-)^T$, which represents the marking change due to the firing of t.

Depending on the available information, Petri nets may permit to represent and study a system qualitatively, based only on the graph structure, as much as quantitatively or dynamically. An interesting *structural analysis* is based on the incidence matrix and it aims to determine the so-called *invariants* of the net. We focus here on T-invariants. Let N be a Petri net, with m transitions and n places, a *T-invariant* (transition invariant) of N is a multiset of transitions whose execution starting from a state will bring the system back to the same state, namely it is an m-dimensional vector in which each component represents the number of times that a transition should fire to take the net from a state M back to M itself. It can be obtained as a solution of the equation

$$\mathcal{A}_N \cdot X = 0, \quad \text{where } X = (x_1, \ldots, x_m)^T \quad \text{and} \quad x_i \in \mathbb{N}, \text{ for } i \in \{1, \ldots, m\}.$$

A T-invariant $X \neq 0$ indicates that the system can cycle on a state M enabling the cycle. As discussed in [13, 18], T-invariants admit two possible interpretations. On the one hand, the components of a T-invariant represent a multiset of interactions (transitions) whose partially ordered execution reproduces a given initial state of the system (marking). On the other hand, the components of a T-invariant may be interpreted as the relative rates of interactions (transitions) which occur permanently and concurrently in a steady state. Minimal T-invariants of a finite Petri net, N, form a basis, $\mathcal{B}(N)$, for the set of semi-positive T-invariant (Hilbert basis [24]). Any T-invariant can be obtained as a linear combination, with positive integer coefficients, of elements of the basis. Uniqueness of the basis $\mathcal{B}(N)$ makes it a characteristic feature of the net N.

Two subclasses of Petri nets will be of interest in the modelling of trophic networks [10]. A *state machine Petri net* is a Petri net where every arc has weight one and every transition has exactly one place in its pre- and post-condition. State machine Petri nets are conservative, namely the total number of tokens of the system remains invariant under the occurrence of transitions. A *free choice* Petri net is characterised by the fact that for any place p, either p has at most one post-transition (i.e. no conflict) or it is the only pre-place of all its post-transitions. The class of state machine Petri nets is strictly included in the class of free choice Petri nets.

Petri nets supply an executable specification: in the case of basic Petri nets, we can play the *token game*, i. e. the non-deterministic firing of all the enabled transitions. More sophisticated and realistic models and simulations can be obtained through extended Petri net models. The most interesting in our context are Continuous Petri nets. In *Continuous Petri nets* [13] the state is no longer discrete. Places contain non-negative real numbers, called marks, usually interpreted as the concentration of the species represented by the place. The instantaneous firing of a transition is carried out like a continuous flow. The firing rate expresses the "speed" of the transformation from input to output places. The rate functions associated with transitions may follow, under simplifying assumptions, known kinetic equations such as the mass action equation.

4 Petri Nets for Analysing Trophic Networks

We start discussing how Petri nets can be used to model and analyse a trophic network. We assume to know only the species (or compartments) and their relations, which is the minimal knowledge generally available on a trophic network. As a running example, we consider the trophic network $\mathcal{T}_V$ of the Venice lagoon in Figure 1. We illustrate how to build corresponding Petri net models and discuss what we can obtain by applying some Petri net analysis techniques. We use the tools Snoopy [14], Charlie [15] and 4ti2 [1] for editing and analysing the Petri net models.

4.1 Modelling trophic networks with Petri nets

Given a trophic network $\mathcal{T}$, a simple Petri net model can be immediately derived by replicating the topological structure of $\mathcal{T}$ in the Petri net. Recall that in the graph representation of $\mathcal{T}$ each species (or compartment) is a node and a relation between two species is a directed arc representing the flux between the two species.

A *structural Petri net model* of a trophic network $\mathcal{T}$ is the net $N_s(\mathcal{T})$ where

- any species (or compartment) becomes a place;
- any flow (relation) between two species S_1 and S_2 in $\mathcal{T}$, becomes a transition having S_1 as a pre-condition and S_2 as a post-condition.
- any outgoing flow from a species S_1 to the external environment (e.g., dissipation) in $\mathcal{T}$, becomes a transition with pre-condition S_1 and empty post-condition; similarly, any incoming flow from the environment to a species S_2, becomes a transition with empty pre-condition and post-condition S_2.

In absence of any information regarding the fluxes, all weights are set to one. Transitions corresponding to interactions among species are referred to as *internal transitions*, while those corresponding to interactions with the environment are referred to as *interface transitions*. Note that the structural Petri net model of a trophic network is a free choice Petri net and, when restricted to internal transitions, it is a state machine Petri net.

By applying the described construction to the running example $\mathcal{T}_V$ in Figure 1, we obtain a structural Petri net model which is depicted in Figure 2 (for the moment, please ignore the rates associated with transitions). The net includes six places (in yellow) representing the six compartments (DET, PHP, BPL, MIZ, MEZ, TAP) of the trophic network, and by as many transitions as the flows of biomass, to which we associate different colors to improve readability. More specifically, respiration flows (producing CO_2) are represented by light blue transitions; defecation flows are represented by brown transitions; mortality flows are represented by purple transitions; input and export flows for DET, as well as the harvesting flow for TAP are represented by red transitions; predation-prey flows are represented by white transitions.

Note that transitions PHP_CO2, representing respiration of PHP, and BPL_DET, representing BPL mortality in the Petri net model of Figure 2, do not have a direct match in the trophic network $\mathcal{T}_V$ of Figure 1. This is due to the fact that, as already mentioned, $\mathcal{T}_V$ was simplified by taking into account also some experimental data. More precisely, the flow corresponding to PHP_CO2 was integrated in CO2_PHP (modelling CO_2 needed for photosynthesis) and BPL_DET was considered irrelevant and thus omitted.

4.2 Structural analysis of trophic networks modelled as Petri nets

Since the structural Petri net model strictly adheres to the graph representation used by ecologists, it obviously enables the usual structural analyses for trophic networks, for example to determine food chains lengths and connectance.

In addition, standard structural analyses for Petri nets can be used, like those based on T-invariants. The presence of T-invariants in a Petri net model of a trophic network is ecologically of interest as it can reveal the presence of steady states. The set of transitions involved in a T-invariant can be seen as a subsystem of the original system, whose equilibrium is autonomously maintained.

Given a trophic network $\mathcal{T}$, consider the set of semi-positive T-invariants of the structural Petri net model $N_s(\mathcal{T})$ and the corresponding Hilbert basis $\mathcal{B}(N_s(\mathcal{T}))$, consisting of the minimal T-invariants. According to the terminology in [13], we classify T-invariants into two groups:

- *internal T-invariants*, consisting of internal transitions only;
- *I/O T-invariants*, which include also interface transitions.

If we consider the elements of the basis, then for any such T-invariant $I = (x_1, \ldots, x_m)$ we have $x_i \leq 1$ for all $i \in \{1, \ldots, m\}$, namely each transition occurs at most once and the invariant is a set rather than a proper multiset. Moreover, since $N_s(\mathcal{T})$, when restricted to the internal transitions, is a state machine, for any pair of transitions t_i, t_j in the same invariant, whenever they share a place in the pre-condition or in the post-condition, they coincide. Therefore:

- *Minimal internal invariants are simple cycles*, involving only internal transitions.
- *Minimal I/O invariants are acyclic paths*, connecting two interface transitions.

In both cases we recover well-known concepts in trophic networks as presented, e.g., in [28]. The internal minimal T-invariants are Ulanowicz simple cycles, which are associated with the internal recycling of matter. The minimal I/O T-invariants are the Ulanowicz straight-through flows, which represent the way energy and matter are provided by the environment, used by the network and then (partially) released back to the environment. The correspondence is at the structural level and the quantities of fluxes are needed for Ulanowicz analyses.

In our case study, the structural Petri net model has an Hilbert basis consisting of 69 minimal T-invariants, nine are internal and sixty are I/O invariants. The internal T-invariants are shown in Table 1. The first two invariants describe the self-predation (cannibalism) of MEZ and MIZ. All the other T-invariants "traverse" the DET place, pointing out that Detritus is the way for recycling matter in this network. The I/O invariants start from source transitions CO2_PHP and input_DET and end in sink transitions PHP_CO2, BPL_CO2, MIZ_CO2, MEZ_CO2, TAP_CO2 and TAP_harvesting. They model trophic chains allowing for respiration of the various compartments and for input and output of matter.

4.3 T-invariant based steady state

In this section we refine the structural Petri net model of a trophic network, turning it into a continuous Petri net model. What we obtain closely resembles the representation of the trophic network usually adopted by ecologists, where

Inv no.	Transitions
1	MEZ_MEZ
2	MIZ_MIZ
3	DET_TAP; TAP_DET
4	DET_BPL; BPL_DET
5	DET_BPL; BPL_MEZ; MEZ_DET
6	DET_BPL; BPL_MIZ; MIZ_DET
7	DET_BPL; BPL_TAP; TAP_DET
8	DET_BPL; BPL_MIZ; MIZ_MEZ; MEZ_DET
9	DET_BPL; BPL_MIZ; MIZ_TAP; TAP_DET

Table 1. Internal minimal T-invariants of the structural Petri net model of $\mathcal{T}_V$.

the system is at a steady state and the input and output flows in all the compartments are balanced (the mass balance assumption). The choice of considering a continuous extension is motivated by the fact that we are modelling fluxes of biomass which better correspond to continuous fluxes.

The continuous Petri net model is still derived only from the network topology by exploiting the minimal T-invariants in a way similar to what is done in [22] for Time Petri nets. A first observation is in order.

Remark 1. In the structural Petri net model of a trophic network $N_s(\mathcal{T})$ any place has typically at least one incoming and one outgoing transition, otherwise the place would unnaturally correspond to a compartment with monotonically increasing or decreasing content. Under this assumption, $N_s(\mathcal{T})$ *is covered by T-invariants*, namely each transition in the Petri net belongs to at least one minimal T-invariant. In fact, when we exclude interface transitions $N_s(\mathcal{T})$ is a state machine, hence for any transition, if we follow the predecessors and successors we will get back to the transition itself (internal T-invariant) or to an interface transition on both sides (I/O T-invariant).

In order to associate rates with the transitions, we assume that each subsystem corresponding to a minimal T-invariant

1. is active and
2. performs all its transitions once per time unit.

The assumption that all minimal subsystems of an ecosystem are active is quite reasonable from an ecological viewpoint. On the contrary the assumption that all subsystems perform all their transitions exactly once per time unit is rather strong and unrealistic. This is the simplest choice which can be taken in absence of further information on the ecosystem. When additional knowledge is available, it could be integrated in the model, as shown in the next section.

Let us consider the structural Petri net model $N_s(\mathcal{T})$ of a trophic network $\mathcal{T}$ as described in Section 4.1 and its Hilbert basis $\mathcal{B}(N_s(\mathcal{T}))$. According to the assumptions above, the rate of a transition t should depend on the number of minimal invariants in which t occurs. Then, for the trophic network $\mathcal{T}$, we define the *simple continuous Petri net model* $N_c(\mathcal{T})$ as the continuous Petri net obtained by considering the structural model $N_s(\mathcal{T})$ as underlying Petri net and by associating to each transition t a constant rate given by:

$$rate(t) = |\{I_i | I_i \in \mathcal{B}(N_s(\mathcal{T})) \ \wedge \ t \in I_i\}|.$$

With such rates, all the transitions in all the invariants in $N_c(\mathcal{T})$ are performed once in one time unit and the system is in a steady state. Moreover, since all transition arcs are 1-weighted, rates and flows per time unit coincide.

Remark 2. The continuous Petri net model of a trophic network satisfies the mass balance assumption, namely, for all compartments the sum of ingoing and outgoing fluxes coincide. This is an immediate consequence of the fact that minimal T-invariants are simple cycles or paths. Hence, given a place p, for any invariant I_i that "crosses" place p, one token is added to p by a transition in I_i and one token is consumed by another transition in I_i, namely the flux flowing through p via I_i is balanced. This holds for any invariant crossing p and for any p. Therefore, the input and output fluxes coincide for any place of the network.

In the simple continuous model $N_c(\mathcal{T})$, the system is represented in a steady state, with the fluxes of biomass balanced in all compartments. This corresponds closely to the ecologists representation of a trophic network as a snapshot of the system at steady state. Note that the continuous Petri net model $N_c(\mathcal{T})$, despite the fact that it makes explicit some additional features, is still based only on the topology of $\mathcal{T}$: biomasses do not play a role in the definition of the rates.

For our case study, the continuous Petri net model resulting from the construction outlined above is shown in Figure 2, where each transition have an associated rate. Note that all places are balanced. We would like to validate our simple continuous model by considering some basic ecological processes and check their plausibility from an ecological point of view. For each compartment we compute the throughput, namely the total amount of flux flowing per unit of time, in order to measure the degree of activity of the compartment. Besides we compute food consumption (total amount of ingested food per time unit), food assimilation (amount of ingested food minus amount of faeces, per time unit), respiration and mortality as percentages of the consumption. Table 2 shows the throughputs, the assimilation and respiration values as resulting from the model compared with those found in the literature.

The values derived from the simple continuous model are quite interesting. Considering the throughput, the various compartments are ordered as follows:

$$\text{DET}>\text{PHP}>\text{BPL}=\text{TAP}>\text{MIZ}>\text{MEZ}.$$

We may distinguish two main groups: lower trophic level compartments (DET, PHP and BPL), having higher throughput, and higher trophic level compartments (TAP, MIZ and MEZ), having lower throughput. This is coherent with the general knowledge on metabolic and growth rates of the two different groups of compartments under consideration.

Assimilation of the top compartment TAP is just over the maximum indicated in the literature, while assimilation requirements for MEZ and MIZ are perfectly met. However, MEZ assimilation is close to the lower bound of the indicated range. This is due to the fact that MEZ is a top level compartment

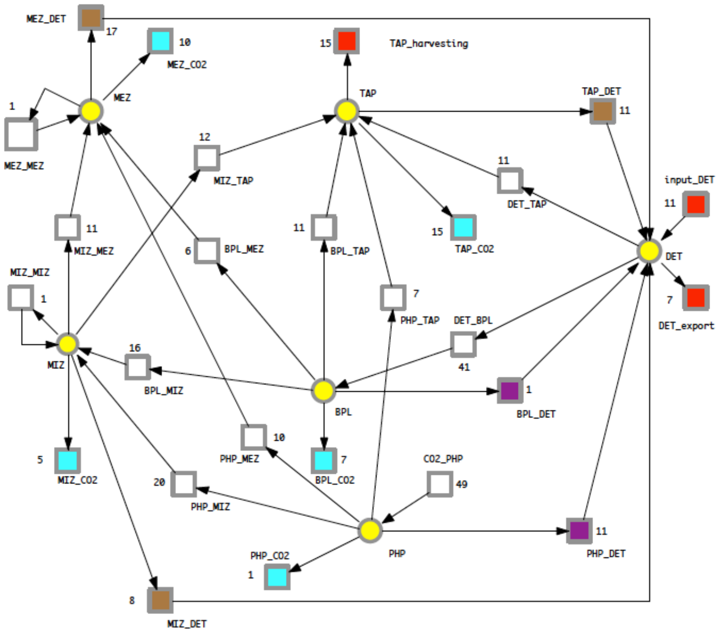


Fig. 2. Continuous Petri net model for the case study

in the network and no predators are modelled for it. This is a quite unrealistic assumption: in natural systems MEZ are actually preyed by other species, like fishes. By adding an external predation on MEZ, we found that its assimilation becomes close to TAP and MIZ assimilation values.

Concerning respiration, TAP and MEZ satisfy the constraints found in the literature, while MIZ and BPL are slightly below the indicated value. Respiration of PHP is instead largely below the lower bound of the indicated range. The low respiration flows for MIZ, BPL and PHP is caused by the fact that there are only a few I/O minimal invariants involving these compartments. This is a misbehaviour of the simple continuous model, that must be somehow overcome.

Concerning mortality, for BPL it is irrelevant and this is in accordance with experimental data (see discussion in Section 2). Mortality of PHP is instead quite high: this is probably due to the fact that some PHP grazers, like fishes usually occurring in lagoon systems, are not modelled.

On the whole, the continuous Petri net model realistically reproduces the main processes of the trophic network considered in the case study. Even if it based only on the network topology, it allows for deriving some quantitative information on trophic network flows, which are coherent with results of experimental measures taken in natural ecosystems. Moreover, the quantitative validation shows that the model is somehow incomplete, signalling that two further predation fluxes, one for MEZ and one for PHP, should be represented in the model.

Compartment	throughput	Literature values	Model values

TAP	41	[25] Respiration $\geq$ 20%
		[25] 37% $\leq$ Assimilation $\leq$ 70%
			Respiration = 36%
			Assimilation = 73%
			Defecation and Mortality = 27%

MEZ	28	[12] Respiration $\geq$ 20%
		[23, 9] 40% $\leq$ Assimilation $\leq$ 80%
			Respiration = 37%
			Assimilation = 39%
			Defecation and Mortality = 61%

MIZ	37	[12] Respiration $\geq$ 20%
		[23, 9] 40% $\leq$ Assimilation $\leq$ 80%
			Respiration = 14%
			Assimilation = 78%
			Defecation and Mortality = 22%

BPL	41	[26, 5] Respiration $\geq$ 20%
		Assimilation = Consumption
			Respiration = 17%
			Assimilation = Consumption
			Mortality = 2,4%

PHP	49	[32, 3] 10% $\leq$ Respiration $\leq$ 30%
		Assimilation = Consumption
			Respiration = 2%
			Assimilation = Consumption
			Mortality = 22%

DET	58	not relevant	not relevant

Table 2. Literature values and measured values for the continuous Petri net model of the case study.

4.4 Introducing ecological constraints in the Petri net model

In the previous section we underlined some misbehaviours of the simple continuous Petri net model. These are somehow expected since the model is only based on the topology of the system and it relies on the strong assumption that all subsystems proceed at the same speed. In order to adjust the model and make it closer to the real trophic network, one can follow two directions:

1. Drop the assumption that all the subsystems perform their path exactly once in one time unit and "speedup" some subsystems.
2. Use additional knowledge on the trophic network besides the topology, such as the metabolism of the species or their diet, and impose some constraints on the rates of the corresponding transitions.

We next examine more closely these two alternatives and apply them to our case study.

Speeding-up subsystems. Recall that any linear combination of minimal T-invariants is a T-invariant and a possible steady state of the network. Let us consider a generic linear combination of all minimal T-invariants:

$$\sum_{I_i \in \mathcal{B}(N_c(\mathcal{T}))} k_i I_i \ , \qquad k_i \in \mathbb{R}.$$

The simple continuous model $N_c(\mathcal{T})$ corresponds to a steady state given by a linear combination of all the minimal T-invariants where all the k_i are set to one. The refined continuous Petri net model, $N_{cs}(\mathcal{T})$, is obtained from $N_c(\mathcal{T})$ by dropping the assumption that all subsystems have the same speed and setting the invariants constants k_i to values possibly greater than one. In $N_{cs}(\mathcal{T})$ the rate associated with each transition is generalised to:

$$\mathit{rate}(t) = \sum_{I_i \in \mathcal{B}(N_s(\mathcal{T})),\ t \in I_i} k_i .$$

No.	Transition	rate	rate after speedup	No.	Transition	rate	rate after speedup	No.	Transition	rate	rate after speedup
1	CO$_2$_PHP	49	60	10	BPL_MIZ	16	17	19	MEZ_CO$_2$	10	10
2	input_DET	11	14	11	BPL_TAP	11	11	20	TAP_DET	11	11
3	PHP_MIZ	20	22	12	MIZ_MIZ	1	1	21	TAP_CO$_2$	15	15
4	PHP_MEZ	10	10	13	MIZ_DET	8	8	22	TAP_Harv.	15	15
5	PHP_DET	11	11	14	MIZ_CO$_2$	5	8	23	DET_TAP	11	11
6	PHP_TAP	7	7	15	MIZ_MEZ	11	11	24	DET_Export	7	7
7	DET_BPL	41	44	16	MIZ_TAP	12	12	25	PHP_CO$_2$	1	10
8	BPL_CO$_2$	7	9	17	MEZ_MEZ	1	1	26	BPL_DET	1	1
9	BPL_MEZ	6	6	18	MEZ_DET	17	17				

Table 3. Rates of the continuous Petri net model before and after speedup.

The refined continuous Petri net model $N_{cs}(\mathcal{T})$ still represents the trophic network at a steady state and with all compartments balanced, since the input and output fluxes are balanced in each place for each minimal T-invariant.

We applied this idea to the case study and speed up the invariants involving respiration of PHP, BPL and MIZ, since the respiration flows of these compartments do not satisfy the ranges indicated in the literature (see Table 2). The new rates for the transitions are shown in Table 3.

Concerning PHP, it receives in input CO$_2$ and partially release it for respiration. The unique I/O T-invariant for this process is {CO2_PHP; PHP_CO2}. By speeding up this invariant to run ten times per unit of time, the respiration flow for PHP becomes the 16% of its total consumption, within the range indicated by the literature (see Table 2).

Concerning BPL, it is fed by the Detritus and part of the ingested food is used for respiration. The invariants involving transition BPL_CO2 are {CO2_PHP; PHP_ DET; DET_BPL; BPL_CO2} and {input_DET; DET_BPL; BPL_CO2}. By allowing the second invariant to run three times per unit of time, respiration of BPL become the 20% of its total consumption.

Concerning MIZ, we could speedup the invariants involving MIZ_CO2, namely {CO2_PHP; PHP_ DET; DET_MIZ; MIZ_CO2} and {input_DET; DET_MIZ; BPL_MIZ}. By allowing them to run three and two times per unit of time, respectively, respiration of BPL becomes the 20% of its total consumption. Assimilation of BPL becomes the 80% of the consumption, still in the range indicated in Table 2.

Including constraints in the model. The second alternative for improving the model consists in "embedding" into the continuous Petri net model of the trophic network some available information regarding the metabolism of the species or their diet. We work under the simplifying assumption that flux constraints imposed on the model are linear. This assumption is generally satisfied by the constraints on metabolic fluxes and on the diet partitions. For our case study, some metabolic constraints taken from the literature are given in Table 2.

We define a continuous Petri net model which structurally coincides with $N_s(\mathcal{T})$ and whose transition rates satisfy a set of linear inequalities. As in the previous cases, the transition rates are derived from the "speed" k_i of each minimal invariant, but now we are interested only in invariants that satisfy the constraints. These can be obtained as solutions of a system of inequalities

$$\begin{array}{ll} A_N \cdot X & = 0 \\ C \cdot X & \geq 0 \end{array} \tag{1}$$

where A_N is the incidence matrix of $N_s(\mathcal{T})$. We can consider the minimal such T-invariants, referred to as the constrained Hilbert basis $\mathcal{B}_C(N_s(\mathcal{T}))$, so that any solution of (1) will be a linear combination of elements in $\mathcal{B}_C(N_s(\mathcal{T}))$.

A continuous Petri net model $N_c(\mathcal{T}, \mathcal{C})$ for the trophic network $\mathcal{T}$ satisfying the constraints $\mathcal{C}$ is defined as follows. The underlying Petri net is $N_s(\mathcal{T})$ and each transition t is associated with a constant rate:

$$\mathit{rate}(t) = |\{I_i : I_i \in \mathcal{B}_C(N_s(\mathcal{T})) \ \wedge \ t \in I_i\}|.$$

In this way each transition in each constrained invariant I_i in $\mathcal{B}_C(N_s(\mathcal{T}))$ can be performed once in one time unit.

When applied to our case study, this approach produces a linear system of equalities and inequalities, where the inequalities express the literature knowledge summarised in Table 2. By considering only the inequalities given by the lower bounds, the constrained Hilbert basis contains 349 minimal invariants. The induced rate constants for the extended network automatically satisfy the given ecological constraints.

The two approaches could be combined, by determining the constrained invariants and by setting for them possibly different speeds.

Simulations on continuous models with constant rates do not provide meaningful information. Some hints on how to further refine the model to do simulation analyses are given in the conclusions.

5 Conclusions and Future Work

In this paper we explored the use of Petri nets for representing and analysing trophic networks and our preliminary results are encouraging. A trophic network naturally translates into a structural Petri net model which allows for recovering classical trophic networks concepts and analyses. The structural model can be refined into a continuous Petri net model that closely resembles the representation of the trophic network usually adopted by ecologists, where the system is at a steady state and the input and output flows are balanced in all the compartments. Despite the fact that the Petri net models proposed are still simplistic (in particular, the continuous models have constant rates, independent of the masses), in our case study of the Venice lagoon, the analysis of the continuous Petri net model shows that it realistically reproduces the main ecological processes. Furthermore, it shows that the continuous Petri net model can be fruitfully used for an early stage validation of the trophic network under study. Two refinements of the continuous Petri net are considered: the first is based on a fine tuning of the speed of the minimal T-invariants, while the second one is based on a systematic embedding of some ecological knowledge expressed as linear inequalities into the calculation of the Hilbert basis. This however might have scalability problems, since the constraints increase the size of the Hilbert basis, and the problem of determining the Hilbert basis is already in EXPSPACE.

Future work deals with making the Petri net model more realistic and dynamic, by adding biomass information on compartments. The knowledge of biomasses at a steady state can, in fact, be used to derive constants for a continuous model governed, e.g., by the mass action equation. We believe that introducing rates dependent on biomasses could allow for interesting simulations, describing, not only the steady state but also the transient behaviour leading to such state. Additionally, on such model perturbations of the biomasses and of the speed of the various interactions could be used for performing what-if analyses.

Acknowledgements. We are grateful to Monika Heiner and Andrea Marin for many inspiring discussions.

References

1. 4ti2 team. 4ti2—a software package for algebraic, geometric and combinatorial problems on linear spaces. Available at www.4ti2.de.
2. S. Allesina and R. E. Ulanowicz. Cycling in ecological networks: Finn's index revisited. *Computational Biology and Chemistry*, 28:227–233, 2004.
3. R.S.K. Barnes and R.N. Hughes. *An introduction to Marine Ecology.* Wiley, 1999.
4. D. Brigolin and R. Pastres. Influence of intra-seasonal variability of metabolic rates on the output of a steady-state food web model. In Jordán F. and Jørgensen S.E., editors, *Models of the Ecological Hierarchy: From Molecules to the Ecosphere*, Developments in Environmental Modelling, pages 165–179. Elsevier, 2012.
5. C.A. Carlson, P.A. Del Giorgio, and G.J. Herndl. Microbes and the dissipation of energy and respiration: from cells to ecosystems. *Oceanography*, 20(2):89–100, 2007.
6. V. Christensen. Ecopath a software balancing steady-state models and calculating network characteristics. *Ecological modelling*, 61:169–185, 1992.
7. V. Christensen and C. J. Walters. Ecopath with Ecosim: methods, capabilities and limitations. *Ecological modelling*, 172(2):109–139, 2004.
8. V. Christensen, C. J. Walters, and D. Pauly. Ecopath with ecosim: a users guide. *Fisheries Centre, University of British Columbia, Vancouver*, 154, 2005.
9. R.J. Conover. Factors affecting the assimilation of organic matter by zooplankton and the question of superfluous feeding. *Limnology and Oceanography*, 11(3):346–354, 2003.
10. J. Desel and J. Esparza. *Free Choice Petri Nets.* Cambridge University Press, 2005.
11. J. Esparza and M. Nielsen. Decidability issues for Petri Nets - a survey. *Journal Inform. Process. Cybernet. EIK*, 30(3):143–160, 1994.
12. Vézina A. F. and M. L. Pace. An inverse model analysis of planktonic food webs in experimental lakes. *Canadian Journal of Fisheries and Aquatic Sciences*, 51(9):2034–2044, 1994.
13. M. Heiner, D. Gilbert, and R. Donaldson. Petri Nets for Systems and Synthetic Biology. In *Proc. of SFM'08*, volume 5016 of *LNCS*, pages 215–264. Springer, 2008.

14. M. Heiner, M. Herajy, F. Liu, C. Rohr, and M. Schwarick. Snoopy a unifying Petri net tool. In *Proc. of Petri Nets 2012*, volume 7347 of *LNCS*, pages 398–407. Springer, 2012.
15. M. Heiner, M. Schwarick, and J. Wegener. Charlie an extensible petri net analysis tool. In *Proc. of Petri Nets 2015*, LNCS. Springer, 2015. to appear.
16. R. E. Heymans, J. J. Ulanowicz and C. Bondavalli. Network analysis of the south florida everglades graminoid marshes and comparison with nearby cypress ecosystems. *Ecological Modelling*, 149:5–23, 2002.
17. F. Jordán, M. Scotti, and C. Priami. Process algebra-based computational tools in ecological modelling. *Ecological Complexity*, 8(4):357–363, 2011.
18. I. Koch and M. Heiner. Petri nets. In B. H. Junker and F. Schreiber, editors, *Analysis of Biological Networks*, Book Series in Bioinformatics, pages 139–179. Wiley, 2008.
19. T. Murata. Petri Nets: Properties, Analysis, and Applications. *Proceedings of IEEE*, 77(4):541–580, 1989.
20. E.P. Odum. The strategy of ecosystem development. *Science*, 164(3877):262–270, 1969.
21. Petri nets tools. http://www.informatik.uni-hamburg.de/TGI/PetriNets/tools.
22. L. Popova-Zeugmann, M. Heiner, and I. Koch. Timed Petri Nets for modelling and analysis of biochemical networks. *Fundamenta Informaticae*, 67:149–162, 2005.
23. Parsons T. R., Takahashi M., and Hargrave B. *Biological oceanographic processes.* Pergamon Press, 1984.
24. A. Schrijver. *Theory of linear and integer programming.* Interscience series in discrete mathematics and optimization. Wiley, 1999.
25. I. Sorokin and O. Giovanardi. Trophic characteristics of the manila clam. *ICES Journal of Marine Science*, 52(5):853–862, 1995.
26. Reinthaler T., Winter C., and Herndl G. J. Relationship between bacterioplankton richness, respiration, and production in the Southern North Sea. *Applied and environmental microbiology*, 5(7):2260–2266, 2005.
27. R. E. Ulanowicz. A phenomenological perspective of ecological development. *Aquatic Toxicology and Environmental Fate*, 9:73–81, 1986.
28. R. E. Ulanowicz. Quantitative methods for ecological network analysis and its application to coastal ecosystems. *Treatise on Estuarine and Coastal Science*, 9:35–57, 2011.
29. D. van Oevelen, K. van den Meersche, F. R. Meysman, K. Soetaert, J. Middelburg, and A. Vézina. Quantifying Food Web Flows Using Linear Inverse Models. *Ecosystems*, 13:32–45, 2010.
30. M. Vasconcellos, S. Mackinson, K. Sloman, and D. Pauly. The stability of trophic mass-balance models of marine ecosystems: a comparative analysis. *Ecological Modelling*, 100:125–134, 1997.
31. A.F. Vézina and T. Platt. Food web dynamics in the ocean. I. Best-estimates of flow networks using inverse methods. *Marine Ecology - Progress Series*, 42:269–287, 1988.
32. R. G. Wetzel. *Limnology. Lake and River Ecosystems.* Elsevier, 2001.

A coloured Petri net approach for spatial Biomodel Engineering based on the modular model composition framework Biomodelkit

Mary Ann Blätke[1]* and Christian Rohr[2]*

[1] Otto-von-Guericke University Magdeburg,
Chair of Regulatory Biology
Universitätsplatz 2, D-39106 Magdeburg, Germany
mary-ann.blaetke@ovgu.de
http://www.regulationsbiologie.de/

[2] Brandenburg University of Technology Cottbus-Senftenberg,
Chair of Data Structures and Software Dependability,
Postbox 10 13 44, D-03013 Cottbus, Germany,
christian.rohr@b-tu.de,
http://www-dssz.informatik.tu-cottbus.de

Abstract. Systems and synthetic biology require multiscale biomodel engineering approaches to integrate diverse spatial and temporal scales in order to understand and describe the various interactions in biological systems. Our BioModelKit framework for modular biomodel engineering allows to compose multiscale models from a set of modules, each describing an individual molecular component in the form of a Petri net. In this framework, we do now propose a feature for spatial modelling of molecular biosystems. Our spatial modelling methodology allows to represent the local positioning and movement of individual molecular components represented as modules. In the spatial model, interactions between components are restricted by their local positions. In this context, we use coloured Petri nets to scale the modular composed spatial model, such that each molecular component can exist in an arbitrary number of instances. Thus, a modular composed spatial model can be mapped to the cellular arrangement and different cell geometries.

Keywords: Modular Model Composition, Spatial Modelling, Multiscale Biomodel Engineering, Coloured Petri nets

1 Introduction

Systems biology aims at describing and understanding complex biological processes on a systems level. Therefore, systems biology employs modelling and simulation as indispensable tools to describe, predict and understand biological systems in an integrative and quantitative context. Besides complex interactions,

* Corresponding authors

M Heiner, AK Wagler (Eds.): BioPPN 2015, a satellite event of PETRI NETS 2015,
CEUR Workshop Proceedings Vol. 1373, 2015.

models do also need to integrate diverse temporal and spatial scales spanning the biological systems. Multiscale biomodel engineering goes beyond standard modelling approaches in systems biology and addresses physical problems as important features at multiple scales in time and space [1]. Current challenges and methodologies used so far in multiscale biomodel engineering have been reviewed in [2] and [3].

Here, we focus on the spatial aspects in multiscale biomodel engineering, which have been mostly neglected in the description of intracellular processes until now [1]. In particular, we demonstrate, based on the BioModelKit framework for modular biomodel engineering [4], how to extend plain models of intracellular processes to spatial models without their reimplementation. The models in our case are composed from modules, where each module describes the functionality of a certain molecular component in the form of a Petri net. The use of coloured Petri nets in our approach allows to represent different numbers of module instances for each component. To our knowledge, the methodology for spatial modelling in the context of modular model composition, which we suggest in this paper, is unique.

As modelling tool, we chose Snoopy [5], because it supports low-level and coloured Petri net network classes, as well as the concept of logical (fusion) nodes and hierarchical modelling.

In the next section, we will shortly describe the BioModelKit framework and summarize the use of coloured Petri nets for multiscale modelling. Afterwards, in Section 3, we introduce our spatial modelling methodology as a new feature of the BioModelKit framework. Section 4 applies the introduced methodology for spatial modelling to a simple example of a molecular interaction between two proteins represented as modules. In the last section, we give a short summary and outlook.

2 Previous Work

2.1 BioModelKit Framework for Modular Model Composition

The BioModelKit framework (BMK framework) is a tool for modular biomodel engineering [4], see Fig. 1. The main motivation behind BMK framework was to develop a modelling environment, where modules are specifically designed for the purpose of model composition. The modularisation approach used in BMK framework was inspired by the natural composition of biomolecular systems, where molecular components (genes, mRNAs and proteins) are the main building blocks. Thus, each molecular component is represented as a self-contained module, describing the underlying functionality using the formal language of Petri nets. Interface networks, which are part of each module describe the interactions with other molecular components and are used to automatically couple respective modules [4].

Since, the functionality of genes, mRNAs and proteins is diverse, different module types have been defined in BMK framework, as well as allelic influence

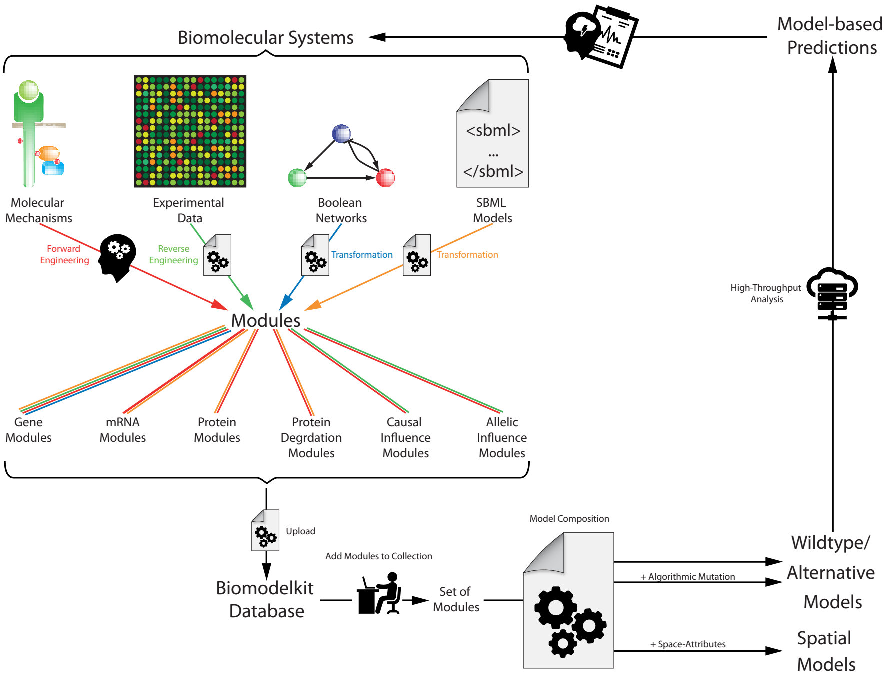


Fig. 1: Overview of the BioModelKit Framework for Modular Model Composition.

modules and causal influence modules to capture also correlations with missing mechanistic descriptions [6,7]. Modules can be generated by forward and reverse engineering approaches or by transforming boolean models or models provided in the systems biology markup language (SBML) into modules [8,9], see Fig. 1.

The web-interface of BMK framework (www.biomodelkit.com [4]) includes a feature to submit modules and to create a model annotation file in the BMK markup language (BMKml, *unpublished work*). The submitted module and its annotation have to be curated by an administrator before publicly releasing them by storing their content in a relational MySQL database (BMKdb). Another feature of the BMK framework is a model composition algorithm, which allows to automatically compose comprehensive models from a set of chosen modules. In addition, the composed model can also be modified by applying algorithms mimicking single/double gene knock-outs or structural mutations of the included molecular components (*unpublished work*).

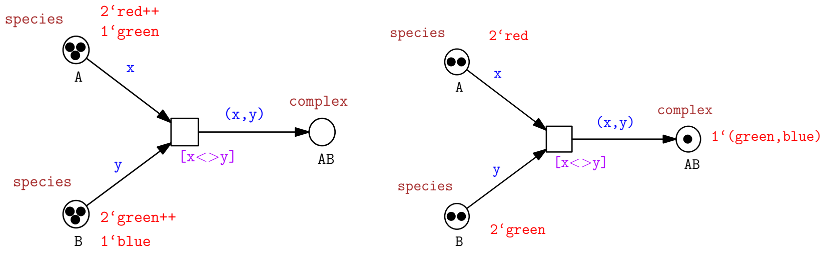
2.2 Coloured Petri Nets

We use Petri nets ($\mathcal{PN}$) as modelling paradigm, which gives us a complete formalised and standardised framework, as well as an intuitive way of modelling concurrent behaviour.

In systems biology, as well as in other fields, it's quite common that parts of larger models have similar structures. In such a case simplifying the model via reusing that part, instead of having redundant structures is demanded. Coloured Petri nets ($\mathcal{PN}^\mathcal{C}$) are a modelling paradigm that fits well in such a case.

We are using Snoopy [10] as modelling and simulation tool, thus we describe $\mathcal{PN}^\mathcal{C}$ how they are defined there.

```
coloursets:
    enum species := red, green, blue
    product complex := species, species
variables:
    species x
    species y
```



(a) Before Firing (b) After Firing

Fig. 2: Example $\mathcal{PN}^\mathcal{C}$ of abstract complex formation.

We use the coloured Petri net in Fig. 2 as an example. It represents an abstract complex formation of two species of different kind into one complex. The model contains two coloursets, first a simple colourset named *species* of type *enum*, including the colours *red*, *green* and *blue*. Second a product colourset named *complex*, its colours are 2-tuples of the *species* colourset. The net consists of three places A, B and AB and one transition. The colourset *species* is associated with the places A and B and the place AB has colourset *complex*. The variables x and y are used in the arc inscriptions and the transition guard. The transition guard $x <> y$ determines that only tokens of different colour are valid bindings for the variables x and y. The arc inscriptions x and y on the incoming

We summarize the following net classes together under the term *Petri net* ($\mathcal{PN}$): Qualitative Petri net ($\mathcal{QPN}$), eXtended Petri net ($\mathcal{XPN}$), Continuous Petri net ($\mathcal{CPN}$), Stochastic Petri net ($\mathcal{SPN}$) and Hybrid Petri net ($\mathcal{HPN}$). The same goes for the coloured Petri nets ($\mathcal{PN}^\mathcal{C}$).

arcs of the transition define its precondition, i.e. there have to be at least one token on place A and one token on place B and they have to be of different colour due to the guard. The arc inscription (x, y) on the outgoing arc of the transition defines the production of one *complex* token. In Fig. 2a the place A has two tokens of colour *red* and one token of colour *green* and the place B has two tokens of colour *green* and one token of colour *blue*. This gives the following bindings for the variables x and y: $(red, green)$, $(red, blue)$, $(green, blue)$. We selected the binding $(green, blue)$ and let the transition fire. One *green* token from place A and one *blue* token from place B are consumed and one $(green, blue)$ token is produced on place AB, see Fig. 2b.

Much more extensive descriptions how to use coloured Petri nets in systems biology are given in [11,12,13]. Besides the animation of the coloured Petri net, it is possible to unfold every $\mathcal{PN}^{\mathcal{C}}$ into an uncoloured $\mathcal{PN}$ [11]. So it is possible to apply any analysis and simulation technique available for uncoloured Petri nets on coloured Petri nets too.

Up to this, modelling biochemical systems using coloured Petri nets did not incorporate spatial aspects or movement in space. But this can be included in the model as shown by Gilbert et al. [14]. Therefore the space is discretised into a grid of one, two or three dimensions and a position in the grid is represented by a single place. This works fine if there is no need to distinguish between the entities on each position. One can model the diffusion of substances using this approach quite well, as presented in [14].

This can be extended to more complex reaction-diffusion systems, as shown in [15]. More examples of using coloured Petri nets for modelling of biological systems including spatial aspects are [16,17].

All models above have in common that they model space by discretisation into a grid and having one subnet (ranging from a single place to a complex network) per grid position. This is handy, if the entities moving around have no internal behaviour or state and there is no need to distinguish them. But if that is the case, the internal network has to move around as well and this leads to some issues on modelling and simulation. Parvu et al. [17] used this approach for a model of phase variation in bacterial colony growth. The bacteria have two different states, i.e. two places A and B representing the two states and two transitions for changing the state are needed. In order to let the bacteria move around, the whole subnet is needed in every grid position. Incorporating this in the coloured model is straightforward, but the size of the unfolded model increases drastically. This has an impact on the analysis and simulation of the model and may lead to inconvenient run times.

While the approaches of representing space via discretisation into grid-places fit well in the shown cases, it is not practical in our use case, because we have complex subnets moving around. We present our approach of incorporating space by adding coordinate places in the following section.

3 Spatial Modelling Methodology for Modular Composed Models

Before we start with the formal description of the spatial transformation algorithm, we have to introduce some general definitions which apply to our modular modelling approach. A module M_i is defined by a quintuple $M_i = (P_i, T_i, f_i, v_i, m_{0,i})$ according to the general definition of quantitative Petri nets. Each module M_i consists of $n_{ci} + 1$ components, the main component C_0 and n_{ci} interacting components. Thus, each module M_i represents a set of components $\mathbf{C}_i = \{C_0, C_1, \dots, C_{n_{ci}}\}$. The mapping of a place $p^i_j \in P_i$ of a module M_i to a set of components $\mathbf{C}_i^{p^i_j} \subseteq \mathbf{C}_i$ is given by the relation $g : P_i \rightarrow \mathbf{C}_i$, such that $g(p^i_j) = \mathbf{C}_i^{p^i_j}$. A place p^i_j with $|g(p^i_j)| > 1$ represents a complex of $|g(p^i_j)|$ components. The set $K_i = \{\mathbf{C}_i^{p^i_j} \mid |g(p^i_j)| > 1\}$ contains all complexes among the components in $\mathbf{C}_i$. A transition $t^i_j \in T_i$ of a module M_i with $|g(\bullet t^i_j) \cap g(t^i_j \bullet)| > 1$ represents an interaction with at least two different involved components. The total set of all interacting transitions in a Module M_i is given by $T_i^{IA} = \{\forall t^i_j \in T_i : |g(\bullet t^i_j) \cap g(t^i_j \bullet)| > 1\}$.

A set of modules defines a modular composed model $\mathcal{M} = \{M_1, \dots, M_n\}$, where n is the number of modules. Consequently, the modular composed model can also be defined as $\mathcal{M} = (P^{\mathcal{M}}, T^{\mathcal{M}}, f^{\mathcal{M}}, v^{\mathcal{M}}, m_0^{\mathcal{M}})$ according to the general definition of quantitative Petri nets with the following relations:

- $P^{\mathcal{M}} = \bigcup P_i$, where $M_i \in \mathcal{M}$ - total finite and non-empty set of places.
- $T^{\mathcal{M}} = \bigcup T_i$, where $M_i \in \mathcal{M}$ - total finite and non-empty set of transitions.
- $f^{\mathcal{M}} = \bigcup f_i$, where $M_i \in \mathcal{M}$ - total set of directed arcs, weighted by a non-negative integer value.
- $v^{\mathcal{M}} : \bigcup T_i \rightarrow H$, where $M_i \in \mathcal{M}$ - total set of firing rates.
- $m_0^{\mathcal{M}} \bigcup P_i \rightarrow \mathbb{N}_0$, $\exists p^i_{k'}, p^j_{k''}$ with $p^i_{k'} \in P_i, p^j_{k''} \in P_j$, where $p^i_{k'} = p^j_{k''}, \{p^i_{k'} : p^j_{k''}\} \rightarrow p^{\mathcal{M}}_k \in P^{\mathcal{M}}$, $m_0^{\mathcal{M}}(p^{\mathcal{M}}_k) = max(m_{0,i}(p^i_{k'}), m_{0,j}(p^j_{k''}))$

In addition to the definitions above, the following relations can be derived:

- $\mathbf{C}^{\mathcal{M}} = \bigcup \mathbf{C}_i$, where $M_i \in \mathcal{M}$ - total set of components
- $g^{\mathcal{M}} : \bigcup P_i \rightarrow \bigcup C_i$, where $M_i \in \mathcal{M}$ - total set of place component relations
- $T_{IA}^{\mathcal{M}} \subseteq T^{\mathcal{M}} = \bigcup T_i^{IA}$, where $M_i \in \mathcal{M}$ - total set of all interacting transitions
- $K^{\mathcal{M}} = \bigcup K_i$, where $M_i \in \mathcal{M}$ - total set of complexes.

Spatial Transformation Algorithm For the spatial transformation of the flat modular composed model the following procedure needs to be executed.

Step 1: Explicit Encoding of Local Positions. The position of each component $C_i \in C^{\mathcal{M}}$ is explicitly encoded by d places $p^{C_i}_1, \dots p^{C_i}_d$ (termed coordinate places), which can be interpreted as coordinates, where d, $d \geq 1$, defines the number of axes (e.g. 1D, 2D or 3D grid). The marking $m(p^{C_i}_j)$ of a place $p^{C_i}_j$ defines the current coordinate value, which must be restricted by a lower $m_L(p^{C_i}_j)$ and upper bound $m_U(p_j^{C_i})$ to represent the boundaries of the encoded grid, such that, $m_{L/U}(p_j^{C_i}) > 0$ and $m_L(p_j^{C_i}) < m_U(p_j^{C_i})$.

Step 2: Local Restriction of Interactions. To restrict the executability of each transition $t \in T_{IA}^{\mathcal{M}}$, the firing rate $h(t)$ must be multiplied by a boolean relation $b(t)$ describing a defined neighbourhood relation: $h^{IA}(t) = b(t) * h(t)$, $t \in T_{IA}^{\mathcal{M}}$. If the neighbourhood relation claims that the distance between components involved in the interaction represented by a transition $t \in T_{IA}^{\mathcal{M}}$ must be zero, $b(t)$ has to be defined as follows:

$$b(t) = \begin{cases} 1, & \sum_{i=1}^{|g(\bullet t) \cup g(t\bullet)|-1} \sum_{j=i+1}^{|g(\bullet t \cup g(t\bullet)|} \sum_{k=1}^{d} (m(p_k^{C_i}) - m(p_k^{C_j}))^2 = 0 \\ 0, & \sum_{i=1}^{|g(\bullet t) \cup g(t\bullet)|-1} \sum_{j=i+1}^{|g(\bullet t) \cup g(t\bullet)|} \sum_{k=1}^{d} (m(p_k^{C_i}) - m(p_k^{C_j}))^2 \neq 0 \end{cases}$$

In addition, read edges, connecting each transition $t \in T_{IA}^{\mathcal{M}}$ and the coordinate places of the respective components have to be added, such that $f_{ReadEdge}(p_{1-d}^{g(\bullet t) \cup g(t\bullet)}, t) = 1$.

Step 3: Explicit Encoding of Local Position Changes. To encode the position changes for a component $C_i \in C^{\mathcal{M}}$ two different scenarios have to be considered dependent on the state of interaction:

1. Local position change of individual components:
For each component $C_i \in \mathbf{C}^{\mathcal{M}}$ and each coordinate place $p_j^{C_i} \in \{p_1^{C_i}, \ldots, p_d^{C_i}\}$ two transitions $t_{j,L}^{C_i}$ and $t_{j,U}^{C_i}$ are needed to incrementally decrease or increase the amount of tokens. The transition $t_{j,L}^{C_i}$ subtracts tokens from the coordinate place $p_j^{C_i}$ till $m(p_j^{C_i}) = m_L(p_j^{C_i})$. Therefore, the following edges have to be introduced $f^{\mathcal{M}}(p_j^{C_i}, t_{j,L}^{C_i}) = 1$ and $f_{ReadEdge}^{\mathcal{M}}(p_j^{C_i}, t_{j,L}^{C_i}) = m_L(p_j^{C_i}) + 1$. The transition $t_{j,U}^{C_i}$ adds tokens to the coordinate place $p_j^{C_i}$ till $m(p_j^{C_i}) = m_U(p_j^{C_i})$. Therefore, the following edges have to be introduced $f^{\mathcal{M}}(t_{j,U}^{C_i}, p_j^{C_i}) = 1$ and $f_{InhibitorEdge}^{\mathcal{M}}(p_j^{C_i}, t_{j,U}^{C_i}) = m_U(p_j^{C_i})$. To ensure that the position of the component C_i can only be changed if it does not interact with another component C_j, $i \neq j$, additional inhibitory edges for each transition $t_{j,L/U}^{C_i}$ have to be introduced: $f_{InhibitorEdge}(P_{IS}^{C_i}, t_{j,L/U}^{C_i}) = 1$, where $P_{IS}^{C_i} \subseteq P^{\mathcal{M}}$ and $P_{IS}^{C_i} = \{\forall p \in P^{\mathcal{M}} : C_i \in g(p) \wedge |g(p)| > 1\}$.
2. Local position change of complexes:
The local position of components forming a complex $k_i \in K^{\mathcal{M}}$ have to be updated consistently during the local position change. For each complex $k_i \in K^{\mathcal{M}}$ and each dimension j, $1 \leq j \leq d$, two transitions $t_{j,L}^{k_i}$ and $t_{j,U}^{k_i}$ are needed to incrementally decrease or increase the amount of tokens. The transition $t_{j,L}^{k_i}$ removes tokens from the set of coordinate places $\bigcup_{C_h \in k_i} p_j^{C_h}$ till at least for one component $C_h \in \mathbf{C}^{\mathcal{M}}$ the condition $m(p_j^{C_h}) = m_L(p_j^{C_h})$ is fulfilled. Therefore, for each component $C_h \in k_i$ the following edges have to be introduced $f^{\mathcal{M}}(p_j^{C_h}, t_{j,L}^{k_i}) = 1$ and $f_{ReadEdge}^{\mathcal{M}}(p_j^{C_h}, t_{j,L}^{k_i}) = m_L(p_j^{C_h}) + 1$. The transition $t_{j,U}^{k_i}$ adds tokens to the set of coordinate places $\bigcup_{C_h \in k_i} p_j^{C_h}$ till at least for one component $C_h \in \mathbf{C}^{\mathcal{M}}$ the condition $m(p_j^{C_h}) = m_U(p_j^{C_h})$ is fulfilled. Therefore, for each component $C_h \in k_i$ the following edges have to be

introduced $f^{\mathcal{M}}(t_{j,U}^{k_i}, p_j^{C_h}) = 1$ and $f_{InhibitorEdge}^{\mathcal{M}}(p_j^{C_h}, t_{j,U}^{k_i}) = m_U(p_j^{C_h})$. To ensure that the position of the complex k_i can only be changed if it is actually formed additional read edges for each transition $t_{j,L/U}^{k_i}$ have to be introduced: $f_{ReadEdge}(P_{IS}^{k_i}, t_{j,L/U}^{k_i}) = 1$, where $P_{IS}^{k_i} \subseteq P^{\mathcal{M}}$ and $P_{IS}^{k_i} = \{\forall p \in P^{\mathcal{M}} \mid k_i = g(p)\}$. Furthermore, it must be excluded for each component $C_h \in k_i$ that interacts with other components using a different binding site. Therefore, all co-existing interactions have to be determined $K_{coex}^{k_i} = \{\forall k_j \in K^{\mathcal{M}} : k_i \cap k_j \neq \varnothing \mid k_j \neq k_i\}$. All places representing a complex $k_j \in K_{coex}^{k_i}$ have to be added to each transition $t_{j,L/U}^{k_i}$ using an inhibitory edge: $f_{InhibitoryEdge}(P_{COEX}^{k_i}, t_{j,L/U}^{k_i}) = 1$, where $P_{COEX}^{k_i} = \{\forall p \in P^{\mathcal{M}} : g(p) = k, k \in K_{coex}^{k_i}\}$.

To allow the movement of co-existing complexes which use different interaction sites simultaneously, the above described procedure has to be applied to all possible combination of co-existing complexes, compare Section 4.

For simplicity reasons the firing rate of each transition $t_{j,L/U}^{k_i}$ and $t_{j,L/U}^{C_h}$ is given by Fick's laws of diffusion [18]. Furthermore, we assume equidistant subvolumes with the width and hight $h = 1$ and set all diffusion coefficient to one. Please note, it is straightforward to define the diffusion coefficients more precisely based on experimental results.

Step 4: Encoding of Component Instances by Applying Coloured Petri nets. A colourset $\sigma_{C_i}^{simple}$ with $1 - q_{C_i}$ colours needs to be specified for each component $C_i \in \mathbf{C}^{\mathcal{M}}$, where $q_{C_i} \in \mathbb{N}$ defines the number of instances for a component C_i. The total set of simple coloursets is given by $\Sigma^{simple} = \{\sigma_{C_1}^{simple}, \ldots, \sigma_{|\mathbf{C}^{\mathcal{M}}|}^{simple}\}$. For each $\sigma_{C_i}^{simple} \in \Sigma^{simple}$ a variable a^{C_i} needs to be specified. All edges $f(p,t)$ and $f(t,p)$ of the flat model $\mathcal{M}$ for which it is true, that a component $C_i \in \mathbf{C}^{\mathcal{M}}$, where $C_i \in g(p)$ and $|g(p)| = 1$ are extended to the multiset expression a^{C_i}‘$f(p,t)$, or respectively a^{C_i}‘$f(t,p)$. The total set of simple coloursets Σ^{simple} is mapped to a subset of places according to the relation $S^{simple} : \Sigma^{simple} \to P^{\mathcal{M}}$, such that $S^{simple}(\sigma_{C_i}^{simple}) = \{p \in P^{\mathcal{M}} \mid C_i \in g(p) \wedge |g(p)| = 1\}$.

Each complex $k_i \in K^{\mathcal{M}}$, where k_i represents a subset of components, such that $k_i \subseteq C^{\mathcal{M}}$, is represented by a compound colourset of type product $\sigma_{k_i}^{compound} = \prod_{C_j \in k_i} \sigma_{C_j}^{simple}$. The total set of compound coloursets is given by $\Sigma^{compound} = \{\sigma_{k_1}^{compound}, \ldots, \sigma_{|K^{\mathcal{M}}|}^{compound}\}$. The total set of compound coloursets $\Sigma^{compound}$ is mapped to remaining subset of places according to the relation $S^{compound} : \Sigma^{compound} \to P^{\mathcal{M}}$, such that $S^{compound}(\sigma_{k_i}^{compound}) = \{p \in P^{\mathcal{M}} \mid k_i = g(p)\}$. All edges $f(p,t)$ and $f(t,p)$ of the flat model $\mathcal{M}$ for which its is true, that a complex $k_i \in K^{\mathcal{M}}$, where $k_i = g(p)$ are extended to the multiset expression $\bigcup_{C_j \in k_i} a^{C_j}$‘$f(p,t)$, or respectively $\bigcup_{C_j \in k_i} a^{C_j}$‘$f(p,t)$. The marking and firing rates are kept constant over all place and transition instances, such that marking of each place $p \in P^{\mathcal{M}}$ is represented by $all()`m_0(p)$ and the firing rates of each transition $t \in T^{\mathcal{M}}$ $all()`h(t)$, where $all()$ is a function that extracts all instances of a coloured node.

4 Example

For demonstration purposes we introduce in Fig. 3 a running example of a modular composed model, which consists of two modules, a module for Protein A and a module for Protein B. The module of Protein A describes the complex formation and cleavage between the two ligand binding domains of Protein A and Protein B ($ProteinA\_LBD$, $ProteinB\_LBD$). The formation of the complex between Protein A and Protein B ($ProteinA\_LBD\_\_\_ProteinB\_LBD$) is the trigger for the phosphorylation of a tyrosine residue at Protein B ($ProteinB\_TYR$, $ProteinB\_TYRp$). To phosphorylate Protein B, the catalytic domain of Protein A needs to be in an active state ($ProteinA\_CD\_active$). The catalytic domain of Protein A can switch between being active or inactive ($ProteinA\_CD\_active$, $ProteinA\_CD\_inactive$). In the module of Protein B, the subnet describing the interaction between Protein A and Protein B is redundant. Redundant subnets are called interface networks (indicated by logical (fusion) places and transitions shaded in grey), and are used to automatically couple modules. An additional subnet in the module of Protein B explains the complex formation and cleavage between the phosphorylated Tyrosine of Protein B ($ProteinB\_TYRp$) and a SH2 domain of Protein C ($ProteinC\_SH2$). The Module of Protein C is not given in this example. Since this paper is not dealing with kinetic aspects, we assume mass action kinetics and set all kinetic coefficient to one. It is straightforward to replace this assumption with more detailed kinetic descriptions.

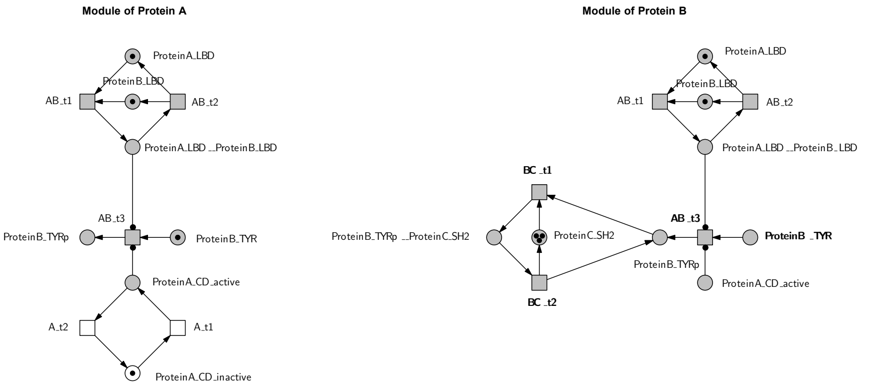


Fig. 3: Running example with two protein modules.

The module of Protein A (M_1) contains two components $\mathbf{C}_1 = \{\ ProteinA, ProteinB\}$ and the module of Protein B (M_2) contains three components $\mathbf{C}_2 = \{ProteinA, ProteinB, ProteinC\}$. For the composed Model $\mathcal{M} = \{M_1, M_2\}$, we get the following mapping of places to the components:

- $g^{\mathcal{M}}(\{ProteinA\_LBD, ProteinA\_CD\_active, ProteinA\_CD\_inactive\}) = ProteinA$
- $g^{\mathcal{M}}(\{ProteinB\_LBD, ProteinB\_TYR, ProteinB\_TYRp\}) = ProteinB$
- $g^{\mathcal{M}}(\{ProteinB\_SH2\}) = ProteinC$
- $g^{\mathcal{M}}(\{ProteinA\_LBD\_\_\_ProteinB\_LBD\}) = \{ProteinA, ProteinB\}$
- $g^{\mathcal{M}}(\{ProteinB\_TYRp\_\_\_ProteinC\_SH2\}) = \{ProteinB, ProteinC\}$

Furthermore places $ProteinA\_LBD\_\_\_ProteinB\_LBD$ and $ProteinB\_TYRp\_\_\_ProteinC\_SH2$ represent two complexes $k_1 = \{ProteinA, ProteinB\}$ and $k_2 = \{ProteinB, ProteinC\}$. The set of interacting transitions is given by $T_{IA}^{\mathcal{M}} = \{AB\_r1, AB\_r2, AB\_r3, BC\_r1, BC\_r2\}$.

For the spatial model we assume a two dimensional grid ($d = 2$) of the size 5×5 for each component given by the constants $xDimA = xDimB = xDimC = 5$ and $yDimA = yDimB = yDimC = 5$.

Step 1 of the spatial transformation algorithm introduces two coordinate places representing the x- and y-coordinate of each component, e.g. for component $ProteinA$ we add two places $ProteinA\_X$ and $ProteinA\_Y$, see Fig. 4. We chose the marking of the places representing the local position according to the following assumption: component $ProteinA$ is initially positioned at (1,1), component $ProteinB$ at (3,3) and component $ProteinC$ at (4,4).

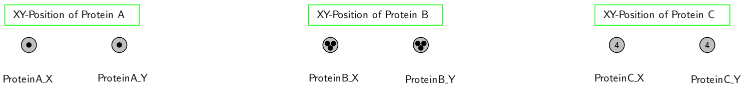


Fig. 4: Encoding of the local positions for each component in the composed modular model.

Step 2 of the spatial transformation algorithm restricts the interaction of components dependent on their local position. The restriction applies only to transitions in the set $T_{IA}^{\mathcal{M}}$. We assume that components can only interact, if their local positions are identical, meaning the distance between them must be zero. Therefore, the firing rates of transitions AB_t1, AB_t2 and AB_t3 must be multiplied with the boolean expression $b_{AB}(t)$:

$$b_{AB}(t) = \begin{cases} 1, dist_{AB} = 0 \\ 0, dist_{AB} \neq 0 \end{cases} \quad \text{with}$$

$$dist_{AB} = (m(ProteinA\_X) - m(ProteinB\_X))^2 + (m(ProteinA\_Y) - m(ProteinB\_Y))^2$$

The places representing the local positions of component $ProteinA$ and component $ProteinB$ have to be connected by read edges to the transitions AB_t1, AB_t2 and AB_t3, compare Fig. 5. Accordingly, the firing rates of transitions BC_t1 and BC_t2 must be multiplied with the boolean expression $b_{BC}(t)$:

$b_{AB}(t) = \begin{cases} 1, dist_{BC} = 0 \\ 0, dist_{BC} \neq 0 \end{cases}$ with
$dist_{BC} = (m(ProteinB\_X) - m(ProteinC\_X))^2 + (m(ProteinB\_Y) - m(ProteinC\_Y))^2$. The places representing the local positions of component $ProteinB$ and component $ProteinC$ have to be connected by read edges to the transitions BC_t1 and BC_t2, compare Fig. 5.

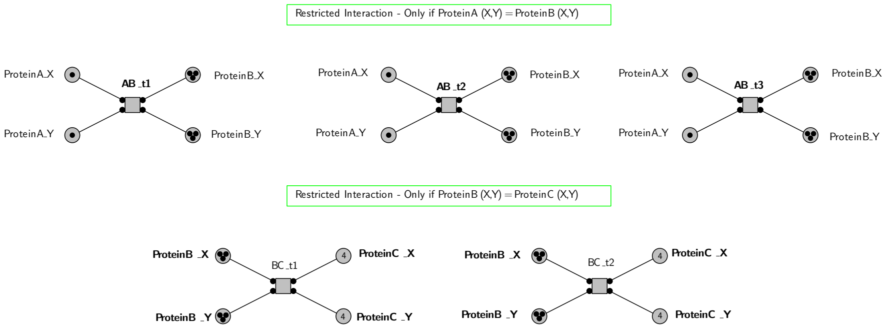


Fig. 5: Restriction of interactions depending on the local position of the involved components.

Step 3 of the spatial transformation algorithm encodes the local position change in respect to the interaction state of the components. In our example we assume only movement along the horizontal and vertical axes. For each axis two transitions are needed to either increase or decrease the marking value of the respective coordinate place of a component with respect to the grid size. A component can only move as a single entity if it is not interacting at the same time with other components. Therefore we have to check if the corresponding places representing the interaction states (complexes) are empty, e.g. in case of component $ProteinB$ it can only move as single entity if places $ProteinA\_LBD\_\_\_ProteinB\_LBD$ and $ProteinB\_TYRp\_\_\_ProteinC\_SH2$ are empty, see Figure 6(A). To move components forming a complex or which build co-existing complexes, the coordinates of all involved components have to be updated simultaneously, see Figure 6(B). Form the definitions above, we know that component $ProteinB$ can form a complex with $ProteinA$ and $ProteinC$. Thus, these two complexes can co-exist because component $ProteinB$ uses different interaction sites. To move the complex of component $ProteinA$ and component $ProteinB$ we need to check whether the corresponding place $ProteinA\_LBD\_\_\_ProteinB\_LBD$ is marked and if the place $ProteinB\_TYRp\_\_\_ProteinC\_SH2$ is empty. In contrast, to move the complex of component $ProteinB$ and component $ProteinC$ we need to check whether the place $ProteinB\_TYRp\_\_\_ProteinC\_SH2$ is marked and if place $ProteinA\_LBD\_\_\_ProteinB\_LBD$ is empty. And to move the co-existing complex of component $ProteinA$, component $ProteinB$ and component $ProteinC$, we need to check if both places $ProteinA\_LBD\_\_\_ProteinB\_LBD$ and $ProteinB\_TYRp\_\_\_ProteinC\_SH2$ are marked.

Step 4 of the spatial transformation algorithm has to be applied to represent more than one instance of each component, compare Fig. 7 and 8. In our example the number of instances for each component is three, which we represent by the constants $numA = numB = numC = 3$. For each component $C_i \in \mathbf{C}^{\mathcal{M}}$ we define a simple colourset:

- int csProteinA := 1 - numA,
- int csProteinB := 1 - numB,
- int csProteinC := 1 - numC

where colourset *csProteinA* is mapped to the places with the relation $g(p)$ =*ProteinA*, colourset *csProteinB* is mapped to the places with the relation $g(p)$ =*ProteinB* and colourset *csProteinC* is mapped to the places which fulfil relation $g(p)$ =*ProteinC*. The coordinate places of each component have to be bound to the respective colourset as well. Furthermore, we need to define a variable for each simple colourset:

- csProteinA A
- csProteinB B
- csProteinC C

All in-going and out-going arcs of places bound to one of the simple coloursets defined above have to carry the respective variable as arc expression. Since, we have two binary complexes $k_1 = \{ProteinA, ProteinB\}$ and $k_2 = \{ProteinB, ProteinC\}$, we need to define a compound colourset for each as product of the respective simple coloursets:

- product csProteinA_ProteinB := csProteinA, csProteinB
- product csProteinB_ProteinC := csProteinB, csProteinC

where colourset *csProteinA_ProteinB* is mapped to the places with the relation $g(p) = \{ProteinA, ProteinB\}$ and colourset *csProteinB_ProteinC* is mapped to the places with the relation $g(p) = \{ProteinB, ProteinC\}$. All in-going and out-going arcs of places bound to one of the compound coloursets defined above have to carry a 2-tuple of respective variables as arc expression.

Fig. 9 presents one exemplifying stochastic simulation run of the final spatial model of Fig. 7 and 8. In Fig. 9(A), we depict the movement of all instances of components *ProteinA*, *ProteinB* and *ProteinC* on separate two-dimensional grids of the size 5×5. During the simulation time three complexes between instances of component *ProteinA* and component *ProteinB* could be obtained ($ProteinA\_1+ ProteinB\_1$,$ProteinA\_1+ProteinB\_2$,$ProteinA\_3+ProteinB\_3$), as well as two complexes between instances of component *ProteinB* and component *ProteinC* ($ProteinB\_1$ + $ProteinC\_1$,$ProteinB\_3$ + $ProteinC\_3$). The simulation result also shows that the distance between the corresponding instances of components forming a complex is zero, which means that they can only move has one entity. Subfigure (1) and (2) of Fig. 9(B) shows that the instance $ProteinB_1$ is interacting with instance $ProteinA_1$ and instance $ProteinC_1$ at the same time near the end of the simulation run, thus the two complex states co-exist.

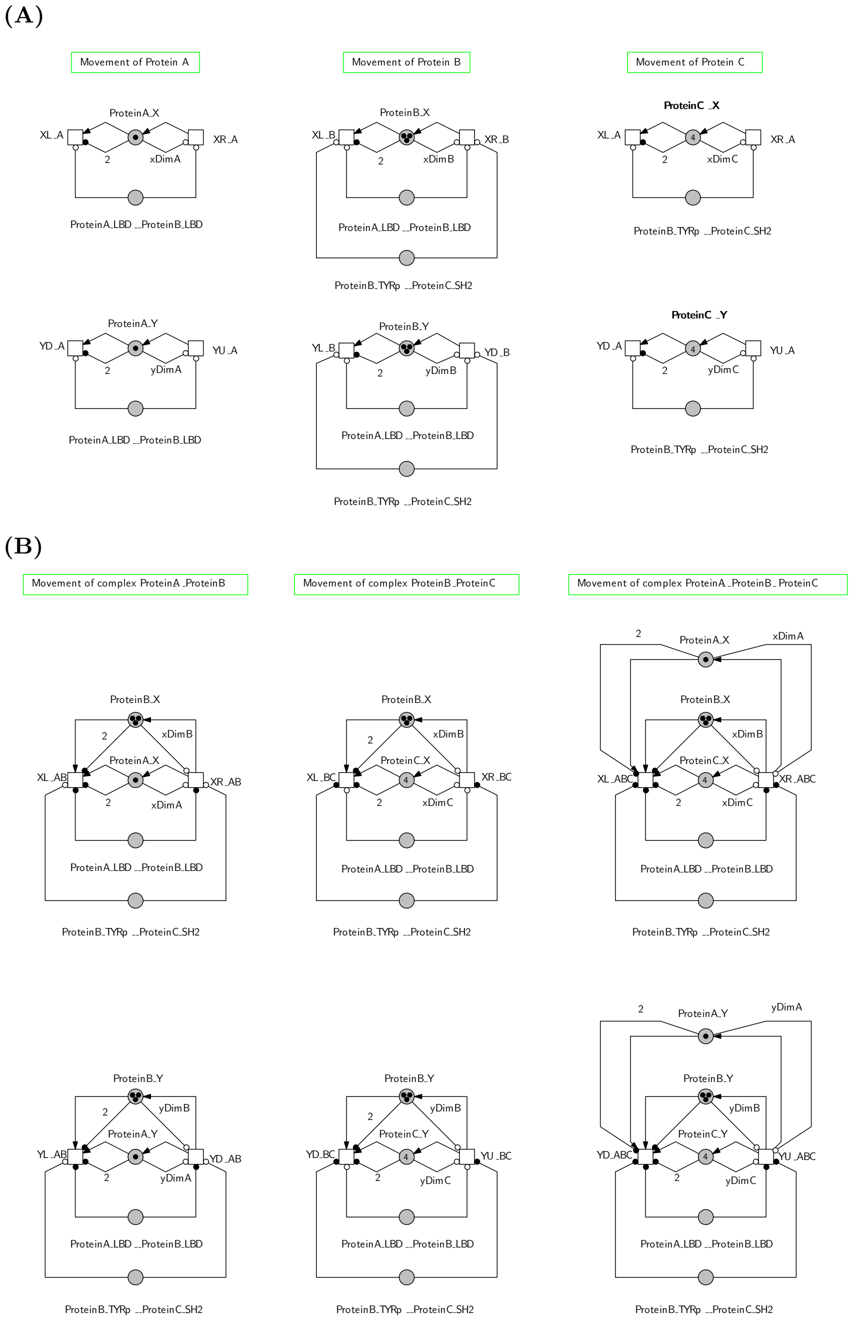
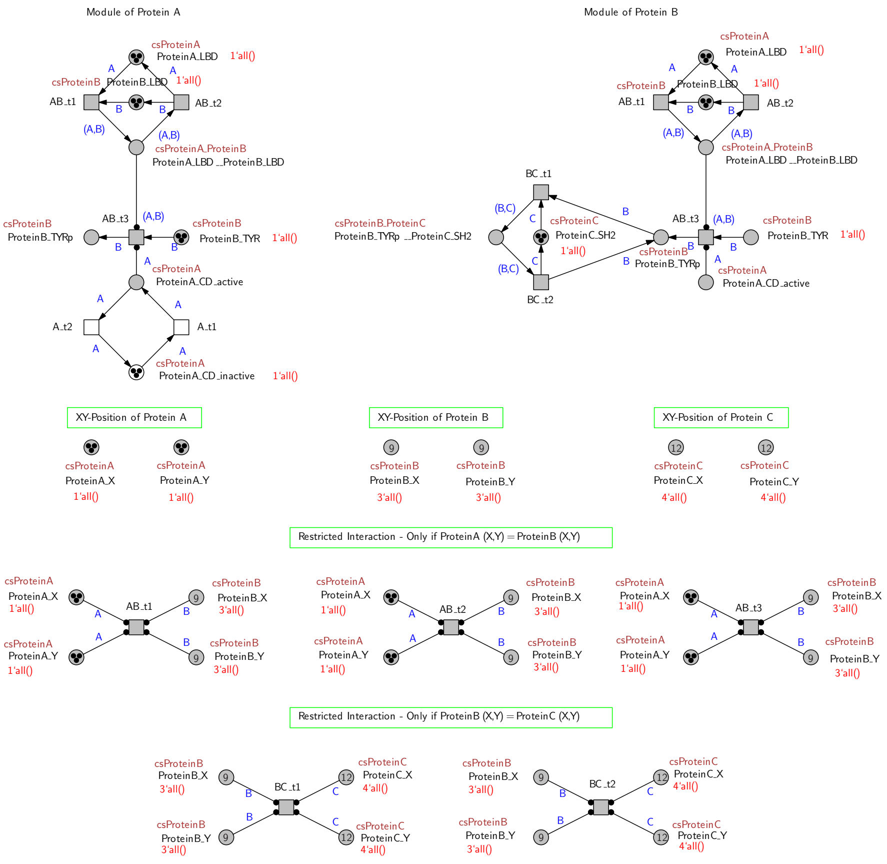

(A)
Movement of Protein A
Movement of Protein B
Movement of Protein C
ProteinA_X
XL_A
XR_A
2
xDimA
ProteinA_LBD _ProteinB_LBD
ProteinB_X
XL_B
XR_B
2
xDimB
ProteinA_LBD _ProteinB_LBD
ProteinB_TYRp _ProteinC_SH2
ProteinC _X
XL_A
XR_A
4
2
xDimC
ProteinB_TYRp _ProteinC_SH2
ProteinA_Y
YD_A
YU_A
2
yDimA
ProteinA_LBD _ProteinB_LBD
ProteinB_Y
YL_B
YD_B
2
yDimB
ProteinA_LBD _ProteinB_LBD
ProteinB_TYRp _ProteinC_SH2
ProteinC _Y
YD_A
YU_A
4
2
yDimC
ProteinB_TYRp _ProteinC_SH2
(B)
Movement of complex ProteinA _ProteinB
Movement of complex ProteinB _ProteinC
Movement of complex ProteinA _ProteinB _ProteinC
ProteinB_X
2
xDimB
ProteinA_X
XL_AB
XR_AB
2
xDimA
ProteinA_LBD _ProteinB_LBD
ProteinB_TYRp _ProteinC_SH2
ProteinB_X
2
xDimB
ProteinC_X
XL_BC
XR_BC
4
2
xDimC
ProteinA_LBD _ProteinB_LBD
ProteinB_TYRp _ProteinC_SH2
2
ProteinA_X
xDimA
ProteinB_X
xDimB
ProteinC_X
XL_ABC
XR_ABC
4
2
xDimC
ProteinA_LBD _ProteinB_LBD
ProteinB_TYRp _ProteinC_SH2
ProteinB_Y
2
yDimB
ProteinA_Y
YL_AB
YD_AB
2
yDimA
ProteinA_LBD _ProteinB_LBD
ProteinB_TYRp _ProteinC_SH2
ProteinB_Y
2
yDimB
ProteinC_Y
YD_BC
YU_BC
4
2
yDimC
ProteinA_LBD _ProteinB_LBD
ProteinB_TYRp _ProteinC_SH2
2
ProteinA_Y
yDimA
ProteinB_Y
yDimB
ProteinC_Y
YD_ABC
YU_ABC
4
2
yDimC
ProteinA_LBD _ProteinB_LBD
ProteinB_TYRp _ProteinC_SH2

Fig. 6: Local position change dependent on the state of interaction. (A) components are not in complex, (B) components are in complex with each other.



- simple coloursets:
  - int ProteinA := 1 - numA
  - int ProteinB := 1 - numB
  - int ProteinB := 1 - numC
- compound coloursets:
  - product ProteinA__ProteinB := ProteinA, ProteinB
  - product ProteinB__ProteinC := ProteinB, ProteinC
- variables:
  - ProteinA A
  - ProteinB B

- constants:
  - int numA := 3
  - int numB := 3
  - int numC := 3
  - int xDimA := 5
  - int yDimA := 5
  - int xDimB := 5
  - int yDimB := 5
  - int xDimC := 5
  - int yDimC := 5

Fig. 7: Part 1: Instantiation of modules using coloured Petri nets.

5 Conclusions & Outlook

We presented a new approach for incorporating spatial aspects into modular composed models. A new approach was necessary, because existing techniques (see Section 2.2) are not suitable for model composition. In particular, we demon-

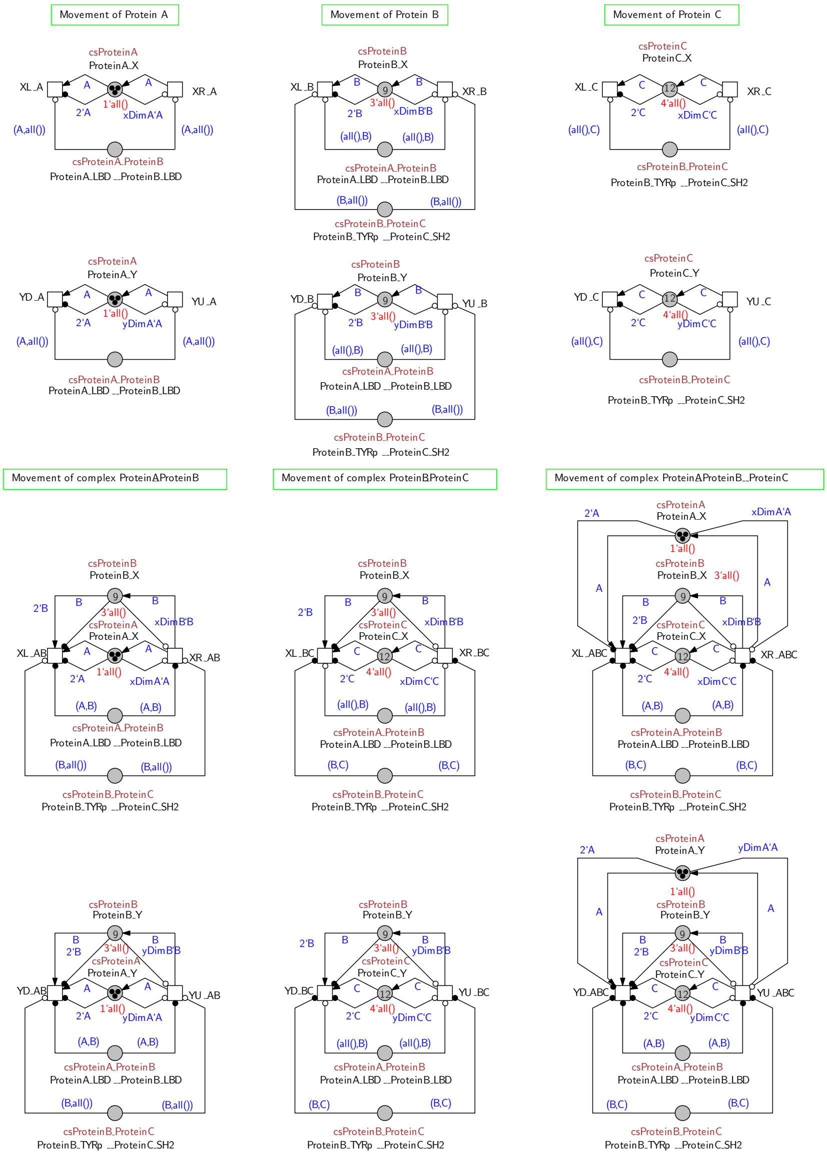


Fig. 8: Part 2: Instantiation of modules using coloured Petri nets.

strated based on the BioModelKit framework for modular biomodel engineering [4], how to extend plain models of intracellular processes to spatial models without their reimplementation. The models in our case are composed from modules, where each module describes the functionality of a certain molecular com-

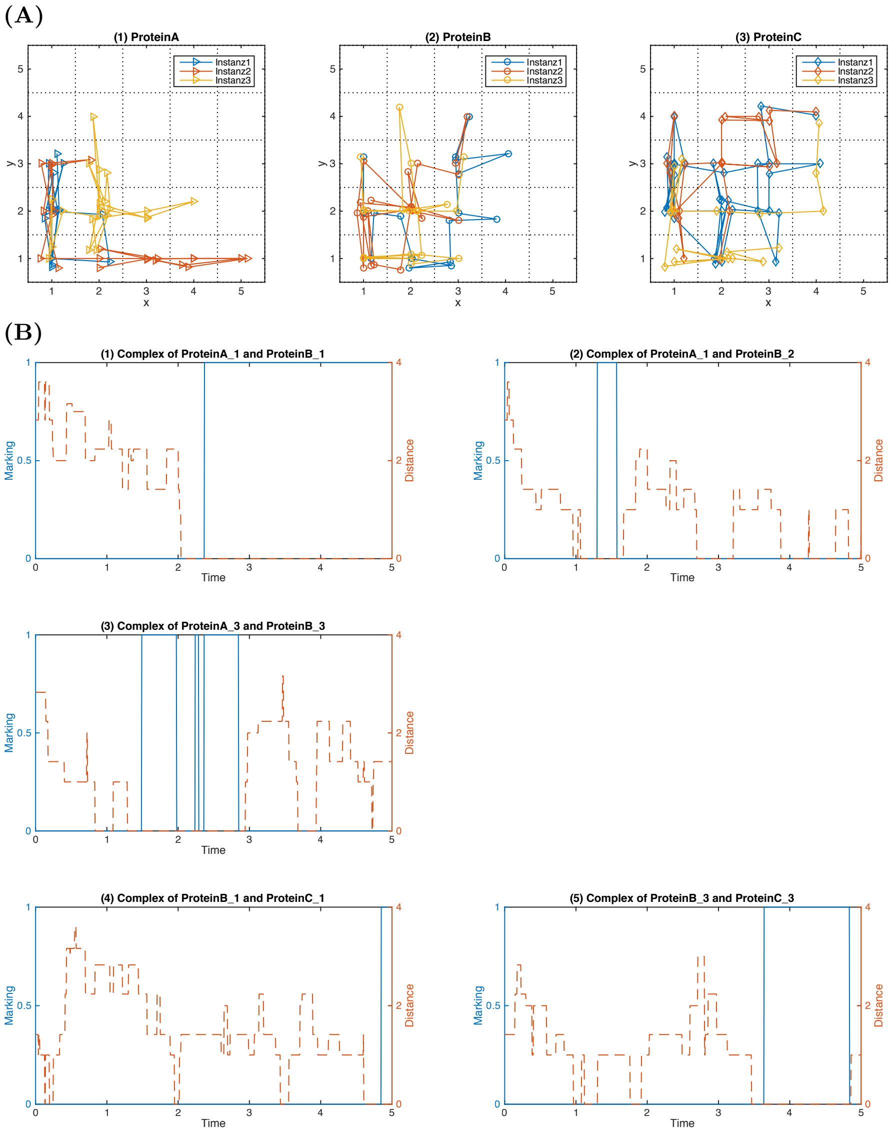


Fig. 9: Stochastic Simulation of the Spatial Model.

ponent in the form of a Petri net. To transform a flat modular composed model into a spatial model the following steps have to be performed: (1) components have to be equipped with individual spatial attributes to represent their localisation and movement in the biomolecular system, (2) components are only allowed to interact if they fulfil certain neighbourhood conditions, (3) the movement of components depends on their state of interaction, e.g. interacting components forming a complex can only move as one entity. The use of coloured Petri nets in our approach allows us to represent individual numbers of module instances for each component.

In our approach we use d different places per module to hold the spatial informations. The value of d is usually 1, 2 or 3 for one-, two- or three-dimensional space. The position of a module is characterized by the number of tokens on these places, e.g. $ProteinA\_X = 3$ and $ProteinA\_Y = 2$ is position (3,2) in two-dimensional space. Furthermore, we add transitions to each module to enable movement and interaction of modules. The structure of the non-spatial modules remains the same, while converting it into a spatial module. So it is possible to revert it back again easily.

The use of places holding spatial information does not restrict our approach to discrete space, but allows us to model continuous space as well by using continuous places. This is not possible using the grid-places approach presented in Section 2.2.

The whole process does not depend on the module and can be applied easily to any module of the BMKdb. Thus it fits quite well in the BMK framework presented in Section 2.1. The spatial transformation algorithm will be a new feature in the next release of the BMK online tool.

Further investigation is needed in terms of simulation. Adding space surely increases the computational complexity and the question is, how can we challenge this.

References

1. Heiner, M., Gilbert, D.: Biomodel engineering for multiscale systems biology. Progress in Biophysics and Molecular Biology **111**(2-3) (April 2013) 119–128
2. Dada, J.O., Mendes, P.: Multi-scale modelling and simulation in systems biology. Integrative Biology **3**(2) (February 2011) 86–96
3. Qu, Z., Garfinkel, A., Weiss, J.N., Nivala, M.: Multi-scale modeling in biology: How to bridge the gaps between scales? Progress in Biophysics and Molecular Biology **107**(1) (October 2011) 21–31
4. Blätke, M.A., Dittrich, A., Rohr, C., Heiner, M., Schaper, F., Marwan, W.: JAK/STAT signalling–an executable model assembled from molecule-centred modules demonstrating a module-oriented database concept for systems and synthetic biology. Molecular BioSystems **9**(6) (2013) 1290–1307
5. Rohr, C., Marwan, W., Heiner, M.: Snoopy–a unifying Petri net framework to investigate biomolecular networks. Bioinformatics **26**(7) (2010) 974–975
6. Marwan, W., Blätke, M.A.: A module-based approach to biomodel engineering with Petri nets. In: Proceedings of the 2012 Winter Simulation Conference (WSC 2012), Berlin. 978-1-4673-4781-5/12, IEEE (2012)
7. Blätke, M.A., Heiner, M., Marwan, W.: Predicting phenotype from genotype through automatically composed Petri nets. In: Proc. 10th International Conference on Computational Methods in Systems Biology (CMSB 2012), London. Volume 7605 of LNCS/LNBI., Springer (2012) 87–106

8. Jehrke, L.: Modulare Modellierung und graphische Darstellung boolscher Netzwerke mit Hilfe automatisch erzeugter Petri-Netze und ihre Simulation am Beispiel eines genregulatorischen Netzwerkes (2014)
9. Soldmann, M.: Transformation monolithischer SBML-Modelle biomolekularer Netzwerke in Petri-Netz Module (2014)
10. Heiner, M., Herajy, M., Liu, F., Rohr, C., Schwarick, M.: Snoopy – a unifying Petri net tool. In: Proc. PETRI NETS 2012. Volume 7347 of LNCS., Springer (June 2012) 398–407
11. Liu, F.: Colored Petri Nets for Systems Biology. PhD thesis, BTU Cottbus, Dep. of CS (January 2012)
12. Liu, F., Heiner, M., Yang, M.: Colored Petri Nets for Multiscale Systems Biology – Current Modeling and Analysis Capabilities in Snoopy. In: Proc. 7th International Conference on Systems Biology (ISB 2013), IEEE (August 2013) 24 – 30
13. Liu, F., Heiner, M.: 9. In: Petri Nets for Modeling and Analyzing Biochemical Reaction Networks. Springer (2014) 245–272
14. Gilbert, D., Heiner, M., Liu, F., Saunders, N.: Colouring Space - A Coloured Framework for Spatial Modelling in Systems Biology. In Colom, J., Desel, J., eds.: Proc. PETRI NETS 2013. Volume 7927 of LNCS., Springer (June 2013) 230–249
15. Liu, F., Blätke, M., Heiner, M., Yang, M.: Modelling and simulating reaction–diffusion systems using coloured Petri nets. Computers in Biology and Medicine **53** (October 2014) 297–308 online July 2014.
16. Gao, Q., Gilbert, D., Heiner, M., Liu, F., Maccagnola, D., Tree, D.: Multiscale Modelling and Analysis of Planar Cell Polarity in the Drosophila Wing. IEEE/ACM Transactions on Computational Biology and Bioinformatics **10**(2) (2013) 337–351 online 01 August 2012.
17. Parvu, O., Gilbert, D., Heiner, M., Liu, F., Saunders, N.: Modelling and Analysis of Phase Variation in Bacterial Colony Growth. In Gupta, A., Henzinger, T., eds.: Proc. CMSB 2013. Volume 8130 of LNCS/LNBI., Springer (September 2013) 78–91
18. Fick, A.: V. on liquid diffusion. Philosophical Magazine Series 4 **10**(63) (1855) 30–39

On modeling internal organs and meridian system based on traditional Chinese medicine

Qi-Wei Ge[1], Ren Wu[2], and Mitsuru Nakata[1]

[1] Yamaguchi University, 1677-1 Yoshida, Yamaguchi-shi 753-8513, Japan
[2] Yamaguchi Junior College, 1346-2 Daido, Hofu-shi 747-1232, Japan
gqw@yamaguchi-u.ac.jp, wu@yamaguchi-jc.ac.jp, mnakata@yamaguchi-u.ac.jp

Abstract. In traditional Chinese medicine, internal organs imply five viscera and six bowels. Five viscera mean liver, heart, spleen, lung and kidney, and six bowels mean gallbladder, small intestine, stomach, large intestine, urinary bladder and triple energizer. Also, meridian system represents the passage of metabolites in the human body. In this paper, we deal with construction of Petri net model of internal organs and meridian based on traditional Chinese medicine. At first, we introduce relations of mutual generation and mutual restriction between five viscera to make a basic Petri net model of five viscera based on five-elements theory. Analyzing the relation between five viscera and six bowels, we propose a model of internal organs that include five viscera and six bowels. After that, through investigating the syndrome of internal organs as well as the function of pericardium meridian, we propose a Petri net model including internal organs and meridians by combining the model of internal organs with meridians. Finally, we do simulation of the proposed model by using CPN Tools to show how our model works.

Keywords: traditional Chinese medicine, five-elements theory, five viscera and six bowels, meridian system, modeling, Petri net

1 Introduction

Traditional Chinese medicine or oriental medicine has been widely applied in treating disease in China as well as in various asian countries since ancient times. This is because of its less secondary effect and possible curing for ahead sick and incurable disease. Especially acupuncture and moxibustion therapy that stimulate acupuncture points in meridian system to treat disease have spread rapidly since the times when acupuncture and moxibustion therapy were admitted by WHO in 1989 and 361 acupuncture points were standardized by WHO in 2006. However mechanism of meridian system is still not scientifically elucidated and many of related researches and treatments have been made empirically and clinically. Therefore it is required to develop new knowledge to elucidate acupuncture and moxibustion treatment [1].

In traditional Chinese medicine, internal organs include five viscera and six bowels. Five viscera mean liver, heart, spleen, lung and kidney, and six bowels

M Heiner, AK Wagler (Eds.): BioPPN 2015, a satellite event of PETRI NETS 2015,
CEUR Workshop Proceedings Vol. 1373, 2015.

mean gallbladder, small intestine, stomach, large intestine, urinary bladder and triple energizer. Five viscera and six bowels imply the systematic functions of human body rather than internal organs of human anatomy. Five viscera and six bowels are closely related each other and have the correspondence relationship between liver and gallbladder, heart and small intestine, spleen and stomach, lung and large intestine, kidney and urinary bladder, respectively. Once one becomes sick, the other has high possibility of abnormalities. Such relationship is expressed in five-elements theory of traditional Chinese medicine [2]. In Chinese medicine, human body is also thought of a complex and interconnected system, and consists of meridian system that connects skin to inner organs from head to foot. The elements of meridians are considered as the acupuncture points of the body. Stimulating acupuncture points on the body, various diseases can be treated and prevented [2].

Recently, five viscera of traditional Chinese medicine have been studied through modeling and quantitatively analyzing [3]. Fusing five-elements theory and fuzzy system theory, Sun et al. have proposed a fuzzy model (called Sun's model hereafter) of five viscera by focusing on the physiological equilibrium states of liver, heart, spleen, lung and kidney [4]. Based on the evolution law of five viscera, Guo et al. have proposed a quantitative measurement model in order to realize five-elements theory [5]. Nevertheless these models have a common problem that they are difficult to be used in simulating the behaviour of five viscera as well as six bowels. On the other hand, Petri net is a modeling and analyzing tool of systems and can represent and analyze static structure and dynamic behaviour of a system. There have been many success stories on modeling biological systems and elucidating the mechanisms by Petri nets. On modeling and simulation of meridian system using Petri net [6], P.A. Heng et al. have presented an intelligent virtual environment for Chinese acupuncture learning and training using state-of-the-art virtual reality technology in order to develop a comprehensive virtual human model for studying Chinese medicine [7]; J. Pan and M. Zhou have modeled and analyzed meridian system by adopting Petri net methods [8]. However, these studies deals with meridian system only, without taking into account of the internal organs, five viscera and six bowels. We aim to propose a Petri net model for both internal organs and meridian system in order to finally elucidate the mechanism of meridian system as well as the internal organs.

In this paper, we are to propose a method of modeling both internal organs and meridian system by using discrete Petri nets as the first step towards elucidating meridian system. Section 2 introduces five-elements theory in traditional Chinese medicine and gives basic knowledge of Petri nets. Section 3 describes a model of internal organs including both five viscera and six bowels based on five-elements theory. Section 4 introduces a concept of "syndrome" used in traditional Chinese medicine and describes syndrome of five viscera and six bowels as well as the function of a meridian, pericardium meridian, to propose a Petri net model including internal organs and meridian system. Section 5 shows simulation of the proposed model by using CPN Tools [9] to show how our model works.

2 Five-Elements Theory and Petri Nets

2.1 Five-Elements Theory

According to ancient Chinese five-elements theory, the five elements, wood, fire, earth, metal and water are indispensable to the daily life of mankind. And in five-elements theory of traditional Chinese medicine, five viscera, liver, heart, spleen, lung and kidney, are mapped to the five elements respectively. Liver flows Qi through over the body free of all care, as a tree getting taller; Heart warms the body as fire; Spleen produces nutrients, as soil that produce all things; Lung takes down Qi and Bodily Fluid, as astringent action of the metal; Kidney pools Mind and adjusts the moisture of the body, as water that flows to the low place from on high [2].

In five-elements theory, there are generation and restriction relationships between five viscera. Generation is that of mother-to-child relationship to give birth to the other party and is circulating in the order of wood → fire → earth → metal → water [2]. Restriction is to suppress the other party in the order of wood → earth → water → fire → metal [2]. In traditional Chinese medicine, health is maintained if generation and restriction relationships are balanced, and hence cause of the disease and methods of treatment can be investigated from the interrelationship of five viscera. Fig.1 shows the generation and restriction relationships.

In addition, five viscera and six bowels have relations that interact with each other. That is, liver and gallbladder, heart and small intestine, spleen and stomach, lung and large intestine, and kidney and urinary bladder interact with each other, respectively. Triple energizer consists of upper energizer, middle energizer and lower energizer, which are the paths for Qi and Bodily Fluid to pass. Since triple energizer does not corresponds to any one of five viscera, we are not to deal with it in this paper.

2.2 Petri Nets and Colored Petri Nets

A Petri net is one of several mathematical modeling languages for the description of concurrent systems [6][10]. A Petri net is a weighted directed bipartite graph and consists of two types of nodes, transitions (i.e. events that may occur, signified by bars) and places (i.e. conditions, signified by circles). Places may contain a number of marks called tokens. Any token distribution over the places will represent a configuration of the net called a marking. The directed arcs with weights describe which places are pre- and/or postconditions for which transitions (signified by arrows). A Petri net is expressed by a 5-tuple $PN = (P, T, A, W, M_0)$. Here, P=$\{p_1, p_2, ...p_{|P|}\}$ is a set of places, $T = \{t_1, t_2, ...t_{|T|}\}$ is a set of transitions, $A \subseteq (P \times T) \cup (T \times P)$ is a set of arcs, W is weight function $A \mapsto \{1, 2, ...\}$ and M_0 is initial marking $P \mapsto \{0, 1, 2, ...\}$.

Colored Petri nets (CPN) is extended from Petri nets by adding colors to tokens and is a discrete-event modeling language combining the capabilities of Petri nets with the capabilities of a high-level programming language. It allows

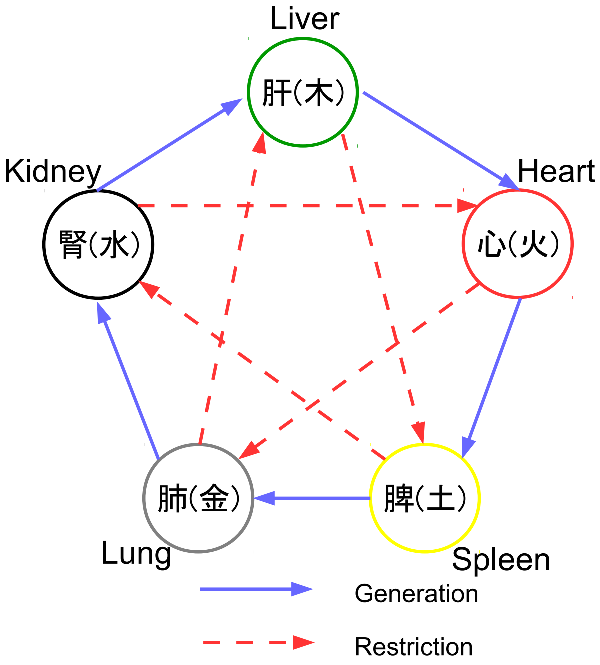


Fig. 1. Generation and restriction relationships between five viscera.

tokens to have a data value attached to them. This attached data value is called token color. A Petri net is a tuple $CPN{=}(P, T, A, \Sigma, C, N, E, G, I)$ [11], where, P, T and A are the same as Petri net, Σ is a set of color sets and contains all possible colors, operations and functions. C is a color function and maps places into colors. N is a node function and maps A into $(P{\times}T){\cup}(T{\times}P)$. E is an arc expression function and maps each arc into the expression. G is a guard function and maps each transition into guard expression. I is an initialization function and maps each place into an initialization expression.

3 Construction of Petri Net Model of Internal Organs

3.1 A Control Model of Five Viscera

In Sun's model [4], physiological equilibrium states are quantitatively defined in domain $(-1, 1)$ individually for liver, heart, spleen, lung and kidney, and a fuzzy model had been proposed based on five-elements theory. The domain is divided into $(-1, b_1)[b_1, a_1)[a_1, a_2](a_2, b_2](b_2, 1)$ as shown in Fig.2, which respectively five states, weak, little weak, equilibrium, little strength, strength. These five states respectively represent dysfunction and no power of generation $((-1, b_1))$, delicate health and weak power of generation $([b_1, a_1))$, health and stable state $([a_1, a_2])$,

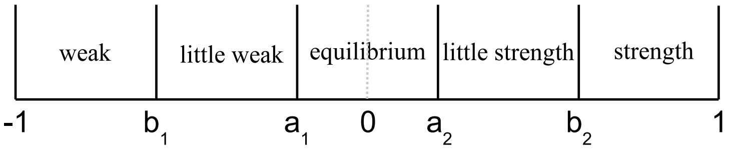


Fig. 2. Five viscera's state.

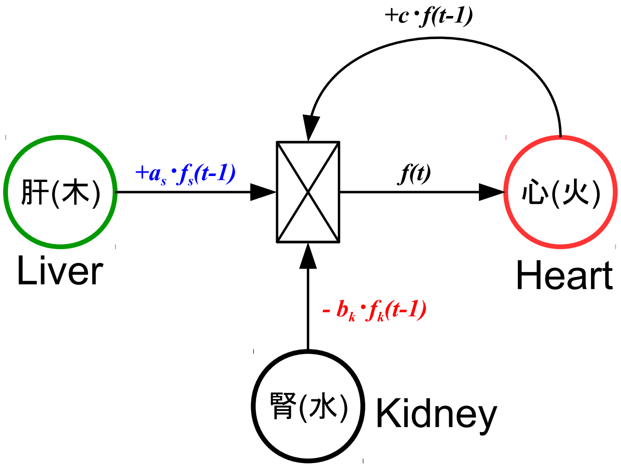


Fig. 3. Generation and restriction relationship.

Excess and disease state with power of restriction ($(a_2, b_2]$), and severe state of excess $(b_2, 1)$.

Among five viscera, there are generation and restriction relationships. In the case of heart, it is generated by liver but restricted by kidney, which is shown in Fig.3. Meanwhile, liver may loss its energy itself. Therefore, the state of liver at time t, $f(t)$, is expressed by the following equation that is modified from Sun's model [4]:

$$f(t) = a_s{\cdot}f_s(t-1) - b_k{\cdot}f_k(t-1) + c{\cdot}f(t-1) \tag{1}$$

where, t is the time, a_s, b_k, c are non-negative parameters, $a_s{\cdot}f_s(t-1)$ represents generation affection (f_s is the state of liver) and $b_k{\cdot}f_k(t-1)$ represents restriction affection (f_k is the state of kidney).

3.2 A Petri Net Model of Five Viscera

Here, we propose a Petri net model of five viscera based on Sun's model. Firstly, we give a model for a single viscus as shown in Fig.4. Places, p_{in} and p_{state}, are

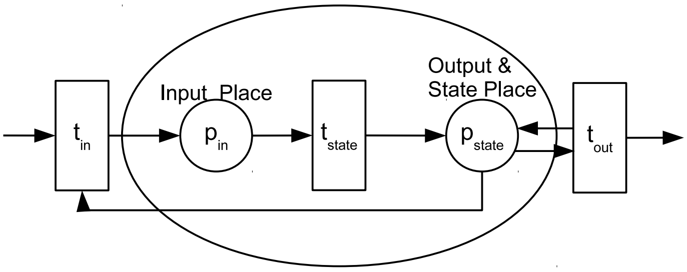


Fig. 4. A Petri net model for a single viscus.

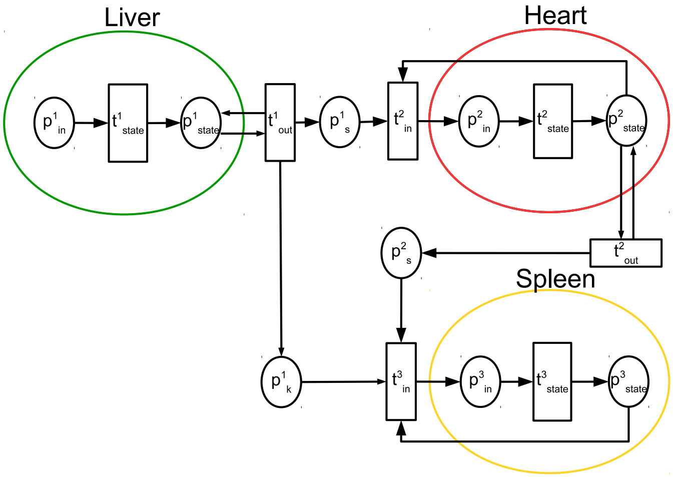


Fig. 5. A model including generation and restriction.

called input place and state place, and the token in state place is called state token whose value expresses the state of the viscus. t_{in} is called input transition that works to calculate the state value of Eq. (1). t_{state} is called state transition and works to generate a state of the viscus. The token with calculated state value passes through p_{in} and t_{state} and then arrive at p_{state}. t_{out} is called output transition and works to generate and restrict other viscera.

Fig.5 shows a Petri net model including relationship of generation and restriction between viscera. This model is comprised of three single viscus models, the models of liver, heart and spleen, which are connected by places p^1_s, p^1_k and p^2_s. Places p^1_s and p^1_k express liver's generation and restriction affection to heart and spleen, respectively, and these two places are respectively called generation-output place and restriction-output place of liver. Tokens in generation-output and restriction-output places are respectively called generation token and restriction token. Similarly, p^2_s is generation-output place of heart.

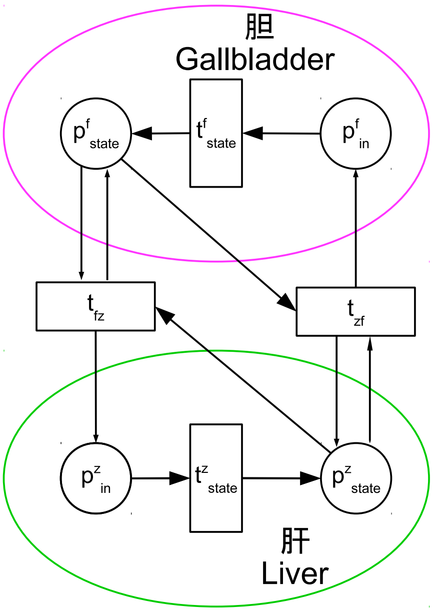


Fig. 6. A Petri net mode of liver and gallbladder.

Since liver generates heart and restricts spleen, state token of liver flows through output transition t^1_{out} to generation-output place p^1_s (that is also an input place of heart model) and to restriction-output place p^1_k (that is also an input place of spleen model). In the same way, state token of heart flows to generation-output place p^2_s that is an input place of spleen model.

3.3 A Petri Net Model of Internal Organs

The Petri net shown in Fig.6 can be treated as a model of any one pair of five viscera and six bowels (say liver and gallbladder). Hence to make a full model that includes both five viscera and six bowels, we need only to consider how to connect these models together into one model.

Five viscera and six bowels are in the relationship of the front and back, such as liver and gallbladder, heart and small intestine, spleen and stomach, lung and large intestine, and kidney and urinary bladder. Each of these pairs interacts with each other to maintain life. Such a pair, for example liver and gallbladder, is modeled by Petri net as shown in Fig.6. This model is made by adding transitions t_{zf} and t_{fz} and connecting them to the single model of liver and gallbladder. t_{zf} and t_{fz} represent the affections from liver to gallbladder and from gallbladder to liver, respectively.

Synthesizing the models we have made till now, we can complete a model of internal organs. The process is summarized as follows: (1) Make a single model

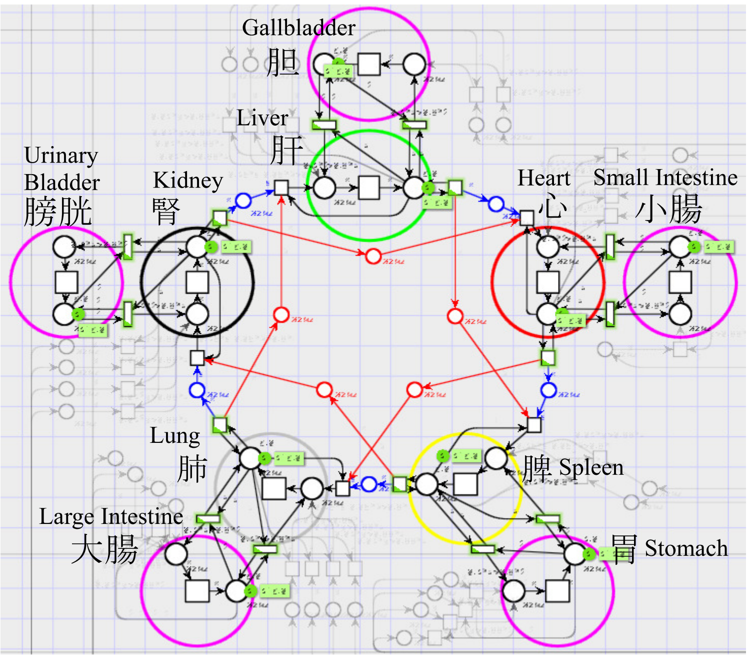


Fig. 7. A complete Petri net model of internal organs.

for each of five viscera as Fig.4; (2) Connect these single models according to generation and restriction relationship between five viscera as Fig.5; (3) Add to the model of each viscus by the model of its pair partner of six bowels as Fig.6. Then the complete model is obtained as shown in Fig.7.

4 A Combined Model of Internal Organs and Meridian Based on Syndrome

4.1 Meridians and Acupuncture Points

In traditional Chinese medicine, meridian system is considered as important channels to transfer Qi (means energy) and blood in human body. Meridian system is divided into meridians and collaterals [12], [13]. Meridians are main part of meridian system and represent paths that trend almost upside down. Collaterals play the role of branches and connect all the way to the whole body.

Meridians consist of twelve principal meridians and eight extra meridians. Twelve principal meridians include three Yin and three Yang meridians of hand and also three Yin and three Yang meridians of foot. Here, Yin and Yang are the concept of duality forming a whole and mean respectively sunny side and shady side (e.g., the palm and the back of the hand). Yin meridians belong to

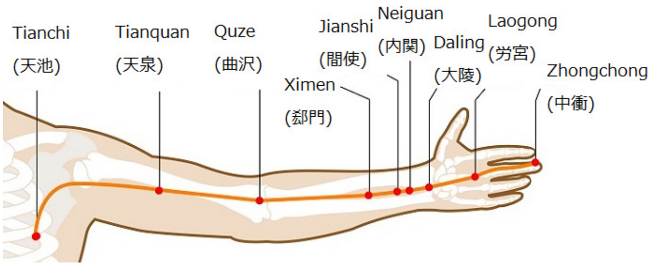


Fig. 8. Pericardium meridian containing nine acupuncture points (from [14]).

five viscera and Yang meridians belong to six bowels. Each principal meridian has acupuncture points. In eight extra meridians, only Renmai and Dumai have acupuncture points. Twelve principal meridians along with Renmai and Dumai are thought of main meridians [12]. Collaterals consist of fifteen collaterals and other collaterals (tiny collateral, superficial collaterals and musculature that is not connected to internal organs).

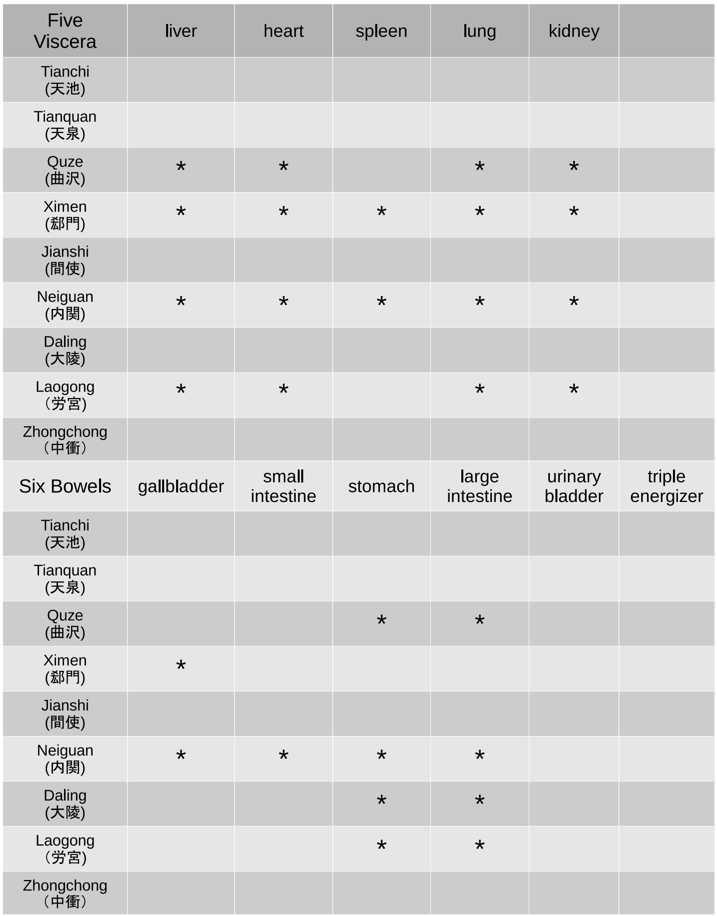
Acupuncture points exist in twelve principal meridians, Renmai and Dumai. Through these acupuncture points, Qi passes inside and outside of body and thus various diseases can be prevented and cured by stimulating these acupuncture points [2, 12, 13]Here, we consider the modeling of pericardium meridian. Pericardium meridian is an important Yin meridian of hand and is closely related to heart. It begins near the heart, goes down each arm to the palms and then goes to the tip of the middle finger. As can be found in Fig.8, pericardium meridian contains nine acupuncture points, Tianchi, Tianquan, Quze, Ximen, Jianshi, Neiguan, Daling, Laogong and Zhongchong.

4.2 Relationship between Internal Organs and Pericardium Meridian Based on Syndrome

In traditional Chinese medicine, "syndrome" is a measure to evaluate the state and physical condition of human body [2, 12, 13]. The acupuncture points of pericardium meridian are efficacious against various symptoms. For example, Quze corresponds thirsty; Ximen heartbeat, shortness of breath and tenosynovitis; Neiguan motion sickness, hangover, hiccup and chronic gastritis; Daling rheumatoid arthritis and halitosis; Laogong stomatitis, thirsty and forgetful; and so on [15].

On the other hand, when the states of internal organs are no longer normal, various functions will be modulated by each organ. For example, in liver the dispersing and dredging function and the capability to store the blood are probably decreased; in heart abnormalities may occur in the blood stream; in spleen the functions of transportation and transformation, which send up the lucid Yang and govern the blood, may be decreased; in lung dissipating and exerting effect may become weak; and in kidney functions of storing the essence of

Table 1. The relationship between internal organs and acupuncture points of pericardium meridian.



Five Viscera	liver	heart	spleen	lung	kidney	
Tianchi (天池)						
Tianquan (天泉)						
Quze (曲沢)	*	*		*	*	
Ximen (郄門)	*	*	*	*	*	
Jianshi (間使)						
Neiguan (内関)	*	*	*	*	*	
Daling (大陵)						
Laogong （劳宮)	*	*		*	*	
Zhongchong （中衝）						
Six Bowels	gallbladder	small intestine	stomach	large intestine	urinary bladder	triple energizer
Tianchi (天池)						
Tianquan (天泉)						
Quze (曲沢)			*	*		
Ximen (郄門)	*					
Jianshi (間使)						
Neiguan (内関)	*	*	*	*		
Daling (大陵)			*	*		
Laogong （劳宮)			*	*		
Zhongchong （中衝）						

life and regulating the metabolism of water are probably decreased. In the case of six bowels, the abnormalities related to digest, absorption and excretion may probably occur. [2, 12, 13].

Combining these syndromes due to the modulations of internal organs with the effect of acupuncture points of pericardium meridian, we can make a table representing the relationship between each internal organ and each acupuncture point of pericardium meridian as shown in Table.1, in which notion "*" shows the efficacious relations. For example, the acupuncture point Quze is supposed to be efficacious against the symptoms related to liver, heart, lung and kidney, as well as stomach and large intestine.

4.3 Modeling Internal Organs and Pericardium Meridian

Here, we construct a model of pericardium meridian and combine it with the model of internal organs proposed in the last section according to Table 1. As we have stated, meridians and acupuncture points connect skin and inner organs of body. Stimulating an acupuncture point, the signal is transmitted to the meridian and then circulation of Qi and flood is improved, which makes internal organs active. In the case of Quze, when it is stimulated, liver, heart, lung and kidney, as well as stomach and large intestine, are activated.

Based on the above, we modeled the influence of a single acupuncture on a single organ as shown in Fig.9. Acupuncture point is represented by a transition, stimulation of the acupuncture point is represented by firing the transition. Fitting together with state token of the single organ, the token generated by the firing goes through transition t_p to the input place of the organ, and it further moves to the state place (Output&State Place). Thus activation of the organ is represented.

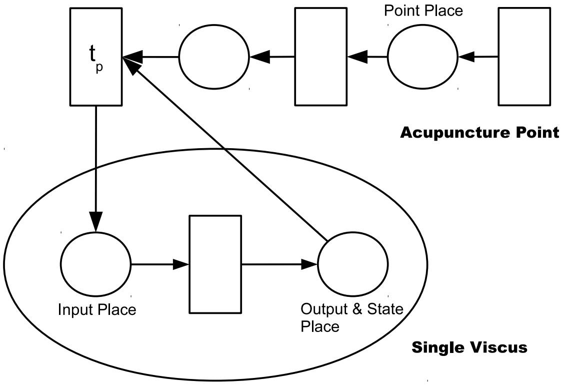


Fig. 9. A combined model of single acupuncture point and single organ.

Applying the same way as Fig.9 for all the organs according to Table 1, we finally get a complete model of internal organs and pericardium meridian as shown in Fig.10. Note that plural tokens generated by plural acupuncture points are accumulated at one place.

5 Simulation using CPN Tools

We have done simulation by using CPN Tools [9] for the model of Fig.10. The parameters are set as follows:

(1) Data type for each place is defined by REAL;

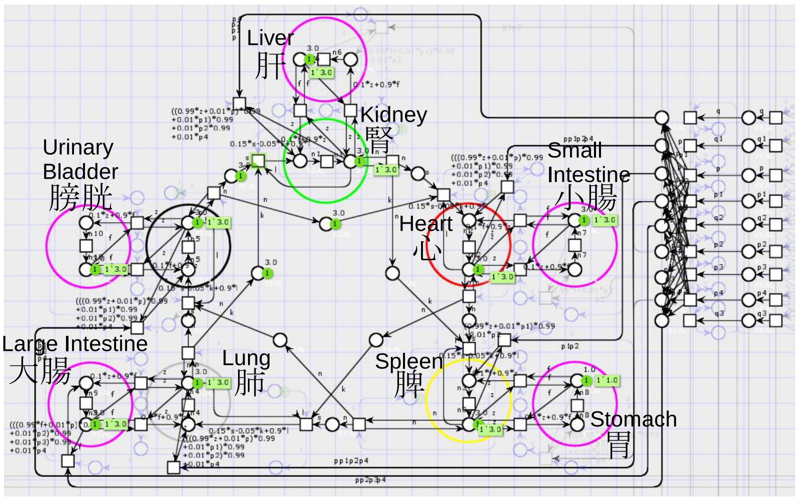


Fig. 10. A complete Petri net model of internal organs and pericardium meridian.

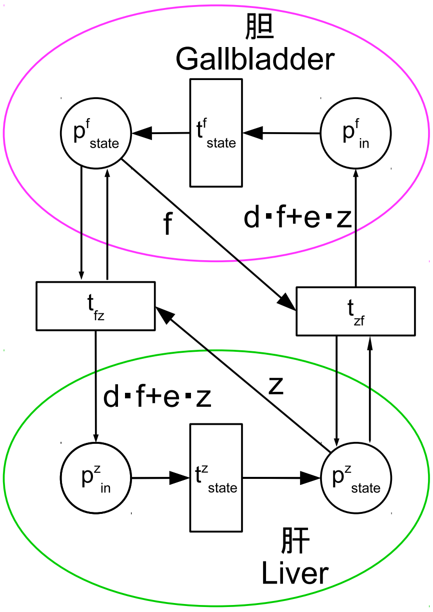


Fig. 11. Parameter setting for model of liver and gallbladder.

(2) Value of state token of each organ is defined in domain $(0.5, 5.5)$, which means that $(0.5, 1.5)[1.5, 2.5)[2.5, 3.5](3.5, 4.5](4.5, 5.5)$ respectively five states, weak, little weak, equilibrium, little strength, strength;

(3) The parameters a_s, b_k, c in equation $f(t) = a_s{\cdot}f_s(t-1) - b_k{\cdot}f_k(t-1) + c{\cdot}f(t-1)$ are defined as $a_s = 0.15 b_k = 0.05 c = 0.90$;

(4) For each pair of five viscera and six bowels (e.g. liver and gallbladder as shown in Fig.11), set expression z for arc (p^z_{state}, t_{fz}), f for (p^f_{state}, t_{zf}), $d{\cdot}f + e{\cdot}z$ for (t_{zf}, p^f_{in}) and $d{\cdot}z + e{\cdot}f$ for (t_{fz}, p^z_{in}), where d and e are defined as $d = 0.90 e = 0.10$;

(5) One stimulation of each puncture point provides 0.01 active influence on the related organ.

Note that all the constants are temporarily decided for the simulation.

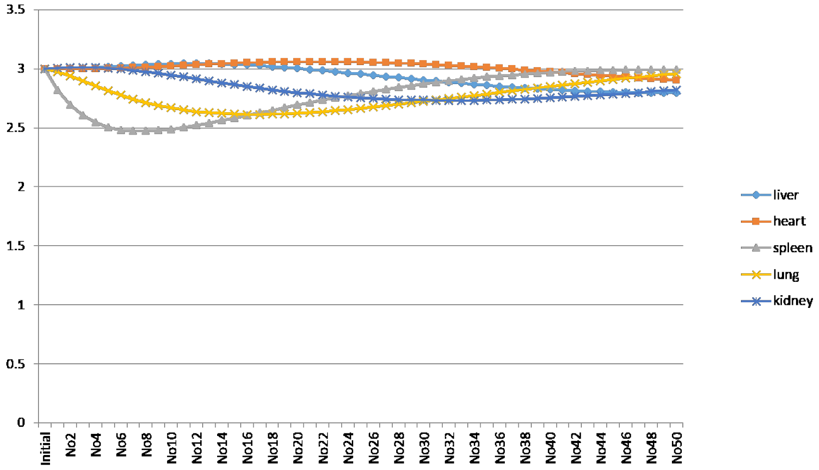


Fig. 12. Change of states of five viscera.

Simulations have been done based on the above parameters. We suppose that initially stomach is in weak state with state token value “1” and all the other organs are in normal state with “3”. Among acupuncture points, Quze, Neiguan, Daling and Laogong have influence on stomach, and we choose Daling to do the stimulation for 50 times. Simulation results are shown in Fig.12 and Fig.13.

From Fig.12 and Fig.13 we can observe the following phenomenons. At first, weak stomach adversely affects spleen directly, and thus spleen weakens rapidly and its state value becomes lower than 2.5. Due to weakening of spleen and also the relationship of generation and restriction between five viscera, all the viscera gradually weaken except heart. On the other hand, spleen directly strengthens stomach and the acupuncture point, Daling, is stimulated continuously. These

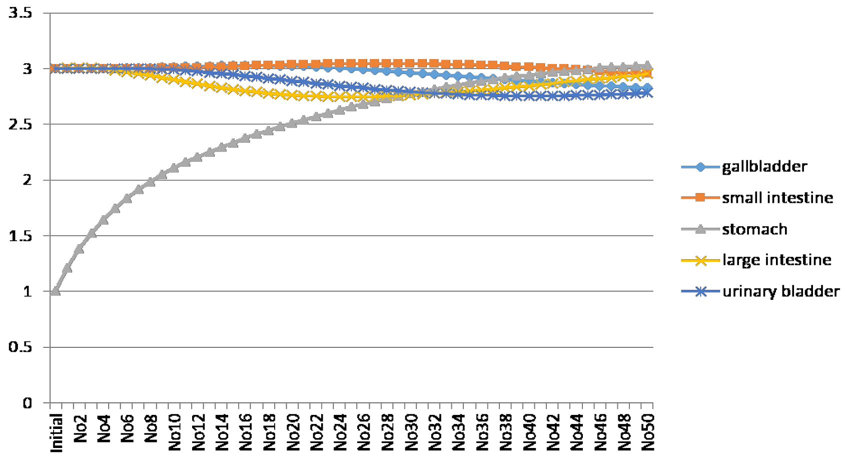


Fig. 13. Change of states of six bowels.

actions strengthen stomach rapidly and as the result stomach recovers its state value to 2.5 when Daling is stimulated 22 times. After that, all the organs recover gradually and finally get to health state with state values almost 3 when Daling is stimulated 50 times. Of particular interest is that any organs of six bowels are almost not affected by stomach except large intestine.

6 Conclusions

We have proposed a method of constructing Petri net model for internal organs and meridians based on traditional Chinese medicine. According to five-elements theory and the control model proposed by Sun et al., we have proposed a Petri net model of five viscera by considering the generation and restriction relationship between five viscera. Taking into account of the relationship of front and back between five viscera and six bowels (except triple energizer), we have made a model of internal organs. Through investigating the syndrome of internal organs as well as the function of pericardium meridian, we have proposed a Petri net model including internal organs and meridians by combining the model of internal organs with meridians. Finally, we do simulation to show how our model works by using CPN Tools.

It should be pointed out that this work does not intend to be a scientific contribution to medical science in the sense as it is usually understood. As the future works, we are to (1) decide parameters for all the transitions and places, as well as for tokens and arcs on the basis of the data of acupuncture treatment site; (2) do simulation to verify the validity of the proposed model and the parameters, in order to improve the model; (3) develop a method to construct a Petri net model for the whole human body.

Acknowledgement

The authors would like to thank Ms. Misato Maesaka for her efforts in doing simulation and handling simulation data. This work was supported by JSPS KAKENHI Grant Number 25540135 (Grant-in-Aid for Challenging Exploratory Research).

References

1. S. Shinohara, “A proposal of the 2nd Japan meridian and point committee”, IDO NO NIPPON, vol.71, no.5, pp.142-154 (2012) (in Japanese)
2. A. Hyodo, *The Visual Encyclopedia of East Asian Medicine*, Shinsei Publishing (2012) (in Japanese).
3. X.W. Li, Y.M. Wang, X. Liu and Y. Zhang, “The summarization on the quantitative models of five elements”, BME & Clin Med., vol.16, no.4, pp.411-414 (2012) (in Chinese)
4. C.L. Sun, X.Y. Li and L.C. Zhao, “Fuzzy modeling and analysis based on five elements theory for the system of five organs system”, Journal of Anshan Normal University, vol.13, no.6, pp.1-4 (2011) (in Chinese)
5. W.Y. Guo, J.Q. Wu and S.Wang, “Five elements system modeling and solving”, Journal of Shanghai Second Polytechnic University, vol.25, no.4, pp.253-256 (2008) (in Chinese).
6. J.L. Peterson, *Petri Net Theory and the Modeling of Systems*, Prentice Hall, Englewood Cliffs, NJ, USA (1981).
7. P.A. Heng, T.T. Wong, R. Yang, Y.P. Chui, Y. Xie, K.S. Leung and P.C. Leung, “Intelligent inferencing and peptic simulation for Chinese acupuncture Learning and Training”, IEEE Trans. Inf. Technol. Biomed., vol. 10, no. 1, pp. 28-41 (2006)
8. J. Pan and M. Zhou, “Modeling and analysis of meridian systems using Petri nets”, International Journal of Intelligent Control and Systems, vol. 10, no. 3, pp. 226-234 (2005)
9. http://cpntools.org/start
10. T. Murata, “Petri nets: properties, analysis and applications”, IEEE Proceedings, vol. 77, no. 4, 541-580 (1989).
11. K. Jensen, *Coloured Petri Nets* (2 ed.), Berlin: Heidelberg (1996).
12. X.M. Shi, *Acupuncture and Moxibustion*, Beijing: China Press of Traditional Chinese Medicine (2002) (in Chinese).
13. X.Y. Shen, *Science of Meridian and Acupuncture*, Beijing: China Press of Traditional Chinese Medicine (2007) (in Chinese).
14. http://www.all-about-acupuncture.com/meridian-charts.html
15. http://www.sennenq.co.jp/top.html

Full structural model refinement as type refinement of colored Petri nets

Diana-Elena Gratie[1,2,3] and Ion Petre[1,2,3]
{dgratie,ipetre}@abo.fi

[1] Turku Centre for Computer Science
[2] Computational Biomodeling Laboratory
[3] Department of Computer Science, Åbo Akademi University
Joukahaisenkatu 3-5, FIN-20520 Turku

Abstract. In this paper we propose a method for implementing a full structural model refinement of a (biological) model represented as a (colored) Petri net. We build on the full structural data refinement definition of C. Gratie and Petre, and the type refinement of colored Petri nets introduced by Charles Lakos. Given a (biological) reaction-based model and a desired full structural refinement of it, we propose a general coloring scheme for a colored Petri net implementation of the model and give an algorithm for adding the refinement details in the Petri net model. We then prove that the construction is a type refinement, and that by our choice of color sets the resulting refined colored Petri net implements the full structural refinement of the given model.

Keywords: Colored Petri nets, type refinement, reaction network, structural model refinement.

1 Introduction

Model refinement, the process of adding more details to an existing model, is an important step in the model building cycle. Many refinement methods have been proposed for different modeling frameworks and formalisms, e.g., action systems [1], Petri nets [17, 11], kappa [4], biochemical reaction networs [7], π-calculus [16], etc. We bridge here two modelling frameworks and their respective ways of implementing refinement, namely reaction network models with structural refinement and colored Petri nets with type refinement.

Type refinement of colored Petri nets has been introduced in [11], and consists of refining the color sets of places such that the new color sets are polymorphic with the initial color sets. The authors see this as adding some supplementary data to a given data type represented as a color set, e.g. include in the entry of a book in a library not only its title and authors, but also the maximum number of days it can be borrowed.

The concept of *(full) structural refinement* of a reaction network (bio-)model has been introduced in [7] (where it was called data refinement), with a focus on

M Heiner, AK Wagler (Eds.): BioPPN 2015, a satellite event of PETRI NETS 2015,
CEUR Workshop Proceedings Vol. 1373, 2015.

an ODE-based representation of a model and its refinement. A sufficient condition for the refined model to preserve the fit of the original one was discussed in [6] for mass-action models. We follow in this paper the terminology of [6]. We use the main concepts of *species refinement* and *(full) structural refinement* for models represented as (colored) Petri nets, and give a methodology for implementing full structural refinements as type refinements of colored Petri nets. An approach to implementing model refinement in the colored Petri net framework has been exemplified for a model of the eukaryotic heat shock response mechanism in [8]. The authors present there two coloring schemes that can be used for the particular refinement they were implementing. We derive here a general coloring scheme for model refinement that can be used when implementing a full structural data refinement of a model.

We assume the reader is familiar with (colored) Petri nets, but we recall some of the basic definitions so that the paper is self-contained.

The paper is structured as follows: in Section 2 we present reaction network (also called reaction-based) models and the notions of *species refinement* and *(full) structural refinement* of such models, with a discussion on the explosion of the model induced by a refinement, in terms of number of species and reactions that the initial model refines to. In Section 3 we recall some notions and notations for Petri nets and their colored version, give a coloring scheme and discuss how a reaction network model can be implemented as a (colored) Petri net. We continue in Section 4 with proposing a type refinement based on a refinement relation ρ and prove that the chosen type refinement results in a colored Petri net that is the implementation of the full structural $\rho-$refinement of the initial model. We draw our conclusions and discuss about the model size and successive refinements in Section 5.

2 Model Refinement

In systems biology, model refinement comprises two aspects: the structural side and the quantitative side. The structural side handles the newly introduced species and presents a methodology for computing the new set of reactions, while the quantitative side deals with changes in the kinetic constants of the model and ways of setting the new parameters in such a way that previous data is used. *Quantitative model refinement* was introduced in [15, 4] for rule-based models, and for reaction-based models in [13, 7]. We recall here the structural refinement of reaction network models, as presented in [7] and based on the terminology of [6]. We are only interested in the structural refinement, so we will not focus on any quantitative details.

A *reaction-based model* M consists of a finite set of *species* $\mathscr{S} = \{A_1, \ldots, A_m\}$ and a finite set of *reactions* $\mathscr{R} = \{r_1, \ldots, r_n\}$ using only species in $\mathscr{S}$. A reaction $r_j \in \mathscr{R}$ can be formulated as a rewriting rule of the form:

$$r_j \ : \ c_{1,j}A_1 + \ldots + c_{m,j}A_m \xrightarrow{k_{r_j}} c'_{1,j}A_1 + \ldots + c'_{m,j}A_m, \qquad (1)$$

with the meaning that $c_{i,j}$ copies of species A_i are consumed by the reaction and $c'_{i,j}$ copies of species A_i are produced, $i = 1..m$. Constants $c_{1,j}, \dots, c_{m,j}, c'_{1,j}, \dots, c'_{m,j} \in \mathbb{N}$ are the *stoichiometric coefficients* of r_j and $k_{r_j} \geq 0$ is the *kinetic rate constant* of reaction r_j. We denote by $\mathbf{r_j^-} = (c_{1,j}, \dots, c_{m,j})$ the vector of stoichiometric coefficients on the left hand side of the reaction, for the species being consumed in reaction r_j, and by $\mathbf{r_j^+} = (c'_{1,j}, \dots, c'_{m,j})$ the vector of stoichiometric coefficients on its right hand side, those of species being produced. Without a risk of ambiguity, reaction r_j can then be written as $\mathbf{r_j^-} \xrightarrow{k_{r_j}} \mathbf{r_j^+}$.

Example 1. A biological system with two irreversible reactions that encode the dimerization of a molecule P can be represented as a reaction-based model $M = (\mathscr{S}, \mathscr{R})$ where $\mathscr{S} = \{P, P_2\}$ and $\mathscr{R} = \{2P \to P_2,\ P_2 \to 2P\}$. P represents the monomeric molecule and P_2 is the dimer that is formed from two P monomers.

Data refinement is the type of refinement of a model that consists in adding details related to the species of the model, i.e., it replaces a species with several of its subspecies. The subspecies may account for post-translational modifications of macromolecules, or distinguish between possible variants of some trait.

All species are considered to be refined at once, thus each species in an initial model is replaced by a non-empty set of refined species to yield a refined model, as dictated by a *species refinement relation* ρ. This is formalized in Definition 1.

Definition 1 ([6]). *Given two sets of species $\mathscr{S}$ and $\mathscr{S}'$, and a relation $\rho \subseteq \mathscr{S} \times \mathscr{S}'$, we say that ρ is a* species refinement relation *iff it satisfies the following conditions:*

1. *for each $A \in \mathscr{S}$ there exists $A' \in \mathscr{S}'$ such that $(A, A') \in \rho$;*
2. *for each $A' \in \mathscr{S}'$ there exists* exactly one *$A \in \mathscr{S}$ such that $(A, A') \in \rho$.*

We denote $\rho(A) = \{A' \in \mathscr{S}' \mid (A, A') \in \rho\}$. We say that all species $A' \in \rho(A)$ are siblings.

Intuitively, each species $A \in \mathscr{S}$ is replaced in the refined model with the set of species $\rho(A)$. For the case where $\rho(A)$ is a singleton set, one may consider that species A does not change, even if its refined counterpart is denoted by a different name in $\mathscr{S}'$; such a refinement of a species is called *trivial.*

Next we recall the definitions of refinement of a vector (of stoichiometric coefficients), of a reaction, and of a reaction-based model.

Definition 2 ([6]). *Let $\mathscr{S} = \{A_1, \dots, A_m\}$ and $\mathscr{S}' = \{A'_1, \dots, A'_p\}$ be two sets of species, and $\rho \subseteq \mathscr{S} \times \mathscr{S}'$ a species refinement relation.*

1. *Let $\alpha = (\alpha_1, \dots, \alpha_m) \in \mathbb{N}^{\mathscr{S}}$ and $\alpha' = (\alpha'_1, \dots, \alpha'_p) \in \mathbb{N}^{\mathscr{S}'}$. We say that α' is a ρ-*refinement *of α if*
$$\sum_{\substack{1 \leq j \leq p \\ A'_j \in \rho(A_i)}} \alpha'_j = \alpha_i, \text{ for all } 1 \leq i \leq m\ .$$
We denote by $\rho(\alpha)$ the set of all $\rho-$refinements of α.

2. *Let $r:\ r^- \to r^+$ and $r':\ {r'}^- \to {r'}^+$ be two reactions over $\mathscr{S}$ and $\mathscr{S}'$, resp. We say that r' is a ρ-*refinement *of r if*
$$ {r'}^- \in \rho(r^-) \text{ and } {r'}^+ \in \rho(r^+) \ . $$
We denote by $\rho(r)$ the set of all $\rho-$refinements of r. Note that $\rho(r) = \rho(r^-) \times \rho(r^+)$.
3. *Let $M = (\mathscr{S}, \mathscr{R})$ and $M' = (\mathscr{S}', \mathscr{R}')$ be two reaction-based models, and $\rho \subseteq \mathscr{S} \times \mathscr{S}'$ a species refinement relation. We say that M' is a ρ-*structural refinement of *M if*
$$ \mathscr{R}' \subseteq \bigcup_{r \in \mathscr{R}} \rho(r) \text{ and } \rho(r) \cap \mathscr{R}' \neq \varnothing \ \forall r \in \mathscr{R} \ . $$
In case $\mathscr{R}' = \bigcup_{r\in\mathscr{R}} \rho(r)$, we say M' is the full structural ρ-refinement *of M, denoted $M' = M_\rho$.*

Model explosion. Note that a vector of coefficients $\alpha' \in \mathbb{N}^{\mathscr{S}}$ that respects the sum condition $\sum_{\substack{1\le j\le p \\ A'_j \in \rho(A_i)}} \alpha'_j = \alpha_i$, for all $1 \le i \le m$ can be seen as a way of choosing α_i elements from a bag containing elements of $|\rho(A_i)|$ types, where the selection may contain several elements of the same type. The total number of different ways in which one may choose k elements from a bag with elements of n types (assuming enough copies of each type are available) is $\left(\!\binom{n}{k}\!\right) = \binom{n+k-1}{k}$, the so-called *multiset coefficient, n multichoose k*.

A reaction r_j of the form (1) can refine to $\prod_{1\le i\le m} \left(\!\binom{|\rho(A_i)|}{c_{i,j}}\!\right) \cdot \left(\!\binom{|\rho(A_i)|}{c'_{i,j}}\!\right)$ different reactions. The number stems from the number of possible ways of choosing $c_{i,j}$ ($c'_{i,j}$, resp.) copies from the possible refinements of a species $A_i \in \mathscr{S}$. The number of reactions in a full structural $\rho-$refinement of a model with n reactions is thus:
$$ \sum_{1\le j\le n} \prod_{1\le i\le m} \left(\!\binom{|\rho(A_i)|}{c_{i,j}}\!\right) \cdot \left(\!\binom{|\rho(A_i)|}{c'_{i,j}}\!\right) \ . $$

Example 2. Consider the reaction-based model $M = (\mathscr{S}, \mathscr{R})$ from Example 1. One possible refinement for this model is to consider that molecule P can be in two states: acetylated ($P^{(1)}$) and non-acetylated($P^{(0)}$). Then the dimer P_2 could have none ($P_2^{(0)}$), one ($P_2^{(1)}$) or both ($P_2^{(2)}$) of its composing monomers acetylated. Consider a set of species $\mathscr{S}' = \{P^{(0)}, P^{(1)}, P_2^{(0)}, P_2^{(1)}, P_2^{(2)}\}$. A relation $\rho \subseteq \mathscr{S} \times \mathscr{S}'$ that would capture such a refinement is $\rho = \{(P, P^{(0)}), (P, P^{(1)}), (P_2, P_2^{(0)}), (P_2, P_2^{(1)}), (P_2, P_2^{(2)})\}$. One can easily see that ρ is a refinement relation, based on Definition 1.

A full structural ρ-refinement of M is the model $M' = (\mathscr{S}', \mathscr{R}')$, where $\mathscr{R}' =$
$$ \begin{array}{lll} \{2P^{(0)} \to P_2^{(0)}, & 2P^{(0)} \to P_2^{(1)}, & 2P^{(0)} \to P_2^{(2)}, \\ 2P^{(1)} \to P_2^{(0)}, & 2P^{(1)} \to P_2^{(1)}, & 2P^{(1)} \to P_2^{(2)}, \\ P^{(0)} + P^{(1)} \to P_2^{(0)}, & P^{(0)} + P^{(1)} \to P_2^{(1)}, & P^{(0)} + P^{(1)} \to P_2^{(2)}, \\ P_2^{(0)} \to 2P^{(0)}, & P_2^{(0)} \to 2P^{(1)}, & P_2^{(0)} \to P^{(0)} + P^{(1)}, \\ P_2^{(1)} \to 2P^{(0)}, & P_2^{(1)} \to 2P^{(1)}, & P_2^{(1)} \to P^{(0)} + P^{(1)}, \\ P_2^{(2)} \to 2P^{(0)}, & P_2^{(2)} \to 2P^{(1)}, & P_2^{(2)} \to P^{(0)} + P^{(1)}\}. \end{array} $$

3 Modeling Biological Systems as (Colored) Petri Nets

Many biological models are implemented as Petri nets due to the graphical, intuitive formalism, and the many simulation strategies they offer. We start our discussion over refinement and implementations of models as Petri nets from the standard version of Petri nets. We then continue with colored Petri nets.

3.1 Preliminaries

We assume the reader is familiar with the basic notions and notations related to Petri nets and we refer to [5], [14] for details. We also assume that the reader is familiar with constructing a standard Petri net associated to a reaction-based model; we refer to [2] for details.

In order to implement a reaction-based model as a Petri net, one represents each species via a place, and each reaction via a transition having as pre-places the places representing the reactants of the reaction, and as post-places the places representing the products of the reaction, with each arc expression being the stoichiometry of the represented species in that reaction, see [2].

Definition 3 (Implementation of a reaction network model as a Petri net). *Given a reaction-based model $M = (\mathscr{S}, \mathscr{R})$, and a Petri net $N = (P, T, A, f, M_0)$ with $|\mathscr{S}| = |P|$ and $|\mathscr{R}| = |T|$, we say that the Petri net N* structurally implements *model M if there exists a bijection $\delta : \mathscr{S} \cup \mathscr{R} \to P \cup T$ mapping species of M into places of N and reactions of M into transitions of N ($\delta(x) \in P$, for all $x \in \mathscr{S}$ and $\delta(x) \in T$ for all $x \in \mathscr{R}$) such that for every reaction $r_j \in \mathscr{R}$ and its corresponding transition $t = \delta(r_j)$ and for every species $S_i \in \mathscr{S}$ the following conditions hold:*

1. *if $c_{i,j} > 0$ then $(\delta(S_i), t) \in A$ and $f(\delta(S_i), t) = c_{i,j}$, otherwise $(\delta(S_i), t) \notin A$;*
2. *if $c'_{i,j} > 0$ then $(t, \delta(S_i)) \in A$ and $f(t, \delta(S_i)) = c'_{i,j}$, otherwise $(t, \delta(S_i)) \notin A$.*

Example 3. An example of a Petri net structural implementation of the model described in Example 1 is given in Figure 1. The bijection δ is defined such that $\delta(P) =$ P_, $\delta(P_2) =$ P_2, $\delta(2P \to P_2) =$ T_fw, $\delta(P_2 \to 2P) =$ T_bw. One can easily see that the arc multiplicities respect the two conditions in Definition 3.

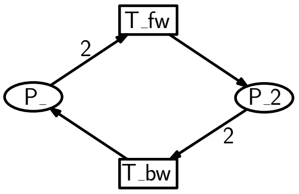


Fig. 1. Standard Petri net structural implementation of a dimerization model (only multiplicities greater than 1 are displayed)

There exist two ways of defining colored Petri nets, one proposed by Kurt Jensen in [9], and an equivalent one adapted from the first definition, by Charles Lakos in [11]. In this paper we consider the definition of colored Petri nets proposed by Lakos because it does not explicitly include transition guards (that we are not using in our construction) and because of the definition of type refinement of colored Petri nets proposed in [11]. We use the following less well known notations: Σ denotes a universe of non-empty color sets with an associated partial order $<:\subseteq \Sigma \times \Sigma$ indicating that values from one color set X with $X <: Y$ can be used in contexts expecting values of Y. Π_Y is a projection function mapping values of X into values of Y. $\Phi\Sigma = \{X \to Y \mid X, Y \in \Sigma\}$ denotes the functions over Σ, and $\mu\mathbf{X} = \{X \to \mathbb{N}\}$ denotes the multisets over X. $E^-, E^+ : \mathbb{Y} \to \mathbb{M}$ represent the *incremental negative and positive, resp. changes* of the occurrence of a step Y, and are given by the linear extension of: $E^-((t,c)) = \sum_{p\in P}\{p\} \times E((p,t))(c)$ and $E^+((t,c)) = \sum_{p\in P}\{p\} \times E((t,p))(c)$, $\forall t \in T,\ \forall c \in C(t)$.

Definition 4 ([11]). *A* colored Petri net *is a tuple* $N = (P, T, A, C, E,\ \Sigma,\ \mathbb{M},\ \mathbb{Y},\ M_0)$ *where:*

- P *is the finite set of places;*
- T *is the finite set of transitions, such that* $P \cap T = \varnothing$*;*
- $A \subseteq P \times T \cup T \times P$ *is the finite set of arcs;*
- Σ *is a universe of non-empty color sets with an associated partial order;*
- $C : P \cup T \to \Sigma$ *is the color set function, assigning color sets to places and (modes) of transitions;*
- $E : A \to \Phi\Sigma$ *is the arc expression function, where* $E(p,t), E(t,p) : C(t) \to \mu C(p)$*;*
- $\mathbb{M} = \mu\{(p,c) \mid p \in P, c \in C(p)\}$ *is the set of markings;*
- $\mathbb{Y} = \mu\{(t,c) \mid t \in T, c \in C(t)\}$ *is the set of steps;*
- M_0 *the initial marking, with* $M_0 \in \mathbb{M}$.

Arc expressions may contain variables, which are seen as symbols whose value is determined by the color (mode) of the transition the arc is connected with.

For any colored Petri net with finite color sets there exists a standard Petri net that is behaviorally equivalent, see [10]. The process of transforming a colored Petri net into its standard Petri net equivalent is called *unfolding*. We give in the following the definition of the unfolding of a colored Petri net as adapted from [10] to the notations we use.

Definition 5 ([10]). *Given a colored Petri net* $N = (P, T, A, \Sigma, C, E, \mathbb{M}, \mathbb{Y}, M_0)$*, its unfolded Petri net is denoted by* $N^* = (P^*, T^*, A^*, f^*, M_0^*)$*, where:*

- P^* *is the set of place instances, pairs* (p,c) *with* $p \in P$ *and* $c \in C(p)$*;*
- T^* *is the set of transition instances, pairs* (t,c) *with* $t \in T$ *and* $c \in C(t)$*;*
- $A^* = \{((p,c),(t,c')) \in P^* \times T^* \mid E((p,t))(c')(c) > 0\} \cup \{((t,c'),(p,c)) \in T^* \times P^* \mid E((t,p))(c')(c) > 0\}$*;*
- $f^*((p,c),(t,c')) = E((p,t))(c')(c),\ \forall((p,c),(t,c')) \in A^*$ *and* $f^*((t,c'),(p,c)) = E((t,p))(c')(c),\ \forall((t,c'),(p,c)) \in A^*$*;*
- $M_0^*((p,c)) = M_0(p,c)$.

3.2 Coloring a Standard Petri Net

A colored Petri net representation of a model can be obtained from a standard Petri net implementation of the model by assigning to each place a color set with just one element. We propose here a general coloring scheme that uses record color sets (i.e. a data structure containing a finite collection of fields, each with a name and an associated data type) and can easily be extended to incorporate refinement details by adding new fields. Each place is assigned its own record color set with one field that has exactly one value. Each transition is assigned a color set that is a multiset of color sets of its pre- and post-places, where the multiplicity of each color set is given by the multiplicity of the arc connecting the place and the transition. It is basically a multiset with elements of different types. For example, the color set *CS_T_fw* in Figure 2 is a collection of two elements of type *CS_P* and one element of type *CS_P2*. Note that this is not the only possible coloring scheme and moreover it may not be optimal (in terms of number of variables and data structures used), but it is general. One may use integers, records, sets, Cartesian products, or whatever coloring scheme better suits the system being modeled.

A further change that is required when turning a standard Petri net into a colored one is assigning to each arc a with arc function $f(a) = k$ where $k \in \mathbb{N}$ the expression $E(a) = \mathsf{v_1} + + \ldots + +\mathsf{v_k}$ where ++ denotes multiset addition and $\mathsf{v_i}$:$C(p)$ are typed variables with $i = 1..k$, and p is the place of arc a. Intuitively, we use a different variable for each token that may traverse an arc. The total number of variables needed in a model is thus $\sum_{a\in A} f(a)$. A further change is in the initial marking, where each place p is assigned the same number of tokens as in the standard network, and all tokens have as color the one color in p's color set. We call such a colored Petri net the *trivial coloring* of the initial network.

We denote by $\mathcal{C}(x)$ the one color in the color set of a place/transtition x. In order to identify precisely the variables used in the expression of an arc $(x, y) \in A$ we denote the variables by $\mathsf{v_{x,y,i}}$, where $i = 1..f((x, y))$. We also use the shorthand notation $\mathsf{v_{a,i}}$ to denote the i-th variable on arc $a \in A$.

Definition 6 (Trivial coloring of a Petri net). *Given a standard Petri net* $N = (P, T, A, f, M_0)$*, we call a* trivial coloring of N *a colored Petri net* $T(N) = (P, T, A, \Sigma, C, E, \mathbb{M}, \mathbb{Y}, M_0')$ *such that:*

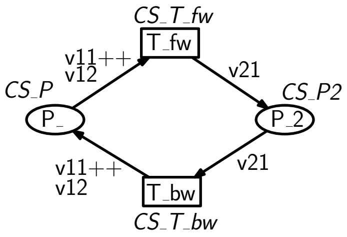
- $\Sigma = \bigcup_{p\in P} C_p \cup \bigcup_{t\in T} C_t$ *where* $C_t : \{C_p \mid p \in P\} \to \mathbb{N}$ *is a multiset such that:*

$$C_t(C_p) = \begin{cases} 0 & (p,t) \notin A \textit{ and } (t,p) \notin A \\ f((p,t)) & (p,t) \in A \textit{ and } (t,p) \notin A \\ f((t,p)) & (p,t) \notin A \textit{ and } (t,p) \in A \\ f((p,t)) + f((t,p)) & \textit{otherwise} \end{cases};$$

- $C : P \cup T \to \Sigma$*, such that* $C(x)$ *is a record color set defined as above if* $x \in P$ *and a multiset defined as above if* $x \in T$*;*

- $E(a) = {}^{++}\sum_{1\leq i\leq f(a)} \mathsf{v_{a,i}} = \mathsf{v_{a,1}} ++ \cdots ++ \mathsf{v_{a,f(a)}}$, *for all* $a \in A$, *where* $\mathsf{v_{a,i}} :$ $C(p)$ *with* p *being the place of arc* a*;*
- $\mathbb{M}$ *is the set of markings;*
- $\mathbb{Y}$ *is the set of steps;*
- $M'_0(p) = M_0(p)\,`\mathcal{C}(p)$, *for all* $p \in P$.

Example 4. An example of a trivial coloring of the Petri net described in Example 3 is given in Figure 2.



colset *CS_P* = record id:int with 0..0;
colset *CS_P2* = record id:int with 0..0;
colset *CS_T_fw* = multiset with *CS_P*, *CS_P*, *CS_P2*;
colset *CS_T_bw* = multiset with *CS_P*, *CS_P*, *CS_P2*;

Fig. 2. Trivial coloring of a Petri net structural implementation of a dimerization model

Definition 7 (Implementation of a reaction-based model as a colored Petri net). *We say that a colored Petri net* N structurally implements *a given reaction-based model* M *iff* N^**, the unfolding of* N*, structurally implements model* M *in the sense of Definition 3.*

Proposition 1. *The unfolding* $T(N)^*$ *of a trivial coloring* $T(N)$ *of a standard Petri net* N *is equivalent to the initial net* N *(as every color set has exactly one color).*

Proposition 2. *If a standard Petri net* N *structurally implements a reaction-based model* M*, then its trivial coloring* $T(N)$ *structurally implements the same model* M*.*

Proof. By Proposition 1, N and $T(N)^*$ are equivalent, thus the unfolding of $T(N)$ structurally implements model M and, by Definition 7, $T(N)$ structurally implements M.

3.3 Type Refinement of Colored Petri Nets

Refinements of Petri nets have been a subject of interest for many years. In particular, we are concerned here with the work of Charles Lakos, who has identified and formalized three types of refinements: *type refinement*, *subnet refinement* and *node refinement*, see [11] for details. The concepts of type and node refinement have been further extended by Choppy et. al., see [3]. We prove in this paper that a full structural refinement of a model can be implemented via a type refinement of the colored Petri net representing the model.

We recall now the definition of type refinement of a colored Petri net as it was proposed in [11].

Definition 8 ([11]). *Let N and N' be two colored Petri nets. A morphism $\Phi : N \to N'$ captures a* type refinement of a colored Petri net *if:*

1. *Φ is the identity function on P, T, A;*
2. *$C(x) <: \Phi(C)(x)$, for all $x \in P \cup T$;*
3. *$\Phi(1\ `(x, c)) = 1\,`(x, \Pi_{\Phi(C)(x)}(c))$ for all $x \in P \cup T$ and for all $c \in C(x)$;*
4. *$\Phi(E^-(1\,`(t, m)))(p) = \Pi_{\Phi(C)(p)}(E(p, t)(m)) = \Phi(E)(p, t)(\Pi_{\Phi(C)(t)}(m))$, for all $(p, t) \in A$ and for all $(t, m) \in \mathbb{Y}$;*
5. *$\Phi(E^+(1\,`(t, m)))(p) = \Pi_{\Phi(C)(p)}(E(t, p)(m)) = \Phi(E)(t, p)(\Pi_{\Phi(C)(t)}(m))$, for all $(t, p) \in A$ and for all $(t, m) \in \mathbb{Y}$.*

A morphism that captures a type refinement is a *system morphism*, see [11], which means that it is a *behavior-respecting* mapping of two colored Petri nets. Expressing structural refinement as a type morphism will thus guarantee that the behavior of the initial network is preserved in the refined network. Moreover, as discussed in [12], type refinement ensures bisimilarity between the initial and the refined network.

Note that for every refined state or action there exists a corresponding abstract state or action, resp. via the projection from subtype to supertype. Also note that in Definition 8, N denotes the refined network.

4 Full Structural Refinement as Type Refinement of Colored Petri Nets

In this section we prove that the full structural refinement of a reaction-based model implemented as a Petri net can be implemented as a type refinement of the trivial coloring of the Petri net. We give a coloring strategy (type refinement) for implementing a full structural data refinement of a model represented as a Petri net, and conclude by proving that our construction indeed implements the required full structural data refinement.

4.1 Implementing a Full Structural Model Refinement via a Type Refinement in a Colored Petri Net Model

Intuitively, species refinement implies replacing each species with a non-empty set of species. This can be done in a colored Petri net by replacing for each place representing a species its default color set by a new record or enumeration color set having as many elements as the set of species that its corresponding species refines to. Or, assuming color sets defined as records, by replacing a single value field with a new field with as many possible values as the cardinality of the refined subspecies set. Formally, we need to define a morphism from the refined colored Petri net to the initial colored Petri net that respects all the properties of a type refinement, as described in [11] and presented in Section 3.3.

Definition 9 (Colored Petri net implementation of a structural refinement of a reaction network model). *We say that a colored Petri net N structurally implements the full structural refinement of a model M as described by a refinement relation ρ iff the unfolding of N, N^*, structurally implements the full structural refinement of M, $\rho(M)$ in the sense of Definition 3.*

We describe next a type refinement of a given trivial coloring of a Petri net implementation of a reaction-based model M that captures the full structural data refinement of M as described by a given refinement relation ρ.

Algorithm 1 TypeRef

```
function TypeRef($N, \rho$)
    $\varSigma' \leftarrow \varnothing$;
                                        ▷ create the new color sets based on the old ones;
    for all $p \in P$ do
        $cs \leftarrow C(p)$;
        define a new color set $cs'$ that extends $cs$ with a new field with $\rho(\delta^{-1}(p))$ values;
        $\varSigma' \leftarrow \varSigma' \cup \{cs'\}$;
        $C'(p) \leftarrow cs'$;
    end for
    for all $t \in T$ do
        define $cs$ as a multiset $cs : \{C'(p) \mid p \in P\} \rightarrow \mathbb{N}$ such that $cs(C'(p)) = C(t)(C(p)),\ \forall p \in P$;
        $\varSigma' \leftarrow \varSigma' \cup \{cs\}$;
        $C'(t) \leftarrow cs$;
    end for
    ▷ re-type the arc expressions: for each variable in an arc expression, create one having as type the new color set of the place that the arc is connected to; the new arc expression is a multiset sum of these variables;
    $E' \leftarrow \varnothing$;
    for all $e \in E$ do
        $p \leftarrow$ the place connected to $e$;
        $V \leftarrow$ set of variables appearing in $e$;
        $V' \leftarrow \varnothing$;
        for all $v_i \in V$ do
            define $v_i' : C'(p)$;
            $V' \leftarrow V' \cup \{v_i'\}$
        end for
        $e' \leftarrow {}^{++}\sum_{v \in V'} v$;                ▷ ${}^{++}\sum$ denotes multiset addition;
        $E' \leftarrow E' \cup \{e'\}$;
    end for
    $\mathbb{M}' \leftarrow \mu\{(p, c) \mid p \in P, c \in C'(p)\}$;
    $\mathbb{Y}' \leftarrow \mu\{(t, c) \mid t \in T, c \in C'(t)\}$;
    $\mathbb{M}'_0$ is designed such that $\sum_{c \in C'(p)} \mid \mathbb{M}'_0(p, c) \mid = \mid \mathbb{M}_0(p, \mathcal{C}(p)) \mid, \forall p \in P$;
    $N' \leftarrow (P, T, A, \varSigma', C', E', \mathbb{M}', \mathbb{Y}', M_0')$;
return $N'$;
end function
```

Let $N = (P, T, A, \Sigma, C, E, \mathbb{M}, \mathbb{Y}, M_0)$ be a trivially colored Petri net that implements a reaction-based model $M = (\mathscr{S}, \mathscr{R})$ with correspondence function δ. Let $\rho \subseteq \mathscr{S} \times \mathscr{S}'$ be a full structural refinement relation that refines model M to model $M' = (\mathscr{S}', \mathscr{R}')$. We build a colored Petri net $N' = (P, T, A, \Sigma', C', E', \mathbb{M}', \mathbb{Y}', M_0')$ and then show that the construction is a type refinement. Moreover, we show that the resulting network implements the full structural refinement $\rho(M)$. The procedure takes as input a trivially colored Petri net that implements M, and the refinement function ρ. It then updates the color sets of the network such that the color set of each place is extended with a new field that will account for the new subtypes of the species that the place stands for. Each transition gets as color set a multiset of the color sets of its pre- and post-places, with multiplicities dictated by the cardinality of each arc expression, just like in the trivial coloring. Note that this means that the refined transition color sets are subtypes of the initial transition color sets, as multisets of subtypes of a color set that is a multiset of supertypes, with identical multiplicities.

Using a distinct variable for each token on every arc is important because it allows for exact identification of each token. One can thus encode all possible combinations of in- and out- tokens for a transition t, i.e. the full set of refinements of the reaction encoded by transition t.

Proposition 3. *Given a trivially colored Petri net N that is an implementation of a reaction-based model M, and a full structural refinement relation ρ of M, the colored Petri net $N' = \textsc{TypeRef}(N, \rho)$ is a type refinement of the initial network.*

Proof. Based on the construction described in Algorithm 1, we detail here the type refinement morphism between the two networks.

Note that N is trivially colored, so all color sets have exactly one color. The projection from any color in a color set of Σ' onto its corresponding supertype color set is the one color in the supertype color set: $\Pi_{C(x)}(c) = \mathcal{C}(x)$, for any $x \in P \cup T$, and any color $c \in C'(x)$.

We now describe a morphism $\Phi_\rho : N' \to N$ between the two networks, that is a type morphism.

1. $\Phi_\rho(x) = x$ for all $x \in P \cup T \cup A$.
2. $\Phi_\rho(C')(x) = C(x)$. By definition of the color sets in N', the color set of each place and of each transition in N' is a subtype of the color set of the same place/transition in N, i.e. $C'(x) <: \Phi_\rho(C')(x)$. Moreover, for any color $c \in C'(x) : \Pi_{\Phi_\rho(C')(x)}(c) = \Pi_{C(x)}(c) = \mathcal{C}(x)$.
3. $\forall x \in P \cup T : \forall c \in C'(x) : \Phi_\rho(1\,`(x, c)) = 1\,`(x, \Pi_{C(x)}(c)) = 1\,`(x, \mathcal{C}(x))$: for every colored place/transition in N' with color c, the morphism Φ_ρ returns the same place/transition (because Φ_ρ is the identity on $P \cup T$), having as color the projection of c on the color set of x as given by the morphism Φ_ρ, namely $\mathcal{C}(x)$.
4. $\forall (p, t) \in A : \forall (t, m) \in \mathbb{Y}' : \Phi_\rho(E'(p, t)) = E(p, t)$ and the multiset of colored tokens consumed from place p at the firing of transition t in mode m is $E'(p, t)(m)$. By construction of E', the number of consumed tokens is $E(p, t)(\mathcal{C}(t))$. The projection of every color in $C'(p)$ is $\mathcal{C}(p)$, thus we get:

$$\Phi_\rho(E^-(1`(t,m))(p)) = \Pi_{\Phi_\rho(C')(p)}(E'(p,t)(m)) = E(p,t)(\mathcal{C}(t)) = \\ = E(p,t)(\Pi_{C(t)}(m)) = \Phi_\rho(E')(p,t)(\Pi_{\Phi_\rho(C')(t)}(m)).$$

5. Similarly, $\forall(t,p) \in A : \forall(t,m) \in \mathbb{Y}' : \Phi_\rho(E'(t,p)) = E(t,p)$ and the multiset of colored tokens added to place p at the firing of transition t in mode m is $E'(t,p)(m)$. By construction of E', the number of produced tokens is $E(t,p)(\mathcal{C}(t))$. The projection of every color in $C'(p)$ is $\mathcal{C}(p)$, thus we get:

$$\Phi_\rho(E^+(1`(t,m))(p)) = \Pi_{\Phi_\rho(C')(p)}(E'(t,p)(m)) = E(t,p)(\mathcal{C}(t)) = \\ = E(t,p)(\Pi_{C(t)}(m)) = \Phi_\rho(E')(t,p)(\Pi_{\Phi_\rho(C')(t)}(m)).$$

Because the morphism Φ_ρ respects all conditions for being a type refinement of a Petri net it follows that Algorithm 1 computes a type refinement of its input Petri net.

Theorem 1. *Given a reaction-based model $M = (\mathscr{S}, \mathscr{R})$, a structural refinement relation $\rho \subseteq \mathscr{S} \times \mathscr{S}'$, and a colored Petri net $N = (P, T, A, \Sigma, C, E, \mathbb{M}, \mathbb{Y}, M_0)$ that is trivially colored and implements model M with function $\delta : \mathscr{S} \cup \mathscr{R} \to P \cup T$, the colored Petri net* TypeRef(N, ρ) *implements the full structural ρ-refinement of model M.*

Proof. Let N' denote the refined colored Petri net TypeRef(N, ρ), and let $M' = (\mathscr{S}', \mathscr{R}')$ denote the full structural ρ-refinement M_ρ. By construction of the refined colored Petri net N' there exists a type morphism between N' and N, as detailed in the proof of Proposition 3.

First, note that N is trivially colored and thus the network is equivalent to its unfolding (see Proposition 1). With a slight abuse of notation, we will use x to denote the unfolded equivalent of a place/transition $x \in P \cup T$, $(x, \mathcal{C}(x))$.

We show now that the unfolding of N' implements the full structural refinement of M. Let $N^* = \{P^*, T^*, A^*, f^*, M_0^*\}$ be the unfolding of N'. The color set of a place $p \in P'$ has $\mid \rho(\delta^{-1}(p)) \mid$ elements, where each color represents one refined species $S' \in \mathscr{S}'$, $(\delta^{-1}(p), S') \in \rho$. The places of N^* represent pairs (p, c) such that $p \in P$ and $c \in C'(p)$. Given that every place p has a symbolic correspondence with one species $S = \delta^{-1}(p)$ in $\mathscr{S}$, and the colors of places in N' can be thought of as the refinements of S, there exists a one-to-one correspondence between places in P^* and species in $\mathscr{S}'$. Let $\delta_\rho : \mathscr{S}' \to P^*$, with $\delta_\rho(S') = (\delta(S), c) \in P^*$ where $(S, S') \in \rho$ and no two siblings are mapped to the same value.

δ_ρ can be extended to map also reactions in $\mathscr{R}'$ to (t, m) pairs. The color m of a transition t uniquely identifies its pre- and post-places in the unfolded network, and the arc inscriptions. By definition of the color sets of transitions as multisets over the color sets of neighbouring places, it follows that every possible combination of colored tokens flowing through a transition is captured by a transition color. This means that a transition t in N' encodes all possible refinements $\rho(r)$ of the reaction $r = \delta^{-1}(t)$ that transition t stands for in N.

A transition $(t, m) \in T^*$ encodes the reaction

$$\sum_{(p,c)\in {}^\bullet(t,m)} f^*((p,c),(t,m))\delta_\rho^{-1}((p,c)) \to \sum_{(p,c)\in(t,m)^\bullet} f^*((t,m),(p,c))\delta_\rho^{-1}((p,c)).$$

The reaction $r' = \delta_\rho^{-1}(t,m)$ that a transition $(t,m) \in T^*$ implements in N'^* is a ρ-refinement of the reaction $r = \delta^{-1}(t)$ that transition t implements in N. This comes from the type refinement conditions 4 and 5 (see Definition 8). The incremental effects of executing a step (t,m) in the refined network equal the incremental effects of executing the step $(t, \Pi_{C(t)}(m))$ in the initial network. The negative incremental effect E^- encodes the left hand side of a reaction, and the positive incremental effect E^+ encodes the right hand side.

We detail here the negative incremental effect of a step, and relate it to its meaning in the model M'. $E^-(1`(t,m)) = \sum_{(p,t)\in A} p \times E((p,t))(m)$. In the unfolded network N^* a transition (t,m) is connected to places via edges $((p,c),(t,m)) \in A^*$ where $f^*((p,c),(t,m)) = E((p,t))(m)(c)$. Summing over all unfolded instances of a place in N^* yields

$$\sum_{c\in C'(p)} f^*((p,c),(t,m)) = \sum_{c\in C'(p)} E((p,t))(m)(c) =\mid E((p,t))(m) \mid .$$

Note that the arc expressions in N and N' are the same, which means that their cardinality is also the same. N implements model M, thus $|E((p,t))| = c_{i,j}$ and $|E((t,p))| = c'_{i,j}$ where $c_{i,j}$ is the stoichiometric coefficient of species $S_i = \delta^{-1}(p)$ on the left hand side of reaction $r_j = \delta^{-1}(t)$ and $c'_{i,j}$ is the soichiometric coefficient of S_i on the right hand side of r_j. Arc multiplicities in N^* represent stoichiometries, and for any place p of N' its unfolded places $\{(p,c) \mid c \in C'(p)\}$ represent the sibling species in $\rho(\delta^{-1}(p))$.

A similar argument can be made for the right hand side of a reaction, starting from the positive incremental effect of a step. With both the left and the right hand side of a reaction represented by (t,m) being a ρ-refinement of the left or right, respectively hand side of the reaction $\delta^{-1}(t)$, it follows that (t,m) implements a ρ-refinement of the reaction implemented by t.

5 Discussion

In this paper we have made a connection between the notions of type refinement of a colored Petri net proposed in [11] and that of full structural refinement of reaction network models proposed in [6]. The connection is based on modeling a reaction network system as a Petri net and using a coloring scheme that allows for easy type refinement. Starting from a Petri net implementation of a reaction-based model, we proposed a general coloring scheme that uses record color sets and further detailed the construction and how the color sets can be refined. We proved that the colored Petri net obtained by coloring the initial Petri net with our coloring strategy is also an implementation of the model implemented by the initial net. We further proved that our strategy is in fact using a type refinement that implements a full structural refinement of a model.

The size of the refined colored Petri net model We discuss here about the size of the colored Petri net model obtained by refining a given model, in terms of number of places and transitions.

A type refinement of a colored Petri net preserves the structure of the network unchanged, i.e. the number of places and transitions does not change. But the semantics of each place and transition is different, and we will therefore consider the unfolding of the colored Petri net.

Given $N = (P, T, A, \Sigma, C, E, \mathbb{M}, \mathbb{Y}, M_0)$ a trivial colored Petri net implementation of a reaction-based model $M = \mathscr{S}, \mathscr{R}$, a refinement relation $\rho \subseteq \mathscr{S} \times \mathscr{S}'$ and a colored Petri net $N' = (P, T, A, \Sigma', C', E, \mathbb{M}', \mathbb{Y}', M_0')$ which is the implementation of the full structural $\rho-$refinement of M by Algorithm 1 with function $\delta : \mathscr{S} \cup \mathscr{R} \to P \cup T$, we discuss the size of the unfolding of N', denoted byN^*.

N has by construction $|\mathscr{S}|$ places and $|\mathscr{R}|$ transitions. In N' by construction each place representing a species $S \in \mathscr{S}$ has $\rho(S)$ colors, and will therefore unfold to $\rho(S)$ places. The total number of unfolded places is $\sum_{S\in\mathscr{S}} |\rho(S)| = |\mathscr{S}'|$. The total number of possible colors of a transition depends on the number of colors in the color set of the pre- and post-places of the transition, and on the cardinality of the arc expressions of arcs connected on either end with the transition. A transition $t \in T$ will thus unfold to

$$\prod_{p\in {}^\bullet t} \left(\!\!\binom{|\rho(\delta^{-1}(p))|}{E((p,t))}\!\!\right) \cdot \prod_{p\in t^\bullet} \left(\!\!\binom{|\rho(\delta^{-1}(p))|}{E,((t,p))}\!\!\right)$$

transitions in N^*, which yields a total number of transitions in N^* equal to

$$\sum_{t\in T} \left(\prod_{p\in {}^\bullet t} \left(\!\!\binom{|\rho(\delta^{-1}(p))|}{E((p,t))}\!\!\right) \cdot \prod_{p\in t^\bullet} \left(\!\!\binom{|\rho(\delta^{-1}(p))|}{E,((t,p))}\!\!\right)\right).$$

Depending on the refinement function ρ, this number can be much larger than the number of transitions in the colored network N', which successfully avoids this explosion in number of places and transitions of the network.

Consecutive full structural refinements Very often models go through several steps of refinement, as new information about the modeled system is available, and a more detailed representation is needed. We discuss in this paragraph how subsequent full structural refinements of a model can be implemented using our approach. The problem can be formulated as follows. Given a reaction-based model $M = (\mathscr{S}, \mathscr{R})$ and two refinement relations $\rho \subseteq \mathscr{S} \times \mathscr{S}'$ and $\rho' \subseteq \mathscr{S}' \times \mathscr{S}''$, obtain the full structural $\rho'-$refinement of the full structural $\rho-$refinement of M. In our construction, we start from a trivial coloring of a Petri net implementation of a model. This is however not a limitation of the approach, since subsequent refinements can be implemented as one single refinement that is the composition of the two (or more) successive refinements to be implemented.

We conclude that colored Petri nets can be used to implement full structural refinements of reaction-based models. The major advantage of using the colored Petri nets formalism lies in their ability to represent the fully structurally refined system in a compact way, using the same network structure and adding all refinement details in the colors of places and transitions.

References

1. Ralph-Johan Back and Joakim Wright. *Refinement calculus: a systematic introduction*. springer Heidelberg, 1998.
2. Claudine Chaouiya. Petri net modelling of biological networks. *Briefings in bioinformatics*, 8(4):210–219, 2007.
3. Christine Choppy, Laure Petrucci, and Alfred Sanogo. Coloured petri nets refinements. In *PNSE+ ModPE*, pages 187–201. Citeseer, 2013.
4. Vincent Danos, Jérôme Feret, Walter Fontana, Russell Harmer, and Jean Krivine. Rule-based modelling and model perturbation. *Transactions on Computational Systems Biology XI*, pages 116–137, 2009.
5. René David and Hassane Alla. Petri nets for modeling of dynamic systems: A survey. *Automatica*, 30(2):175–202, 1994.
6. Cristian Gratie and Ion Petre. Fit-preserving data refinement of mass-action reaction networks. In Arnold Beckmann, Erzsébet Csuhaj-Varjú, and Klaus Meer, editors, *Language, Life, Limits*, volume 8493 of *Lecture Notes in Computer Science*, pages 204–213. Springer, 2014.
7. Bogdan Iancu, Elena Czeizler, Eugen Czeizler, and Ion Petre. Quantitative refinement of reaction models. *International Journal of Unconventional Computing*, 8(5-6):529–550, 2012.
8. Bogdan Iancu, Diana-Elena Gratie, Sepinoud Azimi, and Ion Petre. On the implementation of quantitative model refinement. In Adrian-Horia Dediu, Carlos Martín-Vide, and Bianca Truthe, editors, *Algorithms for Computational Biology*, volume 8542 of *Lecture Notes in Computer Science*, pages 95–106. Springer International Publishing, 2014.
9. Kurt Jensen. Coloured petri nets: A high level language for system design and analysis. *Advances in Petri nets 1990*, pages 342–416, 1991.
10. Kurt Jensen. Coloured petri nets. volume 1, basic concepts. *EATCS Monographs in Theoretical Computer Science*, 1, 1992.
11. Charles Lakos. Composing abstractions of coloured petri nets. In *Application and Theory of Petri Nets 2000*, pages 323–342. Springer, 2000.
12. Charles Lakos and Glenn Lewis. A catalogue of incremental changes for coloured petri nets. Technical report, the International Conference of Application and Theory of Petri Nets, 1999.
13. Andrzej Mizera, Eugen Czeizler, and Ion Petre. Self-assembly models of variable resolution. *LNBI Transactions on Computational Systems Biology*, pages 181–203, 2011.
14. Tadao Murata. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580, 1989.
15. Elaine Murphy, Vincent Danos, Jérôme Feret, Jean Krivine, and Russell Harmer. *Elements of Computational Systems Biology*, chapter Rule Based Modelling and Model Refinement, pages 83–114. Wiley Book Series on Bioinformatics. John Wiley & Sons, Inc., 2010.
16. Marco Pistore and Davide Sangiorgi. A partition refinement algorithm for the pi-calculus. In *Proceedings of CAV'96, volume 1102 of Lecture Notes in Computer Science*, pages 200–1. Springer-Verlag, 1996.
17. Ichiro Suzuki and Tadao Murata. A method for stepwise refinement and abstraction of petri nets. *Journal of Computer and System Sciences*, 27(1):51 – 76, 1983.

Dependent shrink for Petri net models of signaling pathways

Atsushi Mizuta[1], Qi-Wei Ge[2], and Hiroshi Matsuno[1]

[1] Graduate School of Science and Engineering, Yamaguchi University
Yamaguchi 753-8512, Japan
[2] Faculty of Education, Yamaguchi University
Yamaguchi 753-8512, Japan
{v014vc,gqw}@yamaguchi-u.ac.jp, matsuno@sci.yamaguchi-u.ac.jp

Abstract. Retention-free Petri net has been used in modeling of signaling pathways, which is a timed Petri net such that total input and total output token flows are equivalent at any place. Previously we have investigated the dependency of transitions in retention-free Petri net. In this paper, we introduce a modeling method for signaling pathway by using Petri net, giving properties of retention-freeness by considering arc weight. Based on the obtained properties, we propose an algorithm to find shrinkable transitions and to shrink them into a single transition. This algorithm eventually provides a set of transitions whose firing frequencies are dependent. As an example, we apply the algorithm to IL-3 signaling pathway Petri net model to show the usefulness of our proposed algorithm.

Keywords: signaling pathway, Petri net, retention-free Petri net, dependent shrink

1 Introduction

Li et al. [1] have proposed a qualitative modeling method by paying attention to the molecular interactions and mechanisms using discrete Petri nets. Furthermore, Miwa et al. [2] modeled it with timed Petri net, which is an extended Petri net on the concept of time, proposing a method to have firing frequency conditional expressions based on its structure information. At the same time, they introduced "retention-free" Petri net for defining smooth signal flows in signaling pathways.

In Petri net model of signaling pathway, firing frequency of each transition should be measured by biological experiments. However, such biological data of reactions are very few. As a method to cope with this problem, Murakami et al. [3] proposed an approach to check the retention-freeness of a given Petri net based on firing frequencies of transitions of this Petri net. According to this method, Matsumoto et al. [4] formally described the concept of dependent shrink after giving formal definitions of dependent subnet. Dependent shrink is a concept to express a dependent subnet which is shrunk into a single transition.

M Heiner, AK Wagler (Eds.): BioPPN 2015, a satellite event of PETRI NETS 2015,
CEUR Workshop Proceedings Vol. 1373, 2015.

Place
Transition
Directed arc

Fig. 1. Elements of Petri nets

The advantage of this concept is that all the firing frequencies of transitions in the subnet can be computationally obtained from the firing frequency of that shrunk transition. Namely, by only getting the reaction speed of a reaction that corresponds to that shrunk transition, all other reaction speeds in dependent subnet can be estimated by the proposed procedure in this paper.

In this paper we propose an algorithm to do equivalent transformation for retention-free Petri nets. Concretely, we first classify dependent shrink pattern according to the patterns of input and output transitions of a place and then perform the dependent shrink operations of the patterns. Finally, we reconstruct the Petri net based on the dependent shrink result.

2 Basic Definitions and Properties

In this section, we briefly give the necessary definitions and properties of Petri nets. For detailed definitions the reader is suggested to refer to [5].

Definition 1. *A Petri net denoted as $PN = (T, P, E, \alpha, \beta)$ that is a bipartite graph, where $E = E^+ \cup E^-$ and*

- *T: a set of transitions $\{t_1, t_2, \cdots, t_{|T|}\}$*
- *P: a set of places $\{p_1, p_2, \cdots, p_{|P|}\}$*
- *E^+: a set of arcs from transitions to places $e = (t, p)$*
- *E^-: a set of arcs from places to transitions $e = (p, t)$*
- *α_e: is the weight of arc $e = (p, t)$*
- *β_e: is the weight of arc $e = (t, p)$* □

Definition 2. *Let PN be a Petri net*

1. *${}^\bullet t$ ($t^\bullet$) is the set of input (or output) places of t, and ${}^\bullet p$ ($p^\bullet$) is the set of input (or output) transitions of p.*
2. *A transition without input arc is called source transition and the set of source transitions are denoted by $T_{sour} = \{t_1^{sour}, \cdots, t_a^{sour}\}(a \geq 1)$.*
3. *A transition without output arc is called sink transition and the set of sink transitions is denoted by $T_{sink} = \{t_1^{sink}, \cdots, t_b^{sink}\}(b \geq 1)$.*
4. *A transition t is called $P_s - synchronous$ transition if there exists a set of input places P_s that for any $p \in P_s$, $p^\bullet = \{t\}$ holds, and is defined by $T_{sync} = \{t_1^{sync}, \cdots, t_c^{sync}\}(c \geq 1)$.*

5. *A place can hold a positive integer that represents a number of tokens. An assignment of tokens in places expressed in form of a vector M is called a marking, which varies during the execution of a Petri net. Given with an initial marking M_0, the Petri net is called Marked Petri net and denoted by $MPN = (PN, M_o)$.* □

Fig. 2 shows source(i) and sink(ii) transition and Fig. 3 shows synchronous transitions. Note that, we use discrete Petri nets in this paper.

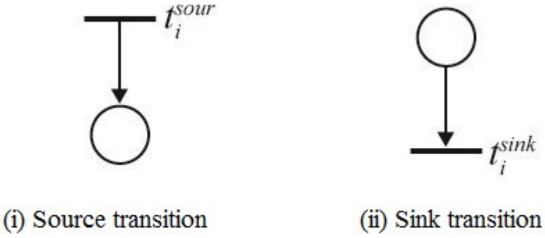


Fig. 2. Source and sink transition

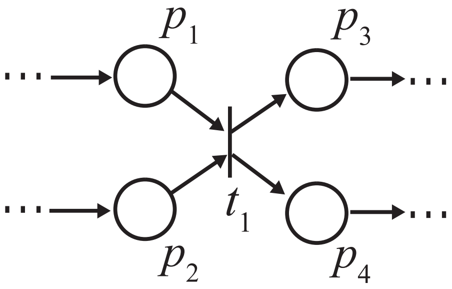


Fig. 3. Synchronous transition

2.1 Modeling rules

Li et al. [1] gave the following modeling rules for signaling pathways based on Petri net representation.

1. Places denote static elements including chemical compounds, conditions, states, substances, and cellular organelles participating in the biological pathways. Tokens indicate the presence of these elements. The number of

tokens can be regarded as a representation of the amount of chemical substances. Current assignment of tokens to the places is expressed in form of a vector, namely a *marking* as defined above.

2. Transitions denote active elements including chemical reactions, events, actions, conversions, and catalyzed reactions. A transition fires by taking off tokens from its individual input places and creating new tokens that are distributed to its output places if its input places has at least as many tokens in it as arc weight from the place to the transition.
3. Directed arcs connecting the places and the transitions represent the relations between corresponding static elements and active elements. Arc weights α and β (defined in Definition 1) describe the quantities of substances required before and after a reaction, respectively. Especially in case of modeling a chemical reaction, arc weights represent quantities given by stoichiometric equations of the reaction itself. Note that, weight of an arc is omitted if the weight is 1.
4. Since an enzyme itself plays a role of catalyzer in biological pathways and there occurs no consumption in biochemical reactions, an enzyme is exceptionally modeled in Definition 3 below.
5. An inhibition function in biological pathways is modeled by an inhibitor arc.

Definition 3. [1] *An enzyme in a biological pathway is modeled by a place, called enzyme place, as shown in Fig. 4.*

1. *Enzyme place p_i has a self-loop with the same weight connected from and to transition t_s. Once an enzyme place is occupied by a token, the token will return to the place again to keep the firable state, if the transition t_s is fired.*
2. *Let t_p and t_d denote a token provider of p_i and a sink output transition of p_i, respectively, where the firing of t_p represents an enzyme activation reaction and the firing of t_d implies a small natural degradation in a biological pathway. p_i holds up token(s) after firing transition t_p and the weights of the arcs satisfy $\alpha(p_i, t_d) \ll \alpha(p_i, t_s)$.* □

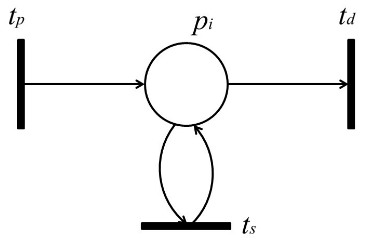


Fig. 4. An enzyme place in Petri net model

[Firing rule of Petri net] A transition t is firable if each input place p_I of PN has at least α_e tokens, where α_e denotes the weight of an arc $e = (p_I, t)$. Firing of a transition t removes α_e tokens from each input place p_I of t and deposit $\beta_e(e = (t, p_O))$ tokens to each output place p_O of t, where β_e denotes the weight of an arc $e = (t, p_O)$. A source transition is always firable.

Definition 4. *A timed Petri net TPN is defined by $TPN = (PN, D)$, where D is a set of positive number expressing firing delay times (or delay time for short) of transitions in T.* □

[Firing rule of timed Petri nets] (i) If the firing of a transition t_i is decided, tokens required for the firing are reserved. We call these tokens as reserved tokens. (ii) When the delay time d_i of t_i passed, t_i fires to remove the reserved tokens from the input places of t_i and put non-reserved tokens into the output places of t_i. In a timed Petri net, firing times of a transition t_i per unit time is called *firing frequency* f_i. $\overline{f_i}$ represents the maximum firing frequency of t_i. The delay time d_i of t_i is given by the reciprocal of $\overline{f_i}$.

Definition 5. [2] *With the firing of transition t_I, token amounts flowed into place p per unit time is called "input token-flow", and is denoted by $TF_{t_I,p}$. On the other hand, with the firing of transition t_O, token amounts flowed out of place p per unit time is called "output token-flow", and is denoted by TF_{p,t_O}. $TF_{t_I,p}$ and TF_{p,t_O} (shown in Fig. 5) are defined by following equations, respectively:*

$$TF_{t_I,p} = f_I \cdot \beta_I \tag{1}$$

$$TF_{p,t_O} = f_O \cdot \alpha_O, \tag{2}$$

where f_I and f_O are firing frequencies of t_I and t_O, respectively; β_I and α_O are the weights of $e = (t_I, p)$ and $e = (p, t_O)$, respectively. □

Based on this definition, the following equation hold.

Proposition 1. [2] *Let p be a place with input transitions $\{t_{I_i} | t_{I_i} \in {}^\bullet p\}$ and out put transitions $\{t_{O_j} | t_{O_j} \in p^\bullet\}$. Then $\sum_{i=1}^{m} TF_{t_{I_i},p}$ and $\sum_{j=1}^{n} TF_{p,t_{O_j}}$ are the total input token-flow and the total output token-flow for place p, respectively. Furthermore, when firing frequency f take the maximum firing rate $\overline{f}$, input token-flow $TF_{t_I,p}$ and output token flow TF_{p,t_O} become the maximum, $\overline{FT_{t_I,p}}$ and $\overline{FT_{p,t_O}}$, respectively. These maximum token-flows satisfy following equations.*

$$\sum_{i=1}^{m} TF_{t_{I_i},p} \leq \sum_{i=1}^{m} \overline{FT_{t_{I_i},p}} \tag{3}$$

$$\sum_{j=1}^{n} TF_{p,t_{O_j}} \leq \sum_{j=1}^{n} \overline{FT_{p,t_{O_j}}} \tag{4}$$

□

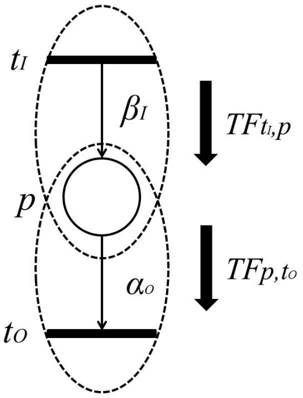


Fig. 5. Token flows

The following requirement is trivial.

Proposition 2. [2] *In a timed Petri net, a total output token-flow is not more than a total input token-flow for each place p:*

$$\sum_{i=1}^{m} TF_{t_{I_i},p} \geq \sum_{j=1}^{n} TF_{p,t_{O_j}}, \tag{5}$$

□

Definition 6. [2] *A timed Petri net TPN is called Retention-free Petri net (RFPN) (satisfying Proposition1) if a total input token-flow and a total output token-flow are equivalent at any place of TPN; that is,*

$$TF_{t_{I_i},p} = TF_{p,t_{O_j}} \tag{6}$$

□

Definition 7. [3] *Each unreserved token deposited to input place p is assigned to be reserved by the transition t_{O_j} that satisfies the following expression:*

$$\begin{aligned}&\{(\frac{c_j}{\alpha_j})/\sum_{k=1}^{n}(\frac{c_k}{\alpha_k}) - s_j\} = \\ &min\{(\frac{c_i}{\alpha_i})/\sum_{k=1}^{n}(\frac{c_k}{\alpha_k}) - s_i \mid i = 1, 2, \cdots, n\}\end{aligned} \tag{7}$$

When the number of reserved tokens of t_{O_j} is not less than a required token number for the firing, the firing of t_{O_j} is decided. After the delay time d_{O_j} of t_{O_j} passed, t_{O_j} fires to remove the reserved tokens from the input place of t_{O_j} and deposit unreserved tokens into the output places of t_{O_j}. □

In the above expression (7), s_j is the firing probability of transition t_{O_j}, which represents the proportion of the firing frequency of each transition to the total firing frequencies of the transitions in conflict. A probability s_j is assigned to corresponding transition t_{O_j}, which is given as a constant in advance according to the event. A variable c is an accumulated number of tokens that t_{O_j} has been reserved so far, and thus $\lfloor \frac{c_j}{\alpha_j} \rfloor$ represents the number of firing times of transition t_{O_j} from the beginning.

Expression (7) is designed to reserve the token to such a transition t_i that has the largest difference between calculated firing probability $\frac{c_j}{\alpha_j} / \sum_{k=1}^{n} \frac{c_k}{\alpha_k}$ and given firing probability s_j among all the transitions in conflict.

Definition 8. [2] *If output transitions of p are in conflict, the maximum firing frequency of t_{O_j} must satisfy the following expression:*

$$\frac{s_j \cdot \alpha_j}{\sum_{k=1}^{n} s_k \cdot \alpha_k} \cdot \sum_{i=1}^{m} TF_{t_{I_i},p} = f_{O_j} \cdot \alpha_j, \tag{8}$$

where α_j is the weight of $e = (p, t_{O_j})$ and s_j is the firing probability of t_{O_j}. $\frac{s_j \cdot \alpha_j}{\sum_{k=1}^{n} s_k \cdot \alpha_k}$ represents the ratio of the token amount deposited to t_{O_j} to the total token-flow from p to each output transition $p^\bullet$. □

3 Shrink of Dependent Subnet

The equation (8) shows a relationship of firing frequency about input and output transitions, which are dependent each other. Based on this dependency, a set of transitions can be obtained, by which firing frequencies of all transitions in a Petri net model can be calculated. Note that the transitions in the set determined in this way correspond to the reactions whose speeds need to be measured by biological experiments.

3.1 Dependent subnet

Dependent subnet, obtained as follows, is a Petri net induced from a set of transitions which are dependent on each other.

Definition 9. [4] *If firing frequency of a transition t is determined by the firing frequency of transition α, this transition is called α-dependent transition. The subnet induced by the set of α-dependent transitions and transition α is called α-dependent subnet, denoted by PN_α.* □

Definition 10. [4] *For a given set A of transitions, A-dependent transition is a set of transitions whose firing frequencies are determined by the firing frequencies of transitions in A. The subnet induced by A-dependent transition and A is called A-dependent subnet, denoted by PN_A.* □

3.2 Dependent shrink

For a set of α-dependent transition T, dependent shrink is a procedure to substitute the set of α-dependent transition T to a single transition t.

Definition 11. [4] *If two transitions t_i and t_j exist are dependent each other, these two transitions can be shrunk into a single transition.* □

Note that the proofs of the following propositions are omitted to save the space of this paper.

Proposition 3. *As shown in Fig. 6, if a place p has one input transition t_I and one output transition t_O, these two transitions can be shrunk into a single transition t', where the weight of new input arc $\alpha' = \alpha_1$, and the weight of new output arc $\beta' = \beta_2 \cdot \frac{\beta_1}{\alpha_2}$.* □

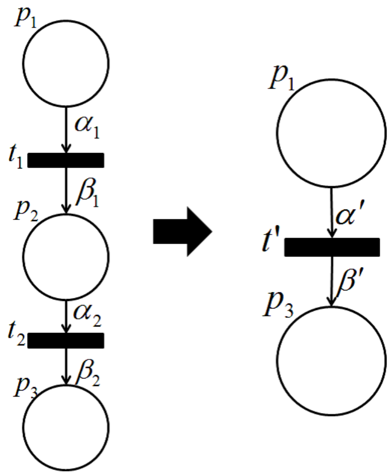


Fig. 6. Markgraph

Proposition 4. *As shown in Fig. 7, if place p has multiple output transitions $T_O = \{t_{O.1}, t_{O.2}, \cdots, t_{O.k}\}$, T_O can be shrink into a single transition t', where* *the weights of input arc α' and new output arc β' are defined by the following formulas;*

$$\alpha' = \frac{s_{O.1} \cdot \alpha_1 + s_{O.2} \cdot \alpha_2 + \cdots + s_{O.k} \cdot \alpha_k}{s_{O.1}} \tag{9}$$

$$\begin{aligned} \beta'_1 &= s_{O.1} \cdot \beta_1 \\ \beta'_2 &= s_{O.2} \cdot \beta_2 \\ &\vdots \\ \beta'_k &= s_{O.k} \cdot \beta_k \end{aligned} \tag{10}$$

□

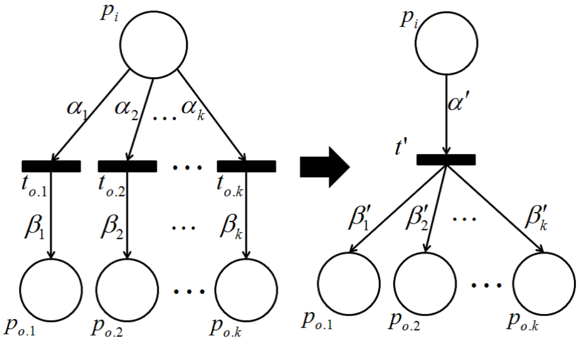


Fig. 7. Conflict structure

Proposition 5. *As shown is Fig. 8, if place p has multiple output transitions of two types, with self-loop $T_l = \{t_{l.1}, t_{l.2}, \cdots, t_{l.k}\}$ and without self-loop $T_O = \{t_{O.1}, t_{O.2}, \cdots, t_{O.k}\}$,$T_l$ and T_O can be shrunk into a single transition, where input arc α' and new output arc β' are defined by the following formulas;*

$$\begin{aligned} \alpha' = \Big(&(s_{O.1} \cdot \alpha_{O.1} + s_{O.2} \cdot \alpha_{O.2} + \cdots + s_{l.k2} \cdot \alpha_{l.k2}) \\ &-(s_{l.1} \cdot \beta_{l.1} + s_{l.2} \cdot \beta_{l.2} + \cdots + s_{l.k2} \cdot \beta_{l.k2})\Big)/s_{O.1} \end{aligned} \tag{11}$$

$$\begin{aligned} \beta'_1 &= s_{O.1} \cdot \beta_1 \\ \beta'_2 &= s_{O.2} \cdot \beta_2 \\ &\vdots \\ \beta'_{k2} &= s_{l.k2} \cdot \beta_{k2} \end{aligned} \tag{12}$$

Note that, in equation (11) $\alpha' > 0$ *should be held.* □

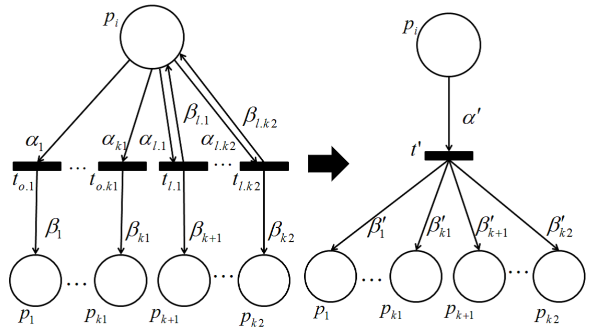


Fig. 8. Self-loop structure

4 Dependent Shrink Algorithm and an Example

In this section, we propose a dependent shrink algorithm based on the dependent shrink method. Furthermore, we apply this algorithm to IL-3 signaling pathway Petri net model (shown in Fig. 10), which is transformed from IL-3 phenomenon model (shown in Fig. 9) obtained from the website [10]. Note that IL-3 is a glycoprotein and is known to be involved in the immune response [6–9].

4.1 Outline of shrink process

The shrink process of dependent subnet can be briefly described as follows:

step1: *Shrink of self loop structure*
A place randomly selected from a Petri net is stored in a queue after the conversion of the self-loops and the structures of conflict of it.

step2: *Shrink of conflict structure*
If a place picked up from the queue has a self-loop or a transition of one-input and one-output, this place is shrunk.

step3: *Changing weight of the input arc*
If shrunk Petri net has a multiple input place, it re-stores to the queue, performing the above step2 again. This procedure is repeated until the queue becomes empty.

The variables used in the algorithm are as follows:

- PN_0 is a given signaling pathway Petri net model constituted by $T_0, P_0, and\ E_0$.
- N is a variable that stores Petri net after dependent shrink, constituted by $T, P, andE$.
- Q is a queue.
- X is a set of place initially set as a given place set P_0.
- f is a flag, by which dependent shrink pattern is determined.

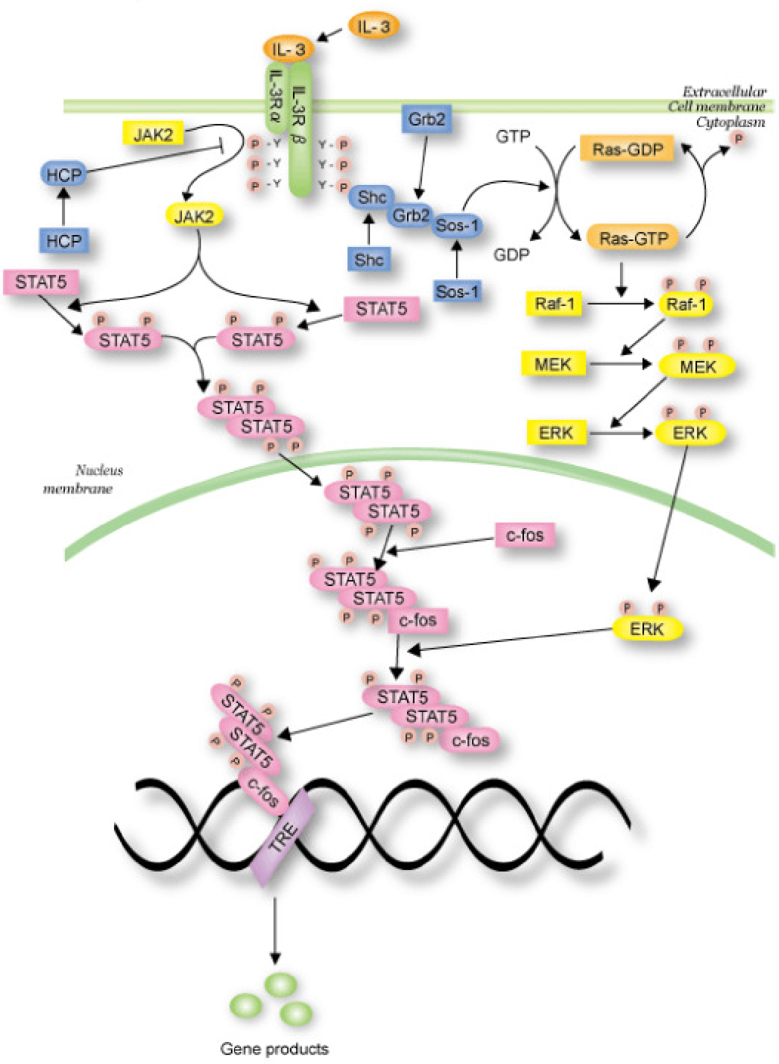


Fig. 9. IL-3 phenomenon model

4.2 Dependent shrink algorithm

The following algorithm is used to shrink dependent subnets into a single transition in order to find transitions with interdependent firing frequency.

Algorithm: Dependent shrink
Input: $PN_0 = (T_0, P_0, E_0)$
Output: Shrunk Petri net $N = (T, P, E)$

```
**Main**($PN_0$)
1°$T \leftarrow T_0$, $P \leftarrow P_0$, $E \leftarrow E_0$, $N \leftarrow (T, P, E)$
2°$X \leftarrow P$, $Q \leftarrow \phi$
3°**while** $(X \neq \phi)$
    Pull an element $x$ from $X(X \leftarrow X - \{x\})$
    **Enqueue**$(Q, x)$
    **Shrink1**$(N, x)$
4°**Shrink2**$(N, Q)$
**Shrink1**$(N, x)$
1°**if** $(|^\bullet x \cap x^\bullet| \geq 1)$ **then**
        $f \leftarrow 1$
        **Arcweight**$(N, x, f)$
2°**if** $(|x^\bullet| \geq 2)$ **then**
    $f \leftarrow 2$
    **Arcweight**$(N, x, f)$
**Shrink2**$(N, Q)$
1°**while** $(|Q| \geq 1)$
    $x \leftarrow$ **Dequeue**$(Q)$
        **if** $(|^\bullet x \cap x^\bullet| \geq 1)$ **then**
            $f \leftarrow 1$
            **Enqueue**$(Q, x)$
        **else if** $(|^\bullet x| = |x^\bullet| = 1)$ **then**
            $f \leftarrow 3$
        **else if** $(|^\bullet x| \geq 2)$ **then**
            $f \leftarrow 4$
            **Enqueue**$(Q,\ x)$
        **if** $(f \neq 4)$ **then**
            **Arcweight**$(N, x, f)$
**Arcweight**$(N, x, f)$
1°**if** $(f = 1)$ **then**
    $\forall t' \in {}^\bullet x \cap x^\bullet$
        $\alpha(x, t') = \alpha(x, t') - \beta(t', x)$
        **if** $(\alpha(x, t') < 0)$ **then**
            $\beta(t', x) = |\alpha(x, t')|$
            $E \leftarrow E - \{(x, t')\}$
        **else if** $(\alpha(x, t') > 0)$ **then**
            $E \leftarrow E - \{(t', x)\}$
        **else if** $(\alpha(x, t') = 0)$ **then**
            $E \leftarrow E - \{(t', x), (x, t')\}$
2°**else if** $(f = 2)$ **then**
    $T \leftarrow T \cup \{t'\}$
    $E \leftarrow E \cup \{(x, t')\} \cup \{(u, t')|u \in {}^\bullet z, z \in x^\bullet\} \cup \{(t', v)|v \in z^\bullet, z \in x^\bullet\}$
    Choose $z' \in x^\bullet$.
    $\forall z \in x^\bullet - \{t'\}$
        $\alpha(x, t') = \alpha(x, t') + s(z) * \alpha(x, z)$
        $\forall v \in z^{\bullet}, z \in x^{\bullet}$
            $\beta(t', v) = s(z) * \beta(z, v)$
        $\forall u \in {}^{\bullet}z, z \in x^{\bullet}$
            $\alpha(u, t') = s(z) * \alpha(u, z)/s(z')$
    $\alpha(x, t') = \alpha(x, t')/s(z')$
    $T \leftarrow T - \{z | z \in x^{\bullet} - \{t'\}\}$
3°else if $(f = 3)$ then
    $T \leftarrow T \cup \{t'\}$
    Let $z_i, z_o$ be $\{z_i\} = {}^{\bullet}x$, $\{z_o\} = x^{\bullet}$ (due to $|{}^{\bullet}x| = |x^{\bullet}| = 1$).
    $E \leftarrow E \cup \{(u, t') | u \in {}^{\bullet}z_i \cup {}^{\bullet}z_o\} \cup \{(t', v) | v \in z_i^{\bullet} \cup z_o^{\bullet}\}$
        $\forall u \in {}^{\bullet}z_i$
            $\alpha(u, t') = \alpha(u, z_i)$
        $\forall u \in {z_i}^{\bullet}$
            $\beta(t', u) = \beta(z_i, u)$
        $\forall v \in {}^{\bullet}z_o$
            $\alpha(v, t') = \beta(z_i, x) * \alpha(v, z_o)/\alpha(x, z_o)$
        $\forall v \in {z_o}^{\bullet}$
            $\beta(t', v) = \beta(z_i, x) * \beta(z_o, v)/\alpha(x, z_o)$
    $T \leftarrow T - \{z_i | z_i \in {}^{\bullet}x\} - \{z_o | z_o \in x^{\bullet}\}$
    $P \leftarrow X - \{x\}$
```

When the above algorithm is applied, a dependent subnet, say S, is transformed into a single transition, say t_S. Obviously S and t_S possess the same input and output places. As the result, for each input place, p, the tokens flowed out of p per unit time are the same before and after dependent shrink. Similarly, the tokens flowed into an output place per unit time are the same.

4.3 An Example for Signaling Pathway Petri Net Model

Here we give an example to show an application of our proposed algorithm. The algorithm is applied to dependent shrink for IL-3 Petri net model (see Fig.10 (a)), obtained from the website [10]. As the result, the original IL-3 Petri net model shown in Fig.10 (a) is shrunk into Fig.10 (c). This means that the firing frequency of all transition in Fig.10 (a) are denpendent each other. In the intermediate shrunk net (see Fig. 10 (b)), by assuming the firing frequency of the input transition f_I be 1, the weights of input and output arcs be $\frac{1}{2}$ and 1, respectively, then the firing frequency of output transition f_O is $\frac{1}{2}$. In this way, we can calculate all of the firing frequency of transitions in the IL-3 Petri net model from one firing frequency in this model.

5 Conclusion

In this paper, after giving basic definitions of Petri net and modeling method, we introduced dependent shrink method and its properties to find dependent subnet. Further, we designed an algorithm of dependent shrink and applied it to IL-3 signaling pathway Petri net model as an example.

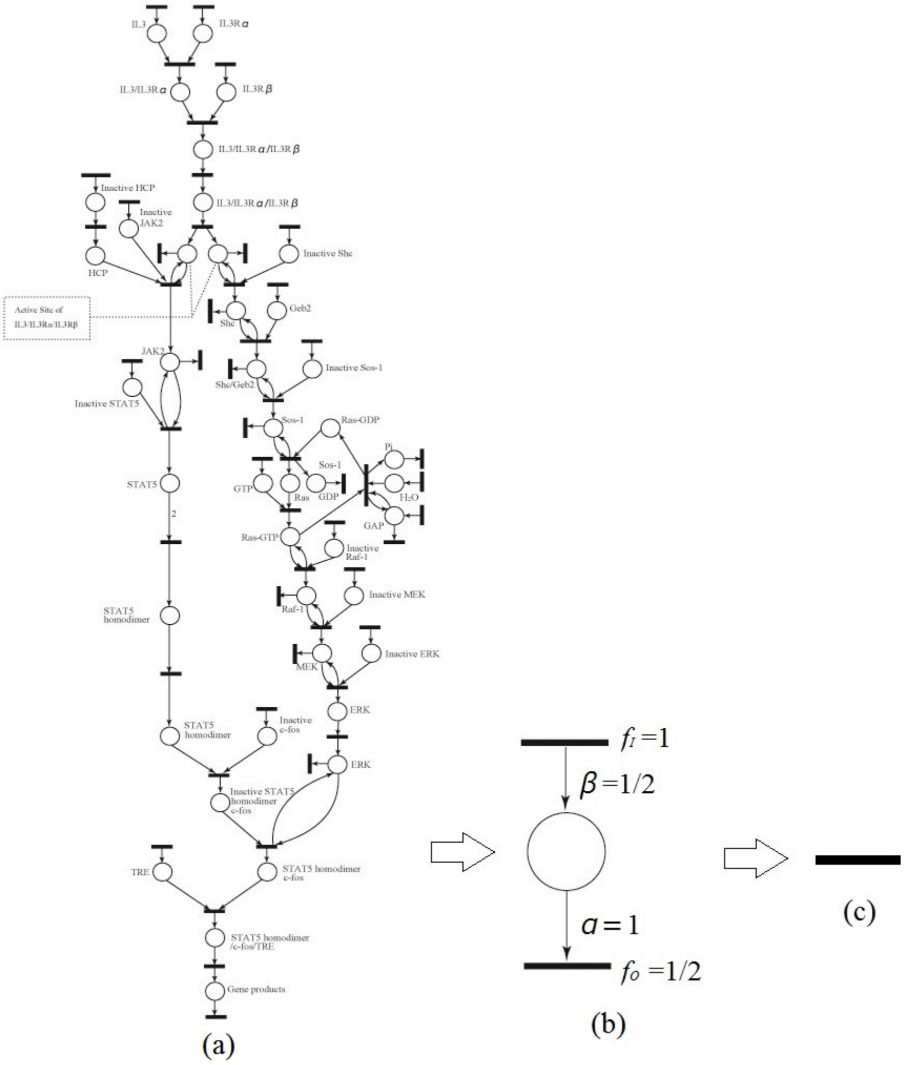


Fig. 10. Shrunk IL-3 Petri net model

By applying the dependent shrink algorithm, IL-3 Petri net model is converted to a simple model, with which we could find the transitions which are dependent each other.

This algorithm allows us to obtain firing frequencies of all transitions in a dependent subnet only by measuring reactions corresponding to the transitions by biological experiments in the simple model.

In this paper, we only discussed discrete Petri nets. By extending transitions to include firing speed, it is possible to extend our method to continuous Petri nets.

As a future work, we need to improve our algorithm so that it can indicate transitions corresponding to measurable reactions by biological experiment. Also the uniqueness of our algorithm needs to be investigated.

Acknowledgement

We would like to thank Dr. Adrien Fauré at Yamaguchi University for his support in proofreading the manuscript. This work was partially supported by Grant-in-Aid for Scientific Research (B) (23300110) from Japan Society for the Promotion of Science.

References

1. C. Li, S. Suzuki, Q.W. Ge, M. Nakata, H. Matsuno, S. Miyano, "Structural modeling and analysis of signaling pathways based on Petri net", *Journal of Bioinformatics and Computational Biology*, Vol.4, No.5, pp.1119-1140, 2006.
2. Y. Miwa, Y. Murakami, Q.W. Ge, C. Li, H. Matsuno, S. Miyano, "Delay time determination for the timed Petri net model of a signaling pathway based on its structural information", *IEICE on Fundamentals of Electronics, Communications and Computer Sciences*, Vol.E93-A, No12, pp2717-2729, 2010 (in Japanese).
3. Y. Murkami, Q.W. Ge, H. Matsuno, "Consideration on the token retention-free in timed Petri net model based on the signaling pathway characteristics", Technical Report of IEICE, Vol.111, No.453, MSS2011-77, pp.29-34, 2012 (in Japanese).
4. T. Matsumoto, Q.W. Ge, H. Matsuno, "An algorithm for finding dependent subnets in a retention-free Petri net," Technical Report of IEICE, MSS2012-50, pp.27-32, 2013 (in Japanese).
5. J.L. Peterson, *Petri net theory and the modeling of systems*, Prentice Hall, 1981.
6. K. Sugamura, K. Miyazono, K. Miyazawa, N. Tanaka, *Saitokain zoushokuinshi yougo raiburari* (*Term Library of Cytokines and Growth factor*), Yodosha, 2005 (in Japanese).
7. W. Chen, M.O. Daines, G.K. Khurana Hershey, "Turning off signal transducer and activator of transcription (STAT): the negative regulation of STAT signaling," The Journal of allergy and clinical immunology, 114(3), pp.476-489, quiz 490, 2004.
8. T. Kisseleva, S. Bhattacharya, J. Braunstein, C. W. Schindler, "Signaling through the JAK/STAT pathway," recent advances and future challenges, Gene, 285(1-2), pp.1-24, 2002.
9. R.P. de Groot, P.J. Coffer, L. Koenderman, "Regulation of proliferation, differentiation and survival by the IL-3/IL-5/GM-CSF receptor family," Cellular signalling, 10(9), pp.619-628, 1998.
10. Petri Net Pathways, `http://genome.ib.sci.yamaguchi-u.ac.jp/pnp/`(accessed April 20th)

Author Index