Monika Heiner
Ralf Hofestädt

Biological Processes & Petri Nets

3rd International Workshop BioPPN 2012
Hamburg, 25 June 2012

Proceedings

CEUR Workshop Proceedings, Volume 852

Editors & Publishers:

Monika Heiner
Brandenburgische Technische Universität Cottbus, Institut für Informatik,
03013 Cottbus, Germany
monika.heiner@informatik.tu-cottbus.de

Ralf Hofestädt
Bielefeld University, Faculty of Technology, Bioinformatics Department, Germany
ralf.hofestaedt@uni-bielefeld.de

CEUR Workshop Proceedings (ISSN 1613-0073), Volume 852
http://CEUR-WS.org/Vol-852/

also available as Preprint FBI-HH-M-347/12
published by
Fachbereich Informatik, Universität Hamburg
Vogt-Kölln-Straße 30
22527 Hamburg, Germany

BibTeX entry for online proceedings:

```
@proceedings{bioppn2012,
  editor    = {Monika Heiner and Ralf Hofest\"adt},
  title     = {Proceedings of the 3rd International Workshop on
               Biological Processes \& Petri Nets (BioPPN 2012),
               satellite event of Petri Nets 2012, Hamburg, Germany, June 25, 2012},
  booktitle = {Biological Processes \& Petri Nets},
  location  = {Hamburg, Germany},
  publisher = {CEUR-WS.org},
  series    = {CEUR Workshop Proceedings},
  volume    = {Vol-852},
  year      = {2012},
  url       = {http://CEUR-WS.org/Vol-852/}
}
```

Preface

These proceedings contain the five peer-reviewed contributions as well as the (extended) abstracts of the three complementary accepted presentations given at the Third International Workshop on *Biological Processes & Petri Nets* (BioPPN 2012), held as a satellite event of PETRI NETS 2012, in Hamburg, Germany, at June 25, 2012.

The workshop has been organised to provide a platform for researchers aiming at fundamental research and real life applications of Petri nets 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.

We received three types of contributions: research papers, work-in-progress papers and posters. All submissions have been reviewed by four to five reviewers coming from or being recommended by the workshop's Program Committee. The list of reviewers comprises 18 professionals of the field. The five accepted peer-reviewed papers (with an acceptance rate of 78%) involve 23 authors coming from three different countries. In summary, the workshop proceedings enclose theoretical contributions as well as biological applications, demonstrating the interdisciplinary nature of the topic.

The workshop programme was complemented by the invited talk *Petri Nets - an Integrative Framework for Advanced Biomodel Engineering* given by Wolfgang Marwan from the Magdeburg Centre for Systems Biology (MaCS), Otto-von-Guericke University Magdeburg, Germany.

For more details see the workshop's website http://www-dssz.informatik.tu-cottbus.de/BME/BioPPN2012.

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

June 2012

Monika Heiner
Ralf Hofestädt

Table of Contents

Program Committee

David Angeli	Imperial College London, UK
Gianfranco Balbo	University of Torino, Italy
Rainer Breitling	University of Glasgow, UK
Jorge Carneiro	Instituto Gulbenkian de Ciência, Portugal
Claudine Chaouiya	Instituto Gulbenkian de Ciência, Portugal
Ming Chen	Zhejiang University, China
David Gilbert	Brunel University, UK
Simon Hardy	Université Laval, Canada
Monika Heiner	Brandenburg University of Technology, Germany
Ralf Hofestädt	Bielefeld University, Germany
Peter Kemper	College of William and Mary, USA
Hanna Klaudel	University of Evry, France
Hiroshi Matsuno	Yamaguchi University, Japan
Satoru Miyano	University of Tokyo, Japan
Ps Thiagarajan	National University of Singapore, Singapore

Additional Reviewers

B

Beccuti, Marco
C

Cordero, Francesca
H

Herajy, Mostafa
R

Rohr, Christian
S

Schwarick, Martin
Sugii, Manabu

Proc. BioPPN 2012, a satellite event of PETRI NET 2012

A Database-supported Modular Modelling Platform for Systems and Synthetic Biology

Wolfgang Marwan

Lehrstuhl für Regulationsbiologie and Magdeburg Centre for Systems Biology,
Otto-von-Guericke Universität, Universitätsplatz 2, 39106 Magdeburg, Germany
`wolfgang.marwan@ovgu.de`

Abstract. Petri nets are employed as a multifunctional integrative framework for biomodel engineering. We describe the general concept of a modular modelling approach that consideres the functional coupling of components of genome, transcriptome, and proteome with complex cellular phenotypes. For this purpose, the effects of genes and their mutated alleles on downstream components are modeled by composable, metadata-containing Petri net models organized in a database with version control, accessible through a web interface. Gene modules are coupled to protein modules through mRNA modules by specific interfaces designed for the automatic, database-assisted composition. Automatically assembled executable models may then consider cell type-specific gene expression patterns and take the resulting protein concentrations into account. With a sufficient number of protein modules in the database, the composed Petri nets can predict complex effects of gene mutations or uncover complex genotype/phenotype relationships. In this context, forward and reverse engineered modules are fully compatible.

Comparison of Metabolic Pathways by Considering Potential Fluxes

Paolo Baldan[1], Nicoletta Cocco[2] and Marta Simeoni[2]

[1] Dipartimento di Matematica, Università di Padova
via Trieste 63, 35121 Padova, Italy
email: baldan@math.unipd.it
[2]Dipartimento di Scienze Ambientali, Statistica e Informatica,
Università Ca' Foscari Venezia,
via Torino 155, 30172 Venezia Mestre, Italy
email: cocco@dsi.unive.it,simeoni@dsi.unive.it

Abstract. Comparison of metabolic pathways is useful in phylogenetic analysis and for understanding metabolic functions when studying diseases and in drugs engineering. In the literature many techniques have been proposed to compare metabolic pathways, but most of them focus on structural aspects, while behavioural or functional aspects are generally not considered. In this paper we propose a new method for comparing metabolic pathways of different organisms based on a similarity measure which considers both homology of reactions and functional aspects of the pathways. The latter are captured by relying on a Petri net representation of the pathways and comparing the corresponding T-invariant bases, which represent potential fluxes in the nets. The approach is implemented in a prototype tool, CoMeta, which allows us to test and validate our proposal. Some experiments with CoMeta are presented.

1 Introduction

The life of an organism depends on its metabolism, the chemical system which generates the essential components - amino acids, sugars, lipids and nucleic acids - and the energy necessary to synthesise and use them. Subsystems of metabolism dealing with some specific function are called metabolic pathways. Comparing metabolic pathways of different species yields interesting information on their evolution and it may help in understanding metabolic functions. This is important for metabolic engineering and for studying diseases and drugs design.

In the recent literature many techniques have been proposed for comparing metabolic pathways of different organisms. Each approach chooses a representation of metabolic pathways which models the information of interest, proposes a similarity or a distance measure and possibly supplies a tool for performing the comparison.

Representations of metabolic pathways at different degrees of abstraction have been considered. A pathway can be simply viewed as a *set* of components of interest, which can be reactions, enzymes or chemical compounds. In other approaches pathways are decomposed into a set of paths, leading from an initial metabolite to a final one. The most detailed representations model a metabolic pathway as a graph. Clearly, more detailed models produce more accurate comparison results, in general at the price of being more complex.

The distances in the literature generally focus on static, topological information of the pathways, disregarding the fact that they represent dynamic processes. In this paper we propose to take into account also behavioural aspects: we represent the pathways as Petri nets (PNs) and compare also aspects related to their behaviour as captured by T-invariants. Petri nets seem to be particularly natural for representing and modelling metabolic pathways (see, e.g., [8] and references therein). The graphical representations used by biologists for metabolic pathways and the ones used in PNs are similar; the stoichiometric matrix of a metabolic pathway is analogous to the incidence matrix of a PN; the flux modes and the conservation relations for metabolites correspond to specific properties of PNs. In particular minimal (semi-positive) T-invariants correspond to elementary flux modes [43] of a metabolic pathway, i.e., minimal sets of reactions that can operate at a steady state. The space of semi-positive T-invariants has a unique basis of minimal T-invariants which is characteristic of the net and we use it in the comparison. Hence we propose a similarity measure between pathways which considers both homology of reactions, represented by the Sørensen index on the multisets of enzymes in the pathways, and similarity of potential fluxes in the pathways, obtained by comparing the corresponding T-invariant bases. We developed a prototype tool, CoMeta, implementing our proposal. Given a set of organisms and a set of metabolic pathways, CoMeta automatically gets the corresponding data from the KEGG database, builds the corresponding Petri nets, computes the T-invariants and the similarity measures and shows the results of the comparison among organisms as a phylogenetic tree. We performed several experiments with CoMeta and, although further investigations are definitively needed, the approach appears to be promising and worth to be pursued.

The paper is organised as follows. In Section 2 we introduce metabolic pathways and give a classification of various proposals for metabolic pathways comparison. In Section 3 we show how a Petri net can model a metabolic pathway and present our proposal. In Section 4 we briefly illustrate the tool CoMeta and we present some experiments. A short conclusion follows in Section 5.

2 Comparison of Metabolic Pathways

In this section we briefly introduce metabolic pathways and classify various proposals for the comparison of metabolic pathways in the literature.

2.1 Metabolic pathways

Biologists usually represent a metabolic pathway as a network of *chemical reactions*, catalysed by one or more *enzymes*, where some molecules (*reactants* or *substrates*) are transformed into others (*products*). Enzymes are not consumed in a reaction, even if they are necessary and used while the reaction takes place. The product of a reaction is the substrate for other ones.

To characterise a metabolic pathway, it is necessary to identify its components (namely the reactions, enzymes, reactants and products) and their relations. Quantitative relations can be represented through a *stoichiometric matrix*, where rows represent molecular species and columns represent reactions. An element of the matrix, a *stoichiometric coefficient* n_{ij}, represents the degree to which the i-th chemical species participates in the j-th reaction. By convention, the coefficients for reactants are negative, while those for products are positive. The kinetic of a pathway is determined by the rate associated to each reaction. It is represented by a *rate equation*, which depends on the concentrations of the reactants and on a *reaction rate coefficient* (or *rate constant*) which includes all the other parameters (except for concentrations) affecting the rate.

Information on metabolic pathways are collected in databases. In particular the *KEGG PATHWAY* database [2] (KEGG stands for *Kyoto Encyclopedia of Genes and Genomes*) contains the main known metabolic, regulatory and genetic pathways for different species. It integrates genomic, chemical and systemic functional information [23]. The pathways are manually drawn, curated and continuously updated from published materials. They are represented as maps which are linked to additional information on reactions, enzymes and genes, which may be stored in other databases. KEGG can be queried through *KGML* (KEGG Markup Language) [1], a language based on XML.

2.2 Comparison techniques for metabolic pathways

Many proposals exist in the literature for comparing metabolic pathways and whole metabolic networks in different organisms. Each proposal is based on some simplified representation of a metabolic pathway and on a related definition of similarity score (or distance measure) between two pathways. Hence we can group the various approaches in three classes, according to the structures they use for representing and comparing metabolic pathways. Such structures are:

- *Sets.* Most of the proposals in the literature represent a metabolic pathway (or the entire metabolic network) as the set of its main components, which can be reactions, enzymes or chemical compounds (see, e.g., [17, 18, 29, 22, 14, 13, 10, 48, 33]). This representation is simple and efficient and very useful when entire metabolic networks are compared. The comparison is based on suitable set operations.
- *Sequences.* A metabolic pathway is sometimes represented as a set of sequences of reactions (enzymes, compounds), i.e., pathways are decomposed into a set of selected paths leading from an initial component to a final one (see, e.g., [49, 30, 11, 27, 50]). This representation may provide more information on the original pathways, but it can be computationally more expensive. It requires methods both for identifying a suitable set of paths and for comparing them.

- *Graphs.* In several approaches, a metabolic pathway is represented as a graph (see, e.g., [20, 34, 16, 52, 28, 6, 12, 24, 31, 26, 7, 5]). This is the most informative representation in the classification, as it considers both the chemical components and their relations. A drawback can be the complexity of the comparison techniques. In fact the graph and subgraph isomorphism problems are GI-complete (graph isomorphism complete) and NP-complete, respectively. For this reason efficient heuristics are used and simplifying assumptions are introduced, which produce further approximations.

The similarity measure (or distance) and the comparison technique strictly depend on the chosen representation. When using a set-based representation, the comparison between two pathways roughly consists in determining the number of common elements. A similarity measure commonly used in this case is the *Jacard index* defined as:

$$J(X, Y) = \frac{|X \cap Y|}{|X \cup Y|}$$

where X and Y are the two sets to be compared. When pathways are represented by means of sequences, *alignment* techniques and *sum of scores with gap penalty* may be used as similarity measures. In the case of graph representation, more complex algorithms for *graph homeomorphism* or *graph isomorphism* are used and some approximations are introduced to reduce the computational costs.

In any case the definition of a similarity measure between two metabolic pathways relies on a similarity measure between their components. Reactions are generally identified with the enzymes which catalyse them, and the most used similarity measures between two reactions/enzymes are based on:

- *Identity.* The simplest similarity measure is just a boolean value: two enzymes can either be identical (similarity = 1) or different (similarity = 0).
- *EC hierarchy.* The similarity measure is based on comparing the unique *EC number* (Enzyme Commission number) associated to each enzyme, which represents its catalytic activity.
  The EC number is a 4-level hierarchical scheme, $d_1.d_2.d_3.d_4$, developed by the International Union of Biochemistry and Molecular Biology (IUBMB) [51]. For instance, *arginase* is numbered by $EC : 3.5.3.1$, which indicates that the enzyme is a hydrolase ($EC : 3. * . * .*$), and acts on the "carbon nitrogen bonds, other than peptide bonds" (sub-class $EC : 3.5. * .*$) in linear amidines (sub-sub-class $EC : 3.5.3.*$). Enzymes with similar EC classifications are functional homologues, but do not necessarily have similar amino acid sequences.
  Given two enzymes $e = d_1.d_2.d_3.d_4$ and $e' = d'_1.d'_2.d'_3.d'_4$, their similarity $S(e, e')$ depends on the length of the common prefix of their EC numbers:

  $$S(e, e') = \max\{i : d_i = d'_i\}/4$$

  For instance, the similarity between *arginase* ($e = 3.5.3.1$) and *creatinase* ($e' = 3.5.3.3$) is 0.75.

- *Information content.* The similarity measure is based on the EC numbers of enzymes together with the information content of the numbering scheme. This is intended to correct the large deviation in the distribution in the enzyme hierarchy. For example, the enzymes in the class 1.1.1 range from $EC1.1.1.1$ to $EC1.1.1.254$, whereas there is a single enzyme in the class 5.3.4. Given an enzyme class h, its information content is defined as $I(h) = -log_2 C(h)$, where $C(h)$ denotes the number of enzymes in h. The similarity between two enzymes e_i and e_j is $I(h_{ij})$, where h_{ij} is their lowest common upper class.
- *Sequence alignment.* The similarity measure is obtained by aligning the genes or the proteins corresponding to the two enzymes and by considering the resulting alignment score.

3 Behavioural Aspects in Metabolic Pathways Comparison

In this section we briefly discuss how to represent a metabolic pathway as a Petri net. Then we define a similarity measure between two metabolic pathways modelled as Petri nets, which takes into account the flows in the pathways by comparing their minimal T-invariants. Such measure is combined with a more standard one which considers homology of reactions.

3.1 Metabolic pathways as Petri nets

PNs are a well known formalism introduced in computer science for modelling discrete concurrent systems. PNs have a sound theory and many applications both in computer science and in real life systems (see [32] and [15] for surveys on PNs and their properties). A large number of tools have been developed for analysing properties of PNs. A quite comprehensive list can be found at the Petri net World site [3].

In some seminal papers Reddy et al. [37, 35, 36] and Hofestädt [21] proposed Petri nets (PNs) for representing and analysing metabolic pathways. Since then, a wide range of literature has grown on the topic [8]. The structural representation of a metabolic pathway by means of a PN can be derived by exploiting the natural correspondence between PNs and biochemical networks. In fact places are associated with molecular species, such as metabolites, proteins or enzymes; transitions correspond to chemical reactions; input places represent the substrate or reactants; output places represent reaction products. The incidence matrix of the PN is identical to the stoichiometric matrix of the system of chemical reactions. The number of tokens in each place indicates the amount of substance associated with that place. Quantitative data can be added to refine the representation of the behaviour of the pathway. In particular, extended PNs may have an associated transition rate which depends on the kinetic law of the corresponding reaction. Large and complex networks can be greatly simplified by avoiding an explicit representation of enzymes and by assuming that ubiquitous substances are in a constant amount. In this way, however, processes involving these substances, such as the energy balance, are not modelled.

Once metabolic pathways are represented as Petri nets, we consider their behavioural aspects as captured by the *T-invariants* (transition invariants) of the nets which, roughly, represents potential cyclic behaviours in the system. More precisely a T-invariant is a (multi)set of transitions whose execution starting from a state will bring the system back to the same state. Alternatively, the components of a T-invariant may be interpreted as the relative firing rates of transitions which occur permanently and concurrently, thus characterising a steady state. Therefore presence of T-invariants in a metabolic pathway is biologically of great interest as it can reveal the presence of steady states, in which concentrations of substances have reached a possibly dynamic equilibrium.

Although space limitations prevent us from a formal presentation of nets and invariants, it is useful to recall that the set of (semi-positive) T-invariants can be characterised finitely, by resorting to its Hilbert basis [40].

Remark 1 (Unique basis). The set of T-invariants of a (finite) Petri net N admits a unique basis which is given by the collection $\mathcal{B}(N)$ of minimal T-invariants.

The above means that any T-invariant can be obtained as a linear combination (with positive integer coefficient) of minimal T-invariants. Uniqueness of the basis $\mathcal{B}(N)$ allows us to take it as a characteristic feature of the net.

The problem of determining the Hilbert basis is EXPSPACE since the size of such basis can be exponential in the size of the net. Still, in our experience, the available tools like INA [47] work fine on Petri nets arising from metabolic pathways.

In a PN model of a metabolic pathway, a minimal T-invariant corresponds to an elementary flux mode, a term introduced in [43] to refer to a minimal set of reactions that can operate at a steady state. It can be interpreted as a minimal self-sufficient subsystem which is associated to a function. Minimal T-invariants are important in model validation techniques (see, e.g., [19, 25]) and they may provide insights into the network behaviour. By assuming both the fluxes and the pool sizes constants, with some further simplifying assumption, the stoichiometry of the network restricts the space of all possible net fluxes to a rather small linear subspace. Such subspace can be analysed in order to capture possible behaviours of the pathway and its functional subunits [38, 39, 41–44].

3.2 A combined similarity measure between pathways

Metabolic pathways are complex networks of biochemical reactions describing fluxes of substances. Such fluxes arise as the composition of elementary fluxes, i.e., cyclic fluxes which cannot be further decomposed. Most of the techniques briefly illustrated in Section 2 compare pathways on the basis of homology of their reactions, that is they determine a point to point functional correspondence. Some proposals consider also the topology of the network, but still most techniques are eminently static and ignore the flow of metabolites in the pathway.

Here we propose a comparison between metabolic pathways based on the combination of two similarity scores derived from their Petri net representation. More precisely, we consider a "static" score, *R_score* (reaction score), taking into account the homology of reactions occurring in the pathways and a "behavioural" score, *I_score* (invariant score), taking into account the dynamics of the pathway as expressed by the T-invariants.

Both *R_score* and *I_score* are based on the *Sørensen index* [46] extended to multisets as below, where X_1 and X_2 are multisets and $\cap$ and $|\cdot|$ are intersection and cardinality generalised to multisets. [1]

$$S\_index(X_1, X_2) = \frac{2|X_1 \cap X_2|}{|X_1| + |X_2|}$$

Given two pathways represented as Petri nets P_1 and P_2, the *R_score* is computed by comparing their reactions. Each reaction is actually represented by the EC numbers of the associated enzymes. More precisely, if X_1 and X_2 denotes the multisets of the EC numbers in P_1 and $P2$ respectively, we define the *R_score* as

$$R\_score(X_1, X_2) = S\_index(X_1, X_2).$$

The similarity considered between enzymes is the identity, but finer similarity measures between enzymes, such as the one determined by the EC hierarchy, could be easily accommodated in this setting. We choose a multiset representation since an EC number may occur more than once in a pathway and we opted for the Sørensen index as it fits better to multisets than the Jacard index.

The distance based on reactions is then defined as follows

$$d_R(P_1, P_2) = 1 - R\_score(X_1, X_2).$$

The behavioural component of the similarity is obtained by comparing the Hilbert bases of minimal T-invariants. Each invariant is represented as a multiset of EC numbers, corresponding to the reactions occurring in the invariant, and the similarity between two invariants is given, as before, by the *S_index*. Note that when T-invariants are sets of transitions (rather than proper multisets) they can be seen as subnets of the net at hand, and the similarity between two T-invariants coincides with the *R_score* of the corresponding subnets. More generally, transitions can occur in an invariant with some multiplicity, which influences the similarity score.

A heuristic match between the two bases $\mathcal{B}(P_1)$ and $\mathcal{B}(P_2)$ is performed and the *S_index* values corresponding to the matching pairs are accumulated into I_SCORE(P_1, P_2) as described by the algorithm in Fig. 1.

---

[1] Formally, a multiset is a pair (X, m_X) where X is the *underlying set* and $m_X : X \to \mathbb{N}^+$ is the *multiplicity function*, associating to each $x \in X$ a positive natural number indicating the number of its occurrences. Then $|(X, m_X)| = \sum_{z \in X} m_X(z)$ and $(X, m_X) \cap (Y, m_Y) = (X \cap Y, m_{X \cap Y})$ where $m_{X \cap Y}(z) = min(m_X(z), m_Y(z))$ for each $z \in X \cap Y$.

```
function I_Score($(P_1, P_2)$);
  input:   two metabolic pathways $P_1$ and $P_2$;
  output:  the similarity measure between $\mathcal{B}(P_1)$ and $\mathcal{B}(P_2)$;
begin
  $I_1 = \mathcal{B}(P_1)$; $I_2 = \mathcal{B}(P_2)$;
  $score = 0$;
  $card = \max\{|I_1|, |I_2|\}$;
  while $(I_1 \neq \emptyset \ \wedge \ I_2 \neq \emptyset)$ do
  begin
    $(X_1, X_2)$ = Find_max_Sim($I_1, I_2$);    {Returns a pair of T-invariants, $(X_1, X_2)$,
                                                  in $I_1 \times I_2$ such that $S\_index(X_1, X_2)$
                                                  is maximum}
    $score = score + S\_index(X_1, X_2)$;
    $I_1 = I_1 - \{X_1\}$;
    $I_2 = I_2 - \{X_2\}$;
  end;
  $score = score/card$;
  return score
end Compute_I_Score;
```

Fig. 1. Comparing bases of T-invariants

Again, pathways similarity based on minimal T-invariants induces a distance:

$$d_I(P_1, P_2) = 1 - I\_score(P_1, P_2)$$

The two distances are combined by taking a weighted sum as below, where $\alpha \in [0, 1]$:

$$d_D(P_1, P_2) = \alpha \ d_R(P_1, P_2) + (1 - \alpha) \ d_I(P_1, P_2)$$

The parameter α allows the analyst to move the focus between homology of reactions and similarity of functional components as represented by the T-invariants.

Two organisms O_1 and O_2 can be compared by considering n metabolic pathways $P_1, \ldots, P_n$. In this case the distances between the two organisms with respect to the various metabolic pathways P_j, $j \in [1, n]$, need to be combined. The simplest solution consists in taking the average distance:

$$d_D(O_1, O_2) = \frac{\sum_{j=1}^{n} d_D(P_j^1, P_j^2)}{n}$$

When a pathway P_j occurs in one of the two organisms but not in the other, the corresponding pathway distance $d_D(P_j^1, P_j^2)$ in the formula above is taken to be 1.

4 Experimenting with CoMeta

In this section we briefly illustrate the tool CoMeta (Comparing METAbolic pathways) which implements our proposal, and we report on some experiments.

CoMeta is a user-friendly tool written in Java and running under Windows and Linux. Due to space limitation, we just list its main integrated functionalities:

- *Select organisms and pathways:* CoMeta proposes the lists of KEGG organisms and pathways and allows the user to select the ones to be compared. Such lists can be saved and then recovered for further processing.
- *Retrieve KEGG information:* the KEGG files corresponding to the selected organisms and pathways are automatically downloaded by CoMeta from the KEGG database [2].
- *Translate into PNs:* CoMeta automatically translates the selected organisms and pathways into corresponding Petri nets, by using an extension of the tool MPath2PN [9]. The PNML files describing the Petri nets thus obtained are available for further processing.
- *Compute T-invariants:* CoMeta uses the tool INA [47] to compute the bases of semi-positive T-invariants of the PN representations.
- *Compute Distances:* CoMeta automatically computes the reactions and invariants distances as defined in Section 3.2, and allows the user to specify the parameter α used for computing the combined distance. Distance matrices can be exported as text files. Moreover, CoMeta allows the user to inspect the details of the comparison between any pair of organisms (T-invariants bases, matches between invariants, reactions and invariants scores, etc.).
- *Show Phylogenetic trees:* the combined distance matrix may be the input of a phylogenetic tree construction method. Currently CoMeta implements the UPGMA and Neighbour Joining methods, and displays the corresponding phylogenetic trees.

4.1 Experiments

In order to validate our proposal CoMeta has been applied to many sets of organisms. We next show some interesting experiments.

Experiment 1. In the first experiment we consider the *glycolysis* pathway in five eucaryotes: *Homo sapiens* (HSA), *Rattus norvegicus* (RNO), *Meleagris gallopavo* (MGP), *Sus scrofa* (SSC), *Saccharomyces cerevisiae* (SCE).

The combined distance has been computed with the parameter α ranging in $\{0.00, 0.25, 0.50, 0.75, 1.00\}$. The corresponding phylogenetic trees built with the UPGMA method are shown in Figure 2.

The tree in Figure 2 (left) is built with $\alpha = 1$, i.e., by considering in the comparison only homology of reactions. In this case *Homo sapiens* and *Rattus norvegicus* are closely classified because they have the same *glycolysis* pathway, but *Meleagris gallopavo* is incorrectly close to them. The tree does not change for $\alpha = 0.75$. The tree in Figure 2 (right) is obtained with $\alpha = 0.5$, hence besides homology of reactions, it considers also the similarity of T-invariants

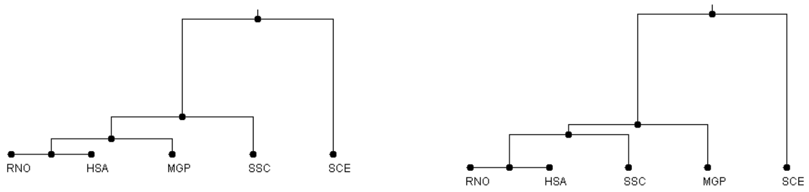


Fig. 2. UPGMA trees for Experiment 1, with $\alpha = 1$ (left) and $\alpha \leq 0.5$ (right).

(with weigth 0.5). This modifies the classification which now matches exactly the standard NCBI taxonomy [4]. With α smaller than 0.5, i.e., by increasing the relevance of the T-invariants in the computation of the distance, we obtain the same phylogenetic tree.

In this experiment the classification based on glycolysis obtained by considering only the distance on reactions does not match the NCBI taxonomy and it improves by taking into account the distance on T-invariants, i.e., the combined distance produces a better classification. This is not always true as shown by the next experiment.

Experiment 2 In this experiment we consider four eucaryotes − *Homo sapiens* (HSA) *Rattus norvegicus* (RNO) *C. elegans* (CEL) *Drosophila melanogaster* (DME) − and a bacterium − *E. coli* (ECO) − and three metabolic pathways, *glycolysis*, *pyruvate metabolism* and *purine metabolism.*

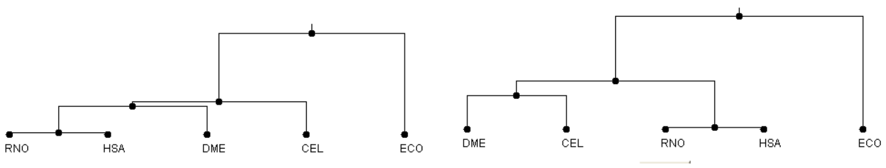


Fig. 3. UPGMA trees for experiment 2, with $\alpha = 1$ (left) and $\alpha \leq 0.75$ (right).

The results obtained with α ranging in $\{0.00, 0.25, 0.50, 0.75, 1.00\}$ are shown in Figure 3. The phylogenetic trees are built with the UPGMA method. The tree on the left of Figure 3, corresponds to $\alpha = 1$ and thus it considers only similarity of reactions in the comparison. This classification matches exactly the standard NCBI taxonomy [4] of the considered organisms. The tree in Figure 3 (right), corresponds to consider similarity both of reactions and of T-invariants, with $\alpha = 0.75$. The classification changes and it does not match any longer the standard NCBI taxonomy. This remains true by increasing the relevance of T-invariants i.e., with $\alpha = 0.50$ or smaller.

In this experiment, by considering the distance based on reactions we get a classification of the organisms matching the NCBI taxonomy. This is no longer true when considering also T-invariants. This could be due to the fact that the reference NCBI taxonomy considers many characteristics of the organisms, not just a few metabolic functions as we do. In general, this shows that further experiments are necessary for understanding how to use our combined distance.

Experiment 3 The third experiment is conducted on a set of 16 organisms, mainly bacteria, w.r.t. the *glycolysis* pathway. It has been originally used in [20] as a test case and then considered also in [10]. The organisms and their reference NCBI taxonomy are show in Figure 4.

Focusing on an experiment already studied in the literature helps in comparing our technique with other proposals, although, as clarified below, a precise comparison is quite difficult for the variability of data sources and reference classifications.

Cod.	Organism	Reign
afu	A. fulgidus	Archea
mja	M. jannaschii	Archea
cpn	C. pneumoniae	Bacteria
mge	M. genitalum	Bacteria
mpn	M. pneumoniae	Bacteria
hin	H. influenzae	Bacteria
syn	Synechocystis	Bacteria
dra	D. radiodurans	Bacteria
mtu	M. tuberculosis	Bacteria
tpa	T. pallidum	Bacteria
bsu	B. subtilis	Bacteria
aae	A. aeolicus	Bacteria
tma	T. maritima	Bacteria
eco	E. coli	Bacteria
hpy	H. pylori	Bacteria
sce	Saccharomyces cerevisiae	Eucaryotes

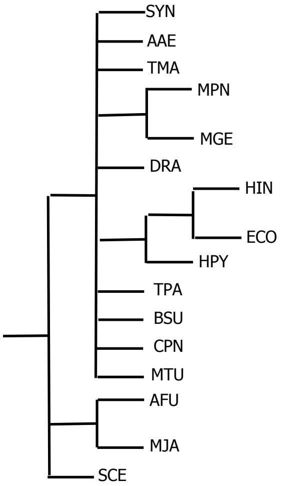


Fig. 4. Left: organisms for experiment 3. Right: reference NCBI taxonomy

As in the previous experiments, α ranges in $[0, 1]$, phylogenetic trees are built using the UPGMA method and they are compared with the reference NCBI classification of the 16 organisms. In order to perform such a comparison, following [20, 10], we used the *cousins* tool [53, 45] with threshold 2. The tool compares unordered trees with labelled leaves by counting the sets of common cousin pairs up to a certain cousin distance.[2] The outcome is reported in the table in Fig-

---

[2] A *cousin pair* is a triple consisting of a pair of leaves and their cousin distance: 0 if they are siblings (same parent), 0.5 if the parent of one of them is the grandparent of the other, 1 if they are cousins (same grandparent but not same parent), 1.5 if their first common ancestor is the grandparent of one of them and the great-grandparent

ure 5 (left). Our best result, 0.3163265, corresponds to the phylogenetic tree in Figure 5 (right) and to the combined distance with $\alpha \in [0.50, 0.75]$.

α	Similarity value
0.00	0.25
0.25	0.2673267
0.50	0.3163265
0.75	0.3163265
1.00	0.2621359

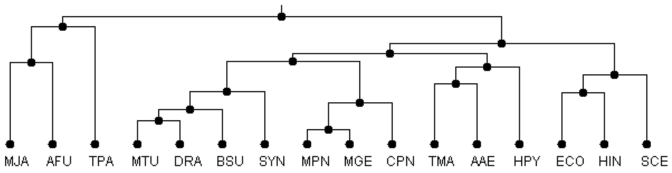


Fig. 5. Results for experiment 3. Left: similarity values computed with *cousins*. Right: UPGMA phylogenetic tree ($\alpha \in [0.50, 0.75]$).

Our results cannot be immediately compared with those in [20, 10]. In fact, the reference NCBI classification of the 16 organisms (and apparently also the corresponding KEGG data) has been changing in the meantime. Nevertheless, the experiment suggests that our technique produces results which are at least comparable with those in [20, 10].

In particular, in [20] a pathway is represented as an *enzyme graph* and a distance is defined which takes into account both the structure of the graph and the similarity between corresponding nodes. A phylogenetic tree is built with the resulting distance matrix by using the Neighbour Joining method. According to the authors, *cousins* provides a similarity value of 0.26 between their phylogenetic tree and their reference NCBI taxonomy and this outperforms the results of the phylogenies obtained by NCE, 16SrRNA and [29]. Hence our results improves those obtained in [20]. Although space limitations prevent us to report the details here, this is true also when we use Neighbour Joining trees.

Instead, in [10] a heuristic comparison algorithm is proposed which computes the intersection and symmetric difference of the sets of compounds, enzymes, and reactions in the metabolic pathways of different organisms. Their algorithm gives in output a similarity matrix which is used by a fuzzy equivalence relations-based (FER) hierarchical clustering method to compute the classification tree. The authors were not able to reproduce the experiment in [20]. In the *cousins* comparison w.r.t. the reference NCBI taxonomy their best result has a similarity value of 0.3195876, which is very close to our best result.

5 Conclusions

Biological questions related to evolution and to differences among organisms can be answered by comparing their metabolic pathways. In this paper we propose a new similarity measure for metabolic pathways which combines a similarity

---

of the other one, 2 if they are second cousins (same great-grandparent but not same grandparent) and so on.

based on reactions and a similarity based on behavioural aspects such as potential fluxes, which correspond to the minimal T-invariants of the Petri net representation of a pathway.

We implemented a tool, CoMeta, to experiment with our proposal. It is not easy to compare the results we obtained with those in the literature. Nevertheless experiments made with CoMeta showed that:

- Our combined measure produces valid phylogenetic classifications.
- Neither the comparison based on reactions nor the one based on T-invariants gives always correct results. The refinement due to the introduction of the behavioural measure can be useful, but further investigations are necessary to determine how to combine properly the two measures.
- Measures based on more sophisticated representations of a pathway (e.g., using graphs rather than sets, or considering also compounds besides enzymes) not necessarily give better results than our combined measure, as our third experiment shows. However also this hypothesis needs further experiments to be verified.

We are performing extensive studies on the distributions of the two proposed distances. This could reveal correlations between them, and, possibly, give insights on the ranges for the α parameter (influence of the T-invariants on the combined distance) providing the best results. We are also extending CoMeta to deal with a more refined similarity measure on EC numbers, the hierarchical similarity. We plan to add also the Tanimoto index (extended Jacard index) as an alternative to the Sørensen index. This would allow us to compare and evaluate different measures. When comparing organisms on large sets of pathways, a further extension would be to associate weights to the pathways. Weights could be chosen by the user in order to put more emphasis on some pathways of interest or could be derived on the basis of characteristics of the pathways, like their size.

Moreover, it would be very interesting to compare different organisms by considering their whole metabolic networks. This would allow one to identify more properly the T-invariants corresponding to functional units in the metabolic network. In fact, when considering single pathways some T-invariants can be not recognisable since they might be split in different pathways. However, the additional information deriving from the partitioning in well established functional pathways would be lost. Additionally, comparing full metabolic networks could be not viable from a computational point of view since in the worst case Hilbert bases can be exponential in the size of the original net.

CoMeta is part of a larger project to integrate various tools for representing and analysing metabolic pathways through Petri nets. We intend to use the distance matrices computed by CoMeta for different analyses. CoMeta is freely available at: http://www.dsi.unive.it/∼simeoni/CometaTool.tgz.

Acknowledgements. We are grateful to Paolo Besenzon and Silvio Alaimo for their contribution to the implementation of the tools used for the experiments.

References

1. Kegg Markup Language manual. http://www.genome.ad.jp/kegg/docs/xml.
2. KEGG pathway database - Kyoto University Bioinformatics Centre. http://www.genome.jp/kegg/pathway.html.
3. Petri net tools. http://www.informatik.uni-hamburg.de/TGI/PetriNets/tools.
4. Taxonomy - site guide - NCBI. http://www.ncbi.nlm.nih.gov/guide/taxonomy/.
5. F. Ay, M. Dang, and T. Kahveci. Metabolic network alignment in large scale by network compression. *BMC Bioinformatics*, 13 (Suppl 3), 2012.
6. F. Ay, T. Kahveci, and V. de Crecy-Lagard. Consistent alignment of metabolic pathways without abstraction. In *Int. Conf. on Computational Systems Bioinformatics (CSB)*, pages 237–248. 2008.
7. F. Ay, M. Kellis, and T. Kahveci. SubMAP: Aligning metabolic pathways with subnetwork mappings. *Journal of Computational Biology*, 18(3):219–235, 2011.
8. P. Baldan, N. Cocco, A. Marin, and M Simeoni. Petri nets for modelling metabolic pathways: a survey. *Natural Computing*, 9(4):955–989, 2010.
9. P. Baldan, N. Cocco, F. De Nes, M. Llabrés Segura, and M. Simeoni. MPath2PN - Translating metabolic pathways into Petri nets. In M. Heiner and H. Matsuno, editors, *BioPPN2011 Int. Workshop on Biological Processes and Petri Nets*, CEUR Workshop Proceedings, pages 102–116, 2011.
10. J. Casasnovas, J.C. Clemente, J. Miró-Julià, F. Rosselló, K. Satou, and G. Valiente. Fuzzy clustering improves phylogenetic relationships reconstruction from metabolic pathways. In *Proc. of the 11th Int. Conf. on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, 2006.
11. M. Chen and R. Hofestadt. Web-based information retrieval system for the prediction of metabolic pathways. *IEEE Trans. on NanoBioscience*, 3(3):192–199, 2004.
12. Q. Cheng, R. Harrison, and A. Zelikovsky. MetNetAligner: a web service tool for metabolic network alignments. *Bioinformatics*, 25(15):1989–1990, 2009.
13. JC. Clemente, K. Satou, and G. Valiente. Reconstruction of phylogenetic relationships from metabolic pathways based on the enzyme hierarchy and the gene ontology. *Genome Informatics*, 16(2):45–55, 2005.
14. O. Ebenhöh, T. Handorf, and R. Heinrich. A cross species comparison of metabolic network functions. *Genome Informatics*, 16(1):203–213, 2005.
15. J. Esparza and M. Nielsen. Decidability issues for Petri Nets - a survey. *Journal Inform. Process. Cybernet. EIK*, 30(3):143–160, 1994.
16. C.V. Forst, C. Flamm, I. L. Hofacker, and P. F. Stadler. Algebraic comparison of metabolic networks, phylogenetic inference, and metabolic innovation. *BMC Bioinformatics*, 7(67), 2006.
17. C.V. Forst and K. Schulten. Evolution of metabolism: a new method for the comparison of metabolic pathways using genomics information. *Journal of Computational Biology*, 6(3/4):343–360, 1999.
18. C.V. Forst and K. Schulten. Phylogenetic analysis of metabolic pathways. *Journal of Molecular Evolution*, 52(16):471–489, 2001.
19. M. Heiner and I. Koch. Petri Net Based Model Validation in Systems Biology. In *Petri Nets and Other Models of Concurrency - ICATPN 2004*, volume 3099 of *LNCS*, pages 216–237. Springer, 2004.
20. M. Heymans and A. M. Singh. Deriving phylogenetic trees from the similarity analysis of metabolic pathways. *Bioinformatics*, 19(1):i138–i146, 2003.

21. R. Hofestädt. A Petri net application of metbolic processes. *Journal of System Analysis, Modelling and Simulation*, 16:113–122, 1994.
22. S. H. Hong, T. Y. Kim, and S. Y. Lee. Phylogenetic analysis based on genome-scale metabolic pathway reaction content. *Appl. Microbiol. Biotechnology*, 65:203–210, 2004.
23. M. Kanehisa, M. Araki, S. Goto, M. Hattori, M. Hirakawa, M. Itoh, T. Katayama, S. Kawashima, S. Okuda, T. Tokimatsu, and Y. Yamanishi. KEGG for linking genomes to life and the environment. *Nuc. Acids Research*, pages 480–484, 2008.
24. G. W. Klau. A new graph-based method for pairwise global network alignment. *BMC Bioinformatics*, 10(Suppl 1), 2009.
25. 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 & Sons, 2008.
26. O. Kuchaiev, T. Milenkovic, V. Memisevic, W. Hayes, , and N. Przulj. Topological network alignment uncovers biological function and phylogeny. *J. R. Soc. Interface*, 7(50):1341–1354, 2010.
27. Y. Li, D. de Ridder, M.J.L. de Groot, and M.J.T. Reinders. Metabolic pathway alignment between species using a comprehensive and flexible similarity measure. *BMC Systems Biology*, 2008.
28. Z. Li, S. Zhang, Y. Wang, X.S. Zhang, and L. Chen. Alignment of molecular networks by integer quadratic programming. *Bioinformatics*, 23(13):1631–1639, 2007.
29. S. Liao, L.and Kim and J.F. Tomb. Genome comparisons based on profiles of metabolic pathways. In *Proc. of the 6th Int. Conf. on Knowledge-Based Intelligent Information and Engineering Systems (KES 02)*, pages 469–476, 2002.
30. E. Lo, T. Yamada, M. Tanaka, M. Hattori, S. Goto, C. Chang, and M. Kanehisa. A method for customized cross-species metabolic pathway comparison. In *Proc. of Genome Informatics 2004*. GIW 2004 Poster Abstract: P068, 2004.
31. A. Mithani, G.M. Preston, and J. Hein. Rahnuma: Hypergraph based tool for metabolic pathway prediction and network comparison. *Bioinformatics*, 25(14):1831–1832, 2009.
32. T. Murata. Petri Nets: Properties, Analysis, and Applications. *Proceedings of IEEE*, 77(4):541–580, 1989.
33. S. Oehm, D. Gilbert, A. Tauch, J. Stoye, and A. Goessmann. Comparative Pathway Analyzer - a web server for comparative analysis, clustering and visualization of metabolic networks in multiple organisms. *Nuc. Acids Research*, 36:433–437, 2008.
34. R.Y. Pinter, O. Rokhlenko, E. Yeger-Lotem, and M. Ziv-Ukelson. Alignment of metabolic pathways. *Bioinformatics*, 21(16):3401–3408, 2005.
35. V. N. Reddy. Modeling Biological Pathways: A Discrete Event Systems Approach. Master's thesis, The Universisty of Maryland, M.S. 94-4, 1994.
36. V. N. Reddy, M.N. Liebman, and M.L. Mavrovouniotis. Qualitative Analysis of Biochemical Reaction Systems. *Comput. Biol. Med.*, 26(1):9–24, 1996.
37. V. N. Reddy, M. L. Mavrovouniotis, and M. N. Liebman. Petri net representations in metabolic pathways. In *ISMB93: First Int. Conf. on Intelligent Systems for Molecular Biology*, pages 328–336. AAAI press, 1993.
38. C. H. Schilling, D. Letscherer, and B. O. Palsson. Theory for the systemic definition of metabolic pathways and their use in interpreting metabolic function from a pathway-oriented perspective. *Journal of Theoretical Biology*, 203:229–248, 2000.
39. C. H. Schilling, S. Schuster, B. O. Palsson, and R. Heinrich. Metabolic pathway analysis: basic concepts and scientific applications in the post-genomic era. *Biotechnol. Prog.*, 15:296–303, 1999.

40. A. Schrijver. *Theory of linear and integer programming.* Wiley-Interscience series in discrete mathematics and optimization. Wiley, 1999.
41. S. Schuster, T. Dandekar, and D. A. Fell. Detection of elementary flux modes in biochemical networks: a promising tool for pathway analysis and metabolic engineering. *Trends Biotechnology*, 17(March):53–60, 1999.
42. S. Schuster, D. A. Fell, and T. Dandekar. A general definition of metabolic pathway useful for systematic organization and analysis of complex metabolic networks. *Nature Biotechnology*, 18(March):326–332, 2000.
43. S. Schuster and C. Hilgetag. On elementary flux modes in biochemical reaction systems at steady state. *Journal of Biological Systems*, 2:165–182, 1994.
44. S. Schuster, T. Pfeiffer, F. Moldenhauer, I. Koch, and T. Dandekar. Exploring the pathway structure of metabolism: decomposition into subnetworks and application to Mycoplasma pneumoniae. *Bioinformatics*, 18(2):351–361, 2002.
45. D. Shasha, J. T. L. Wang, and S. Zhang. Unordered tree mining with applications to phylogeny. In *20th Int. Conf. on data engineering*, pages 708–719. IEEE Computer Society, 2004.
46. T. Sørensen. A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on danish commons. *Biologiske Skrifter / Kongelige Danske Videnskabernes Selskabg*, 5(4):1–34, 1948.
47. P.H. Starke and S. Roch. The Integrated Net Analyzer. *Humbolt University Berlin*, 1999. www.informatik.hu-berlin.de/ starke/ina.html.
48. Y. Tohsato. A method for species comparison of metabolic networks using reaction profile. *Inf. and Media Technology*, 2(1):109–114, 2007.
49. Y. Tohsato, H. Matsuda, and A. Hashimoto. A multiple alignment algorithm for metabolic pathway analysis using enzyme hierarchy. In *Proc. Int. Conf. Intell. Syst. Mol. Biol.*, pages 376–383, 2000.
50. Y. Tohsato and Y. Nishimura. Metabolic pathway alignment based on similarity between chemical structures. *Inf. and Media Technology*, 3(1):191–200, 2008.
51. E. C. Webb. *Enzyme nomenclature 1992: recommendations of the Nomenclature Committee of the International Union of Biochemistry and Molecular Biology on the nomenclature and classification of enzymes.* San Diego: Published for the International Union of Biochemistry and Molecular Biology by Academic Press, 1992.
52. S. Wernicke and F. Rasche. Simple and fast alignment of metabolic pathways by exploiting local diversity. *Bioinformatics*, 23(15):1978–1985, 2007.
53. K. Zhang, J.T.L. Wang, and D. Shasha. On the editing distance between undirected acyclic graphs. *Int. Journal of Foundations of Computer Science*, 3(1):43–57, 1996.

A Database-supported Modular Modelling Platform for Systems and Synthetic Biology

Mary Ann Blätke and Wolfgang Marwan

Lehrstuhl für Regulationsbiologie and Magdeburg Centre for Systems Biology,
Otto-von-Guericke Universität, Universitätsplatz 2, 39106 Magdeburg, Germany
`maryann.blaetke@ovgu.de`

Extended Abstract

Background. Facilitating the handling of the increasing number of kinetic models is a challenging task in biomodel engineering. Usually advanced mathematical skills are required to generate, curate, update or to just perform simulations with a kinetic model. This makes modelling less popular in the life sciences. In addition, an on-going trend in biomodel engineering is the representation of biological systems in the form of monolithic models, which are inherently complex due to the numerous interwoven components. This makes them hardly accessible to others, especially for curation and updating. Moreover, monolithic models usually cannot be linked without extensive restructuring. Therefore, biomodel engineering would benefit from a universal and unifying modelling platform facilitating the use of models and making them more appealing even for wet lab life scientists.

Results. We developed a modular modelling concept, where modules focus on the reactions of each individual protein with its specific interaction partners as described by a Petri net [1–4]. These Petri net modules are graphically displayed, can be executed individually and may be coupled with via specific connection interfaces. Our concept is supported by a prototype database with a publically accessible web-interface [1]. In addition to the network structure of each module, the database contains metadata for documentation purposes. Therefore, each module corresponds to a wiki-like minireview. The database can manage multiple versions of each module. The organization of molecule-oriented modules in a database facilitates the automatic composition into coherent models containing an arbitrary number of molecular species chosen ad hoc by the user. Petri nets composed from modules can be executed as ODEs, stochastic, hybrid, or merely qualitative models and exported in SMBL format.

Conclusion. The modular modelling concept and its extension by a supportive database, facilitates the curation, documentation, version control, and update of individual modules and the subsequent automatic composition of complex models, without requiring mathematical skills [1–4]. Modules can be recombined according to user-defined scenarios, e.g., considering the gene expression patterns in given cell types, under certain physiological conditions, or states of disease. As synthetic biology application we propose we propose the fully automated generation of synthetic or synthetically rewired network models by composition of metadata-guided automatically modified modules representing altered protein binding sites.

References

1. 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. submitted (2012)
2. Blätke, M.A., Heiner, M., Marwan, W.: Predicting Phenotype from Genotype Through Automatically Composed Petri Nets. submitted (2012)
3. Blätke, M., Meyer, S., Marwan, W.: Pain signaling - a case study of the modular petri net modeling concept with prospect to a protein-oriented modeling platform. In: Proc. of the International Workshop on Biological Processes & Petri Nets (BioPPN), satellite event of Petri Nets 2011. CEUR Workshop Proceedings, vol. 724, pp. 117–134. CEUR-WS.org (June 2011)
4. Blätke, M., Meyer, S., Stein, C., Marwan, W.: Petri net modeling via a modular and hierarchical approach applied to nociception. In: Proc. International Workshop on Biological Processes & Petri Nets (BioPPN), satellite event of Petri Nets 2010. pp. 131–143. ISBN: 978-972-8692-53-7 (June 2010)

Petri nets in VANTED: Simulation of Barley Seed Metabolism

Anja Hartmann[1,*], Hendrik Rohn[1], Kevin Pucknat[1], and Falk Schreiber[1,2]

[1] Leibniz Institute of Plant Genetics and Crop Plant Research (IPK),
Corrensstrasse 3, 06466 Gatersleben, Germany
`{hartmann,rohn,pucknat,schreibe}@ipk-gatersleben.de`
[2] Martin Luther University Halle-Wittenberg, Institute of Computer Science,
Von-Seckendorff-Platz 1, 06120 Halle, Germany

Abstract. Petri nets are a mathematical language, which provide a unified environment for modeling, simulation, and formal analysis of biological systems. To support the applicability of Petri nets for biological users we implemented a Petri net add-on for the widely used VANTED framework. VANTED supports Petri net reconstruction, simulation capabilities to be able to investigate dynamic system behavior, and analysis algorithms for calculating intrinsic net properties. VANTED furthermore supports advanced visualization and exploration techniques, which can be used to examine even larger Petri nets in an interactive manner. We use this framework for the simulation-based analysis of a large stoichiometric model of central barley seed metabolism and discuss problems and obstacles during this process.

Keywords: Petri net, Simulation, Analysis, VANTED, Metabolic model

1 Introduction

Metabolic network models can be analyzed using various approaches, such as topological analysis (e. g., centralities), stoichiometric analysis (e. g., Flux Balance Analysis) or kinetic modeling, each corresponding to a different level of detail and a different level of available information. Petri nets can be used in order to quantitatively model, simulate, and analyze biological systems without the need for detailed and difficult to obtain measurements, such as enzyme activities or metabolite levels. Until now several tools were released to utilize the power of Petri nets, but their focus is often on analytical approaches, such as calculating invariants, hence their applicability is limited to small networks.

We developed a Petri net add-on for VANTED [6] focusing on the needs for a metabolic modeling pipeline, which enables simulation, interactive exploration, and interpretation of various properties based on continuous and discrete place-transition nets [22]. VANTED is a network editing framework, which supports researchers in the interpretation of experimental data visualized as charts in the context of biological networks, and thus is a widely-used tool in systems biology. To benefit from the broad functionalities of VANTED and to complement other modeling approaches, we extended Vanted for the analysis and simulation offered by Petri nets through the Petri net add-on. An example of the combination of experimental data and Petri net models is the integration of omics data represented as charts inside the Petri net nodes. This Petri net models enable the comparison of metabolic effects (places) or enzyme activities (transitions) with simulation results. Vanted can furthermore be used to simulate metabolic models by the use of stoichiometric analysis methods, such as flux balance analysis.

The second section describes the software architecture and evaluates the known Java-based Petri net tools, highlights special properties of metabolic Petri net models, and explains exploration techniques. The third section shows the application of Vanted for the simulation-based analysis of a large stoichiometric model of central barley seed metabolism and discusses problems and obstacles during this process.

2 Methods and Tool

2.1 Software Architecture

In order to evaluate tools which may be used as an extension for Vanted, a survey of Java-based Petri net tools and libraries was performed (see Table 1). The requirements were availability under an open source license, support for discrete and continuous Petri nets, simulation in single- and multiple steps, analysis of place- and transition-invariants, and calculation of reachability [23].

Table 1. Evaluation of Java-based open source Petri net tools and libraries. Licenses abbreviations: Lesser General Public License (LGPL), Berkeley Software Distribution (BSD), General Public License (GPL), Open Software License (OSL), Academic Free License (AFL), Eclipse Public License (EPL).

Name	**License**	**Petri-net Types**		**Simulation**		**Analysis**		
				Single-step	**Multi-step**	**Invariant**		**Reachability**
		Discrete	**Continuous**			**P**	**T**	
Jfern	LGPL	✓	✓	✓	✓			
Tortuga	LGPL	✓		✓				
HISim	BSD	✓	✓	✓	✓			
JARP	GPL	✓		✓				
PIPE2	OSL3.0	✓		✓	✓	✓	✓	✓
Petri-LLD	GPL	✓		✓				
JPetriNet	AFL	✓		✓				
Renew	LGPL	✓	✓	✓	✓			
FERN	LGPL	✓		✓	✓			
PNEditor	GPLv3	✓		✓				
TAPAAL	OSL3+BSD	✓		✓				
WoPeD	LGPL	✓		✓	✓			✓
KIT-HORUS	EPL	✓		✓	✓			

The Petri net tools and libraries JFern [9], HISim [12], and Renew [13] can process discrete and continuous Petri nets which can be simulated in single- and multiple steps, however they do not support analytical methods. Pipe2 [2] and WoPeD [11] can handle discrete Petri nets and simulate in single- and multiple steps. Pipe2 enables the computation of reachability and invariants, whereas WoPeD enables only the computation of reachability. All other tools do neither support analytical methods nor the simulation of continuous Petri nets. FERN [14] and KIT-Horus [15] can simulate discrete Petri nets in single- and multiple steps whereas Tortuga [16], Petri-LLD [17], JPetriNet [18], PNEditor [19], JARP [20], and TAPAAL [21] enable the simulation of just discrete Petri nets in single steps.

It has become apparent that no single tool was able to satisfy all requirements. Therefore we decided to incorporate two tools. JFern [9] is a compact and native Java library being able to handle object-oriented-, timed-, high-level-, and place-transition nets. It is used to perform basic Petri net operations on continuous and discrete place-transition nets. As the library does not comprise analytical methods, Pipe2 [2] is used to perform reachability, and invariant analysis. Both libraries are complemented by the capabilities of the VANTED framework itself. It provides various network importers (e. g., for SBML-, KGML-, and GML files) and direct access to network databases (e. g., MetaCrop [10], KEGG [7]). For network exchange with the Petri net community PNML [5] import and export functionality was implemented. Various image exporters enable the intuitive communication of simulation and analysis results within the scientific community. Finally, networks can be edited, semi-automatically transformed into Petri nets, layouted, and extended with other systems biology data, such as gene expression and metabolic data.

VANTEDs workflow is structured in three parts: In order to create valid Petri nets, manual editing or semi-automatic network transformations can be used in the reconstruction step. The simulation step enables to interactively follow the flow of tokens through the Petri net. Finally, complementary analytical functions to calculate intrinsic net properties can be performed in the analysis step and examined using exploration techniques. These steps are explained in detail in the following sections and can be performed using the Petri net add-on available for download under http://www.vanted.org/petrinet.

2.2 Model Reconstruction

A Petri net can be manually reconstructed with VANTED. This tedious process can be circumvented by accessing metabolic networks from files or public databases. The process of transforming such networks into syntactically valid Petri nets is supported by semi-automatic transformation functionalities which, for example, enable to assign place or transition roles to selected nodes. A user may choose to transform all selected nodes into places and unselected nodes are implicitly assigned as transitions (or vice versa). Another transformation converts hypergraphs (e. g. networks consisting of metabolites represented as nodes connected by hyperedges representing enzymes) in Petri nets by transforming all existing nodes into places and splitting edges between these places into two arcs by inserting transition nodes. The correctness of generated Petri nets is validated before analysis and simulation is performed. VANTED examines that arcs run only from places to transitions or vice versa and checks for unconnected elements in the Petri net. Also the type of the Petri net is considered, as VANTED checks for consistent use of discrete or continuous Petri net elements by examining tokens, place-capacities and arc-weights.

Special focus was put on unique properties of metabolic Petri net reconstruction in accordance to the review of Baldan et al. [1]. The modeling of spatial properties, such as compartmentation is solved by adding suffixes to each metabolite name, thus distinguishing pathways distributed between different compartments. Multiple occurring metabolites in different reactions, such as ATP and CO_2 are represented as logical places, thereby preventing multiple edge crossings throughout the network. External metabolites for token import are modeled as source transitions (regular import) or source places (controlled import by specifying a certain amount of tokens), whereas token export is realized by creating sink transitions (export if the precondition is achieved) or sink places (traceable accumulation). In case of reversible reactions VANTED enables the use of hierarchical transitions, which are visualized as single reactions but internally split up into two distinct forward and backward reactions.

2.3 Model Simulation

Simulation of the token flow through the Petri net is an important possibility to determine dynamic properties of metabolic networks. Initially, tokens are set for all places and a user-defined number of simulation steps is performed. The marking is visualized for each step, resulting in an animation of token flow throughout the Petri net. For faster simulation results a number of steps can be performed in the background, resulting in the visualization of the final marking. For further analysis the marking of each simulation step can be exported as a text file and imported into other tools, such as MS Excel or mapped to the corresponding places in VANTED in order to examine the number of assigned tokens for each simulation step.

2.4 Model Analysis

Petri net theory provides a variety of analytical methods in order to gain information on metabolic network behavior. VANTED enables the calculation and visualization of the reachability graph, as well as place- and transition-invariants. Such methods enable the detection of structural inconsistencies, such as traps, boundedness, liveness, safeness, and deadlocks, thereby semantically checking the Petri net structure (for definitions see [23]). In order to support interpretation of analytical approaches, interactive visualization techniques were developed based on the brushing and linking concept [8]: Calculated invariants are listed in a dialog, which supports a mouse-over selection effect, resulting in a linked immediate visualization of the particular invariant in the Petri net view. Analogously, the calculated reachability graph is visualized as an additional network and also reacts on mouse-over events, see Figure 1. Here a node mouse-over triggers the visualization of the particular marking in the Petri net view which represents a reachable state depending on the initial marking (S0 in part a). Mouse-over an arc results in the highlighting of a firing transition (t2 in part b). This interaction technique enables users to understand non-deterministic features of Petri nets and enables detailed investigation of dynamic changes of marking even in large reachability graphs.

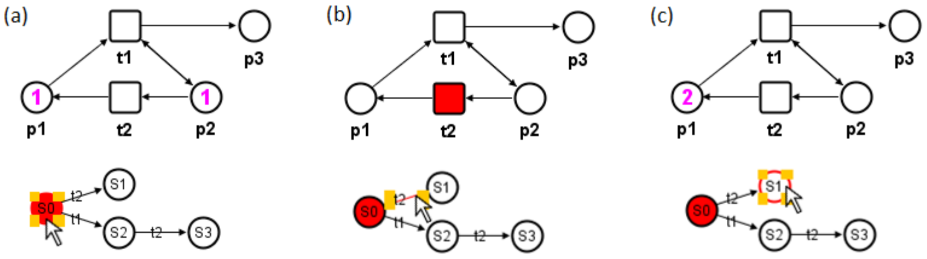


Fig. 1. Interactive visualization of the Petri net reachability analysis. (a) Initial marking is represented by the initial state S0 (marked) of the reachability graph, which is visualized in the Petri net. (b) Mouse-over an edge of the reachability graph results in a visualization of the firing transition (t2), which would lead to the next reachable state S1 (c).

3 Simulation of a Comprehensive Barley Seed Petri net

The barley seed Petri net is based on a stoichiometric model of central seed metabolism consisting of 234 metabolites and 257 reactions in different compartments, with the aim of getting a systemic understanding of barley seed storage metabolism and to study grain yield [3]. As the calculation of invariants for such large models is not feasible, the goal of this use case is to show how such large models can be converted into valid Petri nets and be investigated using the simulation capabilities of VANTED.

The stoichiometric model defined as an SBML file (see supplementary material of [3]) is imported into VANTED, layouted, and all compartmentation represented by splitting places and adding suffixes to the place names. No place-capacities were set and arc-weights (representing stoichiometry) were automatically recognized. The boundary is defined according to the goal of observing biomass accumulation by adding source transitions to allow regular import of external metabolites and sink places to trace the accumulation of produced metabolites. The starch degradation pathway was excluded from the simulation in order to prevent a drain of starch into this pathway. This is because increasing starch levels would otherwise be degraded immediately instead of being used for biomass synthesis. An initial marking of 10 tokens for each place was set and the simulation was performed over 1000 steps. The marking gets visualized and exported for each step, animating the accumulation or depletion of metabolites over time. Figure 2 comprises important metabolites in the starch metabolism with the number of assigned tokens over these steps.

After a transient oscillation at the beginning of the simulation, starch and biomass converge into a stable marking, because no mitochondrial energy (ATP) is available (step 20). As soon as enough ATP is produced (step 100), starch is metabolized into biomass and depletes at step 165. The reason for the limitation of biomass production by lack of starch is the energy household which apparently is a deadlock: Mitochondrial ATP is not transported to the cytosol or plastid, because the export of ATP from the mitochondria demands an import of ADP from the cytosol. Instead the cytosolic ADP gets depleted very early due to the drain of other reactions. As the production of starch precursors G1P and ADPglc depends on cytosolic and plastidic ATP, starch cannot be produced.

The basic issue is the small pool of ATP and ADP compared to the large number of reactions consuming these substances in the cytosol. Such an inbalance increases the impact of non-deterministic firing of reactions being in conflict. This non-deterministic decision causes e. g. the consumption of free ADP and thereby preventing the necessary transport of ADP into mitochondria in order to export ATP. As the behavior of biochemical networks is inherently governed by stochastic laws [4], the utilization of stochastic Petri nets could resolve such dynamic conflicts.

To understand the causality of such complex interactions between different compartments, additional tests should be applied in order to solve this deadlock by adding or excluding biochemical reactions in form of transitions.

4 Conclusion and Outlook

We have shown a promising approach to simulate large metabolic models with a new Petri net add-on for the VANTED framework. Based on the broad functionality of VANTED a transformation of existing metabolic models into Petri nets and their simulation is powerful and easy to use. The barley seed model, developed for the optimization of biomass production, proved to be difficult to transfer directly into a Petri net. Typical behavior, such as non-deterministic firing of transitions and preferred firing for branches with less token uptake needs further adjustments of the metabolic model. Nevertheless, based on the results we are confident that the enhanced reconstruction and simulation process in VANTED supports users in the analysis of such problems. In the future, we plan to use other Petri net classes such as stochastic Petri nets in order to improve modeling quality of metabolic network properties.

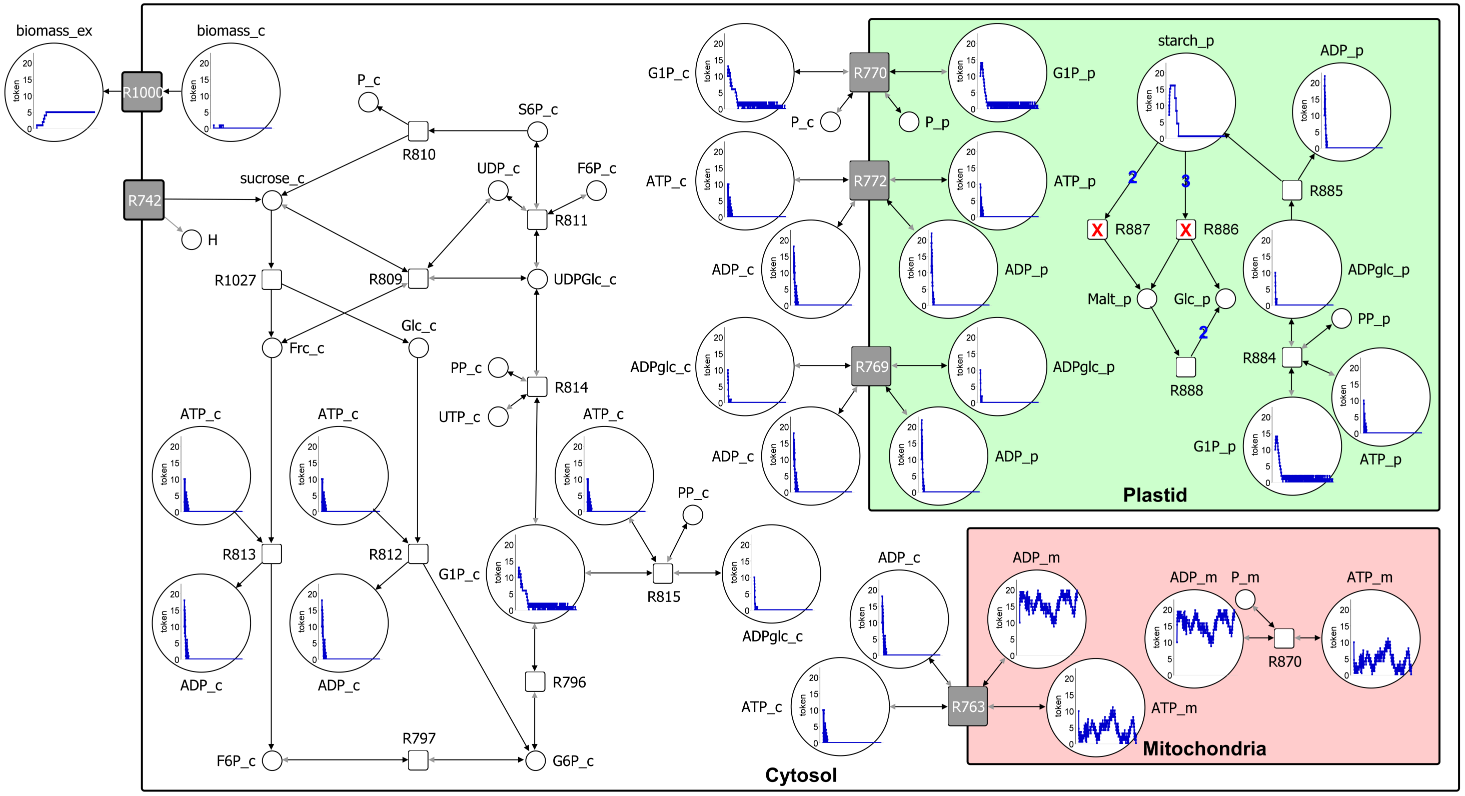


Fig. 2. Schematic overview of the condensed barley seed Petri net focusing on starch and energy metabolism. Charts depict token assignments per simulation step for important metabolites of the barley seed Petri net. The carbon source sucrose is imported into the cytosol (suffix "_c") of the cell through the source transition R742 without restrictions. Different reactions transform sucrose together with the consumption of energy (ATP) in starch precursors in the cytosol (G1P, ADPglc). The precursor ADPglc is produced either in the cytosol (R815) and afterwards transported into the plastid (R769, suffix "_p") or G1P is transported into the plastid (R770) and ADPglc produced in the plastid (R884). The last step of the starch biosynthesis (R885) and also both steps of the starch degradation (R886, R887), which are excluded from simulation as indicated by the cross, are located in the plastid. The starch precursors G1P and ADPglc located in the cytosol and plastid are consumed together with starch in order to build up biomass. Necessary energy (ATP) is produced in the respiratory chain of the mitochondria (R870) and delivered to the cytosol (R763), as well as to the plastid (R772) by exchanging ADP. In the mitochondria the energetic metabolites ATP and ADP act in an alternating manner. The first 165 steps of the simulation show a clear effect of the initial marking of 10 tokens, after which most of the metabolites converge into a stable marking.

References

1. Baldan, P., Cocco, N., Marin, A., Simeoni, M.: Petri nets for modelling metabolic pathways: a survey. Natural Computing 9, pp. 955–989 (2010)
2. Bonet, P., Llado, C., Puigjaner, R., Knottenbelt, W. J.: PIPE v2.5: A Petri Net Tool for Performance Modelling. In: 23rd Latin American Conference on Informatics (CLEI 2007)
3. Grafahrend-Belau, E., Schreiber, F., Koschützki, D., Junker, B. H.: Flux balance analysis of barley seeds: a computational approach to study systemic properties of central metabolism. Plant Physiology 1(149), pp. 585–598 (2009)
4. Heiner, M.: Petri nets for Systems and Synthetic Biology. Natural Computing 10(2), pp. 633–638 (2011)
5. Hillah, L. M., Kindler, E., Kordon, F., Petrucci, L., Treves, N.: A primer on the Petri Net Markup Language and ISO/IEC 15909-2. Petri Net Newsletter 76, pp. 9–28 (2009)
6. Junker, B. H., Klukas, C., Schreiber, F.: VANTED: A system for advanced data analysis and visualization in the context of biological networks. BMC Bioinformatics 7(109) (2006)
7. Kanehisa, M., Goto, S., Sato, Y., Furumichi, M., and Tanabe, M.: KEGG for integration and interpretation of large-scale molecular datasets. Nucleic Acids Research 40, pp. D109-D114 (2012)
8. Martin, A. R., Ward, M. O.: High dimensional brushing for interactive exploration of multivariate data. In: Proceedings on Visualization, pp. 271–278 (1995)
9. Nowostawski, M.: JFern Manual, Version 4.0.0 (2009)
10. Schreiber, F., Colmsee, C., Czauderna,T., Grafahrend-Belau, E., Hartmann, A., Junker, A., Junker, B. H., Klapperstück, M., Scholz, U., Weise, S.: MetaCrop 2.0: managing and exploring information about crop plant metabolism. Nucleic Acids Research 40, pp. D1173-D1177 (2012)
11. Eckleder, A., Freytag, T.: WoPeD A tool for teaching, analyzing and visualizing workflow nets. Petri Net Newsletter 75 (2008)
12. Amengual, A.: A computational model of attachment secure responses in the Strange Situation. Technical report, International Computer Science Institute (2009)
13. Friedrichs, F. D.: Referenznetze mit Anschriften in Scheme. PhD Thesis, University Hamburg (2007)
14. Erhard, F., Friedel, C. C., Zimmer, R.: FERN - a Java framework for stochastic simulation and evaluation of reaction networks. BMC Bioinformatics, 1(9), pp. 356 (2008)
15. Klink, S., Li, Y., Oberweis, A.: INCOME2010 - a toolset for developing process-oriented information systems based on petri nets. In: Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops, Simutools '08, pp. 14:1-14:8. ICST (2008)
16. Kuhl, F.: Tortuga, The MITRE Corporation (2004), `http://code.google.com/p/tortugades`
17. Brusey, J.: Petri(LLD) Tutorial. Department of Engineering, Cambridge University (2006)
18. Azevedo, M.: JPetriNet, Sapucai Valley University - Brazil (2004), `http://jpetrinet.sourceforge.net`
19. Riesz, M.: PNEditor, Slovak University of Technology (2010), `http://www.pneditor.org`

20. Padilha, R. S.: JARP Petri Nets Analyzer, Industrial Automation and Control Engeneering of the Federal University of Santa Catarina - Brazil (2001), `http://jarp.sourceforge.net`
21. Byg, J., Jörgensen, K. Y., Srba, J.: TAPAAL: Editor, Simulator and Verifier of Timed-Arc Petri Nets. In: Proceedings of the 7th International Symposium on Automated Technology for Verification and Analysis (ATVA '09), pp. 5799:84–5799:89, Lecture Notes in Computer Science, Springer-Verlag, Berlin-Heidelberg (2009)
22. Bernardinello, L., de Cindio, F.: A survey of basic net models and modular net classes. In: Rozenberg, Grzegorz: Advances in Petri Nets: The DEMON Project, Lecture Notes in Computer Science, pp. 609:304–351, London, 1992. Springer-Verlag
23. Murata, T.: Petri Nets: Properties, Analysis and Applications. In: Proceedings of the IEEE 77(4), pp. 541–580 (1989)

Proc. BioPPN 2012, a satellite event of PETRI NET 2012

A Hybrid Petri Net Model of the Eukaryotic Cell Cycle

Mostafa Herajy and Martin Schwarick

Brandenburg University of Technology at Cottbus,
Computer Science Institute,
Data Structures and Software Dependability,
Postbox 10 13 44, 03044 Cottbus, Germany
http://www-dssz.informatik.tu-cottbus.de/

Abstract. System level understanding of the repetitive cycle of cell growth and division is crucial for disclosing many unexpected principles of biological organisms. The deterministic or stochastic approach are alone not sufficient to study such cell regulation due to the complex reaction network and the existence of reactions with different time scales. Thus, integration of both approaches is necessary to study such biochemical networks. In this paper we present a hybrid Petri net model to study the eukaryotic cell cycle using Generalised Hybrid Petri Nets. The proposed model is intuitively and graphically represented through Petri net primitives. Moreover, it can capture intrinsic and extrinsic noises and deploys stochastic as well as deterministic reactions. Additionally, self-modifying weights are motivated and introduced to Snoopy – a tool for animating and simulating Petri nets.

Keywords: Generalised hybrid Petri nets; hybrid modelling; eukaryotic cell cycle

1 Introduction

The reproduction of eukaryotic cells is controlled by a complex regulatory network of reactions known as cell cycle [17,18,21]. Through it, cells grow, replicate and divide into two daughter cells [12,19]. This regulation cycle consists of four phases: S phase (synthesis) and M phase (mitosis) separated by two gap phases: G1 and G2 [21]. During the S phase, the cell replicates all of its components, while it divides each component more or less evenly between the two daughter cells at the end of the M phase [12]. After the S phase, there is another gap (G2) where the cell ensures that the duplication of DNA has completed and prepares itself for mitosis. Newborn cells are not replicated and located at the G1 gap. Furthermore, the processes of synthesis and mitosis alternate with each other during the reproduction process. Understanding such control cycles is crucial for revealing defects in cell growth which underlies many human diseases (e.g., cancer) [22].

In the eukaryotic cell cycle, the alternation between the S and the M phase as well as the balance of growth and division is governed by the activity of a family of cyclin-dependent protein kinases (CDK) [12]. Therefore, many computational models have been constructed to study the control system of CDK (e.g., in [1,12,17,18,21]). Some of these models are based on the deterministic approach which represents changes of species concentrations as continuous variables that evolve deterministically and continuously with respect to time. However, such approach does not capture the variability of cell size due to the fluctuation of some species which usually exist in low numbers of molecules [4]. Motivated by this argument, a number of stochastic models have been created and simulated using either a stochastic simulation algorithm (e.g., [12]) or by introducing noise to the model through Langevin equation [20]. However, the stochastic approach is computationally expensive, particularly when the model under study contains reactions of high rates or species of large numbers of molecules.

Similarly, the eukaryotic cell cycle model exhibits high reaction rates of some reactions while some other reactions have low rates. The latter types are responsible for the intrinsic noise due to molecular fluctuations [14]. The existence of reactions of different time scales (fast and slow) suggests the simulation using a hybrid approach. In [14] and [19] two different hybrid approaches are used to simulate the progression of cell cycle.

Correspondingly, Generalised Hybrid Petri Nets ($GHPN_{bio}$) have been introduced, in [10] and [11], to represent and simulate stiff biochemical networks where fast reactions are represented and simulated continuously, while slow reactions are carried out stochastically. $GHPN_{bio}$ provide rich modelling and simulation functionalities by combining all features of Continuous Petri Nets [2] and Extended Stochastic Petri Nets [15], including three types of deterministic transitions. Moreover, the partitioning of the reaction networks can either be done off-line before the simulation starts or on-line while the simulation is in progress. The implementation of $GHPN_{bio}$ is available as part of Snoopy [8] - a tool to design and animate or simulate hierarchical graphs, among them qualitative, stochastic, continuous and hybrid Petri nets. Indeed, the cell cycle model is an ideal case where the majority of $GHPN_{bio}$ features can be demonstrated.

In this paper we present another argument to motivate the hybrid simulation of the cell cycle control system. The cell cycle model contains some components which would be better represented as continuous processes (e.g., volume growth), while other reactions of low rates are vital to represent them as stochastic processes. For instance, Mura and Csikasz-Nagy constructed in [17] a stochastic version of the model in [1] using stochastic Petri nets. However, they got stuck with the problem of representing cell growth processes which evolve continuously and exponentially with respect to time using stochastic Petri net primitives. Indeed cell growth is a typical example where continuous transitions could be used. Moreover, our proposed model is graphically and intuitively represented in terms of Petri nets.

The paper is organised as follows: we start off by a brief introduction of Generalised Hybrid Petri Nets. After that, some related work is pinpointed. Next, we present our hybrid Petri net model of the eukaryotic cell cycle and describe in detail some of its key modelling components. In Section 5 we show the simulation results produced by Snoopy's hybrid simulation engine. Finally, we sum up with conclusions and outlook.

2 Related Work

Mura and Csikasz-Nagy constructed in [17] a stochastic Petri net model based on the work of [1] to study the effect of noise on cell cycle progression. However, some components could not intuitively be represented using SPN primitives only (e.g., cell growth). Moreover, their model is based on phenomenological rate laws (e.g., Michaelis-Menten) which do not work well with stochastic simulation algorithms [12]. Sabouri-Ghomi et al. [18], and Kar et al. [12], asserted that applying Gillespie's stochastic simulation algorithm [3] directly to phenomenological rate laws might produce incorrect results. Therefore, they unpacked the phenomenological deterministic model of Tyson-Novak [21] in terms of elementary mass action kinetics. The Tyson-Novak model is based on a bistable switch between the complex CycB-Cdk1 (denoted by variable X) and the complex Cdh1-APC (denoted by the variable Y). CycB-Cdk1 phosphorylates Cdh1-APC and free Cdh1-APC catalyses the degradation of CycB-Cdk1. To model a complete cell cycle, Kar et al. [12] unpacked the effect of Cdc20 and Cdc14 which are lumped in the variable Z in the Tyson-Novak model. High activity of CycB-Cdk1 promotes the synthesis of Cdc20 which activates Cdc14. Finally the dephosphorylated Cdc14 activates Cdh1-APC. The Kar et al. model accounts for both intrinsic and extrinsic noises. Intrinsic noise is due to the fluctuation of species of low numbers of molecules, while extrinsic noise is due to the unequal division of the cell between the two daughter cells [12].

In [19], a hybrid model which combines ordinary differential equations (ODEs) and discrete Boolean networks has been constructed to adapt quantitative as well as qualitative parts in the same model. The latter approach requires less knowledge about realistic kinetic rate constants. Liu et al. [14] simulate the stochastic model of [12] by statically partition the model reactions into slow and fast ones. However, as they have reported in their paper, the resulting hybrid model requires more simulation time than the original stochastic one. Therefore, they have (re)packed fast reactions in terms of phenomenological rate lows. The model presented in this paper differs from the Liu et al. one in the intuitive graphical representation and execution time of the hybrid model.

In this paper a hybrid Petri net model of the eukaryotic cell cycle is presented. The model is hybrid in the sense that it combines continuous, stochastic and immediate transitions to represent deterministic, stochastic and control components. Our main goal is to show how such class of models are intuitively represented and executed using hybrid Petri net primitives. Using Snoopy's simulator, it can be simulated either deterministically, stochastically or in a hybrid way.

3 Generalised Hybrid Petri Nets

To model stiff biochemical networks, $GHPN_{bio}$ [11] combine both stochastic and continuous elements in one and the same model. Indeed, continuous and stochastic Petri nets complement each other. The fluctuation and discreteness can be conveniently modelled using stochastic simulation and at the same time, the computationally expensive parts can be simulated deterministically using ODE solvers. Modelling and simulation of stiff biochemical networks are outstanding functionalities that $GHPN_{bio}$ provide for systems biology.

Generally speaking, biochemical systems can involve reactions from more than one type of biological networks, for example gene regulation, metabolic pathways or signal transduction pathways. Incorporating reactions which belong to distinct (biological) networks, tends to result into stiff systems. This follows from the fact that gene regulation networks species may contain a few number of molecules, while metabolic networks species may contain a large number of molecules [13].

In the rest of this section, we will give a brief introduction of $GHPN_{bio}$ in terms of the graphical representation of its elements as well as the firing rules and connectivity between continuous and stochastic net parts.

3.1 Elements

The $GHPN_{bio}$ elements are classified into three categories: places, transitions and arcs.

$GHPN_{bio}$ offer two types of places: discrete and continuous. Discrete places (single line circle) hold non-negative integer numbers which represent e.g., the number of molecules of a given species (tokens in Petri net notions). On the other hand, continuous places - which are represented by shaded line circle - hold non-negative real numbers which represent the concentration of a certain species.

Furthermore, $GHPN_{bio}$ offer five transition types: stochastic, immediate, deterministically delayed, scheduled, and continuous transitions [7]. Stochastic transitions which are drawn in Snoopy as a square, fire with an exponentially distributed random delay. The user can specify a set of firing rate functions, which determine the random firing delay. The transitions' pre-places can be used to define the firing rate functions of stochastic transitions. Immediate transitions (black bar) fire with zero delay, and have always highest priority in the case of conflicts with other transitions. They may carry weights which specify the relative firing frequency in the case of conflicts between immediate transitions. Deterministically delayed transitions (represented as black squares) fire after a specified constant time delay. Scheduled transitions (grey squares) fire at user-specified absolute time points. Continuous transitions (shaded line square) fire continuously in the same way like in continuous Petri nets. Their semantics are governed by ODEs which define the change in the transitions' pre- and post-places. More details about the biochemical interpretation of deterministically delayed, scheduled, and immediate transitions can be found in [9] and [15]. To

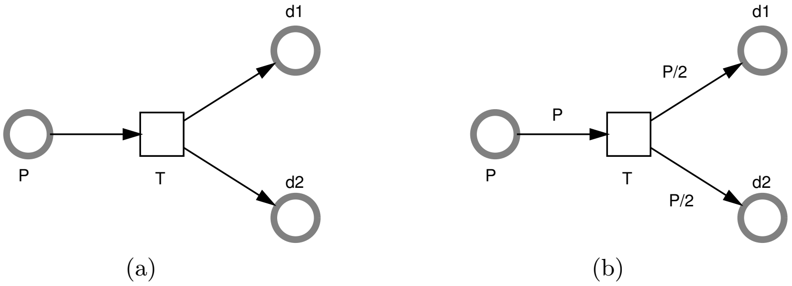


Fig. 1. Self-modifying weight illustrated by a simple biological example. (a) cell division cannot be modelled (b) cell division can intuitively be modelled.

simplify the presentation, we occasionally refer to stochastic, immediate, deterministically delayed or scheduled transitions as discrete transitions.

The connection between those two types of nodes (places and transitions) takes place using a set of different arcs (edges). $GHPN_{bio}$ offer six types of arcs: standard, inhibitor, read, equal, reset and modifier arcs. Standard arcs connect transitions with places or vice versa. They can be discrete, i.e., carry non-negative integer-valued weights (stoichiometry in the biochemical context), or continuous i.e., carry non-negative real-valued weights. In addition to their influence on the enabling of transitions, they affect also the place marking when a transition fires by adding (removing) tokens from the transition's post-places (pre-places). For more details see [10].

To support the special modelling requirements of some biological models (e.g., cell cycle model), arc weights are allowed to be a pre-place of a transition [23] or even a function which is defined in terms of the transition's pre-places [16].

Consider the following simple biological example. When a cell divides the mass between two daughter cells, each daughter takes approximately half of the mass. This example cannot be modelled using standard Petri nets as shown in Figure 1a. In Figure 1b, Using self-modifying weight; the ongoing arc of the transition "t" has weight equal to the marking of the place "P", while each of the two outgoing arcs has weight equal to the half marking of place "P".

Motivated by the case study of this paper, self-modifying weights are introduced to all arc types supported by Snoopy (standard, read, inhibitor, and equal arc). For more detail see Section 4.2.

Extended arcs like inhibitor, read, equal, reset, and modifier arcs can only be used to connect places to transitions, but not vice versa. A transition connected with an inhibitor arc is enabled if the marking of the pre-place is less than the arc weight. Contrary, a transition connected with a read arc is enabled if the marking of the pre-place is greater than or equal to the arc weight. Similarly, a transition connected using an equal arc is enabled if the marking of the pre-place is equal to the arc weight.

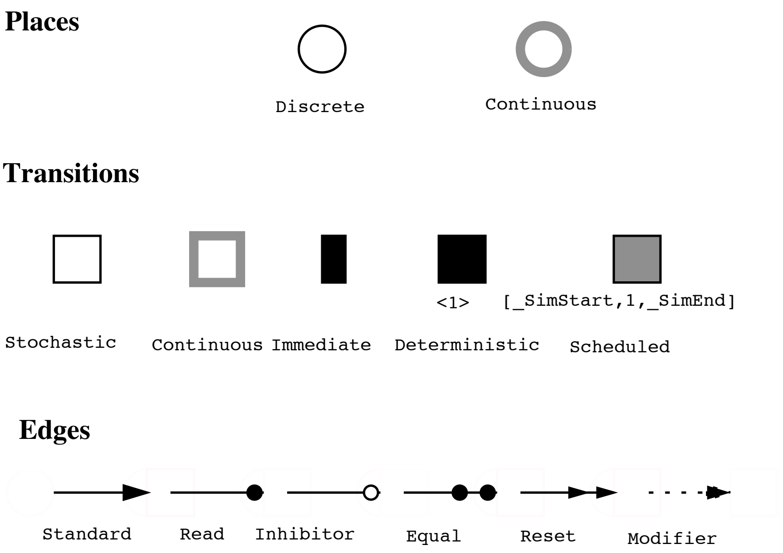


Fig. 2. Graphical representation of the $GHPN_{bio}$ elements. Places are classified as discrete and continuous, transitions as continuous, stochastic, immediate, deterministically delayed, and scheduled, and edges as standard, inhibitor, read, equal, reset, and modifier.

The other two remaining arcs do not affect the enabling of transitions. A reset arc is used to reset a place marking to zero when the corresponding transition fires. Modifier arcs permit to include any place in the transitions' rate functions and simultaneously preserve the net structure restriction.

The connection rules and their underlying formal semantics are discussed in more details below. Figure 2 provides a graphical illustration of all elements. Although this graphical notation is the default one, they can be easily customised using the Petri nets editing tool, Snoopy.

3.2 Connection Rules

A critical question arises when considering the combination of discrete and continuous elements: how are these two different parts connected with each other? Figure 3 provides a graphical illustration of how the connection between different elements of $GHPN_{bio}$ takes place.

Firstly, we will consider the connection between continuous transitions and the other elements of $GHPN_{bio}$. Continuous transitions can be connected with continuous places in both directions using continuous arcs (i.e., arc with real-valued weight). This means that continuous places can be pre- or post-places of continuous transitions. These connections typically represent deterministic biological interactions.

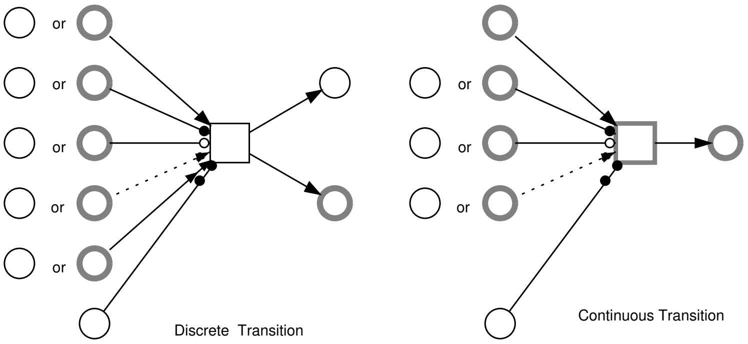


Fig. 3. Possible connections between $GHPN_{bio}$ elements. The restrictions are: discrete places can not be connected with continuous transitions using standard arcs, continuous places can not be tested with equal arcs, and continuous transitions can not use reset arcs.

Continuous transitions can also be connected with discrete places, but only by one of the extended arcs (inhibitor, read, equal, and modifier). This type of connection allows a link between discrete and continuous parts of the biochemical model.

Discrete places are not allowed to be connected with continuous transitions using standard arcs, because the firing of continuous transitions is governed by ODEs which require real values in the pre- and post-places. Hence, this cannot take place in the discrete world.

Secondly, discrete transitions can be connected with discrete or continuous places in both directions using standard arcs. However, the arc's weight needs to be considered. The connection between discrete transitions and discrete places takes place using arcs with non-negative integer numbers, while the connection between continuous places and discrete transitions is weighted by non-negative real numbers. The general rule to determine the weight type of arcs is the type of the connected place.

4 The Model

Figure 4 shows the hybrid Petri net model based on the previous one introduced by Kar et al. in [12]. Proteins, genes and mRNAs are represented by places. Transitions represent reactions. We use the same kinetic parameters and initial values. For the sake of compactness will not repeat them again here. Initial markings are shown inside the places. Moreover, we use Snoopy's logical node features to simplify connections between different nodes. For example, place X and Y are involved in many reactions which decreases the network's readability. We repeat them multiple times with same names to keep the model understandable (logical places). Likewise the transition "divide" (logical transitions). Furthermore, the increase of cell volume size is intuitively represented using a continuous transition with a rate $\mu \cdot V$, where μ is the growth factor and V is the cellular volume.

The model contains three different transition types: continuous, stochastic, and immediate. Continuous transitions simulate the corresponding reactions deterministically, while stochastic transitions carry them out stochastically. The latter transitions are responsible for molecular fluctuations. Immediate transitions monitor the model evolution and perform the division when the free number of molecules of Cdh1_APC reaches a certain threshold ($\hat{Y} = Y + YX + XY$).

In the sequel we present in more detail some of the model's key components and the corresponding $GHPN_{bio}$ representations.

4.1 Decision to Perform Division

When the number of molecules of $\hat{Y}$ becomes greater than a certain threshold (in our case 1200), the cell can divide the mass and other components between the two daughter cells. In Figure 5, this process is represented by an immediate transition "check" with the weight $\hat{Y} > threshold$. Recall that immediate transition weights determine the firing frequencies of immediate transitions in case of conflicts. A weight of zero means that a transition cannot fire at all. Therefore, when the transition "check" has weight greater than zero, it adds a token to the place "ready_to_divide" which signals the transition "divide" to carry out the division. To give the transition "divide" a chance to fire before re-checking the value of $\hat{Y}$, an inhibitor arc is used to constrain this case.

An interesting characteristic of the model is the division process. Although the division can take place when the value of $\hat{Y}$ is greater than a certain threshold, it does not do that all the times. For example, at the beginning of the simulation, the initial value of $\hat{Y}$ satisfies the dividing criterion. However; the cell should not divide because it is still at G1 phase which means that it has to replicate before it can divide. We model these cases by adding a new immediate transition which detects the critical value of $\hat{Y}$, before checking for division. Therefore the transition "critical" monitors the value of $\hat{Y}$. When the value of $\hat{Y}$ goes below a certain threshold, it enables the division process.

4.2 Cell Division and Self-modifying Weights

When a cell divides, it divides all of its components more-or-less evenly between two daughter cells. This is another ideal case to demonstrate self-modifying weights [23]. In Figure 5, when the transition "divide" fires, it removes all of the current marking of the place V and adds $V/2$ to it. To permit uneven division of the cell volume and other components, arc weights can be a function which operates on the current place marking [16]. However, we restrict the places used in arc weights to the transitions' pre-places to maintain the Petri net structure.

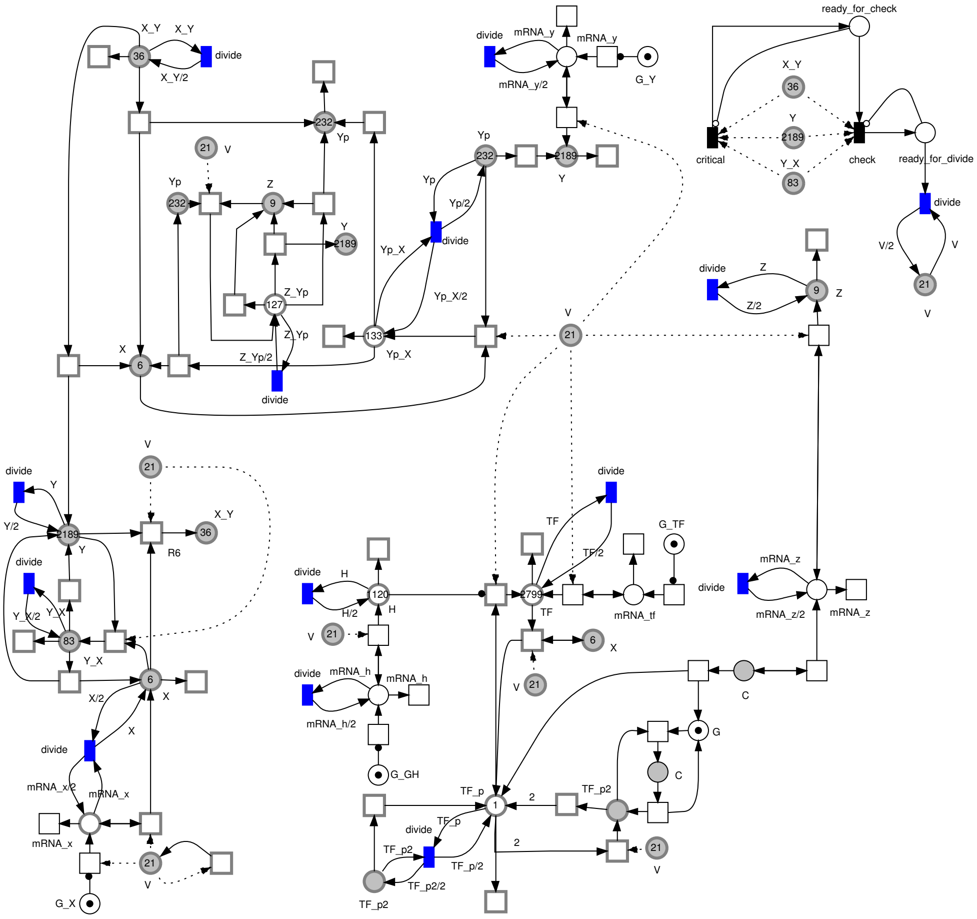


Fig. 4. A Generalised Hybrid Petri Nets representation of the eukaryotic cell cycle. The model employs different types of transitions: continuous, stochastic and immediate. All reactions affecting m_RNAs are represented and simulated stochastically. Repetitive nodes (places and transitions) with same names are logical nodes. When the transition "divide" fires, it divides the current place marking more or less equally. The type of division (equal, or unequal) depends on the outgoing arc weight and its effect takes place using self-modifying weight.

4.3 Transition Partitioning

The model in Figure 4 contains transitions which fire at different rates. For instance, transition "R3", as illustrated in Figure 6a, fires more frequently than "R1". Slow transitions should be simulated stochastically to account for molecular fluctuations, while fast transitions need to be simulated continuously to

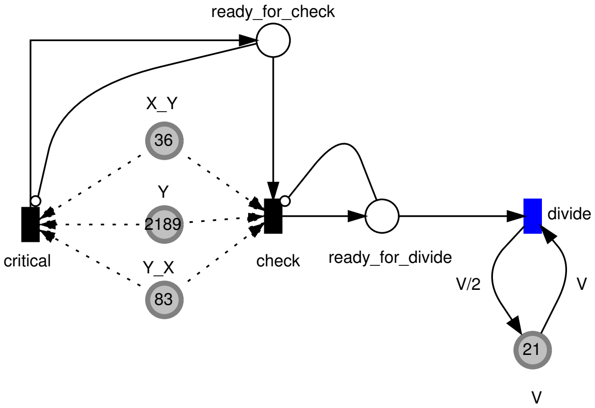


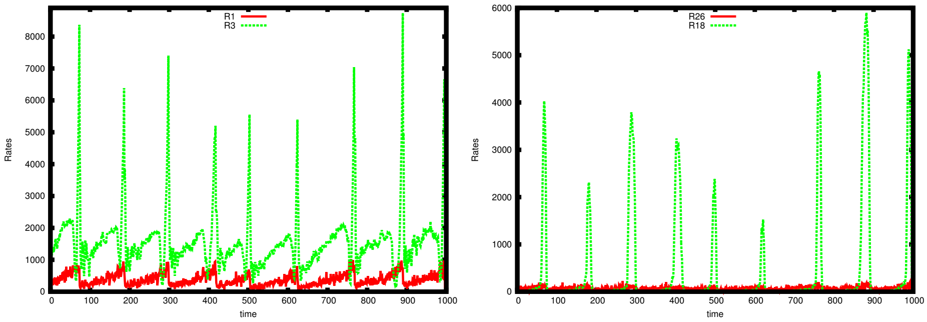
Fig. 5. A sub-net for modelling the decision of the division process. The transition "critical" monitors the value of $\hat{Y}$ and adds a token to "ready_for_check" when $\hat{Y} < 300$. Later, when the value of $\hat{Y}$ increases and becomes greater than a threshold (1200), the transition "check" fires and adds a token to "ready_for_divide" which signals the transition "divide" to perform the division. Inhibitor arcs are used as a check point for the sequence of events: critical → check → divide.

increase the numerical efficiency. Indeed, the latter types consume the majority of computational resources.

In this model, transitions are partitioned statically before the simulation starts. The transition type is decided by executing a single run and analyse the results as in Figure 6. Increasing (decreasing) the accuracy of the model results involves converting more continuous (stochastic) transitions into stochastic (continuous) ones. Similarly, controlling the speed of the model simulation will require the opposite procedure.

Another approach to do the partitioning is to perform it dynamically during the simulation. Using this technique, a transition changes its type from stochastic to continuous or vice versa according to the current firing rate. $GHPN_{bio}$ provide the user with a trade-off between efficiency and accuracy by permitting the user to specify two thresholds: $a_{0_{min}}$ and $a_{0_{max}}$, the minimum and maximum cumulative propensity, respectively. Moreover, two other thresholds are required to perform dynamic partitioning: the place marking threshold and the transition rate threshold. The former is used to ensure that species concentrations are large enough to be simulated continuously, while the latter is used to partition transitions into fast and slow based on their rates. For a transition to be simulated continuously its rate has to exceed the rate threshold and the marking of all its pre-place must be greater than the marking threshold.

Nevertheless, in both cases cell growth has to be represented and simulated continuously. Using off-line partitioning, this can be easily told to the simulator by drawing a continuous transition. However, in the case of dynamic partitioning;



(a) (b)

Fig. 6. Example of different transition firing rates. (a) transition "R3: X+Y → Y_X" fires more frequently than transition "R1: mRNA_x → mRNA_x+X" and (b) transition "R18: H → H+TF" fires much more than "R26: mRNA_tf→ mRNA_tf +TF".

the transition rate threshold should be set less than the expected rate of cell growth.

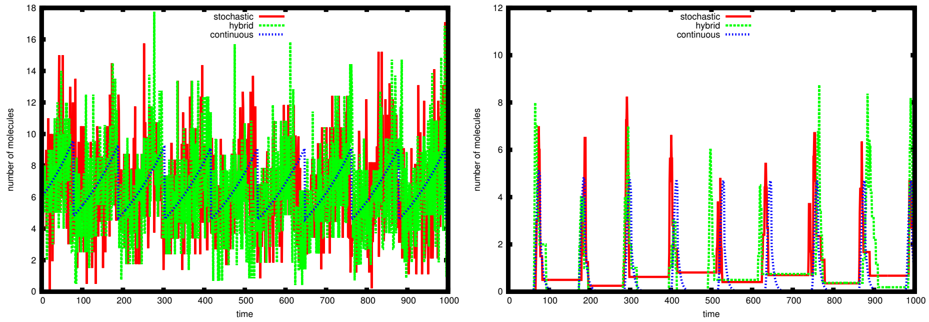
5 Simulation Results

In this section, we compare the simulation results of the following scenarios: when all of the reactions are simulated stochastically, when reactions related only to mRNAs are simulated stochastically and when all reactions are simulated continuously. In all cases cell growth is simulated using a continuous transition. Figure 7 shows the results of the three approaches using species with low numbers of molecules (mRNA_x and mRNA_z). Since reactions related to mRNAs are simulated stochastically in hybrid and stochastic simulations, their results are close to each other.

Figure 8 shows time course simulation results of proteins X and Y. In hybrid and stochastic simulations, X and Y are affected with fluctuations of mRNAs. while in continuous one there is no such effect.

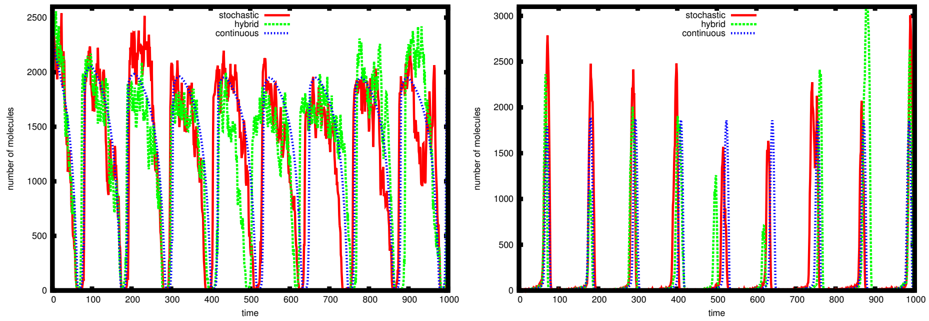
Figure 9 compares continuous and hybrid simulation results of the volume size (V). Using continuous simulation, cell divides all the time equal and the model produces no variability in its volume size. The hybrid simulation shows variability in the volume size because species of low numbers of molecules (e.g., mRNAs) are simulated stochastically which account for the molecular fluctuations and therefore, they are responsible for the intrinsic noise [12].

As a conclusion, the hybrid simulation approach can reproduce the results of the stochastic approach. However, substantially amount of simulation time could be saved. Fortunately, the resulting system of ordinary differential equations (ODEs) of this model is not stiff (for more details, see [5] and [6]). Therefore, an explicit ODE solver can be used to increase the performance of the hybrid simulation engine in connection with the stochastic simulation algorithms (SSA).



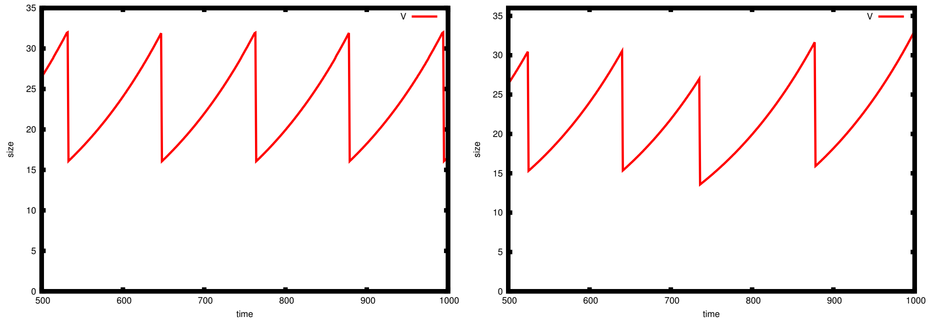
(a) (b)

Fig. 7. Time course result of the model in Figure 4 using Snoopy simulator (a) mRNA_x and (b) mRNA_z.



(a) (b)

Fig. 8. Time course result of species with large number of molecules; (a) Y and (b) X.



(a) (b)

Fig. 9. Continuous and hybrid simulation results for the cellular volume (V); (a) continuous result and (b) hybrid simulation result.

6 Conclusions and Outlook

In this paper we have presented a hybrid Petri net model of the eukaryotic cell cycle. The model can be executed using either continuous, stochastic or hybrid simulators. It employs continuous, stochastic and immediate transitions to intuitively represent the entire model logic.

The model is implemented using Snoopy which is available free of charge at `http://www-dssz.informatik.tu-cottbus.de/snoopy.html`.

Self-modifying weight is a new added feature to Snoopy which is currently not available in the official Snoopy release.

From the simulation results we notice that hybrid simulation produces results close to the stochastic one while simulation efficiency could be preserved. Indeed, the reactions of this model could easily be separated into slow and fast reactions, which makes it an ideal case study for hybrid simulation algorithms.

Self-modifying arcs are of paramount importance to model such biological cases since they provide a direct tool to program some biological phenomenon (e.g., cell division). Therefore, we intend to add more functionalities into this direction to permit more user-defined operators depending on transition's preplaces.

So far the partitioning of the reactions into stochastic and deterministic ones is carried out using a heuristic approach (see Section 4.3). However, as it has been risen during the review process; a better justification for the partitioning could be performed. For instance, the fast processes can be regarded as processes that could be described by quasi (or pseudo)-steady state approach, assuming that they reach equilibrium rapidly. In other words, they could be better described by setting the corresponding ODE to zero and solving for the fast variables. In contrast, continuous dynamics could be seen as more appropriate for abundant molecules whose concentration display a small coefficient of variation, and stochastic dynamics for those molecules evolving at low copy number.

The presented model could be viewed as a sub-net in a bigger network of reactions (e.g., modelling budding yeast cell cycle or Fission yeast cells). Snoopy's hierarchical nodes might simplify such task as they provide an easy tool to insert a sub net in a bigger one.

7 Acknowledgements

The authors acknowledge the comments of Monika Heiner during the implementation of $GHPN_{bio}$ and the preparation of this manuscript.

References

1. Chen, K., Calzone, L., Csikasz-Nagy, A., Cross, F., Novak, B., Tyson, J.: Integrative analysis of cell cycle control in budding yeast. Mol. Biol. Cel 5(8), 3841–3862 (2004)

2. Gilbert, D., Heiner, M.: From Petri Nets to Differential Equations - An Integrative Approach for Biochemical Network Analysis. In: Donatelli, S., Thiagarajan, P. (eds.) Petri Nets and Other Models of Concurrency - ICATPN 2006, Lecture Notes in Computer Science, vol. 4024, pp. 181–200. Springer Berlin / Heidelberg (2006)
3. Gillespie, D.: A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. J. Comput. Phys. 22(4), 403 – 434 (1976)
4. Gillespie, D.: Stochastic simulation of chemical kinetics. Annual review of physical chemistry 58(1), 35–55 (2007)
5. Hairer, E., Nørsett, S., Wanner, G.: Solving Ordinary Differential Equations I: Nonstiff Problems, Springer Series in Comput. Mathematics, vol. 8. Springer-Verlag, second edn. (1993)
6. Hairer, E., Wanner, G.: Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems, Springer Series in Comput. Mathematics, vol. 14. Springer-Verlag, second edn. (1996)
7. Heiner, M., Gilbert, D., Donaldson, R.: Petri nets for systems and synthetic biology. In: Formal Methods for Computational Systems Biology, Lecture Notes in Computer Science, vol. 5016, pp. 215–264. Springer Berlin / Heidelberg (2008)
8. Heiner, M., Herajy, M., Liu, F., Rohr, C., Schwarick, M.: Snoopy – a unifying Petri net tool. In: proc. of 33rd International Conference on Application and Theory of Petri Nets and Concurrency (2012)
9. Heiner, M., Lehrack, S., Gilbert, D., Marwan, W.: Extended stochastic Petri nets for model-based design of wetlab experiments. In: Transactions on Computational Systems Biology XI, pp. 138–163. Springer, Berlin, Heidelberg (2009)
10. Herajy, M., Heiner, M.: Hybrid representation and simulation of stiff biochemical networks. Nonlinear Analysis: Hybrid Systems (2012), to appear
11. Herajy, M., Heiner, M.: Hybrid representation and simulation of stiff biochemical networks through generalised hybrid Petri nets. Tech. Rep. 02/2011, Brandenburg University of Technology Cottbus, Dept. of CS (2011)
12. Kar, S., Baumann, W.T., Paul, M.R., Tyson, J.J.: Exploring the roles of noise in the eukaryotic cell cycle. Proceedings of the National Academy of Sciences of the United States of America 106(16), 6471–6476 (2009)
13. Kiehl, T., Mattheyses, R., Simmons, M.: Hybrid simulation of cellular behavior. Bioinformatics 20, 316–322 (2004)
14. Liu, Z., Pu, Y., Li, F., Shaffer, C., Hoops, S., Tyson, J., Cao, Y.: Hybrid modeling and simulation of stochastic effects on progression through the eukaryotic cell cycle. J. Chem. Phys 136(34105) (2012)
15. Marwan, W., Rohr, C., Heiner, M.: Petri nets in Snoopy: A unifying framework for the graphical display, computational modelling, and simulation of bacterial regulatory networks, Methods in Molecular Biology, vol. 804, chap. 21, pp. 409–437. Humana Press (2012)
16. Matsuno, H., Tanaka, Y., Aoshima, H., Doi, A., Matsui, M., Miyano, S.: Biopathways representation and simulation on hybrid functional Petri net. In silico biology 3(3) (2003)
17. Mura, I., Csikász-Nagy, A.: Stochastic Petri net extension of a yeast cell cycle model. Journal of Theoretical Biology 254(4), 850 – 860 (2008)
18. Sabouri-Ghomi, M., Ciliberto, A., Kar, S., Novak, B., Tyson, J.J.: Antagonism and bistability in protein interaction networks. Journal of Theoretical Biology 250(1), 209 – 218 (2008)
19. Singhania, R., Sramkoski, R.M., Jacobberger, J.W., Tyson, J.J.: A hybrid model of mammalian cell cycle regulation. PLoS Comput Biol 7(2), e1001077 (02 2011)

20. Steuer, R.: Effects of stochasticity in models of the cell cycle: from quantized cycle times to noise-induced oscillations. Journal of Theoretical Biology 228(3), 293 – 301 (2004)
21. Tyson, J., Novak, B.: Regulation of the eukaryotic cell cycle: Molecular antagonism, hysteresis, and irreversible transitions. Journal of Theoretical Biology 210(2), 249 – 263 (2001)
22. Tyson, J., Novak, B.: A Systems Biology View of the Cell Cycle Control Mechanisms. Elsevier, San Diego, CA, (2011)
23. Valk, R.: Self-modifying nets, a natural extension of Petri nets. In: Proceedings of the Fifth Colloquium on Automata, Languages and Programming. pp. 464–476. Springer-Verlag, London, UK, (1978)

Modelling Atopic Dermatitis using Petri Nets

Marta E Polak

Clinical and Experimental Sciences, University of Southampton, Faculty of Medicine,
Mailpoint 825, Level F, South Block, Sir Henry Wellcome Laboratories, Southampton
General Hospital, Southampton, SO16 6YD, United Kingdom,
`m.e.polak@soton.ac.uk`

1 Background

Atopic dermatitis (AD) is a disorder of inflammation in the skin and is strongly associated with other inflammatory epithelial atopic conditions (asthma, allergic rhinitis and food allergy). The pathogenesis of AD results from complex interactions between susceptibility genes encoding skin barrier molecules and markers of the inflammatory response, host environments, infectious agents, and specific immunologic responses [1-3]. Despite the long-held knowledge about the multiple immunological processes involved in AD pathogenesis (including role of dendritic cells, kerationocytes and T lymphocytes, interactions between different cell types, effect of allergen exposure, trauma and infection with Staphylococcus Aureus on the AD exacerbations) the precise reason for the bias towards a cutaneous Th2 response in AE still remains obscure. It seems likely that understanding of the complex cross-talk between different components of the cutaneous immune network is fundamental to understanding of the Th2 polarised inflammation in AD.

To gain in-depth understanding and be able to predict the behavior of the system, we sought to create an integrated in silico model compositionally from relatively simple individual components that are amenable to detailed mathematical analysis. The simple, initial model described in this contribution will serve as a high level architectural specification of AD skin and immune system interactions. In the future we plan to harness data from microarray analysis in order to refine the individual model components by incorporating the reconstructed signal transduction pathways within cellular components.

2 Model description

Petri nets have been widely used in modelling biochemical, genetic, signaling and metabolic networks [7-9]. We however feel that the modelling of AD requires a multistep approach, embedding the investigations of the internal genetic or metabolic signaling cascades within the detailed analysis of the interactions between the cellular components of the cutaneous immune system. The preliminary model of cell-to-cell signaling in AD exposed to allergen and infected with S.A. incorporates three cell types (KC, DC and T cells), where KC and T cells are simplistically represented as places. The signaling within DC is represented by two independent pathways, TNF-α–NFκB-dependent IL-12p70 initiating Th1 responses and TSLP-STAT3-OX-40L pathway inducing Th2 responses. The signaling proteins are represented as places and are connected by activating (black arrows) and inhibiting (red diamond arrows) edges.

The model was constructed using yEd graphical software, and BioLayout Express3D [10] was used to simulate the signal flow allowing investigating the dynamic behaviour of the network dependently on presence/absence of inhibition edges, conservation or consumption of the tokens at the transitions, and the value of initial stimulation given by the number of tokens. To represent healthy skin and AD skin firings via TNF-α or TSLP were inhibited, respectively, and the simulation for altered number of tokens at each starting place was repeated as for the initial state of the network. Similarly, the effect of S.A. infection (0-1000 tokens) and combined effect of Der-p-1 and S.A. in exclusive Th1/Th2 or both pathways enabled models was investigated.

As assumed, stimulation with der-p-1 in AD model induced only Th2 responses, which were greatly enhanced in the presence of Staphylococcal infection. While in normal skin model infection with SA led to induction of both Th1 and Th2 responses, stimulation of AD model resulted in Th2 skewing. Induction of Th2 polarisation depended on blocking of the Th1 signaling, rather than enhancing Th2 signaling.

Acknowledgements

I am very grateful to Dr. Mike Ardern-Jones and Dr. Pawel Sobocinski for critical review of the manuscript. I would also like to thank Prof. Tom Freeman for advice on Petri nets and model construction. This study has been funded by British Skin Foundation.

References

1. Boguniewicz M, Leung DY. Atopic dermatitis: a disease of altered skin barrier and immune dysregulation. Immunol Rev. 242(1):233-46.(2011)
2. Irvine AD, McLean WH, Leung DY. Filaggrin mutations associated with skin and allergic diseases. N Engl J Med. 365(14):1315-27 (2011)
3. Hamid Q, Boguniewicz M, Leung DY. Differential in situ cytokine gene expression in acute versus chronic atopic dermatitis. J Clin Invest. 1994 Aug;94(2):870-6.
4. Niebuhr M, Werfel T. Innate immunity, allergy and atopic dermatitis. Curr Opin Allergy Clin Immunol. 10(5):463-8. (2010)
5. Zaba LC, Krueger JG, Lowes MA. Resident and "inflammatory" dendritic cells in human skin. J Invest Dermatol. 129(2):302-8 (2009)
6. Soumelis V, Reche PA, Kanzler H, Yuan W, Edward G, Homey B, Gilliet M, Ho S,Antonenko S, Lauerma A, Smith K, Gorman D, Zurawski S, Abrams J, Menon S,McClanahan T, de Waal-Malefyt Rd R, Bazan F, Kastelein RA, Liu YJ. Human epithelial cells trigger dendritic cell mediated allergic in-flammation by producing TSLP. Nat Immunol. 3(7):673-80 (2002)

7. Ruths D, Muller M, Tseng JT, Nakhleh L, Ram PT. The signaling petri net-based simulator: a non-parametric strategy for characterizing the dynamics of cell-specific signaling networks. PLoS Comput Biol. 29;4(2):e1000005 (2008)
8. Heiner M, Koch I, Will J. Model validation of biological pathways using Petri nets–demonstrated for apoptosis. Biosystems. 75(1-3):15-28 (2004)
9. Moore JH, Boczko EM, Summar ML. Connecting the dots between genes,biochemistry, and disease susceptibility: systems biology modeling in hu-man genetics. Mol Genet Metab. 84(2):104-11. (2005)
10. Theocharidis A, van Dongen S, Enright AJ, Freeman TC. Network visualization and analysis of gene expression data using BioLayout Express(3D). Nat Protoc. 4(10):1535-50 (2009)

Proc. BioPPN 2012, a satellite event of PETRI NET 2012

PNlib – A Modelica Library for Simulation of Biological Systems Based on Extended Hybrid Petri Nets

Sabrina Proß[1], Sebastian Jan Janowski[2], Bernhard Bachmann[1],
Christian Kaltschmidt[3], and Barbara Kaltschmidt[3]

[1]University of Applied Sciences Bielefeld, Department of Engineering and Mathematics,
Bielefeld, Germany
{sabrina.pross, bernhard.bachmann}@fh-bielfeld.de
[2]Bielefeld University, Faculty of Technology, Bielefeld, Germany
{sebastian.janowski}@uni-bielefeld.de
[3]Bielefeld University, Faculty of Biology, Bielefeld, Germany
{c.kaltschmidt,barbara.kaltschmidt}@uni-bielefeld.de

Abstract. We present a new Petri net simulation environment to enable the processing of experimental data to gain usable new insights about biological systems. Therefore, a powerful mathematical modeling concept – xHPNbio (extended Hybrid Petri Nets for biological applications) – has been defined which is properly adapted to the demands of biological processes. This specification is used for the PNlib (Petri Net library), realized by means of the object-oriented modeling language Modelica, which can be easily integrated for simulation processing in any other network modeling tool, as described in the last part of this paper. There, we briefly describe VANESA, a user-friendly biological network-modeling tool that uses the PNlib and the xHPNbio formalism for the simulation of biological networks.

Keywords: hybrid systems, Petri nets, biological processes, Modelica, xHPN, xHPNbio, VANESA, PNlib.

1 Introduction

Modern computer techniques and large memory capacities make it possible to produce an enormous amount of molecular data stored in huge databases. This data is indispensable for the scientific progress but does not necessarily lead to insight about the functionality of biological systems. To improve the understanding of molecular mechanisms, modern techniques focus on network analysis. The question which is posed here is, what model formalism is appropriate and which simulator? Numerous model formalisms have been proposed for modeling and simulation biological systems (see e.g. [1]). Generally, a distinction must be made between qualitative and quantitative approaches. Qualitative models represent only the fundamental compounds, their interaction mechanisms, and the relationships between them while quantitative models describe, in addition, the time-related changes of the components.

Furthermore, quantitative model formalisms can be divided into discrete and continuous approaches as well as deterministic and stochastic techniques.

In the recent years, Petri nets with their various extensions are becoming increasingly popular. They have been proven to be as universal graphical modeling concept for representing biological systems in nearly all degrees of abstraction. They support the qualitative modeling approach as well as the quantitative one. Furthermore, the biological processes can be modeled discretely as well as continuously and, in addition, discrete and continuous processes can also be combined within a Petri net model to so-called hybrid Petri nets first introduced by David and Alla (e.g. [2]). The Petri net formalism with all its extensions is so powerful that all other formalisms are included. Hence, only one formalism is needed regardless of the approach (qualitative vs. quantitative, discrete vs. continuous vs. hybrid, deterministic vs. stochastic) which is appropriate for the respective system. The Petri net formalism is easy to understand for researchers from different disciplines. It is such an ideal way for intuitive representing and communicating experimental data and new knowledge of molecular mechanisms. Besides, Petri nets allow hierarchical structuring of models and therefore offer the possibility of different detailed views for every observer of the model.

Despite several works and publications with Petri net approaches, there is a serious problem relating to the lacking unity of concepts, notations, and terminologies. Therefore, to show the research community the power of Petri nets, we have analyzed the demands for carefully modeling biological systems and specially developed a Petri net formalism which is called xHPNbio (extended Hybrid Petri Net for biological applications).

This Petri net concept is the specification of the new simulator based on the object-oriented modeling language Modelica. A user-friendly graphical model reconstruction in addition to a well-prepared visualization of simulation results is achieved by connection the new Petri net simulator to VANESA, a powerful and easy-to-use biological modeling tool [3]. This new approach is already in use in the area of dynamic system modeling for hypothesis generation and testing of reconstructed database and lab-validated biological networks.

2 Related Works

Reddy et al. proposed the application of Petri net formalism (introduced by Carl Adam Petri in 1962) for biological network modeling in order to represent and analyze metabolic pathways in a qualitative manner [4]. Thereby, places represent biological compounds such as metabolites, enzymes, and cofactors which are part of biochemical reactions. These biochemical reactions are modeled by transitions and their stoichiometry is represented by the arc weights. Besides, the tokens indicate the presence of compounds.

Moreover, Hofestädt and Thelen expanded the approach of Reddy by introducing functional Petri nets to enable quantitative modeling of biochemical networks [5]. Thereby, the arc weights are functions, which depend on concrete markings of places in order to model kinetic effects.

Due to the fact that a random behavior of molecular reactions at low concentrations has been observed in many experiments, Goss and Peccoud introduced stochastic Petri nets [6]. A stochastic transition does not fire instantaneously but rather with a time delay following an exponential distribution which may depend on the token numbers of the places.

A reasonable way for modeling concentrations of biological compounds is by places containing real token numbers instead of integers and transitions which fire as a continuous flow specified by an assigned speed. The transformation from the discrete to the continuous Petri net concept was first introduced by David and Alla in 1987 and they replaced the term token by mark because tokens relate mostly to integer quantities [7].

Furthermore, Alla and David and proposed combing the discrete and the continuous Petri net concept to so-called hybrid Petri nets [8]. A hybrid Petri net contains discrete places with integer tokens and discrete transitions with time delays as well as continuous places with non-negative real marks and continuous transitions with firing speeds .Matsuno et al. used this approach for modeling gene regulatory networks by discrete and continuous processes [9]. They improved this approach further by adding the properties of functional Petri nets to it so that the arcs as well as the speeds of the transitions are functions depending on the marks of the places [10]. In addition, they extended the hybrid functional Petri nets by two specific arcs, called test and inhibitor arcs [10], to accomplish the modeling of inhibition and activation mechanisms of biological reactions. Chen and Hofestädt as well as Doi et al. demonstrated the applicability of this approach by modeling molecular networks [11, 12]. Moreover, Nagasaki et al. extended the hybrid functional Petri nets further by types with which various data types can be regarded in order to model more complex biological processes which involve various kinds of biological information and data [13]. They called this approach hybrid functional Petri nets with extensions (HFPNe).

Despite these mentioned works and publications, there is a serious problem regarding the lacking unity of concepts, notations, and terminologies. The definition of Petri nets is not standardized; every author has his/her own definitions which are partly not precise enough, not common, or contradictory. Hence, to show the research community the power of Petri nets, they have to be defined precisely together with the corresponding processes, which are essential for the simulation. This has been done in this paper; based on the mentioned Petri net concepts, formalism has been developed which is able to represent nearly all kinds of biological processes. It is called xHPNbio (extended Hybrid Petri Nets for biological applications).

Two common tools are already available for modeling biological processes with the Petri net formalism. The first one is the commercial tool Cell Illustrator and the second one is the freely available tool Snoopy.

The Cell Illustrator is a commercial, widely-used tool available as a Java Web Start application that enables to draw, model, elucidate, and simulate complex biological processes and systems based on extended hybrid functional Petri nets [14]. Discrete and continuous processes can be connected to perform hybrid simulations. The drawback of the Cell Illustrator is that the simulation is like a “black box”. There is no information about how the Petri nets and the corresponding processes are defined which are necessary for modeling and simulation, e.g. how conflicts in Petri nets are resolved, how the hybrid simulation is performed, and which integrators are used. In addition, there is no possibility to adapt solver settings in order to achieve reliable simulation results.

Snoopy is a freely available unifying Petri net framework to investigate biomolecular networks [15]. A Petri net can be modeled time-free (qualitative model) or its behavior can be associated with time (quantitative model) such as stochastic, continuous, and hybrid Petri nets; thereby, different models are convertible into each other. It is also possible to structure the models hierarchically in order to manage complex networks. The drawback of Snoopy is that a continuous Petri net is interpreted as a graphical representation of a system of ordinary differential equations. Hence, the general Petri net property of non-negative marks cannot be held during simulation. Additionally, conflict situations of hybrid Petri nets are trapped not completely and, thus, negative markings can occur. Furthermore, places cannot be provided with capacities and no functions can be assigned to arcs in hybrid Petri nets.

Hence, these problems led to the development of a new Petri net simulation environment specified by the established xHPNbio formalism. The xHPNbio elements are modeled object-oriented which allows an easy way to maintain, extend, and modify them. Furthermore, the hybrid simulation is performed by an appropriate Modelica-tool. With this several solver settings can be adapted in order to achieve reliable simulation results. Moreover, the xHPNbio formalism is already integrated in VANESA, an easy-to-use biological modeling tool. Using VANESA scientists are able to reconstruct and simulate biological pathways either by drag-and-drop or by loading networks from databases and transforming in the appropriate xHPNbio formalism in one software application.

3 Extended Hybrid Petri Nets for Biological Applications (xHPNbio)

The xHPNbio formalism comprises three different processes, called transitions: discrete, stochastic, and continuous, two different states, called places: discrete and continuous, and four different arcs: normal, inhibition, test, and read arc.

Discrete places contain a non-negative integer quantity, called tokens or *marks* while continuous places contain a non-negative real quantity, called marks. These marks initiate transitions to fire according to specific conditions. These firings lead mostly to changes of the marks in the connected places.

Discrete transitions are provided with *delays* and *firing conditions* and fire first when the associated delay is passed and the conditions are fulfilled. These fixed delays can be replaced by exponentially distributed random values, then, the corresponding transition is called *stochastic transition*. Thereby, the characteristic parameter λ of the exponential distribution can depend functionally on the markings of several places (cp. [16]) and is recalculated at each point in time when the respective transition becomes active or when one or more markings of involved places change[1]. Based on the characteristic parameter, the next putative firing time $\tau = time + \mathrm{Exp}(\lambda)$ of the transition can be evaluated and it fires when this point in time is reached.

Both – discrete and stochastic transitions - fire by removing the arc weight from all input places and adding the arc weight to all output places. On the contrary, the firing of continuous transitions takes places as a continuous flow determined by the firing speed which can depend functionally on markings and/or time. Places and transitions are connected by "normal" arcs which are weighted by non-negative integer and real numbers, respectively. But also functions can be written at the arcs depending on the current markings of the places and/or time.

Places can also be connected to transitions by test, inhibition, and read arcs. Then their markings do not change during the firing process. In the case of test and inhibitor arcs, the markings are only read to influence the time of firing while read arcs only indicate the usage of the marking in the transition, e.g. for firing conditions or speed functions. If a place is connected to a transition by a test arc, the marking of the place must be greater than the arc weight to enable firing. If a place is connected to a transition by an inhibitor arc, the marking of the place must be less than the arc weight to enable firing. In both cases the markings of the places are not changed by firing. The same place can be connected to the same transition by a test and, in addition, by a normal arc as well as by an inhibitor and normal arc. These arcs are called *double arcs.*

It is important to mention that a discrete transition always fires in a discrete manner by removing and adding marks after a delay is passed regardless of whether a discrete or a continuous place is connected to it. However, a continuous transition always fires in a continuous flow so that a discrete place can only be connected to continuous transitions if it is input as well as output of the transition with arcs of the same weight. In this way, the continuous transition can only be influenced by the discrete place but the discrete marking cannot be changed by continuous firing. Hence, the conversion from discrete to continuous markings and vice versa is always performed by discrete transitions connected to continuous places.

A formal definition of an xHPNbio is given below. Therefore, at first the xHPN-formalism is introduced which is then expanded to xHPNbio by providing the Petri net elements with a biological meaning.

Definition 1. The tuple $(PD, PC, TD, TS, TC, F, G, \mathcal{T}, \mathcal{I}, \mathcal{R}, f, c_l, c_u, e, \wp, d, h, v, s, m_0)$ is a *xHPN* if

- $PD = \{pd_1, pd_2, \ldots, pd_{pd}\}$ is a finite set of discrete places,
- $PC = \{pc_1, pc_2, \ldots, pc_{pc}\}$ is a finite set of continuous places,
- $TD = \{td_1, td_2, \ldots, td_{td}\}$ is a finite set of discrete transitions,
- $TS = \{ts_1, ts_2, \ldots, ts_{ts}\}$ is a finite set of stochastic transitions,

---

[1] The involved places may change their markings only in a discrete manner. Continuous changes of involved places are not allowed because then the putative firing times have to be recalculated the whole time as the continuous change takes place.

- $TC = \{tc_1, tc_2, \dots, tc_{tc}\}$ is a finite set of continuous transitions,
- $PD, PC, TD,$ TS, and TC are pairwise disjoint,
- $F \subseteq (PD \times TD \cup PD \times TS \cup PD \times TC \cup PC \times TC \cup PC \times TD \cup PC \times TS)$ is a set of normal arcs from places to transitions, where $(p_i \to t_j)$ denotes the arc from place p_i to transition t_j,
- $G \subseteq (TD \times PD \cup TD \times PC \cup TS \times PD \cup TS \times PC \cup TC \times PC \cup TC \times PD)$ is set of normal arcs from transitions to places, where $(t_j \to p_i)$ denotes the arc from transition t_j to place p_i,
- $\mathcal{T} \subseteq (PD \times TD \cup PD \times TS \cup PD \times TC \cup PC \times TC \cup PC \times TD \cup PC \times TS)$ is a set of test arcs, where $(p_i \to t_j)_{\mathcal{T}}$ denotes the test arc from p_i to t_j,
- $\mathcal{J} \subseteq (PD \times TD \cup PD \times TS \cup PD \times TC \cup PC \times TC \cup PC \times TD \cup PC \times TS)$ is a set of inhibitor arcs, where $(p_i \to t_j)_{\mathcal{J}}$ denotes the inhibitor arc from p_i to t_j,
- $\mathcal{R} \subseteq (PD \times TD \cup PD \times TS \cup PD \times TC \cup PC \times TC \cup PC \times TD \cup PC \times TS)$ is a set of read arcs, where $(p_i \to t_j)_{\mathcal{R}}$ denotes the read arc from p_i to t_j,
- $f: (F \cup G \cup \mathcal{T} \cup \mathcal{J}, \mathrm{m}) \to \{\mathbb{N}_0: p_i \in PD, \mathbb{R}_{\geq 0}: p_i \in PC\}$ is an arc weight function which assigns every arc connected to a discrete place a non-negative integer and all others a non-negative real number depending on a concrete marking m, where $f(p_i \to t_j)$ denotes the weight of the arc from place p_i to transition t_j,
- if $p_i \in PD$, $t_j \in TC$ then $(p_i \to t_j) \in F$ if and only if $(t_j \to p_i) \in G$ and $f(p_i \to t_j) = f(t_j \to p_i)$,
- $c_l: \{PD \to \mathbb{N}_0, PC \to \mathbb{R}_{\geq 0}\}$ are the minimum capacities of the places,
- $c_u: \{PD \to \mathbb{N}_0, PC \to \mathbb{R}_{\geq 0}\}$ are the maximum capacities of the places,
- $e: (PD \cup PC) \to \{prio, prob\}$ are the resolution types of the places for type-1-conflicts either priority or probability resolution,
- $\wp: (F \cup G) \to \{\mathbb{N}: e(p_i) = prio \ \wedge t_j \in TD,\ (F \cup G) \to [0,1]: e(p_i) = prob \wedge t_j \in TD\}$ is an enabling function which assigns every arc connected to a discrete transition t_j either a priority or a probability according to the resolution type of the place p_i,
- if $e(p_i) = prio$ then $\wp(p_i \to t_k) \neq \wp(p_i \to t_l)\ \forall t_k, t_l \in TD_{out}(p_i)$ and $\wp(t_k \to p_i) \neq \wp(t_l \to p_i)\ \ \forall t_k, t_l \in TD_{in}(p_i)$, if $e(p_i) = prob$ then $\sum_{t_k \in TD_{out}(p_i)} \wp(p_i \to t_k) = 1$ and $\sum_{t_k \in TD_{in}(p_i)} \wp(t_k \to p_i) = 1$,
- $d: TD \to \mathbb{R}_{\geq 0}$ is a delay function which assigns every discrete transition a positive, real-valued delay,
- $h: (TS, m) \to \mathbb{R}_{\geq 0}$ is a hazard function which assigns every stochastic transition a positive, real-valued random delay depending on a concrete marking m,
- $v: (TC, m) \to \mathbb{R}_{\geq 0}$ is a maximum speed function which assigns every continuous transition a positive, real-valued maximum speed depending on a concrete marking m,
- $s: (TD \cup TS \cup TC, mv) \to \{true, false\}$ is a condition function which assigns every transition a condition depending on all possible model variables (mv) e.g. time,
- $m_0: \{PD \to \mathbb{N}_0, PC \to \mathbb{R}_{\geq 0}\}$ is the initial marking which must satisfy the condition $c_l(p_i) \leq m_0(p_i) \leq c_u(p_i)\ \forall\, p_i \in (PD \cup PC)$.

Definition 2. An *xHPNbio* is an xHPN (see Definition 1) with a concrete transformation of xHPN elements to biological ones. This transformation is summarized in the following table by mentioning also some examples of the biological meaning.

xHPN	Biological meaning
Places	*Biological compounds*
	metabolites, enzymes, substances, substrates, products, signals, genes, proteins, cells, complexes, activators, inhibitors, repressors, RNA
Transitions	*Biological processes*
	biochemical reactions, metabolic reactions, interactions, regulatory reactions, signal transduction reactions, chemical reactions, binding, phosphorylation
Marks	*Quantities of biological compounds*
	molecules, concentrations, cells
Normal Arcs	*Connections of biological compounds and processes*
Test arcs	*Activation of biological processes*
	transcription process, activation in gene regulation, enzyme activity, activation mechanisms
Inhibitor arcs	*Inhibition of biological processes*
	repression of gene regulation, inhibition mechanisms
Read arcs	*Needs for biological processes*
	catalysis
Arc weights	*Biological coefficients*
	stoichiometric coefficients, yield coefficients
Min/max. capacities	*Reasonable biological capacities*
	biological knowledge
Delays	*Duration of biological processes*
Hazard functions	*Random duration of biological processes*
	stochastic kinetics
Maximum speeds	*Rate of biological processes*
	kinetics effects/laws
xHPNbio	*Biological systems*
	metabolic networks, signal transduction networks, regulatory networks, chemical networks, cell cycle, cell communication, diseases, population dynamics, flux networks, cultivation processes

This xHPN formalism has been transformed to the modeling language Modelica (see section 4) to enable graphical modeling, hybrid simulation, and animation. The execution of a hybrid simulation requires the definition for activating and firing transitions as well as the resolution of possible conflicts.

A *discrete/stochastic transition* t_j in an xHPN is *active* if the markings of all input places $\left(P_{in}(t_j)\right)$ do not fall below the minimum capacities when the arc weights are removed, and the maximum capacities of all output places $(P_{out}(t_j))$ may not be exceeded when the arc weights are added. Additionally, the input places connected by test arcs must have more marks than the arc weights and the places connected by inhibitor arcs must have less marks than the arc weights; read arcs do not influence the activation of a transition.

However, the activation process of *continuous transitions* requires a differentiation between connected continuous and discrete places. A continuous transition t_j is active if all continuous input places $\left(PC_{in}(t_j)\right)$ have either a marking greater than their minimum capacities or they are fed by at least one input transition, i.e. the input speed I_i is not zero. Additionally, all continuous output places $\left(PC_{out}(t_j)\right)$ have either a marking less than their maximum capacities or they are emptied by at least one output transition, i.e. the output speed O_i is not zero. The connected discrete places have to fulfill the same conditions as mentioned above for activating a discrete transition. In addition, the markings of input places connected by test arcs have to be greater than the arc weights and markings of places connected by inhibitor arcs have to be less than the arc weights.

Definition 3. The tuple $(PD, PC, TD, TS, TC, F, G, \mathcal{T}, \mathcal{I}, \mathcal{R}, f, c_l, c_u, e, \wp, d, h, v, s, m_0)$ is an xHPN. A *discrete/stochastic transition* $t_j \in (TD \cup TS)$ is *active* if and only if

$$\forall\, p_i \in P_{in}(t_j) : \begin{cases} m(p_i) - f(p_i \to t_j) \geq c_l(p_i) & if\ (p_i \to t_j) \in F \\ m(p_i) > f\left((p_i \to t_j)_{\mathcal{T}}\right) & if\ (p_i \to t_j)_{\mathcal{T}} \in \mathcal{T} \\ m(p_i) < f\left((p_i \to t_j)_{\mathcal{I}}\right) & if\ (p_i \to t_j)_{\mathcal{I}} \in \mathcal{I}, \end{cases}$$

and

$$\forall\, p_i \in P_{out}(t_i) :\ m(p_i) + f(t_j \to p_i) \leq c_u(p_i)$$

and the condition s_j must be fulfilled.
A *continuous transition* $t_j \in TC$ is *active* if and only if

$\forall\, p_i \in PC_{in}(t_j)$:

$$\begin{cases} m(p_i) > c_l(p_i) \vee (m(p_i) = c_l(p_i) \wedge I_i > 0) & if\ (p_i \to t_j) \in F \\ m(p_i) > f\left((p_i \to t_j)_{\mathcal{T}}\right) & if\ (p_i \to t_j)_{\mathcal{T}} \in \mathcal{T} \wedge (p_i \to t_j) \notin F \\ m(p_i) > f\left((p_i \to t_j)_{\mathcal{T}}\right) \vee \left(m(p_i) = f\left((p_i \to t_j)_{\mathcal{T}}\right) \wedge I_i > 0\right) & if\ (p_i \to t_j)_{\mathcal{T}} \in \mathcal{T} \wedge (p_i \to t_j) \in F \\ m(p_i) < f\left((p_i \to t_j)_{\mathcal{I}}\right) & if\ (p_i \to t_j)_{\mathcal{I}} \in \mathcal{I}, \end{cases}$$

and

$$\forall\, p_i \in PC_{out}(t_j) :\ m(p_i) < c_u(p_i) \vee (m(p_i) = c_u(p_i) \wedge O_i > 0).$$

and

$$\forall\, p_i \in PD_{in}(t_j) : \begin{cases} m(p_i) - f(p_i \to t_j) \geq c_l(p_i) & if\ (p_i \to t_j) \in F \\ m(p_i) > f\left((p_i \to t_j)_{\mathcal{T}}\right) & if\ (p_i \to t_j)_{\mathcal{T}} \in \mathcal{T} \\ m(p_i) < f\left((p_i \to t_j)_{\mathcal{I}}\right) & if\ (p_i \to t_j)_{\mathcal{I}} \in \mathcal{I}, \end{cases}$$

and

$$\forall\, p_i \in PD_{out}(t_j) :\ m(p_i) + f(p_i \to t_j) \leq c_u(p_i)$$

and the condition s_j must be fulfilled.

An active transition has to be enabled by all input and output places to become firable. Thereby, enabled discrete transitions wait until the assigned delay is elapsed and stochastic transitions fire first when the putative firing time is reached. However, continuous transitions fire immediately when they are enabled.

Several conflicts can occur when the places have to enable their connected active transitions. Possibly, a discrete place or a continuous place connected to discrete transitions has not enough marks to enable all output transitions simultaneously or cannot receive marks from all active input transitions due to the maximum capacity. Then a conflict arises that has to be resolved (*type-1-conflict*). This can be either done by providing the transitions with priorities or probabilities. In the first case, a deterministic process decides which place enables which transitions and in the second case the enabling is performed at random; thereby transitions assigned with a high probability are chosen preferentially.

Another conflict can occur between a continuous place and two or more continuous transitions when the input speed is not sufficient to fire all output transitions with the instantaneous speed $\tilde{v}_j$ (see equation (1)) (*type-2-output-conflict*) or when the output speed is not sufficient to fire all input transitions with the speed of equation (1) (*type-2-input-conflict*). This conflict is solved by sharing the speeds proportional to the assigned maximum speeds (see [17]).

If a conflict occurs between a place and continuous as well as discrete/stochastic transitions, the discrete/stochastic transitions take always priority over the continuous transitions (*type-3-conflict*).

A last conflict can occur when a discrete place has not enough marks to enable all connected continuous transitions (*type-4-conflict*). This is solved by prioritization of the involved transitions.

The *firing* is then performed in the following way. Discrete transitions fire by removing as much marks as the arc weights from all input places and by adding as many marks as the arc weights to all output places. However, the firing process of continuous transitions take place as a continuous flow with a maximum speed assigned to every transition. The recalculation of a *discrete marking* is described by an algebraic equation while a *continuous marking* is recalculated by a differential equation describing the flow of the continuous firing and an algebraic equation representing the firings of discrete transitions.

Definition 4. The tuple $(PD, PC, TD, TS, TC, F, G, \mathcal{T}, \mathcal{I}, \mathcal{R}, f, c_l, c_u, e, \wp, d, h, v, s, m_0)$ is an xHPN. The *firing process* of an active *continuous transition* $t_j \in TC$ is described by a negative mark change of all continuous input places which is expressed by the differential equation

$$\frac{dm(p_i)}{dt} = -f(p_i \to t_j) \cdot \tilde{v}_j \qquad \forall\, p_i \in PC_{in}(t_j)$$

and a positive mark change of all continuous output places which is expressed by the differential equation

$$\frac{dm(p_i)}{dt} = f(t_j \to p_i) \cdot \tilde{v}_j \qquad \forall\, p_i \in PC_{out}(t_j),$$

where $\tilde{v}_j$ is the *instantaneous speed* of transition t_j and calculated by the following equation if the transition is not involved in a type-2-conflict [2, 17]

$$\tilde{v}_j = \min\left\{ \min_{p_i \in PI_{in}(t_j)} \left(\frac{1}{f(p_i \to t_j)} \sum_{t_k \in TCF_{in}(p_i)} f(t_k \to p_i) \cdot \tilde{v}_k \right), \min_{p_i \in PI_{out}(t_j)} \left(\frac{1}{f(t_j \to p_i)} \sum_{t_k \in TCF_{out}(p_i)} f(p_i \to t_k) \cdot \tilde{v}_k \right), v_j \right\} \tag{1}$$

where $PI_{in}(t_j)$ is the set of continuous input places of t_j with $m(p_i) = c_l(p_i)$, $PI_{out}(t_j)$ is the set of continuous output places of t_j with $m(p_i) = c_u(p_i)$, $TCF_{in}(p_i) \subseteq TC$ is the set of all continuous firing input transitions, $TCF_{out}(p_i) \subseteq TC$ is the set of all continuous firing output transitions. If the transition is involved in a type-2-conflict, this speed has to be adapted appropriately (see [17]).

An active *discrete transition* $t_j \in TD$ waits d_j time units before it *fires* and a *stochastic transition* $t_j \in TS$ fires when the putative firing time τ_j is reached which is calculated based on the hazard function h. Both fire by removing the arc weight from all input places

$$m'(p_i) = m(p_i) - f(p_i \to t_j) \;\; \forall\, p_i \in P_{in}(t_j)$$

and by adding the arc weight to all output places

$$m'(p_i) = m(p_i) + f(t_j \to p_i) \;\; \forall\, p_i \in P_{out}(t_j).$$

The *marking* of a discrete place $p_i \in PD$ can be recalculated by the following algebraic equation

$$m'(p_i) = m(p_i) + \sum_{t_j \in TDF_{in}(p_i)} f(t_j \to p_i) - \sum_{t_j \in TDF_{out}(p_i)} f(p_i \to t_j),$$

whereby $TDF_{in}(p_i) \subseteq (TD \cup TS)$ is the set of all discrete/stochastic firing input transitions and $TDF_{out}(p_i) \subseteq (TD \cup TS)$ is the set of all discrete/stochastic firing output transitions.

The *continuous mark change* of a continuous place $p_i \in PC$ is performed with the aid of the following differential

$$\frac{dm(p_i)}{dt} = \sum_{t_j \in TCF_{in}(p_i)} f(t_j \to p_i) \cdot \tilde{v}_j - \sum_{t_j \in TCF_{out}(p_i)} f(p_i \to t_j) \cdot \tilde{v}_j$$

and, in addition, by the following algebraic equation for the discrete mark change caused by firing connected discrete transitions

$$m_{dis}(p_i) = \sum_{t_j \in TDF_{in}(p_i)} f(t_j \to p_i) - \sum_{t_j \in TDF_{out}(p_i)} f(p_i \to t_j).$$

At these discrete firing times the continuous marking is reinitialized by

$$m'(p_i) = m(p_i) + m_{dis}(p_i).$$

4 The Petri Net Library in Modelica (PNlib)

The xHPN definition and the corresponding definitions for activation, enabling, and firing mentioned above have been implemented by means of the object-oriented modeling language Modelica to enable graphical hierarchical modeling, hybrid simulation, and animation [18]. Modelica is developed and promoted by the Modelica Association since 1996 for modeling, simulation, and programming primarily of physical and technical systems and processes. Additionally, the Modelica standard library is available from the Modelica Association to model mechanical (1D/3D), electrical (analog, digital, machines), thermal, fluid, control systems, and hierarchical state machines. Furthermore, several libraries have been developed in the last decade for specific applications. An overview can be found on the Modelica homepage (www.modelica.org). The development of the language and libraries is ongoing and driven by several European projects (EUROSYSLIB, MODELISAR, OPENPROD, and MODRIO). Since the year 2000, Modelica is used successfully in the industry which is documented in the proceedings of many Modelica conferences and journals.

The Modelica models are described on the textual level by discrete, algebraic, and differential equations and by schematics on the graphical level. A schematic consists of connected *components* which are defined by other components or on the lowest level by equations in the Modelica syntax. The components have connectors which describe the interaction between them. By drawing a line from one component to another, a connection is established to enable interactions. In this manner a model is constructed. Several components can be structured in libraries, called *packages*, which provides hierarchical modeling. Moreover, the *wrapping technique* enables the representation of sub-models consisting of several connected components by a specific adapted icon in order to simplify the modeling process. Then, the sub-models can be used several times in the same or in different models and, in addition, it offers an easy-to-use-model at the top level with an intuitive and familiar adapted view.

For graphical modeling, simulation, and animation an appropriated environment is needed. Several commercial and open- source tools are available. A full list can be found on the Modelica homepage (www.modelica.org).

Each of the xHPN components - transitions, places, and arcs - is modeled by an own Modelica model which are organized and structured in a Modelica package, called *PNlib* (*Petri Net library*). All components are defined by discrete (event-based), algebraic, and differential equations (cp. [19]).

The main process in the place model is the recalculation of the marking after firing a connected transition. In the case of the discrete place model, this is realized by the discrete equation

```
when fire pre(reStart) then
   t = if fire then pre(t)+firingSumIn−firingSumOut
        else reStartTokens;
end when;
```

whereby `pre(t)` accesses the marking `t` immediately before the transitions fire. To this amount, the arc weight sum of all firing input transitions is added and the arc weight sum of all firing output transitions is subtracted from it. Additionally, the tokens are reset to `reStartTokens` when the user-defined condition `reStart` becomes true; this could be a global condition.

The marking of continuous places can change continuously as well as discretely. This is implemented by the following construct

```
der(t) = conMarkChange;
when discreteFire then
   reinit(t, t+discreteMarkChange);
end when;
when reStart then
   reinit(t,reStartMarks);
end when;
```

whereby the `der-operator` access the derivative of the marking `t` according to time. The continuous mark change is performed by a differential equation while the discrete mark change is performed by the `reinit`-operator within a discrete equation. This operator causes a re-initialization of the continuous marking every time when a connected discrete transition fires. Additionally, the marking is re-initialized by `reStartMarks` when the condition `reStart` becomes true.

The main process of the transition is to check if it can fire. When it is possible, discrete and stochastic transitions wait as long as the (random) delay is passed while the continuous transitions fires continuously with a speed calculated in the transition model. Via connector variables, the places report the transitions their markings and the transitions report when they fire. The conflicts which could arise in xHPN models have to be resolved by the mentioned methods to have successful and reliable simulation (see [17]).

5 Application

In this section we briefly demonstrate the PNlib possibilities in the software application VANESA (www.vanesa.sf.net). VANESA is a powerful and easy-to-use network modeling tool to members of the laboratory, which combines different fields of studies such as life science, database consulting, modeling and visualization for a semi-automatic and lab-validated reconstruction of biological networks. The application helps to model experimental results that can be expanded with database information to perform modern biological network analysis. Based on project experimental data and the integrated databases in VANESA, users are able to explore and reconstruct any biological system, which can be further transformed, simulated, and analyzed in the language of Petri nets.

Therefore, VANESA uses the PNlib and xHPNbio formalism for simulation processing. One feature of VANESA is the possibility to automatically transform any kind of network into the language of an xHPNbio. Thus, users are able to model and simulate dynamic systems within one tool. Petri net simulations are performed in the background, not visible to users. Simulation results are visualized in plots and animated within the active window (see Fig. 1).

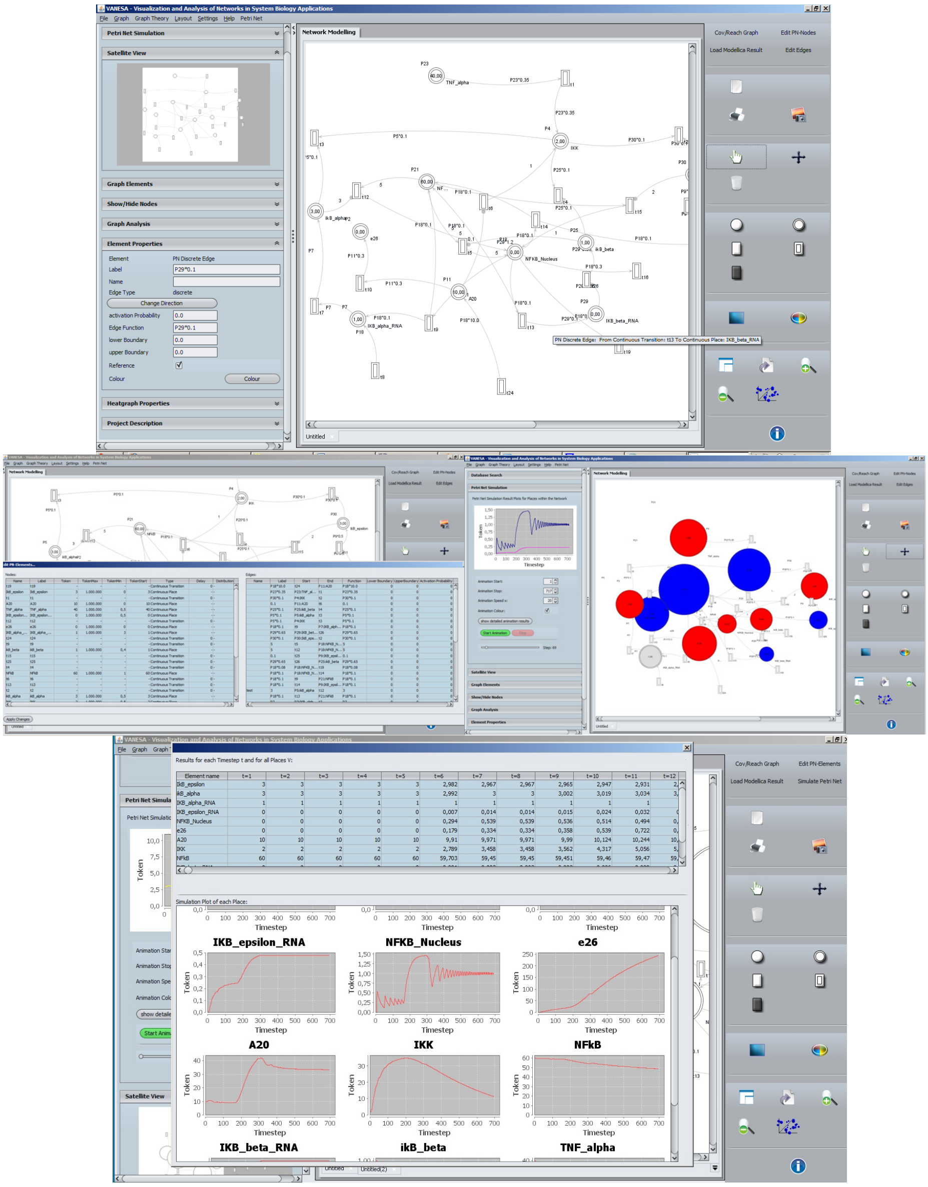
6 Conclusions

A powerful Petri net simulation environment has been developed to enable the processing of experimental data to usable new insights about biological systems. The mathematical modeling concept xHPNbio serves as specification for the presented simulation environment. The xHPNbio elements are modeled object-oriented by discrete, algebraic, and differential equations in the Modelica language. This allows an easy way to maintain, extend, and modify them. The hybrid simulation is performed by an appropriate Modelica tool which usually comprises several possibilities to adapt the solver settings in order to achieve reliable simulation results.

The mathematical modeling concept xHPNbio, was specially developed based on the demands of biological processes, and is so powerful but also so universal and generic that it is an ideal all-round-tool for modeling and simulation of nearly all kinds of processes such as business processes, production processes, logistic processes, work flows, traffic flows, data flows, multi-processor systems, communication protocols, and functional principals.

The PNlib will be freely available on the Modelica homepage (www.modelica.org) soon. However, in this paper we have already presented a software application, called VANESA, which uses the PNlib for simulation processes of biological networks.

In addition, a future goal is to provide an open source Petri net simulation tool for up till now the PNlib works only with the commercial Modelica tool Dymola. This demands a further development of the open source Modelica tool OpenModelica to get the PNlib to work with it because some Modelica features are not yet supported. The University of Applied Sciences Bielefeld is already closely involved in the further development of the OpenModelica tool [20].



Results for each Timestep t and for all Places V:

Element name	t=1	t=2	t=3	t=4	t=5	t=6	t=7	t=8	t=9	t=10	t=11	t=12
IkB_epsilon	3	3	3	3	3	2,982	2,967	2,967	2,965	2,947	2,931	2,
ikB_alpha	3	3	3	3	3	2,992	3	3	3,002	3,019	3,034	3,
IKB_alpha_RNA	1	1	1	1	1	1	1	1	1	1	1	
IKB_epsilon_RNA	0	0	0	0	0	0,007	0,014	0,014	0,015	0,024	0,032	
NFKB_Nucleus	0	0	0	0	0	0,294	0,539	0,539	0,536	0,514	0,494	0,
e26	0	0	0	0	0	0,179	0,334	0,334	0,358	0,539	0,722	0,
A20	10	10	10	10	10	9,91	9,971	9,971	9,99	10,124	10,244	10,
IKK	2	2	2	2	2	2,789	3,458	3,458	3,562	4,317	5,056	5,
NFkB	60	60	60	60	60	59,703	59,45	59,45	59,451	59,46	59,47	59,

Simulation Plot of each Place:

Fig. 1. An example of a simulation in VANESA. Starting point is the reconstruction of a biological network (NF-κB), which is automatically transformed into the xHPNbio formalism. The top picture shows the reconstruction of the biological network by data mining and information fusion. The middle left picture shows the xHPN user settings for the following simulation in the PNlib of Modelica. The middle right picture and the bottom picture show visualized and animated simulation results in VANESA. Simulation results can be plotted and also mapped on each element within the model. Plotted curves give detailed insight into system dynamics, whereas animated simulation results give insights about a certain time step. Shape and color of the elements present amount and developing of values (red increasing, blue decreasing).

References

1. Wiechert W.: Modeling and simulation: tools for metabolic engineering. Journal of biotechnology 94(1), 37–63 (2002)
2. David R., Alla H.: On Hybrid Petri Nets. Discrete Event Dynamic Systems: Theory and Applications(11), 9–40 (2001)
3. Janowski, S.J., Kormeier B., Töpel T., and Hofestädt R.:. Modeling of Cell-to-Cell communication processes with Petri Nets using the example of Quorum Sensing, in: E. Wingender (Ed.), Biological Petri Nets, 162. Amsterdam: IOS Press, 182 - 203 (2011)
4. Reddy V.N., Mavrovouniotis M.L., Liebman M.N.: Petri net representations in metabolic pathways. Proceedings of 1st International Conference on Intelligent Systems for Molecular Biology, 328-336 (1993)
5. Hofestädt R., Thelen S.: Quantitative modeling of biochemical networks. In Silico Biology 1(1),39–53 (1998)
6. Goss P., Peccoud J.: Quantitative modeling of stochastic systems in molecular biology by using stochastic Petri nets. Proceedings of the National Academy of Sciences of the United States of America 95(12), 6750-6755 (1998)
7. David R., Alla H.: Continuous Petri nets. Proceedings of 8th European Workshop on Application and Theory of Petri nets, 275-294 (1987)
8. Alla, H. and David, R., Continuous and hybrid Petri nets, Journal of Circuits, Systems, and Computers, 8:159- 188 (1998)
9. Matsuno H., Doi A., Nagasaki M., Miyano S.: Hybrid Petri net representation of gene regulatory network. Pacific Symposium on Biocomputing 5, 341-352 (2000)
10. Matsuno H., Tanaka Y., Aoshima H., Doi A., Matsui M., Miyano S.: Biopathways representation and simulation on hybrid functional Petri net. In Silico Biology 3(3), 389–404 (2003)
11. Chen M., Hofestädt R.: Quantitative Petri net model of gene regulated metabolic networks in the cell. In Silico Biology 3(3), 347–365 (2003)
12. Doi A., Fujita S., Matsuno H., Nagasaki M., Miyano S.: Constructing biological pathway models with hybrid functional Petri nets. In Silico Biology 4(3), 271–291 (2004)
13. Nagasaki M., Doi A., Matsuno H., Miyano S.: A Versatile Petri Net Based Architecture for Modeling and Simulation of Complex Biological Processes. Genome Informatics 15(1), 180-197 (2004)
14. Nagasaki M., Saito A., Jeong E., Li C., Kojima K., Ikeda E., Miyano S.: Cell Illustrator 4.0: A computational platform for systems biology. In Silico Biology 10(1), 5–26 (2010)
15. Rohr C., Marwan W., Heiner M.: Snoopy—a unifying Petri net framework to investigate biomolecular networks. Bioinformatics 26(7), 974 (2010)
16. Heiner M., Gilbert D., Donaldson R.: Petri nets for systems and synthetic biology. Proceedings 8th International Conference on Formal Methods for Computational Systems Biology, 215–264 (2008)
17. Proß S.: Hybrid Modeling and Optimization of Biological Processes. PhD thesis (in preparation) (2012)
18. Modelica Association: Modelica - A Unified Object-Oriented Language for Physical Systems Modeling Language Specification Version 3.2 (2010)
19. Proß S., Bachmann B.: An Advanced Environment for Hybrid Modeling of Biological Systems Based on Modelica. Journal of Integrative Bioinformatics(8), 1–34 (2011)
20. Braun W., Bachmann B., Proß S.: Synchronous Events in the OpenModelica Compiler with a Petri Net Library Application. Proceedings of 3rd International Workshop on Equation-Based Object-Oriented Modeling Languages and Tools, 63–70 (2010)

Simulative Model Checking of Steady State and Time-Unbounded Temporal Operators

Christian Rohr

Brandenburg University of Technology Cottbus,
Chair of Data Structures and Software Dependability,
Postbox 10 13 44, D-03013 Cottbus, Germany,
`rohrch@tu-cottbus.de`,
`http://www-dssz.informatik.tu-cottbus.de`

Abstract. When working with large stochastic models simulation remains the only possible analysis technique. Therefore, simulative model checking is the way to go. While finite time horizon algorithms are well known for probabilistic linear-time temporal logic, we provide an infinite time horizon procedure as well as steady state computation, based on exact stochastic simulation algorithms. We demonstrate the approach on models of the RKIP inhibited ERK pathway and angiogenetic process.

Keywords: simulative model checking, stochastic Petri net, steady state, unbounded temporal operator, probabilistic linear-time temporal logic

1 Introduction

Stochastic modelling of biochemical reaction networks is getting more and more popular. This also increases the demand for efficient analysis of such models. While small and medium-sized models can be analysed numerically, we focus on large or unbounded models. Therefore we use stochastic simulation to overcome the problem of state space explosion.

We use stochastic Petri nets ($\mathcal{SPN}$) [1] as modelling paradigm, which gives us a complete formalised and standardised framework, as well as an intuitive way of modelling concurrent behaviour. A number of biochemical species N involved in the biological model are represented as places $p_1 \dots p_N$, and the reactions between them refer to the transitions $t_1 \dots t_M$. The kinetics of a reaction is defined as possibly state-dependent rate function h_t assigned to the transition. Places and transitions are connected via directed arcs. Each arc contains the stoichiometric value of the associated species. The semantics of such an $\mathcal{SPN}$ is defined as continuous-time Markov chain (CTMC).

The dynamic behaviour of stochastic models can be analysed in different ways. We showed in [2] that numerical analysis is currently efficient up to 1×10^9 states. Beyond this limit, stochastic simulation remains the only possible technique. Stochastic simulation may be performed with approximate or exact methods. An approximate method is τ-leaping [3], which generates an approximate realisation of the stochastic process. Its advantage is the ability to jump over several transitions and thus be more efficient in trace generation than exact methods. But for simulative model checking, we need to know the exact occurrences of each transition. Therefore, only exact simulation algorithms are suitable for the purpose of simulative model checking, like Gillespie's direct method [4] or the next reaction method [5] by Gibson & Bruck.

In this paper, we extend the finite time horizon model checking algorithm of probabilistic linear-time temporal logic to an infinite time horizon and provide an algorithm to compute simulatively steady state formulas.

2 Stochastic Simulation

In biochemical reaction networks (with n molecular species and k reactions), the molecular reactions between the species are random processes, because it is impossible to predict the time at which the next reaction will occur. Stochastic modelling has therefore become an important tool to fully understand the system behaviour of such reaction networks.

The stochasticity can be described in a time-dependent manner by the Chemical Master Equation. In probability theory, this identifies the evolution as a continuous-time Markov chain (CTMC), with the integrated master equation obeying a Chapman-Kolmogorov equation. When working with biological systems, it may be infeasible to set up the CTMC as the state space $\mathcal{X} \subseteq \mathbb{N}^n$ can be very large or even infinite. The largeness of CTMCs makes simulation an important analysis technique: instead of computing the CTMC directly, simulation aims at imitating the CTMC by generating different paths of the CTMC, i.e., a sequence of discrete random variable $X_l(t)$. The discrete random variable $X_l(t)$ describes the number of molecules of species S_l, $l \in \{1, \ldots, n\}$ present at time t. The system state at time t is thus a discrete n-dimensional random vector $X(t) = (X_1(t), \ldots, X_n(t)) \in \mathcal{X}$. Given the system is in state $X(t)$, the probability that a transition/reaction of type $j \in \{1, \ldots, k\}$ will occur in the infinitesimal time interval $[t + \tau, t + \tau + \mathrm{d}\tau)$ is given by:

$$P(t + \tau, j | X(t))\mathrm{d}\tau = a_j(X(t)) \exp\left(-a_0(X(t))\tau\right) \mathrm{d}\tau$$

For each reaction j, the rate is given by the propensity function a_j, where $a_j(x)\mathrm{d}\tau$ is the conditional probability that a reaction of type j occurs in the infinitesimal time interval $[t, t + \mathrm{d}\tau)$, given state $X(t)$ at time t. The sum of the propensities of all possible transitions in the current state $X(t)$ is given by $a_0(X)$. Thus, the different (enabled) transitions in the net compete in a race condition and the fastest one determines next state and the time elapsed. In the new state, the race condition is started anew.

To analyse or understand the behaviour of a biochemical reaction network, many trajectories need to be simulated for a good approximation of the underlying CTMC. Although in principle known a long time before, Gillespie was the first who developed a supporting theory for a stochastic simulation of chemical kinetics [4]. He presented the Stochastic Simulation Algorithm (SSA; often also called Gillespie's algorithm), which is a Monte Carlo procedure for numerically generating CTMC. Since Gillespie's seminal work, several variants and different implementations and optimisations of the SSA have been proposed. Basically, each variant performs the following steps:

1. Initialise time $t = t_0$ and the system's state X at time t_0.
2. Repeat:
   (a) determine time increment $\tau \in \mathbb{R}$
   (b) select next reaction type j depending on the current state $X(t)$
   (c) perform state transition imposed by reaction of type j and update state vector X
   (d) update time $t = t + \tau$.
   until simulation time is reached.

The SSA simulates every state transition event, one at a time, and updates the system after each state transition. To determine the time increment τ and to select the next reaction requires to generate random numbers. Different realisations of the CTMC are obtained by different initialisations of the random number generator. Since reliable statements about the system behaviour (variance) can only be made based on many simulations, the usefulness of the simulation approach depends on the simulation time for each individual trajectory. Accelerating simulations is therefore desirable without changing the basic ideas of the algorithm.

Many variants of the SSA aim at reducing the computational cost of selecting the next reaction that will occur. Cao et al. [6] keep the reactions with larger propensities at the beginning of the list. The position of each reaction in the list is thereby determined after some pre-simulations. McCollum et al. [7] maintain a loosely sorted order of the reactions as the simulation proceeds. Instead of arranging the reactions in a linear list, Gibson & Bruck [5] propose to use advanced data structures (trees) to speed up the search for the next reaction that will occur. However, the time to manage the advanced data structures partially compensates the speed-up due to faster search [8].

Further performance increases of the SSA are obtained if only those propensities are recalculated that actually have changed after a state transition, whereas all others are reused (e.g., [8, 5]).

An additional speed-up to the SSA is provided by the approximate method τ-leaping [3], in which time t is advanced by a preselected amount τ and the numbers of firings of the individual transitions during the time interval $[t, t + \tau)$ are approximated by Poisson random numbers. Thus, instead of (sequentially) tracing every single state transition, several reactions are executed in parallel. With τ-leaping, it is assumed that all propensity functions are approximately constant in $[t, t + \tau)$, which is referred to as the leap condition. To ensure this, it is important to select τ sufficiently small, but also large enough to accelerate simulation.

3 Model Checking

Stochastic simulation produces traces through the state space of the model. For the specification of temporal formulas we define the probabilistic extension of the Linear-time Temporal Logic (LTL) [9] with numerical constraints [10], which is called Probabilistic Linear-time Temporal Logic with numerical constraints (PLTLc) [11]. The grammar of all PLTLc formulas is given in Definition 1.

Definition 1. *Probabilistic Linear-time Temporal Logic with Constraints*

$$\begin{aligned}
\psi &:= \mathcal{P}_{\bowtie x}\,[\phi] \mid \mathcal{P}_{=?}\,[\phi] \\
&\quad \bowtie\, \in \{<, \leq, \geq, >\}\,,\ x \in [0,1] \\
\phi &:= \mathbf{X}^{\,I}\phi \mid \mathbf{F}^{\,I}\phi \mid \mathbf{G}^{\,I}\phi \mid \phi\ \ \mathbf{U}^{\,I}\phi \mid \neg\phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid \sigma \\
I &:= [x_1, x_2] = \big\{x \in \mathbb{R}^{+} \mid\ x_1 \leq x \leq x_2\big\}\,,\ \mathit{omit}\ I = [0, \infty) \\
\sigma &:= \neg\sigma \mid \sigma \wedge \sigma \mid \sigma \vee \sigma \mid \mathit{value} \trianglelefteq \mathit{value} \mid \mathit{true} \mid \mathit{false} \\
&\quad \trianglelefteq\, \in \{<, \leq, \geq, >, =, \neq\} \\
\mathit{value} &:= \mathit{value} \sim \mathit{value} \mid \mathit{Place} \mid \$\mathit{Variable} \mid \mathit{Int} \mid \mathit{Real} \mid \mathit{function} \\
&\quad \sim\, \in \{+, -, *, /\}
\end{aligned}$$

The probability operator $\mathcal{P}$ has two different modes. If it is used with the question mark as $\mathcal{P}_{=?}\,[\phi]$ then it will return the probability $Pr(\phi)$ that ϕ is true. In the second case, $\mathcal{P}_{\bowtie x}\,[\phi]$ returns *true*, if $Pr(\phi) \bowtie x$ is fulfilled, *false* otherwise. In simulative model checking we compute a confidence interval (*c.i.*); in consequence of that, we have to introduce an additional return value in the second case. For simplicity, we assume the *c.i.* to have a lower and an upper bound $B_l, B_u \in \mathbb{R}_{\geq 0}$, such that the probability $Pr(\phi)$, which is not known in our case, is $B_l \leq Pr(\phi) \leq B_u$.

$$\mathcal{P}_{\bowtie x}[\phi] = \begin{cases} \mathit{true} & \text{if } x \bowtie [B_l, B_u] \wedge x \notin [B_l, B_u] \\ \mathit{false} & \text{if } x \not\bowtie [B_l, B_u] \wedge x \notin [B_l, B_u] \\ \mathit{unknown} & \text{if } x \in [B_l, B_u] \end{cases}$$

The operators $\neg$, $\wedge$, $\vee$ are the standard boolean operators *not*, *and*, *or*. Whereas $\mathbf{X}$, $\mathbf{F}$, $\mathbf{G}$, $\mathbf{U}$ denote the temporal operators *NEXT*, *FINALLY*, *GLOBALLY* and *UNTIL*. The *NEXT* operator ($\mathbf{X}^{\,I}\phi$) refers to *true* in the next state and within the time interval I. The *UNTIL* operator ($\phi_1\,\mathbf{U}^{\,I}\phi_2$) indicates that a state where ϕ_2 holds is eventually reached within the time interval I, while ϕ_1 continuously holds. The *FINALLY* operator ($\mathbf{F}^{\,I}\phi$) means that at some point within the time interval I a state where ϕ holds is eventually reached. Whereas the *GLOBALLY* operator ($\mathbf{G}^{\,I}\phi$) refers to the condition ϕ continuously holding true within the time interval I. The latter two are syntactic sugar, as they rely on the equivalences $\mathbf{F}\,\phi \equiv \mathit{true}\ \mathbf{U}\,\phi$ and $\mathbf{G}\,\phi \equiv \neg\mathbf{F}\,\neg\phi$.

A trace T fulfils a linear-time temporal logic formula ϕ if the following $\models$ relations hold:

$$\begin{aligned}
T \models \mathbf{X}\,\phi &\iff T^{(1)} \models \phi \\
T \models \mathbf{F}\,\phi &\iff \exists i \in \mathbb{N} : T^{(i)} \models \phi \\
T \models \mathbf{G}\,\phi &\iff \forall i \in \mathbb{N} : T^{(i)} \models \phi \\
T \models \phi_1\,\mathbf{U}\,\phi_2 &\iff \exists i \in \mathbb{N} : T^{(i)} \models \phi_2 \text{ and } \forall j \in \mathbb{N} \wedge j < i : T^{(j)} \models \phi_1 \\
T \models \neg\phi &\iff T \not\models \phi \\
T \models \phi_1 \wedge \phi_2 &\iff T \models \phi_1 \wedge T \models \phi_2 \\
T \models \phi_1 \vee \phi_2 &\iff T \models \phi_1 \vee T \models \phi_2 \\
T \models v_1 \trianglelefteq v_2 &\iff evalState(v_1, T^{(i)}) \trianglelefteq evalState(v_2, T^{(i)})
\end{aligned}$$

The function $evalState(v, T^{(i)})$ assigns a numerical value to the expression v by looking up the tokens that each place $x \in P(v)$ has in state $T^{(i)}$ of trace T.

In the next sections we present an algorithm to compute steady state formulas, and afterwards an algorithm to compute time-unbounded temporal operators in a simulative manner. Time-bounded algorithms for simulative model checking are well known, i.e., [10].

3.1 Steady State Computation

In steady state simulation, the measures of interest are defined as limits, as the length of the simulation goes to infinity. There is no natural event to terminate the simulation, so the length of the simulation is made large enough to get "good" estimates of the quantities of interest. Steady-state simulation generally poses two problems:

1. The existence of a transient phase may cause the estimate to be biased.
2. The simulation runs are long, and usually one cannot afford to carry out many independent simulations.

These are several methods that allow to cope with these problems to some extent. Among them are: the batch means method, the method of independent replicas, and the regeneration method. Each of these methods has its advantages and disadvantages. In our implementation we use a sample batch means algorithm to compute the steady state.

We choose Skart [12], which is an automated sequential procedure for on-the-fly steady state simulation output analysis, because it is specifically designed to handle observation-based statistics and usually requires a smaller initial sample size compared with other well-known simulation analysis procedures [12]. This algorithm partitions a long simulation run into batches, computes an average statistics for each batch and constructs an interval estimate using the batch means. Based on this interval estimate Skart decides whether a steady state is reached or more samples were needed. A detailed description of the algorithm is given in [12].

We extend PLTLc with the steady state operator $\mathcal{S}$. Definition 2 states the syntax of it. The return values are quite the same as for the probability operator $\mathcal{P}$. Inside of $\mathcal{S}$ are only state formulas allowed, i.e., no temporal operators.

Definition 2. *Extension of PLTLc with steady state operator $\mathcal{S}$.*

$$\psi := \mathcal{P}_{\bowtie x}\left[\phi\right] \mid \mathcal{P}_{=?}\left[\phi\right] \mid \mathcal{S}_{\bowtie x}\left[\sigma\right] \mid \mathcal{S}_{=?}\left[\sigma\right]$$
$$\bowtie \in \left\{<, \leq, \geq, >\right\}, \ x \in [0, 1]$$

Steady-state formulas are computed with Algorithm 1. At first the simulation trace is created until the steady state is reached (line 4–8). To get an unbiased result, we cut off the first n states, which bias the steady state (line 9). The steady state probability is now the ratio $^{T_o}/_{T_s}$ of the occupation time T_o of the fulfilling states and the simulation time T_s (line 11–19). But this gives correct results only for those Petri nets, where the reachability graph consists of only one strongly connected component. The complexity of this decision is the same as for constructing the reachability graph. To solve this problem, one has to make several steady state computations and average the values. In that way the individual steady state estimates are weighted according to the strongly connected components.

Algorithm 1 Steady state computation for one simulation run

```
Require: $\boldsymbol{trace} \leftarrow (m_0, t_0)$
 1: procedure EVALSTEADYSTATE($\sigma$)
 2:     $steadyStateReached \leftarrow false$
 3:     $pos \leftarrow 0$
 4:     repeat
 5:         $\boldsymbol{trace} \leftarrow \boldsymbol{trace}$ + NEXTSTATE($\boldsymbol{trace}^{(pos)}$)
 6:         $pos \leftarrow pos + 1$
 7:         $steadyStateReached \leftarrow$ CHECKSTEADYSTATE($\boldsymbol{trace}$)
 8:     until $steadyStateReached = true$
 9:     $cutOff \leftarrow$ GETSTEADYSTATECUTOFF
10:     $T_o \leftarrow 0,\ T_s \leftarrow 0$
11:     for $i \leftarrow cutOff,\ |\boldsymbol{trace}|$ do
12:         $(s_i, t_i) \leftarrow \boldsymbol{trace}^{(i)}$                ▷ state $s_i$, sojourn time $t_i$ in $s_i$
13:         $T_s \leftarrow T_s + t_i$
14:         $res \leftarrow$ EVALSTATE($\sigma, s_i$)
15:         if $res = true$ then
16:             $T_o \leftarrow T_o + t_i$
17:         end if
18:     end for
19:     return $T_o/T_s$
20: end procedure
```

3.2 Verification of Time-Unbounded Until

In Algorithm 2 we present the algorithm for checking time-unbounded until formulas. It is nearly the same as for time-bounded formulas except that one has to stop the simulation trace at some time point. The decision of doing that is the crucial part of the algorithm. We assume that reaching the steady state is a reasonable stopping criteria (line 38). A system in a steady state has numerous properties that are unchanging in time. This implies that for any property p of the system, the partial derivative with respect to time is zero:

$$\frac{\partial p}{\partial t} = 0$$

If a system is in steady state, then the recently observed behaviour of the system will continue into the future. In stochastic systems, the probabilities that various states will be repeated will remain constant.

4 MARCIE: An Implementation

MARCIE [13] is a tool for analysing generalised stochastic Petri nets ($\mathcal{GSPN}$), supporting qualitative and quantitative analyses including model checking capabilities. Particular features are symbolic state space analysis including efficient saturation-based state space generation, evaluation of standard Petri net properties as well as Computational Tree Logic model checking. Further it offers symbolic Continuous Stochastic Logic model checking and permits to compute expectations for rewards which can be added to the core $\mathcal{GSPN}$. Most of MARCIE's features are realised on top of an Interval Decision Diagram (IDD) implementation [14]. IDDs are used to efficiently encode interval logic functions representing marking sets of bounded Petri nets. Thus, MARCIE falls into the category of symbolic analysis tools. However, it additionally comprises approximative and simulative engines, which work explicitly, to support also stochastic analysis of very large and unbounded nets. It includes two exact simulation algorithms, firstly Gillespie's direct method [4], and secondly the next reaction method by Gibson & Bruck [5].

Parallelising the simulation is a good way to speed-up the computation. We use the Message Passing Interface (MPI) to develop a portable and scalable simulation tool for large-scale models. The desired number of simulations runs is equally distributed to the worker processes. Therefore the run-time decreases with the number of workers.

MARCIE is completely written in C++, and makes use of the libraries GMP, pthreads, flex/bison and boost. It comprises about 45,000 lines of source code. MARCIE is available for non-commercial use; we provide statically linked binaries for Linux and Mac OS X. The tool, the manual and a benchmark suite can be found on our website http://www-dssz.informatik.tu-cottbus.de/marcie.html. MARCIE itself comes with a textual user interface. Options and input files can also be specified by a generic Graphical User Interface (GUI), written in Java, which can be easily configured by means of a XML description. The GUI is part of our Petri net analyser Charlie [15].

Algorithm 2 Unbounded Until for one simulation run

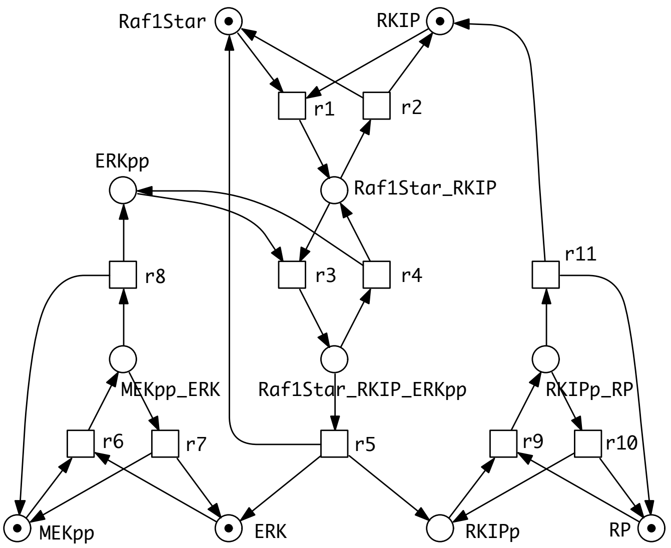
```
Require: $\boldsymbol{trace} \leftarrow (m_0, t_0)$
procedure EVALFORMULA($\phi$, $pos$, $\boldsymbol{trace}$)
    $steadyStateReached \leftarrow false$, $res \leftarrow false$
    repeat
        switch $\phi$
            case $\sigma$ :
                $(s_{pos}, t_{pos}) \leftarrow \boldsymbol{trace}^{(pos)}$
                $res \leftarrow$ EVALSTATE($\sigma$, $s_{pos}$)
                return ($pos$, $res$)
            case $\neg\phi_1$ :
                ($pos_1$, $res_1$) $\leftarrow$ EVALFORMULA($\phi_1$, $pos$, $\boldsymbol{trace}$)
                return ($pos_1$, $\neg res_1$)
            case $\phi_1 \wedge \phi_2$ :
                ($pos_1$, $res_1$) $\leftarrow$ EVALFORMULA($\phi_1$, $pos$, $\boldsymbol{trace}$)
                ($pos_2$, $res_2$) $\leftarrow$ EVALFORMULA($\phi_2$, $pos$, $\boldsymbol{trace}$)
                return ($min$($pos_1$, $pos_2$), $res_1 \wedge res_2$)
            case $\phi_1 \vee \phi_2$ :
                ($pos_1$, $res_1$) $\leftarrow$ EVALFORMULA($\phi_1$, $pos$, $\boldsymbol{trace}$)
                ($pos_2$, $res_2$) $\leftarrow$ EVALFORMULA($\phi_2$, $pos$, $\boldsymbol{trace}$)
                return ($max$($pos_1$, $pos_2$), $res_1 \vee res_2$)
            case $X\phi_1$ :
                if $pos = |\boldsymbol{trace}|$ then
                    $\boldsymbol{trace} \leftarrow \boldsymbol{trace} +$ NEXTSTATE($\boldsymbol{trace}^{(pos)}$)
                end if
                $pos \leftarrow pos + 1$
                ($pos_1$, $res_1$) $\leftarrow$ EVALFORMULA($\phi_1$, $pos$, $\boldsymbol{trace}$)
                return ($pos_1$, $res_1$)
            case $\phi_1 U \phi_2$ :
                ($pos_2$, $res_2$) $\leftarrow$ EVALFORMULA($\phi_2$, $pos$, $\boldsymbol{trace}$)
                if $res_2 = true$ then
                    return ($pos_2$, $res_2$)
                end if
                ($pos_1$, $res_1$) $\leftarrow$ EVALFORMULA($\phi_1$, $pos$, $\boldsymbol{trace}$)
                if $res_1 = false$ then
                    return ($pos_1$, $res_1$)
                end if
                $pos \leftarrow pos_1$
        end switch
        $steadyStateReached \leftarrow$ CHECKSTEADYSTATE($\boldsymbol{trace}$)
        if $pos = |\boldsymbol{trace}|$ then
            $\boldsymbol{trace} \leftarrow \boldsymbol{trace} +$ NEXTSTATE($\boldsymbol{trace}^{(pos)}$)
        end if
        $pos \leftarrow pos + 1$
    until $steadyStateReached = true$
    return ($pos$, $res$)
end procedure
```

5 Case Studies

In this section we demonstrate our approach on the models of the RKIP inhibited ERK pathway and angiogenetic process. All Petri nets were modelled with Snoopy [16] and analysed with MARCIE [13]. The experiments were carried out on a machine with 4x AMD Opteron™ 6276 with 2.3 GHz and 256GB RAM running CentOS 6.

5.1 RKIP inhibited ERK pathway

This model shows the influence of the Raf Kinase Inhibitor Protein (RKIP) on the Extracellular signal Regulated Kinase (ERK) signalling pathway. A model of non-linear ordinary differential equations was originally published in [17]. Later on, it was discussed as qualitative and continuous Petri nets in [18], and as three related Petri net models in [19]. The stochastic Petri net $\mathcal{SPN}_{ERK}$ comprises 11 places and 11 transitions connected by 34 arcs and is shown in Fig. 1. All transition rate functions use mass action kinetics with the original parameter



$$\text{Raf1Star} + \text{RKIP} \overset{r1,r2}{\leftrightarrow} \text{Raf1Star\_RKIP}$$
$$\text{Raf1Star\_RKIP} + \text{ERKpp} \overset{r3,r4}{\leftrightarrow} \text{Raf1Star\_RKIP\_ERKpp}$$
$$\text{Raf1Star\_RKIP\_ERKpp} \overset{r5}{\rightarrow} \text{Raf1Star} + \text{ERK} + \text{RKIPp}$$
$$\text{ERK} + \text{MEKpp} \overset{r6,r7}{\leftrightarrow} \text{MEKpp\_ERK} \overset{r8}{\rightarrow} \text{ERKpp} + \text{MEKpp}$$
$$\text{RKIPp} + \text{RP} \overset{r9,r10}{\leftrightarrow} \text{RKIPp\_RP} \overset{r11}{\rightarrow} \text{RKIP} + \text{RP}$$

Fig. 1. Stochastic Petri net of the RKIP inhibited ERK pathway, including textual representation of the chemical reactions.

values from [17]. The model is scalable by the initial amount of tokens in the places RKIP, MEKpp, ERK and RP. The more initial tokens on each of these places, the bigger the state space of the Petri net. Table 1 shows the number of reachable states for different initial markings.

Table 1. The size of the state space for different initial markings of $\mathcal{SPN}_{ERK}$ computed with MARCIE's symbolic state space generation. All places which carry one token in Fig. 1 have now initially N tokens.

N	|states|	N	|states|	N	|states|	N	|states|
5	1,974	20	1,696,618	40	79,414,335	100	1.591×10^{10}
10	47,047	25	5,723,991	50	2.834×10^{8}	250	3.582×10^{12}
15	368,220	30	15,721,464	60	8.114×10^{8}	500	2.231×10^{14}

We first check the reachability of a state at some time in the future, such that the number of tokens on place *MEKpp* is between 60% and 80% of N:

$$\mathcal{P}_{=?}\left[\mathbf{F}\ [MEKpp \geq N \cdot 0.6 \wedge MEKpp \leq N \cdot 0.8]\right].$$

In any case such a state was reached, therefore the probability of the formula is 1, see Table 2.

Table 2. Reachability analysis for different initial markings N of $\mathcal{SPN}_{ERK}$. The total time is given for different numbers of workers.

N	1	2	4	8	16	32	64	result
20	0m56s	0m28s	0m14s	0m7s	0m3s	0m2s	0m0s	[1,1]
30	1m17s	0m38s	0m19s	0m10s	0m4s	0m3s	0m1s	[1,1]
40	1m38s	0m48s	0m24s	0m12s	0m6s	0m3s	0m1s	[1,1]
50	1m57s	0m59s	0m30s	0m15s	0m7s	0m4s	0m2s	[1,1]
60	2m23s	1m9s	0m35s	0m17s	0m8s	0m5s	0m3s	[1,1]

Since we know now that such a state is eventually reached, we want to compute the steady state probability of being in such a state, where the number of tokens on place *MEKpp* is between 60% and 80% of N:

$$\mathcal{S}_{=?}\left[MEKpp \geq N \cdot 0.6 \wedge MEKpp \leq N \cdot 0.8\right].$$

This is the same property as in [2], so we can verify the correctness of the results. The results in Table 3 show first that the resulting confidence interval covers the probability computed by the Jacobi method in [2]. Second the algorithm scales nearly linear with the number of worker processes. A very interesting behaviour regards the relationship between the state space size and the total run-time of the computation. One could expect an increase of the run-time, but it stays the same. This is a result of the level semantics described in [20].

Table 3. Steady state analysis for different initial markings N of $\mathcal{SPN}_{ERK}$. The total time is given for different numbers of workers.

N	1	2	4	8	16	32	64	result
20	7m31s	3m46s	1m53s	0m57s	0m27s	0m17s	0m10s	[0.77482, 0.77534]
30	7m34s	3m43s	1m51s	0m57s	0m28s	0m17s	0m11s	[0.83277, 0.83325]
40	7m40s	3m43s	1m56s	0m57s	0m28s	0m18s	0m11s	[0.87416, 0.87470]
50	7m43s	3m57s	1m57s	1m0s	0m30s	0m17s	0m11s	[0.90437, 0.90486]
60	7m53s	3m59s	1m56s	1m1s	0m28s	0m17s	0m12s	[0.92641, 0.92696]

5.2 Angiogenesis

Angiogenesis is a complex phenomenon that goes from a molecular level to macroscopic events. This Petri net models a part of the signal transduction pathway involved in the angiogenetic process and was originally published in [21]. The stochastic Petri net $\mathcal{SPN}_{ANG}$ comprises 39 places and 64 transitions connected by 185 arcs.

The model is scalable by the initial amount of tokens in the places *Akt*, *DAG*, *Gab1*, *KdStar*, *Pip2*, *P3k*, *Pg* and *Pten*. The more initial tokens on each of these places, the bigger the state space of the Petri net. The number of reachable states for different initial markings are shown in Table 4. As in the previous case study we check for reachability first. Now we want to know the probability of eventually reaching a state where no tokens reside on place *Akt*:

Table 4. The size of the state space for different initial markings of $\mathcal{SPN}_{ANG}$ computed with MARCIE's symbolic state space generation. The places *Akt*, *DAG*, *Gab*1, *KdStar*, *Pip*2, *P3k*, *Pg* and *Pten* carry initially N tokens.

N	|states|	N	|states|	N	|states|	N	|states|
1	96	4	2,413,480	7	2.181×10^{9}	10	4.537×10^{11}
2	5,384	5	29,224,050	8	1.464×10^{10}	15	5.207×10^{14}
3	144,188	6	277,789,578	9	8.623×10^{10}	20	1.428×10^{17}

$$\mathcal{P}_{=?}\left[\mathbf{F}\ [Akt = 0]\right].$$

In contrast to $\mathcal{SPN}_{ERK}$, Table 5 shows that the probability ranges from about $0.44\,(N = 1)$ to $0.9\,(N = 6)$. That means a state where no tokens lay on place *Akt* is not always reached, because the CTMC consists of several strongly connected components and in some of them such a state does not exist. Secondly we compute the steady state probability of being in a state that has no tokens on place *Akt*:

$$\mathcal{S}_{=?}\left[Akt = 0\right].$$

The results in Table 6 show that the steady state probability is nearly the same as in the reachability case as the overall steady state probability consists of

Table 5. Reachability analysis for different initial markings N of $\mathcal{SPN}_{ANG}$. The total time is given for different numbers of workers.

N	1	2	4	8	16	32	64	result
1	12m10s	6m13s	3m3s	1m29s	0m49s	0m25s	0m13s	[0.44542, 0.44642]
2	60m19s	30m18s	14m47s	7m14s	3m36s	1m52s	1m17s	[0.81292, 0.81370]
3	65m37s	35m41s	18m31s	8m20s	4m2s	2m1s	1m55s	[0.94319, 0.94365]
4	74m43s	37m14s	19m52s	9m45s	4m58s	2m18s	3m0s	[0.98189, 0.98216]
5	79m45s	38m37s	18m54s	9m20s	4m37s	2m22s	3m35s	[0.99380, 0.99396]
6	79m37s	39m57s	19m50s	9m14s	4m34s	2m11s	3m3s	[0.99760, 0.99770]

two parts, first the probability of reaching a strongly connected component and second the steady state probability inside these component. The result means the steady state probability inside a strongly connected component, where a state exists with $Akt = 0$, is almost 1. That's why the overall steady state probability almost coincides with the reachability probability.

Table 6. Steady state analysis for different initial markings N of $\mathcal{SPN}_{ANG}$. The total time is given for different numbers of workers.

N	1	2	4	8	16	32	64	result
1	7m1s	3m17s	1m42s	0m47s	0m26s	0m13s	0m9s	[0.43773, 0.44771]
2	28m25s	14m10s	6m54s	3m33s	1m34s	0m51s	0m36s	[0.80446, 0.81237]
3	58m42s	30m19s	17m54s	7m32s	3m46s	2m30s	1m20s	[0.92772, 0.93284]
4	94m27s	49m59s	24m39s	11m53s	6m11s	3m25s	2m11s	[0.97859, 0.98140]
5	133m56s	66m44s	33m24s	17m22s	8mm44s	4m49s	3m12s	[0.98923, 0.99121]
6	170m57s	85m25s	42m54s	22m12s	11m13s	6m57s	3m30s	[0.99649, 0.99758]

6 Related work

To compute the transient probability of the formula $\mathcal{P}_{=?}[\phi_1 \, \mathbf{U} \, \phi_2]$ in state s means to compute the probability distribution starting in s and making states absorbing, which satisfy $\neg\phi_1 \vee \phi_2$. The resulting linear system of equations can be solved numerically by iterative methods like Gauss-Seidel or Jacobi. There are several tools available that support such solvers, among them MARCIE [13]. The drawback of numerical solvers is their restriction to bounded CTMCs and the complexity is typically $\mathrm{O}(n)$ and in worst case $\mathrm{O}(n^2)$. On the other hand they compute an "exact" result. The same methods were used to compute the steady state distribution of bounded CTMCs for computing the steady state probability of formulas like $\mathcal{S}_{=?}[\phi]$.

Statistical model checking [22–24] is a quite similar approach to simulative model checking, but differs in some details. Hypothesis testing, i.e., sequential probability ratio test (SPRT), has good performance compared to the computation of point estimates, but it can only check formulas like $\mathcal{P}_{\bowtie x}$. In the end, the user gets a result of *true* or *false* and has no idea of the scale of the estimated probability.

The Monte Carlo Model Checker MC2 [11] computes a point estimate of a Probabilistic LTL logic (with numerical constraints) formula to hold for a model. MC2 does not include any simulation engine but works offline by taking a set of sampled trajectories generated by any simulation or ODE solver software.

Last not least, a combination of simulation and reachability analysis were used to compute time-unbounded formulas in [22, 25]. But this approach suffers from the same restrictions of bounded state spaces as the numerical methods.

7 Conclusions

In this paper we presented an infinite time horizon model checking algorithm plus steady state operator for probabilistic linear-time temporal logic. We verified the results of the simulative approach against the numerical solutions of the Jacobi and Gauss-Seidel methods. We proved the efficiency of our algorithm and the scalability by using several worker processes through MPI.

As our algorithm is based on stochastic simulation, its run-time does not directly depend on the size of the state space, as for the numerical methods, but on the rate functions of the transitions and the structural size of the Petri net. That is the greater the sum of the transitions rates, the smaller the time steps are, and the more simulation steps need to be done to reach a certain time point.

The main drawback of simulation-based methods remains. The achieved accuracy depends on the number of simulation runs and grows exponentially with the expected accuracy. Therefore methods of choice for bounded and medium-sized models are numerical, otherwise simulation plays out its strength.

References

1. Marsan, M.A., Balbo, G., Conte, G., Donatelli, S., Franceschinis, G.: Modelling with Generalized Stochastic Petri Nets. Wiley Series in Parallel Computing, John Wiley and Sons (1995) 2nd Edition.
2. Heiner, M., Rohr, C., Schwarick, M., Streif, S.: A comparative study of stochastic analysis techniques. In: Proc. CMSB 2010, ACM (2010) 96–106
3. Gillespie, D.T.: Approximate accelerated stochastic simulation of chemically reacting systems. The Journal of Chemical Physics **115**(4) (2001) 1716–1733
4. Gillespie, D.: Exact stochastic simulation of coupled chemical reactions. The Journal of Physical Chemistry **81(25)** (1977) 2340–2361
5. Gibson, M.A., Bruck, J.: Efficient exact stochastic simulation of chemical systems with many species and many channels. The Journal of Physical Chemistry **A 104** (2000) 1876–1889
6. Cao, Y., Gillespie, D.T., Petzold, L.R.: Adaptive explicit-implicit tau-leaping method with automatic tau selection. J Chem Phys **126**(22) (Jun 2007) 224101

7. McCollum, J.M., Peterson, G.D., Cox, C.D., Simpson, M.L., Samatova, N.F.: The sorting direct method for stochastic simulation of biochemical systems with varying reaction execution behavior. Comput. Biol. Chem. **30**(1) (2006) 39–49
8. Cao, Y., Li, H., Petzold, L.: Efficient formulation of the stochastic simulation algorithm for chemically reacting systems. J Chem Phys **121**(9) (Sep 2004) 4059–4067
9. Pnueli, A.: The temporal logic of programs. In: Proceedings of the 18th IEEE Symposium on the Foundations of Computer Science, IEEE Computer Society Press (1977) 46–57
10. Fages, F., Rizk, A.: On the analysis of numerical data time series in temporal logic. In: Proc. CMSB 2007, LNCS/LNBI 4695, Springer (2007) 48–63
11. Donaldson, R., Gilbert, D.: A Monte Carlo model checker for probabilistic LTL with numerical constraints. Technical report, University of Glasgow, Dep. of CS (2008)
12. Tafazzoli, A., Wilson, J.R., Lada, E.K., Steiger, N.M.: Skart: A skewness- and autoregression-adjusted batch-means procedure for simulation analysis. In: Winter Simulation Conference. (2008) 387–395
13. Schwarick, M., Rohr, C., Heiner, M.: Marcie - model checking and reachability analysis done efficiently. In: Proc. 8th International Conference on Quantitative Evaluation of SysTems (QEST 2011), IEEE CS Press (September 2011) 91–100
14. Tovchigrechko, A.: Model Checking Using Interval Decision Diagrams. PhD thesis, BTU Cottbus, Dep. of CS (2008)
15. Franzke, A.: A concept for redesigning Charlie. Technical report, BTU Cottbus, Dep. of CS (2008)
16. Rohr, C., Marwan, W., Heiner, M.: Snoopy–a unifying Petri net framework to investigate biomolecular networks. Bioinformatics **26**(7) (2010) 974–975
17. Cho, K.H., Shin, S.Y., Kim, H.W., Wolkenhauer, O., McFerran, B., Kolch, W.: Mathematical modeling of the influence of RKIP on the ERK signaling pathway. In: Proc. CMSB 2003, LNCS 2602, Springer (2003) 127–141
18. Gilbert, D., Heiner, M.: From Petri nets to differential equations - an integrative approach for biochemical network analysis. In: Proc. ICATPN 2006, LNCS 4024, Springer (2006) 181–200
19. Heiner, M., Donaldson, R., Gilbert, D. In: Petri Nets for Systems Biology, in Iyengar, M.S. (ed.), Symbolic Systems Biology: Theory and Methods. Jones and Bartlett Publishers, Inc. (2010)
20. Calder, M., Duguid, A., Gilmore, S., Hillston, J.: Stronger computational modelling of signalling pathways using both continuous and discrete-state methods. In: Proc. CMSB 2006, LNBI 4210, Springer (2006) 63–78
21. Napione, L., Manini, D., Cordero, F., Horvath, A., Picco, A., Pierro, M.D., Pavan, S., Sereno, M., Veglio, A., Bussolino, F., Balbo, G.: On the Use of Stochastic Petri Nets in the Analysis of Signal Transduction Pathways for Angiogenesis Process. In: Proc. CMSB 2009, LNCS/LNBI 5688, Springer (2009) 281–295
22. Younes, H., Clarke, E., Zuliani, P.: Statistical verification of probabilistic properties with unbounded until. In: Formal Methods: Foundations and Applications. Volume 6527 of Lecture Notes in Computer Science. Springer (2011) 144–160
23. Basu, S., Ghosh, A.P., He, R.: Approximate model checking of PCTL involving unbounded path properties. In: ICFEM. (2009) 326–346
24. Ballarini, P., Forlin, M., Mazza, T., Prandi, D.: Efficient parallel statistical model checking of biochemical networks. In: Proc. PDMC. (2009) 47–61
25. Zapreev, I.S.: Model checking Markov chains : techniques and tools. PhD thesis, University of Twente, Enschede (March 2008)

Studying prostate cancer as a network disease by qualitative computer simulation with Stochastic Petri Nets

Nicholas Stoy[1] Sophie Chen[2] Andrzej M. Kierzek[3]

[1] St Georges University of London, Cranmer Terrace, London SW17 0RE,
`nicholasstoy@Yahoo.co.uk`,
[2] OPCaRT, Surrey Research Park, Guildford, GU2 7AF, UK
`s.chen@opcart-lab.org`
[3] Faculty of Health and Medical Sciences, University of Surrey, Guildford, GU2 7XH, UK
`a.kierzek@surrey.ac.uk`

Abstract. Despite increased understanding of cancer pathogenesis, translating this knowledge into therapy remains challenging. Radical progress depends on utilizing molecular biology knowledge to understand processes by which genetic information is executed in response to microenvironmental perturbations. Understanding of the molecular machinery of the cell requires computer simulation of the complex network of molecular interactions and Petri net offers ideal framework. Lack of quantitative data about molecular amounts and transition rates necessitates development of qualitative methods providing useful insight with minimal knowledge on quantitative parameters. Here we show work in progress on the modeling of molecular interaction network involved in the evolution of prostate cancer. We use statistical model checking of qualitative model and show that the effects of genetic and pharmacological perturbations on prostate cancer evolution can be predicted by the number of token game trajectories reaching nodes representing proliferation and cell death events.

Keywords: Stochastic Petri net, statistical model checking, Prostate cancer

1 Introduction

Despite increased understanding of cancer pathogenesis, translating this knowledge into therapy remains challenging. Radical progress depends on utilizing advances in both DNA sequencing technology and molecular biology to understand processes by which genetic information is executed in response to microenvironmental perturbations. However, the expression of genetic information in response to environmental signals is performed by complex molecular machinery involving hundreds of thousands of components influencing each other through non-linear interactions. While the century of biochemistry and molecular biology resulted in accumulation of voluminous data on molecular components and their interactions the size and complexity of the system makes it impossible to use this knowledge without computer models. This situation motivated intense research towards reconstruction or reverse engineering the network of interacting cellular components in the form of computer simulation capable of predicting effects of genetic and environmental perturbations on cellular behavior [1-4]. The major bottleneck in this effort is the lack of quantitative parameters describing molecular interactions. The existence and sign (activation, inhibition) of the molecular interaction are much more amenable to experimental studies than quantitative measurements of molecular amounts or transition rates. This encourages development of qualitative simulation approaches where valuable insight can be provided by analysis of molecular interaction network connectivity, with very limited quantitative data. In this contribution we present the Petri Net model of the molecular interaction network involved in the evolution of prostate cancer. We show that statistical model checking of the qualitative model, with discretized molecular activities and uniform transition rates provides valuable predictions of the effects of genetic and pharmacological perturbations on cancer progression.

2 Petri Net model of Prostate Cancer Network.

Molecular species were represented as places and interactions were represented by transitions. Edges and read edges were used to represent activation interactions whilst inhibitory interactions were represented by molecular state-change transitions rather than by inhibitory edges, as compiled from research data. Our model building started from interactions involved in the control of the PTEN gene. Subsequently, we have included transitions involving molecules controlling PTEN and proceeded until major inputs for environmental signals were covered. The model, which is still very much 'work in progress', currently includes growth factors, insulin, Wnt and androgen receptors activating the signaling networks of PI3K-PTEN-Akt, Ras-Raf_MEK-ERK, beta-catenin, c-Myc, mTor, p53 and p27. The intrinsic apoptotic pathway and a very basic representation of the cell cycle was included in the network. Two special places were introduced into the network to represent biological outcomes of cell death and proliferation. The model has been build using a Snoopy Petri Net tool [5] in Extended Petri Net mode. The final Petri Net model contains 251 nodes and 195 transitions.

3 Statistical Model Checking.

Cancer develops when the cell escapes the cell death program and starts proliferating out of the control of growth factor signals. Thus in our simulations we investigated reachability of the places representing cell death and proliferation. In particular, we were interested in whether the proliferation place is reached before the cell death place is reached. Since, we do not know molecular amounts and transition rates in the system we have attempted qualitative simulation.

We allowed the state of each place to vary between 0 and 2 tokens, thus representing absent, low activity and high activity states of molecular components. Even with this radical discretization the model was still too large to determine reachability of proliferation place by analysis of full reachability graph. We have therefore used a statistical model checking approach. For each simulation we have generated ensemble of token game trajectories starting from the same initial conditions and determined the number of trajectories in which the marking of Proliferation place changed from 0 to 1 while the marking of Cell Death place remained at initial value of 0. In short, we determined the number of token game trajectories in which proliferation occurred before cell death. We decided to use Gillespie algorithm [6] numerical simulation of Continuous Time Markov Chain dynamics to generate trajectory samples. Since, we have no knowledge of rate constants we have assumed that all transitions are equally likely to occur and set their rate constants to 1. This implies that we did not interpret Gillespie algorithm time as a real time, with physical time units. We used it exclusively to order the sequence of transitions and to limit the maximal trajectory length. The stochastic Petri net with Gillespie algorithm allowed perturbing the system by adjusting rates and thus altering probability of the occurrence of selected events. This is advantageous over similar approach previously used in biological context [7]. Trajectories were run until the cell death place changed its state or simulation time reached 100 arbitrary time units. For each of the numerical experiments we have run 10000 independent trajectories. The numbers of trajectories reaching proliferation before cell death were expressed as fractions and the 99% binomial probability confidence intervals (99% CI) were calculated to establish significance of the differences between simulation outcomes. The fractions which were outside of their 99% CIs were considered to be significantly different.

Simulations were performed by the extended version of SurreyFBA software [8], which allows statistical model checking in general molecular interaction networks. The Python script has been written to automatically convert Petri Nets in Snoopy file format to simulation software files. Confidence intervals were calculated by binconf() function of Hmisc R package using Wilsons method.

4 Results

We have first run 10000 token game trajectories for the original model. Within the simulation time of 100 arbitrary time units, the proliferation place has been reached before the cell death place in 4779 of trajectories. The 99% CI of the binomial probability was (0.4650524, 0.490777). Subsequently, we have inactivated p53 gene in the model by removing all tokens from the node representing p53 DNA. The number of token game trajectories in which proliferation was observed before cell death was 7223 and the 99% CI was (0.7106193, 0.7336859). Therefore, inactivation of p53 gene resulted in the significant rise of the proliferation. This is in agreement with voluminous experimental data on p53 gene showing that inactivation of p53 results in the increased chances of proliferation and cancer. Motivated by this result we have investigated other perturbations. Simulation of the network where the state of testosterone input node was set to 0 tokens resulted in proliferation probabilities in the 99% confidence interval of (0.2349789, 0.2571581). Therefore, removal of testosterone input resulted in decreased chances of cell proliferation happening before cell death. This result is also in agreement with experimental data. Next, we tested whether our model reproduces the effect of GSK-3B enzyme inhibitors of potential use in cancer chemotherapy [9, 10]. As these drug decrease nuclear stability of an AR-GSK-3B complex and increase (deactivating) nuclear export of the AR, we modeled its effect by increasing the rate of transition removing tokens from the place representing the phosphorylated AR-GSK-3B nuclear complex. In the perturbed system the rate of inactivation transition was set to 1000, while all other transition rates were set to 1. The proliferation probability estimated by running 10000 trajectories was in the 99% CI of (0.3255304, 0.349885). Thus, the simulation reproduced the action of the drug. While numerical experiments shown above provide encouraging validation for the model, we have obtained quite disappointing results in simulation of the PTEN gene inactivation. The 99% CI of proliferation probability was (0.4678475, 0.4935781) i.e. there was no significant difference between the number of trajectories reaching proliferation in original wild type model and PTEN knock-out. There are cogent reasons why this may be the case because the network control of PTEN has been shown in another systems biology study to be different in the basal state than in the insulin-stimulated state [10] and so more sophisticated dynamic Petri net modeling is likely to clarify the situation, particularly as the present result is in contradiction to experimental data indicating an important role for loss of PTEN activity in the evolution of prostate cancer. The qualitative balance of PTEN versus PI3K sensitivity is another area that may also require adjustment in our model to fit data from the literature. Thus, although the simplified Petri net model of prostate cancer presented here highlights gaps in current knowledge and the requirement for meticulous incorporation of static and dynamic network behaviour from the literature, the success of this qualitative approach nevertheless demonstrates that it is useful to focus experimental effort on determination of a full set qualitative interactions, before proceeding to difficult quantitative experiments.

5 Conclusions

In this contribution we show preliminary results that demonstrate the applicability of Stochastic Petri nets for qualitative modeling of molecular interaction networks involved in cancer. We discretize molecular activities to 3 levels, set all transition rates to 1 and used Gillespie algorithm simulation to generate a sample of transition sequences leading to alternative biological outcomes of death and proliferation. We show that despite this radical qualitative approximation the method captures the effects of best studied perturbations on these key biological behaviours. The method is computationally efficient and allows accurate determination of probabilities for large-scale models. Our test shows that it is possible to perform qualitative simulations of the numerous large-scale network reconstructions that were so far investigated exclusively by calculation of network connectivity statistics [1, 2], which does not allow prediction of perturbation effects. We believe that Stochastic Petri nets can be used to reconstruct genome scale models of molecular interaction networks and apply them to prediction of the effects of gene polymorphism and pharmacological interventions in complex network diseases such as prostate cancer.

References

1. Calzone, L., Gelay, A., Zinovyev, A., Radvanyi, F., Barillot, E.: A comprehensive modular map of molecular interactions in RB/E2F pathway. Molecular systems biology 4, 173 (2008)
2. Oda, K., Kitano, H.: A comprehensive map of the toll-like receptor signaling network. Molecular systems biology 2, 2006 0015 (2006)
3. Price, N.D., Reed, J.L., Palsson, B.O.: Genome-scale models of microbial cells: evaluating the consequences of constraints. Nature reviews. Microbiology 2, 886-897 (2004)
4. Kohn, K.W.: Molecular interaction map of the mammalian cell cycle control and DNA repair systems. Molecular biology of the cell 10, 2703-2734 (1999)
5. Rohr, C., Marwan, W., Heiner, M.: Snoopy–a unifying Petri net framework to investigate biomolecular networks. Bioinformatics 26, 974-975 (2010)
6. Gillespie, D.T.: Exact stochastic simulation of coupled chemical reactions. J. Phys. Chem 81, 2340-2361 (1977)
7. Ruths, D., Muller, M., Tseng, J.T., Nakhleh, L., Ram, P.T.: The signaling petri net-based simulator: a non-parametric strategy for characterizing the dynamics of cell-specific signaling networks. PLoS computational biology 4, e1000005 (2008)
8. Gevorgyan, A., Bushell, M.E., Avignone-Rossa, C., Kierzek, A.M.: SurreyFBA: a command line tool and graphics user interface for constraint-based modeling of genome-scale metabolic reaction networks. Bioinformatics 27, 433-434 (2011)
9. Schutz, S.V., Schrader, A.J., Zengerling, F., Genze, F., Cronauer, M.V., Schrader, M.: Inhibition of glycogen synthase kinase-3beta counteracts ligand-independent activity of the androgen receptor in castration resistant prostate cancer. PloS one 6, e25341 (2011)
10. Lequieu, J., Chakrabarti, A., Nayak, S., Varner, J.D.: Computational modeling and analysis of insulin induced eukaryotic translation initiation. PLoS computational biology 7, e1002263 (2011)

Author Index