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

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

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

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

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

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

1 Introduction

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

* Corresponding authors

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

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

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

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

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

2 Previous Work

2.1 BioModelKit Framework for Modular Model Composition

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

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

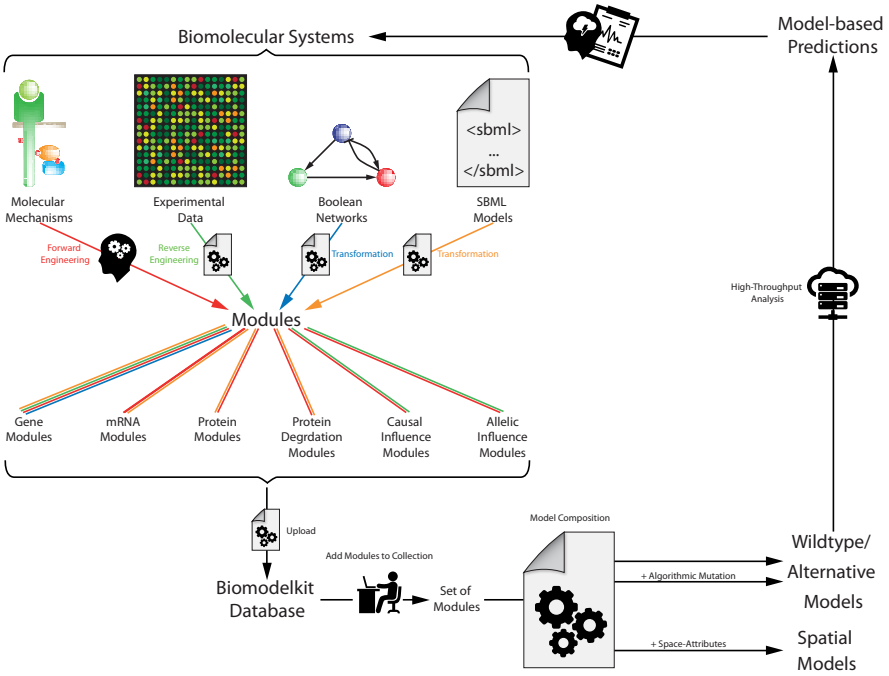


Fig. 1: Overview of the BioModelKit Framework for Modular Model Composition.

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

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

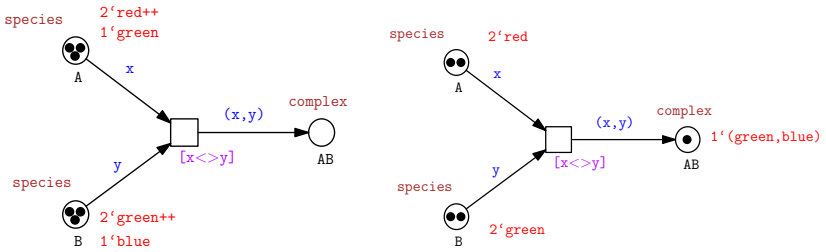
2.2 Coloured Petri Nets

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

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

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

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



(a) Before Firing (b) After Firing

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

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

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

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

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

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

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

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

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

3 Spatial Modelling Methodology for Modular Composed Models

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

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

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

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

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

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

Step 1: Explicit Encoding of Local Positions. The position of each component $C_i \in C^{\mathcal{M}}$ is explicitly encoded by d places $p_1^{C_i}, \ldots p_d^{C_i}$ (termed coordinate places), which can be interpreted as coordinates, where d, $d \geq 1$, defines the number of axes (e.g. 1D, 2D or 3D grid). The marking $m(p_j^{C_i})$ of a place $p_j^{C_i}$ defines the current coordinate value, which must be restricted by a lower $m_L(p_j^{C_i})$ and

upper bound $m_U(p_j^{C_i})$ to represent the boundaries of the encoded grid, such that, $m_{L/U}(p_j^{C_i}) > 0$ and $m_L(p_j^{C_i}) < m_U(p_j^{C_i})$.

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

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

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

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

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

introduced $f^{\mathcal{M}}(t^{k_i}_{j,U}, p^{C_h}_j) = 1$ and $f^{\mathcal{M}}_{InhibitorEdge}(p^{C_h}_j, t^{k_i}_{j,U}) = m_U(p^{C_h}_j)$. To ensure that the position of the complex k_i can only be changed if it is actually formed additional read edges for each transition $t^{k_i}_{j,L/U}$ have to be introduced: $f_{ReadEdge}(P^{k_i}_{IS}, t^{k_i}_{j,L/U}) = 1$, where $P^{k_i}_{IS} \subseteq P^{\mathcal{M}}$ and $P^{k_i}_{IS} = \{\forall p \in P^{\mathcal{M}} \mid k_i = g(p)\}$. Furthermore, it must be excluded for each component $C_h \in k_i$ that interacts with other components using a different binding site. Therefore, all co-existing interactions have to be determined $K^{k_i}_{coex} = \{\forall k_j \in K^{\mathcal{M}} : k_i \cap k_j \neq \varnothing \mid k_j \neq k_i\}$. All places representing a complex $k_j \in K^{k_i}_{coex}$ have to be added to each transition $t^{k_i}_{j,L/U}$ using an inhibitory edge: $f_{InhibitoryEdge}(P^{k_i}_{COEX}, t^{k_i}_{j,L/U}) = 1$, where $P^{k_i}_{COEX} = \{\forall p \in P^{\mathcal{M}} : g(p) = k, k \in K^{k_i}_{coex}\}$.
To allow the movement of co-existing complexes which use different interaction sites simultaneously, the above described procedure has to be applied to all possible combination of co-existing complexes, compare Section 4.

For simplicity reasons the firing rate of each transition $t^{k_i}_{j,L/U}$ and $t^{C_h}_{j,L/U}$ is given by Fick's laws of diffusion [18]. Furthermore, we assume equidistant subvolumes with the width and hight $h = 1$ and set all diffusion coefficient to one. Please note, it is straightforward to define the diffusion coefficients more precisely based on experimental results.

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

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

by $all()`m_0(p)$ and the firing rates of each transition $t \in T^{\mathcal{M}}$ $all()`h(t)$, where $all()$ is a function that extracts all instances of a coloured node.

4 Example

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

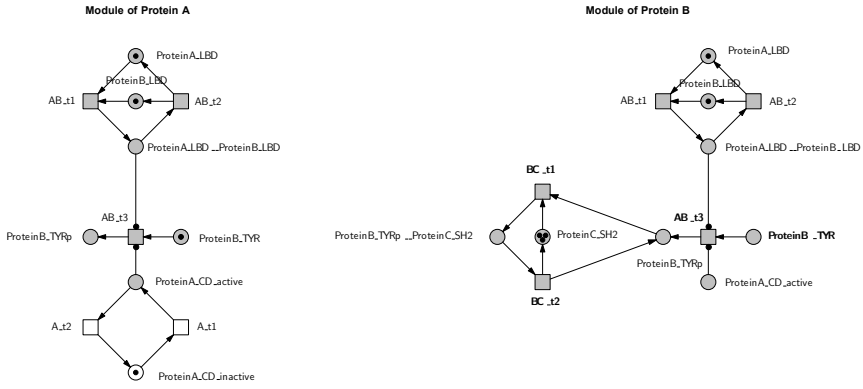


Fig. 3: Running example with two protein modules.

The module of Protein A (M_1) contains two components $\mathbf{C}_1 = \{ ProteinA, ProteinB\}$ and the module of Protein B (M_2) contains three

components $\mathbf{C}_2 = \{ProteinA, ProteinB, ProteinC\}$. For the composed Model $\mathcal{M} = \{M_1, M_2\}$, we get the following mapping of places to the components:

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

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

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

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

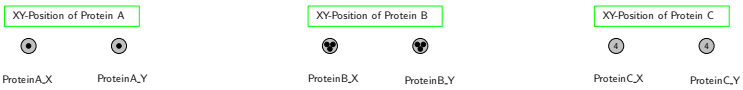


Fig. 4: Encoding of the local positions for each component in the composed modular model.

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

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

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

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

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

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

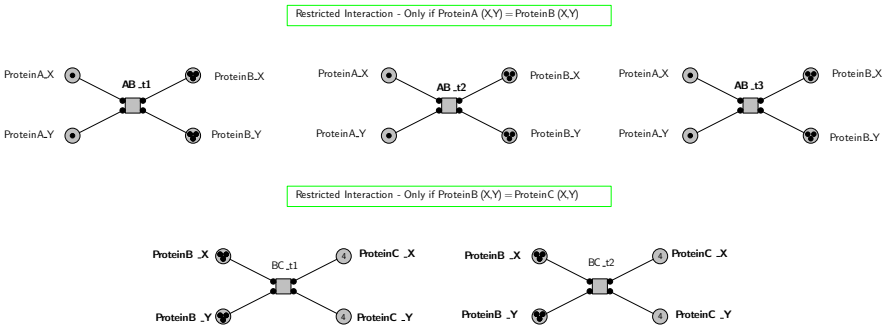


Fig. 5: Restriction of interactions depending on the local position of the involved components.

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

plex of component $ProteinA$, component $ProteinB$ and component $ProteinC$, we need to check if both places $ProteinA\_LBD\_\_\_ProteinB\_LBD$ and $ProteinB\_TYRp\_\_\_ProteinC\_SH2$ are marked.

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

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

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

- csProteinA A
- csProteinB B
- csProteinC C

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

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

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

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

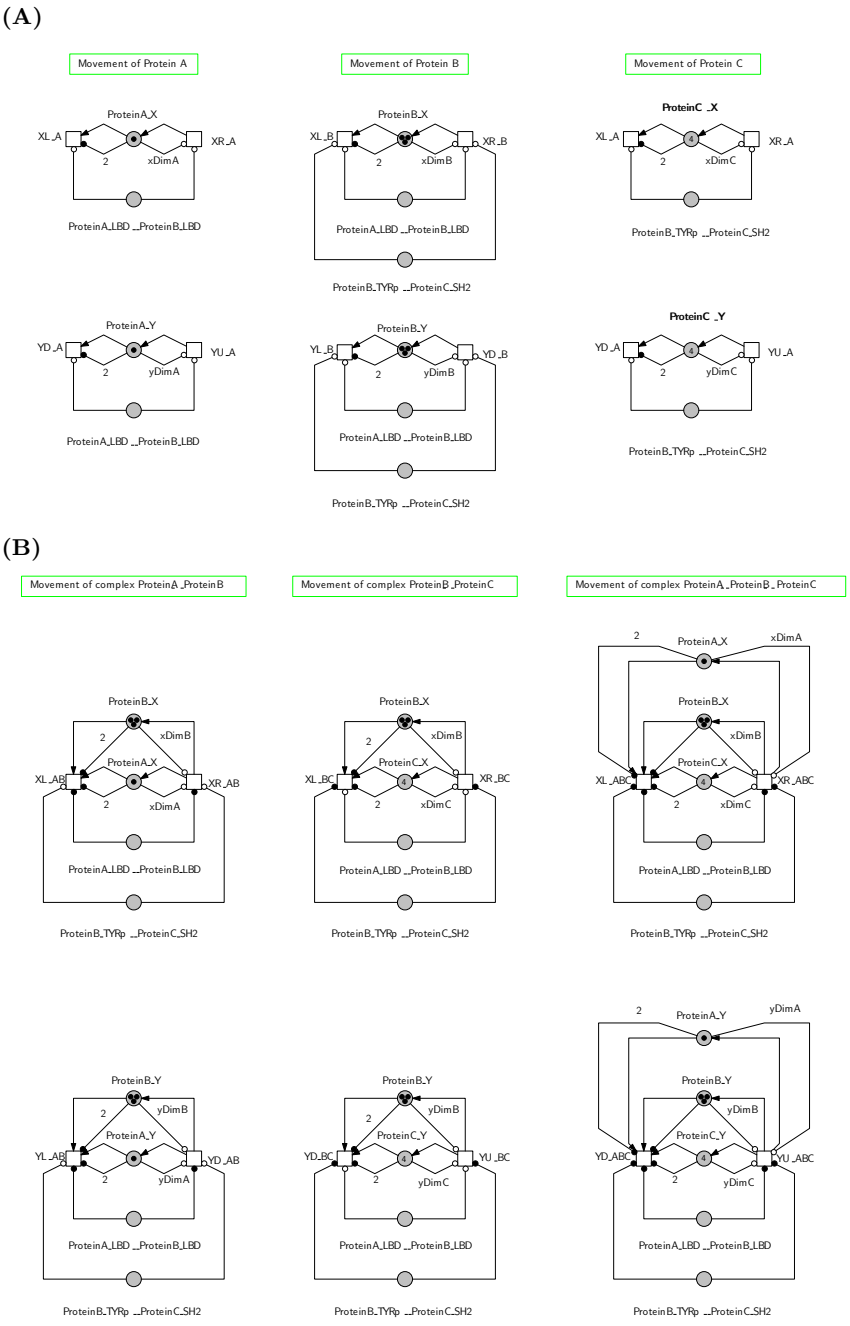
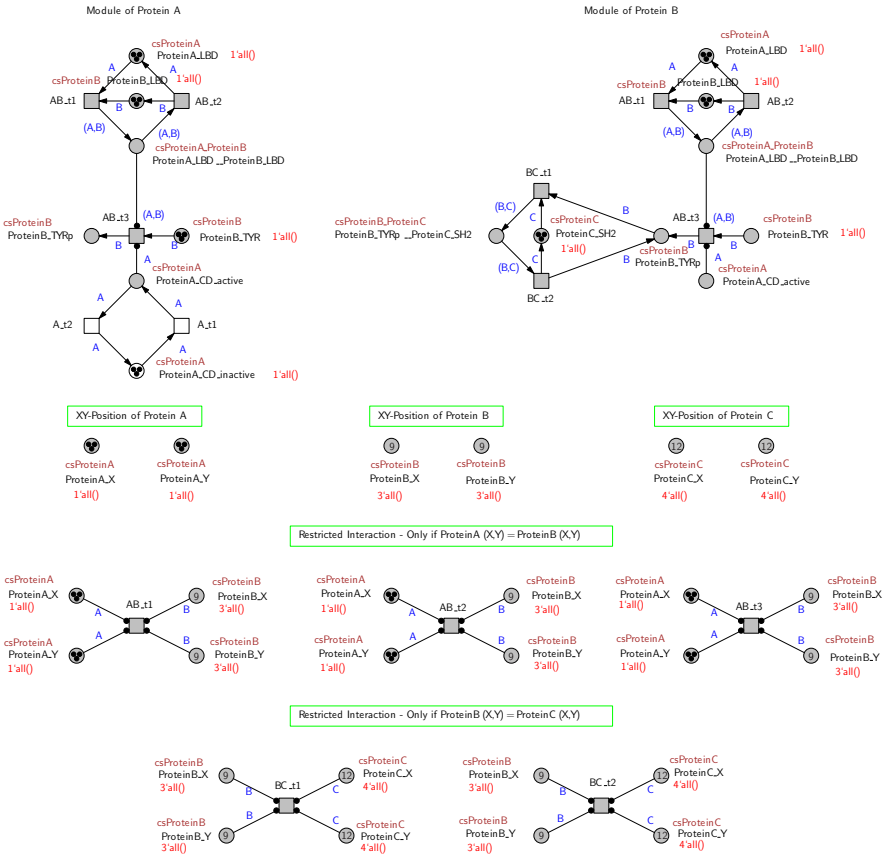


Fig. 6: Local position change dependent on the state of interaction. (A) components are not in complex, (B) components are in complex with each other.



– simple coloursets:
  - int ProteinA := 1 - numA
  - int ProteinB := 1 - numB
  - int ProteinB := 1 - numC
– compound coloursets:
  - product ProteinA_ProteinB := ProteinA, ProteinB
  - product ProteinB_ProteinC := ProteinB, ProteinC
– variables:
  - ProteinA A
  - ProteinB B
– constants:
  - int numA := 3
  - int numB := 3
  - int numC := 3
  - int xDimA := 5
  - int yDimA := 5
  - int xDimB := 5
  - int yDimB := 5
  - int xDimC := 5
  - int yDimC := 5

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

5 Conclusions & Outlook

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

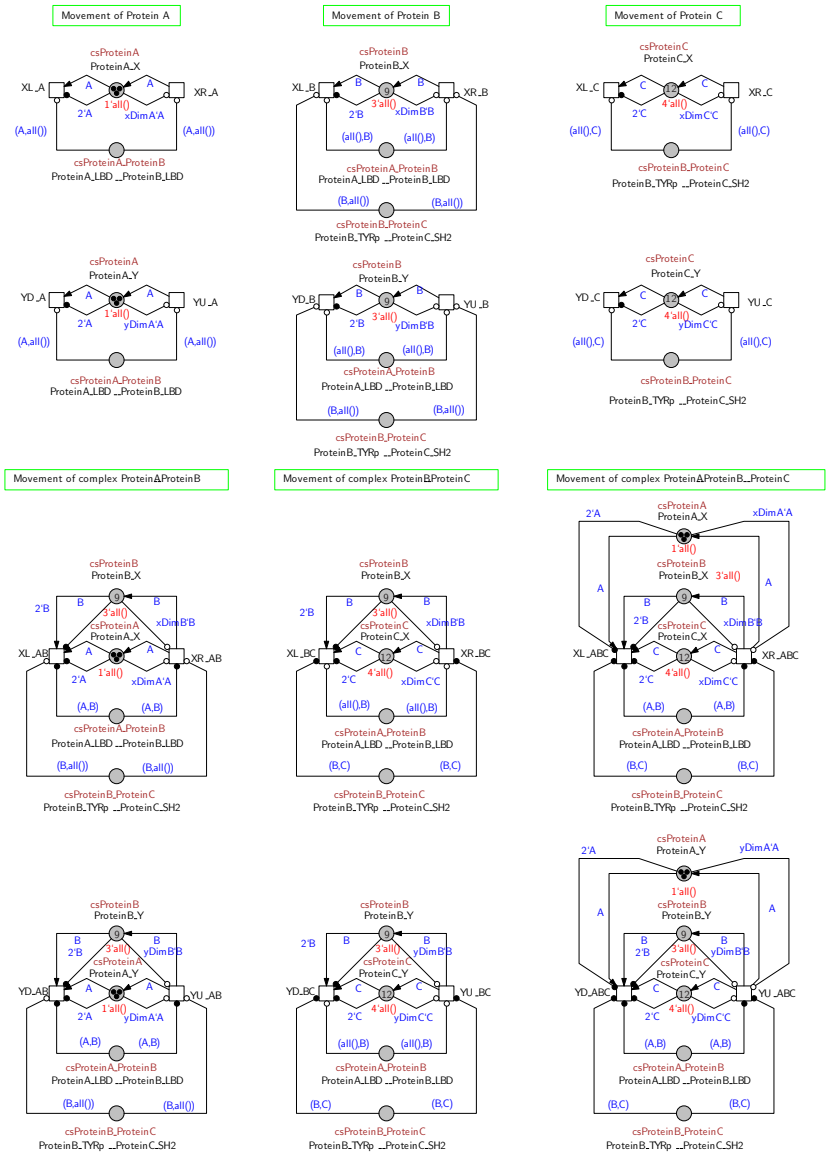


Fig. 8: Part 2: Instantiation of modules using coloured Petri nets.

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

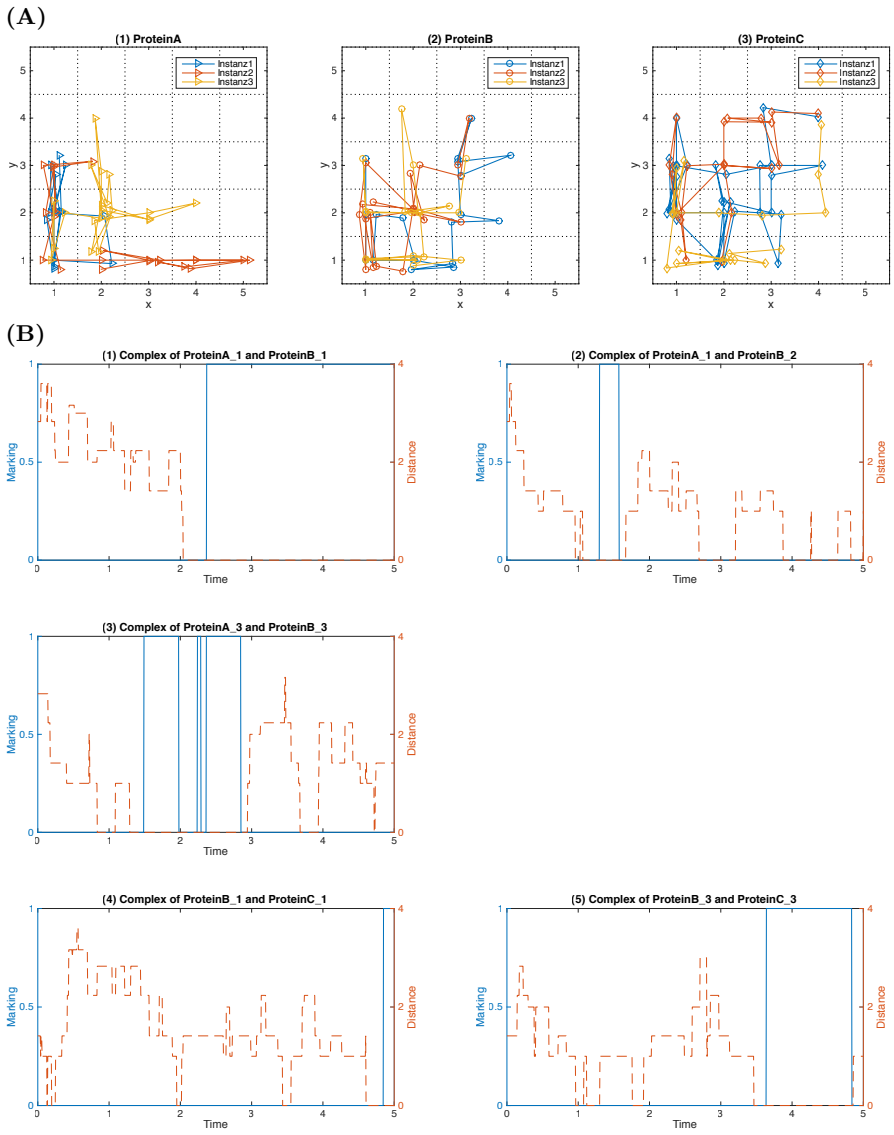


Fig. 9: Stochastic Simulation of the Spatial Model.

ponent in the form of a Petri net. To transform a flat modular composed model into a spatial model the following steps have to be performed: (1) components have to be equipped with individual spatial attributes to represent their localisation and movement in the biomolecular system, (2) components are only allowed to interact if they fulfil certain neighbourhood conditions, (3) the movement of

components depends on their state of interaction, e.g. interacting components forming a complex can only move as one entity. The use of coloured Petri nets in our approach allows us to represent individual numbers of module instances for each component.

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

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

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

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

References

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

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