Colouring Space - A Coloured Framework for Spatial Modelling in Systems Biology

David Gilbert[1], Monika Heiner[2], Fei Liu[3], and Nigel Saunders[1]

[1] School of Information Systems, Computing and Mathematics
Brunel University, Uxbridge, Middlesex UB8 3PH, UK
{david.gilbert,nigel.saunders}@brunel.ac.uk
[2] Computer Science Institute, Brandenburg University of Technology
Postbox 10 13 44, 03013 Cottbus, Germany
monika.heiner@informatik.tu-cottbus.de
[3] Harbin Institute of Technology
West Dazhi Street 92, 150001 Harbin, China
liufei@hit.edu.cn

Abstract. In this paper we introduce a technique to encode spatial attributes of dynamic systems using coloured Petri nets and show how it can be applied to biological systems within the spirit of BioModel Engineering. Our approach can be equally applied to qualitative, stochastic, continuous or hybrid models of the same physical system, and can be used as the basis for multiscale modelling. We illustrate our approach with two case studies, one from the continuous and one from the stochastic paradigm. In this paper we only discuss the case of finite colours, and by unfolding our method can take advantage of all the analytical machinery and simulation techniques that have been developed for the uncoloured family of Petri net classes.

Keywords: Coloured Petri nets, qualitative, stochastic, continuous, hybrid Petri nets, spatial modelling, biomolecular networks, Systems Biology, BioModel Engineering.

1 The Coloured Framework

In this paper we build on [16,20], where we have introduced our methodology for the use of a structured family of Petri net classes which enables the investigation of biological systems using various complementary modelling abstractions comprising the qualitative and quantitative paradigms. In the following we focus on the use of the coloured subset of the previously introduced framework [20] – coloured qualitative Petri nets ($\mathcal{QPN}^{\mathcal{C}}$), coloured stochastic Petri nets ($\mathcal{SPN}^{\mathcal{C}}$), coloured continuous Petri nets ($\mathcal{CPN}^{\mathcal{C}}$), and coloured hybrid Petri nets ($\mathcal{HPN}^{\mathcal{C}}$); Fig. 1 recalls our coloured framework.

We extend our approach by considering biochemical processes evolving in space, which we illustrate with two case studies. In our spatial modelling approach we discretise space using coloured Petri nets, and in this paper we investigate the use of finite discrete colour sets. This ensures the following three

J.-M. Colom and J. Desel (Eds.): PETRI NETS 2013, LNCS 7927, pp. 230–249, 2013.

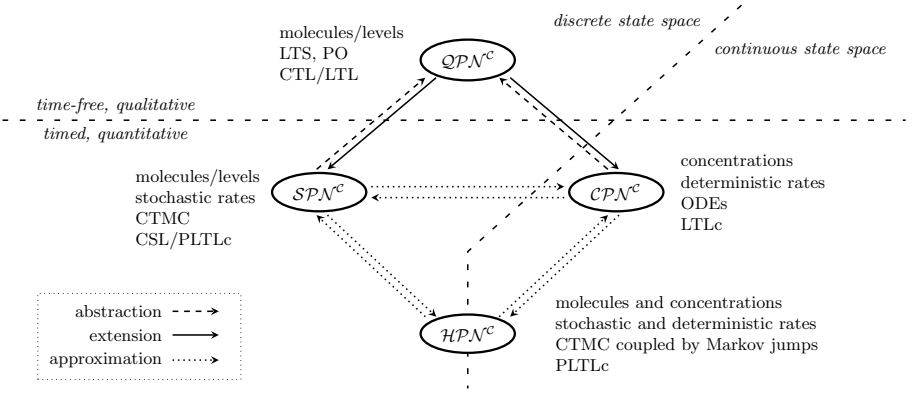


Fig. 1. The coloured unifying framework integrating four degrees of abstraction

features which are crucial for our BioModel Engineering principles (uniformity, reuse and conciseness) [17].

First and most importantly, the spatial modelling principle can be equally applied to all paradigms (qualitative, stochastic, continuous, and hybrid), i.e., once a Petri net model has been enriched with colour-encoded space, it can be easily transformed into any other net class while preserving all spatial attributes.

Second, all space-related information is encoded in colour and corresponding net annotations, such as colour sets, functions, and guards, which can be effortlessly reused in many models. Moreover, changing the notion of space or just some spatial attributes only requires the adaptation of those colour-related definitions, and the net structure itself needs not to be touched.

Third, the use of *a priori* finitely discretised space preserves the analysibility of the models, in particular we retain the discrete state space in both the qualitative and stochastic settings. All analysis and simulation techniques, which have been developed for uncoloured Petri nets over the last two decades, can be immediately reused by automatic unfolding.

The main contributions of our paper are

- a framework to encode space by coloured Petri nets, which can be equally applied in a qualitative, stochastic, continuous, or hybrid setting,
- a set of basic colour-related definitions which can be easily applied to a wide range of spatial scenarios,
- two substantial biological case studies illustrating the framework.

This paper is organised as follows. In the next section we recall some related work to set the background of our contribution. Afterwards we introduce our modelling approach of colour-encoded space by means of a popular case study in the continuous paradigm (Section 3), before applying it to a second case study in the stochastic paradigm (Section 4). We conclude our paper with a brief overview of the tools used and the summary.

2 Related Work

In the following we assume basic knowledge of the Petri net terminology; see [19] for an introduction and formal definitions in the biochemical context.

Existing Uses of Petri Nets in Systems Biology. Petri nets are a natural and established notation for describing reaction networks – both share the bipartite property. Petri nets enjoy a formal semantics and are particularly attractive to biologists, because they can 'buy in' to the executable representation. The intuitive visualisation is complemented by a rich set of sophisticated analysis techniques, supported by reliable tools. Petri nets can serve as an umbrella formalism comprising a family of related (qualitative, stochastic, continuous, hybrid) models, sharing structure, but differing in their kinetics [16].

A recent survey [2] has shown how Petri nets can be applied to various types of biological processes at different abstraction levels, illustrating this with a rich set of case studies. Most of these focus on the molecular level; however examples at the multi-cellular level include the signal-response behaviour of an organism [28], and developmental processes in multi-cellular pattern formation [7, 9, 23].

Current Challenges to Systems Biology Due to Complexity and Multiscale Issues. A drawback of current modelling approaches, including Petri nets, are their limitation to relatively small networks. Biological systems can be represented as networks which themselves typically contain regular (network) structures, and/or repeated occurrences of network patterns. This organisation occurs in a hierarchical manner, reflecting the physical and spatial organisation of the organism. Thus a further challenge is to represent the structure inherent in biological systems.

Coloured Petri Nets are high-level nets and a well-established modelling formalism. They have been used for over 20 years for the specification and analysis of communication protocols, distributed systems, automated production systems, work flows, and VLSI chip design [22]. They allow the description of similar network structures in a concise and well-founded way, providing a flexible template mechanism for network designers, and their combination with hierarchical structuring mechanisms is extremely powerful [21].

In coloured Petri nets, tokens can be distinguished via their colours. This allows for the discrimination of species (molecules, metabolites, proteins, secondary substances, genes, etc.). In addition, colours can be used to distinguish between sub-populations of a species in different locations (cytosol, nucleus, etc.).

Each place is assigned a colour set, specifying the kind of tokens which can reside on the place. A guard is associated with each transition, specifying which coloured tokens are required for firing, and each arc is allocated an inscription specifying the kind of tokens flowing through it. Coloured Petri nets with finite colour sets can be automatically unfolded into uncoloured Petri nets, which then permits the application of all of the existing powerful Petri net analysis and simulation techniques. Vice versa, uncoloured Petri nets can be folded into coloured Petri nets, if partitions of the place and transition sets are given. These partitions of the uncoloured net define the colour sets of the coloured net. As

with hierarchical Petri nets, the conversion between uncoloured and coloured Petri nets changes the style of representation, but does not change the actual net structure of the underlying reaction network.

An attractive advantage of coloured Petri nets is their possibility to easily increase the size of a model consisting of many similar subnets by just adding colours, compare Fig. 2. This permits, e.g., concise representations of the uncoloured multi-cellular models of C. Elegans discussed in [7, 9]. These models consist of six (almost) identical network patterns, one for each cell. In a coloured version, the network pattern can be represented only once and the different cells are reflected in the coloured annotations of the net [23]. Another scenario for deploying colour to simulate a bacterial infection can be found in [8].

Colouring Space. In this paper we deploy colour to specify (biochemical) processes evolving in space. We develop a spatial specification style which can be equally applied in all modelling paradigms. This facilitates smooth movement between the modelling paradigms and the qualitative, stochastic, continuous or hybrid interpretation of the same Petri net. See [23] for more details and formal definitions of the structured family of coloured Petri net classes used in this paper, and [26] for all tool-related features.

In the continuous paradigm, our approach using discretised space corresponds to discretising partial differential equations. An alternative approach to model and solve partial differential equations using (discrete) Petri nets, based on the probably simplest time concept possible for this purpose (maximal steps, maximal auto-concurrency) is discussed in [3, 4]. A more elaborated comparison with other approaches to treat spatial properties is beyond the scope of this paper.

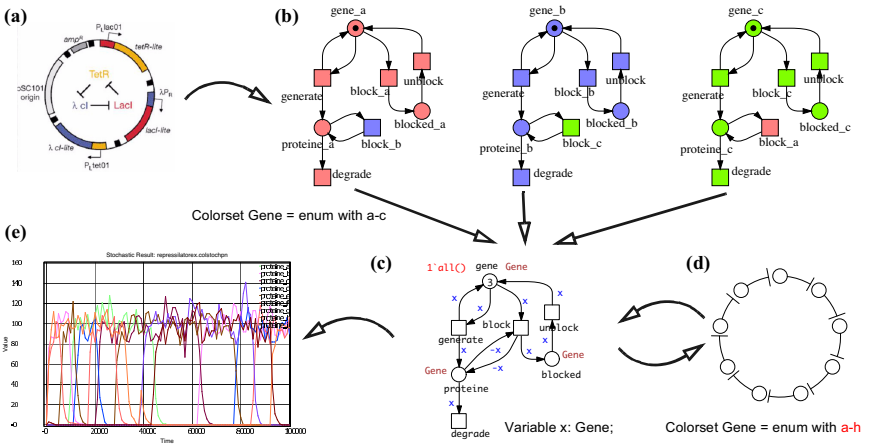


Fig. 2. The repressilator - a genes regulatory cycle [6]. (a) Schematic diagram for three genes. (b) Uncoloured Petri net model for three genes using logical transitions. (c) Folding of similar subnets into a coloured Petri net. (d) Schematic diagram for the generalised repressilator with nine genes. Modelling is accomplished by adjusting the colour set.(e) Stochastic simulation plot of the underlying uncoloured stochastic Petri net. See [20] for the explanation of annotations.

3 Continuous Paradigm

In this section we focus on the continuous part of the framework, illustrating it by means of a case study elaborated over two spatial dimensions.

3.1 Case Study 1: Diffusion

Background. We focus here on diffusion, which is a basic process occurring in biochemical systems with parameters over time and space. It can be regarded as the simplest form of passive mobility. Diffusion goes from regions of higher concentration to regions of lower concentration (Ficks laws) [10] where the diffusion flux is proportional to the minus gradient of concentrations.

Example 1. *One molecular species (here cyclic adenosine monophosphate – cAMP) diffuses continuously in space; i.e., it evolves simultaneously over time and space. The state-dependent diffusion rate follows mass/action kinetics, i.e., the rate is defined by the product of the species involved times some constant, summing up all dependencies on pressure, temperature, etc. The observation shall start with a high concentration (e.g., 100) in the middle of the space, with all other space positions initially set to 0.*

We are going to discuss this example in different scenarios, specifically 1- and 2-dimensional space (1D, 2D), using coloured Petri nets. We use the concept of colour to efficiently represent repeated structures in a continuous Petri net - i.e. to encode repeated elements of a set of ODEs. Each repeated element is associated with a colour, represented by a positive integer; sets of colours are thus discrete and finite. More specifically, we apply colour to represent spatial location; thus in a 1D scenario locations (their addresses) are 1-tuples, in a 2D scenario locations are 2-tuples, and in 3D they are triples.

3.2 Diffusion in One Dimension

We discretise the space and assume an 1-dimensional grid dividing the space into grid positions; see Fig. 3.

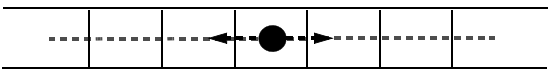


Fig. 3. General scheme of discrete one-dimensional space (1D grid)

A corresponding continuous Petri net is given in Fig. 4 modelling a discrete, 1-dimensional space comprising five grid positions - the five Petri net places $cAMP_i$, while the Petri net transitions model diffusion between neighbouring grid positions. The two outer places stand for the equivalence classes of the boundary space positions and beyond.

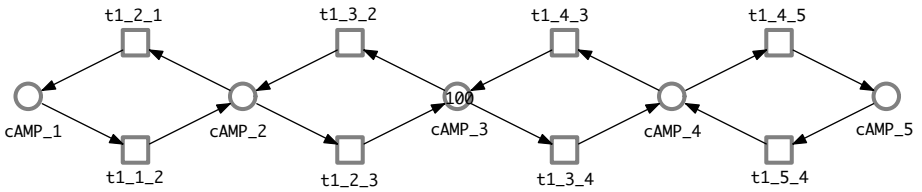


Fig. 4. Continuous Petri net for diffusion in one dimension. The space is discretised into five positions. The value of the middle position is initially set to 100, all other positions to zero, which is the default value, usually not given in graphics.

A continuous Petri net uniquely defines a system of Ordinary Differential Equations (ODEs) [14,31], with one equation for each place (variable). The rates of pre-transitions increase its value, thus defining plus terms, while the rates of post-transitions decrease its value, thus defining minus terms. Denoting the rate of a transition t_j by $v(t_j)$, and the set of pre-transitions (post-transitions) of a place c by ${}^\bullet c$ ($c^\bullet$), we get the generating Equation (1).

$$\frac{dc_i}{dt} = \sum_{t_j \in {}^\bullet c_i} v(t_j) - \sum_{t_j \in c_i{}^\bullet} v(t_j) \tag{1}$$

Assuming the diffusion rates $v(t_j)$ to follow mass/action kinetics with the common rate parameter k, we get the Equations (2)–(6) for the continuous Petri net in Fig. 4; for sake of readability we abbreviate $cAMP_i$ by c_i.

$$\frac{dc_1}{dt} = k \cdot c_2 - k \cdot c_1 \tag{2}$$

$$\frac{dc_2}{dt} = k \cdot c_1 + k \cdot c_3 - 2 \cdot k \cdot c_2 \tag{3}$$

$$\frac{dc_3}{dt} = k \cdot c_2 + k \cdot c_4 - 2 \cdot k \cdot c_3 \tag{4}$$

$$\frac{dc_4}{dt} = k \cdot c_3 + k \cdot c_5 - 2 \cdot k \cdot c_4 \tag{5}$$

$$\frac{dc_5}{dt} = k \cdot c_4 - k \cdot c_5 \tag{6}$$

We obtain a general pattern for an arbitrary, but static size of the discrete, 1-dimensional space by folding the (continuous) Petri net in Fig. 4 into a coloured (continuous) Petri net. For this purpose we introduce the following definitions.

```
const D1 = int with 5;              // grid size
const MIDDLE = int with D1/2+1;
```

```
colorset Grid1D = int with 1–D1;        // grid positions
var x,y : Grid1D;

fun bool neighbour1D(Grid1D x, Grid1D xn) {
   // xn is neighbour of x
   (xn=x–1 | xn=x+1) & (1<=xn) & (xn<=D1) };
```

In this paper we consider finite space. Thus grid positions at the border have fewer neighbours than inner grid positions. We obtain the coloured continuous Petri net given in Fig. 5, where colours serve as addresses in the spatial grid. Changing the grid position of a token now just means recolouring the token.

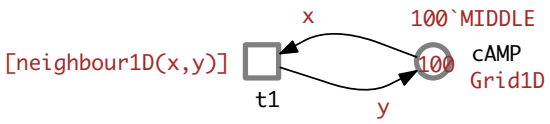


Fig. 5. Coloured continuous Petri net for diffusion in one dimension. The initial marking assigns the value 100 to the MIDDLE grid position, $v(t1) = MassAction(k)$.

Unfolding the coloured Petri net in Fig. 5 with $D1 = 5$ yields exactly the continuous Petri net given in Fig. 4, and thus in turn the ODEs (2)–(6). Changing the constant $D1$ adapts the model pattern to a specific grid size, which permits convenient model scaling, e.g., to increase the spatial resolution.

3.3 Diffusion in Two Dimensions

The generalisation to the 2-dimensional case using a Cartesian grid, see Fig. 6, is rather straightforward. We basically need to extend the definitions required for annotating the coloured Petri net while keeping the Petri net structure as it is.

We start off with a neighbourhood relation where inner grid positions have four neighbours, see Fig. 6(a), which is encoded in the function $neighbour2D4$. The corresponding coloured Petri net is given in Fig. 7(a), and its unfolding for $D1 = D2 = 5$ in Fig. 8. All transitions follow the same kinetic rate pattern.

```
const D1 = int with 5;          // grid size first dimension
const D2 = D1;                  // grid size second dimension
const MIDDLE = int with D1/2+1;

colorset CD1 = int with 1–D1;       // row index
colorset CD2 = int with 1–D2;       // column index
colorset Grid2D = product with CD1 x CD2;       // 2D grid
```

```
var x,a : CD1;
var y,b : CD2;

fun bool neighbour2D4 (CD1 x, CD2 y, CD1 xn, CD2 yn) {
  // (xn,yn) is one of the up to four neighbours of (x,y)
          (xn=x & yn=y-1) | (xn=x & yn=y+1)
   | (yn=y & xn=x-1) | (yn=y & xn=x+1)
   & (1<=xn & xn<=D1) & (1<=yn & yn<=D2) };
```

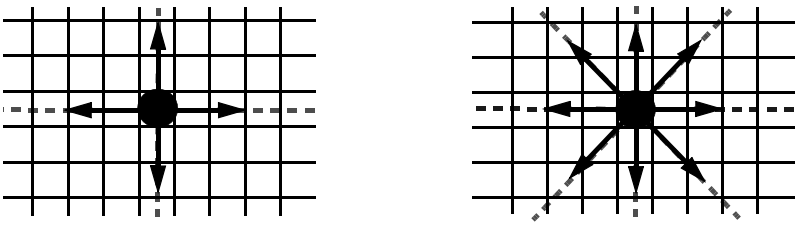
Next we consider a variation of the neighbourhood relation where each inner grid position has eight neighbours; see Fig. 6(b). We introduce three functions.

```
fun bool neighbour2D8(CD1 x,CD2 y,CD1 xn,CD2 yn) {
  // (xn,yn) is one of the up to eight neighbours of (x,y)
          (xn=x-1 | xn=x | xn=x+1 ) & (yn=y-1 | yn=y | yn=y+1)
   & (!(xn=x & yn=y))
   & (1<=xn & xn<=D1) & (1<=yn & yn<=D2) };

fun bool lateral(CD1 x,CD2 y,CD1 xn,CD2 yn) {
          (xn=x & yn=y-1) | (xn=x & yn=y+1)
   | (yn=y & xn=x-1) | (yn=y & xn=x+1) };

fun bool diagonal(CD1 x,CD2 y,CD1 xn,CD2 yn) {
          (xn=x-1 & yn=y-1) | (xn=x+1 & yn=y-1)
   | (xn=x-1 & yn=y+1) | (xn=x+1 & yn=y+1) };
```

The latter two functions are used to appropriately set the rate functions, assuming that it takes longer to reach a diagonal neighbour than a lateral one:

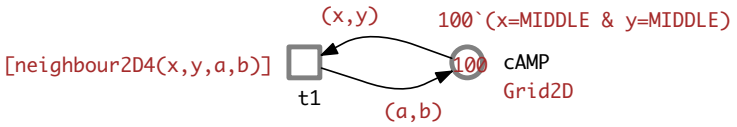
$$v(t1) = \begin{cases} lateral(x,y,a,b) & : \quad MassAction(k) \\ diagonal(x,y,a,b) & : \quad MassAction(k/DIAGONAL) \end{cases}, \tag{7}$$

with DIAGONAL $= \sqrt{2}$. The corresponding coloured Petri net is given in Fig. 7(b), and its unfolding for D1=D2=5 in Fig. 8.

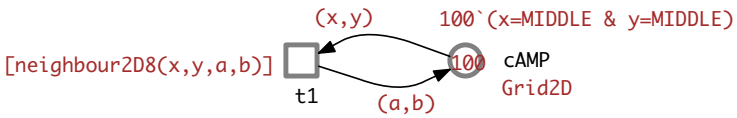


(a) four neighbours (2D4 grid) (b) eight neighbours (2D8 grid)

Fig. 6. General scheme of discrete two-dimensional space with two different neighbourhood relations



(a) four neighbours



(b) eight neighbours

Fig. 7. $\mathcal{CPN}^{\mathcal{C}}$ for diffusion in two dimensions with two different neighbourhood relations. The difference consists of the neighbour function used as transition guard and the rate functions; (a) $v(t1) = MassAction(k)$, (b) see Equation (7).

Remarks: The coloured Petri nets in Fig. 5 and 7 all share the same structure, they differ in their colour-related annotations. It is obvious how to adjust the definitions to other neighbourhood relations.

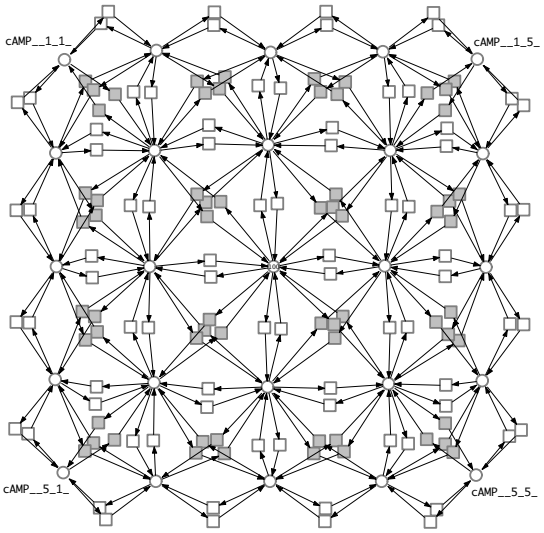


Fig. 8. Continuous Petri nets for diffusion in two dimensions with four neighbours (white transitions only), and eight neighbours (including grey transitions). These Petri nets have been generated by unfolding the two $\mathcal{CPN}^{\mathcal{C}}$ in Fig. 7 with $D1 = D2 = 5$.

3.4 Computational Experiments

For the time being, all computational experiments are undertaken by unfolding coloured Petri nets which is automatically performed in the background, and numerically solving the underlying ODEs which again are generated automatically. Both transformation steps and the continuous simulation itself are features of Snoopy, the tool used in this paper, see also Section 5. In other words, the $\mathcal{CPN}^{\mathcal{C}}$ serve as a kind of very high-level description of ODEs. Note that the mapping from $\mathcal{CPN}^{\mathcal{C}}$ to ODEs is unique but not vice-versa [31].

The key challenge when unfolding coloured Petri nets is to compute all transition instances, which suffers from combinatorial explosion. However, when the number of transition instances is only determined by guards (logical expressions), which is the case in our scenario, a constraint satisfaction approach [32] can be employed. As each coloured transition can be considered separately, the unfolding can be easily parallelised by multiple threads to take advantage of state-of-the-art multi-core computer architectures. We have used the efficient search strategies of *Gecode* [12] to substantially improve the unfolding efficiency of coloured Petri nets; for more details see [23, 27].

The unfolding of any $\mathcal{CPN}^{\mathcal{C}}$ version of our gradient example yields extremely large continuous Petri nets. It is easy to see that in the 2-dimensional case the number of places always equals $D1D2$, while the number of transitions amounts to $4D1D2 - 2(D1 + D2)$ for a 2D4 grid, and $8D1D2 - 6(D1 + D2) + 4$ for a 2D8 grid, respectively. To give an example, the unfolding of an 120×120 2D4 grid (used in Fig. 10, last row) generates 14,400 places and 57,120 transitions, with an unfolding time of about 25 seconds (on a standard laptop computer). The generated continuous Petri net in turn is transformed into ODEs according to formula (1), i.e., the number of places determines the number of ordinary differential equations, and the number of transitions the total number of terms in these equations to be simulated.

The actual simulation, i.e., the numerical integration of the generated ODEs takes a couple of seconds and yields time traces for each unfolded place, see Fig. 9. These traces are converted into heat maps, one for each time step, i.e., a sequence of heat maps eventually visualises the evolution in time and space, see Fig. 10.

We performed a couple of experiments to test the scalability of our model. Model scaling also usually requires adjustments of the initial marking and rate parameters. To maximise the flexibility of our model we slightly changed the specification style of the initial marking. We introduced a couple of constants (including LB – lower bound, UB – upper bound to specify a rectangle) which eventually permit the specification of the range of grid positions set to 100 in the initial marking in a better adjustable manner:

```
100 '((LB <=x & x <= UB) & (LB <=y & y <= UB)) .
```

To reach equivalent states in the same simulation time, we need to scale the parameters by the square of the resolution factor; see Fig. 10 for some results.

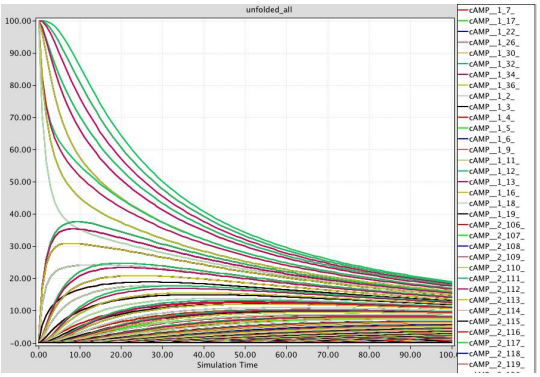


Fig. 9. Simulation plot of the ODEs generated from a $\mathcal{CPN}^{\mathcal{C}}$, illustrating approaching to the future steady state where all concentrations will be equal

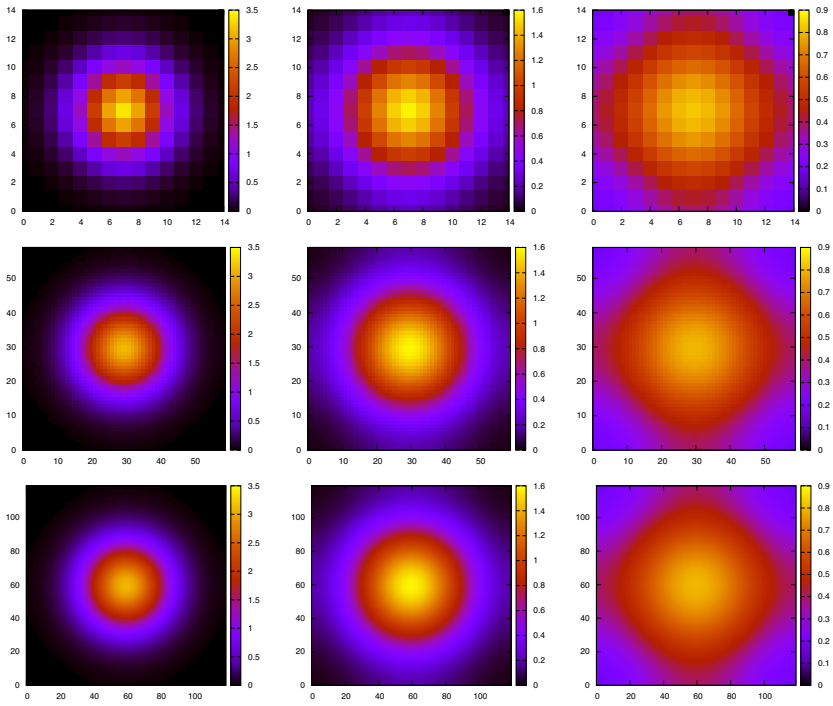


Fig. 10. Continuous simulation results for diffusion in two dimensions with four neighbours in space resolutions 15×15, 60×60, and 120×120. The three snapshots given for each resolution are taken at simulation time 25 (left), 50 (middle), and 100 (right column).

4 Stochastic Paradigm

In this section we focus on the stochastic part of the framework, reusing the colour definitions which we introduced in the previous section. Diffusion can be treated stochastically using the laws of Brownian motion, for example embodied in the Gilllespie algorithm [15]. However, the mapping is straightforward and instead we present a more sophisticated and challenging biological example.

4.1 Case Study 2: Bacterial Colony Growth – Phase Variation

We study phase variation in bacterial cell colonies which grow in space. We developed a Coloured Stochastic Petri Net which allows us to substantially extend the method applied in [30] to computationally predict the sector-like patterning characteristic of such colonies, see e.g., Figure 1 in [1].

Background. A common microbial stochastic mechanism is phase variation, in which gene expression is controlled by a reversible genetic mutation, rearrangement, or modification. Phase variation has traditionally been considered in the context of 'contingency genes' in which a sub-population is continuously generated which is pre-adapted to repeated environmental transitions, often to immune selective changes. However recent re-consideration, in the light of stochastic processes in genes under other forms of regulation, suggests an important potential role in bacterial specialization and differentiation, and the generation of structured bacterial populations.

Example 2. *We consider a colony of bacteria with two phenotypes A and B, which develop over time by cell division. Cell division may involve cell mutation, and back-mutation alternates phenotypes; see Fig. 11. The observation should start with one bacterium of phenotype A. We are interested in the proportion of phenotypes in the cell generations, and how their spatial distribution evolves over time.*

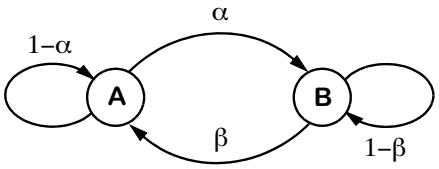


Fig. 11. Phase variation within bacterial colonies - basic scheme. Mutation from A to B happens at rate α, and backward mutation at rate β.

4.2 Step-Wise Modelling

In the following we describe the step-wise approach which we have employed to construct our $\mathcal{SPN}^{\mathcal{C}}$ model. We start off with a basic model of phase variation between two states in bacterial colonies as discussed in [30] which did not model spatial aspects, and encode this as a stochastic Petri net. Next we enrich the basic model with a 2D8 grid, where the parent remains in-situ, and the child is

displaced by one grid position. Finally, we refine our model by controlling colony spreading and thickness. Our stochastic spatial model permits us to describe the development of sector-like pattering typical of phase variation in bacterial colonies.

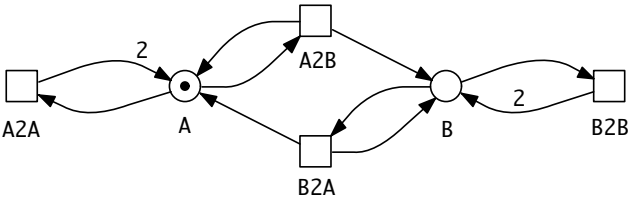
Step 1 – Basic Model of Phase Variation. We start with the equations taken from the previous *deterministic* model of phase variation [30], which describe *synchronous* growth in cell colonies with two phenotypes A and B, modelled here by two corresponding variables indexed by the discrete time steps. These equations include the assumption that "if phase variation occurs, the progeny consists of one A and one B."

$$A_{n+1} = 2d_A(1-\alpha)A_n + d_A\alpha A_n + d_B\beta B_n \tag{8}$$

$$B_{n+1} = 2d_B(1-\beta)B_n + d_B\beta B_n + d_A\alpha A_n \tag{9}$$

Here, d_A and d_B specify the fitness, i.e., the proportions of A or B, respectively, that survive to division.

Previously [30], behaviour was explored by iterating the equations on a spreadsheet. We develop a Petri net model that is directly executable by playing the token game which facilitates its comprehension, and permits the exploration of the behaviour by standard analysis and simulation techniques. Our initial stochastic Petri net, see Fig. 12, corresponds to Equations (8)–(9), but adopts an *asynchronous* modelling approach so that cells divide individually.



$v(A2B) = d_A\alpha A$, $v(A2A) = d_A(1-\alpha)A$
$v(B2A) = d_B\beta B$, $v(B2B) = d_B(1-\beta)B$

Fig. 12. Stochastic Petri net ($\mathcal{SPN}$) corresponding to Equations (8)–(9)

Model parameters (taken from [30])

- *mutation rates* α (forward), and β (backward): in the range of $10^{-2} - 10^{-5}$; e.g. *high*: $\alpha = \beta = 0.0025$, *medium*: $\alpha = \beta = 0.0005$, *low*: $\alpha = \beta = 0.00005$; α and β could also take different values;
- *relative fitness* f – ratio of phenotype survival probability: $f = d_A/d_B$; typical values: f = 1.0 (no fitness difference), 0.99, 0.9, 0.5.

Derived Measures of Interest

- *Total number of bacteria.* The n-th generation in a synchronous model yields 2^n bacteria. Vice versa, if we know the total number *total* of bacteria generated by asynchronous cell division, then we can obtain the corresponding synchronous generation counter n by

$$n = \log_2 total \tag{10}$$

  For example, 26 synchronous generations (which may develop in about 24 hours) end up with a total population size of approximately $67 \cdot 10^6$.
- *Proportion of A and B.*

$$propA = \frac{A}{A+B}; \quad propB = \frac{B}{A+B} \tag{11}$$

Simulating the stochastic model allows us to observe asynchronous population growth such that cells divide individually. Each event (firing of a transition) corresponds to the division of one cell. Consequently, the size of the population will grow in steps by 1, see Fig. 13, in contrast with the synchronous model. Depending on the setting for the output steps of the simulator we may not be able to observe all events in the simulation trace.

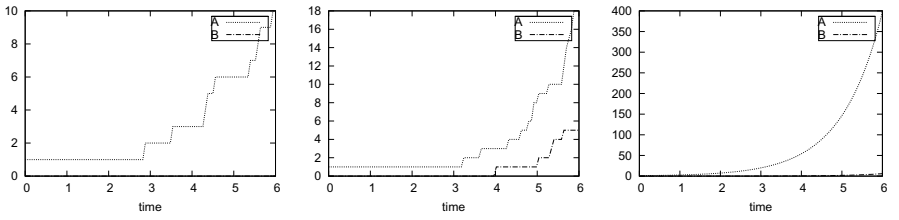


Fig. 13. Two single stochastic simulation runs, and one continuous run; $\alpha = \beta = 0.0025$, $d_A = d_B = 1$, i.e., no fitness advantage

Observations

- Keeping a relative fitness of 1 while extending the simulation time allows us to observe that the variables A and B will finally be almost identical, meaning their proportions will finally approach 50%.
- Likewise, keeping the mutation rates equal and giving one mutant a fitness advantage over the other, e.g. using a fitness ratio of 0.9, then the mutants with the greater fitness will finally outnumber the mutants with lower fitness and the proportion of the latter ones in the total population approaches zero over time.

Starting from simulation traces like the ones given in Fig. 13, all diagrams presented and discussed in [30] can be derived by some post-processing.

To prepare for the modelling of cell colonies in space we fold our first (uncoloured) Petri net. For this purpose we introduce two colour sets, $Phenotype = \{a, b\}$, and $DivisionType = \{replicate, mutate\}$. These definitions allow us to fold the two places A and B into one coloured place *cell* with the colour set *Phenotype*, and to fold the four transitions into the coloured transition *division*. We obtain the basic model given in Fig. 14.

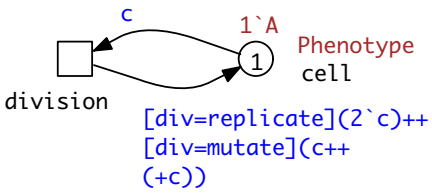


Fig. 14. $\mathcal{SPN}^{\mathcal{C}}$ as $\mathcal{SPN}$ short-hand notation; unfolding this $\mathcal{SPN}^{\mathcal{C}}$ generates the $\mathcal{SPN}$ in Fig. 12. See listing in Fig 15 for the related definitions.

The derivation of our final model, see Fig. 15, from the basic model, see Fig. 14, requires three further steps: adding space, controlling colony spreading, and controlling thickness. We deliberately ignore some complexities, e.g. nutrition and oxygen which are responsible for the vertical structure of the bacterial colony, to design a simple, but powerful model.

Step 2 – Adding Space. We assume that the 3D colony is represented by a 2D grid with a finite capacity on each grid position, and there is an equal maximal height over all of the cell colony, i.e., all grid positions have the same capacity. We derive a colour set from the cross product of the *Grid2D* and *Phenotype* colour sets. Adding space requires making a decision regarding the destination of the offspring. Initially, we assume that the offspring always goes to one of the neighbouring positions which is chosen stochastically.

In this case study we are concerned with mutation rates and their influence on the system behaviour. So their total values have to be kept constant. Introducing space means technically to multiply transitions (basically one for each direction per grid position). To counterbalance this effect, we scale the transition rates by dividing them by the number of grid positions and by N, with N being the number of neighbours. With other words, all transitions (which we get by unfolding) make four equivalence classes, and the sum of all rates in one equivalence class is kept constant, independent of the grid size and the neighbourhood relation. Thus, the total rates in the phase variation model with space are the same as in the phase variation model without space.

Step 3 – Controlling Colony Spreading. Cells do not actively move; as a result of cell division they can either pile up on the parent's grid position or be displaced to a neighbouring position. To model this phenomenon, we add an alternative transition which allows an offspring to stay with its parent. Thus, the rate functions need now to be scaled by N+1.

To control the tendency between staying with the parent (*division*1) or going to a neighbouring position (*division*2), we introduce a preference factor γ, which may vary between 0 and $\gamma_{\max}$ without changing the total division rate (sum of rates of *division*1 and *division*2). For this purpose, we define $\gamma_H = \gamma/\gamma_{\max}$, and $\gamma_N = (\gamma_{\max} - \gamma)/\gamma_{\max}$ to further scale the rate functions correspondingly.

Increasing γ increases the preference to stay with the parent, while decreasing γ increases the preference to displace. Setting γ to $\gamma_{\max}$ precludes the ability to go to a neighbour, thus the size of the colony is restricted by the capacity of one grid position. Setting γ to zero precludes staying with the parent. Cells then have the tendency to first occupy all grid positions, before the thickness increases simultaneously over the whole colony patch.

Step 4 – Controlling Thickness. The bacteria generated by cell division can pile up on top of each other and thus increase the colony thickness at that grid position. This thickness is limited because of the cells' requirements for access to oxygen and nutrients. In order to control the thickness we introduce a constant POOLSIZE, which limits the maximum number of cells at a certain grid position. We set POOLSIZE to give room for 26 generations. See the listing in Fig. 15 for a summary of all required colour-related definitions.

4.3 Computational Experiments

All computational experiments are done on the automatically unfolded Petri nets. Unfolding our coloured Petri net for a 101×101 grid yields an uncoloured Petri net with 30,603 places and 362,404 transitions with an unfolding time of 630 seconds. The unfolded Petri net is simulated using the Gillespie algorithm [15]. One stochastic simulation run takes about 40 minutes. The output comprises a pair of simulation traces for each grid position, corresponding to the two phenotypes A and B, similar to Fig. 9, with each run behaving differently.

The analysis considers the development over time of the proportion of the given genotype in the total population, and the patterning into characteristic segments. This requires converting the stochastic simulations into 2D representations, see Fig. 16, and analysing the development of the 2D sector-like patterns over time. We expect that the model will finally permit the prediction of mutation rates and fitness by counting and measuring pattern segments, which in the future could give new insights into the population dynamics of mutation. Currently, our model predicts behavior which has not been measured so far in the wet lab — the model generates a time series description of the evolution of the patterns in cell colony (indicated in Fig. 16), while wet lab data just give a snapshot of the final state.

5 Tools

All Petri nets in this paper were constructed with Snoopy [29], recently extended to support coloured Petri nets [20, 23]. Simulations were done with Snoopy's built-in stochastic simulator and Marcie [18]. Simulation traces have been further

```
const D1 = int with 101;
const D2 = D1;
const MIDDLE = int with D1/2+1;
const POOLSIZE = int with 7000;
const POOLSIZE_1 = int with POOLSIZE-1;

colorset Phenotype = enum with A, B;
colorset DivisionType = enum with replicate, mutate;
colorset CD1 = int with 1-D1;
colorset CD2 = int with 1-D2;

colorset Grid2D = product with CD1 x CD2;
colorset Grid = product with Grid2D x Phenotype;

var c : Phenotype;
var div : DivisionType;
var x,xn : CD1;
var y,yn : CD2;

fun bool neighbour2D8(CD1 x,CD2 y,CD1 xn,CD2 yn) { ... };
fun bool lateral(CD1 x,CD2 y,CD1 xn,CD2 yn) { ... };
fun bool diagonal(CD1 x,CD2 y,CD1 xn,CD2 yn) { ... };
```

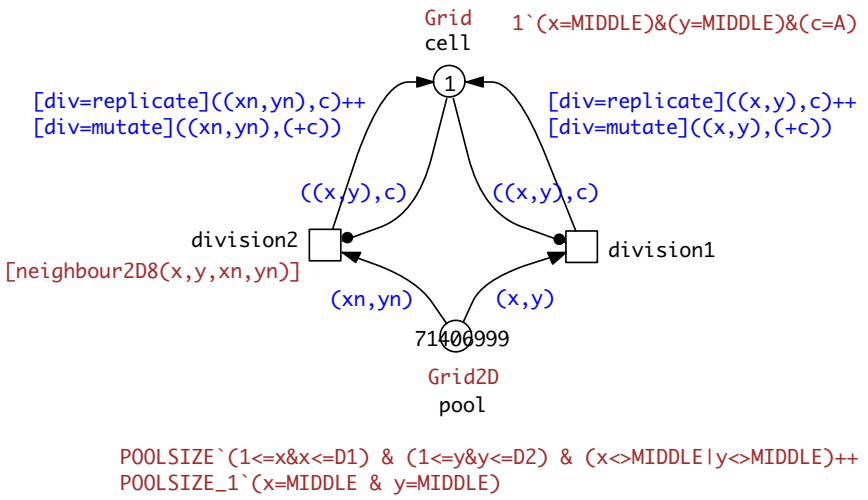


Fig. 15. $\mathcal{SPN}^{\mathcal{C}}$ for the final spatial model of phase variation. The integers on the places give the total number of tokens of any colour.

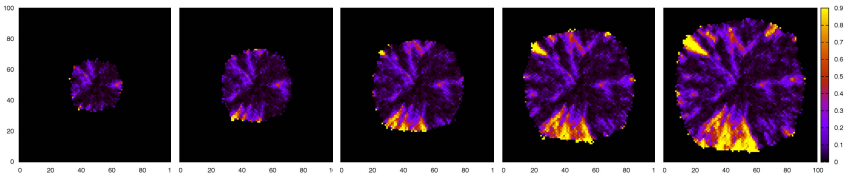


Fig. 16. 2D representation of a single stochastic run showing the development of binary phase variation in a cell colony over time and space, and illustrating the development of sector-like patterns. Density of the two phenotypes is represented by yellow and dark blue, respectively. Each run looks differently due to the built-in stochasticity.

processed by customised Java (Python) programs, and finally visualised with Gnuplot (matplotlib). Snoopy and the models in Snoopy format can be obtained from `http://www-dssz.informatik.tu-cottbus.de`. Thus, all our results can be easily reproduced by the interested reader.

6 Summary

In this paper we have deployed colour to specify biochemical processes evolving in time and space. The spatial modelling style presented can be applied to a wide range of biological and also technical application scenarios. The framework we have introduced covers qualitative, stochastic, continuous and hybrid modelling paradigms It exploits existing simulation techniques and analytical machinery by unfolding to uncoloured nets.

Due to page restrictions we have only presented continuous and stochastic case studies, but it is obvious how to apply our spatial modelling approach to qualitative and hybrid examples. It is also straightforward how to extend the colouring principle to a 3-dimensional space, or how to adapt it to different notions of space; e.g., using polar coordinates.

The coloured Petri nets which we have presented might give an impression of simplicity, which just underlines the power of abstraction by folding into coloured models. Crucially, this technique enables a new approach to multiscale modelling, and we have elsewhere illustrated this by using coloured stochastic and continuous Petri nets to model planar cell polarity in Drosophila fly wing [11, 13,17]. A 2-dimensional space is organised as a regular honeycomb lattice of cells which is interpreted over a regular grid by tuning the neighbourhood functions. Each position in the grid contains a subgrid describing the intracellular level. Further case studies deploying coloured Petri nets for spatial modelling problems can be found in [5, 13, 24, 25].

Our modelling style supports BioModel engineering by the established *separation of concerns* principle. Changing the notion of space just requires the appropriate adaptation of the definition of the colour sets, the functions specifying the neighbourhood relation, and the transition rate functions. The net

structure itself needs not to be altered. All colour-related definitions can be reused via Snoopy's export/import functionality.

Our current ongoing work includes the development of visualisation and model checking over spatial patterns in multiple dimensions and scales, as well as non-rectangular geometries. Future work will address computational challenges due to the fact that currently simulations must be performed at the unfolded level rather than at the coloured level.

Acknowledgments. This research has been partially funded by the British EPSRC Research Grant EP I036168/1 and the German BMBF Research Grant 0315449H. The computational experiments undertaken in this project would not have been possible without the constant support by Mostafa Herajy, Martin Schwarick, and Christian Rohr. We acknowledge as well the support by Mary Ann Blätke and Jan Wegener in producing some of the data plots.

References

1. Appelmelka, B.J., Monteirob, M.A., Martinc, S.L., Morand, A.P., Vandenbroucke-Graulsa, C.M.: Why Helicobacter pylori has Lewis antigens. Trends in Microbiology 8(12), 565–570 (2000)
2. Baldan, P., Cocco, N., Marin, A., Simeoni, M.: Petri Nets for Modelling Metabolic Pathways: a Survey. J. Natural Computing (9), 955–989 (2010)
3. Bertens, L.: Computerised modelling for developmental biology: an exploration with case studies. Ph.D. thesis, Leiden University (September 2012)
4. Bertens, L., Kleijn, J., Hille, S., Koutny, M., Heiner, M., Verbeek, F.: Modeling biological gradient formation: combining partial differential equations and Petri nets. Tech. Rep. CS-TR-1379, Univ. of Newcastle upon Tyne, School of CS (2013)
5. Blätke, M., 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 BioSystem (2013)
6. Blossey, R., Cardelli, L., Phillips, A.: Compositionality, Stochasticity and Cooperativity in Dynamic Models of Gene Regulation. HFSP Journal 1(2), 17–28 (2008)
7. Bonzanni, N., Krepska, E., Feenstra, K., Fokkink, W., Kielmann, T., Bal, H., Heringa, J.: Executing multicellular differentiation: quantitative predictive modelling of c. elegans vulval development. Bioinformatics 25, 2049–2056 (2009)
8. Carvalho, R., Kleijn, J., Meijer, A., Verbeek, F.: Modeling Innate Immune Response to Early Mycobacterium Infection. Computational and Mathematical Methods in Medicine 2012, Article ID 790482 (2012)
9. Chen, L., Masao, N., Kazuko, U., Satoru, M.: Simulation-based model checking approach to cell fate specification during C. Elegans vulval development by hybrid functional Petri net with extension. BMC Systems Biology 42, 3 (2009)
10. Fick, A.: Über Diffusion (in German). Annalen der Physik 170(1), 59–86 (1855)
11. 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 99(PrePrints) (2012)
12. Gecode: Gecode: An open constraint solving library (2011), `http://www.gecode.org`

13. Gilbert, D., Heiner, M.: Petri nets for multiscale Systems Biology. Brunel University, Uxbridge (2011), http://multiscalepn.brunel.ac.uk/
14. Gilbert, D., Heiner, M.: From Petri nets to differential equations - an integrative approach for biochemical network analysis. In: Donatelli, S., Thiagarajan, P.S. (eds.) ICATPN 2006. LNCS, vol. 4024, pp. 181–200. Springer, Heidelberg (2006)
15. Gillespie, D.: Exact stochastic simulation of coupled chemical reactions. The Journal of Physical Chemistry 81(25), 2340–2361 (1977)
16. Heiner, M., Gilbert, D.: How Might Petri Nets Enhance Your Systems Biology Toolkit. In: Kristensen, L.M., Petrucci, L. (eds.) PETRI NETS 2011. LNCS, vol. 6709, pp. 17–37. Springer, Heidelberg (2011)
17. Heiner, M., Gilbert, D.: Biomodel engineering for multiscale systems biology. Progress in Biophysics and Molecular Biology (2012)
18. Heiner, M., Rohr, C., Schwarick, M.: MARCIE – Model Checking and Reachability Analysis Done Efficiently. In: Colom, J.-M., Desel, J. (eds.) PETRI NETS 2013. LNCS, vol. 7927, pp. 389–399. Springer, Heidelberg (2013)
19. Heiner, M., Gilbert, D., Donaldson, R.: Petri nets for systems and synthetic biology. In: Bernardo, M., Degano, P., Zavattaro, G. (eds.) SFM 2008. LNCS, vol. 5016, pp. 215–264. Springer, Heidelberg (2008)
20. Heiner, M., Herajy, M., Liu, F., Rohr, C., Schwarick, M.: Snoopy - A Unifying Petri Net Tool. In: Haddad, S., Pomello, L. (eds.) PETRI NETS 2012. LNCS, vol. 7347, pp. 398–407. Springer, Heidelberg (2012)
21. Huber, P., Jensen, K., Shapiro, R.: Hierarchies in coloured Petri nets. In: Rozenberg, G. (ed.) APN 1990. LNCS, vol. 483, pp. 313–341. Springer, Heidelberg (1991)
22. Jensen, K., Kristensen, L.: Coloured Petri nets: modelling and validation of concurrent systems. Springer (2009)
23. Liu, F.: Colored Petri Nets for Systems Biology. Ph.D. thesis, BTU Cottbus, Dep. of CS (January 2012)
24. Liu, F., Heiner, M.: Modeling membrane systems using colored stochastic Petri nets. Nat. Computing (2013)
25. Liu, F., Heiner, M.: Multiscale modelling of coupled Ca^{2+} channels using coloured stochastic Petri nets. IET Systems Biology (2013)
26. Liu, F., Heiner, M., Rohr, C.: Manual for Colored Petri Nets in Snoopy. Tech. Rep. 02-12, Brandenburg University of Technology Cottbus, Dep. of CS (March 2012)
27. Liu, F., Heiner, M., Yang, M.: An efficient method for unfolding colored Petri nets. In: Proc. 2012 Winter Simulation Conference (WSC 2012). IEEE, Berlin (2012)
28. Marwan, W., Sujathab, A., Starostzik, C.: Reconstructing the regulatory network controlling commitment and sporulation in *Physarum polycephalum* based on hierarchical Petri net modeling and simulation. J. of Theoretical Biology 236(4), 349–365 (2005)
29. Rohr, C., Marwan, W., Heiner, M.: Snoopy - a unifying Petri net framework to investigate biomolecular networks. Bioinformatics 26(7), 974–975 (2010)
30. Saunders, N., Moxon, E., Gravenor, M.: Mutation rates: estimating phase variation rates when fitness differences are present and their impact on population structure. Microbiology 149(2), 485–495 (2003)
31. Soliman, S., Heiner, M.: A unique transformation from ordinary differential equations to reaction networks. PlosONE 5(12), e14284 2010)
32. Tsang, E.P.K.: Foundations of Constraint Satisfaction. Academic Press, London (1993)