An Approximate Execution of Rule-Based Multi-level Models

Tobias Helms, Martin Luboschik, Heidrun Schumann,
and Adelinde M. Uhrmacher

University of Rostock, Institute of Computer Science,
Albert-Einstein-Straße 22, 18059 Rostock, Germany
{tobias.helms,martin.luboschik,heidrun.schumann,
adelinde.uhrmacher}@uni-rostock.de
http://www.uni-rostock.de

Abstract. In cell biology, models increasingly capture dynamics at different organizational levels. Therefore, new modeling languages are developed, e.g., like ML-Rules, that allow a compact and concise description of these models. However, the more complex models become the more important is an efficient execution of these models. τ-leaping algorithms can speed up the execution of biochemical reaction models significantly by introducing acceptable inaccurate results. Whereas those approximate algorithms appear particularly promising to be applied to hierarchically structured models, the dynamic nested structures cause specific challenges. We present a τ-leaping algorithm for ML-Rules which tackles these specific challenges and evaluate the efficiency and accuracy of this adapted τ-leaping based on a recently developed visual analysis technique.

Keywords: computational biology, rule-based modeling, multi-level modeling, tau-leaping, efficient execution.

1 Introduction

The size and complexity of models in systems biology have steadily been increasing in the last decade, also promoted by the development of modeling languages whose syntax allows a more compact and succinct description of models [6,13]. Consequently, the need for an efficient execution has increased, so that many improvements of Gillespie's original stochastic simulation algorithm (SSA) [8] have been developed, e.g., by using more efficient data structures or performing tasks concurrently [7,17,5]. However, exact variants of the SSA still execute every single event that occurs inside the system. Thus, with large propensities (due to high numbers of molecules, large stochastic rate constants or diffusion constants, and multi-scale models), the time step between successive events might decrease drastically, rendering the simulation progress very slow [2].

Multi-level models describe a system at different levels, e.g., combining intracellular and intercellular dynamics. Multi-level rules (ML-Rules) [21] is a rule-based modeling formalism developed to model systems operating at different

A. Gupta and T.A. Henzinger (Eds.): CMSB 2013, LNBI 8130, pp. 19–32, 2013.

organizational levels. Therefore, an ML-Rules model can represent dynamically and arbitrarily nested biochemical reaction networks. So far, the simulation algorithm of ML-Rules bases on the exact SSA and thus, shares its drawbacks. In contrast, approximate algorithms trade accuracy for execution speed. One famous family of approximate algorithms are leap methods, e.g., τ-leaping [9], which abandon the idea of single event executions in favor of larger time jumps and an estimation of all events within the intervals. These methods can gain a significant performance improvement compared to exact SSA procedures [15]. Based on the τ-leaping variant presented in [3], we develop a τ-leaping approach to compute ML-Rules models and present methods to tackle the specific challenges caused by the dynamic nesting structure. As earlier experiments showed that the parameters of τ-leaping algorithms influence both speed and accuracy [15,18], we use visual analysis to illuminate this influence. As case studies serve a Wnt/β-catenin pathway model [22], a fission yeast model [21], and a lipid raft model [10].

2 Background

We use the τ-leaping variant of Cao [3] as the basis of our τ-leaping approach for ML-Rules. In the following, this algorithm is explained in more detail. Additionally, a brief description of ML-Rules is given.

2.1 Tau-Leaping

The τ-leaping algorithm was introduced by Gillespie et al. [9] to speed up the simulation of well-stirred biochemical reaction networks. Instead of simulating every single reaction that occurs inside the system, as done by exact algorithms, τ-leaping performs "leaps" along the time line. For each leap, τ-leaping calculates the number of firings for each reaction during this leap and executes all reaction firings simultaneously. The length of a single leap is denoted by τ. More formally, a τ-leap can be described by

$$\mathbf{X}(t+\tau) = \mathbf{X} + \sum_{r \in R(\mathbf{X})} K_r(\tau; \mathbf{X}, t) \cdot \mathbf{v}(r) \tag{1}$$

with $R(\mathbf{X})$ being the set of all potential reactions given the current state $\mathbf{X}$, $\mathbf{v}(r)$ as the state change map for the reaction r and $K_r(\tau; \mathbf{X}, t)$ representing the firings' number of r during τ for $\mathbf{X}$. Restricting the size of τ to a value sufficiently small that the propensity $a(r)$ remains nearly constant during the leap for each reaction $r \in R(\mathbf{X})$ allows an approximation of $K_r(\tau; \mathbf{X}, t)$ by a Poisson random variable $P(a(r), \tau)$ with mean and variance $a(r)\tau$. This condition on the selection of τ is called the *leap condition*. The degree of acceptable propensity changes is bounded by the error parameter ϵ.

Many improvements of Gillespie's original approach have been developed recently [3,11]. Since we use the mechanism of Cao [3] as the basis of our algorithm,

this method shall be described briefly. Initially, the reaction set $R(\mathbf{X})$ is computed. Afterwards, $R(\mathbf{X})$ is divided into a set of non critical reactions $R_{ncr}(\mathbf{X})$, and a set of critical reactions $R_{cr}(\mathbf{X}) = R(\mathbf{X})\backslash R_{ncr}(\mathbf{X})$ using the parameter $n_c \in \mathbb{N}$. A reaction is assigned as critical if this reaction cannot be fired more than n_c times, i.e., at least one reactant would be completely consumed after n_c firings of this reaction. The separation is done to reduce the probability of negative populations caused by the unbounded Poisson distribution. Therefore, critical reactions are only allowed to fire at most once during a τ-leap.

Next, one candidate for τ, denoted τ', is computed based on $R_{ncr}(\mathbf{X})$. The set $RS_{ncr}(\mathbf{X})$ of reactant species of all $R_{ncr}(\mathbf{X})$ is determined initially, i.e., the set of species which are a reactant in at least one reaction of $R_{ncr}(\mathbf{X})$. For each reactant species $rs \in RS_{ncr}(\mathbf{X})$, three values are computed. Firstly, the function $g(rs)$, which is used to "guarantee that bounding the relative change of states is sufficient for bounding the relative change of propensity functions" [23] is computed by the equation from [23]:

$$g(rs) = h(rs) + \frac{h(rs)}{n(rs)} \sum_{i=1}^{n(rs)-1} \frac{i}{\mathbf{X}(rs) - i} \tag{2}$$

$h(rs)$ denotes the highest order of reactions in which rs is a reactant species. $n(rs)$ denotes the highest amount of rs which is consumed by any of the highest order reactions. The changes' mean and variance of rs of all reactions in $R_{ncr}(\mathbf{X})$ are computed afterwards by

$$\hat{\mu}(rs) = \sum_{r \in R_{ncr}(\mathbf{X})} v_r(rs) \cdot a(r) \qquad \hat{\sigma}^2(rs) = \sum_{r \in R_{ncr}(\mathbf{X})} v_r(rs)^2 \cdot a(r) \tag{3}$$

With the help of these equations, τ' is computed by

$$\tau' = \min_{rs \in RS_{ncr}(\mathbf{X})} \left\{ \frac{\max\{\epsilon \cdot \mathbf{X}(rs)/g(rs), 1\}}{|\hat{\mu}(rs)|}, \frac{\max\{\epsilon \cdot \mathbf{X}(rs)/g(rs), 1\}^2}{\hat{\sigma}^2(rs)} \right\} \tag{4}$$

If τ' is smaller than a multiple α of the propensity sum of all reactions, $1/a(R(\mathbf{X}))$, a number N_{SSA} of exact iterations is performed instead of a τ-leap. Very small τ values would cause many firing numbers being set to zero, so that the algorithm tries to overcome the critical region in the state space by falling back to a much more simpler and, in this case, often faster exact simulation.

If τ' is sufficiently large, a second τ candidate, denoted τ'', is sampled from $\text{Exp}(1/a(R_{cr}(\mathbf{X})))$, i.e., τ'' represents the time interval until the next critical reaction will fire. The minimum of τ' and τ'' is used as the next τ value. If τ'' is smaller than τ', exactly one critical reaction is selected, which will fire once during this τ-leap. After computing τ, the number of reaction firings can be computed and these reactions can be executed simultaneously. If any negative population occurs after executing these reactions, all changes are discarded, τ' is halved and the procedure is repeated until a valid τ-leap is executed.

2.2 ML-Rules

Multi-level rules (ML-Rules) is a rule-based formalism which can be used to create hierarchical models, including models with downward and upward causation between different hierarchy levels [21]. It has been realized as part of the modeling and simulation framework JAMES II [14]. Models are described by species definitions, a start state, and rule schemata. A species definition declares a species type, i.e., a unique name (e.g., A, $Cell$) and a tuple of attributes. A concrete species is defined by its type, attribute values, and sub species. Furthermore, species are treated population-based, i.e., identical species are summarized and an amount value is added to the representative of them. Species are identical if they have the same type, the same attribute values, identical sub species, and are enclosed by the same species. A rule scheme comprises a set of reactant patterns, a set of products and a kinetic rate. Reactant patterns describe species by their names, their desired attributes (optionally expressed by variables), and by sub reactant patterns, i.e., nested patterns can reach across an arbitrary number of levels. For example, the reactant pattern $A[B]$ describes species of type A, which contain at least one species B. Products are defined analogously to reactant patterns. The kinetic rate of a rule scheme can be an arbitrary expression, i.e., ML-Rules is not fixed to mass action kinetics. Such expressions can comprise simple arithmetics, conditions, and functions. Mass action kinetics can be modeled by bounded variables. For example, the rule schema

$$B^b + C^c \xrightarrow{b \cdot c \cdot r_c} D$$

bounds the variable b to the amount of the selected species B, the variable c to the amount of the selected species C and uses these variables and a constant reaction rate r_c to compute the reaction propensity. Additionally, the hierarchy above the reactant species is considered to compute the propensity of a reaction (see figure 1).

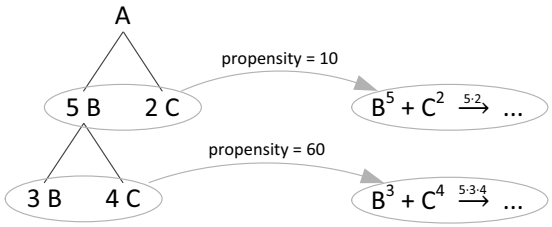

Fig. 1. Illustration of the rule schema instantiation of ML-Rules (adapted from [20, p. 143]). The rule scheme $B^b + C^c \xrightarrow{b \cdot c} \ldots$ is applied to the species on the left. Two reactions are instantiated. Due to the hierarchical multiplicity, a propensity of 60 $(12 \cdot 5)$ is assigned to the lower reaction.

So far, ML-Rules models have been executed by simulation algorithms based on the SSA [21]. These deal with dynamic hierarchical structures, complex rule schemata, and unbounded sets of species and reactions. To improve the performance of executing a model, a component-based simulation algorithm was developed recently to tailor the algorithm to specific requirements of concrete models. Furthermore, reinforcement learning techniques were used to adapt the algorithm due to changing model requirements during one simulation run [12].

3 τ-leaping for ML-Rules

The principle method of our τ-leaping approach for ML-Rules follows the one presented in [3] (see sec. 2.1). The algorithm is implemented inside JAMES II and will be part of the release 0.9.3. At first, all reactions are split into the sets of critical and non critical reactions. In contrast to the previous ML-Rules simulation algorithm, propensities of non critical reactions are computed locally, as the changes inside a context during a τ-leap depend neither on the number of context copies nor on the number of species higher up the hierarchy. Consequently, hierarchical multiplicities are not included in the propensity computation (see figure 2).

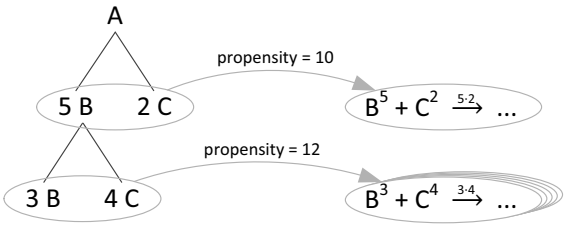

Fig. 2. Hierarchical multiplicities are not included in the propensity calculation of non critical reactions, because reactions are considered individually, i.e., for each of the five Bs, a reaction number of the lower reaction is computed and the corresponding number of reaction firings is applied to the specific B species

Next, each reaction context (rc) is considered separately and local τ_l' values are computed for each of these reaction contexts based on equation 4. Consequently, a τ_l' value reflects the relative changes inside its reaction context. τ' represents the minimum of all τ_l' values. After computing a suitable τ' value, it is checked whether τ' is too small and if so, a number N_{SSA} of SSA steps is performed. Otherwise, τ'' is computed for the critical reactions. If τ'' is smaller than τ', τ is set to τ'' and one critical reaction is selected which will fire exactly once. Otherwise, τ is set to τ' and no critical reaction will fire.

In contrast to flat chemical reaction networks, the computation of reaction firings and the simultaneous execution of all these reactions are complex tasks,

because they have to be executed recursively along the hierarchy of the model. Thereby, each species is treated individually, because it is assumed that even identical species probably have different numbers of reaction firings during a τ-leap. The algorithm starts from the topmost reaction context of the model, i.e., the context which is not enclosed by a species:

1. Compute the number of reaction firings for the current reaction context rc and τ.
2. Remove all reaction reactants from rc.
3. For each remaining enclosed species of rc, execute this algorithm.
4. Add all reaction products to rc.

If the model state is invalid after a τ-leap, i.e., the amount of at least one species is negative, the changes are discarded, τ' is halved, and the algorithm is repeated.

3.1 Reaction Splitting

We applied the described τ-leaping algorithm to realistic ML-Rules models and observed that τ-leaping performed poorly. The number of executed reactions during a τ-leap was too small. The used models describe processes of cells. Typically, they comprise rules to diffuse species into or out of cells, rules to manipulate species inside a cell whose products and kinetic rates depend on the enclosing cell (e.g., on its volume), and rules to manipulate cells. For example, the following rule describes the diffusion of a protein into a cell:

$$Cell[sol?]^c + Protein^p \xrightarrow{c \cdot p \cdot r_c} Cell[Protein + sol?]$$

Multiple firings of a reaction based on this rule would add one protein to a number of cells. Additionally, such reactions tend to be critical, if the number of cells is small. The question is how to handle these reactions differently so that several proteins can enter the same cell within one τ-leap and the number of critical reactions is reduced. During calculating τ and the firing rates of reactions the cell in the above example can be ignored as its attributes do not change. Therefore, such reactions are split into two reactions, one describes the changes outside the cell and one describes the changes inside the cell. Referring to the example above, a concrete reaction would be split into the following two reactions:

$$\begin{aligned} &\text{1. Inside the cell} \quad : \quad \xrightarrow{p \cdot r_c} Protein \\ &\text{2. Outside the cell} : Protein \xrightarrow{p \cdot r_c} \end{aligned}$$

The creation of these two reactions is possible, because all necessary information, e.g., the cell which is used and the values of the variables, are known after creating the basic reaction. The propensity of these reactions is reduced by the factor of the cells' amount, because in τ-leaping each cell is considered individually. These two reactions can now be handled as ordinary reactions, whereas the original reaction is not considered during the next τ-leap any more. Finally, the execution algorithm has only to ensure that both new reactions have the same

firing number. Currently, we allow reaction splittings if the rule of a reaction contains exactly one pair of reactant and product pattern which satisfies the following conditions:

1. The species types and the attributes of the reactant and the product pattern are the same and the amount of both must be 1.
2. The amount of the top species of the reactant pattern is used as a factor for the reaction rate.
3. Both patterns comprise two hierarchy levels.

These conditions are a first rather restrictive approach, however, which facilitates automating reaction splitting, i.e., everything is done transparently to the user. Future approaches could generalize these conditions, so that more complex reactions could be split.

3.2 Population-Based τ-leaps

During the computation of the reaction firing numbers and the corresponding reactions, species are considered individually, i.e., the population-based approach cannot be used (see sec. 3). This individual consideration of species during a τ-leap can lead to a high computational effort. That is why we created a new parameter $\mu \in \mathbb{N}^+ \cup \infty$, which introduces the concept of populations to the execution of τ-leaps. Basically, during the computation of the reaction firing numbers and the corresponding reactions, it is used to partition a set of identical species into μ equally sized groups. Afterwards, all individual species of one group will evolve equally during the current τ-leap. If μ is greater than the amount of a species (guaranteed by $\mu = \infty$), one group is created for each individual of this species, i.e., all species are treated individually. Eventually, if $\mu = 1$, species are completely treated population-based, i.e., all identical species evolves equally. Consequently, reducing the value of μ on the one hand can decrease the accuracy of the simulation results but on the other hand can also decrease the computational effort of the τ-leaps.

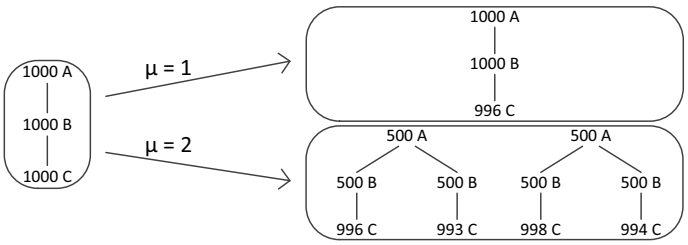

Fig. 3. Illustration of the impact of different μ values on one τ-leap. The rule scheme $C^c \xrightarrow{c \cdot r_c}$ is applied to the left species, i.e., one C is consumed after firing the corresponding reaction once. In this example, the concrete reaction firing numbers are chosen manually for illustration.

Figure 3 illustrates the concept of μ. The rule scheme $C^c \xrightarrow{c \cdot r_c}$ is applied to the species on the left, i.e., one reaction is created, which would consume exactly one C. If $\mu = 1$, species are treated completely population-based. Thus, one Poisson number is sampled for the reaction firing number and the amount of C is simply reduced according to that number, i.e., the reaction fires equally frequent inside all Bs. Afterwards, the whole species comprises again 1000 identical As, each enclose 1000 identical Bs. If $\mu = 2$, the As are separated into two groups, each comprises 500 As. For each group of As, the Bs are separated analogously. One reaction firing number is then computed for each group of Bs, and the amount of C is reduced accordingly. Thus, a total of four Poisson numbers are sampled. Finally, if μ would be set to ∞, 1000 groups of As would be created, each containing 1000 groups of Bs, i.e., one million Poisson numbers would be sampled.

4 Evaluation

We evaluate the developed τ-leaping approach for ML-Rules based on experiments with three different models, namely a realistic model of the Wnt/β pathway in neural progenitor cells [22], a cell cycle model [21], and a lipid raft model [10]. To analyze the effects of different parameter settings, we use a recently developed visual analysis technique [18].

4.1 The Wnt/β-Catenin Pathway Model

The Wnt/β-catenin pathway model described in [22] comprises five species types and twelve rule schemes. It defines one species type to represent cells and one species type to represent nuclei inside cells. The amount of cells can be defined by a parameter. Moreover, three other types are defined, representing Wnt proteins, β-catenin proteins, and Axin proteins. The Axin protein species type has one attribute reflecting the phosphorylation state of such species. Inside a cell, Axin proteins are phosphorylized and dephosphorylized (partially dependent on the number of Wnt proteins). β-catenin proteins are constantly moved inside and outside the nucleus of a cell. Depending on the number of β-catenin proteins inside the nucleus, dephosphorylized Axin proteins are synthesized inside a cell. Furthermore, Wnt proteins, Axin proteins, and β-catenin proteins are degraded constantly. Despite the number of β-catenin proteins, the degree of β-catenin degradation also depends on the number of phosphorylized Axin proteins.

We use 480 configurations of τ-leaping for the analysis, built from the cross product of twenty ϵ values (0.01, 0.02, ..., 0.2), four α values (5, 10, 15, 20), and six μ values (1, 2, 4, 6, 8, 10). n_c is always set to 10 and N_{SSA} is always set to 100. For each parameter setting, 100 replications with the simulation end time 200 are executed. The model state is observed after 0.4 time units have elapsed, i.e., 500 observations are made per simulation. Once we executed the experiment of the model with one cell, once with ten cells.

Referring to the performance, all configurations of τ-leaping need significantly less execution time per replication on average compared to the SSA. For one

simulation of the model with one cell, the SSA need ≈ 39 s on average ($\sigma \approx 2.5\ s$) on the used computer, the fastest τ-leaping configuration ($\epsilon = 0.2$, $\alpha = 5$, $\mu = 1$) need ≈ 0.5 s on average ($\sigma \approx 0.05\ s$), and the slowest one ($\epsilon = 0.01$, $\alpha = 5$, $\mu = 10$) need ≈ 1.8 s on average ($\sigma \approx 1.3\ s$). For one simulation of the model with ten cells, the execution times of the SSA and of most τ-leaping configurations nearly increase by thirty, e.g., the SSA need ≈ 1125 s on average ($\sigma \approx 76\ s$). The ϵ parameter influences the performance significantly, i.e., the higher ϵ is chosen, the lower is the execution time. The α parameter influence the execution time negligibly. As excepted, for the model with one cell, the μ parameter has neither an effect on the simulation results nor on the execution time, because there is only one cell and one nucleus, i.e., population-based τ-leaps do not occur. For the model with ten cells, on the contrary, μ has an impact on the performance, i.e., τ-leaping performs better with $\mu = 1$ than with the other used values for μ (e.g., τ-leaping ($\epsilon = 0.2$, $\alpha = 5$) need $\approx 8\ s$ with $\mu = 1$ and $\approx 17\ s$ with $\mu \in \{2, 4, 6, 8, 10\}$). Interestingly, the difference depends on the value of ϵ, i.e., the higher ϵ is chosen, the smaller is the difference (if $\epsilon > 0.07$, the difference diminishes almost completely). We found out that only for $\mu = 1$, population-based τ-leaps are executed frequently due to identical cells. For the other values of μ, each cell differs permanently from each other after a short period of the simulation, so that only a few population-based τ-leaps are executed at the beginning. The executed SSA steps caused by small τ values are the main reason for this behavior. Nevertheless, the impact of these SSA steps should diminish with a higher number of cells, i.e., even with higher μ values several cells should be identical frequently during the simulation and thus these μ values should have an effect on the performance.

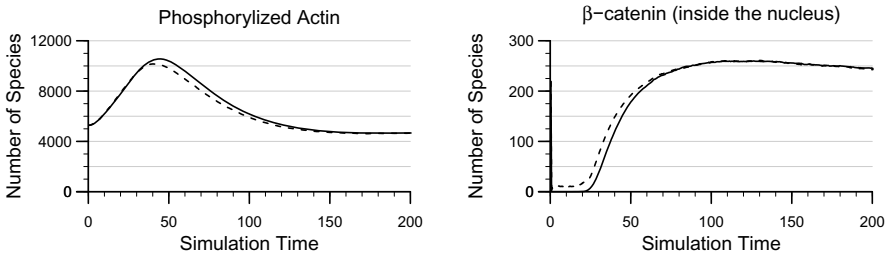

Fig. 4. The average results of the SSA (solid lines) compared to τ-leaping (dotted lines) with $\epsilon = 0.2$, $\alpha = 5$, and $\mu = 10$ (i.e., the least accurate configuration we used) of the species amount of phosphorylized Actin proteins and of β-catenin proteins inside the nucleus for the Wnt/β-catenin pathway model with one cell.

However, the interesting aspect is how much accuracy is traded for this gain of efficiency and especially due to their impact on the execution time how is the impact of the parameters ϵ and μ on the accuracy. Most importantly, all configurations of τ-leaping produce similar results compared to the results of the

SSA (e.g., see figure 4). Due to the high number of parameters, configurations, and observations, we use visual analysis to examine the impact of ϵ and μ on the accuracy, because it enables us to interactively explore data, e.g., to observe relations, correlations, and interdependencies between parameters and results. Precisely, we use the software developed in [18]. It visually connects the used parameter values and the computed accuracy measures with the corresponding simulation results of one species. Focusing on the first two, the software uses color coding and maps low values of the five parameters to white and high values to black, low accuracy values (in this case, represented by the p-value of the paired Wilcoxon rank sum test [24, p. 513]) of the observations to white and high values to saturated cyan (see figure 5). Each line corresponds to one parameter setting aligned to the according accuracy values evolving over time. The software now enables us to scroll trough, sort, and select parameter settings by concrete parameters and accuracy values of specific time intervals.

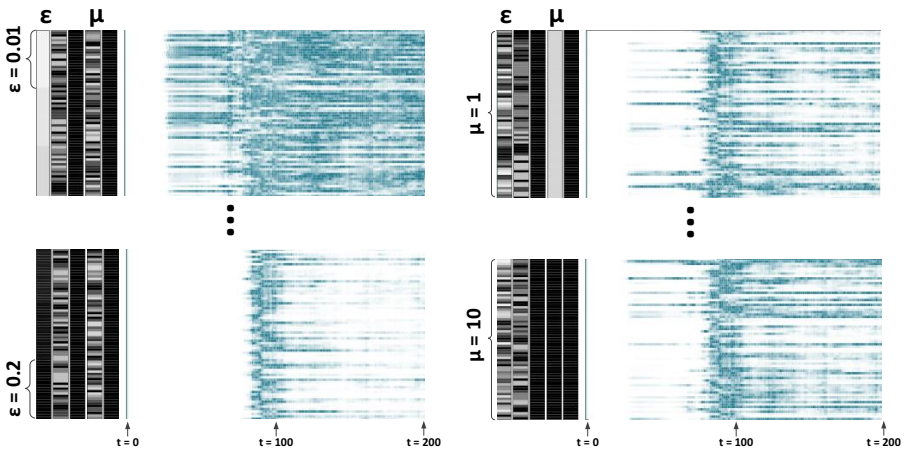

Fig. 5. Accuracy illustrations of parameter settings sorted by the value of ϵ (left) and by the value of μ (right) on the example of the dephosphorylized Axin proteins of the Wnt/β-catenin pathway model with ten cells. The more saturated the cyan of a field, the more accurate are the corresponding results.

To analyze the impact of ϵ on the accuracy, we started by sorting the parameter settings according to ϵ and scrolling through the configurations (see the left illustrations of figure 5). Characteristic for τ-leaping, the smaller ϵ is chosen, the more accurate the results get, which becomes visible by dark cyan regions in the top left illustration of figure 5. Vice versa, the accuracy decreases if ϵ is increased, especially in the beginning of the simulation (see the bottom left illustration in figure 5). Thus, referring to the impact of ϵ, the algorithm behaves as expected. Sorting the parameter settings according to μ, we observed another behavior: No relation between its value and the accuracy can be revealed

(see the right illustrations of figure 5). As this might be specific for the chosen model, further experiments are needed to analyze the impact of μ on accuracy in more detail. Also, the parameter α does not influence the result accuracy in our model either, i.e., the computed τ' is usually chosen sufficiently high so that eventually only few SSA steps are executed. Interestingly, at several time points the accuracy of all configurations decreases slightly, which results in faint but noticeable vertical lines in the visualization (e.g., see figure 6). The reason for this is not yet clear, e.g., whether this is caused by large τ values in comparison to smaller observation rates or whether it is caused by the execution of SSA steps.

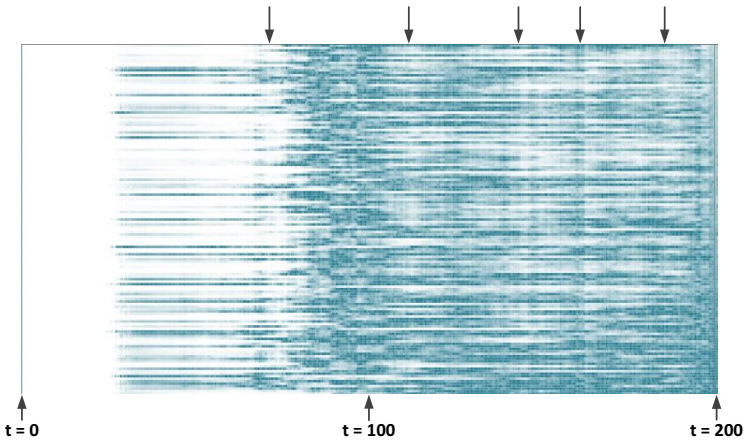

Fig. 6. The accuracy illustrations of some τ-leaping configurations again on the example of the dephosphorylized Axin proteins of the Wnt/β-catenin pathway model with ten cells. The illustrations show noticeable vertical lines, i.e., many configurations behave similar at these times. Some of these lines are marked by the arrows on top.

4.2 The Lipid Raft Model

The lipid raft model describes the synthesis, degradation and diffusion of lipid rafts in cell membranes [10]. It comprises receptor species, which diffuse inside and outside lipid rafts. While degrading a lipid raft, all containing receptors are removed. For this evaluation, the model was reimplemented in ML-Rules. We use the same 480 configurations of the experiments with the Wnt/β-catenin pathway model for the evaluation (see 4.1). Regarding the results, again ϵ is the most important parameter. However, if $\epsilon > 0.01$, the accuracy of the results decreases significantly. In contrast to the Wnt/β-catenin pathway model, the α parameter also significantly influences the results (if $\epsilon = 0.01$). For example, if $\alpha = 5$, many SSA steps are executed, so that the results are still relatively accurate. If $\alpha \in \{10, 15, 20\}$, SSA steps are executed rarely and the accuracy of the results decrease dramatically. Thus, accurate results are only achieved with $\epsilon = 0.01$ and $\alpha = 5$. The according configurations perform 16% better

on average than the SSA, which needs $\approx$ 72 s. The μ parameter again only slightly influence the execution time of the simulations and does not noticeable influence the simulation results. The evaluation of the experiments with the lipid raft model again shows the importance of an appropriate configuration of all parameters of τ-leaping, i.e., using inappropriate configurations can lead to fast execution times but totally wrong results.

4.3 The Fission Yeast Model

The fission yeast model [21] represents the cell division and mating process of fission yeast cells. It comprises intracellular as well as intercellular reactions and contains a simple grid-based spatial level to describe specific diffusion reactions. In this model, a cell species contains two attributes, representing the volume and the phase of the cell. Further species types describe different pheromones and proteins. However, initial experiments show that τ-leaping performs worse for this model than the SSA. Reactions which change attributes of cells are the most frequent reactions in the model. Typically, the number of cells is small (1 - 100), so that either these reactions are declared as critical, or the computed τ' values are small. Consequently, τ-leaping only summarizes a small number of reactions per τ-leap, i.e., two reactions are summarized per τ-leap on average. All in all, these results show that τ-leaping for ML-Rules behaves like existing τ-leaping algorithms, i.e., if the most frequent reactions involve species with small amounts, τ-leaping cannot exploit its ability to execute leaps.

5 Related Work

τ-leaping is a promising algorithm to improve the performance of biochemical reaction networks [9]. Consequently, many improvements and extensions have been developed over the last years. For example, many τ-leaping variants deal with the problem of negative populations [1,25]. However, only a few variants focus on structured models. Related approaches are those τ-leaping variants that consider space. For example, the binomial spatial τ-leaping algorithm [19] and S-τ [16] are based on a grid-like structuring of space into subvolumes where diffusion events happen between and reaction events within those subvolumes. Similarly to our approach a τ candidate is calculated separately for each subvolume. Calculating τ candidates locally is also a strategy adopted for dynamical probabilistic P Systems [4]. However, those approaches do not have hierarchical contexts similar to those of ML-Rules: The former approaches due to not supporting hierarchies, the latter due to not supporting hierarchical rules and population-based membranes.

The performance of τ-leaping depends on the model and its configuration, i.e., the trade-off between execution time and accuracy [15,16]. To explore these dependencies in the multi-dimensional space of parameter settings and different accuracies, formulating and testing hypotheses iteratively would be the usual case. Instead, visual analysis enables the user to execute an exploratory investigation of the data helping him to get impressions of the data to formulate hypotheses specifically.

6 Conclusion

This paper presents a τ-leaping algorithm for the rule-based multi-level formalism ML-Rules, which supports dynamic nesting. It extends the traditional τ-leaping by treating identical individuals as populations (along with the parameter μ to control the grouping) and by calculating τ'_l values in the respective contexts (among which the minimum is selected as the overall τ'). The evaluation shows that the algorithm behaves like other τ-leaping variants, e.g., small ϵ values cause more accurate results than high ϵ values and the improvement of τ-leaping depends strongly on the used model. Depending on the model, the execution time can be decreased significantly, e.g., the execution time for one simulation of the used Wnt/β-catenin pathway model can be reduced by 2 orders on average. The new parameter μ, which configures the population-based execution of τ-leaps, rarely influences the execution time and does not affect the accuracy of the results. However, this relation is likely caused by the used models and thus, deserves further investigations - as do the areas of poor accuracy which have been revealed in our visual exploratory analysis. Currently, the developed τ-leaping approach only supports mass action kinetics. Since ML-Rules is not constrained to mass action kinetics, it has to be investigated which types of kinetics can be supported by τ-leaping.

Acknowledgments. This research is supported by the German Research Foundation, via research grants CoSA (UH 66/7-3) and VASSiB (part of SPP 1335).

References

1. Cao, Y., Gillespie, D.T., Petzold, L.R.: Avoiding negative populations in explicit Poisson tau-leaping. The Journal of Chemical Physics 123(5) (2005)
2. Cao, Y., Gillespie, D.T., Petzold, L.R.: The slow-scale stochastic simulation algorithm. The Journal of Chemical Physics 122(1), 014116 (2005)
3. Cao, Y., Gillespie, D.T., Petzold, L.R.: Efficient step size selection for the tau-leaping simulation method. The Journal of Chemical Physics 124(4) (2006)
4. Cazzaniga, P., Pescini, D., Besozzi, D., Mauri, G.: Tau Leaping Stochastic Simulation Method in P Systems. In: Hoogeboom, H.J., Păun, G., Rozenberg, G., Salomaa, A. (eds.) WMC 2006. LNCS, vol. 4361, pp. 298–313. Springer, Heidelberg (2006)
5. Dematté, L., Prandi, D.: GPU computing for systems biology. Briefings in Bioinformatics 11(3), 323–333 (2010)
6. Faeder, J.R.: Toward a comprehensive language for biological systems. BMC Systems Biology 9(68) (2011)
7. Gibson, M.A., Bruck, J.: Efficient Exact Stochastic Simulation of Chemical Systems with Many Species and Many Channels. The Journal of Chemical Physics 104(9), 1876–1889 (2000)
8. Gillespie, D.T.: Exact stochastic simulation of coupled chemical reactions. The Journal of Physical Chemistry 81(25), 2340–2361 (1977)
9. Gillespie, D.T.: Approximate accelerated stochastic simulation of chemically reacting system. The Journal of Chemical Physics 115(4), 1716–1733 (2001)

10. Haack, F., Burrage, K., Redmer, R., Uhrmacher, A.M.: Studying the role of lipid rafts on protein receptor bindings with Cellular Automata. IEEE/ACM Transactions on Computational Biology and Bioinformatics (accepted for publication, 2013)
11. Harris, L.A., Clancy, P.: A "partitioned leaping" approach for multiscale modeling of chemical reaction dynamics. The Journal of Chem. Physics 125(14) (2006)
12. Helms, T., Ewald, R., Rybacki, S., Uhrmacher, A.M.: A Generic Adaptive Simulation Algorithm for Component-based Simulation Systems. In: Proc. 27th Workshop on Principles of Adv. and Dist. Simulation, PADS 2013 (2013)
13. Henzinger, T.A., Jobstmann, B., Wolf, V.: Formalisms for Specifying Markovian Population Models. In: Bournez, O., Potapov, I. (eds.) RP 2009. LNCS, vol. 5797, pp. 3–23. Springer, Heidelberg (2009)
14. Himmelspach, J., Uhrmacher, A.M.: Plug'n simulate. In: Proc. 40th Annual Simulation Symposium (ANSS 2007), pp. 137–143 (2007)
15. Jeschke, M., Ewald, R.: Large-Scale Design Space Exploration of SSA. In: Heiner, M., Uhrmacher, A.M. (eds.) CMSB 2008. LNCS (LNBI), vol. 5307, pp. 211–230. Springer, Heidelberg (2008)
16. Jeschke, M., Ewald, R., Uhrmacher, A.M.: Exploring the Performance of Spatial Stochastic Simulation Algorithms. The Journal of Computational Physics 230(7), 2562–2574 (2011)
17. Li, H., Petzold, L.: Logarithmic Direct Method for Discrete Stochastic Simulation of Chemically Reacting Systems. Technical report, Department of Computer Science, University of California: Santa Barbara (2006)
18. Luboschik, M., Rybacki, S., Ewald, R., Schwarze, B., Schumann, H., Uhrmacher, A.M.: Interactive Visual Exploration of Simulator Accuracy: A Case Study for Stochastic Simulation Algorithms. In: Proc. 44th Winter Simulation Conference, WSC 2012 (2012)
19. Marquez-Lago, T.T., Burrage, K.: Binomial tau-leap spatial stochastic simulation algorithm for applications in chemical kinetics. The Journal of Chemical Physics 127(10) (2007)
20. Maus, C.: Toward Accessible Multilevel Modeling in Systems Biology - A Rule-based Language Concept. PhD thesis, University of Rostock, Germany (2013)
21. Maus, C., Rybacki, S., Uhrmacher, A.M.: Rule-based multi-level modeling of cell biological systems. BMC Systems Biology 5(166) (2011)
22. Mazemondet, O., John, M., Leye, S., Rolfs, A., Uhrmacher, A.M.: Elucidating the Sources of β-Catenin Dynamics in Human Neural Progenitor Cells. PLoS ONE 7(8), e42792 (2012)
23. Sandmann, W.: Streamlined formulation of adaptive explicit-implicit tau-leaping with automatic tau selection. In: Proc. 41st Winder Simulation Conference (WSC 2009), pp. 1104–1112 (2009)
24. Sheskin, D.J.: Handbook of Parametric and Nonparametric Statistical Procedures, 4th edn. Chapman & Hall/CRC (2007)
25. Tian, T., Burrage, K.: Binomial leap methods for simulating stochastic chemical kinetics. The Journal of Chemical Physics 121(21), 10356–10364 (2004)