BMC Bioinformatics

METHODOLOGY ARTICLE Open Access

Potential based, spatial simulation of dynamically nested particles

Till Köster*, Philipp Henning and Adelinde M. Uhrmacher

Abstract

Background: To study cell biological phenomena which depend on diffusion, active transport processes, or the locations of species, modeling and simulation studies need to take space into account. To describe the system as a collection of discrete objects moving and interacting in continuous space, various particle-based reaction diffusion simulators for cell-biological system have been developed. So far the focus has been on particles as solid spheres or points. However, spatial dynamics might happen at different organizational levels, such as proteins, vesicles or cells with interrelated dynamics which requires spatial approaches that take this multi-levelness of cell biological systems into account.

Results: Based on the perception of particles forming hollow spheres, ML-Force contributes to the family of particle-based simulation approaches: in addition to excluded volumes and forces, it also supports compartmental dynamics and relating dynamics between different organizational levels explicitly. Thereby, compartmental dynamics, e.g., particles entering and leaving other particles, and bimolecular reactions are modeled using pair-wise potentials (forces) and the Langevin equation. In addition, forces that act independently of other particles can be applied to direct the movement of particles. Attributes and the possibility to define arbitrary functions on particles, their attributes and content, to determine the results and kinetics of reactions add to the expressiveness of ML-Force. Its implementation comprises a rudimentary rule-based embedded domain-specific modeling language for specifying models and a simulator for executing models continuously. Applications inspired by cell biological models from literature, such as vesicle transport or yeast growth, show the value of the realized features. They facilitate capturing more complex spatial dynamics, such as the fission of compartments or the directed movement of particles, and enable the integration of non-spatial intra-compartmental dynamics as stochastic events.

Conclusions: By handling all dynamics based on potentials (forces) and the Langevin equation, compartmental dynamics, such as dynamic nesting, fusion and fission of compartmental structures are handled continuously and are seamlessly integrated with traditional particle-based reaction-diffusion dynamics within the cell. Thereby, attributes and arbitrary functions allow to flexibly describe diverse spatial phenomena, and relate dynamics across organizational levels. Also they prove crucial in modeling intra-cellular or intra-compartmental dynamics in a non-spatial manner, and, thus, to abstract from spatial dynamics, on demand which increases the range of multi-compartmental processes that can be captured.

Keywords: Space, Simulation, Nesting, Force, Multi-level, Modeling, Attributed

Background

Space plays an important role in cell biological dynamics, such as cell signalling [1]. To study cell biological phenomena which depend on diffusion, active transport processes, or the locations of species, modeling and simulation studies need to take space into account. Over the last decade, a variety of spatial modeling and simulation methods and tools has been developed to support these simulation studies. They offer different approaches how to describe a spatial model, e.g., reaction-based, rule-based, or graphically, and they differ referring to what kind of spatial dynamics can be described, e.g., whether concentrations, populations, or individual particles are considered, whether a deterministic or stochastic approach is pursued, and whether movement takes place in discretized or continuous space [2, 3]. The kind of spatial

*Correspondence: till.koester@uni-rostock.de
[1] Institute of Computer Science, University of Rostock, Albert-Einstein-Straße 22, 18059 Rostock, Germany

dynamics that are supported by tools largely determines the results and questions that can be answered by a simulation study [4–6] and has been subject to different categorizations to structure the portfolio of spatial simulation methods from which the user can select.

TAKAHASHI et al. [7] distinguished spatial simulation approaches according to the representation of space. We adopt this structuring and slightly adapt it for the purpose of this paper as presented in Fig. 1. We distinguish between:

1. Stochastic non-spatial dynamics: To take the stochasticity of the modelled system into account, the propensity of reactions are sampled from exponential distributions and determine which reaction will occur at which time [9].
2. Compartmental dynamics: Many tools allow to constrain the dynamics of species to specific compartments. However also, compartments themselves can be subject to dynamics. For example, compartments can fuse and divide [10].
3. Individual particles moving in continuous space: Particles can be identified by their unique position in space, bimolecular reactions are triggered by collisions of particles, and typically particles diffuse by Brownian motion [11, 12].
4. Partial differential equations: Spatial gradients of concentrations are calculated deterministically [13].
5. Spatial stochastic dynamics: Multiple particles can occupy a position within a lattice space, and diffuse between neighboring positions [14].

Particle-based approaches (in the above categorization *c)*) are the subject of a further categorization suggested by SCHÖNEBERG et al. [8] (Fig. 1, right lower corner). Spatial particle-based simulation tools are categorized into those that support

- *level 1 – free diffusion*: basic 3D diffusion of point particles and reactions between them are considered,
- *level 2 – confined diffusion*: diffusion can be constrained to compartments,
- *level 3 – excluded volume*: point particles are replaced by volumetric entities, and
- *level 4 – potentials for particle-particle interaction*: instead of a simple rejection of movements assuming rigid cells, potentials are used to determine the interaction of particles in terms of exclusion of movement or reactions.

According to this categorization [8], Smoldyn is characterised as *level 2* particle-based approach: point particles which do not hamper each others movement are equipped with binding and unbinding radii, they can be constrained to compartments. A more recent version includes collision strategies to reach *level 3* [15]. SpringSaLaD [16], ReaDDy [17], and SRSim [18] are based on forces to model

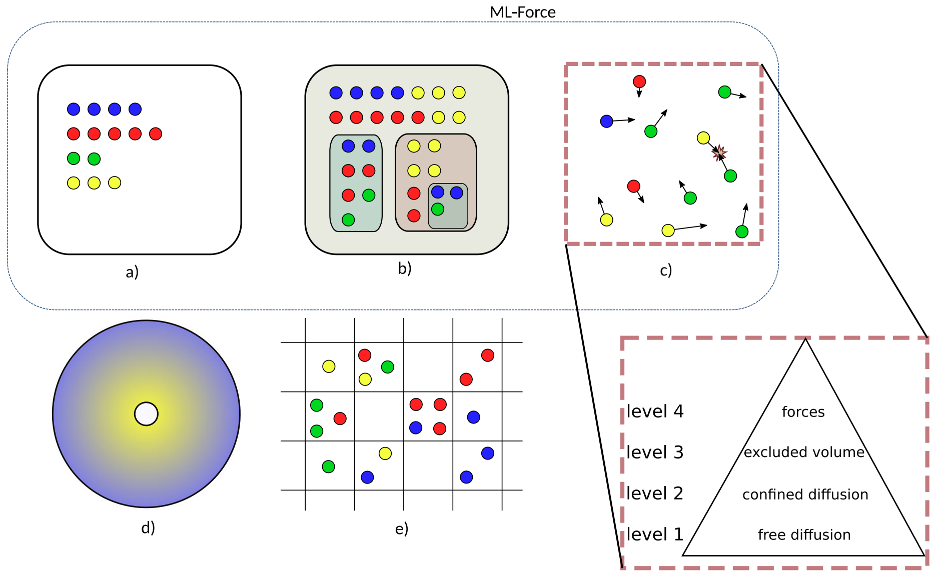

Fig. 1 Representations of space which is based on and slightly adapts Figure 1 of [7]. We distinguish between (**a**) stochastic non spatial dynamics, (**b**) compartmental dynamics, (**c**) individual particles moving in continuous space, (**d**) partial differential equations, and (*e*) spatial stochastic dynamics. Type (*c*) individual particles moving in continuous space is further described by four levels, according to SCHÖNEBERG et al. [8]. Each level builds upon the capabilities of the underlying ones, thus the representation as a pyramid. For example a method that provides excluded volumes should conceptually also be able to provide free diffusion and confined diffusion. ML-Force is characterized by a-b-c(4)

the particle-particle interactions (*level 4*). Still, the levels introduced do not imply that a tool that supports *level 3* provides all interesting features that a tool working at *level 2* offers to study the system of interest. E.g., DONOVAN [19], whose spatial simulation works at *level 2*, supports arbitrarily complex 3D mesh geometries, whereas in ML-Space [20] (*level 3*) only rigid spheres are considered.

Merging both characterizations, approaches that combine different spatial representations can be characterized. The Two-Regime method [21] and KLANN et al. [22] allow to combine particle and RDME dynamics within one model: *c(3)-e*. Similarly in [23], particles and partial differential equations are combined to focus on the spatial region of interest: *c(3)-d*. [24] couples a partial differential equations solver and Smoldyn *c(2)-d*. ML-Space combines particles at *level 3* (as excluded volumes are considered), compartmental dynamics, and spatial, stochastic simulation (RDME): *b-c(3)-e* [20].

In addition, simulation environments offer the possibility to select different spatial semantics for one model specification. For example, in VCELL [13], rule-based models can be interpreted by a particle-based simulator (i.e., Smoldyn) or a partial differential equation solver, both confined to realistically geometrical compartmental structures (*c(2)*, *d*). This list is far from being complete, and the characterization of the simulation tools may only depict a specific state in their development.

Against this background, we propose a new spatial particle-based modeling and simulation approach for cell biological systems, i.e., *ML-Force* (Fig 1). It combines:

- *Potential-based particle dynamics*: The excluded volumes that are represented by the large amount of macro-molecules within the cell affect the physico-chemical kinetics of various intracellular processes [25]. To capture the effects of molecular crowding, space exclusions need to be considered. In addition, interactions based on potentials are an effective means to study the formation of clusters [18]. Additionally, potentials allow us to capture directed movements, such as the transport of vesicles [26] (see "Results" section). Similar to ReaDDy [17], SRSim [18], or SpringSaLaD [16], ML-Force will use potentials for particle-particle interaction: *c(4)*.
- *Compartmental dynamics*: Intra-cellular space is further structured by compartments and vesicles, most of which are subject to frequent changes in terms of numbers, content, and inter-connectivity. A prominent example is the endosomal system. Vesicles form at the membrane. Here they acquire their cargo and engulf protein receptor complexes. Those are transported towards the inner cell, where part is degraded and part is recycled. The vesicles themselves move through stages of early sorting, recycling to late endosomes, closely interacting with each other – processes which include frequent fission and fusion [26] (Fig. 2). ML-Force uses the same force-based approach that it uses for the interaction of particles and global force functions, to model the compartmental dynamics. Similar to ML-Space, ML-Force supports both: compartmental and particle dynamics. However, unlike ML-Space it uses potentials *b-c(4)*.
- *Rule-based approach with arbitrary attributes and functions*: As other spatial simulators, ML-Force supports rule-based modeling [29]. However in ML-Force, attributes are not constrained to a finite set of values. Similarly, as in ML-Rules [30], attributes

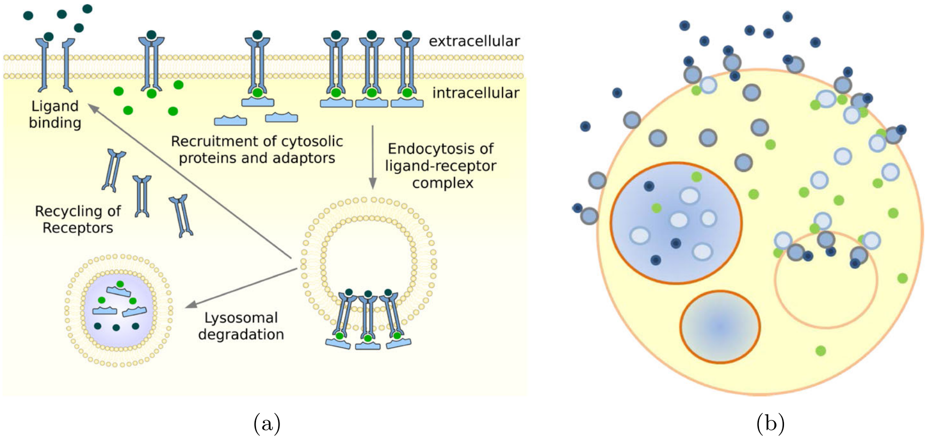

Fig. 2 Illustration (combined from [27, 28]) of a receptor-ligand binding kinetics, which includes receptor protein coupling, internalization, and recycling, whose components can be interpreted as nested arrangements of objects within objects that interact at intersection points, can be approximated to spherical particles of various sizes, move in continuous space and undergo processes of creation, degradation, internalization, externalization, fusion, and fission. **a** Biological view, **b** Hollow spheres view

are of arbitrary type and arbitrary functions are supported to determine the result and kinetics of reactions. The possibility to define attributed species and arbitrary functions increases significantly the expressiveness of a modeling formalism [31]. It allows to express spatial stochastic models (RDME) on a lattice (e) within a modeling approach that relies on a standard stochastic simulation algorithm for execution (a). By attributing species with indices that locate them in a specific point in the lattice (which represent a subvolume), e.g., $A(x,y), B(x,y)$, reactions can be constrained to species within one subvolume, e.g., $A(x_a,y_a) + B(x_b,y_b) \rightarrow 2\,B(x_b,y_b) @ \text{if}\, x_a = x_b \wedge y_a = y_b$ then k else 0. Furthermore diffusion between neighboring subvolumes can be defined, e.g., $B(x_b,y_b) \rightarrow B\Big(\text{oneOf}\big((x{+}1,y),(h{-}1,y),(x,y{+}1),(x,y{-}1)\big)\Big) @k_d$. Thus, attributed species and arbitrary functions can be used to add space to otherwise non-spatial approaches. This is an observation also made in colored PETRI-Nets [32] and in expressive rule-based modeling languages such as ML-Rules [30].

In ML-Force, equipping particles with attributes and arbitrary functions allows e.g. to describe some part of the cellular dynamics in a non-spatial manner. For example, the reaction $B + C \rightarrow D\,@\,k$ which takes place in particle A can be transformed into a non-spatial stochastic first order reaction (a). Therefore, the particle A is equipped with the corresponding attributes and a reaction is defined $A(b,c) \rightarrow A(d)\,@\,b \cdot c \cdot k$. The function $b \cdot c \cdot k$ calculates the propensity of the reaction to take place. The possibility to constrain reactions based on arbitrary functions defined on the attributes, allows us to include non-spatial stochastic dynamics (a) which brings ML-Force up to *a-b-c(4)* (see Fig. 1).

Methods

ML-Force adopts a rule-based approach to specify the spatial biochemical dynamics. Firstly the mathematical concepts and abstractions are presented. This is followed by a short description of its implementation and embedded domain-specific modeling language.

Mathematical concepts and abstractions

As the name suggests, forces are central in ML-Force: they govern the interactions between and most of the activity of particles in a continuous manner. This is in contrast to the discrete event view chosen by simulators such as ML-Space or other agent-based approaches which interpret compartmental dynamics such as the proliferation of cells as discrete events [20, 33].

Representation of particles

ML-Force is a particle-based approach. Particles represent *hollow spheres*. As particles can contain other particles, they can be used to model cellular compartments. Particles are spherically symmetric and deformable, their size, properties and nested content may vary. Particles p_m are categorized into classes called *species*: $p_m \in S_i$. Species are characterized by a set of arbitrary attributes, $S(a_1, \dots a_n)$, e.g., radius, phosphorylation sites, or phases. In our implementation of ML-Force we require particles to have at least the attributes radius r (non-zero) and density ρ. The information about its radius together with information about the system's temperature and the viscosity of its environment allows to derive the diffusion coefficient of a particle. In combination with the density, this information is used to determine the actual velocity (see "Spatial propagation" section). Particles may contain other particles, $p_m(v_1, \dots v_n)[p_j + \dots + p_k]$. All particles belonging to one class share the same attributes and attribute-types. Attribute values and content of particles may vary among particles of the same species. The interactions between and the activity of particles is governed by rules of behavior that operate on the particles, their attributes and their content. They rely on arbitrary functions for accessing and updating attributes and content of particles. Particles belonging to the same species share the same behavioral patterns.

Spatial propagation

For particles to interact they need to be in close proximity. In ML-Force, we base our model of propagation on the LANGEVIN equation [34]

$$m_i \ddot{\boldsymbol{x}}_{\boldsymbol{i}}(t) = -\gamma \dot{\boldsymbol{x}}_{\boldsymbol{i}}(t) + \boldsymbol{f}(t) + \boldsymbol{F}_{\text{ext}}(\boldsymbol{x}_i, t)\,, \qquad (1)$$

with the dot denoting time derivative and boldness indicating vectors. A Langevin equation is an ordinary differential equation with a random term added. This equation can be intuitively understood as an extension to NEWTON's $\boldsymbol{F}(\boldsymbol{x},t) = m\ddot{\boldsymbol{x}}(t)$ by adding both friction (friction coefficient γ) which is linear to the velocity and a random force $\boldsymbol{f}$ which describes the accumulated effect of the system interaction with the individual particle. This interaction between the particles and the system results in the diffusion of the particles. In particular $\boldsymbol{f}$ is commonly sampled from a Gaussian distribution to model the aggregate system behavior.

Values for the constants involved can generally be found using known equations like STOKES-EINSTEIN relation ($D_i = \frac{k_B T}{\gamma_i} = \frac{k_B T}{6\pi \eta r_i}$), as well as system properties like viscosity η.

The $\boldsymbol{F}_{\text{ext}}(\boldsymbol{x},t)$ expression is the sum of all external forces on a particle. Two types of external forces are distinguished:

- global forces are those, that act on one particle, independent of other particles. Global forces functions may be specified as arbitrary functions of the individual particle state (most importantly its position) and the global state of the simulator. This may be used to model, for example, some global current in a blood stream or some other directed cellular motion (as is done in "A model of vesicular transport" section below).
- pair-wise forces are those, that originate in the interaction of two particles like the non-reaction force (Eq. 2), reaction force (Eq. 7) or a division force (Eq. 6).

Non-reactive force

In order to model excluded volumes, we introduce a soft sphere potential between two non-reacting particles. This is a common and natural way to model excluded volumes in force-based approaches. In addition, a hard-sphere potential would imply more efforts numerically, as it would lead to a discontinuous force. An established choice for the modeling of elastic (soft-sphere) collision is the HERTZ collision theory [35]:

$$F_{non-react}(d) = \frac{4}{3}E^{*}\sqrt{r \cdot d^{3}} \tag{2}$$

where d is the overlap of the particles, $E^{*} = \left(\frac{1-\nu_1^2}{E_1} + \frac{1-\nu_2^2}{E_2}\right)^{-1}$ results from the YOUNG'S modulus E and POISSON'S ratio ν and $r = \left(\frac{1}{r_1} + \frac{1}{r_2}\right)^{-1}$ from their radii. However, for many sparse applications, any kind of polynomial would suffice, as the exact dynamics of the elastic collision do not matter, when the mean time between collisions is sufficiently large.

Reactions in mL-Force

Reactions between particles in ML-Force include zero, first and second order reactions. ML-Force supports a rule-based description of models. Whereas lower order reactions are sampled from an exponential distribution, second order reactions are calculated, similarly as the movement of particles, based on forces.

Lower order reactions

Simulation of zeroth and first order reaction, e.g., $\perp \rightarrow p_j$, or $p_i \rightarrow p'_i$, or $p_i \rightarrow p_j + p_k$ is a well understood problem (e.g., in SpringSaLaT [16]). The most basic approach is to sample the rate every (sufficiently small) time-step and check it against a random number.

In ML-Force, rates can be defined by arbitrary functions that access the attributes and contents of the particles involved in a reaction, e.g., for a first order reaction:

$$p_i(v_1, \ldots v_n)[p_1 + \ldots + p_m] \rightarrow \ldots @\lambda(k, v_1, \ldots v_n, [p_1, \ldots, p_m]) \ . \tag{3}$$

here λ denotes a function defined by the user, k a parameter, which might refer to the system globally, e.g., in terms of temperature and viscosity, or to a kinetic constant. The result of the function λ serves as the parameter for the exponential random distribution, that describes the probability density of the time until the reaction occurs. Thus in ML-Force, zero and first order reactions are treated semantically the same as in stochastic simulation algorithms [9]. In stochastic, non-spatial and spatial, simulation approaches (see Fig. 1a and e) exponentially distributed stochastic events form the basis of all, including second order, reactions.

In ML-Force the rate is sampled every time step. Instead of propagating the simulation until the exact time point a reaction occurs, we check (stochastically), if the reaction occurs in the current time step. Sampling the rate every time step is useful if the information $(v_1, \ldots, v_n, [p_1, \ldots, p_m])$, on which the rates depend, frequently changes. If this information remains largely constant over time, a scheduling approach may be more efficient computationally, as events do not have to be rescheduled frequently [36]. In executing ML-Force models so far, zeroth and first order reactions have only been responsible for a small fraction of the required computing-time.

In addition to computational considerations, the error introduced by this procedure needs to be considered. Due to the discretization of time (explicit time steps) no-more than one reaction per particle may take place per time step. This is important to keep in mind for future models. In the application domain the mean time between reactions tend to be large compared to the very small time steps that result from the fast diffusion of particles. As long as this is the case, the error is negligible. With resonable technical effort, this problem could also be avoided by allowing multiple reactions per time step.

One kind of lower order reaction is the changing of particle size, in terms of a particle's radius r. In ML-Force arbitrary functions can be applied. The new value of the radius r, i.e., v'_1, of a particle $p_i \in S_j(r, \ldots)$ is calculated in dependence of some global parameters k, and the current state of the particle in terms of its attribute values $v_1, \ldots v_n$, and its content $p_1 \ldots p_m$, such that

$$\begin{aligned} & p_i(v_1, \ldots v_n)[p_1 + \ldots + p_m] \\ & \rightarrow p_i(\lambda(k, v_1, \ldots v_n, [p_1, \ldots, p_m]), v_2 \ldots, v_n)[p_1 + \ldots + p_m] \ldots @\lambda(\ldots) \ . \end{aligned}$$

By default, whenever a particle is created, it is first put into the system as a very small particle and then quickly grows continuously to its desired size. This continuous change of particle radii is also used, whenever the radius of a particle is changed in the model by any kind of reaction. Carrying the changes out suddenly (so not slowly over multiple integration steps) would potentially lead to integration errors, especially in systems with nesting. A sudden change in size could for example lead to a high particle overlap, which in turn would create a high unrealistic potential energy leading to a strong force and thus a disturbance in the system, e.g., numerical heating. Slow changes in size, allow for each artificial change in energy to slowly dissipate e.g., in friction due to the numerically very stable Langevin model. Naturally, it is also possible to create a particle within or at the surface of another particle. Those new positions are chosen randomly. At the surface this means that the new particle's shell barely touches the old particle's shell.

Compartment division

An interesting kind of first order reaction is that of a cellular compartment (or specifically a cell) dividing.

$$\begin{aligned} p_i(v_1,\dots v_n)[p_1+\dots+p_m] \rightarrow\, & p_{i1}\big(\lambda_{11}(y),\dots,\lambda_{n1}(y)\big)\big[\lambda_{c1}(y)\big] \\ & + p_{i2}\big(\lambda_{12}(y),\dots,\lambda_{n2}(y)\big)\big[\lambda_{c2}(y)\big] \qquad @\lambda(\dots),\{a\}\,, \end{aligned} \tag{4}$$

with $y = k, v_1, \dots v_n, [p_1, \dots, p_m]$ or in shorthand, omitting attributes and contained particles

$$p_i \rightarrow p_{i1} + p_{i2} @\lambda(\dots), \{a\} \ . \tag{5}$$

In ML-Force this is realized in a continuous and nesting compatible manner. As a first order reaction, the event that the division starts is sampled from an exponential distribution. The simulator temporarily introduces an "ignoring" relation between particles. This can be best explained using an example: Imagine a particle p_i containing many particles $\hat{p}_1 \dots \hat{p}_m$. If now the reaction for p_i to split into p_{i1} and p_{i2} triggers, a few things happen. First of all, each particle $\hat{p}_1 \dots \hat{p}_m$ gets assigned to remain in either p_{i1} or p_{i2}. By default, this assignment happens randomly. If required by the model, a specific splitting function may be specified by the user. Therefore, two arbitrary functions, i.e., $\lambda_{c1}, \lambda_{c2}$ are applied. The results of these applications form the new content of the newly generated particles, i.e., p_{i1} and p_{i2} respectively. The functions λ_{i1} to λ_{n1}, respectively λ_{i2} to λ_{n2} determine the new attribute values of the particles. If attribute values remain the same, simply the old values v_i can be used.

Each $\hat{p}_i$ that will remain in p_{i1} is set to ignore the particle p_{i2} and vice versa. Also p_{i1} and p_{i2} are set to ignore one another. Now an external force is applied to push p_{i1} and p_{i2} apart

$$F_{\text{divison}} = m_i \cdot a_{\text{division}} \ . \tag{6}$$

The force is calculated based on the acceleration a that is part of this particular reaction type (Eq.: 4, 5) in ML-Force and the mass of the involved particles (which again depend on the respective radii and their density ρ). This is chosen, instead of a uniform force, to create a uniform motion of all particles. In principle however, any force expression suitable to the model could be used here, like a half sine wave for example. Once p_{i1} and p_{i2} are fully outside one another, all the newly introduced ignore relations are removed and regular propagation proceeds.

The principle of continuous change in particle size also applies here. At the beginning both particles have the same size. If p_{i1} and p_{i2} are to change their size throughout the division (see example in "The yeast model" section), this change is carried out throughout the division process. Initially, both p_{i1} and p_{i2} share position and radius. The further they move out, the closer their radii become to the desired ones. The fusion of compartments could be added in an identical (if reversed) fashion. However so far, it has not been implemented yet.

Bimolecular reactions

Handling bimolecular reactions in particle-based approaches is rather challenging. Whilst solutions exist when not considering excluded volumes (e.g. Smoldyn [12]), there has not yet been agreement among the community on the best way to approach this problem. This is evident from the many different approaches of recent tools [5, 12, 16, 17, 37]. Two processes contribute to the macroscopic rate,

- the *diffusion process* describes how likely it is that two particle actually move into the vicinity of each other. This upper bound to the reaction rate is well described by the SMOLUCHOWSKI-Theory [38] (elaborated in the Additional file 1)
- the *microscopic activation process* describes how likely it is for two particle to react once they are close to each other [39, 40].

The rate is considered to be diffusion limited if every interaction results in a reaction, all other cases are considered to be activation limited. We propose to use forces for both processes. This means also the microscopic activation process is based on forces. In case a reaction can occur between two particles, the non-reactive force (Eq.: 2) has to be replaced by a reactive force. We used a rather simple approach which is calculated based on a degree of overlapping required d^* and the work to cross the energy barrier between the two particles. Both, the degree of overlapping and the energy barrier, are specific for the reaction to occur and have to be defined by the modeler.

$$F_{react} = \frac{W}{d^*} \tag{7}$$

where d^* denotes the required overlap of the particles and W the work required to cross the energy barrier. It should be noted due to the modular design (see "Simulator and implementation" section) within the simulator these functions can easily be changed. Suppose we were to look at the reaction $p_i + p_j \rightarrow p_k\{d^*, W\}$. Whenever p_i and p_j overlap, they start to repel each other with the force F_{react}. Once they overlap more than d^*, we consider the reaction to be triggered. Thus, the energy barrier and the required overlap of the microscopic interaction potential determines the likelihood of the reaction to take place. The higher the energy barrier, the less likely the particles will overlap sufficiently. If the energy barrier W is negligible, the diffusion is the only limit to the macroscopic reaction rate in the reaction process, it is *diffusion limited*.

This type of bimolecular reaction handling with forces, allows also for self-consistent handling of dynamic nesting (and "unnesting") reactions, i.e., one particle shuttles into (or out of) another particle. In principle the same concepts apply as for the regular bimolecular reaction. Looking at the reaction $p_i + p_j \rightarrow p_i[p_j]\{d^*, W\}$ (and the same for $p_i[p_j] \rightarrow p_i + p_j\{d^*, W\}$), first the particles need to diffuse into proximity of one another. Once particles overlap, they repel each other. If p_i's diameter has fully passed over p_j's outline, the nesting reaction is triggered. However at this point p_j is already fully contained in p_i so there is no jerkiness in the simulation. For example, if there is no unnesting reaction defined for these particles, the only thing, that changes, is that, if p_j were to move towards again touching p_i, it would perceive the force $F_{non-react}$ (pulling inwards). This ensures a soft-sphere-style excluded volume. Similar as for the regular $p_i + p_j \rightarrow p_k$ type bimolecular reaction, a high energy barrier makes for a less, and a low for a more likely reaction.

The advantage of this force-based approach, is the smooth integration of the nesting process. On the other hand, it does require a very small timestep, compared to other methods. Furthermore, in the current situation, the parameterization of bimolecular reactions is not optimal. Ideally, the modeler would specify the macroscopic rate, instead of the microscopic barrier. In the future we plan on investigating this further and, using principles from thermodynamics, provide an automatic conversion method. This would mean, by modeling the system as a thermodynamic ensemble we could determine the statistical likelihood of a particle surpassing a certain energy threshold during interaction and put this into explicit mathematical relation to the reaction rate.

Simulator and implementation

The simulator has been implemented using `C++14` and has been tested on both Windows and Linux (using CMake-build system) and is available at https://git.informatik.uni-rostock.de/mosi/ml-force-publication.

The user-interface is built around a rudimentary embedded domain specific language (DSL). An example and explanation can be found in the provided code snippet below. In the long run, it would be advantageous to create an external DSL using techniques like transpiling, to still have the performance benefits of a compiling language. A more in-depth comparison and discussion of DSLs can be found in the literature [41]. The developed embedded DSL bears some similarities with the ℓ-language [42] which also supports an imperative rather than a declarative approach towards modeling.

As can be seen in Fig. 3, the software pursues a component-based design with a clear separation of concern [43]. Its components are arranged around the main *simulation engine*. The simulation engine itself is very basic and designed to make as little assumptions about the underlying semantics as possible. This simulation engine interacts with the components via a well defined, small interface. Each of the key ingredients of the simulation is separated into exchangeable components, namely the integrator, collision detection engine, and the visualizer, as well as the overarching modeling layer which provides the interface between the simulation engine and DSL. They are invoked by the simulator and implement the functionality introduced in the previous section. The simulator is responsible for executing the lower and higher order reactions by calling the specified functions of the model. First it is checked whether lower order reactions trigger in this step. If so, those are executed. Based on the forces a numerical integrator calculates the new positions of the particles. To identify the particles that interact, a collision detection engines is invoked. Afterward the simulator calculates the bimolecular forces and executes the biomolecular reactions accordingly. The calculated state can be subject to visualizations or other reporting mechanisms.

Integrator Currently integration is carried out using a two-step kick-drift-kick integrator [44] of the stochastic LANGEVIN equation. This integration includes velocity, and is thus fairly costly numerically. However as an alternative, one could select an integrator with immediate dissipation of momentum, that would allow for a considerably larger timestep. The choice for the more complex integrator was made as ML-Force makes heavy use of various forces and carrying out the more complex integration, helps in terms of correctness. In the current implementation the time step is chosen adaptively based on the smallest particle and the fastest velocity currently in the system, as $\Delta t = \frac{r_{\min}}{v_{\max}\, g}$ with g as the parameter of *granularity*, that describes the precision of the integration.

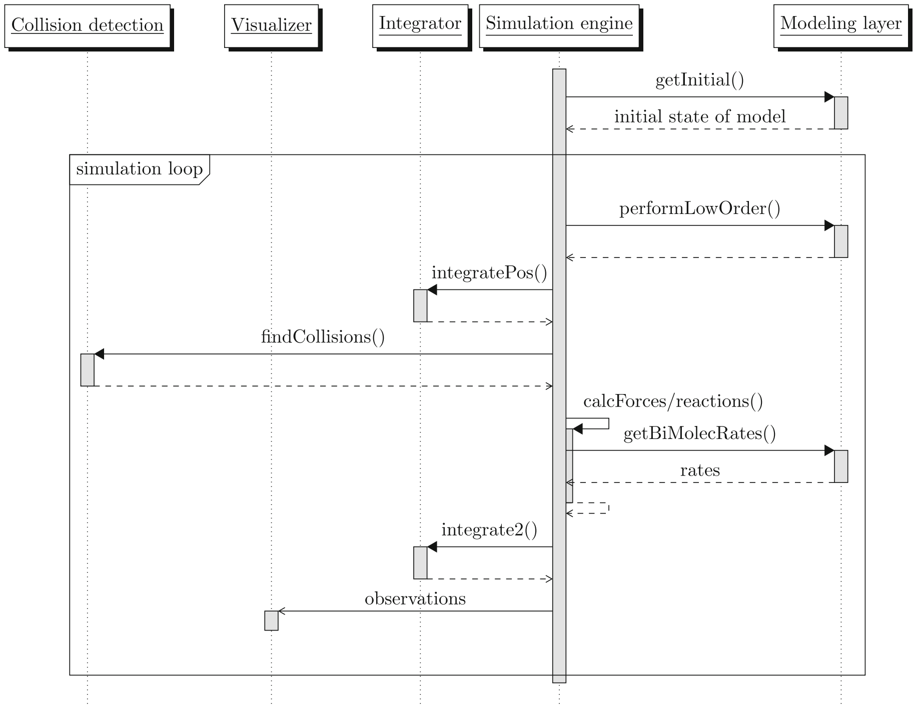

Fig. 3 The different components of the software and their respective tasks, as called upon by the main simulator in an UML sequence diagram. The different components of the software are interchangeable and only interact via their interface to the main *Simulator*-component. This simulator also provides the layer between the model (as specified using the DSL).The software implementation is described on in "Simulator and implementation" section. The second integration of the particles is optional. For example the current integrator advances the velocity in the second integration step

Collision detection engine Finding possible interactions of particles using the collision detection engine is one of the key drivers of runtime in the current implementation. Several collision detection engines have been implemented, using different algorithms and parallelization. For example, for highly nested particles, the collision-detection problem lends itself to be solved by a specifically tailored algorithm and a massively parallel execution on the GPU [28].

Visualizer Currently 3 basic visualizers have been created. One to just create an output text file, and two based on CImg [45], one for live viewing and one for generating movie files. The visualizers are encapsulated in a separate thread to minimize overhead. One more advanced visualization has been implemented that maps the simulation to 3D space using OpenGL [46]. It offers some useful features for the case studies (see "Results" section), such as tracking individual particles in space.

Modeling layer The modeling layer makes heavy use of λ-functions in order to encapsulate model specific behavior into single function calls for the simulator. λ-functions here are a `C++11` feature that allows to define anonymous functions, that can be used as parameters in a functional-like programming style. This is facilitated by various templates that have been defined in the DSL. For example, after each reaction, the simulator invokes the *post reaction function* (see e.g. afterFuncA in listing 2). In the modeling layer this function is specified by the user and denotes the results of applying a rule, such as particles changing state, being degraded or created.

This simplifies the implementation of the simulator and gives a unified and lean interface for implementing further features. The simulator also provides a set of units (via the `units::` namespace) to the modeler, allowing for consistent parameterization. For simplicity the reactants in the DSL are always denoted as A and B. The particular type of A and B is then specified via the *as_A* etc. functions. In general the DSL is very expressive, as all rates and reactions are described as λ-functions. These can be of arbitrary complexity. However, it should be noted that the current DSL presents only a proof of concept rather than a language that allows a succinct description of ML-Force models. Therefore, further efforts will be dedicated to improving the design of the language.

Listing 1 This is an extract from the lipid raft model that describes how a receptor moves into a lipid raft. The embedded domain specific language is built around objects. *Species*, *reaction* or *property* objects can be put in relation to one another. Some macro-functions are made available, to minimize boilerplate code (e.g. the `ADD_SPECIES`-macro does not only create a species object, but also sets a string property with that object's name). The `*_FUNC` are wrappers for different λ-functions that can be used to specify arbitrary complex behavior and rates.

```
/* declaring species and properties */
 ADD_SPECIES(lipid_raft);
 radius(lipid_raft) = 200 * units::nano_meter;
local_density(lipid_raft) = nature::protein_density
     ;
local_viscosity(lipid_raft) = 10E-3 * units::pascal
      * units::second;
colorType(lipid_raft) = 8;

ADD_SPECIES(protein);
local_density(protein) = nature::protein_density;
radius(protein) = 3.7 * units::nano_meter;
colorType(protein) = 1;

/* declaring A + B -> B[A] reaction */
reaction Protein_in(as_{A}(protein), as_{B}(
     lipid_raft));
Protein_in.barrier = 0. * units::kilogram *
                 pow(units::nano_meter/units::second
                      ,2);
Protein_in.AinB();

/* inital state */
putHere(lipid_raft, box_size / 2, box_size / 2, 0);
putSomewhere(protein, 200);
```

Results

In order to test the ML-Force simulator and its implementation, we conduct a few case studies. These include simple test cases to investigate correctness and more complex models. The latter are inspired by realistic cell biological models, make use of the features in ML-Force, and, thus, demonstrate their usefulness. Both, the basic test models and the more realistic ones, can be found in the Additional file 1.

Testing correctness

As other particle-based simulation approaches [16, 17, 37, 47] we test the correctness of our simulator, based on a set of simple reactions which we simulate to compare the achieved results to theory. To those tests belong the creation of a particle with a constant rate as a 0 order reaction ($\emptyset \xrightarrow{k} A$), the decay with a rate $A \xrightarrow{k} \emptyset$ as first order reaction, and irreversible and reversible second order reactions $A + B \leftrightarrow C$ in the diffusion limited (intrinsic rate $k_{int} = \infty$) case. A comparison of simulation results and theory shows overall a good agreement with the theory (see Additional file 1). However, those tests also reveal the decisive role of selected parameters to determine forces ("Bimolecular reactions" section) and time steps. For their selection, suitable computational support should be provided. As a form of pre-processing step, energy barriers respectively accelerations for individual reactions could be generated by automatically fitting simulation results to theory. A similar strategy can help selecting suitable time steps, so that the diffusion of particles is correctly simulated (see Additional file 1).

A model of vesicular transport

Heinrich and Rapoport proposed a model of vesicular transport [48]. Although the model describes a spatial process, it was formalized by means of differential equations.

The vesicular transport model refers to two cellular compartments which exchange membrane-bound soluble N-ethyl-maleimide–sensitive factor attachment protein receptors (SNAREs) and cargo proteins with the help of differently coated vesicles. These are budding from the compartments and move, driven by motor-proteins, to the other compartment where they are fused. Thereby the coat of vesicles defines, which type of SNAREs (X or Y), cargo proteins and motor-proteins are bound to them. Since different types of motor-proteins move in different direction, SNARE X and Y accumulate in different compartments.

By modeling this process in a spatial regime, the problem starts, as in each particle-based approach, with the description of the species. Each species needs a radius and a diffusion coefficient. The ML-Force simulator determines the diffusion based on the temperature, solvent viscosity and particle radius according to the Stokes-Einstein equation (see Additional file 1). The cell, compartments and the vesicles are represented as particles while the SNAREs are attributes of particles, which change during the budding and fusion processes. As Klann et al. [2] has already shown through the translation of the original ODE model into a spatial agent-based model, this transport relies on a directed movement of vesicles. Therefore Klann et al. added a cytoskeleton structure to the cell on which the motor-proteins moved along. In ML-Force the direction is determined by forces which take the type of particles and their attributes into account. An external force field is introduced which is shaped like a dipole field with the compartments as poles. Based on the coat of the vesicles a force along this field is added to their movement, which lead them to one of the compartments. To test the sorting mechanism of this simple vesicular transport model we study a system with 2 types of SNAREs (X and Y) with the same amount. As in [48], we initialize the model: the first compartment is nine times bigger ($V_1 \approx 0.118\ \mu m^3$) than the second one ($V_2 \approx 0.013\ \mu m^3$) and contains 90% of the SNAREs (X and Y) ($N_1^{X/Y}$ = 90000 and $N_2^{X/Y}$ = 10000). Since the budding process of the vesicles depends on the volume of the compartments, they should reach an equal size and the SNAREs should be sorted.

As shown in Fig. 4 both compartments reach an equal volume and SNARE X accumulates in compartment 1 while SNARE Y accumulates in compartment 2 (not shown).

To show that the directed motion is essential for the sorting, as a control we run the simulation without the external force field and with compartments with equal initial size ($V_{1/2} \approx 0.065\ \mu m^3$) and equally initially distributed SNAREs ($N_{1/2}^{X/Y} = 50000$). In a purely Brownian model, i.e., a model where all particles perform Brownian motion, no sorting takes place and both compartments are just shrinking as shown in Fig. 5. It can also be seen that the vesicle's slowly diffusion around the compartments and there behavior is independent of their coating. In contrast the vesicles in the model with a directed movement either re-fuse with the compartment of origin or move to the other compartment based on their coating. As a result vesicles in this model fuse with a specific compartment after a short time and by this they are sorting the SNAREs. This simplified model illustrates how external forces can simulate the directed motion in a biological system.

The yeast model

A more complex model that showcases more of ML-Force's abilities is based on an illustrative yeast model [30], which in turn uses the model of the cell cycle proposed by TYSON [49]. In the original model each yeast cell can grow and contains five different proteins whose amounts oscillate periodically and trigger the fission of a yeast cell. Each cell has a mating type (P or M) which can change during the cell fission. Depending on the type the yeast releases pheromones (M-factor and P-factor, respectively) to inhibit the cell cycle of cells with opposite mating type. In addition, the type M cells secrete a protease called Sxa2 which inactivates the P-factor pheromone that stems from the type P cells.

With the ML-Force model, we want to check whether cells of identical mating types accumulate in some areas and inhibit the cell cycle of cells with opposite mating type in this area. In ML-Force, each species of the system can be modeled as a particle, as shown in the upper part of Fig. 6. The problem of this approach are the different time scales of the dynamics. The cell cycle lasts about 120 min, while the diffusion of the proteins needs a time step in the sub nanosecond regime. In addition, for our question, a spatial simulation of the intra-cellular dynamics is not required. Therefore, we represent the pheromones and the cells as particles and the proteins involved in the cell cycle (as well as the cell cycle) as attributes of the cells, as shown in the lower part of Fig. 6. Whereas cells only move during the fission process in the x-y plane, pheromones and Sxa2 diffuse in the extracellular medium and disappear when they reach the borders of the test volume.

As shown in Fig. 7 after 900 min there are twice as many P-type cells as M-type cells. Furthermore some of the M-type cells are old (large) and have a highly inhibited cell cycle, which makes it likely that these cell will enter apoptosis before they can divide.
Attributed particles and arbitrary functions allow to model the intra-cellular dynamics non-spatially as first order reaction.

Listing 2 The creation of cyclin inside the cell is realized as a first order reaction that changes the attribute Y.

```
// 1) cyclin synthesis in cell 0 -> Y
reaction Y_synthesis(as_{A}(cell));
Y_synthesis.rateFunc = R_FUNC(k1);
Y_synthesis.afterFuncA = G_FUNC(Y(A) = Y(A) + 1;);
```

Nevertheless these non-spatial dynamic can influence the particle, here by triggering the division of a cell.

Listing 3 After the attribute 'Ma' of the cell drops below the threshold t9 the division reaction is executed and new cell namely B is created.

```
9) cell division (transition from M->G1)
reaction division(as_{A}(cell));
division.reacPossibleFunc = B_FUNC(Ma(A) < t9 phase
     (A) == 3);// #M_A < t9
division.rate = k9;
division.divide(as_{B}(cell), 1.E-8);
division.afterFuncA =
G_FUNC(volume(B) = 0.5 * volume(A); volume(A) = 0.5
      * volume(A););
```

It is also possible for a particle to influence the attributes (non-spatial) of particle.

Listing 4 The collision of the two particles cell and F_M results in the 'death' of F_M and the change of the attribute 'receptor_occupied' of the particle cell.

```
/* cell(n) + F_M -> cell(n+1) */
reaction bind_pherome_{M}(as_{A}(cell), as_{B}(F_M)
     );
bind_pherome_M.barrier = 0.;
bind_pherome_M.setOverlapp(0.01);
bind_pherome_M.reacPossibleFunc =
    B_FUNC(receptor_occupied(A) < receptor_total(A)
         && mating(A) == 1);
bind_pherome_M.afterFuncA =
    G_FUNC(receptor_occupied(A) = receptor_occupied
         (A) + 1;);
bind_pherome_M.Bdead();
```

For example this is the case in the yeast model when a pheromone reacts with a cell (both particles) and inhibits the cell cycle of the cell. This combination of spatial and non-spatial behaviors enables the simulation of systems which include processes on different temporal scales.

Lipid rafts model

The role of lipid rafts in inducing and promoting receptor accumulation within the cell membrane and the recruitment and binding of proteins from the cytosol has been

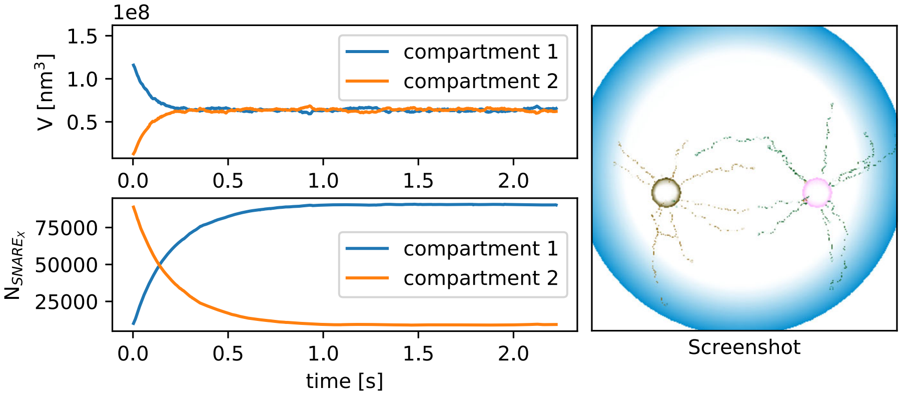

Fig. 4 Volume and amount of SNARE$_X$ over time and Snapshot from the vesicle transport model. In the case of a directed movement the volume of the compartments become equal and SNARE$_X$ accumulate in compartment 1 as shown on the left. The right side shows a screenshot from the simulation where you can see the movement along the external force field. The blue sphere is the cell with a radius of 2.5 μm

subject to several computational studies [50, 51]. A small model of receptor lipid raft interaction which is inspired by a sub-model of Wnt signaling [52] shall illustrate the ability of ML-Force to support dynamic nesting. The model consists of two proteins (LRP 5/6 and CK1-γ) which diffuse in the membrane and may enter or leave lipid rafts. In the model, the mobility of proteins is reduced in lipid rafts by a 10 times larger viscosity in the rafts compared to the rest of the membrane. Both proteins have different affinities to enter the lipid raft which is modeled by different energy barriers in the nesting reactions. LRP 5/6 has to pass a barrier of $12 \cdot 10^{-21}$ J which was estimated from the distribution of the kinetic energy for demonstration purposes. It shows how the energy barrier at a bimolecular reaction can be used to slow down the reaction. CK1-γ can enter the lipid raft freely ($E_{barrier}$ = 0 J). By choosing a larger energy barrier for LRP 5/6 while they are entering the lipid raft we model their lower affinity to the lipid raft and can show how the energy barrier at a bimolecular reaction can be used to slow down the reaction. To show how these functions influence the accumulation of the proteins in the lipid raft we run a 2D simulation. In this simple experiment, our model contains a single lipid raft which covers 25% of the surface. The 10 times larger viscosity of the lipid raft causes a 10 times lower diffusion coefficient in the raft compared to the rest of the membrane. The simulation starts with 200 LRP 5/6 and 200 CK1-γ particles outside the lipid raft. To enter the lipid raft, the particles need to overcome the energy barriers described above. After a particle entered a lipid raft they can leave it again without any energy barrier.

As expected, Fig. 8 shows that due to the slower diffusion in the lipid raft CK1-γ accumulates in the lipid raft.

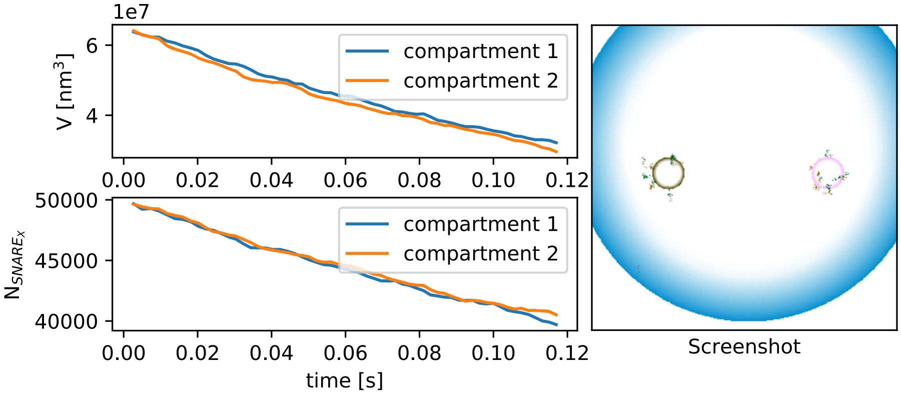

Fig. 5 Volume and amount of SNARE$_X$ over time and snapshot from the vesicle transport model. In the case of pure brownian motion the volume of the compartments decreases and no sorting of SNAREs takes place as shown on the left. The right side shows a screenshot from the simulation where you can see the random movement of the vesicles. The blue sphere is the cell with a radius of 2.5 μm

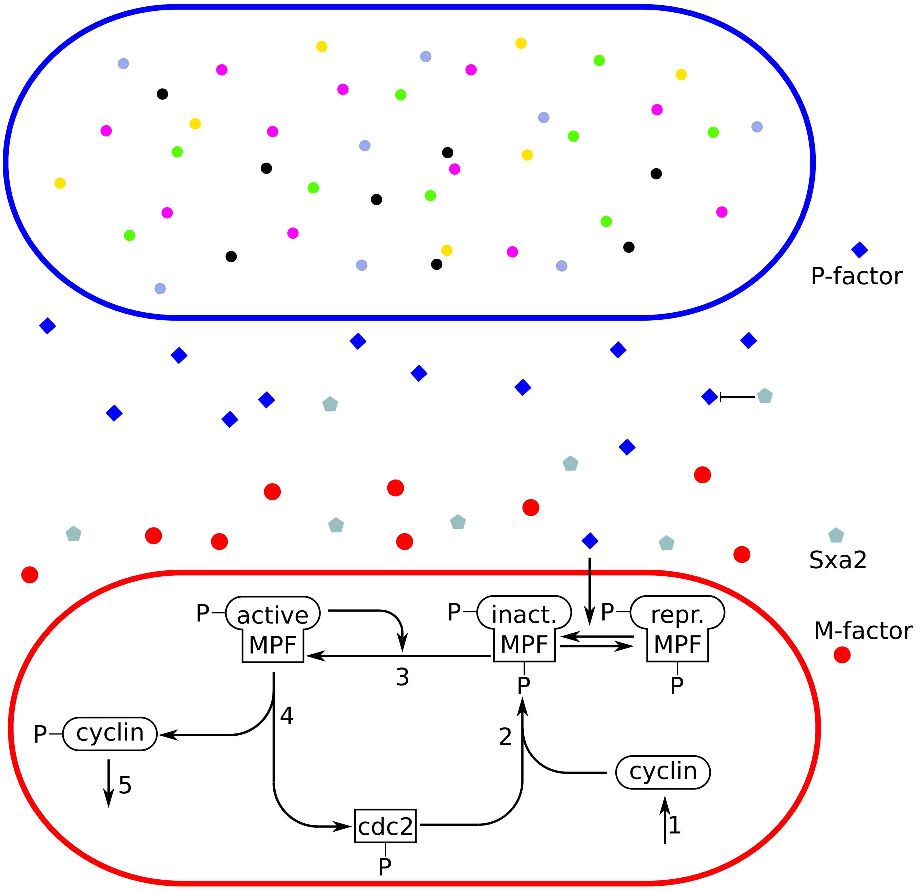

Fig. 6 Sketch of the spatial/non-spatial cell cycle. Cell cycle of a yeast cell with five/six different proteins. The upper cell shows the cell cycle modeled with particles. As the proteins interaction within the cell occurs on a vastly different timescale than the behavior of the cell this is not a feasible approach. Instead the internal dynamics are modeled by a non-spatial approach as shown at the bottom

The amount of LRP 5/6 which accumulates in the lipid raft is lower due to the energy barrier which hampers receptors to enter the lipid raft.

The model shows how the repulsive force during a bimolecular reaction can lower the reaction rate. Also the dynamic nesting of particles can change their behavior (here the diffusion of the nested particle) during simulation. It is also possible to restrict reactions to only occur if particles are nested inside a specific type of particle, such as the phosphorylation of LRP 5/6 which is constrained to lipid rafts.

Discussion

The benchmark models which range from the multicellular yeast model to the subcellular lipid raft model have shown the usefulness of the realized features. ML-Force expands upon the state of the art. The combination of compartmental dynamics and particle simulation based on forces is a unique feature of ML-Force. This feature has been used in the lipid raft model ("Lipid rafts model" section). ML-Force provides a consistent force-based semantics for spatially resolved dynamic nesting with excluded volumes. Arbitrary functions and attributes have proven a necessity for a concise and computationally feasible realization of the yeast model in ML-Force ("The yeast model" section) where they are used for the non-spatial stochastic simulation of the cell cycle. In ML-Force, intra-particle dynamics can be simulated either spatially resolved by nested particles or in a non-spatial stochastic manner. Applying lower and higher spatial resolutions on demand facilitates modeling and simulating complex spatial models. The vesicle transport model ("A model of vesicular transport" section) relied on the possibility to let particles grow and to define global force functions that apply to all particles independently of other particles to simulate the directed movement of vesicles. The global force functions provide an additional means for the modeler for abstraction (in this case from the cytoskeleton

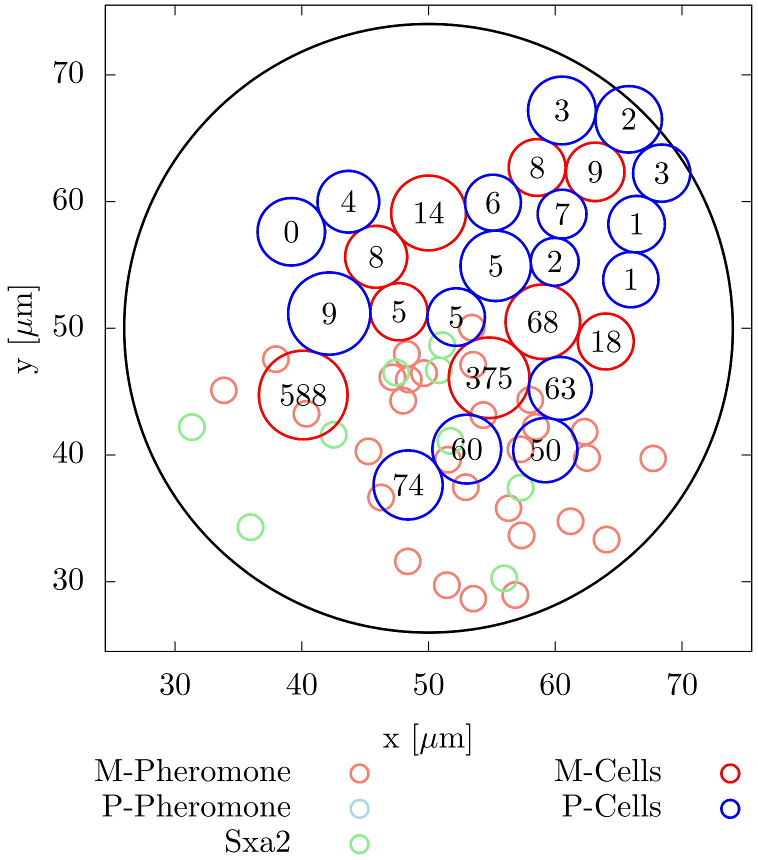

Fig. 7 Spread of yeast cells. Distribution of yeast cells after 900 min. The numbers in the cells indicate how many pheromones are bound to them. For example the M-type cell on the left site has a highly inhibited cell cycle with 568 bounded pheromones. It is very likely that this cell will enter apoptosis, before it divides

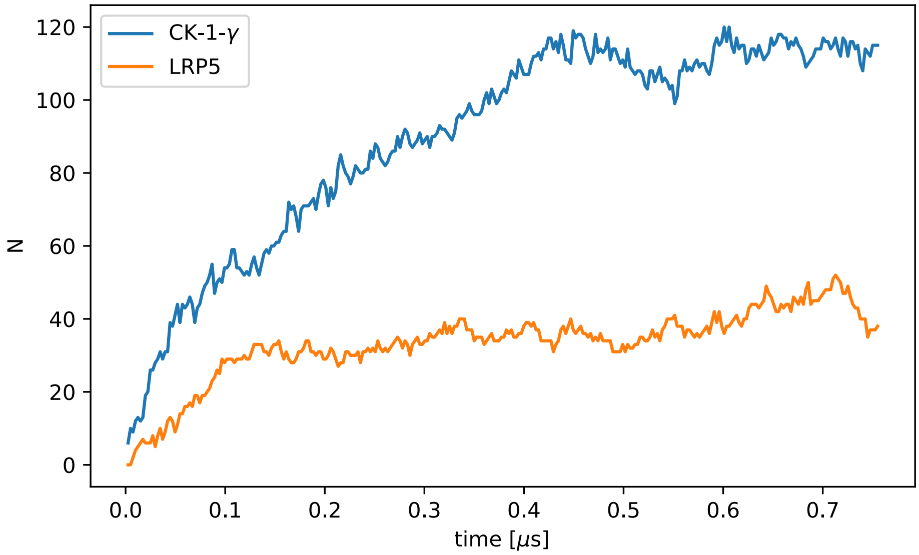

Fig. 8 Accumulation of proteins in lipid rafts. Amount of LPR 5/6 and CK1-γ inside the lipid raft. Due to the lower affinity of LRP 5/6 to the lipid raft (realized by an energy barrier) their concentration is lower then the one of CK1-γ

structure on which the motor-proteins move). Along all other dynamics this is seamlessly integrated into the force-based semantics of ML-Force.

Conclusions

The particle-based modeling and simulation approach ML-Force combines excluded volumes and forces with support for dynamic nesting (compartmental dynamics) and the ability of constraining cellular dynamics by arbitrary attributes and functions. Whereas other simulation approaches have treated compartmental dynamics, such as particles entering or leaving a compartment, or compartmental fission or fusion, discretely, in ML-Force those are simulated in a smooth, continuous manner. Thereby, structural dynamics are seamlessly integrated into the particle-based simulation governed by the LANGEVIN equation. However, this adds to the requirements the integrator has to face and consequently the induced calculation efforts.

ML-Force utilizes a rule-based modeling approach. In combination with attributed species of arbitrary types and arbitrary functions that work on particles, their attributes and content, this contributes to the expressiveness and flexibility of ML-Force. In particular, part of a spatial model can be executed in a non-spatial stochastic manner. Abstracting from spatial details on demand reduces the complexity of the model and the induced calculation effort and allows applying ML-Force to a wider range of cell biological systems. The possibility to define force functions that act on all particles independently of others provides another valuable means for abstraction in ML-Force.

Currently, ML-Force is being used in a combined in-vitro and in-silico study to analyze the impact of external electrical fields on cellular membranes. Future work will be aimed at enhancing the readability of ML-Force models by a more declarative expression of rules. In addition, user support for selecting time steps and suitable parameters for force calculation needs to be provided. Also advanced numerical integration schemes that are exploited by other particle-based simulators and their impact on compartmental dynamics shall be analyzed.

Supplementary information

Supplementary information accompanies this paper at https://doi.org/10.1186/s12859-019-3092-y.

Additional file 1: The additional file contains tests of of basic functions and the code of the models from the case study.

Abbreviations

CK1-γ: casein kinase 1-γ; DSL: domain specific language; LRP5/6: low-density lipoprotein receptor-related protein 5/6; ML: multi-level; RDME: reaction diffusion master equation; SNARE: soluble N-ethyl-maleimide–sensitive factor attachment protein receptors;

Acknowledgements

Felix Hauptmann implemented the OpenGL visualizer.

Authors' contributions

TK conceived the method. TK designed and implemented the Software. PH carried out the Case study. TK, PH and AMU wrote the manuscript. AMU supervised the project. All authors have read and approved the manuscript.

Funding

Financial support (design of the study, analysis, and interpretation of simulations and in writing the manuscript) was provided by the Deutsche Forschungsgemeinschaft (DFG) research grant 'ESCeMMo' (UH-66/13) and the DFG collaborative research centre 'ELAINE' (CRC 1270). We acknowledge financial support by Deutsche Forschungsgemeinschaft (DFG) and Universität Rostock within the funding programme Open Access Publishing.

Availability of data and materials

The datasets used and/or analysed during the current study are available from the corresponding author on reasonable request.

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Received: 14 March 2019 Accepted: 10 September 2019
Published online: 27 November 2019

References

1. Scott J. D., Pawson T. Cell Signaling in Space and Time: Where Proteins Come Together and When They're Apart. Science. 2009;326(5957): 1220–4. https://doi.org/10.1126/science.1175668 Accessed 23 July 2019.
2. Klann M, Koeppl H. Spatial Simulations in Systems Biology: From Molecules to Cells. Int J Mol Sci. 2012;13(6):7798–827. https://doi.org/10.3390/ijms13067798.
3. Bittig AT, Uhrmacher AM. Spatial modeling in cell biology at multiple levels. In: Proceedings of the 2010 Winter Simulation Conference. Baltimore; 2010. p. 608–19. https://doi.org/10.1109/WSC.2010.5679125.
4. Ridgway D, Broderick G, Ellison MJ. Accommodating space, time and randomness in network simulation. Curr Opin Biotechnol. 2006;17(5): 493–8. https://doi.org/10.1016/j.copbio.2006.08.004.
5. Takahashi K, Tănase-Nicola S, Wolde PRt. Spatio-temporal correlations can drastically change the response of a MAPK pathway. Proc Natl Acad Sci. 2010;107(6):2473–8. https://doi.org/10.1073/pnas.0906885107.
6. Cowan AE, Moraru II, Schaff JC, Slepchenko BM, Loew LM. Chapter 8 - Spatial Modeling of Cell Signaling Networks. In: Asthagiri AR, Arkin AP, editors. Methods in Cell Biology, Computational Methods in Cell Biology, vol. 110, pp. 192-221. Cambridge: Academic Press; 2012. https://doi.org/10.1016/B978-0-12-388403-9.00008-4. http://www.sciencedirect.com/science/article/pii/B9780123884039000084. Accessed 23 July 2019.
7. Takahashi K, Arjunan SNV, Tomita M. Space in systems biology of signaling pathways - towards intracellular molecular crowding in silico. FEBS Lett. 2005;579(8):1783–8. https://doi.org/10.1016/j.febslet.2005.01.072.
8. Schöneberg J, Ullrich A, Noé F. Simulation tools for particle-based reaction-diffusion dynamics in continuous space. BMC Biophysics. 2014;7(1):11. https://doi.org/10.1186/s13628-014-0011-5.
9. Gillespie DT. Stochastic Simulation of Chemical Kinetics. Annu Rev Phys Chem. 2007;58(1):35–55. https://doi.org/10.1146/annurev.physchem.58.032806.104637. Accessed 23 July 2019.
10. Regev A, Panina EM, Silverman W, Cardelli L, Shapiro E. BioAmbients: an abstraction for biological compartments. Theor Comput Sci. 2004;325(1): 141–67. https://doi.org/10.1016/j.tcs.2004.03.061. Accessed 23 July 2019.
11. Kerr RA, Bartol TM, Kaminsky B, Dittrich M, Chang J-CJ, Baden SB, Sejnowski TJ, Stiles JR. Fast Monte Carlo Simulation Methods For BIOLOGICAL Reaction-Diffusion Systems In Solution And On Surfaces. SIAM J Sci Comput Publ Soc Ind Appl Math. 2008;30(6):3126. https://doi.org/10.1137/070692017. Accessed 23 July 2019.

12. Andrews SS, Addy NJ, Brent R, Arkin AP. Detailed Simulations of Cell Biology with Smoldyn 2.1. PLoS Comput Biol. 2010;6(3):. https://doi.org/10.1371/journal.pcbi.1000705.
13. Blinov ML, Schaff JC, Vasilescu D, Moraru II, Bloom JE, Loew LM. Compartmental and Spatial Rule-Based Modeling with Virtual Cell. Biophys J. 2017;113(7):1365–72. https://doi.org/10.1016/j.bpj.2017.08.022.
14. Hattne J, Fange D, Elf J. Stochastic reaction-diffusion simulation with MesoRD. Bioinformatics. 2005;21(12):2923–4. https://doi.org/10.1093/bioinformatics/bti431. Accessed 23 July 2019.
15. Andrews SS. Smoldyn: particle-based simulation with rule-based modeling, improved molecular interaction and a library interface. Bioinformatics. 2017;33(5):710–7. https://doi.org/10.1093/bioinformatics/btw700.
16. Michalski P, Loew L. SpringSaLaD: A Spatial, Particle-Based Biochemical Simulation Platform with Excluded Volume. Biophys J. 2016;110(3): 523–29. https://doi.org/10.1016/j.bpj.2015.12.026.
17. Schöneberg J, Noé F. ReaDDy - A Software for Particle-Based Reaction-Diffusion Dynamics in Crowded Cellular Environments. PLoS ONE. 2013;8(9):74261. https://doi.org/10.1371/journal.pone.0074261.
18. Gruenert G, Ibrahim B, Lenser T, Lohel M, Hinze T, Dittrich P. Rule-based spatial modeling with diffusing, geometrically constrained molecules. BMC Bioinformatics. 2010;11(1):307. https://doi.org/10.1186/1471-2105-11-307.
19. Donovan RM, Tapia J-J, Sullivan DP, Faeder JR, Murphy RF, Dittrich M, Zuckerman DM. Unbiased Rare Event Sampling in Spatial Stochastic Systems Biology Models Using a Weighted Ensemble of Trajectories. PLOS Comput Biol. 2016;12(2):1004611. https://doi.org/10.1371/journal.pcbi.1004611.
20. Bittig AT, Uhrmacher AM. ML-Space: Hybrid Spatial Gillespie and Particle Simulation of Multi-level Rule-based Models in Cell Biology. IEEE/ACM Transactions on Computational Biology and Bioinformatics. 2016;PP(99): 1–16. https://doi.org/10.1109/TCBB.2016.2598162.
21. Flegg MB, Chapman SJ, Erban R. The two-regime method for optimizing stochastic reaction-diffusion simulations. J R Soc Interf. 2012;9(70):859–68. https://doi.org/10.1098/rsif.2011.0574.
22. Klann M, Ganguly A, Koeppl H. Hybrid spatial Gillespie and particle tracking simulation. Bioinformatics. 2012;28(18):549–55. https://doi.org/10.1093/bioinformatics/bts384.
23. Kim Y, Stolarska MA, Othmer HG. A hybrid model for tumor spheroid growth in vitro i: theoretical development and early results. Mathematical Models and Methods in Applied Sciences. 2007;17(supp01):1773–98. https://doi.org/10.1142/S0218202507002479. Accessed 23 July 2019.
24. Schaff JC, Gao F, Li Y, Novak IL, Slepchenko BM. Numerical Approach to Spatial Deterministic-Stochastic Models Arising in Cell Biology. PLOS Comput Biol. 2016;12(12):1005236. https://doi.org/10.1371/journal.pcbi.1005236. Accessed 3 May 2018.
25. Hall D, Minton AP. Macromolecular crowding: qualitative and semiquantitative successes, quantitative challenges. Biochim Biophys Acta (BBA) - Protein Proteomics. 2003;1649(2):127–39. https://doi.org/10.1016/S1570-9639(03)00167-5.
26. Neefjes J, Jongsma MML, Berlin I. Stop or Go? Endosome Positioning in the Establishment of Compartment Architecture, Dynamics, and Function. Trends Cell Biol. 2017;27(8):580–94. https://doi.org/10.1016/j.tcb.2017.03.002.
27. Peng D, Warnke T, Haack F, Uhrmacher AM. Reusing simulation experiment specifications to support developing models by successive extension. Simul Model Pract Theory. 2016;68:33–53. https://doi.org/10.1016/j.simpat.2016.07.006.
28. Köster T, Perumalla K, Uhrmacher A. Efficient Simulation of Nested Hollow Sphere Intersections: For Dynamically Nested Compartmental Models in Cell Biology. In: Proceedings of the 2017 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation SIGSIM-PADS '17, pp. 173–183. New York: ACM; 2017. https://doi.org/10.1145/3064911.3064920.
29. Faeder JR, Blinov ML, Hlavacek WS. Rule-Based Modeling of Biochemical Systems with BioNetGen. In: Maly, IV, (ed.) Systems Biology. Methods in Molecular Biology pp. 113-167. Totowa: Humana Press; 2009. https://doi.org/10.1007/978-1-59745-525-1_5. Accessed 23 July 2019.
30. Maus C, Rybacki S, Uhrmacher AM. Rule-based multi-level modeling of cell biological systems. BMC Syst Biol. 2011;5(1):166. https://doi.org/10.1186/1752-0509-5-166.
31. John M, Lhoussaine C, Niehren J, Uhrmacher AM. The Attributed Pi-Calculus with Priorities. In: Transactions on Computational Systems Biology XII. Lecture Notes in Computer Science, pp. 13?76. Berlin: Springer; 2010. https://link.springer.com/chapter/10.1007/978-3-642-11712-1_2.
32. Pârvu O., Gilbert D., Heiner M., Liu F., Saunders N., Shaw S. Spatial-Temporal Modelling and Analysis of Bacterial Colonies with Phase Variable Genes. ACM Trans Model Comput Simul. 2015;25(2):13–11325. https://doi.org/10.1145/2742546.
33. Wang Z, Butner JD, Kerketta R, Cristini V, Deisboeck TS. Simulating Cancer Growth with Multiscale Agent-Based Modeling. Semin Cancer Biol. 2015;30:70–8. https://doi.org/10.1016/j.semcancer.2014.04.001.
34. Coffey WT, Kalmykov YP, Waldron JT. The Langevin Equation: With Applications to Stochastic Problems in Physics, Chemistry and Electrical Engineering, Revised. Singapore; River Edge, NJ: World Scientific Pub Co Inc; 2004. https://www.worldscientific.com/worldscibooks/10.1142/8195.
35. Hertz H. Über die Berührung fester elastischer Körper. Journal für die reine und angewandte Mathematik (Crelle's Journal). 1882;92:156–71. https://doi.org/10.1515/crll.1882.92.156.
36. Jeschke M, Ewald R. Large-Scale Design Space Exploration of SSA. In: Computational Methods in Systems Biology, Lecture Notes in Computer Science, pp. 211-230. Rostock: Springer; 2008. https://link.springer.com/chapter/10.1007/978-3-540-88562-7_17.
37. Johnson ME, Hummer G. Free-Propagator Reweighting Integrator for Single-Particle Dynamics in Reaction-Diffusion Models of Heterogeneous Protein-Protein Interaction Systems. Phys Rev X. 2014;4(3):031037. https://doi.org/10.1103/PhysRevX.4.031037.
38. von Smoluchowski M. Versuch einer mathematischen Theorie der Koagulationskinetik kolloider Lösungen. Z für Phys Chem. 1917;92:129–68.
39. Collins FC, Kimball GE. Diffusion-controlled reaction rates. J Colloid Sci. 1949;4(4):425–37. https://doi.org/10.1016/0095-8522(49)90023-9.
40. Noyes RM. Effects of diffusion rates on chemical kinetics. Prog React Kinet. 1961;1:129–60. https://ci.nii.ac.jp/naid/10016743949/. Accessed 12 June 2018.
41. Van Deursen A, Klint P, Visser J. Domain-specific languages: An annotated bibliography. SIGPLAN Not. 2000;35(6):26–36. https://doi.org/10.1145/352029.352035.
42. Zunino R, Nikolic D, Priami C, Kahramanogulları O, Schiavinotto T. l: An Imperative DSL to Stochastically Simulate Biological Systems. In: Bodei, C, Ferrari, G, Priami, C, (eds.) Programming Languages with Applications to Biology and Security: Essays Dedicated to Pierpaolo Degano on the Occasion Of His 65th Birthday, pp. 354-357. Cham: Springer; 2015. https://doi.org/10.1007/978-3-319-25527-9_23.
43. Himmelspach J, Uhrmacher AM. Plug'n Simulate. In: Simulation Symposium, 2007. ANSS '07. 40th Annual; 2007. p. 137–43. https://doi.org/10.1109/ANSS.2007.34.
44. Mannella R. Numerical Stochastic Integration for Quasi-Symplectic Flows. SIAM J Sci Comput. 2006. https://doi.org/10.1137/040620965.
45. The CImg Library - C++ Template Image Processing Toolkit. http://cimg.eu/. Accessed 11 Feb 2019.
46. OpenGL - The Industry Standard for High Performance Graphics. https://www.opengl.org/. Accessed 11 Feb 2019.
47. Yogurtcu O. N., Johnson M. E. Theory of bi-molecular association dynamics in 2d for accurate model and experimental parameterization of binding rates. J Chem Phys. 2015;143(8):084117. https://doi.org/10.1063/1.4929390. Accessed 23 July 2019.
48. Heinrich R, Rapoport TA. Generation of nonidentical compartments in vesicular transport systems. J Cell Biol. 2005;168(2):271–80. https://doi.org/10.1083/jcb.200409087.
49. Tyson JJ. Modeling the cell division cycle: cdc2 and cyclin interactions. Proc Natl Acad Sci. 1991;88(16):7328–32. https://doi.org/10.1073/pnas.88.16.7328.
50. Nicolau DV, Burrage K, Parton RG, Hancock JF. Identifying Optimal Lipid Raft Characteristics Required To Promote Nanoscale Protein-Protein Interactions on the Plasma Membrane. Mol Cell Biol. 2006;26(1):313–23. https://doi.org/10.1128/MCB.26.1.313-323.2006.
51. Haack F, Burrage K, Redmer R, Uhrmacher AM. Studying the Role of Lipid Rafts on Protein Receptor Bindings with Cellular Automata. IEEE/ACM Trans Comput Biol Bioinformatics. 2013;10(3):760–70. https://doi.org/10.1109/TCBB.2013.40.
52. Haack F, Lemcke H, Ewald R, Rharass T, Uhrmacher AM. Spatio-temporal Model of Endogenous ROS and Raft-Dependent WNT/Beta-Catenin Signaling Driving Cell Fate Commitment in Human Neural Progenitor Cells. PLOS Comput Biol. 2015;11(3):1004106. https://doi.org/10.1371/journal.pcbi.1004106.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.