View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by D-Scholarship@Pitt

OPEN ACCESS Freely available online

PLOS COMPUTATIONAL BIOLOGY

Exact Hybrid Particle/Population Simulation of Rule-Based Models of Biochemical Systems

Justin S. Hogg[☯], **Leonard A. Harris**[☯], **Lori J. Stover, Niketh S. Nair, James R. Faeder***

Department of Computational and Systems Biology, University of Pittsburgh School of Medicine, Pittsburgh, Pennsylvania, United States of America

Abstract

Detailed modeling and simulation of biochemical systems is complicated by the problem of combinatorial complexity, an explosion in the number of species and reactions due to myriad protein-protein interactions and post-translational modifications. Rule-based modeling overcomes this problem by representing molecules as structured objects and encoding their interactions as pattern-based rules. This greatly simplifies the process of model specification, avoiding the tedious and error prone task of manually enumerating all species and reactions that can potentially exist in a system. From a simulation perspective, rule-based models can be expanded algorithmically into fully-enumerated reaction networks and simulated using a variety of network-based simulation methods, such as ordinary differential equations or Gillespie's algorithm, provided that the network is not exceedingly large. Alternatively, rule-based models can be simulated directly using particle-based kinetic Monte Carlo methods. This "network-free" approach produces exact stochastic trajectories with a computational cost that is independent of network size. However, memory and run time costs increase with the number of particles, limiting the size of system that can be feasibly simulated. Here, we present a hybrid particle/population simulation method that combines the best attributes of both the network-based and network-free approaches. The method takes as input a rule-based model and a user-specified subset of species to treat as population variables rather than as particles. The model is then transformed by a process of "partial network expansion" into a dynamically equivalent form that can be simulated using a population-adapted network-free simulator. The transformation method has been implemented within the open-source rule-based modeling platform BioNetGen, and resulting hybrid models can be simulated using the particle-based simulator NFsim. Performance tests show that significant memory savings can be achieved using the new approach and a monetary cost analysis provides a practical measure of its utility.

Citation: Hogg JS, Harris LA, Stover LJ, Nair NS, Faeder JR (2014) Exact Hybrid Particle/Population Simulation of Rule-Based Models of Biochemical Systems. PLoS Comput Biol 10(4): e1003544. doi:10.1371/journal.pcbi.1003544

Editor: Jorg Stelling, ETH Zurich, Switzerland

Received March 6, 2013; **Accepted** February 3, 2014; **Published** April 3, 2014

Copyright: © 2014 Hogg et al. This is an open-access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Funding: We gratefully acknowledge funding from NIH/NIGMS grant P41GM103712, NIH/NIBIB grant T32EB009403, NSF Expeditions in Computing Grant (award 0926181), and NSF grant CCF-0829788. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

* E-mail: faeder@pitt.edu

☯ These authors contributed equally to this work.

This is a *PLOS Computational Biology* Methods article.

Introduction

Rule-based modeling

Cell signaling encompasses the collection of cellular processes that sample the extracellular environment, process and transmit that information to the interior of the cell, and regulate cellular responses. In a typical scenario, molecules outside of the cell bind to cognate receptors on the cell membrane, resulting in conformational changes or clustering of receptors. A complex series of protein binding and biochemical events then occurs, ultimately leading to the activation or deactivation of proteins that regulate gene expression or other cellular processes [1]. A typical signaling protein possesses multiple interaction sites with activities that can be modified by direct chemical modification or by the effects of modification or interaction at other sites. This complexity at the protein level leads to a combinatorial explosion in the number of possible species and reactions at the level of signaling networks [2].

Combinatorial complexity poses a major barrier to the development of detailed, mechanistic models of biochemical systems. Traditional modeling approaches that require manual enumeration of all potential species and reactions in a network are infeasible or impractical [2–4]. This has motivated the development of rule-based modeling languages, such as the BioNetGen language (BNGL) [5,6], Kappa [7,8], and others [9–12], that provide a rich yet concise description of signaling proteins and their interactions [13]. The combinatorial explosion problem is avoided by representing interacting molecules as structured objects and using pattern-based rules to encode their interactions. In the graph-based formalisms of BNGL and Kappa, molecules are represented as graphs and biochemical interactions by graph-rewriting rules. Rules are *local* in the sense that only the properties of the reactants that are transformed, or are required for the transformation to take place, affect their ability to react. As such, each rule defines a class of reactions that share a common set of transformations (e.g., the formation of a bond between molecules) and requirements for those transformations to take place (e.g., that one or more components have a particular covalent modification). The number of reactions encoded by a rule varies depending on

Author Summary

Rule-based modeling is a modeling paradigm that addresses the problem of combinatorial complexity in biochemical systems. The key idea is to specify only those components of a biological macromolecule that are directly involved in a biochemical transformation. Until recently, this "pattern-based" approach greatly simplified the process of model *building* but did nothing to improve the performance of model *simulation*. This changed with the introduction of "network-free" simulation methods, which operate directly on the compressed rule set of a rule-based model rather than on a fully-enumerated set of reactions and species. However, these methods represent every molecule in a system as a particle, limiting their use to systems containing less than a few million molecules. Here, we describe an extension to the network-free approach that treats rare, complex species as particles and plentiful, simple species as population variables, while retaining the exact dynamics of the model system. By making more efficient use of computational resources for species that do not require the level of detail of a particle representation, this hybrid particle/population approach can simulate systems much larger than is possible using network-free methods and is an important step towards realizing the practical simulation of detailed, mechanistic models of whole cells.

the specifics of the model; a rule-based encoding is considered compact if it contains rules that encode large numbers of reactions. Overviews of rule-based modeling with BNGL can be found in Sec. S3.1 of Text S1 and Refs. [6,14]. A description of the graph-theoretic formalism underlying BNGL is provided in Sec. S4.1 of Text S1, building on a previous graph-theoretical treatment [15].

Network-based and network-free simulation of rule-based models

An important characteristic of rule-based models is that they can encode both finite and *infinite* reaction networks. If the network is finite and "not too large" ($\lesssim$10000 reactions [16]) it can be generated from the rule-based model algorithmically by a process known as "network generation" [5,6,14,15,17]. Network generation begins by applying the rules of a rule-based model to a set of initial "seed" species, which define the initial state of the model system, to generate new species and reactions. The new species are then matched against the existing species to determine whether or not they are already present in the network [18]. Any species that are not already present are added to the network and an additional round of rule application is performed. This iterative process continues until an iteration is encountered in which no new species are generated. The resulting system of reactions can then be simulated using a variety of network-based deterministic and stochastic simulation methods. For example, network-based simulation methods currently implemented within BioNetGen include SUNDIALS CVODE [19] for ordinary differential equation (ODE)-based simulations, Gillespie's stochastic simulation algorithm (SSA; direct method with dynamic propensity sorting) [20,21], and the accelerated-stochastic "partitioned-leaping algorithm" [22].

The rule-based methodology also provides a way to simulate models with prohibitively large or infinite numbers of species and reactions. This "network-free" approach involves representing molecular complexes as particles and applying rule transformations to those particles at runtime using a kinetic Monte Carlo update scheme [23,24]. At each simulation step, reactant patterns are matched to the molecular complexes within the system to calculate rule propensities. The rule to next fire is then selected probabilistically as in the SSA [20] and the particle(s) to participate in the transformation is (are) selected randomly from the set of matches. When the rule fires, transformations are applied to the reactant complexes to create the products. Since the reactants and products are determined at runtime there is no need to enumerate all species and reactions *a priori* as in network-based methods. This procedure is a particle-based variant of Gillespie's algorithm [23,24] and a generalization of the "n-fold way" of Bortz et al. [25], which was originally developed to accelerate the simulation of Ising spin systems. An efficient, open-source implementation that is compatible with BNGL models is NFsim, the "network-free simulator" [16]. Other network-free simulation tools for rule-based models include RuleMonkey [26], DYNSTOC [27], SRsim [28], and KaSim [24]. A recent paper [29] compares the rejection-based sampling technique [23] used in NFsim with the rejection-free approach employed in RuleMonkey. For models of multivalent ligand-receptor binding, rejection-based sampling was shown to be more efficient in the vicinity of the solution-gel phase boundary, while rejection-free sampling was more efficient for simulating the dynamics within the gel phase.

Since only the current set of molecular complexes and the transformations that can be applied to them are tracked, network-free methods can efficiently simulate systems that are intractable to network-based methods [16,23,24,29]. However, the explicit representation of every molecule in the system is a major shortcoming of the approach. As such, network-free methods can require large amounts of computational memory for systems that contain large numbers of particles, a potential barrier to simulating systems such as the regulatory networks of a whole cell [30,31]. A typical eukaryotic cell, for example, contains on the order of 10^3-10^4 protein-coding genes, 10^4-10^5 mRNA molecules, and 10^9-10^{10} protein molecules [32,33], along with much larger numbers of metabolites, lipids, and other small molecules. Simulating a cell at this level of detail using a network-free approach would be impractical. There is a need, therefore, for new approaches that can reduce the memory requirements of network-free simulation methods.

Computational complexity

A common measure of the computational cost of an algorithm is its *computational complexity*. In basic terms, computational complexity measures how the computational cost increases as an algorithm is applied to increasingly larger data sets [34]. For the simulation methods considered in this paper, two types of computational complexity are important: (i) *space complexity*, the number of memory units consumed during the execution of an algorithm; (ii) *time complexity*, the number of computational steps required to complete an algorithm.

Network-based exact-stochastic simulation methods, like Gillespie's SSA [20,35,36], treat species as lumped variables with a population counter. Therefore, their space complexity is constant in the number of particles in the system. However, representing the reaction network has a space complexity that is linear (or worse if a reaction dependency graph is used [37,38]) in the number of reactions. Network-based SSA methods are thus space efficient for systems with large numbers of particles, but less so for systems with large numbers of reactions. The time complexity of SSA methods is more difficult to quantify. It depends on model-specific factors such as the number of reactions in the network and the values of rate constants and species concentrations, as well as methodological factors such as how the next reaction to fire in the system is selected [20,21,37–41] and how reaction propensities are updated

after each reaction firing [37,38]. However, for our purposes, what matters is that the time cost *per event* (reaction firing) for these methods is constant in the number of particles in the system and increases with the number of reactions in the network.

Network-free methods, in contrast, represent each particle individually. Thus, their space complexity is *linear* in the number of particles. This is the primary shortcoming of these methods, as it limits the size of system that can be feasibly simulated. However, since reactions are not enumerated, their space complexity is linear in the number of *rules*, rather than the number of reactions. This is a key advantage for models where very large reaction networks are encoded by a small number of rules. Network-free methods also have an advantage over network-based methods in that their time complexity per event also scales with the number of rules, rather than the number of reactions. Since the number of rules in a rule-based model is typically far less than the number of reactions, this can be a substantial improvement. For example, NFsim has been demonstrated to significantly outperform network-based SSA methods for a family of Fcε receptor signaling models with large reaction networks [16]. We also note that for many models network-free methods have a time cost per event that is constant in the number of particles. However, for systems in which large aggregates form (e.g., models with polymerization dynamics [42,43]) the cost can be significantly higher, scaling with the number of particles [16,24]. Nevertheless, network-free methods are still usually the best option in these cases because these types of models tend to encode very large reaction networks [16].

In Table 1, we summarize the space and time complexities for different network-based SSA variants and for the network-free algorithm. Of most relevance to the current work are the entries that show: (i) the space complexity of network-based methods is constant in the number of particles and linear (or worse) in the reaction network size; (ii) the space complexity of network-free methods is linear in the number of particles and independent of the reaction network size, depending instead on the number of rules; (iii) the time complexity of network-based methods depends on the number of reactions in the network while for network-free methods it depends on the number of rules. Network-based methods are thus the best choice for systems with large numbers of particles and a small to moderate reaction network, and network-free methods are the best choice for systems with a large reaction network and small to moderate numbers of particles. However, neither method is optimal for systems that contain *both* a large number of particles and a large reaction network.

Combining network-based and network-free methodologies

The key idea pursued in this work is that memory consumption can be reduced in network-free simulators if simple species and small molecular complexes that exist in the system in large numbers are treated as population variables with counters rather than as particles. However, retaining the ability to address combinatorial complexity requires retaining the particle representation for species and complexes that are comprised of many molecules and/or have a large number of internal states. Here, we present an approach, termed the hybrid particle/population (HPP) simulation method, that accomplishes this. Given a user-defined set of species to treat as population variables, the HPP method partially expands the network around these population species and then simulates the partially-expanded model using a population-adapted particle-based method. By treating complex species as structured particles, HPP capitalizes on the reduced time complexity with respect to network size characteristic of the network-free approach. However, for the subset of species treated as population variables, we take advantage of the constant memory requirements of the network-based methodology. It is important to emphasize that in the HPP approach it is the *system* that is represented in a hybrid manner, as a collection of particles and population variables. The underlying simulator remains the same particle-based variant of Gillespie's algorithm that is used in existing network-free simulators [23,24], but with small modifications to support population variables. This distinguishes HPP from other types of hybrid methods that combine different simulation methodologies, e.g., ODE/SSA integrators [44–53].

Related work

While numerous rule-based modeling frameworks have been developed, little has been done with regard to hybrid particle/

Table 1. Space and time complexities for network-based (SSA) and network-free (NF) stochastic simulation algorithms.

	SSA		**NF**	
	***Particles* (*P*)**	***Reactions* (*R*)**	***Particles* (*P*)**	***Rules* ($\bar{R}$)**
Space	$O(1)$	$O(R)^a$, $O(d(R)\cdot R)^b$	$O(P)$	$O(\bar{R})^a$
Time (per event)	$O(1)$	$O(d(R))^c$, $O(d(R)\log_2 R)^d$, $O(R)^e$	$O(1)$, $O(P)^f$	$O(\bar{R})^g$

[a]No dependency graph.
[b]Dependency graph [37,38].
[c]Logarithmic classes (with dependency graph) [21,39,40].
[d]Next-reaction method (with dependency graph) [37].
[e]Direct method (with or without dependency graph) [20].
[f]Polymerizing systems in gel phase [23,42] (see Fig. 5B).
[g]Direct method-like implementation.
Scalings are shown with respect to particle number, P, and number of reactions, R, or rules, $\bar{R}$. For combinatorially-complex models, $\bar{R} \ll R$. Note that time complexity is given on a "per event" (reaction/rule firing) basis. If a reaction dependency graph [37] is used, the space and time complexities of SSA methods with respect to R depend on d, the maximum number of reactions updated after each reaction firing [37,38]. In combinatorially-complex models, d often increases with R (see Figure S1 of the supporting information). The time complexity of SSA methods with respect to R also depends on the method used for selecting the next reaction to fire in the system. Scalings are shown for three different SSA variants that use different selection methods [20,21,37,39,40]. Also note that optimized variants of the direct method [21,38,41] have been shown to outperform methods with lower asymptotic complexity in some cases [38]. Space and time complexities of the NF algorithm with respect to assume no dependency graph and that the next rule to fire is selected as in Gillespie's direct method [20], although in principle other variants are possible.
doi:10.1371/journal.pcbi.1003544.t001

Table 2. Summary of example models used to test the performance of the HPP method.

Model	Rules	Reactions	Species	Particles ($f=1$)	Population species	Rules after PNE	t_end (s)
TLBR [16,42,65]	4	∞	∞	5.3×10^6	2	9	500
Actin [16,43]	21	∞	∞	1.2×10^6	2	25	1000
FcεRI [16,70,81]	24	58 276	3744	6.9×10^6	1/6	25/38	2400
EGFR [18,71,72]	113	415 858	18 950	2.2×10^6	29	159	1200

Number of particles is for an NFsim simulation of a full cell volume ($f=1$). Fractional cell volumes as low as 0.001 and as high as 1 are used in the performance analyses (see "Example models" for details). Number of rules after PNE includes the population-mapping rules (one per population species).
doi:10.1371/journal.pcbi.1003544.t002

population simulation. Kappa [7,8] has the concept of "tokens," which are structureless population-type species. Modelers can write hybrid models in terms of both structured "agents" and structureless tokens and simulate them using KaSim 3, the most recent version of the Kappa-compatible network-free simulator (https://github.com/jkrivine/KaSim). However, there is no facility for transforming a model written exclusively in terms of agents into a hybrid form, as in our HPP method. Bittig et al. [10] have developed a spatial rule-based language called ML-Space that builds upon the multi-level language ML-Rules [9]. "Entities" that are assigned optional attributes such as shape, volume, and position in continuous space are automatically treated as particles diffusing via Brownian motion, while those without these attributes are treated as population variables reacting and diffusing within a discretized space (subvolumes). For non-spatial models, the population-based network-free algorithm (PNFA) of Liu et al. [54] employs a similar philosophy: all multi-state (structured) species are automatically treated as particles, while single-state species are treated as population variables. Both ML-Space and PNFA lack a general representation of intermolecular bonding, which makes it difficult to account for combinatorial complexity associated with aggregation processes [2,29]. Falkenberg et al. [55] have proposed a hybrid deterministic/stochastic method that specifically addresses the problem of aggregation. Their approach first calculates occupancy probabilities as a function of time for all binding-site types by treating them as population variables and numerically integrating an associated set of deterministic ODEs describing the binding/unbinding kinetics. An ensemble of system states is then obtained by randomly distributing bonds, based on these probabilities, among a finite number of discrete molecules. The method assumes that inter- and intra-molecular bond formations occur with equal rates. Thus, although efficient for problems with high symmetry, its applicability to more general cases may be limited.

Other approaches aimed at improving the efficiency of rule-based simulations include "on-the-fly" network generation [17,56,57], where the reaction network is gradually built up by adding reactions only when new species appear in the system. The approach has only been developed within the context of discrete-stochastic simulation and has been shown to be significantly less efficient than network-free approaches when applied to combinatorially-complex models [23,58]. An alternative approach to reducing computational cost is exact model reduction (EMR) [59–64]. EMR aims to reduce the state space of a rule-based model while preserving the exact system dynamics with respect to observable quantities. These methods can achieve dramatic reductions in model complexity when applied within the context of ODEs, so long as the model does not contain significant cooperative or allosteric interactions [62,64]. EMR for stochastic simulations, however, has so far been less successful (see http://infoscience.epfl.ch/record/142570/files/stochastic_fragments.pdf).

Methods

Example models

We have tested the performance of the HPP method by applying it to four example models, summarized in Table 2 and discussed in further detail below. All of the models are biologically relevant and are either taken directly from the literature or are based on models taken from the literature. Complete BNGL encodings, HPP configuration files (containing actions for loading models, defining population maps, and executing simulations), and partially-expanded versions of all example models are provided as Texts S5, S6, S7, S8, S9, S10, S11, S12, S13, S14, S15, S16, S17 of the supporting information.

Trivalent-ligand bivalent-receptor. The trivalent-ligand bivalent-receptor (TLBR) model is a simplified representation of receptor aggregation following multivalent ligand binding. TLBR has biological relevance to antigen-antibody interaction at the cell surface, where bivalent $\mathrm{IgE-Fc\varepsilon RI}$ receptor complexes aggregate in the presence of multivalent antigen [65]. A theoretical study of the TLBR system was presented by Goldstein and Perelson [65], who derived analytical conditions for a solution-gel phase transition in terms of binding equilibrium constants, free ligand concentration, and receptors per cell. A more recent study considered the effects of steric constraints and ring closure on the solution-gel phase transition [42].

Despite its simplicity, the TLBR system experiences a state-space explosion near the solution-gel phase boundary. A computational study by Sneddon et al. using NFsim [16] reproduced the analytical results of Goldstein and Perelson. Due to large excesses of ligand and receptor under certain conditions, TLBR is a natural test case for HPP. We simulated the TLBR system using HPP with free ligand and receptor treated as population species. All simulations were performed with parameters as defined in Monine et al. [42], which lie within the solution-gel phase coexistence region. A cell-scale simulation assumed $1\,\mathrm{nl}$ extracellular volume per cell (10^6 cells/ml) with $8.3\,\mathrm{nM}$ ligand and 3×10^5 receptors per cell. Simulations were performed at fractional cell volumes, f, ranging from 0.001 to 0.1 with a lumping rate constant $\mathtt{k_lump} = 10000/\mathrm{s}$ (see below).

Actin polymerization. Actin polymerization plays a key role in cell morphology and motility [66,67]. Roland et al. [43] presented a dynamic model of actin polymerization featuring

filament elongation by monomer addition, stabilization by ATP hydrolysis, and severing mediated by actin depolymerizing factor (ADF)/cofilin. Sneddon et al. [16] presented a rule-based formulation of the Roland et al. model and replicated their results using NFsim. The model features an excess of actin monomer and ADF molecules. Therefore, we speculated that substantial memory reduction would be possible using the hybrid approach. We applied HPP to the Sneddon et al. rule-based model of actin dynamics (hereafter referred to as the Actin model) with actin monomer and ADF treated as population species. A cell-scale simulation assumed 1 pl intracellular volume with 1 μM actin monomer and 1 μM ADF/cofilin. Simulations were performed at fractional cell volumes, f, ranging from 0.01 to 1 with a lumping rate constant `k_lumps` = 10000/s

FcεRI **signaling.** FcεRI is a membrane receptor that binds IgE antibodies. Signaling through FcεRI regulates basophilic histamine release in response to IgE antibody-antigen interaction [68]. Faeder et al. [69,70] developed a rule-based model of FcεRI receptor assembly and activation in which receptor dimerization/clustering is mediated by chemically cross-linked IgE, which serve as multivalent ligands. Dimerized receptors are transphosphorylated, leading to Syk and Lyn recruitment and phosphorylation. Sneddon et al. [16] presented several extensions of the Faeder et al. model, including the *gamma2* variant with two γ phosphorylation sites. Particle-based NFsim simulations of the *gamma2* model were found to be substantially faster than network-based SSA simulations.

Due to the excess of free ligand, the HPP method was applied to the *gamma2* model to reduce memory consumption. The method was applied with two different sets of population species. In the first case, only free ligand was treated as a population species (FcεRI : 1). In the second, cytosolic Lyn and all four phosphorylation states of cytosolic Syk were also treated as populations (FcεRI : 6). A cell-scale simulation assumed 1 pl intracellular volume with 1 nl extracellular space per cell (10^6 cells/ml), 10 nM ligand, and 4×10^5 receptors per cell. Simulations were performed at fractional cell volumes, f, ranging from 0.001 to 0.1 with a lumping rate constant `k_lumps` = 10000/s.

EGFR signaling. A model of signaling through the epidermal growth factor receptor (EGFR), beginning with ligand binding and concluding with nuclear phospho-ERK activity, was constructed by combining three existing models: (i) a rule-based model of EGFR complex assembly [18]; (ii) a Ras activation model [71]; (iii) a pathway model of Raf, MEK and ERK activation [72]. Ras activation was coupled to the EGFR complex assembly by treating receptor-recruited Sos as the Ras GEF. Activated Ras was coupled to the Raf/MEK/ERK cascade through RasGTP-Raf binding and subsequent phosphorylation of Raf. Parameters for the combined model were obtained from the respective models. However, parameters governing Ras-GEF (i.e., Sos) activity had to be changed from their original values [71] in order to account for the known GEF-mediated activation of Ras [73]. Specifically, we used $K_{M,\mathrm{GDP}} = K_{M,\mathrm{GTP}} = 1.56 \times 10^{-7}$ M and $D = 1000$ (unitless).

Free EGF and Raf-, MEK-, and ERK-based species were treated as population species in the hybrid variant. Ras-based species were also treated as populations except for those that include a Sos molecule. A cell-scale simulation assumed 0.94 pl cytosolic and 0.22 pl nuclear volume, with 0.94 pl extracellular space, 10 nM ligand, and 4×10^5 receptors per cell. Simulations were performed at fractional cell volumes, f, ranging from 0.01 to 1 with a lumping rate constant `k_lumps` = 100000/s.

Performance metrics

HPP was evaluated for peak memory use, CPU run time, and accuracy as compared to particle-based NFsim simulations. For models where network generation is possible (FcεRI and EGFR), comparisons were also made to SSA simulations (as implemented within BioNetGen [6]). All simulations were run on a 2× Intel Xeon E5520 @ 2.27 GHz (8 cores, 16 threads, x86_64 instruction set) with 74 GB of RAM running the GNU/Linux operating system. To ensure that each process had access to 100% of the compute cycles of a thread, no more than 12 simulations were run simultaneously.

Peak memory. Average peak memory usage for each simulation method was calculated based on seven independent simulation runs. Peak memory for each run was evaluated by peak virtual memory allocation reported by the operating system with the command "cat/proc/<PID>/status". For all tested models, peak memory was achieved early in the simulation and remained steady throughout (data not shown).

CPU run time. Average CPU run time for each simulation method was calculated based on seven independent simulation runs using clock time as a metric. Clock time for each run was recorded using the Time::HiRes Perl module. Run time included initialization as well as the simulation phase. Partial network expansion for HPP simulations was a one time cost, typically a few seconds, and was not included in the calculation.

Accuracy. Simulation accuracy was quantified using several approaches. First, since HPP, NFsim, and SSA are all exact-stochastic methods, they should all produce statistically the same number of reaction firings. To verify this, for all tested models the total number of reaction firings was recorded for each of 40 independent simulation runs of each method (firings of population-mapping rules were subtracted from the total in HPP simulations). The Mann-Whitney U test [74,75] was then used to test the null hypothesis that none of the methods produces a larger number of reaction firings.

For the TLBR and Actin models, we further compared equilibrium distributions for key observables. These include the number of receptor clusters in the TLBR model and the length of actin polymers in the Actin model. 10 000 samples were collected over 100 000 seconds of simulated time and distributions were compared by binning samples (20 bins) and performing a two-sample chi-squared test [76]. For the FcεRI and EGFR models, we compared dynamic trajectories for key observables. These include $\gamma-$phosphorylated receptor and receptor-recruited, $\alpha-$ phosphorylated Syk in the FcεRI model, and activated Sos and nuclear phosphorylated ERK in the EGFR model. Due to complications of autocorrelation, a statistical test was not applied to the dynamic trajectory comparison. Instead, moving averages and 5–95% frequency envelopes, based on 40 simulation runs of each method using a sampling window of 10 s, were plotted for inspection by eye.

Software

All HPP and NFsim simulations reported in this work were run using NFsim version 1.11, which is available for download at http://emonet.biology.yale.edu/nfsim. All simulations (SSA included) were invoked through BioNetGen version 2.2.4, which implements the hybrid model generator and is distributed with NFsim 1.11. Instructions for running simulations with BioNetGen (ODE, SSA, and HPP) can be found in Secs. S3.2 and S3.3 of Text S1 and Refs. [6,14]. NFsim and BioNetGen source code are available at http://code.google.

com/p/nfsim and http://code.google.com/p/bionetgen, respectively. Additional documentation for BioNetGen can be found at http://bionetgen.org.

Results

A hybrid particle/population simulation approach

In this section, we first present an approach, termed "partial network expansion," for transforming a rule-based model into a dynamically-equivalent, partially-expanded form. We then describe a simple modification to the network-free simulation protocol that permits simulation of the transformed model as a collection of both particles and population variables. We refer to the combination of these methods as the hybrid particle/population (HPP) simulation method. The basic workflow is shown in Fig. 1.

The HPP approach is analogous to the coupled procedure of network generation and simulation described above, where a rule-based model is first transformed into a *fully-expanded* reaction network and then simulated as a collection of population variables (i.e., species) using a network-based simulator. The obvious differences are that in HPP the network is only partially expanded and the system can only be simulated stochastically using a population-adapted network-free simulator. The partial network expansion algorithm has been implemented within the open-source rule-based modeling package BioNetGen [5,6,14] and resulting hybrid models can be simulated using version 1.11 (or later) of the network-free simulator NFsim [16], which has been modified to handle population-type species. For convenience, we adhere in this paper to the BNGL syntax, which is summarized in Sec. S3.1 of Text S1 of the supporting material. However, the HPP method is generally applicable to any rule-based modeling language for which there exists a network-free simulator capable of handling a mixed particle/population system representation, e.g., KaSim 3.x for Kappa language models (see https://github.com/jkrivine/KaSim).

Population species and population-mapping rules. Given a rule-based model, the first step in the HPP approach is to select a subset of species to treat as "lumped" population variables. There are no hard-and-fast rules for doing this but, generally speaking, species that are good candidates for a population treatment (i) have a small number of components and internal states, (ii) participate in a small number of rules, and (iii) maintain a large population throughout the course of a simulation. An example is a simple ligand species that exists in great excess in the extracellular environment and interacts with cell surface receptors. It is our experience that these simple rules of thumb, combined with the experience and intuition of the modeler, are usually sufficient for selecting an adequate set of population species. However, in some cases a more systematic approach may be desirable. We will return to this topic below.

For now, however, let us assume that we have selected a suitable set of population species. The next step in the HPP approach is to map each of these to an associated *unstructured* species. The mapping is accomplished by defining a *population-mapping rule*, which follows the same syntactic conventions as a standard BNGL rule. For example, the rule

```
Egf(r)→pop_Egf()k_lump
```

maps the unbound EGF ligand, Egf(r), to the unstructured species pop_Egf(). To avoid confusion, we will henceforth refer to species on the reactant side of a population-mapping rule, such as Egf(r), as *structured population species* and to those on the product side as *unstructured population species*. Importantly, unstructured population species differ from conventional unstructured molecules in BNGL in that they possess a property, called a *count*, which records their current population (see Sec. S3.3 of Text S1 and Texts S4, S7, S10, S13, S14, and S17 to see how the population keyword is used to make this distinction). The action of the population-mapping rule above is thus to delete the Egf(r) molecule and to *increment by one* the count of pop_Egf(). The role of the rate parameter `k_lump`, termed the *lumping rate constant*, will be explained in detail below.

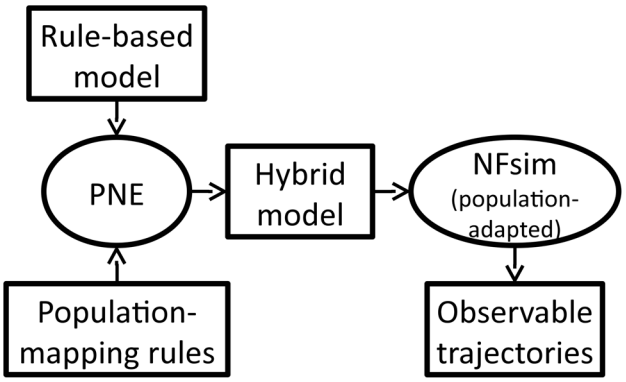

Figure 1. Basic workflow of the HPP simulation method. Given a rule-based model and a user-specified set of population-mapping rules (which define the population species), partial network expansion (PNE) is performed to generate a hybrid version of the original model. The hybrid model is then passed to a population-adapted network-free simulator (e.g., NFsim 1.11), which generates the time-evolution trajectories for all observable quantities specified in the original model.
doi:10.1371/journal.pcbi.1003544.g001

Partial network expansion. Ultimately, our goal in the HPP method is to replace in the simulation environment large numbers of indistinguishable particles with small numbers of lumped objects containing population counters (the unstructured population species), thus significantly reducing memory usage. In order to accomplish this without losing any information regarding the dynamics of the system, we must partially expand the rule set of the original model until all interactions and transformations in which the structured population species participate *as reactants* (see below) are enumerated. We can then swap the structured species with their unstructured counterparts, which have been specified via the population-mapping rules. We refer to this procedure as partial network expansion (PNE).

The PNE algorithm is comprised of three basic steps, which are applied to each rule of a rule-based model:

1. For each reactant pattern in the rule, identify all matches of that pattern into the set of structured population species. Also collect a self-match of the reactant pattern *unless it equals* one of the population species (this can only happen if the reactant pattern is a fully-specified species; see below for further discussion).
2. Derive an expanded set of rules by applying the rule to all possible combinations (the cartesian product) of the pattern matches collected in Step 1.
3. For each derived rule from Step 2, replace each instance of a structured population species with its unstructured population counterpart.

The result is an expanded rule set consisting of three general types of rules: (i) particle rules, in which all reactants are

conventional reactant patterns; (ii) mixed particle/population rules, where at least one reactant is a conventional reactant pattern and one is an unstructured population species; (iii) pure population *reactions*, where all reactants are unstructured population species. This expanded rule set has the property that every possible action of the original rule set on the population species is enumerated while actions on particle objects remain pattern-based (i.e., non-enumerated). For a more formal presentation of the PNE algorithm, complete with pseudocode, we direct the reader to Sec. S4.2 of Text S1.

Role of the population-mapping rules. After completion of PNE, the final step in transforming a rule-based model into a form that can be simulated as a hybrid particle/population system is to append the population-mapping rules to the expanded rule set. The reason for doing this is not immediately obvious. We have seen above that the population-mapping rules specify which structured species are to be replaced in the transformed model with population variables. However, an obvious question to ask is why we have chosen to specify this information via a set of reaction rules, rather than simply as a list of species to be lumped. The answer is combinatorial complexity.

As explained above, systems that are combinatorially complex are comprised of a relatively small number of constituent parts but exhibit an explosion in the number of potential species and reactions due to the myriad number of ways in which these parts can be connected and arranged. Rule-based modeling is effective in representing these systems because it focuses only on the portions of molecular complexes that affect biochemical reactivity, not on entire species. However, a consequence of this approach is that there is often ambiguity regarding the products of a reaction rule. A rule may describe the breaking of a bond between two molecules, for example, but the exact composition of the resulting complexes is left necessarily ambiguous (see Fig. 2).

With regard to the HPP approach, this ambiguity in the products of a reaction rule complicates the process of PNE. Application of a reaction rule to one complex may produce a population species, whereas application of the same rule to a different complex may not. Distinguishing between cases where population species are produced and where they are not is difficult, and may even be impossible if the system is combinatorially complex. Thus, the strategy that we have adopted here is to expand the network out only to the point where all population species *on the reactant side* are enumerated and to handle the ambiguity in products by adding the population-mapping rules to the rule set. The role of the population-mapping rules is thus to detect any instances of structured population species that appear in the simulation environment as products of a rule application and to gather them up into the unstructured population pool.

This returns us to the issue of the lumping rate constant, k_lump. In Step 1 of the PNE algorithm, if a reactant pattern equals a population species then we discard the self-match (the structured version of the population species). To see why we do this, consider the binding rule depicted in Fig. 2A. However, different from Figs. 2B–D, assume that molecules A and B have only *one* binding site each. If we choose to lump the unbound molecules then we must define the following population-mapping rules:

$$\mathrm{A(b)}\rightarrow\mathrm{pop_A()k_lump},$$

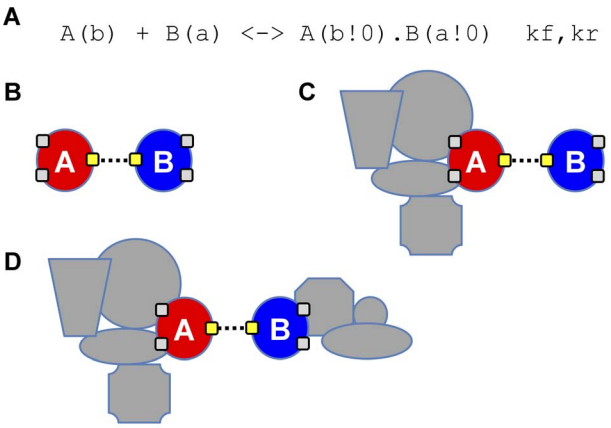

Figure 2. Simple illustration of ambiguity in the products of reaction rules. (A) A simple rule encodes the reversible binding of two molecule types, A and B. (B)–(D) If both molecules have multiple binding sites then they may be present within arbitrarily complex complexes. Breaking the bond between A and B thus produces a variety of product species, some of which may correspond to population species and others not. Dashed line represents a bond addition/deletion operation.
doi:10.1371/journal.pcbi.1003544.g002

$$\mathrm{B(a)}\rightarrow\mathrm{pop_B()k_lump}.$$

Obviously, these structured population species are equivalent to the reactant patterns in Fig. 2A. However, let us choose *not* to discard the self-matches in this case. PNE would then generate the following four derived rules:

$$\mathrm{A(b)+B(a)}\rightarrow\mathrm{A(b!0).B(a!0)kf},$$

$$\mathrm{pop_A()+B(a)}\rightarrow\mathrm{A(b!0).B(a!0)kf},$$

$$\mathrm{A(b)+pop_B()}\rightarrow\mathrm{A(b!0).B(a!0)kf},$$

$$\mathrm{pop_A()+pop_B()}\rightarrow\mathrm{A(b!0).B(a!0)kf}.$$

We see that the first three of these rules have conventional (structured) reactant patterns. However, if k_lump is sufficiently large then particle instances of A(b) and B(a) will never exist in the system long enough to be matched to these patterns. Thus, these rules can be safely discarded, which is equivalent to discarding the self-match in Step 1 of the PNE algorithm. Retaining only the fourth derived rule (the pure population version) simplifies the process and keeps the size of the derived rule set to a minimum.

The consequence of this is obviously that the HPP method is formally exact *only* for an infinite lumping rate constant. From a practical point of view, this could be a problem if the network-free simulator being used does not support infinite rates (e.g., NFsim currently does not). However, our performance tests indicate that as long as k_lump is "large" with respect to the model dynamics then essentially exact results can be obtained (see 'Performance

	Molecule Type	*Description*
1	`L(r)`	extracellular ligand
2	`R(l,a~0~P,b~0~P)`	membrane receptor with two phosphorylation sites
3	`A(r,b~0~P)`	cytosolic molecule (recruits to phosphorylated receptor)
4	`B(r,c)`	cytosolic molecule (recruits to phospho-receptor or phospho-A)
5	`C(b)`	cytosolic molecule (binds to B)

	Reaction Rule	*Rate constant(s)*	*Description*
1	`L(r) + R(l) <-> L(r!1).R(l!1)`	`kp1,km1`	receptor-ligand binding
2	`L(r!1).R(l!1,a~0) -> L(r!1).R(l!1,a~P)`	`k2`	site `a` phosphorylation
3	`L(r!1).R(l!1,b~0) -> L(r!1).R(l!1,b~P)`	`k2`	site `b` phosphorylation
4	`R(a~P) -> R(a~0)`	`k3`	site `a` dephosphorylation
5	`R(b~P) -> R(b~0)`	`k3`	site `b` dephosphorylation
6	`R(a~P) + A(r) <-> R(a~P!1).A(r!1)`	`kp4,km4`	phosphorylated `R` binding `A`
7	`R(b~P) + B(r) <-> R(b~P!1).B(r!1)`	`kp5,km5`	phosphorylated `R` binding `B`
8	`B(c) + C(b) <-> B(c!1).C(b!1)`	`kp6,km6`	`B` and `C` binding
9	`R(a~P!1).A(r!1,b~0) -> R(a~P!1).A(r!1,b~P)`	`k7`	recruited `A` phosphorylation
10	`A(b~P) -> A(b~0)`	`k8`	`A` dephosphorylation
11	`A(b~P) + B(r) <-> A(b~P!1).B(r!1)`	`kp9,km9`	phosphorylated `A` binding `B`

Population-mapping rules:

	Structured species		*Population species*	*Lumping rate constant*
1	`L(r)`	->	`P1()`	`k_lump`
2	`A(r,b~0)`	->	`P2()`	`k_lump`
3	`A(r,b~P)`	->	`P3()`	`k_lump`
4	`A(r,b~P!1).B(r!1,c)`	->	`P4()`	`k_lump`
5	`A(r,b~P!1).B(r!1,c!2).C(b!2)`	->	`P5()`	`k_lump`
6	`B(r,c)`	->	`P6()`	`k_lump`
7	`B(r,c!1).C(b!1)`	->	`P7()`	`k_lump`
8	`C(b)`	->	`P8()`	`k_lump`

Figure 3. Simple receptor activation model in BNGL format. Abridged; see Text S2 of the supporting material for the complete model and Text S3 for the population-mapping rules.
doi:10.1371/journal.pcbi.1003544.g003

analyses'). Nevertheless, we have implemented in BioNetGen a "safe" mode for PNE that retains all of the self-matches and, hence, produces exact results for *any* value of k_lump (see Sec. S3.3 of Text S1 for instructions on how to call this method). For a select number of examples, we have confirmed that both approaches give essentially identical results for sufficiently large k_lump and that the "safe" mode is less efficient (data not shown).

Simple example of PNE. PNE is best illustrated through an example. In Fig. 3, we present a simple rule-based model of receptor activation (for brevity, parameters, initial populations, and output observables are omitted; see Text S2 of the supporting material for the complete model in BNGL format). The model includes a ligand, L, its cognate receptor, R, and three cytosolic proteins, A, B, and C, that are recruited to the phosphorylated receptor. The 16 rules (six unidirectional and five reversible), describing ligand-receptor binding, receptor phosphorylation/dephosphorylation, and protein recruitment, encode a reaction network comprised of 56 species and 287 reactions. In applying the HPP method, eight species are selected for lumping: free ligand, free A, B and C, and complexes of A, B and C that exclude the receptor. Receptor complexes are treated as particles because there are many possible receptor configurations (48 total).

In Fig. 4, a step-by-step application of PNE to rule 11f (forward) of Fig. 3 is presented. First, both reactant patterns are matched to the structured population species. Reactant pattern 1 has one match, while reactant pattern 2 has two. Note that since neither reactant pattern exactly equals a species (i.e., is isomorphic to one)

Reaction rule: `A(b~P) + B(r) -> A(b~P!1).B(r!1) kp9`

Description: "Bind component sites b~P and r"

1: Find Reactant Pattern (RP) Matches

	Matches to RP 1	*Population species*		*Matches to RP 2*	*Population species*
1	`A(b~P)`	identity automorphism	1	`B(r)`	identity automorphism
2	`A(r,b~P)`	`P3()`	2	`B(r,c)`	`P6()`
			3	`B(r,c!1).C(b!1)`	`P7()`

2: Rule Expansion: Apply rule to each reactant set in the cartesian product of reactant matches

1	`A(b~P) + B(r)`	->	`A(b~P!1).B(r!1)`	kp9
2	`A(b~P) + B(r,c)`	->	`A(b~P!1).B(r!1,c)`	kp9
3	`A(b~P) + B(r,c!1).C(b!1)`	->	`A(b~P!2).B(r!2,c!1).C(b!1)`	kp9
4	`A(r,b~P) + B(r)`	->	`A(r,b~P!1).B(r!1)`	kp9
5	`A(r,b~P) + B(r,c)`	->	`A(r,b~P!1).B(r!1,c)`	kp9
6	`A(r,b~P) + B(r,c!1).C(b!1)`	->	`A(r,b~P!2).B(r!2,c!1).C(b!1)`	kp9

3: Rule Rewriting: Substitute structured species graphs with unstructured population species

1	`A(b~P) + B(r)`	->	`A(b~P!1).B(r!1)`	kp9
2	`A(b~P) + P6()`	->	`A(b~P!1).B(r!1,c)`	kp9
3	`A(b~P) + P7()`	->	`A(b~P!2).B(r!2,c!1).C(b!1)`	kp9
4	`P3() + B(r)`	->	`A(r,b~P!1).B(r!1)`	kp9
5	`P3() + P6()`	->	`P4()`	kp9
6	`P3() + P7()`	->	`P5()`	kp9

Figure 4. Partial network expansion (PNE) applied to Rule 11f of Fig. 3. See Text S4 of the supporting material for the complete, partially-expanded model.
doi:10.1371/journal.pcbi.1003544.g004

the self match (identity automorphism) is added to the reactant match list in both cases. Next, the rule is applied to each possible reactant set (the cartesian product of the reactant match lists). This results in a set of six derived rules. The structured population species are then replaced in these rules by their associated unstructured species, resulting in one pure particle rule (the original rule), three mixed particle/population rules, and two pure population reactions. Including the population-mapping rules, the hybrid model contains a total of 42 rules, more than the original 16 but significantly less than the 287 reactions of the fully-expanded network. The complete partially-expanded HPP model in BNGL format can be found in Text S4 of the supporting material.

Population-adapted network-free simulation. Although modified relative to the original, the hybrid model generated from PNE remains properly a rule-based model. As such, it can, in principle, be simulated with any of the network-based (after network generation) and network-free simulation methods described above. However, the advantage of recasting the original model into the hybrid form is that it can be represented as a collection of particles and population objects and simulated using a modified network-free method that has the following attributes: (i) a population count property for each molecule object; (ii) a transformation that performs population increments and decrements; (iii) a method for calculating population-weighted propensities (rates). Examples of population-adapted network-free simulators are NFsim 1.11 and KaSim 3.x.

The population-weighted propensity of a rule R_μ can be calculated as

$$a_\mu = \frac{k_\mu}{s_\mu} \prod_{r=1}^{M_\mu} \left(\sum_{x=1}^{X} \rho(x) \eta_{\mu,r}(x) \right). \quad (1)$$

Here, k_μ is the rate constant (more generally, the "single-site rate law" [6]), s_μ is the symmetry factor (see Note 4.21 of Ref. [6]), M_μ is the number of reactant patterns in the rule (i.e., the *molecularity*), X is the total number of complexes in the system, $\rho(x)$ is the population of complex x (unity in the case of particles), and $\eta_{\mu,r}(x)$ is the number of matches of reactant pattern r into complex x (unity or zero for unstructured population species, i.e., the species either is the reactant or it is not). The difference between Eq. 1 and the formula used for calculating propensities in standard network-free simulators is the term $\rho(x)$; a fully particle-based network-free

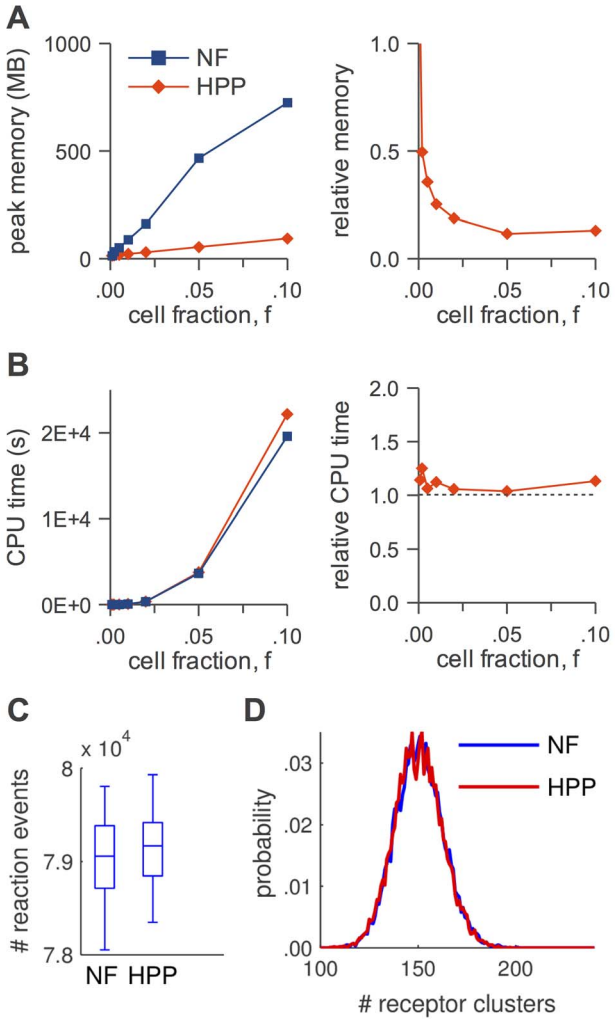

Figure 5. HPP performance analysis for the TLBR model. (A) peak memory usage (*left*: absolute, *right*: relative to NFsim); (B) CPU run time (*left*: absolute, *right*: relative to NFsim); (C) number of reaction events fired during a simulation ($f=0.01$); (D) equilibrium distribution of number of clusters ($f=0.01$). The slight deviation from linearity for 'NF' in (A) is an artifact of how memory is allocated in NFsim.
doi:10.1371/journal.pcbi.1003544.g005

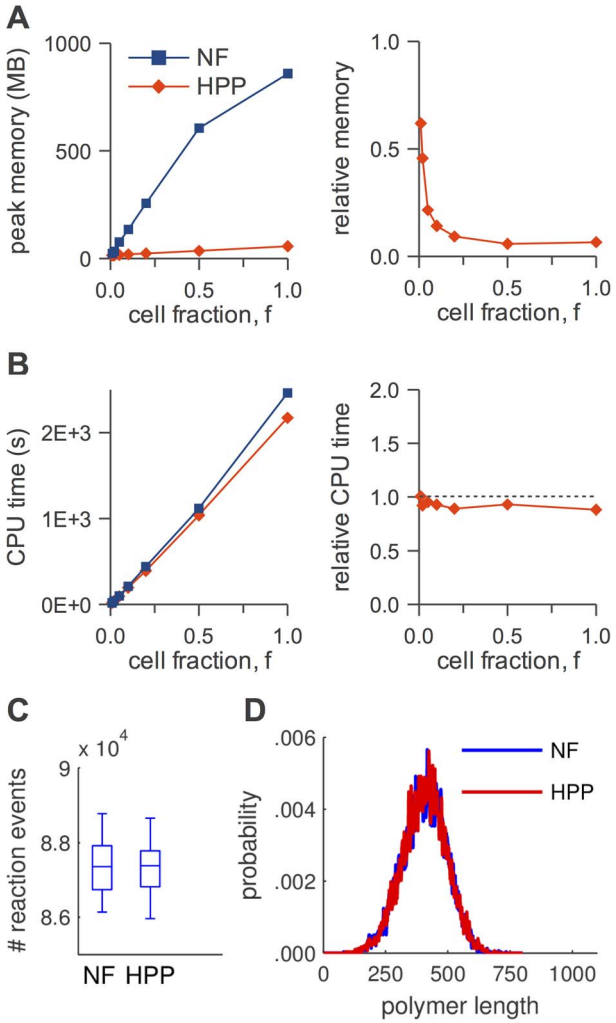

Figure 6. HPP performance analysis for the actin polymerization model. (A) peak memory usage (*left*: absolute, *right*: relative to NFsim); (B) CPU run time (*left*: absolute, *right*: relative to NFsim); (C) number of reaction events fired during a simulation ($f=0.01$); (D) equilibrium distribution of actin polymer lengths ($f=0.01$). The slight deviation from linearity for 'NF' in (A) is an artifact of how memory is allocated in NFsim.
doi:10.1371/journal.pcbi.1003544.g006

calculation is recovered if all $\rho(x)=1$. Conversely, the difference between Eq. 1 and the formula used in network-based SSA simulators is the term $\eta_{\mu,r}(x)$; a fully population-based calculation is recovered if all $\eta_{\mu,r}(x)=0$ or 1, in which case X is the total number of species in the network. Equation 1 thus generalizes the concept of propensity for hybrid systems comprised of both particles and population variables.

Also note that for symmetric population reactions, e.g., pop_A()+pop_A()→A(a!0).A(a!0), the possibility of a null event must be calculated in order to prevent reactions involving the same molecule. This is accomplished by rejecting the event with probability $1/\rho(x)$. Furthermore, since population species have zero components, if complex x is a population species and $\eta_{\mu,r}(x)=1$, then $\eta_{\mu,r}(y)=0$ for all $y\neq x$. This property is useful because it guarantees that a reactant pattern matches either particles or population species exclusively, never a mixture of both. Thus, once a rule has been selected to fire, the particles to participate in that rule can be selected from a uniform distribution rather than from a population-weighted distribution.

Performance analyses

Peak memory use and CPU run time. In Figs. 5–8, panels A, we show absolute and relative (with respect to NFsim) peak memory use as a function of cell fraction, f, for all models considered. We see that in all tested cases HPP requires less memory than NFsim. For NFsim, we also see the expected linear relationship (Table 1) between peak memory use and particle number (i.e., cell fraction; the slight deviation from linearity is an artifact of how memory is allocated in NFsim). For HPP, peak memory use also scales linearly with particle number, but with a smaller slope. This is the expected behavior since as the cell fraction is increased (keeping concentrations constant) a portion of the added particles, and hence memory cost, is always absorbed by the population portion of the system. Furthermore, in cases where network generation is possible (FcεRI, Fig. 7A; EGFR, Fig. 8A), we see the expected constant relationship between memory usage and particle number for the SSA (Table 1). We also see that the SSA requires more memory than both NFsim and HPP for all cell

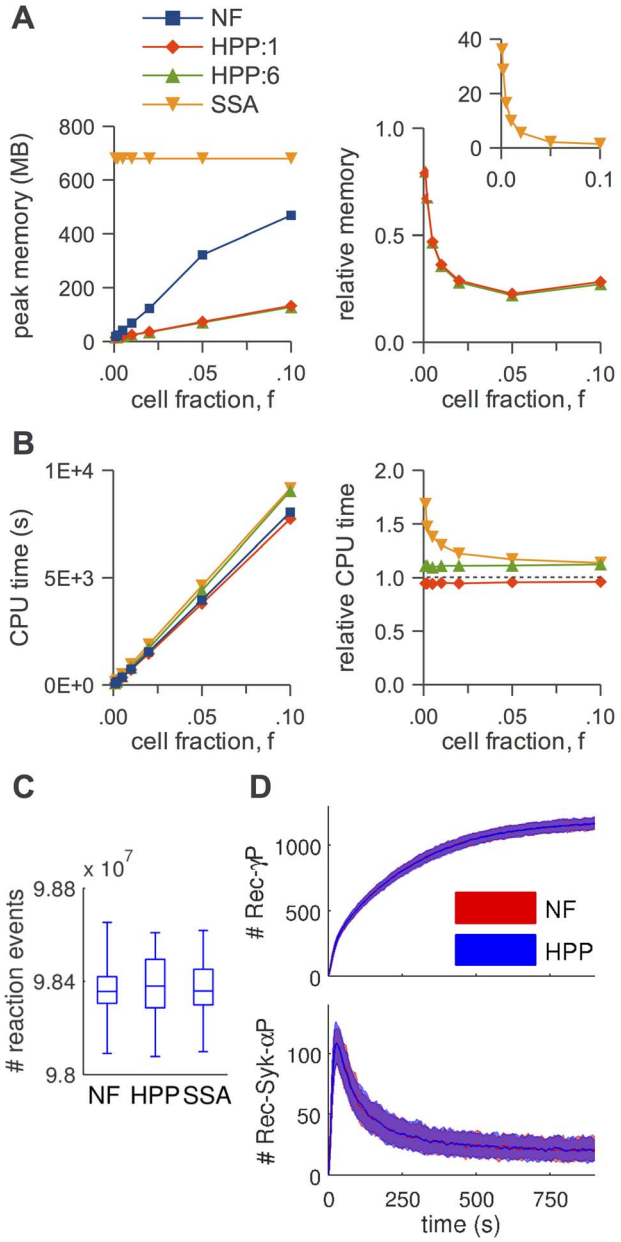

Figure 7. HPP performance analysis for the FcεRI signaling model. (A) peak memory usage (*left*: absolute, *right*: relative to NFsim); (B) CPU run time (*left*: absolute, *right*: relative to NFsim); (C) number of reaction events fired during a simulation ($f=0.01$); (D) timecourses (means and 5–95% frequency envelopes; $f=0.01$) for $\gamma-$phosphorylated receptor (*top*) and receptor-recruited, $\alpha-$phosphorylated Syk (*bottom*). The slight deviation from linearity for 'NF' in (A) is an artifact of how memory is allocated in NFsim. SSA timecourses are virtually indistinguishable from those in (D) and have been omitted for clarity.
doi:10.1371/journal.pcbi.1003544.g007

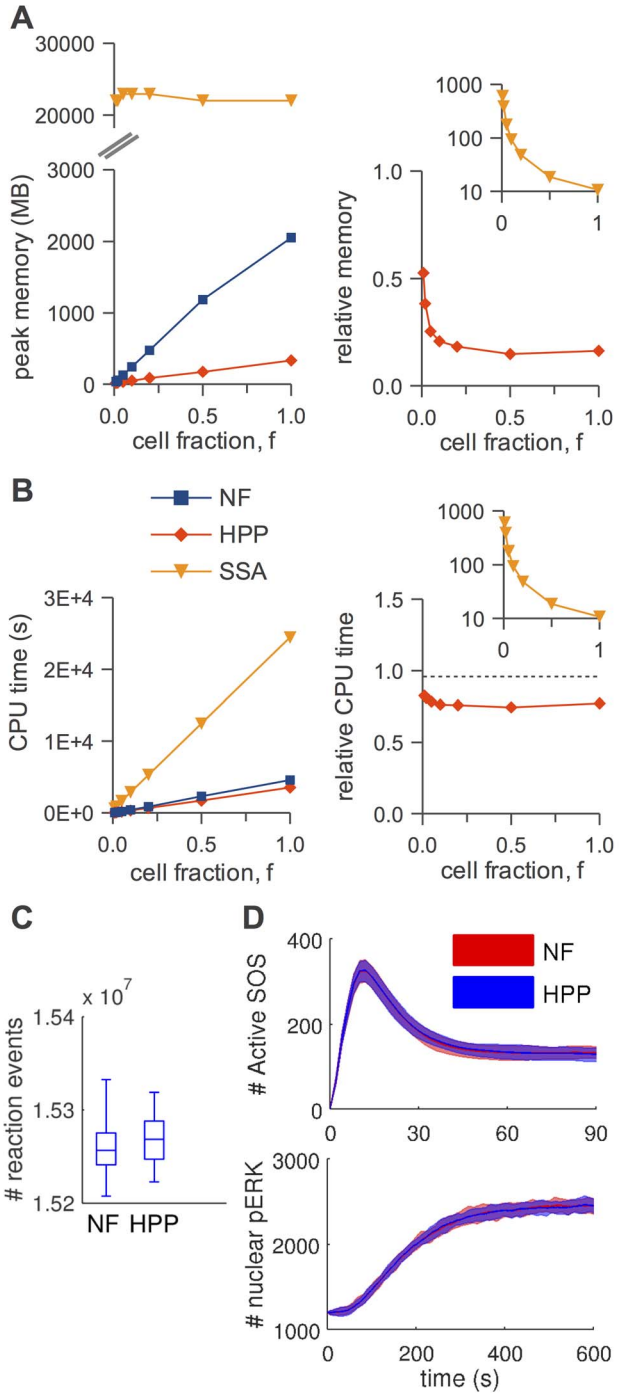

Figure 8. HPP performance analysis for the EGFR signaling model. (A) peak memory usage (*left*: absolute, *right*: relative to NFsim); (B) CPU run time (*left*: absolute, *right*: relative to NFsim); (C) number of reaction events fired during a simulation ($f=0.05$); (D) timecourses (means and 5–95% frequency envelopes; $f=0.05$) for activated Sos (*top*) and nuclear phosphorylated ERK (*bottom*). The slight deviation from linearity for 'NF' in (A) is an artifact of how memory is allocated in NFsim. Due to high computational expense, SSA statistics were not collected in (C) and (D).
doi:10.1371/journal.pcbi.1003544.g008

fractions considered. This is due to the high memory cost of the dependency update graph [38] used in the SSA implementation within BioNetGen, which scales with the product of the number of reactions in the network and the number of reactions updated after each reaction firing (see Table 1).

In Figs. 5–8, panels B, we show absolute and relative (with respect to NFsim) CPU run times as a function of cell fraction. Generally speaking, HPP and NFsim run times are comparable in all cases, indicating that the reductions in memory use seen in Figs. 5–8, panels A, are not achieved at the cost of increased run

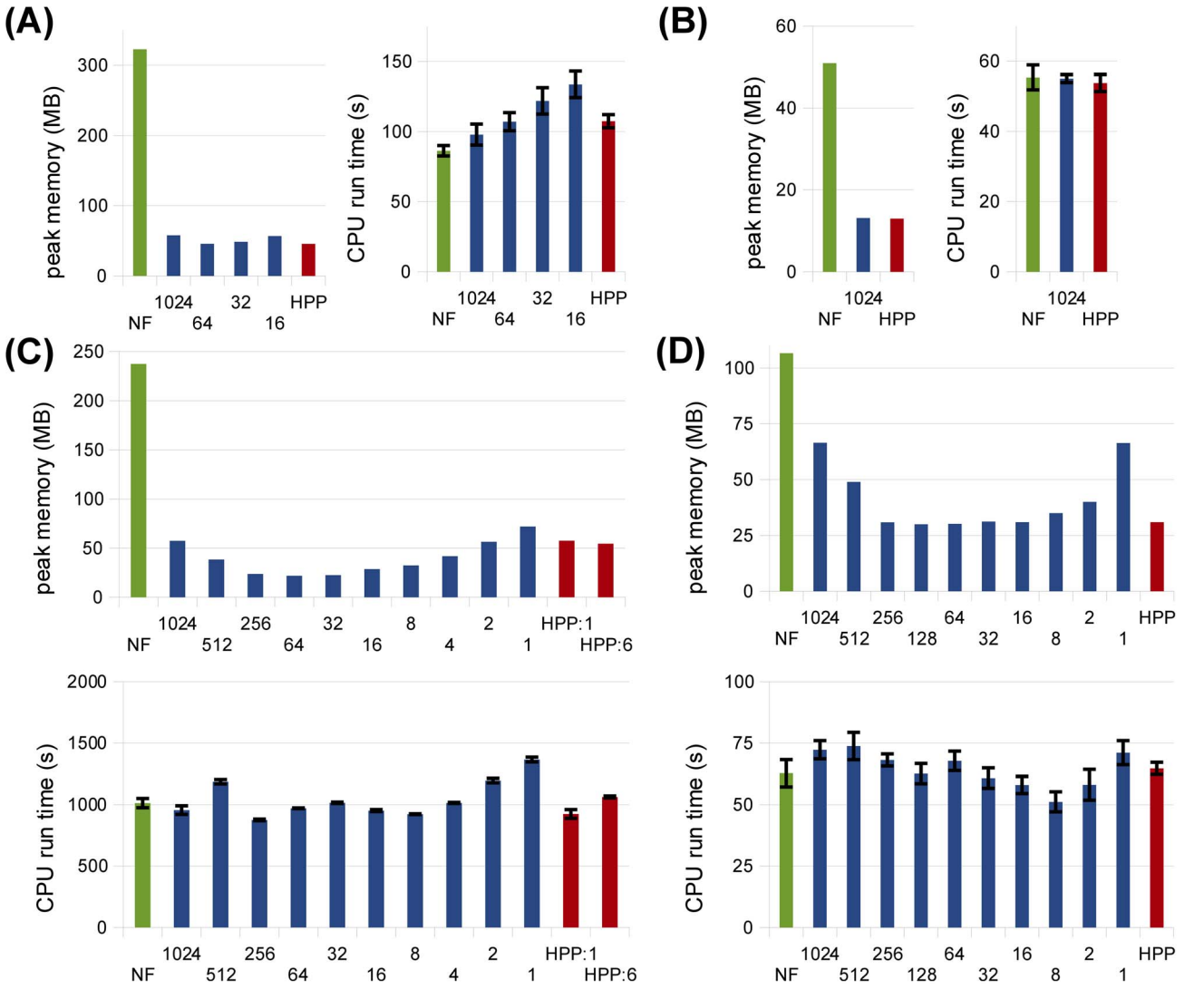

Figure 9. HPP performance analyses for various lumping thresholds at cell fraction $f = 0.01$. (A) TLBR; (B) Actin; (C) FcεRI; (D) EGFR. In all plots, threshold values for different lumping sets are shown on the x-axis. For TLBR and Actin, some thresholds yield the same set of population species as larger thresholds and are thus omitted from the figures. For TLBR, results for thresholds <16 are omitted due to impractically large partial networks in those cases. Results for NFsim ('NF') and the hand-picked lumping sets from Figs. 5–8 ('HPP') are shown in all plots for comparison. Error bars show standard error (three samples).
doi:10.1371/journal.pcbi.1003544.g009

times. In fact, HPP is slightly faster than NFsim in most cases. This is because operations on population species (e.g., increment/decrement) are less costly than the graph operations applied to particles (e.g., subgraph matching). Also note in Fig. 5B the expected quadratic relationship between run time and particle number for the TLBR model (Table 1), which is due to the formation of a super aggregate near the solution-gel phase boundary [23,42]. In Figs. 7B and 8B, we see that the SSA is slower than both NFsim and HPP for all cell fractions considered. The difference is most pronounced at small cell fractions and is much more significant for EGFR than for FcεRI. This is expected since previous work [16] has shown that network-free methods perform particularly well for systems with small numbers of particles and large networks (the EGFR network is significantly larger than the FcεRI network; Table 2). Finally, we see in Fig. 7B that the CPU run time increases as we increase the number of species treated as populations in the FcεRI model, even though the memory usage remains constant (Fig. 7A). This is interesting because it suggests that the FcεRI : 1 variant, with free ligand as the only population species, is near-optimally lumped for the cell fractions considered. We revisit the issue of optimal lumping sets below.

Accuracy. In Figs. 5–8, panels C, we show distributions of the number of reaction firings per simulation run for each of the simulation methods considered. It is evident that for all models the distributions, as illustrated by box plots, are similar for NFsim, HPP, and SSA (the latter for FcεRI only; Fig. 7C). Statistically speaking, the two-sided Mann-Whitney U test [74,75] was unable to reject the null hypothesis in all cases at the 5% significance level (TLBR: $p = 0.25$; Actin: $p = 0.90$; FcεRI: $p = 0.27$; EGFR: $p = 0.07$). There is no evidence, therefore, that HPP does not generate statistically identical numbers of reaction firings to both NFsim and SSA. This is as expected since all methods are exact-stochastic approaches.

In Figs. 5–8, panels D, we compare distributions obtained from NFsim and HPP simulations of all models. In Fig. 5D, we show equilibrium distributions of the number of receptor clusters in the TLBR model ($f=0.01$). In Fig. 6D, equilibrium distributions of polymer lengths in the Actin model are shown ($f=0.01$). In both cases, the NFsim and HPP distributions are statistically indistinguishable (TLBR: $p=0.50$; Actin: $p=0.66$). In Fig. 7D, time courses for $\gamma-$phosphorylated receptor and receptor-recruited, $\alpha-$phosphorylated Syk are shown ($f=0.01$). In Fig. 8D, time courses for membrane-recruited (active) SOS and nuclear phospho-ERK are shown ($f=0.05$). Although we did not perform any statistical tests, visual inspection of the trajectories clearly shows that in all cases the NFsim and HPP results are virtually identical.

Systematic approach to selecting population species. All of the HPP results presented in Figs. 5–8 were obtained with "hand-picked" sets of population species chosen based on modeler experience and intuition. The significant memory savings seen in these plots imply that this approach will often be sufficient in practice. However, it is fair to ask whether a more systematic approach to selecting population species can achieve additional memory savings. In order to address this question, we considered a variety of different lumping sets for each example model and compared their performance in terms of memory usage and CPU run time. The lumping sets were chosen based on average species populations calculated over the course of a single NFsim pre-simulation at cell fraction $f=0.01$. Specifically, at periodic intervals, the full set of complexes in the system was collected, each complex canonically labeled, and the number of instances of each label (i.e., species) counted. Average values over the entire simulation were then calculated for each species. Sets of population species were constructed by lumping all species with an average population greater than a range of pre-defined thresholds. For convenience, we chose thresholds of $2^n, n\in[0,10]$. Average species populations obtained from each NFsim pre-simulation are provided in supplementary Dataset S1. The script that implements this method (for a single threshold) has been included in the recent BioNetGen 2.2.5 release (auto_hpp.pl in the Perl2 subdirectory).

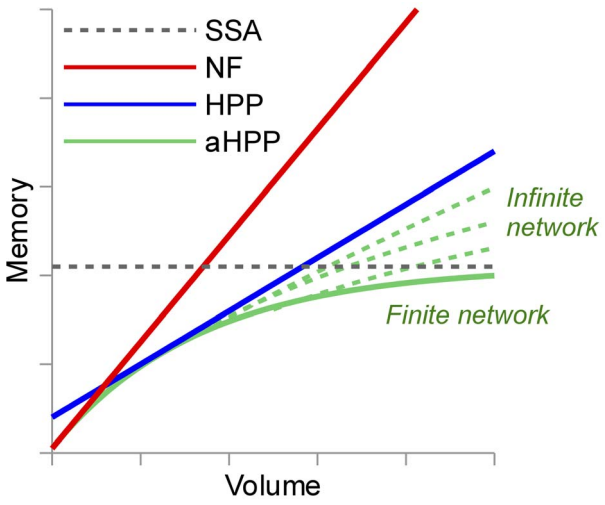

Figure 10. Memory use vs. simulated volume for different simulation methods, including a hypothetical automated HPP (aHPP). For finite networks, aHPP memory use plateaus once the entire reaction network has been generated. For infinite networks, the scaling at large volumes should fall somewhere between constant and linear (no worse than HPP) depending on the model (see Sec. S2 of Text S1 for an analysis).
doi:10.1371/journal.pcbi.1003544.g010

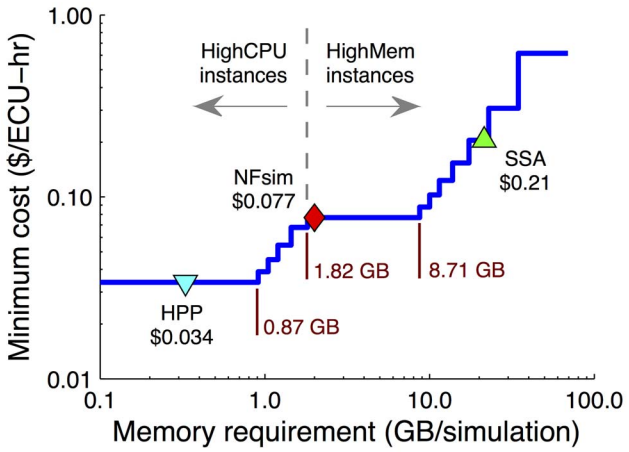

Figure 11. Cost of running simulations on the Amazon Elastic Compute Cloud (EC2). The minimum cost as a function of memory requirement was calculated based on January 2012 pricing (http://aws.amazon.com/ec2/) of all *Standard*, *High-CPU*, and *High-Memory* EC2 instances (see Sec. S1 of Text S1 for details of the calculation). Also included are values for NFsim, HPP, and SSA simulations of the EGFR model at cell fraction $f=1$.
doi:10.1371/journal.pcbi.1003544.g011

In Fig. 9, we show peak memory use and CPU run times for HPP simulations of each model at each lumping set considered. In general, these results illustrate the success of the hand-picked lumping sets, which produced memory savings close to the optimal in most cases. There was, however, some room for improvement in the FcεRI model (Fig. 9C). This is because the fourth and fifth most populated species for this model were complexes comprised of five molecular subunits (see Dataset S1). Since we did not anticipate this result, these high-population species were not included in the hand-picked lumping set. The majority of the memory savings seen in Fig. 9C for thresholds >32 are due to lumping of these species. Thus, our results also illustrate the value of using a more systematic approach to selecting population species in some cases.

It is also interesting to note in Figs. 9C and 9D the presence of an optimal lumping threshold between the maximum and minimum values considered. At high thresholds, most species are treated as particles and higher memory use is expected. At low thresholds, however, the higher memory use is due to the larger size of the partially-expanded network. Also interesting is that the run time results in Fig. 9 show a weak (if any) dependence on the chosen threshold, despite the fact that the time complexity of network-free methods scales linearly with rule set size (Table 1). Presumably, this is because the lower cost operations (increment/decrement) associated with the population species offset the increased cost of larger rule sets. This robustness of the time cost with respect to the size of the lumping set is a positive attribute of the HPP method.

Discussion

We have presented a hybrid particle/population simulation approach for rule-based models of biological systems. The HPP approach is applied in two stages (Fig. 1): (i) transformation of a rule-based model into a dynamically-equivalent hybrid form by partially expanding the network around a selected set of

population species; (ii) simulation of the transformed model using a population-adapted network-free simulator. The method is formally exact for an infinite population lumping rate constant, but can produce statistically exact results in practice provided that a sufficiently large value is used (Figs. 5–8, panels C and D). As currently implemented, the primary advantage of the HPP method is in reducing memory usage during simulation (Figs. 5–8, panels A). Importantly, this is accomplished with little to no impact on simulation run time (Figs. 5–8, panels B).

We have shown that peak memory use for HPP scales linearly with particle number (with a slope that is smaller than for NFsim; Figs. 5–8, panels A) and confirmed that when network generation is possible SSA memory use is approximately independent of particle number (Figs. 7A and 8A). At the system volumes that we have considered here, HPP memory use is significantly less than for SSA. However, the linear scaling of HPP and the constant scaling of SSA indicate that with further increases in the system volume there will invariably come a point where HPP memory use exceeds that of SSA. This is because species that are rare at small volumes, and hence chosen to be treated as particles, become plentiful at large volumes. Intuitively, a partially-expanded network should never require more memory than a fully-enumerated network. However, as currently implemented, there is no way to strictly enforce this restriction because HPP requires that population species be chosen prior to PNE.

In Fig. 9, we have shown how a systematic approach to choosing population species can optimize memory usage for a given system volume. However, this approach requires running an NFsim pre-simulation, which may not be feasible for systems with extremely large numbers of particles (e.g., whole cells). Thus, we propose to develop a more general version of HPP that dynamically tracks the populations of species during the course of a simulation and automatically selects those to treat as population variables based on some criteria, e.g., that their population exceeds a certain threshold. In this automated version of HPP (aHPP), PNE would be performed every time a new species is lumped. If all species in the system become lumped then the network will naturally become fully enumerated. Hence, the memory load will never exceed that of the fully-expanded network. In Fig. 10, we provide a qualitative sketch of how we expect the memory usage of this hypothetical aHPP method to scale with system volume (particle number). Included for comparison are scalings for HPP, NFsim, and SSA. For models with finite networks (such as FcεRI and EGFR), aHPP memory use should plateau once the entire reaction network has been generated. For models with infinite networks (such as TLBR and Actin), we expect aHPP memory use at large volumes to scale somewhere between constant and linear (no worse than HPP) depending on the model. A detailed analysis of the space complexity of a hypothetical, ''optimal'' aHPP method is provided in Sec. S2 of supplementary Text S1.

In order to frame our results within a real-world context, we have estimated the cost of simulation based on hourly rates of on-demand instances on the Amazon Elastic Compute Cloud (EC2). In Fig. 11, we show the hourly cost (per ''effective compute unit'') of simulation as a function of required memory per simulation (details of the calculation can be found in Sec. S1 of Text S1). Also included in the plot are values for HPP (0.3 GB), NFsim (2.1 GB), and SSA (22.0 GB) simulations of the EGFR model at cell fraction $f=1$ (Fig. 8A). Our calculations show that below 1.82 GB of required memory *High-CPU* instances are the most cost effective. Above this threshold *High-Memory* instances are the better option. The HPP simulation falls below this cutoff while both NFsim and SSA lie above. There is a quantifiable benefit, therefore, to reducing memory usage in this case; HPP simulations on the EC2 would be ~ 2.5 and ~ 33 times less expensive, respectively, than NFsim and SSA (HPP is slightly faster than NFsim and significantly faster than SSA; Fig. 8B). Thus, the reduction in memory usage offered by HPP is not simply of academic interest but can impact, in a tangible way, the cost of doing computational research.

Finally, even greater benefits are possible if, in addition to reducing memory usage, the speed of HPP simulations can be increased. τ leaping [36,77–79] is an approach for accelerating stochastic simulations of chemically reactive systems. With a few exceptions (e.g., Ref. [80]), $\tau-$leaping has been applied primarily to fully-enumerated reaction networks. We believe that the HPP method provides a unique setting for the application of τ leaping because, unlike in pure particle-based methods, there exists a partial network of reactions that act on population species. Thus, a network-based $\tau-$leaping method can be applied exclusively to the population component of a system while retaining the network-free approach in the particle component. We have recently implemented a $\tau-$leaping variant in BioNetGen, known as the partitioned-leaping algorithm [22], and are actively working on integrating it with the HPP.

Supporting Information

Dataset S1 Average species populations from NFsim pre-simulations ($f=0.01$) of all example models considered in Fig. 9.
(XLSX)

Figure S1 Average number of reactions that must be updated after each reaction firing (i.e, dependencies) for a collection of FcεRI signaling models of varying network size (all models are included in the BioNetGen 2.2.5 release available at http://bionetgen.org).
(EPS)

Text S1 Sec. S1: Details of the monetary cost analysis shown in Fig. 11; Sec. S2: Space complexity analyses for the network-based SSA, network-free, HPP, and hypothetical aHPP methods (Fig. 10); Sec. S3: Overview of BNGL, model files, and running HPP simulations with BioNetGen/NFsim; Sec. S4: BNGL formalism and the formal foundation of the PNE algorithm (with pseudocode).
(PDF)

Text S2 Complete BNGL file for the simple receptor activation model of Fig. 3 (receptor_activation.bngl).
(TXT)

Text S3 HPP configuration file for the simple receptor activation model, including population mapping rules and instructions for executing NFsim and HPP simulations (run_receptor_activation.bngl).
(TXT)

Text S4 Partially-expanded (HPP) version of the simple receptor activation model of Fig. 3 generated using the method outlined in Fig. 4 (receptor_activation_hpp.bngl).
(TXT)

Text S5 BNGL file for the TLBR model (tlbr.bngl).
(TXT)

Text S6 HPP configuration file for the TLBR model (run_tlbr.bngl).
(TXT)

Text S7 HPP version of the TLBR model (tlbr_hpp.bngl).
(TXT)

Text S8 BNGL file for the Actin model (actin_simple.bngl).
(TXT)

Text S9 HPP configuration file for the Actin model (run_actin_simple.bngl).
(TXT)

Text S10 HPP version of the Actin model (actin_simple_hpp.bngl).
(TXT)

Text S11 BNGL file for the FcεRI model (fceri_gamma2.bngl).
(TXT)

Text S12 HPP configuration file for the FcεRI model (run_fceri_gamma2.bngl).
(TXT)

Text S13 HPP version of the FcεRI model with free ligand treated as the only population species (fceri_gamma2_hpp1.bngl).
(TXT)

Text S14 HPP version of the FcεRI model with free ligand, cytosolic Lyn and all four phosphorylation states of cytosolic Syk treated as population species (fceri_gamma2_hpp6.bngl).
(TXT)

Text S15 BNGL file for the EGFR model (egfr_extended.bngl).
(TXT)

Text S16 HPP configuration file for the EGFR model (run_egfr_extended.bngl).
(TXT)

Text S17 HPP version of the EGFR model (egfr_extended_hpp.bngl).
(TXT)

Acknowledgments

We thank Michael Sneddon for assistance in preparing the NFsim 1.11 distribution (population-adapted).

Author Contributions

Conceived and designed the experiments: JSH LAH JRF. Performed the experiments: JSH. Analyzed the data: JSH LAH JRF. Contributed reagents/materials/analysis tools: JSH LAH LJS NSN JRF. Wrote the paper: JSH LAH JRF.

References

1. Alberts B, Johnson A, Lewis J, Raff M, Roberts K, et al. (2002) Molecular Biology of the Cell. Garland Science, 4th edition.
2. Hlavacek WS, Faeder JR, Blinov ML, Perelson AS, Goldstein B (2003) The complexity of complexes in signal transduction. Biotechnol Bioeng 84: 783–794.
3. Hlavacek WS, Faeder JR, Blinov ML, Posner RG, Hucka M, et al. (2006) Rules for modeling signal-transduction systems. Sci STKE 2006: re6.
4. Aldridge BB, Burke JM, Lauffenburger DA, Sorger PK (2006) Physicochemical modelling of cell signalling pathways. Nat Cell Biol 8: 1195–1203.
5. Blinov ML, Faeder JR, Goldstein B, Hlavacek WS (2004) BioNetGen: software for rule-based modeling of signal transduction based on the interactions of molecular domains. Bioinformatics 17: 3289–3291.
6. Faeder JR, Blinov ML, Hlavacek WS (2009) Rule-based modeling of biochemical systems with BioNetGen. In: Methods in Molecular Biology, Clifton, N.J.: Humana Press, volume 500. pp. 113–167.
7. Danos V, Laneve C (2004) Formal molecular biology. Theor Comput Sci 325: 69–110.
8. Danos V, Feret J, Fontana W, Harmer R, Krivine J (2007) Rule-based modelling of cellular signalling. Lect Notes Comput Sci 4703: 17–41.
9. Maus C, Rybacki S, Uhrmacher AM (2011) Rule-based multi-level modeling of cell biological systems. BMC Syst Biol 5: 166.
10. Bittig AT, Haack F, Maus C, Uhrmacher AM (2011) Adapting rule-based model descriptions for simulating in continuous and hybrid space. In: Proceedings of the 9th International Conference on Computational Methods in Systems Biology. New York, NY, USA: ACM, CMSB '11, pp. 161–170.
11. Lis M, Artyomov MN, Devadas S, Chakraborty AK (2009) Efficient stochastic simulation of reaction–diffusion processes via direct compilation. Bioinformatics 25: 2289–2291.
12. Angermann BR, Klauschen F, Garcia AD, Prustel T, Zhang F, et al. (2012) Computational modeling of cellular signaling processes embedded into dynamic spatial contexts. Nat Meth 9: 283–289.
13. Faeder J (2011) Toward a comprehensive language for biological systems. BMC Biol 9: 68.
14. Sekar JAP, Faeder JR (2012) Rule-based modeling of signal transduction: A primer. In: Methods In Molecular Biology, Clifton, N.J.: Humana Press, volume 880. pp. 139–218.
15. Blinov ML, Yang J, Faeder JR, Hlavacek WS (2006) Graph theory for rule-based modeling of biochemical networks. Lect Notes Comput Sci 4230: 89–106.
16. Sneddon MW, Faeder JR, Emonet T (2011) Efficient modeling, simulation and coarse-graining of biological complexity with NFsim. Nat Meth 8: 177–183.
17. Faeder JR, Blinov ML, Goldstein B, Hlavacek WS (2005) Rule-based modeling of biochemical networks. Complexity 10: 22–41.
18. Blinov ML, Faeder JR, Goldstein B, Hlavacek WS (2006) A network model of early events in epidermal growth factor receptor signaling that accounts for combinatorial complexity. Biosystems 83: 136–151.
19. Hindmarsh AC, Brown PN, Grant KE, Lee SL, Serban R, et al. (2005) SUNDIALS: suite of nonlinear and differential/algebraic equation solvers. ACM Trans Math Software 31: 363–396.
20. Gillespie DT (1976) A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. J Comput Phys 22: 403–434.
21. Fricke T, Wendt D (1995) The Markoff-automaton: a new algorithm for simulating the timeevolution of large stochastic dynamic systems. Int J Mod Phys C 6: 277–306.
22. Harris LA, Clancy P (2006) A "partitioned leaping" approach for multiscale modeling of chemical reaction dynamics. J Chem Phys 125: 144107.
23. Yang J, Monine MI, Faeder JR, Hlavacek WS (2008) Kinetic Monte Carlo method for rule-based modeling of biochemical networks. Phys Rev E 78: 031910.
24. Danos V, Feret J, Fontana W, Krivine J (2007) Scalable simulation of cellular signaling networks. Lect Notes Comput Sci 4807: 139–157.
25. Bortz AB, Kalos MH, Lebowitz JL (1975) A new algorithm for Monte Carlo simulation of Ising spin systems. J Comput Phys 17: 10–18.
26. Colvin J, Monine MI, Gutenkunst RN, Hlavacek WS, Von Hoff DD, et al. (2010) RuleMonkey: software for stochastic simulation of rule-based models. BMC Bioinformatics 11: 404.
27. Colvin J, Monine MI, Faeder JR, Hlavacek WS, Von Hoff DD, et al. (2009) Simulation of large-scale rule-based models. Bioinformatics 25: 910–917.
28. Grünert G, Dittrich P (2011) Using the SRSim software for spatial and rule-based modeling of combinatorially complex biochemical reaction systems. Lect Notes Comput Sci 6501: 240–256.
29. Yang J, Hlavacek WS (2011) The efficiency of reactant site sampling in network-free simulation of rule-based models for biochemical systems. Phys Biol 8: 055009.
30. Tomita M (2001) Whole-cell simulation: a grand challenge of the 21st century. Trends Biotechnol 19: 205–210.
31. Karr JR, Sanghvi JC, Macklin DN, Gutschow MV, Jacobs JM, et al. (2012) A whole-cell computational model predicts phenotype from genotype. Cell 150: 389–401.
32. Alon U (2006) An Introduction to Systems Biology: Design Principles of Biological Circuits. New York: Chapman & Hall/CRC.
33. Moran U, Phillips R, Milo R (2010) Snapshot: Key numbers in biology. Cell 141: 1262–1262.e1.
34. Cormen TH, Leiserson CE, Rivest RL, Stein C (2001) Introduction to Algorithms. Cambridge: MIT University Press.
35. Gillespie DT (1977) Exact stochastic simulation of coupled chemical reactions. J Phys Chem 81: 2340–2361.
36. Gillespie DT (2007) Stochastic simulation of chemical kinetics. Annu Rev Phys Chem 58: 35–55.
37. Gibson MA, Bruck J (2000) Efficient exact stochastic simulation of chemical systems with many species and many channels. J Phys Chem A 104: 1876–1889.
38. Cao Y, Li H, Petzold L (2004) Efficient formulation of the stochastic simulation algorithm for chemically reacting systems. J Chem Phys 121: 4059–4067.
39. Slepoy A, Thompson AP, Plimpton SJ (2008) A constant-time kinetic Monte Carlo algorithm for simulation of large biochemical reaction networks. J Chem Phys 128: 205101.
40. Schulze TP (2008) Efficient kinetic Monte Carlo simulation. J Comput Phys 227: 2455–2462.

41. McCollum JM, Peterson GD, Cox CD, Simpson ML, Samatova NF (2006) The sorting direct method for stochastic simulation of biochemical systems with varying reaction execution behavior. Comput Biol Chem 30: 39–49.
42. Monine MI, Posner RG, Savage PB, Faeder JR, Hlavacek WS (2010) Modeling multivalent ligandreceptor interactions with steric constraints on configurations of cell-surface receptor aggregates. Biophys J 98: 48–56.
43. Roland J, Berro J, Michelot A, Blanchoin L, Martiel JL (2008) Stochastic severing of actin filaments by actin depolymerizing factor/cofilin controls the emergence of a steady dynamical regime. Biophys J 94: 2082–2094.
44. Haseltine EL, Rawlings JB (2002) Approximate simulation of coupled fast and slow reactions for stochastic chemical kinetics. J Chem Phys 117: 6959–6969.
45. Kiehl TR, Mattheyses RM, Simmons MK (2004) Hybrid simulation of cellular behavior. Bioinformatics 20: 316–322.
46. Takahashi K, Kaizu K, Hu B, Tomita M (2004) A multi-algorithm, multi-timescale method for cell simulation. Bioinformatics 20: 538–546.
47. Vasudeva K, Bhalla US (2004) Adaptive stochastic-deterministic chemical kinetic simulations. Bioinformatics 20: 78–84.
48. Burrage K, Tian T, Burrage P (2004) A multi-scaled approach for simulating chemical reaction systems. Prog Biophys Mol Bio 85: 217–234.
49. Salis H, Kaznessis Y (2005) Accurate hybrid stochastic simulation of a system of coupled chemical or biochemical reactions. J Chem Phys 122: 054103.
50. Salis H, Sotiropoulos V, Kaznessis YN (2006) Multiscale Hy3S: Hybrid stochastic simulation for supercomputers. BMC Bioinformatics 7: 93.
51. Griffith M, Courtney T, Peccoud J, Sanders WH (2006) Dynamic partitioning for hybrid simulation of the bistable HIV-1 transactivation network. Bioinformatics 22: 2782–2789.
52. Wylie DC, Hori Y, Dinner AR, Chakraborty AK (2006) A hybrid deterministic-stochastic algorithm for modeling cell signaling dynamics in spatially inhomogeneous environments and under the influence of external fields. J Phys Chem B 110: 12749–12765.
53. Liu Z, Pu Y, Li F, Shaffer CA, Hoops S, et al. (2012) Hybrid modeling and simulation of stochastic effects on progression through the eukaryotic cell cycle. J Chem Phys 136: 034105.
54. Liu Z, Mobassera UJ, Shaffer CA, Watson LT, Cao Y (2010) Multistate modeling and simulation for regulatory networks. In: Johansson B, Jain S, Montoya-Torres J, Hugan J, Yücesan E, editors, Proceedings of the 2010 Winter Simulation Conference. pp. 631–642.
55. Falkenberg CV, Blinov ML, Loew LM (2013) Pleomorphic ensembles: Formation of large clusters composed of weakly interacting multivalent molecules. Biophys J 105: 2451–2460.
56. Lok L, Brent R (2005) Automatic generation of cellular reaction networks with Moleculizer 1.0. Nat Biotechnol 23: 131–136.
57. Blinov ML, Faeder JR, Yang J, Goldstein B, Hlavacek WS (2005) 'On-the-fly' or 'generate-first' modeling? Nat Biotechnol 23: 1344–1345.
58. Creamer M, Stites E, Aziz M, Cahill J, Tan C, et al. (2012) Specification, annotation, visualization and simulation of a large rule-based model for ERBB receptor signaling. BMC Syst Biol 6: 107.
59. Borisov NM, Markevich NI, Hoek JB, Kholodenko BN (2005) Signaling through receptors and scaffolds: independent interactions reduce combinatorial complexity. Biophys J 89: 951–966.
60. Borisov NM, Markevich NI, Hoek JB, Kholodenko BN (2006) Trading the micro-world of combinatorial complexity for the macro-world of protein interaction domains. Biosystems 83: 152–166.
61. Conzelmann H, Fey D, Gilles ED (2008) Exact model reduction of combinatorial reaction networks. BMC Syst Biol 2: 78.
62. Borisov NM, Chistopolsky AS, Faeder JR, Kholodenko BN (2008) Domain-oriented reduction of rule-based network models. IET Syst Biol 2: 342–351.
63. Feret J, Danos V, Krivine J, Harmer R, Fontana W (2009) Internal coarse-graining of molecular systems. Proc Natl Acad Sci USA 106: 6453–6458.
64. Danos V, Feret J, Fontana W, Harmer R, Krivine J (2010) Abstracting the differential semantics of rule-based models: Exact and automated model reduction. In: 2010 25th Annual IEEE Symposium on Logic in Computer Science. pp. 362–381.
65. Goldstein B, Perelson AS (1984) Equilibrium theory for the clustering of bivalent cell surface receptors by trivalent ligands. Application to histamine release from basophils. Biophys J 45: 1109–1123.
66. Pollard TD, Borisy GG (2003) Cellular motility driven by assembly and disassembly of actin filaments. Cell 112: 453–465.
67. Lacayo CI, Pincus Z, VanDuijn MM, Wilson CA, Fletcher DA, et al. (2007) Emergence of largescale cell morphology and movement from local actin filament growth dynamics. PLOS Biol 5: e233.
68. Stone KD, Prussin C, Metcalfe DD (2010) IgE, mast cells, basophils, and eosinophils. J Allergy Clin Immun 125: S73–S80.
69. Goldstein B, Faeder JR, Hlavacek WS, Blinov ML, Redondo A, et al. (2002) Modeling the early signaling events mediated by FcεRI. Mol Immunol 38: 1213–1219.
70. Faeder JR, Hlavacek WS, Reischl I, Blinov ML, Metzger H, et al. (2003) Investigation of early events in FcεRI-mediated signaling using a detailed mathematical model. J Immunol 170: 3769–3781.
71. Stites EC, Trampont PC, Ma Z, Ravichandran KS (2007) Network analysis of oncogenic Ras activation in cancer. Science 318: 463–467.
72. Fujioka A, Terai K, Itoh RE, Aoki K, Nakamura T, et al. (2006) Dynamics of the Ras/ERK MAPK cascade as monitored by uorescent probes. J Biol Chem 281: 8917–8926.
73. Overbeck AF, Brtva TR, Cox AD, Graham SM, Huff SY, et al. (1995) Guanine nucleotide exchange factors: activators of Ras superfamily proteins. Mol Reprod Dev 42: 468–476.
74. Wilcoxon F (1945) Individual comparisons by ranking methods. Biometrics Bull 1: 80–83.
75. Mann HB (1947) On a test of whether one of two random variables is stochastically larger than the other. Ann Math Stat 18: 50–60.
76. Pearson K (1900) On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. Philos Mag 50: 157–175.
77. Gillespie DT (2001) Approximate accelerated stochastic simulation of chemically reacting systems. J Chem Phys 115: 1716–1733.
78. Gillespie DT, Petzold LR (2003) Improved leap-size selection for accelerated stochastic simulation. J Chem Phys 119: 8229–8234.
79. Cao Y, Gillespie DT, Petzold LR (2006) Efficient step size selection for the tau-leaping simulation method. J Chem Phys 124: 044109.
80. Vlachos DG (2008) Temporal coarse-graining of microscopic-lattice kinetic Monte Carlo simulations via $\mathcal{T}$ leaping. Phys Rev E 78: 046713.
81. Goldstein B, Faeder JR, Hlavacek WS (2004) Mathematical and computational models of immunereceptor signalling. Nat Rev Immunol 4: 445–456.