Chapter 8

Rule-Based Modeling Using Wildcards in the Smoldyn Simulator

Steven S. Andrews

Abstract

Many biological molecules exist in multiple variants, such as proteins with different posttranslational modifications, DNAs with different sequences, and phospholipids with different chain lengths. Representing these variants as distinct species, as most biochemical simulators do, leads to the problem that the number of species, and chemical reactions that interconvert them, typically increase combinatorially with the number of ways that the molecules can vary. This can be alleviated by "rule-based modeling methods," in which software generates the chemical reaction network from relatively simple "rules." This chapter presents a new approach to rule-based modeling. It is based on wildcards that match to species names, much as wildcards can match to file names in computer operating systems. It is much simpler to use than the formal rule-based modeling approaches developed previously but can lead to unintended consequences if not used carefully. This chapter demonstrates rule-based modeling with wildcards through examples for signaling systems, protein complexation, polymerization, nucleic acid sequence copying and mutation, the "SMILES" chemical notation, and others. The method is implemented in Smoldyn, a spatial and stochastic biochemical simulator, for both generate-first and on-the-fly expansion, meaning whether the reaction network is generated before or during the simulation.

Key words Rule-based modeling, Particle-based simulation, Wildcards, Reaction networks, Spatial simulation, Stochastic simulation, Brownian dynamics

1 Introduction

Since about the time that Boyle posited that matter was composed of minute particles "associated into minute masses or clusters" [1], now recognized as molecules, the dominant paradigm in chemistry has been to classify molecules into chemical species. This paradigm forms the foundation of chemical kinetics [2, 3] and is supported by the finding that different molecules of the same species are completely indistinguishable from each other

Electronic supplementary material: The online version of this chapter (https://doi.org/10.1007/978-1-4939-9102-0_8) contains supplementary material, which is available to authorized users.

William S. Hlavacek (ed.), *Modeling Biomolecular Site Dynamics: Methods and Protocols*, Methods in Molecular Biology, vol. 1945, https://doi.org/10.1007/978-1-4939-9102-0_8,

[4, 5]. Correspondingly, most modern biochemical simulation software represents molecules as members of species, treating all members of a single species identically (*see* reviews [6, 7]). However, many biological molecules do not fit neatly into these classes. For example, a cell might have a hundred or more DNA molecules, each with a different sequence. Similarly, a cell might have thousands of copies of some protein, but the copies vary according to whether they are bound to other proteins, bound to cofactors, or post-translationally modified with phosphate, methyl, or other moieties.

Several approaches have been developed to represent this molecular variation in computational models. One is to represent every multimer as an explicit graph, including its component monomers and their interconnections (e.g., [8–12]). Here, every molecule is its own entity and the concept of a species as a class of molecules is unnecessary. A second approach is to maintain the species concept, but to include states in the molecule definitions. For example, some biochemical simulators allow molecules to have modification states [13–15], surface-binding states [16], or an entire hierarchy of states [17]. A third approach is to define each molecular variant as a separate species, with minimal variation within species. The possible variations can lead to a combinatorial expansion in the number of species [18], leading to the development of so-called *rule-based modeling* methods for automating reaction network expansion from "rules" that describe molecular complexation and modifications (e.g., [19–21]).

The Smoldyn simulator represents molecular variation using this last approach, offering two different types of rule-based modeling [22]. Smoldyn is a widely used biochemical simulator that represents molecules as individual particles in 1D, 2D, or 3D space; these molecules diffuse, react with each other, and interact with surfaces [16, 23–25]. In one type of rule-based modeling, Smoldyn sends any rules in the user's input file that are written in the BioNetGen language (BNGL) to the BioNetGen software [19, 26]. BioNetGen expands the rules to lists of species and reactions. Then, Smoldyn reads the species and reactions; computes diffusion coefficients, graphical display parameters, and surface interactions for the new species; and runs the simulation [22]. Smoldyn's second type of rule-based modeling, which is the focus of this chapter, is based upon wildcard characters. Here, the modeler uses wildcard characters to specify groups of species to the Smoldyn software in much the same way as wildcard characters can be used to specify groups of files to the operating system. When used in chemical reactions, these wildcards can be used to define new species and new reactions.

Conventional rule-based modeling languages, including BNGL and Kappa [26, 27], are formal languages that are designed around an underlying model of how protein complexation and

modification generally work. The wildcard approach is different: it is simply a well-defined set of text-replacement tools with which modelers can create their own models and notational schemes. This offers substantial versatility and generally simplifies input files. However, this expanded freedom can also produce incorrect reaction networks if not used carefully. To address this, the main text of this chapter describes how wildcards work and the subsequent Notes section presents examples that illustrate how to use the method effectively.

2 Materials

Download Smoldyn from http://www.smoldyn.org. Smoldyn is free, open source, and licensed under the relatively permissive LGPL. The download package comes with install scripts, a detailed user's manual, over 100 example input files, related software tools (including BioNetGen), and, if desired, the source code. Install on Macs and Windows with the install scripts, which is generally easy. Install on Linux computers by compiling the source code with CMake and Make, which is also straightforward. Smoldyn runs on most laptops and larger computers that are less than 5 years old, as well as many older computers. Support is available by e-mailing support@smoldyn.org.

3 Methods

3.1 Running Smoldyn

To simulate a model in Smoldyn, start by describing the model in the Smoldyn language using a plain text file. Reference [28] and the Smoldyn User's Manual (included in the download package) describe how to write input files and give suggestions for parameter values.

Run Smoldyn at a shell prompt (a "Terminal" or "Command Line" application) by typing `smoldyn myfile.txt`, where `myfile.txt` is the configuration file name. Upon starting, Smoldyn reads model parameters from the configuration file, calculates and displays simulation parameters, and runs the simulation. As the simulation runs, Smoldyn displays the simulated system to a graphics window and saves quantitative data to one or more output files.

3.2 Wildcards for Matching

Molecules in Smoldyn are classified into chemical species and can also adopt any of five physical *states*. These states are in solution (e.g., a cell's cytoplasm) or the four surface-bound states called "front," "back," "up," and "down." Originally, the former two surface-bound states were for peripheral membrane proteins and the latter two were for integral membrane proteins, although they

are all essentially equivalent in practice. All molecules of a single species and state behave identically, meaning that they have the same diffusion coefficients, graphical display parameters, surface interaction rates, and chemical reaction rates. Any other molecular variation needs to be expressed using separate species. For example, if a model includes the yeast Fus3 protein, which can bind to zero, one, or two phosphate groups [29], then each of its phosphorylation states would need to be represented as a separate species. Alternatively, if a model includes a receptor that diffuses at one rate in normal membrane regions and more slowly in lipid rafts, then this variation would again need to be represented using separate species.

These groups of species can be easily represented using wildcards. For example, if the three Fus3 species were named Fus3, Fus3p, and Fus3pp, then the *species pattern* Fus3* would represent all three species. Also, if the receptors mentioned above were named R_normal and R_raft, then the species pattern R_* would represent both species. More generally, a species pattern is defined as a species name that may or may not include wildcard characters. In both of these examples, the "*" wildcard is used to represent variable portions of the species names.

Smoldyn supports *text-matching* and *structural* wildcards, where the former ones match to specific portions of the species names and the latter ones enable logical operations within species patterns. The text-matching wildcards include "*", which matches to any zero or more characters, "?", which matches to any one character, and [...], which matches to any one character from a specified list. The structural wildcard characters include "|", which is an OR operator, "&", which is a permutation operator, and {...}, which specifies the order of operation for the other two structural wildcards (the normal order of operations is that "&" takes precedence over "|"). The structural wildcards are most easily explained through examples, this time using the generic protein monomer names A, B, and C: the pattern A|B matches to either A or B; the pattern A&B matches to either AB or BA; the pattern A&B|C matches to AB, BA, or C; and the pattern A&{B|C} matches to AB, BA, AC, or CA. *See* Table 1.

Internally, when Smoldyn parses the user's input file and expects a species name, it inputs the given text as a species pattern. The pattern may be as simple as a single species name but could also include one or more wildcard characters. If the pattern does not include structural wildcards, then it is an *elementary pattern*. On the other hand, if it does include structural wildcards, such as the pattern A&*, then Smoldyn first expands it to a list of *elementary patterns*; in this A&* example, Smoldyn would expand it to the elementary patterns A* and *A. Next, Smoldyn scans through its list of species names to see which ones can match the elementary pattern(s). These matching species form a *species group*. If the

Table 1
Smoldyn wildcards

Symbol	Meaning	Matching example	Reaction example
?	Any 1 character	A? matches to AB, AC, etc.	A? → B?
*	0 or more characters	A* matches to A, AB, etc.	A* → B*
[...]	1 listed character	A[a-c] matches to Aa, Ab, Ac	A[u,p] → B[0,1]
|	OR operator	A|B|C matches to A, B, C	A|B → a|b
&	Permutation	A&B matches to AB, BA	A&B → a&b
{...}	Grouping	A{B|C} matches to AB, AC	A{b|c} → A{c|b}
$*n*	*n*'th match	Not applicable	A?? → B$2$1

pattern arose in a statement that defines species attributes (e.g., `difc`, for specifying the diffusion coefficient), then Smoldyn assigns the same attribute value to all species within the species group. Alternatively, if the pattern arose in a command that outputs information about molecules (e.g., `molcountspecies`, which counts the number of molecules of a given species or species group), then Smoldyn combines the appropriate information for all of the molecules that are in the species group.

3.3 Wildcards for Substitutions

Smoldyn also supports wildcards in chemical reaction definitions, where they can be used to specify multiple chemical reactions at once. Smoldyn inputs each chemical reaction equation as a *reaction pattern*, which again may include wildcards but does not have to.

First, consider elementary reaction patterns, meaning reaction patterns that do not contain structural wildcards. In this case, Smoldyn substitutes any text that the wildcards match for the reactants into the corresponding wildcards in the products. For example, the reaction Ste5 + Fus3* → Ste5-Fus3* specifies that any of the three Fus3 species described above (Fus3, Fus3p, and Fus3pp) can associate with the Ste5 protein [29]. In this case, the respective products would be Ste5-Fus3, Ste5-Fus3p, and Ste5-Fus3pp. If the same text-matching wildcard is used multiple times on each side of the equation, then Smoldyn corresponds the first instance in the reactants to the first instance in the products, the second to the second, and so on. For example, if Ste5 can also be phosphorylated, then Ste5* + Fus3* → Ste5*-Fus3* specifies that the binding reaction occurs for all phosphorylation states of both proteins, and that they maintain their phosphorylation states during the reaction. The correspondence can also be given explicitly using the "$*n*" wildcard on the product side of a reaction, using any value of *n* from 1 to 9, where it represents the *n*'th item of matching text. For example, the previous reaction could also be

written as Ste5* + Fus3* → Ste5$1-Fus3$2. Text-matching wildcards in the reactants do not have to appear in the products; for example, Fus3* → X shows that all three Fus3 species decay to the same product. On the other hand, text-matching wildcards in the products must appear in the reactants, meaning that Smoldyn would not accept the reaction X → Fus3*.

Much like the case for species patterns, Smoldyn expands reaction patterns that include structural wildcards to lists of elementary reaction patterns and then performs matching and substitution on these elementary patterns. In the reaction pattern A&* → X*, for example, Smoldyn would first expand it to the elementary reaction patterns A* → X* and *A → X*; Smoldyn would then perform matching and substitution on these two elementary reaction patterns. There are a few possible types of expansions. (1) If the reactant and product sides expand to the same number of elementary patterns, then Smoldyn assumes that they correspond to each other sequentially. For example, Smoldyn expands the reaction pattern A|B → C|D to the two reactions A → C and B → D. (2) Smoldyn accepts patterns that expand to only one elementary pattern on either the reactant or the product side, in each case creating a list of reactions that have either the same reactant or product. For example, A|B → X expands to A → X and B → X. Also, X → A|B expands to X → A and X → B. However, (3) Smoldyn does not accept patterns that expand to different numbers of elementary patterns on the reactant and product sides. For example, Smoldyn rejects the reaction pattern A|B|C → D|E.

In addition to the chemical reaction equation, Smoldyn allows modelers to specify several other reaction parameters. These include the reaction rate constant, how any dissociation products should be arranged, whether molecule serial numbers should be retained, and others. These parameters are entered in the same way for single reactions, reactions defined using wildcards, and reactions defined as rules, described next.

3.4 Reaction Network Expansion

In most cases, Smoldyn acts on input file statements as it encounters them. For example, if Smoldyn encounters a `difc` statement in an input file, it immediately sets the diffusion coefficient for all species that match the given species pattern to the given value. Likewise, if Smoldyn encounters a `reaction` statement, it immediately creates reactions for all currently defined species that match the given reaction pattern. In this case, Smoldyn issues either a warning or an error if any product names arise that are not currently defined species. Smoldyn does not revisit these statements during the simulation.

On the other hand, if the input file statement is suffixed with the text "`_rule`", such as in `difc_rule` or `reaction_rule`, then Smoldyn does not act on the statement immediately but instead stores it for future use (after a little preliminary parsing). Smoldyn

acts on these statements later on during *rule expansion*. Smoldyn supports two approaches for rule expansion. First, if it encounters an `expand_rules` statement in the input file (followed by "all" or a number), it expands the rules at that point. In this so-called *generate-first* approach [30], Smoldyn reads through the rules sequentially and acts on them using the currently defined species. In doing so, if it finds that a reaction specifies a product species that has not been defined, then Smoldyn creates the species. Smoldyn repeats this process for a user-specified number of iterations or until it has fully expanded the reaction network. This generate-first approach is often convenient for small reaction networks because Smoldyn displays all species and reactions before the simulation begins, making it easy to confirm that the network agrees with expectations (*see* **Notes 1–3**). Second, the rules can be expanded using the *on-the-fly* approach [30], in which Smoldyn acts on the rules at every time step during the simulation, but only as required. In particular, Smoldyn only generates the reactions for a species once the first molecule of the species has actually arisen in the simulation. This prevents the generation of unused species and reactions, which can be a large fraction of the possible ones [31]. This improves simulation efficiency for large reaction networks and can often enable simulations to run with reaction networks that would be infinitely large if they could be fully expanded (*see* **Notes 4–6**).

3.5 Properties of New Species

As mentioned above, the Smoldyn species properties include their diffusion coefficients, graphical display parameters, and surface interaction behaviors. These properties are typically assigned using the `difc`, `color`, `display_size`, `action`, and `rate` statements in the input file, where the last two define molecule-surface interaction behaviors. However, if Smoldyn acts on these statements before it performs reaction network expansion (which always happens when using on-the-fly expansion), then they do not apply to newly generated species. The rule statements described above, such as `difc_rule`, are one way to address this problem. An alternate and often better approach is that Smoldyn can assign species properties automatically by computing reaction product properties from the reactant properties.

It does so using the following assumptions: (1) reactants diffuse as though they are roughly spherical, (2) reactant volumes add upon binding, and (3) molecule diffusion coefficients scale as the inverse of the molecule's radius [22]. This last assumption follows from the Stokes-Einstein equation, which appears to be reasonably accurate even within cells [28, 32]. These assumptions lead to the following equations for the product of the generic reaction $A + B \rightarrow AB$:

$$r_{AB} = \sqrt[3]{r_A^3 + r_B^3}$$

$$D_{AB} = \left(D_A^{-3} + D_B^{-3}\right)^{-1/3}$$

where r_A and r_B are the reactant radii, D_A and D_B are the reactant diffusion coefficients, and r_{AB} and D_{AB} are the product radius and diffusion coefficient. Smoldyn assigns the product's graphical display radius from the r_{AB} equation. Next, Smoldyn computes the product's display color using a radius-weighted average of the reactant colors. For each of the red, green, and blue colors, it computes the product brightness value using

$$v_{AB} = \frac{r_A v_A + r_B v_B}{r_A + r_B}$$

where v_A and v_B are the reactant brightness values and v_{AB} is the product brightness value. Finally, Smoldyn determines surface interactions for products using the method that the new species behaves like the reactant that has the "greater action," where the possible actions are ordered with increasing value as transmission, reflection, absorption, and porting (which is for hybrid simulations [33]). For example, if a surface reflects reactant A and transmits reactant B, then reflection is the greater action, so the surface reflects product AB.

3.6 Symmetric Species

Reaction networks that include structurally symmetric species often include multiple reactions that form the same products, which increases the effective reaction rate. Consider the A-B-B-A complex for example (*see* **Note 3**). It can lose an A monomer from either the left or the right sides, whereas the A-B-B complex can only lose an A monomer from the left side, so the former reaction should proceed twice as fast (assuming that all of these A-B bonds are chemically identical). Smoldyn accounts for this by watching for repeated reactions as it expands reaction patterns, and incrementing the associated *reaction multiplicity* when they arise. Smoldyn multiplies the reaction multiplicity by the requested reaction rate constant to compute the total reaction rate constant.

An exception arises to this multiplicity computation if the reaction rule for a bimolecular reaction can match to both possible orderings of a single pair of reactants. For example, the rule * + * → ** can match to the two reactants A and AA as either A + AA or AA + A (*see* **Note 5**). Because these two possible reaction orderings typically reflect two different chemical bonds being formed, Smoldyn only considers one of the two orderings (the one in which the reactant's internal indices are in increasing order). These computations are designed to yield the results that one would normally expect but are nevertheless complicated when used with symmetric complexes, so it is worth checking that the simulation parameters are as desired.

4 Notes

The following notes illustrate the use of wildcards for rule-based modeling using several example problems. These models, and additional files that I used for their analysis, are available in the Smoldyn download package in the subdirectory examples/S94_archive/Andrews_2019. Further information about the models is also available in this chapter's Supplementary Materials (available at the publisher's website, the Smoldyn website, and the bioRxiv preprint server).

1. *Simple reaction networks with low symmetry.* Reaction networks that are conceptually simple and have low symmetry are typically easy to define using wildcards. This is illustrated with an example of second messenger signaling, where extracellular "first messengers" bind to cell receptors, which then release intracellular "second messengers" [34, 35]. Figure 1a shows a simple model in which a transmembrane receptor (R) can bind an extracellular ligand (L) and/or an intracellular messenger protein (M); a messenger that is bound to a ligand-bound receptor gets phosphorylated (Mp), and phosphorylated messengers lose their phosphates spontaneously (such as from unmodeled phosphatases). The network, which comprises nine species and ten reactions (Fig. 1b), can be expressed with the following four rules using wildcards:

```
rxnlr      L(fsoln) + R*(up) <-> LR*(up)    krl_on krl_off
rxnrm      *R(up) + M*(bsoln) <-> *RM*(up)  krm_on krm_off
rxnphos    LRM(up) -> LRMp(up)              k_phos
rxnunphos  Mp(soln) -> M(soln)              k_unphos
```

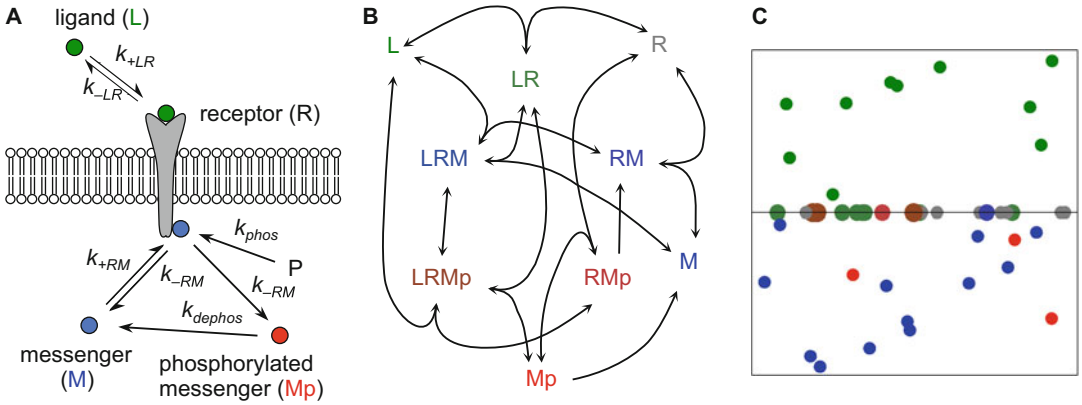

Fig. 1 Model of second messenger signaling. (**a**) Cartoon of the model, showing the components and their interactions. (**b**) The complete reaction network, where species are shown with the same colors as those generated by Smoldyn. (**c**) Snapshot of this model simulated in Smoldyn, again using the same color scheme. The line across the middle represents the membrane, the region above the line is the extracellular region, and the region below the line is the cytoplasm

Each line shows the rule name, the reaction rule, and the reaction rate constants. Note that the use of wildcards, which in this case is just the "*" character, enabled each rule to represent a separate process in a clear manner. Also note that the reactant and product states (the spatial localizations given within parentheses) are straightforward to define and reasonably intuitive. Smoldyn uses them to correctly place all receptor complexes at the membrane, ligands in the extracellular space, and messengers in the cytosol (Fig. 1c).

2. *More complicated networks with low symmetry.* Figure 2a shows a slightly more complicated example, but one that still includes asymmetric complexes. It shows a model of transposon excision that was developed to answer the question of how DNA transposons regulate their copy numbers so that they do not overproduce themselves and then kill their hosts [36] (transposons are mobile sections of DNA that can be amplified as they move from one location in the genome to another). In the model, the A-B species is a transposon with ends "A" and "B", and T_2 is a transposase dimer, an enzyme that binds to and cuts transposon ends. The transposase can be nonspecifically bound to DNA ($T_{2,nsb}$) or freely diffusing in the nucleus (T_2). At low transposase concentrations: a T_2 binds to a transposon end to form a singly bound transposon (T_2A-B or A-BT_2), this DNA forms a loop, the same T_2 binds to the other transposon end (AT_2B), and the transposase cuts out the transposon (the reaction with rate k_3). In the model, the transposition products conserve the reactant amounts and create an X molecule as a transposition counter although, in actuality, transpositions can produce additional transposons, amplifying the transposon in the genome. At high T_2 concentrations: transposases bind to singly bound transposons to create doubly bound transposons

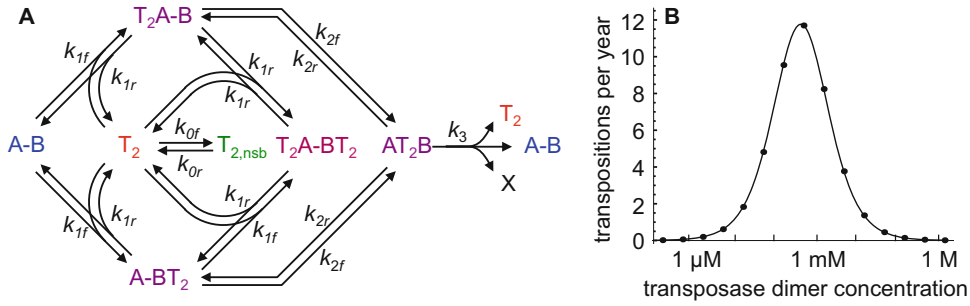

Fig. 2 Model of transposase dynamics modified from Ref. [36]. (**a**) Reaction network, where A-B is a transposon and T_2 is a transposase dimer. Colors are those generated by Smoldyn. (**b**) Transposition rate for a single transposon as a function of the total transposase dimer concentration within the nucleus. Points represent simulation data generated with ordinary differential equations and simulated in Mathematica and the line represents the analytical theory for the transposition rate, described in the main text

(T_2A-BT_2). These cannot undergo transposition, thereby regulating the process. This model can be expressed using wildcards as

```
rxnT2nsb      T2 <-> T2nsb                              k0f k0r
rxnABbind     A-B*|*A-B + T2 <-> T2A-B*|*A-BT2          k1f k1r
rxnassemble   T2&A-B <-> AT2B                           k2f k2r
rxnexcise     AT2B -> A-B + T2 + X                      k3
```

The OR operators in `rxnABbind` indicate that T_2 can bind to either the left of A-B* (A-B or A-BT_2) or the right of *A-B (A-B or T_2A-B). The permutation operator in `rxnassemble` indicates that both T_2A-B and A-BT_2 react to form AT_2B.

Expanding these reaction rules with Smoldyn produced the reaction network shown in Fig. 2a, as anticipated. The physiological rate constants [36] vary extremely widely (e.g., $k_{0f} = 10^5\ \mathrm{s}^{-1}$ and $k_{2f} = 4.3 \times 10^{-4}\ \mathrm{s}^{-1}$), meaning that Smoldyn would have to use short time steps to resolve the fast reactions but also run for a very long time to observe the slow reactions, so I simulated these reactions deterministically using Mathematica instead. Figure 2b compares the simulated transposition rates with steady-state values derived from analytical theory, showing excellent agreement.

I derived the analytical result shown in Fig. 2b in several steps while using the parameter values presented in Ref. [36]. First, the equilibrium constant of reaction 0 showed that most transposase is nonspecifically bound rather than freely diffusing. Next, I found that the equilibrium concentrations of the unbound, singly bound, and doubly bound transposons are

$$\frac{[\text{A-B}]}{[\text{A-B}_{\text{sum}}]} = \frac{1}{1 + 2K[\text{T}_{2,\text{tot.}}] + K^2[\text{T}_{2,\text{tot.}}]^2}$$

$$\frac{[\text{T}_2\text{A-B}]}{[\text{A-B}_{\text{sum}}]} = \frac{[\text{A-BT}_2]}{[\text{A-B}_{\text{sum}}]} = \frac{K[\text{T}_{2,\text{tot.}}]}{1 + 2K[\text{T}_{2,\text{tot.}}] + K^2[\text{T}_{2,\text{tot.}}]^2}$$

$$\frac{[\text{T}_2\text{A-BT}_2]}{[\text{A-B}_{\text{sum}}]} = \frac{K^2[\text{T}_{2,\text{tot.}}]^2}{1 + 2K[\text{T}_{2,\text{tot.}}] + K^2[\text{T}_{2,\text{tot.}}]^2}$$

where $[\text{A-B}_{\text{sum}}]$ is the sum of these three transposon concentrations (which does not include AT_2B) and K is the transposon association constant, which are defined as

$$[\text{A-B}_{\text{sum}}] = [\text{A-B}] + [\text{T}_2\text{A-B}] + [\text{A-BT}_2] + [\text{T}_2\text{A-BT}_2]$$

$$K = \frac{k_{0r}k_{1f}}{(k_{0r} + k_{0f})k_{1r}}.$$

The reverse reaction rate constant k_{2r} is much smaller than the transposition rate constant, k_3, allowing it to be ignored. As a result, the steady-state transposition rate is

$$\phi = k_{2f}[\text{T}_2\text{A-B}] + k_{2f}[\text{A-BT}_2].$$

Substituting in for the singly bound transposon concentrations and then addressing the fact that $[\text{A-B}_{\text{sum}}]$ does not include the concentration of AT_2B lead to the final result:

$$\phi = \frac{2k_{2f}K\left[\text{T}_{2,\text{tot.}}\right][\text{A-B}_{\text{tot.}}]}{1 + 2K\left[\text{T}_{2,\text{tot.}}\right] + K^2\left[\text{T}_{2,\text{tot.}}\right]^2} \times \left[\frac{2k_{2f}K\left[\text{T}_{2,\text{tot.}}\right]}{(2k_{2r} + k_3)\left(1 + 2K\left[\text{T}_{2,\text{tot.}}\right] + K^2\left[\text{T}_{2,\text{tot.}}\right]^2\right)} + 1\right]^{-1}$$

This is the steady-state transposition rate, shown in Fig. 2b with a solid line. The former term represents the dominant effect of transposon regulation, showing that transposition is slow with low and high transposase concentrations but fast with intermediate transposase concentrations. The latter term in this equation is much less important and can be ignored if $k_3 >> k_{2f}$; however, it is included here because experimental evidence suggests that k_3 is actually only about three times larger than k_{2f} [36]. *See* the Supplementary Materials for details.

3. *Symmetric complexes, modeled with asymmetric notation*. Reaction networks that include structurally symmetric protein complexes, such as dimers and higher oligomers [37], generally require a little more care. In particular, it is often the case that a single complex can be represented correctly in multiple ways, leading to the question of whether the model notation should just include one of the ways, or all of them. Which approach is simplest depends on the specific problem; this note shows an example of the former approach, in which each complex is represented in just one way.

   Figure 3a shows a simple model of reversible dimer assembly for a symmetric complex that has the form A-B-B-A, a form that is loosely based upon receptor tyrosine kinases such as the epidermal growth factor and insulin receptors [38]. The model includes the monomers A and B, dimers AB and BB, trimer

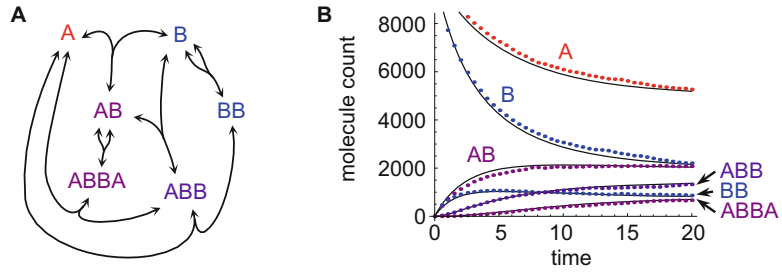

Fig. 3 Model of symmetric complexation using asymmetric notation. (**a**) Reaction network for binding between A and B components that can assemble into the A-B-B-A complex. (**b**) Black lines show reaction kinetics computed from manual reaction network expansion and simulated with ordinary differential equations using Mathematica; colored points show reaction kinetics from Smoldyn's expansion of wildcard rules and then simulation. Colors in both panels are those generated by Smoldyn. Simulation parameters: `AB_ON` = 10, `AB_OFF` = 0.05, `BB_ON` = 8, `BB_OFF` = 0.03, 10,000 initial A molecules, 10,000 initial B molecules, volume of 100^3, time from 0 to 20 with steps, in Smoldyn simulation, of 0.05

ABB, and tetramer ABBA. The notation is asymmetric in that it includes the species AB but not the species BA, which would be chemically identical. Similarly, it includes ABB but not BBA. It can be expressed with the reaction rules:

```
rxnABon     A + B|BB|BB|ABB -> AB|ABB|ABB|ABBA     AB_ON
rxnABoff    AB|ABB|ABBA|ABBA -> A + B|BB|ABB|ABB   AB_OFF
rxnBBon1    B + B|AB -> BB|ABB                     BB_ON
rxnBBon2    AB + AB -> ABBA                        BB_ON
rxnBBoff    *BB* -> *B + *B                        BB_OFF
```

These rules make heavy use of the OR operator. For example, the first reaction rule shows that A can bind to any of B, BB, BB, or ABB, and the products are, respectively, AB, ABB, ABB, and ABBA. The repeated BB reactants in this rule reflect the fact that A can bind to either the left or the right side of BB so the rate constant for this reaction should be twice the listed value (`AB_ON`). Similarly, in the second reaction rule, ABBA dissociates twice to A + ABB to reflect the two A-B bonds in ABBA. These rules are somewhat inelegant in that they do not reflect the symmetry of the system, include strings of OR operators, and only include irreversible reactions despite the fact that the model reactions are reversible. This inelegance arises from the decision to use asymmetric notation and from limitations in the wildcard approach. Nevertheless, these rules are substantially simpler than the full list of 12 reactions.

The reaction network that Smoldyn computed from these rules was identical to ones that arose from BioNetGen and manual expansion [22], validating the rule approach. Figure 3b shows that a Smoldyn simulation that was defined with these rules agreed well with a deterministic simulation of the same network, computed using ordinary differential equations.

4. *Symmetric complexes, modeled with symmetric notation.* This note continues on the topic of symmetric complexes, but now using symmetric notation. In this case, complexes that can be represented correctly in multiple ways are not represented with just one of the possibilities, but with all of them. This increases network complexity due to the greater number of species and reactions, but can simplify the reaction rules through preservation of the network symmetry.

   E. coli bacteria have several mechanisms for locating their cell division plane at the cell center, one of which is to prevent division elsewhere with the Min system [39, 40]. In this system, the combined actions of the MinD and MinE proteins create a spatiotemporal oscillation between the cell poles that keeps the co-localized MinC away from the cell center; MinC inhibits division apparatus formation, thus inhibiting cell division away from the cell center. This system has been modeled extensively [41, 42] but few models explicitly represent MinD or MinE dimerization [43], despite the fact that both have dissociation constants that are comparable to their intracellular concentrations [44, 45]. Interestingly, MinD only dimerizes when bound to ATP [46] and MinD only hydrolyzes ATP when it is dimeric [47].

   Figure 4a shows a model of MinD nucleotide binding and dimerization. All species are MinD proteins, but bound to different cofactors: "T" represents MinD bound to ATP, "D" represents MinD bound to ADP, and "A" represents MinD bound to neither ("A" stands for apo). Pairs of these symbols, such as "TT", represent dimers. Three of the dimers are heterodimers that the model represents using both possible orderings, such as DT and TD. The model can be described with the following rules, of which the first three represent nucleotide substitution, and subsequent ones represent dimerization, dimer dissociation, and ATP hydrolysis:

```
rxnAtoD    *A* <-> *D*    KATOD KDTOA
rxnAtoT    *A* <-> *T*    KATOT KTTOA
rxnDtoT    *D* <-> *T*    KDTOT KTTOD
rxndimer   T + T -> TT    KDIMER
rxndissoc   ?? -> ? + ?   KDISS
rxnhydro    ?&T -> ?&D    KHYDRO
```

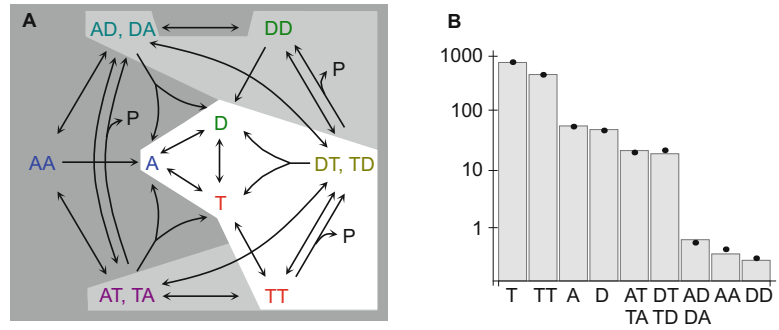

Fig. 4 Model of *E. coli* MinD dimerization and nucleotide binding. (**a**) Reaction network. Background shading illustrates on-the-fly simulation for a simulation in which A, D, T, TT, and either DT or TD have arisen. White regions are explored, light gray are generated but not explored, and dark gray are not generated; see the main text. (**b**) Species abundance in a single cell at steady state using physiologically reasonable parameter estimates. Bars are deterministic values computed by simulating the network in Mathematica using ordinary differential equations and points are averages of Smoldyn simulation values ($n = 20$)

Maintaining the reaction network symmetry in the model notation enabled simple and elegant reaction rules in this case. Note the use of the "?" wildcard: `rxndissoc` uses it to indicate that all dimers dissociate with the same rate constant and `rxnhydro` uses it to represent any monomer within a MinD dimer. Also, use of the permutation operator in the last rule shows that any dimer with a "T" in it, regardless of whether the "T" is the first or second symbol, is able to perform hydrolysis.

Figure 4a illustrates on-the-fly network generation for this model using background shading. It depicts the situation in which the only species that have arisen in the simulation so far are A, D, T, TT, and DT and/or TD. They are over a white background to show that this region of the network has been explored. Species and reactions in the adjacent light gray regions have been generated by Smoldyn so that they could be used, but they have not actually arisen in the simulation so far. Species and reactions in the dark gray regions have not been generated yet (and may not require generation), which saves computation and computer memory.

Figure 4b shows the number of molecules of each species at steady state, where the bars are from a deterministic simulation in Mathematica and the points are average values from a Smoldyn simulation. It shows that most of the MinD is bound to ATP and is either monomeric or dimeric. These results were computed from physiologically reasonable parameters for a

single cell but do not account for membrane or MinE interactions.

The parameters are described in the Supplementary Materials and summarized here. The cell volume was set to 1 fl, from Refs. [48, 49], cells contained 2000 MinD protein copies [50], and the ATP concentration was 1.54 mM, which came out to 930,000 molecules [48, 51]. The ADP concentration was set eightfold lower than the ATP concentration [48, 52], leading to 0.68 mM ADP, which was 116,000 molecules. From the 29.6 kDa molecular weight of MinD [53], I estimated its diffusion coefficient as 2.6 $\mu m^2/s$ [28], in good agreement with the value used in a prior model [54]. Nucleotide exchange was shown to occur for MinD [55], for which I assumed that exchange of ADP with ATP had a rate constant of 1 s^{-1}, from the same model [54]. Combining this with the ATP concentration yielded `KDTOT` as 650 M^{-1} s^{-1}, which is 1.1×10^{-6} μm^3 s^{-1}. ATP competes three times more effectively than ADP for binding to MinD [56], so I set `KTTOD` to be 1/3 of `KDTOT`, making it 0.37×10^{-6} μm^3 s^{-1}. I assumed the same values for nucleotide gain, so `KATOT` was 1.1×10^{-6} μm^3 s^{-1} and `KATOD` was 0.37×10^{-6} μm^3 s^{-1}. I also assumed that 20 times more MinD is bound to ATP than unbound, which combined with prior numbers to give both `KTTOA` and `KDTOA` as 0.05 s^{-1}. MinD is predominantly dimeric when over 2 μM and monomeric at lower concentrations [45], so I assumed a 2 μM dissociation constant. Further assuming a dissociation reaction rate constant (`KDISS`) of 1 s^{-1} led to the dimerization rate constant (`KDIMER`) of 5×10^5 M^{-1} s^{-1}, which is 8.5×10^{-4} μm^3 s^{-1}. Finally, MinD hydrolyzes ATP with a maximum rate of 2.5 nmoles of ATP per mg of protein per minute [57] which converts to a `KHYDRO` value of 1.2×10^{-3} s^{-1}.

5. *Polymerization with identical monomers.* Cellular polymers include (1) microtubules and actin, which are important for cell structure and intracellular transport; (2) intermediate filaments, which provide mechanical strength; (3) DNA and RNA, which encode genetic information; (4) polysaccharides, which provide structure and store energy; and sometimes (5) amyloid fibrils, which can cause neurodegenerative diseases [58–60]. Most of these polymers assemble at one or both ends, although some can also anneal, meaning that two polymers join end to end.

   Figure 5a shows a polymer model that assembles and disassembles at one end (the model is called "polymer_end1"). It can be expressed with the reaction rule

```
rxnend    * + A <-> *A    KF KR
```

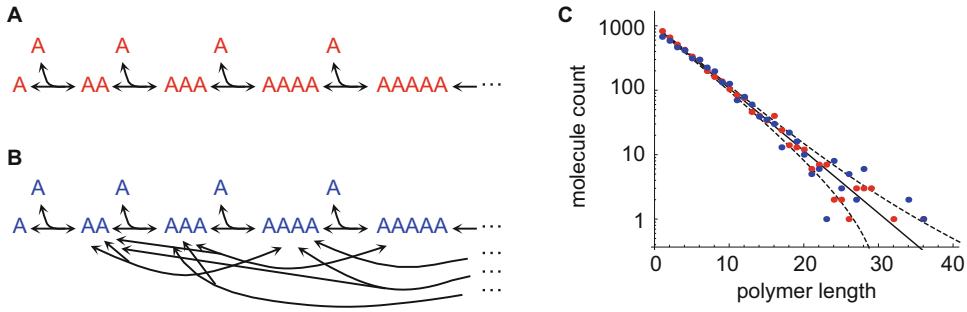

Fig. 5 Models of polymerization. (**a**) Reaction network for polymers that can add or lose units from a single end. (**b**) Part of a reaction network for polymers that can add or lose units from their ends, and can also break and anneal. (**c**) Equilibrium length distributions of polymers from a simulation of the end-polymerization model ("polymer_end1" model, red points), a simulation of the breaking and annealing model ("polymer_mid" model, blue points), and analytical theory (solid black line). Dashed lines show the theoretical standard deviations

where "A" is a single polymer unit and `KF` and `KR` are the forward and reverse reaction rate constants. The isolated asterisk is adequate in this rule because this model does not include other species, but would create unintended reactions otherwise. A simulation that started with 20,000 monomers and used on-the-fly expansion resulted in an exponential polymer length distribution at equilibrium, shown with red dots in Fig. 5c. This agreed with theory [61] (black lines in Fig. 5c) which is described in the Supplementary Materials and summarized below. On completion, this model had 40 species and 77 reactions.

Some limitations of the wildcard method were interesting. (1) This simulation represented polymer lengths by listing their units rather than with numbers (e.g., "AAA" rather than "A3"), so polymers were limited to 256 units because that is the longest species name that Smoldyn allows. (2) Smoldyn represents these polymers as spheres rather than as extended filaments; this is clearly inaccurate for stiff polymers, although actually reasonably accurate for highly flexible polymers which tend to collapse into loose clusters [61]. In the latter case, Smoldyn computed polymer radii as increasing as $L^{1/3}$, where L is the polymer length due to its default method for computing properties for new species (Subheading 3.5, above), whereas the ideal scaling for freely jointed chains is $L^{1/2}$ [60, 62]. And (3) Smoldyn computed the polymer diffusion coefficients as decreasing as $L^{-1/3}$, as compared to the $L^{-0.6}$ scaling that is typically observed experimentally for polymers [60].

A more serious flaw with this model is that Smoldyn assigns the same reaction rate to all association reactions. This is the correct behavior for the given reaction rule, but does not account for the fact that the A + A → AA reaction can happen in either of the two ways: either of the two reactant monomers can end up at the "left" end of the product. The following reaction rules (model "polymer_end2") fix this flaw:

```
rxn1    A + A <-> AA        2*KF KR
rxn2    *AA + A <-> *AAA    KF KR
```

Here, monomer association proceeds twice as fast as association of higher polymers. Results from these latter rules agreed with a comparable model written in BNGL [22]. At equilibrium, they also showed an exponential length distribution for all polymers with more than one monomer (*see* Supplementary Materials).

Figure 5b shows a model in which polymers can also anneal and break (model "polymer_mid"). It can be expressed with the reaction rule

```
rxnmid    * + * <-> **    2*KF KR
```

As above, the association reaction rate was doubled to account for the fact that either of the two reactants can end up on the "left" side of the product. This follows from the fact that Smoldyn only considers a single ordering for any particular pair of reactants; for example, it generates the reaction A + AA → AAA but not also AA + A → AAA. This model reached equilibrium much faster than the former ones but produced essentially the same exponential length distribution as the "polymer_end1" model (blue dots in Fig. 5c). This model led to a much larger reaction network, with 151 species and 4037 reactions, because each species can participate in many more reactions.

To derive the theoretical length distribution for the polymer_end1 model, define K_a as the association constant:

$$K_a = \frac{k_f}{k_r}.$$

It is the equilibrium constant for each of the association reactions, so

$$K_a = \frac{[\mathrm{A}_2]}{[\mathrm{A}][\mathrm{A}]} = \frac{[\mathrm{A}_3]}{[\mathrm{A}_2][\mathrm{A}]} = \cdots = \frac{[\mathrm{A}_n]}{[\mathrm{A}_{n-1}][\mathrm{A}]}$$

where A_n represents an n-mer. Rearrangement leads to

$$[A_n] = K_a^{n-1}[A]^n$$

for all n equal to 2 or larger. This equation shows the exponential length distribution at equilibrium, in which long polymers are less abundant than short polymers. To solve for the monomer concentration in this equation, we use the fact that the simulations conserved the total subunit concentration:

$$[A_{tot.}] = [A] + 2[A_2] + \cdots + n[A_n] + \cdots.$$

Substituting in the above solution for $[A_n]$, summing the infinite series, and then solving for [A] yield the monomer concentration

$$[A] = [A_{tot.}]\frac{1 + 2K_a[A_{tot.}] - \sqrt{1 + 4K_a[A_{tot.}]}}{2K_a^2[A_{tot.}]^2}.$$

Finally, these are equilibrium concentrations, so the standard deviations of the populations can be well approximated as the square roots of the mean populations. Derivations for the theoretical length distributions for the polymer_end2 and polymer_mid models are similar and described in the Supplementary Materials.

6. *Polymer sequences and chemical structures.* The pattern-matching aspects of the wildcard method enable it to be used to define reactions that are specific to individual polymer sequences and chemical structures.

   The central dogma of molecular biology is that cells transcribe DNA to mRNA and then translate mRNA to protein [58]. Figure 6a shows that this process can be modeled using wildcards if sequences are reasonably short. The reaction rule

```
rxnTransc    Dna* -> Dna$1 + Rna$1    KTRANSC
```

   performs transcription, where "Dna" and "Rna" are prefixes that indicate the sequence type and the "$1" portions of the products show that the same text gets substituted into each one. Ideally, this rule would not only preserve the sequence, which it does, but also replace all T symbols, for DNA thymine bases, with U symbols, for RNA uracil bases. However, there is no easy way to do this with the wildcard method as it is currently designed. A wildcard approach that used regular expressions, which are more sophisticated pattern-matching approaches, would solve this problem but would also be more difficult to use. The following reaction rules perform translation by modeling ribosome ("Rib") binding to the beginning

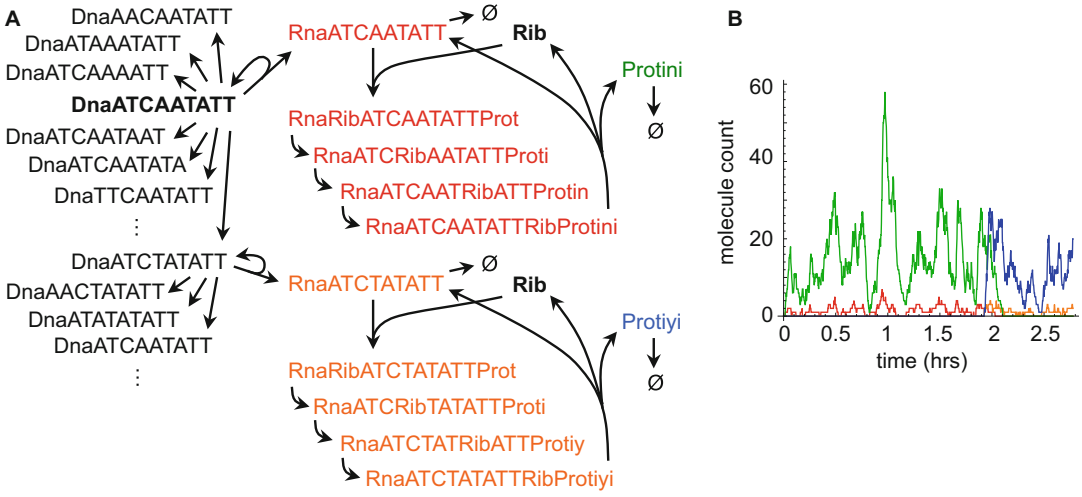

Fig. 6 Model of DNA transcription and then RNA translation. (**a**) Diagram of the model, showing some of the species that arose during a particular simulation run that used on-the-fly network expansion. The starting DNA sequence is shown in black and bold face. It could mutate to other sequences, shown above and below in black. It was also transcribed to RNA, shown in red, and these RNAs were translated one codon at a time to produce the polypeptide Protini, shown in green. Mutation actually happened at simulation time of about 1.9 h, at which point the new DNA was transcribed to the RNA shown in orange and it was translated to the protein shown in blue. (**b**) Copy numbers of the RNA and protein molecules from the same simulation, using the same colors as panel A

of an mRNA sequence, translation of each codon, and finally dissociation of RNA, ribosome, and protein ("Prot" prefix):

```
rxnRibBind     Rna*[A,T,C,G] + Rib -> RnaRib*[A,T,C,G]Prot     KTRANSL
rxnTranslI     Rna*RibAT[T,C,A]* -> Rna*AT[T,C,A]Rib*i         KTRANSL
rxnTranslN     Rna*RibAA[T,C]* -> Rna*AA[T,C]Rib*n              KTRANSL
... 17 more amino acids ...
rxnRibUnbind   Rna*RibProt* -> Rna* + Rib + Prot*              KTRANSL
```

In `rxnTranslI` reaction rule, for example, any of the RNA codons ATT, ATC, and ATA (using T instead of U) code for isoleucine, so the product shows that the ribosome moves forward by three base pairs and an "i", for isoleucine, is appended to the growing protein. Three final reaction rules encode for DNA mutations and RNA and protein degradation, respectively:

```
rxnMut        Dna*?* -> Dna*{A|T|C|G}*     KMUT
rxnRnaDeg     Rna*[A,T,C,G] -> 0            KRNADEG
rxnProtDeg    Prot* -> 0                    KPROTDEG
```

Figure 6b shows results from a simulation of this model that started with one DNA molecule, `DnaATCAATATT`. Initially, it

was transcribed to `RnaATCAATATT` and then translated over multiple steps to `Protini` (isoleucine-asparagine-isoleucine). At simulation time of about 1.9 h, the DNA mutated, leading to a slightly different RNA sequence and production of protein iyi. The protein molecule counts show a large variation because they amplify the RNA counts, which have high variation due to their low copy numbers [63].

Wildcards can also be used to define reactions based on chemical structures that are not sequence data. In particular, they are useful in conjunction with the SMILES notation [64], a scheme that allows most chemical complexes to be uniquely expressed using a single line of normal text characters (e.g., ethanol, CH_3CH_2OH is CCO in SMILES notation). As an example, the *E. coli* lipid synthesis pathway includes several enzymes that act repeatedly on lipids, adding a two-carbon group with each repetition [65]. Each enzyme is specific to a particular chemical functional group but has low specificity with regard to the lipid chain length. This can be represented using wildcards starting with the ten-carbon lipid cis-3-decenoyl-ACP, written in SMILES notation as [ACP]C(=O)C/C=C\CCCCCC. Here, ACP is an abbreviation for acyl carrier protein, the C(=O) portion represents a carbonyl group, the /C=C\ portion represents a cis-conformation double bond, and the CCCCCC portion represents a saturated hydrocarbon tail. The reaction rules are

```
FabB + ACP-C(=O)C{CC|/C=C\}* -> FabB + ACP-C(=O)CC(=O)C{CC|/C=C\}*
FabG + ACP-C(=O)CC(=O)C* -> FabG + ACP-C(=O)CC(O)C*
FabZ + ACP-C(=O)CC(O)C* -> FabZ + ACP-C(=O)C/C=C/*
FabI + ACP-C(=O)C/C=C/* -> FabI + ACP-C(=O)CCC*
```

In the first reaction, FabB adds a carbonyl and extra carbon, C(=O)C, to the chain. Next, FabG reduces the newly added carbonyl to a hydroxyl, C(O); FabZ reduces the hydroxyl to a trans-conformation carbon-carbon double bond, /C=C/; and then FabI reduces the double bond to a single bond, CC. The end result is that the cis-3-decenoyl-ACP gets lengthened by two carbons to cis-3-dodecenoyl-ACP. Application of these rules to this longer lipid adds yet more carbons.

Both the nucleic acid sequence model and this lipid synthesis model would undoubtedly be simpler and more generalizable if they were developed using software designed specifically for the tasks. However, the fact that they can be developed using wildcards shows the method's versatility.

Acknowledgments

I thank Ronnie Chalmers, Akintunde Emiola, Jim Faeder, and Karen Lipkow for useful discussions. Much of this work was carried out during a visit to the Isaac Newton Institute for Mathematical Sciences, for which I thank Radek Erban, David Holcman, Sam Isaacson, and Konstantinos Zygalakis, who were the program organizers, and the institute staff. I also thank Roger Brent, Erick Matsen, and Harlan Robbins for providing space for me at the FHCRC, where the work was completed. This work was supported by a Simons Foundation grant awarded to SSA and by EPSRC grant EP/K032208/1 awarded to the Isaac Newton Institute.

References

1. Boyle R (1661) The Sceptical Chymist. F. Cadwell, London
2. Waage P, Guldberg CM (1864) Studier over affiniteten. Forhandlinger: Videnskabs-Selskabet i Christiania, p 35–40
3. Michaelis L, Menten ML (1913) Die kinetik der invertinwirkung. Biochem Z 49:333–369
4. Gibbs JW (1875–1878) On the equilibrium of heterogeneous substances. In: Transactions of the Connecticut Academy, vol 3. The Academy, New Haven, pp 108–248, 343–524
5. Jaynes ET (1992) The Gibbs paradox. In: Smith CR, Erickson GJ, Neudorfer PO (eds) Maximum Entropy and Bayesian Methods. Kluwer Academic Publishers, Dordrecht, pp 1–22
6. Alves R, Antunes F, Salvador A (2006) Tools for kinetic modeling of biochemical networks. Nat Biotechnol 24:667–672
7. Andrews SS, Dinh T, Arkin AP (2009) Stochastic models of biological processes. In: Meyers RA (ed) Encyclopedia of complexity and system science, vol 9. Springer, New York, pp 8730–8749
8. Bray D, Lay S (1997) Computer-based analysis of the binding steps in protein complex formation. Proc Natl Acad Sci U S A 94:13493–13498
9. Goldman J, Andrews SS, Bray D (2004) Size and composition of membrane protein clusters predicted by Monte Carlo analysis. Eur Biophys J 33:506–512
10. Sneddon MW, Faeder JR, Emonet T (2011) Efficient modeling, simulation and coarse-graining of biological complexity with NFsim. Nat Methods 8:177–183
11. Gruenert G, Ibrahim B, Lenser T, Lohel M, Hinze T, Dittrich P (2010) Rule-based spatial modeling with diffusing, geometrically constrained molecules. BMC Bioinformatics 11:307
12. Schöneberg J, Noé F (2013) ReaDDy – a software for particle-based reaction-diffusion dynamics in crowded cellular environments. PLoS One 8:e74261
13. Morton-Firth CJ, Bray D (1998) Predicting temporal fluctuations in an intracellular signalling pathway. J Theor Biol 192:117–128
14. 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, ACM, pp 161–70
15. Stefan MI, Bartol TM, Sejnowski TJ, Kennedy MB (2014) Multi-state modeling of biomolecules. PLoS Comp Biol 10:e1003844
16. Andrews SS, Addy NJ, Brent R, Arkin AP (2010) Detailed simulation of cell biology with Smoldyn 2.1. PLoS Comp Biol 6: e1000705
17. Tolle DP, Le Novère N (2010) *Meredys*, a multi-compartment reaction-diffusion simulator using multistate realistic molecular complexes. BMC Systems Biol 4:24
18. Hlavacek WS, Faeder JR, Blinov ML, Posner RG, Hucka M, Fontana W (2006) Rules for modeling signal-transduction systems. Sci STKE 2006:re6
19. 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 20:3289–3291
20. Lok L, Brent R (2005) Automatic generation of cellular reaction networks with Moleculizer 1.0. Nat Biotech 23:131–136

21. Danos V, Feret J, Fontana W, Harmer R, Hayman J, Krivine J, Thompson-Walsh C, Winskel G (2012) Graphs, rewriting, and pathway reconstruction for rule-based models. In: LIPIcs-Leibniz International Proceedings in Informatics, Schloss Dagstuhl-Leibniz-Zentrum fuor Informatik, vol 18. Dagstuhl Publishing, Germany
22. Andrews SS (2017) Smoldyn: particle-based simulation with rule-based modeling, improved molecular interaction and a library interface. Bioinformatics 33:710–717
23. Andrews SS, Bray D (2004) Stochastic simulation of chemical reactions with spatial resolution and single molecule detail. Phys Biol 1:137–151
24. Andrews SS (2009) Accurate particle-based simulation of adsorption, desorption, and partial transmission. Phys Biol 6:46015
25. Andrews SS (2018) Particle-based stochastic simulators. In: Jaeger D, Jung R (eds) Encyclopedia of computational neuroscience. Springer, New York
26. Faeder JR, Blinov ML, Hlavacek WS (2009) Rule-based modeling of biochemical systems with BioNetGen. Methods Mol Biol 500:113–167
27. Danos V, Laneve C (2004) Formal molecular biology. Theor Comput Sci 325:69–110
28. Andrews SS (2012) Spatial and stochastic cellular modeling with the Smoldyn simulator. Methods Mol Biol 804:519–542
29. Bardwell L (2005) A walk-through of the yeast mating pheromone response pathway. Peptides 26:339–350
30. Blinov ML, Faeder JR, Yang J, Goldstein B, Hlavacek WS (2005) 'On-the-fly' or 'generate-first' modeling? Nat Biotechnol 23:1344–1345
31. Suderman R, Deeds EJ (2013) Machines vs. ensembles: effective MAPK signaling through heterogeneous sets of protein complexes. PLoS Comp Biol 9:e1003278
32. Dix JA, Verkman AS (2008) Crowding effects on diffusion in solutions and cells. Annu Rev Biophys 37:247–263
33. Robinson M, Andrews SS, Erban R (2015) Multiscale reaction-diffusion simulations with Smoldyn. Bioinformatics 31:2406–2408
34. Alberts B, Johnson A, Lewis J, Raff M, Robers K, Walter P (2008) Molecular biology of the cell. Garland Science, New York
35. Sutherland EW, Oye I, Butcher RW (1964) The action of epinephrine and the role of the adenyl cyclase system in hormone action. Recent Prog Horm Res 21:623–646
36. Claeys Bouuaert C, Lipkow K, Andrews SS, Liu D, Chalmers R (2013) The autoregulation of a eukaryotic DNA transposon. elife 2: e00668
37. Marianayagam NJ, Sunde M, Matthews JM (2004) The power of two: protein dimerization in biology. Trends Biochem Sci 29:618–625
38. Hubbard SR, Till JH (2000) Protein tyrosine kinase structure and function. Annu Rev Biochem 69:373–398
39. Lutkenhaus J (2007) Assembly and dynamics of the bacterial MinCDE system and spatial regulation of the Z ring. Annu Rev Biochem 76:539–562
40. Loose M, Kruse K, Schwille P (2011) Protein self-organization: lessons from the Min system. Annu Rev Biophys 40:315–336
41. Howard M, Kruse K (2005) Cellular organization by self-organization: mechanisms and models for Min protein dynamics. J Cell Biol 168:533–536
42. Kruse K, Howard M, Margolin W (2007) An experimentalist's guide to computational modelling of the Min system. Mol Microbiol 63:1279–1284
43. Cytrynbaum E, Marshall BDL (2007) A multistranded polymer model explains MinDE dynamics in *E. coli* cell division. Biophys J 93:1134–1150
44. Zhang Y, Rowland S, King G, Braswell E, Rothfield L (1998) The relationship between hetero-oligomer formation and function of the topological specificity domain of the *Escherichia coli* MinE protein. Mol Microbiol 30:265–273
45. Hu Z, Lutkenhaus J (2003) A conserved sequence at the C-terminus of MinD is required for binding to the membrane and targeting MinC to the septum. Mol Microbiol 47:345–355
46. Hu Z, Saez C, Lutkenhaus J (2003) Recruitment of MinC, an inhibitor of Z-ring formation, to the membrane in *Escherichia coli*: role of MinD and MinE. J Bact 185:196–203
47. Andrews SS, Moghaddam A, Groves JT (2006) Quantification of reaction rates in the *E. coli* Min system. American Chemical Society National Meeting, San Francisco, CA
48. Milo R, Jorgensen P, Moran U, Weber G, Springer M (2010) BioNumbers – the database of key numbers in molecular and cell biology. Nucleic Acids Res 38:D750–DD53
49. Neidhardt FC, Umbarger HE (1996) In: Neidhardt FC (ed) Chemical composition of *Escherichia coli in Escherichia coli* and *Salmonella*. ASM Press, Washington, DC

50. Shih Y-L, Fu X, King GF, Le T, Rothfield L (2002) Division site placement in *E. coli*: mutations that prevent formation of the MinE ring lead to loss of the normal midcell arrest of growth of polar MinD membrane domains. EMBO J 21:3347–3357
51. Yaginuma H, Kawai S, Tabata KV, Tomiyama K, Kakizuka A, Komatsuzaki T, Noji H, Imamura H (2014) Diversity in ATP concentrations in a single bacterial cell population revealed by quantitative single-cell imaging. Sci Rep 4:6522
52. Tran QH, Unden G (1998) Changes in the proton potential and the cellular energetics of *Escherichia coli* during growth by aerobic and anaerobic respiration or by fermentation. Eur J Biochem 251:538–543
53. de Boer PAJ, Crossley RE, Rothfield LI (1989) A division inhibitor and a topological specificity factor coded for by the minicell locus determine proper placement of the division septum in *E. coli*. Cell 56:641–649
54. Huang KC, Meir Y, Wingreen NS (2003) Dynamic structures in *Escherichia coli*: spontaneous formation of MinE rings and MinD polar zones. Proc Natl Acad Sci U S A 100:12724–12728
55. de Boer PAJ, Crossley RE, Hand AR, Rothfield LI (1991) The MinD protein is a membrane ATPase required for the correct placement of the *Escherichia coli* division site. EMBO J 10:4371–4380
56. Lackner LL, Raskin DM, de Boer PAJ (2003) ATP-dependent interactions between *Escherichia coli* Min proteins and the phospholipid membrane in vitro. J Bacteriol 185:735–749
57. Hu Z, Lutkenhaus J (2001) Topological regulation of cell division in *E. coli*. Spatiotemporal oscillation of MinD requires stimulation of its ATPase by MinE and phospholipid. Mol Cell 7:1337–1343
58. Alberts B, Bray D, Lewis J, Raff M, Roberts K, Watson JD (1994) Molecular biology of the cell. Garland Publishing, New York
59. Ross CA, Poirier MA (2005) What is the role of protein aggregation in neurodegeneration? Nat Rev Mol Cell Biol 6:891–898
60. Andrews SS (2014) Physical models and computational methods for modeling cytoskeletal and DNA filaments. Phys Biol 11:011001
61. Flory PJ (1953) Principles of polymer chemistry. Cornell University Press, Ithaca, NY
62. Doi M, Edwards SF (1986) The theory of polymer dynamics. Oxford University Press, Oxford
63. Berg OG (1978) A model for the statistical fluctuations of protein numbers in a microbial population. J Theor Biol 71:587–603
64. Weininger D (1988) SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. J Chem Inf Comput Sci 28:31–36
65. Emiola A, Andrews SS, Heller C, George J (2016) Crosstalk between the lipopolysaccharide and phospholipid pathways during outer membrane biogenesis in *Escherichia coli*. Proc Natl Acad Sci U S A 113:3108–3113