Rule-based Modeling Using Wildcards

Steven S. Andrews

1. Fred Hutchinson Cancer Research Center, Seattle, WA USA. 2. Seattle University, Seattle, WA USA. 3. Isaac Newton Institute, Cambridge, UK

Introduction

Many biological molecules exist in multimeric complexes or can be post-translationally modified. Representing all these variants as distinct chemical species leads to a combinatorial explosion of species and reactions. Rule-based modeling[1], in which software generates the reaction network from user-supplied rules alleviates this.

Rule-based modeling is typically performed with the formal BioNetGen or Kappa languages, but their formality makes them rigid and non-intuitive. I addressed this by developing a rule-based modeling approach that is based on wildcards that match to species names[2], much as wildcards can match to file names in computer operating systems. Use with several real-world problems showed the method to be flexible and intuitive[3].

I implemented rule-based modeling with wildcards in the Smoldyn software[4], a biochemical simulator that represents each molecule of interest as an individual particle. These particles, diffuse, react, and interact with surfaces much as real molecules do.

Wildcard matching

Species *names* in input files get replaced by species *patterns*, each of which can match to multiple individual species (forming a "species group"). Example:

`Fus3*` matches to: `Fus3`, `Fus3p`, and `Fus3pp`

Table of Wildcards

symbol	meaning	matching example	reaction example
?	any 1 character	A? matches AB, AC, etc.	A? ⟶ B?
*	0 or more char.	A* matches A, AB, etc.	A* ⟶ B*
[...]	1 listed char.	A[a-c] matches Aa, Ab, Ac	A[u,p] ⟶ B[0,1]
|	OR operator	A|B|C matches A, B, C	A|B ⟶ a|b
&	permutation	A&B matches AB, BA	A&B ⟶ a&b
{...}	grouping	A{B|C} matches AB, AC	A{b|c} ⟶ A{c|b}
$*n*	*n*'th match	not applicable	A?? ⟶ B$2$1

The ?, *, and [...] wildcards are "text-matching" wildcards. Patterns including these ("elementary patterns") can be easily checked against species names to detect matches. The |, &, and {...} wildcards are "structural" wildcards. Smoldyn expands patterns with structural wildcards into a list of elementary patterns and then detects matches with those.

Smoldyn maintains a list of species names that match each species pattern to prevent redundant text parsing.

Wildcard substituting

Wildcard substitution arises in reactions, where wildcards on the right side correspond to those on the left using the same order. Example:

`Ste5* + Fus3*` ⟶ `Ste5*-Fus3*`

Ordering can also be specified with the $*n* wildcard.

Wildcard substitutions can produce new species names. If the reaction is declared as a "reaction_rule", Smoldyn adds these new names to its list of species during rule expansion, which Smoldyn can perform before the simulation ("generate-first" approach) or as needed during the simulation ("on-the-fly" approach).

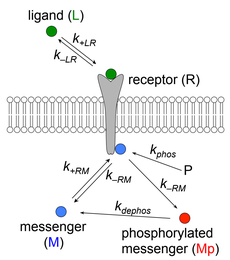
For the reaction rule A + B → AB, the properties of AB can be given with rules, or using Smoldyn's default approach in which the product radius, diffusion coefficient, and color are computed from reactant properties with:

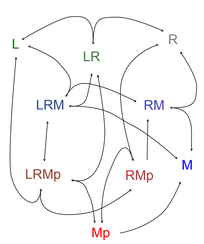
$$r_{AB} = \sqrt[3]{r_A^3 + r_B^3} \qquad D_{AB} = \left(D_A^{-3} + D_B^{-3}\right)^{-1/3} \qquad v_{AB} = \frac{r_A v_A + r_B v_B}{r_A + r_B}$$

Examples

Second messenger signaling

Extracellular “first messengers” bind to cell receptors, which release intracellular “second messengers.” Here, receptor (R) can bind ligand (L) and/or a messenger protein (M); a messenger that is bound to a ligand-bound receptor gets phosphorylated (Mp), and phosphorylated messengers lose phosphates spontaneously (e.g. by unmodeled phosphatases).





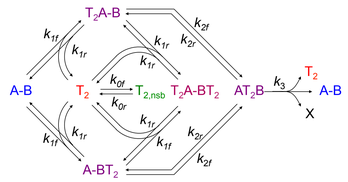
Wildcard rules

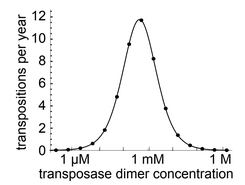
```
rxnlr      L(fsoln) + R*(up) <-> LR*(up)      krl_on krl_off
rxnrm      *R(up) + M*(bsoln) <-> *RM*(up)    krm_on krm_off
rxnphos    LRM(up) -> LRMp(up)                k_phos
rxnunphos  Mp(soln) -> M(soln)                k_unphos
```

This model has 3 proteins and 4 reaction rules, with one rule for each physical process. They expand to 9 species and 10 reactions.

Transposase dynamics

DNA transposons regulate their copy numbers to avoid killing their hosts by overproducing[5]. A-B is a transposon with ends ‘A’ and ‘B’ and T_2 is a transposase dimer, which binds and cuts transposon ends. T_2 can non-specifically bind DNA ($T_{2,nsb}$) or be free in the nucleus (T_2). At low T_2 concentration: T_2 binds a transposon end to form singly-bound transposon (T_2A-B or A-BT_2), the DNA forms a loop, the same T_2 binds the other transposon end (AT_2B), and the T_2 cuts out the transposon (reaction rate k_3). At high T_2 concentration: singly-bound transposons bind new T_2 creating doubly-bound transposons (T_2A-BT_2), which prevent transposition and regulate the process.



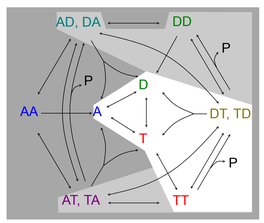


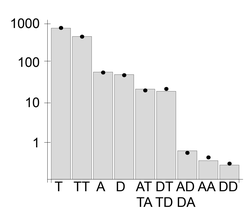
Wildcard rules

```
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
```

E. coli MinD

E. coli locate their cell division plane in part through spatiotemporal oscillations of Min proteins. Of them, MinD binds ATP (T), ADP (D), or no nucleotide (A); it also dimerizes when bound to ATP (T+T -> TT) and MinD hydrolyzes ATP when dimeric[3,6] (e.g. TT -> DT).





(Left) The expanded portion of the network during on-the-fly expansion. (Right) Steady-state molecule counts with deterministic (bars) and stochastic (dots) results.

Wildcard rules

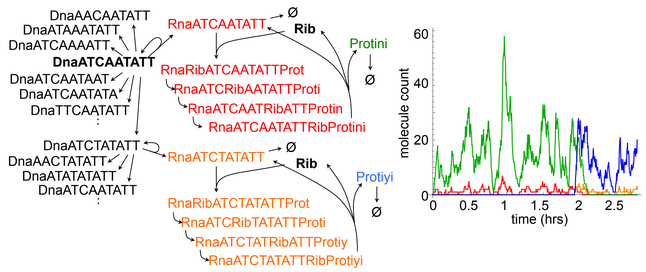
```
rxnAtoD     *A* <-> *D*      KATOD   KDTOA
rxnAtoT     *A* <-> *T*      KATOT   KTTOA
rxnDtoT     *D* <-> *T*      KDTOT   KTTOD
rxndimer    T + T -> TT      KDIMER
rxndissoc   ?? -> ? + ?      KDISS
rxnhydro    ?&T -> ?&D       KHYDRO
```

Sequences

Transcription from DNA to mRNA, and then translation to protein can be represented with wildcards.

Wildcard rules

```
rxnTransc     Dna* -> Dna$1 + Rna$1                          KTRANSC
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
rxnMut        Dna*?* -> Dna*{A|T|C|G}*                       KMUT
rxnRnaDeg     Rna*[A,T,C,G] -> 0                             KRNADEG
rxnProtDeg    Prot* -> 0                                     KPROTDEG
```



Wildcards also work with chemical structures using the SMILES notational system. For example, the following rules represent main steps of *E. coli* lipid synthesis[7].

Wildcard rules

```
FabB + ACP-C(=O)C{CCl/C=C\}* -> FabB + ACP-C(=O)CC(=O)C{CCl/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*
```

Conclusions

- Rule-based modeling with wildcards is often better than formal methods because it is more versatile and intuitive.
- However, wildcards are less good for: very large complexes and complexes with complicated symmetry.
- Smoldyn supports rule-based modeling with both wildcards and the BNGL language.

References

1. Blinov, Faeder, Goldstein, Hlavacek, *Bioinf.* 20:3289, 2004.
2. Andrews, *Bioinf.* 33:710, 2017.
3. Andrews, *Meth. Mol. Biol.* accepted 2016; BioArXiv, 2017.
4. Andrews, Addy, Brent, Arkin, *PLoS Comp. Biol.* 6:e1000705, 2010.
5. Clays Bouuaert, Lipkow, Andrews, Liu, Chalmers, *eLife* 2:e00668, 2013.
6. Andrews, Moghaddam, Groves, *in* "American Chemical Society", San Francisco, CA, 2006.
7. Emiola, Andrews, Heller, George, *Proc. Natl. Acad. Sci. USA* 113:3108, 2016.

Funding: Simons Foundation grant awarded to SSA and EPSRC grant EP/K032208/1 awarded to the Isaac Newton Institute.