Maximal Aggregation of Polynomial Dynamical Systems

Supporting Information

Luca Cardelli, Mirco Tribastone, Max Tschaikowski, Andrea Vandin

List of Figures

List of Tables

Listings

This appendix provides accompanying material for the main text. Section S1 collects all technical results. Instead, Section S2 provides more details on the case studies considered in the main text, and complements them with further models.

S1. Technical Results

Using the notation and definitions of section "Model Definitions" in the main text, we hereby provide the necessary proofs to:

- formally relate forward (Section S1.1) and backward (Section S1.2) equivalence to the syntactic conditions in Equations 6 and 7 from the main text, respectively;
- our partition refinement algorithm, stating its correctness and complexities (Section S1.3);
- our notion of reduced RN, and formally relate its ODEs to those of the original RN (Section S1.4).

S1.1 Characterization of Forward Equivalence

Here we state and prove the first main theorem. It relates a forward equivalence on the ODE variables $x_1, \ldots, x_n$ with the **fr**-conditions on the corresponding species $S_1, \ldots, S_n$, of the reaction network (RN) encoding. Throughout this document, with a slight abuse of terminology we may refer to a partition of species as a forward/backward equivalence to indicate the corresponding partition of ODE variables.

In the following, let (S, R) be an RN, $\mathcal{R} \subseteq S \times S$ an equivalence relation, and $\mathcal{H} = S/\mathcal{R}$. Also we denote the choice function of $\mathcal{H}$ as $\mu : S \to S$, defined such that $\mathcal{H} = \{\mu^{-1}(Y) \mid Y \in \mu(S)\}$. We lift μ to multisets by applying it element-wise, e.g., $\mu(S_i + S_j) = \mu(S_i) + \mu(S_j)$. For any $H \in \mathcal{H}$ set $x_H := \sum_{S_i \in H} x_i$ and let $Y^H \in S$ denote the representative of H, i.e. such that $\mu^{-1}(Y^H) = H$. Finally, for any multiset ρ we use ρ_i to denote the multiplicity of species S_i in ρ, and $|\rho|$ to denote the size of ρ (including multiplicities).

The following two propositions are used in the proof of the theorem.

Proposition 1. *The equivalence relation $\mathcal{R}$ satisfies the condition of Eq. (6) in the main text if and only if, for all $H \in \mathcal{H}$ and $\rho \in \mathcal{MS}(S)$, it holds that*

$$\frac{\phi(\rho, H)}{[\rho]!} = \frac{\phi(\mu(\rho), H)}{[\mu(\rho)]!}$$

Proof. If $\mathcal{R}$ satisfies Eq. (6), it holds that

$$\frac{\phi(S_i + \rho, H)}{[S_i + \rho]!} = \frac{\phi(S_j + \rho, H)}{[S_j + \rho]!}$$

for all $S_i, S_j \in H$. An iterative application of this identity in which each $S_i \in \rho$ is replaced by $S_k = \mu(S_i)$ yields the "if" direction. To see the "only if" direction, note that

$$\frac{\phi(\rho, H)}{[\rho]!} = \frac{\phi(\mu(\rho), H)}{[\mu(\rho)]!} = \frac{\phi(\rho', H)}{[\rho']!}$$

for any ρ' that satisfies $\mu(\rho) = \mu(\rho')$. Since $\mu(S_i + \rho) = \mu(S_j + \rho)$ if $S_i, S_j \in \mathcal{H}$, this yields the claim. □

Proposition 2. *Let G be a set, and $\mathcal{H}$ a partition of G. Then it holds*

$$\sum_{\rho \in \mathcal{MS}_n(G)} \prod_{S_k \in G} x_k^{\rho_k} = \sum_{\rho \in G^n} \frac{\prod_{S_k \in G} x_k^{\rho_k}}{[\rho]!}$$

where $\mathcal{MS}_n(G) := \{\rho \in \mathcal{MS}(G) \mid |\rho| = n\}$ denotes the set of multisets in $\mathcal{MS}(G)$ of size n.

Proof. It is well known that the multinomial coefficient $[\rho]!$ gives the number of distinct permutations of a multiset ρ. Hence, the statement easily follows by noting that for each $\rho \in \mathcal{MS}_n(G)$, G^n contains $[\rho]!$ instances that are equal to ρ up to permutation. □

In the theorem below, we remark that the set $\{\tilde{\rho} \mid (\tilde{\rho} + S_k \xrightarrow{\alpha} \pi) \in R, k \in \{i, j\}\}$ might contain the multiset $\emptyset$ to account for the case of reactions with one species only (i.e., either S_i or S_j) as reagent. Instead, reactions with $\emptyset$ as reagents, which encode constants, do not affect the notion of forward equivalence, and are therefore ignored by **fr**.

Theorem 1. *$\mathcal{H}$ is a forward equivalence if and only if for any two blocks $H, H' \in \mathcal{H}$ and all species $S_i, S_j \in H$ it holds*

$$\mathbf{fr}(S_i, \rho, H') = \mathbf{fr}(S_j, \rho, H') \quad \textit{for all} \quad \rho \in \left\{\tilde{\rho} \mid (\tilde{\rho} + S_k \xrightarrow{\alpha} \pi) \in R, k \in \{i, j\}\right\}.$$

Proof. The theorem can be proved by showing that, for each $H \in \mathcal{H}$, the sum of the ODEs of the species in H can be expressed in terms of the aggregated variables of the blocks of $\mathcal{H}$ only, if and only if $\mathcal{H}$ satisfies the condition in Eq. (6). Set $a_r := \max\{|\rho| \mid (\rho \xrightarrow{\alpha} \pi) \in R\}$ to denote the maximal size of the reagents in R. Then such sum is:

$$\begin{aligned}
\sum_{S_i \in H} \sum_{\rho \xrightarrow{\alpha} \pi \in R} (\pi_i - \rho_i) \cdot \alpha \cdot \prod_{S_t \in S} x_t^{\rho_t} &= \sum_{S_i \in H} \sum_{\rho \in \mathcal{MS}(S)} \prod_{S_t \in S} x_t^{\rho_t} \cdot \phi(\rho, S_i) \\
&= \sum_{\rho \in \mathcal{MS}(S)} \prod_{S_t \in S} x_t^{\rho_t} \cdot \phi(\rho, H) \\
&= \sum_{n=1}^{a_r} \sum_{\rho \in S^n} \prod_{S_t \in S} x_t^{\rho_t} \cdot \frac{\phi(\rho, H)}{[\rho]!} \qquad \text{(by Proposition 2)} \qquad \text{[SE1]}
\end{aligned}$$

We start by showing that we can rewrite each summand of the outer summation of Eq. (SE1) in terms of the aggregated variables whenever $\mathcal{H}$ satisfies the condition of Eq. (6).

$$\begin{aligned}
\sum_{\rho \in S^n} \prod_{S_t \in S} x_t^{\rho_t} \cdot \frac{\phi(\rho, H)}{[\rho]!} &= \sum_{\rho \in S^n} \prod_{S_t \in S} x_t^{\rho_t} \cdot \frac{\phi(\mu(\rho), H)}{[\mu(\rho)]!} \qquad \text{(by Proposition 1)} \\
&= \sum_{\rho \in S^{n-1}} \sum_{S_n \in S} \prod_{S_t \in S} x_t^{\rho_t} \cdot x_n \cdot \frac{\phi(\mu(\rho + S_n), H)}{[\mu(\rho + S_n)]!} \\
&= \sum_{\rho \in S^{n-1}} \sum_{H_n \in \mathcal{H}} \sum_{S_n \in H_n} \prod_{S_t \in S} x_t^{\rho_t} \cdot x_n \cdot \frac{\phi(\mu(\rho + S_n), H)}{[\mu(\rho + S_n)]!} \\
&= \sum_{\rho \in S^{n-1}} \sum_{H_n \in \mathcal{H}} \prod_{S_t \in S} x_t^{\rho_t} \cdot x_{H_n} \cdot \frac{\phi(\mu(\rho + Y^{H_n}), H)}{[\mu(\rho + Y^{H_n})]!} \qquad \text{[SE2]}
\end{aligned}$$

The same transformations can be repeated for each $n-1$ entries of the remaining cartesian product S^{n-1} of Eq. (SE2), obtaining an expression in the form

$$\sum_{H_1 \in \mathcal{H}} x_{H_1} \sum_{H_2 \in \mathcal{H}} x_{H_2} \cdots \sum_{H_n \in \mathcal{H}} x_{H_n} \cdot \frac{\phi(\mu(\sum_{1 \le i \le n} Y^{H_i}), H)}{[\mu(\sum_{1 \le i \le n} Y^{H_i})]!}$$

This completes the proof of the "if" direction.

We now address the "only if" direction by showing that $\mathcal{H}$ satisfies the condition of Eq. (6) whenever we can rewrite Eq. (SE1) in terms of the aggregated variables. To this end, we assume that, for all $\tilde{H} \in \mathcal{H}$, there exists a polynomial $\wp_{\tilde{H}}$ of degree less than or equal to $N = a_r$ in $|\mathcal{H}| = |\{H_1, \ldots, H_m\}| = m$ variables that satisfies

$$\sum_{n=1}^{N} \sum_{\rho \in \mathcal{MS}_n(S)} \prod_{S_i \in S} x_i^{\rho_i} \cdot \phi(\rho, \tilde{H}) = \wp_{\tilde{H}}(x_{H_1}, \ldots, x_{H_m})$$

for all $x \in \mathbb{R}^S$, where $\mathcal{MS}_n(S)$ is a short-hand for $\{\rho \in \mathcal{MS}(S) \mid |\rho| = n\}$. We will also use σ_{Y^H} to denote the multiplicity in the multiset σ of the representative of block H.

Since there exist unique coefficients $\gamma_\rho^{\tilde{H}}$, where $\rho \in \bigcup_{n=1}^{N} \mathcal{MS}_n(\mu(S))$, such that

$$\wp_{\tilde{H}}(x_{H_1}, \ldots, x_{H_m}) = \sum_{n=1}^{N} \sum_{\sigma \in \mathcal{MS}_n(\mu(S))} \gamma_\sigma^{\tilde{H}} \cdot \prod_{H \in \mathcal{H}} x_H^{\sigma_{Y^H}}$$

for all $x \in \mathbb{R}^S$, we thus infer that

$$\sum_{n=1}^{N} \sum_{\rho \in \mathcal{MS}_n(S)} \prod_{S_i \in S} x_i^{\rho_i} \cdot \phi(\rho, \tilde{H}) = \sum_{n=1}^{N} \sum_{\sigma \in \mathcal{MS}_n(\mu(S))} \gamma_\sigma^{\tilde{H}} \cdot \prod_{H \in \mathcal{H}} \Big(\sum_{S_i \in H} x_i\Big)^{\sigma_{Y^H}} \qquad \text{[SE3]}$$

for all $V \in \mathbb{R}^S$.

Given some $\rho \in \mathcal{MS}_n(S)$, we set $\sigma := \mu(\rho)$ and compute how many times the monomial $\prod_{S_i \in S} x^{\rho_i}$ appears in the polynomial

$$\prod_{H \in \mathcal{H}} x_H^{\sigma_{Y^H}} = \prod_{H \in \mathcal{H}} \big(\sum_{S_i \in H} x_i \big)^{\sigma_{Y^H}} \qquad \text{[SE4]}$$

That is, we want to determine the coefficient of the monomial $\prod_{S_i \in S} x_i^{\rho_i}$ in the polynomial that arises if (SE4) is multiplied out. Note that $\sum_{S_i \in H} \rho_i$ gives the number of elements in ρ that are in $H \in \mathcal{H}$ and that $\sum_{S_i \in H} \rho_i = (\mu(\rho))_{Y^H} = \sigma_{Y^H}$. Consequently, if $H = \{S_1^H, \ldots, S_{|H|}^H\}$ and H^{S_i} denotes the block of $\mathcal{H}$ containing S_i, the number of monomials contributed by $\big(\sum_{S_k \in H^{S_i}} x_k \big)^{\sigma_i}$ is given by the multinomial coefficient

$$\binom{(\sum_{S_i \in H} \rho_i)!}{\rho_1^H, \ldots, \rho_{|H|}^H} = \frac{(\sum_{S_i \in H} \rho_i)!}{\prod_{S_i \in H} \rho_i!}$$

where ρ_i^H denotes the occurrences in ρ of S_i^H. This implies that the monomial $\prod_{S_i \in S} x_i^{\rho_i}$ appears exactly

$$\prod_{H \in \mathcal{H}} \frac{(\sum_{S_i \in H} \rho_i)!}{\prod_{S_i \in H} \rho_i!} = \frac{\prod_{H \in \mathcal{H}} (\sum_{S_i \in H} \rho_i)!}{\prod_{S_i \in S} \rho_i!}$$

times in the polynomial $\prod_{S_i \in \mu(S)} \big(\sum_{S_k \in H^{S_i}} x_k \big)^{\sigma_i}$. Using this and (SE3) we conclude that

$$\phi(\rho, \tilde{H}) = \gamma^{\tilde{H}}_{\mu(\rho)} \cdot \frac{\prod_{H \in \mathcal{H}} (\sum_{S_i \in H} \rho_i)!}{\prod_{S_i \in S} \rho_i!}$$

for any $\rho \in \mathcal{MS}_n(S)$. Now, let us assume that we are given $\rho, \rho' \in \mathcal{MS}_n(S)$ with $\mu(\rho) = \sigma = \mu(\rho')$. Then, it holds that

$$\phi(\rho, \tilde{H}) \cdot \frac{\prod_{S_i \in S} \rho_i!}{\prod_{H \in \mathcal{H}} (\sum_{S_i \in H} \rho_i)!} = \gamma^{\tilde{H}}_{\sigma} = \phi(\rho', \tilde{H}) \cdot \frac{\prod_{S_i \in S} \rho_i'!}{\prod_{H \in \mathcal{H}} (\sum_{S_i \in H} \rho_i')!}.$$

Since $\prod_{H \in \mathcal{H}} (\sum_{S_i \in H} \rho_i)! = \prod_{H \in \mathcal{H}} (\sum_{S_i \in H} \rho_i')!$ because $\mu(\rho) = \mu(\rho')$, this is equivalent to

$$\phi(\rho, \tilde{H}) \cdot \prod_{S_i \in S} \rho_i! = \phi(\rho', \tilde{H}) \cdot \prod_{S_i \in S} \rho_i'!$$

Thanks to the fact that $\mu(\rho) = \mu(\rho')$, the above implies

$$\frac{\phi(\rho, \tilde{H})}{[\rho]!} = \frac{\phi(\rho', \tilde{H})}{[\rho']!}$$

and Proposition 1 yields the claim. □

S1.2 Characterization of Backward Equivalence

In this section we prove the characterization theorem that relates a backward equivalence with an RN. In order to do so, we first provide an alternative characterization of backward equivalence. This is more natural, but it is not based on the notion of splitter, and hence it is not amenable to efficient partition refinement. Given a species $S_i \in S$ and a set of multisets $\mathcal{M} \in \mathcal{MS}(S)$, we might write

$$\phi(\mathcal{M}, S_i) := \sum_{\rho \in \mathcal{M}} \phi(\rho, S_i)\,.$$

Proposition 3 (Alternative characterization of backward equivalence)**.** *Let (S, R) be an RN and $\mathcal{H}$ a partition of S. Then $\mathcal{H}$ is a backward equivalence if and only if for any block $H \in \mathcal{H}$, and all species $S_i, S_j \in H$ we have*

$$\phi(\mathcal{M}, S_i) = \phi(\mathcal{M}, S_j) \quad \textit{for all} \quad \mathcal{M} \in \{\rho \mid (\rho \xrightarrow{\alpha} \pi) \in R\}/\approx_{\mathcal{H}}\,. \qquad \text{[SE5]}$$

Proof. We say that $x \in \mathbb{R}^S$ is *uniform* on $\mathcal{H}$ whenever $x_i = x_j$ for all $H' \in \mathcal{H}$ and $S_i, S_j \in H'$. Next we define $[\![\rho]\!]_x := \prod_{S_i \in S} x_i^{\rho_i}$ and set $Q := \{\rho \mid \rho \xrightarrow{\alpha} \pi \in R\}/\approx_{\mathcal{H}}$. Fix some arbitrary $H \in \mathcal{H}$ and $S_l, S_j \in H$ and note that for any $S_k \in S$ we have:

$$\mathcal{P}_k(x) = \sum_{(\rho \xrightarrow{\alpha} \pi) \in R} \alpha \cdot (\pi_k - \rho_k)[\![\rho]\!]_x = \sum_{\mathcal{M} \in Q} \sum_{\rho \in \mathcal{M}} \phi(S_k, \rho)[\![\rho]\!]_x = \sum_{\mathcal{M} \in Q} \underbrace{\Big(\sum_{\rho \in \mathcal{M}} \phi(S_k, \rho) \Big)}_{c(x_k, \mathcal{M})} [\![\rho_0]\!]_x$$

whenever x is uniform on $\mathcal{H}$, with ρ_0 being a chosen representative multiset in $\mathcal{M}$. Observe also that the function $x \mapsto \mathcal{P}_k(x)$, where x is uniform on $\mathcal{H}$, defines a polynomial in $|Q|$ variables with the monomials $\{c(x_k, \mathcal{M}) \cdot [\![\rho_0]\!]_x \mid \mathcal{M} \in Q$ and ρ_0 being the representative of $\mathcal{M}\}$. Finally, recall that the multi dimensional version of Taylor's theorem implies that two real polynomials are equivalent if and only if they have the same monomials. Thus, $\mathcal{P}_l(x) = \mathcal{P}_j(x)$ for all x that are uniform on $\mathcal{H}$ if and only if $c(x_i, \mathcal{M}) = c(x_j, \mathcal{M})$ for all $\mathcal{M} \in Q$. □

We now actually prove that the conditions from Eq. (7) in the main text characterize backward equivalence. In the theorem below, we remark that $\{\rho \mid (\rho + S_k \xrightarrow{\alpha} \pi) \in R, S_k \in S\}/\approx_{\mathcal{H}}$ might contain the equivalence class $\{\emptyset\}$, corresponding to reactions with one reagent (S_k) only.

Theorem 2. *Let (S, R) be an RN and $\mathcal{H}$ a partition of S. Then $\mathcal{H}$ is a backward equivalence if and only if for any two blocks $H, H' \in \mathcal{H}$, any two $S_i, S_j \in H$, it holds that*

$$\phi(\emptyset, S_i) = \phi(\emptyset, S_j) \qquad \text{and} \qquad \mathbf{br}(S_i, \mathcal{M}, H') = \mathbf{br}(S_j, \mathcal{M}, H') \text{ for all } \mathcal{M} \in \big\{\rho \mid (\rho + S_k \xrightarrow{\alpha} \pi) \in R, S_k \in S\big\}/\approx_{\mathcal{H}} .$$

Proof. The proof consists in a reduction of the condition in Eq. (SE5) to the condition in Eq. (7) of the main text. The case of reactions with $\emptyset$ as reagents is trivial, as the two conditions coincide. The rest of the proof hence focuses on reactions with non-empty reagents. We remark that for any two $\rho, \rho' \in \mathcal{MS}(S)$ we write $\rho \approx_{\mathcal{H}} \rho'$ if the two multisets have same number of species of each block of $\mathcal{H}$. Hence, for any $\mathcal{M} \in \mathcal{MS}(S)/\approx_{\mathcal{H}}$ and $H \in \mathcal{H}$ we use $\mathcal{M}_H$ to denote $\sum_{S_i \in H} \rho_i$, with ρ any multiset in $\mathcal{M}$.

The proof is based on the following fact:

1. For each $\mathcal{M} \in \mathcal{MS}(S)/\approx_{\mathcal{H}}$ and $H \in \mathcal{H}$ such that $\mathcal{M}_H > 0$ there exists a $\mathcal{M}^H \in \mathcal{MS}(S)/\approx_{\mathcal{H}}$ defined such that $\mathcal{M}^H_H = \mathcal{M}_H - 1$, and for all $H' \neq H \in \mathcal{H}$ we have $\mathcal{M}^H_{H'} = \mathcal{M}_{H'}$.

For all $H \in \mathcal{H}$ such that $\mathcal{M}_H > 0$ we can rewrite the left-hand side of Eq. (SE5) as follows:

$$\sum_{S_z \in H} \sum_{\rho \in \mathcal{M}^H} \frac{\phi(S_z + \rho, S_i)}{|S_z + \rho|_H} = \mathbf{br}(S_i, \mathcal{M}^H, H) \qquad \text{[SE6]}$$

Hence we can rewrite Eq. (SE5) as

$$\mathbf{br}(S_i, \mathcal{M}^H, H) = \mathbf{br}(S_j, \mathcal{M}^H, H)$$

The proof is closed by noting that by Point 1 we have that for any pair of H and $\mathcal{M}$ considered in Eq. (7), there exists an $\mathcal{M}' \in \mathcal{MS}(S)/\approx_H$ such that $\mathcal{M}' = \mathcal{M}^H$ (and vice versa). All pairs H and $\mathcal{M}$ not considered in Point 1 are such that $\mathbf{br}(S_k, \mathcal{M}, H) = 0$ for all $S_k \in S$. □

Note that in the left-hand side of Eq. (SE6) we add the scaling $|S_z + \rho|_H = |\{(S_l, m) \in (S_z + \rho) \mid S_l \in H\}|$. This is needed to account for the fact that each multiset $S_z + \rho$ might be considered more than once in the left-hand side of Eq. (SE6). Consider for example the case $S_z + \rho = S_1 + S_2$, with $S_1, S_2 \in H$. This multiset will be considered two times: $S_z = S_1$ and $\rho = S_2$; $S_z = S_2$ and $\rho = S_1$.

S1.3 Partition-refinement Algorithm

In this section we present the details of our algorithm to iteratively compute the largest forward/backward equivalence that refines a given initial input partition of variables. In particular, the RN characterizations of forward and backward equivalence are given in the seminal style of Larsen and Skou's probabilistic bisimulation (47), whereby, roughly speaking, two states are equivalent if their behavior toward any equivalence class is the same. Indeed, the notion of $\mathbf{fr}[S_i, \rho, H]$ is in such desired format: S_i is the species for which the equivalence is being checked, H is an equivalence class of "target" states, named *splitter* in the literature, while ρ plays the role of a *label*, identifying partner species

reacting with S_i. The case of $\mathbf{br}[S_i, \mathcal{M}, H]$ is similar. However, while for forward equivalence the labels range over the set of multisets of species (including the empty one $\emptyset$ to indicate reactions with one reagent), in backward equivalence the labels range over blocks of multisets containing same number of species equivalent according to the partition to be checked (again, together with the distinguished set $\{\emptyset\}$ for reactions with one reagent). When used within the partition refinement algorithm, splitting a partition block leads to a refinement of the labels considered in backward equivalence. In other words, unlike for the forward case, the set of labels must be updated at every iteration. However, as we will see, this does not worsen the complexity of the backward case, as the same set will be actually used to represent the labels in both cases.

This correspondence suggests to employ a partition refinement approach similar to that for computing the largest probabilistic bisimulation (48), iteratively refining an input partition considering a *splitter* block at a time that tells apart the behavior of two species. Intuitively, the algorithm proceeds by steps in which i) it considers one of the blocks H of the current partition as a splitter, ii) refines any block that does not satisfy the $\mathbf{fr}[\cdot, \cdot, H]$ or $\mathbf{br}[\cdot, \cdot, H]$ condition, and iii) terminates when all blocks (including the intermediate generated ones) have been considered as splitter.

Reactions with $\emptyset$ as reagent do not affect forward equivalence, and are hence ignored by $\mathbf{fr}$. Instead, they are considered in the backward case by the first condition of Eq. (7). This condition does not depend on the partition to be refined (i.e., no block H' appears in it), and it is hence dealt with using an initialization phase that refines the initial partition in sub-blocks that satisfy the condition.

Data Structures. We are now ready to present our algorithm. First, we introduce the data structures employed. This is relevant in order to study the computational complexity. In particular, in order to achieve tight time and space bounds, we make use of pointer-based data structures only. Hence we assume that species, partition blocks and reactions are stored once and then referred to by other data structures via pointers.

Notation. For ease of presentation we fix an RN (S, R), and set $s := |S|$, $r := |R|$. Also, we use $p_r := \max\{\sum_{S_i \in S} \mathbb{1}_{\{\rho_i > 0\}} \mid (\rho \xrightarrow{\alpha} \pi) \in R\}$ and $p_p := \max\{\sum_{S_i \in S} \mathbb{1}_{\{\pi_i > 0\}} \mid (\rho \xrightarrow{\alpha} \pi) \in R\}$ to denote the maximum number of different species which appear as reagents or products of reactions in R, respectively, and $p := \max(p_r, p_p)$. Also, we use $a_r := \max\{|\rho| \mid (\rho \xrightarrow{\alpha} \pi) \in R\}$ for the maximum arity of reagents in R.

We define $\mathcal{L}(R) := \{\rho \mid \exists (S_i + \rho \xrightarrow{\alpha} \pi) \in R\}$ as the set of all *labels*, and write $l := |\mathcal{L}(R)|$, which can be bounded by $\mathcal{O}(p_r \cdot r)$. In fact, each reaction $\rho \xrightarrow{\alpha} \pi$ gives rise to one label per different species in ρ, which is bound by p_r. Each label is obtained by decreasing the multiplicity of each species S_i such that $\rho_i > 0$ by one. For example, $\{(1, S_i), (2, S_j)\}$ gives rise to the labels $\{(1, S_i), (1, S_j)\}$ and $\{(2, S_j)\}$.

If we disregard pathological RNs with species not appearing in any reaction (which can always be removed from the set S in a pre-processing step), we can bound s by $\mathcal{O}((p_r + p_p) \cdot r) = \mathcal{O}(p \cdot r)$. This is because each reaction can have at most p_r and p_p different species as reagents and products, respectively.

We remark that, in general, p is bounded by s. However, we prefer to explicitly use it because it allows us to relate the complexity of the algorithm to the degree of the polynomials appearing in the derivatives of the ODE system. Indeed, we have that the maximum degree d is equal to the arity of the largest reagents in R (counting multiplicities), and to the arity minus 1 of the largest products. Hence, we have $p \leq d + 1$. Similarly, we have $a_r = d$.

RN representation. Species are stored in a list. A reaction is a structure with a real-valued field for its rate, and two lists of pairs in the form `(multiplicity,species)` for the reagents and products. The list for reagents is sorted with respect to a total ordering on the species. In addition, reactions are provided with a real field `mcoeff` used to store the multinomial coefficient ($[\rho]!$) of its reagents (ρ). Storing R thus requires $\mathcal{O}(p \cdot r)$ space.

In order to efficiently compute the multinomial coefficients of the reagents, we assume that a_r is known, and that a vector `factorials` of a_r entries is precomputed such that the position i contains the value of $i!$. However, it could be computed in $\mathcal{O}(a_r)$ time, because the value in the entry i is computed by multiplying the value of the entry $i - 1$ by i.

Each label is generated by decreasing the multiplicity of a species in a multiset of reagents by 1. Hence, a label can be stored in constant space as a pair `(reagents,species)`. For example, the label ρ_{S_i} obtained by decreasing the multiplicity of species S_i by 1 in the multiset ρ can be stored as (ρ, S_i). This encoding is not unique, as a label can be obtained from different multisets by decreasing different species (e.g., the label $\{(2, S_i), (2, S_j)\}$ can be obtained from $\{(3, S_i), (2, S_j)\}$, $\{(2, S_i), (3, S_j)\}$ or $\{(2, S_i), (2, S_j), (1, S_k)\}$, for any species S_k), but we will store it only once. Furthermore, since reagents are lists of pairs `(multiplicity,species)` sorted with respect to the species, it is possible to compare two labels in $\mathcal{O}(p_r)$ time, even if different encodings are used for the same label. Storing $\mathcal{L}(R)$ thus takes $\mathcal{O}(l)$ space. The set $\mathcal{L}(R)$ is stored as a sorted list, for which insertions and searches cost $\mathcal{O}(p_r \cdot \log l)$ time, because both operations require to compare $\mathcal{O}(\log l)$ labels, each made by up to p_r pairs. In order to improve performance we also maintain a vector `pos` for each reaction $\rho \xrightarrow{\alpha} \pi$ containing p_r integers. The i-th entry contains

```
1  LargestEquivalence(χ,S,R,H) :=
2    Init(R)
3    if(χ = B)
4      H = BackwardPrepartitioning(S,R,H,M)
5    spls = shallow copy of H
6    while(spls ≠ ∅)
7      H_sp = pop(spls)
8      Split(χ,S,R,M,H,H_sp,spls)
```

Algorithm S1. Computation of the largest equivalences.

the position in $\mathcal{L}(R)$ of the label obtained by decreasing by one the multiplicity of the species in position i in the list which stores ρ. For ease of presentation we use $\texttt{pos}[S_i]$ to denote the label obtained by decreasing the multiplicity of S_i. This does not worsen the space complexity to store reactions.

We make use of two vectors, `nzs` and `out`, indexed by species. Each entry $\texttt{nzs}[S_i]$ points to a list of pairs `(reaction,netStoichiometry)` containing all reactions $\rho \xrightarrow{\alpha} \pi$ such that the *net stoichiometry of* S_i is non-zero, i.e., $(\pi_i - \rho_i) \neq 0$. Note that each reaction may appear in the lists in `nzs` for at most $p_r + p_p$ species, thus requiring $\mathcal{O}(p \cdot r)$ space to store `nzs`. The vector `out` is similar, but each $\texttt{out}[S_i]$ entry points to a list of reactions having S_i in their reagents. Each reaction may appear in `out` for at most p_r species. Thus the space required by `out` is $\mathcal{O}(p_r \cdot r)$.

In the algorithm we build sets of elements. Insertions in sets can be implemented in constant time because an element is never added to a set more than once in our algorithm.

Refinable partition. A partition is stored as a doubly-linked list of pointers to its blocks. Each block record contains an integer to store its size and pointers to two doubly-linked lists that divide the species into `marked` and `unmarked`, as a result of operations that are used to split blocks, discussed later. Each species has a pointer to its block in the current partition. Thus, finding the block for a species, marking, and unmarking take constant time. Also, it is possible to scan the species of a block in time linear with respect to its size, and to split it in time linear with respect to the number of marked states.

The operation of splitting a block H creates a new block H_1 containing the marked species of H, while H maintains those that are not marked. This requires to assign the list pointed by $H.\texttt{marked}$ to $H_1.\texttt{unmarked}$ and to assign an empty list to $H.\texttt{marked}$. These operations are done in constant time, while a time linear with respect to the originally marked species of H is necessary to update their reference to the new block H_1. If instead H originally contained just marked or unmarked species, then no split is actually performed, and marked species get unmarked at no further cost.

Splitters. The list of pointers `spls` refers to the blocks of the current partition that will be used as splitters. An $s \times l$ matrix `M` of real numbers is maintained to efficiently compute the values of **fr** and **br**. Columns respect the order of $\mathcal{L}(R)$. A *possible majority candidate* (pmc) of an array `A` of size s is either the value which appears more than $\lfloor s/2 \rfloor$ times in `A`, or any other value if it does not exist. We calculate the *pmc row* of the matrix `M` by extending the algorithm from (17) to vectors in a straightforward manner.

We denote the row of species S_i in `M` by $\texttt{M}[S_i]$, that is $\texttt{M}[S_i] \in \mathbb{R}^l$. In the course of splitting, we sort species according to the lexicographical order on their rows in `M`. Clearly, sorting a set $H \in \mathcal{H}$ takes $\mathcal{O}(l \cdot |H| \cdot \log |H|)$ time, as $\mathcal{O}(|H| \cdot \log |H|)$ comparisons are needed, each requiring $\mathcal{O}(l)$ time.

This leads to an overall $\mathcal{O}(a_r + s + p \cdot r + l + s \cdot l)$ space complexity, which, given that s and l are bound by $\mathcal{O}(p \cdot r)$, is at most $\mathcal{O}(a_r + s \cdot p \cdot r)$. Other auxiliary lists and sets of pointers presented later will respect the space bound given above.

Algorithm Overview. We are now ready to provide the partition refinement algorithm. Algorithm S1 provides the parametric procedure `LargestEquivalence` for computing the largest ODE equivalence that refines a given initial partition $\mathcal{H}$ of species of an RN (S, R). The first argument (χ) can be either `F` or `B`, specifying the notion of ODE equivalence to be computed (i.e., forward or backward equivalence, respectively).

Before the actual partition refinement it is necessary to perform some initialization steps. In particular, we compute the multinomial coefficients of the reagents of each reaction, as well as the set of labels. Also, we have to initialize `M`. Consequently, our algorithm starts (Line 2) by invoking the `Init` procedure.

Init (Algorithm S2). This procedure iterates the set of reactions once (Line 2), and for each reaction $\rho \xrightarrow{\alpha} \pi$ it scans the pairs `(multiplicity,species)` used to store ρ (Line 6), requiring $\mathcal{O}(r \cdot p_r)$ time. For each pair (m, S_i) in ρ,

```
1  Init(R) :=
2    forall((ρ --α--> π) ∈ R)
3      pos = new vector of size p_r
4      coeff = 1
5      size =0
6      forall((m, X) ∈ ρ)
7        ρ_{S_i} = (ρ, S_i)  //label obtained decreasing by 1 the multiplicity of S_i in ρ
8        add ρ_{S_i} to L(R)
9        pos[S_i] = position of ρ_{S_i} in L(R)
10       size = size + m
11       coeff = coeff · factorials[m]
12     coeff = factorials[size]/coeff
13   M = build an s × l matrix of reals
```

Algorithm S2. Initialization: compute the multinomial coefficients and initialize the set of labels and `M`.

```
1  BackwardPrepartitioning(S,R,H):=
2    V = build an array of size s initialized to 0
3    //Iterate once R to compute φ(∅, S_i) for all species
4    forall((∅ --α--> π) ∈R)
5      forall((m, S_i) ∈ π)
6        V[S_i]=V[S_i] + m · α
7    //Refine H according to the computed values
8    forall(H ∈ H)
9      Sort and split H wrt V[S_i], for all S_i ∈ H, yielding H_1, ..., H_b
10     Add H_1, ..., H_b to H'
11   return H'
```

Algorithm S3. Pre-partition $\mathcal{H}$ wrt the first condition of Eq. (7).

Line 7 creates the label ρ_{S_i} obtained from ρ by decreasing by one the multiplicity of S_i (taking constant time). Then, Line 8 adds ρ_{S_i} to $\mathcal{L}(R)$ in $\mathcal{O}(p_r \cdot \log l)$ time, as discussed.

While scanning the pairs (m, S_i) of ρ, we also compute the multinomial coefficient of ρ (Lines 11-12). We access the precomputed factorial of each multiplicity m in `factorials`, and store in `coeff` the products of all the factorials of the multiplicities of ρ. Then, the computation of $[\rho]!$ is completed in Line 12 by dividing the factorial of the size of ρ by `coeff`.

Note that `Init` also builds the `pos` vector of each reaction. It is initialized in Line 3, and set in Line 9. Finally, the $s \times l$ matrix of reals `M` is built and initialized to 0 in $\mathcal{O}(s \cdot l)$ time. This leads to an overall $\mathcal{O}(r \cdot p_r^2 \cdot \log l + s \cdot l)$ time complexity for the procedure `Init`.

Backward pre-partitioning (Algorithm S3). This procedure provides the coarsest refinement of $\mathcal{H}$ which satisfies the first condition of backward equivalence given in Eq. (7). It refines $\mathcal{H}$ according to the value $\phi(\emptyset, S_i)$ of each species S_i belonging to the same block. In particular, in this procedure we create a vector `V` of size s used to store the $\phi(\emptyset, \cdot)$ values of all species. We use `V[`S_i`]` to denote the entry of `V` corresponding to S_i, which will contain $\phi(\emptyset, S_i)$. We can thus compute all $\phi(\emptyset, \cdot)$ values in one iteration of R only (Lines 4-6), requiring $\mathcal{O}(r \cdot p)$ time. Then, we refine $\mathcal{H}$ (Lines 8-10). This can be done, for each initial block $H \in \mathcal{H}$, by sorting the species $S_i \in H$ according to the value stored in their entry of `V`. After sorting, all species belonging to the same sub-block will be alongside each other, and it is easy to transform them into new blocks in $\mathcal{O}(|H|)$ time. The sorting of each block requires $\mathcal{O}(|H| \cdot \log |H|)$ time, and the total time spent in sorting is thus $\mathcal{O}(\sum_{H \in \mathcal{H}} |H| \cdot \log |H|) \leq \mathcal{O}(\sum_{H \in \mathcal{H}} |H| \cdot \log s) = \mathcal{O}(s \cdot \log s)$. Overall, this yields $\mathcal{O}(r \cdot p + s \cdot \log s)$ time complexity. Given that $s \leq p \cdot r$, this can be bounded by $\mathcal{O}(r \cdot p \cdot \log s)$.

Iterative Refinement (Algorithm S1, Lines 5-8). This procedure performs the iterative partition refinement. If $\chi = F$, the blocks of $\mathcal{H}$ are split into sub-blocks of species with same $\mathbf{fr}(\cdot, \rho, H_{sp})$ for all $\rho \in \mathcal{L}(R)$. Instead, if $\chi = B$, blocks are split with respect to their $\mathbf{br}(\cdot, \mathcal{M}, H_{sp})$ for all labels $\mathcal{M} \in \{\rho \mid (\rho + S_k \xrightarrow{\alpha} \pi) \in R, S_k \in S\} / \approx_{\mathcal{H}}$

Line 5 creates the linked list `spls`, consisting of initial candidate splitters as pointers to each $H \in \mathcal{H}$: all blocks of $\mathcal{H}$ are considered as initial candidate splitters. Then, Lines 6-8 iterate while there are candidate splitters to be considered: after selecting a splitter (H_{sp}) and removing it from `spls`, `Split` is invoked to refine each block of $\mathcal{H}$

with respect to H_{sp}.

We now provide an overview of the `Split` procedure (Algorithm S4). A detailed presentation is given in the next section, together with the complexity results. `Split` first computes either $\mathbf{fr}(S_i, \rho, H_{sp})$ for all $S_i \in S$ and $\rho \in \mathcal{L}(R)$ (forward case) or $\mathbf{br}(S_i, \mathcal{M}, H_{sp})$, for all $S_i \in S$ and $\mathcal{M} \in \left\{\rho \mid (\rho + S_k \xrightarrow{\alpha} \pi) \in R, S_k \in S\right\}/\approx_{\mathcal{H}}$ (backward case). The rates are computed for all species and labels at once and are stored in `M`. We recall that backward equivalence uses different labels than forward equivalence. Nevertheless, as will be discussed in the next section, the number of labels used in the backward case is bounded by l as well, and hence `M` can be safely used in both cases. Then, the algorithm iterates over the set of blocks containing a species for which at least one non-zero **fr** or **br** rate has been computed. Each partition block H is split in sub-blocks with either same $\mathbf{fr}(\cdot, \rho, H_{sp})$ for all labels ρ ($\chi = F$), or same $\mathbf{br}(\cdot, \mathcal{M}, H_{sp})$ for all labels $\mathcal{M}$ ($\chi = B$), updating the list of splitters `spls` accordingly. Following the usual approach of Paige and Tarjan (18), a sub-block with maximal size is not added to `spls`. However, this is done only if the block that is split (i.e., H) has been already used as a splitter, as otherwise the algorithm would be incorrect, see the discussion in (17).

```
1  Split(χ,S,R,M,H,H_sp,spls) :=
2    T_S = ∅ //Set of species S_i with at least a non-zero fr/br[S_i,·,H_sp]
3    T_H = ∅ //Set of blocks containing the species in T_S
4    forall(S_j ∈ H_sp)
5      if(χ = F)
6        ComputeFR(S_j,M) //Compute fr(S_i,ρ,S_j) for all S_i,ρ. Populate T_S
7      else
8        ComputeBR(S_j,H_sp,M) //Compute br(S_i,M,S_j) for all S_i,M. Populate T_S
9    //M[S_i][ρ] stores fr(S_i,ρ,H_sp), or br(S_i,ρ_M,H_sp), with ρ_M a chosen ρ' ∈ M
10   forall(S_i ∈ T_S)
11     H = get block of S_i
12     Discard label of H, if any
13     if(M[S_i] is not a zero row) //Discard spurious species from T_S
14       if(H contains no marked states) //Add only once H to T_H
15         Add H to T_H
16       Mark S_i in H
17   while(T_H ≠ ∅)
18     H = pop(T_H)
19     H_1 = marked states of H
20     H = not marked states of H
21     if(H = ∅)
22       Give the identity of H to H_1
23     else
24       Make H_1 a new block
25     pmc = PMCRow(H_1,M)
26     H_2 = {S_i ∈ H_1 | M[S_i] not equal to the pmc-row}
27     H_1 = H_1 \ H_2
28     if(H_2 = ∅)
29       b = 1 //No need to split H_1.
30     else
31       Sort and split H_2 according to M[S_i], yielding H_2,...,H_b
32       Make each of H_2,...,H_b a new block
33     if(H ∈spls)
34       Add H_1,...,H_b except H to spls
35     else
36       Add [H,]^? H_1,...,H_b to spls except a sub-block of maximal size
37   while(T_S ≠ ∅)
38     S_i = pop(T_S)
39     touched[S_i]=false
40     forall(ρ ∈ L(R))
41       M[S_i][ρ]=0
```

Algorithm S4. The Split procedure.

The* Split *Procedure. We now provide a more detailed description of our `Split` procedure, shown in Algorithm S4. It begins (Line 2) by initializing the set of pointers T_S that will refer to all species S_i for which either there exists a multiset of species $\rho \in \mathcal{L}(R)$ such that $\mathbf{fr}(S_i, \rho, H_{sp}) \neq 0$ if $\chi = F$, or for which there exists an equivalence class of multisets $\mathcal{M}$ such that $\mathbf{br}(S_i, \mathcal{M}, H_{sp}) \neq 0$ if $\chi = B$. Similarly, Line 3 initializes the set T_H which will point to the blocks of the species in T_S. We remark that only the blocks in T_H may be split due to the current splitter H_{sp}. If $\chi = F$, Lines 4-8 compute $\mathbf{fr}(S_i, \rho, H_{sp})$ and store it in $\mathtt{M}[S_i][\rho]$ for each S_i and ρ. This is done by `ComputeFR` in Algorithm S5. The procedure scans all the reactions in the `nzs` list of each $S_j \in H_{sp}$. For each reaction $(\rho \xrightarrow{\alpha} \pi) \in \mathtt{nzs}[Y]$ we scan all pairs $(m, S_i) \in \rho$, and compute $\mathbf{fr}(S_i, \rho_{S_i}, Y_{sp})$, with ρ_{S_i} the label obtained from ρ by decreasing by one m. This ensures that all the labels of interest for the reaction are considered. We can see that the contributions given by each reaction are divided by $[\rho]!$, the multinomial coefficient of the reagents of the reaction. Also, we remark that the $ns(S_j)$ factor corresponds to the $(\pi_j - \rho_j)$ one used in the characterization of forward equivalence. The actual updates on the entries of `M` are performed in Line 5 invoking the simple sub-routine `Update` in Lines 8-12 of Algorithm S5, which also updates T_S if necessary (by using an auxiliary vector of booleans `touched`, with one entry per species). We note that the index of the column in `M` of each label ρ_{S_i} is stored in $\mathtt{pos}[S_j]$, as discussed.

If $\chi = B$, Lines 4-8 of Algorithm S4 compute $\mathbf{br}(S_i, \mathcal{M}, H_{sp})$ and store it in $\mathtt{M}[S_i][\rho_{\mathcal{M}}]$ for each $S_i \in S$ and $\mathcal{M} \in \left\{\rho \mid (\rho + S_k \xrightarrow{\alpha} \pi) \in R, S_k \in S\right\} / \approx_{\mathcal{H}}$, with $\mathcal{H}$ the current partition. The symbol $\rho_{\mathcal{M}}$ denotes one of the labels in $\mathcal{L}(R)$ such that $\rho_{\mathcal{M}} \in \mathcal{M}$, chosen as discussed below. The label $\rho_{\mathcal{M}}$ will be used in this iteration of the algorithm to univocally identify $\mathcal{M}$. The **br** rates are computed by `ComputeBR` of Algorithm S6. It is similar to `ComputeFR`, but it scans the reactions in the `out` lists of each species $S_j \in H_{sp}$ rather than in the `nzs` lists. In addition, some extra steps (Lines 4-12) are necessary to compute labels. According to the splitter-based characterization of backward equivalence, each reaction $(S_j + \rho' \xrightarrow{\alpha} \pi) \in \mathtt{out}[S_j]$ contributes rates to **br** that have the label corresponding to the equivalence class of multisets $\mathcal{M}$ to which ρ' belongs. In Lines 5-11 we build a label $\rho_{\mathcal{M}'} \in \mathcal{L}(R)$ which *characterizes* $\mathcal{M}'$ in this iteration of the algorithm. Essentially, we just replace each species in ρ' with one chosen species of its block, hence collapsing the pairs of ρ' regarding species belonging to the same block. For doing this we provide each block $H' \in \mathcal{H}$ with a field `label` used to point to a label in $\mathcal{L}(R)$ assigned to H'. This will actually be a species in $H' \cap \mathcal{L}(R)$. In particular, in Line 6 we get the block of S'_j, H'. Then, if no label is currently assigned to H', we set S'_j as *label* of H'. We do this for each $(m, S'_j) \in \rho'$,* and add $(m, H'.label)$ to $\rho_{\mathcal{M}'}$. In case an entry $(m', H'.label)$ already exists in $\rho_{\mathcal{M}'}$, it is replaced with $(m + m', H'.label)$. Each insertion costs $\mathcal{O}(\log p_r)$ time, because $\rho_{\mathcal{M}'}$ is sorted with respect to the species field. Once $\rho_{\mathcal{M}'}$ has been computed, in Line 12 its column in `M` is obtained in $\mathcal{O}(p_r \cdot \log l)$ time.

We are finally ready to invoke the `update` routine to update the $\mathtt{M}[S_i][\rho_{\mathcal{M}}]$ entry of each species S_i appearing as reagent or product in the considered reaction (Lines 13-16). Reagents are decreased by their multiplicity times the rate of the reaction, while products are increased by the same amount. Note that all the reactions such that the considered ρ' belongs to the same label $\mathcal{M}'$ will contribute to the same $\rho_{\mathcal{M}'}$ column of `M`. Therefore, this computes the summation over all the multisets ρ such that $\rho \in \mathcal{M}'$ considered in the splitter-based characterization of backward equivalence, see Eq. (7). We note that in Lines 7-8, for each reaction we count the number of different reagent species belonging to the splitter block H_{sp}. The rate of the reaction is then divided by counter ct in Lines 14 and 16, as required by the splitter-based notion of backward equivalence.

Now that T_S and `M` have been populated, Lines 10-16 of Algorithm S4 build T_H and mark all species in T_S as discussed in the section about data structures. Note that Line 13 discards species in T_S whose `M`-rows have only zeros. This can happen because positive and negative values can sum up to zero (see, e.g., lines 14 and 16 of Algorithm S6). In addition, Line 12 reinitializes all `label` fields of the blocks in T_H, a super-set of those to which `ComputeBR` might have assigned a label.

It is now possible to refine $\mathcal{H}$ and update the list of candidate splitters. This is done by splitting each block $H \in T_H$ according to the computed **fr** or **br** values (Lines 17-36). As discussed in the section about data structures, the split of a block unmarks its species at no extra computational cost. Lines 19-20 perform the discussed split operation. In particular they split (in constant time) the species in H which appear in T_H (the marked ones) from those which do not appear in T_H (the unmarked ones). Those $S_i \in H$ that yield $\mathtt{M}[S_i][\cdot] \neq 0$ form the block H_1, while the others remain in H. If the new H is empty, then H_1 contains the same elements originally present in H and thus receives its identity. Otherwise H_1 is made to a new block in $O(|H_1|)$ time.

Lines 25-27 further split H_1 by moving some of its elements in a new block H_2 in $O(|H_1|)$ time. In particular, we calculate the `pmc`-row in order to split H_1 into (a new) H_1 and H_2. If more than half of the species of the original H_1 have equal values in their `M`-row, the new block H_1 will contain those species with the `pmc`-row; otherwise, the new block will contain any sub-set of H_1 with same row in `M`. In both cases the obtained H_1 does not need to be further

*We actually scan the pairs $(m, S'_j) \in (S_j + \rho')$. Then, if $S'_j = S_j$ we consider $(m - 1, S'_j)$, and we ignore the pair if $m - 1 = 0$.

```
1  ComputeFR(S_j, M) :=
2    forall((ρ -α-> π, ns(S_j), [ρ]!, pos) ∈ nzs[S_j])
3      forall((m, S_i) ∈ ρ)
4        //Add fr(S_i, ρ_{S_i}, S_j) to M[S_i, ρ_{S_i}]
5        Update(S_i, pos[S_i], (α·ns(S_j))/[ρ]!)
6
7  //Sub-routine to update M and T_S
8  Update(S_i, col, val) :=
9    if(not(touched[S_i]))
10     touched[S_i] = true
11     add S_i to T_S
12   M[S_i][col] = M[S_i][col] + val
```

Algorithm S5. Compute **fr** wrt the splitters.

```
1  ComputeBR(S_j, H_sp, M) :=
2    forall((S_j + ρ' -α-> π) ∈ out[S_j])
3      ct = 1
4      ρ_{M'} = emptylist
5      forall((m, S'_j) ∈ ρ')
6        H' = get block of S'_j
7        if(H' = H_sp and S'_j ≠ S_j)
8          ct = ct + 1
9        if(H' does not have a label)
10         H'.label = S'_j
11       add (m, S'_j.label) to ρ_{M'}
12     col = position of ρ_{M'} in L(R)
13     forall((m, S'_j) ∈ (S_j + ρ'))
14       Update(S'_j, col, (-α·m)/ct)
15     forall((m, S_i) ∈ π)
16       Update(S_i, col, (α·m)/ct)
```

Algorithm S6. Compute **br** wrt the splitters.

split. Instead H_2 might need to be split further. We note that H_2 might be empty, meaning that there was no need in splitting H_1. In such case H_1 remains unchanged; in the opposite case, instead, H_2 is split in Lines 31-32 and the obtained sub-blocks are added to $\mathcal{H}$. We remark that we are guaranteed that each sub-block of H_2 has at most half the elements originally in H. Moreover, it is worth noting that splitting blocks in $\mathcal{H}$ affects `spls` because `spls` stores pointers to the elements of $\mathcal{H}$.

Finally, we add the so obtained sub-blocks to `spls` by storing the corresponding pointers in `spls`. As discussed, we do not add a sub-block with maximal size if the original H has already been used as splitter (Line 36). Note that $[H,]^? H_1$ means that we add only one of the two blocks to `spls` if Line 22 gave the identity of H to H_1. Instead, in Line 34 there is no need to add the new H to `spls` because it is already there (i.e., the original H was there, and hence the refined H inherited its presence). The procedure terminates by resetting the vector `touched`, used to build T_S, and the rows of `M` regarding the species in T_S.

Theorem 3 (Algorithm complexity)**.** *Algorithm S1 calculates the coarsest forward/backward equivalences that refine an input partition $\mathcal{H}$. Its time complexity is $\mathcal{O}(r \cdot p^2 \cdot l \cdot \log s) \leq \mathcal{O}\big(r^2 \cdot p^3 \cdot \log(s)\big)$, while its space complexity is $\mathcal{O}(a_r + r \cdot p \cdot s)$.*

Proof. The proof lifts the ideas of (17) to RNs. As in (17), we extend Algorithm S1 by an adjacent species $X_\perp$, a set of compound blocks $\mathcal{C}$, and two additional operations. In particular, we add the command $\mathcal{C} = \{S, \{X_\perp\}\}$ after the command on Line 5 and we add $\mathcal{C} = (C \setminus \{C_{H_{sp}}\}) \cup \{H_{sp}, C_{H_{sp}} \setminus H_{sp}\}$ after the command on Line 7. At the beginning of each iteration of the while loop on Line 6, any compound block $C \in \mathcal{C}$ on Line 7 can be represented as a unique union of blocks from the current partition $\mathcal{H}$. Here, $C_{H_{sp}}$ refers to the unique compound block such that $H_{sp} \subseteq C_{H_{sp}}$. By replacing statements $W(s_1, B') = W(s_2, B')$ from (17) with $\mathbf{fr}(X_1, \cdot, H_{sp}) = \mathbf{fr}(X_1, \cdot, H_{sp})$ (or $\mathbf{br}(X_1, \cdot, H_{sp}) = \mathbf{br}(X_1, \cdot, H_{sp})$), Lemma 1 from (17) carries over in a verbatim manner, thus showing that our algorithm is correct if and only if its extended version is correct. The same applies to Lemma 2 from (17) which ensures that the algorithm "does not split too much", meaning that the coarsest forward or backward equivalence that refines the partition is returned by the algorithm. By repeating the argumentation from (17), we show the first invariant: At each new iteration on Line 7, each compound block $C \in \mathcal{C}$ can be written as a union of blocks from the current partition $\mathcal{H}$ up to one single block from *spls*. That is, for each $C \in \mathcal{C}$, there exist unique blocks $H_1, \ldots, H_k \in \mathcal{H}$ such that $C = H_1 \cup \ldots \cup H_k$ with $H_1, \ldots, H_{k-1} \in$ *spls* and $H_k \notin$ *spls*. Using this invariant, we copy-paste the proof for the termination from (17). Afterwards, we observe that the proof of the second invariant carries over, where the second invariant reads as: For any $H \in \mathcal{H}$, $X_1, X_2 \in H$ and $C \in \mathcal{C}$, it holds that $\mathbf{fr}(X_1, \cdot, C) = \mathbf{fr}(X_1, \cdot, C)$ (or $\mathbf{br}(X_1, \cdot, C) = \mathbf{br}(X_1, \cdot, C)$). Using the first and the second invariant, the proof of correctness follows as in (17).

We now turn to the proof of complexity. We first consider the complexity of the algorithm without considering the `Init` procedure. Space complexity has been already considered. Arguing as in (17), one can show that any $S_i \in S$ appears at most $\lfloor \log(s) + 1 \rfloor$ times in a splitter. Thus, if $\chi = F$, Algorithm S1 needs $\mathcal{O}(\sum_{S_i \in S} \log(s) \cdot |\mathtt{nzs}[S_i]| \cdot p_r) \leq \mathcal{O}(\log(s) \cdot r \cdot p^2)$ number of steps if we ignore lines 13, 25, 31 and 40 in Algorithm S4. The p_r factor comes from the fact that `ComputeFR` has to iterate over the pairs $(m, S_i) \in \rho$.

Instead, if $\chi = B$, Algorithm S1 needs $\mathcal{O}(\sum_{S_i \in S} \log(s) \cdot |\mathtt{out}[S_i]| \cdot p) = \mathcal{O}(\log(s) \cdot r \cdot p_r \cdot p) \leq \mathcal{O}(\log(s) \cdot r \cdot p^2)$

number of steps if we ignore lines 13, 25, 31 and 40 in Algorithm S4, as well as lines 11 and 12 of Algorithm S6. The p_r factor comes from the fact that `out` has size $\mathcal{O}(r \cdot p_r)$. By extending the possible majority candidate from (17) to vectors and by observing that comparing and rewriting vectors in $\mathbb{R}^l$ needs $\mathcal{O}(l)$ steps, we conclude that Algorithm S1 performs at most $\mathcal{O}(l \cdot \log(s) \cdot r \cdot p^2)$ steps if we ignore the sorting on Line 31 in Algorithm S4. Note that lines 11 and 12 of Algorithm S6 actually contribute only an extra $\log(p_r)$ and $\log(l)$ factor per step, which is absorbed by the l one of the other lines.

In order to see that the sorting does not cost too much, we proceed as in (17) and introduce the concept of middle blocks. A sub-block H_i that was created by splitting some $H \in H_T$ where H, in turn, was added to H_T by processing H_{sp}, is called middle block if $\mathcal{R}(X, \cdot, H_{sp}) \neq 0$ and $\mathcal{R}(X, \cdot, C_{H_{sp}} \setminus H_{sp}) \neq 0$ for all $X \in H_i$, with $\mathcal{R} \in \{\mathbf{fr}, \mathbf{br}\}$. Let M_j be the union of all middle blocks that are created through splitting during the j-th iteration of the while loop on Line 6 in Algorithm S1 and let J denote the total number of iterations performed by the while loop on Line 6 in Algorithm S1. By arguing as in (17), it can be shown that it suffices to prove that sorting of blocks of size $2|M_1|, \ldots, 2|M_J|$ does not exceed our complexity bound. We achieve this by generalizing the argumentation of (17) as follows.

Denoting the number of middle blocks where a species S_i appears by $\#_m(S_i) := |\{1 \leq j \leq J \mid S_i \in M_j\}|$, the cumulative size of all middle blocks during the calculation of the algorithm is given by $\sum_{j=1}^{J} |M_j| = \sum_{S_i \in S} \#_m(S_i)$. In order to estimate $\#_m(S_i)$, we introduce the quantities

$$\#_c^F(S_i) := |\{C \in \mathcal{C} \mid \exists \lambda \in \mathcal{L}(R)\ \exists S_j \in C(\mathbf{fr}(S_i, \lambda, S_j) \neq 0)\}|$$
$$\#_c^B(S_i) := |\{C \in \mathcal{C} \mid \exists \lambda \in \mathcal{L}(R)\ \exists S_j \in C(\mathbf{br}(S_i, \mathcal{M}, S_j) \neq 0),\ \text{with } \mathcal{M} \text{ the equivalence class of } \lambda\}|$$

Note that backward equivalence actually uses different labels than forward equivalence. However, the number of labels used in the backward case is bound by $\mathcal{L}(R)$, as discussed. Note also that $\#_c^\chi(S_i)$ depends on $\mathcal{C}$, where $\chi \in \{F, B\}$. Thus, since each new iteration refines $\mathcal{C}$, the value $\#_c^\chi(S_i)$ can only increase during the course of the algorithm. Also, we remark that $\sum_{S_i \in S} \#_c^\chi(S_i) \leq 2 \cdot r \cdot p^2$. In fact, for every reaction (r), we can have at most p_r combinations of species S_i and labels λ, and $p_r + p_p$ species S_j leading to non-zero $\mathbf{fr}$ and $\mathbf{br}$.

Let us assume that $\chi = F$. By definition, any element $S_k \in H_i$ of a middle block H_i that is created during the j-th iteration of the while loop satisfies $\mathbf{fr}(S_k, \cdot, H_{sp}) \neq 0$ and $\mathbf{fr}(S_k, \cdot, C_{H_{sp}} \setminus H_{sp}) \neq 0$, meaning that there exist $\lambda, \lambda' \in \mathcal{L}(R)$ such that $\mathbf{fr}(S_k, \lambda, H_{sp}) \neq 0$ and $\mathbf{fr}(S_k, \lambda', C_{H_{sp}} \setminus H_{sp}) \neq 0$. Consequently, a splitting of $C_{H_{sp}}$ into H_{sp} and $C_{H_{sp}} \setminus H_{sp}$ increases $\#_m(S_k)$ by one and respects $\#_m(S_k) \leq \#_c^F(S_k)$. This yields $\sum_{S_i \in S} \#_m(S_i) \leq \sum_{S_i \in S} \#_c^F(S_i) \leq 2 \cdot r \cdot p^2$ in the case of $\chi = F$.

Instead, if $\chi = B$, any element $S_k \in H_i$ of a middle block H_i that is created during the j-th iteration of the while loop satisfies $\mathbf{br}(S_k, \mathcal{M}', H_{sp}) \neq 0$ and $\mathbf{br}(S_k, \mathcal{M}'', C_{H_{sp}} \setminus H_{sp}) \neq 0$ for some equivalence classes of multisets $\mathcal{M}', \mathcal{M}''$. Note that $\mathcal{C}$ and $\mathcal{H}$ refers to the value present in iteration $1 \leq j \leq J$; in particular, in contrast to the case $\chi = F$, the set of labels of $\mathbf{br}$ may vary from iteration to iteration. The last statement ensures, however, the existence of at least a $\lambda' \in \mathcal{L}(R)$ such that $\lambda' \in \mathcal{M}'$ that provides a non-zero contribution to $\mathbf{br}(S_k, \mathcal{M}', H_{sp})$. That is, such that

$$\sum_{S_l \in H_{sp}} \frac{\phi(S_l + \lambda', S_k)}{|S_l + \lambda'|_{H_{sp}}} \neq 0\,.$$

Similarly, the last statement also ensures the existence of at least a $\lambda'' \in \mathcal{L}(R)$ such that $\lambda'' \in \mathcal{M}''$ that provides a non-zero contribution to $\mathbf{br}(S_k, \mathcal{M}'', C_{H_{sp}} \setminus H_{sp})$. This, in turn, implies the existence of $S_p \in H_{sp}$ and $S_s \in C_{H_{sp}} \setminus H_{sp}$ such that λ' and λ'' provide a non-zero contribution to, respectively, $\mathbf{br}(S_k, \mathcal{M}', S_p)$ and $\mathbf{br}(S_k, \mathcal{M}'', S_s)$. Consequently, a splitting of $C_{H_{sp}}$ into H_{sp} and $C_{H_{sp}} \setminus H_{sp}$ increases $\#_m(S_k)$ by one and respects $\#_m(S_k) \leq \#_c^B(S_k)$. This yields $\sum_{S_k \in S} \#_m(S_k) \leq \sum_{S_k \in S} \#_c^B(S_k) \leq 2 \cdot r \cdot p^2$ in the case of $\chi = B$.

Armed with the bound $\sum_{S_i \in S} \#_m(S_i) \leq 2 \cdot r \cdot p^2$, we extend the argumentation of (17) as follows. Since $|M_j| \leq s$ for all $1 \leq j \leq J$, the cumulative cost of the sorting is given by $\mathcal{O}(\sum_{j=1}^{J} l 2|M_j| \log(2|M_j|)) \leq \mathcal{O}(\sum_{j=1}^{J} l|M_j| \log(2s)) \leq \mathcal{O}\big(l \cdot r \cdot p^2 \cdot (\log(s) + \log(2))\big) = \mathcal{O}\big(l \cdot r \cdot p^2 \cdot \log s\big)$. As in the calculation of the possible majority candidate before, the factor l arises from vector comparison.

Hence, the time complexity of Algorithm S1 is $\mathcal{O}\big(l \cdot r \cdot p^2 \cdot \log s\big)$ if ignoring the `Init` procedure, which has been shown to have $\mathcal{O}(r \cdot p_r^2 \cdot \log l + s \cdot l)$ time complexity. Given that both s and l can be bound by $\mathcal{O}(r \cdot p)$, we can bound the time complexity of Algorithm S1 by $\mathcal{O}\big(r \cdot p^2 \cdot l \cdot \log(s)\big) \leq \mathcal{O}\big(r^2 \cdot p^3 \cdot \log(s)\big)$. □

S1.4 Reduced Reaction Network

The partition-refinement algorithm discussed in the previous section provides a partition of species that satisfies the forward/backward equivalence criteria. To obtain a reduced model, we now provide a notion of *reduced RN* up to a

forward/backward equivalence $\mathcal{H}$, where each species is associated with an equivalence class. From the reduced RN, a reduced ODE system can be straightforwardly obtained applying mass-action kinetics. For doing this we use the notion of *choice function* $\mu : S \to S$ of a partition $\mathcal{H}$, which associates each species with a representative of its block in $\mathcal{H}$. The notion of choice function can be easily generalized to sets and multisets of species. Also, we denote by $\#(S_i)$ the size of the block of $\mathcal{H}$ that contains S_i.

Definition 1. *Let (S, R) be an RN and $\mathcal{H}$ a partition of S. Then, the reduction of (S, R) up to $\mathcal{H}$ is $(S, R)^{\mathcal{H}} := (\mu(S), R^{\mathcal{H}})$, where $R^{\mathcal{H}}$ is obtained with the following algorithm.*

(0) Let $\mathcal{R}$ be a multiset of reactions, initialized with the empty multiset.

(1) For each reaction $(\rho \xrightarrow{\alpha} \pi) \in R$, add the following reaction to $\mathcal{R}$:

$$\mu(\rho) \xrightarrow{\alpha'} \mu(\pi), \qquad \text{with} \quad \alpha' = \frac{\alpha}{\prod_{S_i \in S} \#(S_i)^{\rho_i}}.$$

(2) $R^{\mathcal{H}} = \{\rho \xrightarrow{m \cdot \alpha} \pi \mid (m, \rho \xrightarrow{\alpha} \pi) \in \mathcal{R}\}$.

Essentially, step (1) transforms reactions between species into reactions between the representatives of the equivalence classes. The original rate is scaled down by the product of the sizes of the blocks containing the reagents. Instead, step (2) merges all reactions that have the same reagents, products, and rate after the transformation (see for instance Fig. 2 in the main text); the pair $(m, \rho \xrightarrow{\alpha} \pi)$ indicates that there are m occurrences of reaction $\rho \xrightarrow{\alpha} \pi$ in multiset $\mathcal{R}$.

Theorem 4 (Reduction complexity)**.** *Let (S, R) be an RN, $\mathcal{H}$ a forward or backward equivalence and μ its choice function. Then, $(S, R)^{\mathcal{H}}$ is computed in at most $\mathcal{O}\big(|R| \cdot |S| \cdot \log(|R| \cdot |S|)\big)$ steps.*

Proof. Computing the set $\mu(S)$ requires $\mathcal{O}(|S|)$ time, assuming that species are provided with a pointer to their block, which in turn has a pointer to its representative. Also, we assume that each block is provided with a boolean flag, used to add only once the representative species of each block $H \in \mathcal{H}$.

Step (1) requires to iterate (once) the reactions, taking $\mathcal{O}(|R|)$ time. In particular, for each reaction we have in turn to iterate its reagents and products to apply μ. This requires $\mathcal{O}(|S|)$ time. Assuming that the size of each block of the species in ρ is also stored within the block, the computation of the scaling factor requires just to iterate (the blocks of) the species in ρ. This requires again $\mathcal{O}(|S|)$ time per reaction. Finally, in order to perform efficiently step (2), we sort the computed reagents and products, taking $\mathcal{O}(|S| \cdot \log(|S|))$ time. To sum up, step (1) requires $\mathcal{O}\big(|R| \cdot |S| \cdot \log |S|\big)$ time.

Instead, step (2) can be computed by first sorting the reactions obtained from (1), requiring $\mathcal{O}(|R| \cdot \log(|R|) \cdot |S|)$ time, where the $|S|$ factor comes from the fact that in order to compare two reactions it is necessary to scan once their (sorted) reagents and products (and their rate). Then, (2) is completed by iterating (once) the reactions to actually collapse them, taking $\mathcal{O}(|R| \cdot |S|)$ time. □

We now formalize the relation existing between the notions of forward equivalence and reduced RN. We first relate the derivatives in the original model with those in the reduced one.

Proposition 4. *Let (S, R) be an RN, $\mathcal{H} = \{H_1, \ldots, H_m\}$ be a forward equivalence and μ its choice function. Let $\mathcal{P}$ and $\hat{\mathcal{P}}$ be the vectors of polynomials in Eq. (1) from the main text of the ODEs underlying (S, R) and $(S, R)^{\mathcal{H}}$, respectively. Then it holds:*

$$\sum_{S_i \in H} \mathcal{P}_i(x) = \hat{\mathcal{P}}_H\big(\sum_{S_i \in H_1} x_i, \ldots, \sum_{S_i \in H_m} x_i\big)$$

for all $H \in \mathcal{H}$ and $x \in \mathbb{R}^S_{\geq 0}$.

Proof. We want to show that

$$\sum_{S_i \in H} \sum_{(\rho \xrightarrow{\alpha} \pi) \in R} (\pi_i - \rho_i) \cdot \alpha \cdot \prod_{S_j \in S} x_j^{\rho_j} = \sum_{(\rho \xrightarrow{\alpha} \pi) \in R} \frac{(\mu(\pi)_{Y^H} - \mu(\rho)_{Y^H})}{\prod_{S_k \in S} \#(S_k)^{\rho_k}} \cdot \alpha \cdot \prod_{H' \in \mathcal{H}} x_{H'}^{\mu(\rho)_{Y^{H'}}} \qquad \text{[SE7]}$$

where $\mu(\cdot)_{Y^H}$ denotes the multiplicity of the representative species of block H in the multiset $\mu(\cdot)$. We note that $\mu(\rho')_{Y^H} = \sum_{S_l \in H} \rho'_l$ for any $\rho' \in \mathcal{MS}(S)$. For $a_r := \max\{|\rho| \mid (\rho \xrightarrow{\alpha} \pi) \in R\}$ the maximum arity of reagents in R, the left-hand-side of Eq. (SE7) can be rewritten as:

$$\sum_{n=1}^{a_r} \sum_{H_1 \in \mathcal{H}} x_{H_1} \sum_{H_2 \in \mathcal{H}} x_{H_2} \cdots \sum_{H_n \in \mathcal{H}} x_{H_n} \cdot \frac{\phi(\sum_{l=1}^{n} Y^{H_l}, H)}{[\sum_{l=1}^{n} Y^{H_l}]!}$$

The right-hand-side of Eq. (SE7) can be rewritten as:

$$\sum_{n=1}^{a_r} \sum_{\rho \in \mathcal{MS}_n(S)} \frac{\prod_{H' \in \mathcal{H}} x_{H'}^{\mu(\rho)_{Y^{H'}}}}{\prod_{S_k \in S} \#(S_k)^{\rho_k}} \sum_{(\rho \xrightarrow{\alpha} \pi) \in R} (\sum_{S_l \in H} \pi_l - \sum_{S_l \in H} \rho_l) \cdot \alpha = \sum_{n=1}^{a_r} \sum_{\rho \in \mathcal{MS}_n(S)} \frac{\prod_{H' \in \mathcal{H}} x_{H'}^{\mu(\rho)_{Y^{H'}}}}{\prod_{S_k \in S} \#(S_k)^{\rho_k}} \cdot \phi(\rho, H) \qquad \text{[SE8]}$$

We now focus on one $1 \leq n \leq a_r$ only of the right-hand side of Eq. (SE8). We recall that for each $\rho \in \mathcal{MS}_n(S)$ there exist $[\rho]!$ entries in S^n equal to ρ up to permutation, obtaining:

$$\begin{aligned}
\sum_{\rho \in \mathcal{MS}_n(S)} \frac{\prod_{H' \in \mathcal{H}} x_{H'}^{\mu(\rho)_{Y^{H'}}}}{\prod_{S_k \in S} \#(S_k)^{\rho_k}} \cdot \phi(\rho, H) &= \sum_{\rho \in S^n} \frac{\prod_{H' \in \mathcal{H}} x_{H'}^{\mu(\rho)_{Y^{H'}}}}{\prod_{S_k \in S} \#(S_k)^{\rho_k}} \cdot \frac{\phi(\rho, H)}{[\rho]!} \\
&= \sum_{\rho \in S^n} \frac{\prod_{H' \in \mathcal{H}} x_{H'}^{\mu(\rho)_{Y^{H'}}}}{\prod_{S_k \in S} \#(S_k)^{\mu(\rho)_k}} \cdot \frac{\phi(\mu(\rho), H)}{[\mu(\rho)]!} \qquad \text{(by Proposition 1)} \qquad \text{[SE9]}
\end{aligned}$$

For every $\rho \in \mathcal{MS}_n(S)$, the set of multisets $\{\rho' \mid \mu(\rho') = \mu(\rho)\}$ has size $\prod_{S_k \in S} \#(S_k)^{\mu(\rho)_k}$. Also, Eq. (SE9) does not depend on which elements from such sets are considered. We can hence rewrite Eq. (SE9) as:

$$\sum_{\substack{\rho \in S^n \\ \rho = \mu(\rho)}} \left(\prod_{H' \in \mathcal{H}} x_{H'}^{\mu(\rho)_{Y^{H'}}} \right) \cdot \frac{\phi(\mu(\rho), H)}{[\mu(\rho)]!} = \sum_{\substack{\rho \in S^{n-1} \\ \rho = \mu(\rho)}} \sum_{H_n \in \mathcal{H}} x_{H_n} \cdot \left(\prod_{H' \in \mathcal{H}} x_{H'}^{\mu(\rho)_{Y^{H'}}} \right) \cdot \frac{\phi(\rho + Y^{H_n}, H)}{[\rho + Y^{H_n}]!}$$

The same can be repeated for all remaining $n-1$ entries of the cartesian product S^{n-1}, obtaining

$$\sum_{H_1 \in \mathcal{H}} x_{H_1} \sum_{H_2 \in \mathcal{H}} x_{H_2} \cdots \sum_{H_n \in \mathcal{H}} x_{H_n} \cdot \frac{\phi(\sum_{i=1}^{n} Y^{H_i}, H)}{[\sum_{i=1}^{n} Y^{H_i}]!}$$

This closes the proof. □

Theorem 5 (Forward Reduced RN)**.** *Let (S, R) be an RN, and $\mathcal{H}$ be a forward equivalence. Let S_{H_j} be the species in $(S, R)^{\mathcal{H}}$ corresponding to block H_j for any $H_j \in \mathcal{H}$. Let x_i denote the ODE variable for species S_i in (S, R) and $\hat{x}_H$ the variable of the species S_H in $(S, R)^{\mathcal{H}}$. Then, for all $H \in \mathcal{H}$ it holds*

$$\sum_{S_i \in H} x_i(t) = \hat{x}_H(t) \qquad \text{at all time points } t$$

for all $\hat{x}(0)$ such that $\hat{x}_{H'}(0) = \sum_{S_k \in H'} x_k(0)$ for all $H' \in \mathcal{H}$.

Proof. The statement follows directly from Proposition 4. □

We now move our attention towards the relation existing between the notions of backward equivalence and reduced RN.

Proposition 5. *Let (S,R) be an RN, $\mathcal{H}=\{H_1,\dots,H_m\}$ be an equivalence over S and μ a choice function. Let $\mathcal{P}$ and $\hat{\mathcal{P}}$ be the vectors of polynomials in Eq. (1) from the main text of the ODEs underlying (S,R) and $(S,R)^{\mathcal{H}}$, respectively. Then it holds:*

$$\sum_{S_i\in H}\mathcal{P}_i(x)=\hat{\mathcal{P}}_H\Big(\sum_{S_i\in H_1}x_i,\dots,\sum_{S_i\in H_m}x_i\Big)$$

for all $H\in\mathcal{H}$ and $x\in\mathbb{R}^S_{\geq 0}$ that are uniform on $\mathcal{H}$.

Proof. We say that $x(t)\in\mathbb{R}^S_{\geq 0}$ is *uniform* on $\mathcal{H}$ whenever $x_i(t)=x_j(t)$ for all $H'\in\mathcal{H}$ and $S_i,S_j\in H'$. Also, we remark that for any two $\rho,\rho'\in\mathcal{MS}(S)$ we write $\rho\approx_{\mathcal{H}}\rho'$ if the two multisets have same number of species of each block of $\mathcal{H}$. Hence, for any $\mathcal{M}\in\mathcal{MS}(S)/\approx_{\mathcal{H}}$ and $H\in\mathcal{H}$ we use $\mathcal{M}_H$ to denote $\sum_{S_i\in H}\rho_i$, with $\rho\in\mathcal{MS}(S)$. We want to show that

$$\sum_{S_i\in H}\sum_{(\rho\xrightarrow{\alpha}\pi)\in R}(\pi_i-\rho_i)\cdot\alpha\cdot\prod_{S_j\in S}x_j^{\rho_j}=\sum_{(\rho\xrightarrow{\alpha}\pi)\in R}\frac{(\mu(\pi)_{Y^H}-\mu(\rho)_{Y^H})}{\prod_{S_k\in S}\#(S_k)^{\rho_k}}\cdot\alpha\cdot\prod_{H'\in\mathcal{H}}x_{H'}^{\mu(\rho)_{Y^{H'}}}\qquad\text{[SE10]}$$

where $\mu(\cdot)_{Y^H}$ denotes the multiplicity of the representative species of block H in the multiset $\mu(\cdot)$. We note that $\mu(\rho')_{Y^H}=\sum_{S_l\in H}\rho'_l$ for any $\rho'\in\mathcal{MS}(S)$.

The left-hand-side of Eq. (SE10) can be rewritten as:

$$\begin{aligned}
\sum_{S_i\in H}\sum_{\rho\in\mathcal{MS}(S)}\phi(\rho,S_i)\cdot\prod_{S_j\in S}x_j^{\rho_j}&=\sum_{S_i\in H}\sum_{\rho\in\mathcal{MS}(S)}\phi(\rho,S_i)\cdot\prod_{H'\in\mathcal{H}}\prod_{S_j\in H'}x_j^{\rho_j}\\
&=\sum_{S_i\in H}\sum_{\rho\in\mathcal{MS}(S)}\phi(\rho,S_i)\cdot\prod_{H'\in\mathcal{H}}\prod_{S_j\in H'}x_{Y^{H'}}^{\rho_j}&&\text{(by } x \text{ being uniform on } \mathcal{H})\\
&=\sum_{S_i\in H}\sum_{\mathcal{M}\in\mathcal{MS}(S)/\approx_{\mathcal{H}}}\sum_{\rho\in\mathcal{M}}\phi(\rho,S_i)\cdot\prod_{H'\in\mathcal{H}}x_{Y^{H'}}^{\sum_{S_j\in H'}\rho_j}&&\text{[SE11]}\\
&=\sum_{S_i\in H}\sum_{\mathcal{M}\in\mathcal{MS}(S)/\approx_{\mathcal{H}}}\sum_{\rho\in\mathcal{M}}\phi(\rho,S_i)\cdot\prod_{H'\in\mathcal{H}}x_{Y^{H'}}^{\mathcal{M}_{H'}}&&\text{(see below)}\\
&=\sum_{S_i\in H}\sum_{\mathcal{M}\in\mathcal{MS}(S)/\approx_{\mathcal{H}}}\left(\left(\prod_{H'\in\mathcal{H}}x_{Y^{H'}}^{\mathcal{M}_{H'}}\right)\cdot\sum_{\rho\in\mathcal{M}}\phi(\rho,S_i)\right)\\
&=\sum_{S_i\in H}\sum_{\mathcal{M}\in\mathcal{MS}(S)/\approx_{\mathcal{H}}}\left(\left(\prod_{H'\in\mathcal{H}}x_{Y^{H'}}^{\mathcal{M}_{H'}}\right)\cdot\phi(\mathcal{M},S_i)\right)\\
&=\sum_{\mathcal{M}\in\mathcal{MS}(S)/\approx_{\mathcal{H}}}\left(\prod_{H'\in\mathcal{H}}x_{Y^{H'}}^{\mathcal{M}_{H'}}\right)\cdot\sum_{S_i\in H}\phi(\mathcal{M},S_i)\\
&=\sum_{\mathcal{M}\in\mathcal{MS}(S)/\approx_{\mathcal{H}}}\left(\prod_{H'\in\mathcal{H}}x_{Y^{H'}}^{\mathcal{M}_{H'}}\right)\cdot\phi(\mathcal{M},H)
\end{aligned}$$

where we rewrote Eq. (SE11) thanks to the fact that for any $\rho\in\mathcal{M}$ we have same $\mathcal{M}_{H'}=\sum_{S_j\in H'}\rho_j$.

Instead, the right-hand-side of Eq. (SE10) can be rewritten as:

$$\begin{aligned}
\sum_{(\rho\xrightarrow{\alpha}\pi)\in R}\Big(\sum_{S_i\in H}\pi_i-\sum_{S_i\in H}\rho_i\Big)\cdot\alpha\cdot\frac{\prod_{H'\in\mathcal{H}}x_{H'}^{\mu(\rho)_{Y^{H'}}}}{\prod_{S_k\in S}\#(S_k)^{\rho_k}}&=\sum_{\mathcal{M}\in\mathcal{MS}(S)/\approx_{\mathcal{H}}}\sum_{\rho\in\mathcal{M}}\left(\frac{\prod_{H'\in\mathcal{H}}x_{H'}^{\mu(\rho)_{Y^{H'}}}}{\prod_{S_k\in S}\#(S_k)^{\rho_k}}\right)\cdot\phi(\rho,H)&&\text{[SE12]}\\
&=\sum_{\mathcal{M}\in\mathcal{MS}(S)/\approx_{\mathcal{H}}}\sum_{\rho\in\mathcal{M}}\left(\frac{\prod_{H'\in\mathcal{H}}x_{H'}^{\mathcal{M}_{H'}}}{\prod_{H'\in\mathcal{H}}|H'|^{\mathcal{M}_{H'}}}\right)\cdot\phi(\rho,H)&&\text{(see below)}\\
&=\sum_{\mathcal{M}\in\mathcal{MS}(S)/\approx_{\mathcal{H}}}\left(\frac{\prod_{H'\in\mathcal{H}}x_{H'}^{\mathcal{M}_{H'}}}{\prod_{H'\in\mathcal{H}}|H'|^{\mathcal{M}_{H'}}}\right)\cdot\phi(\mathcal{M},H)&&\text{[SE13]}
\end{aligned}$$

where we rewrote the right-hand side of Eq. (SE12) due that the facts that: (numerator) $\mu(\rho)_{Y^{H'}} = \sum_{S_j \in H'} \rho_j = \mathcal{M}_{H'}$ for any $\rho \in \mathcal{M}$; (denumerator) $\#(S_z) = \#(S_{z'})$ for any $H'' \in \mathcal{H}$ and $S_z, S_{z'} \in H'$.

Considering Eq. (SE13), we can rewrite $\prod_{H' \in \mathcal{H}} x_{H'}^{\mathcal{M}_{H'}}$ as $\prod_{H' \in \mathcal{H}} |H'|^{\mathcal{M}_{H'}} \cdot \prod_{H' \in \mathcal{H}} x_{Y^{H'}}^{\mathcal{M}_{H'}}$. In fact:

$$\prod_{H' \in \mathcal{H}} x_{H'}^{\mathcal{M}_{H'}} = \prod_{H' \in \mathcal{H}} \Big(\sum_{S_k \in H'} x_k \Big)^{\mathcal{M}_{H'}} = \prod_{H' \in \mathcal{H}} \big(|H'| \cdot x_{Y^{H'}} \big)^{\mathcal{M}_{H'}} = \prod_{H' \in \mathcal{H}} \big(|H'|^{\mathcal{M}_{H'}} \cdot x_{Y^{H'}}^{\mathcal{M}_{H'}} \big) = \prod_{H' \in \mathcal{H}} |H'|^{\mathcal{M}_{H'}} \cdot \prod_{H' \in \mathcal{H}} x_{Y^{H'}}^{\mathcal{M}_{H'}}$$

Hence, we can close the proof by rewriting Eq. (SE13) as

$$\sum_{\mathcal{M} \in \mathcal{MS}(S)/\approx_{\mathcal{H}}} \left(\left(\prod_{H' \in \mathcal{H}} x_{Y^{H'}}^{\mathcal{M}_{H'}} \right) \cdot \phi(\mathcal{M}, H) \right)$$

□

The following theorem states that whenever the partition $\mathcal{H}$ considered in Proposition 5 is a backward equivalence, we can recover all information in (S, R) from $(S, R)^{\mathcal{H}}$.

Theorem 6 (Backward Reduced RN)**.** *Let (S, R) be an RN, and $\mathcal{H}$ be a backward equivalence. Let S_{H_j} be the species in $(S, R)^{\mathcal{H}}$ corresponding to block H_j, for any $H_j \in \mathcal{H}$. Let x_i denote the ODE variable for species S_i in (S, R), and $\hat{x}_H$ the variable for S_H in $(S, R)^{\mathcal{H}}$. Then, for all $H \in \mathcal{H}$ and $S_i \in H$ it holds:*

$$x_i(t) = \frac{\hat{x}_H(t)}{|H|} \qquad \text{at all time points } t$$

for all $x(0)$ uniform on $\mathcal{H}$, and $\hat{x}(0)$ such that $\hat{x}_{H'}(0) = \sum_{S_k \in H'} x_k(0)$ for all $H' \in \mathcal{H}$.

Proof. The statement follows directly from Proposition 5, and by the fact that $\mathcal{H}$ is a backward equivalence. In particular, the latter guarantees that if $x(0)$ is uniform on $\mathcal{H}$, then $x(t)$ is uniform on $\mathcal{H}$ at all time points t, allowing to apply Proposition 5 for all t. □

Theorem 7 (Correspondence of reduced RN and reduced ODE System)**.** *Let (S, R) be an RN and $\mathcal{H}$ be an equivalence relation. Let $\mathcal{P}$ and $\hat{\mathcal{P}}$ be the vectors of polynomials in Eq. (1) from the main text of the ODEs underlying (S, R) and $(S, R)^{\mathcal{H}}$, respectively. Then, it holds that $\hat{\mathcal{P}} = \tilde{\mathcal{P}}$ when $\tilde{\mathcal{P}}$ is as in Eq. (4) from the main text.*

Proof. By Proposition 4, it holds that

$$\sum_{S_i \in H} \mathcal{P}_i(x) = \hat{\mathcal{P}}_H\Big(\sum_{S_i \in H_1} x_i, \ldots, \sum_{S_i \in H_m} x_i \Big)$$

for all $H \in \mathcal{H}$ and all $x \in \mathbb{R}^S$. From this, we infer that $\hat{\mathcal{P}}(y) = \hat{\mathcal{P}}A(x) = A\mathcal{P}(x) = A\mathcal{P}\overline{A}A(x) = A\mathcal{P}\overline{A}(y)$ if $y = Ax$, where $A = (a_{ij})$ is the aggregation matrix underlying $\mathcal{H}$ and $\overline{A} = (\overline{a}_{ij})$ is given by $\overline{a}_{ij} = a_{ji}/(\sum_k a_{jk})$. In the last computation, the third equality follows from (20). To see also the backward case, we first note that Proposition 5 ensures

$$\sum_{S_i \in H} \mathcal{P}_i(x) = \hat{\mathcal{P}}_H\Big(\sum_{S_i \in H_1} x_i, \ldots, \sum_{S_i \in H_m} x_i \Big)$$

for all $H \in \mathcal{H}$ and all $x \in \mathbb{R}^S$ that are uniform on $\mathcal{H}$. With this, it holds that $A\mathcal{P}\overline{A}(y) = A\mathcal{P}(x) = \hat{\mathcal{P}}A(x) = \hat{\mathcal{P}}A\overline{A}(y) = \hat{\mathcal{P}}(y)$, where the first and the third equalities follow from the choice of $\overline{A}$ and the fact that x is uniform on $\mathcal{H}$. □

S2. Case Studies

With reference to the section "Applications" in the main text, we hereby further demonstrate the applicability of forward and backward equivalence to biological models from the literature, and comment on the nature of the obtained reductions. In particular, we discuss the following:

- models of multisite protein phosphorylation and equivalent binding sites in mechanisms of complex formation (Section S2.1);

- models of site interactions that are controlled or dependent upon other sites, which cannot be found through domain-specific reduction that exploit assumptions of independence within interaction domains (Section S2.2);
- forward equivalence for the model of early events for the signaling pathway of FCϵRI (32) (Section S2.3);
- backward equivalence for a model with Michaelis-Menten kinetics, using a translation into a polynomial ODE system (Section S2.4);
- forward and backward equivalence in the *continuous homologous* of discrete Boolean models of regulatory networks through a polynomial ODE encoding (Section S2.5).

The BioNetGen language. Sections S2.1, S2.2 and S2.3 consider models written in BioNetGen (49), a rule-based language for modeling protein-interaction networks. In this language, chemical reaction networks (CRNs) with mass-action kinetics are specified intensionally in terms of interactions among basic molecules. For example, consider a kinase $E(s)$ and a substrate protein $S(p1, p2)$, with s, $p1$ and $p2$ being the *binding sites* of the two molecules. Then, the term:

$$E(s!1).S(p1\sim P, p2\sim U!1)$$

represents the complex species formed upon the binding of E and S via s and $p2$ (denoted by the '.' operator linking molecule E and S, and '!1' which tags the sites through which the binding occurs). Each binding site might be provided with an internal state, denoted by $\sim$, used e.g., to represent its phosphorylation ($\sim P$) or unphosphorylation ($\sim U$). The dynamics of the system are specified in terms of rules that can be understood as an interaction pattern between units of complexes that may occur in any molecular context. For example, a rule of form:[†]

```
E(s) + S(p1~U) <-> E(s!1).S(p1~U!1)  k1, k2
```
[SE14]

describes reversible binding (indicated by the double arrow `<->`) with rates k1 and k2, respectively, between the kinase E and any substrate protein S with unphosphorylated and unbound binding site $p1$, independently on the status or binding of $p2$. Instead, a rule of form:

```
S(p1~U!+) -> S(p1~P!+)  k3
```
[SE15]

describes the fact that S phosphorylates in $p1$ whenever bound with other molecules via $p1$. In other words, '`!+`' acts as a wild-card allowing to omit the molecule with which S is bound.

Starting from a set of *seed species*, i.e., basic complexes, the rules are exhaustively applied to enumerate all possible complexes and their reactions as a mass-action CRN. For example, consider the following three of seed species: $E(s)$, $S(p1\sim U, p2\sim U)$, and $E(s!1).S(p1\sim U, p2\sim P!1)\}$. Then, Eq. (SE14) generates the reactions (and species):

$$\begin{aligned}
E(s) + S(p1\sim U, p2\sim U) &\xrightarrow{k1} E(s!1).S(p1\sim U!1, p2\sim U),\\
E(s!1).S(p1\sim U!1, p2\sim U) &\xrightarrow{k2} E(s) + S(p1\sim U, p2\sim U),\\
E(s) + E(s!1).S(p1\sim U, p2\sim P!1) &\xrightarrow{k1} E(s!2).E(s!1).S(p1\sim U!2, p2\sim P!1),\\
E(s!2).E(s!1).S(p1\sim U!2, p2\sim P!1) &\xrightarrow{k1} E(s) + E(s!1).S(p1\sim U, p2\sim P!1).
\end{aligned}$$

Instead, Eq. (SE15) generates:

$$\begin{aligned}
E(s!1).S(p1\sim U!1, p2\sim U) &\xrightarrow{k3} E(s!1).S(p1\sim P!1, p2\sim U),\\
E(s!2).E(s!1).S(p1\sim U!2, p2\sim P!1) &\xrightarrow{k3} E(s!2).E(s!1).S(p1\sim P!2, p2\sim P!1)
\end{aligned}$$

Reproducibility and tool support: ERODE. The reduction techniques presented in the main text and further discussed in this manuscript have been implemented in ERODE (50), a tool for the evaluation and reduction of ordinary differential equations. ERODE is available together with a manual and examples at http://sysma.imtlucca.it/tools/erode/.

All experiments presented in this paper have been performed using ERODE. For models written in BioNetGen we provide links to the original BioNetGen files (extension .bngl), which can be compiled in its underlying CRN (extension .net) using the BioNetGen tool. ERODE supports .net files generated using version 2.2.5 of BioNetGen available at http://mmbios.org/index.php/bionetgen-2-2-5-stable. Such generated .net files can be imported into ERODE using specifications as in Listing S7. The models in Sections S2.4 and S2.5 are provided directly as ERODE specifications, which include the command to reproduce the discussed reductions.

[†] In this manuscript, we will use verbatim form to refer to BioNetGen rules, and math form to refer to reactions of reaction networks.

```
1 begin model importedFromBNG
2   importBNG(fileIn="bngFile.net")
3   reduceNFB(reducedFile="bngFileFE.ode")
4   reduceNBB(reducedFile="bngFileBE.ode")
5 end model
```

Listing S7. ERODE snippet code to import and reduce BioNetGen files. Line 2 loads a .net file (using a path relative to that of the ERODE file itself). Then, Line 3 and Line 4 reduce the CRN up to forward and backward equivalence, respectively. In both cases the obtained reduced CRN is stored in the file specified in the `reducedFile` parameter.

S2.1 Equivalent-site Assumption in Biochemical Models

Here we detail how forward and backward equivalences may reveal the assumption of equivalent sites in a number of computational models of protein interaction networks. We first study multisite protein phosphorylation, which was exemplified with the CRN in Fig. 1 in the main text. We consider both a linear ODE model and a mass-action one. Then we present an instance in a rule-based model of the Mitogen Activated Protein Kinase (MAPK) pathway. Finally, using a model of insulin-receptor binding, we show how a similar aggregation can be found in mechanistic models of complex formation, when a receptor protein features multiple binding sites.

Linear multisite protein phosphorylation. We consider the linear ODE model of multisite protein phosphorylation studied in (21, Supplementary Doc S2). It describes the kinetics of a protein Y with $n = 3$ sites that can phosphorylate/dephosphorylate independently of each other (i.e., according to the *random mechanism*). This leads to an ODE system with $2^n = 8$ equations, each tracking the concentration of proteins with a specific configuration of its sites. We denote each such configuration by subscripting Y with a triplet $a_1a_2a_3$ where each a_i can be either '0' (dephosphorylated) or 'P' (phosphorylated). The ODE system can be written thus:

$$\begin{aligned}
\dot{Y}_{000} &= -(\alpha_{x00} + \alpha_{0x0} + \alpha_{00x})Y_{000} + \beta_{x00}Y_{P00} + \beta_{0x0}Y_{0P0} + \beta_{00x}Y_{00P} \\
\dot{Y}_{P00} &= \alpha_{x00}Y_{000} - (\alpha_{Px0} + \alpha_{P0x} + \beta_{x00})Y_{P00} + \beta_{Px0}Y_{PP0} + \beta_{P0x}Y_{P0P} \\
\dot{Y}_{0P0} &= \alpha_{0x0}Y_{000} - (\alpha_{xP0} + \alpha_{0Px} + \beta_{0x0})Y_{0P0} + \beta_{xP0}Y_{PP0} + \beta_{0Px}Y_{0PP} \\
\dot{Y}_{00P} &= \alpha_{00x}Y_{000} - (\alpha_{x0P} + \alpha_{0xP} + \beta_{00x})Y_{00P} + \beta_{x0P}Y_{P0P} + \beta_{0xP}Y_{0PP} \\
\dot{Y}_{PP0} &= \alpha_{Px0}Y_{P00} + \alpha_{xP0}Y_{0P0} - (\alpha_{PPx} + \beta_{xP0} + \beta_{Px0})Y_{PP0} + \beta_{PPx}Y_{PPP} \\
\dot{Y}_{P0P} &= \alpha_{x0P}Y_{00P} + \alpha_{P0x}Y_{P00} - (\alpha_{PxP} + \beta_{x0P} + \beta_{P0x})Y_{P0P} + \beta_{PxP}Y_{PPP} \\
\dot{Y}_{0PP} &= \alpha_{0xP}Y_{00P} + \alpha_{0Px}Y_{0P0} - (\alpha_{xPP} + \beta_{0xP} + \beta_{0Px})Y_{0PP} + \beta_{xPP}Y_{PPP} \\
\dot{Y}_{PPP} &= \alpha_{PPx}Y_{PP0} + \alpha_{PxP}Y_{P0P} + \alpha_{xPP}Y_{0PP} - (\beta_{PPx} + \beta_{PxP} + \beta_{xPP})Y_{PPP}
\end{aligned}$$

where the α's and the β's denote the phosphorylation and dephosphorylation rates, respectively. In the subscripts, 'x' indicates the position of the site which is involved in the reaction; for instance, α_{x0P} is the phosphorylation rate of the first site when the protein is in configuration Y_{00P}. The mathematical simplification that these rates are independent from the specific location of the involved site translated into setting:

$$\begin{aligned}
\alpha_{x00} &= \alpha_{0x0} = \alpha_{00x} \equiv \alpha_1 \\
\alpha_{Px0} &= \alpha_{P0x} = \alpha_{xP0} = \alpha_{0Px} = \alpha_{0xP} = \alpha_{x0P} \equiv \alpha_2 \\
\alpha_{PPx} &= \alpha_{PxP} = \alpha_{xPP} = \alpha_3 \\
\beta_{x00} &= \beta_{0x0} = \beta_{00x} \equiv \beta_1 \\
\beta_{Px0} &= \beta_{P0x} = \beta_{xP0} = \beta_{0Px} = \beta_{0xP} = \beta_{x0P} \equiv \beta_2 \\
\beta_{PPx} &= \beta_{PxP} = \beta_{xPP} = \beta_3
\end{aligned}$$

Under these conditions, a self-consistent ODE system with variables Y_0, Y_1, Y_2, Y_3 is given in (21, Supplementary Doc S2) which tracks the following sums of solutions

$$Y_0 := Y_{000} \qquad Y_1 := Y_{P00} + Y_{0P0} + Y_{00P} \qquad Y_2 := Y_{PP0} + Y_{P0P} + Y_{0PP} \qquad Y_3 := Y_{PPP}$$

Indeed, we can confirm the ad hoc lumping scheme in (21, Supplementary Doc S2) by proving that $\{\{Y_{000}\}, \{Y_{P00}, Y_{0P0}, Y_{00P}\}, \{Y_{P00}, Y_{0P0}, Y_{00P}\}\}$ is a forward equivalence. (In addition, the same partition is also a backward equivalence.)

Model	*Original CRN*		*Reduced CRN*	
	$\|S\|$	$\|R\|$	$\|S\|$	$\|R\|$
$E2$	18	48	12	24
$E3$	66	288	22	60
$E4$	258	1536	37	120
$E5$	1026	7680	58	210
$E6$	4098	36864	86	336
$E7$	16386	172032	122	504
$E8$	65538	786432	167	720
$E9$	262146	3538944	222	990

Table S1. Maximal forward/backward reductions for the multisite phosphorylation models of (33). En **is the model with** n **independent and identical binding sites;** $|S|$ **= number of species;** $|R|$ **= number of reactions. For each model, the computed forward and backward equivalences coincide.**

It aggregates proteins that have equal states of the phosphorylation sites up to permutation, essentially explaining that the identity of equivalent sites may be abstracted away in the reduced model.

Mass-action multisite phosphorylation We now consider a family of synthetic benchmark models published in (33) to study the performance of a network-free simulator for CRNs. It consists of a collection of multisite phosphorylation models where a substrate protein, S, can be phosphorylated and dephosphorylated by a kinase, E, and phosphatase, D. This basic pattern occurs in different models where n, the number of independent phosphorylation sites, is varied between 2 and 9, see (33, Supplementary Note 7).[‡] We denote by En the model with n sites. Since each site has 4 distinct states (phosphorylated and unbound, unphosphorylated and unbound, phosphorylated and bound to a phosphatase, unphosphorylated and bound to a kinase) these dynamics lead to CRNs with $4^n + 2$ species and $6n(4^{n-1})$ reactions. Binding is assumed to take place according to mass-action dynamics, leading to an underlying ODE systems with polynomial derivatives of degree two. The assumption of equivalent sites is implemented in the model by having rate parameters that are independent of the identity of the site involved in the reaction.

Table S1 shows the results of the computation of the maximal forward/backward aggregations for these models, obtained by initializing the partition-refinement algorithm with the trivial singleton partition where all original species are included in one block. Both forward and backward reductions find the same coarsest partition. A manual inspection of the as-obtained equivalence classes reveals that the equivalences aggregate complexes that have equal states of the phosphorylation sites up to permutation. For instance, using BioNetGen, the input modeling language used in (33), a sample equivalence class for model $E2$ is given by:

$$\left\{E(s!1).S(p1\sim P, p2\sim U!1), E(s!1).S(p1\sim U!1, p2\sim P)\right\}$$

It describes the equivalence between two complexes where the substrate protein S has one phosphorylated site, and one unphosphorylated site to which the kinase E is bound.

Double phosphorylation in MAPK pathways. In (25) the authors study idealized models of the Mitogen Activated Protein Kinase (MAPK) pathway, which transmits signals using a linear three-level sequence of kinases, MAP3K, MAP2K, and MAPK, where the latter two are known to require double phosphorylation, e.g., (51). The *Scaff-22* model of (25) describes the series of cascade events where phosphorylation occurs at a scaffold protein that is able to recruit kinases from all tiers. Similarly to the previous model, the rates of activation of the doubly-phosphorylated kinases are equal for both the phosphorylation sites and the order with which they phosphorylate (when the kinase is bound to the scaffold protein) is random. The resulting CRN consists of 85 species and 487 reactions. The corresponding source BioNetGen file is described in Appendix B of (25).

The maximal forward equivalence explains the assumption of independence between the phosphorylation sites of MAP2K/MAP3K, yielding a reduced CRN with 56 species and 264 reactions. The largest equivalence class collapses distinct chemical species in the original model consisting of a scaffold protein that is bound to both MAP2K and

[‡] As stated in (33), the corresponding source BioNetGen files are available at http://www.nfsim.org.

MAPK with equal numbers of phosphorylated sites, as in the following sample equivalence class:

$$\big\{MAP2K(R1\sim Y,R2\sim Yp,s!1).MAPK(R1\sim Yp,R2\sim Y,s!2).Scaff(map2k!1,map3k,mapk!2),$$
$$MAP2K(R1\sim Y,R2\sim Yp,s!1).MAPK(R1\sim Y,R2\sim Yp,s!2).Scaff(map2k!1,map3k,mapk!2),$$
$$MAP2K(R1\sim Yp,R2\sim Y,s!1).MAPK(R1\sim Yp,R2\sim Y,s!2).Scaff(map2k!1,map3k,mapk!2),$$
$$MAP2K(R1\sim Yp,R2\sim Y,s!1).MAPK(R1\sim Y,R2\sim Yp,s!2).Scaff(map2k!1,map3k,mapk!2)\big\}.$$

Here, using the BioNetGen notation, the equivalence class relates species where both the MAP2K and the MAPK kinase have one phosphorylated site (either $R1$ or $R2$, denoted by state Yp).

Insulin receptor binding. Insulin receptors are dimers that can bind two distinct molecules of insulin. An assumption of equivalence between the two binding sites must however take into account negative cooperativity, i.e., that the binding affinity for the second molecule is lower than that for the first molecule (52). The following is an excerpt from the CRN of a model of early events in insulin and EGF signaling presented in (53, 54) which focuses on the mechanistic description of insulin/receptor (reversible) binding only:

$$Ir(b1,b2)+Ins(s)\xrightarrow{k_1}Ir(b1!1,b2).Ins(s!1)$$
$$Ir(b1!1,b2).Ins(s!1)\xrightarrow{k_{-1}}Ir(b1,b2)+Ins(s)$$
$$Ir(b1,b2)+Ins(s)\xrightarrow{k_1}Ir(b1,b2!1).Ins(s!1)$$
$$Ir(b1,b2!1).Ins(s!1)\xrightarrow{k_{-1}}Ir(b1,b2)+Ins(s)$$
$$Ir(b1!1,b2).Ins(s!1)+Ins(s)\xrightarrow{k_2}Ir(b1!1,b2!2).Ins(s!1).Ins(s!2)$$
$$Ir(b1!1,b2!2).Ins(s!1).Ins(s!2)\xrightarrow{k_{-2}}Ir(b1!1,b2).Ins(s!1)+Ins(s)$$
$$Ir(b1,b2!1).Ins(s!1)+Ins(s)\xrightarrow{k_2}Ir(b1!1,b2!2).Ins(s!1).Ins(s!2)$$
$$Ir(b1!1,b2!2).Ins(s!1).Ins(s!2)\xrightarrow{k_{-2}}Ir(b1,b2!1).Ins(s!1)+Ins(s)$$

The different affinities are modeled by assuming that $k_1\neq k_2$ and $k_{-1}\neq k_{-2}$. Under these conditions, forward and backward equivalences still reveal the symmetry in the two binding sites, $b1$ and $b2$, yielding for instance the equivalence class $\{Ir(b1,b2!1).Ins(s!1),Ir(b1!1,b2).Ins(s!1)\}$. This reduction pattern occurs in all complexes of the signaling pathway of (53, 54) that are equal up to the identity of the site involved in the insuline/receptor binding. In the full original model (53, Supplementary File 2), consisting of 2768 species and 38320 reactions, we found 52 occurrences of such a pattern.

S2.2 Controlled Interaction Domains

In models of protein-interaction networks with molecular complexes that exhibit internal states, dependencies may occur among such states. This has been discussed in the main text with regard to the oligomerization model for EFGR (28) and for the FcϵRI signaling pathway of (32). In this section we provide detailed information on how such situations of controlled interaction domains block the use of domain-specific model-reduction techniques, since these crucially exploit independence of dynamics across different sites (5–7, 30).

For this, here we consider a simple model of *ordered phosphorylation* taken from (6, Supplement 6), where the ability of one site being phosphorylated depends on the state of other sites of the same complex. The model is defined in BioNetGen, and is reported in Table S2.

This model consists of a substrate protein, S, with two phosphorylation sites, Y_1 and Y_2, and a receptor R with two activation sites, a_1 and a_2. The activation is governed by spontaneous and independent events (rules R1-R2). The receptor may reversibly bind to S regardless of the local states of both R and S (rule R3). However, the phosphorylation of Y_1/Y_2 is governed by A's activation states: when R is bound to S and a_1 is active, then R may phosphorylate S at Y_1 (rule R4); the phosphorylation of Y_2 requires that both a_2 be active and Y_1 phosphorylated (rule R5). Similarly to deactivation of a_1/a_2, dephosphorylation of Y_1/Y_2 is modeled as a spontaneous reaction that is independent of the state of all other sites (rules R6-R7). The full CRN generated from such BioNetGen model consists of 24 species and 88 reactions.

This model is used in (6) as an example to show that the domain-specific reduction technique therein introduced (operating at the rule-based language level) may yield incorrect results when the input model exhibits site dependency.

Id	Rule	Rates
R1	`R(a1~U) <-> R(a1~A)`	`0.40, 0.60`
R2	`R(a2~U) <-> R(a2~A)`	`0.60, 0.40`
R3	`R(s) + S(r) <-> R(s!1).S(r!1)`	`0.03, 1.00`
R4	`R(s!1,a1~A).S(r!1,Y1~Y) -> R(s!1,a1~A).S(r!1,Y1~pY)`	`0.30`
R5	`R(s!1,a2~A).S(r!1,Y1~pY,Y2~Y) -> R(s!1,a2~A).S(r!1,Y1~pY,Y2~pY)`	`0.70`
R6	`S(Y1~pY) -> S(Y1~Y)`	`0.70`
R7	`S(Y2~pY) -> S(Y2~pY)`	`0.30`

Table S2. BioNetGen model of ordered phosphorylation from (6, Supplement 6).

Instead, we applied our algorithm to the CRN obtained from the BioNetGen model in Table S2 where all activation and phosphorylation events are modeled with distinct, site-specific kinetic parameters. The maximal forward equivalence reveals the following three equivalence classes:

1. $R := \{\ R(a1{\sim}U, a2{\sim}U, s),\ R(a1{\sim}A, a2{\sim}U, s),\ R(a1{\sim}U, a2{\sim}A, s),\ R(a1{\sim}A, a2{\sim}A, s)\ \}$
2. $S := \{\ S(Y1{\sim}Y, Y2{\sim}Y, r),\ S(Y1{\sim}pY, Y2{\sim}Y, r),\ S(Y1{\sim}pY, Y2{\sim}pY, r),\ S(Y1{\sim}Y, Y2{\sim}pY, r)\ \}$
3. $R.S := \{$
   $R(a1{\sim}U, a2{\sim}U, s!1).S(Y1{\sim}Y, Y2{\sim}Y, r!1),\ R(a1{\sim}A, a2{\sim}U, s!1).S(Y1{\sim}Y, Y2{\sim}Y, r!1),$
   $R(a1{\sim}U, a2{\sim}A, s!1).S(Y1{\sim}Y, Y2{\sim}Y, r!1),\ R(a1{\sim}A, a2{\sim}A, s!1).S(Y1{\sim}Y, Y2{\sim}Y, r!1),$
   $R(a1{\sim}A, a2{\sim}U, s!1).S(Y1{\sim}pY, Y2{\sim}Y, r!1),\ R(a1{\sim}U, a2{\sim}U, s!1).S(Y1{\sim}pY, Y2{\sim}Y, r!1),$
   $R(a1{\sim}A, a2{\sim}A, s!1).S(Y1{\sim}pY, Y2{\sim}Y, r!1),\ R(a1{\sim}U, a2{\sim}A, s!1).S(Y1{\sim}pY, Y2{\sim}Y, r!1)$
   $R(a1{\sim}A, a2{\sim}A, s!1).S(Y1{\sim}pY, Y2{\sim}pY, r!1),\ R(a1{\sim}U, a2{\sim}A, s!1).S(Y1{\sim}pY, Y2{\sim}pY, r!1)$
   $R(a1{\sim}A, a2{\sim}U, s!1).S(Y1{\sim}pY, Y2{\sim}pY, r!1),\ R(a1{\sim}U, a2{\sim}U, s!1).S(Y1{\sim}pY, Y2{\sim}pY, r!1)$
   $R(a1{\sim}A, a2{\sim}A, s!1).S(Y1{\sim}Y, Y2{\sim}pY, r!1),\ R(a1{\sim}U, a2{\sim}A, s!1).S(Y1{\sim}Y, Y2{\sim}pY, r!1)$
   $R(a1{\sim}A, a2{\sim}U, s!1).S(Y1{\sim}Y, Y2{\sim}pY, r!1),\ R(a1{\sim}U, a2{\sim}U, s!1).S(Y1{\sim}Y, Y2{\sim}pY, r!1)\ \}$

They aggregate, respectively: i) all unbound forms R, i.e., regardless of the states of its a_1/a_2 activation sites; ii) all unbound forms of S, i.e., regardless of the states of its Y_1/Y_2 activation sites; all bound R-S complexes, regardless of the states of any of theirs sites. Intuitively, in this model forward equivalence internalizes the ordered phosphorylation events (which take place only when the complex is formed) within the S-R equivalence class. Indeed, the reduced RN computed according to the algorithm discussed in the main text and in Section S1.4 gives two equations only

$$R + S \xrightarrow{k_3} R.S \qquad\qquad R.S \xrightarrow{k_{-3}} R + S$$

which describe the macroscopic binding/unbinding mechanism governing the three equivalence classes (at the same rates k_3, k_{-3} as those in the original model).

S2.3 FcϵRI Model

Here we detail the relationship between the reduction by forward equivalence and the simplified version model of the model of early events for the signaling pathway of FCϵRI, published in (32) and discussed in the main text. The same model (name `fceri_ji.bngl`) and variants thereof are presented in (33) to study the performance of a network-free simulation for rule-based systems.

Table S3 shows the rule-based interactions for the original model using the BioNetGen syntax. For instance, rule R1 specifies a reversible ligand-binding reaction, which represents FcϵRI and IgE, respectively. This rule may subsume a reaction between the seed species $Rec(a)$ and $Lig(l, l)$ (together with its corresponding reverse one), i.e.:

$$Rec(a) + Lig(l, l) \xrightarrow{kp1} Rec(a!1).Lig(l!1, l), \quad Rec(a!1).Lig(l!1, l) \xrightarrow{km1} Rec(a) + Lig(l, l),$$

This generates the new species $Rec(a!1).Lig(l!1, l)$ where the receptor is bound to the ligand. To this new species, rule R2 may apply because, as discussed, `Lig(l,l!+)` in its left hand side matches any molecular complex containing a ligand that is bound through one site only, while the other is free. Applying rule R2 to the seed species $Rec(a)$ and

Id	*Rule*	*Rates*
R1	`Rec(a) + Lig(l,l) <-> Rec(a!1).Lig(l!1,l)`	`kp1, km1`
R2	`Rec(a) + Lig(l,l!+) <-> Rec(a!2).Lig(l!2,l!+)`	`kp2, km2`
R3	`Rec(b~Y) + Lyn(U,SH2) <-> Rec(b~Y!1).Lyn(U!1,SH2)`	`kpL, kmL`
R4	`Lig(l!1,l!2).Lyn(U!3,SH2).Rec(a!2,b~Y!3).Rec(a!1,b~Y) ->` `Lig(l!1,l!2).Lyn(U!3,SH2).Rec(a!2,b~Y!3).Rec(a!1,b~pY)`	`pLb`
R5	`Lig(l!1,l!2).Lyn(U!3,SH2).Rec(a!2,b~Y!3).Rec(a!1,g~Y) ->` `Lig(l!1,l!2).Lyn(U!3,SH2).Rec(a!2,b~Y!3).Rec(a!1,g~pY)`	`pLg`
R6	`Rec(b~pY) + Lyn(U,SH2) <-> Rec(b~pY!1).Lyn(U,SH2!1)`	`kpLd, kmLs`
R7	`Lig(l!1,l!2).Lyn(U,SH2!3).Rec(a!2,b~pY!3).Rec(a!1,b~Y) ->` `Lig(l!1,l!2).Lyn(U,SH2!3).Rec(a!2,b~pY!3).Rec(a!1,b~pY)`	`pLbs`
R8	`Lig(l!1,l!2).Lyn(U,SH2!3).Rec(a!2,b~pY!3).Rec(a!1,g~Y) ->` `Lig(l!1,l!2).Lyn(U,SH2!3).Rec(a!2,b~pY!3).Rec(a!1,g~pY)`	`pLgs`
R9	`Rec(g~pY) + Syk(tSH2) <-> Rec(g~pY!1).Syk(tSH2!1)`	`kpS, kmS`
R10	`Lig(l!1,l!2).Lyn(U!3,SH2).Rec(a!2,b~Y!3).Rec(a!1,g~pY!4).Syk(tSH2!4,l~Y) ->` `Lig(l!1,l!2).Lyn(U!3,SH2).Rec(a!2,b~Y!3).Rec(a!1,g~pY!4).Syk(tSH2!4,l~pY)`	`pLS`
R11	`Lig(l!1,l!2).Lyn(U,SH2!3).Rec(a!2,b~pY!3).Rec(a!1,g~pY!4).Syk(tSH2!4,l~Y) ->` `Lig(l!1,l!2).Lyn(U,SH2!3).Rec(a!2,b~pY!3).Rec(a!1,g~pY!4).Syk(tSH2!4,l~pY)`	`pLSs`
R12	`Lig(l!1,l!2).Syk(tSH2!3,a~Y).Rec(a!2,g~pY!3).Rec(a!1,g~pY!4).Syk(tSH2!4,a~Y) ->` `Lig(l!1,l!2).Syk(tSH2!3,a~Y).Rec(a!2,g~pY!3).Rec(a!1,g~pY!4).Syk(tSH2!4,a~pY)`	`pSS`
R13	`Lig(l!1,l!2).Syk(tSH2!3,a~pY).Rec(a!2,g~pY!3).Rec(a!1,g~pY!4).Syk(tSH2!4,a~Y) ->` `Lig(l!1,l!2).Syk(tSH2!3,a~pY).Rec(a!2,g~pY!3).Rec(a!1,g~pY!4).Syk(tSH2!4,a~pY)`	`pSSs`
R14	`Rec(b~pY) -> Rec(b~Y)`	`dm`
R15	`Rec(g~pY) -> Rec(g~Y)`	`dm`
R16	`Syk(tSH2!+,l~pY) -> Syk(tSH2!+,l~Y)`	`dm`
R17	`Syk(tSH2!+,a~pY) -> Syk(tSH2!+,a~Y)`	`dm`
R18	`Syk(tSH2,l~pY) -> Syk(tSH2,l~Y)`	`dc`
R19	`Syk(tSH2,a~pY) -> Syk(tSH2,a~Y)`	`dc`

Table S3. Model of early events for the signaling pathway of FCϵRI, originally published in (32), where the actual rate values can be retrieved. Variants are proposed in (33) in order to study a network-free simulator for rule-based models. Removing the rules in red yields the exactly reduced model `fceri_ji_red.bngl` of (32) which abstracts from the phosphorylation state of `Syk`'s site `l`. The maximal forward equivalence abstracts from both phosphorylation sites of `Syk`. Its reduced CRN is equivalent to the one obtained by removing the red and the blue rules from the model.

$Rec(a!1).Lig(l!1,l)$ will yield a new species, $Rec(a!1).Rec(a!2).Lig(l!1,l!2)$, that represents the IgE dimer recruiting both FCϵRI units.

As discussed in the main text, the full CRN for this model consists of 354 species and 3 680 reactions. The maximal forward equivalence aggregates species that have equal structure up to the phosphorylation status of both `Syk`'s sites `a` and `l` (representing the activation loop and linker, respectively). The assumption of independence mentioned in the main text between the dynamics of `Syk`'s sites and those that lead to a structural change in the complex can be explained by visual inspection of the rules. Indeed, none of the rules that lead to a change of structure (i.e., R1, R2, R3, R6, and R9) specify a context dependent on the phosphorylation state of `Syk`'s sites; conversely, none of the rules related to `Syk`'s phosphorylation events (R10, R11, R12, R13, R16, R17, R18, and R19) lead to a change of structure in the molecular complex involved.

In (33, Supplementary Note 8) the authors observe, without a formal proof, that an exactly reduced model may be obtained by abstracting away the phosphorylation state of `Syk`'s site `l`. This is achieved in the rule-based model by removing R10 and R11, which define a transphosphorylation of `l` by `Lyn` when it is free or bound to `SH2`, respectively. The resulting CRN has 172 species and 1433 reactions (name `fceri_ji_red.bngl`).

Here, we first observe that the very same CRN is obtained by also removing rules R16 and R18 because they are never fired since `Syk`'s site `l` is never phosphorylated. We can prove that this model is an exact reduction in the sense of a forward equivalence by verifying that the partition that aggregates the original model into blocks of complexes with the same structure up to the phosphorylation state of site `l` is a forward equivalence, Indeed, the reduced CRN up to such equivalence corresponds to the CRN of `fceri_ji_red.bngl`. In addition, as discussed in the main text,

the maximal forward equivalence can aggregate more, abstracting from the state of both phosphorylation sites of `Syk`, leading to a reduced CRN with 105 species and 732 reactions. This is equivalent to a CRN obtained by the original rule-based model by removing all rules that concern `Syk`'s phosphorylation events, i.e., R10–R13 and R16–R19. Both reductions can be generated using the ERODE specification provided as Supplementary File 1.

S2.4 Septation Initiation Netwrok in fission yeast

We now consider a model of the Septation Initiation Network (SIN) in fission yeast presented in (55). With this we show how backward equivalence explains the symmetric behavior analyzed in (55); in doing so, we show that our technique can be in applied in some cases of ODE models with other nonlinearities (e.g., rational expressions, sigmoids, and trigonometric functions). For this one requires simple algebraic manipulations that lead to a polynomial ODE system with additional auxiliary variables which represent algebraic constraints. However, in general care has to be taken in order to ensure the well-definedness of the auxiliary variables at all time points. For example, we study the minimal model of asymmetry established as presented in (55, Supplemetary Text S1) and shown below for completeness:

$$\dot{SIN}_{old} = k_{S_{on}}(SIN_{tot} - SIN_{new} - SIN_{old}) - k_{S_{off}} Byr4_{old} SIN_{old} \quad \text{[SE16]}$$

$$\dot{SIN}_{new} = k_{S_{on}}(SIN_{tot} - SIN_{new} - SIN_{old}) - k_{S_{off}} Byr4_{new} SIN_{new} \quad \text{[SE17]}$$

$$\dot{Byr4}_{old} = (k_{bias} + k_{B_{on}}(Cdc11OldT - Cdc11_{oldP}))(Byr4_{tot} - Byr4_{new} - Byr4_{old}) - k_{B_{off}} Byr4_{old} \quad \text{[SE18]}$$

$$\dot{Byr4}_{new} = k_{B_{on}}(Cdc11NewT - Cdc11_{newP})(Byr4_{tot} - Byr4_{new} - Byr4_{old}) - k_{B_{off}} Byr4_{new} \quad \text{[SE19]}$$

$$\dot{Cdc11}_{oldP} = k_{11P} SIN_{old} \frac{Cdc11OldT - Cdc11_{oldP}}{Js1 + Cdc11OldT - Cdc11_{oldP}} - k_{11dP} \frac{Cdc11_{oldP}}{Js1 + Cdc11_{oldP}} \quad \text{[SE20]}$$

$$\dot{Cdc11}_{newP} = k_{11P} SIN_{new} \frac{Cdc11NewT - Cdc11_{newP}}{Js1 + Cdc11NewT - Cdc11_{newP}} - k_{11dP} \frac{Cdc11_{newP}}{Js1 + Cdc11_{newP}} \quad \text{[SE21]}$$

where $k_{S_{on}}$, $k_{S_{off}}$, k_{bias}, $k_{B_{on}}$, $k_{B_{off}}$, k_{11P}, k_{11dP}, and $Js1$ are given kinetic parameters; $Cdc11OldT = Cdc11NewT$ is a given concentration level of scaffold protein $Cdc11$, while SIN_{tot} and $Byr4_{tot}$ are the total concentration levels of SIN (whose members are modeled as a single variable) and $Byr4$ (which represents the limiting component of the complex $Cdc16$-$Byr4$). The subscripts 'old' and 'new' refer to components that were existing already in the mother cell and those in the daughter cell, respectively. This model features rational expressions in Equations (SE20)-(SE21), which define Michaelis-Menten kinetics for the phosphorylation and dephosphorylation of $Cdc11$ promoted by SIN (the quantity $Cdc11OldT - Cdc11_{oldP}$ represents the concentration of dephosphorylated $Cdc11$ in the mother cell).

Following for instance (2), we first translate this ODE into an equivalent one by first introducing new variables:

$$y_{old} = \frac{1}{Js1 + Cdc11OldT - Cdc11_{oldP}}$$
$$y_{new} = \frac{1}{Js1 + Cdc11NewT - Cdc11_{newP}}$$
$$z_{old} = \frac{1}{Js1 + Cdc11_{oldP}}$$
$$z_{new} = \frac{1}{Js1 + Cdc11_{newP}}.$$

This makes Equations (SE20)-(SE21) polynomial ODEs in these new variables:

$$\dot{Cdc11}_{oldP} = k_{11P} \cdot SIN_{old}(Cdc11OldT - Cdc11_{oldP}) y_{old} - k_{11dP} \cdot Cdc11_{oldP} \cdot z_{old} \quad \text{[SE22]}$$

$$\dot{Cdc11}_{newP} = k_{11P} \cdot SIN_{new}(Cdc11NewT - Cdc11_{newP}) y_{new} - k_{11dP} \cdot Cdc11_{newP} \cdot z_{new} \quad \text{[SE23]}$$

To the as-obtained equations we couple differential equations for the new variables:

$$\begin{aligned}
\dot{y}_{old} &= \frac{1}{(Js1 + Cdc11OldT - Cdc11_{oldP})^2} \dot{Cdc11}_{oldP} \\
&= y_{old}^2 \big(k_{11P} \cdot SIN_{old}(Cdc11OldT - Cdc11_{oldP}) y_{old} - k_{11dP} \cdot Cdc11_{oldP} \cdot z_{old}\big) && \text{[SE24]} \\
\dot{y}_{new} &= \frac{1}{(Js1 + Cdc11NewT - Cdc11_{newP})^2} \dot{Cdc11}_{newP} \\
&= y_{new}^2 \big(k_{11P} \cdot SIN_{new}(Cdc11NewT - Cdc11_{newP}) y_{new} - k_{11dP} \cdot Cdc11_{newP} \cdot z_{new}\big) && \text{[SE25]} \\
\dot{z}_{old} &= -\frac{1}{(Js1 + Cdc11_{oldP})^2} \dot{Cdc11}_{oldP} \\
&= -z_{old}^2 \big(k_{11P} \cdot SIN_{old}(Cdc11OldT - Cdc11_{oldP}) y_{old} - k_{11dP} \cdot Cdc11_{oldP} \cdot z_{old}\big) && \text{[SE26]} \\
\dot{z}_{new} &= -\frac{1}{(Js1 + Cdc11_{newP})^2} \dot{Cdc11}_{newP} \\
&= -z_{new}^2 \big(k_{11P} \cdot SIN_{new}(Cdc11NewT - Cdc11_{newP}) y_{new} - k_{11dP} \cdot Cdc11_{newP} \cdot z_{new}\big) && \text{[SE27]}
\end{aligned}$$

Thus, from the polynomial ODE system consisting of Equations (SE16)–(SE19), (SE22)–(SE27) one can get the solution of original ODE system of Equations (SE16)–(SE21) whenever the respective initial conditions are consistent, e.g., when $y_{old}(0) = 1/(Js1 + Cdc11OldT - Cdc11_{oldP}(0))$. This shows that initial conditions on this polynomial ODE system may not be chosen arbitrarily. For example, one must ensure that $Cdc11_{oldP}(0) \neq Js1 + Cdc11OldT$ because otherwise it would yield an undefined initial condition.

Now, on this polynomial ODE system we can check that when $k_{bias} = 0$ the partition

$$\big\{\{SIN_{old}, SIN_{new}\}, \{Byr4_{old}, Byr4_{new}\}, \{Cdc11_{oldP}, Cdc11_{newP}\}, \{y_{old}, y_{new}\}, \{z_{old}, z_{new}\}\big\}$$

is a backward equivalence. This explains the symmetry between mother and daughter cells discussed in (55) when the binding rate $k_{B_{on}}$ is not subject to a bias.

S2.5 Logic models of regulatory networks

In this section we provide more details on how both backward and forward equivalence can capture symmetries in Boolean networks.

As discussed in the main text, each species S_i in a Boolean network has associated a Boolean variable $b_i \in \{0, 1\}$ that describes its activation status. The dynamics of a species is described by a Boolean expression B_i, its *Boolean update function*, which collects the positive (promotion) and negative (inhibition) contribution that a subset of other species $\{b_{i1}, \ldots, b_{iN}\}$ has on it. Given the current state of the variables at time step t, denoted by $b_{i1}(t), \ldots, b_{iN}(t)$, the next state of b_i at step $t + 1$ is given by:

$$b_i(t + 1) = B_i(b_{i1}(t), \ldots, b_{iN}(t))$$

This is a model of synchronous Boolean networks, where all activation statuses at time step $t + 1$ are computed synchronously by evaluating each update function at time step t.

The technique from (38) gives an ODE system where each Boolean variable b_i is replaced by a real variable x_i, and each Boolean function B_i is associated with a *perfect continuous homologous* $\overline{B}_i$, a real function that coincides with B_i for values in $\{0, 1\}$. Formally, the obtained ODE system is component-wise defined for each species S_i as:

$$\dot{x}_i = \frac{\overline{B}_i(x_{i1}, \ldots, x_{iN}) - x_i}{\tau_i}$$

with $B_i(b_{i1}, \ldots, b_{iN}) = \overline{B}_i(x_{i1}, \ldots, x_{iN})$ whenever $b_{ij} = x_{ij}$ for all $j \in \{1, \ldots, N\}$. The term $\overline{B}_i$ describes the production of species S_i, while $-x_i$ is a first-order decay term. The additional parameter τ_i can be interpreted as the lifetime of species S_i. In particular, here we use the *BooleCube* continuous homologous of (38). It is obtained by multi-linearly interpolating the Boolean function, leading to an encoding with polynomial multivariate ODE systems.

When the network is analyzed according to its Boolean semantics, an important issue is the study of its attractors. Reduction techniques for Boolean networks have been developed with the objective of reducing the number of variables whilst preserving these dynamical properties, e.g., (56). Backward and forward equivalence offer an orthogonal reduction technique for the exact simplification of the dynamics arising from the polynomial ODE interpretation.

```
1  EGF       = true
2  ERBB1     = EGF
3  ERBB2     = EGF
4  ERBB3     = EGF
5  ERBB2_3   = ERBB2 ∧ ERBB3
6  ERBB1_2   = ERBB1 ∧ ERBB2
7  ERBB1_3   = ERBB1 ∧ ERBB3
8  AKT1      = (¬IGF1R ∧ ¬ERBB1 ∧ ¬ERBB2_3 ∧ ¬ERBB1_2 ∧ ERBB1_3) ∨
9              (¬IGF1R ∧ ¬ERBB1 ∧ ¬ERBB2_3 ∧ ERBB1_2) ∨
10             (¬IGF1R ∧ ¬ERBB1∧ ERBB2_3) ∨ (¬IGF1R ∧ ERBB1) ∨ (IGF1R)
11 MEK1      = (¬IGF1R ∧ ¬ERBB1 ∧ ¬ERBB2_3 ∧ ¬ERBB1_2 ∧ ERBB1_3) ∨
12             (¬IGF1R ∧ ¬ERBB1 ∧ ¬ERBB2_3 ∧ ERBB1_2) ∨
13             (¬IGF1R ∧ ¬ERBB1∧ ERBB2_3) ∨ (¬IGF1R ∧ ERBB1) ∨ (IGF1R)
14 IGF1R     = (¬ER_alpha ∧ AKT1 ∧ ¬ERBB2_3) ∨ (ER_alpha ∧ ¬ERBB2_3)
15 ER_alpha  = (¬MEK1 ∧ AKT1) ∨ (MEK1)
16 MYC       = (¬MEK1 ∧ ¬ER_alpha ∧ AKT1) ∨ (¬MEK1 ∧ ER_alpha) ∨ (MEK1)
17 CyclinD1  = (¬MEK1 ∧ ER_alpha ∧ AKT1 ∧ MYC) ∨ (MEK1 ∧ ER_alpha ∧ MYC)
18 p27       = ¬CDK4 ∧ ¬CDK2 ∧ ¬AKT1 ∧ ¬MYC
19 p21       = ¬CDK4 ∧ ¬AKT1 ∧ ¬MYC
20 CyclinE1  = MYC
21 CDK4      = ¬p21 ∧ CyclinD1 ∧ ¬p27
22 CDK6      = CyclinD1
23 CDK2      = ¬p21 ∧ ¬p27 ∧ CyclinE1
24 pRB1      = CDK4 ∧ CDK6
```

Listing S8. Boolean model for tyrosine kinase ERBB2 from (57). The left-hand side is the state of a Boolean variable at the next time step, given by the evaluation of the right-hand side proposition with the values at the current time step. In the synchronous model, all rules are applied simultaneously at every time step.

Backward equivalence. Let us consider the Boolean model for tyrosine kinase ERBB2 used in (57) to study the activation of the tumor suppressor retinoblastoma protein pRB through the ERBB-receptor. The complete model is given in Listing S8, which has been generated automatically using the Ginsim tool (36) from the Ginsim specification http://ginsim.org/node/39. The model has one *input* species, EGF (Line 1), whose activation triggers the activation of three ERBB receptors (Lines 2-4). This in turn leads to a cascade of interactions resulting in the activation of the output species $pRB1$ (Line 24).

We computed the largest backward equivalence on the BooleCube encoding of this model using an initial partition with two blocks, one containing the input EGF, and one containing the remaining species (see Supplementary File 2). This guarantees that the obtained backward reduction can be used for any value assigned to the input species, while it is assumed that the other species are all initialized with 0 (false). The algorithm returned the non-singleton equivalence classes $\{x_{\text{ERBB1}}, x_{\text{ERBB2}}, x_{\text{ERBB3}}\}$, $\{x_{\text{ERBB1\_3}}, x_{\text{ERBB2\_3}}\}$, and $\{x_{\text{ATK1}}, x_{\text{MEK1}}\}$. As in the main text, we report such equivalence classes by distinguished background colors, in Fig. S1, using the Ginsim graphical representation of the Boolean network. In particular, boxes represent species, while green and red arrows represent promotion and inhibition relations, respectively. The graphical representation abstracts from the actual update functions, and only maintains the promotion/inhibition relations. For example, Fig. S1 contains an inhibitor arrow from ERBB2_3 to IGF1R due to the term ¬ERBB2_3 appearing in the update function of IGF1R.

The computed backward equivalence can be explained in terms of the original Boolean dynamics of Listing S8. Specifically, the two species AKT1 and MEK1 have the same Boolean update function, hence they will have same value at any point in time if initialized equally. A similar argument holds for species ERBB1, ERBB2, and ERBB3. Interestingly, this in turn guarantees that also ERBB2_3, ERBB1_2 and ERBB1_3 will always have same value, as they have same update function up to the three equivalent species ERBB1, ERBB2, and ERBB3.

This "chain of symmetries" is more evident in Fig. 6 from the main text, depicting the Boolean network for T-cell receptor signaling studied in (37, 38). The figure provides the graphical representation of the Ginsim specification available at http://ginsim.org/node/78. We computed the largest backward equivalences of the BooleCube encoding of the two model variants considered in the main text starting with an initial partition with a singleton block per input variable, which are x_{CD45}, x_{CD8}, and x_{TCRlig}. The computed backward equivalences coincide with the equivalence classes depicted in Fig. 6 from the main text (see Supplementary File 3). In this model, the two species x_{JNK} and x_{NFAT} form a backward equivalence class due to a chain of equivalences of length 4 involving the backward equivalence classes $\{x_{\text{DAG}}, x_{\text{IP3}}\}$, $\{x_{\text{PKCth}}, x_{\text{Ca}}\}$ and $\{x_{\text{SEK}}, x_{\text{IKK}}, x_{\text{Calcin}}\}$. Again, this is due to the symmetries present in the

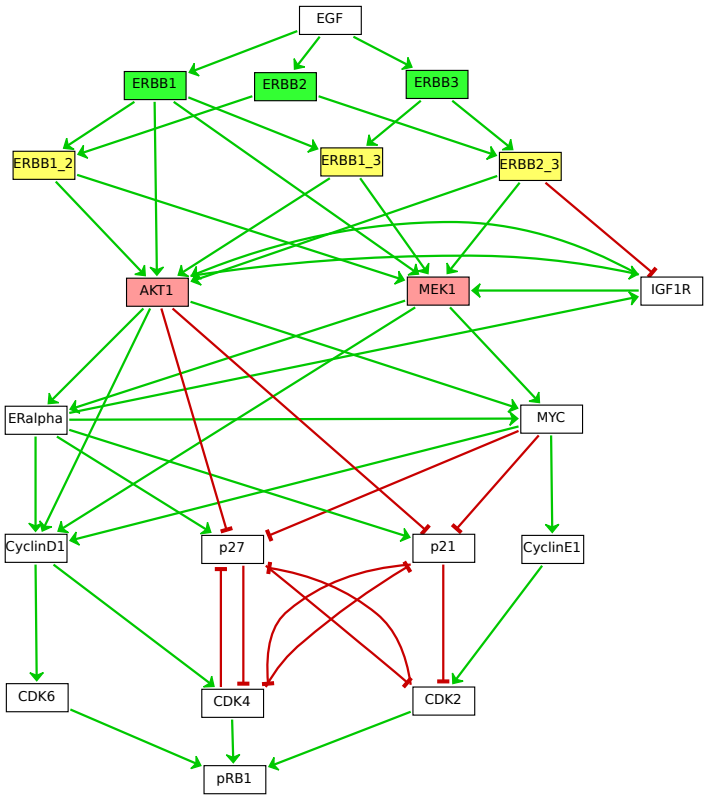

Fig. S1. Graphical representation, using (36), of the Boolean model for tyrosine kinase ERBB2 from (57). Using the BoolCube multivariate polynomial ODE encoding of (38), we fixed an initial partition where the input variables are singletons, ensuring that the largest backward equivalence that refines this partition reveals nodes with equivalent dynamics for any choice of the input values. Each non-trivial backward equivalence class is represented with colored nodes with the same background.

Boolean update functions of these nine species, provided in Listing S9, which also carry over to their BooleCube encoding, given in Listing S10. Indeed, we see that the variables in each backward equivalence class have same Boolean update functions and same derivatives, up to renaming of equivalent variables.

Another example is the model proposed in (58) to study the FcϵRI signaling pathway during mast cell activation. Fig. S2 provides a graphical representation of the model, using the Ginsim specification available at http://ginsim.org/node/180. Here, $x_{\text{NF-kB}}$ and $x_{\text{CALCINEURIN}}$ are backward equivalent due to a chain of symmetries of length three, involving the equivalence classes $\{x_{\text{DAG}}, x_{\text{IP3}}\}$ and $\{x_{\text{PKC}}, x_{\text{Ca}}\}$. As for the previous Boolean networks, we computed the largest backward equivalence of the BooleCube encoding of the model starting from an initial partition with singleton blocks for the input species, cCbl, PIP2 and Ag, obtaining the partition depicted in Fig. S2 (see Supplementary File 4).

Forward equivalence. Applying the forward equivalence criterion to ODE models of Boolean networks gives a reduced model where the variables related to non-trivial equivalence classes do not live in the domain $[0, 1]$. Nevertheless, forward equivalence can still be meaningfully applied by isolating variables of interest in initial singleton partition blocks, as discussed in the main text. Additionally, here we show that when different variables of interest are aggregated, the reduced model may still be useful for the analysis.

Fig. S3 shows the largest forward equivalence for the BoolCube encoding of the T-cell model in Fig. 6 from the main text. For both variants (with and without the feedback loops indicated by the dashed arrows) the ODE variables related to the outputs CRE, AP1, NFKB and NFAT, form a forward equivalence class. In addition, we note that also forward equivalence leads to "chains of symmetries" as discussed for backward equivalence. Intuitively, Rsk and Ca form a forward equivalence class because they affect the dynamics of variables belonging to the same forward equivalence class (respectively, CREB and Calcin) in the same way. A similar consideration holds for CREB and Calcin, which affect the dynamics of CRE and NFAT, respectively, in the same way. The outputs form a forward equivalence class because they do not affect the dynamics of other variables. Each output variable only appears in its own derivative, hence their ODEs can be trivially rewritten in terms of their sums. This can be explained by considering the update functions of the mentioned variables, shown in Listing S11, whose ODE encodings are given in Listing S12. Indeed, the ODEs in Listing S12 can be rewritten in terms of sums of variables within the mentioned

```
DAG     = PLCg_a
IP3     = PLCg_a

PKCth   = DAG
Ca      = IP3

SEK     = PKCth
IKK     = PKCth
Calcin  = Ca

JNK     = SEK
NFAT    = Calcin
```

Listing S9. Excerpt of the Boolean model for T-cell receptor signaling studied in (37, 38), depicted in Fig. 6 from the main text for species collapsing onto non-trivial equivalence classes. The update functions of such species remain unchanged in the two model variants considered in the main text.

$$
\begin{aligned}
\dot{x}_{\mathtt{DAG}} &= x_{\mathtt{PLCg\_a}} - x_{\mathtt{DAG}} \\
\dot{x}_{\mathtt{IP3}} &= x_{\mathtt{PLCg\_a}} - x_{\mathtt{IP3}} \\
\\
\dot{x}_{\mathtt{PKCth}} &= x_{\mathtt{DAG}} - x_{\mathtt{PKCth}} \\
\dot{x}_{\mathtt{Ca}} &= x_{\mathtt{IP3}} - x_{\mathtt{Ca}} \\
\\
\dot{x}_{\mathtt{SEK}} &= x_{\mathtt{PKCth}} - x_{\mathtt{SEK}} \\
\dot{x}_{\mathtt{IKK}} &= x_{\mathtt{PKCth}} - x_{\mathtt{IKK}} \\
\dot{x}_{\mathtt{Calcin}} &= x_{\mathtt{Ca}} - x_{\mathtt{Calcin}} \\
\\
\dot{x}_{\mathtt{JNK}} &= x_{\mathtt{SEK}} - x_{\mathtt{JNK}} \\
\dot{x}_{\mathtt{NFAT}} &= x_{\mathtt{Calcin}} - x_{\mathtt{NFAT}}
\end{aligned}
$$

Listing S10. Excerpt of the BoolCube encoding of the Boolean model for T-cell receptor signaling studied in (37, 38) depicted in Fig. 6 from the main text. We provide the ODEs of the variables considered in Listing S9.

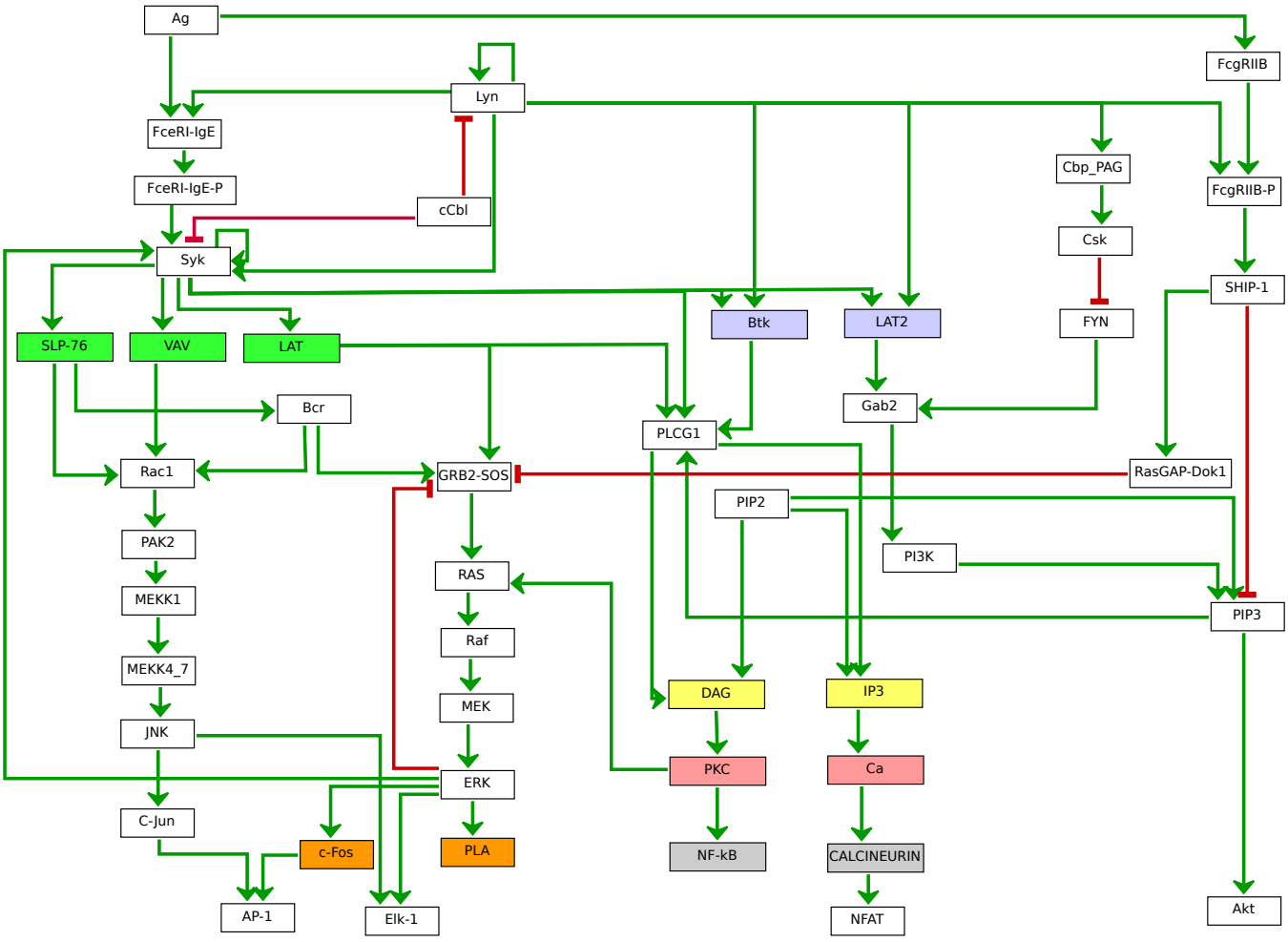

Fig. S2. Graphical representation, using the Ginsim tool (36), of a Boolean model of the FcϵRI signaling pathway during mast cell activation in wild type and mutant conditions from (58). Using the BoolCube encoding, we fixed an initial partition where the input variables are singletons. Each non-trivial backward equivalence class is represented with colored nodes with the same background.

forward equivalence classes, as shown below:

$$
\begin{aligned}
\dot{x}_{\mathrm{Rsk}} + \dot{x}_{\mathrm{Ca}} &= x_{\mathrm{ERK}} + x_{\mathrm{IP3}} - (x_{\mathrm{Rsk}} + x_{\mathrm{Ca}}) \\
\dot{x}_{\mathrm{CREB}} + \dot{x}_{\mathrm{Calcin}} &= (x_{\mathrm{Rsk}} + x_{\mathrm{Ca}}) - (x_{\mathrm{CREB}} + x_{\mathrm{Calcin}}) \qquad \text{[SE28]} \\
\dot{x}_{\mathrm{CRE}} + \dot{x}_{\mathrm{NFAT}} + \dot{x}_{\mathrm{AP1}} + \dot{x}_{\mathrm{NFKB}} &= x_{\mathrm{Fos}} \cdot x_{\mathrm{Jun}} + 1 - x_{\mathrm{IkB}} + (x_{\mathrm{CREB}} + x_{\mathrm{Calcin}}) - (x_{\mathrm{CRE}} + x_{\mathrm{AP1}} + x_{\mathrm{NFkB}} + x_{\mathrm{NFAT}})
\end{aligned}
$$

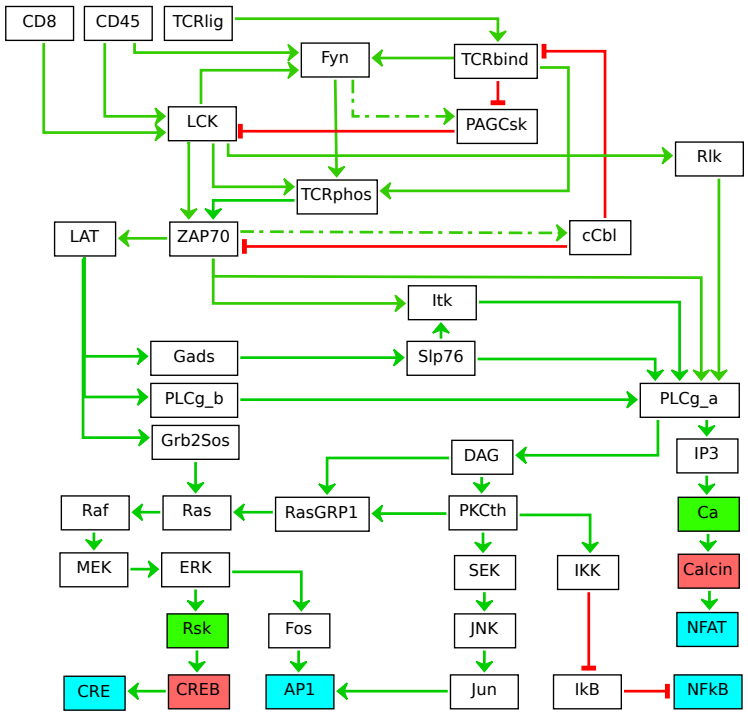

Fig. S3. Graphical representation, using (36), of the Boolean model for T-cell receptor signaling studied in (37, 38). Using the BoolCube encoding, we computed the largest forward equivalence of the network. Each non-trivial forward equivalence class is represented with colored nodes with the same background. The condition of forward equivalence holds in both model variants discussed in the main text. The equivalence class {CRE, AP1, NFKB, NFAT} consists of all the outputs of the network. Hence, in this case forward equivalence can be used for studying properties regarding the full activation/deactivation of the network, as done in (38).

```
1  Rsk     = ERK
2  Ca      = IP3
3
4  CREB    = Rsk
5  Calcin  = Ca
6
7  CRE     = CREB
8  NFAT    = Calcin
9  AP1     = Fos ∧ Jun
10 NFKB    = ¬IKB
```

Listing S11. Excerpt of the Boolean model for T-cell receptor signaling studied in (37, 38) depicted in Fig. 6 from the main text. Fig. S3 depicts the non-trivial forward equivalence classes of the model. We provide the Boolean update functions of the species collapsing in non-trivial equivalence classes. These are the same in the two variants of the model considered in the main text.

$$
\begin{aligned}
1 \quad & \dot{x}_{\mathtt{Rsk}} && = x_{\mathtt{ERK}} - x_{\mathtt{Rsk}} \\
2 \quad & \dot{x}_{\mathtt{Ca}} && = x_{\mathtt{IP3}} - x_{\mathtt{Ca}} \\
3 \quad & && \\
4 \quad & \dot{x}_{\mathtt{CREB}} && = x_{\mathtt{Rsk}} - x_{\mathtt{CREB}} \\
5 \quad & \dot{x}_{\mathtt{Calcin}} && = x_{\mathtt{Ca}} - x_{\mathtt{Calcin}} \\
6 \quad & && \\
7 \quad & \dot{x}_{\mathtt{CRE}} && = x_{\mathtt{CREB}} - x_{\mathtt{CRE}} \\
8 \quad & \dot{x}_{\mathtt{NFAT}} && = x_{\mathtt{Calcin}} - x_{\mathtt{NFAT}} \\
9 \quad & \dot{x}_{\mathtt{AP1}} && = x_{\mathtt{Fos}} \cdot x_{\mathtt{Jun}} - x_{\mathtt{AP1}} \\
10 \quad & \dot{x}_{\mathtt{NFKB}} && = 1 - x_{\mathtt{IkB}} - x_{\mathtt{NFkB}}
\end{aligned}
$$

Listing S12. Excerpt of the BoolCube encoding of the Boolean model for T-cell receptor signaling studied in (37, 38), showing only the variables in Listing S11. The output variables appear only (with negative sign) in their own derivative. This is because output variables do not affect the dynamics of other species, meaning that they do not appear in other update functions. This guarantees that we can rewrite the original ODE system in a self-consistent one where only one variable provides cumulative information about all outputs. This allows the study of full activation/deactivation of the network outputs, as performed in (38).

The reduced ODE system can be used to replicate an experiment done in (38) where the authors study: (a) full activation, when all the outputs of the network get activated without the feedback loops (dashed lines in Fig. 6 from the main text and in Fig S3); (b) full deactivation, when the outputs tend to zero when in presence of the feedback loops. In this case the variables of interest consist of all outputs; in the reduced model, full activation (respectively, full deactivation) will then be indicated by the variable representing the equivalence class of the outputs approaching 4.0 (respectively, 0.0). The results are shown in Fig. S4.

For completeness, we report on the reductions up to forward equivalence for the other networks examined in this section. Specifically, the Boolean model for tyrosine kinase ERBB2 from (57) (Fig. S1) could not be reduced. The network of the FcεRI signaling pathway of Fig. S2 has a non-trivial forward equivalence class for the six outputs (AP-1, Elk-1, PLA, NK-kN, NFAT, and Akt), and a singleton block for every other variable.

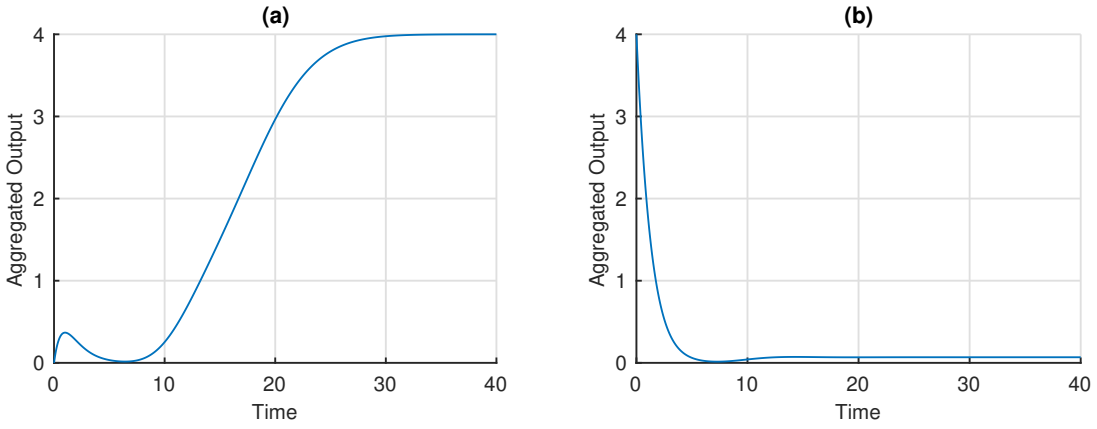

Fig. S4. Analysis regarding full activation and full deactivation of the two variants of the T-cell model in Fig. 6 from the main text using the BoolCube encoding reduced with forward equivalence. (a) Feedback loops deactivated. Following (38) we set all inputs to 1 and all other variables to 0. In this case the network gets fully activated (aggregated output approaching 4.0). (b) Feedback loops activated. As in (38) we set all inputs and outputs to 1, and all other variables to 0. In this case th network gets fully deactivated (aggregated output approaching 0.0).