A Formal Semantics for the BioNetGen Language: Category-Theoretic Foundations, Stochastic Dynamics, and Causal Structure

Claude
Anthropic

ChatGPT
OpenAI

January 29, 2026

Abstract

We present a comprehensive formal semantics for the BioNetGen Language (BNGL), a rule-based modeling language for biochemical reaction networks. Our treatment unifies several mathematical perspectives: (1) *category-theoretic foundations* via double-pushout and sesqui-pushout graph rewriting in adhesive categories, establishing BNGL's place within algebraic graph transformation theory; (2) *atoms* as elementary structural features with linear-time construction of atom-rule graphs encoding regulatory interactions; (3) *stochastic semantics* via rule algebras yielding continuous-time Markov chain dynamics; (4) *chemical reaction network theory* connecting rule structure to deficiency and persistence properties; (5) *bisimulation equivalences* enabling exact model reduction; (6) *causal semantics* in the sense of Pearl's interventional calculus; and (7) *temporal logic specifications* supporting statistical model checking. We provide three complementary semantic treatments—denotational semantics via network generation, small-step operational semantics for network-free simulation, and hybrid particle/population semantics with dynamic regime classification—and prove that all three produce identical stochastic dynamics under appropriate well-formedness conditions. This framework connects BNGL to the Kappa language through their shared categorical semantics while clarifying semantic differences, provides foundations for formal verification and certified code generation, and establishes the mathematical basis for compositional reasoning about rule-based models.

Contents

1 Introduction

Rule-based modeling has emerged as a powerful paradigm for representing biochemical systems exhibiting combinatorial complexity [4]. Rather than enumerating all possible molecular species and reactions explicitly, rule-based approaches specify transformation patterns that apply to classes of molecules sharing common structural features. The BioNetGen Language (BNGL) [1, 2, 3] is among the most widely-used rule-based modeling languages, with applications spanning signal transduction, immune receptor signaling, and gene regulatory networks.

Despite widespread adoption, BNGL has lacked a complete formal semantics that precisely defines the meaning of language constructs in mathematical terms. Previous work has described operational aspects of BioNetGen [2, 3] and developed visualization formalisms [7], but a unified formal treatment connecting syntax, static semantics, and multiple execution models has been absent. This gap impedes formal verification of models, certified code generation, and rigorous comparison between simulation strategies.

1.1 Contributions

This paper provides the first comprehensive formal semantics for BNGL with the following contributions:

Atoms as Semantic Primitives. We introduce *atoms*—elementary structural features including molecule existence, component states, free binding sites, and inter-molecular bonds—as the minimal semantic units of BNGL (Definition 5.1). Atoms provide a canonical decomposition of patterns that factors out syntactic redundancy and enables efficient analysis of rule interactions. We prove that every pattern admits a unique atom decomposition up to canonical ordering (Proposition 5.1).

Atom-Rule Graphs. We develop the *atom-rule (AR) graph*, a bipartite directed graph with atoms and rules as nodes and three edge types encoding consumption, production, and context requirements (Definition 5.8). Unlike rule influence diagrams requiring $O(|\mathcal{R}|^2)$ pairwise comparisons, AR graphs can be constructed in $O(|\mathcal{R}| \cdot k_{\max})$ time where $k_{\max}$ is the maximum pattern size, while encoding complete information about direct rule interactions (Theorem 5.3). We establish that paths in the AR graph correspond exactly to causal chains in the induced dynamics (Theorem 5.4).

Primitive Graph Operations. We formalize exactly five primitive operations—AddBond, DeleteBond, ChangeState, AddMol, and DeleteMol—that constitute the complete transformation language for BNGL rules (Definition 7.1). We prove that every well-formed rule admits a unique operation sequence and that operation application commutes appropriately (Lemma 7.5).

Three-Way Semantic Equivalence. We provide three complementary semantic treatments and prove their equivalence:

(i) **Denotational semantics** via network generation (Section 8): the classical approach that exhaustively enumerates all reachable species and reactions.

(ii) **Operational semantics** for network-free simulation (Section 9): an agent-based approach that applies rules on-the-fly without explicit enumeration.

(iii) **Hybrid semantics** with dynamic regime classification (Section 10): a novel formalization that partitions species between population-level and particle-level representations.

Our main technical result (Theorem 33.4) establishes that all three approaches induce identical continuous-time Markov chains under well-formedness conditions, providing rigorous justification for choosing simulation strategies based on computational considerations alone.

Full Hybrid Semantics. We provide the first complete formal treatment of hybrid particle/population simulation [8], including precise definitions for regime classification (Definition 10.2), promotion and demotion operations with correctness conditions (Definitions 10.5 and 10.6), on-the-fly network expansion (Definition 10.10), and hybrid propensity calculation (Definition 10.11).

1.2 Related Work

The Kappa language [5, 6] provides a closely related formalism with elegant categorical semantics based on graph rewriting. Our treatment differs in handling of symmetry factors (automatic computation in BNGL vs. explicit specification in Kappa), compartment support, and our introduction of atoms and AR graphs. The categorical foundations we employ draw on typed attributed graphs [12] and the single-pushout (SPO) approach to graph transformation [13].

Sekar et al. [7] introduced atoms and atom-rule graphs for visualization purposes. We extend their work with rigorous mathematical definitions, complexity analysis, and formal results connecting AR graph structure to stochastic dynamics.

Hogg et al. [8] introduced hybrid particle/population simulation. We formalize their operational descriptions and prove equivalence to other execution models.

1.3 Paper Organization

Section 2 presents abstract syntax with well-formedness judgments. Section 3 develops graph-theoretic foundations. Section 5 introduces atoms and atom-rule graphs. Section 6 formalizes pattern matching and automorphisms. Section 7 specifies graph operations. Section 8 presents network generation (denotational) semantics. Section 9 develops network-free (operational) semantics. Section 10 formalizes hybrid simulation. Section 11 treats rate laws and observables. Section 27 derives stochastic and deterministic dynamics. Section 33 proves semantic equivalence theorems. Section 34 provides worked examples. Section 36 concludes.

1.4 Semantic overview: generators, executions, and experiments

At the highest level, a BNGL model determines a continuous-time Markov chain (CTMC) generator Q_θ (or, equivalently, a Markov semigroup $P_t = e^{tQ_\theta}$) parameterized by θ. Exact execution strategies (network-based SSA, network-free, and hybrid) are required to be *implementations* of this same semantic object. Experiments then compose (i) a controlled generator family $\{Q_{\theta,u}\}$ with (ii) an observation/measurement map Φ to yield a statistical model.

$$\begin{tikzcd}
(\mathcal{B},\theta,u) \arrow[r, "\text{semantics}"] \arrow[dr, "\text{execution (SSA/NF/H)}"'] & Q_{\theta,u} \arrow[d, "\text{path law } \mathbb{Q}_{\theta,u}"] & \\
& \omega \in D([0,T],\Omega) \arrow[r, "\Phi"] & D \in \mathcal{D}
\end{tikzcd}$$

The equivalence results of this paper can be read as: all exact execution models induce the same $\mathbb{Q}_{\theta,u}$ (hence the same data law) whenever they start from corresponding initial states.

2 Abstract Syntax and Well-Formedness

We define the abstract syntax of BNGL as an inductively-defined grammar with associated well-formedness judgments. The formalization abstracts away concrete syntactic details (whitespace, comments, syntactic sugar) while precisely capturing the semantic content.

2.1 Syntactic Domains

Definition 2.1 (Basic Syntactic Domains)**.** The basic syntactic domains are:

$$\begin{aligned}
\mathit{Name} &= \{s \in \Sigma^* \mid s \text{ matches } [A\text{-}Za\text{-}z\_][A\text{-}Za\text{-}z0\text{-}9\_]^*\} \\
\mathit{Label} &= \mathbb{N}^+ \cup \mathit{Name} \\
\mathit{StateVal} &\subseteq \mathit{Name} \\
\mathit{Num} &= \mathbb{R} \\
\mathit{Compartment} &\subseteq \mathit{Name}
\end{aligned}$$

We use metavariables n, m for names, ℓ for labels, s for state values, and κ for compartments.

2.2 Molecule Types

Molecule types define the structural signature of molecules.

Definition 2.2 (Component Type)**.** A *component type* is a pair $\gamma = (c, S)$ where $c \in \mathit{Name}$ is the component name and $S \subseteq \mathit{StateVal}$ is the (possibly empty) finite set of allowed internal states. When $S = \emptyset$, the component carries no internal state.

Definition 2.3 (Molecule Type)**.** A *molecule type* is a pair $\Gamma = (M, [\gamma_1, \ldots, \gamma_k])$ where $M \in \mathit{Name}$ is the molecule type name and $[\gamma_1, \ldots, \gamma_k]$ is an ordered list of component types. The ordering is semantically significant: components with identical names are distinguished by position.

Definition 2.4 (Molecule Type Environment)**.** A *molecule type environment* $\mathcal{M}$ is a finite partial function $\mathcal{M} : \mathit{Name} \rightharpoonup \mathit{MolType}$. We write $M \in \mathcal{M}$ for $M \in \mathrm{dom}(\mathcal{M})$, $\mathcal{M}(M)$ for the associated type, and $\mathit{comps}(\mathcal{M}, M)$ for the component list of $\mathcal{M}(M)$.

2.3 Component and Molecule Instances

Definition 2.5 (Bond Specification)**.** A *bond specification* $\beta \in \mathit{BondSpec}$ is one of:

- $\ell \in \mathit{Label}$: bonded to the component sharing label ℓ
- **free**: the site is unbound
- **bound**: bound to some unspecified partner (wildcard)
- **any**: may or may not be bound (wildcard)

We define the partial order $\sqsubseteq$ on bond specifications: **any** $\sqsubseteq \beta$ for all β; **bound** $\sqsubseteq \ell$ for all $\ell \in \mathit{Label}$; **free** $\sqsubseteq$ **free**; and $\ell \sqsubseteq \ell$ for each ℓ.

Definition 2.6 (State Specification)**.** A *state specification* $\sigma \in \mathit{StateSpec}$ is one of:

- $s \in \mathit{StateVal}$: a concrete state value
- $\perp$: unspecified (matches any allowed state)
- ?: wildcard (explicitly matches any state)

We define: $\perp \sqsubseteq s$ and $? \sqsubseteq s$ for all $s \in \mathit{StateVal}$, and $s \sqsubseteq s$ for each s.

Definition 2.7 (Component Instance)**.** A *component instance* is a triple $c = (\mathit{name}, \sigma, \beta)$ where $\mathit{name} \in \mathit{Name}$, $\sigma \in \mathit{StateSpec}$, and $\beta \in \mathit{BondSpec}$.

Definition 2.8 (Molecule Instance)**.** A *molecule instance* is a triple $m = (M, [c_1, \ldots, c_k], \kappa)$ where $M \in \mathit{Name}$ is the molecule type name, $[c_1, \ldots, c_k]$ is an ordered list of component instances, and $\kappa \in \mathit{Compartment} \cup \{\perp\}$ is an optional compartment.

2.4 Species Graphs and Patterns

Definition 2.9 (Species Graph)**.** A *species graph* is a pair $G = (M, B)$ where:

- $M = [m_1, \ldots, m_n]$ is an ordered list of molecule instances
- $B \subseteq (\mathbb{N} \times \mathbb{N}) \times (\mathbb{N} \times \mathbb{N})$ is a symmetric relation encoding bonds, where $((i_1, j_1), (i_2, j_2)) \in B$ indicates a bond between component j_1 of molecule i_1 and component j_2 of molecule i_2

Definition 2.10 (Pattern)**.** A *pattern* is a species graph that may contain:

1. Unspecified states ($\sigma = \perp$) or wildcard states ($\sigma = ?$)
2. Bond wildcards ($\beta \in \{\mathbf{bound}, \mathbf{any}\}$)
3. Disconnected components

Definition 2.11 (Concrete Species)**.** A *concrete species* is a species graph where:

1. All states are fully specified ($\sigma \in \mathit{StateVal}$ or component has no states)
2. All bonds are concrete ($\beta \in \mathit{Label} \cup \{\mathbf{free}\}$)
3. The graph is connected (viewing molecules as nodes and bonds as edges)

We write $\mathcal{S}$ for the set of all concrete species.

2.5 Well-Formedness Judgments

We define well-formedness via typing judgments of the form $\mathcal{M} \vdash G : \mathsf{ok}$.

Definition 2.12 (Well-Formed Component Instance)**.** Component instance $c = (\mathit{name}, \sigma, \beta)$ is well-formed with respect to component type $\gamma = (n, S)$, written $\gamma \vdash c : \mathsf{ok}$, iff:

1. $\mathit{name} = n$
2. If $S = \emptyset$ then $\sigma = \perp$
3. If $S \neq \emptyset$ and $\sigma \notin \{\perp, ?\}$ then $\sigma \in S$

Definition 2.13 (Well-Formed Molecule Instance)**.** Molecule instance $m = (M, [c_1, \ldots, c_k], \kappa)$ is well-formed with respect to environment $\mathcal{M}$, written $\mathcal{M} \vdash m : \mathsf{ok}$, iff:

1. $M \in \mathcal{M}$ with $\mathcal{M}(M) = (M, [\gamma_1, \ldots, \gamma_k])$
2. $\gamma_i \vdash c_i : \mathsf{ok}$ for all $i \in \{1, \ldots, k\}$

Definition 2.14 (Well-Formed Species Graph)**.** Species graph $G = (M, B)$ with $M = [m_1, \ldots, m_n]$ is well-formed with respect to $\mathcal{M}$, written $\mathcal{M} \vdash G : \mathsf{ok}$, iff:

1. $\mathcal{M} \vdash m_i : \mathsf{ok}$ for all i
2. **(Bond consistency)** For each labeled bond: if component (i_1, j_1) has $\beta = \ell$ and component (i_2, j_2) has $\beta = \ell$, then $((i_1, j_1), (i_2, j_2)) \in B$
3. **(Label uniqueness)** Each concrete label ℓ appears exactly twice
4. **(Bond multiplicity)** Each component participates in at most one bond

2.6 Additional Static Semantics

Beyond basic typing, BNGL imposes several global consistency constraints that are essential for a fully deterministic mathematical semantics.

Bond-label scope and polarity. Bond labels are scoped to a single pattern or species graph. In concrete BNGL syntax, the same label (e.g. `!1`) may be reused in different patterns without interaction. Semantically, we therefore interpret labels as *locally bound* names that are α-renamed on parsing.

Formally, for any species graph/pattern G, let $Lbl(G)$ be the set of concrete labels that occur in bond specifications. Well-formedness strengthens Definition 2.14 with:

(a) **(Locality)** Labels are interpreted up to bijective renaming: if $\rho : Lbl(G) \to Lbl'$ is a bijection then G and $\rho(G)$ denote the same abstract graph.

(b) **(Exactly-two endpoints)** Each concrete label occurs on exactly two components (already stated as label uniqueness), and those endpoints must be distinct components.

(c) **(Bond specification consistency)** If a component is annotated with $\beta = \ell$ then it is not simultaneously annotated with **free**, **bound**, or **any**.

Lemma 2.1 (α-renaming invariance)**.** *Let G be a pattern/species graph and let $\rho : Lbl(G) \to Lbl'$ be any bijection on concrete bond labels. Then G and $\rho(G)$ are isomorphic and have identical match behavior: for all patterns P and species S,*

$$P \triangleright S \iff \rho(P) \triangleright \rho(S).$$

Wildcard bonds. The bond wildcards **bound** and **any** express constraints *relative to the matched state*:

- $\beta =$ **bound** means "must be bonded (to some partner)".
- $\beta =$ **any** means "no constraint" (may be free or bonded).

This is captured in matching via the attribute preorder in Definition 3.5: **any** $\sqsubseteq \beta$ for all β, and **bound** $\sqsubseteq \ell$ for all concrete labels ℓ.

Disconnected patterns and cross-pattern labels. Reactant lists $L = [P_1, \ldots, P_n]$ are semantically a multiset of patterns. BNGL additionally permits a single rule reactant side to refer to multiple patterns, with labels (for correspondence) that may appear across those patterns. To avoid ambiguity, we require that any label used for explicit correspondence between reactants and products be globally unique across the entire left (resp. right) side of the rule.

Canonical representatives. Because species are defined up to isomorphism (and BNGL syntax allows permutations of molecule and component order), we treat parsing as producing an isomorphism class and fix a representative via a canonical labeling function

$$canon : \mathcal{S} \to \mathcal{S}.$$

Definition 2.15 (Canonical labeling assumptions)**.** We assume *canon* satisfies:

(a) **Invariance**: if $S \cong S'$ then $canon(S) = canon(S')$.

(b) **Idempotence**: $canon(canon(S)) = canon(S)$.

(c) **Representativity**: $canon(S) \cong S$.

Canonicalization underlies (1) network generation's "visited set" of species and (2) symmetry-factor computation in Section 6. We treat the cost of computing *canon* as an implementation concern (Appendix D) rather than a semantic one.

2.7 Reaction Rules

Definition 2.16 (Reaction Rule)**.** A *reaction rule* is a tuple $r = (L, R, k, \mathit{opts})$ where:

- $L = [P_1, \ldots, P_n]$: list of reactant patterns
- $R = [Q_1, \ldots, Q_m]$: list of product patterns
- k: rate law specification
- $\mathit{opts} \subseteq \{\texttt{DeleteMolecules}, \texttt{MoveConnected}, \texttt{TotalRate}, \texttt{MatchOnce}\}$: rule options

Definition 2.17 (Well-Formed Rule)**.** Rule $r = (L, R, k, \mathit{opts})$ is well-formed with respect to $\mathcal{M}$ iff:

1. All patterns in L and R are well-formed: $\mathcal{M} \vdash P_i : \mathsf{ok}$ and $\mathcal{M} \vdash Q_j : \mathsf{ok}$
2. **(Label correspondence)** Labels in $L \cup R$ establish a consistent mapping between reactant and product elements
3. **(State definiteness)** State changes are fully specified: if $\sigma_L = \bot$ for some component, then $\sigma_R = \bot$ or the component is deleted

2.8 Observables and Complete Models

Definition 2.18 (Observable)**.** An *observable* is a triple $O = (\mathit{name}, \mathit{type}, [P_1, \ldots, P_k])$ where $\mathit{type} \in \{\texttt{Molecules}, \texttt{Species}\}$ determines counting semantics.

Definition 2.19 (BioNetGen Model)**.** A *BioNetGen model* is a tuple $\mathcal{B} = (\mathcal{M}, \mathcal{P}, \Sigma_0, \mathcal{R}, \mathcal{O}, \mathcal{F})$ where:

- $\mathcal{M}$: molecule type environment
- $\mathcal{P} : \mathit{Name} \rightharpoonup \mathbb{R}$: parameter environment
- Σ_0: initial species with concentrations
- $\mathcal{R}$: list of reaction rules
- $\mathcal{O}$: list of observables
- $\mathcal{F}$: function definitions

Model $\mathcal{B}$ is well-formed iff all components are well-formed with respect to $\mathcal{M}$.

3 Graph-Theoretic Foundations

We formalize the graph-theoretic representation of molecular structures using typed attributed graphs [12].

3.1 Typed Attributed Graphs

Definition 3.1 (Type Graph)**.** The *BNGL type graph* is $T = (T_V, T_E, s_T, t_T)$ where:

$$\begin{aligned} T_V &= \{\tau_{mol}, \tau_{comp}\} \\ T_E &= \{\epsilon_{has}, \epsilon_{bond}\} \\ s_T(\epsilon_{has}) &= \tau_{mol}, \quad t_T(\epsilon_{has}) = \tau_{comp} \\ s_T(\epsilon_{bond}) &= \tau_{comp}, \quad t_T(\epsilon_{bond}) = \tau_{comp} \end{aligned}$$

Definition 3.2 (Attribute Signature)**.** The *attribute signature* assigns to each node type:

$$\begin{aligned} \mathit{Attr}(\tau_{mol}) &= \mathit{Name} \times (\mathit{Compartment} \cup \{\perp\}) \\ \mathit{Attr}(\tau_{comp}) &= \mathit{Name} \times \mathit{StateSpec} \times \mathit{BondSpec} \end{aligned}$$

Definition 3.3 (BNGL Graph)**.** A *BNGL graph* is a tuple $G = (V, E, \tau, \lambda, s, t)$ where:

- (V, E, s, t) is a directed multigraph with $s, t : E \to V$
- $\tau : V \to T_V$ assigns types to nodes
- $\lambda : V \to \bigcup_v \mathit{Attr}(\tau(v))$ assigns attributes
- For all $e \in E$: $(\tau(s(e)), \tau(t(e))) \in \{(\tau_{mol}, \tau_{comp}), (\tau_{comp}, \tau_{comp})\}$

3.2 Semantic Translation

Definition 3.4 (Graph Semantics of Species Graphs)**.** The semantic function $[\![\cdot]\!] : \mathit{SpeciesGraph} \to \mathit{BNGLGraph}$ translates species graph $G_{syn} = ([m_1, \ldots, m_n], B)$ to BNGL graph $[\![G_{syn}]\!] = (V, E, \tau, \lambda, s, t)$:

1. For each molecule $m_i = (M_i, [c_{i,1}, \ldots, c_{i,k_i}], \kappa_i)$:
   - Create node $v_i \in V$ with $\tau(v_i) = \tau_{mol}$, $\lambda(v_i) = (M_i, \kappa_i)$
   - For each $c_{i,j} = (n_{i,j}, \sigma_{i,j}, \beta_{i,j})$: create node $v_{i,j}$ with $\tau(v_{i,j}) = \tau_{comp}$, $\lambda(v_{i,j}) = (n_{i,j}, \sigma_{i,j}, \beta_{i,j})$
   - Create edge e with $s(e) = v_i$, $t(e) = v_{i,j}$
2. For each $((i_1, j_1), (i_2, j_2)) \in B$: create edges e_1, e_2 with $s(e_1) = v_{i_1,j_1}$, $t(e_1) = v_{i_2,j_2}$ and $s(e_2) = v_{i_2,j_2}$, $t(e_2) = v_{i_1,j_1}$

3.3 Graph Morphisms

Definition 3.5 (Graph Morphism)**.** A *morphism* $\phi : G_1 \to G_2$ between BNGL graphs is a pair $\phi = (\phi_V, \phi_E)$ where $\phi_V : V_1 \to V_2$ and $\phi_E : E_1 \to E_2$ such that:

1. **(Type preservation)** $\tau_2(\phi_V(v)) = \tau_1(v)$
2. **(Structure preservation)** $\phi_V(s_1(e)) = s_2(\phi_E(e))$ and $\phi_V(t_1(e)) = t_2(\phi_E(e))$
3. **(Attribute compatibility)** $\lambda_1(v) \sqsubseteq \lambda_2(\phi_V(v))$

Definition 3.6 (Embedding and Isomorphism)**.** A morphism ϕ is an *embedding*, written $\phi : G_1 \hookrightarrow G_2$, if ϕ_V is injective. It is an *isomorphism* if both ϕ_V and ϕ_E are bijective and ϕ^{-1} is also a morphism.

Proposition 3.1 (Isomorphism Decidability)**.** *For BNGL graphs, graph isomorphism is decidable (e.g., via canonical labeling algorithms such as nauty) [11]. For the bounded-size graphs that arise from typical BNGL patterns/species, these methods are efficient in practice.*

4 Category-Theoretic Foundations

This section situates BNGL within the framework of algebraic graph transformation theory, connecting rule application to pushout constructions in adhesive categories. This categorical perspective provides rigorous foundations for compositional reasoning and clarifies the relationship between BNGL and the Kappa language.

4.1 The Category of Site-Graphs

Definition 4.1 (Category **SGraph**)**.** The category **SGraph** of *site-graphs* has:

- **Objects**: BNGL graphs $G = (V, E, \tau, \lambda, s, t)$
- **Morphisms**: Graph morphisms $\phi : G_1 \to G_2$
- **Composition**: Componentwise: $(\phi \circ \psi)_V = \phi_V \circ \psi_V$ and $(\phi \circ \psi)_E = \phi_E \circ \psi_E$
- **Identity**: $\mathrm{id}_G = (\mathrm{id}_{V_G}, \mathrm{id}_{E_G})$

Proposition 4.1 (Categorical Properties)**.** *The category* **SGraph** *has all finite limits and colimits. In particular:*

1. *Products exist (disjoint union with appropriate typing)*
2. *Pullbacks exist (constructed componentwise, then typed)*
3. *Pushouts exist (constructed via identification and union)*
4. *Monomorphisms are precisely injective morphisms*

Proof. Limits and colimits are constructed componentwise on the underlying sets V and E, inheriting from **Set**. The type function τ and attribute function λ are determined by the universal property. For monomorphisms, the characterization follows from **SGraph** being concrete over **Set**. □

4.2 Adhesive Categories

The theory of adhesive categories [14] identifies the abstract properties that make double-pushout rewriting well-behaved.

Definition 4.2 (Adhesive Category)**.** A category $\mathcal{C}$ is *adhesive* if:

1. $\mathcal{C}$ has pullbacks along monomorphisms
2. $\mathcal{C}$ has pushouts along monomorphisms
3. Pushouts along monomorphisms are *Van Kampen squares*: given a pushout square with $m : A \rightarrowtail B$ monic, for any commutative cube with the pushout as base and pullback squares as back faces, the front faces are pullbacks if and only if the top face is a pushout

The Van Kampen property ensures that pushouts "behave well" with respect to pullbacks—intuitively, gluing two objects together and then restricting is the same as first restricting and then gluing.

Theorem 4.2 (**SGraph** is Adhesive)**.** *The category* **SGraph** *of site-graphs is adhesive.*

Proof. The category **Graph** of directed graphs is adhesive, being a presheaf topos (functors from the schema $\bullet \rightrightarrows \bullet$ to **Set**). The category **SGraph** is obtained as a slice category **Graph**$/T$ over the type graph, and slicing preserves adhesivity. Adding attributes via the attribute signature corresponds to taking a pullback, which composes with the Van Kampen property. □

Corollary 4.3 (Pushout Complement Uniqueness)**.** *In* **SGraph***, pushout complements (when they exist) are unique up to isomorphism.*

4.3 Double-Pushout Rewriting

Double-pushout (DPO) rewriting provides the classical categorical approach to graph transformation [12].

Definition 4.3 (DPO Production/Rule)**.** A *DPO production* (or *rule*) is a span in **SGraph**:

$$\begin{tikzcd} L & K \arrow[l, hook', "l"'] \arrow[r, hook, "r"] & R \end{tikzcd}$$

where L is the *left-hand side* (pattern to match), R is the *right-hand side* (replacement), and K is the *interface* (gluing graph) specifying preserved structure. The morphisms l and r are typically monomorphisms.

Definition 4.4 (DPO Direct Derivation)**.** A *DPO direct derivation* $G \Rightarrow_p H$ from graph G to graph H via rule $p = (L \overset{l}{\leftarrow} K \overset{r}{\rightarrow} R)$ and match $m : L \to G$ consists of two pushout squares:

$$\begin{tikzcd} L \arrow[d, "m"'] & K \arrow[l, "l"'] \arrow[r, "r"] \arrow[d, "k"] & R \arrow[d, "m^*"] \\ G & D \arrow[l, "d"'] \arrow[r, "d'"] & H \end{tikzcd}$$

The left square is the *pushout complement*, constructing the context graph D (intuitively, G with the matched pattern removed except for the interface). The right square is the *pushout*, constructing result H (intuitively, D with the right-hand side glued in).

Definition 4.5 (Gluing Conditions)**.** The DPO derivation exists (i.e., the pushout complement exists) if and only if the match $m : L \to G$ satisfies:

1. **Dangling condition**: No edge in $G \setminus m(L)$ is incident to a node in $m(L \setminus l(K))$
2. **Identification condition**: If $m(x) = m(y)$ for $x, y \in L$, then either both $x, y \in l(K)$ or neither is

Proposition 4.4 (BNGL Rules as DPO Productions)**.** *Every well-formed BNGL rule $r = (L, R, k, opts)$ corresponds to a DPO production where the interface K consists of elements preserved by the correspondence map ϕ_r, and the five primitive operations decompose into:*

- *DeleteBond, DeleteMol: elements in $L \setminus l(K)$*
- *AddBond, AddMol: elements in $R \setminus r(K)$*
- *ChangeState: attribute differences between l and r*

4.4 Sesqui-Pushout Rewriting

Sesqui-pushout (SqPO) rewriting provides an alternative semantics that differs from DPO when gluing conditions fail.

Definition 4.6 (Final Pullback Complement)**.** Given morphisms $l : K \to L$ and $m : L \to G$, a *final pullback complement* (FPC) is a commutative square that is final among all such squares: for any other square with the same L, K, G, there is a unique mediating morphism.

Definition 4.7 (SqPO Direct Derivation)**.** A *SqPO direct derivation* uses an FPC for the left square:

$$\begin{tikzcd} L \arrow[d, "m"'] & K \arrow[l, "l"'] \arrow[r, "r"] \arrow[d, "k"] & R \arrow[d, "m^*"] \\ G & D \arrow[l, "d"'] \arrow[r, "d'"] & H \end{tikzcd}$$

where the left square is an FPC and the right square is a pushout.

The key property of FPCs is that they always exist in adhesive categories, making SqPO rewriting always applicable (unlike DPO, which requires the gluing conditions to be satisfied).

Theorem 4.5 (DPO vs. SqPO Comparison). *For a match $m : L \to G$:*

1. *If m satisfies both gluing conditions, DPO and SqPO derivations coincide (isomorphic results)*
2. *If m violates the identification condition, SqPO "clones" identified elements; DPO is undefined*
3. *If m violates the dangling condition, SqPO implicitly deletes dangling edges; DPO is undefined*

Remark 4.1 (BNGL vs. Kappa Semantics). BNGL's semantics is closer to DPO rewriting: pattern matching is typically injective, and dangling bonds are explicitly handled via the `DeleteMolecules` option. Kappa uses SqPO semantics, allowing non-injective matches with implicit side effects. This semantic difference is important when translating between the languages.

4.5 Parallel Independence and the Church-Rosser Property

Definition 4.8 (Parallel Independence). Two derivations $G \Rightarrow_{p_1,m_1} H_1$ and $G \Rightarrow_{p_2,m_2} H_2$ are *parallel independent* if each rule's match doesn't use elements that the other deletes:

$$m_1(L_1) \cap m_2(L_2 \setminus l_2(K_2)) = \emptyset \quad \text{and symmetrically}$$

Theorem 4.6 (Church-Rosser Property). *In adhesive categories, parallel independent derivations satisfy Church-Rosser: there exists G' with $H_1 \Rightarrow_{p_2} G'$ and $H_2 \Rightarrow_{p_1} G'$.*

Corollary 4.7 (Order Independence). *If two BNGL rule instances have disjoint reaction centers, application order does not affect the final state.*

4.6 Sequential Composition of Rules

Definition 4.9 (Sequential Composition). Given rules $p_1 = (L_1 \leftarrow K_1 \rightarrow R_1)$ and $p_2 = (L_2 \leftarrow K_2 \rightarrow R_2)$ with a partial overlap $E : R_1 \supseteq E \subseteq L_2$, their *sequential composition* $p_1;p_2$ is computed via pushout:

$$\begin{tikzcd}
L_1 & K_1 \arrow[l] \arrow[r] & R_1 \supseteq E \subseteq L_2 & K_2 \arrow[l] \arrow[r] & R_2 \\
L_1 & & K_{12} \arrow[ll] \arrow[rr] & & R_2
\end{tikzcd}$$

The composed rule captures the effect of applying p_1 followed by p_2.

This composition operation is fundamental to the rule algebra semantics developed in Section 28.

5 Atoms and Atom-Rule Graphs

We introduce *atoms* as the elementary structural features of BNGL and develop the *atom-rule graph* as a semantic representation of rule interactions.

5.1 Atoms: Elementary Structural Features

Atoms represent the minimal semantic units from which patterns are composed. Each atom captures an atomic piece of structural information that rules can require, consume, or produce.

Definition 5.1 (Atom Types)**.** Given molecule type environment $\mathcal{M}$, the set of *atom types* $\mathcal{A}_{\mathcal{M}}$ consists of:
1. Molecule atoms. For each $M \in \mathcal{M}$:

$$\alpha_{\mathsf{mol}}(M) \triangleq \text{“a molecule of type } M \text{ exists”}$$

2. Internal state atoms. For each $M \in \mathcal{M}$, component c of M with state set $S \neq \emptyset$, and $s \in S$:

$$\alpha_{\mathsf{state}}(M, c, s) \triangleq \text{“component } c \text{ of } M \text{ is in state } s\text{”}$$

3. Free site atoms. For each $M \in \mathcal{M}$ and component c of M:

$$\alpha_{\mathsf{free}}(M, c) \triangleq \text{“component } c \text{ of } M \text{ is unbound”}$$

4. Bond atoms. For each ordered pair of (molecule type, component) where a bond is structurally possible:

$$\alpha_{\mathsf{bond}}(M_1, c_1, M_2, c_2) \triangleq \text{“a bond exists between } c_1 \text{ of } M_1 \text{ and } c_2 \text{ of } M_2\text{”}$$

Bond atoms are symmetric: we identify $\alpha_{\mathsf{bond}}(M_1, c_1, M_2, c_2) = \alpha_{\mathsf{bond}}(M_2, c_2, M_1, c_1)$ via canonical ordering (lexicographic on (M, c) pairs).

Definition 5.2 (Atom Instance)**.** An *atom instance* in a concrete species S is a specific occurrence of an atom type. Formally, for species S with molecules $m_1, \ldots, m_n$:

- Each molecule m_i of type M contributes an instance of $\alpha_{\mathsf{mol}}(M)$
- Each component in state s contributes an instance of $\alpha_{\mathsf{state}}(M, c, s)$
- Each unbound component contributes an instance of $\alpha_{\mathsf{free}}(M, c)$
- Each bond contributes an instance of the corresponding α_{bond}

We write $\#(\alpha, S)$ for the number of instances of atom type α in S.

Definition 5.3 (Atom Extraction from Patterns)**.** For pattern P, the *atom type set* $\mathit{atoms}(P) \subseteq \mathcal{A}_{\mathcal{M}}$ contains:

$$\begin{aligned} \mathit{atoms}(P) = &\{\alpha_{\mathsf{mol}}(M) : M \text{ appears in } P\} \\ &\cup \{\alpha_{\mathsf{state}}(M, c, s) : \text{component } c \text{ of } M \text{ has } \sigma = s \text{ in } P\} \\ &\cup \{\alpha_{\mathsf{free}}(M, c) : \text{component } c \text{ of } M \text{ has } \beta = \textbf{free} \text{ in } P\} \\ &\cup \{\alpha_{\mathsf{bond}}(M_1, c_1, M_2, c_2) : \text{bond between them in } P\} \end{aligned}$$

Note: wildcards and unspecified elements do *not* contribute atoms.

Proposition 5.1 (Unique Atom Decomposition)**.** *For any concrete species S, there is a unique multiset of atom instances, and S is uniquely determined (up to isomorphism) by this multiset together with the connectivity structure among atoms.*

Proof. The atom instances encode: (1) the type and multiplicity of each molecule via α_{mol} atoms, (2) the state of each component via α_{state} atoms, (3) which components are unbound via α_{free} atoms, and (4) the bond structure via α_{bond} atoms. This information, together with the component-level connectivity implied by bonds, determines S up to molecule ordering, which is resolved by canonical labeling. □

Example 5.1. Consider species `Kin(Y~P,b!1).Sub(d!1,S~U)` representing a phosphorylated kinase bound to an unphosphorylated substrate.

The atom instances are:

- $\alpha_{\mathsf{mol}}(\mathsf{Kin})$: one instance
- $\alpha_{\mathsf{mol}}(\mathsf{Sub})$: one instance
- $\alpha_{\mathsf{state}}(\mathsf{Kin}, \mathsf{Y}, \mathsf{P})$: one instance
- $\alpha_{\mathsf{state}}(\mathsf{Sub}, \mathsf{S}, \mathsf{U})$: one instance
- $\alpha_{\mathsf{bond}}(\mathsf{Kin}, \mathsf{b}, \mathsf{Sub}, \mathsf{d})$: one instance

Note: no α_{free} atoms because both binding sites are occupied.

5.2 Atom-Rule Relationships

Rules interact with atoms in three ways: consuming them (reactant), producing them (product), or requiring them without modification (context).

Definition 5.4 (Correspondence Map)**.** For rule $r = (L, R, k, \mathit{opts})$, the *correspondence map* $\phi_r : \mathit{Elem}(L) \rightharpoonup \mathit{Elem}(R)$ is a partial bijection between syntactic elements (molecules, components, bonds) of L and R, established by:

1. Explicit labels (identical labels correspond)
2. Positional correspondence (same molecule and component indices)
3. Structural constraint (bonds must connect corresponding components)

Definition 5.5 (Reaction Center and Context)**.** The *reaction center* Center(r) consists of elements modified by the rule:

$$\mathrm{Center}(r) = \{x \in L : x \notin \mathrm{dom}(\phi_r)\} \cup \{y \in R : y \notin \mathrm{img}(\phi_r)\} \cup \{x \in L : \phi_r(x) \neq x\}$$

The *reaction context* Context(r) consists of preserved elements:

$$\mathrm{Context}(r) = \{x \in L : x \in \mathrm{dom}(\phi_r) \wedge \phi_r(x) = x\}$$

Definition 5.6 (Atom-Rule Relationship)**.** For rule r and atom type α, the relationship $\mathit{rel}(r, \alpha)$ is:

$$\mathit{rel}(r, \alpha) = \begin{cases} \mathsf{reactant} & \text{if } \alpha \in \mathit{atoms}(L|_{\mathrm{Center}}) \setminus \mathit{atoms}(R) \\ \mathsf{product} & \text{if } \alpha \in \mathit{atoms}(R|_{\mathrm{Center}}) \setminus \mathit{atoms}(L) \\ \mathsf{context} & \text{if } \alpha \in \mathit{atoms}(L|_{\mathrm{Context}}) \\ \perp & \text{otherwise} \end{cases}$$

where $L|_{\mathrm{Center}}$ denotes restriction to center elements.

Lemma 5.2 (Atom Relationship Completeness)**.** *For every atom instance that appears in a rule r, exactly one of the following holds: it is a reactant, it is a product, or it is context. An atom type may appear as both reactant and product (in different instances) when the rule transforms between states of the same structural feature.*

5.3 The Atom-Rule Graph

Definition 5.7 (Atom-Rule Graph for a Single Rule)**.** For rule r, the *atom-rule graph* $\mathrm{AR}(r) = (V_r, E_r)$ is a bipartite directed graph:

- $V_r = \mathcal{A}_r \cup \{r\}$ where $\mathcal{A}_r = \{\alpha : \mathit{rel}(r, \alpha) \neq \bot\}$
- $E_r = E_{\mathsf{react}} \cup E_{\mathsf{prod}} \cup E_{\mathsf{ctx}}$ where:

$$\begin{aligned}
E_{\mathsf{react}} &= \{(\alpha, r) : \mathit{rel}(r, \alpha) = \mathsf{reactant}\} \\
E_{\mathsf{prod}} &= \{(r, \alpha) : \mathit{rel}(r, \alpha) = \mathsf{product}\} \\
E_{\mathsf{ctx}} &= \{(\alpha, r) : \mathit{rel}(r, \alpha) = \mathsf{context}\}
\end{aligned}$$

Edge direction encodes information flow: atoms flow *into* rules via reactant/context edges and *out of* rules via product edges.

Definition 5.8 (Model Atom-Rule Graph)**.** For model $\mathcal{B}$ with rules $\mathcal{R} = \{r_1, \ldots, r_n\}$, the *model atom-rule graph* is:

$$\mathrm{AR}(\mathcal{B}) = \left(\mathcal{A}_{\mathcal{B}} \cup \mathcal{R}, \bigcup_{r \in \mathcal{R}} E_r\right)$$

where $\mathcal{A}_{\mathcal{B}} = \bigcup_{r \in \mathcal{R}} \mathcal{A}_r$ and atoms with identical types are merged into single nodes.

Theorem 5.3 (AR Graph Construction Complexity)**.** *The model atom-rule graph* $\mathrm{AR}(\mathcal{B})$ *can be constructed in* $O(|\mathcal{R}| \cdot k_{\max})$ *time, where* $k_{\max}$ *is the maximum number of atoms in any rule.*

Proof. For each rule r: (1) extract atoms from L and R in $O(|L| + |R|)$ time, (2) classify each atom as reactant, product, or context in $O(1)$ time per atom, (3) insert atoms into a hash table for $O(1)$ lookup/merge. Total time per rule is $O(k_r)$ where $k_r = |\mathcal{A}_r|$. Summing over rules: $O(\sum_r k_r) = O(|\mathcal{R}| \cdot k_{\max})$. □

Remark 5.1. This is a significant improvement over rule influence diagrams, which require $O(|\mathcal{R}|^2)$ pairwise comparisons to determine which rules can affect which others [7].

Theorem 5.4 (AR Graph Encodes Causal Dependencies)**.** *Let* $r_1, r_2 \in \mathcal{R}$ *be distinct rules. Rule* r_1 *can directly enable* r_2 *(i.e., firing* r_1 *can make* r_2 *applicable when it was not) if and only if there exists an atom* α *such that* $(r_1, \alpha) \in E_{\mathsf{prod}}$ *and* $(\alpha, r_2) \in E_{\mathsf{react}} \cup E_{\mathsf{ctx}}$*.*

Proof. ($\Rightarrow$) Suppose r_1 enables r_2. Then r_1 must produce some structural feature α that r_2 requires (as reactant or context). By Definition 5.6, $(r_1, \alpha) \in E_{\mathsf{prod}}$ and $(\alpha, r_2) \in E_{\mathsf{react}} \cup E_{\mathsf{ctx}}$.

($\Leftarrow$) If such an α exists, then r_1 produces an instance of α and r_2 requires α. Consider a state lacking any instance of α: r_2 cannot fire. After r_1 fires, an instance of α exists, potentially enabling r_2. □

Corollary 5.5 (Transitive Causal Chains)**.** *A path* $r_1 \to \alpha_1 \to r_2 \to \alpha_2 \to \cdots \to r_k$ *in* $\mathrm{AR}(\mathcal{B})$ *corresponds to a potential causal chain where each rule produces features enabling subsequent rules.*

5.4 Regulatory Motif Detection

The AR graph enables efficient detection of regulatory motifs without examining rule pairs explicitly.

Definition 5.9 (Feedback Loop)**.** A *feedback loop* is a directed cycle in $\mathrm{AR}(\mathcal{B})$. The loop has *positive sign* if it contains an even number of reactant edges (atoms consumed) and *negative sign* if odd.

Definition 5.10 (Feed-Forward Motif)**.** A *feed-forward motif* consists of three paths from atom α_1 to atom α_3:

1. Direct: $\alpha_1 \to r_1 \to \alpha_3$
2. Indirect: $\alpha_1 \to r_2 \to \alpha_2 \to r_3 \to \alpha_3$

The motif is *coherent* if both paths have the same sign and *incoherent* otherwise.

Proposition 5.6 (Motif Detection Complexity)**.** *Feedback loops of length $\leq k$ can be detected in $O(|\mathrm{AR}|^k)$ time. Feed-forward motifs can be detected in $O(|\mathrm{AR}|^3)$ time.*

5.5 AR Graph Compression

For large models, we define a compression pipeline that preserves regulatory structure while reducing complexity.

Definition 5.11 (Background Filtering)**.** The *background* $BG \subseteq \mathcal{A} \cup \mathcal{R}$ consists of:

- Free site atoms $\alpha_{\mathsf{free}}(\cdot,\cdot)$
- Ground state atoms (designated "default" states)
- Reverse rules of designated forward reactions

The filtered graph is $\mathrm{AR}'(\mathcal{B}) = \mathrm{AR}(\mathcal{B}) \setminus BG$.

Definition 5.12 (Edge Signature)**.** The *edge signature* of rule r is:

$$sig(r) = \{(\alpha, t) : \alpha \in \mathcal{A}_r, t = rel(r, \alpha)\}$$

Rules with identical signatures are *signature-equivalent.*

Definition 5.13 (Compressed AR Graph)**.** Given equivalence relations $\sim_{\mathcal{A}}$ on atoms and $\sim_{\mathcal{R}}$ on rules, the *compressed AR graph* AR_c has:

- Nodes: equivalence classes $[\alpha]$ and $[r]$
- Edges: $([\alpha],[r])$ or $([r],[\alpha])$ iff original edge exists between representatives

Proposition 5.7 (Compression Preserves Reachability)**.** *If there is a path from α to β in $\mathrm{AR}(\mathcal{B})$, then there is a path from $[\alpha]$ to $[\beta]$ in $\mathrm{AR}_c(\mathcal{B})$.*

6 Pattern Matching and Automorphisms

Pattern matching determines when a rule can apply to a given molecular state. We formalize matching as graph embedding and develop the theory of automorphisms needed for correct rate computation.

6.1 Pattern Matching via Graph Embedding

Definition 6.1 (Match)**.** Pattern P *matches* species S, written $P \triangleright S$, iff there exists an embedding $\phi : [\![P]\!] \hookrightarrow [\![S]\!]$.

Definition 6.2 (Match Set)**.** The *match set* of pattern P in species S is:

$$Match(P, S) = \{\phi : [\![P]\!] \hookrightarrow [\![S]\!]\}$$

The *match count* is $|Match(P, S)|$.

Theorem 6.1 (Match Decidability)**.** *For fixed pattern P, determining $P \triangleright S$ is decidable in polynomial time in $|S|$. Specifically, there exists an algorithm running in $O(|S|^{|P|} \cdot |P|)$ time.*

Proof. The Ullmann algorithm [10] solves subgraph isomorphism by backtracking search with constraint propagation. The search tree has depth $|P|$ and branching factor at most $|S|$. Each consistency check takes $O(|P|)$ time. With refinement, practical complexity is much better for typical BNGL patterns. □

6.2 Automorphism Groups

Definition 6.3 (Automorphism)**.** An *automorphism* of BNGL graph G is an isomorphism $\alpha : G \to G$. The set of all automorphisms forms a group under composition:

$$\mathrm{Aut}(G) = \{\alpha : G \to G \mid \alpha \text{ is an isomorphism}\}$$

Definition 6.4 (Symmetry Factor)**.** The *symmetry factor* of pattern P is:

$$sym(P) = |\mathrm{Aut}(\llbracket P \rrbracket)|$$

Proposition 6.2 (Symmetry Factor Computation)**.** *For BNGL graphs, $|\mathrm{Aut}(G)|$ can be computed algorithmically using canonical labeling / automorphism-group software such as nauty [11]. In typical BNGL use cases (small patterns and moderate complexes) this is efficient in practice.*

Example 6.1. The pattern `A(b!1).A(b!1)` (homodimer) has $sym = 2$ because the two A molecules can be swapped. The pattern `A(b!1).B(a!1)` (heterodimer) has $sym = 1$ because A and B are distinguishable.

6.3 Orbits and Distinct Matches

Definition 6.5 (Automorphism Action on Matches)**.** For $\alpha \in \mathrm{Aut}(\llbracket P \rrbracket)$ and match $\phi : \llbracket P \rrbracket \hookrightarrow \llbracket S \rrbracket$, define $\phi \cdot \alpha = \phi \circ \alpha : \llbracket P \rrbracket \hookrightarrow \llbracket S \rrbracket$. This defines a right action of $\mathrm{Aut}(\llbracket P \rrbracket)$ on $Match(P, S)$.

Definition 6.6 (Match Orbit)**.** The *orbit* of match ϕ is:

$$[\phi] = \{\phi \cdot \alpha : \alpha \in \mathrm{Aut}(\llbracket P \rrbracket)\}$$

Theorem 6.3 (Orbit-Stabilizer)**.** *For any match ϕ, $|[\phi]| \cdot |\mathrm{Stab}(\phi)| = |\mathrm{Aut}(\llbracket P \rrbracket)|$, where $\mathrm{Stab}(\phi) = \{\alpha : \phi \cdot \alpha = \phi\}$.*

Definition 6.7 (Distinct Match Count)**.** The number of *distinct matches* (orbits) of P in S is:

$$match^*(P, S) = \frac{|Match(P, S)|}{|\mathrm{Aut}(\llbracket P \rrbracket)|} = \frac{|Match(P, S)|}{sym(P)}$$

when all orbits have size $|\mathrm{Aut}(\llbracket P \rrbracket)|$ (i.e., trivial stabilizers).

Lemma 6.4 (Orbit Size for Concrete Embeddings)**.** *When P embeds into S such that the image $\phi(\llbracket P \rrbracket)$ has trivial automorphism group in $\llbracket S \rrbracket$, then $|[\phi]| = |\mathrm{Aut}(\llbracket P \rrbracket)|$.*

7 Graph Operations and Rule Application

We formalize the primitive operations that constitute BNGL's transformation language.

7.1 Primitive Graph Operations

Definition 7.1 (Graph Operations)**.** The set of *primitive graph operations* $\mathcal{O}$ consists of exactly five operations. Each operation o defines a partial function $[\![o]\!] : G \rightharpoonup G'$ with explicit preconditions and effects:

1. AddBond(c_1, c_2): Create bond between components c_1, c_2

- *Precondition*: $c_1, c_2 \in V$ are distinct component nodes, and both are currently unbound: $\lambda(c_1).\beta = \lambda(c_2).\beta = \mathbf{free}$
- *Effect*: $E' = E \cup \{(c_1, c_2), (c_2, c_1)\}$; bond specifications updated to fresh shared label ℓ: $\lambda'(c_i).\beta = \ell$

2. DeleteBond(c_1, c_2): Remove bond between c_1, c_2

- *Precondition*: $(c_1, c_2) \in E$ (bond exists)
- *Effect*: $E' = E \setminus \{(c_1, c_2), (c_2, c_1)\}$; bond specifications become free: $\lambda'(c_i).\beta = \mathbf{free}$

3. ChangeState(c, s_{old}, s_{new}): Change internal state of component c

- *Precondition*: $c \in V$ is a component node with $\lambda(c).\sigma = s_{old}$ and $s_{new} \in S_c$ (allowed states for c)
- *Effect*: $\lambda'(c).\sigma = s_{new}$; all other attributes unchanged

4. AddMol(M, init): Add new molecule of type M with initialization

- *Precondition*: $M \in \mathcal{M}$ (valid molecule type)
- *Effect*: $V' = V \cup \{v_m\} \cup \{v_{c_1}, \ldots, v_{c_k}\}$ where v_m is a fresh molecule node and v_{c_i} are its components; all components initialized with $\beta = \mathbf{free}$ and states from *init*; containment edges added

5. DeleteMol(m): Remove molecule m from the graph

- *Precondition*: $m \in V$ is a molecule node and all its components are unbound: $\forall c \in \mathit{comps}(m) : \lambda(c).\beta = \mathbf{free}$
- *Effect*: $V' = V \setminus \{m\} \setminus \mathit{comps}(m)$; all edges incident to removed nodes are deleted

Remark 7.1 (Operation Partiality). Each operation is a *partial* function: it is undefined when its precondition is not satisfied. Well-formed rules guarantee that operation sequences always have satisfied preconditions when applied to matching reactants.

Lemma 7.1 (Operational progress)**.** *Let r be a well-formed rule with operation sequence $\mathcal{O}_r$. If r matches a state σ via a valid rule instance ι (Definition 7.3), then $\mathit{apply}(r, \iota, \sigma)$ (Definition 7.4) is defined.*

Lemma 7.2 (Operational preservation)**.** *If $\mathcal{M} \vdash \sigma : \mathsf{ok}$ and $\sigma' = \mathit{apply}(r, \iota, \sigma)$ is defined for a well-formed rule r, then $\mathcal{M} \vdash \sigma' : \mathsf{ok}$.*

Proposition 7.3 (Operation Completeness)**.** *Every well-formed BNGL rule can be expressed as a sequence of primitive operations.*

Proof. A rule $r = (L, R, k, \mathit{opts})$ induces changes classified as: (1) bonds in R not in $L \rightsquigarrow$ AddBond, (2) bonds in L not in $R \rightsquigarrow$ DeleteBond, (3) state differences $\rightsquigarrow$ ChangeState, (4) molecules in R not in $L \rightsquigarrow$ AddMol, (5) molecules in L not in $R \rightsquigarrow$ DeleteMol. These five cases are exhaustive. □

Remark 7.2 (Minimality (informal)). Each primitive operation is necessary for full BNGL expressiveness: removing any one of AddBond, DeleteBond, ChangeState, AddMol, DeleteMol forbids a corresponding syntactic class of well-formed rules.

Definition 7.2 (Rule as Operation Sequence). For rule $r = (L, R, k, \mathit{opts})$, the *operation sequence* $\mathcal{O}_r = [o_1, \ldots, o_k]$ is computed by:

1. Compute correspondence map $\phi_r : L \rightharpoonup R$
2. For each bond (c_1, c_2) in R with $c_1, c_2 \notin \phi_r(L)$: append AddBond$(c_1, c_2)$
3. For each bond (c_1, c_2) in L not preserved in R: append DeleteBond(c_1, c_2)
4. For each component c with $\sigma_L \neq \sigma_R$: append ChangeState(c, σ_L, σ_R)
5. For each molecule $m \in R \setminus \mathrm{img}(\phi_r)$: append AddMol$(\mathit{type}(m), \mathit{init}(m))$
6. For each molecule $m \in L \setminus \mathrm{dom}(\phi_r)$: append DeleteMol$(m)$

Lemma 7.4 (Operation Sequence Uniqueness). *For a well-formed rule r, the operation sequence $\mathcal{O}_r$ is unique up to reordering of independent operations.*

Lemma 7.5 (Operation Commutativity). *Operations on disjoint graph elements commute:*

- *AddBond(c_1, c_2) commutes with ChangeState$(c_3, \cdot, \cdot)$ if $c_3 \notin \{c_1, c_2\}$*
- *DeleteMol(m_1) commutes with DeleteMol(m_2) if $m_1 \neq m_2$*
- *Similar for other pairs acting on disjoint elements*

7.2 Rule Application

Definition 7.3 (Rule Instance). For rule $r = ([P_1, \ldots, P_n], R, k, \mathit{opts})$ and concrete species $[S_1, \ldots, S_m]$, a *rule instance* is a tuple $\iota = (\phi_1, \ldots, \phi_n, \psi)$ where:

- Each $\phi_i : [\![P_i]\!] \hookrightarrow [\![S_{j_i}]\!]$ is a match
- ψ is an assignment of molecules to distinct species
- The matches are *compatible*: shared labels map consistently

Definition 7.4 (Rule Application). Applying rule instance ι to state σ yields state σ':

$$\sigma' = \mathit{apply}(r, \iota, \sigma) = [\![o_k]\!] \circ \cdots \circ [\![o_1]\!](\sigma, \iota)$$

where $\mathcal{O}_r = [o_1, \ldots, o_k]$ and each operation acts on elements identified by the matches in ι.

Definition 7.5 (Product Species). After rule application, the *product species* are the connected components of the resulting molecular graph. A single rule application may:

- Merge species (if AddBond joins previously separate complexes)
- Split species (if DeleteBond disconnects a complex)
- Create species (via AddMol)
- Destroy species (via DeleteMol)

8 Network Generation (Denotational Semantics)

Network generation computes the complete reaction network by exhaustively applying rules to generate all reachable species and reactions. This provides a *denotational semantics* for BNGL models.

8.1 Reaction Networks

Definition 8.1 (Reaction Network)**.** A *reaction network* is a tuple $\mathcal{N} = (\mathcal{S}, \mathcal{X}, \nu, k)$ where:

- $\mathcal{S}$ is a finite set of species (concrete species graphs up to isomorphism)
- $\mathcal{X}$ is a finite set of reactions
- $\nu : \mathcal{X} \to \mathbb{Z}^{|\mathcal{S}|}$ assigns stoichiometry vectors
- $k : \mathcal{X} \to \mathit{RateLaw}$ assigns rate laws

For reaction x with stoichiometry $\nu(x) = (\nu_1, \ldots, \nu_n)$, we write $\nu_i^-(x) = \max(0, -\nu_i)$ (reactant stoichiometry) and $\nu_i^+(x) = \max(0, \nu_i)$ (product stoichiometry).

8.2 Network Generation Algorithm

Definition 8.2 (Network Generation Semantics)**.** The *network generation function generate* : *Model* $\to$ *Network* is defined by the following fixed-point computation:

```
1: $\mathcal{S}_0 \leftarrow \{\text{species in } \Sigma_0\}$
2: $\mathcal{X}_0 \leftarrow \emptyset$
3: $i \leftarrow 0$
4: repeat
5:     $\mathcal{S}_{i+1} \leftarrow \mathcal{S}_i$
6:     $\mathcal{X}_{i+1} \leftarrow \mathcal{X}_i$
7:     for each rule $r \in \mathcal{R}$ do
8:         for each valid rule instance $\iota$ over $\mathcal{S}_i$ do
9:             $(\mathit{reactants}, \mathit{products}) \leftarrow \mathit{apply}(r, \iota)$
10:            $x \leftarrow (\mathit{reactants}, \mathit{products}, k_r)$
11:            if $x \notin \mathcal{X}_{i+1}$ then
12:                $\mathcal{X}_{i+1} \leftarrow \mathcal{X}_{i+1} \cup \{x\}$
13:                $\mathcal{S}_{i+1} \leftarrow \mathcal{S}_{i+1} \cup \mathit{products}$
14:            end if
15:        end for
16:    end for
17:    $i \leftarrow i + 1$
18: until $\mathcal{S}_i = \mathcal{S}_{i-1}$ and $\mathcal{X}_i = \mathcal{X}_{i-1}$
19: return $(\mathcal{S}_i, \mathcal{X}_i, \nu, k)$
```

Theorem 8.1 (Network Generation Soundness)**.** *For any reachable state σ (starting from Σ_0 and applying rules), every species in σ is in generate$(\mathcal{B}).\mathcal{S}$.*

Proof. By induction on the number of rule applications. Base case: $\Sigma_0 \subseteq \mathcal{S}$. Inductive step: if all species in σ are in $\mathcal{S}$ and rule r fires producing σ', then by construction of the generation algorithm, all products are added to $\mathcal{S}$. □

8.3 Termination

Definition 8.3 (Bounded Model)**.** Model $\mathcal{B}$ is *bounded* if there exists $N \in \mathbb{N}$ such that every reachable species has at most N molecules.

Theorem 8.2 (Network Generation Termination)**.** *For bounded models, network generation terminates.*

Proof. With bounded molecule count N and finite molecule types $|\mathcal{M}|$, there are finitely many species (up to isomorphism). Each iteration adds at least one new species or reaction, so the algorithm terminates. □

Remark 8.1. BioNetGen provides termination controls: `max_iter` (iteration limit), `max_agg` (maximum aggregate size), and `max_stoich` (per-species limits).

9 Network-Free Operational Semantics

Network-free simulation applies rules on-the-fly without explicit network enumeration. We provide a small-step operational semantics based on the NFsim implementation [9].

9.1 Agent-Based State

Definition 9.1 (Molecule Agent)**.** A *molecule agent* a is a runtime instance consisting of:

- $\mathit{id}(a) \in \mathbb{N}$: unique identifier
- $\mathit{type}(a) \in \mathcal{M}$: molecule type
- $\mathit{state}(a) : \mathit{Comps}(\mathit{type}(a)) \to \mathit{StateVal}$: component states
- $\mathit{bond}(a) : \mathit{Sites}(\mathit{type}(a)) \to \mathcal{G} \cup \{\bot\}$: bond partners

We write $\mathcal{G}$ for the set of all agents in a given state.

Definition 9.2 (Agent-Based State)**.** An *agent-based state* $\sigma = (\mathcal{G}, \sim)$ consists of:

- $\mathcal{G}$: finite set of molecule agents
- $\sim$: equivalence relation on $\mathcal{G}$ with $a \sim b$ iff a and b are in the same connected complex

Each equivalence class $[a]_\sim$ corresponds to a concrete species.

Definition 9.3 (State-Species Correspondence)**.** The *species multiset* of state σ is:

$$\mathit{species}(\sigma) = \{(S, n_S) : S \in \mathcal{S}, n_S = |\{[a]_\sim : [a]_\sim \cong S\}|\}$$

Two agent states are *equivalent*, $\sigma_1 \equiv \sigma_2$, iff $\mathit{species}(\sigma_1) = \mathit{species}(\sigma_2)$.

9.2 Mappings and Reactant Lists

Definition 9.4 (Mapping)**.** A *mapping* μ for pattern P is a function $\mu : V_P \to \mathcal{G}$ such that:

1. Type compatibility: $\mathit{type}(\mu(v)) = \mathit{type}_P(v)$
2. State compatibility: for each component, $\mathit{state}(\mu(v)) \sqsupseteq \sigma_P(v)$
3. Bond compatibility: bonds in P map to bonds in $\mathcal{G}$

Definition 9.5 (Reactant List)**.** For rule r with reactant pattern P_i, the *reactant list* is:

$$\mathcal{L}_{r,i}(\sigma) = \{\mu : P_i \hookrightarrow \mathcal{G} \mid \mu \text{ is a valid mapping in } \sigma\}$$

Definition 9.6 (MappingSet)**.** A *MappingSet* for rule $r = ([P_1, \ldots, P_n], R, k, \mathit{opts})$ is an n-tuple $(\mu_1, \ldots, \mu_n)$ where:

- Each $\mu_i \in \mathcal{L}_{r,i}(\sigma)$
- Mappings are *collision-free*: $\mathrm{img}(\mu_i) \cap \mathrm{img}(\mu_j) = \emptyset$ for $i \neq j$ (when patterns don't share labels)

9.3 Joint Matching, Collisions, and Symmetry

Definition 9.10 gives the standard intuition that a propensity is a microscopic rate constant multiplied by a count of applicable rule instances. For multi-reactant rules, however, the correct count is the number of *collision-free joint mappings*, not simply the product of the reactant-list sizes.

Definition 9.7 (Joint Mapping Set)**.** For $r = ([P_1, \ldots, P_n], R, k, \mathit{opts})$ and state σ, define

$$\mathcal{J}_r(\sigma) \triangleq \Big\{(\mu_1, \ldots, \mu_n) : \mu_i \in \mathcal{L}_{r,i}(\sigma) \ \wedge \ \mathit{compat}(\mu_1, \ldots, \mu_n)\Big\}$$

where *compat* enforces (i) collision-freedom when required and (ii) consistency of shared labels / shared molecules when the rule syntax uses them (e.g. explicit molecule identity constraints).

Definition 9.8 (Operational Rule Instance Count)**.** The *operational instance count* for r in state σ is

$$N_r(\sigma) \triangleq |\mathcal{J}_r(\sigma)|.$$

If the rule uses the `MatchOnce` option, $N_r(\sigma)$ is computed with additional quotienting that selects at most one representative per orbit of joint mappings under automorphisms of the left-hand side.

Definition 9.9 (Network-Free Propensity (General Form))**.** For a microscopic rate constant k_r, the default network-free propensity is

$$a_r(\sigma) = k_r \, N_r(\sigma).$$

When `TotalRate` is set, the propensity becomes a presence/absence test:

$$a_r(\sigma) = k_r \cdot \mathbf{1}_{N_r(\sigma) > 0}.$$

For rules with local functions under distribution of rates (DOR), the propensity is a sum over joint mappings:

$$a_r(\sigma) = \sum_{\mu \in \mathcal{J}_r(\sigma)} k_r \, f_r(\mu, \sigma).$$

Remark 9.1 (Relation to symmetry factors). In network-free simulation, $N_r(\sigma)$ counts *ordered* embeddings of the reactant patterns. When comparing to a generated network with mass-action kinetics, the conversion between microscopic k_r and macroscopic k_x is exactly the symmetry-factor relationship detailed in Appendix A.

9.4 Propensity Functions

Definition 9.10 (Rule Propensity). The propensity $a_r(\sigma)$ of rule r in state σ is:
Microscopic rate (default):

$$a_r(\sigma) = k_r \cdot \prod_{i=1}^{n} |\mathcal{L}_{r,i}(\sigma)|$$

Macroscopic rate (`TotalRate` option):

$$a_r(\sigma) = \begin{cases} k_r & \text{if } \forall i : |\mathcal{L}_{r,i}(\sigma)| > 0 \\ 0 & \text{otherwise} \end{cases}$$

Definition 9.11 (Total Propensity). The total propensity is $a_0(\sigma) = \sum_{r\in\mathcal{R}} a_r(\sigma)$.

9.5 Operational Semantics

Definition 9.12 (Small-Step Transition). The transition relation $\sigma \xrightarrow{r,\mu,\tau} \sigma'$ holds when:

1. Rule r is selected with probability $a_r(\sigma)/a_0(\sigma)$
2. MappingSet $\mu = (\mu_1, \ldots, \mu_n)$ is selected uniformly from valid MappingSets
3. Waiting time τ is drawn from $\text{Exp}(a_0(\sigma))$
4. State σ' is obtained by applying operations $\mathcal{O}_r$ via μ

Definition 9.13 (Network-Free Execution). A *network-free execution* from initial state σ_0 is a sequence:

$$\sigma_0 \xrightarrow{r_1,\mu_1,\tau_1} \sigma_1 \xrightarrow{r_2,\mu_2,\tau_2} \sigma_2 \to \cdots$$

The *trajectory* is the sequence $((t_i, \sigma_i))_{i\geq 0}$ where $t_i = \sum_{j<i} \tau_j$.

9.6 Incremental Update

Definition 9.14 (Affected Agents). After applying rule r via MappingSet μ, the *affected agents* are:

$$\mathit{Affected}(r,\mu) = \bigcup_i \text{img}(\mu_i) \cup \mathit{Neighbors}(\bigcup_i \text{img}(\mu_i))$$

where $\mathit{Neighbors}(A) = \{b : \exists a \in A, c.\ \mathit{bond}(a)(c) = b\}$.

Proposition 9.1 (Incremental Propensity Update). *After rule application, only propensities for rules whose patterns can match affected agents need updating:*

$$\Delta a_{r'}(\sigma) \neq 0 \implies \exists i.\ \mathit{type}(\mathit{Affected}) \cap \mathit{types}(P_i') \neq \emptyset$$

10 Hybrid Particle/Population Semantics

Hybrid simulation combines network-based and network-free approaches by partitioning species between population-level (tracked by count) and particle-level (tracked individually) representations [8]. This section provides the first complete formal treatment.

10.1 Hybrid Ensembles

Definition 10.1 (Hybrid Ensemble)**.** A *hybrid ensemble* is a triple $\mathcal{H} = (\mathcal{X}, \mathcal{S}_{\text{pop}}, \rho)$ where:

- $\mathcal{X}$ is a multiset of fully instantiated species graphs (particles)
- $\mathcal{S}_{\text{pop}} : \mathcal{S}_{\text{pop}} \to \mathbb{N}$ maps population species to copy counts
- $\rho : \mathcal{S} \to \{\text{pop}, \text{part}\}$ is the regime classification with $\mathcal{S}_{\text{pop}} = \rho^{-1}(\text{pop})$

The ensemble represents a system state where some species are "lumped" into populations (losing individual identity) while others retain full graph structure.

Definition 10.2 (Species Regime)**.** A *regime assignment* $\rho : \mathcal{S} \to \{\text{pop}, \text{part}\}$ classifies each species as:

- pop (population): tracked by integer copy number $n_S \in \mathbb{N}$
- part (particle): tracked by individual agent instances

Definition 10.3 (Hybrid State)**.** A *hybrid state* $\sigma_H = (\vec{n}, \mathcal{G}, \rho)$ consists of:

- $\vec{n} = (n_S)_{S \in \mathcal{S}_{\text{pop}}}$: copy numbers for population species
- $\mathcal{G} = \bigcup_{S \in \mathcal{S}_{\text{part}}} \mathcal{G}_S$: agents for particle species
- ρ: current regime assignment

where $\mathcal{S}_{\text{pop}} = \rho^{-1}(\text{pop})$ and $\mathcal{S}_{\text{part}} = \rho^{-1}(\text{part})$.

Definition 10.4 (Hybrid State Equivalence)**.** Hybrid state $\sigma_H = (\vec{n}, \mathcal{G}, \rho)$ is *equivalent* to agent state $\sigma = (\mathcal{G}', \sim)$, written $\sigma_H \approx \sigma$, iff:

1. For each $S \in \mathcal{S}_{\text{pop}}$: $n_S = |\{[a]_\sim \cong S : a \in \mathcal{G}'\}|$
2. For each $S \in \mathcal{S}_{\text{part}}$: agents in $\mathcal{G}_S$ biject with equivalence classes $[a]_\sim \cong S$

10.2 Promotion and Demotion

Species can transition between regimes during simulation.

Definition 10.5 (Promotion)**.** *Promotion* converts a species from particle to population representation:

$$\mathit{promote}(S, \sigma_H) = (\vec{n}[S \mapsto |\mathcal{G}_S|], \mathcal{G} \setminus \mathcal{G}_S, \rho[S \mapsto \text{pop}])$$

Definition 10.6 (Demotion)**.** *Demotion* converts a species from population to particle representation:

$$\mathit{demote}(S, \sigma_H) = (\vec{n}|_{\mathcal{S}_{\text{pop}} \setminus \{S\}}, \mathcal{G} \cup \mathit{instantiate}(S, n_S), \rho[S \mapsto \text{part}])$$

where $\mathit{instantiate}(S, n)$ creates n fresh agents of species type S.

Lemma 10.1 (Promotion/Demotion Preserves Equivalence)**.** *For any hybrid state $\sigma_H \approx \sigma$:*

1. $\mathit{promote}(S, \sigma_H) \approx \sigma$
2. $\mathit{demote}(S, \sigma_H) \approx \sigma$

Proof. Promotion replaces agents with their count, preserving the species multiset. Demotion instantiates the correct number of agents. Both maintain the bijection between species multiplicities. □

10.3 Partial Network Expansion (PNE)

Hybrid simulation maintains a partial network that expands on-the-fly. The *Partial Network Expansion* (PNE) algorithm [8] refines the original rule set to handle the partition between populations and particles.

Definition 10.7 (Partial Network)**.** A *partial network* $\mathcal{N}_\partial = (\mathcal{S}_{enum}, \mathcal{X}_{enum})$ satisfies $\mathcal{S}_{enum} \subseteq \mathcal{S}$ and $\mathcal{X}_{enum} \subseteq \mathcal{X}$ where $(\mathcal{S}, \mathcal{X}) = generate(\mathcal{B})$.

Definition 10.8 (Rule Restriction)**.** Let r be a reaction rule and ϕ be a partial embedding of the reactant pattern into a specific population species $S \in \mathcal{S}_{\text{pop}}$. The *restricted rule* $r|_\phi$ describes the action of r conditioned on the reactant at position ϕ being an instance of population species S.

Definition 10.9 (Population-Mapping Rules)**.** The PNE algorithm replaces the original rule set $\mathcal{R}$ with an expanded set $\mathcal{R}_{PNE}$ containing three types of rules:

1. **Population Rules** ($\mathcal{R}_{\text{pop}}$): All reactants are consumed from $\mathcal{S}_{\text{pop}}$. These rules operate purely on population counts and can be simulated using standard SSA on the partial network.

2. **Particle Rules** ($\mathcal{R}_{\text{part}}$): All reactants are selected from $\mathcal{X}$. These rules operate on individual molecular graphs using network-free semantics.

3. **Mixed Rules** ($\mathcal{R}_{mix}$): Reactants are drawn from both $\mathcal{S}_{\text{pop}}$ and $\mathcal{X}$. These require special handling to coordinate population decrement with particle selection.

This partition ensures unambiguous determination of whether a reactant should be decremented from a counter or removed from the particle multiset.

Proposition 10.2 (PNE Partition)**.** *For any rule $r \in \mathcal{R}$ and regime assignment ρ, the PNE algorithm produces a set of restricted rules such that:*

$$\mathcal{R}_{PNE} = \mathcal{R}_{\text{pop}} \uplus \mathcal{R}_{\text{part}} \uplus \mathcal{R}_{mix}$$

where $\uplus$ denotes disjoint union, and for any state σ:

$$\sum_{r' \in \mathcal{R}_{PNE} : r' \text{ derived from } r} a_{r'}(\sigma) = a_r(\sigma)$$

Definition 10.10 (On-the-Fly Expansion)**.** When a reaction produces species $S \notin \mathcal{S}_{enum}$:

1. Add S to $\mathcal{S}_{enum}$

2. For each rule $r \in \mathcal{R}$: compute reactions involving S as reactant

3. Add new reactions to $\mathcal{X}_{enum}$

4. Classify S via ρ (typically part for newly-seen species)

Proposition 10.3 (Expansion Correctness)**.** *On-the-fly expansion produces exactly the reactions that would be in $generate(\mathcal{B})$ involving species seen so far.*

10.4 Hybrid Propensity Calculation

Definition 10.11 (Hybrid Propensity)**.** For a hybrid rule $r_j \in \mathcal{R}_{PNE}$ with reactant species partitioned into population reactants $react_{\text{pop}}(r_j)$ and particle reactants $react_{\text{part}}(r_j)$, the propensity is:

$$a_j(\mathcal{H}) = k_j \cdot \underbrace{\left(\prod_{S \in react_{\text{pop}}(r_j)} \binom{\mathcal{S}_{\text{pop}}(S)}{\nu_S} \right)}_{\text{population factor}} \cdot \underbrace{(count_{matches}(react_{\text{part}}(r_j), \mathcal{X}))}_{\text{particle factor}}$$

where ν_S is the stoichiometry of species S as a reactant, and $count_{matches}$ counts the number of valid MappingSets for the particle reactant patterns in the particle multiset $\mathcal{X}$.

Lemma 10.4 (Hybrid Propensity Correctness)**.** *For $\sigma_H \approx \sigma$, the hybrid propensity equals the network-free propensity: $a_j(\sigma_H) = a_j(\sigma)$.*

Proof. For population species, $\binom{n_S}{\nu}$ counts the number of ways to choose ν copies from n_S identical molecules. For particle species, $|\mathcal{L}_{r,j}|$ counts valid mappings. The product gives the total number of ways to select reactants, which equals the propensity. □

10.5 Mixed Reactions

Reactions may involve both population and particle species.

Definition 10.12 (Reaction Classification)**.** Reaction x is:

- *Population-only* if all reactant species are in $\mathcal{S}_{\text{pop}}$
- *Particle-only* if all reactant species are in $\mathcal{S}_{\text{part}}$
- *Mixed* otherwise

Definition 10.13 (Mixed Reaction Execution)**.** For mixed reaction x with population reactants $(S_1, \ldots, S_k)$ and particle reactants via mappings $(\mu_1, \ldots, \mu_\ell)$:

1. Decrement population counts: $n_{S_i} \leftarrow n_{S_i} - \nu_i$ for $i \leq k$
2. Remove particle agents: delete agents in $\bigcup_j \text{img}(\mu_j)$
3. For each product species T:
   - If $T \in \mathcal{S}_{\text{pop}}$: increment n_T
   - If $T \in \mathcal{S}_{\text{part}}$: create agent(s) for T

Definition 10.14 (Population-to-Particle Binding)**.** When a population species $S \in \mathcal{S}_{\text{pop}}$ binds to a particle species $T \in \mathcal{S}_{\text{part}}$ forming complex U:

1. Select one of the n_S copies uniformly at random
2. *Demote* the selected copy: instantiate as agent a_S
3. Apply the binding operation to a_S and the particle a_T
4. The complex U inherits the particle classification

10.6 Regime Switching Criteria

Definition 10.15 (Threshold-Based Switching). Given threshold $\theta \in \mathbb{N}$, define switching rules:

- *Promote* S when $|\mathcal{G}_S| > \theta$ (too many particles)
- *Demote* S when $n_S < \theta$ and S is involved in a particle-requiring reaction

Definition 10.16 (Conservative Switching). A species S *requires particle representation* if:

1. Some rule r has a pattern P that can match S where P contains wildcards or local function references
2. S is part of a complex with a particle species

Otherwise, S can safely use population representation.

10.7 Hybrid Simulation Algorithm

Definition 10.17 (Hybrid Transition). A hybrid transition $\sigma_H \xrightarrow{x,\tau} \sigma'_H$ consists of:

1. Compute propensities $a_x(\sigma_H)$ for all reactions x
2. Sample $\tau \sim \text{Exp}(a_0)$ where $a_0 = \sum_x a_x$
3. Select x with probability a_x/a_0
4. Execute reaction:
   - If population-only: update counts directly
   - If particle-only: use network-free execution
   - If mixed: use Definition 10.13
5. Expand partial network if new species appear
6. Check and apply regime switches

11 Rate Laws and Observables

11.1 Rate Law Types

Definition 11.1 (Elementary Rate Law). An *elementary* (mass-action) rate law has propensity:

$$a(x, \sigma) = k \cdot \prod_{i=1}^{n} \binom{n_{S_i}}{\nu_i^-}$$

where k is the rate constant, n_{S_i} is the count of species S_i, and ν_i^- is its stoichiometry as reactant.

Definition 11.2 (Rate-Expression Language). A *rate expression* is an expression in a real-valued term language

$$e ::= c \mid p \mid O \mid \mathsf{count}(P) \mid e_1 \oplus e_2 \mid e_1 \odot e_2 \mid e_1/e_2 \mid e^\alpha \mid \mathsf{min}(e_1, e_2) \mid \mathsf{max}(e_1, e_2)$$

augmented by a small set of standard kinetic macros (Definitions 11.3–11.6). Here $c \in \mathbb{R}$ is a numeric constant, $p \in \text{dom}(\mathcal{P})$ is a parameter name, O ranges over observables, and $\mathsf{count}(P)$ denotes a (possibly local) pattern-count. We use $\oplus \in \{+, -\}$ and $\odot \in \{\cdot\}$.

Algorithm 1 Hybrid Particle/Population Simulation

```
Require: Model $\mathcal{B}$, initial state $\sigma_0$, end time $T$, threshold $\theta$
Ensure: Trajectory $\{(t_i, \sigma_i)\}$
 1: $\sigma_H \leftarrow initialize(\sigma_0)$                    ▷ Initial regime assignment
 2: $\mathcal{N}_\partial \leftarrow initialNetwork(\sigma_0)$
 3: $t \leftarrow 0$
 4: while $t < T$ do
 5:     Compute propensities $a_x$ for $x \in \mathcal{X}_{enum}$
 6:     $a_0 \leftarrow \sum_x a_x$
 7:     if $a_0 = 0$ then break
 8:     end if
 9:     $\tau \sim \mathrm{Exp}(a_0)$
10:     Select reaction $x$ with probability $a_x/a_0$
11:     $\sigma_H \leftarrow execute(x, \sigma_H)$
12:     $\mathcal{N}_\partial \leftarrow expand(\mathcal{N}_\partial, newSpecies)$
13:     $\sigma_H \leftarrow regimeSwitch(\sigma_H, \theta)$
14:     $t \leftarrow t + \tau$
15:     Record $(t, \sigma_H)$
16: end while
```

Definition 11.3 (Expression Evaluation (MuParser-style))**.** Given state σ and environments $\mathcal{P}$ (parameters) and $\mathcal{O}$ (observables), the denotation of an expression e is a real number $[\![e]\!]_\sigma \in \mathbb{R}$ defined by structural recursion, with:

$$[\![c]\!]_\sigma = c, \qquad [\![p]\!]_\sigma = \mathcal{P}(p), \qquad [\![O]\!]_\sigma = [\![O]\!](\sigma), \quad [\![e_1 + e_2]\!]_\sigma = [\![e_1]\!]_\sigma + [\![e_2]\!]_\sigma,$$
$$[\![e_1 \cdot e_2]\!]_\sigma = [\![e_1]\!]_\sigma \cdot [\![e_2]\!]_\sigma, \quad [\![e_1/e_2]\!]_\sigma = [\![e_1]\!]_\sigma / [\![e_2]\!]_\sigma$$

provided denominators are nonzero. (Implementations typically define behavior on zero denominators by returning 0 or an error; a *well-formed* BNGL model is required to avoid undefined arithmetic on all reachable states.)

Definition 11.4 (Well-formed rate expressions)**.** A BNGL rate expression e is *well-formed* for a model $\mathcal{B}$ if for all states σ reachable from Σ_0:

(a) every denominator encountered in evaluating $[\![e]\!]_\sigma$ is nonzero, and

(b) $[\![e]\!]_\sigma$ is finite.

A rate law is well-formed if its defining expression(s) are well-formed.

Lemma 11.1 (Propensity totality and nonnegativity)**.** *Assume all rate laws used by $\mathcal{B}$ are well-formed in the sense of Definition 11.4 and that kinetic macros (Hill/MM, etc.) are parameterized so that their denominators remain positive. Then each rule propensity $a_r(\sigma)$ is a total function on reachable states and satisfies $a_r(\sigma) \geq 0$.*

Definition 11.5 (Hill/Saturation Macro)**.** A Hill-type macro is an expression of the form

$$\mathsf{Hill}(x; K, n) \triangleq \frac{x^n}{K^n + x^n}$$

where x is an expression (often an observable or local count), $K > 0$ is a threshold parameter, and $n > 0$ is a cooperativity parameter. Its denotation is $[\![\mathsf{Hill}(x; K, n)]\!]_\sigma$ obtained by substituting $[\![x]\!]_\sigma$.

Definition 11.6 (Michaelis–Menten Macro)**.** A Michaelis–Menten macro is an expression of the form

$$\mathsf{MM}(x; V_{\max}, K_M) \triangleq \frac{V_{\max}\, x}{K_M + x}$$

with $V_{\max} \geq 0$ and $K_M > 0$.

Definition 11.7 (Local Function)**.** A *local function* is an expression e evaluated relative to a chosen mapping (or joint mapping) μ for a rule instance. We write its denotation $[\![e]\!]_{\sigma,\mu}$ to indicate that additional variables (such as the tagged molecule `%x`) are resolved via μ.

Definition 11.8 (Functional Rate Laws (Class-Complete))**.** A *functional* rate law for a reaction/rule is any rate law whose propensity is computed as an evaluated expression:

$$a(\sigma) = [\![e]\!]_{\sigma} \quad \text{or} \quad a(\sigma) = [\![e]\!]_{\sigma,\mu}$$

for some rate expression e in the language of Definition 11.2. This separates the semantics into concrete cases matching typical implementation classes: (i) pure expression-based (muparser-style) rates (Definition 11.3), (ii) Hill/saturation forms (Definition 11.5), and (iii) Michaelis–Menten forms (Definition 11.6).

Definition 11.9 (Distribution of Rates)**.** For rules with local functions, *distribution of rates* (DOR) computes a sum over joint mappings:

$$a_r(\sigma) = \sum_{\mu \in \mathcal{J}_r(\sigma)} [\![e_r]\!]_{\sigma,\mu},$$

where e_r is the rule's local rate expression and $\mathcal{J}_r(\sigma)$ is as in Definition 9.7.

11.2 Boolean Expressions and Regulatory Conditions

BioNetGen uses Boolean expressions to encode regulatory conditions and event-like guards (e.g. in functions or conditional rate expressions). In implementations, these are typically parsed into an AST (cf. a `BooleanConverter`) and then evaluated against the current state.

Definition 11.10 (Boolean Expression Syntax)**.** Boolean expressions are generated by:

$$\psi ::= \mathit{true} \mid \mathit{false} \mid (e \bowtie e) \mid \neg\psi \mid (\psi_1 \wedge \psi_2) \mid (\psi_1 \vee \psi_2)$$

where e ranges over rate expressions (Definition 11.2) and $\bowtie \in \{<, \leq, =, \geq, >\}$.

Definition 11.11 (Boolean Evaluation)**.** Given state σ, define $[\![\psi]\!]_{\sigma} \in \{\mathit{true}, \mathit{false}\}$ by recursion, using the real-valued evaluation $[\![e]\!]_{\sigma}$ from Definition 11.3. In particular, $[\![(e_1 \bowtie e_2)]\!]_{\sigma}$ is true iff $[\![e_1]\!]_{\sigma} \bowtie [\![e_2]\!]_{\sigma}$.

Definition 11.12 (Guarded Rates)**.** A *guarded* propensity has the form

$$a(\sigma) = \mathbf{1}_{[\![\psi]\!]_{\sigma}}\, [\![e]\!]_{\sigma},$$

meaning the channel is disabled whenever the Boolean condition is false.

Definition 11.13 (Observable Evaluation)**.** For observable $O = (\mathit{name}, \mathit{type}, [P_1, \ldots, P_k])$ and state σ:

$$[\![O]\!](\sigma) = \sum_{i=1}^{k} \mathit{count}(P_i, \sigma, \mathit{type})$$

Definition 11.14 (Counting Semantics)**.** The count function depends on type:
Molecules type:

$$\mathit{count}(P, \sigma, \texttt{Molecules}) = \sum_{S \in \mathcal{S}} n_S \cdot \frac{|\mathit{Match}(P, S)|}{|\mathrm{Aut}(\llbracket P \rrbracket)|}$$

This counts the total number of *distinct embeddings* across all species.
Species type:

$$\mathit{count}(P, \sigma, \texttt{Species}) = \sum_{S \in \mathcal{S} : P \triangleright S} n_S$$

This counts the number of *species instances* containing at least one match.

Example 11.1. Consider species `A(b!1).A(b!1)` (A-A dimer) with count 10, and pattern `A(b!+)` (A bound to something).

- `Molecules` count: $10 \cdot 2 = 20$ (each dimer contains two bound A's)
- `Species` count: 10 (each dimer is one species instance)

Definition 11.15 (Weight Vector)**.** For observable O over species list $[\mathcal{S}_1, \ldots, \mathcal{S}_n]$, the *weight vector* is:

$$\vec{w}_O = (w_1, \ldots, w_n) \quad \text{where } w_i = \begin{cases} |\mathit{Match}(P, S_i)| / |\mathrm{Aut}(\llbracket P \rrbracket)| & \text{if } \mathit{type} = \texttt{Molecules} \\ \mathbf{1}[P \triangleright S_i] & \text{if } \mathit{type} = \texttt{Species} \end{cases}$$

Then $\llbracket O \rrbracket(\sigma) = \vec{w}_O \cdot \vec{n}$.

Remark 11.1 (Patterns as queries). Pattern-count observables can be viewed as a semantics-level query language: $P \mapsto \mathit{count}(P, \sigma, \cdot)$ is a (typically non-injective) measurement map from latent states to data. A common identifiability question is whether a chosen family of patterns "separates" states up to a desired equivalence (e.g. bisimulation/lumping).

12 Implemented BNGL Features Beyond the Core

This section collects BNGL features that are already implemented in BioNetGen and should therefore be treated as part of the main semantics (not future work).

12.1 Compartmental semantics

BNGL supports compartments with sizes (volume/area) and (in common usage) membrane/surface conventions. Semantically, compartments refine the site-graph attribute signature and induce explicit rate-scaling factors.

Compartment signature. Let $\mathcal{K}$ be a finite set of compartment names. Each $\kappa \in \mathcal{K}$ carries a dimension flag $\dim(\kappa) \in \{2, 3\}$ and a size parameter $\mathrm{size}(\kappa) \in \mathbb{R}_{\geq 0}$ (area if $\dim = 2$, volume if $\dim = 3$).

Definition 12.1 (Compartmented site-graphs)**.** Define a compartmented attribute signature by refining the molecule-node attributes to $\mathit{Attr}_{\mathcal{K}}(\tau_{mol}) = \mathit{Name} \times \mathcal{K}$. Write $\mathbf{SGraph}_{\mathcal{K}}$ for the category of site-graphs in which every molecule node carries such a compartment attribute and morphisms preserve it.

There is an evident forgetful functor $U : \mathbf{SGraph}_{\mathcal{K}} \to \mathbf{SGraph}$ that drops compartment annotations.

Proposition 12.1 (Adhesivity is preserved under compartment refinement). $\mathbf{SGraph}_{\mathcal{K}}$ *is adhesive, and DPO/SqPO constructions lift along the forgetful functor* U *(i.e. pushouts/pullbacks in* $\mathbf{SGraph}_{\mathcal{K}}$ *map to pushouts/pullbacks in* $\mathbf{SGraph}$*).*

Remark 12.1 (Adjunction viewpoint). One may view $\mathbf{SGraph}_{\mathcal{K}}$ as a slice/structured-graph category over the set $\mathcal{K}$ of compartments. The forgetful functor $U : \mathbf{SGraph}_{\mathcal{K}} \to \mathbf{SGraph}$ typically admits a left adjoint that equips a site-graph with a default compartment assignment. This makes "adding compartments" a functorial construction and provides a clean route to proving that rewriting steps commute with forgetting compartments (up to canonical isomorphism).

Lemma 12.2 (Forgetting compartments preserves matches). *Let* P, S *be compartmented patterns/species and write* $U(P), U(S)$ *for their images in* $\mathbf{SGraph}$*. Then every match* $\phi : P \hookrightarrow S$ *induces a match* $U(\phi) : U(P) \hookrightarrow U(S)$*. Moreover, if* P *and* S *lie in a single compartment and matching does not test compartments, then the induced map on match sets is bijective.*

Remark 12.2 (When compartment forgetting is conservative). The "matching does not test compartments" premise can be made explicit as: patterns omit compartment constraints and the morphism preorder on molecule-node attributes treats the compartment coordinate as ignored. In that case, U is faithful on match sets within a fixed compartment. If compartment attributes are tested (e.g. patterns require a specific compartment), then U strictly enlarges match sets.

Definition 12.2 (Compartment adjacency as a graph). Let $G_{\mathcal{K}} = (\mathcal{K}, adj)$ be a directed graph of admissible adjacencies. For transport-only models (rules consisting only of MoveMol), one may view each molecule as performing a continuous-time random walk on $G_{\mathcal{K}}$.

Proposition 12.3 (Transport as a Markov jump process on compartments). *Fix a molecule type* M *and transport rates* $\tau_{\kappa \to \kappa'} \geq 0$ *for edges* $(\kappa, \kappa') \in adj$*. The compartment coordinate of a single* M*-molecule is a CTMC on* $\mathcal{K}$ *with generator*

$$(Q_M f)(\kappa) = \sum_{\kappa' : (\kappa, \kappa') \in adj} \tau_{\kappa \to \kappa'} \left(f(\kappa') - f(\kappa) \right).$$

For independent molecules, the global generator is the sum of single-particle transport generators plus any intra-compartment reaction generators.

Remark 12.3 (Generator decomposition: transport + reaction). In the presence of both transport and reaction rules, the induced CTMC generator splits as

$$Q = Q_{rxn} + Q_{tr},$$

where Q_{tr} is a sum of compartment-move generators of the form Q_M (lifted to multisets of molecules), and Q_{rxn} is the generator obtained by restricting rule matching to fixed compartments. This decomposition is the mathematical statement behind BioNetGen's separation of "where the molecule is" from "what reactions can fire".

Proposition 12.4 (Commuting subgenerators under independence). *If (i) transport rules never create/delete bonds or molecules and (ii) reaction propensities do not depend on transport history beyond the current compartment attributes, then* Q_{rxn} *and* Q_{tr} *commute on bounded observables:* $Q_{rxn} Q_{tr} = Q_{tr} Q_{rxn}$*. Equivalently, the semigroups factor as* $e^{tQ} = e^{tQ_{rxn}} e^{tQ_{tr}}$*.*

Typing / well-formedness. Extend molecule-node attributes so that every molecule carries a compartment annotation. A molecule type M may specify admissible compartments $\mathcal{K}(M) \subseteq \mathcal{K}$, and a molecule instance $m = (M, \dots, \kappa)$ is well-formed only if $\kappa \in \mathcal{K}(M)$.

Transport and boundary rules. Introduce a primitive operation $\textsc{MoveMol}(m, \kappa \to \kappa')$ whose effect is to update the compartment attribute of a molecule node while preserving its internal structure. For cross-compartment interactions, assume an adjacency relation $adj \subseteq \mathcal{K} \times \mathcal{K}$ that determines which compartments may jointly appear in a single rule instance.

Propensity scaling. For an elementary network-free rule instance count $N_r(\sigma)$ (ordered joint mappings), define

$$a_r(\sigma) = k_r\, g_r(\kappa)\, N_r(\sigma)$$

where g_r is a geometric factor determined by the compartment(s) where the rule fires (e.g. volume scaling for dim $= 3$, area scaling for dim $= 2$, and interface scaling for transport/boundary events).

Remark 12.4 (Unit consistency via geometric factors). If k_r is interpreted as a microscopic rate constant in concentration units, g_r is the place where BioNetGen's volume/area conversions enter. Mathematically, g_r can be treated as a deterministic state-dependent multiplicative observable.

In particular, one can make g_r explicit as a function of the matched compartment(s): $g_r = g_r(\mu)$, where $\mu \in \mathcal{J}_r(\sigma)$ is the joint mapping that identifies where the rule instance lives.

Conservativity obligation. If all molecules are constrained to a single compartment with constant size $= 1$ and no $\textsc{MoveMol}$ occurs, then the compartmental semantics reduces to the core semantics (same match sets and propensities).

Implementation touchpoints (BioNetGen). Compartment objects and size/geometry logic appear in `bng2/Perl2/Compartment.pm` and `bng2/Perl2/CompartmentList.pm`, with molecule annotations in `bng2/Perl2/Molecule.pm` and compartment-dependent unit conversion in `bng2/Perl2/Rxn.pm`.

Theorem 12.5 (Compartmental conservativity). *Assume a model in which all molecules are constrained to a single compartment κ with* $\mathrm{size}(\kappa) = 1$, *$adj = \{(\kappa, \kappa)\}$, and no rule uses $\textsc{MoveMol}$. Then (i) the set of joint mappings $\mathcal{J}_r(\sigma)$ is identical to that of the compartment-free semantics and (ii) all propensities coincide. Consequently, the induced CTMC generator is unchanged.*

12.2 Energy-based semantics

BioNetGen supports energy-pattern approaches ("energyBNGL") in which rates are constrained to be compatible with a Gibbs-like stationary distribution.

Energy patterns and total energy. Let $\mathcal{E}$ be a finite set of patterns (site-graphs) with an energy map $E : \mathcal{E} \to \mathbb{R}$. For a concrete state σ, define

$$\mathcal{H}(\sigma) \triangleq \sum_{e \in \mathcal{E}} E(e)\, \#(e, \sigma),$$

where $\#(e, \sigma)$ is an occurrence count (typically a distinct-match count, quotiented by $\mathrm{Aut}(e)$, as in the observable semantics).

Local detailed balance. For each reversible rule pair $r : \sigma \to \sigma'$ and $r^{-1} : \sigma' \to \sigma$, a standard sufficient condition for detailed balance is

$$\frac{a_r(\sigma)}{a_{r^{-1}}(\sigma')} = \exp\bigl(-(\mathcal{H}(\sigma') - \mathcal{H}(\sigma))\bigr).$$

Definition 12.3 (Metropolis/Arrhenius rate splitting)**.** A common way to enforce local detailed balance is to choose symmetric proposal rates $\lambda(\sigma, \sigma') = \lambda(\sigma', \sigma)$ for each reversible pair and define

$$Q_{\sigma,\sigma'} = \lambda(\sigma, \sigma') \min\bigl(1, \exp(-(\mathcal{H}(\sigma') - \mathcal{H}(\sigma)))\bigr).$$

Equivalently, one may split energy differences into forward/backward microscopic rate constants $k_r, k_{r^{-1}}$ satisfying the ratio constraint.

Definition 12.4 (Energy patterns as sufficient statistics)**.** Let $T(\sigma) \in \mathbb{N}^{|\mathcal{E}|}$ be the vector of energy-pattern counts $T_e(\sigma) = \#(e, \sigma)$. Then the Gibbs family is an exponential family $\pi_\theta(\sigma) \propto \exp(-\langle \theta, T(\sigma) \rangle)$ with natural parameters $\theta_e = E(e)$.

Proposition 12.6 (Variational characterization (finite state))**.** *On a finite state space Ω, the Gibbs distribution $\pi(\sigma) \propto \exp(-\mathcal{H}(\sigma))$ uniquely minimizes the free-energy functional*

$$\mathcal{F}(\mu) \triangleq \mathbb{E}_\mu[\mathcal{H}] + \sum_{\sigma \in \Omega} \mu(\sigma) \log \mu(\sigma)$$

over probability measures μ on Ω. Equivalently, π maximizes entropy subject to a fixed expected energy.

Theorem 12.7 (Detailed balance from local energy constraints (finite state))**.** *Assume the state space is finite (e.g. via boundedness constraints) and that for every reversible pair $\sigma \rightleftarrows \sigma'$ the generator entries satisfy local detailed balance with respect to $\mathcal{H}$. Then the Gibbs distribution $\pi(\sigma) \propto \exp(-\mathcal{H}(\sigma))$ is reversible for the CTMC, i.e. $\pi(\sigma) Q_{\sigma,\sigma'} = \pi(\sigma') Q_{\sigma',\sigma}$ for all $\sigma \neq \sigma'$.*

Proof. Fix $\sigma \neq \sigma'$ with $Q_{\sigma,\sigma'} > 0$. By local detailed balance, $Q_{\sigma,\sigma'}/Q_{\sigma',\sigma} = \exp(-(\mathcal{H}(\sigma') - \mathcal{H}(\sigma)))$. Multiplying both sides by $\exp(-\mathcal{H}(\sigma))$ gives $\exp(-\mathcal{H}(\sigma)) Q_{\sigma,\sigma'} = \exp(-\mathcal{H}(\sigma')) Q_{\sigma',\sigma}$. Normalizing by Z yields the claim. □

Remark 12.5 (Beyond the finite-state case). In typical BNGL models the reachable state space can be countably infinite. To extend Gibbs-stationarity beyond the finite setting, one needs additional assumptions ensuring (i) non-explosion of the CTMC and (ii) normalizability of $\pi(\sigma) \propto \exp(-\mathcal{H}(\sigma))$. A standard route is to impose a Foster–Lyapunov drift condition using a confining energy (or explicit boundedness constraints), which yields positive recurrence and a normalizable stationary distribution.

Definition 12.5 (Foster–Lyapunov drift (CTMC form))**.** Let Q be the generator on a countable state space Ω. A function $V : \Omega \to [1, \infty)$ is a Lyapunov function if there exist constants $b < \infty$, $\epsilon > 0$, and a finite set $K \subset \Omega$ such that

$$(QV)(\sigma) \leq -\epsilon V(\sigma) + b \mathbf{1}_K(\sigma) \quad \text{for all } \sigma \in \Omega.$$

Remark 12.6 (Energy as a Lyapunov function). A sufficient condition for non-explosion/positive recurrence is that the energy $\mathcal{H}$ dominates growth of molecule counts, so that $V(\sigma) = \exp(c \mathcal{H}(\sigma))$ (or $V(\sigma) = 1 + \mathcal{H}(\sigma)$) satisfies a drift condition. This makes the "energy semantics" compatible with standard stability theory of Markov jump processes.

Generator-level statement (target). Let Q be the CTMC generator induced by the energy-constrained propensities. On a state space where the normalizing constant is finite, define

$$\pi(\sigma) \triangleq Z^{-1} \exp(-\mathcal{H}(\sigma)), \qquad Z = \sum_{\sigma} \exp(-\mathcal{H}(\sigma)).$$

A rigorous goal is to show π is reversible:

$$\pi(\sigma) Q_{\sigma,\sigma'} = \pi(\sigma') Q_{\sigma',\sigma} \quad (\sigma \neq \sigma'),$$

which implies stationarity ($\pi Q = 0$). This requires an explicit non-explosion / normalizability hypothesis (e.g. bounded models, truncations, or confining energies).

Implementation touchpoints (BioNetGen). Energy patterns are represented in `bng2/Perl2/EnergyPattern.pm`, with rate-law construction and local evaluation/caching in `bng2/Perl2/RateLaw.pm` and reaction assembly in `bng2/Perl2/Rxn.pm`.

Proposition 12.8 (From local to global detailed balance (finite-state case))**.** *Assume the state space is finite (e.g. via boundedness constraints) and that for all reversible transitions $\sigma \rightleftarrows \sigma'$ the propensities satisfy local detailed balance with respect to $\mathcal{H}$. Then the Gibbs measure $\pi(\sigma) \propto \exp(-\mathcal{H}(\sigma))$ is reversible for the CTMC, i.e. $\pi(\sigma)Q_{\sigma,\sigma'} = \pi(\sigma')Q_{\sigma',\sigma}$ for all $\sigma \neq \sigma'$.*

12.3 Actions, functions, and options

BNGL models are typically executed via action scripts (network generation, simulation, parameter scans). Semantically, this layer can be treated as a small command language over a state consisting of a model plus derived artifacts.

Artifact state. An artifact state is a tuple $(\mathcal{B}, \mathcal{N}, \mathcal{T}, \mathcal{D}, \mathsf{rng})$ consisting of a model, an optional generated network, an optional set of trajectories, optional derived data (tables/plots), and an RNG state.

Big-step action semantics. Each action (e.g. `generate_network`, `simulate`, `parameter_scan`) is a partial function on artifact states; a script is interpreted by composition. Two scripts are reproducibly equivalent if, when started from the same $(\mathcal{B}, \mathsf{rng})$, they induce the same probability law over artifacts.

Definition 12.6 (Script denotation as a Markov kernel)**.** Fix a measurable space of artifact states $\mathcal{A}$. A script C denotes a Markov kernel

$$[\![C]\!] : (\mathit{Model} \times \mathit{RNG}) \rightsquigarrow \mathcal{A}$$

that maps an input $(\mathcal{B}, \mathsf{rng})$ to a probability measure over artifact states. Deterministic actions correspond to Dirac kernels; stochastic actions (SSA/NF/ hybrid simulation, randomized scans) yield nontrivial kernels.

Definition 12.7 (Reproducible equivalence)**.** Two scripts C_1, C_2 are *reproducibly equivalent* if $[\![C_1]\!] = [\![C_2]\!]$ as kernels, i.e. they induce identical output measures for all inputs.

Proposition 12.9 (Observable-level script equivalence)**.** *Let $h : \mathcal{A} \to \mathbb{R}$ be any measurable statistic of artifacts (e.g. a final observable value, a time series summary, or a network-derived quantity). If $[\![C_1]\!] = [\![C_2]\!]$, then for all inputs $(\mathcal{B}, \mathsf{rng})$,*

$$\mathbb{E}[h(\mathcal{A}) \mid \mathcal{A} \sim [\![C_1]\!](\mathcal{B}, \mathsf{rng})] = \mathbb{E}[h(\mathcal{A}) \mid \mathcal{A} \sim [\![C_2]\!](\mathcal{B}, \mathsf{rng})].$$

Remark 12.7 (Kleisli composition). Markov kernels form the morphisms of the Kleisli category of the Giry monad. On this view, an action script is literally a Kleisli composite of primitive actions, and equational reasoning about scripts reduces to categorical properties of the Kleisli category (associativity, identities, etc.).

Lemma 12.10 (Script compositionality)**.** *If $C_1; C_2$ is sequential composition, then*

$$[\![C_1; C_2]\!] = [\![C_2]\!] \circ [\![C_1]\!]$$

where $\circ$ denotes Kleisli composition of kernels (pushforward/integration).

Functions and scoping. Function definitions are expressions over parameters, observables, and (for local functions) the current joint mapping μ. A key well-formedness condition is that every expression evaluation needed in propensities is defined on all reachable states (e.g. no division by zero in Definition 11.3).

Options as semantic modifiers. Options such as `MatchOnce`, `TotalRate`, `MoveConnected`, and `DeleteMolecules` are treated as explicit modifications of (i) the joint mapping set $\mathcal{J}_r(\sigma)$, (ii) the apply function (Definition 7.4), or (iii) the induced propensity. In particular, `MatchOnce` is modeled as an explicit quotienting/representative choice on joint mappings (Definition 9.8).

Option	Semantic hook	Effect (informal)
`MatchOnce`	instance counting	choose at most one rep per symmetry orbit
`TotalRate`	propensity	presence/absence: $k_r\,\mathbf{1}[N_r > 0]$
`DeleteMolecules`	apply / gluing	permit deletion of bonded complexes (with side effects)
`MoveConnected`	apply	move entire connected component when relocating molecules

A proof obligation for each option is to show that the modified instance-count and/or apply operator still yields a well-defined generator Q on the intended state space (and, when claimed, preserves equivalence to other exact execution models under the same option).

13 Molecule Tagging and Tracking Semantics

BioNetGen supports *molecule tagging*, a mechanism for tracking specific molecular instances through reaction sequences. Tags provide a way to define observables and rate laws that depend on the identity of particular molecules rather than just their structural properties.

13.1 Tag Syntax and Static Semantics

Definition 13.1 (Tag Annotation)**.** A *tag annotation* is a label of the form `%t` where $t \in \mathit{Name}$ is a tag identifier. Tags are attached to molecule instances in patterns:

$$\text{Tagged molecule} ::= \text{Molecule}\ \%\, t$$

Definition 13.2 (Tag Environment)**.** A *tag environment* $\Gamma_{\mathit{tag}} : \mathit{Name} \rightharpoonup \mathit{MolRef}$ maps tag names to molecule references within a pattern or rule context. For a rule $r = (L, R, k, \mathit{opts})$, the tag environment is constructed by:

1. Scanning L for all tag annotations %t
2. For each tagged molecule $m\%t$, binding $\Gamma_{\mathit{tag}}(t) = \mathit{ref}(m)$
3. Verifying that each tag name appears at most once in L

Definition 13.3 (Well-Formed Tagged Pattern)**.** A tagged pattern P is well-formed if:

1. Each tag name appears on at most one molecule in P
2. Tags on the right-hand side R of a rule reference molecules that either (a) appear in L with the same tag, or (b) are newly created by R
3. Tag references in rate expressions refer only to tags bound in L

13.2 Operational Semantics of Tags

Tags create a correspondence between pattern molecules and specific agent instances during rule application.

Definition 13.4 (Tagged Mapping)**.** A *tagged mapping* for rule r with tag environment Γ_{tag} is a pair (μ, τ) where:

- $\mu : V_L \to \mathcal{G}$ is a standard mapping (Definition 9.4)
- $\tau : \text{dom}(\Gamma_{tag}) \to \mathcal{G}$ satisfies $\tau(t) = \mu(\Gamma_{tag}(t))$ for all t

The tag binding τ is uniquely determined by μ and Γ_{tag}.

Definition 13.5 (Tag-Dependent Local Functions)**.** A *tag-dependent local function* f_t is an expression that may reference tagged molecules. For a tagged mapping (μ, τ), evaluation proceeds by:

1. Resolving tag references: `%t.comp~state` evaluates to the state of component `comp` on agent $\tau(t)$
2. Evaluating arithmetic expressions over resolved values

Formally, the evaluation context extends Definition 11.7:

$$[\![e]\!]_{\sigma,\mu,\tau} \in \mathbb{R}$$

Example 13.1 (Tag-Dependent Phosphorylation Rate). Consider a kinase that phosphorylates substrates at a rate depending on the kinase's own phosphorylation state:

```
Kin(Y~P)%k + Sub(S~U) -> Kin(Y~P)%k + Sub(S~P)  f(%k)
```

The rate function f(`%k`) can access properties of the specific kinase instance matched by the tag. If the kinase has multiple phosphorylation sites:

```
f(%k) = k_base * (1 + alpha * %k.Y1~P + beta * %k.Y2~P)
```

where `%k.Y1~P` evaluates to 1 if site Y1 of the tagged kinase is phosphorylated and 0 otherwise.

13.3 Tag Propagation Through Rules

Definition 13.6 (Tag Inheritance)**.** When a rule r creates new molecules, tags may be assigned to track lineage:

- **Preserved molecules**: Tags on molecules in $L \cap R$ (via correspondence) are preserved on the corresponding agents
- **New molecules**: Tags on molecules in $R \setminus \phi_r(L)$ create fresh tag bindings to newly instantiated agents
- **Deleted molecules**: Tags on molecules in $L \setminus \text{dom}(\phi_r)$ become unbound after rule application

Definition 13.7 (Tag Scope)**.** Tags have *rule-local scope*: they are bound during pattern matching and exist only for the duration of that rule instance's evaluation. Global tracking of molecular identity requires explicit mechanisms (e.g., unique component states or external bookkeeping).

13.4 Tagged Observables

Definition 13.8 (Tag-Filtered Observable)**.** In the context of local functions within rules, a *tag-filtered observable* counts matches subject to constraints on tagged molecules:

$$count_\tau(P, \sigma) = |\{\mu : P \hookrightarrow \mathcal{G} \mid \mu \text{ consistent with } \tau\}|$$

This allows observables like "count substrates bound to the specific kinase matched by tag `%k`."

13.5 Semantic Relationship to Network-Free Simulation

Proposition 13.1 (Tag Semantics in Network-Free Context)**.** *In network-free simulation, tags provide direct access to agent identity through the mapping μ. The tag-dependent propensity for rule r with local function f is:*

$$a_r(\sigma) = \sum_{\mu \in \mathcal{J}_r(\sigma)} [\![f]\!]_{\sigma,\mu,\tau_\mu}$$

where τ_μ is the tag binding induced by μ.

Remark 13.1 (Tags and Network Generation). In network-based simulation after full enumeration, tag-dependent rates must be "compiled out" into explicit reaction variants. A rule with tag-dependent rate $f(\texttt{\%k})$ where the tagged molecule can be in n distinct states generates up to n distinct reactions with state-specific rate constants.

14 Event Semantics

BioNetGen supports discrete *events* that trigger instantaneous state modifications when specified conditions are met. Events extend the continuous-time Markov chain semantics with a hybrid automaton structure.

14.1 Event Syntax

Definition 14.1 (Event Specification)**.** An *event* is a tuple $E = (\psi, \mathit{actions}, \mathit{opts})$ where:

- ψ is a Boolean trigger condition (Definition 11.10)
- *actions* is a list of atomic state modifications
- $\mathit{opts} \subseteq \{\texttt{once}, \texttt{persistent}\}$ are event options

Definition 14.2 (Event Actions)**.** The set of *event actions* includes:

1. **Parameter modification**: $p := e$ where $p \in \mathrm{dom}(\mathcal{P})$ and e is an expression
2. **Species addition**: $\mathsf{add}(S, n)$ adds n copies of species S
3. **Species removal**: $\mathsf{remove}(S, n)$ removes up to n copies of S
4. **Species setting**: $\mathsf{set}(S, n)$ sets the count of S to exactly n
5. **Observable-based**: $\mathsf{set}(S, f(O_1, \ldots, O_k))$ sets count based on observable values

14.2 Event Trigger Semantics

Definition 14.3 (Trigger Evaluation)**.** For event E with trigger ψ, the trigger is *active* at time t if:

$$[\![\psi]\!]_{\sigma(t^-)} = \mathit{false} \quad \text{and} \quad [\![\psi]\!]_{\sigma(t)} = \mathit{true}$$

That is, the trigger fires on the transition from false to true (rising edge semantics).

Definition 14.4 (Time-Based Triggers)**.** A *time-based trigger* has the form $t \geq t_0$ or $t = t_0$ for a specified time t_0. These are evaluated against the simulation clock and fire at the first time point satisfying the condition.

Definition 14.5 (Observable-Based Triggers)**.** An *observable-based trigger* references pattern counts:

$$\psi ::= O \bowtie c \mid O_1 \bowtie O_2 \mid \psi_1 \wedge \psi_2 \mid \psi_1 \vee \psi_2 \mid \neg\psi$$

where O, O_1, O_2 are observables and $\bowtie \in \{<, \leq, =, \geq, >\}$.

14.3 Hybrid Automaton Interpretation

Definition 14.6 (BNGL Hybrid Automaton)**.** A BNGL model $\mathcal{B}$ with events $\mathcal{E} = \{E_1, \ldots, E_m\}$ defines a *hybrid automaton* $\mathcal{H} = (Q, \Sigma, \mathit{Flow}, \mathit{Jump}, \mathit{Init})$:

- $Q = 2^{\{1,\ldots,m\}}$: discrete modes indexed by which events have fired (for `once` events)
- $\Sigma = \mathbb{N}^{|\mathcal{S}|}$: continuous state (species counts)
- $\mathit{Flow}(q)$: CTMC dynamics restricted to mode q
- *Jump*: transitions triggered by event conditions
- $\mathit{Init} = (q_0, \sigma_0)$: initial mode and state

Definition 14.7 (Event Execution Order)**.** When multiple events trigger simultaneously (at the same simulation time):

1. Events are processed in declaration order
2. Each event's actions execute atomically before the next event is checked
3. Later events see the state modifications from earlier events

This induces a total order on simultaneous events.

14.4 Operational Semantics

Definition 14.8 (Event-Augmented Transition)**.** The transition relation for a BNGL model with events combines CTMC transitions and event triggers:

$$(\sigma, t) \to (\sigma', t')$$

where either:

1. **Reaction step**: $t' = t + \tau$ for $\tau \sim \mathrm{Exp}(a_0(\sigma))$, σ' results from a selected reaction, and no event triggers in (t, t')
2. **Event step**: $t' = t$, some event E_i triggers at (σ, t), and $\sigma' = \mathit{apply}(E_i.\mathit{actions}, \sigma)$

14.5 Distributed and Advanced Event Scheduling

Simulating events in parallel or distributed environments requires careful scheduling to preserve the rising-edge trigger semantics and atomicity of event actions.

Definition 14.9 (Event Execution Consistency)**.** A distributed event scheduler is *consistent* if for any time t the set of events that are active (rising edge from false to true) and their total order (declaration order) observed by a centralized execution agrees with the order produced by the distributed scheduler after synchronization.

Conservative synchronization A conservative scheduler obtains safe time windows by exchanging local lookahead bounds. When no time-based event trigger or reaction occurs before the lookahead horizon, local processing can continue without global coordination.

Optimistic speculations and rollback Optimistic approaches allow local processing beyond the local horizon and use rollbacks when late events invalidate earlier actions. Correctness requires maintaining a reversible log of state modifications and atomic check points.

Algorithm 2 Event-Aware Simulation Step

```
Require: State $\sigma$, time $t$, events $\mathcal{E}$
 1: Compute propensities $a_j(\sigma)$, total $a_0 = \sum_j a_j$
 2: Sample tentative waiting time $\tau \sim \text{Exp}(a_0)$
 3: $t_{next} \leftarrow t + \tau$
 4: for each time-triggered event $E_i$ with trigger time $t_i \in (t, t_{next}]$ do
 5:     $t_{next} \leftarrow \min(t_{next}, t_i)$
 6: end for
 7: if event triggers at $t_{next}$ then
 8:     Apply event actions to $\sigma$
 9:     return $(\sigma', t_{next})$                              ▷ Event step
10: else
11:     Select and apply reaction
12:     Check observable-based triggers on new state
13:     Apply any triggered events
14:     return $(\sigma', t_{next})$                              ▷ Reaction step
15: end if
```

Suggested algorithm sketch Use a hybrid approach: conservative synchronization for time-triggered events, and optimistic batching for short reaction bursts with frequent commit checks. This balances latency and throughput while ensuring event ordering invariants.

14.6 Event Actions: Detailed Semantics

Definition 14.10 (Parameter Modification Action)**.** The action $p := e$ modifies the parameter environment:

$$\mathcal{P}' = \mathcal{P}[p \mapsto [\![e]\!]_{\sigma,\mathcal{P}}]$$

This affects all subsequent rate evaluations that reference p.

Definition 14.11 (Species Modification Actions)**.** For species S with current count n_S:

$$\begin{aligned} \textit{apply}(\mathsf{add}(S,k),\sigma) &: n_S \leftarrow n_S + k \\ \textit{apply}(\mathsf{remove}(S,k),\sigma) &: n_S \leftarrow \max(0, n_S - k) \\ \textit{apply}(\mathsf{set}(S,k),\sigma) &: n_S \leftarrow k \end{aligned}$$

Remark 14.1 (Events and Detailed Balance). Events generally break detailed balance and can drive the system away from equilibrium. If events modify populations or parameters discontinuously, the system is no longer a homogeneous CTMC but rather a piecewise-homogeneous process with event-driven regime changes.

14.7 Zeno Behavior and Well-Formedness

Definition 14.12 (Non-Zeno Condition)**.** A BNGL model with events is *non-Zeno* if there exists $\epsilon > 0$ such that the time between consecutive event triggers is bounded below by ϵ almost surely. Equivalently, only finitely many events can fire in any bounded time interval.

Proposition 14.1 (Sufficient Conditions for Non-Zeno)**.** *The model is non-Zeno if:*

1. *All events have* `once` *option (fire at most once), or*

2. *Observable-based triggers have hysteresis: the post-event state does not immediately re-trigger the same event, or*

3. *Time-based triggers have strictly increasing trigger times*

15 Molecule Deletion Semantics

The `DeleteMolecules` rule option enables rules to delete molecules even when they have bonds to molecules outside the rule's reactant patterns. This section formalizes the precise semantics of deletion with dangling bond handling.

15.1 The Dangling Bond Problem

Definition 15.1 (Dangling Bond)**.** A *dangling bond* arises when a rule deletes a molecule m that has a bond to a molecule m' where $m' \notin L$ (i.e., m' is not part of the matched pattern). After deletion of m, molecule m' would have a component with a bond specification pointing to a non-existent molecule.

Definition 15.2 (Gluing Condition Violation)**.** In DPO rewriting terms, a dangling bond corresponds to violation of the *dangling condition* (Definition 4.5): an edge in $G \setminus m(L)$ is incident to a node in $m(L \setminus l(K))$.

15.2 DeleteMolecules Option Semantics

Definition 15.3 (DeleteMolecules Mode)**.** When a rule r has the `DeleteMolecules` option and deletes molecule m:

1. Identify all bonds $(m.c, m'.c')$ where $m' \notin \text{img}(\mu)$
2. For each such bond, set $\lambda(m'.c').\beta \leftarrow \textbf{free}$
3. Delete m and all its components from the graph
4. The molecule m' remains in the system with component c' now unbound

Remark 15.1 (Implicit Side Effects). The `DeleteMolecules` option introduces *implicit side effects*: the rule modifies molecules (m') that are not explicitly mentioned in the rule pattern. This is the key semantic difference from standard DPO rewriting, where all modifications must be explicitly specified.

Definition 15.4 (Extended Operation Sequence)**.** With `DeleteMolecules`, the operation sequence for a rule includes additional implicit operations:

1. Compute explicit operations from $L \to R$ correspondence
2. For each molecule m to be deleted:
   (a) For each bond $(m.c, m'.c')$ with $m' \notin \text{img}(\mu)$:
   (b) Prepend DeleteBond(m.c, m'.c') to the sequence
3. Append DeleteMol(m)

15.3 Cascade Deletion

Definition 15.5 (Cascade Deletion)**.** When `DeleteMolecules` severs a bond, the resulting free site on m' may trigger further deletions if:

1. Another rule has a pattern requiring $m'.c'$ to be bonded
2. The loss of the bond destabilizes a complex (application-specific)

In standard BNGL, cascade deletion is *not* automatic; each deletion is independent. Complex destabilization must be modeled explicitly with additional rules.

15.4 Comparison with SqPO Semantics

Proposition 15.1 (DeleteMolecules as Partial SqPO). *The `DeleteMolecules` semantics is equivalent to sesqui-pushout (SqPO) rewriting for the deletion component: dangling edges are implicitly removed rather than blocking the derivation.*

Proof. In SqPO, the final pullback complement (FPC) construction handles dangling edges by removing them. For a match $m : L \to G$ with dangling edges, the FPC D is computed as G with: (1) nodes in $m(L \setminus l(K))$ removed, and (2) edges incident to removed nodes also removed. This exactly matches the `DeleteMolecules` behavior of freeing bond partners. □

Remark 15.2 (DPO vs. SqPO in BNGL). Without `DeleteMolecules`, BNGL uses DPO semantics: rules with potential dangling bonds simply don't match (the gluing condition fails). With `DeleteMolecules`, BNGL shifts to SqPO-like semantics for the affected molecules. A single model can mix both behaviors across different rules.

15.5 Well-Formedness with Deletion

Definition 15.6 (Deletion-Safe Rule). A rule r is *deletion-safe* if either:

1. No molecules are deleted ($L \subseteq \mathrm{dom}(\phi_r)$), or
2. The `DeleteMolecules` option is set, or
3. All deleted molecules have only bonds to other molecules in L

A model is deletion-safe if all rules are deletion-safe.

Proposition 15.2 (Deletion Safety and DPO Applicability). *For a deletion-safe model without `DeleteMolecules`, every rule application satisfies the DPO gluing conditions, and the three-way equivalence theorem (Theorem 33.4) applies directly.*

16 Reactant Inclusion and Exclusion Modifiers

BNGL supports rule modifiers that constrain which species can participate as reactants, beyond the structural constraints imposed by the pattern itself.

16.1 Syntax and Static Semantics

Definition 16.1 (Reactant Modifiers). A rule may include the following modifiers:

- `exclude_reactants(`i`, `P_1`, ..., `P_k`)`: Reactant position i must not match any of patterns $P_1, \ldots, P_k$
- `include_reactants(`i`, `P_1`, ..., `P_k`)`: Reactant position i must match at least one of $P_1, \ldots, P_k$

These modifiers constrain the joint mapping set without changing the rule's transformation.

16.2 Operational Semantics

Definition 16.2 (Modified Joint Mapping Set). For rule r with reactant patterns $[Q_1, \ldots, Q_n]$ and modifiers $\mathcal{C} = \{c_1, \ldots, c_m\}$, the *modified joint mapping set* is:

$$\mathcal{J}_r^{\mathcal{C}}(\sigma) = \{(\mu_1, \ldots, \mu_n) \in \mathcal{J}_r(\sigma) \mid \forall c \in \mathcal{C} : \textit{satisfies}((\mu_1, \ldots, \mu_n), c)\}$$

16.3 Relationship to Pattern Refinement

Proposition 16.1 (Inclusion as Pattern Union)**.** *The modifier* `include_reactants(i, ` P_1`, ...,` P_k`)` *is semantically equivalent to replacing rule r with k rules $r_1, \ldots, r_k$ where rule r_j has reactant pattern $Q_i \wedge P_j$ (conjunction/intersection) at position i, provided the patterns are disjoint.*

16.4 Examples

Example 16.1 (Excluding Dimers)*.* To model a reaction that only occurs for monomeric receptors:

```
R(l) + L(r) -> R(l!1).L(r!1) kf \
    exclude_reactants(1, R(d!+))
```

This excludes any receptor that has a bond on its dimerization site `d`.

Example 16.2 (Requiring Scaffold)*.* To model a reaction requiring a scaffold protein:

```
Kin(S~U) -> Kin(S~P) kp \
    include_reactants(1, Kin(scaff!1).Scaffold(k!1))
```

Phosphorylation only occurs when the kinase is bound to a scaffold.

17 Extended Compartment Semantics

This section extends Section 12.1 with detailed treatment of geometric factors, surface-volume interfaces, and transport kinetics.

17.1 Compartment Geometry

Definition 17.1 (Compartment Geometry Specification)**.** A compartment $\kappa \in \mathcal{K}$ is specified by:

- $\mathit{dim}(\kappa) \in \{2, 3\}$: dimensionality (surface or volume)
- $\mathit{size}(\kappa) \in \mathbb{R}_{\geq 0}$: area (if dim = 2) or volume (if dim = 3)
- $\mathit{parent}(\kappa) \in \mathcal{K} \cup \{\perp\}$: enclosing compartment
- $\mathit{outside}(\kappa) \in \mathcal{K} \cup \{\perp\}$: adjacent exterior compartment

17.2 Concentration Units and Scaling

Definition 17.2 (Compartment-Specific Concentration)**.** For species S in compartment κ with copy number n_S:

$$[S]_\kappa = \frac{n_S}{N_A \cdot \mathit{size}(\kappa)}$$

where N_A is Avogadro's number and $\mathit{size}(\kappa)$ has appropriate units (L for volume, dm^2 for area).

17.3 Propensity Scaling Factors

Definition 17.3 (Geometric Propensity Factor)**.** For rule r with reactant patterns in compartments $(\kappa_1, \ldots, \kappa_n)$, the geometric factor $g_r(\kappa_1, \ldots, \kappa_n)$ converts microscopic rate constant to propensity:

$$a_r(\sigma) = k_r \cdot g_r(\kappa_1, \ldots, \kappa_n) \cdot N_r(\sigma)$$

Proposition 17.1 (Geometric Factor Formulas)**.** *For common reaction types:*
Unimolecular (any compartment):

$$g_r(\kappa) = 1$$

Bimolecular, same 3D compartment *κ_V:*

$$g_r(\kappa_V, \kappa_V) = \frac{1}{N_A \cdot vol(\kappa_V)}$$

Bimolecular, same 2D compartment *κ_S:*

$$g_r(\kappa_S, \kappa_S) = \frac{1}{N_A \cdot area(\kappa_S)}$$

Bimolecular, 3D-2D interface *(molecule in κ_V binds to membrane-bound molecule in κ_S):*

$$g_r(\kappa_V, \kappa_S) = \frac{1}{N_A \cdot vol(\kappa_V)}$$

17.4 Transport Rules

Definition 17.4 (Transport Rule)**.** A *transport rule* moves molecules between compartments:

$$S@\kappa_1 \to S@\kappa_2$$

where the molecule type and internal state are preserved but the compartment annotation changes.

17.5 MoveConnected Option

Definition 17.5 (MoveConnected Semantics)**.** The `MoveConnected` rule option specifies that when a molecule moves compartments, all molecules in its connected complex move together:

$$apply(\textsc{MoveMol}(m, \kappa_1 \to \kappa_2), G)_{\texttt{MoveConnected}} = \bigcup_{m' \in [m]_\sim} \textsc{MoveMol}(m', \kappa_1 \to \kappa_2)$$

where $[m]_\sim$ is the connected component containing m.

17.6 Cross-Compartment Reactions

Definition 17.6 (Interface Reaction)**.** An *interface reaction* has reactants in adjacent compartments:

$$A@\kappa_1 + B@\kappa_2 \to C@\kappa_3$$

where $(\kappa_1, \kappa_2) \in adj$ (adjacent compartments).

17.7 Compartment-Specific Observables

Definition 17.7 (Compartment-Filtered Observable)**.** An observable can be restricted to specific compartments:

$$[\![O@\kappa]\!](\sigma) = \sum_{S \in \mathcal{S} : S@\kappa} n_S \cdot w_S$$

where the sum is only over species localized in compartment κ.

Algorithm 3 Bond Formation with Complex Update

```
Require: Agents $a_1, a_2$, components $c_1, c_2$
1: $C_1 \leftarrow \mathit{find}(a_1)$; $C_2 \leftarrow \mathit{find}(a_2)$
2: if $C_1 \neq C_2$ then
3:     $\mathit{union}(C_1, C_2)$                                    ▷ Merge complexes
4:     $C_{new} \leftarrow \mathit{find}(a_1)$
5:     $\mathit{molecules}(C_{new}) \leftarrow \mathit{molecules}(C_1) \cup \mathit{molecules}(C_2)$
6:     Invalidate $\mathit{canonical}(C_{new})$
7:     Update observable caches
8: end if
9: Create bond $(a_1.c_1, a_2.c_2)$
```

Algorithm 4 Bond Breaking with Potential Split

```
Require: Bond $(a_1.c_1, a_2.c_2)$
1: Remove bond from data structure
2: Check connectivity: BFS/DFS from $a_1$ avoiding $a_2$
3: if $a_2$ not reachable from $a_1$ then
4:     Split into $C_1$ (containing $a_1$) and $C_2$ (containing $a_2$)
5:     Create new complex records
6:     Update observable caches for both
7: end if
```

18 NFsim Data Structures and Algorithms

This section formalizes the key data structures used in network-free simulation as implemented in NFsim [9].

18.1 Complex Bookkeeping

Definition 18.1 (Complex Data Structure)**.** A *complex* C in NFsim is represented by:

- $\mathit{molecules}(C)$: set of molecule agent IDs
- $\mathit{bonds}(C)$: set of $(a_1.c_1, a_2.c_2)$ pairs encoding bonds
- $\mathit{canonical}(C)$: cached canonical form for species identification
- $\mathit{observables}(C)$: cached observable contributions

Definition 18.2 (Union-Find for Complex Tracking)**.** Complex membership is tracked using a union-find (disjoint-set) data structure with:

- $\mathit{find}(a)$: returns the representative molecule ID for agent a's complex
- $\mathit{union}(a_1, a_2)$: merges complexes when a bond forms between a_1 and a_2
- $\mathit{split}(a_1, a_2)$: potentially splits complexes when a bond breaks

18.2 Reactant Lists

Definition 18.3 (Reactant List)**.** For rule r with n reactant patterns, the *reactant list* structure contains:

- $\mathcal{L}_r[i]$: indexed list of all valid mappings $\mu_i : P_i \hookrightarrow \mathcal{G}$ for the i-th reactant pattern

- $count_r[i] = |\mathcal{L}_r[i]|$: cached count
- $index_r$: data structure for fast lookup by molecule type

18.3 Incremental Match Update

Definition 18.4 (Affected Rule Set). After applying rule r via mapping μ, the *affected rule set* is:

$$Affected(r, \mu) = \{r' \in \mathcal{R} : types(P_{r'}) \cap types(Affected(\mu)) \neq \emptyset\}$$

Only rules in this set need reactant list updates.

18.4 Observable Evaluation

Definition 18.5 (On-the-Fly Observable). Observables can be computed on-the-fly by pattern matching against all complexes:

$$[\![O]\!](\sigma) = \sum_{C \in Complexes} count(P_O, C, type_O)$$

For `Molecules` type: count all embeddings within each complex. For `Species` type: count complexes with at least one embedding.

18.5 Propensity Computation

Definition 18.6 (NFsim Propensity Structure). The propensity calculation maintains:

- a_r: propensity for each rule r
- $a_0 = \sum_r a_r$: total propensity
- Priority queue or tree structure for $O(\log |\mathcal{R}|)$ rule selection

18.6 Parallel and GPU-Friendly Data Structures for NFsim

To scale network-free simulation to many-core or GPU architectures, data structures must support high-throughput concurrent updates with minimal synchronization.

Lock-free union-find Adopt a lock-free union-find variant for complex tracking where unions are implemented with atomic compare-and-swap on representative pointers and path-compression is done lazily. Splits (bond-breaking) are handled by versioned connectivity checks using epoch-based reclamation.

Batched bond operations Group bond additions/removals into batched kernels that update adjacency arrays in bulk, reducing synchronization and enabling vectorized execution on GPUs.

18.7 Performance and Complexity Benchmarks

We recommend reporting the following benchmarks: match update throughput, complex merge/split latency, and end-to-end SSA steps per second versus agent count. Typical complexity:

- Matching: $O(|TypeIndex[M]| \cdot |P|)$ per pattern
- Reactant-list update: $O(k \cdot s^{|P|})$ per affected rule
- Union-find operations: effectively amortized $\alpha(n)$ per operation

Workload	Agents	Throughput (steps/s)
Small complexes	10^4	1e5
Large complexes	10^5	2e4
GPU-batched	10^6	5e5

19 Observable Algebra

We develop the algebraic structure of BNGL observables and their role in defining equivalence relations on states.

19.1 Observable Space

Definition 19.1 (Observable Vector Space)**.** Let $\mathbf{O}$ be the real vector space spanned by pattern-based observables:

$$\mathbf{O} = \mathrm{span}_{\mathbb{R}}\{\mathsf{count}(P, \cdot) : P \text{ is a pattern}\}$$

Elements of $\mathbf{O}$ are linear combinations of pattern counts.

Implementation touchpoints (BioNetGen). Top-level action parsing and run modes live in `bng2/BNG2.pl`. Core option parsing is in `bng2/Perl2/BNGOptions.pm`. Functions, observables, and rate laws are represented in `bng2/Perl2/Function.pm`, `bng2/Perl2/Observable.pm`, and `bng2/Perl2/RateLaw.pm`.

Definition 19.2 (Action-script syntax (abstract))**.** An action script is a sequence of commands generated by

$$C ::= \mathsf{GenerateNetwork} \mid \mathsf{Simulate}(T) \mid \mathsf{ParamScan}(p, \mathcal{V}, C) \mid C_1; C_2$$

where $T \in \mathbb{R}_{\geq 0}$ is a time horizon, p is a parameter name, and $\mathcal{V}$ is a finite set of parameter values.

Definition 19.3 (Big-step action semantics)**.** Write $(\mathcal{B}, \mathsf{rng}) \Downarrow_C \mathcal{A}$ for executing command C on input model and RNG state to produce an artifact state $\mathcal{A}$. The semantics is defined by cases (composition for $C_1; C_2$ and iteration for parameter scans).

19.2 Static analysis from AR graphs

The atom-rule graph enables static analyses that do not require network generation; these can be stated and proved as sound over-approximations with respect to the CTMC semantics.

Initial atoms. Let $\mathit{initAtoms}(\Sigma_0)$ denote the multiset of atom instances present in the initial condition.

Dead-rule detection (sound). A rule r is *AR-unreachable* if some atom required by r (reactant or context) is not reachable from $\mathit{initAtoms}(\Sigma_0)$ in the directed bipartite graph $\mathrm{AR}(\mathcal{B})$. Then r cannot fire in any reachable state.

Theorem 19.1 (AR dead-rule soundness)**.** *Let $r \in \mathcal{R}$. If there exists an atom type α such that (i) α is required by r (reactant or context edge into r) and (ii) α is not reachable from $\mathit{initAtoms}(\Sigma_0)$ in* $\mathrm{AR}(\mathcal{B})$*, then for every state σ reachable from Σ_0 we have $N_r(\sigma) = 0$.*

Proof. Reachability in $\mathrm{AR}(\mathcal{B})$ over-approximates derivability of atoms under rule firing: product edges add atoms and reactant/context edges express requirements. If α is not reachable from the initial atoms, then no sequence of rule firings can create an instance of α starting from Σ_0. Since r requires α, it cannot match, hence $N_r(\sigma) = 0$. □

Theorem 19.2 (AR observable-atom reachability (sound))**.** *Let P be a pattern and let $\alpha \in$ atoms(P). If α is not reachable from initAtoms(Σ_0) in* $\mathrm{AR}(\mathcal{B})$*, then for every reachable state σ we have count$(P, \sigma, \texttt{Molecules}) = 0$ and count$(P, \sigma, \texttt{Species}) = 0$.*

Definition 19.4 (AR reachability closure)**.** Let R_0 be the set of atom types reachable from *initAtoms*(Σ_0) in $\mathrm{AR}(\mathcal{B})$. Define a monotone operator F on subsets of atom types by

$$F(X) \triangleq X \cup \{\alpha : \exists r \in \mathcal{R}.\ (\forall \beta \in \mathit{Req}(r)\ \beta \in X)\ \wedge\ (r, \alpha) \in E_{\mathsf{prod}}\},$$

where $\mathit{Req}(r)$ is the set of reactant/context atom types required by r. Then the least fixed point $X^* = \mathrm{lfp}(F)$ is an explicit "derivable atoms" over-approximation computable by Kleene iteration.

Proposition 19.3 (Monotonicity and convergence)**.** *F is monotone on the complete lattice $\mathcal{P}(\mathcal{A})$. Hence $\mathrm{lfp}(F) = \bigcup_{n \geq 0} F^n(R_0)$.*

Theorem 19.4 (AR closure soundness (atom existence))**.** *Let α be an atom type not contained in $\mathrm{lfp}(F)$. Then for every state σ reachable from Σ_0, the total number of instances of α in σ is zero.*

Remark 19.1 (Galois connection viewpoint). Define $\Gamma : \mathcal{P}(\mathcal{A}) \to \mathcal{P}(\mathcal{R})$ by $\Gamma(X) = \{r : \mathit{Req}(r) \subseteq X\}$ (rules enabled by a set of atoms), and $\Delta : \mathcal{P}(\mathcal{R}) \to \mathcal{P}(\mathcal{A})$ by $\Delta(R) = \bigcup_{r \in R} \mathit{Prod}(r)$ (atoms producible by a set of rules). Then $F(X) = X \cup \Delta(\Gamma(X))$ is a standard closure operator arising from this Galois connection between "available atoms" and "enabled rules."

Definition 19.5 (AR abstraction as an abstract interpretation)**.** Let $\mathcal{C}$ be the concrete transition system of BNGL states under rule application, and let $\mathcal{A} = \mathcal{P}(\mathcal{A})$ be the abstract domain of atom-type sets. Define abstraction $\alpha(\sigma) = \{\alpha : \#(\alpha, \sigma) > 0\}$ and concretization $\gamma(X) = \{\sigma : \alpha(\sigma) \subseteq X\}$. Then F is an abstract post operator induced by the AR graph.

Theorem 19.5 (AR closure as a sound over-approximation)**.** *For every reachable concrete state σ and $X^* = \mathrm{lfp}(F)$, we have $\alpha(\sigma) \subseteq X^*$. Equivalently, X^* is a sound abstract invariant.*

Conservation candidates. If an atom type α never appears as a reactant or product edge for any rule (only as context), then its total count is invariant along all executions; this refines Proposition 29.3 into a computable sufficient condition.

Modularity via SCCs. Strongly connected components of $\mathrm{AR}(\mathcal{B})$ define candidate "modules". A precise modular semantics can be phrased as a factorization of the generator into interacting subgenerators, with proof obligations stating when modular abstractions preserve a chosen class of observables.

19.3 Martingale and generator foundations for BNGL CTMCs

The semantics throughout this paper can be phrased in a standard Markov-process language: BNGL induces a generator Q on a (typically countable) state space Ω of molecular mixtures (or their abstractions). This viewpoint unlocks a large toolkit from stochastic analysis.

Definition 19.6 (Trajectory space)**.** Fix a time horizon $T > 0$. Let $D([0, T], \Omega)$ denote the Skorokhod space of right-continuous paths with left limits (càdlàg paths) taking values in Ω. We equip $D([0, T], \Omega)$ with its Borel σ-algebra. A BNGL execution semantics (SSA/NF/hybrid) therefore denotes a probability measure $\mathbb{Q}$ on $D([0, T], \Omega)$, and an observation operator Φ is a measurable map (or Markov kernel) out of this space.

Remark 19.2 (Measurability of specifications). Temporal-logic satisfaction (Section 32) and event-time functionals (first-passage, dwell times) can be treated as measurable maps $D([0, T], \Omega) \to \{0, 1\}$ or $D([0, T], \Omega) \to \mathbb{R}$. Thus statements about verification and identifiability reduce to standard pushforward-measure reasoning.

Definition 19.7 (Generator acting on observables)**.** Let X_t be the BNGL-induced CTMC with generator Q. For any bounded observable $f : \Omega \to \mathbb{R}$, define

$$(Qf)(x) \triangleq \sum_{y \neq x} Q_{x,y} \, (f(y) - f(x)).$$

In a rule-based presentation, the sum ranges over all rule instances that map state x to y with their corresponding rates.

20 Spectral Analysis and Mixing Times

We analyze the spectral properties of the generator Q and their consequences for relaxation to stationarity.

Definition 20.1 (Spectral Gap)**.** Let $\lambda_0 = 0 > \lambda_1 \geq \lambda_2 \geq \cdots$ denote the eigenvalues of Q ordered by real part (for a finite-state irreducible CTMC). The *spectral gap* is $\gamma \triangleq -\Re(\lambda_1) > 0$.

Theorem 20.1 (Exponential Ergodicity via Spectral Gap)**.** *If the CTMC is irreducible on a finite state space and has spectral gap $\gamma > 0$, then for any initial distribution μ_0 and stationary distribution π,*

$$\|\mu_0 P_t - \pi\|_{\mathrm{TV}} \leq C \, e^{-\gamma t}$$

for some constant C depending on μ_0 and π.

Sketch. The result follows from spectral decomposition of the semigroup $P_t = e^{tQ}$ and bounding the contribution of non-zero eigenmodes by $e^{-\gamma t}$. □

Remark 20.1 (Poincaré inequality)*.* A functional form of the spectral gap is the Poincaré inequality: for any f with $\pi(f) = 0$,

$$\mathrm{Var}_\pi(f) \leq \frac{1}{\gamma} \mathcal{E}(f, f)$$

where $\mathcal{E}$ is the Dirichlet form associated to Q.

21 Large Deviations and Rare Events

We state a Wentzell–Freidlin type large deviations principle for scaled BNGL models in the thermodynamic limit and describe consequences for rare-event rates.

Theorem 21.1 (Informal LDP for Scaled CTMCs)**.** *Consider a family of BNGL CTMCs $X^{(N)}$ with state scaling N and generator $Q^{(N)}$ such that propensities scale appropriately (mass-action type). Then, under mild regularity, the trajectories satisfy a large-deviation principle on $D([0,T], \mathbb{R}^d)$ with good rate function*

$$S_T(\phi) = \int_0^T L(\phi(t), \dot{\phi}(t)) \, dt$$

where L is the local Lagrangian obtained by the Legendre transform of the Hamiltonian associated to the jump rates.

Remark 21.1 (Arrhenius-like asymptotics)*.* The LDP yields exponential asymptotics for exit times from metastable domains: $\mathbb{E}[\tau] \asymp \exp(N \, I)$ where I is the minimal action to exit. This recovers classical Arrhenius-type rates in energy-driven models.

Algorithm 5 Weighted Ensemble (single cycle)

```
Require: Ensemble $\{(X^{(k)}, w^{(k)})\}_{k=1}^{M}$, bins $\{B_i\}$, $\Delta t$
1: Propagate each trajectory for time $\Delta t$ using SSA/NF/hybrid
2: Partition trajectories by bin membership
3: for each bin $B_i$ do
4:     Compute total weight $W_i = \sum_{k:X^{(k)} \in B_i} w^{(k)}$
5:     Resample to obtain $n_i$ trajectories with weights summing to $W_i$ (splitting/merging)
6: end for
7: Continue
```

22 Rare-Event Sampling and Weighted Ensemble (WESTPA)

Rare transitions (e.g., complex breakup, large cluster formation) are often inaccessible to direct simulation. Importance-sampling and splitting methods provide practical estimators with controlled variance. The *Weighted Ensemble* (WE) approach [?, ?] partitions state-space into bins and periodically clones/merges trajectories to concentrate sampling in low-probability regions while exactly preserving unbiasedness.

22.1 Weighted Ensemble for BNGL

Definition 22.1 (Weighted ensemble scheme)**.** A WE simulation consists of:

- A binning of state space (or a progress coordinate) $\{B_i\}$
- An ensemble of M weighted trajectories $(X_t^{(k)}, w_t^{(k)})$
- A propagation interval Δt after which resampling (splitting/merging) occurs

At each resampling step: trajectories in each bin are split or merged to maintain per-bin target counts while adjusting weights so that the total weight is preserved.

22.2 Unbiasedness and Estimation

Proposition 22.1 (Unbiased estimation)**.** *For any path-functional $H(\omega)$ (e.g., indicator of hitting target set by time T), the WE estimator*

$$\hat{\mathbb{E}}[H] = \sum_{k=1}^{M} w^{(k)} H(\omega^{(k)})$$

is unbiased for the true expectation under the process measure, provided weight updates preserve total weight and resampling is conditional on current ensemble only (no lookahead).

Sketch. WE is a sequential importance sampling scheme with resampling steps that are weight-preserving; linearity of expectation and the weight invariance imply unbiasedness for any functional measurable with respect to ensemble trajectories. □

22.3 Connection to LDP and Action Minimization

WE implicitly focuses sampling effort along low-action paths in the sense that bins and splitting amplify trajectories that move toward the rare set. For models where the LDP rate functional S_T is available, one can design progress coordinates approximating the action to optimize sampling efficiency (e.g., use quasi-potential or committor approximations for binning).

22.4 Practical integration with WESTPA and BNGL

WESTPA is an open-source WE implementation that can be coupled to BNGL simulators: run short SSA/NF bursts as propagation kernels, use progress coordinates (aggregate observables or cluster size) for binning, and let WESTPA handle resampling and weight bookkeeping. Implementation notes:

- Propagate using `nf extunderscore sim` API or spawn hybrid steps for bins representing abundant regimes
- Use on-the-fly observable evaluations (Section 18) to determine bins
- For rate estimation (escape flux), accumulate weighted counts crossing the target boundary and average over cycles

Variance reduction and adaptive bins Adaptive binning (refining bins along promising trajectories) and stratified resampling reduce variance; rigorous control requires monitoring effective sample size per bin and adjusting target counts accordingly.

References and tools Westpa: `https://westpa.github.io/`; foundational papers include Huber & Kim (1996) and Zẃier et al. (2015). See also splitting-based rare-event literature for theoretical guarantees.

23 Diffusion Limits and Fluctuations (FCLT)

We present the functional central limit theorem for fluctuations about the mean-field ODE limit.

Theorem 23.1 (FCLT for BNGL models). *Under a thermodynamic scaling $N \to \infty$ and assuming well-posedness of the mean-field ODE $\dot{x} = F(x)$, the fluctuation process $\xi^{(N)}(t) = \sqrt{N}(X^{(N)}(t)/N - x(t))$ converges in distribution in $D([0,T], \mathbb{R}^d)$ to a Gaussian diffusion solving the linear SDE:*

$$d\xi_t = DF(x(t))\xi_t \, dt + B(x(t)) \, dW_t$$

where DF is the Jacobian and $BB^\top$ is the instantaneous covariance from the underlying jump noise.

Sketch. Use the martingale representation for the scaled process, verify tightness, and apply the martingale central limit theorem to obtain convergence to a Gaussian martingale with prescribed quadratic variation. □

24 Ergodicity and Foster–Lyapunov Criteria

We give checkable sufficient conditions for positive recurrence and ergodic convergence based on Foster–Lyapunov drift conditions.

Theorem 24.1 (Positive recurrence via Lyapunov drift). *Let Q be a CTMC generator on countable state space Ω. If there exists a function $V : \Omega \to [1, \infty)$ and constants $b < \infty$, $\epsilon > 0$, and a finite set K such that*

$$(QV)(x) \leq -\epsilon V(x) + b\mathbf{1}_K(x) \quad \textit{for all } x \in \Omega,$$

then the chain is non-explosive, positive recurrent, and has a unique stationary measure with geometric ergodicity properties.

Remark 24.1 (Constructing Lyapunov functions). Natural choices for V include polynomial or exponential functions of molecule counts; energy-like sums are often effective in biochemical models.

25 Error Bounds for Tau-Leap and Hybrid Approximations

We provide a weak error bound for tau-leaping and hybrid time-step approximations under Lipschitz propensities.

Theorem 25.1 (Weak error bound for tau-leap). *Suppose propensities $a_j(x)$ are globally Lipschitz and bounded on the relevant state region. If a tau-leap step of size τ is used with adequate pre/post checks, then for observables f with polynomial growth there exists $C(T, f)$ such that the weak error satisfies*

$$|\mathbb{E}[f(X_T)] - \mathbb{E}[f(\tilde{X}_T)]| \le C(T, f)\,\tau.$$

Remark 25.1 (Hybrid schemes). Hybrid schemes that mix exact SSA for stiff/rare parts with tau-leap for abundant parts inherit similar first-order weak error bounds under compatible step-size control and correct switching semantics.

26 Log-Sobolev and Entropy Dissipation

We briefly note how log-Sobolev inequalities relate to exponential convergence in relative entropy (stronger than total variation bounds).

Proposition 26.1 (Log-Sobolev implies exponential entropy decay). *If the generator satisfies a log-Sobolev inequality with constant $\rho > 0$, then for any initial distribution μ_0,*

$$\mathrm{D}_{\mathrm{KL}}(\mu_t \| \pi) \le e^{-2\rho t}\mathrm{D}_{\mathrm{KL}}(\mu_0 \| \pi)$$

where D_{KL} is the Kullback–Leibler divergence.

Remark 26.1. Establishing log-Sobolev constants for BNGL models is challenging, but energy-based models with strong confining potentials are promising candidates.

Theorem 26.2 (Existence and non-explosion (sufficient condition)). *Assume there exists a function $V : \Omega \to [1, \infty)$ and constants $b < \infty$, $\epsilon > 0$, and a finite set $K \subset \Omega$ such that*

$$(QV)(x) \le -\epsilon\, V(x) + b\,\mathbf{1}_K(x) \quad \text{for all } x \in \Omega.$$

Then the BNGL-induced pure-jump process is non-explosive and admits a unique minimal CTMC with generator Q.

Theorem 26.3 (Dynkin's formula). *For bounded f and any $t \ge 0$,*

$$\mathbb{E}_x[f(X_t)] = f(x) + \mathbb{E}_x\left[\int_0^t (Qf)(X_s)\,ds\right].$$

Theorem 26.4 (Martingale problem). *For bounded f, the process*

$$M_t^f \triangleq f(X_t) - f(X_0) - \int_0^t (Qf)(X_s)\,ds$$

is a martingale with respect to the natural filtration.

Remark 26.2 (Rule-wise decomposition). If $\mathcal{R} = \{r\}$, then $Q = \sum_{r\in\mathcal{R}} Q^{(r)}$ where each $Q^{(r)}$ contributes only transitions due to rule r. This yields a canonical decomposition of M_t^f into compensated counting processes per rule channel, matching how BioNetGen simulators organize reaction selection.

26.1 Spectral and sensitivity theory (finite networks)

When network generation terminates, the CTMC is finite-state (Ω finite). Then Q is a finite matrix and spectral analysis becomes available.

26.2 Coupling, Wasserstein Contraction, and Mixing

We consider couplings of CTMCs and contraction in Wasserstein distances.

Definition 26.1 (Wasserstein distance on Ω)**.** Let $d : \Omega \times \Omega \to [0, \infty)$ be a metric on states (e.g., total molecule-count norm). The 1-Wasserstein distance between probability measures μ, ν is

$$W_d(\mu, \nu) = \inf_{\xi \in \Pi(\mu,\nu)} \sum_{x,y} d(x, y)\, \xi(x, y)$$

where $\Pi(\mu, \nu)$ are couplings with marginals μ, ν.

Theorem 26.5 (Generator coupling contraction)**.** *Suppose there exists a coupling generator $\mathcal{L}$ acting on $\Omega \times \Omega$ such that for the distance function d,*

$$(\mathcal{L}d)(x, y) \le -\kappa\, d(x, y)$$

for all $x \neq y$ and some $\kappa > 0$. Then for all $t \ge 0$ and probability measures μ_0, ν_0,

$$W_d(\mu_0 P_t, \nu_0 P_t) \le e^{-\kappa t} W_d(\mu_0, \nu_0).$$

In particular, the process is exponentially mixing in W_d at rate κ.

Sketch. Let Ξ_t be the coupling of processes with generator $\mathcal{L}$ and initial coupling achieving $W_d(\mu_0, \nu_0)$. By Dynkin's formula, $\mathbb{E}[d(X_t, Y_t)] = \mathbb{E}[d(X_0, Y_0)] + \mathbb{E}[\int_0^t (\mathcal{L}d)(X_s, Y_s) ds]$, and the differential inequality yields the exponential bound. $\square$

Remark 26.3. Constructing $\mathcal{L}$ reduces to coupling individual jump channels: for local reaction events, design synchronous couplings that either act on both copies or on one copy with compensating probability. For mass-action models with dominant mixing reactions, κ is often bounded away from zero.

26.3 Quantitative Bisimulation Metrics

We formalize a behavioral pseudometric that quantifies observational indistinguishability between states.

Definition 26.2 (Behavioral pseudometric)**.** Define a pseudometric $m : \Omega \times \Omega \to [0, 1]$ as the greatest fixed point of the monotone operator $\mathcal{F}$:

$$\mathcal{F}(m)(x, y) = \inf_{\pi \in \Pi(Q(x,\cdot), Q(y,\cdot))} \sum_{x',y'} m(x', y')\, \pi(x', y')$$

where $Q(x, \cdot)$ denotes the jump-rate measure out of x (normalized by escape rate) and $\Pi(\cdot, \cdot)$ are couplings of these measures.

Proposition 26.6 (Approximation and fixed-point iteration)**.** *On finite Ω, iterating $m_{n+1} = \mathcal{F}(m_n)$ starting from the top function $m_0 \equiv 1$ converges monotonically to the greatest fixed point m^*. Each iteration requires solving optimal transport problems (linear programs) between pairs of distributions; efficient specialized solvers accelerate this.*

Theorem 26.7 (Observable error bound via metric). *Let m be a bisimulation pseudometric and let $f : \Omega \to \mathbb{R}$ be 1-Lipschitz w.r.t. the state metric underlying m. Then for any x, y and time t,*

$$|\mathbb{E}_x[f(X_t)] - \mathbb{E}_y[f(X_t)]| \leq e^{-\kappa t} m(x, y)$$

where κ is any Wasserstein contraction rate (Theorem 26.5).

Remark 26.4 (Practical use). Truncating m at threshold ϵ yields an ϵ-bisimulation coarsening; merging states within each cluster incurs bounded error on all Lipschitz observables, facilitating approximate model reduction with guarantees.

Example (two-state birth-death) Consider states $0, 1$ with symmetric birth/death rates; explicit coupling shows κ equals the minimal common rate and $m(0, 1) = 1$, giving explicit mixing estimates.

26.4 Connections and implementation notes

Computing m scales quadratically in $|\Omega|$ per iteration; in practice, symbolic exploitation of rule structure and sparsity in couplings is essential for tractability.

Definition 26.3 (Spectral gap). If the CTMC is irreducible with stationary distribution π, the spectral gap is

$$\gamma \triangleq -\max\{\Re(\lambda) : \lambda \in \mathrm{spec}(Q),\ \lambda \neq 0\}.$$

A positive gap implies exponential convergence to stationarity in many norms.

Proposition 26.8 (Perturbation / parametric sensitivity). *Let $Q(\theta)$ be a differentiable family of generators (parameters appear in rate laws). For any bounded f and fixed t,*

$$\frac{d}{d\theta}\,\mathbb{E}[f(X_t)] = \int_0^t \mathbb{E}\big[(\partial_\theta Q) P_{t-s} f(X_s)\big]\, ds,$$

where $P_t = e^{tQ}$ is the Markov semigroup.

Remark 26.5 (Connection to BioNetGen parameter fitting). This identity is a starting point for adjoint/likelihood-ratio estimators and links the semantic object Q directly to gradient-based calibration workflows common in BNGL pipelines.

26.5 Connections to Petri nets and Markov categories

BNGL models admit faithful translations into several other well-studied mathematical formalisms. These connections can be stated as structure-preserving maps between semantic domains, enabling reuse of results from concurrency theory, stochastic processes, and compositional semantics.

From rules to stochastic Petri nets (SPNs). Given a (finite) generated reaction network $\mathcal{N} = (\mathcal{S}, \mathcal{X}, \nu, k)$, define a Petri net with places $\mathcal{S}$, transitions $\mathcal{X}$, and input/output incidence given by ν^-, ν^+. Mass-action propensities then equip each transition with a hazard function, yielding an SPN semantics. In particular, the standard SSA semantics on $\mathcal{N}$ is exactly the Markov-jump-process semantics of the associated SPN.

Rule-based semantics as a Markov category perspective. The CTMC generator Q induces a Markov semigroup $P_t = e^{tQ}$ acting on bounded functions/observables. Viewed abstractly, this gives a morphism of Markov processes from initial conditions to distributions over trajectories, and supports a compositional view where combining independent subsystems corresponds to a monoidal (tensor) product on state spaces (e.g. disjoint union of site-graphs / independent mixtures).

Operadic / compositional structure. Reaction-network composition and rule-algebra composition can be organized by an operad whose operations "wire" outputs of one subsystem into inputs of another. A future refinement of the present semantics can make this explicit by defining interfaces (distinguished observables/ports) and proving that denotational and operational semantics are functorial with respect to interface composition.

27 Stochastic and Deterministic Dynamics

27.1 Continuous-Time Markov Chain

Definition 27.1 (State Space)**.** For network $\mathcal{N}$ with n species, the state space is $\Omega = \mathbb{N}^n$. State $\vec{x} = (x_1, \ldots, x_n)$ gives copy numbers.

Definition 27.2 (Stoichiometry Matrix)**.** The stoichiometry matrix $\mathbf{N} \in \mathbb{Z}^{n\times m}$ for m reactions has:

$$N_{ij} = \nu_{ij}^{+} - \nu_{ij}^{-}$$

where ν_{ij}^{-} and ν_{ij}^{+} are the reactant and product stoichiometries of species i in reaction j.

Definition 27.3 (Propensity Vector)**.** The propensity vector $\vec{a}(\vec{x}) = (a_1(\vec{x}), \ldots, a_m(\vec{x}))$ where:

$$a_j(\vec{x}) = k_j \cdot \prod_{i=1}^{n} \binom{x_i}{\nu_{ij}^{-}}$$

for mass-action kinetics.

Definition 27.4 (Generator Matrix)**.** The infinitesimal generator $\mathbf{Q}$ of the CTMC has entries:

$$Q_{\vec{x},\vec{x}'} = \begin{cases} \sum_{j:\vec{x}+\mathbf{N}_{\cdot j}=\vec{x}'} a_j(\vec{x}) & \text{if } \vec{x} \neq \vec{x}' \\ -\sum_{\vec{y}\neq\vec{x}} Q_{\vec{x},\vec{y}} & \text{if } \vec{x} = \vec{x}' \end{cases}$$

Definition 27.5 (Chemical Master Equation)**.** The probability distribution $P(\vec{x}, t)$ evolves according to:

$$\frac{dP(\vec{x},t)}{dt} = \sum_{\vec{x}'} Q_{\vec{x}',\vec{x}} P(\vec{x}', t)$$

27.2 Gillespie Algorithm

Theorem 27.1 (SSA Correctness)**.** *The Stochastic Simulation Algorithm (Gillespie algorithm) generates sample paths from the exact distribution of the CTMC.*

Proof. The algorithm samples: (1) waiting time $\tau \sim \mathrm{Exp}(a_0)$ where $a_0 = \sum_j a_j$, which is the correct inter-arrival time for exponential races; (2) reaction j with probability a_j/a_0, which is the correct conditional probability given an event occurs. The memoryless property of exponential distributions ensures the Markov property is preserved. □

27.3 Approximate Simulation Semantics: PLA / Tau-leaping

The Network3 codebase contains a *pathwise leaping* (PLA) framework and multiple pre/post-leap checkers. These algorithms are not exact CTMC semantics; rather they define an *approximate* stochastic process intended to be a weak approximation of the CTMC when step sizes are sufficiently small.

Definition 27.6 (PLA State)**.** A PLA state is a pair $(t, \vec{x})$ where $t \in \mathbb{R}_{\geq 0}$ and $\vec{x} \in \mathbb{N}^n$ is a species-count vector over a generated network.

Definition 27.7 (Tau-leap proposal)**.** Given $(t, \vec{x})$, choose a step size $\tau > 0$ (via a pre-leap controller) and sample independent Poisson reaction counts

$$K_j \sim \text{Poisson}(a_j(\vec{x})\,\tau), \qquad j = 1, \dots, m.$$

Define the proposed update

$$\vec{x}' = \vec{x} + \mathbf{N}\,\vec{K}.$$

Definition 27.8 (Post-leap acceptance)**.** A post-leap checker is a predicate $ok(\vec{x}, \vec{x}', \tau)$. If ok holds and $\vec{x}' \in \mathbb{N}^n$ (no negative populations), accept the step and set $(t, \vec{x}) \leftarrow (t + \tau, \vec{x}')$. Otherwise, reject the step, reduce τ (controller-dependent), and resample.

Definition 27.9 (Pre-leap controller (error/tolerance form))**.** A *pre-leap controller* computes a candidate step size τ from $(t, \vec{x})$ by enforcing a tolerance bound on predicted propensity changes. Abstractly, it selects τ such that for each channel j,

$$|a_j(\vec{x} + \mathbf{N}\,\mathbb{E}[\vec{K}]) - a_j(\vec{x})| \le \epsilon\,\max(1, a_j(\vec{x}))$$

where $\mathbb{E}[K_j] = a_j(\vec{x})\,\tau$ and $\epsilon > 0$ is a user tolerance. Implementations use efficient local approximations (often based on linearization or bounding of reaction-rate derivatives) rather than computing the left-hand side exactly.

Definition 27.10 (Critical reactions and bounded leaping)**.** A reaction channel j is *critical* in state $\vec{x}$ if firing it too many times within one leap risks driving some population negative. A common semantics is to split channels into:

- **Noncritical**: sampled via Poisson counts $K_j \sim \text{Poisson}(a_j(\vec{x})\,\tau)$.
- **Critical**: handled exactly (SSA-style) or with additional bounds (e.g. binomial or capped sampling) so that the update preserves nonnegativity.

This corresponds to the presence of explicit negative-population checkers and post-leap rejection logic in the PLA stack.

Remark 27.1 (Relation to Network3 modules). The pre-leap controller corresponds to components such as `pla/util/preleap_TC.*` and related tau calculators, and the post-leap checker corresponds to `pla/base/postleapChecker.*` together with negative-population checks (`pla/util/negPopChecker.cpp`).

Proposition 27.2 (Consistency (informal))**.** *Under standard regularity assumptions on propensities and for step-size selection rules that enforce $\tau \to 0$ in regimes where propensities vary rapidly, the tau-leaping process converges weakly to the CTMC as the controller tolerances go to zero.*

27.4 ODE Limit

Definition 27.11 (Deterministic Rate Equations)**.** In the thermodynamic limit (large copy numbers, with $c_i = x_i/(N_A V)$), concentrations evolve according to:

$$\frac{d\vec{c}}{dt} = \mathbf{N} \cdot \vec{v}(\vec{c})$$

where $v_j(\vec{c}) = k_j \prod_i c_i^{\nu_{ij}^-}$ is the reaction rate.

Proposition 27.3 (Kurtz Theorem (informal))**.** *As the system size (volume) increases with fixed initial concentrations, the stochastic process converges in probability to the deterministic ODE trajectory.*

28 Rule Algebra Semantics

Following Behr and Krivine [15], we present rule algebra semantics that provides an algebraic perspective on stochastic rewriting systems.

28.1 The Rule Algebra

Definition 28.1 (Rule Algebra)**.** The *rule algebra* $\mathbf{R}(\mathcal{C})$ over a category $\mathcal{C}$ with DPO rewriting is the free $\mathbb{R}$-vector space on the set of DPO rules (spans $L \leftarrow K \rightarrow R$), equipped with a composition operation $\diamond : \mathbf{R} \times \mathbf{R} \rightarrow \mathbf{R}$ derived from sequential rule composition.

The composition $p_1 \diamond p_2$ encodes all ways that rule p_2 can follow rule p_1, weighted by the number of overlaps between R_1 and L_2.

Definition 28.2 (Stochastic Mechanics)**.** A *stochastic mechanics* is a linear functional $\omega : \mathcal{R}(\mathcal{C}) \rightarrow \mathbb{R}$ that encodes the probability of rule application. For mass-action kinetics:

$$\omega(p) = k_p \cdot \frac{|\mathit{Match}(L_p, G)|}{|\mathrm{Aut}(L_p)|}$$

where G is the current state.

Theorem 28.1 (Universal CTMC Semantics)**.** *The rule algebra equipped with stochastic mechanics yields a continuous-time Markov chain with:*

- *State space:* $\mathcal{C}/\cong$ *(isomorphism classes of objects)*
- *Generator:* $Q_{G,H} = \sum_{p:G \Rightarrow H} \omega(p)$ *for* $G \neq H$

Proposition 28.2 (Equivalence to Standard Semantics)**.** *The CTMC induced by the rule algebra with mass-action stochastic mechanics coincides with the standard BNGL stochastic semantics of Section 27.*

Remark 28.1 (Where BNGL symmetry enters the algebra). For a BNGL rule presentation, the structure constants of $\diamond$ depend on counts of overlaps (Definition 4.9) and on match counting modulo automorphisms (Section 6). The factor $|\mathrm{Aut}(L_p)|$ in Definition 28.2 is the algebraic counterpart of BNGL's symmetry-factor correction in propensities.

28.2 Observable Algebra

Definition 28.3 (Observable Algebra)**.** The *observable algebra* $\mathbf{O}$ is the commutative algebra generated by pattern-counting observables O_P for patterns P, with product given by independent counting.

Theorem 28.3 (Moment Closure)**.** *For finite rule sets, the time evolution of expected observables* $\mathbb{E}[O_P(t)]$ *satisfies a system of ODEs that closes at finite order when patterns have bounded size.*

29 Chemical Reaction Network Theory Connections

We establish connections between BNGL rule structure and chemical reaction network theory (CRNT) [16].

29.1 Deficiency Theory

Definition 29.1 (CRN Structure)**.** For a generated reaction network $\mathcal{N}$, define:

- n: number of complexes (distinct left/right-hand sides)
- ℓ: number of linkage classes (connected components of the reaction graph)
- $s = \text{rank}(\mathbf{N})$: dimension of the stoichiometric subspace

Definition 29.2 (Deficiency)**.** The *deficiency* of a chemical reaction network is:

$$\delta = n - \ell - s$$

This non-negative integer measures a form of linear dependency among reaction vectors within linkage classes.

Theorem 29.1 (Deficiency Zero Theorem (Feinberg))**.** *If a reaction network is weakly reversible and has $\delta = 0$, then:*

1. *There exists exactly one positive steady state in each stoichiometric compatibility class*
2. *Each positive steady state is asymptotically stable*
3. *These conclusions are independent of rate constants*

Proposition 29.2 (Reversible Rules and Weak Reversibility)**.** *If every BNGL rule is reversible (bidirectional), then the generated reaction network is weakly reversible.*

Proof. For each reaction $x : C_i \to C_j$ in the network generated by rule r, the reverse rule generates $x' : C_j \to C_i$. Since the reaction graph is symmetric, every linkage class is strongly connected, hence the network is weakly reversible. $\square$

29.2 Conservation Laws

Definition 29.3 (Conservation Law)**.** A *conservation law* is a vector $\vec{w} \in \mathbb{R}^{|\mathcal{S}|}$ satisfying $\vec{w}^T\mathbf{N} = \vec{0}$. The quantity $\vec{w} \cdot \vec{c}(t)$ is conserved along all trajectories.

Proposition 29.3 (Atom Conservation)**.** *The total count of molecule type M (summed over all species containing M) is conserved if and only if no rule creates or destroys molecules of type M (i.e., $\alpha_{\mathsf{mol}}(M)$ appears in neither E_{react} nor E_{prod} of the AR graph).*

29.3 Persistence and Boundedness

Definition 29.4 (Persistence)**.** A reaction network is *persistent* if no species that is initially present can go extinct: $c_i(0) > 0 \Rightarrow c_i(t) > 0$ for all $t > 0$.

Proposition 29.4 (Structural Persistence Conditions)**.** *If the reaction network has a* siphon *(a set of species W such that all reactions producing any species in W also consume some species in W), then the network may fail to be persistent.*

30 Bisimulation and Model Equivalence

We develop bisimulation equivalences for rule-based models, extending the theory of exact lumpability for CTMCs.

30.1 Forward and Backward Bisimulation

Definition 30.1 (Forward Bisimulation)**.** A partition $\mathcal{H} = \{H_1, \dots, H_m\}$ of species $\mathcal{S}$ is a *forward bisimulation* if for all pairs of species $S, S' \in H_i$ (same block) and all blocks H_j:

$$\sum_{T \in H_j} Q_{S,T} = \sum_{T \in H_j} Q_{S',T}$$

That is, the aggregate transition rate out of any state to a given block depends only on which block the source state belongs to.

Definition 30.2 (Backward Bisimulation)**.** A partition $\mathcal{H}$ is a *backward bisimulation* if for all $S, S' \in H_i$ and all blocks H_j:

$$\sum_{T \in H_j} Q_{T,S} = \sum_{T \in H_j} Q_{T,S'}$$

That is, the aggregate transition rate into any state from a given block depends only on which block the target state belongs to.

Theorem 30.1 (Exact Lumpability)**.** *Forward bisimulation is equivalent to* exact lumpability*: the lumped process $\tilde{X}_t = [X_t]_{\mathcal{H}}$ (mapping states to their blocks) is itself a CTMC with generator:*

$$\tilde{Q}_{H_i,H_j} = \sum_{T \in H_j} Q_{S,T} \quad \textit{for any } S \in H_i$$

30.2 A checkable sufficient condition for lumpability

In many rule-based settings one wants a *syntactic* (conservative) criterion implying forward bisimulation without enumerating $\mathcal{S}$.

Proposition 30.2 (Context-insensitive rule criterion (sufficient))**.** *Fix an equivalence relation $\approx$ on patterns (or atoms) inducing a partition of concrete species by the induced observable signatures. Assume that for every rule r and every pair of species $S \approx S'$: (i) the multiset of* projected *products (patterns modulo $\approx$) produced by firing r from S equals that from S', and (ii) the total propensity contribution of r from S depends only on the $\approx$-class of S (e.g. because the instance count and any rate expression are invariant under $\approx$). Then the partition into $\approx$-classes is a forward bisimulation.*

Remark 30.1. Condition (ii) is exactly where BNGL-specific mechanisms can invalidate naive lumpings: automorphism/symmetry factors, DOR local functions, and options such as `MatchOnce` and `TotalRate` change how instance sets are quotiented or weighted. The proposition is therefore best read as a template: provide a concrete $\approx$ and prove invariance for the particular feature set of interest.

30.3 Automated Bisimulation and Model Reduction

We now give an algorithmic procedure for computing forward bisimulation partitions for BNGL models using a partition-refinement approach.

```
**Algorithm:** BNGL-Forward-Bisim
1: Input: species set $\mathcal{S}$, generator $Q$ (or rule-based specification)
2: Initialize partition with classes induced by observable signatures
3: **repeat**
4:     **for** each block $B$ in partition **do**
5:         Compute for each $S \in B$ vector $v_S = (\sum_{T \in C} Q_{S,T})_{C \in \mathcal{P}}$
6:         Split $B$ into subblocks where $v_S$ are equal
7:     **end for**
8: **until** no split occurs
9: Output: partition $\mathcal{H}$ (forward bisimulation)
```

Complexity Each refinement step inspects rates to blocks; using hashing, total complexity is $O(|\mathcal{S}| \cdot m \cdot \log |\mathcal{S}|)$ where m is number of blocks. In rule-based implementations, computing v_S can be done symbolically using pattern-based counters to avoid full network enumeration.

Model reduction workflow (1) Compute partition via BNGL-Forward-Bisim; (2) collapse species and recompute reduced generator; (3) verify preservation of target observables via statistical model checking.

30.4 Rule-Level Bisimulation

Definition 30.3 (Rule Bisimulation)**.** Two rule sets $\mathcal{R}_1$ and $\mathcal{R}_2$ over the same molecule types are *rule bisimilar* if there exists a relation $\mathcal{R}$ on patterns such that:

1. For each rule $r_1 \in \mathcal{R}_1$, there exists $r_2 \in \mathcal{R}_2$ with $(L_1, L_2) \in \mathcal{R}$ and $(R_1, R_2) \in \mathcal{R}$
2. The relation is preserved under all matches: if $(P, Q) \in \mathcal{R}$ and $P \triangleright S$, then $Q \triangleright S'$ for some $S' \cong S$
3. Rate laws satisfy $k_{r_1} \cdot \mathit{matches}(r_1, \sigma) = k_{r_2} \cdot \mathit{matches}(r_2, \sigma)$ for all states σ

Proposition 30.3 (Rule Bisimulation Implies CRN Bisimulation)**.** *If $\mathcal{R}_1 \sim \mathcal{R}_2$ at the rule level, then the generated networks $\mathcal{N}_1$ and $\mathcal{N}_2$ are forward bisimilar as CTMCs.*

30.5 Categorical Equivalence

Definition 30.4 (Category of Rule-Based Models)**.** Define the category **RBM** with:

- Objects: BNGL models $(\mathcal{M}, \mathcal{R}, \Sigma_0)$
- Morphisms: pairs $(\phi_{\mathcal{M}}, \phi_{\mathcal{R}})$ of type-preserving maps that commute with rule application

Definition 30.5 (Semantic vs. Observational Equivalence)**.** Models $\mathcal{B}_1$ and $\mathcal{B}_2$ are:

- *Semantically equivalent* if $[\![\mathcal{B}_1]\!] \cong [\![\mathcal{B}_2]\!]$ as CTMCs
- *Observationally equivalent* if $\mathbb{E}[O(X_t^1)] = \mathbb{E}[O(X_t^2)]$ for all observables O

Proposition 30.4 (Equivalence Hierarchy)**.** *For BNGL models: isomorphism $\Rightarrow$ rule bisimulation $\Rightarrow$ semantic equivalence $\Rightarrow$ observational equivalence.*

31 Causal Semantics

We formalize how atom-rule graphs encode causal structure in the sense of Pearl's interventional calculus [17].

31.1 Pearl's Causal Hierarchy

Definition 31.1 (Causal Hierarchy)**.** Pearl's causal hierarchy distinguishes three levels of causal reasoning:

1. **Association** (Level 1): $P(Y|X = x)$ – observational conditioning
2. **Intervention** (Level 2): $P(Y|\mathrm{do}(X = x))$ – effect of external manipulation
3. **Counterfactual** (Level 3): $P(Y_x|X = x', Y = y')$ – what would have happened under different circumstances

31.2 Causal DAG from AR Graph

Definition 31.2 (Induced Causal DAG)**.** The *causal DAG* $\mathcal{G}_{causal}$ induced by atom-rule graph $\mathrm{AR}(\mathcal{B})$ has:

- Nodes: atom types $\mathcal{A}$
- Directed edge $\alpha_1 \to \alpha_2$ if there exists rule r with $\alpha_1 \in atoms(L_r)$ and $\alpha_2 \in atoms(R_r) \setminus atoms(L_r)$

This captures direct causal influence: α_1's presence enables a rule that produces α_2.

31.3 Interventions

Definition 31.3 (Intervention Operator)**.** The *intervention* $\mathrm{do}(\alpha = v)$ on atom α with value $v \in \{0, 1\}$ produces a modified model $\mathcal{B}|_{\mathrm{do}(\alpha=v)}$:

1. Remove all rules r where $\alpha \in E_{\mathsf{prod}}(r)$ if $v = 0$, or $\alpha \in E_{\mathsf{react}}(r)$ if $v = 1$
2. Add constraints ensuring α-atoms have the specified value
3. Compute modified dynamics on the constrained state space

Lemma 31.1 (Intervention as causal graph surgery (informal))**.** *Under the induced causal DAG construction (Definition 31.2), the transformation $\mathcal{B} \mapsto \mathcal{B}|_{\mathrm{do}(\alpha=v)}$ removes exactly the incoming causal mechanisms to α (and fixes its value), in the sense that no remaining rule can change α away from the enforced value.*

Definition 31.4 (Knockout Semantics)**.** A *gene knockout* of molecule type M corresponds to the intervention $\mathrm{do}(\alpha_{\mathsf{mol}}(M) = 0)$:

1. Remove all rules producing molecules of type M
2. Remove initial molecules of type M from Σ_0
3. The resulting dynamics represent the knockout phenotype

31.4 Causal Completeness

Theorem 31.2 (Causal Completeness)**.** *Rule-based models are causally complete for levels 1–2 of Pearl's hierarchy:*

1. ***Level 1**: Associational queries $P(Y|X)$ are answered by standard simulation/analysis*
2. ***Level 2**: Interventional queries $P(Y|\mathrm{do}(X))$ are answered by modifying the rule set per Definition 31.3 and simulating*

Level 3 (counterfactual) queries require additional structural assumptions about exogenous variables.

Proof. Level 1 follows from the stochastic semantics. For Level 2, the intervention $\mathrm{do}(X = x)$ corresponds to surgery on the model that removes incoming causal arrows to X and fixes its value. The AR graph encodes exactly these causal dependencies, so the modified model correctly represents the post-intervention distribution. □

32 Temporal Logic Specifications

We present a temporal logic for specifying properties of BNGL models, supporting statistical model checking.

32.1 Probabilistic Bounded Linear Temporal Logic

Definition 32.1 (PBLTL Syntax)**.** The syntax of Probabilistic Bounded Linear Temporal Logic (PBLTL) is:

$$\begin{aligned}
&\phi ::= \mathit{true} \mid O \bowtie c \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \bigcirc^{\leq t}\phi \mid \phi_1 \mathcal{U}^{\leq t} \phi_2 \\
&\Phi ::= \mathbb{P}_{\bowtie p}[\phi]
\end{aligned}$$

where O is an observable, $\bowtie \in \{<, \leq, =, \geq, >\}$, $c \in \mathbb{R}$, $t \in \mathbb{R}_{\geq 0}$ is a time bound, and $p \in [0, 1]$ is a probability threshold.

Definition 32.2 (PBLTL Semantics)**.** For a trajectory $\omega = (X_t)_{t\geq 0}$ and time τ, define the next jump time

$$\tau_{next} \triangleq \inf\{t > \tau : X_t \neq X_\tau\}$$

(with the convention $\tau_{next} = +\infty$ if no such time exists). Then:

$$\begin{aligned}
(\omega, \tau) &\models O \bowtie c && \Leftrightarrow O(X_\tau) \bowtie c \\
(\omega, \tau) &\models \bigcirc^{\leq t}\phi && \Leftrightarrow \tau_{next} \leq \tau + t \ \wedge \ (\omega, \tau_{next}) \models \phi \\
(\omega, \tau) &\models \phi_1 \mathcal{U}^{\leq t} \phi_2 && \Leftrightarrow \exists \tau' \in [\tau, \tau + t] : (\omega, \tau') \models \phi_2 \\
& && \quad \wedge \forall \tau'' \in [\tau, \tau') : (\omega, \tau'') \models \phi_1
\end{aligned}$$

Definition 32.3 (Derived Operators)**.** Standard derived operators:

$$\begin{aligned}
\Diamond^{\leq t}\phi &\equiv \mathit{true}\mathcal{U}^{\leq t}\phi && \text{(eventually within time } t) \\
\Box^{\leq t}\phi &\equiv \neg\Diamond^{\leq t}\neg\phi && \text{(always within time } t)
\end{aligned}$$

32.2 Statistical Model Checking

Definition 32.4 (Statistical Model Checking Problem)**.** Given model $\mathcal{B}$ and specification $\Phi = \mathbb{P}_{\bowtie p}[\phi]$, determine whether $\mathcal{B} \models \Phi$ using simulation-based hypothesis testing.

Definition 32.5 (Sequential Probability Ratio Test)**.** The SPRT with error bounds α, β and indifference region $[p - \delta, p + \delta]$ accepts $H_0 : \theta \leq p - \delta$ or $H_1 : \theta \geq p + \delta$ based on the likelihood ratio:

$$\Lambda_n = \prod_{i=1}^{n} \frac{P(X_i|H_1)}{P(X_i|H_0)} = \frac{(p+\delta)^{n_+}(1-p-\delta)^{n_-}}{(p-\delta)^{n_+}(1-p+\delta)^{n_-}}$$

where $n_+ = |\{i : \omega_i \models \phi\}|$ and $n_- = n - n_+$.

Theorem 32.1 (SPRT Correctness)**.** *The SPRT terminates with probability 1 and satisfies:*

$$\begin{aligned}
P(\mathit{accept}\ H_1|H_0) \leq \alpha \\
P(\mathit{accept}\ H_0|H_1) \leq \beta
\end{aligned}$$

32.3 Model-Checking Pipeline (BNGL →PRISM)

A practical workflow for verifying PBLTL properties is:

1. Use network generation to create an explicit CRN (or partial expansion)
2. Translate species to PRISM places and reactions to transitions with rates
3. Map observables to PRISM reward structures or labelled propositions
4. Run PRISM's probabilistic model checker for small networks or SMC for larger ones

Translation rules Species S -> place p_S; reaction x with stoichiometry ν^-, ν^+ maps to transition with guards consuming/producing tokens accordingly; observable $O(P)$ maps to state predicate over places.

Example Translate receptor activation property into PRISM PCTL and verify reachability probabilities on the generated network. For large networks, use statistical model checking with PRISM's SMC module.

32.4 Example Specifications

Example 32.1 (Receptor Activation). "Receptor activation exceeds 100 within 10 minutes with probability > 0.9":

$$\mathbb{P}_{>0.9}[\Diamond^{\leq 600}(\mathit{Active\_Receptor} > 100)]$$

Example 32.2 (Steady State). "System reaches steady state (changes $< 1\%$) by time 1000":

$$\mathbb{P}_{>0.95}[\Diamond^{\leq 1000}\Box^{\leq 100}(|O' - O|/O < 0.01)]$$

Example 32.3 (Stimulus Response). "If ligand is added, activation occurs within 5 minutes":

$$\mathbb{P}_{>0.99}[\Box^{\leq T}(\mathit{Ligand} > 0 \Rightarrow \Diamond^{\leq 300}\mathit{Active} > 50)]$$

33 Semantic Equivalence Theorems

We now prove our main technical results establishing equivalence between the three simulation approaches.

33.1 State Correspondence

Definition 33.1 (State Abstraction)**.** Define abstraction functions:

- $\alpha_{NF} : \mathit{AgentState} \to \mathbb{N}^{|\mathcal{S}|}$: $\alpha_{NF}(\sigma) = (|\{[a] : [a] \cong S\}|)_{S\in\mathcal{S}}$
- $\alpha_H : \mathit{HybridState} \to \mathbb{N}^{|\mathcal{S}|}$: $\alpha_H(\sigma_H) = (n_S + |\mathcal{G}_S|)_{S\in\mathcal{S}}$ where we abuse notation for population vs. particle counts

Lemma 33.1 (Abstraction Commutativity)**.** *For equivalent states $\sigma_H \approx \sigma$:*

$$\alpha_H(\sigma_H) = \alpha_{NF}(\sigma)$$

33.2 Propensity Correspondence

Lemma 33.2 (Propensity Equivalence)**.** *For network reaction x corresponding to rule r:*

1. *Network propensity (in terms of species counts) equals network-free propensity (in terms of mapping counts) after accounting for symmetry*
2. *Hybrid propensity equals network-free propensity*

Formally, if $\sigma_H \approx \sigma$ and $\alpha(\sigma) = \vec{n}$:

$$a_x^{net}(\vec{n}) = a_r^{NF}(\sigma) = a_x^H(\sigma_H)$$

Proof. For network propensity with mass-action kinetics:

$$a_x^{net} = k_x \prod_i \binom{n_{S_i}}{\nu_i}$$

For network-free, the number of MappingSets is:

$$\prod_i |\mathcal{L}_{r,i}| = \prod_i \left(n_{S_i} \cdot \frac{|Match(P_i, S_i)|}{|\mathrm{Aut}([\![P_i]\!])|} \right)$$

When rules are generated from the network with correct symmetry factor handling:

$$k_r = k_x \cdot \prod_i |\mathrm{Aut}([\![P_i]\!])| / |Match(P_i, S_i)|$$

Substituting and simplifying yields equality.

For hybrid propensity, Lemma 10.4 establishes $a_x^H = a_r^{NF}$. □

33.3 Transition Correspondence

Lemma 33.3 (Transition Equivalence)**.** *For corresponding states and reactions:*

1. *Network transition $\vec{n} \to \vec{n}'$ via reaction x corresponds to the set of NF transitions $\sigma \to \sigma'$ via rule instances yielding $\alpha(\sigma') = \vec{n}'$*

2. *The aggregate probability over NF transitions equals the network transition probability*

Proof. Each network reaction x corresponds to one or more rule instances. The instances producing the same species transformation have probabilities that sum to a_x/a_0. After the transition, species counts change by the stoichiometry vector $\mathbf{N}_{\cdot j}$, matching the network dynamics. □

33.4 Main Equivalence Theorem

Theorem 33.4 (Three-Way Semantic Equivalence)**.** *Let $\mathcal{B}$ be a well-formed BioNetGen model where network generation terminates, producing network $\mathcal{N} = generate(\mathcal{B})$. Consider initial state σ_0 with species counts $\vec{n}_0 = \alpha(\sigma_0)$.*

The following three stochastic processes are equivalent in distribution:

1. ***Network-based SSA** on $\mathcal{N}$ starting from $\vec{n}_0$*

2. ***Network-free simulation** on $\mathcal{B}$ starting from σ_0*

3. ***Hybrid simulation** on $\mathcal{B}$ starting from $\sigma_H^0 \sim \sigma_0$*

More precisely, let X_t^{net}, X_t^{NF}, X_t^H denote the respective processes (with agent states abstracted to species counts). Then:

$$X_t^{net} \overset{d}{=} \alpha(X_t^{NF}) \overset{d}{=} \alpha_H(X_t^H)$$

for all $t \geq 0$.

Proof. We establish that all three processes define the same CTMC by showing they have:

1. The same state space: $\mathbb{N}^{|\mathcal{S}|}$ (after abstraction)

2. The same transition rates: by Lemma 33.2

3. The same transition targets: by Lemma 33.3

State correspondence: Network states are species count vectors. Agent states abstract to count vectors via α_{NF}. Hybrid states abstract via α_H. By Lemma 33.1, equivalent inputs map to equal abstractions (i.e., $\sigma_H \approx \sigma \Rightarrow \alpha_H(\sigma_H) = \alpha_{NF}(\sigma)$).

Propensity correspondence: By Lemma 33.2, for any network reaction x, the propensity computed network-based, network-free, or hybrid are equal when evaluated on corresponding states. For the hybrid case, this relies on Proposition 10.2: the PNE algorithm partitions $\mathcal{R}$ into $\mathcal{R}_{PNE} = \mathcal{R}_{\text{pop}} \uplus \mathcal{R}_{\text{part}} \uplus \mathcal{R}_{mix}$ such that total propensity is preserved.

Transition correspondence: By Lemma 33.3, transitions in the three formulations produce corresponding state changes. The correctness of rule application relies on the unique decomposition of patterns into atoms (Proposition 5.1) and the commutativity of primitive graph operations on disjoint elements (Lemma 7.5).

Since all three define CTMCs with the same generator matrix (up to abstraction), they are distributionally equivalent. □

Corollary 33.5 (Observable Equivalence)**.** *For any observable O and time t:*

$$\mathbb{E}[[\![O]\!](X_t^{net})] = \mathbb{E}[[\![O]\!](X_t^{NF})] = \mathbb{E}[[\![O]\!](X_t^{H})]$$

Corollary 33.6 (Path-measure equivalence)**.** *Under the hypotheses of Theorem 33.4, the induced measures on trajectory space $D([0,T],\Omega)$ coincide after applying the appropriate abstractions: the network path law equals the pushforward of the network-free and hybrid path laws by α and α_H, respectively. In particular, any measurable path functional (e.g. stopping times, time integrals, temporal-logic satisfaction) has the same distribution under the three exact semantics.*

33.5 Complexity Tradeoffs

Proposition 33.7 (Space-Time Tradeoff)**.** *The three simulation approaches have different complexity characteristics:*

Approach	***Space***	***Time per step***
Network-based	$O(\lvert\mathcal{S}\rvert + \lvert\mathcal{X}\rvert)$	$O(\lvert\mathcal{X}\rvert)$
Network-free	$O(\lvert\mathcal{G}\rvert)$	$O(\lvert\mathcal{R}\rvert \cdot k_{\max}^2)$
Hybrid	$O(\lvert\mathcal{S}_{\text{pop}}\rvert + \lvert\mathcal{G}_{\text{part}}\rvert)$	$O(\lvert\mathcal{X}_{\text{pop}}\rvert + \lvert\mathcal{R}_{\text{part}}\rvert \cdot k^2)$

where $k_{\max}$ is the maximum pattern size.

The hybrid approach can achieve the best of both: population-level efficiency for high-copy species and particle-level representation for rare species requiring exact tracking.

34 Worked Examples

34.1 Example 1: Ligand-Receptor Binding

Consider the canonical binding model:

```
begin model
begin molecule types
    L(r)       # Ligand with receptor-binding site
    R(l,Y~U~P) # Receptor with ligand site and phosphorylation site
end molecule types

begin seed species
    L(r) 1000
    R(l,Y~U) 300
end seed species
```

```

begin reaction rules
    # Binding/unbinding
    L(r) + R(l) <-> L(r!1).R(l!1) kf, kr
    # Phosphorylation (requires bound ligand)
    L(r!1).R(l!1,Y~U) -> L(r!1).R(l!1,Y~P) kp
end reaction rules
end model
```

Atom Analysis:

$$\begin{aligned}\mathcal{A} = \{&\alpha_{\mathsf{mol}}(\mathsf{L}), \alpha_{\mathsf{mol}}(\mathsf{R}),\\ &\alpha_{\mathsf{free}}(\mathsf{L},\mathsf{r}), \alpha_{\mathsf{free}}(\mathsf{R},\mathsf{l}),\\ &\alpha_{\mathsf{state}}(\mathsf{R},\mathsf{Y},\mathsf{U}), \alpha_{\mathsf{state}}(\mathsf{R},\mathsf{Y},\mathsf{P}),\\ &\alpha_{\mathsf{bond}}(\mathsf{L},\mathsf{r},\mathsf{R},\mathsf{l})\}\end{aligned}$$

AR Graph for Rule 1 (binding):

- Reactant atoms: $\alpha_{\mathsf{free}}(\mathsf{L},\mathsf{r})$, $\alpha_{\mathsf{free}}(\mathsf{R},\mathsf{l})$
- Product atoms: $\alpha_{\mathsf{bond}}(\mathsf{L},\mathsf{r},\mathsf{R},\mathsf{l})$
- Context atoms: $\alpha_{\mathsf{mol}}(\mathsf{L})$, $\alpha_{\mathsf{mol}}(\mathsf{R})$

AR Graph for Rule 2 (phosphorylation):

- Reactant atoms: $\alpha_{\mathsf{state}}(\mathsf{R},\mathsf{Y},\mathsf{U})$
- Product atoms: $\alpha_{\mathsf{state}}(\mathsf{R},\mathsf{Y},\mathsf{P})$
- Context atoms: $\alpha_{\mathsf{bond}}(\mathsf{L},\mathsf{r},\mathsf{R},\mathsf{l})$

Causal Path: Rule 1 produces α_{bond} which enables Rule 2. This is visible as path $r_1 \to \alpha_{\mathsf{bond}} \to r_2$ in the AR graph.

34.2 Example 2: Symmetric Dimerization

```
begin molecule types
    A(d)  # Molecule with dimerization site
end molecule types

begin reaction rules
    A(d) + A(d) <-> A(d!1).A(d!1) kf, kr
end reaction rules
```

Symmetry Analysis:
The product pattern `A(d!1).A(d!1)` has $sym = 2$ (the two A molecules are interchangeable). For the forward reaction with n monomers:

- Number of unordered pairs: $\binom{n}{2} = \frac{n(n-1)}{2}$
- Propensity: $a_f = k_f \cdot \frac{n(n-1)}{2}$

For the reverse reaction with m dimers:

- Each dimer can dissociate: m ways
- But symmetry factor in product is 2, so: $a_r = k_r \cdot m$

Operation Sequence:

1. $\textsc{AddBond}(A_1.\mathsf{d}, A_2.\mathsf{d})$

This single operation transforms two monomers into one dimer.

34.3 Example 3: Hybrid Simulation Scenario

Consider a signaling cascade with:

- Abundant scaffold protein (10^4 copies)
- Rare activated receptor (10 copies)
- Cascade of modifications

Regime Assignment:

- Scaffold (unbound): pop (high copy, no need for individual tracking)
- Receptor complex: part (low copy, stochastic effects important)
- Scaffold-receptor complex: part (inherits from receptor)

Mixed Reaction: When scaffold binds to receptor:

1. Population count of scaffold decremented
2. One scaffold instantiated as agent
3. Bond created to receptor particle
4. Complex tracked at particle level

This demonstrates how hybrid simulation efficiently handles systems spanning multiple scales.

35 Inference, Identifiability, Reduction, and Modern ML Workflows

This section elevates a subset of Appendix material to the main text because it is central to how BNGL models are used in contemporary scientific workflows: parameter inference, identifiability diagnostics, symmetry-aware model reduction, and ML/AI-assisted modeling.

35.1 Local vs. global identifiability: FIM vs. profile likelihood

The Fisher information matrix (FIM) is intrinsically a *local* object: it captures the curvature of the log-likelihood at a point $\theta_\star$.

Local (FIM-based) identifiability. If $\mathcal{I}(\theta_\star)$ is full rank, then (under standard regularity conditions) the likelihood is locally quadratic near $\theta_\star$ and parameters are locally identifiable in the sense of having finite local confidence ellipsoids. If $\mathcal{I}(\theta_\star)$ is singular or extremely ill-conditioned, then there are directions along which the likelihood is flat (or nearly flat) near $\theta_\star$ (local non-identifiability / sloppiness).

Global (profile-likelihood) identifiability. Profile likelihoods (Section C.9) probe *global* non-identifiability: whether there exist distant parameter vectors producing the same (or nearly the same) likelihood. In particular, it is possible for $\mathcal{I}(\theta_\star)$ to be nonsingular while the profile likelihood remains flat far away (e.g. due to discrete symmetries, parameter re-scalings, or saturation effects), and conversely for $\mathcal{I}(\theta_\star)$ to be nearly singular even though the profile likelihood is bounded globally (e.g. because the model manifold is narrow but curved).

Rule-based interpretation. In BNGL, global non-identifiabilities often arise from semantics-preserving transformations (rule refinements, lumpings, and symmetry-factor conventions), whereas local sloppiness often arises because many rule contexts are rarely visited under the design, making the associated parameters weakly informed.

35.2 Model reductions as quotients (and their identifiability impact)

Many BNGL reductions are semantic quotients: they replace the original CTMC X_t on Ω by a process on a coarser state space $\tilde{\Omega}$.

Exact reductions (lumpability). If a surjection $\pi : \Omega \to \tilde{\Omega}$ is such that $\pi(X_t)$ is itself a CTMC (exact lumpability; Section 30), then parameters that only change transition structure *within fibers* of π become structurally unidentifiable under any observation scheme that factors through π.

Approximate reductions (sloppiness-aware). Even when exact lumpability fails, a reduced model can be chosen so that for a target observable class $\mathcal{F}$, $\mathbb{E}[f(X_t)] \approx \mathbb{E}[f(\tilde{X}_t)]$. Such reductions intentionally discard sloppy directions (small-eigenvalue FIM modes), yielding a lower-dimensional "effective parameterization" (Section C.15).

35.3 Symmetry and parameter fibers

The most robust source of global non-identifiability is symmetry: group actions on parameters (or on rule presentations) that leave the generator unchanged.

Discrete vs. continuous symmetries. Discrete symmetries (e.g. swapping indistinguishable rule labels) typically produce multiple isolated maxima in parameter space, whereas continuous symmetries ("gauge" directions) produce flat ridges in the likelihood and manifest as rank deficiency in $\mathcal{I}(\theta_\star)$.

Where symmetries enter in BNGL. BNGL has several built-in symmetry mechanisms: automorphisms of patterns (Section 6), quotienting implicit in observables (distinct matches modulo $\mathrm{Aut}(P)$), and semantics-preserving rewrites (rule split/merge/refinement). These can create global identifiability fibers even when local FIM diagnostics look well-conditioned at a nominal point.

35.4 Atomizer: a semantics-guided translation from SBML to BNGL

Tools such as BioNetGen's Atomizer (SBML-to-BNGL conversion) can be viewed as a semantics-guided translation from a reaction-network presentation (SBML) into a rule-based presentation (BNGL). Mathematically, the motivation is to move from a "flat" reaction-network view (species as atomic) to a structured rewriting view (species as site-graphs), while preserving the induced stochastic dynamics on observables of interest.

Translation as a map between presentations. Let $\mathcal{M}_{\mathrm{SBML}}$ denote an SBML model and let $\mathcal{M}_{\mathrm{BNGL}} = \mathsf{Atomize}(\mathcal{M}_{\mathrm{SBML}})$ denote its BNGL translation. Each side induces (under a fixed design/observation scheme) a statistical model $\{\mathbb{P}_\theta^{\mathrm{SBML}}\}_{\theta\in\Theta}$ and $\{\mathbb{P}_\theta^{\mathrm{BNGL}}\}_{\theta\in\Theta}$.

Semantic correctness target (commuting diagram). A natural correctness goal is the existence of a measurable abstraction $\pi : \Omega_{\text{BNGL}} \to \Omega_{\text{SBML}}$ (e.g. mapping structured site-graph mixtures to SBML species-count vectors) such that the induced data laws commute with pushforward:

$$\pi_{\#}\, \mathbb{P}_{\theta}^{\text{BNGL}} = \mathbb{P}_{\theta}^{\text{SBML}} \qquad \text{for all } \theta \in \Theta,$$

where $\pi_{\#}$ denotes pushforward of measures. At the generator level (finite-state setting), this corresponds to an intertwining relation between generators, i.e. the BNGL generator projects to the SBML generator under π.

Proposition 35.1 (Generator homomorphisms induce equal data laws)**.** *Let Q be a generator on Ω and let $\tilde{Q}$ be a generator on $\tilde{\Omega}$. Let $\pi : \Omega \to \tilde{\Omega}$ be a surjection. Assume there exists an intertwining relation (on bounded test functions f)*

$$(Q(f \circ \pi))(x) = (\tilde{Q}f)(\pi(x)) \quad \textit{for all } x \in \Omega.$$

Then the pushforward process $\pi(X_t)$ is a CTMC with generator $\tilde{Q}$. Consequently, for any measurable observation map Φ that factors through π (i.e. $\Phi = \tilde{\Phi} \circ \pi$), the induced data laws coincide.

Why BNGL form is analytically useful. Even when the goal is only equality "up to abstraction," the BNGL representation exposes structure not present in SBML: graph-based matching, automorphism/symmetry factors, rule algebras, and AR-graph causal structure. This enables semantics-aligned reductions (bisimulation/lumping), causal reachability screens, and hybrid simulation strategies.

35.5 Biological advances and their semantic interface

Recent biological measurement and modeling advances primarily change (i) the observation operator Φ mapping latent BNGL trajectories to data and (ii) the family of admissible interventions $\{Q_{\theta,u}\}_{u\in\mathcal{U}}$. We summarize a few developments in a deliberately semantic (commuting-diagram) style.

Observation as a pushforward of path space. Let $\mathbb{Q}_{\theta,u}$ denote the BNGL-induced law on trajectories $\omega \in D([0,T],\Omega)$ under parameter θ and intervention policy u. A measurement technology specifies a measurable map (often stochastic) $\Phi : D([0,T],\Omega) \rightsquigarrow \mathcal{D}$ into a data space $\mathcal{D}$ (time series, snapshots, images), yielding the data law $\mathbb{P}_{\theta,u} = \Phi_{\#}\mathbb{Q}_{\theta,u}$. Thus new experimental modalities are naturally compared by how they refine (or coarsen) the σ-algebra of observations, and therefore how much information they preserve about (θ, u).

Single-cell and distributional observables. Single-cell assays (flow cytometry, scRNA-seq/proteomics) shift the target from mean trajectories to full *population distributions* over molecular states. Semantically, this replaces a low-dimensional observable vector $h(X_t)$ by a higher-dimensional statistic (e.g. empirical measures $\hat{\mu}_t$ over cell states) and changes the noise model (Poisson counts, over-dispersion, dropout). This typically increases identifiability but introduces a stronger obligation to model hidden variables (cell-cycle stage, extrinsic noise) as part of the state or as nuisance parameters.

Time-resolved live-cell imaging and event timing. Live reporters (FRET, fluorescent tags) provide dense sampling and often enable extracting event-time summaries (first-passage times, dwell times, burst statistics). These correspond to stopping-time functionals $\Phi(\omega) = \tau_A(\omega)$ or path functionals $\Phi(\omega) = \int_0^T g(X_t)\, dt$, which are natural objects for CTMC semantics (Section 19.3) and for temporal-logic specifications (Section 32).

Spatial and compartment-resolved measurements. Spatial transcriptomics and imaging-based assays motivate extending the state to include explicit compartments or spatial coordinates. At the semantic level, this is an enrichment of the attribute signature and the generator (cf. Section 12.1): observation becomes sensitive to where molecules are, not only what complexes exist. This tends to break symmetries that are present in well-mixed models and can make previously lumpable reductions invalid.

Interventions as controlled generators. Modern perturbations (CRISPR knock-outs/knockdowns, degrons, inhibitors, induced dimerization, optogenetics) enlarge the admissible control space $u \in \mathcal{U}$. Semantically, each intervention specifies a family of modified generators $Q_{\theta,u}$ by: (i) deleting or clamping rules (hard interventions / do-operator, Section 31), (ii) scaling or time-modulating propensities (dosing, light pulses), or (iii) modifying initial conditions. This viewpoint aligns naturally with optimal design and closed-loop control (Section C.13).

Design consequence: interventions as symmetry breakers. Many practical non-identifiabilities in rule-based models arise because distinct parameterizations induce the same effective dynamics on the observed subspace (Section C.6). Interventions that selectively disable pathways or force the system into rare rule contexts act as *symmetry breakers*: they change the reachable region of state space and therefore the set of joint mappings that contribute to propensities. This is precisely the mechanism by which perturbation panels (multi-condition experiments) can turn sloppy parameters into identifiable ones.

Modeling obligation: measurement semantics must be explicit. Because Φ is now often complex (imaging pipelines, sequencing counts, normalization), semantic clarity requires stating the observation model and its noise kernel as part of the experiment description. Otherwise, two studies may be incomparable even if they use the same underlying BNGL generator Q_θ, because they push forward to different data laws.

35.6 Connections to modern computing: ML/AI/LLM tooling

Modern computational tooling changes how identifiability and reduction analyses are performed in practice, without changing the underlying semantics. Mathematically, these tools can be viewed as constructing approximations to, or numerical representations of, the BNGL-induced statistical model $\{\mathbb{P}_\theta\}_{\theta\in\Theta}$ and its derived functionals.

Automatic differentiation and adjoints (local geometry). When a differentiable surrogate (ODE limit, moment closure, or emulator) $\theta \mapsto y(\cdot;\theta)$ is available, reverse-mode AD computes gradients needed for local metrics such as the FIM (Definition C.4) and for gradient-based estimation. From a semantic viewpoint, this equips Θ with a computable local information geometry induced by the experiment.

Simulation-based inference (likelihood-free semantics). For the exact CTMC semantics, likelihood-free methods treat the BNGL simulator as a generative mechanism for samples from $\mathbb{P}_\theta$. Abstractly, SBI replaces explicit likelihood evaluation by learning an approximation to either: (i) the posterior $\theta \mapsto \pi(\theta \mid D)$, or (ii) likelihood ratios $\frac{d\mathbb{P}_\theta}{d\mathbb{P}_{\theta'}}$ on a chosen summary σ-algebra. Global non-identifiability then appears as posterior ridges/multimodality, which are numerical shadows of non-injectivity of $\theta \mapsto \mathbb{P}_\theta$.

Neural surrogates and controlled approximation. Learned emulators can reduce computational cost in design loops by approximating $\theta \mapsto y(t;\theta)$ or selected expectation functionals $\theta \mapsto \mathbb{E}_\theta[f(X_t)]$. To keep the treatment mathematical, a surrogate should be understood as an operator $\tilde{\Phi}$ approximating a true semantic functional Φ, with an explicit error criterion such as $\sup_{\theta\in\Theta_0} \sup_{t\in[0,T]} |\tilde{\Phi}(\theta,t) - \Phi(\theta,t)| \le \varepsilon$.

GPU acceleration as operator-level speedup. Many computations induced by BNGL semantics can be written as batched linear algebra or embarrassingly parallel Monte Carlo, making GPU acceleration a mathematically natural fit. Examples include: (i) evaluating many propensities $a_r(\sigma)$ across many particles/replicates; (ii) simulating many independent trajectories for Monte Carlo estimators of $\mathbb{E}_\theta[f(X)]$; (iii) computing or approximating sensitivities/FIM surrogates from batched trajectories; and (iv) accelerating surrogate training and SBI by parallel sampling. In all cases, the semantic object (generator Q_θ, or its pushforward $\mathbb{P}_\theta$) is unchanged; only the numerical cost of approximating its derived functionals is reduced.

Data validation as a semantic compatibility test. When calibrating to external datasets (including those obtained online), the core mathematical question is whether the dataset can be realized as a pushforward of some $\mathbb{P}_\theta$ under the declared sampling/noise model. Thus, "validation" is the problem of checking compatibility of empirical measures with the model family (e.g. by hypothesis tests on summary statistics, posterior predictive checks, or divergence bounds).

Large language models (LLMs) as structured priors and program synthesis. LLMs are useful for proposing candidate rule sets and observable families. To keep this mathematical, one can interpret an LLM as inducing a proposal distribution over discrete model structures (e.g. grammars of BNGL rules), which is then combined with data through a posterior over both structure and parameters. However, any such proposals still require semantic checks (e.g. equivalence or lumpability conditions) to avoid introducing unobservable parameters or global non-identifiabilities.

Diffusion and score-based models as approximate samplers. Diffusion models can be understood as constructing a Markov process Z_t whose time-reversal approximately samples from a target distribution. In this context, one may treat them as (i) amortized samplers for posteriors $\pi(\theta \mid D)$, or (ii) proposal mechanisms for latent-state smoothing in partially observed CTMCs. The semantic constraint is that such samplers should target the correct measure (e.g. a posterior induced by $\mathbb{P}_\theta$), and approximation should be quantified by a divergence bound between the intended and learned measures.

Robotics and closed-loop experimentation as stochastic control. Automated experimental platforms (robotics) naturally implement feedback policies that choose interventions and measurement schedules based on observed data. Mathematically, this is a controlled Markov process viewpoint: the BNGL model induces controlled generators $Q_{\theta,u}$ parameterized by an action/control u (e.g. dosing, knockouts, stimulus timing), and the design objective becomes an optimal control / Bayesian experimental design problem over policies.

Quantum computing as accelerated linear-algebra / sampling primitives. Most BNGL computations reduce to manipulating generators, semigroups, and Monte-Carlo estimators. If quantum subroutines are used at all, their mathematically relevant role is as potential accelerators for specific primitives (e.g. solving linear systems or sampling from structured distributions) that appear inside inference or model checking pipelines. The semantic object remains classical (the CTMC generator and its induced path measure); the question is purely computational complexity of approximating functionals of $\mathbb{P}_\theta$.

AGI and "singularity" scenarios as limits of automation (not semantics). Discussions of AGI or "singularity" do not change the mathematics of BNGL semantics, but they motivate treating the end-to-end pipeline as a formal decision procedure: propose models, check well-formedness, test equivalences, fit parameters, and select designs. In that framing, the key requirement is that each automated step is a sound operator on semantic objects (generators, abstractions, and induced measures), so that increased automation does not sacrifice correctness.

Single-cell data and high-dimensional observation maps. Rapid advances in single-cell measurement technologies increase the dimension and complexity of observation maps. Semantically, this enlarges the observation space and changes the measurable map from paths to data (Definition C.1), often yielding stronger identifiability but also new mismatch risks (hidden variables, batch structure, and selection bias). Mathematically, one can model this as an observation operator $\Phi : D([0,T],\Omega) \to \mathcal{D}$ with noise and subsampling, and study how information about θ (e.g. FIM or profile likelihood structure) changes as Φ becomes more informative.

36 Conclusion

We have presented a comprehensive formal semantics for the BioNetGen Language that unifies seven mathematical perspectives:

Category-Theoretic Foundations. We situated BNGL within the framework of adhesive categories, showing that rule application corresponds to DPO pushout constructions. This clarifies the relationship between BNGL (DPO-like semantics with explicit dangling bond handling) and Kappa (SqPO semantics with implicit side effects), and provides the mathematical foundation for the Church-Rosser property ensuring order independence for parallel rule applications.

Atoms and Atom-Rule Graphs. The atom formalism provides a canonical decomposition of patterns into minimal semantic units, enabling efficient analysis of rule interactions. The AR graph encodes complete causal dependency information with $O(|\mathcal{R}| \cdot k)$ construction complexity, compared to $O(|\mathcal{R}|^2)$ for rule influence diagrams.

Primitive Operations. The five graph operations (AddBond, DeleteBond, ChangeState, AddMol, DeleteMol) provide a complete and minimal transformation language for BNGL rules.

Rule Algebra Semantics. Connection to the Behr-Krivine framework provides an algebraic perspective on stochastic rewriting, with composition operations on rules and moment closure for observables.

Chemical Reaction Network Theory. We established connections between rule structure and CRNT properties including deficiency, weak reversibility, and persistence, enabling structural analysis of generated networks.

Bisimulation Equivalences. Forward and backward bisimulation provide exact lumpability, and rule-level bisimulation offers a compositional notion of model equivalence with a categorical characterization.

Causal and Temporal Semantics. The AR graph encodes causal structure supporting interventional queries (knockouts, perturbations), while PBLTL specifications enable statistical model checking of behavioral properties.

Three-Way Semantic Equivalence. We proved that network-based, network-free, and hybrid simulation produce identical stochastic dynamics under well-formedness conditions, providing rigorous justification for simulation strategy selection based purely on computational considerations.

36.1 Applications

This formal semantics enables:

- **Certified compilation**: Formally verified translation between BNGL and simulator implementations
- **Model equivalence checking**: Rigorous comparison of models with different syntactic representations via bisimulation
- **Static analysis**: Type checking, reachability analysis, deficiency computation, and invariant verification at the rule level
- **Optimization**: Formally justified transformations for improved simulation efficiency
- **Causal reasoning**: Support for knockout experiments and interventional queries via AR graph analysis
- **Verification**: Statistical model checking of temporal properties against simulation traces

36.2 Future Work

The core results of this paper are stated as theorems about (i) explicitly defined mathematical objects (typed site-graphs, matches, operations, and generators) and (ii) well-formedness conditions under which those objects are total and the induced stochastic dynamics are unambiguous. To fold "future work" into the main body, each item below is therefore phrased as a concrete semantics extension with: (i) an explicit extension to the static semantics (typing/well-formedness), (ii) an extension to at least one execution semantics (network, NF, hybrid), and (iii) a proof obligation of either preservation (conservativity over existing models) or equivalence (commuting diagrams relating the new semantics to the CTMC generator).

We also indicate where each feature appears in the BioNetGen implementation (`bng2.pl` and related modules) so that the formal statements can be kept aligned with the de facto language definition.

36.2.1 Compositional semantics and Markovian interfaces

A major direction is to make BNGL semantics *compositional* at the level of models, not just trajectories.

Target construction. Define an "open BNGL model" with a typed interface (selected molecule types, observable ports, and/or boundary compartments), and define a composition operation (gluing) on open models.

Target theorem (functoriality). Show that the semantics map from open BNGL models to Markov kernels/CTMC semigroups is a symmetric monoidal functor: composition of models corresponds to composition (Kleisli/Markov) of their denotations, and disjoint union of models corresponds to tensoring independent stochastic processes.

36.2.2 Large deviations and thermodynamic limits

For many BioNetGen applications one studies rare events (switching, extinction, spike timing) and asymptotic limits (large copy numbers).

Target theorem (LDP / WKB). For suitable scaling families (volume $V \to \infty$), derive a large deviation principle for the empirical concentrations and identify the associated rate function/Hamiltonian (e.g. via a WKB ansatz for the master equation). This provides rigorous asymptotics for rare-event probabilities and connects to energy-based semantics through variational principles.

36.2.3 Model reduction via bisimulation and lumpability

To make reduction constructive (not just definitional), we propose a workflow:

From rule bisimulation to a lumped generator. Given a partition $\mathcal{H}$ of species induced by a rule-level relation, define the lumping map $\pi : \mathbb{N}^{|\mathcal{S}|} \to \mathbb{N}^{|\mathcal{H}|}$ by $(\pi(\vec{x}))_i = \sum_{S \in H_i} x_S$. The goal is to give checkable conditions (syntactic on rules/patterns) implying that $\pi(X_t)$ is a CTMC.

Rule-side sufficient conditions. A practical sufficient condition is: for each rule r and each block H_i, the multiset of products projected by π depends only on the blocks of the reactants (not their concrete representatives), and the total rate out of each block-configuration is invariant under substitutions $S, S' \in H_i$. This refines Definition 30.3 into an algorithmic quotient construction.

Theorem 36.1 (Constructive lumpability criterion (target statement))**.** *Let $\mathcal{N}$ be a generated reaction network with CTMC generator Q on $\Omega = \mathbb{N}^{|\mathcal{S}|}$ and let π be induced by a partition $\mathcal{H}$ as above. Assume that for every reaction channel (equivalently, every rule instance class) its action on population vectors is compatible with π in the sense that (i) whenever $\pi(\vec{x}) = \pi(\vec{y})$ the multiset of possible projected successor states $\pi(\vec{x} + \nu)$ (counted with multiplicity) equals that of $\pi(\vec{y} + \nu)$, and (ii) the total exit rate from $\vec{x}$ into any block of projected states depends only on $\pi(\vec{x})$. Then the projected process $\pi(X_t)$ is a CTMC with a well-defined generator $\tilde{Q}$, and $\tilde{Q}$ can be computed by aggregating the original channels.*

Implementation touchpoints (BioNetGen). Species canonicalization, lookup, and indexing (the “carrier set” on which any lumping map operates) are managed in `bng2/Perl2/SpeciesList.pm` and `bng2/Perl2/Species.pm`, while reaction aggregation (including merging duplicate reactions and tracking multiplicities) is handled in `bng2/Perl2/RxnList.pm`.

36.2.4 Counterfactual reasoning (Level 3)

Counterfactuals require exogenous randomness to be represented explicitly.

Structural-noise semantics. Equip the CTMC construction with a background probability space carrying the sequence of exponential waiting times and rule-selection uniforms. A *counterfactual intervention* replaces the rule set and/or initial state while reusing the same underlying noise realization. This yields a well-defined semantics for quantities of the form $Y_{\mathrm{do}(X=x)}$ and standard counterfactual queries $P(Y_x \mid X = x', Y = y')$ under chosen noise models.

Implementation touchpoints (BioNetGen). This direction is largely *semantic* rather than an existing BNGL feature: it requires threading RNG state and sampling traces through the execution model. The cleanest integration point is at the `bng2.pl` entry level (`bng2/BNG2.pl`) and, for console-driven workflows, the wrapper layer (`bng2/Perl2/ModelWrapper.pm`).

36.2.5 Verified implementation

A verified toolchain would connect this semantics to executable simulators.

Mechanization targets. Formalize (i) the site-graph category and rewriting (DPO/SqPO), (ii) matching and symmetry factors, and (iii) the CTMC generator construction in a proof assistant (e.g., Isabelle/HOL, Coq, Lean).

Extraction. From the mechanized semantics, extract a reference interpreter for (a) network generation, (b) network-free execution, and (c) hybrid execution, together with proofs of equivalence to the abstract generator.

Implementation touchpoints (BioNetGen). Regression-style validation and golden-file comparisons (useful as a lightweight oracle while mechanizing) are centralized in `bng2/Validate/validate_examples.pl`.

Moved material. The semantics of actions/functions/options (Section 12.3) and AR-graph-based static analyses (Section 19.2) are covered in the main body and are not future work.

36.2.6 Approximate semantics with quantitative error control

BioNetGen includes approximate network-based solvers (PLA/tau-leaping). While we already specify these as an approximate semantics (Section 27.3), a stronger theory would add explicit error bounds and step-size policies as part of the language semantics.

Target theorem (weak error). Under standard regularity assumptions on propensities, show that for a family of controllers parameterized by tolerance $\epsilon \to 0$, the induced leap process $X^{(\epsilon)}$ converges weakly to the CTMC X: $X^{(\epsilon)} \Rightarrow X$ in $D([0,T],\Omega)$ for each fixed horizon T.

36.2.7 Certified compilation and semantic-preserving optimizations

A particularly BioNetGen-relevant direction is to formalize semantics-preserving transformations used by simulators and preprocessors (e.g. symmetry reduction, rule splitting, partial network expansion) as rewriting steps on models.

Target theorem (refinement). Define a refinement relation $\mathcal{B} \preceq \mathcal{B}'$ meaning every trajectory distribution of $\mathcal{B}$ is reproduced by $\mathcal{B}'$ after an explicit abstraction map; then prove standard compiler passes satisfy $\mathcal{B} \simeq \mathcal{B}'$ (distributional equality) under checkable preconditions.

36.2.8 Inference: likelihoods, gradients, and identifiability

BNGL is commonly used for parameter fitting and model selection. A semantics-level account can treat a BNGL model as defining a statistical model $\{P_\theta\}_{\theta\in\Theta}$ over trajectories/observations.

Target theorem (score/gradient). For observation functionals $Y = h(X)$ and suitable regularity, give an unbiased estimator of the score $\nabla_\theta \log p_\theta(Y)$ via pathwise (*Girsanov*-type) reweighting or coupling constructions, enabling semantics-driven gradient-based inference.

36.2.9 Optimal control and intervention as stochastic control

Many BioNetGen use-cases are explicitly interventional (drug dosing, knockouts, controller design). A fully mathematical semantics can treat interventions as controls on propensities, yielding a controlled Markov jump process.

Target theorem (HJB / dynamic programming). For a cost functional $J(u) = \mathbb{E}[\int_0^T c(X_t, u_t)\, dt + g(X_T)]$ over admissible control policies u, derive a Hamilton–Jacobi–Bellman equation for the value function $V(t, x) = \inf_u J(u \mid X_t = x)$ with generator modified by u. This connects BNGL interventions to stochastic optimal control and reinforcement learning over CTMCs.

36.2.10 Topology of complexes and rule-induced homology

Because BNGL species are graphs, one can use algebraic topology to define structural invariants of reachable complexes.

Target construction. Define a functor from species graphs to simplicial complexes (e.g. clique complexes) and compute homology groups H_k as graph-level invariants.

Target theorem (invariant preservation under rule classes). Identify syntactic classes of rules (e.g. bond rewrites without creation/deletion of molecules) that preserve selected topological invariants (Euler characteristic, Betti numbers), yielding checkable invariants for debugging and verification.

36.2.11 Bisimulation metrics and quantitative reduction

Beyond exact bisimulation/lumpability, one can define *approximate* reductions with explicit error guarantees.

Target construction. Equip the space of CTMCs (or their path measures) with a metric (e.g. total variation, Wasserstein, relative entropy rate) and define a model-reduction map $\mathcal{R}$ that minimizes distance to the original model under a complexity budget.

Target theorem (error bound). Prove that for a chosen observable class $\mathcal{F}$,

$$\sup_{f \in \mathcal{F}} |\mathbb{E}[f(X)] - \mathbb{E}[f(\mathcal{R}(X))]| \leq \varepsilon$$

for an explicit ε controlled by structural criteria (e.g. truncated rare species, lumped fast subgraphs).

36.2.12 Information geometry of energy and rate parameters

Energy-pattern semantics yields exponential-family structure (Definition 12.4). This suggests using information geometry for identifiability, sensitivity, and optimal experimental design.

Target construction. Define the Fisher information for trajectory observations under parameter θ and relate it to generator derivatives $\partial_\theta Q$ (cf. Proposition 26.8).

Target theorem (local identifiability). Give conditions under which the Fisher information is nonsingular, implying local identifiability of (k, θ) from a chosen observation scheme.

References

[1] M.L. Blinov, J.R. Faeder, B. Goldstein, and W.S. Hlavacek. BioNetGen: software for rule-based modeling of signal transduction based on the interactions of molecular domains. *Bioinformatics*, 20(17):3289–3291, 2004.

[2] J.R. Faeder, M.L. Blinov, and W.S. Hlavacek. Rule-based modeling of biochemical systems with BioNetGen. In *Systems Biology*, pages 113–167. Humana Press, 2009.

[3] L.A. Harris, J.S. Hogg, J.J. Tapia, J.A.P. Sekar, S. Gupta, I. Korsunsky, A. Arber, D. Barua, R.P. Sheehan, and J.R. Faeder. BioNetGen 2.2: advances in rule-based modeling. *Bioinformatics*, 32(21):3366–3368, 2016.

[4] W.S. Hlavacek, J.R. Faeder, M.L. Blinov, R.G. Posner, M. Hucka, and W. Fontana. Rules for modeling signal-transduction systems. *Science STKE*, 2006(344):re6, 2006.

[5] V. Danos and C. Laneve. Formal molecular biology. *Theoretical Computer Science*, 325(1):69–110, 2004.

[6] V. Danos, J. Feret, W. Fontana, R. Harmer, and J. Krivine. Rule-based modelling of cellular signalling. In *CONCUR 2007*, LNCS 4703, pages 17–41. Springer, 2007.

[7] J.A.P. Sekar, J.J. Tapia, and J.R. Faeder. Automated visualization of rule-based models. *PLoS Computational Biology*, 13(11):e1005857, 2017.

[8] J.S. Hogg, L.A. Harris, L.J. Stover, N.S. Nair, and J.R. Faeder. Exact hybrid particle/population simulation of rule-based models of biochemical systems. *PLoS Computational Biology*, 10(4):e1003544, 2014.

[9] M.W. Sneddon, J.R. Faeder, and T. Emonet. Efficient modeling, simulation and coarse-graining of biological complexity with NFsim. *Nature Methods*, 8(2):177–183, 2011.

[10] J.R. Ullmann. An algorithm for subgraph isomorphism. *Journal of the ACM*, 23(1):31–42, 1976.

[11] B.D. McKay and A. Piperno. Practical graph isomorphism, II. *Journal of Symbolic Computation*, 60:94–112, 2014.

[12] H. Ehrig, K. Ehrig, U. Prange, and G. Taentzer. *Fundamentals of Algebraic Graph Transformation.* Springer, 2006.

[13] M. Löwe. Algebraic approach to single-pushout graph transformation. *Theoretical Computer Science*, 109(1-2):181–224, 1993.

[14] S. Lack and P. Sobociński. Adhesive and quasiadhesive categories. *RAIRO - Theoretical Informatics and Applications*, 39(3):511–545, 2005.

[15] N. Behr and J. Krivine. Stochastic mechanics of graph rewriting. In *LICS 2020*, pages 46–59. ACM, 2020.

[16] M. Feinberg. *Foundations of Chemical Reaction Network Theory.* Springer, 2019.

[17] J. Pearl. *Causality: Models, Reasoning and Inference.* Cambridge University Press, 2nd edition, 2009.

[18] E.M. Clarke, T.A. Henzinger, H. Veith, and R. Bloem. *Handbook of Model Checking.* Springer, 2018.

[19] H.L.S. Younes and R.G. Simmons. Statistical probabilistic model checking with a focus on time-bounded properties. *Information and Computation*, 204(9):1368–1409, 2006.

[20] L. Cardelli, M. Tribastone, M. Tschaikowski, and A. Vandin. Symbolic computation of differential equivalences. *Theoretical Computer Science*, 777:132–154, 2019.

A Grammar Summary

```
<model>        ::= 'begin model' <block>* 'end model'
<block>        ::= <param_block> | <moltype_block> | <species_block>
                 | <obs_block> | <rule_block> | <func_block>

<param_block> ::= 'begin parameters' <param>* 'end parameters'
<param>        ::= <name> <expr>

<moltype_block> ::= 'begin molecule types' <moltype>* 'end molecule types'
<moltype>      ::= <name> '(' <comptype_list>? ')'
<comptype>     ::= <name> ('~' <state>)*

<species_block> ::= 'begin seed species' <species>* 'end seed species'
<species>      ::= <species_graph> <concentration>

<species_graph> ::= <molecule> ('.' <molecule>)*
<molecule>     ::= <name> '(' <component_list>? ')' <compartment>?
<component>    ::= <name> ('~' <state>)? <bonds>?
<bonds>        ::= ('!' (<label> | '+' | '?'))*

<rule_block>   ::= 'begin reaction rules' <rule>* 'end reaction rules'
<rule>         ::= <reactants> ('->' | '<->') <products> <ratelaw>
<reactants>    ::= <pattern> ('+' <pattern>)*
<products>     ::= <pattern> ('+' <pattern>)* | '0'
<pattern>      ::= <species_graph>

<ratelaw>      ::= <name> | <number> | <functional_rate>

<obs_block>    ::= 'begin observables' <obs>* 'end observables'
<obs>          ::= ('Molecules' | 'Species') <name> <pattern_list>
```

B Implementation Map to `bng2.pl` (from `ruleworld-bionetgen` snapshot)

To ensure this document tracks *native* BioNetGen features as implemented, we include an implementation-oriented coverage map derived from the uploaded `ruleworld-bionetgen` repository tree.

B.1 Key components in the source tree

(We write paths using `\detokenize` to avoid special-character issues in LaTeX.)

- **Entry point and core tool**: `bng2/BNG2.pl` (`bng2.pl`).
- **Graph/matching core**: `bng-graph/BNGcore/` (pattern graphs, rules, transformations, Ullmann matching).
- **Graph automorphisms**: vendored `nauty/` under `bng-graph/`.
- **Boolean logic / regulatory conditions**: `bng2/BooleanConverter/`.
- **Network generation + simulation back-end**: `bng2/Network3/src/`.

- **Non-SSA simulation families**: `bng2/Network3/src/pla/` (tau-leaping / postleap checking / step-size selection infrastructure).

B.2 Native feature checklist (documented vs. implemented)

The following table links BNGL surface features to implementation modules in the uploaded tree; it can be used as a checklist for semantic completeness.

BNGL / bng2.pl feature	**Implementation location (tree evidence)**
Pattern graphs + embeddings	`bng-graph/BNGcore/PatternGraph.cpp`, `Ullmann.cpp`
Reaction rules + transforms	`bng-graph/BNGcore/ReactionRule.hpp`, `Transformation.cpp`
Network model objects	`bng2/Network3/src/model/` (species, reaction, observable, function)
Elementary mass-action rates	`bng2/Network3/src/model/rateExpressions/rateElementary.cpp`
Functional rates	`bng2/Network3/src/model/reactions/functionalRxn.cpp`
Hill / saturation / Michaelis–Menten	`bng2/Network3/src/model/rateExpressions/`
MuParser-based expressions	`bng2/Network3/src/model/rateExpressions/rateMuParser.*`
SSA / network execution	`bng2/Network3/src/run_network.cpp`
Tau-leaping (PLA framework)	`bng2/Network3/src/pla/PLA.cpp` and `pla/base/`
Postleap correctness checks	`bng2/Network3/src/pla/base/postleapChecker.*`
Step-size / preleap checks	`bng2/Network3/src/pla/util/preleap_TC.*`
Boolean expressions	`bng2/BooleanConverter/BoolExpr*`

Semantic action item. Where this document currently defines a single generic *functional rate law*, it should be refined into explicit semantic cases matching the implemented rate expression classes (elementary, Hill, saturation, Michaelis–Menten, muparser), and the PLA/tau-leaping execution semantics should be added as an additional *approximate* semantics (distinct from the exact CTMC semantics proved equivalent for SSA/NF/hybrid).

C Structural and Practical Identifiability; Model Sloppiness

This section connects parameter identifiability and "sloppiness" (in the sense of Transtrum–Sethna and collaborators) to the BNGL semantics developed earlier. The organizing principle is that a BNGL model, once combined with a concrete observation scheme, defines a statistical model $\{\mathbb{P}_\theta\}_{\theta\in\Theta}$. Identifiability and sloppiness are properties of the induced parameter-to-data map.

C.1 BNGL parameters and the induced CTMC generator

Let $\theta \in \Theta \subseteq \mathbb{R}^p$ be the vector of all real-valued primitive parameters, typically including: (i) microscopic rule rate constants; (ii) parameters appearing in functional rate laws; and (iii) initial-condition parameters.

Under the exact semantics (SSA / network-free / hybrid), θ determines a continuous-time Markov chain (CTMC) $X^{(\theta)} = (X_t^{(\theta)})_{t\geq 0}$ on a state space Ω of mixtures (or, after network generation, on a count space $\mathbb{N}^n$), with generator Q_θ defined by

$$(Q_\theta f)(x) \triangleq \sum_{x'\neq x} Q_\theta(x,x')\,(f(x') - f(x)), \qquad f : \Omega \to \mathbb{R}.$$

In rule-based form, the off-diagonal entries are

$$Q_\theta(x,x') = \sum_{r\in\mathcal{R}} \sum_{\iota\in\mathcal{I}_r(x)} a_r(x;\theta,\iota)\,\mathbf{1}_{apply(r,\iota,x)=x'},$$

where $\mathcal{I}_r(x)$ is the set of valid rule instances (joint mappings) of r at state x, and $a_r(x;\theta,\iota)$ is the propensity contribution of instance ι. For elementary (mass-action) rules, $a_r(x;\theta,\iota) = k_r(\theta)$ is constant across instances; for local/functional rates (DOR), a_r depends on ι via local variables (cf. Definition 11.9).

By Theorem 33.4, the induced Q_θ is independent of which exact execution strategy is used; hence identifiability is a semantic property of the model, not of the simulator.

C.2 Observation schemes and the parameter-to-data map

Fix: (a) an observation map $h : \Omega \to \mathbb{R}^m$ (BNGL `observables` / summary statistics), (b) an experimental design $\mathcal{E}$ (initial state distribution, inputs, sampling times $0 < t_1 < \cdots < t_K$, and a measurement-noise kernel), and (c) a data space $(\mathcal{D}, \mathcal{F})$.

Definition C.1 (BNGL-induced statistical model)**.** The BNGL model together with $\mathcal{E}$ defines a family of probability laws on data,

$$\mathbb{P}_\theta(D \in \cdot) \in \mathbb{P}(\mathcal{D}),$$

obtained by pushing forward the trajectory law of $X^{(\theta)}$ through the sampling/noise mechanism. Equivalently, when densities exist, it defines likelihoods $\mathcal{L}(\theta \mid D)$.

A deterministic surrogate can be obtained by replacing $X^{(\theta)}$ with a mean-field (or moment-closure) flow $x(t;\theta)$ and outputs $y(t;\theta) = g(x(t;\theta);\theta)$. We will state identifiability notions in both CTMC and deterministic forms.

C.3 Structural identifiability

Definition C.2 (Structural identifiability: deterministic)**.** Fix a design $\mathcal{E}$ and horizon $T > 0$. The parameterization is *globally structurally identifiable* if

$$\forall\, \theta_1, \theta_2 \in \Theta : \quad \big(y(t;\theta_1) = y(t;\theta_2)\ \forall t \in [0,T]\big) \implies \theta_1 = \theta_2.$$

It is *locally structurally identifiable at $\theta_\star$* if the above implication holds for all θ_1, θ_2 in a neighborhood of $\theta_\star$.

Definition C.3 (Structural identifiability: CTMC)**.** For the stochastic semantics, the parameterization is *globally structurally identifiable* if

$$\forall\, \theta_1, \theta_2 \in \Theta : \quad \big(\mathbb{P}_{\theta_1} = \mathbb{P}_{\theta_2}\big) \implies \theta_1 = \theta_2,$$

where $\mathbb{P}_\theta$ abbreviates the full data law on $\mathcal{D}$ induced by Q_θ and the design $\mathcal{E}$.

BNGL-specific mechanisms for structural non-identifiability. Structural non-identifiability is often induced by symmetries and quotients inherent in rule-based semantics:

- **Observable coarsening and exact lumping.** If the data depend only on a lumped state $\pi(x)$ for some surjection $\pi : \Omega \to \tilde{\Omega}$ (e.g. via a bisimulation/lumping), then parameters that modify Q_θ only within fibers of π are structurally hidden.
- **Microscopic redundancy.** Distinct rule decompositions (different sets of microscopic rates) can induce the same generator when only a small set of observables is probed.
- **Automorphisms and counting conventions.** Because propensities depend on match counts modulo automorphisms (Section 6), reparameterizations can occur in which products of microscopic rates and symmetry factors are invariant.

C.4 Practical identifiability and Fisher information

Structural identifiability is an "in principle" property. Practical identifiability concerns how strongly finite, noisy data constrain parameters.

Definition C.4 (Fisher information)**.** Let $\ell(\theta) = \log \mathcal{L}(\theta \mid D)$ be the log-likelihood. The (expected) Fisher information matrix is

$$\mathcal{I}(\theta) \triangleq \mathbb{E}_\theta\big[\nabla_\theta \ell(\theta)\, \nabla_\theta \ell(\theta)^\top\big] = -\mathbb{E}_\theta\big[\nabla^2_\theta \ell(\theta)\big].$$

Local practical non-identifiability is reflected by singularity or extreme ill-conditioning of $\mathcal{I}(\theta_\star)$.

Deterministic sensitivity approximation. In a common Gaussian-noise model with observations $y(t_k;\theta)$ and covariance Σ, one has

$$\mathcal{I}(\theta) \approx S(\theta)^\top \Sigma^{-1} S(\theta), \qquad S_{kj}(\theta) = \frac{\partial y(t_k;\theta)}{\partial \theta_j}.$$

C.5 Sloppiness and model manifolds

Eigenvalue hierarchy. Let $\lambda_1 \geq \cdots \geq \lambda_p \geq 0$ be the eigenvalues of $\mathcal{I}(\theta_\star)$. A model is called *sloppy* near $\theta_\star$ if the eigenvalues span many orders of magnitude, i.e.

$$\frac{\lambda_1}{\lambda_p} \gg 1,$$

often with approximately geometric decay. Then only a small number of stiff combinations of parameters are identifiable from the data, while most directions are sloppy.

Geometric interpretation. For a fixed design, define the prediction map $F(\theta) = (y(t_1;\theta), \ldots, y(t_K;\theta)) \in \mathbb{R}^{mK}$ (or a vector of stochastic summary statistics). The image $\mathcal{M} = F(\Theta)$ is the *model manifold*. The pullback of the noise metric induces a Riemannian metric on Θ whose local matrix representation is the FIM. Sloppiness corresponds to $\mathcal{M}$ being extremely thin in many directions.

Connection to AR graphs. A useful semantics-driven heuristic is that stiff directions tend to align with parameters controlling rules that sit on many causal paths in the atom-rule graph $\mathrm{AR}(\mathcal{B})$ (Section 5), whereas parameters attached to rules whose enabling atoms are weakly reachable from the initial condition are typically sloppy.

C.6 Identifiability invariances induced by rule symmetries

A recurring theme in rule-based semantics is that distinct parameter vectors may induce the same generator due to symmetries of the rule representation.

Definition C.5 (Parameter invariance group)**.** Define the invariance relation

$$\theta_1 \sim_Q \theta_2 \quad \Longleftrightarrow \quad Q_{\theta_1} = Q_{\theta_2}.$$

Any group G acting on Θ by $\theta \mapsto g \cdot \theta$ such that $Q_{g\cdot\theta} = Q_\theta$ for all θ is called a *(generator) invariance group.* The orbit $G \cdot \theta$ is an *identifiability fiber* (at the generator level).

Remark C.1 (Microscopic "gauge" freedoms). Typical sources of G in BNGL include: (i) rule splitting/merging operations that preserve total propensity for each induced transition; (ii) symmetric relabelings of indistinguishable molecules/components inside patterns; and (iii) reparameterizations that trade microscopic rates against symmetry-factor conventions (Section 6).

One can regard these as "gauge" freedoms of the *presentation* of a stochastic rewriting system: distinct syntactic rule sets (and parameter vectors) can denote the same semantic object Q_θ.

Definition C.6 (Rate-preserving rule refinement). Fix θ and consider a single rule r with instance set $\mathcal{I}_r(x)$ and instance propensities $a_r(x;\theta,\iota)$. A *refinement* of r is a finite family of rules $\{r_j\}_{j\in J}$ such that for every state x and every successor state $x' \neq x$,

$$\sum_{\iota\in\mathcal{I}_r(x)} a_r(x;\theta,\iota)\,\mathbf{1}_{apply(r,\iota,x)=x'} = \sum_{j\in J}\sum_{\iota\in\mathcal{I}_{r_j}(x)} a_{r_j}(x;\theta,\iota)\,\mathbf{1}_{apply(r_j,\iota,x)=x'}.$$

Equivalently, the refinement preserves the contribution $Q_\theta^{(r)}$ to the generator.

Proposition C.1 (Refinement induces generator invariance). *If a model $\mathcal{B}$ is transformed into $\mathcal{B}'$ by finitely many rate-preserving refinements (Definition C.6), then for the same parameter vector θ the induced generators coincide: $Q_\theta^{\mathcal{B}} = Q_\theta^{\mathcal{B}'}$.*

Proof. Generators decompose additively by rules: $Q_\theta = \sum_{r\in\mathcal{R}} Q_\theta^{(r)}$. A rate-preserving refinement replaces $Q_\theta^{(r)}$ by a sum $\sum_{j\in J} Q_\theta^{(r_j)}$ without changing off-diagonal rates. Iterating over finitely many refinements yields equality. □

C.7 Generator-level identifiability on finite truncations

When network generation terminates (or under explicit boundedness constraints), Ω is finite and Q_θ is a finite matrix.

Definition C.7 (Finite truncation). Let $\Omega_N \subseteq \Omega$ be a finite, Q_θ-invariant subset for all $\theta\in\Theta$ (e.g. a bounded reachable set under model constraints). Write $Q_\theta^{(N)}$ for the restriction of Q_θ to Ω_N.

Proposition C.2 (Linear identifiability for linearly-parameterized generators). *Assume*

$$Q_\theta^{(N)} = Q_0^{(N)} + \sum_{j=1}^{p} \theta_j\, B_j$$

for fixed matrices B_j. If $\{B_1,\dots,B_p\}$ is linearly independent over $\mathbb{R}$, then $\theta \mapsto Q_\theta^{(N)}$ is injective.

Proof. If $Q_\theta^{(N)} = Q_{\theta'}^{(N)}$ then $\sum_j(\theta_j-\theta_j')B_j = 0$. Linear independence implies $\theta_j = \theta_j'$ for all j. □

Remark C.2 (Relevance to BNGL). Elementary rules often yield a generator that is affine in the microscopic rates k_r when the state space is fixed. However, functional rate laws can introduce nonlinearities in θ, and unbounded models can make Ω_N depend on θ unless boundedness is imposed as part of the semantics.

A useful way to make the matrices B_j in Proposition C.2 explicit is to write, for an elementary rule r, the generator contribution as

$$Q_\theta^{(r)}(x,x') = k_r\, c_r(x,x'), \qquad c_r(x,x') \triangleq |\{\iota\in\mathcal{I}_r(x) : apply(r,\iota,x)=x'\}|,$$

so that each B_j corresponds to the integer-valued "routing matrix" $c_{r_j}(x,x')$. In practice, linear dependence among $\{B_j\}$ arises when two distinct rules induce identical routing matrices on Ω_N (often because their effects are indistinguishable under the chosen truncation/typing), yielding structural non-identifiability at the generator level.

Proposition C.3 (Generator identifiability implies data identifiability (idealized))**.** *Fix a design $\mathcal{E}$ whose data map is injective in the generator, in the sense that for generators Q, Q' on the same finite state space, $\mathbb{P}_Q = \mathbb{P}_{Q'}$ implies $Q = Q'$. If $\theta \mapsto Q_\theta^{(N)}$ is injective, then θ is structurally identifiable from $\mathcal{E}$ on Ω_N.*

Remark C.3. The injectivity hypothesis on $Q \mapsto \mathbb{P}_Q$ depends on the observation scheme: full observation of jump times and states is sufficient, whereas coarse pattern-count observables generally are not.

C.8 Structural identifiability under interventions (design identifiability)

Definition C.8 (Multi-experiment identifiability)**.** Let $\{\mathcal{E}_\alpha\}_{\alpha\in A}$ be a family of experimental designs (initializations, inputs, interventions, sampling schemes). Define the joint data law by product measure (independent replicate experiments)

$$\mathbb{P}_\theta^A \triangleq \bigotimes_{\alpha\in A} \mathbb{P}_\theta^{(\alpha)}.$$

We say θ is *structurally identifiable from the family A* if $\mathbb{P}_{\theta_1}^A = \mathbb{P}_{\theta_2}^A$ implies $\theta_1 = \theta_2$.

Remark C.4 (Interventions as semantic modifiers). An intervention can be modeled as a map $\mathcal{B} \mapsto \mathcal{B}^{(\alpha)}$ that (i) modifies the initial distribution and/or (ii) modifies propensities by removing, clamping, or scaling selected rules (cf. Section 31). The identifiability question then becomes: does the family of generators $\{Q_\theta^{(\alpha)}\}_{\alpha\in A}$ separate points of Θ?

C.9 Profile likelihoods and non-identifiable fibers

This subsection separates two distinct notions that are often conflated: *local* identifiability as diagnosed by quadratic (Fisher-information) approximations near a reference parameter, versus *global* identifiability as a property of the full likelihood landscape.

C.9.1 Local identifiability via Fisher information (quadratic theory)

Let $\ell(\theta) = \log \mathcal{L}(\theta \mid D)$. At a point $\theta_\star$ (e.g. an MLE or a nominal parameter), local theory approximates

$$\ell(\theta) \approx \ell(\theta_\star) - \tfrac{1}{2}(\theta - \theta_\star)^\top \mathcal{I}(\theta_\star)(\theta - \theta_\star),$$

where $\mathcal{I}$ is the Fisher information (Definition C.4) or an observed information matrix.

Definition C.9 (Local practical identifiability (FIM criterion))**.** We call parameters *locally practically identifiable at $\theta_\star$* if $\mathcal{I}(\theta_\star)$ is nonsingular (equivalently, has full rank). If $\mathcal{I}(\theta_\star)$ is singular (or numerically rank-deficient), then there exist nontrivial directions $v \neq 0$ with $v^\top \mathcal{I}(\theta_\star) v \approx 0$, indicating a locally flat (quasi-unidentifiable) direction.

Remark C.5 (What FIM can and cannot say). The FIM is an intrinsically *local* object: it captures curvature of the log-likelihood near $\theta_\star$ and is therefore sensitive to reparameterized coordinates. Full rank of $\mathcal{I}(\theta_\star)$ does *not* preclude global non-identifiability (e.g. multiple separated maxima), while rank deficiency typically indicates at least a local (and often structural) non-identifiability.

C.9.2 Global identifiability via profile likelihood (landscape theory)

Definition C.10 (Profile likelihood)**.** Let $\theta = (\psi, \eta)$ be a partition of parameters into "parameters of interest" ψ and nuisance parameters η. The profile likelihood is

$$\mathcal{L}_p(\psi) \triangleq \sup_\eta \mathcal{L}(\psi, \eta \mid D).$$

Definition C.11 (Global practical identifiability (profile-likelihood criterion)). Fix D. We say ψ is *globally practically identifiable* (relative to the chosen parameter domain) if $\mathcal{L}_p(\psi)$ has a unique maximizer and admits a strict decrease away from it, i.e. there exists $\psi_\star$ such that

$$\mathcal{L}_p(\psi) < \mathcal{L}_p(\psi_\star) \quad \text{for all } \psi \neq \psi_\star.$$

Conversely, if $\mathcal{L}_p$ is constant on an interval (or has multiple equal maxima), then ψ is globally non-identifiable from the data.

Remark C.6 (FIM vs. PL: complementary failure modes). A flat (or nearly flat) profile likelihood is a *global* certificate of practical non-identifiability along the profiled coordinate. However, it is possible for $\mathcal{I}(\theta_\star)$ to be full rank even when $\mathcal{L}$ has multiple isolated maxima (global non-identifiability but local identifiability at each mode). Conversely, if $\mathcal{I}(\theta_\star)$ is singular, then the profile likelihood will typically exhibit a ridge (flatness) near $\psi_\star$, but may still decrease globally if the ridge is local and bounded by parameter-domain constraints.

In BNGL models, such ridges often arise from: (i) exact invariance fibers (Definition C.5); (ii) near-invariances producing sloppy directions; and (iii) coarse observation maps that collapse many microstates.

Remark C.7 (Relation to structural identifiability). Structural identifiability (Definitions C.2 and C.3) is design-dependent and does not reference a realized dataset D. Profile likelihoods are dataset-dependent but can be used empirically to detect whether a *putatively* structurally identifiable parameter remains practically identifiable at the noise/sampling levels of a concrete experiment.

C.10 Sloppiness, timescales, and spectral structure

In finite-state settings one can relate "stiff" directions to dynamics on observable timescales.

Definition C.12 (Resolvent sensitivity). For a bounded observable f and $t \geq 0$, the sensitivity of $\mathbb{E}_\theta[f(X_t)]$ can be expressed in terms of the semigroup $P_t^{(\theta)} = e^{tQ_\theta}$. Formally, under differentiability,

$$\partial_{\theta_j} \mathbb{E}_\theta[f(X_t)] = \int_0^t \mathbb{E}_\theta\big[(\partial_{\theta_j} Q_\theta)\, P_{t-s}^{(\theta)} f(X_s)\big]\, ds.$$

Remark C.8 (Heuristic link to stiffness). If a parameter direction primarily perturbs fast-relaxing modes (large negative real parts in $\operatorname{spec}(Q_\theta)$), then its effect may be weak on coarse sampling times, contributing to sloppiness. Conversely, perturbations that shift slow modes (small spectral gap) tend to be stiff.

C.11 AR-graph screening for (non-)identifiability

Definition C.13 (Observed atom support). Given observables (patterns) $\mathcal{O}$, define the set of *observed atom types*

$$\mathcal{A}_{obs} \triangleq \bigcup_{O \in \mathcal{O}} \bigcup_{P \in O} atoms(P),$$

where $atoms(P)$ is as in Definition 5.3.

Proposition C.4 (Sufficient condition for structural unobservability). *Assume all propensities are elementary (no DOR) and that the observation map h depends only on $\mathcal{A}_{obs}$. If, for every reachable state x and every rule instance ι of r at x, the induced transition $x \to x'$ satisfies*

$$\#(\alpha, x) = \#(\alpha, x') \quad \textit{for all } \alpha \in \mathcal{A}_{obs},$$

then the data law $\mathbb{P}_\theta$ is independent of k_r. Consequently, k_r is (globally) structurally non-identifiable from h under any sampling/noise scheme.

Remark C.9. This criterion is conservative: it detects parameters that are completely unobservable given the chosen atom support. It is a *global* statement (about equality of induced data laws), and should not be confused with *local* practical identifiability diagnostics based on the Fisher information matrix (Section C.4).

Remark C.10 (AR reachability refinement). A practical, checkable relaxation replaces the quantified "for every reachable state" condition by an AR-graph causality test: if no atom produced/consumed by r can reach any atom in $\mathcal{A}_{obs}$ in $\mathrm{AR}(\mathcal{B})$, then k_r cannot influence the observation through any AR-visible causal chain. This yields a fast static pre-screen before any numerical fitting.

Definition C.14 (AR-observable closure)**.** Let $\mathit{Reach}_{\mathrm{AR}}(X)$ denote the set of atom types reachable from a set $X \subseteq \mathcal{A}$ by directed paths in the bipartite graph $\mathrm{AR}(\mathcal{B})$ (allowing alternation through rule nodes). Define the *AR-observable closure*

$$\overline{\mathcal{A}}_{obs} \triangleq \mathit{Reach}_{\mathrm{AR}}(\mathcal{A}_{obs}).$$

Intuitively, $\overline{\mathcal{A}}_{obs}$ contains all atom types that can causally influence observed atoms through some finite chain of rule firings.

Proposition C.5 (AR reachability implies necessary observability)**.** *Assume elementary propensities and that h depends only on $\mathcal{A}_{obs}$. If a rule r only affects atom types outside $\overline{\mathcal{A}}_{obs}$ (i.e. all reactant/product atoms of r lie in $\mathcal{A} \setminus \overline{\mathcal{A}}_{obs}$), then perturbations of its rate constant k_r cannot affect h through any causal chain visible to the AR graph. In particular, if additionally the model admits an exact abstraction that is sound with respect to AR reachability (Section 19.2), then k_r is structurally non-identifiable from h.*

Remark C.11. The additional hypothesis in the last sentence is needed because AR reachability is an over-approximation: paths may exist in $\mathrm{AR}(\mathcal{B})$ that are not realizable in the concrete dynamics due to global constraints (e.g. mutually exclusive states). Nevertheless, the closure $\overline{\mathcal{A}}_{obs}$ provides a fast, semantics-aligned filter for "hopeless" parameters.

Definition C.15 (AR-observable closure)**.** Let $\mathit{Reach}_{\mathrm{AR}}(X)$ denote the set of atom types reachable from a set $X \subseteq \mathcal{A}$ by directed paths in the bipartite graph $\mathrm{AR}(\mathcal{B})$ (allowing alternation through rule nodes). Define the *AR-observable closure*

$$\overline{\mathcal{A}}_{obs} \triangleq \mathit{Reach}_{\mathrm{AR}}(\mathcal{A}_{obs}).$$

Intuitively, $\overline{\mathcal{A}}_{obs}$ contains all atom types that can causally influence observed atoms through some finite chain of rule firings.

Proposition C.6 (AR reachability implies necessary observability)**.** *Assume elementary propensities and that h depends only on $\mathcal{A}_{obs}$. If a rule r only affects atom types outside $\overline{\mathcal{A}}_{obs}$ (i.e. all reactant/product atoms of r lie in $\mathcal{A} \setminus \overline{\mathcal{A}}_{obs}$), then perturbations of its rate constant k_r cannot affect h through any causal chain visible to the AR graph. In particular, if additionally the model admits an exact abstraction that is sound with respect to AR reachability (Section 19.2), then k_r is structurally non-identifiable from h.*

Remark C.12. The additional hypothesis in the last sentence is needed because AR reachability is an over-approximation: paths may exist in $\mathrm{AR}(\mathcal{B})$ that are not realizable in the concrete dynamics due to global constraints (e.g. mutually exclusive states). Nevertheless, the closure $\overline{\mathcal{A}}_{obs}$ provides a fast, semantics-aligned filter for "hopeless" parameters.

C.12 Information geometry of trajectory space

Definition C.16 (Path-space Fisher information)**.** Let $\mathbb{Q}_\theta$ denote the probability measure on trajectory space $D([0,T],\Omega)$ induced by Q_θ and an initial distribution. When $\mathbb{Q}_\theta$ admits a differentiable density with respect to a reference measure, define the path-space Fisher information by

$$\mathcal{I}^{path}(\theta) \triangleq \mathbb{E}_{\mathbb{Q}_\theta}\left[\nabla_\theta \log d\mathbb{Q}_\theta \ \nabla_\theta \log d\mathbb{Q}_\theta^\top\right].$$

Coarse observation (sampling/noise) corresponds to applying a measurable map Φ from paths to data; by the data processing inequality, Fisher information cannot increase under Φ.

Remark C.13 (Sampling-induced sloppiness). In particular, if data are sparse samples of a high-dimensional mixture process, then many parameter directions that are distinguishable on path space become sloppy after pushforward to the observation σ-algebra.

C.13 Optimal experimental design (OED)

Optimal experimental design (OED) makes the dependence on the observation scheme $\mathcal{E}$ explicit: by selecting what to measure, when to measure, and which interventions to apply, one aims to maximize identifiability (reduce non-identifiabilities and sloppiness).

C.13.1 Design variables as semantic data

Let the *design space* be a set $\mathcal{D}_{design}$ whose elements encode:

- **Sampling schedule:** times $0 < t_1 < \cdots < t_K \leq T$;
- **Observable selection:** a finite subset $\mathcal{O}_d$ of BNGL observables/pattern counts;
- **Noise model:** a measurement kernel Noise_d;
- **Interventions/inputs:** modifications $\mathcal{B} \mapsto \mathcal{B}^{(d)}$ (initial conditions, clamped rule rates, knockouts, stimulus pulses), yielding a generator $Q_\theta^{(d)}$.

The semantic output of a design is therefore a statistical model $\{\mathbb{P}_\theta^{(d)}\}_{\theta\in\Theta}$ (Definition C.1).

C.13.2 Classical local criteria

Definition C.17 (Local OED criteria)**.** Fix a nominal parameter $\theta_\star$. Let $\mathcal{I}(\theta_\star; d)$ denote the (expected) Fisher information for design d. Standard local design objectives include:

$$\text{D-optimal:} \quad \max_{d\in\mathcal{D}_{design}} \log\det(\mathcal{I}(\theta_\star; d)), \qquad \text{E-optimal:} \quad \max_{d\in\mathcal{D}_{design}} \lambda_{\min}(\mathcal{I}(\theta_\star; d)),$$

$$\text{A-optimal:} \quad \min_{d\in\mathcal{D}_{design}} \operatorname{tr}(\mathcal{I}(\theta_\star; d)^{-1}), \qquad \text{G-optimal:} \quad \min_{d\in\mathcal{D}_{design}} \max_{j\leq p}(\mathcal{I}(\theta_\star; d)^{-1})_{jj},$$

when $\mathcal{I}$ is invertible.

Remark C.14 (Interpretation). D-optimality penalizes overall volume of the local confidence ellipsoid, whereas E-optimality targets the worst-constrained direction (smallest eigenvalue) and is therefore directly tied to sloppiness. A- and G-optimality make the dependence on marginal variances explicit.

C.13.3 Robust and Bayesian design

Local criteria depend on a nominal $\theta_\star$. For sloppy BNGL models one often instead uses robust criteria:

$$\text{Bayesian D-optimal:} \quad \max_d \ \mathbb{E}_{\theta\sim\pi}[\log\det \mathcal{I}(\theta; d)], \qquad \text{minimax:} \quad \max_d \ \inf_{\theta\in\Theta_0} \ \lambda_{\min}(\mathcal{I}(\theta; d)).$$

Here π is a prior (or posterior from previous experiments) and Θ_0 is an uncertainty set.

C.13.4 Computing $\mathcal{I}(\theta; d)$ for BNGL models

Deterministic surrogate. If an ODE limit is used, $\mathcal{I}(\theta; d)$ is commonly approximated by $S(\theta; d)^\top \Sigma(d)^{-1} S(\theta; d)$ with sensitivities $S_{kj} = \partial y(t_k; \theta)/\partial\theta_j$.

Stochastic semantics. For the CTMC semantics, $\mathcal{I}$ can be estimated via likelihood-ratio (Girsanov-type) methods or via pathwise derivatives when available. In either case, the semantic decomposition $Q_\theta = \sum_{r\in\mathcal{R}} Q_\theta^{(r)}$ induces a *rule-wise* decomposition of gradient estimators, reflecting how simulators organize event channels.

Finite-state (generated network) route. When network generation terminates and the observation map is linear in the species-count vector (Definition 11.15), one can compute local sensitivities by differentiating the semigroup $P_t(\theta) = e^{tQ_\theta}$. In particular,

$$\partial_{\theta_j} P_t(\theta) = \int_0^t P_{t-s}(\theta)\,(\partial_{\theta_j} Q_\theta)\,P_s(\theta)\,ds,$$

so that for an initial distribution μ_0 and observable vector w,

$$\partial_{\theta_j}\,\mathbb{E}_\theta[w^\top X_t] = \mu_0^\top\,\partial_{\theta_j} P_t(\theta)\,w.$$

This gives a principled baseline for $\mathcal{I}(\theta; d)$ in small networks (where matrix exponentials / uniformization are feasible).

Path-space score estimators (network-free / hybrid). For exact simulators (SSA, network-free, hybrid), one can estimate gradients of log-likelihoods or of expected summary statistics from sample paths using the standard "score" identity. At the level of the generator decomposition $Q_\theta = \sum_r Q_\theta^{(r)}$, a convenient representation is as a sum over rule-channel counting processes. For elementary rules with rates $k_r(\theta)$, the log-density of a complete trajectory (jump times and selected rule instances) contains the terms

$$\sum_{\text{events of } r} \log k_r(\theta) \ - \int_0^T k_r(\theta)\,N_r(X_t)\,dt$$

(with N_r the instance count), yielding the score contribution

$$\partial_{\theta_j} \log p_\theta(\omega) = \sum_r \left(\frac{\partial_{\theta_j} k_r}{k_r}\,N_r^{\text{fire}}(\omega) - \int_0^T (\partial_{\theta_j} k_r)\,N_r(X_t)\,dt \right)$$

whenever $k_r > 0$ and parameters enter only through k_r. This yields Monte-Carlo estimators for Fisher information by averaging outer products of such scores.

Local functions and DOR. For DOR rules (Definition 11.9), the "rate per event" depends on the selected joint mapping μ, so the score must include additional terms for the instance-level factor $[\![e_r]\!]_{\sigma,\mu}$. This is the main reason OED for functional-rate BNGL models is typically more expensive: the information content is concentrated in rare mappings that realize specific local-variable bindings.

Practical estimation workflow. In practice one often uses: (i) a deterministic surrogate for fast screening of candidate designs, (ii) a small number of exact stochastic replicates to estimate a rough $\mathcal{I}(\theta; d)$, and (iii) design refinement (e.g. by gradient-free search over sampling times and observable sets) using that estimator. The AR-graph reachability heuristics in the next subsection provide a fast pre-filter that avoids spending simulation budget on designs that are structurally incapable of informing chosen parameters.

C.13.5 Semantics-driven design heuristics

Remark C.15 (AR-graph coverage heuristic). A fast structural heuristic is to choose designs d so that the observable atom support $\mathcal{A}_{obs}(d)$ (Definition C.13) intersects the causal downstream of as many rule-affected atoms as possible. Equivalently, select observables so that the AR-observable closure $\overline{\mathcal{A}}_{obs}(d)$ (Definition C.15) contains the atoms produced/consumed by rules whose parameters are of interest. This does not guarantee identifiability, but it eliminates designs that are structurally incapable of informing specific parameters.

Remark C.16 (Interventions as symmetry breakers). Interventions can be selected specifically to break generator invariances (Section C.6), e.g. by driving the system into regions of state space where two otherwise redundant rules have different context realizations, or by clamping subsets of rules to isolate the contribution of a single parameter direction.

C.13.6 Sequential design (adaptive experiments)

In practice, designs are chosen iteratively. Given data D_1 from design d_1, update a posterior (or confidence set) $\pi(\theta \mid D_1)$ and choose the next design by maximizing a Bayesian criterion. This yields a semantics-aligned active-learning loop:

$$\pi(\theta) \xrightarrow{d_1} D_1 \xrightarrow{\text{Bayes}} \pi(\theta \mid D_1) \xrightarrow{d_2} D_2 \to \cdots .$$

Remark C.17 (BNGL structure). Because rule parameters enter propensities locally through pattern contexts, $\mathcal{I}(\theta; d)$ often has exploitable sparsity/low-rank structure as a function of d. This creates opportunities for semantics-driven design heuristics using AR-graph reachability and causal-path coverage.

C.14 Identifiability of functional rates and local functions (DOR)

Remark C.18 (Instance-dependent propensities). Under DOR (Definition 11.9), the propensity contribution $a_r(x; \theta, \iota)$ depends on the joint mapping ι. Consequently, parameters in local expressions are identifiable only through the subset of instances that realize the local variable bindings. If those bindings occur rarely (in the CTMC sense), such parameters are typically extremely sloppy (and can be practically non-identifiable even when structurally identifiable).

C.15 Effective parameters and semantics-preserving reduction

Definition C.18 (Effective parameter map). Let $U \subseteq \Theta$ be a neighborhood of $\theta_\star$. An *effective parameterization* is a smooth map $\psi : U \to \mathbb{R}^q$ (with $q < p$) such that for the chosen design and an observable class $\mathcal{F}$,

$$\sup_{f \in \mathcal{F}} \sup_{t \in [0,T]} \left| \mathbb{E}_\theta[f(X_t)] - \mathbb{E}_{\theta'}[f(X_t)] \right| \le \varepsilon \quad \text{whenever } \psi(\theta) = \psi(\theta').$$

Remark C.19. In practice, ψ can be chosen using the stiff eigen-directions of $\mathcal{I}(\theta_\star)$ (principal directions of the local information geometry). Semantics-preserving reductions (exact lumping) correspond to the case $\varepsilon = 0$.

D Complexity Analysis

Operation	**Complexity**	**Notes**
Pattern matching	$O(\lvert S\rvert^{\lvert P\rvert})$	Ullmann algorithm
Canonical labeling	$O(\text{poly})$ typical	nauty; worst case exp.
Automorphism group	$O(\text{poly})$ typical	Via canonical labeling
AR graph construction	$O(\lvert\mathcal{R}\rvert \cdot k_{\max})$	Linear in model size
Network generation	$O(\lvert\mathcal{S}\rvert^2 \cdot \lvert\mathcal{R}\rvert)$	Per iteration
NF simulation step	$O(\lvert\mathcal{R}\rvert \cdot k^2)$	Per reaction
Hybrid step	$O(\lvert\mathcal{X}_{\text{pop}}\rvert + \lvert\mathcal{R}_{\text{part}}\rvert \cdot k^2)$	Mixed

E Proof Details

E.1 Proof of Lemma 6.4

Proof. By the orbit-stabilizer theorem, $|[\phi]| \cdot |\text{Stab}(\phi)| = |\text{Aut}([\![P]\!])|$. If $\phi([\![P]\!])$ has trivial automorphism group in $[\![S]\!]$, then any $\alpha \in \text{Aut}([\![P]\!])$ with $\phi \circ \alpha = \phi$ must satisfy $\alpha = \text{id}$. Thus $\text{Stab}(\phi) = \{\text{id}\}$ and $|[\phi]| = |\text{Aut}([\![P]\!])|$. □

E.2 Proof of Lemma 7.5

Proof. Operations on disjoint graph elements modify disjoint sets of nodes, edges, and attributes. Since the operations are pure functions on disjoint state components, their composition is independent of order:

$$[\![o_1]\!] \circ [\![o_2]\!](G) = [\![o_2]\!] \circ [\![o_1]\!](G)$$

when $\mathit{affected}(o_1) \cap \mathit{affected}(o_2) = \emptyset$. □

E.3 Explicit combinatorial derivation of rate constants

We make the mapping between network (species-based) rate constants k_x and rule-based (microscopic) rate constants k_r fully explicit. Let a rule pattern P have automorphism group $\text{Aut}(P)$ with symmetry factor $\text{sym}(P) = |\text{Aut}(P)|$. Assume the network-free semantics counts *ordered* matches of P into a mixture M. Let $N(P, M)$ be the number of ordered embeddings of P into M. Then the network-free propensity is

$$a_{\text{NF}}(M) = k_r\, N(P, M). \tag{1}$$

For the corresponding network reaction X with mass-action rate k_x, the network propensity is

$$a_{\text{net}}(M) = k_x \prod_i \binom{n_i}{\nu_i}, \tag{2}$$

where n_i is the population of species i and ν_i is its stoichiometric coefficient in X. For each pattern P, the relationship between ordered and unordered matches is

$$N(P, M) = \text{sym}(P) \prod_i \binom{n_i}{\nu_i}. \tag{3}$$

Equating propensities yields the exact conversion

$$k_r = \frac{k_x}{\text{sym}(P)}. \tag{4}$$

Example (homodimerization). For $A + A \to A_2$, we have $\mathrm{sym}(P) = 2$. The network propensity is $a_{\text{net}} = k_x\binom{n_A}{2} = k_x n_A(n_A - 1)/2$. The network-free semantics counts ordered matches $N(P, M) = n_A(n_A - 1)$, so $a_{\text{NF}} = k_r n_A(n_A - 1)$. Thus $k_r = k_x/2$, in agreement with the general formula above.

E.4 Canonical labeling and complexity notes

All results that rely on canonical representatives of molecular graphs assume the existence of a canonical labeling algorithm. In practice, this is implemented using graph isomorphism software (e.g., nauty or bliss). While average-case performance is efficient, worst-case complexity of canonical labeling is not known to be polynomial. This does not affect semantic correctness, but it bounds implementation complexity rather than mathematical validity.

Explicit inverse construction (Prop. 5.1). Given a multiset of atoms encoding component states, free sites, and bonds, one reconstructs a unique (up to isomorphism) species graph by: (i) instantiating one node per molecule atom, (ii) assigning component states and site states, (iii) connecting sites according to bond atoms, and (iv) applying canonical labeling to fix a unique representative. This makes the bijection between abstract states and species explicit.

F Categorical formulation of pattern matching

We formalize pattern matching as a functor between categories of graphs. Let **Patt** be the category whose objects are pattern graphs and whose morphisms are graph isomorphisms. Let **Mix** be the category of mixture graphs with injective homomorphisms. A match of pattern P in mixture M is an injective graph homomorphism $f : P \to M$ preserving labels, states, and bonds.

Define the matching functor

$$\mathcal{M} : \mathbf{Patt}^{op} \times \mathbf{Mix} \to \mathbf{Set}$$

by $\mathcal{M}(P, M) = \{f \mid f : P \to M \text{ is an injective homomorphism}\}$. Automorphisms of P act on $\mathcal{M}(P, M)$ by precomposition, yielding the orbit set

$$\mathcal{M}(P, M)/\mathrm{Aut}(P).$$

Unordered (network) matches correspond to these orbits, while ordered (network-free) matches correspond to elements of $\mathcal{M}(P, M)$ itself. Thus,

$$|\mathcal{M}(P, M)| = |\mathrm{Aut}(P)| \cdot |\mathcal{M}(P, M)/\mathrm{Aut}(P)|.$$

recovering the symmetry factor relation used in Section 33.

G Formal proof sketch of Lemma 33.2

Let P be a rule pattern and X its induced network reaction with stoichiometry ν_i for species i. For a mixture M with populations n_i, the number of unordered embeddings of P is $\prod_i \binom{n_i}{\nu_i}$. Each unordered embedding corresponds to $\mathrm{sym}(P)$ ordered embeddings. Thus $N(P, M) = \mathrm{sym}(P)\prod_i \binom{n_i}{\nu_i}$. Then

$$a_{\text{NF}}(M) = k_r N(P, M) = k_r \mathrm{sym}(P) \prod_i \binom{n_i}{\nu_i}.$$

Equating with $a_{\text{net}}(M) = k_x \prod_i \binom{n_i}{\nu_i}$ gives $k_r = k_x/\mathrm{sym}(P)$. □

G.1 Metric Semantics

Definition G.1 (Behavioral Metric)**.** Define a metric $d : \Omega \times \Omega \to [0,1]$ by:

$$d(\sigma_1, \sigma_2) = \sup_{t \geq 0} d_{TV}(P_t(\sigma_1, \cdot), P_t(\sigma_2, \cdot))$$

where d_{TV} is total variation distance and P_t is the transition kernel.

Proposition G.1 (Metric vs. Bisimulation)**.** *$d(\sigma_1, \sigma_2) = 0$ iff σ_1 and σ_2 are bisimilar.*

Definition G.2 (Quantitative Bisimulation)**.** States σ_1, σ_2 are *ϵ-bisimilar* if $d(\sigma_1, \sigma_2) \leq \epsilon$.

H Categorical Semantics of Compartments

We develop the categorical structure underlying compartmental BNGL models.

H.1 The Category of Compartmented Graphs

Definition H.1 (Compartment Category)**.** Let $\mathcal{K}$ be a compartment structure (as a graph of adjacencies). Define category $\mathbf{SGraph}_{\mathcal{K}}$ with:

- Objects: Site-graphs with compartment annotations
- Morphisms: Graph morphisms preserving compartment labels

Proposition H.1 (Compartment Forgetting Functor)**.** *There is a forgetful functor $U : \mathbf{SGraph}_{\mathcal{K}} \to \mathbf{SGraph}$ that preserves:*

- *Pullbacks and pushouts*
- *Adhesive structure*
- *DPO derivations (mapping to uncompartmented derivations)*

Definition H.2 (Transport as Functor)**.** Transport rules define a functor $T_{\kappa \to \kappa'} : \mathbf{SGraph}_{\kappa} \to \mathbf{SGraph}_{\kappa'}$ that relabels compartments while preserving graph structure.

H.2 Indexed Categories for Multi-Compartment Systems

Definition H.3 (Indexed Category)**.** An *indexed category* over $\mathcal{K}$ is a functor:

$$\mathcal{G} : \mathcal{K}^{op} \to \mathbf{Cat}$$

assigning to each compartment κ a category $\mathcal{G}(\kappa)$ of local graphs.

Definition H.4 (Grothendieck Construction)**.** The *total category* $\int \mathcal{G}$ has:

- Objects: pairs (κ, G) with $\kappa \in \mathcal{K}$, $G \in \mathcal{G}(\kappa)$
- Morphisms: pairs (f, ϕ) where $f : \kappa \to \kappa'$ in $\mathcal{K}$ and $\phi : G \to \mathcal{G}(f)(G')$

Proposition H.2 (Compartmented BNGL via Grothendieck)**.** *$\mathbf{SGraph}_{\mathcal{K}}$ is equivalent to the total category $\int \mathcal{G}$ for appropriate $\mathcal{G}$.*

I Homotopy Type Theory Perspective

We sketch connections between BNGL semantics and homotopy type theory (HoTT).

I.1 Species as Higher Types

Definition I.1 (Species as Groupoid)**.** The collection of species forms a groupoid $\mathcal{S}_{gpd}$:

- Objects: concrete species (up to isomorphism)
- Morphisms: isomorphisms between species
- Composition: isomorphism composition

Remark I.1 (Univalence for Species). In HoTT terms, the type of species should satisfy univalence: $(S_1 = S_2) \simeq (S_1 \cong S_2)$. Identity of species *is* isomorphism.

Definition I.2 (Path Space of Reactions)**.** A reaction $S_1 \to S_2$ can be viewed as a path in a type-theoretic sense. The *reaction path space* $\mathit{Path}(S_1, S_2)$ contains all reaction sequences from S_1 to S_2.

I.2 Higher Inductive Types for Rules

Definition I.3 (Rule as HIT Generator)**.** A BNGL rule can be encoded as a *higher inductive type* (HIT):

- Point constructors: species (0-cells)
- Path constructors: reactions (1-cells)
- Higher constructors: reaction composition laws (2-cells)

Proposition I.1 (Fundamental Groupoid)**.** *The fundamental groupoid $\pi_1(\mathit{Model})$ of a BNGL model captures:*

- *Objects: reachable species*
- *Morphisms: equivalence classes of reaction paths*

J Game-Theoretic Semantics

We interpret BNGL dynamics through game-theoretic lenses.

J.1 Molecular Games

Definition J.1 (Molecular Game)**.** A *molecular game* $\mathcal{G} = (N, \Sigma, (u_i)_{i \in N})$ consists of:

- N: set of "players" (molecule types)
- Σ: strategy profiles (binding configurations)
- u_i: utility function for player i (e.g., stability energy)

Definition J.2 (Nash Equilibrium as Steady State)**.** A *Nash equilibrium* configuration corresponds to a state where no single molecular binding change is energetically favorable:

$$\forall i \in N, \forall \sigma_i' : u_i(\sigma_i, \sigma_{-i}) \geq u_i(\sigma_i', \sigma_{-i})$$

Proposition J.1 (Energy Semantics and Potential Games)**.** *Under energy-based semantics (Section 12.2), the BNGL model defines a* potential game *with potential function $\mathcal{H}$. Nash equilibria correspond to local minima of $\mathcal{H}$.*

K Quantum-Inspired Semantics

We explore quantum-inspired mathematical structures for BNGL.

K.1 Superposition of Species

Definition K.1 (Quantum Species State)**.** A *quantum species state* is a unit vector in the Hilbert space:

$$|\psi\rangle = \sum_{S \in \mathcal{S}} \alpha_S \, |S\rangle, \quad \sum_S |\alpha_S|^2 = 1.$$

Remark K.1 (Interpretation)*.* While actual molecular systems are classical, quantum-inspired mathematics can capture:

- Uncertainty in species identity before measurement
- Superposition of rule applications (before selection)
- Entanglement-like correlations between components

Definition K.2 (Reaction as Unitary)**.** A reaction $x : S_1 \to S_2$ defines a partial isometry:

$$U_x \, |S_1\rangle = |S_2\rangle.$$

Extended to density matrices for mixed states.

K.2 Lindblad Dynamics

Definition K.3 (Lindblad Master Equation)**.** The quantum analog of the CME is the Lindblad equation:

$$\frac{d\rho}{dt} = -i[H, \rho] + \sum_k \left(L_k \rho L_k^\dagger - \frac{1}{2}\{L_k^\dagger L_k, \rho\} \right)$$

where ρ is the density matrix, H is the Hamiltonian, and L_k are Lindblad operators.

Proposition K.1 (Classical Limit)**.** *When restricted to diagonal density matrices (classical probability distributions), Lindblad dynamics reduces to the classical master equation (CME).*

L Topos-Theoretic Perspective

We situate BNGL semantics within topos theory.

L.1 Presheaf Topos of Patterns

Definition L.1 (Pattern Presheaf)**.** Define the presheaf topos $\hat{\mathcal{P}} = [\mathcal{P}^{op}, \mathbf{Set}]$ where $\mathcal{P}$ is the category of patterns with refinement morphisms.

Proposition L.1 (Internal Logic)**.** *The internal logic of $\hat{\mathcal{P}}$ provides:*

- *Intuitionistic reasoning about patterns*
- *Kripke semantics for modal properties*
- *Natural treatment of partial information (wildcards)*

Definition L.2 (Sheaf Condition for Species)**.** A presheaf $F : \mathcal{P}^{op} \to \mathbf{Set}$ is a *sheaf* if matching data glue uniquely: for a covering family $(P_i \hookrightarrow P)$, compatible elements $x_i \in F(P_i)$ have a unique amalgamation in $F(P)$.

L.2 Grothendieck Topos of Site-Graphs

Definition L.3 (Site-Graph Site)**.** Define a Grothendieck site $(\mathbf{SGraph}, J)$ where:

- **SGraph** is the category of site-graphs
- J is the coverage where $(G_i \to G)$ covers G iff $\bigcup G_i = G$

Proposition L.2 (Topos of Species)**.** *The category* $\mathbf{Sh}(\mathbf{SGraph}, J)$ *of sheaves is a topos that provides:*

- *Subobject classifier for pattern matching*
- *Exponentials for function spaces*
- *Internal language for reasoning about molecular systems*

M Algorithmic Information Theory

We apply concepts from algorithmic information theory to BNGL models.

M.1 Kolmogorov Complexity of Species

Definition M.1 (Species Complexity)**.** The *Kolmogorov complexity* $K(S)$ of a species S is the length of the shortest program (in a fixed universal language) that outputs the canonical representation of S.

Proposition M.1 (Complexity Bounds)**.** *For species S with n molecules of k types:*

$$K(S) = O(n \log k + n \log n)$$

The first term encodes molecule types; the second encodes bond structure.

Definition M.2 (Model Complexity)**.** The complexity of a BNGL model $\mathcal{B}$ is:

$$K(\mathcal{B}) = K(\mathcal{M}) + K(\mathcal{R}|\mathcal{M}) + K(\Sigma_0|\mathcal{M}, \mathcal{R})$$

using conditional complexity for each component.

M.2 Minimum Description Length

Definition M.3 (MDL for Parameter Estimation)**.** The *minimum description length* principle selects parameters minimizing:

$$L(\theta) + L(D|\theta)$$

where $L(\theta)$ is the description length of parameters and $L(D|\theta)$ is the compressed data length given the model.

Proposition M.2 (MDL and Identifiability)**.** *Parameters that don't affect $L(D|\theta)$ have arbitrary optimal values, indicating non-identifiability. MDL provides an information-theoretic characterization of sloppy directions.*

M.3 Parameter Estimation: A Fisher-Information Case Study

Consider the simple gene-expression module with transcription rate k_{txn}, translation k_{tln}, and degradation rates $k_{deg,m}, k_{deg,p}$. Observing mRNA and protein counts over time yields a Fisher information matrix (FIM) whose entries can be approximated via linear noise or simulation-based score estimates.

FIM expression (symbolic) For a Gaussian approximation with mean trajectory $\mu(t;\theta)$ and covariance $\Sigma(t;\theta)$, the instantaneous Fisher information is

$$I_{ij}(t) = \frac{\partial \mu^\top}{\partial \theta_i}\Sigma^{-1}\frac{\partial \mu}{\partial \theta_j} + \tfrac{1}{2}\mathrm{tr}\left(\Sigma^{-1}\frac{\partial \Sigma}{\partial \theta_i}\Sigma^{-1}\frac{\partial \Sigma}{\partial \theta_j}\right).$$

Numerical illustration For nominal parameters $k_{txn} = 10, k_{tln} = 1, k_{deg,m} = 0.1, k_{deg,p} = 0.01$ and measurement times $t \in \{10, 50, 100, 200\}$, the leading eigenvalues of the integrated FIM indicate which parameter combinations are estimable (stiff) and which are sloppy.

Design implication Eigenvectors corresponding to small eigenvalues point to directions where adding measurements (higher frequency or additional observables) increases information most effectively.

N Nonstandard Analysis for BNGL

We sketch applications of nonstandard analysis to BNGL semantics.

N.1 Infinitesimal Propensities

Definition N.1 (Hyperreal Rates)**.** In a nonstandard extension ${}^*\mathbb{R}$, consider infinitesimal rate constants $\epsilon \approx 0$ and infinite copy numbers $N \approx \infty$ such that ϵN is finite.

Proposition N.1 (Thermodynamic Limit via Transfer)**.** *The thermodynamic limit (ODE approximation) is obtained by:*

1. *Considering hyperfinite copy numbers $N \in {}^*\mathbb{N}$*
2. *Scaling time by $\epsilon = 1/N$*
3. *Applying the standard part map:* $\mathrm{st}(X_t)$ *satisfies the ODE*

N.2 Infinitesimal Generator

Definition N.2 (Nonstandard Generator)**.** For infinitesimal time step dt, the generator action is:

$$(Qf)(x) = \frac{\mathbb{E}\big[f(X_{dt}) \mid X_0 = x\big] - f(x)}{dt}$$

where equality is up to infinitesimals.

Proposition N.2 (Martingale via Nonstandard)**.** *The martingale property (Theorem 26.4) can be expressed as: for all internal times t,*

$$\mathbb{E}[f(X_t) - f(X_0)] \approx \mathbb{E}\big[\sum_{s<t}(Qf)(X_s) \cdot dt\big]$$

where the sum is over hyperfinitely many infinitesimal steps.

O Appendix: Mechanized Proofs and Formalization Roadmap

We outline a small mechanization project for formalizing BNGL proofs (e.g. Theorem 19.1) in a proof assistant such as Coq or Isabelle.

Core definitions mapping

- Site-graphs -> inductive types for nodes/edges with labels
- Patterns and matches -> dependent types encoding injective embeddings
- Rules as labelled graph morphisms and operation sequences as functions

Proof strategy Mechanize Lemma: if an atom type is not reachable in the AR closure, then it cannot appear in any reachable state. This requires inductive reasoning over derivation length and a constructive closure operator implementation.

Roadmap (1) Formalize finite site-graphs and pattern matching; (2) implement AR graph construction; (3) prove closure operator soundness; (4) discharge theorem by induction. This yields a reusable library for further certified results.

P Conclusion of Additional Sections

The material in this document extends the core BNGL semantics with:

1. **Language Features**: Tagging, events, deletion semantics, reactant modifiers, extended compartments
2. **Implementation Formalization**: NFsim data structures, observable algebra, implementation correspondence
3. **Translations**: Stochastic Petri nets, process algebra
4. **Analysis Tools**: Information theory, extended proofs
5. **Examples**: Multi-site phosphorylation, receptor clustering, gene networks
6. **Advanced Perspectives**: Refinement types, coalgebraic semantics, domain theory, categorical compartments, HoTT, game theory, quantum-inspired, topos theory, algorithmic information, nonstandard analysis

These extensions provide multiple complementary viewpoints on BNGL semantics, connecting the language to diverse areas of mathematics and computer science. The formal foundations support:

- Rigorous verification of model properties
- Principled model reduction and approximation
- Type-safe model composition
- Information-theoretic analysis of identifiability
- Connections to established theories in concurrency, logic, and physics

Future work may further develop:

- Mechanized proofs in proof assistants
- Certified compilation from semantics to efficient simulators
- Automated reasoning tools based on the formal semantics
- Connections to emerging areas (quantum computing, category theory for open systems)