Graph-Theoretical Foundations of Polyubiquitin Architecture

Eric Kummelstedt

August 6, 2025

Introduction

Polyubiquitin chains, due to their complex branched architecture, require a rigorous mathematical formalism for computational manipulation, enumeration, and automated synthesis. In this document, we formalize the polyubiquitin structure using the language of rooted directed trees (a subclass of directed acyclic graphs) with biologically meaningful constraints. Each mathematical definition is accompanied by its corresponding biochemical interpretation.

1. Formal Graph Definition

Definition of Graph

Let $G = (V, E, \lambda, \tau)$ be a rooted directed tree where:

- V is the set of nodes.
- $E \subseteq V \times V$ is the set of directed edges.
- $\lambda : E \to L$ labels edges with lysine types or N-terminal (e.g., $L = \{\text{K6}, \text{K11}, \text{K27}, \text{K29}, \text{K33}, \text{K48}, \text{K63}, \text{M1}$
- $\tau : V \to T$ assigns node types, where $T = \{\texttt{Ub}, \texttt{SMAC}, \texttt{ABOC}, \texttt{Protein}, \texttt{HisTag}\}$.

Biochemical context: Each node represents a biological entity (e.g., ubiquitin monomer, protecting group, host protein), and each edge represents a covalent linkage through a specific lysine or N-terminus. The rooted directed tree structure ensures a unique root and exactly one path from the root to any other node, reflecting the hierarchical nature of polyubiquitin assembly.

Edge Direction and Biochemical Binding Semantics

Edges in G represent covalent conjugation events between a lysine residue of one ubiquitin and the C-terminal glycine (G76) of another. Specifically:

- An edge $(v_i, v_j) \in E$ means that the lysine site $\lambda((v_i, v_j))$ on v_i is bound to the C-terminus of v_j.
- The direction of the edge is from the **lysine-bearing parent** to the **C-terminus-bearing child**.
- Both v_i and v_j must be of type `Ub`.
- The edge label $\lambda((v_i, v_j)) \in L$ specifies the lysine through which the attachment occurs.
- Self-conjugation is disallowed: $v_i \neq v_j$.
- The global structure must remain a rooted directed tree to ensure acyclicity and unique paths.

  Note: By abuse of notation, we may write $\lambda(v_i, v_j)$ to mean $\lambda((v_i, v_j))$, where $(v_i, v_j) \in E$. We now formalize the typing constraint for edges:

$$\forall (v_i, v_j) \in E, \quad \tau(v_i) = \texttt{Ub},\ \tau(v_j) = \texttt{Ub},\ \lambda((v_i, v_j)) \in L$$

Biochemical context: This directionality mirrors the physical reality of ubiquitin linkage: a donor ubiquitin presents a lysine side chain, and an acceptor ubiquitin is attached via its terminal glycine. By enforcing this asymmetric, chemically meaningful edge direction, the graph model faithfully captures the topology and constraints of polyubiquitin chain formation.

Node Degrees

- Each node has in-degree ≤ 1 (at most one parent).
- Nodes of type `Ub` can have out-degree up to 8 (corresponding to 7 lysines + 1 N-terminal site).
- Nodes of type `SMAC` or `ABOC` are always terminal: $\deg^+(v) = 0$, and each can attach to only one outgoing lysine edge.

Biochemical context: Ubiquitin has a single reactive C-terminus (G76) that binds to lysines or N-terminal M1 of another Ub. Protecting groups like SMAC or ABOC do not have outgoing edges.

Biochemical context: Multiple protecting groups (e.g., 5 SMAC) can be present across different lysine sites on the same or different ubiquitin monomers. Each individual protecting group blocks conjugation at a single site.

Root Node

The root node of G is defined as the unique node from which all other nodes are reachable via a directed path. It satisfies the condition of having in-degree zero. This root node may be:

- A node of type `Protein` if a host substrate is conjugated to the first ubiquitin;
- A node of type `HisTag` if a terminally immobilized Ub is used as the anchor;
- Or a node of type `Ub` if the first ubiquitin is unbound at its C-terminus.

Formally, the root node $v_{\text{root}} \in V$ satisfies:

$$\deg^-(v_{\text{root}}) = 0 \quad \text{and} \quad \forall v \in V \setminus \{v_{\text{root}}\}, \exists \text{ path } v_{\text{root}} \rightsquigarrow v.$$

$$\tau(v_{\text{root}}) \in \{\texttt{Protein}, \texttt{HisTag}, \texttt{Ub}\}$$

Clarification: The root is always unique by construction of the tree. It must have in-degree zero and represents the biochemical anchor point of the polyubiquitin chain. If a protein substrate is attached to the first ubiquitin, the protein node becomes the root. If the ubiquitin is free or His-tagged, it itself is the root.

Set of Valid Graphs

Define $\mathcal{G}$ as the set of all constrained, rooted, labeled directed trees that meet the formal requirements:

$\mathcal{G} = \{G = (V, E, \lambda, \tau) \mid G$ is a rooted directed tree satisfying root constraints, node/edge typing, bounded

Biochemical context: This definition captures all valid polyubiquitin architectures, incorporating both structural and biochemical rules such as single-parent attachment, lysine/N-terminal linkage constraints, and support for protecting groups. All transformation functions are closed over this space.

2. Preorder Traversal and Labeling

Let $V^+ = \{v \in V \mid \exists v' \in V, (v, v') \in E$ or $\texttt{can\_have\_children}(v)\}$, where $\texttt{can\_have\_children}$(v) is true if v is of type `Ub` and has at least one unbound, unprotected lysine or N-terminal site.

Define a preorder function $\pi : V^+ \to \mathbb{N}$ such that:

- Preorder traversal begins at the root node.
- Children are visited in priority order: K63 $\prec$ K48 $\prec$ K33 $\prec$ K29 $\prec$ K11 $\prec$ K6 $\prec$ M1.
- This priority order is total and fixed throughout this formalism.

- Only nodes with children **or** with the potential to have children (e.g., ubiquitin monomers) are numbered.
- Nodes that are terminal and cannot accept children (e.g., `SMAC`, `ABOC`) are excluded from preorder indexing.

Since G is a rooted directed tree, the preorder traversal is uniquely defined and deterministic. Unlike general DAGs, trees have a unique path from the root to any node, enabling well-defined traversal order based on lysine priority.

Clarification: Because each ubiquitin node can have at most one outgoing edge per lysine type, there are no ties among children of the same lysine label. As a result, the preorder traversal is uniquely defined by the lysine priority order alone, with no need for arbitrary tie-breaking.

Biochemical context: Only Ub nodes are preordered since they can form linkages. Protecting groups never initiate further linkages and are unnumbered.

3. Edge Labeling

For each edge $e = (v_i, v_j) \in E$ with $v_i \in V^+$, its label is defined as:

$$\text{label}(e) = \pi(v_i)\text{-}\lambda(e)$$

Biochemical context: This ensures every linkage (e.g., "3-K48") is globally unique and encodes both its position in the chain and lysine specificity.

4. Summary of Constraints

- G is a finite rooted directed tree.
- Each node has in-degree ≤ 1.
- Only `Ub` nodes can have out-degree > 0, up to 8.
- Edge labels correspond to lysine/M1 types and must be unique per traversal index.
- Each node may have at most one outgoing edge per lysine type, regardless of protection status.
- Preorder numbering applies only to nodes capable of outgoing connections.
- Protecting groups are terminal and excluded from preorder.

Explanation: For each lysine site (e.g., K48, K63, etc.) on a given `Ub` node, only one outgoing edge with that label is permitted. This edge may represent either a conjugation to another ubiquitin (i.e., a linkage) or a protecting group (e.g., `SMAC`, `ABOC`) that occupies and

blocks that site. The presence of a protecting group still counts as occupying the site, meaning no additional edge with the same lysine label can exist. This rule ensures that each lysine behaves as a chemically unique site—consistent with biochemical constraints—and that the graph maintains a strict one-to-one correspondence between lysine labels and outgoing edges per node.

Biochemical context: These constraints model the structure and logic of polyubiquitin chains as found in biochemical systems, enabling systematic computation, simulation, and synthetic route planning.

5. Graph Transformations: Functions

Mathematical Validity and Closure

Error Conditions. All transformation functions in this framework are defined as non-destructive. If a transformation (e.g., deprotection or conjugation) finds no valid sites on the input graph(s), it returns the original graph unchanged as part of the output set. No transformation function returns an empty set. This behavior ensures that simulation layers always preserve input state unless a valid transformation occurs.

Definition. A **transformation function** is any function

$$f : \mathcal{G} \to \mathcal{P}(\mathcal{G})$$

that maps a valid polyubiquitin graph to a finite set of valid graphs and preserves all structural, typing, and biochemical constraints described in this formalism. This includes acyclicity, node and edge typing, bounded degrees, preorder traversal rules, and root uniqueness.

Let $\mathcal{G}$ denote the space of all valid polyubiquitin graphs, where each graph $G \in \mathcal{G}$ satisfies all constraints described in Sections 2–5. These include:

- Acyclicity: G is a rooted directed tree.
- Labeling: Edge labels $\lambda(e) \in L = \{\text{K6}, \text{K11}, \text{K27}, \text{K29}, \text{K33}, \text{K48}, \text{K63}, \text{M1}\}$.
- Typing: Node labels $\tau(v) \in T = \{\texttt{Ub}, \texttt{SMAC}, \texttt{ABOC}, \texttt{Protein}, \texttt{HisTag}\}$.
- Degree bounds: $\deg^-(v) \leq 1$ and $\deg^+(v) \leq 8$ for $\tau(v) = \texttt{Ub}$.
- Preorder indexing applies only to nodes $v \in V^+$ where $V^+ = \{v \in V \mid \texttt{can\_have\_children}(v)\}$.

Let $F : \mathcal{G} \to \mathcal{G}$ be a transformation function acting on graphs (e.g., deprotection or E2 conjugation).

Theorem 1 (Closure Under Transformation)**.** *For all $G \in \mathcal{G}$ and for all transformation functions F, we have:*

$$F(G) \in \mathcal{G}$$

That is, the output of any transformation is again a valid rooted directed tree.

Proof Sketch. Each F is constructed to only:

1. Remove terminal protecting groups (`SMAC`, `ABOC`) — which do not affect tree structure or preorder constraints.
2. Add new edges $\lambda(e)$ only between permissible pairs (Ub C-termini to Ub lysines).
3. Maintain the bounded in-degree and lysine-specific out-degree of each node.
4. Update preorder traversal π consistently by re-evaluating V^+ and assigning traversal indices in lysine priority order.

Since no operation introduces cycles or violates degree/type constraints, the output remains in $\mathcal{G}$. □

Biological context: This theorem formalizes that any valid biochemical operation—such as enzymatic conjugation or chemical deprotection—always yields another chemically and topologically valid polyubiquitin species. This ensures the integrity of synthetic route simulations and biological interpretations.

Closure Property Theorem

Theorem 1 (Closure under Transformation). Let $G = (V, E, \lambda, \tau)$ be a polyubiquitin graph satisfying all constraints defined in Sections 2–5. Then for any transformation function $f : G \to G'$ where f is a deprotection or E2 conjugation function, the resulting graph $G' = (V', E', \lambda', \tau')$ also satisfies all such constraints.

Proof Sketch: All transformation functions are defined to manipulate only allowable elements of the graph, e.g., removing `SMAC`/`ABOC` nodes or adding edges with valid lysine labels between appropriately typed nodes. Each function explicitly preserves the tree structure, bounded in-degree, lysine-specific out-degree constraints, and valid labeling and traversal rules. Therefore, G' remains a valid polyubiquitin graph.

Biochemical context: This theorem guarantees that the application of simulated biochemical operations (deprotection, conjugation) always yields another biologically valid polyubiquitin species, preserving chemical and topological validity.

We now define the allowed transformations on the polyubiquitin graph structure. These fall into two biologically grounded categories: deprotection and E2-mediated conjugation. Each function transforms one or more graphs into a new set of product graphs, while preserving the tree structure constraints and biochemical interpretations outlined previously.

Graph Union and Merging

To formally define E2 conjugation events between two polyubiquitin species, we must first define the graph union operation that combines two valid polyubiquitin graphs $G_1 = (V_1, E_1, \lambda_1, \tau_1)$ and $G_2 = (V_2, E_2, \lambda_2, \tau_2)$.

Assumption: The node sets V_1 and V_2 are disjoint. This is enforced by explicitly remapping node identifiers to ensure disjointness during the union operation.

Definition: The disjoint union of two graphs is given by:

$$G_1 \bigsqcup G_2 := (V_1 \cup V_2,\ E_1 \cup E_2,\ \lambda_1 \cup \lambda_2,\ \tau_1 \cup \tau_2)$$

Post-processing: After merging, node identifiers are reassigned to avoid conflicts, and a new preorder traversal π' is computed over the merged graph according to the traversal rules defined in Section 3.

Biochemical context: Merging graphs represents combining two molecular species during conjugation, e.g., a free monomer with a polyubiquitin chain. Reindexing preorder numbers ensures consistent global interpretation and edge labeling post-reaction.

5.1. Deprotection Functions

Deprotection functions remove nodes of type `SMAC` or `ABOC`, which act as blocking groups preventing lysine residues or C-termini from participating in conjugation. Three functions are defined:

- `SMAC_deprotect`: removes all `SMAC` nodes in the graph.
- `ABOC_deprotect`: removes all `ABOC` nodes in the graph.
- `Mock_deprotect`: a null function that returns the graph unchanged.

Biochemical context: Protecting groups are used to control the sequence and specificity of reactions. Deprotection corresponds to a chemical or enzymatic unmasking step that frees previously blocked reactive groups for further conjugation. The `Mock_deprotect` function serves as a placeholder or control operation.

5.2. E2 Conjugation Function

Let $\mathcal{P}(\mathcal{G})$ denote the powerset of valid graphs, i.e., the set of all finite subsets of $\mathcal{G}$.

E2 conjugation functions simulate the enzymatic linkage of a ubiquitin monomer (or polyUb species) to another at a specific lysine position. For a given lysine type $\theta \in L$, the function:

$$f_{\text{E2}}^{\theta} : \mathcal{P}(G) \to \mathcal{P}(G')$$

takes as input a set of graphs representing possible reaction species and returns all valid product graphs formed by conjugating all available lysines of type θ to all available free C-termini.

Formally, we define the following predicates:

$$\texttt{is\_free\_K}_\theta(v) := \nexists v' \in V : (v, v') \in E \wedge \lambda((v, v')) = \theta$$

$$\texttt{is\_free\_C}(v) := \deg^-(v) = 0$$

Definition: These predicates define whether a lysine site of type θ or a C-terminus is free and available for conjugation.

Let:

- $\mathcal{L}^\theta(G) = \{v \in V(G) \mid \texttt{is\_free\_K}_\theta(v)\}$ be the set of nodes with unconjugated lysine-θ groups.
- $\mathcal{C}(G) = \{v \in V(G) \mid \texttt{is\_free\_C}(v)\}$ be the set of nodes with free C-termini.

For each pair of graphs (G_1, G_2) from the input set, the function:

- Identifies all mappings $f : \mathcal{L}^\theta(G_1) \to \mathcal{C}(G_2)$.
- Each lysine is mapped to some root node, allowing reuse of roots (non-injective).
- Constructs a new graph G' by computing the disjoint union $G_1 \bigsqcup G_2$ and then inserting edges between compatible conjugation sites. The preorder traversal is recomputed after edge insertion.

Biochemical context: This function models the action of E2 enzymes that catalyze isopeptide bond formation between the C-terminus of one ubiquitin and a specific lysine on another. All free lysines of the given type react in a single step, enabling exhaustive enumeration of all valid conjugation products. For example, two K48 sites and two free C-termini result in four possible mappings.

Reaction Semantics and Cross-Product Behavior

The function f^θ_{E2} is defined to enumerate all structurally valid combinations between free lysines of type θ and all free C-termini across pairs of input graphs. This induces a **cross-product semantics**, meaning that for n available lysines and m free C-termini, up to $n \times m$ distinct conjugation mappings may be produced.

Biochemical context: While this combinatorial enumeration captures all chemically feasible conjugation topologies, it does not model the kinetic likelihood, enzyme preference, or selective constraints of actual E2 enzymes. Such considerations are handled in the simulation layer, which applies filtering to restrict biologically implausible outcomes. The graph-theoretical layer guarantees only structural validity, not biochemical frequency or feasibility.

5.3. Filtering Functions

Note on Preorder Traversal and Identity. Although preorder indices $\pi(v)$ are computed per graph and are not globally persistent identifiers, the simulation model ensures that each transformation adds at most one new ubiquitin node. Therefore, new additions can be unambiguously detected via differences in preorder sets and edge insertions, without requiring globally unique node IDs. This design simplifies the model while remaining correct under single-conjugation semantics.

Note: Filtering functions do not modify graphs. They serve only to classify or compare G_r and G_p and are not subject to the closure requirement.

We now define a set of filtering functions that analyze transformations by comparing the input (reactant) and output (product) graphs of a reaction. These functions enable classification of reaction outcomes and enzyme selection based on structural features of the conjugation event.

1. Identity Filter

Let $F_{\text{id}}(G_r, G_p)$ be a function that returns `True` if the reactant graph G_r and product graph G_p are isomorphic, i.e., no structural transformation has occurred:

$$F_{\text{id}}(G_r, G_p) = \begin{cases} \texttt{True} & \text{if } G_r \cong G_p \\ \texttt{False} & \text{otherwise} \end{cases}$$

Note on Isomorphism and Node Identity: In this framework, each node in the graph possesses a unique identifier (node ID) that distinguishes it from others, even if its biochemical type is the same (e.g., two `Ub` nodes are not interchangeable unless explicitly mapped). Preorder numbering π is computed per-graph based on traversal rules and is not an intrinsic identifier. Consequently, two isomorphic graphs may have structurally equivalent topologies but differ in their preorder assignments. Graph isomorphism used in filtering functions such as $F_{\text{id}}(G_r, G_p)$ therefore abstracts over specific preorder indices and focuses on structure, labels, and biochemical semantics.

Biochemical context: This filter detects null reactions, e.g., mock deprotection on an already unprotected graph, or failed conjugation steps.

2. K48 Enzyme Assignment Filter

Let $F_{\text{K48}}(G_r, G_p)$ be a function that returns the correct enzyme for a K48 conjugation based on the structural context of the conjugation site.

Rule:

- If the K48 site conjugated in G_p was on a ubiquitin in G_r that:
  - Was itself conjugated via its C-terminus to a K48 linkage, **and**
  - For all other lysines (i.e., $L \setminus \{\text{K48}\}$), there were no outgoing edges from any of its lysines in G_r:

$$\forall \ell \in L \setminus \{\text{K48}\},\ \nexists (v_k, v') \in E_r \text{ such that } \lambda((v_k, v')) = \ell$$

  then the enzyme is `Ube2g2`.

- Otherwise, the enzyme is `Ube2K`.

Mathematically: Given a set of new edges $\Delta E = E_p \setminus E_r$, identify $v_k \in V_r$ such that $(v_k, v_{\text{new}}) \in \Delta E$ with $\lambda((v_k, v_{\text{new}})) = \text{K48}$. Then:

$$F_{\text{K48}}(G_r, G_p) = \begin{cases} \texttt{Ube2g2} & \text{if } \tau(v_k) = \texttt{Ub},\ \lambda((v_p, v_k)) = \text{K48},\ \forall \ell \in L \setminus \{\text{K48}\},\ \nexists (v_k, v') \in E_r \text{ such that } \lambda((v \\ \texttt{Ube2K} & \text{otherwise} \end{cases}$$

Clarification: The enzyme assignment rule depends on both the parent and grandparent of the new ubiquitin node. Specifically, it evaluates:

1. Whether the parent ubiquitin (v_k) already has any lysine conjugations besides the new K48 linkage.
2. If not, it further examines the linkage through which v_k was conjugated to its own parent (v_p).
3. If v_k was conjugated to v_p via K48, the enzyme is classified as `Ube2g2`.
4. Otherwise (e.g., conjugated via K63 or any other lysine), the enzyme is `Ube2K`.

Biochemical context: This function classifies the type of K48 elongation reaction. `Ube2g2` catalyzes strictly homotypic K48-to-K48 elongation steps with no other branching from the intermediate node. `Ube2K` is required for all other types of K48-based modifications, including branch formation and K63-originated chains.

3. K63 Enzyme Assignment Filter

Let $F_{\text{K63}}(G_r, G_p)$ be a function that always returns the enzyme `Ubc13/Mms2` for any transformation involving K63 conjugation.

$$F_{\text{K63}}(G_r, G_p) = \texttt{Ubc13/Mms2}$$

Biochemical context: K63-linked polyubiquitin chains are exclusively synthesized by the Ubc13/Mms2 heterodimer, regardless of branching state or the architecture of the polyubiquitin chain.

4. New Ubiquitin Count Filter

Let $F_{\text{newUb}}(G_r, G_p)$ be a function that returns the number of newly added ubiquitin nodes with preorder numbers in the product graph G_p compared to the reactant graph G_r.

Define:

$$V_r^{\text{pre}} = \{v \in V(G_r) \mid v \in V^+ \text{ and } \tau(v) = \texttt{Ub}\}$$
$$V_p^{\text{pre}} = \{v \in V(G_p) \mid v \in V^+ \text{ and } \tau(v) = \texttt{Ub}\}$$

Then the function is given by:

$$F_{\text{newUb}}(G_r, G_p) = |V_p^{\text{pre}} \setminus V_r^{\text{pre}}|$$

Biochemical context: This function measures how many new ubiquitin monomers capable of further conjugation have been added to the product graph, excluding non-reactive protecting groups. It is essential for tracking chain growth during simulation.

5. Deubiquitination (DUB) Transformations

While not formally included in the transformation functions defined above, deubiquitinating enzymes (DUBs) are an essential class of biological modifiers that cleave isopeptide bonds between ubiquitin monomers. Conceptually, a DUB operation is defined as a function:

$$F_{\text{DUB}} : \mathcal{G} \to \mathcal{P}(\mathcal{G})$$

which acts on a valid polyubiquitin graph $G \in \mathcal{G}$ and returns a set of disconnected subgraphs $\{G_1, G_2, \ldots, G_k\} \subseteq \mathcal{G}$. Each resulting graph preserves the structural properties of polyubiquitin chains: bounded in-degree (≤ 1), valid lysine-based out-degree, typed nodes, and rooted tree structure. Thus, although the DUB transformation returns multiple graphs, each output remains an element of $\mathcal{G}$.

Biochemical context: DUBs catalyze the cleavage of specific ubiquitin linkages, effectively pruning or segmenting polyubiquitin chains. This operation allows regulatory recycling and editing of ubiquitin modifications. Although DUBs are not explicitly implemented in this modeling framework, they represent a key direction for future extensions.

Limitations and Extensions

While this model captures the structural and biochemical logic of polyubiquitin conjugation and deprotection with high precision, it currently treats E2-mediated conjugation reactions as strictly irreversible. This reflects their synthetic use in in vitro reconstruction and controlled simulation but differs from biological systems where many reactions are reversible.

Reversibility is handled in this framework via explicit deubiquitination functions (`DUBs`), which model the enzymatic cleavage of ubiquitin linkages. These transformations are defined separately from conjugation steps and preserve structural validity by returning a set of valid subgraphs in $\mathcal{G}$. This design choice preserves formal closure properties while allowing biologically realistic deconstruction of polyubiquitin chains.

Future work may integrate DUBs into simulation layers with kinetics or extend the formalism to support equilibrium modeling.

Scope of Enzyme Filtering. At present, the formalism includes enzyme assignment rules only for K48- and K63-linked conjugation events. These are biologically the most common and well-characterized linkages, and the current enzyme filters (`Ube2g2`, `Ube2K`, `Ubc13/Mms2`) reflect this focus. Other lysine types (e.g., K6, K11, K27, K29, K33, M1) are supported in the edge label space but do not yet have corresponding enzyme filters or selection logic. Future work will generalize the enzyme filtering function $F_{\text{enzyme}}(G_r, G_p)$ to handle additional linkage types and more complex reaction contexts.

Chain Size and Growth Constraints. This formalism imposes no upper bound on the total number of nodes, tree depth, or branching factor. Recursive transformations may yield arbitrarily large trees. This design is intentional: limitations such as steric hindrance, reaction kinetics, or resource constraints are assumed to be modeled in downstream simulation layers or accounted for in experimental contexts.