# Counting Linear Extensions of Modular Partial Orders

Matthieu Dien, Frederic Peschanski

# Counting Linear Extensions of Modular Partial Orders

1st Matthieu Dien
GREYC, *CNRS UMR6072*
*Université de Caen*
Caen, France
https://orcid.org/0000-0002-1825-0097

2nd Frederic Peschanski
LIP6, *CNRS UMR7606*
*Sorbonne Université*
Paris, France
https://orcid.org/0000-0002-4206-3283

***Abstract*—The counting of linear extensions is one of the prominent problems about partial orders. Unfortunately, the problem is computationally hard. In fact, relatively few counting procedures have been proposed in the literature. In this paper, we present such a counting procedure based on the modular decomposition of posets. This allows, first, to identify the series and parallel substructures, for which an efficient recursive counting procedure is known. Second, we propose a way to handle more complex prime substructures using an alternative decomposition scheme based on two complementary rules: the BIT-rule that can "consume" individual elements in a chain, and the SPLIT-rule that can "break" antichains. We develop a symbolic algorithm based on a multivariate integral formula solving the linear extension count than can be constructed along the decomposition. To discuss the complexity of this algorithm, we introduce a novel parameter, the BIT-width. We show that the algorithm is $\mathcal{O}(n^{w+1})$ where $w$ is the BIT-width of the input poset.**

***Index Terms*—Partial orders, Counting linear extensions, Modular decomposition**

## I. Introduction

Counting the number of linear extensions is one of the prominent problem about partial orders. There exists several applications of the problem in various areas such as scheduling or A.I., cf. [1] for references. It is also an important aspect in the quantitative analysis of concurrent processes [2], [3]. Unfortunately, the problem has been shown $\sharp P$-complete in [4], and in practice very few counting procedures have been proposed in the literature.

In this paper, we propose a counting procedure based on the modular decomposition of posets [5], [6]. This idea, already introduced in [7], allows to identify the series and parallel (SP) substructures, for which a straightforward (and efficient) recursive counting procedure is available [8]. While the approach is not new, there are very few attempts at coping with the so-called *prime* substructures of the modular decomposition trees, the "places" that concentrate the algorithmic difficulty of several problems about partial orders. As far as the counting of linear extensions is concerned, the difficulty is twofold: (1) the prime substructures can be arbirarily complex (it's "everything beyond SP"), and (2) the counting procedure must itself be modular. Indeed, it is not sufficient to count the number of linear extensions of the prime substructure in isolation, since it may contain or be contained in other structures.

In this paper, we propose to handle the prime node using a generic decomposition scheme based on a Poset decomposition scheme we introduce in [2]. This so-called BITS-decomposition is a simple graph rewriting system working on the cover graph of posets. It is based on two complementary rewrite rules: the BIT-rule that can "consume" isolated elements in a chain, and the SPLIT-rule that can "break" an elementary antichain. Most interestingly, a multivariate integral formula solving the linear extensions count can be constructed along the decomposition. An important fact is that if only the BIT-rule is used (or if the SPLIT-rule is used only sporadically), then the obtained formula is of a polynomial size (linear size if the SPLIT-rule remains unused). The challenge, then, is to introduce this decomposition scheme and counting procedure in the framework of the modular decomposition.

The paper is organised as follows. In Section II we describe two decomposition schemes for posets our counting procedure is based on: (1) the series-parallel structures, and (2) the BITS-decomposition. The associated counting schemes are also detailed. The modular decomposition of posets and its use for the counting of linear extensions is developed in Section III. The class of BIT-modular posets is then defined. The algorithmic and complexity issues are then discussed in Section IV. We show, for instance, that the algorithm we propose is polynomial for a large class of sparse posets, including for example the N-sparse posets [9] (which are BIT-modular) and the N-extensible posets [10] (which are "quasi" BIT-modular, i.e. BIT-modular with few splits). To further discuss the complexity of the counting algorithm, in Section V we introduce a novel parameter, the BIT-width. We show that the algorithm is $\mathcal{O}(n^{w+1})$ where $w$ is the BIT-width of the input poset.

### *Related work*

The problem of counting linear extensions of a general poset has been addressed from three perspectives in the literature: (1) results on the complexity of the problem, (2) exact solving solutions and (3) approximate solving. In this paper we are only concerned with exact solutions to the problem and so do not go into the details of approximate solutions

As a starting point, the complexity of the problem as been proved $\sharp P$-complete in [4] and also in the case of

height-2 posets and incidence posets, in [11]. In the context of parameterized complexity, the problem has been shown intractable w.r.t. the treewidth of the comparability graph in [12]. However the problem is also shown FPT w.r.t. to the treewidth of the incomparability graph.

From the algorithmic perspectives, the problem is known to be polynomial for trees and Series-Parallel posets [8] and mobile posets [13]. Other results exist when some parameters are bounded: a $\mathcal{O}(n^{k^2})$ algorithm for posets of modular-width[1] $k$ [7], an $\mathcal{O}(n^{t+3})$ algorithm for posets whose comparability graph has treewidth $t$ [14] or an $\mathcal{O}(w \cdot n^w)$ algorithm for posets of width[2] $w$. An notable alternative counting approach is proposed in [15] which is an enumeration algorithm for linear extensions with constant delay.

## II. Two decomposition schemes for partial orders

In this section, we discuss the informal notion of *decomposability* of partial orders, and provides two complementary illustrations when considering the problem of counting linear extensions. As a reminder, a poset (or *partially ordered set*) $P = (X, <_P)$ is an element set $X$ and an ordering relation $<_P$ over $X$ such that $\leq_P = <_P \cup id_X$ is reflexive, transitive and antisymmetric[3]. It is *partial* in that there may be elements $x, y$ in $X$, said *incomparable*, such that neither $x <_P y$ nor $y <_P x$. The size of Poset $P$, denoted by $|P|$, is simply the cardinal of its carrier set $|X|$. Without loss of generality, and to simplify some constructions, in this paper we consider a poset $P$ to be implicitly completed by two distinguished elements $\bot$ (bottom) and $\top$ (top) such that $\bot <_P x <_P \top$ for any $x \in X$. When depicting a poset $P$ we omit the bottom and top elements. We also distinguish the *unit* poset $\mathbb{1} = (\{x\}, \{(\bot, x), (x, \top), (\bot, \top)\})$ for some element $x$. Given a poset $P = (X, <_P)$ and a subset $Y \subseteq X$, we define the *restriction* of $P$ to $Y$ as $P[Y] = (Y, \{(x, y) \in Y^2 \mid x <_P y\})$. We also define the lifted union of two posets $P = (X_P, <_P)$ and $Q = (X_Q, <_Q)$ as: $P \uplus Q = (X_P \cup X_Q, \{(x, y) \mid x <_P y \text{ or } x <_Q y\})$ provided $P$ and $Q$ are compatible, *i.e.* there is no $(x, y)$ such that $x <_P y$ and $y <_Q x$. Figure 1 shows four examples of posets using the usual diagrammatic representation (relations are oriented donward).

A *linear extension* $\lambda = (X, \prec_\lambda)$ of a poset $P$ is a *total* ordering over $X^2$ such that $x <_P y$ only if $x \prec_\lambda y$. The linear extensions count $\sharp\ell e(P)$ is the number of distinct linear extensions of the poset $P$.

For a given problem – in our case the linear extensions count – the objective is to be able to construct a solution through the decomposition into smaller sub-problems. This requires to fulfill two complementary requirements:

- first, to identify subclasses of posets such that the problem becomes solvable, or at least "approachable",
- second, be able to combine (the solutions for) such subclasses so that the problem can be solved (or "approached") for larger classes

In this section we focus on the first requirement. From this point of view, the *series-parallel* (SP) posets play a significant role since in this class many "hard" problem can be solved in polynomial time, the linear extensions count being no exception.

**Definition 1** (Series-Parallel posets)**.** Let $P = (X_P, <_P)$ and $Q = (X_Q, <_Q)$ two posets, assuming $X_P \cap X_Q = \{\top, \bot\}$, i.e disjointness. The series and parallel[4] operators are, respectively: $\begin{cases} P \odot Q = (X_P \cup X_Q, <_P \cup <_Q \cup (X_P \times X_Q)) \\ P \parallel Q = (X_P \cup X_Q, <_P \cup <_Q) \end{cases}$
A (finite) poset is Series-Parallel (SP) if it can be obtained uniquely by (finite) compositions of series or parallel operators from singleton sets. ⌟

For example, the poset $P_1$ of Figure 1 is SP, indeed:

$$P_1 = \{a\} \odot ((\{b\} \parallel \{c\}) \odot ((\{d\} \odot \{g\}) \parallel \{e\}) \parallel \{f\}) \odot \{h\}$$

For this class of posets, the linear extensions count can be solved in polynomial time, using the following formulas.

**Theorem 1** (cf. [8])**.** $\sharp\ell e(P \odot Q) = \sharp\ell e(P) \cdot \sharp\ell e(Q)$
*and* $le(P \parallel Q) = \binom{|P|+|Q|}{|P|} \cdot \sharp\ell e(P) \cdot \sharp\ell e(Q)$

Most importantly, the algorithmic folmulas decompose as the SP poset do, which perfectly illustrates the notion of a *recursively* decomposable class of posets.

We introduce in [2] an alternative – and complementary – decomposition scheme based on (elementary) *graph rewriting principles*. More precisely, this so called BITS-decomposition works on the *cover graph* of posets.

**Definition 2** (Transitive reduction and cover graph)**.** Let $P = (X, <_P)$ be a poset. The transitive reduction relation of $P$ is $\prec_P = \{(x, y) \in X^2 \mid x <_P y \text{ and } \nexists u \in X \text{ such that } x <_P u <_P y\}$. The cover graph of $P$ is the directed acyclic (and intransitive) graph (DAG) with vertex set $X$ and edge relation $\prec_P$. Following this terminology, we say that $x$ *covers* $y$ whenever $x \prec_P y$. ⌟

A graph rewriting system is based on two basic principles: (1) *patterns* that identify the parts of the graph concerned by the rewriting, and (2) *effects* explaining how the matched parts should be transformed. In the case of the BITS-decomposition, the patterns and effects are rather straightforward.

**Definition 3** (BITS-decomposition)**.** Let $P = (X, <_P)$ be a poset. We denote by $P[x \prec y \prec z]$ the poset $P$ identifying a $y$ such that there is $x, z$ and $x$ covers $y$ and $y$ covers $z$, and there is no element $u$ in $X$ distinct from $x$ and $z$ such that $u \prec_P y$ or $y \prec_P u$. We denote by $P[x|y]$ the poset $P$ in which at least two elements $x$ and $y$ are incomparable, i.e. such that neither $x <_P y$ nor $y <_P x$. The BITS-decomposition of poset $P$

[1] The modular-width of a poset is the maximum degree of the prime vertices in its modular decomposition.

[2] The width of a poset is the length of its longest antichain.

[3] In the context of linear extensions, it is common to take the point of view of strict partial orders.

[4] The parallel construction $P \parallel Q$ is of course the same as $P \uplus Q$, but the former notation insists on the disjointness condition.

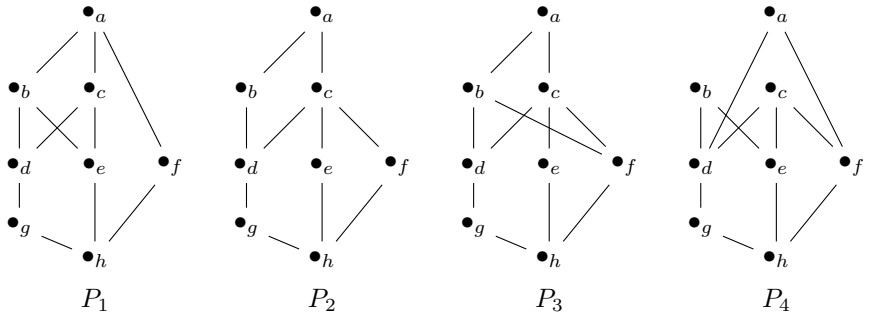



Fig. 1. Diagrammatic representation of four posets of size 8

consists in the repeated and non-deterministic application the following rewrite rules:

BIT: $P[x \prec y \prec z] \Rightarrow P[X \setminus \{y\}]$

SPLIT: $P[x|y] \Rightarrow P_{\lhd x \prec y}$ with $P_{\lhd u \prec v} = P \Cup (X, \{(u,v)\})$

We denote by $P \Rightarrow^* P'$ the decomposition of $P$ into $P'$ by an arbitrary number of rewrites. ⌟

In graphical terms, the element $y$ of pattern $P[x \prec y \prec z]$ corresponds to a vertex with input and output arity exactly 1 in the cover graph of $P$. In a similar way the nodes $x, y$ in a pattern $P[x|y]$ corresponds to arbitrary vertices with no edges between them.

**Lemma 1.** *Given a poset $P = (X, <_P)$, then either there is a $y$ such that $P[x \prec y \prec z]$ for some $x,\ z \in X$ ; otherwise there is a pair $(x, y) \in X^2$ such that $P[x|y]$; or $P = \mathbb{1}$.*

*Proof.* We proceed by contradiction. Suppose that there is a poset $P = (X, <_P)$ such that none of the three patterns hold. By convention there are at least two elements $\bot, \top$ such that $\bot <_P \top$. If there is no other element in the ordering then trivially $P = \mathbb{1}$. So suppose there are $n > 0$ elements $y_1, \ldots, y_n$ distinct from $\bot$ and $\top$ with relations in $<_P$ (if only wrt. $\bot$ and $\top$). If $n = 1$ then by convention $\bot <_P y_1 <_P \top$ thus $P[\bot \prec y_1 \prec \top]$ applies. If $n > 1$ we make the hypothesis that all elements are comparable, since otherwise the pattern $P[x|y]$ would apply. For any $y_i,\ 1 \leq i \leq n$ the pattern $P[x \prec y_i \prec z]$ applies for some $x,\ z \in X$ because $x \prec_P y_i \prec_P z$ is a total order given the fact that the elements cannot be incomparable. □

**Lemma 2.** *Given a poset $P$, then $P \Rightarrow^* \mathbb{1}$.*

*Proof.* By the previous lemma we know that the BITS-patterns are always matching. Each time the BIT-rule applies an element of the poset $P$ is removed. Hence, if this is systematically the only applicable rule ultimately the pattern $\mathbb{1}$ is reached. If otherwise the SPLIT rule is matched, a new relation between *existing* elements $x,\ y$ is added. This rewrite disables pattern $P[x|y]$ since $x$ and $y$ are now comparable. Put in other terms, all incomparable pairs can be made comparable by repeated application of the SPLIT rules. Ultimately the poset shall only contain chains, which can be systematically eliminated using the BIT-rule. □

The BITS rules can thus decompose arbitrary posets, however an interesting poset subclass naturally emerges from the proposed rewriting system.

**Definition 4.** A poset is said BIT-decomposable iff it reduces to $\mathbb{1}$ by the sole, and repeated, application of the BIT-rule.

As detailed in [2], together with the decomposition itself, we can construct a multivariate integral formula for the linear extensions count.

**Definition 5.** Let $P = (X, <_P)$ be a poset. We denote by $\Psi(P)$ a multivariate integral formula such that $P \xrightarrow{1} \Psi(P)$ is inferred from the following rules:

$$\frac{P \setminus \{y\} \xrightarrow{\int_{\alpha(x)}^{\alpha(z)} \Psi . \mathrm{d}y} \Psi'}{P[x \prec y \prec z] \xrightarrow{\Psi} \Psi'}\ \textsf{BIT} \qquad \frac{}{\mathbb{1} \xrightarrow{\Psi} \Psi}\ \textsf{END}$$

$$\frac{P_{\lhd x \prec y} \xrightarrow{\Psi} \Psi_1 \quad P_{\lhd y \prec x} \xrightarrow{\Psi} \Psi_2}{P[x \mid y] \xrightarrow{\Psi} \Psi_1 + \Psi_2}\ \textsf{SPLIT}$$

with $P \setminus \{y\} = P[X \setminus \{y\}]$ and $P_{\lhd u \prec v} = P \Cup (X, \{(u,v)\})$ and $\alpha(\bot) = 0$, $\alpha(\top) = 1$, and $\alpha(x) = x$ otherwise. ⌟

**Theorem 2.** *Let $P$ be a poset. Then $\sharp \ell e(P) = |X|! \cdot \Psi(P)$.*

*Proof idea.* The basic idea is to consider the embedding of the poset $P$ into a polytope $\mathcal{O}(P) = \{v \in [0,1]^X \mid v_x < v_y \text{ whenever } x <_P y\}$ called the *order polytope*, exploiting the fact that $\sharp \ell e(P) = |X|! \cdot Vol(\mathcal{O}(P))$ (cf. [16]).

A detailed proof is given in [2]. □

In Figure 1, only the poset $P_2$ is BIT-decomposable. The nodes with input arities 1 are $\{b, e, f, g\}$ that can be eliminated in an arbitrary order. A possible decomposition is as follows:

$$\Psi(P_2) = \int_0^1 \int_0^h \int_a^h \int_c^h \int_d^h \int_c^h \int_a^d \int_c^h 1 \; . \mathrm{d}f . \mathrm{d}b . \mathrm{d}e . \mathrm{d}g . \mathrm{d}d . \mathrm{d}c . \mathrm{d}a . \mathrm{d}h$$

This gives: $\sharp \ell e(P_2) = 8! \cdot \Psi(P_2) = \dfrac{8!}{1260} = 32.$

## III. Modular counting of linear extensions

The previous section presented two decomposable classes of posets with very distinct and complementary characteristics. In a way, the series-parallel (SP) operators are applied "from the outside" by matching the largest possible series and parallel substructures. The BIT-rule, in contrast, operate "from the inside" by eliminating individual nodes. Moreover, the two schemes are tightly connected to linear extensions counting principles. It seems thus natural to (try to) integrate these principles one way or another. This is the question we adress in this section, and our starting point is the well-known principle of *modular decomposition* (the historical reference appears to be [5] but a useful, modern reference is [6]). While the principle applies on graphs in general, in this paper we only consider the case of posets.

**Definition 6.** Let $P = (X, <_P)$ a poset. A module of $P$ is a subset $M$ of $X$ such that for any $x, x' \in M$ and $y \in X \setminus M$, $x <_P y$ iff $x' <_P y$ and $y <_P x$ iff $y <_P x'$. A module is trivial if it is a singleton. A module $M$ is said strong if for any other module $N$ of $P$ either $N \subset M$, or $M \subset N$ or $M \cap N = \emptyset$. It is said maximally strong no other strong module contains it. Given a partition of $X$ into a set of modules $\mathcal{M}$, the quotient order of $P$ is $P_{/\mathcal{M}} = (\mathcal{M}, <_{\mathcal{M}})$ such that $M_1 <_{\mathcal{M}} M_2$, $M_1, M_2 \in \mathcal{M}$ if and only if there exist $x \in M_1$, $y \in M_2$ and $x <_P y$. ⌟

There is an exponential number of modular partitions of a poset $P$, however there exists a canonical case.

**Theorem 3** (cf. [5])**.** *The partition of maximally strong modules of a poset $P$ is unique.*

**Definition 7** (Modular decomposition, cf. [6])**.**

The modular decomposition of a poset $P = (X, <_P)$ consists in generating a maximal modular partition $\mathcal{M}$ of $P$. Then, for each module $M \in \mathcal{M}$, the maximal modular partition of $P[M]$ is generated, and so on. The decomposition can be arranged in a tree such that each node is associated to a maximal partition and arranged following the inclusion relation. The leaf nodes are exactly the trivial modules $\{x\}$ for every $x \in X$, and the internal nodes correspond to non-trivial (and thus decomposed) modules. An internal node can be linear, complete or prime. A linear node denotes a series construction, and is of the form $\mathbf{L}[t_1, ..., t_n]$ with the $t_i$'s as subtrees. A complete node denotes a parallel construction, and is or the form $\mathbf{C}[t_1, ..., t_n]$. Finally if the underlying poset as an alternative structure, it is said prime and is of the form $\mathbf{P}_Q[t_1, ..., t_n]$ where the poset $Q$ encodes the relationship between the sub-components. The modular decomposition tree $t$ of $P$ is such that $\mathcal{PO}(t) = P$ with:

$$\left[\begin{array}{l} \mathcal{PO}(x) = (\{x\}, \emptyset) \\ \mathcal{PO}(\mathbf{L}[t_1, ..., t_n]) = \mathcal{PO}(t_1) \odot \cdots \odot \mathcal{PO}(t_n) \\ \mathcal{PO}(\mathbf{C}[t_1, ..., t_n]) = \mathcal{PO}(t_1) \parallel \cdots \parallel \mathcal{PO}(t_n) \\ \mathcal{PO}(\mathbf{P}_Q[t_1, ..., t_n]) = Q \Cup (\mathcal{PO}(t_1) \parallel \cdots \parallel \mathcal{PO}(t_n)) \end{array}\right.$$

⌟

The modular decomposition of poset $P_2$ from Figure 1 is depicted in Figure 2 (left). The decomposition tree contains a single prime node, and is otherwise SP. Efficient algorithms exist for constructing such decomposition trees. The initial step, for most algorithms, is to compute the directed acyclic graph corresponding to the transitive closure of the input poset. This step can be performed in $O(n)$. Then quadratic and sub-quadratic algorithms have been proposed in the literature to construct the decomposition tree of a directed graph, cf.[6] for details.

**Proposition 1** (Series-parallel counting)**.**

*Let $P = (X, <_P)$ a series-parallel poset and $t$ its modular decomposition. Then $\sharp\ell e(P) = \sharp\ell e(t)$ with:*

$$\left[\begin{array}{l} \sharp\ell e(x) = 1,\ x \in X \\ \sharp\ell e(\mathbf{L}[t_1, ..., t_n]) = \prod_i \sharp\ell e(t_i) \\ \sharp\ell e(\mathbf{C}[t_1, ..., t_n]) = \binom{\sum_i |t_i|}{|t_1|, ..., |t_n|} \cdot \prod_i \sharp\ell e(t_i) \end{array}\right.$$

*Proof.* This is a direct consequence of the recursive construction of Definition 7, applying the formulas of Theorem 1. □

Evaluating the contribution of the prime nodes to the linear extensions count, in a modular way, relies on the following theorem.

**Theorem 4** (Prime counting)**.** *Let $P = (X, <_P)$ be a partial order with modular decomposition tree $t$ such that its root is a prime node labeled by $Q$* i.e. *$P = \mathbf{P}_Q[t_1, ..., t_n]$. Then:*

$$\sharp\ell e(P) = \sharp\ell e(Q \Cup \bigcup_i lin(\mathcal{PO}(t_i))) \cdot \prod_i \sharp\ell e(\mathcal{PO}(t_i))$$

*where $lin(P)$ is an arbitrary total order on $X$, of size $|P|$.* ⌟

In Figure 2 (right) is depicted the linearization of the poset corresponding to the unique prime node in the decomposition of $P_2$ (from Figure 1).

*Proof.* Proof details are proposed in appendix. A similar result and alternative proof is proposed in [7]. □

**Corollary 1.** $\sharp\ell e(\mathbf{P}_Q[t_1, ..., t_n]) = (\sum_i |t_i|)! \cdot \Psi(Q \cup \bigcup_i lin(\mathcal{PO}(t_i))) \cdot \prod_i \sharp\ell e(t_i)$.

*Proof.* This is a direct consequence of applying Theorem 2 in the context of prime counting. □

**Definition 8.** A BIT-modular poset is such that all prime nodes of its modular decomposition are BIT-decomposable.

In Figure 1, the posets $P_1$, $P_2$ and $P_3$ are examples of BIT-modular posets. Note that if $P_2$ is fully BIT-decomposable, this is not the case of $P_3$. Comparatively, the poset $P_4$ is not BIT-modular because it requires at least an application of the SPLIT-rule.

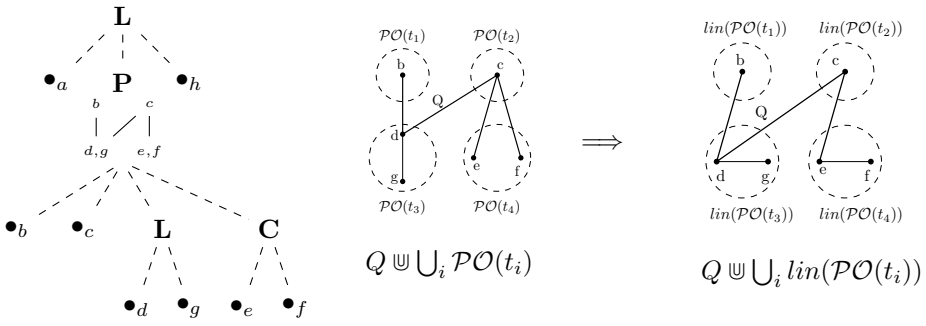



Fig. 2. Modular decomposition of poset $P_2$ (left), and linearization of its prime node (right).

## IV. Algorithmic considerations

From the mathematical results of the previous sections, especially Proposition 1 and Theorem 4, we proposed Algorithm 1 (below) to solve the linear extensions count.

**Algorithm 1** Modular counting of linear extensions

**function** LECount($t$ the decomposition tree of a poset $P$)
  **if** $t = x$ **then return** $1$
  **else if** $t = \mathbf{L}[t_1, ..., t_n]$ **then return** $\prod_i \text{LECount}(t_i)$
  **else if** $t = \mathbf{C}[t_1, ..., t_n]$ **then**
    **return** $\binom{\sum_i |t_i|}{|t_1|,...,|t_n|} \cdot \prod_i \text{LECount}(t_i)$
  **else** $t = \mathbf{P}_Q[t_1, ..., t_n]$
    $\mathring{P} \leftarrow Q \Cup \bigcup_i lin(\mathcal{PO}(t_i))$
    **return** $\Psi(\mathring{P}) \cdot |P|! \cdot \prod_i \text{LECount}(t_i)$

**Theorem 5.** *Given the modular decomposition tree $t$ of a poset $P$, Algorithm 1 counts the number of linear extensions of $P$.*

*Proof.* This is a direct consequence of Proposition 1 and Theorem 4. □

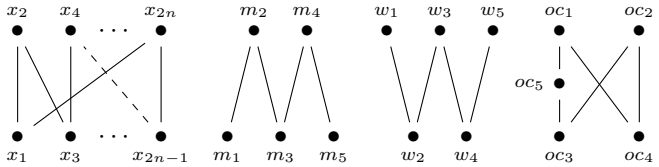



Fig. 3. Left to right: a $n$-Crown, a $M$, a $W$, and an $OC_5$.

Obviously, Algorithm 1 has a super-exponential complexity in the worst-case. If we consider, for example, the $n$-Crown poset $C = (X, <_C)$ (depicted in Figure 3) where $X = \{1, \ldots, 2n\}$ and the relations $x_1 >_C x_2 <_C x_3 >_C \ldots$ and $x_1 >_C x_{2n}$. The poset $C$ is prime, and it is a simple observation that at least $n$ applications of SPLIT rules are needed to decompose the order. Consequently, the associated counting formula is a sum of $2^n$ integral formulas.

As a counterpoint, we are looking for classes of posets for which the algorithm becomes tractable. Series-parallel posets is an obvious starting point.

**Corollary 2.** *Given the modular decomposition tree $t$ of a Series-Parallel poset $P$, Algorithm 1 runs in $\mathcal{O}(|P|)$ arithmetical operations complexity.*

*Proof.* Considering the factorial numbers up to $|P|$ precomputed, this is a direct consequence of the linear size of the modular decomposition tree (see [6] for details). □

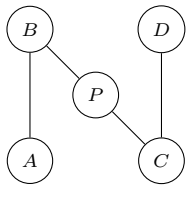
Unsuprisingly, the complexity of the problem comes from the prime modules. Thus, when the prime modules are constrained to live in a given set we can generalize the previous result. Various classes of so-called "quasi" Series-parallel posets have been studied in the literature. An important example is that of *N-sparse* posets, the order-theoretic counterpart of the $P_4$-reducible graphs. Similarly, the $P_4$-extendible graphs have their counterpart of the N-extendible partial orders.

**Definition 9.** from [9] and [17]. $P_4$ is the path graph of length 3. A graph $G$ is:

- $P_4$-reducible if every vertex $v$ of $G$ belongs to at most one induced $P_4$
- $P_4$-extendible if for any subset $W$ of $G$, which contains some $P_4$, there exists at most one $x$ (outside of $W$), such that $x$ forms another $P_4$ with vertices of $W$.

A poset is said N-sparse (resp. N-extendible) if its comparibility graph is $P_4$-reducible (resp. $P_4$-extendible).

A useful characterization is based on the modular decomposition perspective and an elementary poset construction named the *spider product*, denoted by $P \bowtie A, B, C, D$, which can be described as follows:




- When $A, B, C, D$ are singletons, $P \bowtie A, B, C, D$ is a simple spider product.
- When $A, B, C, D$ are of size 1 except one of size 2, $P \bowtie A, B, C, D$ is an extended spider product.

(cf. e.g. [18] for a more formal definition)

**Theorem 6** (from [18] and [19])**.** *A poset is N-sparse if and only if the prime vertices of its modular decomposition are simple spider products $P \bowtie A, B, C, D$ where $P$ is the tree of a N-sparse poset.*

*A poset is N-extendible if and only if the prime vertices of its modular decomposition are:*

- $M, W, OC_5$ *whose leafs are singletons (see Figure 3),*
- *extended spider products* $P \bowtie A, B, C, D$ *where* $P$ *is an N-extended poset.*

**Theorem 7.** *The number of linear extensions of* $N$*-extendible posets (thus,* $N$*-sparse posets too) can be computed in* $\mathcal{O}(n)$ *arithmetical operations.*

*proof (sketch).* First, let deals with the finite patterns $M = m_1 > m_2 < m_3 > m_4 < m_5$, $W = w_1 < w_2 > w_3 < w_4 > w_5$ and $OC_5 = (oc_1 \parallel oc_2) \odot (oc_3 \parallel oc_4) \cup \{oc_1 < oc_5 < oc_3\}$. Because they are of a small size, their number of linear extensions can be precomputed : $\sharp\ell e(M) = \sharp\ell e(W) = 16$ and $\sharp\ell e(OC_5) = 14$. Then it remains to deal with the extended spider product.

Let $P = x_1 < \ldots < x_k$ be a linear order of size $k$ and $Q$ be the poset $P \bowtie A, B, C, D$.

Actually, we can avoid the usage of BITS rules to count the number of linear extensions of $Q$, as follows.

- there are $k+2$ linear extensions with $D < B$ (the number of possible places for $A$)
- there are $k + 2 - 1$ linear extensions with $A > C$ (the number of possible places for $D$ minus one where $D < B$, which is already counted by the previous case)
- there remain $(k+1)(k+2)$ linear extensions of the other kind.

At the end the total count is $(k+2)^2+(k+1)$. Note that these computations are sufficient to deal with the case of N-sparse posets.

Now using the same kind of enumeration, we can count the number of linear extensions of the extended spider product *i.e.* when one of the posets $A, B, C$ or $D$ is a chain or an antichain of size 2:

- when $A$ or $D$ is an antichain of size 2: $\sharp\ell e(Q) = (k+3)^3 - 3 \cdot (k+3)$
- when $A$ or $D$ is a chain of size 2: $\sharp\ell e(Q) = \frac{k+1}{2}(k+4)^2 + 1$
- when $B$ or $C$ is an antichain of size 2: $\sharp\ell e(Q) = 2\left[(k+2)^2 + (2k+3)\right]$
- when $B$ or $C$ is a chain of size 2: $\sharp\ell e(Q) = (k+2)^2 + (2k+3)$

Modifying the `else` clause of Algorithm 1 with those formulas, leads to the linear complexity. □

Finally, the essence of this theorem is that the primes modules are constrained to be finite and belonging to a finite set. Thus, we can easily generalize the result to poset of bounded modular-width.

**Lemma 3.** *Let* $P = \mathbf{P}_Q[t_1, ..., t_k]$ *be a prime poset of size* $n$ *where the* $t_i$ *are total orders. Then, there exists an order of applications of the BITS rules such that the computation of* $\sharp\ell e(P)$ *is carried with* $\mathcal{O}(n^k)$ *arithmetical operations.*

*Proof.* Let us proceed by steps:

1) Reduce the $lin(\mathcal{PO}(t_i))$ components one by one :
   - because $lin(\mathcal{PO}(t_i))$ is a linear order we can apply the BIT rule until it is totally reduced : with $|t_i| = n_i$ and $lin(\mathcal{PO}(t_i)) = x_1 < ... < x_{n_i}$, the BITS formula computed is[5] $\int_{\alpha(\perp_i)}^{\alpha(\top_i)} \int_{\alpha(\perp_i)}^{x_{n_i}} \cdots \int_{\alpha(\perp_i)}^{x_2} 1\, dx_1 1 \ldots dx_{n_i-1}\, dx_{n_i} = \frac{(\alpha(\top_i)-\alpha(\perp_i))^{n_i}}{n_i!}$
   - Then, after reducing all the $t_i$'s the formula is exactly $\psi = \prod_{i=1}^{i=k} \frac{(\alpha(\top_i)-\alpha(\perp_i))^{n_i}}{n_i!}$ *i.e.* an homogeneous polynomial of total degree $n$ with $k$ variables
2) Now, it remains to reduce $Q$:
   a) if Q is not BIT-decomposable, then, at most a linear number (in $k$) of SPLIT rules must be applied to obtain a BIT-decomposable poset
   b) Q is BIT-decomposable: then integrating the polynomial $\psi$ $k$ times ends the process

The first step builds an homogeneous polynomial with, at most, $k$ variables of total degree $n$. By elementary enumeration, that polynomial has, at most, $\binom{n+k-1}{k} = \mathcal{O}(n^k)$ coefficients.

For the second step, if some SPLIT rules must be applied, then the formula to integrate is a sum of at most $k!$ homogeneous polynomials with $k$ variables and degree at most $n$.[6]

It remains to compute the complexity of integrating $k$ times an homogeneous polynomial with $k$ variables and total degree $n$. The integration of a polynomial can be performed by iterating over its coefficients and so, will double the number of coefficients in the worst case. So, the overall operation will cost $\mathcal{O}(2^k\, n^k)$ operations.

Because the $k$ is constant, the total complexity of the whole computations is $\mathcal{O}(n^k)$. □

**Theorem 8.** *Let* $t$ *be the modular decomposition tree of a poset* $P$*. Let* $\{Q_1, ..., Q_\ell\}$ *be the set of posets labeling the prime vertices of* $t$ *and* $k = \max_i |Q_i|$ *(the size of the biggest prime module of* $P$*, also called the* modular-width*). Then, Algorithm 1 counts the number of linear extensions of* $P$ *with complexity* $\mathcal{O}\left(n^{k+1}\right)$ *in terms of arithmetical operations.*

*Proof.* The modular decomposition tree $t$ has $\mathcal{O}(n)$ vertices and so, a linear number of primes vertices of size $k$. By Lemma 3, the number of linear extensions of such vertices can be computed with $\mathcal{O}\left(n^k\right)$ arithmetical operations. So the overall complexity is $\mathcal{O}\left(n^{k+1}\right)$. □

That result largely improves the one of [7] which was $\mathcal{O}(n^{k^2})$.

## V. The BIT-width parameter

For final section of the paper, we introduce a new parameter that is tightly connected to the BITS-decomposition: the *BIT-width* of a poset. Our starting point is the notion of a *BIT-order*,

[5]Here, $\alpha(\perp_i)$ and $\alpha(\top_i)$ correspond to the bounds of $\mathcal{PO}(t_i)$ in $Q$. For example, if $\forall x \in \mathcal{PO}(t_i), \forall y \in \mathcal{PO}(t_j), x < y$ in $Q$ then $\alpha(\perp_j) = x_{n_i}$ and $\alpha(\top_i) = y_1$.

[6]The worst case being unfolding a free order with the SPLIT rule.

i.e. a recording of the order in which the elements of a poset are consumed through applications of the BIT-rule.

**Definition 10.** Let $P$ be a BIT-decomposable poset. A BIT-order $\beta$ of decomposition of $P$ is a sequence of applications of the BIT-rule, identified by the removed elements, that reduces $P$ to $\mathbb{1}$.

To a BIT-order $\beta$ we associate the sequence of polynomials $\boldsymbol{Pol}_\beta$ corresponding to the integrals computed along the decomposition. ⌟

Rolling back into the example of poset $P_2$ (for which a counting formula is given after Theorem 2), using the same BIT-order $\beta = (f, b, e, g, d, c, a, h)$, we obtain the following sequence of polynomials:
$\boldsymbol{Pol}_\beta =$
$(h-c, (d-a)(h-c), (d-a)(h-c)^2, (d-a)(h-c)^2(h-d),$
$-\frac{1}{6}\left(3\,ac^2-2\,c^3+3\,ah^2-h^3-3\left(2\,ac-c^2\right)h\right)(c-h)^2,$
$\frac{2}{45}\,a^6-\frac{4}{15}\,a^5h+\frac{2}{3}\,a^4h^2-\frac{8}{9}\,a^3h^3+\frac{2}{3}\,a^2h^4-\frac{4}{15}\,ah^5+\frac{2}{45}\,h^6, \frac{2}{315}\,h^7, \frac{1}{1260})$

**Definition 11.** Let $\sharp var : \mathbb{K}[X_1, ..., X_n] \mapsto \mathbb{N}$ be the function associating to a polynomial its number of variables. The BIT-width $w(\beta)$ of a BIT-order $\beta$ is the maximal number of variables involved in a polynomial of $\beta$:

$$w(\beta) = \max_{pol \in \boldsymbol{Pol}_\beta} \sharp var(pol).$$

The BIT-width $w(P)$ of a BIT-decomposable poset $P$ is the minimum of the BIT-width of its BIT-order:

$$w(P) = \min_{\beta \text{ a BIT-order of } P} w(\beta) \quad ⌟$$

Going back to the example, $w(\beta) = 4$ but one can check that $w(P_2) = 2$ which is realized (non uniquely) by the BIT-order $(b, a, d, g, e, f, c, h)$.

**Theorem 9.** *Let $P$ be a BIT-decomposable poset of size $n$ and BIT-width $w$ and let $\beta$ be one of its BIT-order of BIT-width $w$. Then, the evaluation of the BIT-formula $\Psi(P)$ has a complexity of $\mathcal{O}(n^{w+1})$.*

*Proof.* Actually, the theorem is just a refinement of Lemma 3. It is sufficient to see that the polynomials involved in the computations of $\Psi(P)$ are homogeneous polynomials of total degree at most $n$ with at most $w$ variables and thus are composed of at most $\mathcal{O}(n^w)$ monomials. So each integral costs $\mathcal{O}(n^w)$ to evaluate and there are $n$ integrals. □

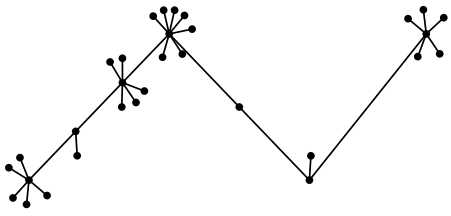


Fig. 4. A caterpillar poset.

The application of this theorem in practice requires a knowledge on how to build a BIT-order of minimal width, which we think is a difficult task in general. Taking the problem upside-down, it is interesting to investigate the poset class of a fixed BIT-width. Interestingly, the posets of BIT-width 1 are the well-known *caterpillar* posets (cf. Figure 4).

**Definition 12.** Let $S$ a poset such that $S_x = (\{x, s_1 \ldots, s_\ell\}, \{\forall i > 0, x < s_i \vee x > s_i\})$ is a *star* poset of center $x$. Let $A = (\{a_{1,1}, \ldots a_{k_1,1}, \ldots a_{1,m} \ldots a_{k_m,m}\}, <_A)$ be a *generalized alternating* poset such that:

$$\left\{ \begin{array}{c} \text{each } (a_{1,i}, \ldots, a_{k_i,i}) \text{ is a chain } (1 \le i \le m) \\ \forall i, a_{k_i,i} <_A a_{1,i+1} \wedge a_{k_{i+1},i+1} >_A a_{1,k_{i+2}} \\ \quad \vee a_{k_i,i} >_A a_{1,i+1} \wedge a_{k_{i+1},i+1} <_A a_{1,k_{i+2}} \end{array} \right\}$$

A *caterpillar* poset is a generalized alternating poset $A$ lifted union a set of star posets $(S_{i,j})$ *i.e* a main thread where each element is the center of star poset. ⌟

**Proposition 2.** *A poset has BIT-width 1 if and only if it is a caterpillar poset.*

*Proof.* First, a BIT-width 1 poset must be built by repeating the following procedure, starting from $P \leftarrow (\{x\}, <)$, $thread \leftarrow x$ and, $center \leftarrow x$:

Choose an element $u$ between $center$ and $thread$
Let $v$ be a fresh element, add $v$ to $P$ with $u < v$ or $v < u$
**if** $u = thread$ **then** $center \leftarrow u$
$thread \leftarrow v$

Then, it is trivial to check that the obtained poset is a caterpillar.

The other implication is proved by building an order of BIT-width 1 for a caterpillar. Such order is obtained by starting from a star at one of the extremities, and so decomposing the star entirely before continuing on the next one along the thread. □

**Corollary 3.** *The number of linear extensions of caterpillars can be counted using $\mathcal{O}(n^2)$ arithmetical operations.*

**Corollary 4.** *Let $E$ be a finite set of prime posets. Let $C$ be the set of posets such that their prime vertices belong to $E$ or are caterpillars. Then, there exists an $\mathcal{O}(n^2)$ to count the linear extensions of posets in $C$.*

Of course, the investigation of the BIT-width parameter is incomplete and many questions remains unanswered. We conclude the presentation with two problems left open for future work, and conjectured difficult.

**Conjecture 1.** *The following problem is NP-complete:*
BIT-WIDTH
INPUT: *a BIT-decomposable poset $P$, an integer $k \ge 3$*
OUTPUT: $w(P) \le k$

**Conjecture 2.** *The following problem is $\sharp P$-complete:*
BIT-MODULAR-COUNT
INPUT: *a BIT-modular poset $P$*
OUTPUT: $\sharp \ell e(P)$

## References


[1] K. Kangas, T. Hankala, T. M. Niinimäki, and M. Koivisto, "Counting linear extensions of sparse posets," in *IJCAI 2016*. IJCAI/AAAI Press, 2016.

[2] O. Bodini, M. Dien, A. Genitrini, and F. Peschanski, "Quantitative and algorithmic aspects of barrier synchronization in concurrency," *Discret. Math. Theor. Comput. Sci.*, vol. 22, no. 3, 2021. [Online]. Available: https://doi.org/10.46298/dmtcs.5820

[3] M. Dien, A. Genitrini, and F. Peschanski, "A combinatorial study of async/await processes," in *Theoretical Aspects of Computing - ICTAC 2022 - 19th International Colloquium, Tbilisi, Georgia, September 27-29, 2022, Proceedings*, ser. Lecture Notes in Computer Science, H. Seidl, Z. Liu, and C. S. Pasareanu, Eds., vol. 13572. Springer, 2022, pp. 170–187. [Online]. Available: https://doi.org/10.1007/978-3-031-17715-6_12

[4] G. Brightwell and P. Winkler, "Counting linear extensions is ♯p-complete," in *Proceedings of the Twenty-Third Annual ACM Symposium on Theory of Computing*, ser. STOC '91. New York, NY, USA: Association for Computing Machinery, 1991, p. 175–181.

[5] T. Gallai, "Transitiv orientierbare graphen," *Acta Mathematica Academiae Scientiarum Hungaricae*, 1967.

[6] M. Habib and C. Paul, "A survey of the algorithmic aspects of modular decomposition," *Computer Science Review*, vol. 4, no. 1, pp. 41–59, 2010.

[7] M. Habib and R. H. Möhring, "On some complexity properties of n-free posets and posets with bounded decomposition diameter," *Discrete Mathematics*, vol. 63, no. 2-3, pp. 157–182, 1987.

[8] R. H. Möhring, *Computationally Tractable Classes of Ordered Sets*, ser. Institut für Ökonometrie und Operations Research: Report, 1987.

[9] A. von Arnim, R. Schrader, and Y. Wang, "The permutahedron of n-sparse posets," *Mathematical Programming*, vol. 75, no. 1, pp. 1–18, 1996.

[10] M. Peter and G. Wambach, "N-extendible posets, and how to minimize total weighted completion time," *Discrete Applied Mathematics*, vol. 99, no. 1, pp. 157–167, 2000. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0166218X99001316

[11] S. Dittmer and I. Pak, "Counting linear extensions of restricted posets," *Electron. J. Comb.*, vol. 27, no. 4, p. 4, 2020. [Online]. Available: https://doi.org/10.37236/8552

[12] E. Eiben, R. Ganian, K. Kangas, and S. Ordyniak, "Counting linear extensions: Parameterizations by treewidth," *Algorithmica*, vol. 81, pp. 1657–1683, 2019.

[13] A. Garver, S. Grosser, J. P. Matherne, and A. Morales, "Counting linear extensions of posets with determinants of hook lengths," *SIAM Journal on Discrete Mathematics*, vol. 35, no. 1, pp. 205–233, 2021.

[14] K. Kangas, M. Koivisto, and S. Salonen, "A faster tree-decomposition based algorithm for counting linear extensions," *Algorithmica*, vol. 82, no. 8, pp. 2156–2173, 2020.

[15] G. Pruesse and F. Ruskey, "Generating linear extensions fast," *SIAM Journal on Computing*, vol. 23, no. 2, pp. 373–386, 1994.

[16] R. P. Stanley, "Two poset polytopes," *Discrete & Computational Geometry*, vol. 1, pp. 9–23, 1986.

[17] M. Peter and G. Wambach, "N-extendible posets, and how to minimize total weighted completion time," *Discrete applied mathematics*, vol. 99, no. 1-3, pp. 157–167, 2000.

[18] B. Jamison and S. Olariu, "On a unique tree representation for p4-extendible graphs," *Discrete Applied Mathematics*, vol. 34, no. 1-3, pp. 151–164, 1991.

[19] ——, "A tree representation for p4-sparse graphs," *Discrete Applied Mathematics*, vol. 35, no. 2, pp. 115–129, 1992.


## Appendix

*Proof of Theorem 4 (Prime Counting).* A proof for a similar theorem can be found in [7]. We propose below a slightly different proof in the context of the definitions given in the present paper.

Given a linear extension $\lambda \in \ell e(P)$ and an index $i,\ 1 \leq i \leq n$, we define the *complement* $\bar{\lambda}_i$ (at index $i$) such that all its elements from $\mathcal{PO}(t_i)$ are replaced by a hole $\square_i$. More formally:

$$\forall k,\ 1 \leq k \leq |\lambda|,\ \bar{\lambda}_i(k) = \begin{cases} \square_i \text{ if } \lambda(k) \in \mathcal{PO}(t_i) \\ \lambda(k) \text{ otherwise.} \end{cases}$$

As an illustration we consider three linear extensions of the prime poset of figure 2: $\lambda = \langle b, c, d, f, e, g\rangle$,
$\lambda' = \langle c, e, b, d, g, f\rangle$ and $\lambda'' = \langle c, f, b, d, g, e\rangle$.
We have, for example, $\bar{\lambda}_3 = \langle b, c, \square_3, f, e, \square_3\rangle$
and $\bar{\lambda}'_3 = \langle c, e, b, \square_3, \square_3, f\rangle$.

We now define the subset $\ell e(P)|_{\bar{\lambda}_i} = \{\kappa \mid \kappa \in \ell e(P) \wedge \bar{\kappa_i} = \bar{\lambda}_i\}$, *i.e.* the linear extensions of $P$ that induce the same holes as $\lambda_i$. By definition 6, since $\mathcal{PO}(t_i)$ is a module, all the total orders obtained by substituing the holes of $\bar{\lambda}_i$ by a linear extension of $\mathcal{PO}(t_i)$ are linear extensions of $P$. This means that the order relation of $\mathcal{PO}(t_i)$ is disjoint from $\ell e(P)|_{\bar{\lambda}_i}$, and thus $\sharp \ell e(P)|_{\bar{\lambda}_i} = \sharp \ell e(\mathcal{PO}(t_i))$.

We now consider all the complements of $\lambda$ at once, forming the set $\ell e(P)|_{\bar{\lambda}} = \bigcap_i \ell e(P)|_{\bar{\lambda}_i}$. Informally, this is the set of linear extensions of $P$ in which all the index holes are inserted. We have, as an illustration, that:
$\bar{\lambda} = \langle \square_1, \square_2, \square_3, \square_4, \square_4, \square_3\rangle$,
$\bar{\lambda}' = \langle \square_2, \square_4, \square_1, \square_3, \square_3, \square_4\rangle$ and $\bar{\lambda}'' = \bar{\lambda}'$.

This last example shows that permutations within a component $lin(\mathcal{PO}(t_i))$ do not impact the complements, as expected. Considering the fact that if $i$ and $j$ are distinct indices then $\square_i$ and $\square_j$ are also distinct, we obtain that $\sharp \ell e(P)|_{\bar{\lambda}} = \prod_{i=1}^{n} \sharp \ell e(\mathcal{PO}(t_i))$.

It remains to count the number of such sets $\ell e(P)|_{\bar{\lambda}}$. For this, we consider the partial order $\mathring{P} = Q \Cup \bigcup_i lin(\mathcal{PO}(t_i))$. For each linear extension $\lambda \in \ell e(\mathring{P})$ we have $\sharp \ell e(\mathring{P})\Big|_{\bar{\lambda}} = 1$. Indeed, all the elements of $\lambda$ are replaced by indexed holes, depending on which set $lin(\mathcal{PO}(t_i))$ is the element from, and this series of holes is of course unique, which means that if $\lambda \neq \lambda'$ then $\ell e(\mathring{P})\Big|_{\bar{\lambda}} \neq \ell e(\mathring{P})\Big|_{\bar{\lambda'}}$. This means that $\ell e(\mathring{P})$ can be bijectively mapped to $\ell e(P)|_{\bar{\lambda}}$ by reading the indices of holes in order, thus:

$$\begin{aligned} \sharp \ell e(P) &= \textstyle\sum \{\sharp X \mid X = \ell e(P)|_{\bar{\lambda}}\} \\ &= \sharp \ell e(Q \Cup \textstyle\bigcup_i lin(\mathcal{PO}(t_i))) \cdot \prod_i \sharp \ell e(\mathcal{PO}(t_i)) \end{aligned}$$

□