ELTE Eötvös Loránd University

Faculty of Science

# Linear extensions of partially ordered sets

Math project I. report

Júlia Éles

Applied mathematics MSc

Supervisor:

Péter Madarasi

HUN-REN Alfréd Rényi Institute of Mathematics, and
Department of Operations Research, Eötvös Loránd University

Budapest
2025

## 1 Introduction

Given a set $P$ and a partial ordering $\prec$ on it. Let us count the number of linear extensions of the partial order, i.e., the number of bijective functions $\pi : P \to \{1, \dots, n\}$ such that $\forall a, b \in P : a \prec b \Rightarrow \pi(a) < \pi(b)$.

The task is equivalent to determining the number of topological orderings of the *cover graph* $G = (V, A)$ of $P$, where $V = P$ and $A = \{uv \in P^2 : u \prec v, \nexists w \in P \setminus \{u, v\} : u \prec w \prec v\}$.

Many have examined this problem before: G. Brightwell and P. Winkler proved in 1991 that the problem is #P-hard [2]. In 2016, Kangas et al. provided a polynomial algorithm for partial orderings with cover graphs of constant treewidth [3]; their dynamic programming-based algorithm has a time complexity of $O(n^{t+4})$, where $t$ is the treewidth of the cover graph. This was later improved by K. Kangas, M. Koivisto, and S. Salonen to $O(n^{t+3})$ in 2020 using a nice tree decomposition of the diagram [4].

## 2 Counting linear extensions in a tree

In general, it is difficult to determine the number of topological orders, but if the covering graph is a tree, there are solutions that are faster than the algorithms mentioned above. In 1990, M. D. Atkinson presented an algorithm with a running time of $O(n^2)$[1].

The algorithm is based on *spectra*. The *spectrum function* associated with the graph $G$ is the function $\sigma_G : V \to \mathbb{Z}_+^n$. The *spectrum* of the vertex $v \in V$ is $\sigma_G(v) = (\sigma_G(v)_1, \sigma_G(v)_2, \dots, \sigma_G(v)_n)$, where $\sigma_G(v)_i$ is the number of topological orders in which the vertex $v$ is in the $i$-th position.

In one step of the algorithm, we split the graph along a directed edge $uv$. Let $G_u$ be the component containing $u$ after the split, and $G_v$ be the component containing $v$. Then, using $\sigma_{G_u}(u)$ and $\sigma_{G_v}(v)$, we can easily determine the value of either $\sigma_G(u)$ or $\sigma_G(v)$. The values of $\sigma_{G_u}(u)$ and $\sigma_{G_v}(v)$ can be calculated recursively. The number of topological orders of the graph is $\sum\limits_{i=1}^{n} \sigma_G(w)_i$ for any vertex $w \in V$.

We examined this algorithm in more detail this semester.

### 2.1 Identifying trees using spectra

We implemented the algorithm described above and first examined whether either the number of linear extensions or the spectrum uniquely identify the tree up to isomorphism. We checked this using a computer and found that neither the number of linear extensions nor the set of spectra uniquely identifies the tree, even in the case of arborescences.

However, we realized that if we assign a *common spectrum* to each pair of vertices, then the set of all such common spectra uniquely identifies the tree.

The *common spectrum function* is a generalization of the previously defined spectrum function, as we specify the positions of two vertices instead of one. Formally, the common spectrum function for the graph $G$ is the function $\tau_G : V^2 \to \mathbb{Z}_+^{n \times n}$, that assigns the common spectrum $\tau_G(u, v)$ to a vertex pair $(u, v)$, a matrix where $\tau_G(u, v)_{ij}$ is the number of topological orders in which $u$ is in the $i$-th place and $v$ is in the $j$-th place.

### 2.2 Calculating the common and the arc spectra

During the semester, we tried to find a method for determining the previously mentioned common spectrum. We were successful with a simpler version of these, the *arc spectrum*, which are a special case of common spectrum: $u$ and $v$ are connected by an arc in the graph, i.e., $uv \in A$.

In order to calculate the arc spectrum of $u$ and $v$, we need the spectrum of $u$ and $v$. We obtain these by cutting the graph along the edge $uv$, where $G_u$ is the component containing $u$ and $G_v$ is the component containing the vertex $v$, and we determine the values of $\sigma_{G_u}(u)$ and $\sigma_{G_v}(v)$ using the above algorithm. Let us denote the number of vertices in $G_u$ by $p$, the number of vertices in $G_v$ by $q$, and the number of vertices in $G$ by $n$ (we know that $p + q = n$).

We would like to determine $\tau_G(u,v)_{rs}$. We know how many orders there are in $G_u$ where vertex $u$ is in the $i$-th position, and how many orders there are in $G_v$ where vertex $v$ is in the $j$-th position. We want to merge two such sequences. There are already $i-1$ elements before vertex $u$, which means that $r-i$ elements must still be selected from sequence of $v$; these elements can be freely "mixed". Similarly, there are already $q-j$ elements after vertex $v$, so $p-s+j$ elements from the other sequence must be placed after $v$. Thus, our combined sequence is divided into three parts by $u$ and $v$, in which the elements in the sequences $u$ and $v$ are mixed. After a quick calculation, we get the following formula:

$$\tau_G(u,v)_{rs} = \sum_{i=1}^{\min(r,p)} \binom{r-1}{i-1} \cdot \sigma_{G_u}(u)_i \cdot \sum_{j=\max(r-i+1,s-p)}^{\min(q,s-i-1)} \binom{s-r-1}{s-j-i} \binom{n-s}{q-j} \cdot \sigma_{G_v}(v)_j.$$

It is also worth examining the common spectrum of vertex pairs connected by a directed path. In this case, we divide the graph along the edges of the path, so that each vertex has its own subgraph in which we can calculate its spectrum. We can then calculate the common spectrum from these spectra in two ways:

1. We calculate the common spectrum of the first and last elements, excluding the intermediate elements. Taking the intermediate elements into account, we recursively calculate the common spectra of the two extreme elements, i.e., the common spectrum of the second and penultimate elements. From the two spectrum obtained in this way, we calculate the common spectrum of first and last elements for the whole graph.
2. We calculate the common spectrum of the first and second elements, then calculate the spectrum of the first and third elements from this and the spectrum of the third element. We continue this until we reach the other end of the path.

## 3 Summary and future plans

In this semester, we examined linear extensions of partial orderings whose cover graph is a tree. We implemented Atkinson's algorithm and extended the concept of spectrum used in it for two vertices, providing an algorithm for special cases where the two vertices are connected by an edge or a directed path.

In the next semester, we would like to provide an algorithm similar to the above for calculating the common spectrum of vertex pairs that are not connected by a directed path. Furthermore, it is also an interesting question whether there is an efficient algorithm for calculating the common spectrum of $3, 4$ or arbitrary $k$ vertices.

## References

[1] M. D. Atkinson. On computing the number of linear extensions of a tree. *Order*, 7(1):23–25, 1990.

[2] G. Brightwell and P. Winkler. Counting linear extensions is #P-complete. In *Proceedings of the Twenty-Third Annual ACM Symposium on Theory of Computing*, pages 175–181, 1991.

[3] K. Kangas, T. Hankala, T. M. Niinimäki, and M. Koivisto. Counting linear extensions of sparse posets. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 603–609, 2016.

[4] K. Kangas, M. Koivisto, and S. Salonen. A faster tree-decomposition based algorithm for counting linear extensions. *Algorithmica*, 82(8):2156–2173, 2020.