AUSTRALASIAN JOURNAL OF COMBINATORICS
Volume **64(1)** (2016), Pages 154–165

The complexity of counting poset and permutation patterns

Joshua Cooper Anna Kirkpatrick

Mathematics Department
University of South Carolina
1523 Greene St., Columbia, SC 29208
U.S.A.

Abstract

We introduce a notion of pattern occurrence that generalizes both classical permutation patterns as well as poset containment. Many questions about pattern statistics and avoidance generalize naturally to this setting, and we focus on functional complexity problems – particularly those that arise by constraining the order dimensions of the pattern and text posets. We show that counting the number of induced, injective occurrences among dimension-2 posets is #**P**-hard; enumerating the linear extensions that occur in realizers of dimension-2 posets can be done in polynomial time, while for unconstrained dimension it is **GI**-complete; counting not necessarily induced, injective occurrences among dimension-2 posets is #**P**-hard; counting injective or not necessarily injective occurrences of an arbitrary pattern in a dimension-1 text is #**P**-hard, although it is in **FP** if the pattern poset is constrained to have bounded intrinsic width; and counting injective occurrences of a dimension-1 pattern in an arbitrary text is #**P**-hard, while it is in **FP** for bounded-dimension texts. This framework naturally leads to a number of open questions, chief among which are (1) is it #**P**-hard to count the number of occurrences of a dimension-2 pattern in a dimension-1 text, and (2) is it #**P**-hard to count the number of permutations avoiding a given pattern?

1 Introduction

A tremendous amount of study has been dedicated to understanding occurrence or non-occurrence of combinatorial substructures: which substructures are avoidable and counting objects that avoid them, what substructures random and random-like objects possess, enumerating substructures in general objects, describing the subclass of objects that have particular substructure counts, etc. Much interesting work (particularly in model theory) has concerned the completely general question of substructure occurrence, but the degree of abstraction involved changes the nature

of which questions are useful to ask. However, here we investigate a somewhat more specific perspective that still allows us to address important questions from two disparate but highly studied areas: permutation patterns and subposet containment. In order to illustrate this unifying viewpoint, we first describe these two topics.

1. A *permutation of* n is a bijection from $[n] = \{1, \ldots, n\}$ to itself for some positive integer n. Given a permutation σ, called a *pattern*, and another permutation τ of n, called the *text*, we say that σ *occurs on* or *matches* the index set $\mathcal{I} \subset [n]$ in τ if $\tau|_{\mathcal{I}}$ is order-isomorphic to σ; that is, if $\mathcal{I} = \{a_1, \ldots, a_k\}$ with $a_1 < \cdots < a_k$, for any $i, j \in [k]$, $\sigma(i) < \sigma(j)$ if and only if $\tau(a_i) < \tau(a_j)$. Interesting questions about pattern occurrence include the complexity of counting the number of occurrences of a pattern, the distribution of pattern counts for random permutations, enumeration of permutations which avoid a given pattern, and the structure of permutations with specified pattern counts.

2. A *poset of size* n is a set (the *ground set*) of cardinality n associated with a partial order of the set, that is, a binary relation which is reflexive, antisymmetric, and transitive. Given posets $P = (S, \prec_P)$ and $Q = (T, \prec_Q)$, we say that Q contains P (as a *subposet*) on the set $U \subset T$ if there is an order-preserving bijection between P and $Q|_U$; that is, if there exists a bijection $f : S \to U$ so that, for $x, y \in S$, $x \prec_P y$ implies $f(x) \prec_Q f(y)$. Furthermore, the containment is said to be *induced* if the implication is in fact biconditional, and *unlabelled* if it is understood only up to order automorphisms of P. Interesting questions about poset containment include the complexity of counting the number of subposets of a given type, the size of the largest subposet of a given poset not containing a fixed subposet, and the nature of linear extensions of a poset (which correspond in the present language to occurrences of a poset in a chain – a total order – of equal size).

To unify these perspectives, we introduce a notion of pattern occurrence that generalizes classical permutation pattern matching as well as poset containment. In the most general formulation, let P and Q be posets which we term the *pattern* and *text* posets, respectively. We say that P *occurs at* a subposet Q' of Q if there exists an onto function $f : P \to Q'$ so that f is order-preserving, i.e., $v \prec_P w$ implies $f(v) \prec_Q f(w)$; in this case, f is called an *occurrence* of P in Q. Furthermore, we say that the occurrence is *induced* if f is in fact an order isomorphism, i.e., f is an occurrence so that $f(v) \prec_Q f(w)$ implies $v \prec_P w$, and we say that the occurrence is *injective* (*bijective*) if f is injective (respectively, $Q' = Q$). One can also speak of *unlabeled* pattern occurrences as equivalence classes of occurrences from a pattern P to a text Q modulo automorphisms of P.

We reformat several classical problems in the language of permutation patterns using the notion of order dimension (sometimes called *Dushnik-Miller dimension* [6]). Given a poset P, a *linear extension* of P is a bijective occurrence of P in a chain C (a totally ordered poset). We use an equivalent definition from poset theory interchangeably with this: a linear extension is a total order $\prec'$ on the ground set of P so that $v \prec_P w$ implies $v \prec' w$ for any $v, w \in P$. A family $\mathcal{R} = \{f_1, \ldots, f_r\}$ of

linear extensions of P is said to be a *realizer* of P if a relation $v \prec w$ is in P if and only if $f_j(v) \prec_C f_j(w)$ for every $j \in [r]$; a realizer $\mathcal{R}$ is *minimal* if and only if it has the fewest possible number of elements among all realizers of P; the cardinality of a minimal realizer of P is the *dimension* $\dim(P)$ of P. The *width* of a poset is the size of its largest antichain, i.e., subset of vertices between which there are no relations. An *automorphism* of a poset is a bijective occurrence of a poset in itself.

One of the main reasons that order dimension gives rise to interesting questions about poset pattern occurrence is the fact that unlabeled, induced, injective pattern occurrence corresponds in a precise way to permutation pattern matching when both pattern and text have dimension 2. In particular, suppose $\dim(P) = \dim(Q) = 2$, and let P have a realizer consisting of $([k], <)$ and $([k], \prec_P^*)$ and Q has a realizer consisting of $([n], <)$ and $([n], \prec_Q^*)$. One can think of P as representing the permutation σ_P of k such that $\sigma_P(i) < \sigma_P(j)$ if and only if $i \prec_P^* j$ and of Q as similarly representing σ_Q of n. Proposition 3.1 provides a rubric connecting permutation pattern matching to poset patterns occurrence in dimension 2.

We also refer to standard texts in computational complexity theory to precisely define hardness of decision and functional complexity problems (e.g., [1]). Roughly, a decision problem is in **P** if the answer can be obtained in polynomial time (in the size of the input instance); it is in **NP** if the answer can be certified in polynomial time; it is **NP**-hard if every problem in **NP** can be reduced to it in polynomial time (i.e., it is at least as hard as all problems in **NP**); it is **NP**-complete if it is **NP**-hard and in **NP**. Similarly, a function problem (a computational problem whose output is an integer instead of only a single bit) is in **FP** if the answer can be obtained in polynomial time (in the size of the input instance); it is in #**P** if it consists of computing the number of correct solutions to a problem in **NP**; it is #**P**-hard if the problem of computing the number of correct solutions to any problem in **NP** can be reduced to this problem in polynomial time; it is #**P**-complete if it is #**P**-hard and in #**P**.

2 Results

In the following, we denote by P a pattern poset and by Q a text poset.

Theorem 2.1. *If* $\dim(P) = \dim(Q) = 2$*, the problem of computing the number of unlabeled, induced, injective occurrences of* P *in* Q *is* #**P***-hard.*

Given a permutation σ of n, there is a poset $D(\sigma)$ associated with σ, whose ground set is $[n]$ and $i \prec_{D(\sigma)} j$ if and only if $i < j$ and $\sigma(i) < \sigma(j)$. In other words, $D(\sigma)$ is the two-dimensional poset with realizer comprised of the ordinary total order $\leq$ on $[n]$ and the pullback $\sigma^*(\leq)$. We can define an automorphism of σ simply to be an automorphism of $D(\sigma)$. A not-necessarily-induced match of a permutation pattern σ in a permutation text τ is an occurrence of $D(\sigma)$ in $D(\tau)$; in the language of permutations, these are maps between the corresponding index sets that preserve co-inversions but not necessarily inversions. (For a permutation $\sigma \in \mathfrak{S}_n$ and $i, j \in [n]$, the pair $\{i, j\}$ is said to be an *inversion* if $(i-j)(\sigma(i)-\sigma(j)) < 0$ and a *co-inversion* if $(i-j)(\sigma(i)-\sigma(j)) > 0$.)

Proposition 2.2. *The problem of counting the number of automorphisms of a dimension-2 poset is in* **FP***; equivalently, the problem of counting the number of automorphisms of a permutation is in* **FP***.*

Note the constrast with poset automorphism counting in general. Indeed, poset automorphism counting is at least as hard as bipartite poset automorphism counting, which is easy to see is polynomial-time equivalent to bipartite graph automorphism counting; bipartite graph isomorphism counting is known to be as hard as general graph isomorphism counting by, for example, [10]. By [11], this is polynomial-time reducible to the graph isomorphism decision problem, and is therefore so-called **GI**-*complete*, a complexity class widely believed to be strictly between **P** and **NP**-hard.

Theorem 2.3. *If* $\dim(P) = \dim(Q) = 2$*, the problem of computing the number of labeled, induced, injective occurrences of* P *in* Q *is* #**P***-hard.*

Corollary 2.4. *For any pattern* P *and text* Q*, the problem of computing the number of (labeled or unlabeled) induced, injective occurrences of* P *in* Q *is* #**P***-hard.*

Theorem 2.5. *If* $\dim(P) = \dim(Q) = 2$*, the problem of computing the number of (labeled or unlabeled) not necessarily induced, injective occurrences of* P *in* Q *is* #**P***-hard.*

Corollary 2.6. *Deciding whether a given dimension-2 poset has a not necessarily induced, injective, unlabeled match in another dimension-2 poset is* **NP***-complete.*

Theorem 2.7. *If* $\dim(Q) = 1$*, the problem of counting the number of (injective or not necessarily injective) occurrences of an arbitrary* P *in* Q *is* #**P***-hard.*

This essentially a restatement of Brightwell and Winkler's famous result that counting the number of linear extensions of a poset is #**P**-hard. By contrast, some special cases of this problem are in fact easy. Before proceeding, we define the *(Gallai) modular decomposition*[1] of a poset. Given $P = (S, \prec)$, define a subset $T \subset S$ to be a *module* of P if, for all $u, v \in T$ and $x \in S \setminus T$, $u \prec x$ if and only if $v \prec x$ and $x \prec u$ if and only if $x \prec v$. A module T is *strong* if, for any module $U \subset S$, $U \cap T \neq \emptyset$ implies $U \subset T$ or $T \subset U$. Thus, the nonempty strong modules of P form a tree order, called the *(Gallai) modular decomposition* of P. A strong module or poset is said to be *indecomposable* if its only proper submodules are singletons and the empty set. It is a result of Gallai ([8]) that the maximal proper strong modules of P form a partition $\mathrm{Gal}(P)$ of T, and it is straightforward to see that the quotient poset $P/\mathrm{Gal}(P)$ is well-defined. The comparability graph $G(P)$ of a poset P has as its vertex set the ground set of P and has an edge $\{x, y\}$ for $x \neq y$ if $x \prec_P y$ or $y \prec_P x$. Specifically, Gallai showed the following.

Theorem 2.8 (Gallai [8])**.** *Given a poset* P *such that* $|P| \geq 2$*, one of the following holds.*

[1]Unfortunately, there are quite a few names in the literature given to modules in addition to *modules*: *autonomous sets*, *intervals*, *homogeneous sets*, *partitive sets*, and *clans*, for example.

1. *(Parallel-Type) If $G(P)$ is not connected, then $\mathrm{Gal}(P)$ is the family of subposets induced by the connected components of $G(P)$ and $P/\mathrm{Gal}(P)$ is an antichain.*

2. *(Series-Type) If the complement $\overline{G(P)}$ of $G(P)$ is not connected, then $\mathrm{Gal}(P)$ is the family of subposets induced by the connected components of $\overline{G(P)}$ and $P/Gal(P)$ is a chain.*

3. *(Indecomposable) Otherwise, $|Gal(P)| \geq 4$ and $P/Gal(P)$ is indecomposable.*

Define the *intrinsic width* $\mathrm{iw}(P)$ of a poset as the maximum width of the posets $P|_T/\mathrm{Gal}(P|_T)$ over all nodes T of the tree order given by the Gallai modular decomposition of P. (So, for example, series-parallel posets are characterized by having intrinsic width 1.) The following strengthens a result of Steiner ([14]), who provides a similar, albeit incomplete, proof of a slightly weaker result.

Theorem 2.9. *If the intrinsic width of a poset is bounded, its number of linear extensions (i.e., bijective occurrences as a pattern in a dimension-1 text poset) can be computed in polynomial time. In particular, given a chain Q, if $\mathrm{iw}(P) \leq k$, there is an algorithm that computes in $O(n^{\max(4,k+1)})$ time the number of occurrences of P in Q.*

Theorem 2.10. *If $\dim(P) = 1$, then counting the number of injective occurrences of P in an arbitrary poset Q is $\#\mathbf{P}$-hard.*

3 Proofs

Proposition 3.1. *The matches of σ_P in σ_Q are in bijection with the unlabeled, induced, injective occurrences of P in Q.*

Proof. Note that the matches of σ_P in σ_Q correspond exactly with certain subsets of $[n]$ of size k, namely, those $\mathcal{I} \in \binom{[n]}{k}$ so that $\sigma_Q|_{\mathcal{I}}$ is order-isomorphic to σ_P. Suppose $\mathcal{I} = \{r_1 < \cdots < r_k\}$, and define $f : [k] \to [n]$ by $f(i) = r_i$. We claim that f provides an isomorphism between P and $Q|_{f(\mathcal{I})}$, and thus is an induced, injective occurrence of P in Q on the set $f(\mathcal{I})$. Note that, for $i, j \in [k]$, if $i \prec_P j$, then $i < j$ and $\sigma_P(i) < \sigma_P(j)$, whence $r_i < r_j$ and $\sigma_Q(r_i) < \sigma_Q(r_j)$, so $r_i = f(i) \prec_Q f(j) = r_j$; furthermore, this argument is reversible, so $i \prec_P j$ if and only if $f(i) \prec_Q f(j)$.

We now show that this map from matches of σ_P in σ_Q to occurrences of P in Q is unique up to automorphisms of P. Suppose that g and h are induced, injective occurrences of P in Q with $g([k]) = h([k])$, so $i \prec_P j$ if and only if $g(i) \prec_Q g(j)$ if and only if $h(i) \prec_Q h(j)$. Define $\tau = h^{-1} \circ g$. We claim that τ is an automorphism of P. Indeed, suppose $i, j \in [k]$; we wish to show that $\tau(i) \prec_P \tau(j)$ if and only if $i \prec_P j$. Indeed, $\tau(i) \prec_P \tau(j)$ if and only if $h^{-1}(g(i)) \prec_P h^{-1}(g(j))$ if and only if $h(h^{-1}(g(i))) \prec_Q h(h^{-1}(g(j)))$ if and only if $g(i) \prec_Q g(j)$ if and only if $i \prec_P j$. Therefore, matches from σ_P to σ_Q correspond bijectively to equivalence classes under automorphisms of P of induced, injective occurrences of P in Q, i.e., unlabeled, induced, injective occurrences of P in Q. □

It is easy to see that the not necessarily induced matches of σ_P in σ_Q are also in bijection with the not necessarily induced, unlabeled, injective occurrences of P in Q.

Proof of Theorem 2.1. Note that, by Proposition 3.1, the problem of computing the number of unlabeled, induced, injective occurrences of P in Q is polynomial-time reducible to the problem of counting matches of σ_P in σ_Q. Since the pattern and text here are arbitrary, by [3], this is a #**P**-hard computational problem. □

The next proof closely resembles in some aspects the argument for Theorem 5 of [7].

Proof of Proposition 2.2. Let $M_1, \ldots, M_k$ be the indecomposable strong modules of P. By Theorem 4.2 of [4], each $P[M_i]$ has at most two automorphisms. In particular, if $(\prec_i^1, \prec_i^2)$ is a realizer of $P[M_i]$ (unique up to ordering by [8]), and τ is the permutation so that $a \prec_i^1 b$ if and only if $\tau_i(a) \prec_i^2 \tau_i(b)$, then $P[M_i]$ has either no nontrivial automorphisms, or else τ_i is its only one. Note that it is certainly polynomial-time to check if τ_i is indeed an automorphism; let $t \leq k \leq |P|$ be the number of such i, so that computing 2^t is in **FP**. It is now straightforward to describe all automorphisms of P. Since automorphisms preserve (strong) modules, all automorphisms of P arise as automorphisms of the indecomposable strong modules composed with automorphisms of the tree corresponding to the Gallai decomposition. Furthermore, series-type nodes have only trivial automorphisms, while parallel-type nodes can be arbitrarily reordered, so the number of automorphisms has size

$$2^t \prod_{P_0 \subset P} |\operatorname{Gal}(P_0)|!$$

where the P_0 vary over all parallel-type strong modules of the Gallai decomposition. By [2], it is possible to compute the entire Gallai decomposition in polynomial time; since

$$\log \prod_{P_0 \subset P} |\operatorname{Gal}(P_0)|! < \sum_{P_0 \subset P} |\operatorname{Gal}(P_0)|^2 \leq (\sum_{P_0 \subset P} |\operatorname{Gal}(P_0)|)^2 \leq 4|P|^2,$$

this shows that computing the number of automorphisms of a dimension-2 poset is in **FP**. (The total number of vertices of a rooted tree none of which have exactly one child is at most twice the number of leaves.) □

Proof of Theorem 2.3. By Proposition 2.2 the problem of computing the number of labeled, induced, injective occurrences of one dimension-2 poset in another is polynomial-time reducible to the problem of computing the number of unlabeled, induced, injective occurrences of one dimension-2 poset in another. This latter problem is #**P**-hard by Theorem 2.1. □

Proof of Corollary 2.4. This follows immediately from Theorems 2.1 and 2.3, since the problem without dimension constraints is more general. □

The next proof involves a modification of the argument of [3], and in fact can be used to provide another proof of the #**P**-hardness of permutation pattern matching because all of the matches involved are in fact induced.

Proof of Theorem 2.5. Since not necessarily induced, unlabeled, injective occurrences of a dimension-2 poset P in a dimension-2 poset Q are equivalent by Proposition 3.1 to not necessarily induced matches of σ_P in σ_Q, we show that the latter problem is #**P**-hard. Suppose Σ is an instance of 3-SAT over n variables $\{x_1, \ldots, x_n\}$, i.e.,

$$\Sigma = C_1 \wedge \cdots \wedge C_m$$

where $C_i = (v_1^i \vee v_2^i \vee v_3^i)$, each v_j^i being a literal of the form $x_{a(i,j)}$ or $\neg x_{a(i,j)}$, $a(i,j) \in [n]$. We assume that no variable occurs both positively and negatively in the same clause. Define a pattern π and text τ permutation as follows. More correctly, for convenience of notation, we define two sequences of distinct reals which can be interpreted as permutations. We treat sequences and words interchangeably, writing concatenation as $(\cdot)$-product. Then

$$\pi = \pi_1^x \cdots \pi_n^x \cdot \pi_1^C \cdots \pi_m^C$$

and

$$\tau = \tau_1^x \cdots \tau_n^x \cdot \tau_1^C \cdots \tau_m^C.$$

Define

$$\pi_i^x = (2n + 2i - 1) \cdot i \cdot (2n - i + 1) \cdot (2n + 2i),$$

and

$$\begin{aligned}\tau_i^x = {} & (4n + 4i - 1) \cdot (2i - 1) \cdot (4n - 2i + 2) \cdot (4n + 4i) \\ & \cdot (4n + 4i - 3) \cdot (2i) \cdot (4n - 2i + 1) \cdot (4n + 4i - 2).\end{aligned}$$

We need a few more definitions before describing the π_i^C and τ_i^C, $1 \le i \le m$. In particular, we describe π_i^C inductively, that is, once π_1^C through π_{i-1}^C have been described. Let u_{ij} for each $j \in [3]$ be any real number strictly between $a(i,j)$ and $2n - a(i,j) + 1$ so that u_{ij} is strictly larger than $u_{i'j}$ for each $1 \le i' < i$. Define

$$\pi_i^C = (4n + 2i - 1) \cdot u_{i1} \cdot u_{i2} \cdot u_{i3} \cdot (4n + 2i).$$

Now, we describe τ_i^C inductively. Let T_j for each $j \in [n]$ be the open interval $(2j - 1, 4n - 2j + 2)$; let F_j be the open interval $(2j, 4n - 2j + 1)$. Now, for each $j \in [n]$, let $T_{ij} = T_j$ if x_j occurs positively in C_i and $T_{ij} = F_j$ if x_j occurs negatively in C_i; similarly, let $F_{ij} = F_j$ or $F_{ij} = T_j$ if x_j occurs positively or negatively in C_i, respectively. (We do not define T_{ij} or F_{ij} if x_j does not occur in C_i.) For each x_j that appears in C_i, choose $t_{ijk} \in T_{ij}$ for each $k \in [4]$ so that

$$t_{ij1} < t_{ij2} < t_{ij3} < t_{ij4}$$

and $t_{ij1} > t_{i'j4}$ for each $1 \le i' < i$. Next, for each x_j that appears in C_i, choose $f_{ijk} \in T_{ij}$ for each $k \in [3]$ so that

$$f_{ij1} < f_{ij2} < f_{ij3}$$

and $f_{ij1} > f_{i'j3}$ for each $1 \leq i' < i$. Finally,

$$\begin{aligned}\tau_i^C =&(8n+14i-1)\cdot q_0(i)\cdot(8n+14i)\cdot\\ &(8n+14i-3)\cdot q_1(i)\cdot(8n+14i-2)\cdot\\ &(8n+14i-5)\cdot q_2(i)\cdot(8n+14i-4)\cdot\\ &(8n+14i-7)\cdot q_3(i)\cdot(8n+14i-6)\cdot\\ &(8n+14i-9)\cdot q_4(i)\cdot(8n+14i-8)\cdot\\ &(8n+14i-11)\cdot q_5(i)\cdot(8n+14i-10)\cdot\\ &(8n+14i-13)\cdot q_6(i)\cdot(8n+14i-12),\end{aligned}$$

where

$$\begin{aligned}q_i(0) =&t_{ia(i,1)1}\cdot t_{ia(i,2)1}\cdot t_{ia(i,3)1},\\ q_i(1) =&t_{ia(i,1)2}\cdot t_{ia(i,2)2}\cdot f_{ia(i,3)1},\\ q_i(2) =&t_{ia(i,1)3}\cdot f_{ia(i,2)1}\cdot t_{ia(i,3)2},\\ q_i(3) =&t_{ia(i,1)4}\cdot f_{ia(i,2)2}\cdot f_{ia(i,3)2},\\ q_i(4) =&f_{ia(i,1)1}\cdot t_{ia(i,2)3}\cdot t_{ia(i,3)3},\\ q_i(5) =&f_{ia(i,1)2}\cdot t_{ia(i,2)4}\cdot f_{ia(i,3)3},\\ q_i(6) =&f_{ia(i,1)3}\cdot f_{ia(i,2)3}\cdot t_{ia(i,3)4}.\end{aligned}$$

We claim that satisfying assignments of Σ are in bijection with matches of π in τ.

Claim 1. Consider any not necessarily induced match of π into τ. We claim that π_i^x matches into τ_i^x for each $i \in [n]$ and π_j^C matches into τ_j^C for each $j \in [m]$. Note that the following is an increasing subsequence of π of length $2n+2m$:

$$\pi_0 = (2n+1)\cdot(2n+2)\cdots(4n+2m-1)\cdot(4n+2m).$$

Let z_k, for each $k \in [2n+2m]$, be the index so that $\pi(z_k) = \pi_0(k)$. Suppose that $\{\tau(z'_k)\}_{k=1}^{2n+2m}$, with $z'_{k+1} > z'_k$ for each $k \in [2n+2m-1]$, is an increasing subsequence τ_0 of τ that can occur as the image of π_0 in some match of π into τ. Then, for each $k \in [2n+2m-1]$, we must have

$$z'_{k+1} - z'_k \geq z_{k+1} - z_k.$$

It is straightforward to see that every such τ_0 has the form

$$\tau_0(r_1,\ldots,r_n;s_1,\ldots,s_m) = w_1^x\cdots w_n^x\cdot w_1^C\cdots w_m^C$$

where w_i^x is a subsequence of τ_i^x of the form $(4n+4i-1-2r_i)\cdot(4n+4i-2r_i)$ for some $r_i \in \{0,1\}$ and w_i^C is a subsequence of τ_i^C of the form $(8n+14i-1-2s_i)\cdot(8n-14i-2s_i)$ for some $s_i \in \{0,1,2,3,4,5,6\}$.

Claim 2. Consider any not necessarily induced match of π into τ; we claim it has a very particular structure, described as follows. By Claim 1, π_0 matches

precisely some τ_0. First, $i\cdot(2n-i+1)$ must match to $(2i-1+r_i)\cdot(4n+2i+2-r_i)$. Then, $u_{i1}\cdot u_{i2}\cdot u_{i3}$ must match to $q_i(s_i)$. These positions are forced because there are precisely two (respectively, three) elements of the sequence between the elements $2n+2i-1$ and $2n+2i$ for each $i\in[n]$ in π_0 and between the elements of w_i^x in τ_0 (respectively, $4n+2i-1$ and $4n+2i$ for each $i\in[m]$ in π_0 and between the elements of w_i^C in τ_0). Furthermore, it is straightforward to see that any such map from π to τ is indeed an (induced!) match, and in fact, the s_i's are determined by the r_i's.

Claim 3. We claim that matches of π into τ, as described above, are in bijection with satisfying assignments. Given a match of π into τ, the corresponding assignment sets x_i equal to true if $r_i=0$ and false if $r_i=1$; the clause C_i is satisfied by the assignment because v_{ij} receives the value $\top$ if the j-th binary digit of s_i is 0 and $\bot$ otherwise, and $s_i\in\{0,1,2,3,4,5,6\}$. Finally, it is clear that every satisfying assignment arises from such a match by choosing the r_i's to reflect the appropriate variable settings. □

Because the matches used in the previous proof may be considered not necessarily induced, and they correspond exactly to satisfying assignments of the 3-CNF formula involved, Corollary 2.6 follows immediately.

Proof of Theorem 2.7. Note that the number of injective occurrences of P in Q is just the number of linear extensions of P times $\binom{|Q|}{|P|}$, the latter quantity being computable in polynomial time, and the former being #**P**-hard by the result [5] of Brightwell and Winkler. The number of not necessarily injective occurrences of P in Q is the number of linear extensions of P times $\binom{|Q|+|P|-1}{|P|}$, the latter quantity being computable in polynomial time, and the former being #**P**-hard by the result [5] of Brightwell and Winkler. □

Before proceeding, note that a *down-set* of a poset P is simply a set D of elements of P so that $x\in D$, $y\in P$, and $y\prec_P x$ implies $y\in D$; we write $x\lessdot_P y$ if y covers x in P, i.e., $x\prec_P y$ and there is no $z\in P$ so that $x\prec_P z$ and $z\prec_P y$.

Proof of Theorem 2.9. First, suppose P has width k (a constant) and ground set $[n]$. Then, by Dilworth's Theorem, there is a chain decomposition $\mathcal{C}=\{C_1,\ldots,C_k\}$; as mentioned in [9], there are well-known $O(n^3)$ algorithms for computing the Dilworth decomposition. Now, we construct the lattice L of down-sets of P from $\mathcal{C}$, keeping track of $|D\cap C_i|$ for each i and the children $D'\lessdot_L D$ of D as we construct the down-sets D. Starting from the empty set (which is the minimal element of L), we iteratively build up all down-sets by considering the least unused element of each C_i one at a time. That is, given some down-set D, we test if $D\cup\{x_i\}$ is also a down-set for each x_i, the least element of C_i which does not appear in D, by checking if x_i satisfies $y_j\not\prec x_i$ for each $j\in[k]$, where y_j is the element (if it exists) of height $|D\cap C_j|+1$ in C_j; this requires at most k^2 comparisons per down-set D. It is straightforward to update the $|D\cap C_j|$ and child lists appropriately for each new down-set. Since

down-sets are uniquely determined by the quantities $|D \cap C_i|$, $i \in [k]$, there are at most n^k such D. Therefore, the algorithm so far has cost $O(n^{\max(3,k)})$ time.

Next, we use L to compute the number of linear extensions of P. Let $f(D)$, for a down-set D of P, denote the number of linear extensions of D; it is easy to see that

$$f(D) = \sum_{D' \lessdot_L D} f(D'),$$

a calculation that requires summing at most a constant (k) number of integers at each step. Again, the number of down-sets is at most n^k, so we obtain $f(P)$, the desired quantity, in time $O(n^{\max(3,k)})$.

Now, suppose $\mathrm{iw}(P) \leq k$. By [12], it is possible to compute the Gallai decomposition of a poset in $O(n^2)$ time. Denote by $I(P)$ the set of *indecomposable-type* nodes T of the Gallai modular decomposition tree of P, and by $I_d(P)$ the set of such nodes at depth d of the tree, with the convention that $[n]$ is at depth $d = 1$. For $T \in I_d(P)$, write

$$P_T = P|_T \cup \bigcup_S L_S$$

where S ranges over $S \in I_{d+1}(P)$ so that $S \subset T$, and L_S denotes any linear extension of $P|_S$. (That is, each maximal proper strong submodule S of $P|_T$ has been replaced by a chain of length $|S|$.) It is straightforward to see that the width of P_T is at most k.

Noting that the cardinality of P_T is $|T|$, the nodes of $I(P)$ at a given depth are pairwise disjoint, and $I_d(P)$ is empty for $d > n$, the time to compute the number of linear extensions of all of them is, by the above argument, at most

$$\begin{aligned} O\left(\sum_{T \in I(P)} |P_T|^{\max(3,k)}\right) &\leq O\left(\sum_{d \geq 1} \left(\sum_{T \in I_d(P)} |P_T|\right)^{\max(3,k)}\right) \\ &\leq O\left(\sum_{d \geq 1} n^{\max(3,k)}\right) = O(n^{\max(4,k+1)}). \end{aligned}$$

Once the number of linear extensions of each of the P_T, $T \in I(P)$, has been computed, we combine these numbers into the number of linear extensions of P by recursing on the nodes of the Gallai decomposition. Indeed, if a node is *series-type*, then the number of linear extensions of the corresponding module is simply the product of the number of linear extensions of its children; if a node is *parallel-type* with children of cardinalities $m_1, \ldots, m_u$, then the number of linear extensions of the corresponding module is the product of the number of linear extensions of its children and the quantity

$$\binom{m_1 + \cdots + m_u}{m_1, \ldots, m_u}.$$

If the node is indecomposable-type, the number of linear extensions of $P|_T$ is the number of linear extensions of P_T times the number of linear extensions of all $P|_S$,

where S is a child node of T. It is easy to check that the numerical computations involved require at most $O(n^{\max(4,k+1)})$ time, so one can compute the total number of linear extensions of P in this amount of time. □

Proof of Theorem 2.10. This also follows from the main result of [5]. If we let R be any poset, P a chain of length $|R|$, and Q the lattice of down-sets of R, then the number of injective occurrences of P in Q is precisely the number of linear extensions of R, which is #**P**-hard to compute. □

Note that counting the number of injective occurrences of a chain in an arbitrary text poset of bounded dimension is in **FP**, because the standard dynamic programming algorithm for counting increasing subsequences of permutation readily generalizes to arbitrary dimension and executes in polynomial time.

4 Conclusion and Problems

There remains a plethora of open questions about poset pattern occurrence. First, if $\dim(P) = 2$ and Q is a chain, is the problem of counting the number of (not necessarily induced, injective) occurrences of P in Q #**P**-hard? In the language of permutations, this is the computational problem of counting, for a given permutation σ, how many permutations of the same size have all of the inversions of σ (and possibly others as well). The problem is also equivalent to asking whether counting linear extensions of dimension-2 posets is hard, a question left open by [5] because the posets whose number of linear extensions the authors compute grow in dimension without bound. Indeed, their gadget $Q_I(p)$ contains as an induced subposet a bipartite poset whose *upper set* is the family of clauses that occur in the 3-SAT instance I and whose *lower set* is the family of variables occurring in I, with an edge between a clause and variable precisely when the clause contains a literal which is the positive or negative of the variable. If we choose the clauses to be all possible disjunctions of two variables out of n (repeating the second to ensure three literals), the resulting subposet is exactly the subposet of the Boolean poset between the doubletons and singletons; Spencer showed ([13]), via a result of Dushnik, that this *Boolean layer* poset has dimension $\Omega(\log \log n)$, and therefore in particular tends to infinity.

What are the complexity of poset pattern recognition problems for other parameter settings than those considered here? What problems/results from the substantial literature on permutation pattern avoidance generalize in an interesting way to not necessarily induced occurrences? Is it #**P**-hard to compute the number of unlabeled dimension-k posets of cardinality n avoiding (i.e., not containing any occurrence of) given patterns, especially, in the case when $k = 2$? Perhaps more important is the closely related question: is it #**P**-hard to compute the number of permutations of n which avoid a given pattern? Given the apparent difficulty of computing the number of pattern-avoiding permutations for various special patterns (1324 being a prominent example), it is natural to suspect that this problem is computationally hard in general.

References

[1] S. Arora and B. Barak, *Computational complexity: A modern approach*, Cambridge University Press, Cambridge, 2009.

[2] H. Buer and R. H. Möhring, A fast algorithm for the decomposition of graphs and posets, *Math. Oper. Res.* **8** (1983), no. 2, 170–184.

[3] P. Bose, J. Bussxi and A. Lubiw, Pattern matching for permutations, *Inf. Proc. Let.* **65** (1998), 277–283.

[4] B. I. Bayoumi, M. H. El-Zahar and S. M. Khamis, Counting two-dimensional posets, *Discrete Math.* **131** (1994), no. 1–3, 29–37.

[5] G. Brightwell and P. Winkler, Counting linear extensions, *Order* **8** (1991), no. 3, 225–242.

[6] B. Dushnik and E. W. Miller, Partially ordered sets, *Amer. J. Math.* **63** (1941), 600–610.

[7] P. Ille and J.-X. Rampon, A counting of the minimal realizations of the posets of dimension two, *Ars Combin.* **78** (2006), 157–165.

[8] T. Gallai, Transitiv orientierbare Graphen, *Acta Math. Acad. Sci. Hungar.* **18** (1967), 25–66.

[9] H. Kierstead, Effective versions of the chain decomposition theorem: The Dilworth theorems, 36–38, *Contemp. Mathematicians*, Birkhäuser Boston, Boston, MA, 1990.

[10] N. M. Korneenko, R. I. Tyshkevich and V. N. Zemlyachenko, The graph isomorphism problem: The theory of the complexity of computations, I, *Zap. Nauchn. Sem. Leningrad. Otdel. Mat. Inst. Steklov. (LOMI)* **118** (1982), 83–158, 215.

[11] R. Mathon, A note on the graph isomorphism counting problem, *Inform. Process. Lett.* **8** (1979), no. 3, 131–132.

[12] R. M. McConnell and J. P. Spinrad, Linear-time modular decomposition and efficient transitive orientation of comparability graphs, *Proc. Fifth Annual ACM-SIAM Symposium on Discrete Algorithms* (1994), 536–545.

[13] J. Spencer, Minimal scrambling sets of simple orders, *Acta Math. Acad. Sci. Hungar.* **22** (1971/72), 349–353.

[14] G. Steiner, Polynomial algorithms to count linear extensions in certain posetsr, *Congr. Numer.* **75** (1990), 71–90.

(Received 15 Sep 2014; revised 11 May 2015)