Maximum k-Chains in Planar Point Sets: Combinatorial Structure and Algorithms*

Stefan Felsner † and Lorenz Wernisch‡
felsner@inf.fu-berlin.de wernisch@inf.fu-berlin.de

Freie Universität Berlin,
Fachbereich Mathematik, Institut für Informatik,
Takustraße 9, 14195 Berlin,
Germany.

Abstract

A chain of a set P of n points in the plane is a chain of the dominance order on P. A k-chain is a subset C of P that can be covered by k chains. A k-chain C is a *maximum k-chain* if no other k-chain contains more elements than C. This paper deals with the problem of finding a maximum k-chain of P in the cardinality and in the weighted case.

Using the skeleton $S(P)$ of a point set P introduced by Viennot we describe a fairly simple algorithm that computes maximum k-chains in time $O(kn \log n)$ and linear space. The basic idea is that the canonical chain partition of a maximum $(k-1)$-chain in the skeleton $S(P)$ provides k regions in the plane, such that a maximum k-chain for P can be obtained as the union of a maximal chain from each of these regions.

By the symmetry between chains and antichains in the dominance order we may use the algorithm for maximum k-chains to compute maximum k-antichains for planar points in time $O(kn \log n)$. However, for large k one can do better. We describe an algorithm computing maximum k-antichains (and, by symmetry, k-chains) in time $O((n^2/k) \log n)$ and linear space. Consequently, a maximum k-chain can be computed in time $O(n^{3/2} \log n)$, for arbitrary k.

The background for the algorithms is a geometric approach to the Greene-Kleitman theory for permutations. We include a skeleton based exposition of this theory and give some hints on connections with the theory of Young tableaux.

The concept of the skeleton of a planar point set is extended to the case of a weighted point set. This extension allows to compute maximum weighted k-chains with an algorithm that is similar to the algorithm for the cardinality case. The time and space requirements of the algorithm for weighted k-chains are $O(2^k n \log(2^k n))$ and $O(2^k n)$ respectively.

Key Words: Algorithms, antichains, chains, orders, point sets, skeletons, Young tableaux.

Subject Classification: 68Q25, 65Y25, 06A07, 05C85.

---

*A preliminary version of this article appeared in *Proc. 25. Ann. ACM Symp. on the Theory of Computing*, pages 146–153, 1993.

†Stefan Felsner has been supported by the Deutsche Forschungsgemeinschaft under grant FE 340/2-1.

‡Lorenz Wernisch has been supported by a grant from the German-Israeli Binational Science Foundation and by the ESPRIT Basic Research Action Program of the EC under project ALCOM II.

1 Introduction

The *dominance order* on points in the plane is given by the relations $p \leq q$ if $p_x \leq q_x$ and $p_y \leq q_y$. Here and throughout the paper p_x and p_y denote the x- and y-coordinates of a point p. The symbol P will always be an n-element set of points in the plane together with the dominance relation. A subset C of P is a *chain* if any two members p, q of C are comparable, i.e., either $p \leq q$ or $q \leq p$. On the other hand, a set $A \subseteq P$ with no two different points comparable is an *antichain*. If a subset C of P can be covered by k chains it is called a *k-chain*. If C is a k-chain but not a $(k-1)$-chain we call C a *strict* k-chain. A k-chain C is *maximum* if no other k-chain contains more elements than C. This paper deals with the problem of finding such k-chains in P. Note that the "greedy method" that repeatedly removes maximum chains may fail in computing a maximum k-chain even for $k = 2$ (see e.g. the point set of Figure 1).

A permutation $\sigma: \{1, \ldots, n\} \rightarrow \{1, \ldots, n\}$ may be represented by points $(i, \sigma(i))$ in the plane. Chains and antichains of a point set correspond to increasing and decreasing subsequences of a permutation. Hence, finding maximum k-chains amounts to computing maximum k increasing subsequences. Fredman [3] shows that finding a maximum increasing subsequence requires $\Omega(n \log n)$ comparisons. Of course, this also gives a lower bound for k-chains, $k \geq 2$. On the other hand, an algorithm to compute a longest increasing subsequence in a permutation in time $O(n \log n)$ pertains to mathematicians folklore. A careful treatment of the algorithm can be found in [3], older sources are e.g. [1] or [8].

Interest in k-chains of orders goes back to Greene and Kleitman [7, 6] who discovered a rich duality between maximum k-chains and maximum ℓ-antichains. From this theory we quote a theorem relating maximum k-chains to maximum ℓ-antichains.

Theorem 1 *For an order P with n elements there exists a partition α of n, such that the Ferrers diagram F_α of α has the following properties:*
(1) *The number of squares in the k longest rows of F_α equals the size of a maximum k-chain, for $1 \leq k \leq n$.*
(2) *The number of squares in the ℓ longest columns of F_α equals the size of a maximum ℓ-antichain, for $1 \leq \ell \leq n$.*

The history of algorithms for maximum k-chains seems to start in some of the many alternative proofs for the Greene-Kleitman Theorems, we mention two of these approaches. In [13] Viennot deals with the case of permutations or point sets respectively and indicates how to find k-chains in time $O((n^2/k) \log n)$. For general orders Frank [2] uses network flows which results in algorithms to compute maximum k-chains in an arbitrary order in time $O(n^3)$. Gavril [4] uses a network designed specifically for k-chain computations and improves the time bound to $O(kn^2)$. Gavril's approach was adapted to handle the weighted case within the same complexity by Sarrafzadeh and Lou [12]. For the case of planar point sets Lou, Sarrafzadeh, and Lee [10, 9] propose algorithms to compute 2- and 3-chains in optimal time $O(n \log n)$. They are motivated to consider k-chains in planar point sets by problems in VLSI design, e.g., multi-layered via minimization for two-sided channels. Maximum k-chains also turn out to be useful in computational geometry, e.g., for counting points in triangles (see [11]).

We describe a fairly simple method to find maximum k-chains for arbitrary k in time $O(kn \log n)$ and linear space. Our approach is based on the useful concept of the *skeleton*

of P introduced by Viennot [14] (see also [13]). We use a maximum $(k-1)$-chain in the skeleton to partition the plane into k regions. Taking a maximum chain from each of the regions already yields a maximum k-chain. Our method leads to a kind of complementary algorithm to the $O((n^2/k)\log n)$ algorithm of Viennot.

In Section 2 the notion of skeletons is introduced. We give an algorithm to compute them and show that a point set P is determined by the skeleton $S(P)$ and two additional chains of *marginal points*. The notion of skeletons leads to a geometric interpretation of the well known bijective correspondence between permutations and pairs of Young tableaux (the Robinson-Schensted correspondence, see, e.g., [8]). The section ends with a brief exposition of this connection.

Section 3 starts with the development of the combinatorial background for the algorithm. The algorithm is described in fairly detailed pseudocode and its correctness is proved. As a byproduct we provide a direct geometric proof for part (1) of Theorem 1 for permutations. Section 4 is devoted to a complete presentation of Viennot's $O((n^2/k)\log n)$ algorithm for k-antichains. A byproduct of the analysis is the direct geometric proof for part (2) of Theorem 1 for permutations. The constructions from Section 3 and Section 4 both imply a result of Greene [5] stating that the shape of the Ferrers diagram F_α in Theorem 1 is just the shape of the Young tableaux corresponding to the permutation.

In Section 5 we extend the concept of skeletons to the case where a real weight $w(p)$ is associated with each point p in P. The algorithm of Section 3 is extended to work with weighted planar point sets and weighted skeletons. This yields a maximum weighted k-chain in time $O(2^k n \log(2^k n))$ and space $O(2^k n)$. Of course, this makes sense only for small values of k. But note that even for constant k no better algorithm than that of Sarrafzadeh and Lou [12] with running time $O(kn^2)$ was known. Unfortunately, it is not obvious how to extend the algorithm of Section 4 computing a maximum k-antichain in a similar way to the weighted case.

2 Skeleton and Young tableaux

Let P be our planar point set with n elements. We will always assume that the points of P are in *general position*, i.e., no two points have the same x- or y-coordinate. Arguments are simpler if we assume this generality. In the case of duplicate coordinates we can perturb the points such that they are in general position without changing the comparability relations. Simply change the values of the x-coordinate of points with the same x-coordinate—by definition they are comparable—by a small amount such they get increasing x-coordinates with increasing y-coordinates. Points with the same y-coordinate are perturbed analogously. Such perturbations can be made in a single sweep, i.e., in time $O(n \log n)$. As is easily seen, the complexity of all algorithms in this paper remains as claimed even if we make such a sweep whenever we start working with a new set of points.

The *height* of a point $p \in P$ is the size of a longest chain with p as maximal element. Of course, two points of the same height cannot be comparable. Hence, collecting points with the same height in the same set yields a partition $\mathcal{A}$ of P into antichains, the *canonical antichain partition*. Observe that this partition is also obtained by a repeated removal of the set of minimal elements. By definition, the number of antichains in a canonical partition is the size of a largest chain in P, a *maximum chain* (see Fig. 1). Since, obviously,

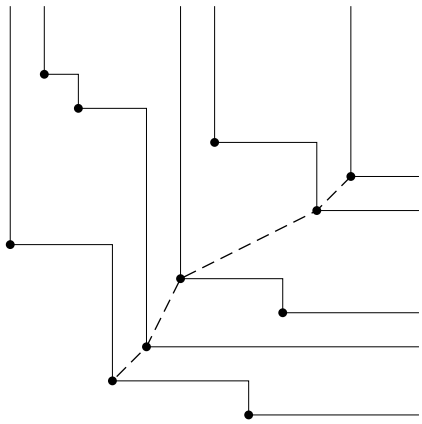

Figure 1: The canonical antichain partition and a maximum chain.

a chain and an antichain can have at most one point in common, there can be no partition into fewer antichains than there are in $\mathcal{A}$, i.e., it is a *minimal antichain partition.*

Following Viennot, we define the *left shadow of point* p as the set of all points (u, v) *dominating* p, i.e., with $u \geq p_x$ and $v \geq p_y$. For a set E of points, the *shadow of* E is the union of the shadows of the points of E, i.e., the set of all points q dominating at least one point of E. The *right shadow* of p is the set of all points (u, v) with $u \leq p_x$ and $v \geq p_y$. The term shadow suggests some light coming from the left below, or from the right below in the case of a right shadow (of course, this should not be taken too verbally, since the "shadows" have a form that is scarcely realizable physically). The *right down shadow* of p is the set of all points (u, v) with $u \leq p_x$ and $v \leq p_y$. The right and right down shadows of a set E are again defined as the union of the corresponding shadows of the points of E.

The *left jump line* or simply *jump line*, $L_{\nearrow}(E)$ or $L(E)$, of a point set E is the topological boundary of the left shadow of E. The *right jump line* $L_{\nwarrow}(E)$ and the *right down jump line* $L_{\swarrow}$ of E are the topological boundaries of the right and the right down shadow of E. Let the unbounded half line of the jump line extending upwards be the *top outgoing line*, and let the unbounded half line extending to the right be the *right outgoing line.* Additionally, we use the term *left outgoing line* when dealing with right or right down jump lines. It is easily seen that the jump line $L(A)$ of an antichain A is a downward staircase with the points of A in its lower corners. Collect the points in the upper corners of $L(A)$ in the set $S_{\nearrow}(A) = S(A)$ this is the set of *skeleton points* or briefly the *skeleton* of the antichain A. Formally, if $(x_1, y_1), \ldots, (x_k, y_k)$ are the points of A ordered by increasing x-coordinate then $S(A)$ consists of the points $(x_2, y_1), \ldots, (x_k, y_{k-1})$. Hence, $L(A)$ has exactly $|A| - 1$ skeleton points (see Fig. 2).

The minimal elements of a point set P form an antichain A such that the rest $P - A$ lies completely in the shadow of A. Hence, by removing A and treating $P - A$ in the same way, we recursively obtain the canonical antichain partition $A = A_0, \ldots, A_{\lambda-1}$ with nonintersecting jump lines $L(A_i)$, $0 \leq i < \lambda$, which will be called the *layers* $L_i(P)$ *of* P.

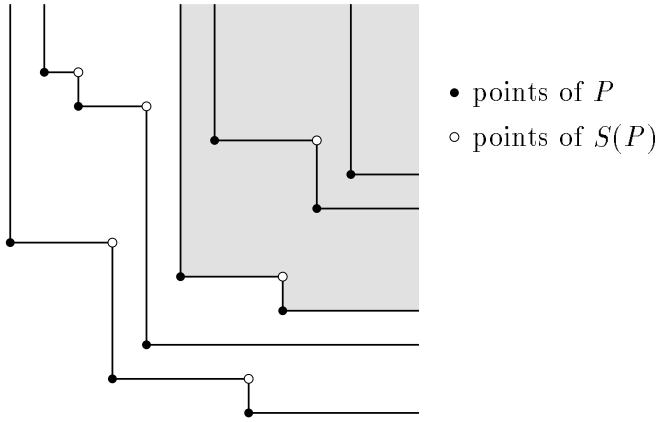

Figure 2: Point set P, its skeleton, and the shadow of 3rd layer.

The *skeleton* or *left skeleton of* P, denoted by $S(P)$ or $S_{\nearrow}(P)$, is then defined as the union of the skeletons $S(A_i)$, $0 \leq i < \lambda$. Since, as noted above, the i-th layer $L_i(P)$ has $|A_i| - 1$ skeleton points, the size of $S(P)$ is $|P| - \lambda$. A picture of a point set P, its skeleton $S(P)$, its antichain layer partition, and the shadow of antichain 2 can be found in Fig. 2. Let us state an easy but quite useful observation.

Lemma 1 *Suppose a point set P is partitioned into k antichains A_i in such a way that the jump lines $L(A_i)$ are pairwise disjoint. Then $A_1, \ldots, A_k$ is the canonical antichain partition of P.*

Proof. Suppose the A_i are ordered by increasing x-coordinates of the top outgoing lines of the $L(A_i)$. Then $P - A_0$ is in the shadow of A_0. Hence, by the definition of the shadow, each point of $P - A_0$ dominates at least one of A_0 and no point of A_0 dominates any other point in P, i.e., A_0 are just the minimal elements of P. Repeat the procedure on $P - A_0$. □

Let us describe a simple algorithm to compute the skeleton, a maximum chain, and the canonical antichain partition of a point set P (see Fig. 3). Essentially, this is the well-known algorithm for longest increasing sequences of permutations (for a geometrically inspired version see [10]). A sweep line L going from left to right halts at every point of P. It contains an ordered set of markers m. A marker m on L has a y-coordinate m_y and m is said to be above a point p if $m_y > p_y$.

Suppose L halts at some point p and the layers have been constructed for all points to the left of L. Find the next layer with right outgoing line above p. If there is no such layer (i.e., the marker found equals the dummy point), open a new one with p as its (yet) sole point. If there is one, add p to this layer and generate a new skeleton point. It is easily seen that the jump lines thus constructed cannot intersect and hence are the layers of a canonical antichain partition, by Lemma 1. Finally, a maximum chain is obtained by

```
SKELETON(P)
insert dummy d_u in L at height +∞; link(d_u) ← nil;
k ← 0; S ← ∅;
for each p ∈ P from left to right do
    insert a new marker m' in L with m'_y ← p_y;
    point(m') ← p;
    m ← next marker below p on L;
    link(p) ← point(m);
    m ← next marker above p on L;
    if m = d_u then
        A_k ← {p}; antichain(m) ← A_k; k ← k + 1;
    else
        add v to antichain(m);
        add skeleton point (L_x, m_y) to S;
        remove m from L;
v ← point(m), m uppermost marker on L; C ← {v};
while link(v) ≠ nil do
    v ← link(v); C ← C ∪ {v};
return S, C, A_0, ..., A_{k-1};
```

Figure 3: Algorithm SKELETON

extracting a point from each of the antichains along a chain of properly established links. With L implemented as a dynamic binary tree we have

Theorem 2 *Algorithm* SKELETON *computes the skeleton, a maximum chain, and the canonical antichain partition of a point set P of size n in time $O(n\log(n))$ and linear space.* □

For the definition of the *right skeleton* $S_{\diagdown}(P)$ use the right shadow and the right jump lines. And for the *right down skeleton* $S_{\diagup}(P)$ use right down shadow and right down jump lines. Of course, with $S_{\diagdown}(P)$ we obtain the canonical chain partition instead of the antichain partition. By symmetry, a lemma corresponding to Lemma 1 but dealing with chains instead of antichains is again true. With $S_{\diagup}(P)$ we again obtain an antichain partition. A layer of this partition contains all the points with the same dual height (the dual height of p is the length of a maximum chain that has p as its minimal element).

It is convenient to conceive the construction of the skeleton as an operator on finite point sets consisting of points in general position, since the points of the skeleton $S(P)$ again have pairwise different x- and y-coordinates. Thus, we may apply operators $S_{\diagup}$ and $S_{\diagdown}$ to $S(P)$. As usual, the k-fold iteration of an operator O will be denoted by O^k; O^0 means identity. $S^k(P)$ will be called the *k-th skeleton* of P. An interesting algebraic property of $S_{\diagup}$ and $S_{\diagdown}$, they are commutative, is shown in [15].

One of the properties that seem to lie behind the usefulness of skeletons is the fact that it is possible to reconstruct P from $S(P)$ with a small amount of additional information. Let $x_{\max}$ be the maximal x-coordinate of points in P, and let $y_{\max}$ be defined analogously. Then the *right marginal points* $M_R(P)$ of P are the points $(x_{\max}+1, y_1), \ldots, (x_{\max}+\lambda, y_\lambda)$,

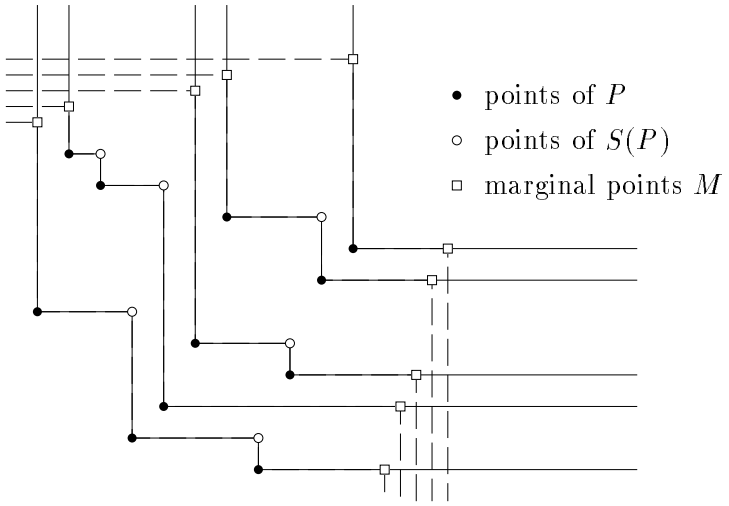

Figure 4: P is the right down skeleton of $S(P) \cup M(P)$.

where λ is the number of layers of P and $y_1, \ldots, y_\lambda$ are the y-coordinates of the right outgoing lines of the layers ordered increasingly (see Fig. 4). Assuming $x_1, \ldots, x_\lambda$ to be the x-coordinates of the top outgoing lines of the layers in increasing order the *top marginal points* $M_T(P)$ of P are $(x_1, y_{\max} + 1), \ldots, (x_\lambda, y_{\max} + \lambda)$ (see Fig. 4). Note that each of $M_R(P)$ and $M_T(P)$ is a chain of length height of P. With $M(P)$ we denote the collection of marginal points of P, i.e., $M(P) = M_R(P) \cup M_T(P)$.

Theorem 3 *A point set P is the right down skeleton of the skeleton $S(P)$ together with the marginal points of P, i.e., $P = S_{\swarrow}(\, S(P) \cup M(P)\,)$.*

Proof. The jump line $L(A)$ of an antichain A nearly coincides with the right down jump line $L_1 = L_{\swarrow}(S(A \cup \{s,t\}))$ of the skeleton of A with an arbitrary point s somewhere on the top outgoing line of $L(A)$ and some point t on the right outgoing line. More precisely, between s and t the two jump lines are equal. As the marginal points on the top and right outgoing lines are chosen so that they form chains, we may bend the original jump lines of P at the marginal points and the bended lines remain nonintersecting. Each bended line L is the right down jump line of the points of $S(P) \cup M(P)$ contained in it. Moreover, the points of $S(P) \cup M(P)$ contained in each of these lines form an antichain and each point is on one of these lines. Hence, the right down version of Lemma 1 applies and we are done. □

A *partition* of an integer n is a sequence of integers $\lambda_0 \geq \lambda_1 \geq \cdots \geq \lambda_{\mu-1} > 0$ such that $n = \lambda_0 + \cdots \lambda_{\mu-1}$. Such a partition may be represented graphically by a Ferrer's diagram also called *Young shape*. This is a shape as that of the two figures in Fig. 6, which consists of μ rows of rectangles or *cells* with λ_i cells in row i, when rows are taken from bottom to top (also called "French notation"). If numbers[1] are put in these cells in increasing order

[1] In the classical theory these are the numbers $1 \ldots n$ which gives a correspondence between pairs of Young tableaux and permutations. In the present context it is convenient to allow real entries.

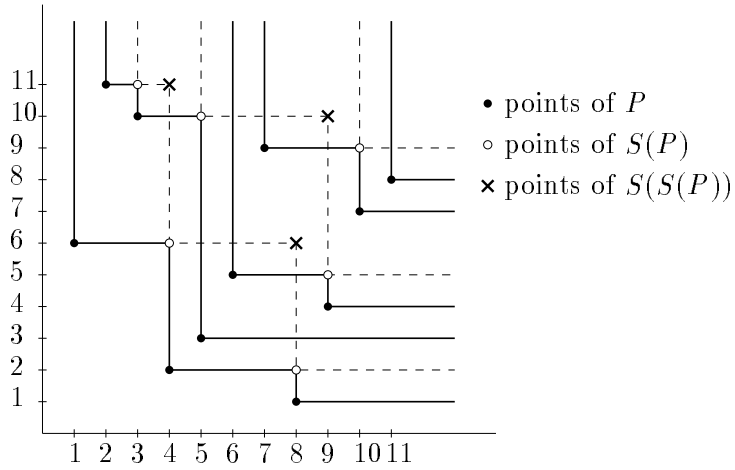

Figure 5: The first two layers $L_1(P)$ and $L_2(P)$ of P.

from left to right and from bottom to top we obtain a *Young tableau*. If $\mathbf{Y}$ is a Young tableau we denote the cell in the i-th column from the right and j-th row from below by $\mathbf{Y}(i,j)$, with $i,j \geq 0$.

Since we will often refer to the number of layers of P, let us adopt the following notation. Let $\mu(P)$ be minimal with $S^{\mu(P)}(P) = \emptyset$. Then $\lambda_i(P)$, $0 \leq i < \mu(P)$, denotes the number of layers of $S^i(P)$. It is convenient to assume $\lambda_i(P) = 0$ for $i \geq \mu(P)$.

Lemma 2 *Let P be a point set then $\lambda_0(P) \geq \lambda_1(P) \geq \cdots \geq \lambda_{\mu(P)-1} > 0$, and $|S^k(P)| = \sum_{k \leq i < \mu} \lambda_i(P)$, where $\mu = \mu(P)$. In particular $\lambda_0(P), \lambda_1(P), \ldots, \lambda_{\mu(P)-1} > 0$ is a partition of n.*

Proof. By Theorem 3, the number of antichains in a minimal antichain partition of $S(P) \cup M(P)$ is the same as $\lambda_0(P)$, the size of the canonical antichain partition of P. Hence, $\lambda_1(P)$, the size of a minimal antichain partition of $S(P)$ is at most $\lambda_0(P)$. The same argument shows the other inequalities. The sum over the $\lambda_i(P)$ is computed easily by using $|S(P)| = |P| - \lambda_0(P)$ and induction. □

Let P be a planar set of n points. We associate two tableaux $\mathbf{P}(P)$ and $\mathbf{Q}(P)$ (the $\mathbf{P}$- and $\mathbf{Q}$-*symbol of* P) with P in the following way. The k-th row of $\mathbf{P}(P)$, $k \geq 0$, are the y-coordinates of the right outgoing lines of $S^k(P)$ in increasing order. The k-th row of $\mathbf{Q}(P)$, $k \geq 0$, are the x-coordinates of the top outgoing lines of $S^k(P)$ in increasing order. Compare the outgoing lines of the first two layers of Fig. 5 with the first two rows of the Young tableaux in Fig. 6. According to Lemma 2, $\mathbf{P}(P)$ and $\mathbf{Q}(P)$ have $\lambda_i(P)$ cells in their i-th row from below and $|P|$ cells altogether. Hence, the shape of the tableaux $\mathbf{P}(P)$ and $\mathbf{Q}(P)$ is a Young shape. We denote the number of cells in the i-th column (from left) with $\lambda_i^*(P)$. Obviously, $\lambda_0^*(P) \geq \lambda_1^*(P) \geq \cdots \geq \lambda_{\ell-1}^*(P)$ and $\sum_{0 \leq i < \ell} \lambda_i^*(P) = |P|$, where $\ell = \lambda_0(P)$. The $\lambda_i^*(P)$ are the *conjugate partition* of the $\lambda_i(P)$ for the integer $|P|$.

Our first observation about the $\mathbf{P}$- and $\mathbf{Q}$-symbol concerns the *inverse* P^{-1} *of* P, which is the point set that is obtained from P by the transposition $(x,y) \to (y,x)$, i.e., by

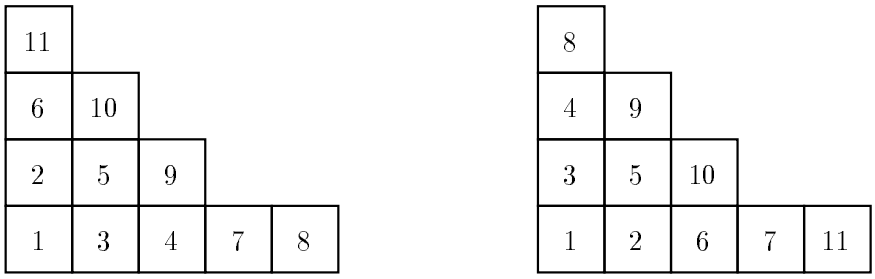

Figure 6: The **P**- and **Q**-symbol of point set P of Fig. 5.

reflection on the diagonal line $x = y$. Obviously, the corresponding **P**- and **Q**-symbols are simply interchanged.

Theorem 4 *For the inverse P^{-1} of a point set P, $\mathbf{P}(P^{-1}) = \mathbf{Q}(P)$ and $\mathbf{Q}(P^{-1}) = \mathbf{P}(P)$.* □

Theorem 5 (Robinson-Schensted) *The tableaux $\mathbf{P}(P)$ and $\mathbf{Q}(P)$ of a point set P are Young tableaux. Moreover, for any two Young tableaux $\mathbf{P}$ and $\mathbf{Q}$ with the same shape there exists a point set P with $\mathbf{P} = \mathbf{P}(P)$ and $\mathbf{Q} = \mathbf{Q}(P)$, i.e., there is a bijection between point sets and pairs of Young tableau with the same shape.*

The reader interested in the proof of this theorem and in a more comprehensive treatment of geometric approaches to the theory of Young tableaux is referred to Viennot [14] and Wernisch [15].

3 Maximum k-chains

Suppose a subset C_S of the skeleton $S(P)$ of a planar point set P is given and let $C_1, \ldots, C_k$ be the canonical chain partition of C_S. The existence of such a chain partition implies that C_S is a k-chain (the converse is false, a k-chain can have a partition into fewer chains). We define the *i-th region of C_S*, for $2 \leq i \leq k$, to be the intersection of the right shadow of C_{i-1} with the complement of the right shadow of C_i, i.e., the region between the jump lines of C_{i-1} and C_i, containing the jump line of C_{i-1} but excluding that of C_i (see Fig. 7). The *first region* is the complement of the right shadow of C_1 and the *$(k+1)$-st region* is the right shadow of C_k. These $k+1$ regions partition the whole plane.

A description of the main steps of an algorithm computing k-chains can be given with this concept of the region (a more detailed description can be found in Fig. 9). The reader may want to visualize the following steps on Fig. 7.

1. Compute the skeleton $S(P)$ of a planar point set P.
2. Compute recursively a $(k-1)$-chain C_{k-1} of $S(P)$.
3. For all regions R defined by C_{k-1}, extract a maximum chain from P intersected with R. The union of all chains is a maximum k-chain for P.

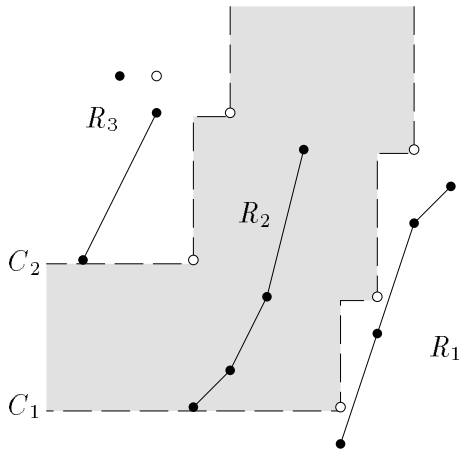

Figure 7: Three regions defined by a maximum 2-chain of the skeleton each containing one chain.

To demonstrate the correctness of the approach we need another definition. An antichain A of P and its jump line are said to *cross region R well* if $R \cap L(A) = R \cap L(A \cap R)$. That is, A crosses R well exactly if the jump line $L(A)$ enters R vertically and leaves R horizontally (see Fig. 8).

The next lemma expresses the key property of well crossing antichains that makes them useful for our purposes.

Lemma 3 *Let R be a region of $C_S \subseteq S(P)$ and let $A_1, \ldots, A_\lambda$ be the canonical antichain partition of P. If I is the set of all indices i such that A_i is crossing R well, then the collection $\{A_i \cap R \mid i \in I\}$ is the canonical antichain partition of the underlying point set $\bigcup_{i \in I} A_i \cap R$.*

Proof. Let R be the ℓ-th region and let $p = L(A_i) \cap L_\diagdown(C_\ell)$ and $p' = L(A_j) \cap L_\diagdown(C_\ell)$ with $i < j$. Note that any two points on $L_\diagdown(C_\ell)$ are comparable. Since $L(A_j)$ is in the shadow of $L(A_i)$ we cannot have $p' \leq p$, hence $p \leq p'$. Since A_i is crossing R well, the line segment of $L(A_i \cap R)$ that ends in p is vertical. The same holds for $L(A_j \cap R)$ and p'. As the two jump lines are disjoint we obtain $p_x < p'_x$. Therefore, we can extend $L(A_i \cap R)$ and $L(A_j \cap R)$ by vertical half lines without introducing an intersection.

A similar argument shows that the y-coordinates of $q = L(A_i) \cap L_\diagdown(C_{\ell-1})$ and $q' = L(A_j) \cap L_\diagdown(C_{\ell-1})$ are related by $q_y < q'_y$. Hence, the corresponding right half lines do not cross. Altogether the jump lines $L(A_i \cup R)$ are pairwise disjoint and, by Lemma 1, $\{A_i \cap R \mid i \in I\}$ is the canonical antichain partition of the underlying set. □

Lemma 4 *Let $C_S \subseteq S(P)$ and let A be an antichain of the canonical partition of P. If m is the number of skeleton points on A that are in C_S then the number of regions of C_S crossed well by A is at least $m + 1$.*

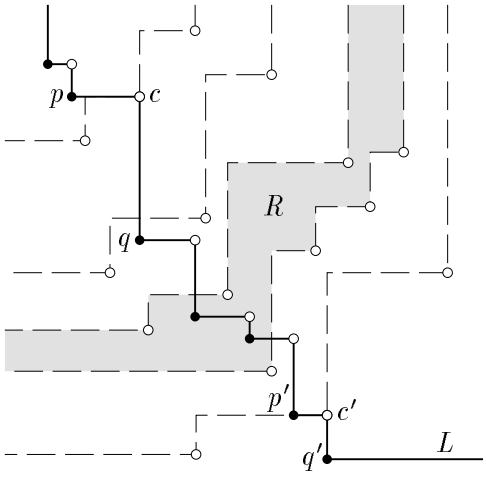

Figure 8: L crosses some region R well in the section between c and c'.

Proof. Let c be a point of C_S on the jump line L of A. Let p and q be the points of A to the left and below c that define it (see Fig. 8). Then it is obvious that L leaves the region containing p horizontally and enters that of q vertically. Of course, the top outgoing line of L is vertical and the right outgoing line is horizontal.

Note that if L leaves one region vertically the region of the next point of A to the right is entered vertically too. Now consider the $m+1$ sections of L from left to right before, between, and after its m points in C_S (possibly m equals 0). Since in each such section A enters its first region vertically and leaves its last region horizontally, there must be some region crossed well by A in between. □

Lemma 5 *Let C_S be a $(k-1)$-chain in the skeleton $S(P)$ of P and let λ be the height of P. Taking a maximum chain of $P \cap R$ in each region R of C_S yields a k-chain of P of size at least $|C_S| + \lambda$.*

Proof. Let $A_1, \ldots, A_\lambda$ be the canonical antichain partition of P. Each point of C_S is a skeleton point of exactly one antichain A_i. If m_i is the number of skeleton points of A_i in C_S, for $1 \leq i \leq \lambda$, then, according to Lemma 4, the antichains A_i cross the regions well in altogether at least $\sum_{1 \leq i \leq \lambda}(m_i + 1) = |C_S| + \lambda$ sections. On the other hand, by Lemma 3, each such section crossing well some region contributes one more point to the maximum chain of that region. □

Note that Lemma 5 does not require the $(k-1)$-chain C_S to be maximum. We now show a kind of reverse to Lemma 5.

Lemma 6 *Let C be a k-chain in P. There exists a $(k-1)$-chain in the skeleton $S(P)$ with size at least $|C| - \lambda$, where λ is the height of P.*

Proof. By Theorem 3, P is the right down skeleton of $S(P) \cup M(P)$. The chains $M_R(P)$ and $M_T(P)$ of marginal points both have size λ. Hence, we may reason similarly as in

the proof of Lemma 5 after reflection of all points on the diagonal $x = -y$, i.e., after the transformation $T(x, y) = (-x, -y)$ of the plane. The right down skeleton P is thus transformed to a skeleton $T(P)$ of $T(S(P) \cup M(P))$. The image $T(C)$ of the k-chain $C \subseteq P$ is a k-chain in the skeleton $T(P)$ defining $k + 1$ regions. Observe that $T(M_R)$ lies in the first region and $T(M_T)$ lies in the $(k + 1)$-st region of $T(C)$ and that both are maximum chains of length λ. We apply Lemma 5 and obtain a $(k + 2)$-chain C'_S in $T(S(P) \cup M(P))$ with $|C'_S| \geq |C| + \lambda$. According to the above observation, we may further assume that $T(M_R)$ and $T(M_T)$ are in C_S. When these two chains are removed from C'_S we obtain a $(k-1)$-chain $C_S \subseteq T(S(P))$ of size at least $|C| - \lambda$ and $T(C_S) \subseteq S(P)$ is the $(k-1)$-chain searched for. □

We denote the k-chain in P obtained from a $(k-1)$-chain C_S in $S(P)$ according to Lemma 5 by $\sigma_k(C_S)$.

Theorem 6 *Let C_1 be a maximum chain of $S^{k-1}(P)$ and $C_i = \sigma_i(C_{i-1})$, for $2 \leq i \leq k$, then C_k is a maximum k-chain in P.*

Proof. By induction, suppose that C_{k-1} is a maximum $(k-1)$-chain in $S(P)$ and let λ be the height of P. If there were a k-chain $C \subseteq P$ with more points than $\sigma_k(C_{k-1})$ then C would have more than $|C_{k-1}| + \lambda$ points, according to Lemma 5. Hence, by Lemma 6, there would be a $(k-1)$-chain of size larger than $|C_{k-1}|$ in $S(P)$, a contradiction. □

Note that the proof of the above theorem also shows that the number of additional points in each application of σ_ℓ to a maximum $(\ell - 1)$-chain of $S^{k-\ell+1}(P)$ is equal to the height of $S^{k-\ell}(P)$, for $2 \leq \ell \leq k$. Hence, we obtain the following corollary that is part (1) of Greene's Theorem for permutations (see Theorem 1).

Corollary 1 *A maximum k-chain of a point set P has size $\sum_{0 \leq i < k} \lambda_i(P)$ where $\lambda_i(P)$ is the height of $S^i(P)$.* □

We are now prepared to provide an algorithm MAXMULTICHAIN (see Fig. 9) that, given a point set P and some k, computes a maximum k-chain of P. Some remarks about this algorithm are in order. To dispose P means to release any memory space holding the points of P, which is necessary to keep the space requirement small. Recall that a maximum $(k-1)$-chain may consist of fewer than $k-1$ chains. This happens if C_S equals S and can be partitioned into less than $k-1$ chains. Hence, there may be fewer than k regions of C_S. If C_S is empty (e.g., when $k = 1$), we assume that R_1 is the whole plane.

The partitioning of P according to the regions R_i of C_S can be done with a single sweep from left to right halting at every point of P. The sweep line L contains its intersection with all the right layers of C_S and is initialized to the y-coordinates of the left outgoing lines of these layers. Now a point $p \in P$ is easily assigned to its region. If the skeleton point immediately above p is in C_S then the height of the intersection point of the corresponding right layer with the sweep line has to be adapted.

Theorem 7 *Algorithm* MAXMULTICHAIN(P, k) *computes a maximum k-chain for a point set P of size n in time $O(kn \log n)$ and linear space.*

```
MAXMULTICHAIN(P, k)
C_S ← ∅; C ← ∅;
if k ≥ 2 then
    S ← skeleton(P);
    M ← marginal points(P);
    dispose P;
    C_S ← MAXMULTICHAIN(S, k − 1);
    P ← right down skeleton(S ∪ M);
R_1, ..., R_l ← regions(C_S);
partition P into P_i ← P ∩ R_i, 1 ≤ i ≤ l;
for i ← 1 to l do
    C ← C ∪ maximum chain(P_i);
return C;
```

Figure 9: Algorithm MAXMULTICHAIN

Proof. According to Theorem 2, the skeleton, marginal points, and the canonical chain partition of a point set of size n can be computed in $O(n \log n)$ time. The partitioning of P described above takes the same amount of time. The computation of the maximum chains in each region take, again by Theorem 2, time $O(\sum_{i=1}^{l} |P_i| \log(|P_i|))$ which is $O(n \log n)$. Since these estimations hold true in each recursive step, we have an overall time $O(kn \log n)$.

As far as the space requirement is concerned, the main problem is the computation of a new skeleton in each recursive step. But P is disposed and only its skeleton together with the marginal points is retained. The number of marginal points in the i-th step equals twice the number $\lambda_i(P)$ of layers of the i-th skeleton $S^i(P)$, $0 \leq i \leq k$. Thus, the amount of space that is needed for k recursions is $O(2 \sum_{i=0}^{k} \lambda_i(P) + |S^k(P)|)$ which, by Lemma 2, is $O(|P|)$. □

4 Maximum k-antichains

In this section we prove some assertions made by Viennot [13] leading to an algorithm that efficiently computes k-antichains of a point set P.

Let A_S be a k-antichain in the skeleton $S(P)$ and let $A_{S,1}, \ldots, A_{S,k}$ be the canonical antichain partition of A_S. It is easily seen that the intersection of P with the right down jump line $L_{\swarrow}(A_{S,i})$ is an antichain in P. Let us denote the k-antichain $\bigcup_{i=1}^{k} L_{\swarrow}(A_{S,i}) \cap P$ of P by $\Delta(A_S)$ (see Fig. 10). On the other hand, given a k-antichain A with canonical partition $A_1, \ldots, A_k$ we define the k-antichain $\delta(A) = \bigcup_{i=1}^{k} L(A_i) \cap S(P)$ of $S(P)$. Recall that a strict k-antichain is one that cannot be covered by less than k antichains.

Lemma 7 *Let P be a planar point set.*

1. *If A_S is a strict k-antichain of $S(P)$ then $\Delta(A_S)$ is a k-antichain of P of size at least $|A_S| + k$.*

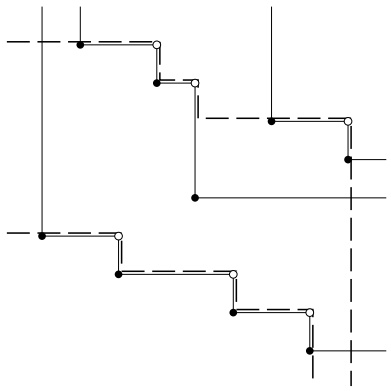

Figure 10: The Δ-operator.

2. *If A is a strict k-antichain of P then $\delta(A)$ is a k-antichain of $S(P)$ of size at least $|A| - k$.*
3. *If A_S is a strict maximum k-antichain then equality holds in 1 and $\Delta(A_S)$ is a strict maximum k-antichain, too.*

Proof. Let $A_{S,1}, \ldots, A_{S,k}$ be the canonical right down antichain partition of A_S. Each skeleton point $s \in S(P)$ has two defining points $p_L(s), p_D(s) \in P$, one with the same y-coordinate to the left, the other with the same x-coordinate below. If we walk along a right down jump line $L_{\swarrow}(A_{S,i})$ from left to right we find between any two consecutive skeleton points s_1, s_2 of $A_{S,i}$ at least one of the defining points $p_D(s_1)$ or $p_L(s_2)$ on the jump line. Otherwise, the defining point $p_D(s_1)$ would have y-coordinate smaller than that of s_2 and point $p_L(s_2)$ would have x-coordinate smaller than that of s_1. But this implies that two layers of the canonical layer structure of P intersect, which is impossible (see Fig. 11). Since the left and down defining point $p_L(s_L)$ and $p_D(s_R)$ of the leftmost and rightmost skeleton points s_L and s_R of $A_{S,i}$ are always on the right down jump line, $P \cap L_{\swarrow}(A_{S,i})$ contains at least one point more than $A_{S,i}$. Summing over all $A_{S,i}$ we get the first inequality of the lemma.

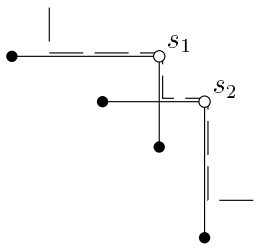

Figure 11: Intersecting layers.

The second inequality is obtained similarly. One may extend P by the two marginal chains and use the transformation $T(x, y) = (-x, -y)$. By Theorem 3, $T(P)$ is the skeleton of $T(S(P) \cup M(P))$. Hence, by the above argument, $|\delta(A)| \geq |\Delta(T(A))| - 2k \geq |T(A)| - k$ since the two chains $T(M_T)$ and $T(M_R)$ contribute at most $2k$ points to the k-antichain $\Delta(T(A))$ in $T(S(P) \cup M(P))$.

Suppose A_S is a strict maximum k-antichain and let $\Delta(A_S)$ be a strict k'-antichain with $k' \leq k$. Then $\delta(\Delta(A_S))$ is a k'-antichain, hence k-antichain, of size at least $|\Delta(A_S)| - k' \geq |A_S| + k - k'$ in $S(P)$ and $k' = k$ since A_S is maximum. Consequently, if A' is a k-antichain of P, $|\Delta(A_S)| - k \geq |A_S| \geq \delta(A') \geq |A'| - k$, which implies that $\Delta(A_S)$ is a maximum k-antichain. □

```
MAXMULTIANTICHAIN(P, k)
λ ← number of layers of (P);
if λ ≤ k then
    return (P, λ);
else
    S ← skeleton(P);
    M ← marginal points(P);
    dispose P;
    (A_S, λ') ← MAXMULTIANTICHAIN(S, k);
    P ← right down skeleton(S ∪ M);
    A ← Δ(A_S);
    if λ' < k then
        add k − λ' arbitrary points of P to A;
    return (A, k);
```

Figure 12: Algorithm MAXMULTIANTICHAIN

Theorem 8 *Let P be a point set and let $k \leq \lambda_0(P)$. There is an ℓ with $0 \leq \ell < \mu(P)$ and $\lambda_{\ell+1}(P) \leq k < \lambda_\ell(P)$. The ℓ-th skeleton $S^\ell(P)$ contains a strict maximum k-antichain A_ℓ and $\Delta^\ell(A_\ell)$ is a maximum k-antichain of P of size $|S^{\ell+1}(P)| + k\ell = \sum_{i=0}^{k} \lambda_i^*(P)$.*

Proof. Since, by Lemma 2, the $\lambda_i(P)$ are decreasing in i and $\lambda_{\mu(P)}(P) = 0$, there is an ℓ such that the inequalities are satisfied. $S^{\ell+1}(P)$ itself is a strict maximum $\lambda_{\ell+1}(P)$-antichain and $\Delta(S^{\ell+1}(P))$ is a strict maximum $\lambda_{\ell+1}(P)$-antichain of $S^\ell(P)$. The size of $\Delta(S^{\ell+1}(P))$ is $|S^{\ell+1}(P)| + \lambda_{\ell+1}(P)$ by Lemma 7. Take a maximum chain C in $S^\ell(P)$; it has size $\lambda_\ell(P)$. Since $\Delta(S^{\ell+1}(P))$ intersects C in at most $\lambda_{\ell+1}(P)$ points and $S^\ell(P)$ has size $|S^{\ell+1}(P)| + \lambda_\ell(P)$, there is no point of $S^\ell(P)$ outside $C \cup \Delta(S^{\ell+1}(P))$ and we may add $k - \lambda_{\ell+1}(P)$ arbitrary points of $S^\ell(P)$ to $\Delta(S^{\ell+1}(P))$ to get a strict maximum k-antichain A_ℓ in $S^\ell(P)$ of size $|S^{\ell+1}(P)| + k$. Now induction and application of Lemma 7 shows the theorem. □

An algorithm computing a maximum k-antichain of P for given P and k is now easy to provide. For algorithm MAXMULTIANTICHAIN see Fig. 12.

Theorem 9 *Algorithm* MAXMULTIANTICHAIN(P, k) *computes a maximum k-antichain for a point set P of size n in time $O((n^2/k)\log n)$ and linear space.*

Proof. Since the algorithm simply mimics the proof of Theorem 8, it certainly computes a maximum k-antichain. The computation of the skeleton, marginal points, and the number of layers takes time $O(n \log n)$. The Δ operator is implemented straightforwardly. For a sweep line going from right to left computing the right down layers of A_S may halt at points of P, too, and check whether they lie on a layer or not. Hence, the time needed for one recursive step is $O(n \log n)$. According to Theorem 8, a maximum k-antichain has size $|S^{\ell+1}| + k\ell \leq n$. Thus, the number $\ell + 1$ of recursions is bounded by $n/k + 1$. That the amount of space needed remains linear is seen as in the proof of Theorem 7. □

5 Maximum weighted k-chains

Given some weight $w\colon P \to \mathbf{R}$ on the points of a set P, we define the weight of a k-chain as the sum of the weights of its points. A *maximum weighted* k-chain has maximum weight among all weights of k-chains of P. Such maximum weighted k-chains can be found in a similar way as maximum k-chains. Unfortunately, the corresponding algorithm is efficient only if k is small.

In the following assume that the weights w are positive integers. With this assumption the weighted case can be simulated by the unweighted one. Though the algorithms work on the weighted point set itself the proofs of correctness are based on the following idea. We expand each point $p \in P$ into a tiny chain $C(p)$ of $w(p)$ points, where tiny means that the chain is contained within a tiny box of sidelength ϵ where ϵ is less than the minimum distance in x- or y-coordinates of the points of P (recall that we assume all points to have different x- and y-coordinates). Denote the expanded set of points by P'. Now consider the skeleton $S(P')$ of P'. Let p, q be two points of P with $p_x < q_x$. It is easily seen that if there are any skeleton points in $S(P')$ having their defining points in the tiny chains $C(p)$ and $C(q)$ then all skeleton points with this property again fit into a box B of sidelength ϵ. In this case, locate a *weighted skeleton point* between p and q (i.e., at (p_x, q_y)) and give it a weight equal to the number of skeleton points contained in box B. The resulting set of weighted skeleton points is the *weighted skeleton* $S(P, w)$.

The weighted skeleton $S(P, w)$ can also be obtained without resorting to set P' of multiplied points. We translate the actions of Algorithm SKELETON (see Fig. 3 of Section 2) that constructs the skeleton $S(P')$ of P' with sweep line L' into actions of an Algorithm SKELETON-WEIGHTED (see Fig. 13) that constructs the corresponding weighted skeleton $S(P, w)$ of P with sweep line L. As a byproduct Algorithm SKELETON-WEIGHTED also computes a maximum weighted chain of P.

If the y-coordinates of a set M of markers on L' differ by an amount smaller than ϵ then they correspond to a *weighted marker* m on L with weight $W(m) = |M|$. The insertion of a point $p \in P$ with weight $w(p)$ in L corresponds to the $w(p)$ insertions of points from $C(p) \subseteq P'$ into L'. Let m be the next marker above p. If $|C(p)|$ is greater or equal to the number $W(m)$ of markers on L' that correspond to m then $W(m)$ new skeleton points in $S(P')$ are generated. Hence, we have to generate a new skeleton point in $S(P, w)$ of weight $W(m)$ and remove marker m. If $|C(p)| > W(m)$ there remain $W = |C(p)| - W(m)$ points of $C(p)$ for insertion. Hence, the next marker on L is searched and the procedure is repeated until there is no further marker on L or there is one marker m_0 with $W(m_0) > W$. Comparing with the corresponding situation in P' we find the necessary action. A skeleton point of weight W is generated and the weight of marker m_0 is updated to $W(m_0) - W$. With this kind of considerations it can be verified that Algorithm SKELETON-WEIGHTED yields the weighted skeleton of (P, w).

For the maximum weighted chain computation observe that either all or none of the points of $C(p)$, for some $p \in P$, are contained in a maximum chain in P'. Thus, a maximum chain in P' corresponds to a maximum weighted chain in P and vice versa. As will be seen later the same holds true of maximum weighted k-chains in P and maximum k-chains in P'.

The weighted skeleton thus computed has many points with equal x- or y-coordinate and we want to compute the weighted skeleton of a skeleton repeatedly. Thus, we perturb

SKELETON-WEIGHTED(P, w)
insert dummy d_u in L with $W(d_u) \leftarrow +\infty$ at height $+\infty$;
$link(d_u) \leftarrow$ nil;
$S \leftarrow \emptyset$;
for each $p \in P$ from left to right **do**
    insert a new marker m' in L with $m'_y \leftarrow p_y$;
    $W(m') \leftarrow w(p)$;
    $point(m') \leftarrow p$;
    $m \leftarrow$ next marker below p on L;
    $link(p) \leftarrow point(m)$;
    $m \leftarrow$ next marker above p on L;
    $W \leftarrow w(p)$;
    while $W \geq W(m)$ **do**
        add skeleton point (L_x, m_y) with weight $W(m)$ to S;
        $W \leftarrow W - W(m)$;
        remove m from L;
        $m \leftarrow$ next marker above p on L;
    if $m \neq d_u$ and $W > 0$ **then**
        add skeleton point (L_x, m_y) with weight W to S;
        $W(m) \leftarrow W(m) - W$;
$v \leftarrow point(m)$, m uppermost marker on L; $C \leftarrow \{v\}$;
while $link(v) \neq$ nil **do**
    $v \leftarrow link(v)$; $C \leftarrow C \cup \{v\}$;
return S, C;

Figure 13: Algorithm SKELETON-WEIGHTED

them according to the simple procedure mentioned in section 2 before we use them in any further computation. Consequently, we may assume that all coordinates of points of the input instance are different.

Computing the weighted skeleton $S(P, w)$ with algorithm SKELETON-WEIGHTED takes time $O((|P| + |S(P, w)|)\log(|P| + |S(P, w)|))$. In contrast to the unweighted case, the weighted skeleton may contain more points than the original point set. Fortunately, there cannot be much more such points.

Lemma 8 *The number of weighted skeleton points in $S(P, w)$ is at most twice the number of points in P.*

Proof. If a skeleton point $s \in S(P, w)$ has no other skeleton point s' above it is assigned to its defining point $p \in P$ below. Otherwise, it is assigned to its defining point to the left. Note that in the second case skeleton point s was generated under the condition $W \geq W(m)$, hence, m was removed and, consequently, there is no skeleton point s' to the right of s. A point $p \in P$ gets assigned the highest skeleton point above it or the furthest skeleton point to its right or both. Therefore, no point p gets assigned more than two skeleton points. □

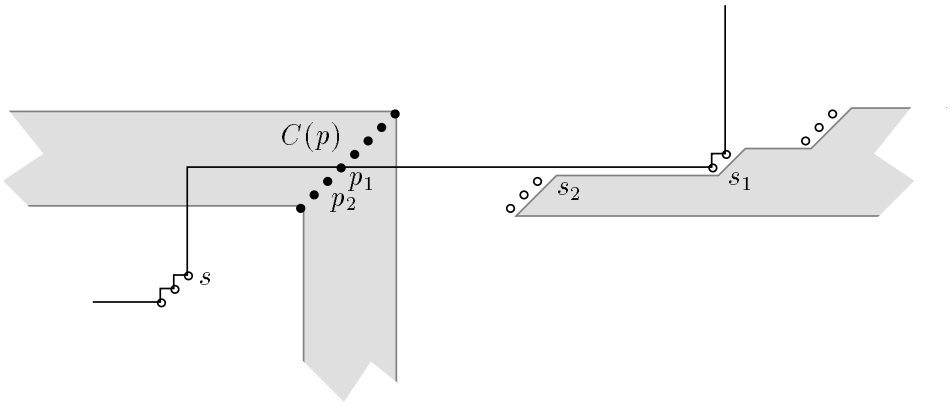

Figure 14: This cannot happen, a jump line of C_S separating $C(p)$. Gray areas do not contain points of P' or $S(P')$.

The interesting fact now is that Algorithm MAXMULTICHAIN (see Fig. 9 of Section 3) may be used nearly without changes to compute a maximum weighted k-chain. The only problem is that we have not yet defined what the marginal points of a weighted point set should be. But we can do without marginal points if we resign to dispose any set of points. This does no harm since the space requirement will be high anyway.

Theorem 10 *Let C_S be a maximum weighted $(k-1)$-chain in $S(P,w)$. Then taking a maximum weighted chain of points $P \cap R$ in each of the k regions R of C_S yields a maximum weighted k-chain of P.*

Proof. By induction, a maximum weighted $(k-1)$-chain C_S of $S(P)$ corresponds to a maximum $(k-1)$-chain C'_S of the skeleton $S(P')$ of P'. The selection of a maximum chain in the point sets $P' \cap R$ for each region R of the k regions of C'_S would give a maximum k-chain of P'. This does not immediately yield a maximum weighted k-chain of P since some points of the chain $C(p) \subseteq P'$ replacing a point p of P might fall in one region and some in another. We claim that this cannot happen. Since there is no skeleton point in the ϵ box containing $C(p)$, it is separated either by a vertical or horizontal segment of a jump line of some chain. It cannot be vertical, for there is no skeleton point below $C(p)$.

Now suppose that a horizontal segment between two skeleton points s and s_1 separates $C(p)$ (see Fig. 14). The y-coordinate of s_1 equals the y-coordinate of some point $p_1 \in C(p)$. Let p_2 be be the immediate predecessor of p_1 in $C(p)$. Since there is no skeleton point to the left of $C(p)$ the point s has to be dominated by p_2 and by s_2, the skeleton point on the outgoing line of p_2. Also $s_2 < s_1$, hence, $C_S \cup \{s_2\}$ is a $(k-1)$-chain. If s_2 is not a member of C_S this contradicts the maximality of C_S. On the other hand, s_2 cannot be a member of C_S, since C_S has a decomposition into $(k-1)$ noncrossing jump lines, one of them joining s and s_1. A jump line containing s_2 however has to leave s_2 upwards and hence crosses the jump line of s and s_1.

We have thus proved that there is a correspondence between the regions R'_i of C'_S and the regions R_i of C_S, for $1 \le i \le k$, in such a way that a chain $C(p)$ is completely contained in R'_i iff p is contained in R_i. Applying Theorem 6 to the expanded point set,

we obtain that all maximum weighted chains from regions R_i together form a maximum weighted k-chain. □

In analogy to the unweighted case, we set $S_w^0(P) = P$ and let $S_w^k(P) = S(S_w^{k-1}(P), w)$ be the *k-th weighted skeleton* of P. We denote the weighted k-chain in P obtained from a weighted $(k-1)$-chain C_S in $S(P,w)$ according to Theorem 10 by $\rho_k(C_S)$.

Theorem 11 *Let C_1 be a maximum weighted chain of $S_w^{k-1}(P)$ and $C_\ell = \rho_\ell(C_{\ell-1})$, for $2 \leq \ell \leq k$, then C_k is a maximum weighted k-chain in P.* □

Theorem 12 *A maximum weighted k-chain of a weighted point set P can be obtained in $O(2^k|P|\log(2^k|P|))$ time and $O(2^k|P|)$ space.*

Proof. As was already pointed out, to construct the weighted skeleton for a weighted point set P takes time $O(|P|\log|P|)$. The point location of points P in the regions defined by a $(k-1)$-chain C_S can be done with the help of a sweep line in time $O(|P|\log|C_S| + |C_S|))$. This amounts to a total running time of $O(\sum_{1\leq i\leq k} 2^i|P|\log(2^i|P|))$, since $|S_w^i| \leq 2^i|P|$ by Lemma 8. The bound on the space is given by $O(\sum_{1\leq i\leq k} 2^i|P|)$. □

A maximum weighted k-chain of a point set P with rational or real weights can be computed by the very same algorithm. The proof of correctness then requires some additional standard rescaling and approximation arguments.

As the algorithm of this section makes sense only for small values of k it would have been nice to have a complementary method for large k. Unfortunately, we have not been able to extend the algorithm of Section 4 to the weighted case so that the running time remains independent from the weights.

References

[1] S. Even, A. Pnueli, and A. Lempel. Permutation graphs and transitive graphs. *J. of the ACM*, 19:400–410, 1972.

[2] A. Frank. On chain and antichain families of partially ordered sets. *J. Comb. Th. (B)*, 29:176–184, 1980.

[3] M. L. Fredman. On computing the length of longest increasing subsequences. *Discr. Math.*, 11:29–35, 1975.

[4] F. Gavril. Algorithms for maximum k-colorings and maximum k-coverings of transitive graphs. *Networks*, 17:465–470, 1987.

[5] C. Greene. An extension of Schensted's theorem. *Adv. in Math.*, 14:254–265, 1974.

[6] C. Greene. Some partitions associated with a partially ordered set. *J. Comb. Th. (A)*, 20:69–79, 1976.

[7] C. Greene and D. J. Kleitman. The structure of sperner k-families. *J. Comb. Th. (A)*, 20:41–68, 1976.

[8] D. E. Knuth. *The Art of Computer Programming III. Sorting and Searching*, volume 3. Addison-Wesley, 1973.

[9] R. D. Lou and M. Sarrafzadeh. An optimal algorithm for the maximum 3-chain problem. *SIAM J. on Comp.*, 22:976–993, 1993.

[10] R. D. Lou, M. Sarrafzadeh, and D. T. Lee. An optimal algorithm for the maximum two-chain problem. *SIAM J. Disc. Math.*, 5:285–304, 1992.

[11] J. Matoušek and E. Welzl. Good splitters for counting points in triangles. *J. of Algorithms*, 13:307–319, 1992.

[12] M. Sarrafzadeh and R. D. Lou. Maximum k-covering of weighted transitive graphs with applications. *Algorithmica*, 9:84–100, 1993.

[13] G. Viennot. Une forme géométrique de la correspondance de Robinson-Schensted. In D. Foata, editor, *Combinatoire et Représentation du Groupe Symétrique*, pages 29–58. Lecture Notes in Mathematics 579, Berlin Heidelberg New-York, 1977.

[14] G. Viennot. Chain and antichain families, grids and young tableaux. In *Orders: Description and Roles*, pages 409–463. North-Holland Math. Stud. 99, Amsterdam-New York, 1984.

[15] L. Wernisch. Dominance relation on point sets and aligned rectangles. *Dissertation*, Freie Universität Berlin, 1994.