Fast RSK Correspondence by Doubling Search

Alexander Tiskin
Department of Mathematics and Computer Science, St. Petersburg State University, Russia

Abstract

The Robinson–Schensted-Knuth (RSK) correspondence is a fundamental concept in combinatorics and representation theory. It is defined as a certain bijection between permutations and pairs of Young tableaux of a given order. We consider the RSK correspondence as an algorithmic problem, along with the closely related k-chain problem. We give a simple, direct description of the symmetric RSK algorithm, which is implied by the k-chain algorithms of Viennot and of Felsner and Wernisch. We also show how the doubling search of Bentley and Yao can be used as a subroutine by the symmetric RSK algorithm, replacing the default binary search. Surprisingly, such a straightforward replacement improves the asymptotic worst-case running time for the RSK correspondence that has been best known since 1998. A similar improvement also holds for the average running time of RSK on uniformly random permutations.

2012 ACM Subject Classification Mathematics of computing → Permutations and combinations

Keywords and phrases combinatorics of permutations, Robinson–Schensted–Knuth correspondence, k-chains, RSK algorithm

Digital Object Identifier 10.4230/LIPIcs.ESA.2022.86

Funding This work was supported by the Russian Science Foundation under grant no. 22-21-00669, `https://www.rscf.ru/en/project/22-21-00669/`.

Acknowledgements I thank Nikolay Vasilyev, Vasilii Duzhin and Artem Kuzmin for advice and fruitful discussions.

1 Introduction

The Robinson–Schensted-Knuth (RSK) correspondence is a fundamental concept in combinatorics and representation theory; for the background on the combinatorial aspects of RSK, see e.g. [20, 17]. It is defined as a certain bijection between pairs of standard Young tableaux and permutations of a given order, and represents a far-reaching generalisation of the longest increasing subsequence problem in a permutation. A common definition of RSK correspondence is algorithmic, via Robinson–Schensted tableau insertions or, alternatively, via the Viennot geometric construction.

The combinatorial properties of RSK are well-studied. In this paper, we consider the RSK correspondence as an algorithmic problem, along with the closely related k-chain problem. In particular, we are interested in both the worst-case and the average asymptotic running time of algorithms for these problems. This aspect of the RSK correspondence seems to have been studied relatively less thoroughly than its combinatorial aspects.

In the rest of this paper, we recall the definition of the RSK correspondence, using the geometric construction of Viennot [23, 24]. We then describe the standard RSK algorithm by Robinson [15] and Schensted [18]. Further, we give a simple, direct decription of the symmetric RSK algorithm, which is implied by the k-chain algorithms of Viennot [24] and of Felsner and Wernisch [9]. Next, we recall the doubling search algorithm of Bentley and Yao [1], and show how it can be used as a subroutine by the symmetric RSK algorithm, replacing the default binary search. Surprisingly, such a straightforward replacement improves the asymptotic worst-case running time for the RSK correspondence from $O(n^{3/2} \log n)$, which has been the best known since [9], to $O(n^{3/2})$. A similar improvement also holds for the average running time of RSK on uniformly random permutations.

30th Annual European Symposium on Algorithms (ESA 2022).
Editors: Shiri Chechik, Gonzalo Navarro, Eva Rotenberg, and Grzegorz Herman; Article No. 86; pp. 86:1–86:10
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

2 The RSK correspondence

Partial orders. We will use the standard terminology related to partial orders: *downset*, *principal downset*, *chain*, *antichain*. We consider two mutually inverse (strict) total orders on $\mathbb{R}$: $<$ and $>$. We also consider the corresponding (strict) partial *dominance* orders on $\mathbb{R}^2$: $\ll$, $\lessgtr$, $\gtrless$, $\gg$, where $(x, y) \ll (x', y')$ if $x < x'$ and $y < y'$, and similarly for the other three orders. Dominance orders $\ll$, $\gg$ are mutually inverse, and so are $\lessgtr$, $\gtrless$. When considering a point set P as a partial order, we will indicate it by a superscript, e.g. $P^{\ll}$. We will always assume that P is finite, and that all x-coordinates in P are distinct, and so are all the y-coordinates.

Young tableaux. Let $\mathbb{N}_+$ denote the set of all positive integers. Given $n \in \mathbb{N}_+$, let $\mathbb{N}_n = \{1, \ldots, n\} \subset \mathbb{N}_+$.

▶ **Definition 1.** *A* Young diagram *of order n is a subset of $\mathbb{N}_+^2$ of cardinality n, that is a downset in the dominance partial order $\ll$. A* Young tableau[1] *of order n is an order-preserving bijection from a Young diagram of order n (called the tableau's* shape*) to a subset of $\mathbb{R}$ with total order $<$.*

We use the so-called French notation for visual representation of Young diagrams and tableaux. The elements of a diagram are represented by cells of an integer grid, arranged in left-aligned rows and bottom-aligned columns. Columns are ordered from left to right, and rows from below upwards. The value of each cell of a tableau is written within that cell; these values increase from left to right in rows, and from below upwards in columns.

▶ **Example 2.** Figure 1 (middle and right columns) gives several examples of Young tableaux with cell values in $\mathbb{N}_{10}$.

Canonical antichain partitioning. The theory of Young tableaux is intimately connected with the combinatorics of permutations. We take a symmetric view of this connection, due to Viennot [23, 24]. A permutation, viewed as a mapping $\pi : \mathbb{N}_n \to \mathbb{N}_n$, is identified with the mapping's graph, i.e. the point set $P_\pi = \{(x, \pi(x)) \mid x \in \mathbb{N}_n\}$.

▶ **Definition 3.** *The* height *of an element in a finite partial order O is the maximum cardinality of a chain in the principal downset generated by that element. A* canonical antichain *is formed by all the elements of a given height. The partitioning of O into disjoint canonical antichains is called the* canonical antichain partition (CAP)*, denoted $cap(O)$.*

Canonical antichains in $\mathbb{R}^2$ are also sometimes called *layers of minima* (*maxima*) [4, 3], *Pareto fronts* [5], or *terraces* [14]. The canonical antichain partition of a point set in $\mathbb{R}^2$ is also sometimes called *greedy cover* [12], *patience sorting* [2], or *non-dominated sorting* [5].

▶ **Example 4.** Figure 1 (top-left) shows a point set P of cardinality 10, and its partitioning $cap(P^{\ll})$ into five antichains.

We recall the following standard result.

▶ **Proposition 5.** *The partitioning $cap(O)$ of a finite partial order has the minimum possible number of antichains among all antichain partitionings of O. This number is also equal to the maximum cardinality of a chain in O.*

Proof. Straightforward; see e.g. [9]. ◀

[1] Young tableaux as defined here are often called "standard", to distinguish them from more general types of tableaux; we omit this qualifier, since it is the only type of Young tableaux we are dealing with.

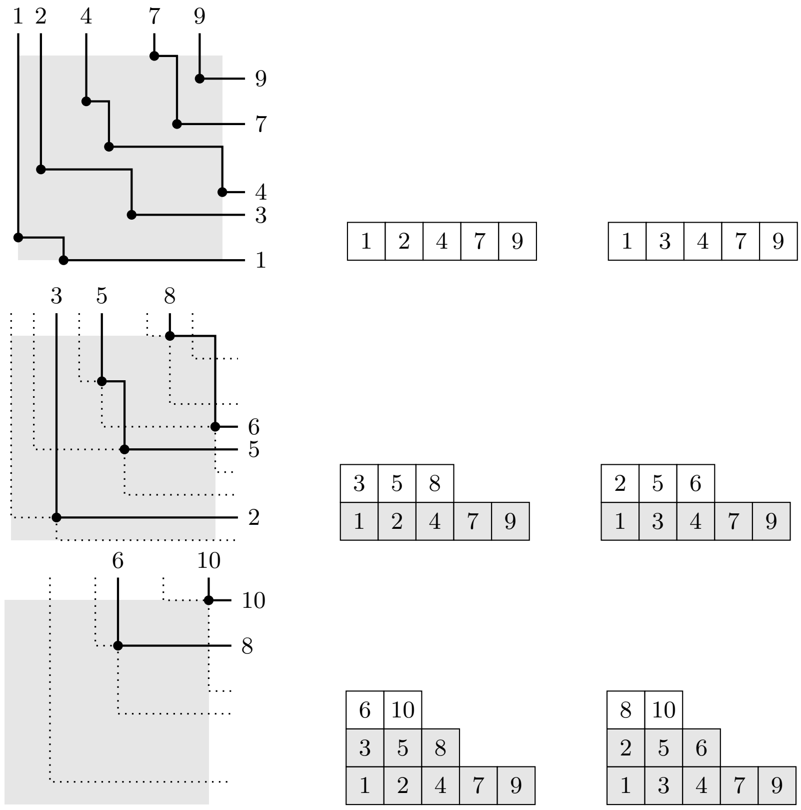

Figure 1 The standard RSK algorithm for $rsk(P^{\ll}) = (H, T)$; tableaux H, T obtained by rows.

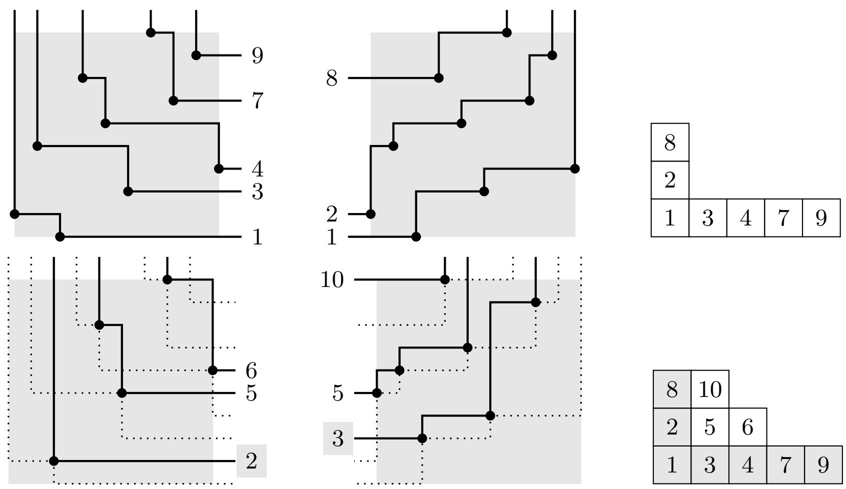

Figure 2 The symmetric RSK algorithm for $rsk(P^{\ll}) = (\_, T)$ and $rsk(P^{\gtrless}) = (\_, T^{\dagger})$: tableau T obtained by principal hooks.

Given a permutation π, the problem of finding the cardinality of $cap(P_\pi^{\lll})$ is equivalent to the problem of finding the length of a longest increasing subsequence (LIS) in π. The LIS problem has a long history, going back to Erdös and Szekeres [8] and Robinson [15]. Based on their ideas, the classical LIS algorithm running in time $O(n \log n)$ was made explicit by Knuth [13], Fredman [10] and Dijkstra [6].

▶ **Definition 6.** *Let P be a point set with dominance order $\lll$. Let $A = \{(x_1, y_1) \lessgtr (x_2, y_2) \lessgtr \ldots \lessgtr (x_r, y_r)\} \subseteq P$ be an antichain of cardinality $r \geq 1$. Values x_1 and y_r will be called respectively the* head *and the* tail *of A. The* skeleton *of A is the antichain $sk(A) = \{(x_2, y_1) \lessgtr (x_3, y_2) \lessgtr \ldots \lessgtr (x_r, y_{r-1})\}$ of cardinality $r - 1$. The* skeleton *of P is the point set $sk(P^{\lll}) = \bigcup_{A \in cap(P^{\lll})} sk(A)$. The heads, tails and skeletons with respect to dominance orders $\lessgtr$, $\gtrless$, $\ggg$ are defined analogously.*

The RSK correspondence. The Robinson–Schensted–Knuth (RSK) correspondence, discovered independently by Robinson [15] and Schensted [18] (see also Romik [17]), is a bijection between permutations of a given order and pairs of Young tableaux of the same order and identical shape. The two tableaux in the pair will be called the *head* and the *tail* tableaux (such a terminology is chosen for its symmetry and consistency with the rest of our exposition, whereas the traditional terminology calls them the *recording* and the *insertion* tableaux).

▶ **Definition 7.** *Let P be a point set with dominance order $\lll$. The* RSK image[2] *of P is a pair of Young tableaux $rsk(P^{\lll}) = (H, T)$, defined recursively as follows. The initial row in H (respectively, T) is formed by the heads (respectively, the tails) of the antichains in $cap(P^{\lll})$. The remaining rows of H, T are formed as $rsk(sk(P^{\lll}))$. The RSK image with respect to dominance orders $\lessgtr$, $\gtrless$, $\ggg$ is defined analogously.*

▶ **Example 8.** Figure 1 (top row) shows the construction of the initial rows in tableaux $rsk(P^{\lll}) = (H, T)$ from a point set $P \subseteq \mathbb{N}_{10}^2$ (top left). Figure 1 (middle and bottom rows) shows the recursive construction of the remaining rows in the tableaux H, T.

The RSK correspondence has some beautiful symmetries, exposed by Schützenberger [19].

▶ **Definition 9.** *Let P be a point set. Its* transpose *is the point set $P^\dagger = \{(x, y) \mid (y, x) \in P\}$ obtained by exchanging the x- and y-coordinates of every point.*

▶ **Observation 10.** *Let π be a permutation. We have $P_\pi^\dagger = P_{\pi^{-1}}$.*

▶ **Definition 11.** *Let Y be a Young diagram. Its* transpose *$Y^\dagger$ is the Young diagram obtained by exchanging the x- and y-coordinates of every cell.*

We now state the theorem by Schützenberger [19] on the symmetries of the RSK correspondence; for completeness, we also present its proof.

▶ **Theorem 12** (Schützenberger)**.** *Let P be a point set, $rsk(P^{\lll}) = (H, T)$. We have*

(i) $rsk(P^{\dagger\lll}) = (T, H)$,
(ii) $rsk(P^{\gtrless}) = (H^{*\dagger}, T^\dagger)$,
(iii) $rsk(P^{\lessgtr}) = (H^\dagger, T^{*\dagger})$,
(iv) $rsk(P^{\ggg}) = (H^*, T^*)$.

Here, H^*, T^* are Young diagrams of the same shape as H, T, called the Schützenberger dual *of H, T, respectively.*

---

[2] The terms "RSK correspondence", "RSK image" as defined here are often called just "Robinson–Schensted", reserving the name "RSK" for a more general type of combinatorial bijection. Since this is an algorithmic study, we use the term RSK throughout, in order to highlight the contribution of Donald Knuth to the development of RSK algorithms.

Proof. Part (i) is obvious by symmetry.

Let us establish part (ii). Let (x_0, y_0) be the point with the least y-coordinate in $sk(P^{\ll})$, so y_0 is the tail of the least-height antichain in $cap(sk(P^{\ll}))$. Let A be an antichain in $cap(P^{\ll})$ such that $(x_0, y_0) \in sk(A)$. Then, there is a pair of points $(x_0, y) \gtrless (x, y_0)$ in A. Let B be the subset of points in P with y-coordinate less than y_0. Subset B must consist of a single chain including point (x_0, y): otherwise, there would be two points with y-coordinate less than y_0 in some antichain A' of $cap(P^{\ll})$, and then $sk(A')$ would contain a point of $sk(P^{\ll})$ with y-coordinate less than y_0, which would contradict the minimality of y_0. Now consider the points of B with the partial order $\gtrless$; these points, including point (x_0, y), form a subset of the least height antichain in $cap(P^{\gtrless})$. Point (x, y_0) must belong to the second-least height antichain in $cap(P^{\gtrless})$, and must have the least y-coordinate in that antichain. Therefore, y_0 is the tail of the second-least height antichain in $cap(P^{\gtrless})$.

We have established that the tail of the least height antichain in $cap(sk(P^{\ll}))$ is equal to the tail of the second-least height antichain in $cap(P^{\gtrless})$. Now (ii) follows by (i) and the recursive construction of Definition 7, and (iii), (iv) follow from (ii) by symmetry. ◀

Multichains. A notion closely related to the subject of this paper is that of a k-chain.

▶ **Definition 13.** *Let O be a finite partial order. A k-chain is a subset of O that can be represented as a union of k chains.*

The connection between the RSK correspondence and k-chains is given by the classical theorem of Greene [11].

▶ **Theorem 14** (Greene). *Let P be a point set with dominance order $\ll$. The maximum cardinality of a k-chain in P is equal to the number of cells in the initial k rows of the shape of $rsk(P^{\ll})$.*

In fact, the RSK algorithm by Felsner and Wernisch [9] is presented entirely in the language of k-chains. While we use the language of Young tableaux in this paper, our results translate immediately into the corresponding statements on maximum k-chains in a set of points, and thus relate to the results of [9].

3 RSK algorithms

Standard RSK algorithm. Definition 7 leads directly to the following standard algorithm for computing the RSK image of a given point set.

Given a point set P, the pair of tableaux $rsk(P^{\ll}) = (H, T)$ are constructed by rows. To obtain the initial rows of H, T, the points in P are scanned in order of increasing x-coordinate. For the subset Q of points seen so far, we maintain the partitioning $cap(Q^{\ll})$; in particular, the heads and the tails of antichains in that partitioning are kept in sorted order. We also maintain the skeleton $sk(Q^{\ll})$ in order of increasing x-coordinate. When the scan of P is complete ($Q = P$), the heads (respectively, tails) of antichains in $cap(P^{\ll})$ become the initial row of tableau H (respectively, T) in $rsk(P^{\ll})$. To obtain the remaining rows of $rsk(P^{\ll})$, we repeat the above procedure on point set $sk(P^{\ll})$. Algorithm 1 gives the algorithm's pseudocode.

▶ **Example 15.** Figure 1 shows the execution of the standard RSK algorithm in three successive iterations: the point set at the beginning of each iteration and its CAP (left column), and the state of the tableaux H and T at the end of the respective iteration (middle and right columns).

Algorithm 1 Standard RSK. The choice of a search method in line 7 is either linear or binary (Section 3) or doubling (Section 4).

1: **procedure** RSK(P) ▷ given point set P sorted by x-coordinate, returns $rsk(P^{\ll})$
2: **if** $P = \emptyset$ **then return** $(\emptyset, \emptyset)$
3: $H_{init} \leftarrow \emptyset$; $T_{init} \leftarrow \emptyset$ ▷ initialise variables for initial rows of H, T
4: $S \leftarrow \emptyset$ ▷ initialise variable for $sk(P^{\ll})$
5: **while** $P \neq \emptyset$ **do**
6: $(x, y) \leftarrow$ point in P with least x-coordinate
7: $y' \leftarrow$ least value in T_{init} greater than y; $+\infty$ if none exists ▷ search
8: **if** $y' = +\infty$ **then**
9: append x to H_{init}; append y to T_{init} ▷ start new antichain
10: **else**
11: replace y' by y in T_{init}; append (x, y') to S ▷ extend antichain
12: remove (x, y) from P
13: $(H_+, T_+) \leftarrow$ RSK(S) ▷ recursive call
14: $H \leftarrow$ tableau with initial row H_{init} and remaining rows H_+
15: $T \leftarrow$ tableau with initial row T_{init} and remaining rows T_+
16: **return** (H, T) ▷ $rsk(P^{\ll}) = (H, T)$

The computation of the initial row in the standard RSK algorithm (before the recursive call in line 13 of Algorithm 1) is essentially identical to the classical algorithm for the LIS problem [13, 10, 6]. In line 7, the canonical antichain for each of the n points can be found by binary search, therefore the whole initial row is obtained in time $O(n \log n)$. In total, there are at most n rows in $rsk(P^{\ll})$, therefore the overall time is $n \cdot O(n \log n) = O(n^2 \log n)$.

Apart from the worst-case running time, it is of interest to consider the average-case running time of RSK algorithms on a uniformly random permutation; in this case, the shape of tableaux H, T turns out to be sampled from the *Plancherel* probability distribution (see, e.g. [17]). Romik [16] established this average-case running time to be $O(n^{3/2} \log n)$.

RSK with linear search. Paradoxically, a speedup can be obtained by replacing binary search with (a carefully controlled) linear search. Indeed, for a given x-coordinate, the value of the search target y' in line 7 of Algorithm 1 can only increase. Therefore, as long these different search invocations are performed as a linear search continuing from the search target of the previous invocation, the combined search time for a given x-coordinate will be $O(n)$, so the overall running time across all x-coordinates is reduced to $n \cdot O(n) = O(n^2)$. This observation may be considered part of the folklore; it is made e.g. by Thomas and Yong [21], who attribute it to an anonymous referee. A simple and elegant alternative description of this algorithm can be obtained by using edge local rules of Viennot [25], giving the same asymptotic running time $O(n^2)$.

Symmetric RSK algorithm. Felsner and Wernisch [9] proposed a more efficient, symmetric approach to developing an RSK algorithm. Their algorithm was described in the language of k-chains. In particular, they gave an algorithm for computing maximum k-chains (and, by symmetry, also k-antichains) of a planar point set in time $O(kn \log n)$. In combination with the algorithm for the same problem by Viennot [24], runnning in time $O((n^2/k) \log n)$, maximum k-chains can be obtained in time $O(n^{3/2} \log n)$ for all k.

Here, we give a simpler, more direct description of this combined algorithm of [24, 9], as an extension of the standard RSK algorithm. The main idea of the symmetric RSK algorithm is to construct the pair of tableaux $rsk(P^{\lll}) = (H, T)$ simultaneously by rows and by columns. The successive rows of tableaux H, T are constructed as in the standard RSK algorithm. At the same time, the successive columns in tableau H (respectively, T) are obtained by running the standard RSK algorithm for $rsk(P^{\lesseqgtr})$ (respectively, $rsk(P^{\gtreqless})$), using the symmetries exposed by Theorem 12.

There is clearly some redundancy in running the standard algorithm three times on partial orders $P^{\lll}$, $P^{\lesseqgtr}$, $P^{\gtreqless}$. However, this constant-factor redundancy allows one to reduce the overall asymptotic running time. Notice that as a result of the first iteration of each of the three runs, we obtain the union of the initial row and initial column in each of H and T; this union is called the initial *principal hook* of the respective tableau. Likewise, as a result of the second iteration, we obtain the second principal hook of both H and T (i.e. the union of the second row and column, minus the initial principal hook). Crucially, while the number of both rows and columns in a Young tableau of order n can be as high as n, the number of its principal hooks is always at most $n^{1/2}$. Thus, the algorithm can be terminated after at most $\lfloor n^{1/2} \rfloor$ iterations made by each of the three simulaneous runs on $P^{\lll}$, $P^{\lesseqgtr}$, $P^{\gtreqless}$. The worst-case running time of the symmetric RSK algorithm is $n^{1/2} \cdot O(n \log n) = O(n^{3/2} \log n)$.

▶ **Example 16.** Figure 2 shows the execution of the symmetric RSK algorithm on the same input point set as in Figure 1. For the sake of brevity, only the computation of tableau T from partial orders $P^{\lll}$, $P^{\gtreqless}$ is shown explicitly, while the symmetric computation of tableau H from partial orders $P^{\lll}$, $P^{\lesseqgtr}$ is omitted. Compared to the three iterations of the standard algorithm in Figure 1, now only two iterations are required.

4 Speeding up RSK by doubling search

Doubling search. The *doubling search* technique (also called *exponential search*) was introduced by Bentley and Yao [1], and represents a hybrid between linear and binary search. Doubling search is particularly efficient for a non-uniform distribution of the target index, skewed towards an end of the array being searched.

We describe doubling search with the starting point at the upper end of the array, in order to be consistent with its intended application as a subroutine for RSK. Given an array a_i, $1 \leq i \leq s$, sorted in increasing order, and a value q distinct from all a_i, we consider the problem of finding the greatest value in a less than q, that is index $k \geq 0$ such that $a_{s-k} < q < a_{s-k+1}$. We assume $a_i = -\infty$ for $i \leq 0$, and $a_{s+1} = +\infty$.

The search begins at the upper end of the array, comparing q against a_s. If $a_s < q$, we have found $k = 0$. Otherwise, the search continues in two phases. In the *doubling phase*, we compare q against a_{s-1}, a_{s-2}, a_{s-4}, a_{s-8}, …, until we find a subtrahend t that is the least power of 2 such that $a_{s-t} < q$. This phase takes $\lfloor \log k \rfloor + 1$ comparisons.

We now know that $1 \leq k \leq t$, and move on to the *binary search phase*. In this phase, we find the exact value of k in this range by binary search, taking at most $\lfloor \log t \rfloor \leq \lfloor \log k \rfloor + 1$ comparisons. Overall, the doubling search algorithm takes at most $2\lfloor \log k \rfloor + 3$ comparisons. Algorithm 2 shows the pseudocode for the doubling search algorithm.

Symmetric RSK with doubling search. Unfortunately, the asymptotic speedup by a factor of $n^{1/2}$ to the standard RSK algorithm, which is provided by the symmetric algorithm, is not compatible with the speedup by a factor of $\log n$ provided by linear search. However, we are still able to obtain both speedups simultaneously by employing doubling search.

Algorithm 2 Doubling search.

procedure DSEARCH(a, q) ▷ given sorted array a and q, returns index for q in a
  if $a_s < q$ **then return** 0
  $t \leftarrow 1$
  while $a_{s-t} > q$ **do** $t \leftarrow 2t$ ▷ doubling phase
  return index k in $\{1, \ldots, t\}$, such that $a_{s-k} < q < a_{s-k+1}$ ▷ binary search

Consider a specific value x for a point's x-coordinate, as the RSK algorithm iterates on P, $sk(P^{\ll})$, $sk(sk(P^{\ll}))$, etc. These iterations form respectively row 1, 2, 3, ... of diagrams H, T. Let l_r denote the length of row r before a point with coordinate x is processed for that row. Let b_r denote the index of the search target y' in row r of a point with coordinate x (as per lines 9 or 11 of Algorithm 1); b_r is undefined if no point with coordinate x is left in iteration r (that is, in the $r-1$-th skeleton of P). Let the *displacement interval* in row r be $\{b_r, b_r + 1, \ldots, \min(l_r, b_{r-1} - 1)\}$, where $b_0 = +\infty$. We denote this interval's length by $d_r = \min(l_r + 1, b_{r-1}) - b_r$; in particular, $d_r = 0$ if $b_r = b_{r-1}$. We also define $d_r = 0$ if b_r is undefined due to no point with coordinate x being left in iteration r.

▶ **Example 17.** Consider the computation of $rsk(P^{\ll})$ by the standard and the symmetric RSK algorithms in Figures 1 and 2. Let us fix $x = 6$.

In the first iteration, the bottom row of tableaux H, T is formed. Just before the processing of point $(6, 3) \in P$ begins, the current state of the tableaux rows is $H_{init} = (1, 2, 4)$, $T_{init} = (1, 5, 6)$, and their common length is $l_1 = 3$. The least value in T_{init} greater than $y = 3$ is 5, and its index in T_{init} is $b_1 = 2$. The displacement interval is between $b_1 = 2$ and $l_1 = 3$ inclusive, and its length is $d_1 = l_1 + 1 - b_1 = 2$.

In the second iteration, the middle row of tableaux H, T is formed. Just before the processing of point $(6, 5) \in sk(P^{\ll})$ begins, the current state of the tableaux rows is $H_{init} = (3, 5)$, $T_{init} = (2, 8)$, and their common length is $l_2 = 2$. The least value in T_{init} greater than $y = 5$ is 8, and its index in T_{init} is $b_2 = 2$. The displacement interval is empty (being defined between $b_2 = 2$ and $b_1 - 1 = 1$ inclusive), and its length is $d_2 = b_1 - b_2 = 0$.

In the third and final iteration (which is absent from the symmetric algorithm in Figure 2), the top row of tableaux H, T is formed. Just before the processing of point $(6, 8) \in sk(sk(P^{\ll}))$ begins, the tableaux rows H_{init}, T_{init} are both empty, and their common length is $l_3 = 0$. The least value in T_{init} greater than $y = 5$ is by convention $+\infty$, and its index in T_{init} is by convention $b_3 = 1$. The displacement interval is between $b_3 = 1$ and $b_2 = 2$, and its length is $d_3 = b_3 - b_2 = 1$.

▶ **Theorem 18.** *The symmetric RSK algorithm with doubling search solves the RSK correspondence problem in worst-case time $O(n^{3/2})$.*

Proof. Without loss of generality, assume that n is a perfect square (otherwise, the input can be extended by extra points with a suitably high y-value). Let $m = n^{1/2}$.

For a fixed x-coordinate, consider the displacement interval in a given row r. The rectangle of tableau cells below and including this interval consists of rd_r cells. All these rectangles for different values of r are pairwise disjoint. The symmetric RSK algorithm terminates after processing at most m rows. The total number of cells in the rectangles defined by the displacement intervals in these rows is obviously at most n:

$$\sum_{r=1}^{m} rd_r \leq n$$

We also have $\sum_{r=1}^m r = \frac{m(m+1)}{2} \le m^2 = n$. Let $d'_r = d_r + 1$. By the above, we have

$$\sum_{r=1}^m r d'_r \le n + n = 2n$$

While working on a point with coordinate x within row r, doubling search makes at most $\lfloor 2 \log d'_r \rfloor + 3$ comparisons. For the total number of comparisons made for the given x-coordinate, we have by the arithmetic-geometric mean inequality and the Stirling lower bound on the factorial (cancelling the rounding down of the logarithms, and omitting the constant factor 2 and the additive term $\sum_{r=1}^m 3 = O(m)$):

$$\sum_{r=1}^m \log d'_r = \sum_{r=1}^m \log \tfrac{r d'_r}{r} = \log \prod_{r=1}^m \tfrac{r d'_r}{r} = \log\Bigl(\tfrac{1}{m!} \prod_{r=1}^m r d'_r\Bigr) \le$$
$$\log\Bigl(\tfrac{1}{m!}\bigl(\tfrac{1}{m}\sum_{r=1}^m r d'_r\bigr)^m\Bigr) \le \log \tfrac{(2n/m)^m}{m!} = \log \tfrac{(2m)^m}{m!} \le \log \tfrac{(2m)^m}{(m/e)^m} =$$
$$m \log(2e) = O(m)$$

There are n different x-coordinates to consider, therefore the algorithm makes $n \cdot O(m) = O(n^{3/2})$ comparisons in total. ◀

5 Conclusion

We have given a simple, direct description of the symmetric RSK algorithm by Felsner and Wernisch [9]. We have shown how this algorithm can be enhanced with doubling search, improving the asymptotic running time from $O(n^{3/2} \log n)$ to $O(n^{3/2})$. It it also worth noticing that the (worst-case) running time of our algorithm is lower than the average-case running time of the standard (or the symmetric) RSK algorithm on uniformly random permutations, as analysed by Romik [16]. Our result implies a similar improvement for the k-chain problem for arbitrary k.

A natural lower bound on the running time of RSK correspondence is provided by the LIS problem, which is a subproblem for RSK, and requires $\Omega(n \log n)$ comparisons in the comparison model [10]. Thus, there remains a substantial gap between the known upper and lower bounds for the asymptotic complexity of the RSK correspondence.

Apart from potential improvements in the algorithm or the lower bound, there is scope for future work in extending the algorithm for more general versions of the RSK correspondence, e.g. that between positive integer matrices and semistandard Young tableaux. An experimental confirmation of the efficiency of our algorithm also remains an endeavor for future work; this is a non-trivial task, since most existing experiments with RSK, e.g. those by Vasilyev and Duzhin [7, 22], concentrate on either Plancherel-random Young diagrams, or on Young diagrams with (near-)maximum dimensions; such a diagram shape seems to be far away from the worst-case shape suggested by the proof of Theorem 18.

References

1 Jon Louis Bentley and Andrew Chi-Chih Yao. An almost optimal algorithm for unbounded searching. *Information Processing Letters*, 5(3):82–87, 1976. doi:10.1016/0020-0190(76)90071-5.

2 Sergei Bespamyatnikh and Michael Segal. Enumerating longest increasing subsequences and patience sorting. *Information Processing Letters*, 76:7–11, 2000. doi:10.1016/S0020-0190(00)00124-1.

3 Henrik Blunck and Jan Vahrenhold. In-Place Algorithms for Computing (Layers of) Maxima. *Algorithmica*, 57:1–21, 2010. doi:10.1007/s00453-008-9193-z.

4 Adam L. Buchsbaum and Michael T. Goodrich. Three-Dimensional Layers of Maxima. *Algorithmica*, 39:275–286, 2004. doi:10.1007/s00453-004-1082-5.

5 Jeff Calder, Selim Esedoḡlu, and Alfred O. Hero. A PDE-based Approach to Nondominated Sorting. *SIAM Journal on Numerical Analysis*, 53:82–104, 2015. doi:10.1137/130940657.

6 E W Dijkstra. Some beautiful arguments using mathematical induction. *Acta Informatica*, 13(1):1–8, 1980. doi:10.1007/BF00288531.

7 V. S. Duzhin and N. N. Vasilyev. Asymptotic behavior of normalized dimensions of standard and strict Young diagrams: growth and oscillations. *Journal of Knot Theory and Its Ramifications*, 25(12):1642002, 2016. doi:10.1142/S0218216516420025.

8 P Erdös and G Szekeres. A combinatorial problem in geometry. *Compositio Mathematica*, 2:463–470, 1935.

9 Stefan Felsner and Lorenz Wernisch. Maximum k-Chains in Planar Point Sets: Combinatorial Structure and Algorithms. *SIAM Journal on Computing*, 28:192–209, 1998. doi:10.1137/S0097539794266171.

10 Michael L Fredman. On computing the length of longest increasing subsequences. *Discrete Mathematics*, 11:29–35, 1975. doi:10.1016/0012-365X(75)90103-X.

11 Curtis Greene. An Extension of Schensted's Theorem. *Advances in Mathematics*, 14:254–265, 1974.

12 Dan Gusfield. *Algorithms on Strings, Trees, and Sequences*. Cambridge University Press, 1997. doi:10.1017/CBO9780511574931.

13 D E Knuth. Permutations, matrices, and generalized Young tableaux. *Pacific Journal of Mathematics*, 34(3):709–727, 1970.

14 S. N. Majumdar and S. Nechaev. Exact asymptotic results for the Bernoulli matching model of sequence alignment. *Physical Review E: Statistical, Nonlinear, and Soft Matter Physics*, 72(2):020901, 2005. doi:10.1103/PhysRevE.72.020901.

15 G de B Robinson. On the representations of the symmetric group. *American Journal of Mathematics*, 60:745–760, 1938. doi:10.2307/2372326.

16 D. Romik. The Number of Steps in the Robinson-Schensted Algorithm. *Functional Analysis and Its Applications*, 39(2):152–155, 2005. doi:10.1007/s10688-005-0030-8.

17 Dan Romik. *The Surprising Mathematics of Longest Increasing Subsequences*. Cambridge University Press, Cambridge, 2014. doi:10.1017/CBO9781139872003.

18 C. Schensted. Longest Increasing and Decreasing Subsequences. *Canadian Journal of Mathematics*, 13:179–191, 1961. doi:10.4153/CJM-1961-015-3.

19 M. P. Schützenberger. Quelques remarques sur une construction de Schensted. *Mathematica Scandinavica*, 12:117–128, 1963. doi:10.7146/math.scand.a-10676.

20 Richard P. Stanley. *Algebraic Combinatorics*. Undergraduate Texts in Mathematics. Springer, New York, NY, 2013. doi:10.1007/978-1-4614-6998-8.

21 Hugh Thomas and Alexander Yong. Longest increasing subsequences, Plancherel-type measure and the Hecke insertion algorithm. *Advances in Applied Mathematics*, 46(1-4):610–642, 2011. doi:10.1016/j.aam.2009.07.005.

22 N. N. Vasiliev and V. S. Duzhin. A Study of the Growth of the Maximum and Typical Normalized Dimensions of Strict Young Diagrams. *Journal of Mathematical Sciences*, 216(1):53–64, 2016. doi:10.1007/s10958-016-2887-x.

23 G. Viennot. Une forme geometrique de la correspondance de Robinson-Schensted. In *Combinatoire et Représentation du Groupe Symétrique*, volume 579 of *Lecture Notes in Mathematics*, pages 29–58. Springer, 1977. doi:10.1007/BFb0090011.

24 G. Viennot. Chain and antichain families, grids and Young tableaux. *Annals of Discrete Mathematics*, 23:409–463, 1984. doi:10.1016/S0304-0208(08)73835-0.

25 X Viennot. Growth diagrams and edge local rules. In *Proceedings of GASCom*, 2018.