Fast RSK correspondence by doubling search

Anonymous author
Anonymous affiliation

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 algorithm 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 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 Author: Please fill in 1 or more \ccsdesc macro

Keywords and phrases Author: Please fill in \keywords macro

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

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. [18, 16]. 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 describe the standard RSK algorithm by Robinson and Schensted (which by itself is the most common defition of the RSK correspondence). We then give a simple, direct decription of the symmetric RSK algorithm, which is implied by the k-chain algorithm of Felsner and Wernisch [9]. Further, 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 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.

2 The RSK correspondence

Planar dominance We will use the standard terminology related to partial orders: *downset*, *principal downset*, *chain*, *antichain*. We will also consider two specific (strict) partial orders between points on the Euclidean plane.

▶ **Definition 1.** *Point* (x, y) *is* dominated *(respecively, anti-dominated) by point* (x', y')*, if* $x \leq x'$ *and* $y \leq y'$ *(respectively,* $y \geq y'$*). Whenever* $x \neq x'$*,* $y \neq y'$*, we denote dominance by* $(x, y) \ll (x', y')$*, and anti-dominance by* $(x, y) \lessgtr (x', y')$

When considering a point set P as a partial order, we will assume the dominance order, unless indicated otherwise. We will also always assume 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 2.** *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. 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* $\mathbb{R}$ *of cardinality* n*.*

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. Column indices increase from left to right, and row indices 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 3.** Figure 1 (middle, right) gives several examples of Young tableaux.

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 [20, 21]. A permutation $\pi : \mathbb{N}_n \to \mathbb{N}_n$ is identified with the point set $P_\pi = \{(x, \pi(x)) \mid x \in \mathbb{N}_n\}$.

▶ **Definition 4.** *The* height *of an element in a finite partial order 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 a partial order* P *into disjoint canonical antichains is called the* canonical antichain partition (CAP)*, denoted* $cap(P)$*.*

Canonical antichains are also sometimes called *layers of minima* (*maxima*) [4, 3], *Pareto fronts* [5], or *terraces* [13]. The canonical antichain partition is also sometimes called the partial order's *greedy cover* [11], *patience sorting* [2], or *non-dominated sorting* [5].

▶ **Example 5.** Figure 1 (top-left) shows a point set P of cardinality 10, and the partitioning $cap(P)$ into five antichains.

We recall the following classical theorem, relied upon by [9].

▶ **Theorem 6** (see e.g. [9])**.** *Partitioning* $cap(P)$ *has the minimum possible number of antichains among all antichain partitionings of* P*. This number is also equal to the maximum cardinality of a chain in* P*.*

Proof. See e.g. [11]; the connection between antichain partitions and chains is via Dilworth's theorem. ◀

[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.

The problem of finding the cardinality of $cap(P)$ is equivalent to the problem of finding the length of a longest increasing subsequence (LIS) in a corresponding permutation. The LIS problem has a long history, going back to Erdös and Szekeres [8] and Robinson [14]. Based on their ideas, a classical LIS algorithm running in time $O(n \log n)$ was made explicit by Knuth [12], Fredman [10] and Dijkstra [6].

▶ **Definition 7.** *Let $A = \{(x_1, y_1) \lessdot (x_2, y_2) \lessdot \ldots \lessdot (x_r, y_r)\}$ be an antichain of cardinality r. Values x_1 and y_r will be called respectively the* head *and the* tail *of A. The* skeleton *of A is the antichain $sk(A) = \{(x_2, y_1) \lessdot (x_3, y_2) \lessdot \ldots \lessdot (x_r, y_{r-1})\}$ of cardinality $r - 1$. Given a point set P, the* skeleton *of P is the point set $sk(P) = \bigcup_{A \in cap(P)} sk(A)$.*

The RSK correspondence The Robinson–Schensted–Knuth (RSK) correspondence, discovered independently by Robinson [14] and Schensted [17] (see also Romik [16]), 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 8.** *Let P be a set of points. The* RSK image[2] *of P is a pair of Young tableaux $rsk(P) = (H, T)$, defined recursively as follows. The initial row in H (respectively, T) is formed by the heads (respectively, the tails) of the antichains in $cap(P)$. The remaining rows of H, T are formed as $rsk(sk(P))$.*

▶ **Example 9.** Figure 1 (top) shows the construction of the initial rows in tableaux H (top middle) and T (top right) from a set of points $P \subseteq \mathbb{N}_{10}^2$ (top left). Figure 1 (middle, bottom) shows the recursive construction of the remaining rows in the tableaux H, T.

▶ **Definition 10.** *Let P be a set of points. The* transpose *of P is the point set obtained by exchanging the x- and y-coordinates of each point: $P^* = \{(y, x) \mid (x, y) \in P\}$. The* x-reversal *(respectively,* y-reversal*) of P is the point set obtained by negating the first (respectively, the second) coordinate: $P^{\mp} = \{(-x, y) \mid (x, y) \in P\}$, $P^{\pm} = \{(x, -y) \mid (x, y) \in P\}$.*

▶ **Observation 11.** *Let π be a permutation. We have $P_\pi^* = P_{\pi^{-1}}$.*

▶ **Proposition 12.** *Let P be a set of points, $rsk(P) = (H, T)$. We have* (i) *$rsk(P^*) = (T, H)$,* (ii) *$rsk(P^{\pm}) = (H, T')$,* (iii) *$rsk(P^{\mp}) = (H', T)$, where H', T' are some Young diagrams.*

Proof. Statement (i) is obvious by symmetry.

Let us establish statement (iii). Let (x_0, y_0) be the point with the least y-coordinate in $sk(P)$, so y_0 is the tail of the least-height antichain in $sk(P)$. Let A be an antichain in P such that $(x_0, y_0) \in sk(A)$. Then, there is a pair of points $(x, y_0) \lessdot (x_0, y)$ in A. The subset of P with y-coordinate less than y_0 must consist of a single chain including point (x_0, y) (otherwise, there would be two points with y-coordinate less than y_0 on some antichain B of P, and then $sk(B)$ would contain a point of $sk(P)$ with y-coordinate less than y_0). Consider the points corresponding to this chain in $P^{\mp}$, with x-coordinates negated. These points, including point $(-x_0, y)$, form a subset of the least height antichain in $P^{\mp}$. Point $(-x, y_0)$

[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.

Table 1 Standard RSK

```
1:  procedure RSK(P)          ▷ given point set P sorted by x-coordinate, returns rsk(P)
2:      H_1 ← ∅; T_1 ← ∅                ▷ initialise variables for initial rows of H, T
3:      S ← ∅                                        ▷ initialise variable for sk(P)
4:      while P ≠ ∅ do
5:          (x, y) ← point in P with least x-coordinate
6:          y' ← least value in T_1 greater than y; +∞ if none exists      ▷ binary search
7:          if y' = +∞ then
8:              append x to H_1; append y to T_1                ▷ start new antichain
9:          else
10:             replace y' by y in T_1; append (x, y') to S       ▷ extend antichain
11:         remove (x, y) from P
12:     (H_+, T_+) ← RSK(S)                                      ▷ recursive call
13:     H ← tableau with initial row H_1 and remaining rows H_+
14:     T ← tableau with initial row T_1 and remaining rows T_+
15:     return (H, T)                                        ▷ rsk(P) = (H, T)
```

must belong to the second-least height antichain in $P^{\mp}$, and must be the $\lessgtr$-minimal point on that antichain. Therefore, y_0 is the tail of the second-least height antichain in $P^{\mp}$.

We have established that the tail of the least height antichain in $sk(P)$ is equal to the tail of the second-least height antichain in $P^{\mp}$. Now (iii) follows by (i) and the recursive construction of Definition 8, and (ii) follows from (iii) by symmetry. ◀

Multichains The following definition and proposition are not essential for this paper. However, we include them for highlighting the connection with [9].

▶ **Definition 13.** *Let P be a set of points. A k-chain is a subset of P that can be represented as a union of k chains.*

▶ **Proposition 14** ([9])**.** *Let P be a set of points. The maximum cardinality of a k-chain in P is equal to the number of cells in the initial k rows of $rsk(P)$.*

3 RSK algorithms

Standard RSK algorithm The classical definition of the RSK correspondence leads directly to an algorithm for computing the RSK image of a given point set.

Given a point set P, the pair of tableaux $rsk(P) = (H, T)$ 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 P' of points seen so far, we maintain the partitioning $cap(P')$; in particular, the heads and the tails of antichains in that partitioning are kept in sorted order. We also maintain the skeleton $sk(P')$ in order of increasing x-coordinate. When the scan of P is complete, the heads (respectively, tails) of antichains in $cap(P)$ become the initial row of tableau H (respectively, T) in $rsk(P)$. To obtain the remaining rows of $rsk(P)$, we repeat the above procedure on point set $sk(P)$. Table 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), and the state of the tableaux H and T at the end of the respective iteration (middle, right).

The computation of the initial row in the standard RSK algorithm (before the recursive call in line 13 of Table 1) is essentially identical to the classical algorithm for the LIS problem [12, 10, 6]. The processing of each of n points in set P is done be binary search, therefore the whole initial row is obtained in time $O(n \log n)$. In total, there are at most n rows in $rsk(P)$, therefore the overall time is $n \cdot O(n \log n) = O(n^2 \log n)$.

Romik [15] analysed the average-case running time of the standard RSK algorithm on a uniformly random permutation (in this case, the shape of diagrams H, T follows the *Plancherel* distribution). The average-case running time in this setting is $O(n^{3/2} \log n)$.

Symmetric RSK algorithm Felsner and Wernisch [9] proposed a more efficient, symmetric version of the RSK algorithm. Their algorithm was described in the language of k-chains. In particular, they gave two algorithms for computing maximum k-chains (and, by symmetry, also k-antichains) of a planar point set: one with running time $O(kn \log n)$, another $O((n^2/k) \log n)$. By running both algorithms simultaneously, 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 algorithm as an extension of the standard RSK algorithm. The main idea of the symmetric RSK algorithm is to construct the pair of tableaux $rsk(P) = (H, T)$ 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 on point set $P^{\pm}$ (respectively, $P^{\mp}$). The correctness of the symmetric RSK algorithm follows from Proposition 12.

There is clearly some redundancy in running the standard algorithm three times on point sets P, $P^{\pm}$, $P^{\mp}$. However, this constant-factor redundancy allows one to reduce the overall asymptotic running time. Notice that after the first iteration of each of the three runs, we have obtained 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, after 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, $P^{\pm}$, $P^{\mp}$. 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 (only the computation of tableau T on point sets P, $P^{\mp}$ is shown explicitly, while the symmetric computation of tableau H on point sets P, $P^{\pm}$ 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 guaranteed to be as fast asymptotically as each of the binary and the linear search alone, even 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

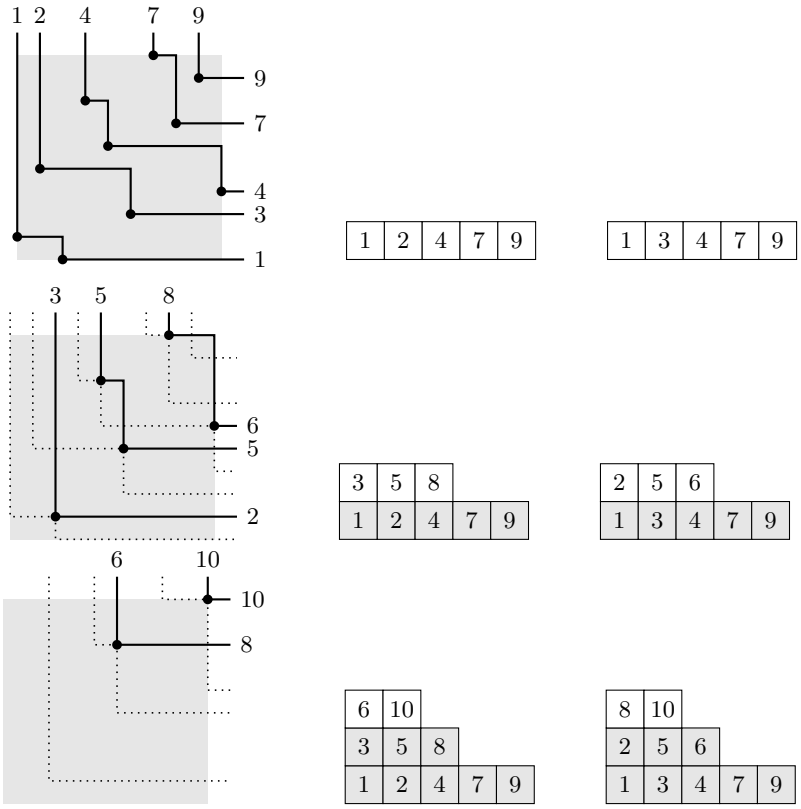

Figure 1 The standard RSK algorithm: running on P, returning tableaux H, T

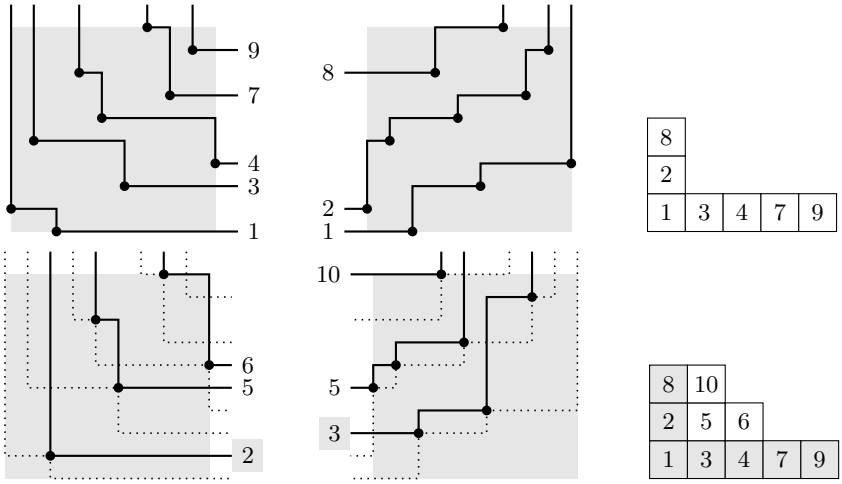

Figure 2 The symmetric RSK algorithm: shown running on P, $P^{\mp}$ only, returning T

Table 2 Doubling search

```
1: procedure DSearch(a, q)        ▷ given sorted array a and q, returns index for q in a
2:     if a_s < q then return 0
3:     t ← 1
4:     while a_{s−t} > q do t ← 2t                                   ▷ doubling phase
5:     return index k in {1, ..., t}, such that a_{s−k} < q < a_{s−k+1}   ▷ binary search
```

a_i, $1 \le i \le s$, 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 $k \ge 0$ such that $a_{s-k} < q < a_{s-k+1}$. We assume $a_i = -\infty$ for $i \le 0$, 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 $k \in \{1, \ldots, t\}$, 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 $\lfloor \log t \rfloor \le \lfloor \log k \rfloor + 1$ comparisons. Overall, the doubling search algorithm takes at most $2\lfloor \log k \rfloor + 3$ comparisons. Table 2 shows the pseudocode for the doubling search algorithm.

Symmetric RSK with doubling search We now present a simple improvement to the symmetric RSK algorithm, replacing binary search within each iteration by doubling search.

Consider a specific value x for a point's x-coordinate, as the RSK algorithm iterates on P, $sk(P)$, $sk(sk(P))$, 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 insertion index in row r of a point with coordinate x (as per lines 8 or 10 of Table 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 $b_{-1} = +\infty$. 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.

▶ **Theorem 17.** *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:

$$\sum_{r=1}^{m} r d_r \le n$$

While searching for an insertion index of a point with coordinate x into 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 $\sum_{r=1}^{m} 3 = O(m)$):

$$\sum_{r=1}^{m} \log d_r = \sum_{r=1}^{m} \log \frac{rd_r}{r} = \log \prod_{r=1}^{m} \frac{rd_r}{r} = \log\left(\frac{1}{m!} \prod_{r=1}^{m} rd_r\right) \leq$$

$$\log\left(\frac{1}{m!}\left(\frac{1}{m}\sum_{r=1}^{m} rd_r\right)^m\right) \leq \log \frac{(n/m)^m}{m!} = \log \frac{m^m}{m!} \leq \log \frac{m^m}{(m/e)^m} = m \log e = O(m)$$

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 the default binary search in this algorithm can be replaced by doubling search, improving the asymptotic running time from $O(n^{3/2} \log n)$ to $O(n^{3/2})$. It should also be noted 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 [15]. 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, 19], 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 17.

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.
2. Sergei Bespamyatnikh and Michael Segal. Enumerating longest increasing subsequences and patience sorting. *Information Processing Letters*, 76:7–11, 2000.
3. Henrik Blunck and Jan Vahrenhold. In-Place Algorithms for Computing (Layers of) Maxima. *Algorithmica*, 57:1–21, 2010.
4. Adam L. Buchsbaum and Michael T. Goodrich. Three-Dimensional Layers of Maxima. *Algorithmica*, 39:275–286, 2004.
5. Jeff Calder, Selim Esedoḡlu, and Alfred O. Hero. A PDE-based Approach to Nondominated Sorting. *SIAM Journal on Numerical Analysis*, 53:82–104, jan 2015.
6. E W Dijkstra. Some beautiful arguments using mathematical induction. *Acta Informatica*, 13(1):1–8, 1980.
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.

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.

10 Michael L Fredman. On computing the length of longest increasing subsequences. *Discrete Mathematics*, 11:29–35, 1975.

11 Dan Gusfield. *Algorithms on Strings, Trees, and Sequences*. Cambridge University Press, 1997.

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

13 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.

14 G de B Robinson. On the representations of the symmetric group. *American Journal of Mathematics*, 60:745–760, 1938.

15 D. Romik. The Number of Steps in the Robinson-Schensted Algorithm. *Functional Analysis and Its Applications*, 39(2):152–155, apr 2005.

16 Dan Romik. *The Surprising Mathematics of Longest Increasing Subsequences*. Cambridge University Press, Cambridge, 2014.

17 C. Schensted. Longest Increasing and Decreasing Subsequences. *Canadian Journal of Mathematics*, 13:179–191, nov 1961.

18 Richard P. Stanley. *Algebraic Combinatorics*. Undergraduate Texts in Mathematics. Springer, New York, NY, 2013.

19 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.

20 G. Viennot. Une forme geometrique de la correspondance de Robinson-Schensted. *Combinatoire et Représentation du Groupe Symétrique*, pages 29–58. 1977.

21 G. Viennot. Chain and Antichain Families Grids and Young Tableaux. *Annals of Discrete Mathematics*, 23:409–463, 1984.