Stack sorting with increasing and decreasing stacks

G. Cerbai, L. Ferrari

Dipartimento di Matematica e Informatica "U. Dini", Universitá degli Studi di Firenze,
Viale Morgagni 65, 50134 Firenze, Italy
giuliocerbai14@gmail.com,luca.ferrari@unifi.it

Permutation Patterns 2018, Dartmouth College, 9-13 July 2018.

Stack sorting (and relatives...)

General formulation:

INPUT - a permutation π;

MACHINE - a network of devices (may be stacks, queues, etc...), connected in series or in parallel (or in some more fancy way...);

OUTPUT - another permutation $f(\pi)$, which is hopefully the identity, otherwise somehow "closer" to the identity than the original permutation π.

Stack sorting (and relatives...)

General formulation:

INPUT - a permutation π;

MACHINE - a network of devices (may be stacks, queues, etc...), connected in series or in parallel (or in some more fancy way...);

OUTPUT - another permutation $f(\pi)$, which is hopefully the identity, otherwise somehow "closer" to the identity than the original permutation π.

Stack sorting (and relatives...)

General formulation:

INPUT - a permutation π;

MACHINE - a network of devices (may be stacks, queues, etc...), connected in series or in parallel (or in some more fancy way...);

OUTPUT - another permutation $f(\pi)$, which is hopefully the identity, otherwise somehow "closer" to the identity than the original permutation π.

Typical questions

- Characterize the permutations that can be sorted by a given network.
- Enumerate sortable permutations with respect to their length.
- If the network is too complex, find a specific algorithm that sorts "many" input permutations and characterize such permutations.

Typical questions

- Characterize the permutations that can be sorted by a given network.
- Enumerate sortable permutations with respect to their length.
- If the network is too complex, find a specific algorithm that sorts "many" input permutations and characterize such permutations.

Typical questions

- Characterize the permutations that can be sorted by a given network.
- Enumerate sortable permutations with respect to their length.
- If the network is too complex, find a specific algorithm that sorts "many" input permutations and characterize such permutations.

Stack sorting and patterns

(Meta)Theorem

The set of permutations which can be sorted by a given network is a permutation class.

This is no longer true if we impose restrictions on the procedure, i.e. if we choose a specific algorithm to be used for the given network (e.g., West-2-stack sortable permutations).

Stack sorting and patterns

(Meta)Theorem

The set of permutations which can be sorted by a given network is a permutation class.

This is no longer true if we impose restrictions on the procedure, i.e. if we choose a specific algorithm to be used for the given network (e.g., West-2-stack sortable permutations).

Stack sorting and patterns

(Meta)Theorem

The set of permutations which can be sorted by a given network is a permutation class.

This is no longer true if we impose restrictions on the procedure, i.e. if we choose a specific algorithm to be used for the given network (e.g., West-2-stack sortable permutations).

Our starting point: the machine $D^k I$

$k+1$ stacks in series:

- the first k stacks are decreasing (i.e. elements are maintained in decreasing order from top to bottom);
- the last stack is increasing.

- $k = 0$: Stacksort;
- $k = 1$: DI machine, introduced by Rebecca Smith (2014).

Our starting point: the machine $D^k I$

$k+1$ stacks in series:

- the first k stacks are decreasing (i.e. elements are maintained in decreasing order from top to bottom);
- the last stack is increasing.

- $k = 0$: Stacksort;
- $k = 1$: DI machine, introduced by Rebecca Smith (2014).

Our starting point: the machine $D^k I$

$k+1$ stacks in series:

- the first k stacks are decreasing (i.e. elements are maintained in decreasing order from top to bottom);
- the last stack is increasing.

- $k = 0$: Stacksort;
- $k = 1$: DI machine, introduced by Rebecca Smith (2014).

Our starting point: the machine $D^k I$

$k+1$ stacks in series:

- the first k stacks are decreasing (i.e. elements are maintained in decreasing order from top to bottom);
- the last stack is increasing.

- $k = 0$: Stacksort;
- $k = 1$: DI machine, introduced by Rebecca Smith (2014).

Our starting point: the machine $D^k I$

$k+1$ stacks in series:

- the first k stacks are decreasing (i.e. elements are maintained in decreasing order from top to bottom);
- the last stack is increasing.

- $k = 0$: Stacksort;
- $k = 1$: DI machine, introduced by Rebecca Smith (2014).

Results on the *DI* machine

Theorem (Smith, 2014)

The permutations sorted by a decreasing stack followed by an increasing one form the class $Av(3241, 3142)$.

Corollary (Smith, 2014; Kremer, 2000)

The number of DI-sortable permutations of length n is equal to the $(n-1)$*-st large Schröder number.*

Results on the *DI* machine

Theorem (Smith, 2014)

The permutations sorted by a decreasing stack followed by an increasing one form the class $Av(3241, 3142)$.

Corollary (Smith, 2014; Kremer, 2000)

The number of DI-sortable permutations of length n is equal to the $(n-1)$*-st large Schröder number.*

Operations of the $D^k I$ machine

- d_0: push the next element of the input permutation into the first decreasing stack D_1;
- d_i, for $i = 1, \ldots, k-1$: pop an element from D_i and push it into the next decreasing stack D_{i+1};
- d_k: pop an element from D_k and push it into the increasing stack I;
- d_{k+1}: pop an element from the increasing stack I and output it (by placing it on the right of the list of elements that have already been output).

Legal operation: when it does not violate the restrictions on the stacks.

Special case: d_{k+1} is legal either if we are pushing into the output the smallest among the elements not already in the output or if all the other operations are not legal

Operations of the $D^k I$ machine

- d_0: push the next element of the input permutation into the first decreasing stack D_1;
- d_i, for $i = 1, \ldots, k-1$: pop an element from D_i and push it into the next decreasing stack D_{i+1};
- d_k: pop an element from D_k and push it into the increasing stack I;
- d_{k+1}: pop an element from the increasing stack I and output it (by placing it on the right of the list of elements that have already been output).

Legal operation: when it does not violate the restrictions on the stacks.

Special case: d_{k+1} is legal either if we are pushing into the output the smallest among the elements not already in the output or if all the other operations are not legal

Operations of the $D^k I$ machine

- d_0: push the next element of the input permutation into the first decreasing stack D_1;
- d_i, for $i = 1, \ldots, k-1$: pop an element from D_i and push it into the next decreasing stack D_{i+1};
- d_k: pop an element from D_k and push it into the increasing stack I;
- d_{k+1}: pop an element from the increasing stack I and output it (by placing it on the right of the list of elements that have already been output).

Legal operation: when it does not violate the restrictions on the stacks.

Special case: d_{k+1} is legal either if we are pushing into the output the smallest among the elements not already in the output or if all the other operations are not legal

Operations of the $D^k I$ machine

- d_0: push the next element of the input permutation into the first decreasing stack D_1;
- d_i, for $i = 1, \ldots, k-1$: pop an element from D_i and push it into the next decreasing stack D_{i+1};
- d_k: pop an element from D_k and push it into the increasing stack I;
- d_{k+1}: pop an element from the increasing stack I and output it (by placing it on the right of the list of elements that have already been output).

Legal operation: when it does not violate the restrictions on the stacks.

Special case: d_{k+1} is legal either if we are pushing into the output the smallest among the elements not already in the output or if all the other operations are not legal

Operations of the $D^k I$ machine

- d_0: push the next element of the input permutation into the first decreasing stack D_1;
- d_i, for $i = 1, \ldots, k-1$: pop an element from D_i and push it into the next decreasing stack D_{i+1};
- d_k: pop an element from D_k and push it into the increasing stack I;
- d_{k+1}: pop an element from the increasing stack I and output it (by placing it on the right of the list of elements that have already been output).

Legal operation: when it does not violate the restrictions on the stacks.

Special case: d_{k+1} is legal either if we are pushing into the output the smallest among the elements not already in the output or if all the other operations are not legal

Operations of the $D^k I$ machine

- d_0: push the next element of the input permutation into the first decreasing stack D_1;
- d_i, for $i = 1, \ldots, k-1$: pop an element from D_i and push it into the next decreasing stack D_{i+1};
- d_k: pop an element from D_k and push it into the increasing stack I;
- d_{k+1}: pop an element from the increasing stack I and output it (by placing it on the right of the list of elements that have already been output).

Legal operation: when it does not violate the restrictions on the stacks.

Special case: d_{k+1} is legal either if we are pushing into the output the smallest among the elements not already in the output or if all the other operations are not legal

Operations of the $D^k I$ machine

- d_0: push the next element of the input permutation into the first decreasing stack D_1;
- d_i, for $i = 1, \ldots, k-1$: pop an element from D_i and push it into the next decreasing stack D_{i+1};
- d_k: pop an element from D_k and push it into the increasing stack I;
- d_{k+1}: pop an element from the increasing stack I and output it (by placing it on the right of the list of elements that have already been output).

Legal operation: when it does not violate the restrictions on the stacks.

Special case: d_{k+1} is legal either if we are pushing into the output the smallest among the elements not already in the output or if all the other operations are not legal

k-sortable permutations

$$B(k) = \{\pi \in S \mid \text{ there is a sequence of legal operations } d_{i_1}, \ldots, d_{i_s} \text{ that sorts } \pi\}$$

Goal: understand the basis of $B(k)$.

k-sortable permutations

$B(k) = \{\pi \in S \mid$ there is a sequence of legal operations $d_{i_1}, \ldots, d_{i_s}$ that sorts $\pi\}$

Goal: understand the basis of $B(k)$.

The case $k = 2$

Theorem

For $j \geq 0$, define the permutation:

$$\alpha_j = 2j+4, 3, a_1, b_1, a_2, b_2, \ldots, a_j, b_j, 1, 5, 2$$

where:

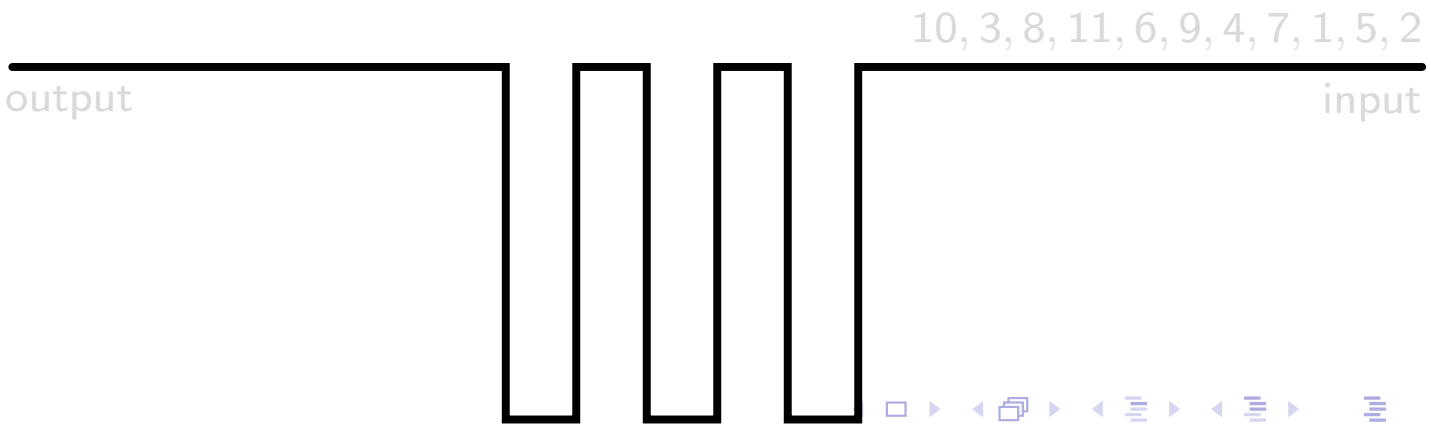
$$\begin{cases} A_j = (a_1, \ldots, a_j) = (2j+2, 2j, 2j-2, \ldots, 6, 4), \\ B_j = (b_1, \ldots, b_j) = (2j+5, 2j+3, 2j+1, \ldots, 9, 7). \end{cases}$$

Then the set of permutations $\{\alpha_j\}_{j\geq 0}$ constitutes an infinite antichain each of whose elements is not 2-sortable. Moreover, α_j is minimal with respect to such a property, i.e. if we remove any element of α_j we obtain a 2-sortable permutation. As a consequence, the basis of $B(2)$ is infinite, since it contains the infinite antichain $\{\alpha_j\}_{j\geq 0}$.

Proof

α_j is not 2-sortable: induction on j.

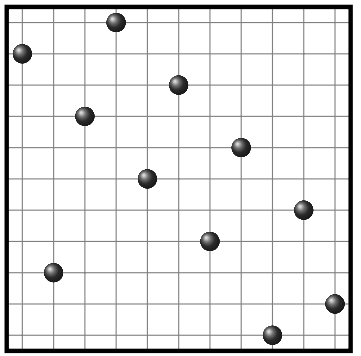
- $j = 0$: the permutation 43152 is not 2-sortable.
- Generic j:

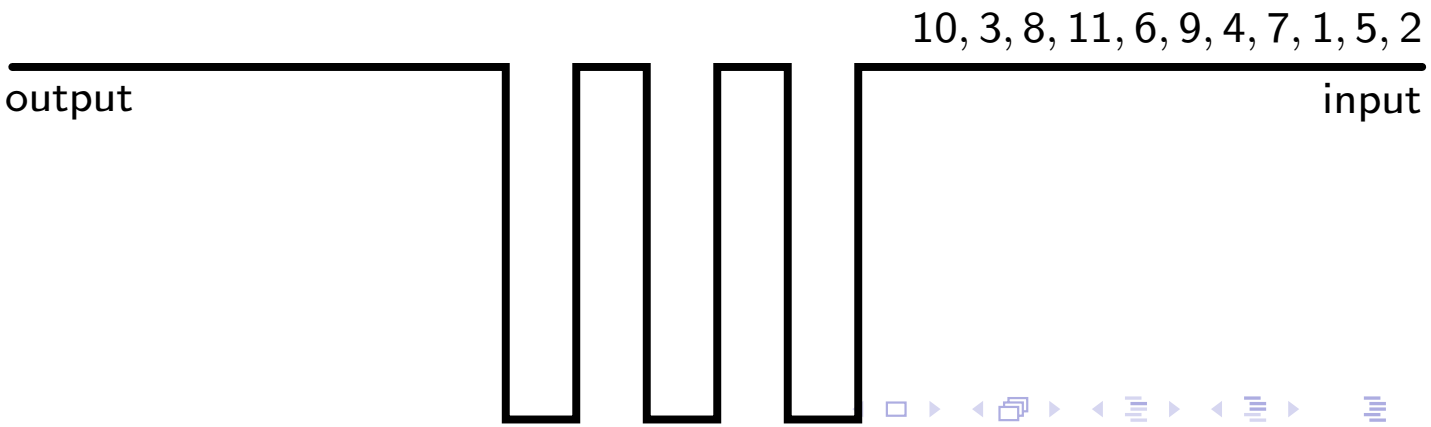


Proof

α_j is not 2-sortable: induction on j.

- $j = 0$: the permutation 43152 is not 2-sortable.
- Generic j:

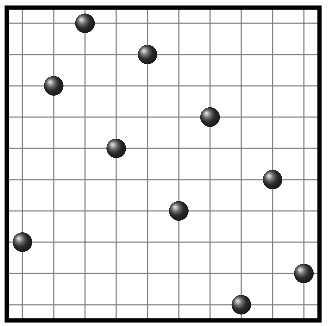


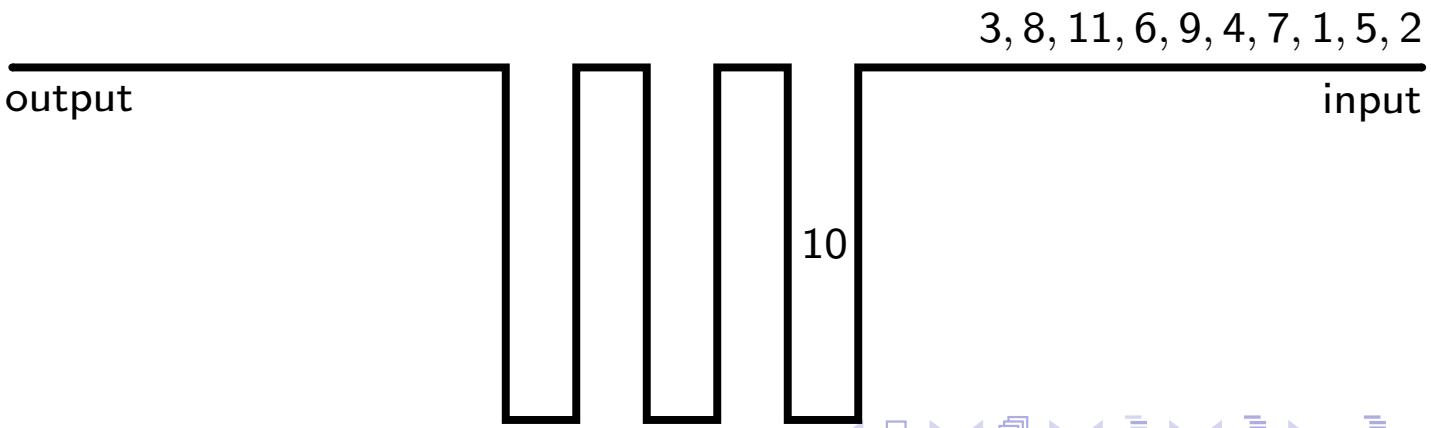


Proof

α_j is not 2-sortable: induction on j.

- $j = 0$: the permutation 43152 is not 2-sortable.
- Generic j:

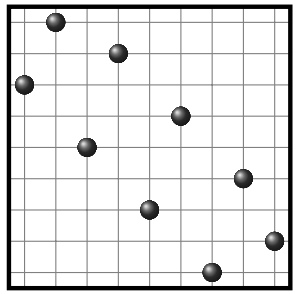


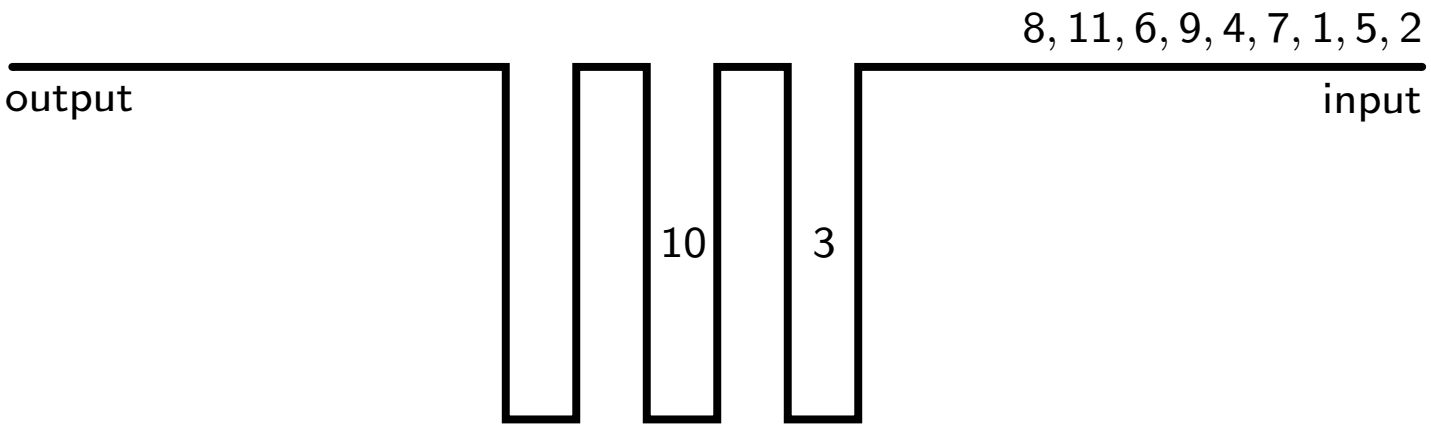


Proof

α_j is not 2-sortable: induction on j.

- $j = 0$: the permutation 43152 is not 2-sortable.
- Generic j:

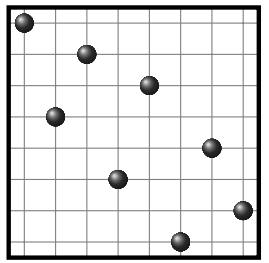


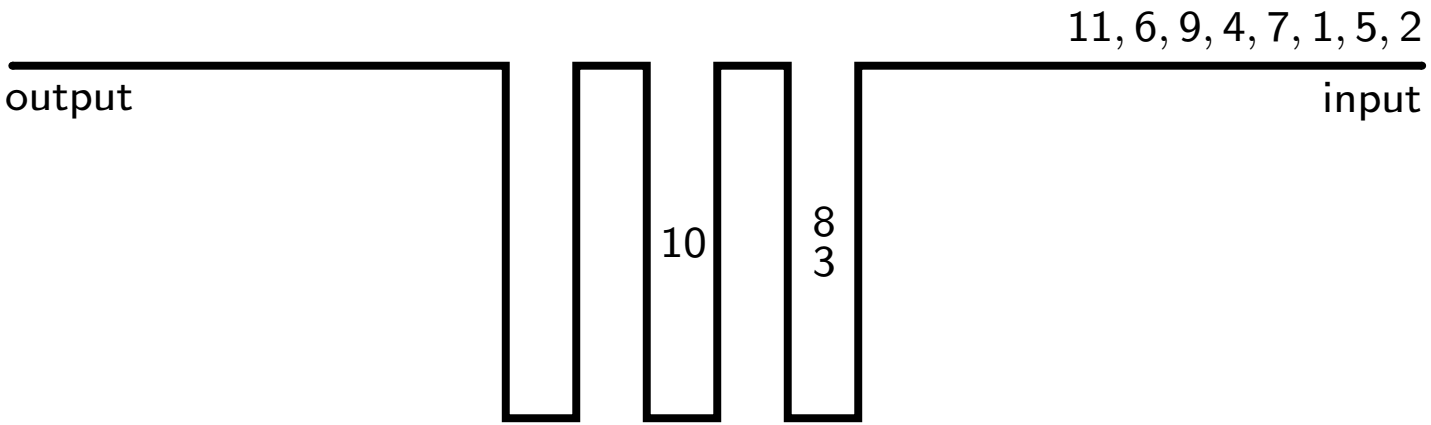


Proof

α_j is not 2-sortable: induction on j.

- $j = 0$: the permutation 43152 is not 2-sortable.
- Generic j:

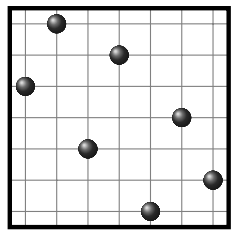


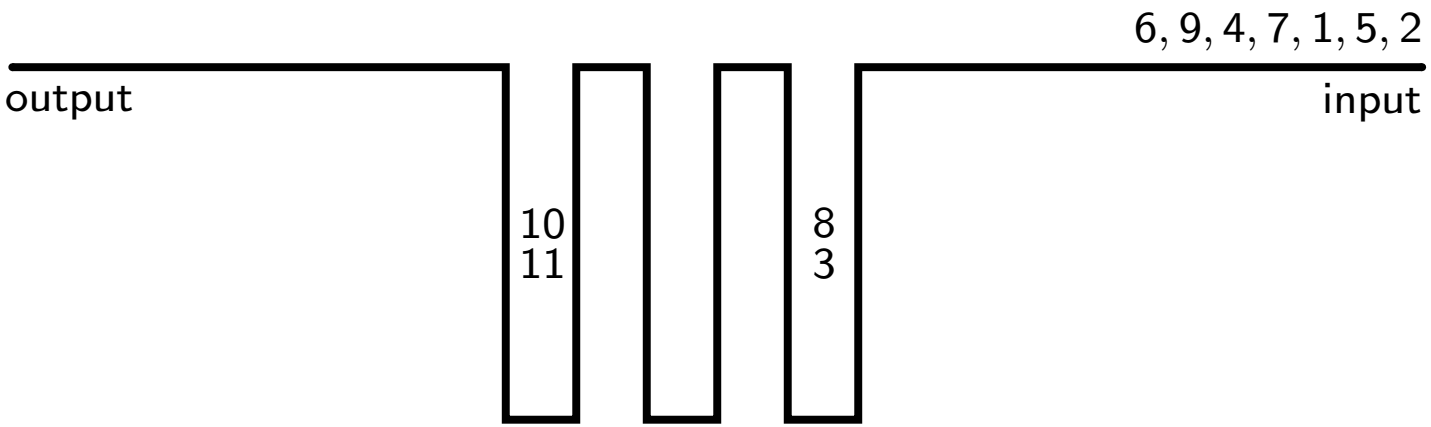


Proof

α_j is not 2-sortable: induction on j.

- $j = 0$: the permutation 43152 is not 2-sortable.
- Generic j:

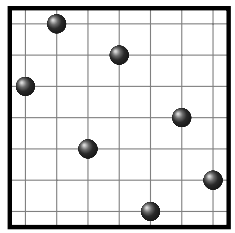


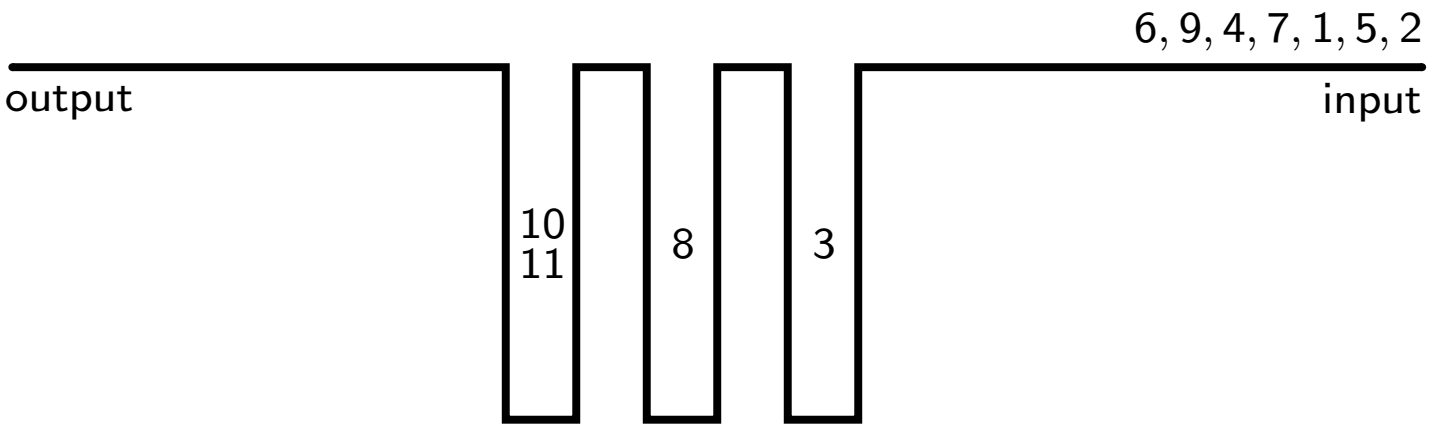


Proof

α_j is not 2-sortable: induction on j.

- $j = 0$: the permutation 43152 is not 2-sortable.
- Generic j:

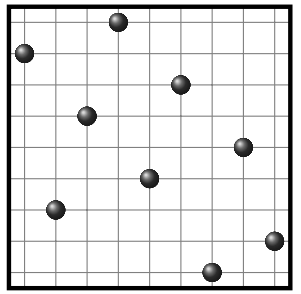


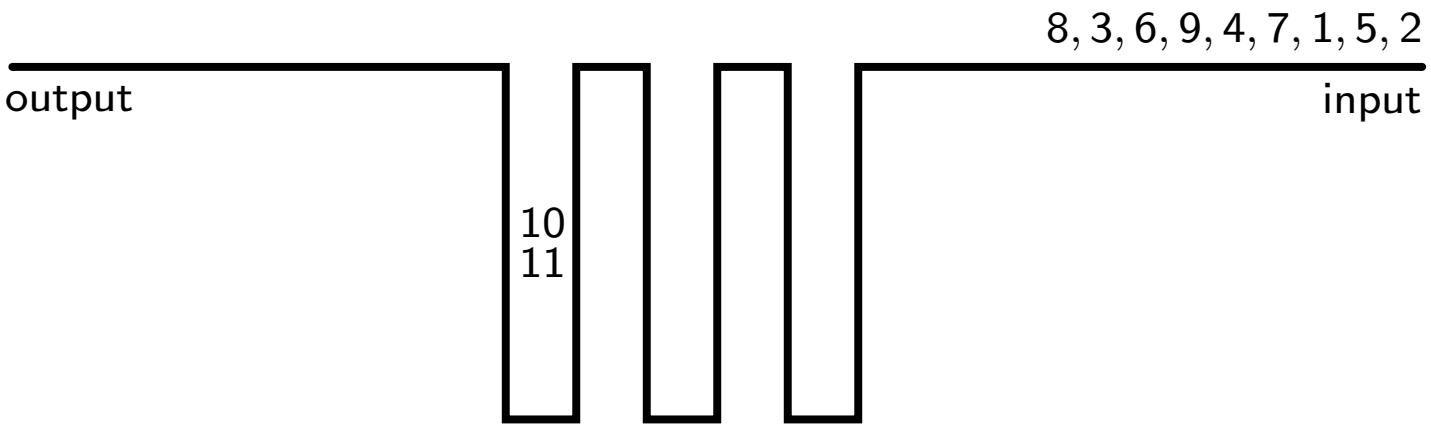


Proof

α_j is not 2-sortable: induction on j.

- $j = 0$: the permutation 43152 is not 2-sortable.
- Generic j:





Proof

All patterns of α_j are 2-sortable: case-by-case analysis.

- Remove $2j + 4$.
- Remove 3.
- Remove a_i.
- Remove b_i.
- Remove either 1 or 5 or 2.

Proof

All patterns of α_j are 2-sortable: case-by-case analysis.

- Remove $2j + 4$.
- Remove 3.
- Remove a_i.
- Remove b_i.
- Remove either 1 or 5 or 2.

Proof

All patterns of α_j are 2-sortable: case-by-case analysis.

- Remove $2j + 4$.
- Remove 3.
- Remove a_i.
- Remove b_i.
- Remove either 1 or 5 or 2.

Proof

All patterns of α_j are 2-sortable: case-by-case analysis.

- Remove $2j + 4$.
- Remove 3.
- Remove a_i.
- Remove b_i.
- Remove either 1 or 5 or 2.

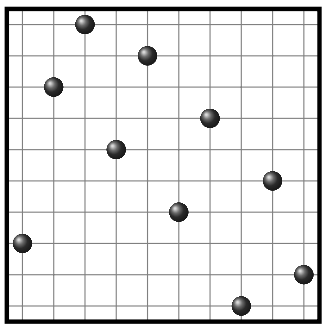
Proof

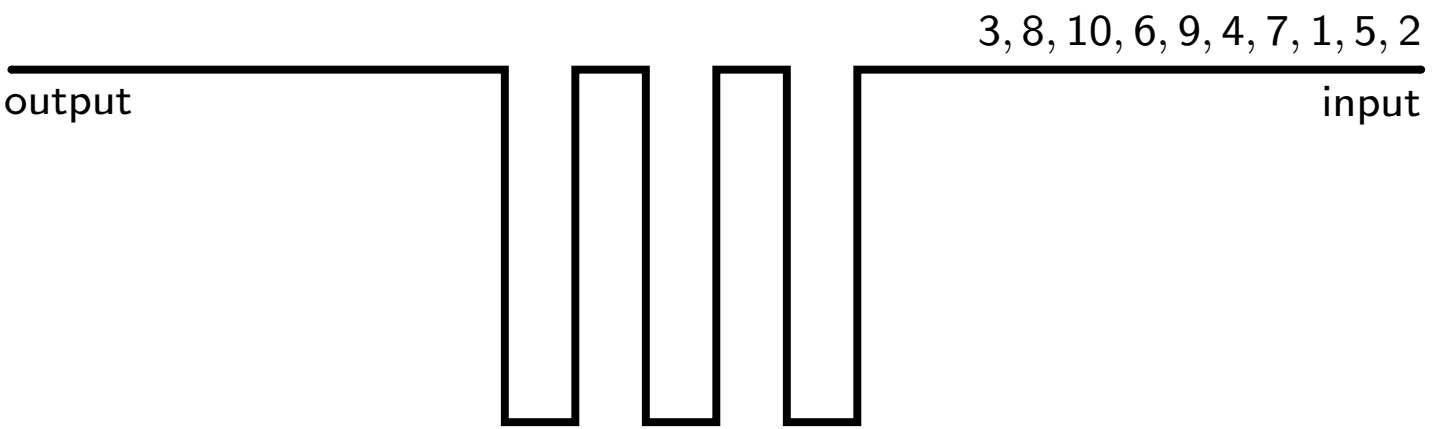
All patterns of α_j are 2-sortable: case-by-case analysis.

- Remove $2j + 4$.
- Remove 3.
- Remove a_i.
- Remove b_i.
- Remove either 1 or 5 or 2.

Proof

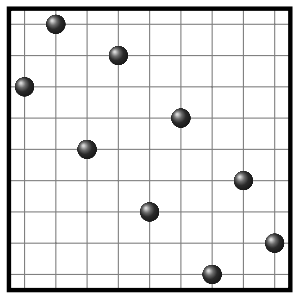
All patterns of α_j are 2-sortable: case-by-case analysis.
Remove $2j + 4$:

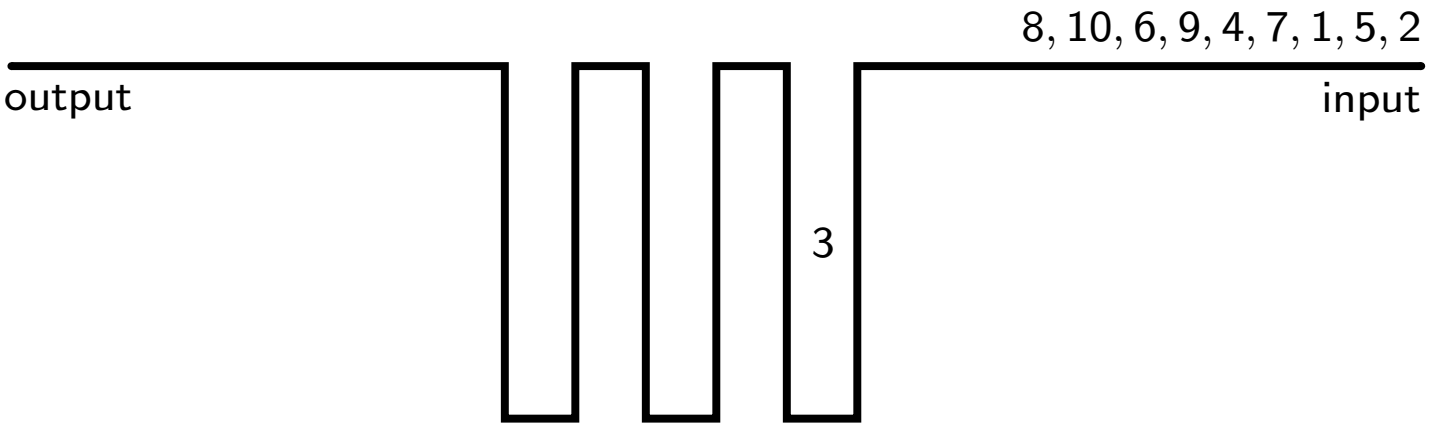




Proof

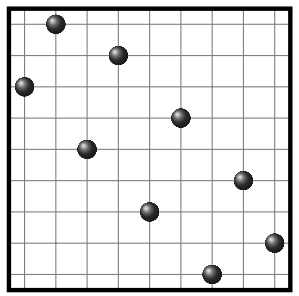
All patterns of α_j are 2-sortable: case-by-case analysis.
Remove $2j + 4$:

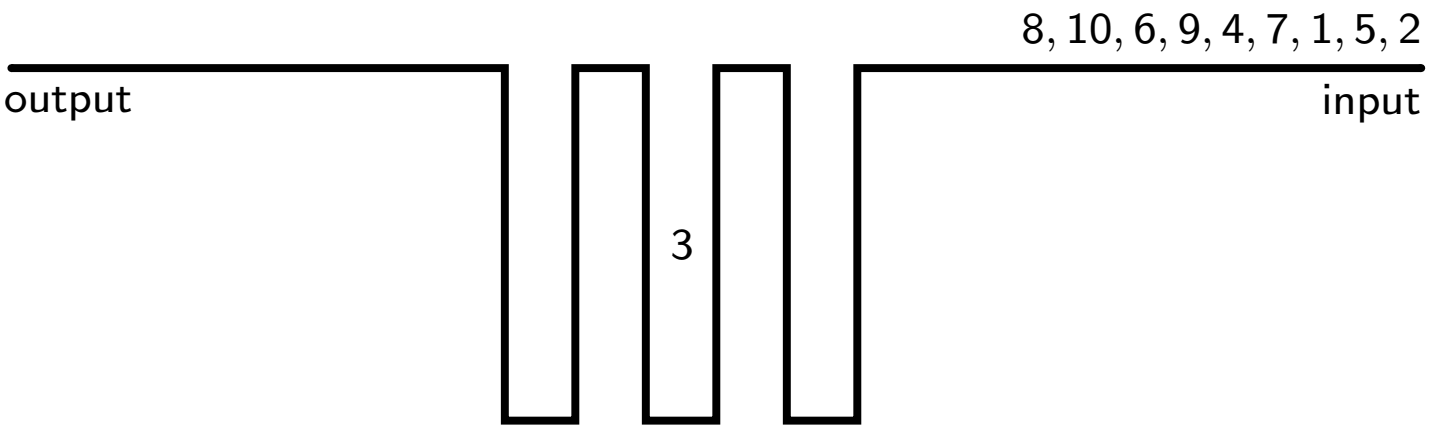




Proof

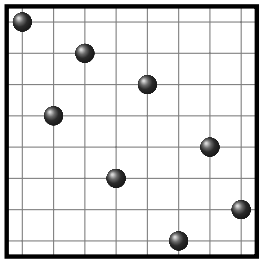
All patterns of α_j are 2-sortable: case-by-case analysis.
Remove $2j + 4$:

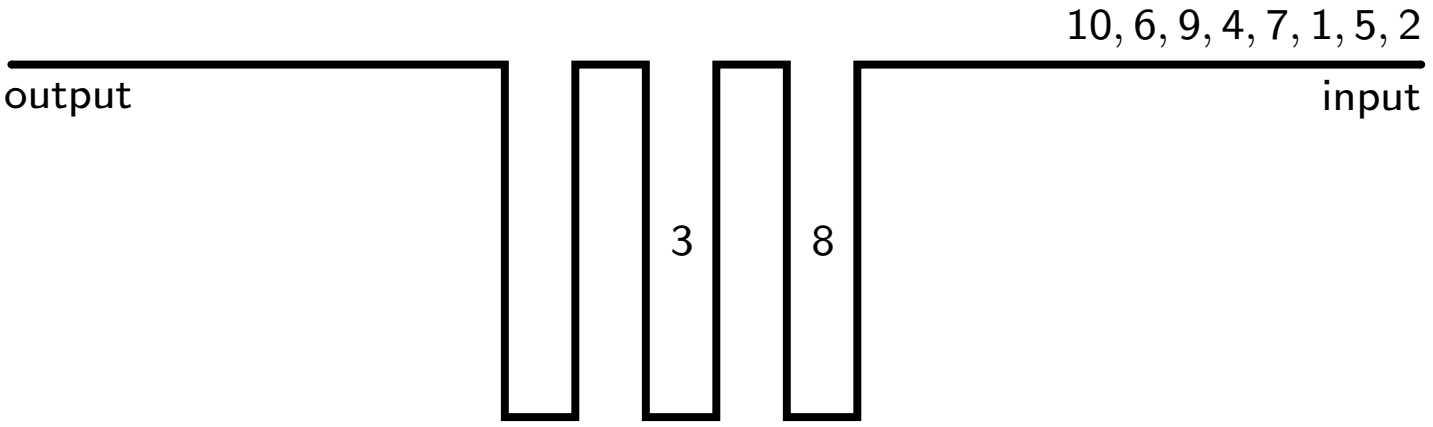




Proof

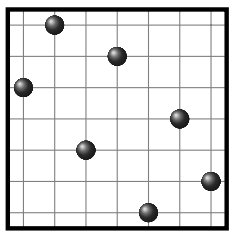
All patterns of α_j are 2-sortable: case-by-case analysis.
Remove $2j + 4$:

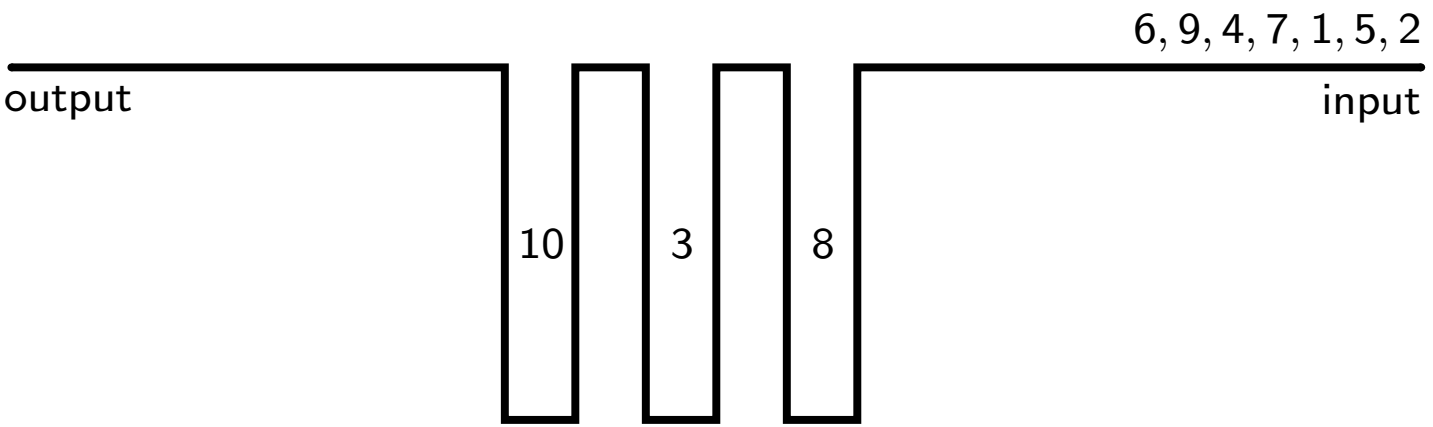




Proof

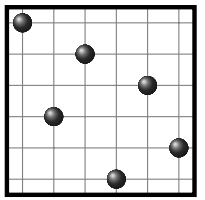
All patterns of α_j are 2-sortable: case-by-case analysis.
Remove $2j + 4$:

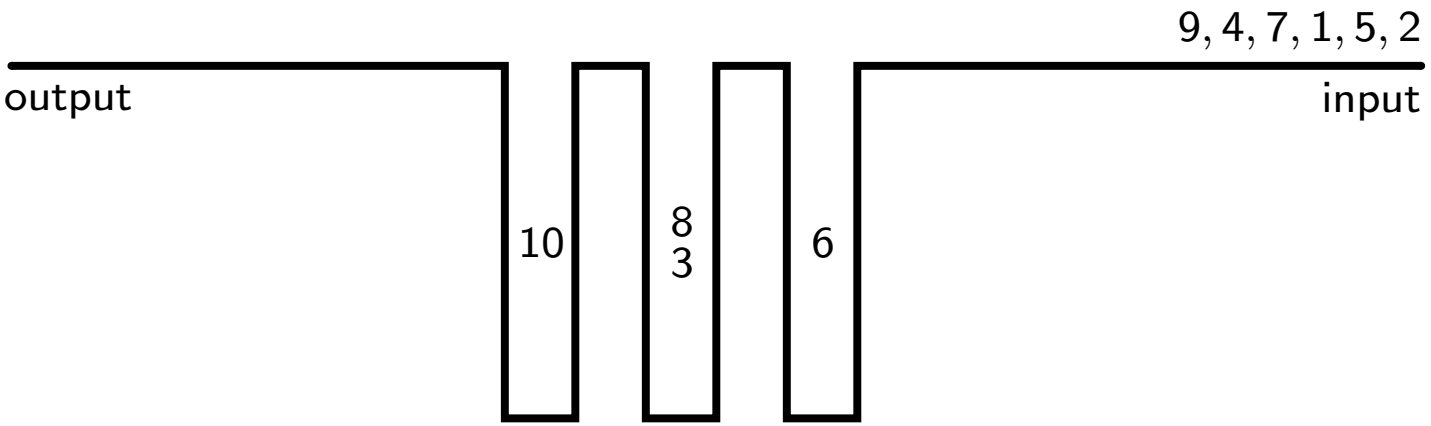




Proof

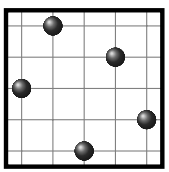
All patterns of α_j are 2-sortable: case-by-case analysis.
Remove $2j + 4$:

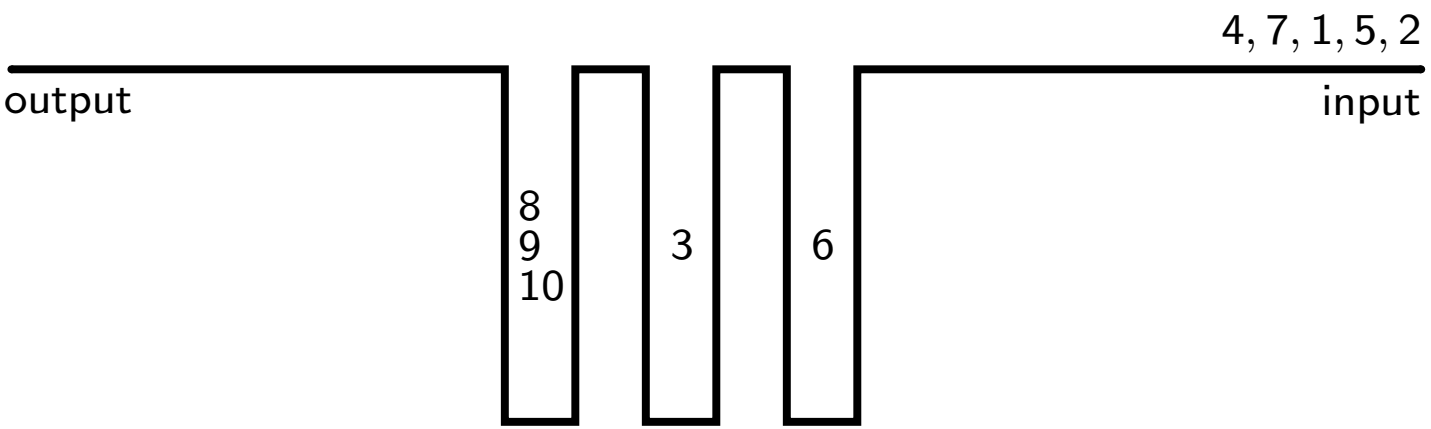




Proof

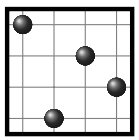
All patterns of α_j are 2-sortable: case-by-case analysis.
Remove $2j + 4$:

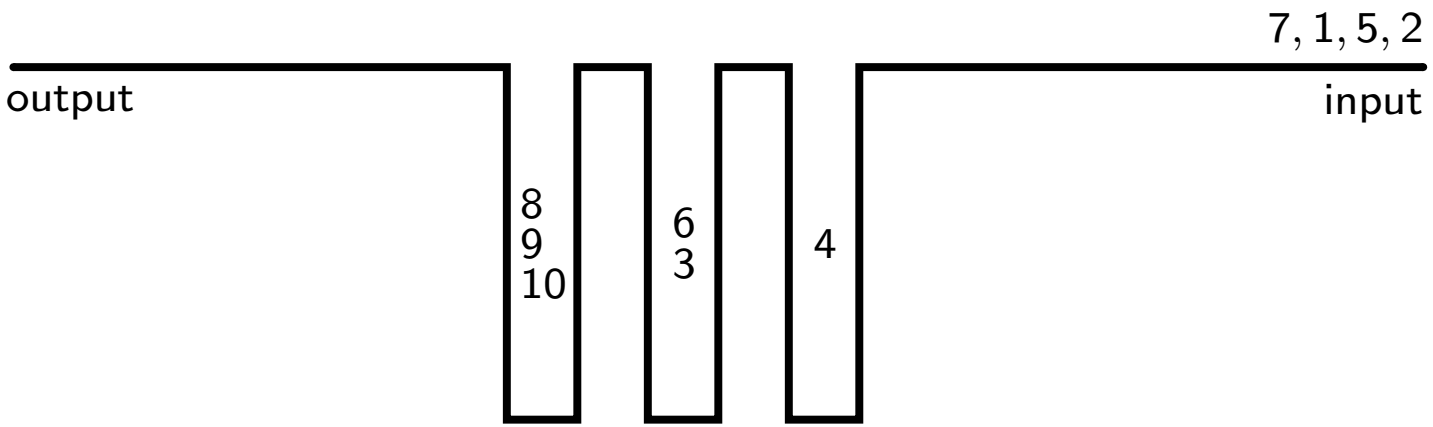




Proof

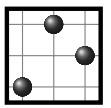
All patterns of α_j are 2-sortable: case-by-case analysis.
Remove $2j + 4$:

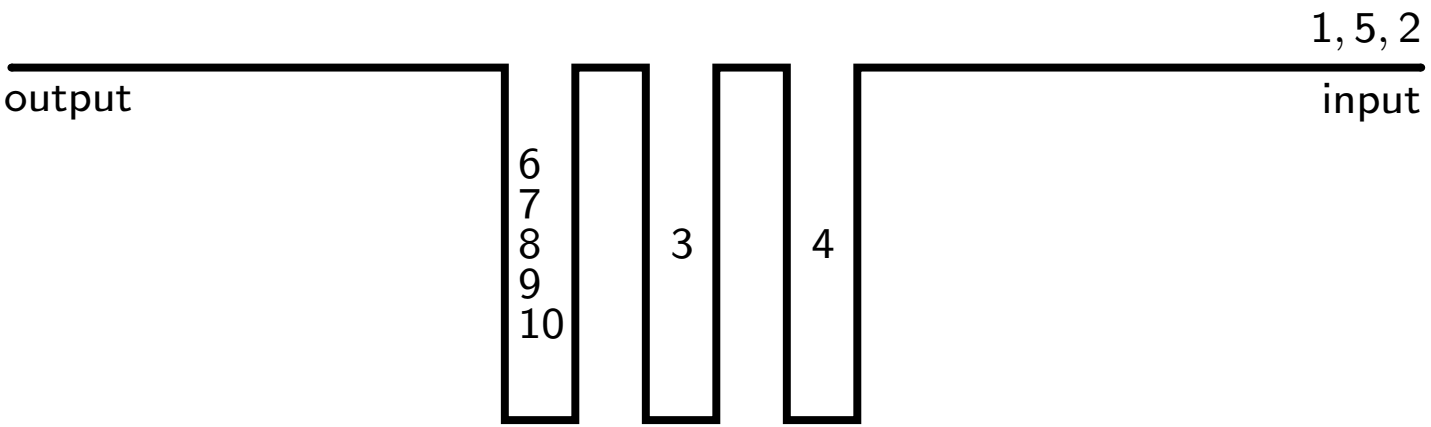




Proof

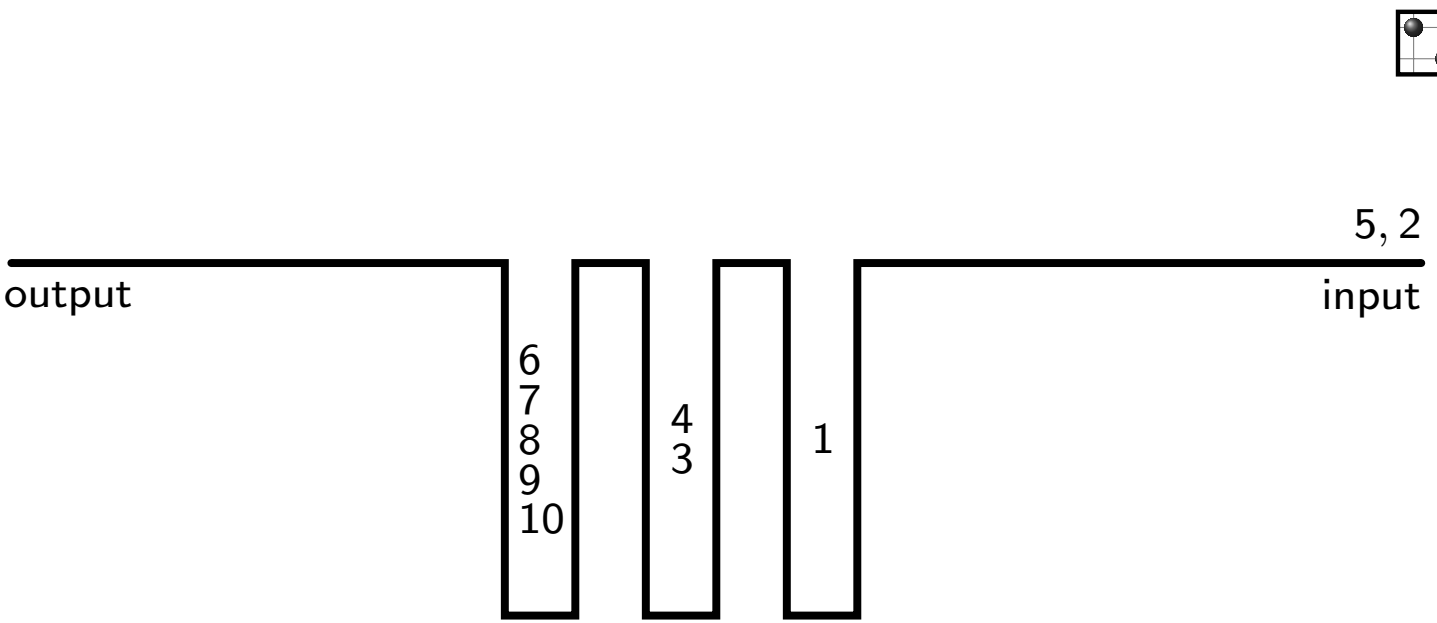
All patterns of α_j are 2-sortable: case-by-case analysis.
Remove $2j + 4$:





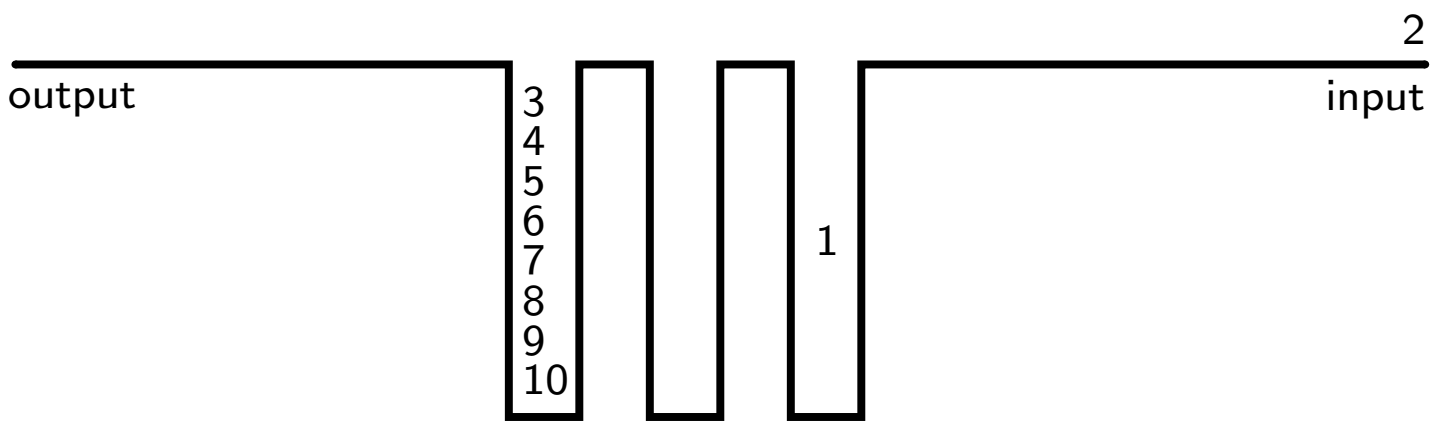
Proof

All patterns of α_j are 2-sortable: case-by-case analysis.
Remove $2j + 4$:



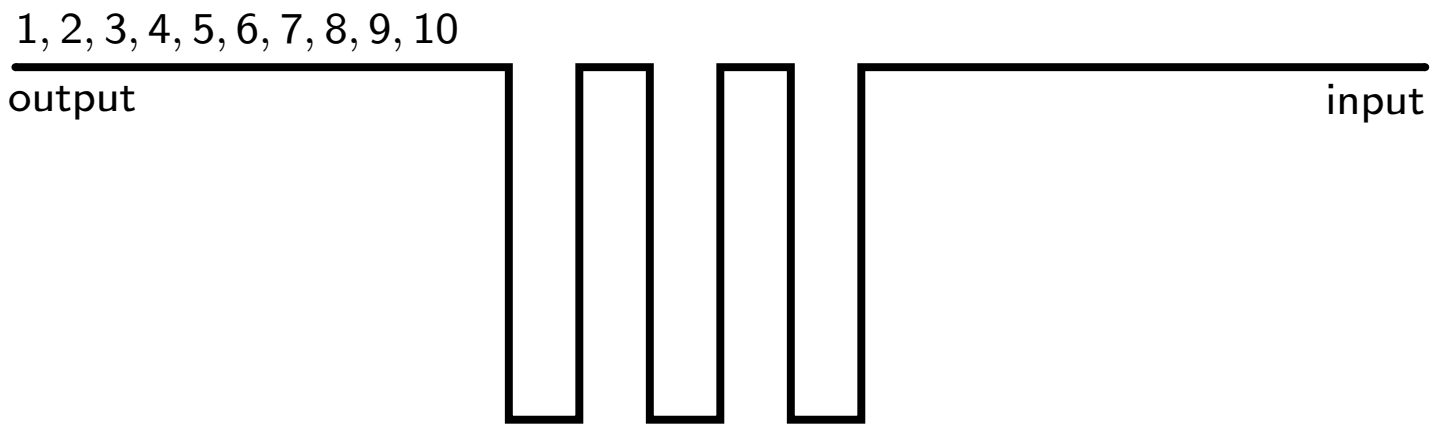
Proof

All patterns of α_j are 2-sortable: case-by-case analysis.
Remove $2j + 4$:



Proof

All patterns of α_j are 2-sortable: case-by-case analysis.
Remove $2j + 4$:



First algorithm: left greedy

Priorities of operations: $d_i \rhd d_j$ whenever $i > j$.

$B_{lg}(k) = \{\pi : \pi \text{ is sorted by the } D^k I \text{ machine using the left-greedy procedure}\}$

Proposition

For every $k \geq 1$, $B_{lg}(k) = Av(231)$.

First algorithm: left greedy

Priorities of operations: $d_i \rhd d_j$ whenever $i > j$.

$B_{lg}(k) = \{\pi : \pi \text{ is sorted by the } D^k I \text{ machine using the left-greedy procedure}\}$

Proposition

For every $k \geq 1$, $B_{lg}(k) = Av(231)$.

First algorithm: left greedy

Priorities of operations: $d_i \rhd d_j$ whenever $i > j$.

$B_{lg}(k) = \{\pi : \pi \text{ is sorted by the } D^k I \text{ machine using the left-greedy procedure}\}$

Proposition

For every $k \geq 1$, $B_{lg}(k) = Av(231)$.

Comparison with Stacksort

Thus the left greedy algorithm sorts precisely the same permutations as Stacksort does.

However, the two algorithms are not equivalent: for instance, when $k = 1$, on the input permutation 2341:

- the left greedy DI machine outputs 2134;
- Stacksort outputs 2314.

Comparison with Stacksort

Thus the left greedy algorithm sorts precisely the same permutations as Stacksort does.

However, the two algorithms are not equivalent: for instance, when $k = 1$, on the input permutation 2341:

- the left greedy DI machine outputs 2134;
- Stacksort outputs 2314.

Comparison with Stacksort

Thus the left greedy algorithm sorts precisely the same permutations as Stacksort does.

However, the two algorithms are not equivalent: for instance, when $k = 1$, on the input permutation 2341:

- the left greedy DI machine outputs 2134;
- Stacksort outputs 2314.

Comparison with Stacksort

$$\phi_k : S_n \longrightarrow S_n$$

$\phi_k(\pi) =$ the permutation which exits the last (i.e., the k-th) decreasing stack.

Of course, ϕ_k preserves the property of being a 231-avoider.
What further properties does ϕ_k have?
For instance, for any given π, the sequence $\{\phi_k(\pi)\}_{k\in\mathbb{N}}$ eventually becomes constant. But we do not know when this happens precisely.
For $\pi = 36257418$:

$$k = 1 : 36275418,$$
$$k = 2 : 37652841,$$
$$k = 3 : 37652841,$$
$$k = 4 : 38765241,$$
$$k = 5 : 38765241,$$
$$k = 6 : 38765241.$$

Comparison with Stacksort

$$\phi_k : S_n \longrightarrow S_n$$

$\phi_k(\pi) =$ the permutation which exits the last (i.e., the k-th) decreasing stack.

Of course, ϕ_k preserves the property of being a 231-avoider.
What further properties does ϕ_k have?
For instance, for any given π, the sequence $\{\phi_k(\pi)\}_{k\in\mathbb{N}}$ eventually becomes constant. But we do not know when this happens precisely.
For $\pi = 36257418$:

$$k = 1 : 36275418,$$
$$k = 2 : 37652841,$$
$$k = 3 : 37652841,$$
$$k = 4 : 38765241,$$
$$k = 5 : 38765241,$$
$$k = 6 : 38765241.$$

Comparison with Stacksort

$$\phi_k : S_n \longrightarrow S_n$$

$\phi_k(\pi) =$ the permutation which exits the last (i.e., the k-th) decreasing stack.

Of course, ϕ_k preserves the property of being a 231-avoider.
What further properties does ϕ_k have?
For instance, for any given π, the sequence $\{\phi_k(\pi)\}_{k\in\mathbb{N}}$ eventually becomes constant. But we do not know when this happens precisely.
For $\pi = 36257418$:

$$k = 1 : 36275418,$$
$$k = 2 : 37652841,$$
$$k = 3 : 37652841,$$
$$k = 4 : 38765241,$$
$$k = 5 : 38765241,$$
$$k = 6 : 38765241.$$

Second algorithm: almost left greedy

Priorities of operations: $d_{k+1} > d_{k-1} > d_{k-2} > \cdots > d_1 > d_0 > d_k$.
This means that the algorithm always performs the leftmost possible operation, except for the push operation into the increasing stack, which is performed last.

$B_{alg}(k) = \{\pi : \pi \text{ is sorted by the } D^k I \text{ machine using the almost left-greedy procedure}\}$

Unfortunately, $B_{alg}(k)$ is not in general a class: for $k = 2$, 631425 is sortable, but its pattern 52314 isn't.

Second algorithm: almost left greedy

Priorities of operations: $d_{k+1} > d_{k-1} > d_{k-2} > \cdots > d_1 > d_0 > d_k$.
This means that the algorithm always performs the leftmost possible operation, except for the push operation into the increasing stack, which is performed last.

$B_{alg}(k) = \{\pi : \pi \text{ is sorted by the } D^k I \text{ machine using the almost left-greedy procedure}\}$

Unfortunately, $B_{alg}(k)$ is not in general a class: for $k = 2$, 631425 is sortable, but its pattern 52314 isn't.

Second algorithm: almost left greedy

Priorities of operations: $d_{k+1} > d_{k-1} > d_{k-2} > \cdots > d_1 > d_0 > d_k$.
This means that the algorithm always performs the leftmost possible operation, except for the push operation into the increasing stack, which is performed last.

$B_{alg}(k) = \{\pi : \pi \text{ is sorted by the } D^k I \text{ machine using the almost left-greedy procedure}\}$

Unfortunately, $B_{alg}(k)$ is not in general a class: for $k = 2$, 631425 is sortable, but its pattern 52314 isn't.

The case $k = 2$

For the almost left greedy D^2I machine we only have some partial results.

Proposition

Let π be an almost left-greedy D^2I sortable permutation; then:

- *π avoids 3214;*
- *π avoids the following barred patterns, each of which is obtained by suitably adding barred elements to the pattern 52314:*
  - $63\bar{1}425$;
  - $7\bar{2}\bar{1}4536$, $7\bar{3}\bar{1}4526$;
  - $\bar{7}\bar{2}8\bar{1}4536$, $\bar{7}\bar{3}8\bar{1}4526$;
  - $\bar{8}\bar{2}7\bar{1}4536$, $\bar{8}\bar{3}7\bar{1}4526$.

The case $k = 2$

For the almost left greedy D^2I machine we only have some partial results.

Proposition

Let π be an almost left-greedy D^2I sortable permutation; then:

- *π avoids* 3214*;*
- *π avoids the following barred patterns, each of which is obtained by suitably adding barred elements to the pattern* 52314*:*
  - $63\bar{1}425$;
  - $7\bar{2}\bar{1}4536$, $7\bar{3}\bar{1}4526$;
  - $\bar{7}\bar{2}8\bar{1}4536$, $\bar{7}\bar{3}8\bar{1}4526$;
  - $8\bar{2}7\bar{1}4536$, $8\bar{3}7\bar{1}4526$.

The case $k = 2$

For the almost left greedy D^2I machine we only have some partial results.

Proposition

Let π be an almost left-greedy D^2I sortable permutation; then:

- *π avoids* 3214*;*
- *π avoids the following barred patterns, each of which is obtained by suitably adding barred elements to the pattern* 52314*:*
  - $63\bar{1}425$;
  - $7\bar{2}\bar{1}4536$, $7\bar{3}\bar{1}4526$;
  - $\bar{7}\bar{2}8\bar{1}4536$, $\bar{7}\bar{3}8\bar{1}4526$;
  - $\bar{8}\bar{2}7\bar{1}4536$, $\bar{8}\bar{3}7\bar{1}4526$.

The case $k = 2$

For the almost left greedy D^2I machine we only have some partial results.

Proposition

Let π be an almost left-greedy D^2I sortable permutation; then:

- *π avoids* 3214*;*
- *π avoids the following barred patterns, each of which is obtained by suitably adding barred elements to the pattern* 52314*:*
  - $63\bar{1}425$*;*
  - $7\bar{2}\bar{1}4536$, $7\bar{3}\bar{1}4526$*;*
  - $\bar{7}\bar{2}8\bar{1}4536$, $\bar{7}\bar{3}8\bar{1}4526$*;*
  - $\bar{8}\bar{2}7\bar{1}4536$, $\bar{8}\bar{3}7\bar{1}4526$.

The case $k = 2$

For the almost left greedy D^2I machine we only have some partial results.

Proposition

Let π be an almost left-greedy D^2I sortable permutation; then:

- *π avoids* 3214*;*
- *π avoids the following barred patterns, each of which is obtained by suitably adding barred elements to the pattern* 52314*:*
  - $63\bar{1}425$*;*
  - $7\bar{2}\bar{1}4536$, $7\bar{3}\bar{1}4526$*;*
  - $\bar{7}\bar{2}8\bar{1}4536$, $\bar{7}\bar{3}8\bar{1}4526$*;*
  - $\bar{8}\bar{2}7\bar{1}4536$, $\bar{8}\bar{3}7\bar{1}4526$.

The case $k = 2$

For the almost left greedy D^2I machine we only have some partial results.

Proposition

Let π be an almost left-greedy D^2I sortable permutation; then:

- *π avoids* 3214*;*
- *π avoids the following barred patterns, each of which is obtained by suitably adding barred elements to the pattern* 52314*:*
  - $63\bar{1}425$;
  - $7\bar{2}\bar{1}4536$, $7\bar{3}\bar{1}4526$;
  - $\bar{7}\bar{2}8\bar{1}4536$, $\bar{7}\bar{3}8\bar{1}4526$;
  - $\bar{8}\bar{2}7\bar{1}4536$, $\bar{8}\bar{3}7\bar{1}4526$.

The case $k = 2$

For the almost left greedy D^2I machine we only have some partial results.

Proposition

Let π be an almost left-greedy D^2I sortable permutation; then:

- *π avoids* 3214*;*
- *π avoids the following barred patterns, each of which is obtained by suitably adding barred elements to the pattern* 52314*:*
  - $63\bar{1}425$;
  - $7\bar{2}\bar{1}4536$, $7\bar{3}\bar{1}4526$;
  - $\bar{7}\bar{2}8\bar{1}4536$, $\bar{7}\bar{3}8\bar{1}4526$;
  - $\bar{8}\bar{2}7\bar{1}4536$, $\bar{8}\bar{3}7\bar{1}4526$.

The case $k = 2$

Proposition

Let π be a permutation that is not almost left-greedy D^2I sortable. Then one of the following cases holds:

- *π contains 3214;*
- *π contains one of the barred patterns listed above;*
- *π contains an occurrence of 52314 that extends to 82714536 (resp., 83714526) which in turn is part of one of the following patterns:*
  - 931825647 (resp., 941825637);
  - 1021493675 8 (resp., 1021593674 8);
  - 1031492675 8 (resp., 1031592674 8);
  - 10211493675 8 (resp., 10211593674 8);
  - 10311492675 8 (resp., 10311592674 8);
  - 11210149367 58 (resp., 10211593674 8);
  - 11310149267 58 (resp., 10311592674 8);

The case $k = 2$

Proposition

Let π be a permutation that is not almost left-greedy D^2I sortable. Then one of the following cases holds:

- *π contains* 3214*;*
- *π contains one of the barred patterns listed above;*
- *π contains an occurrence of* 52314 *that extends to* 82714536 *(resp.,* 83714526*) which in turn is part of one of the following patterns:*
  - 931825647 *(resp.,* 941825637*);*
  - 1021493675 8 *(resp.,* 1021593674 8*);*
  - 1031492675 8 *(resp.,* 1031592674 8*);*
  - 10211493675 8 *(resp.,* 10211593674 8*);*
  - 10311492675 8 *(resp.,* 10311592674 8*);*
  - 11210149367 58 *(resp.,* 10211159367 48*);*
  - 11310149267 58 *(resp.,* 10311159267 48*);*

The case $k = 2$

Proposition

Let π be a permutation that is not almost left-greedy D^2I sortable. Then one of the following cases holds:

- *π contains* 3214*;*
- *π contains one of the barred patterns listed above;*
- *π contains an occurrence of* 52314 *that extends to* 82714536 *(resp.,* 83714526*) which in turn is part of one of the following patterns:*
  - 931825647 *(resp.,* 941825637*);*
  - 1021493675 8 *(resp.,* 1021593674 8*);*
  - 1031492675 8 *(resp.,* 1031592674 8*);*
  - 10211493675 8 *(resp.,* 10211593674 8*);*
  - 10311492675 8 *(resp.,* 10311592674 8*);*
  - 11210149367 58 *(resp.,* 10211593674 8*);*
  - 11310149267 58 *(resp.,* 10311592674 8*);*

The case $k = 2$

Proposition

Let π be a permutation that is not almost left-greedy D^2I sortable. Then one of the following cases holds:

- *π contains* 3214*;*
- *π contains one of the barred patterns listed above;*
- *π contains an occurrence of* 52314 *that extends to* 82714536 *(resp.,* 83714526*) which in turn is part of one of the following patterns:*
  - 931825647 *(resp.,* 941825637*);*
  - 10214936758 *(resp.,* 10215936748*);*
  - 10314926758 *(resp.,* 10315926748*);*
  - 102114936758 *(resp.,* 102115936748*);*
  - 103114926758 *(resp.,* 103115926748*);*
  - 112104936758 *(resp.,* 102115936748*);*
  - 113104926758 *(resp.,* 103115926748*);*

The case $k = 2$

Proposition

Let π be a permutation that is not almost left-greedy D^2I sortable. Then one of the following cases holds:

- *π contains* 3214*;*
- *π contains one of the barred patterns listed above;*
- *π contains an occurrence of* 52314 *that extends to* 82714536 *(resp.,* 83714526*) which in turn is part of one of the following patterns:*
  - 9 3 1 8 2 5 6 4 7 *(resp.,* 9 4 1 8 2 5 6 3 7*);*
  - 10 2 1 4 9 3 6 7 5 8 *(resp.,* 10 2 1 5 9 3 6 7 4 8*);*
  - 10 3 1 4 9 2 6 7 5 8 *(resp.,* 10 3 1 5 9 2 6 7 4 8*);*
  - 10 2 11 1 4 9 3 6 7 5 8 *(resp.,* 10 2 11 1 5 9 3 6 7 4 8*);*
  - 10 3 11 1 4 9 2 6 7 5 8 *(resp.,* 10 3 11 1 5 9 2 6 7 4 8*);*
  - 11 2 10 1 4 9 3 6 7 5 8 *(resp.,* 10 2 11 1 5 9 3 6 7 4 8*);*
  - 11 3 10 1 4 9 2 6 7 5 8 *(resp.,* 10 3 11 1 5 9 2 6 7 4 8*);*

The case $k = 2$

Proposition

Let π be a permutation that is not almost left-greedy D^2I sortable. Then one of the following cases holds:

- *π contains* 3214*;*
- *π contains one of the barred patterns listed above;*
- *π contains an occurrence of* 52314 *that extends to* 82714536 *(resp.,* 83714526*) which in turn is part of one of the following patterns:*
  - 9 3 1 8 2 5 6 4 7 *(resp.,* 9 4 1 8 2 5 6 3 7*);*
  - 10 2 1 4 9 3 6 7 5 8 *(resp.,* 10 2 1 5 9 3 6 7 4 8*);*
  - 10 3 1 4 9 2 6 7 5 8 *(resp.,* 10 3 1 5 9 2 6 7 4 8*);*
  - 10 2 11 1 4 9 3 6 7 5 8 *(resp.,* 10 2 11 1 5 9 3 6 7 4 8*);*
  - 10 3 11 1 4 9 2 6 7 5 8 *(resp.,* 10 3 11 1 5 9 2 6 7 4 8*);*
  - 11 2 10 1 4 9 3 6 7 5 8 *(resp.,* 10 2 11 1 5 9 3 6 7 4 8*);*
  - 11 3 10 1 4 9 2 6 7 5 8 *(resp.,* 10 3 11 1 5 9 2 6 7 4 8*);*

The case $k = 2$

Proposition

Let π be a permutation that is not almost left-greedy D^2I sortable. Then one of the following cases holds:

- *π contains* 3214*;*
- *π contains one of the barred patterns listed above;*
- *π contains an occurrence of* 52314 *that extends to* 82714536 *(resp.,* 83714526*) which in turn is part of one of the following patterns:*
  - 9 3 1 8 2 5 6 4 7 *(resp.,* 9 4 1 8 2 5 6 3 7*);*
  - 10 2 1 4 9 3 6 7 5 8 *(resp.,* 10 2 1 5 9 3 6 7 4 8*);*
  - 10 3 1 4 9 2 6 7 5 8 *(resp.,* 10 3 1 5 9 2 6 7 4 8*);*
  - 10 2 11 1 4 9 3 6 7 5 8 *(resp.,* 10 2 11 1 5 9 3 6 7 4 8*);*
  - 10 3 11 1 4 9 2 6 7 5 8 *(resp.,* 10 3 11 1 5 9 2 6 7 4 8*);*
  - 11 2 10 1 4 9 3 6 7 5 8 *(resp.,* 10 2 11 1 5 9 3 6 7 4 8*);*
  - 11 3 10 1 4 9 2 6 7 5 8 *(resp.,* 10 3 11 1 5 9 2 6 7 4 8*);*

The case $k = 2$

Proposition

Let π be a permutation that is not almost left-greedy D^2I sortable. Then one of the following cases holds:

- *π contains* 3214*;*
- *π contains one of the barred patterns listed above;*
- *π contains an occurrence of* 52314 *that extends to* 82714536 *(resp.,* 83714526*) which in turn is part of one of the following patterns:*
  - 9 3 1 8 2 5 6 4 7 *(resp.,* 9 4 1 8 2 5 6 3 7*);*
  - 10 2 1 4 9 3 6 7 5 8 *(resp.,* 10 2 1 5 9 3 6 7 4 8*);*
  - 10 3 1 4 9 2 6 7 5 8 *(resp.,* 10 3 1 5 9 2 6 7 4 8*);*
  - 10 2 11 1 4 9 3 6 7 5 8 *(resp.,* 10 2 11 1 5 9 3 6 7 4 8*);*
  - 10 3 11 1 4 9 2 6 7 5 8 *(resp.,* 10 3 11 1 5 9 2 6 7 4 8*);*
  - 11 2 10 1 4 9 3 6 7 5 8 *(resp.,* 10 2 11 1 5 9 3 6 7 4 8*);*
  - 11 3 10 1 4 9 2 6 7 5 8 *(resp.,* 10 3 11 1 5 9 2 6 7 4 8*);*

The case $k = 2$

Proposition

Let π be a permutation that is not almost left-greedy D^2I sortable. Then one of the following cases holds:

- *π contains* 3214*;*
- *π contains one of the barred patterns listed above;*
- *π contains an occurrence of* 52314 *that extends to* 82714536 *(resp.,* 83714526*) which in turn is part of one of the following patterns:*
  - 9 3 1 8 2 5 6 4 7 *(resp.,* 9 4 1 8 2 5 6 3 7*);*
  - 10 2 1 4 9 3 6 7 5 8 *(resp.,* 10 2 1 5 9 3 6 7 4 8*);*
  - 10 3 1 4 9 2 6 7 5 8 *(resp.,* 10 3 1 5 9 2 6 7 4 8*);*
  - 10 2 11 1 4 9 3 6 7 5 8 *(resp.,* 10 2 11 1 5 9 3 6 7 4 8*);*
  - 10 3 11 1 4 9 2 6 7 5 8 *(resp.,* 10 3 11 1 5 9 2 6 7 4 8*);*
  - 11 2 10 1 4 9 3 6 7 5 8 *(resp.,* 10 2 11 1 5 9 3 6 7 4 8*);*
  - 11 3 10 1 4 9 2 6 7 5 8 *(resp.,* 10 3 11 1 5 9 2 6 7 4 8*);*

The case $k = 2$

Proposition

Let π be a permutation that is not almost left-greedy D^2I sortable. Then one of the following cases holds:

- *π contains* 3214*;*
- *π contains one of the barred patterns listed above;*
- *π contains an occurrence of* 52314 *that extends to* 82714536 *(resp.,* 83714526*) which in turn is part of one of the following patterns:*
  - 9 3 1 8 2 5 6 4 7 *(resp.,* 9 4 1 8 2 5 6 3 7*);*
  - 10 2 1 4 9 3 6 7 5 8 *(resp.,* 10 2 1 5 9 3 6 7 4 8*);*
  - 10 3 1 4 9 2 6 7 5 8 *(resp.,* 10 3 1 5 9 2 6 7 4 8*);*
  - 10 2 11 1 4 9 3 6 7 5 8 *(resp.,* 10 2 11 1 5 9 3 6 7 4 8*);*
  - 10 3 11 1 4 9 2 6 7 5 8 *(resp.,* 10 3 11 1 5 9 2 6 7 4 8*);*
  - 11 2 10 1 4 9 3 6 7 5 8 *(resp.,* 10 2 11 1 5 9 3 6 7 4 8*);*
  - 11 3 10 1 4 9 2 6 7 5 8 *(resp.,* 10 3 11 1 5 9 2 6 7 4 8*);*

The case $k = 2$

Proposition

Let π be a permutation that is not almost left-greedy D^2I sortable. Then one of the following cases holds:

- *π contains* 3214*;*
- *π contains one of the barred patterns listed above;*
- *π contains an occurrence of* 52314 *that extends to* 82714536 *(resp.,* 83714526*) which in turn is part of one of the following patterns:*
  - 9 3 1 8 2 5 6 4 7 *(resp.,* 9 4 1 8 2 5 6 3 7*);*
  - 10 2 1 4 9 3 6 7 5 8 *(resp.,* 10 2 1 5 9 3 6 7 4 8*);*
  - 10 3 1 4 9 2 6 7 5 8 *(resp.,* 10 3 1 5 9 2 6 7 4 8*);*
  - 10 2 11 1 4 9 3 6 7 5 8 *(resp.,* 10 2 11 1 5 9 3 6 7 4 8*);*
  - 10 3 11 1 4 9 2 6 7 5 8 *(resp.,* 10 3 11 1 5 9 2 6 7 4 8*);*
  - 11 2 10 1 4 9 3 6 7 5 8 *(resp.,* 10 2 11 1 5 9 3 6 7 4 8*);*
  - 11 3 10 1 4 9 2 6 7 5 8 *(resp.,* 10 3 11 1 5 9 2 6 7 4 8*);*

The case $k = 2$

In fact, we can generate an infinite sequence of permutations $(\gamma_n)_{n\in\mathbb{N}}$, with $\gamma_n \in S_{3n+2}$, such that $\gamma_n \leq \gamma_{n+1}$ for all n, and permutations having even index are sortable, whereas permutations having odd index are not.

Formally,

$$\gamma_n = 3n+2,\ \underbrace{2\ 3n+1\ 1}_{231},\ \underbrace{4\ 3n\ 3}_{231}\cdots\underbrace{2n-2\ 2n+3\ 2n-3}_{231}\ \underbrace{2n\ 2n+1\ 2n-1}_{231}\ 2n+2,$$

and we have:

- $\gamma_1 = 52314 \notin B_{alg}(2)$;
- $\gamma_2 = 82714536 \in B_{alg}(2)$;
- $\gamma_3 = 11\ 2\ 10\ 14936758 \notin B_{alg}(2)$;
- ⋮

The case $k = 2$

In fact, we can generate an infinite sequence of permutations $(\gamma_n)_{n\in\mathbb{N}}$, with $\gamma_n \in S_{3n+2}$, such that $\gamma_n \leq \gamma_{n+1}$ for all n, and permutations having even index are sortable, whereas permutations having odd index are not.

Formally,

$$\gamma_n = 3n+2,\ \underbrace{2\ 3n+1\ 1}_{231},\ \underbrace{4\ 3n\ 3}_{231}\cdots\underbrace{2n-2\ 2n+3\ 2n-3}_{231}\ \underbrace{2n\ 2n+1\ 2n-1}_{231}\ 2n+2,$$

and we have:

- $\gamma_1 = 52314 \notin B_{alg}(2)$;
- $\gamma_2 = 82714536 \in B_{alg}(2)$;
- $\gamma_3 = 11\ 2\ 10\ 14936758 \notin B_{alg}(2)$;

⋮

The case $k = 2$

In fact, we can generate an infinite sequence of permutations $(\gamma_n)_{n\in\mathbb{N}}$, with $\gamma_n \in S_{3n+2}$, such that $\gamma_n \leq \gamma_{n+1}$ for all n, and permutations having even index are sortable, whereas permutations having odd index are not.

Formally,

$$\gamma_n = 3n+2,\ \underbrace{2\ 3n+1\ 1}_{231},\ \underbrace{4\ 3n\ 3}_{231}\cdots\underbrace{2n-2\ 2n+3\ 2n-3}_{231}\ \underbrace{2n\ 2n+1\ 2n-1}_{231}\ 2n+2,$$

and we have:

- $\gamma_1 = 52314 \notin B_{alg}(2)$;
- $\gamma_2 = 82714536 \in B_{alg}(2)$;
- $\gamma_3 = 11\ 2\ 10\ 14936758 \notin B_{alg}(2)$;

⋮

The case $k = 2$

In fact, we can generate an infinite sequence of permutations $(\gamma_n)_{n\in\mathbb{N}}$, with $\gamma_n \in S_{3n+2}$, such that $\gamma_n \leq \gamma_{n+1}$ for all n, and permutations having even index are sortable, whereas permutations having odd index are not.

Formally,

$$\gamma_n = 3n+2,\ \underbrace{2\ 3n+1\ 1}_{231},\ \underbrace{4\ 3n\ 3}_{231}\cdots\underbrace{2n-2\ 2n+3\ 2n-3}_{231}\ \underbrace{2n\ 2n+1\ 2n-1}_{231}\ 2n+2,$$

and we have:

- $\gamma_1 = 52314 \notin B_{alg}(2)$;
- $\gamma_2 = 82714536 \in B_{alg}(2)$;
- $\gamma_3 = 11\ 2\ 10\ 14936758 \notin B_{alg}(2)$;

$\vdots$

What next?

There are several interesting things that are still to explore.

- Smarter algorithms for the $D^k I$ machine? An optimal one (at least in the $k = 2$ case)?
- What about making two passes from Rebecca's DI machine? Analogies with West-2-stack-sortable permutations?
- Enumerations?

What next?

There are several interesting things that are still to explore.

- Smarter algorithms for the $D^k I$ machine? An optimal one (at least in the $k = 2$ case)?
- What about making two passes from Rebecca's DI machine? Analogies with West-2-stack-sortable permutations?
- Enumerations?

What next?

There are several interesting things that are still to explore.

- Smarter algorithms for the $D^k I$ machine? An optimal one (at least in the $k = 2$ case)?
- What about making two passes from Rebecca's DI machine? Analogies with West-2-stack-sortable permutations?
- Enumerations?