

A code for square permutations

ENRICA DUCHI

IRIF, Université Paris Diderot

Permutation Patterns 2019

Summary of the talk

Square permutations and a generating tree

A code for square permutations

Extension to convex permutominoes

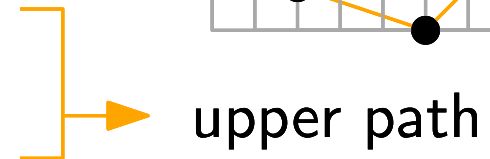
Square permutations

Square permutation

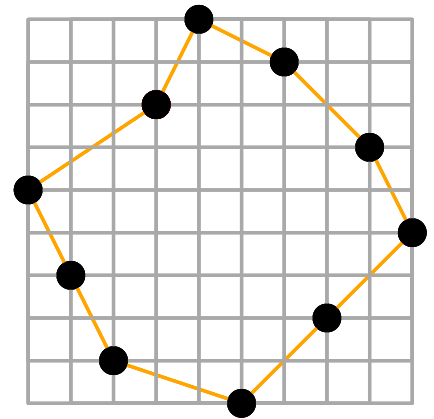
- = permutation without interior point
- = all points are upper left, upper right, down left or down right

Upper left points form the upper left path

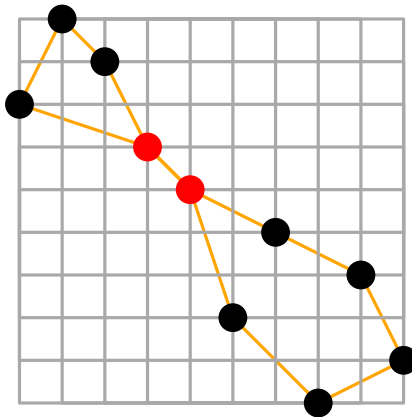
Upper right points form the upper right path



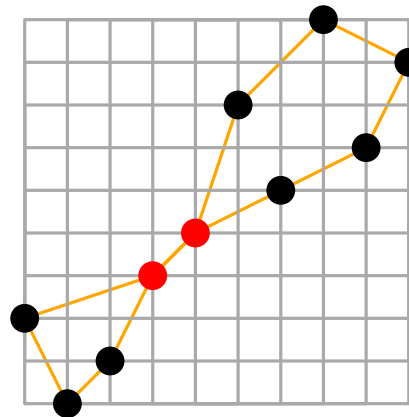
idem for down-left, down-right, and left, right and down path.



Square permutations can have **double points**:



down-left and upper-right



upper-left and down-right

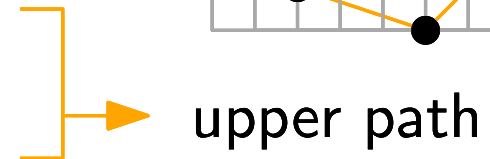
Square permutations

Square permutation

- = permutation without interior point
- = all points are upper left, upper right, down left or down right

Upper left points form the upper left path

Upper right points form the upper right path

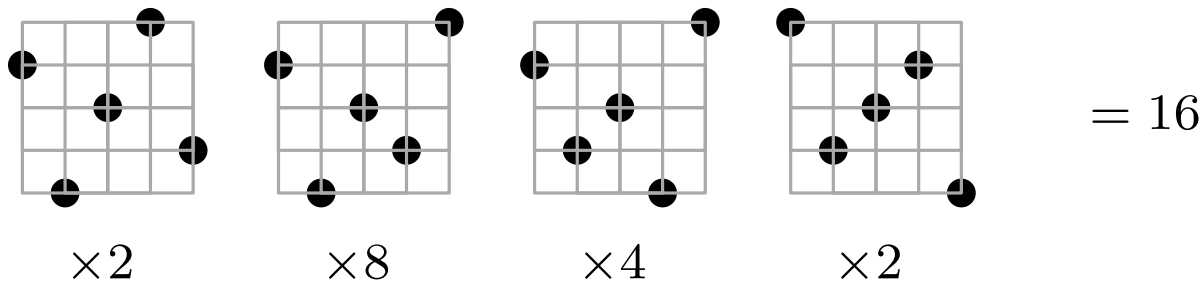


idem for down-left, down-right, and left, right and down path.

Our aim is to count these square permutations

First terms: 1, 2, 6, 24, 104=120-16, ...

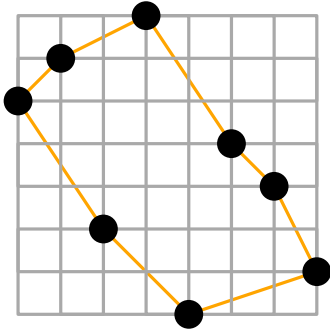
Indeed the smallest non-square permutations have size 5:



Enumeration via generating trees

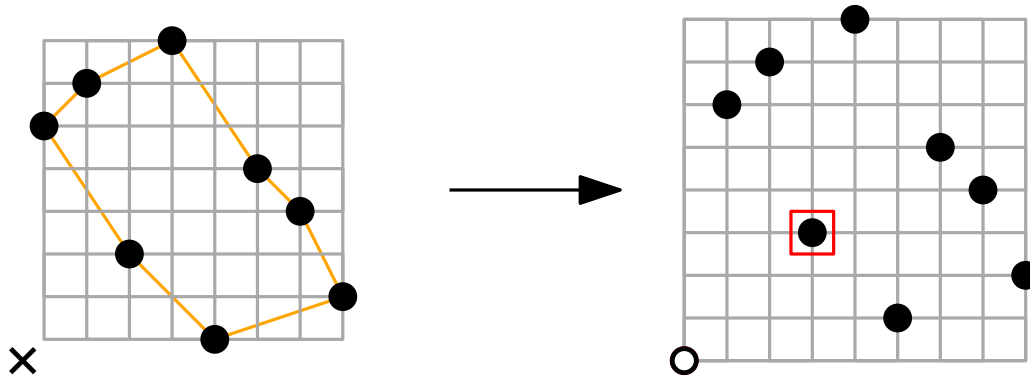
How to grow a **square** permutation?

Insert new points without creating internal points !



How to grow a **square** permutation?

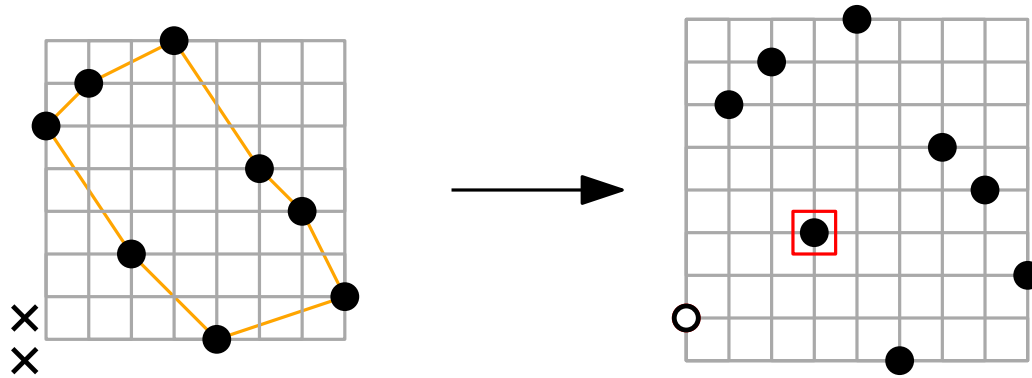
Insert new points without creating internal points !



No!

How to grow a **square** permutation?

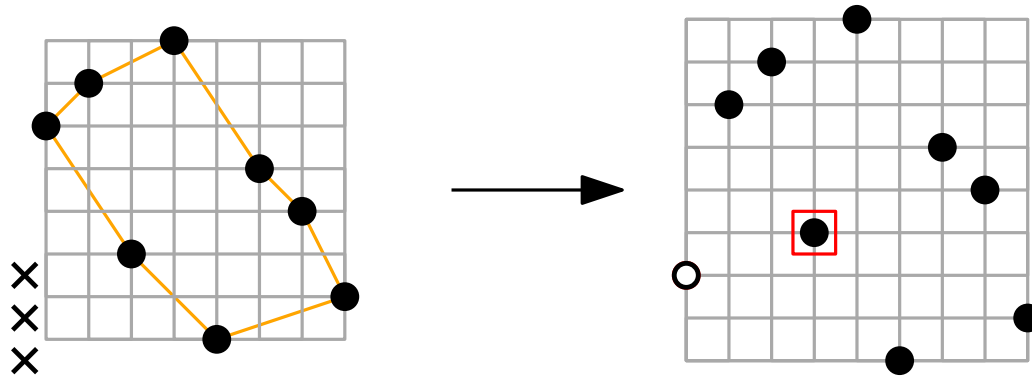
Insert new points without creating internal points !



No!

How to grow a **square** permutation?

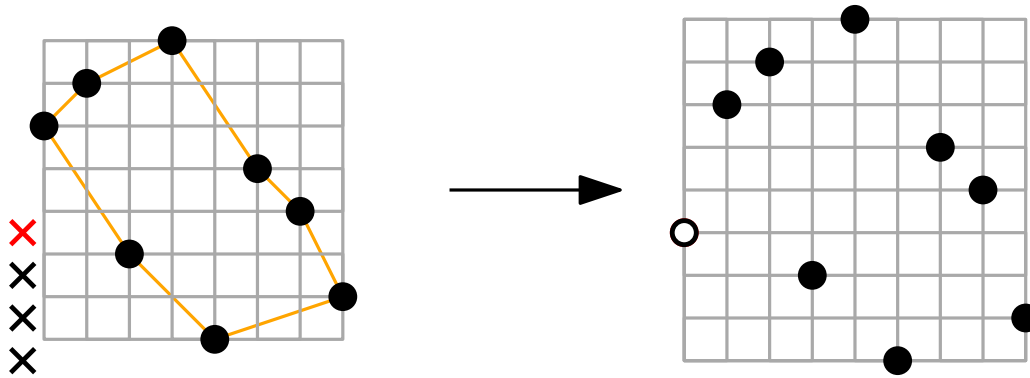
Insert new points without creating internal points !



No!

How to grow a **square** permutation?

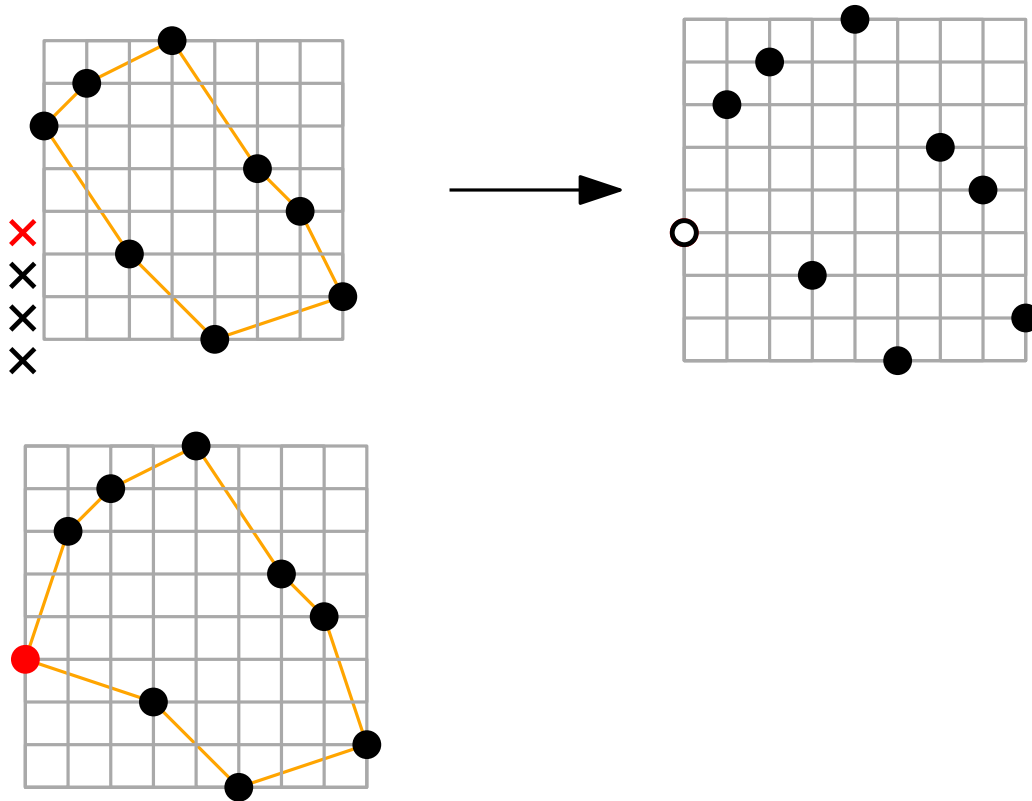
Insert new points without creating internal points !



Yes!

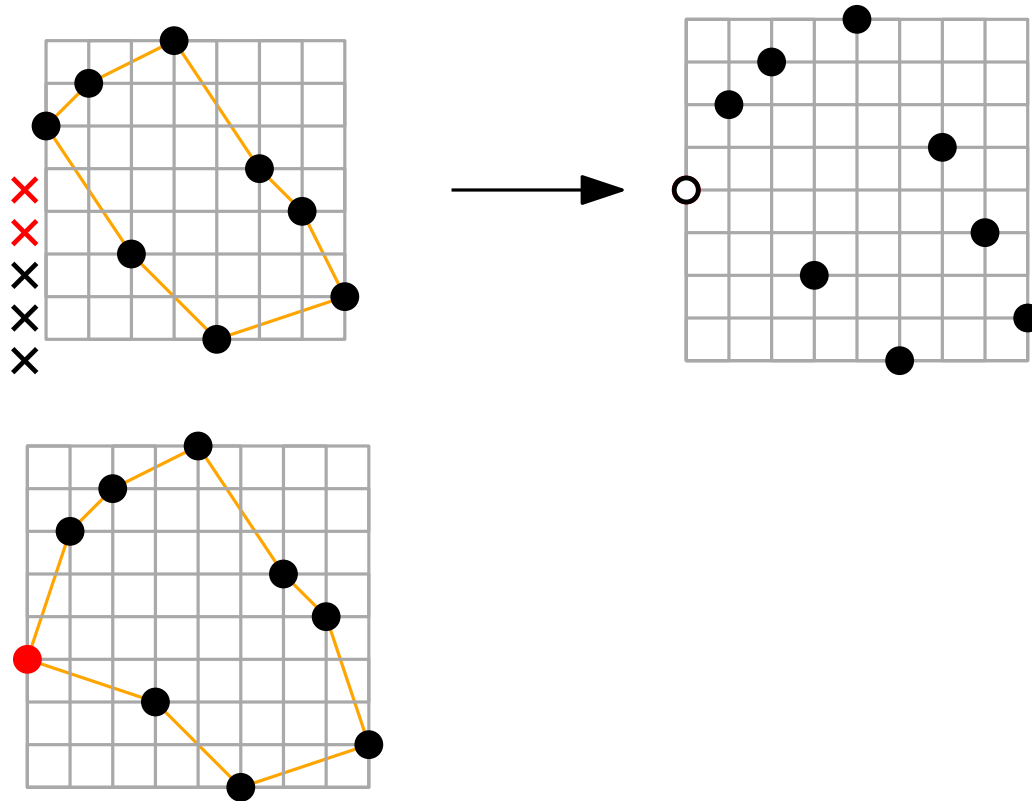
How to grow a **square** permutation?

Insert new points without creating internal points !



How to grow a **square** permutation?

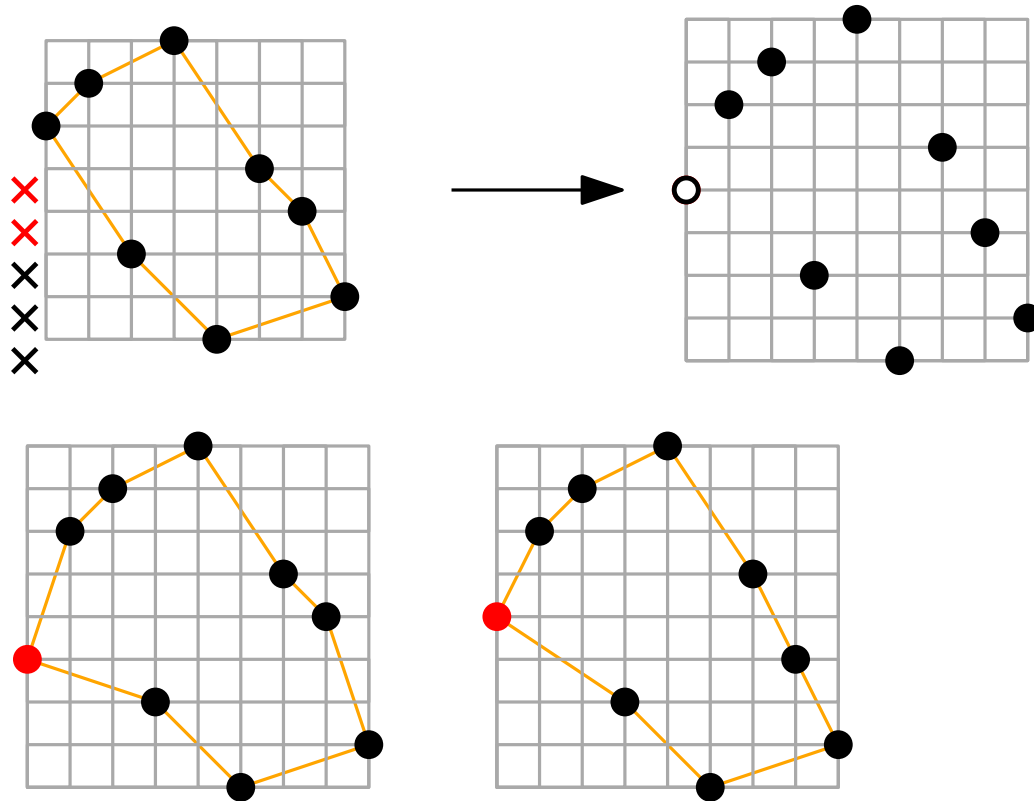
Insert new points without creating internal points !



Yes!

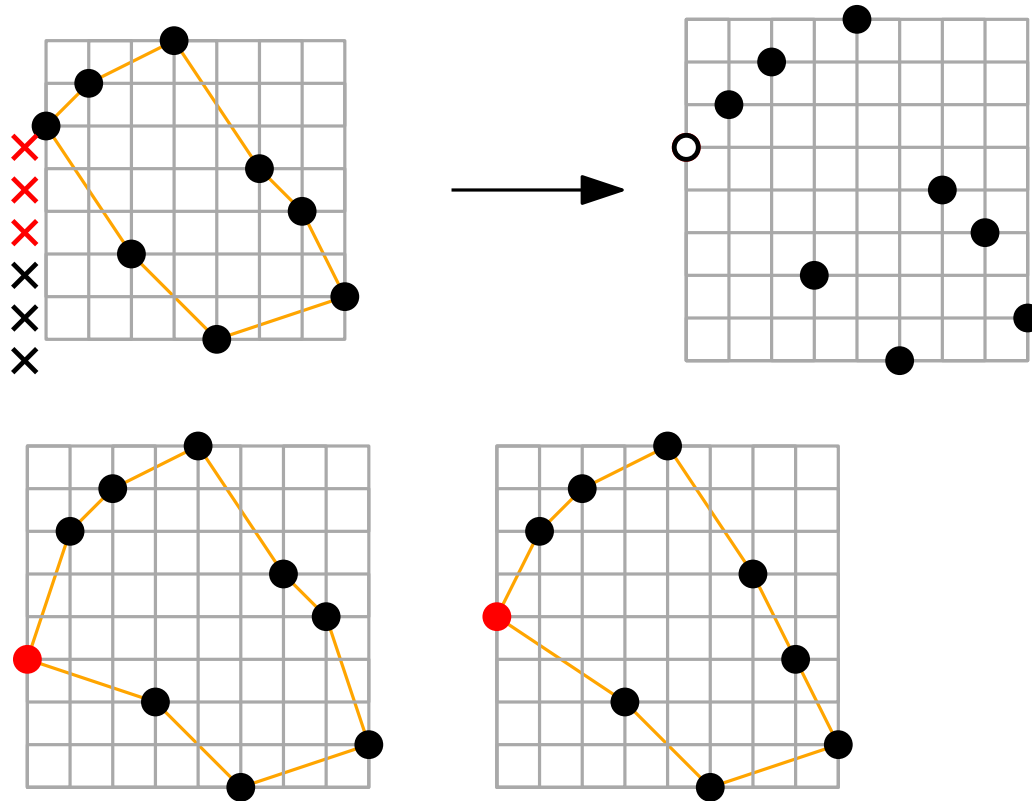
How to grow a **square** permutation?

Insert new points without creating internal points !



How to grow a **square** permutation?

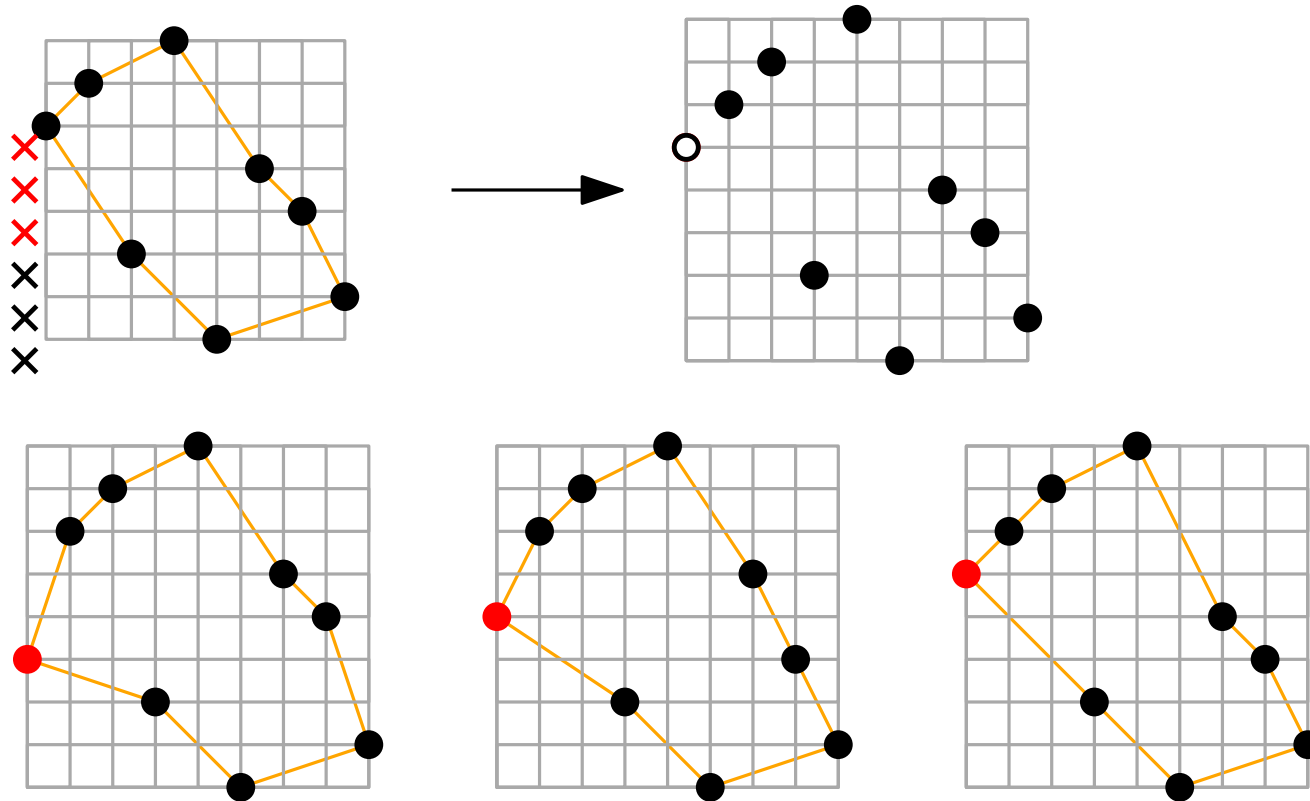
Insert new points without creating internal points !



Yes!

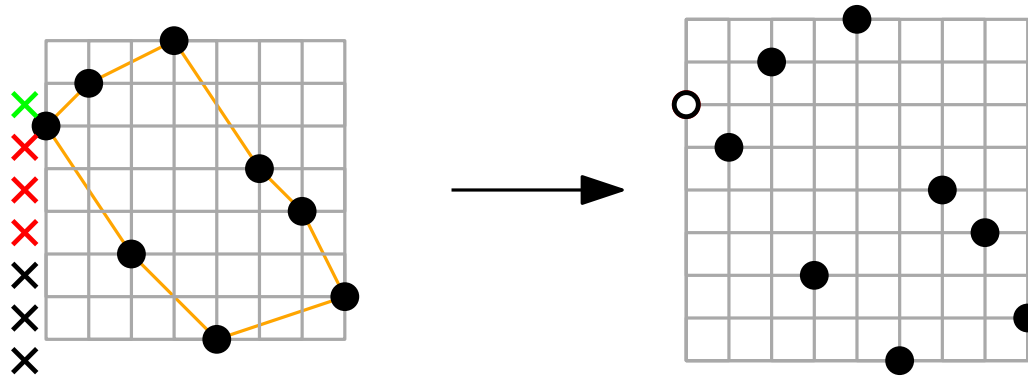
How to grow a **square** permutation?

Insert new points without creating internal points !

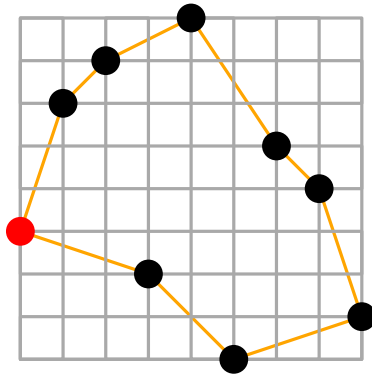
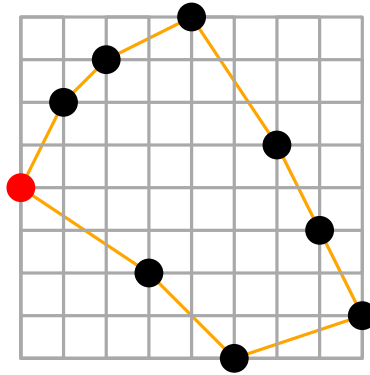
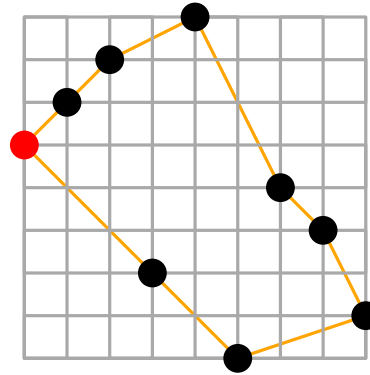


How to grow a **square** permutation?

Insert new points without creating internal points !

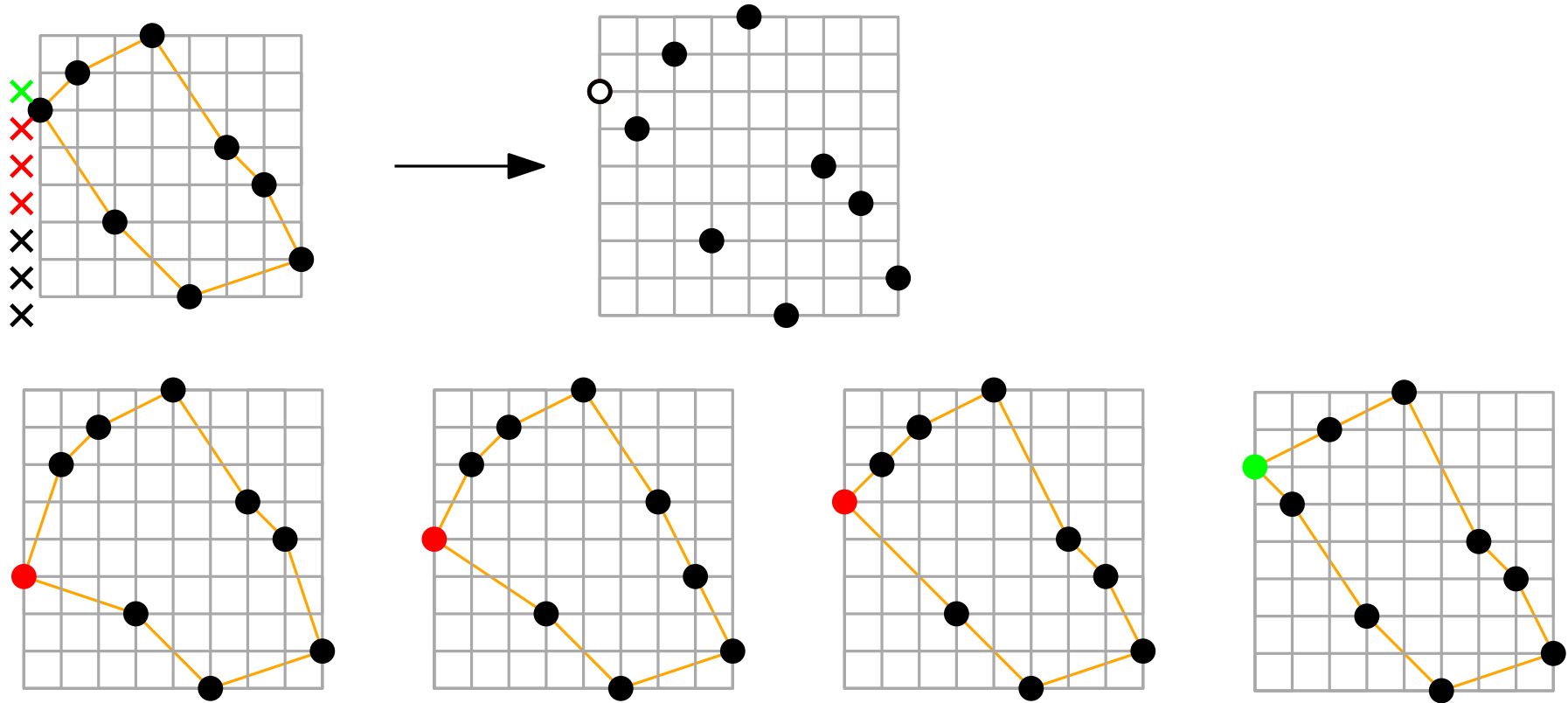


Yes!



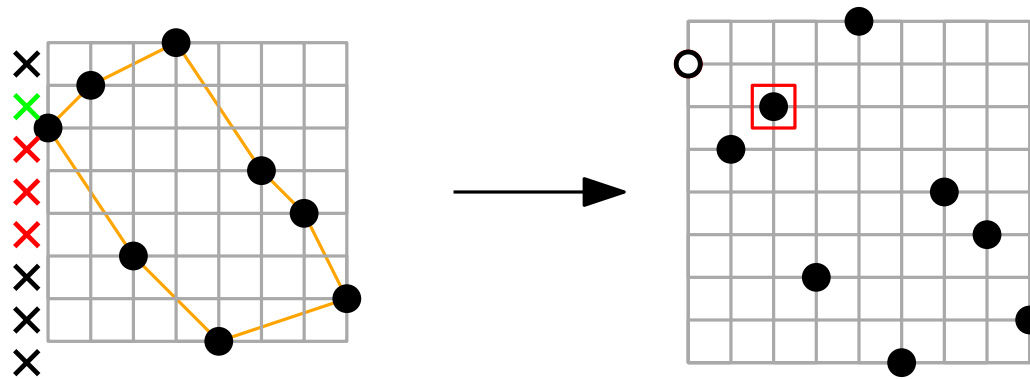
How to grow a **square** permutation?

Insert new points without creating internal points !

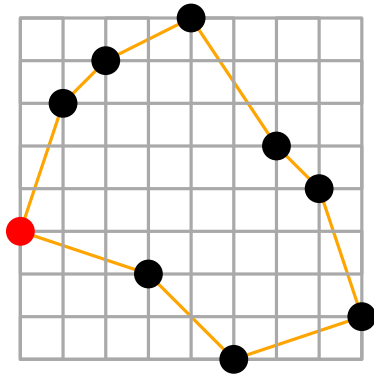
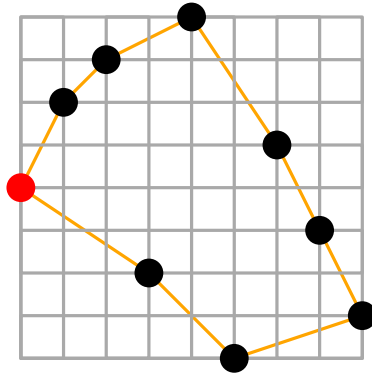
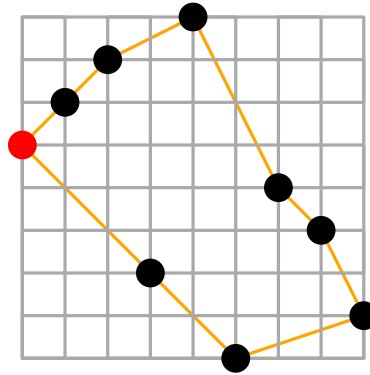
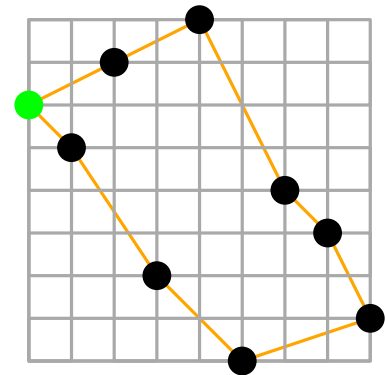


How to grow a **square** permutation?

Insert new points without creating internal points !

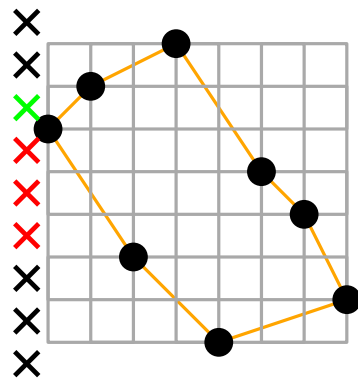
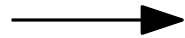
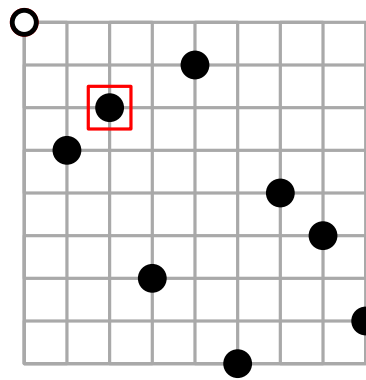


No!

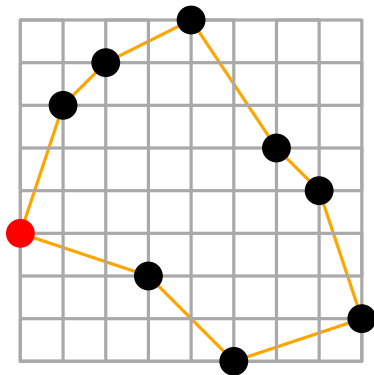
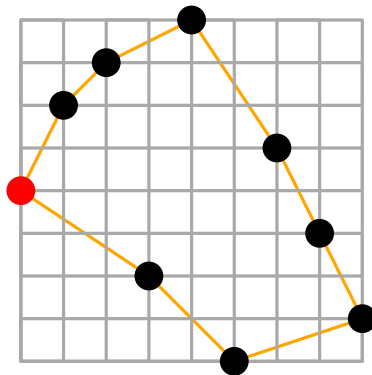
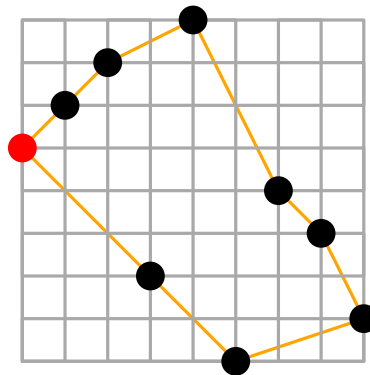
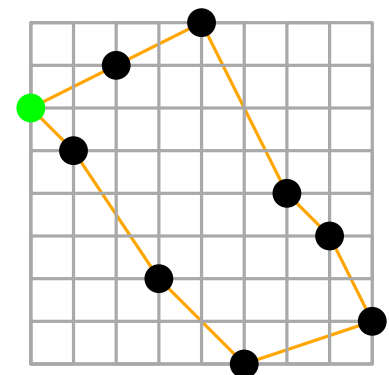


How to grow a **square** permutation?

Insert new points without creating internal points !

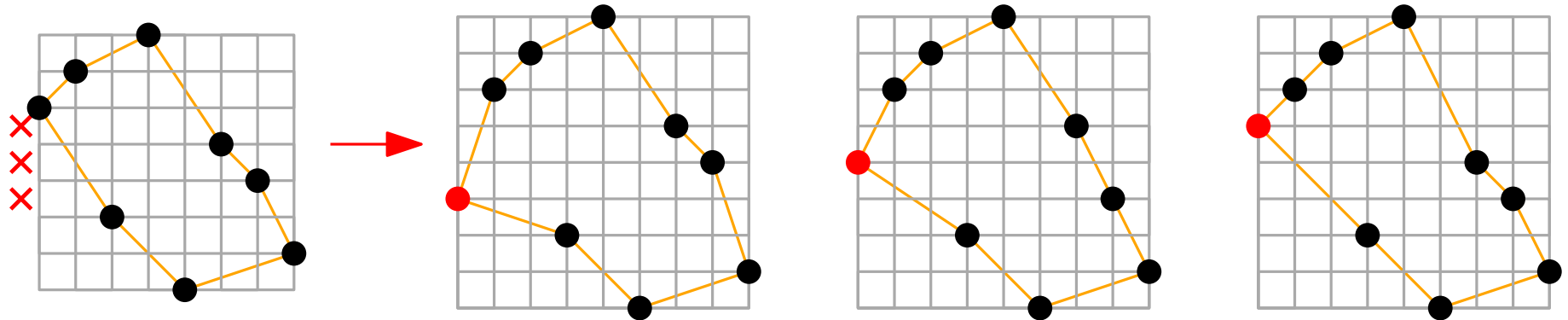


No!



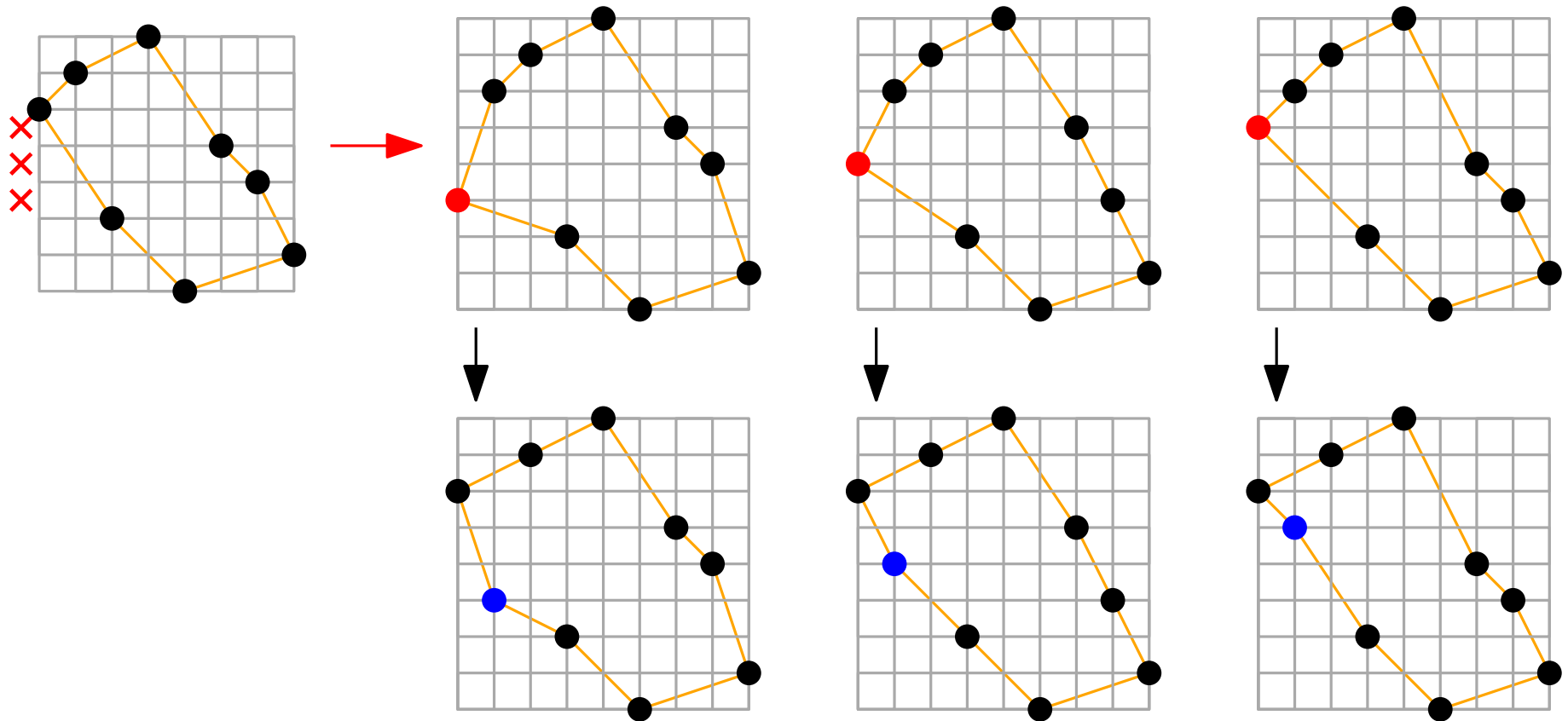
How to grow a square permutation?

The red **active sites**, $\times$, generate permutations σ' with $\sigma'(1) < \sigma'(2)$



How to grow a square permutation?

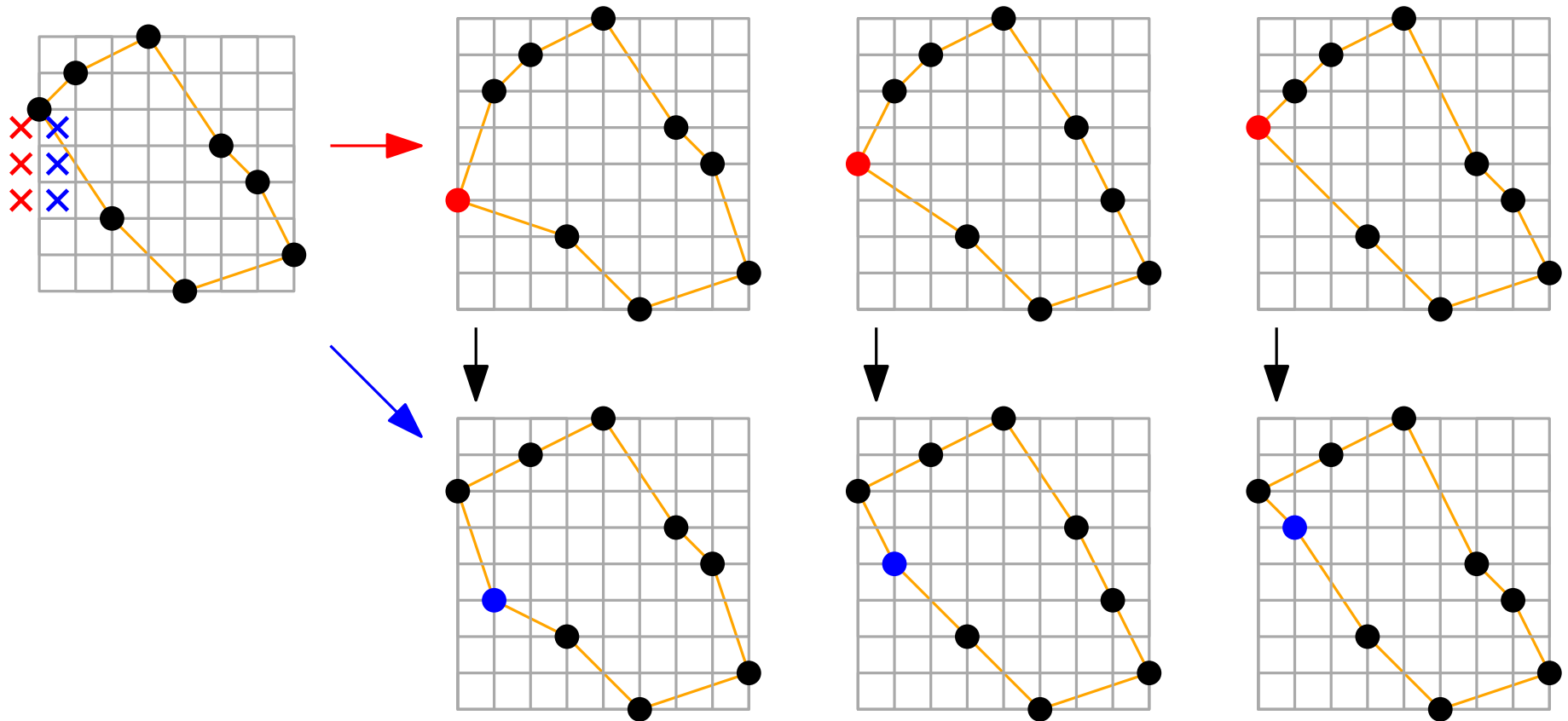
The red **active sites**, $\times$, generate permutations σ' with $\sigma'(1) < \sigma'(2)$



To generate σ' with $\sigma'(1) > \sigma'(2)$ we switch the first two column

How to grow a square permutation?

The red **active sites**, $\times$, generate permutations σ' with $\sigma'(1) < \sigma'(2)$



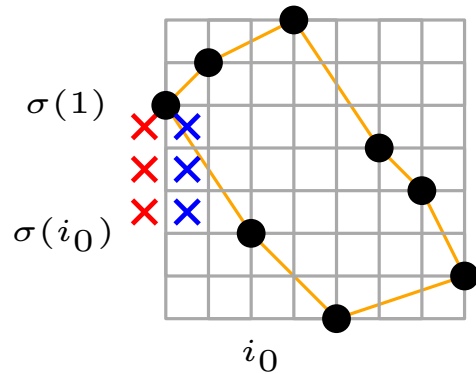
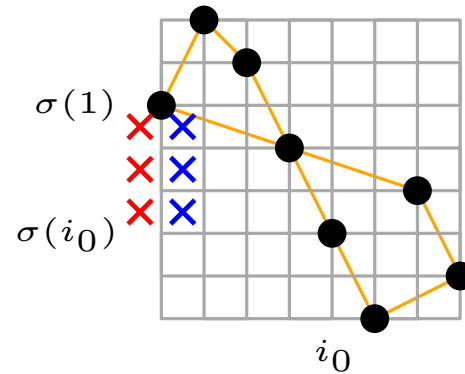
To generate σ' with $\sigma'(1) > \sigma'(2)$ we switch the first two column

Equivalently we generate them by using blue active sites, $\times$

How to grow a square permutation?

Insertions are possible between $\sigma(1)$ and $\sigma(i_0)$ where

$(i_0, \sigma(i_0))$ is the leftmost lower point which is not a double point.

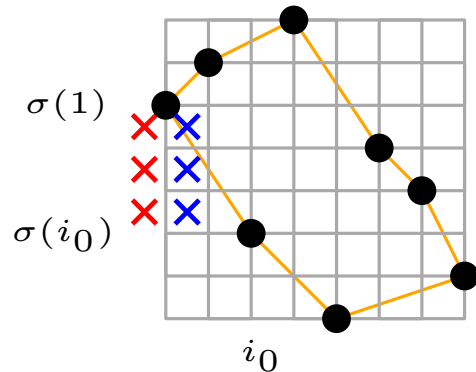
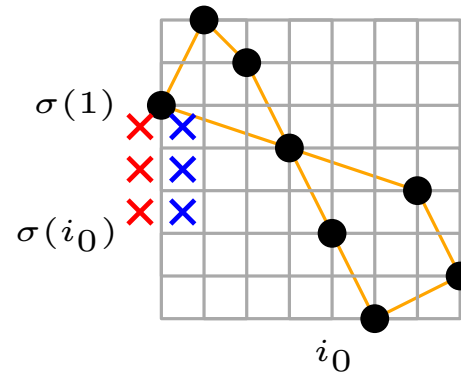


Indeed observe that double points do not block insertion.

How to grow a square permutation?

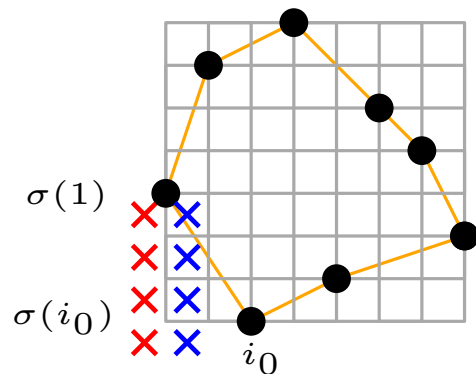
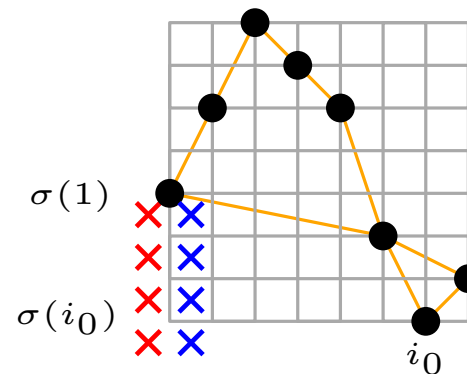
Insertions are possible between $\sigma(1)$ and $\sigma(i_0)$ where

$(i_0, \sigma(i_0))$ is the leftmost lower point which is not a double point.



Indeed observe that double points do not block insertion.

When $\sigma(i_0) = 1$, insertion is also possible at the bottom row:



How to grow a square permutation?

Let the label $k(\sigma)$ of a permutation σ be its number of red active sites.

How to grow a square permutation?

Let the label $k(\sigma)$ of a permutation σ be its number of red active sites.

Moreover let $\theta(\sigma)$ denote the set of $2k(\sigma)$ permutations obtained from σ by insertion at an active site.

$$\theta \left(\begin{array}{|c|c|c|c|} \hline \times & \times & & \\ \hline \times & \times & & \\ \hline & & & \\ \hline \end{array} \right) = \left\{ \begin{array}{|c|c|c|c|} \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline \end{array} \right\}$$

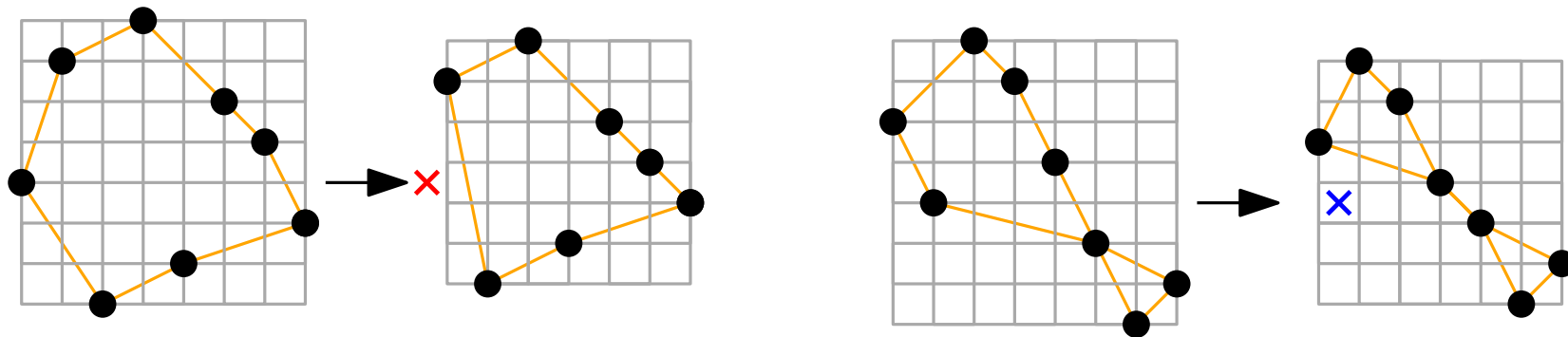
How to grow a square permutation?

Let the label $k(\sigma)$ of a permutation σ be its number of red active sites.

Moreover let $\theta(\sigma)$ denote the set of $2k(\sigma)$ permutations obtained from σ by insertion at an active site.

$$\theta\left(\begin{array}{|c|c|c|c|} \hline \bullet & \times & \times & \\ \hline \times & \times & & \\ \hline & & \bullet & \\ \hline & & \bullet & \\ \hline & & & \bullet \\ \hline \end{array}\right) = \left\{ \begin{array}{|c|c|c|c|} \hline \bullet & & & \\ \hline \bullet & & & \\ \hline & & \bullet & \\ \hline & & \bullet & \\ \hline & & & \bullet \\ \hline \end{array}, \begin{array}{|c|c|c|c|} \hline \bullet & & & \\ \hline & \bullet & & \\ \hline & & \bullet & \\ \hline & & \bullet & \\ \hline & & & \bullet \\ \hline \end{array}, \begin{array}{|c|c|c|c|} \hline \bullet & & & \\ \hline \bullet & & & \\ \hline \bullet & & & \\ \hline & & \bullet & \\ \hline & & & \bullet \\ \hline \end{array}, \begin{array}{|c|c|c|c|} \hline \bullet & \bullet & & \\ \hline & & \bullet & \\ \hline & & \bullet & \\ \hline & & & \bullet \\ \hline & & & \\ \hline \end{array} \right\}$$

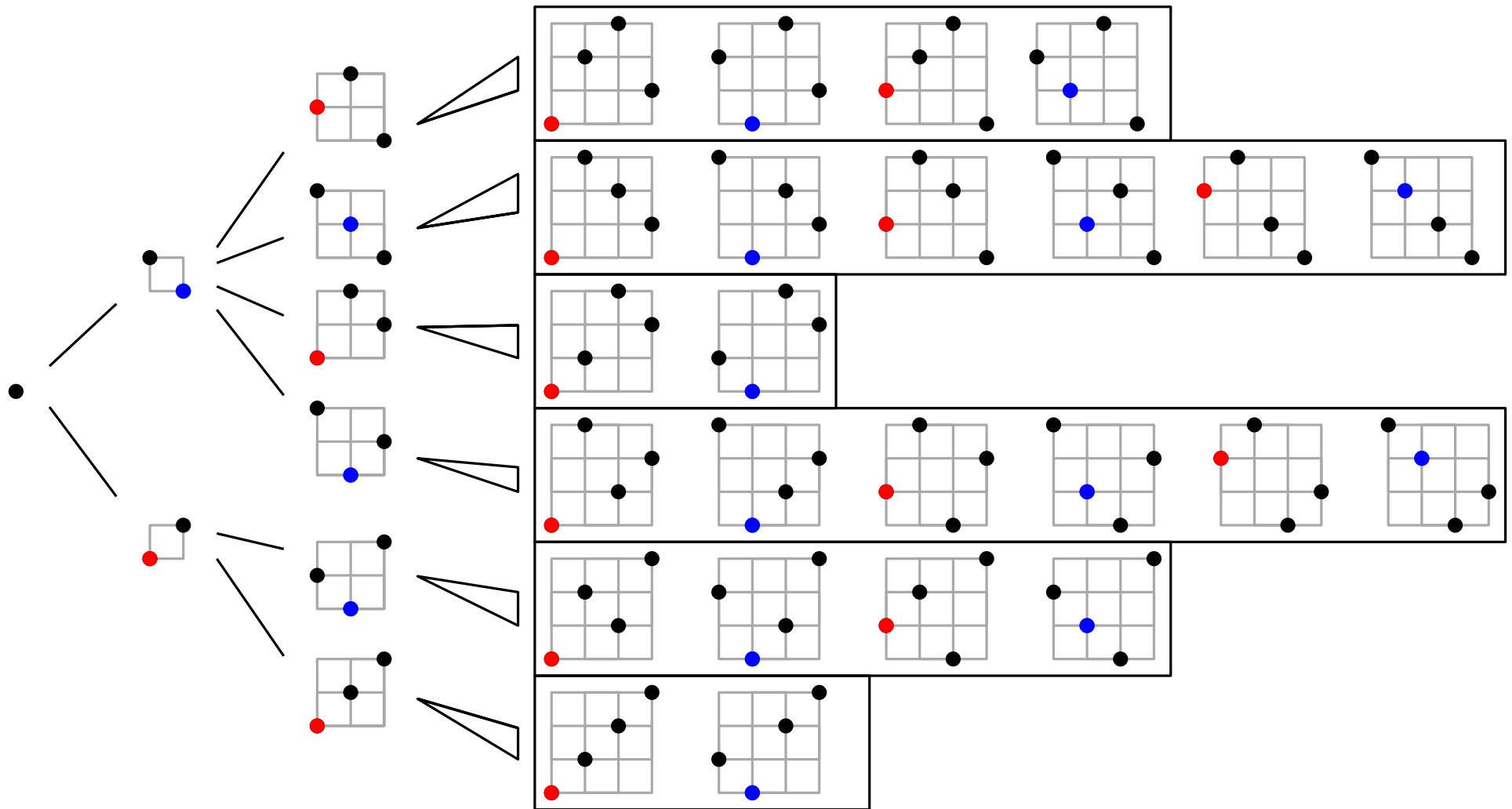
Proposition. For any square permutation σ' of size $n \geq 2$, there is a unique square permutation σ such that $\sigma' \in \theta(\sigma)$.



σ is obtained from σ' by removing the lowest among $\sigma(1)$ and $\sigma(2)$.

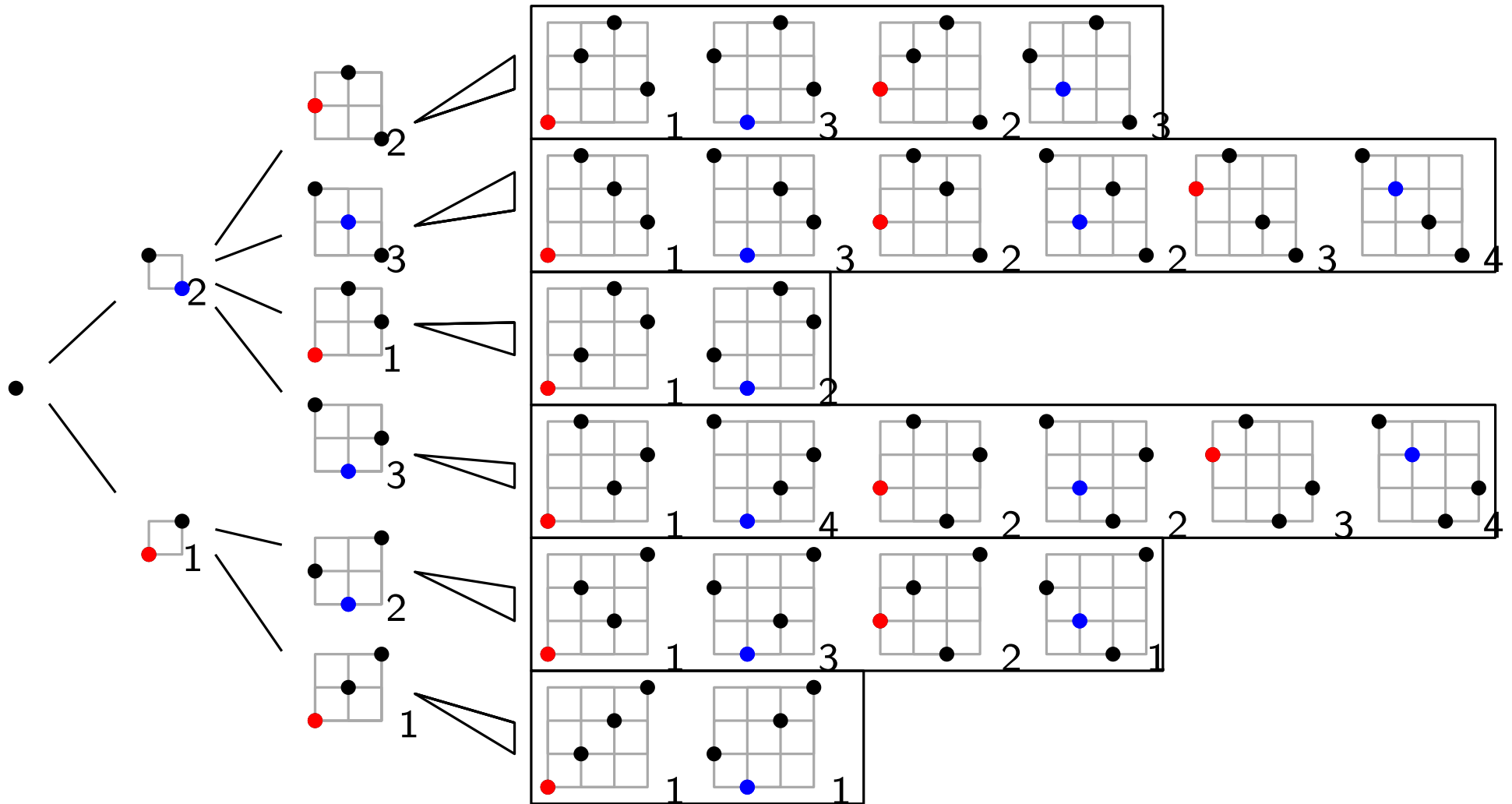
A generating tree for square permutations

Corollary. The mapping θ produces a generating tree for square permutations.



A generating tree for square permutations

Corollary. The mapping θ produces a generating tree for square permutations.



Each node with label k has $2k$ children.

But to count we need a more precise description of the shape of the tree

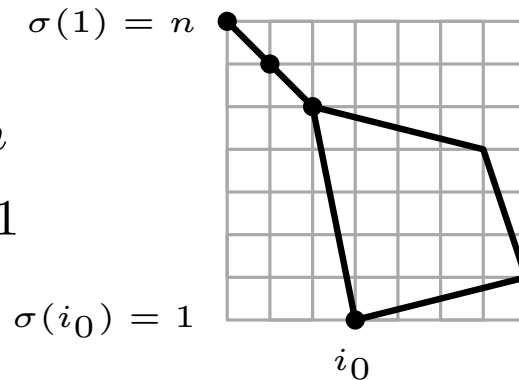
A growth rule

We classify permutations according to the value $\sigma(1)$ and $\sigma(i_0)$.

Type A:

$$\sigma(1) = n$$

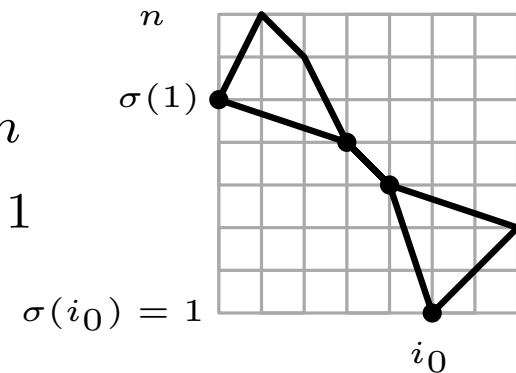
$$\sigma(i_0) = 1$$



Type B:

$$\sigma(1) \neq n$$

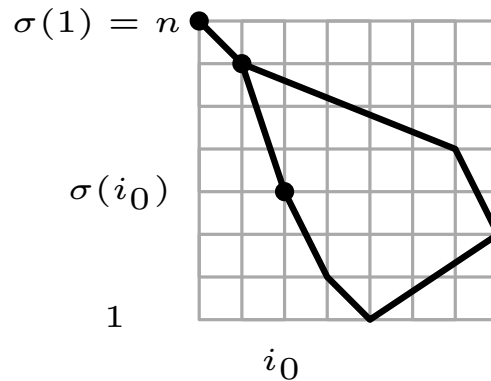
$$\sigma(i_0) = 1$$



Type C:

$$\sigma(1) = n$$

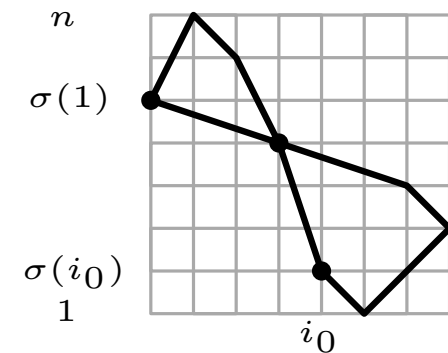
$$\sigma(i_0) \neq 1$$



Type D:

$$\sigma(1) \neq n$$

$$\sigma(i_0) \neq 1$$



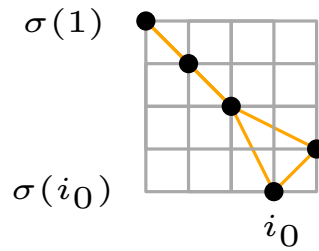
A growth rule

Describe the type of the children of a permutation with label k :

Type A :

$$\sigma(1) = n$$

$$\sigma(i_0) = 1$$



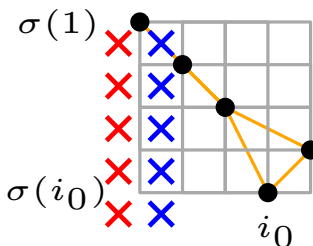
A growth rule

Describe the type of the children of a permutation with label k :

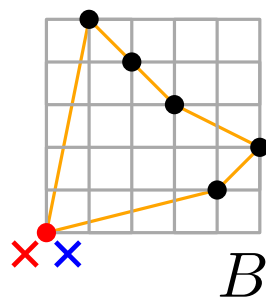
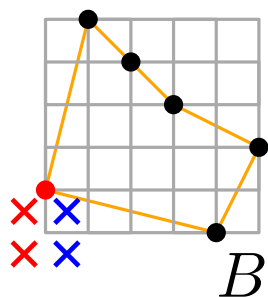
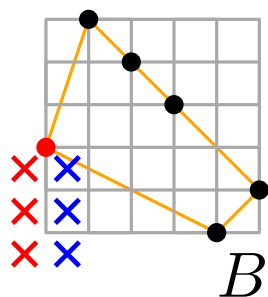
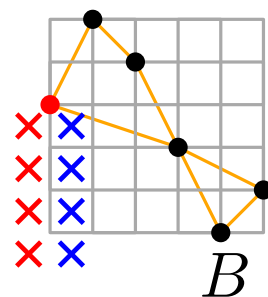
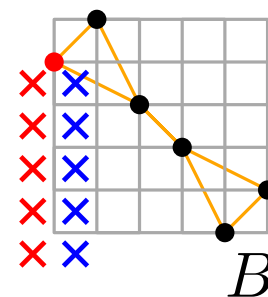
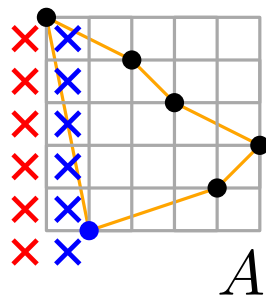
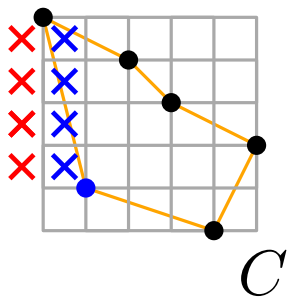
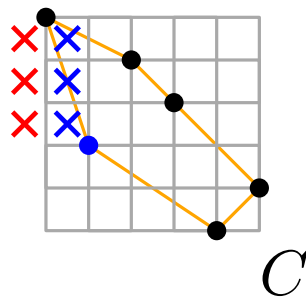
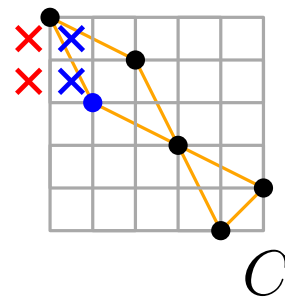
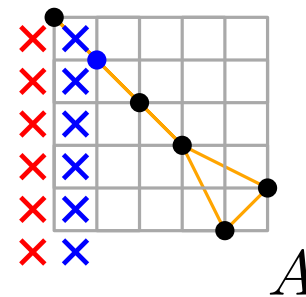
Type A:

$$\sigma(1) = n$$

$$\sigma(i_0) = 1$$



$$(k)_A \xrightarrow{\theta} (1)_B(2)_B \dots (k-1)_B(k)_B \\ (k+1)_A(k-1)_C \dots (2)_C(k+1)_A$$



The generating function

- The permutation $\bullet$ has two children with respective label $(2)_A$ and $(1)_B$

$$\begin{aligned}
 (k)_A &\xrightarrow{\theta} \frac{(1)_B(2)_B \dots (k-1)_B(k)_B}{(k+1)_A(k-1)_C \dots (2)_C(k+1)_A} & (k)_C &\xrightarrow{\theta} \frac{(1)_D(2)_D \dots (k-1)_D(k)_D}{(k)_C \dots (3)_C(2)_C(k+1)_C} \\
 (k)_B &\xrightarrow{\theta} \frac{(1)_B(2)_B \dots (k-1)_B(k)_B}{(k+1)_B(k-1)_D \dots (2)_D(1)_D} & (k)_D &\xrightarrow{\theta} \frac{(1)_D(2)_D \dots (k-1)_D(k)_D}{(k)_D(k-1)_D \dots (2)_D(1)_D}
 \end{aligned}$$

The resulting system:

$$F_A(u) = t^2 u^2 + 2tuF_A(u)$$

$$F_B(u) = t^2 u + t \frac{uF_A(1) - uF_A(u)}{1-u} + t \frac{uF_B(1) - u^2 F_B(u)}{1-u}$$

$$F_C(u) = t \frac{u^2 F_A(1) - F_A(u)}{1-u} + t \frac{u^2 F_C(1) - u^2 F_C(u)}{1-u}$$

$$F_D(u) = t \frac{uF_B(1) - F_B(u)}{1-u} + t \frac{uF_C(1) - uF_C(u)}{1-u} + 2t \frac{uF_D(1) - uF_D(u)}{1-u}$$

The generating function

- The permutation $\bullet$ has two children with respective label $(2)_A$ and $(1)_B$

$$\begin{aligned}
 (k)_A &\xrightarrow{\theta} \frac{(1)_B(2)_B \dots (k-1)_B(k)_B}{(k+1)_A(k-1)_C \dots (2)_C(k+1)_A} & (k)_C &\xrightarrow{\theta} \frac{(1)_D(2)_D \dots (k-1)_D(k)_D}{(k)_C \dots (3)_C(2)_C(k+1)_C} \\
 (k)_B &\xrightarrow{\theta} \frac{(1)_B(2)_B \dots (k-1)_B(k)_B}{(k+1)_B(k-1)_D \dots (2)_D(1)_D} & (k)_D &\xrightarrow{\theta} \frac{(1)_D(2)_D \dots (k-1)_D(k)_D}{(k)_D(k-1)_D \dots (2)_D(1)_D}
 \end{aligned}$$

The resulting system:

$$F_A(u) = t^2 u^2 + 2tuF_A(u)$$

$$F_B(u) = t^2 u + t \frac{uF_A(1) - uF_A(u)}{1-u} + t \frac{uF_B(1) - u^2 F_B(u)}{1-u}$$

$$F_C(u) = t \frac{u^2 F_A(1) - F_A(u)}{1-u} + t \frac{u^2 F_C(1) - u^2 F_C(u)}{1-u}$$

$$F_D(u) = t \frac{uF_B(1) - F_B(u)}{1-u} + t \frac{uF_C(1) - uF_C(u)}{1-u} + 2t \frac{uF_D(1) - uF_D(u)}{1-u}$$

The number of square permutations

The generating series of square permutations of size at least 2 is

$$F_S = F_A(1) + F_B(1) + F_C(1) + F_D(1) = \frac{2t^2(1-3t)}{(1-4t)^2} - \frac{4t^3}{(1-4t)^{3/2}}$$

Extracting the coefficients with the binomial formula yields:

Theorem [Mansour/Severini 2007, this proof by D./Poulalhon 2008]

The number of square permutations of size $n \geq 2$ is

$$|\mathcal{S}_n| = (2n+4)4^{n-3} - 4(2n-5)\binom{2n-6}{n-3}$$

Another proof by Albert, Linton, Ruskuc, Vatter Waton, 2011

The number of square permutations

The generating series of square permutations of size at least 2 is

$$F_S = F_A(1) + F_B(1) + F_C(1) + F_D(1) = \frac{2t^2(1-3t)}{(1-4t)^2} - \frac{4t^3}{(1-4t)^{3/2}}$$

Extracting the coefficients with the binomial formula yields:

Theorem [Mansour/Severini 2007, this proof by D./Poulalhon 2008]

The number of square permutations of size $n \geq 2$ is

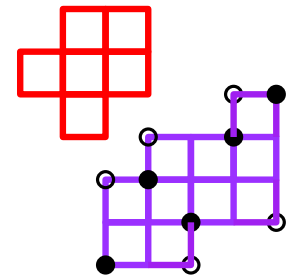
$$|\mathcal{S}_n| = (2n+4)4^{n-3} - 4(2n-5)\binom{2n-6}{n-3}$$

$$\mathcal{C}_n = \{ \text{convex polyominoes of size } n+1 \}$$

$$\mathcal{CT}_n = \{ \text{convex permutominoes of size } n \}$$

$$|\mathcal{C}_n| = (2n+5)4^{n-3} - 4(2n-5)\binom{2n-6}{n-3}$$

$$|\mathcal{CT}_n| = (2n+4)4^{n-3} - (2n-3)\binom{2n-4}{n-2}$$



[Delest Viennot, 84]

[Rinaldi et al., 2007]

The number of square permutations

The generating series of square permutations of size at least 2 is

$$F_S = F_A(1) + F_B(1) + F_C(1) + F_D(1) = \frac{2t^2(1-3t)}{(1-4t)^2} - \frac{4t^3}{(1-4t)^{3/2}}$$

Extracting the coefficients with the binomial formula yields:

Theorem [Mansour/Severini 2007, this proof by D./Poulalhon 2008]

The number of square permutations of size $n \geq 2$ is

$$|\mathcal{S}_n| = (2n+4)4^{n-3} - 4(2n-5)\binom{2n-6}{n-3}$$

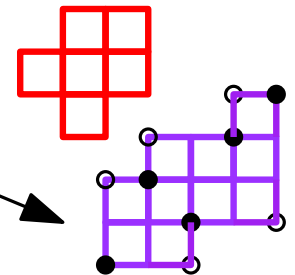
Bernini, Disanto, Pinzani, Rinaldi, 2007

$\mathcal{C}_n = \{ \text{convex polyominoes of size } n+1 \}$

$\mathcal{CT}_n = \{ \text{convex permutominoes of size } n \}$

$$|\mathcal{C}_n| = (2n+5)4^{n-3} - 4(2n-5)\binom{2n-6}{n-3}$$

$$|\mathcal{CT}_n| = (2n+4)4^{n-3} - (2n-3)\binom{2n-4}{n-2}$$



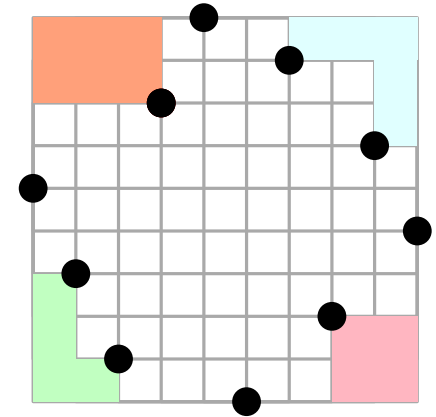
[Delest Viennot, 84]

[Rinaldi et al., 2007]

Subfamilies of square permutations

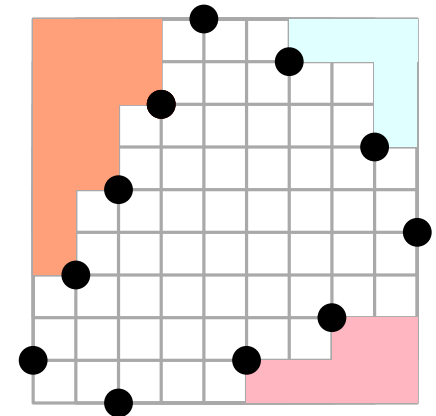
Square permutation

- = permutation without internal point
- = all points are UL, UR, DL or DR.
- = permutations avoiding 16 patterns of length 5



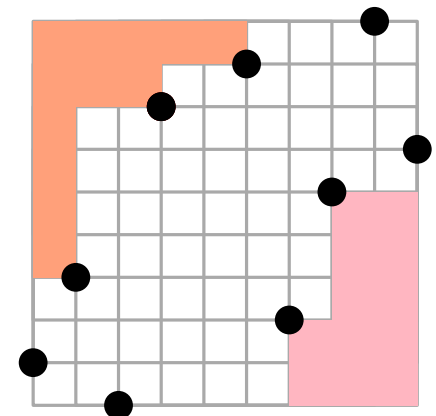
Triangular permutation

- = permutation without internal or DL points
- = all points are UL, UR and DR
- = permutations avoiding 3214 3241 4213 4231



Parallel permutation

- = permutation without internal, DL or UR points
- = all points are UL or DR
- = 321-avoiding permutations



Enumerative results

$\mathcal{S}_n = \{ \text{square permutations of size } n \}$

$$|\mathcal{S}_n| = (2n + 4) 4^{n-3} - 4(2n - 5) \binom{2n-6}{n-3}$$

$\mathcal{T}_n^{\nearrow} = \{ \text{triangular permutations of size } n \}$

$$|\mathcal{D}_n| = \binom{2n-2}{n-1} \quad |\mathcal{T}_n^{\nearrow}| = \binom{2n-2}{n-1}$$

$\mathcal{P}_n = \{ \text{parallel permutations of size } n \}$

$$|\mathcal{PP}_n| = \frac{1}{n+1} \binom{2n}{n} \quad |\mathcal{P}_n| = \frac{1}{n+1} \binom{2n}{n}$$

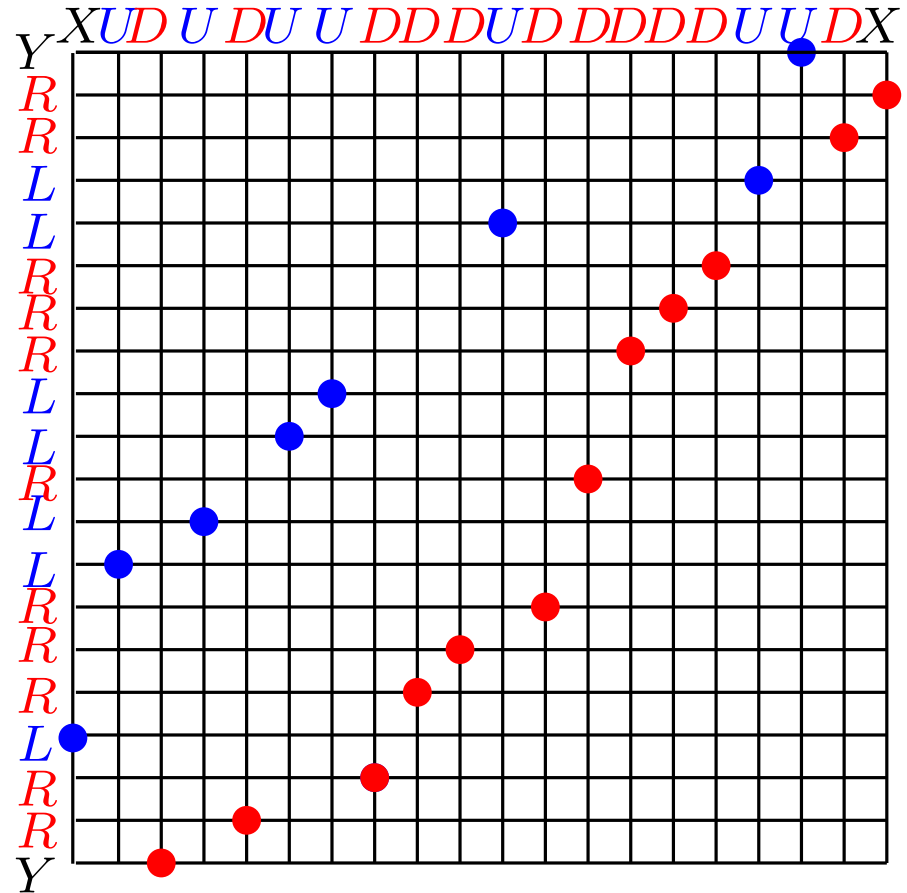
Interpretation of the difference in the formula

A code for square permutations

The idea is to encode square permutations by two words

- $u_1 \dots u_n$ (horizontal code)
- and $v_1 \dots v_n$ (vertical code)

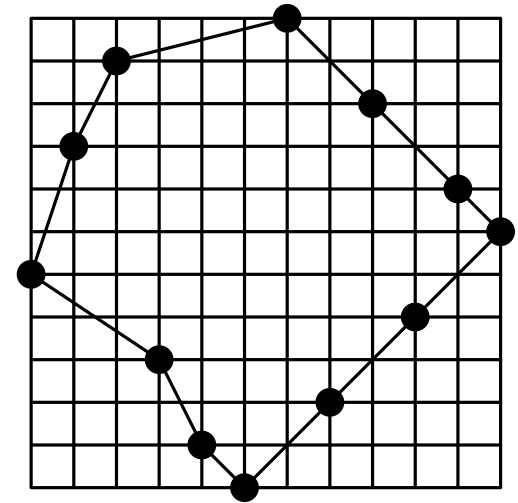
of the same length



Two words do not necessarily encode a square permutation

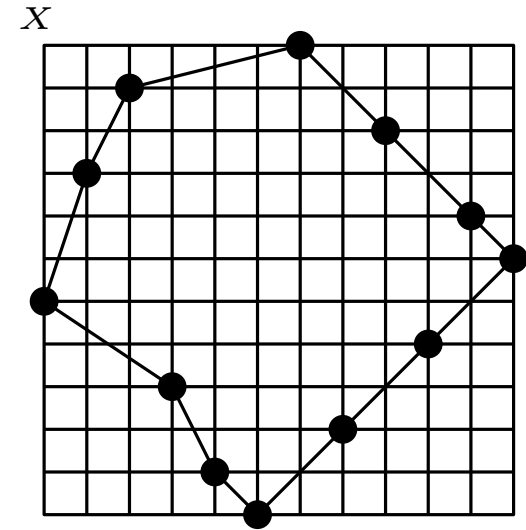
A code for square permutations

- the horizontal word $u_1 \dots u_n$



A code for square permutations

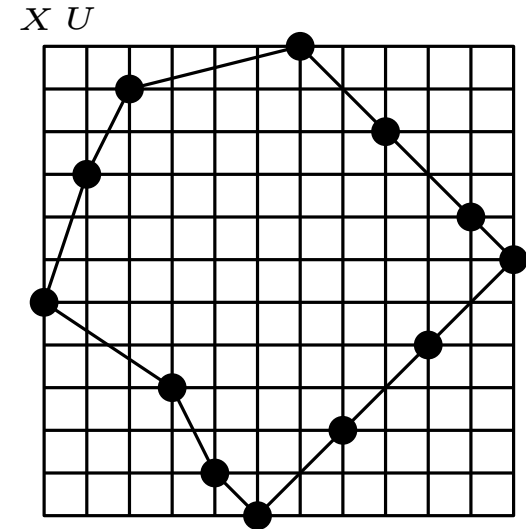
- the horizontal word $u_1 \dots u_n$



$u_i = X$ for extremal points in the vertical borders of the bounding box.

A code for square permutations

- the horizontal word $u_1 \dots u_n$

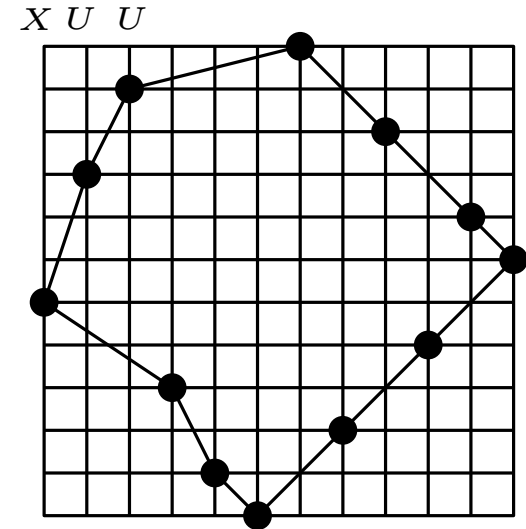


$u_i = X$ for extremal points in the vertical borders of the bounding box.

$$u_i = \begin{cases} U, & \text{if } (i, \sigma(i)) \text{ is an upper point} \\ D, & \text{otherwise} \end{cases}$$

A code for square permutations

- the horizontal word $u_1 \dots u_n$

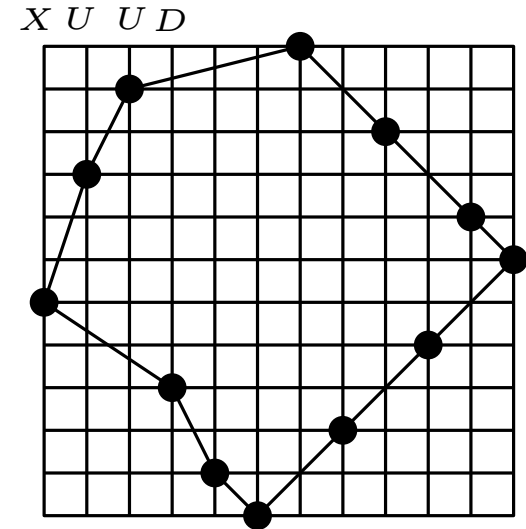


$u_i = X$ for extremal points in the vertical borders of the bounding box.

$$u_i = \begin{cases} U, & \text{if } (i, \sigma(i)) \text{ is an upper point} \\ D, & \text{otherwise} \end{cases}$$

A code for square permutations

- the horizontal word $u_1 \dots u_n$

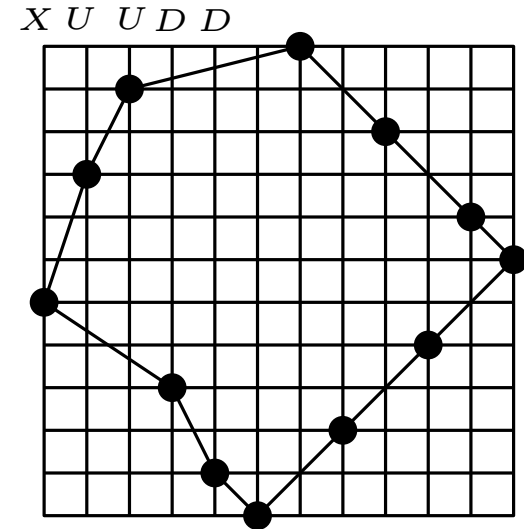


$u_i = X$ for extremal points in the vertical borders of the bounding box.

$$u_i = \begin{cases} U, & \text{if } (i, \sigma(i)) \text{ is an upper point} \\ D, & \text{otherwise} \end{cases}$$

A code for square permutations

- the horizontal word $u_1 \dots u_n$

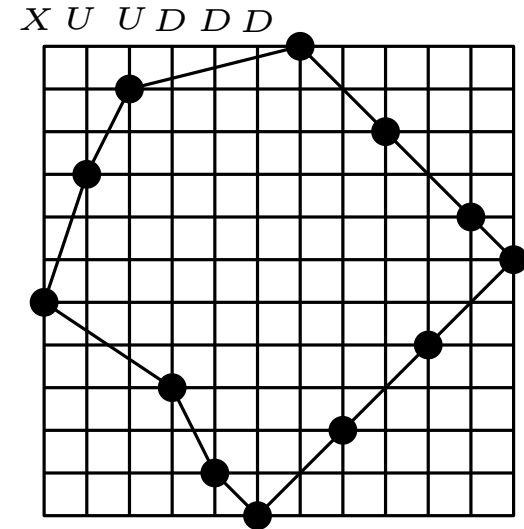


$u_i = X$ for extremal points in the vertical borders of the bounding box.

$$u_i = \begin{cases} U, & \text{if } (i, \sigma(i)) \text{ is an upper point} \\ D, & \text{otherwise} \end{cases}$$

A code for square permutations

- the horizontal word $u_1 \dots u_n$

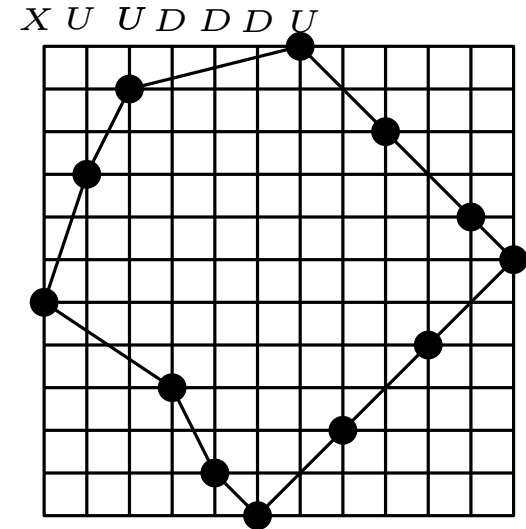


$u_i = X$ for extremal points in the vertical borders of the bounding box.

$$u_i = \begin{cases} U, & \text{if } (i, \sigma(i)) \text{ is an upper point} \\ D, & \text{otherwise} \end{cases}$$

A code for square permutations

- the horizontal word $u_1 \dots u_n$

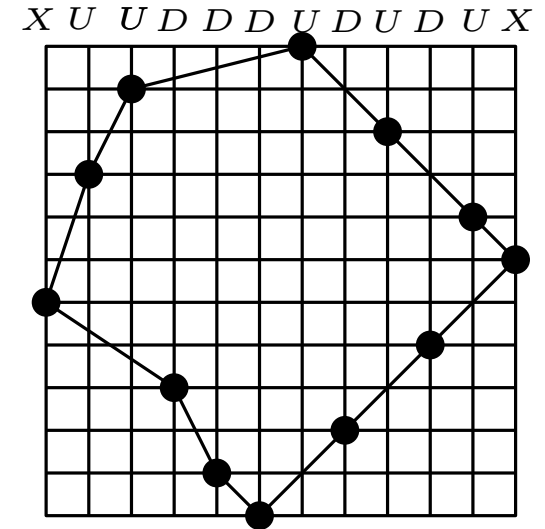


$u_i = X$ for extremal points in the vertical borders of the bounding box.

$$u_i = \begin{cases} U, & \text{if } (i, \sigma(i)) \text{ is an upper point} \\ D, & \text{otherwise} \end{cases}$$

A code for square permutations

- the horizontal word $u_1 \dots u_n$
- the vertical word $v_1 \dots v_n$

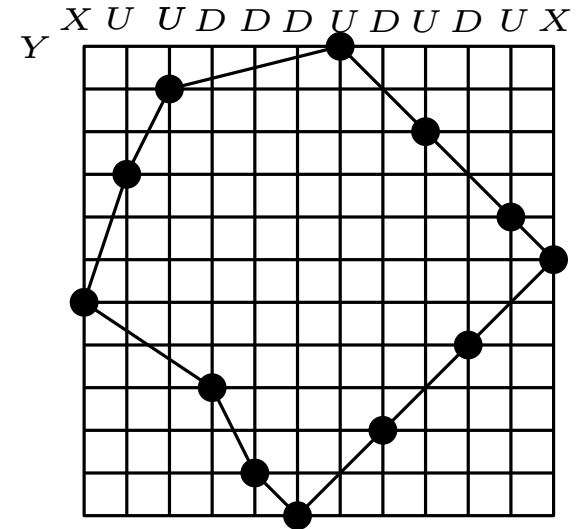


$u_i = X$ for extremal points in the vertical borders of the bounding box.

$$u_i = \begin{cases} U, & \text{if } (i, \sigma(i)) \text{ is an upper point} \\ D, & \text{otherwise} \end{cases}$$

A code for square permutations

- the horizontal word $u_1 \dots u_n$
- the vertical word $v_1 \dots v_n$



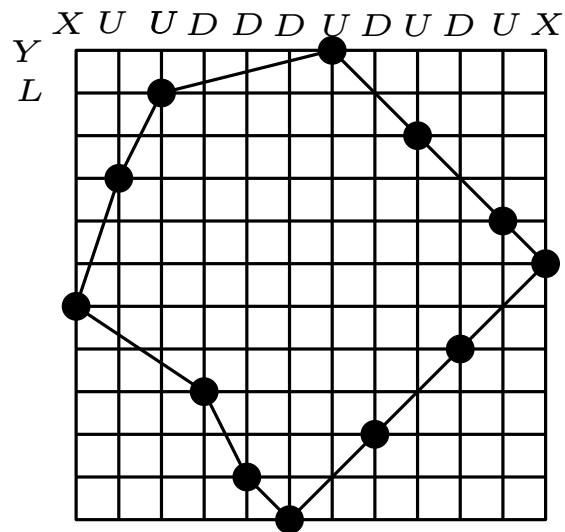
$u_i = X$ for extremal points in the vertical borders of the bounding box.

$$u_i = \begin{cases} U, & \text{if } (i, \sigma(i)) \text{ is an upper point} \\ D, & \text{otherwise} \end{cases}$$

$v_i = Y$ for extremal points in the horizontal borders of the bounding box.

A code for square permutations

- the horizontal word $u_1 \dots u_n$
- the vertical word $v_1 \dots v_n$



$u_i = X$ for extremal points in the vertical borders of the bounding box.

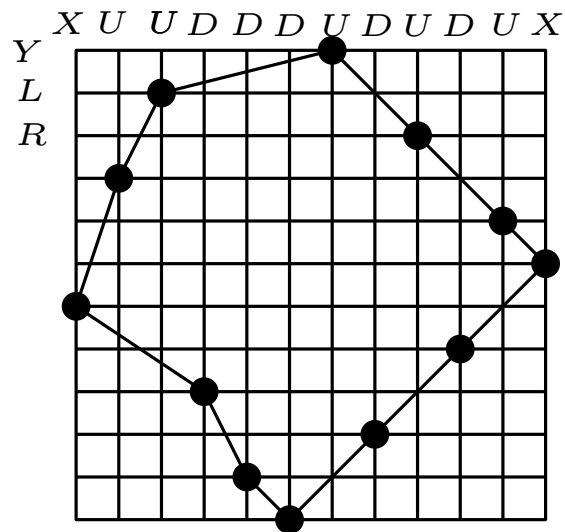
$$u_i = \begin{cases} U, & \text{if } (i, \sigma(i)) \text{ is an upper point} \\ D, & \text{otherwise} \end{cases}$$

$v_i = Y$ for extremal points in the horizontal borders of the bounding box.

$$v_i = \begin{cases} L, & \text{if } (i, \sigma(i)) \text{ is left point which is not an upper right one} \\ R, & \text{otherwise} \end{cases}$$

A code for square permutations

- the horizontal word $u_1 \dots u_n$
- the vertical word $v_1 \dots v_n$



$u_i = X$ for extremal points in the vertical borders of the bounding box.

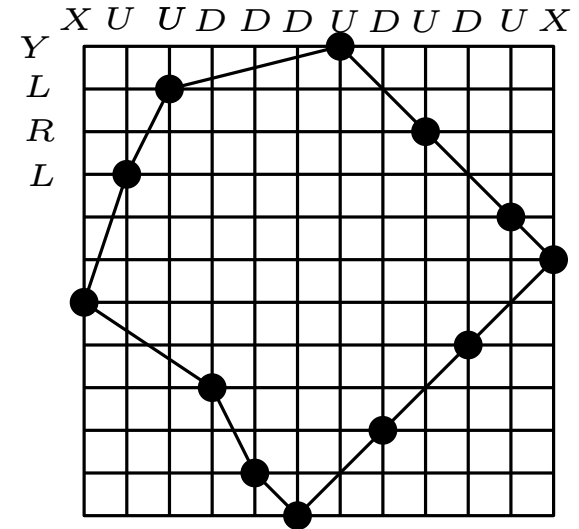
$$u_i = \begin{cases} U, & \text{if } (i, \sigma(i)) \text{ is an upper point} \\ D, & \text{otherwise} \end{cases}$$

$v_i = Y$ for extremal points in the horizontal borders of the bounding box.

$$v_i = \begin{cases} L, & \text{if } (i, \sigma(i)) \text{ is left point which is not an upper right one} \\ R, & \text{otherwise} \end{cases}$$

A code for square permutations

- the horizontal word $u_1 \dots u_n$
- the vertical word $v_1 \dots v_n$



$u_i = X$ for extremal points in the vertical borders of the bounding box.

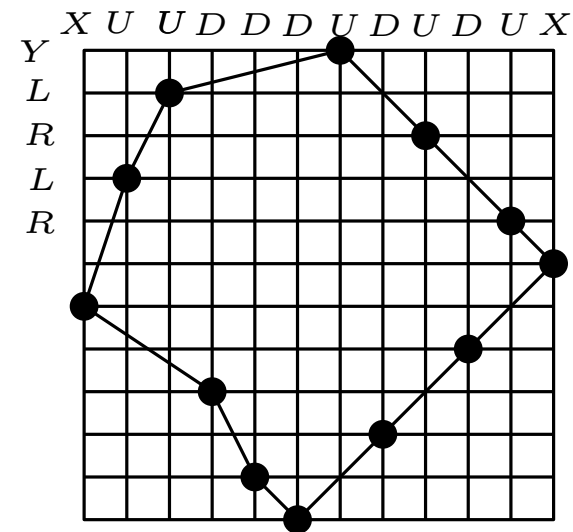
$$u_i = \begin{cases} U, & \text{if } (i, \sigma(i)) \text{ is an upper point} \\ D, & \text{otherwise} \end{cases}$$

$v_i = Y$ for extremal points in the horizontal borders of the bounding box.

$$v_i = \begin{cases} L, & \text{if } (i, \sigma(i)) \text{ is left point which is not an upper right one} \\ R, & \text{otherwise} \end{cases}$$

A code for square permutations

- the horizontal word $u_1 \dots u_n$
- the vertical word $v_1 \dots v_n$



$u_i = X$ for extremal points in the vertical borders of the bounding box.

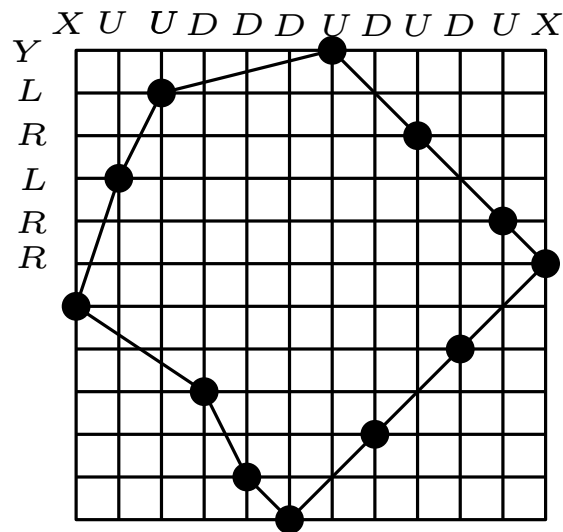
$$u_i = \begin{cases} U, & \text{if } (i, \sigma(i)) \text{ is an upper point} \\ D, & \text{otherwise} \end{cases}$$

$v_i = Y$ for extremal points in the horizontal borders of the bounding box.

$$v_i = \begin{cases} L, & \text{if } (i, \sigma(i)) \text{ is left point which is not an upper right one} \\ R, & \text{otherwise} \end{cases}$$

A code for square permutations

- the horizontal word $u_1 \dots u_n$
- the vertical word $v_1 \dots v_n$



$u_i = X$ for extremal points in the vertical borders of the bounding box.

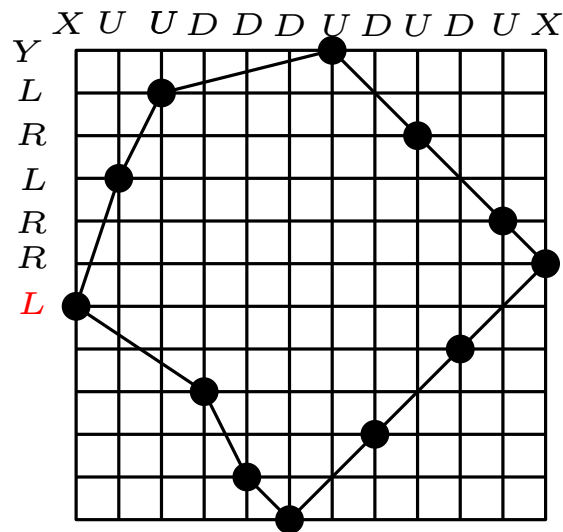
$$u_i = \begin{cases} U, & \text{if } (i, \sigma(i)) \text{ is an upper point} \\ D, & \text{otherwise} \end{cases}$$

$v_i = Y$ for extremal points in the horizontal borders of the bounding box.

$$v_i = \begin{cases} L, & \text{if } (i, \sigma(i)) \text{ is left point which is not an upper right one} \\ R, & \text{otherwise} \end{cases}$$

A code for square permutations

- the horizontal word $u_1 \dots u_n$
- the vertical word $v_1 \dots v_n$



$u_i = X$ for extremal points in the vertical borders of the bounding box.

$$u_i = \begin{cases} U, & \text{if } (i, \sigma(i)) \text{ is an upper point} \\ D, & \text{otherwise} \end{cases}$$

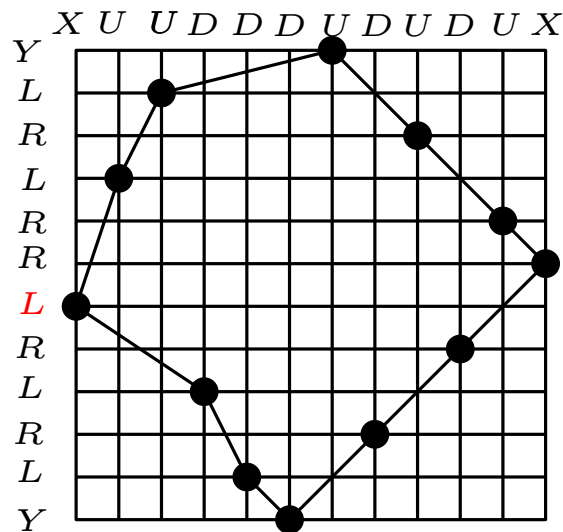
$v_i = Y$ for extremal points in the horizontal borders of the bounding box.

$$v_i = \begin{cases} L, & \text{if } (i, \sigma(i)) \text{ is left point which is not an upper right one} \\ R, & \text{otherwise} \end{cases}$$

A code for square permutations

- the horizontal word $u_1 \dots u_n$
- the vertical word $v_1 \dots v_n$

and by marking one letter L or Y .



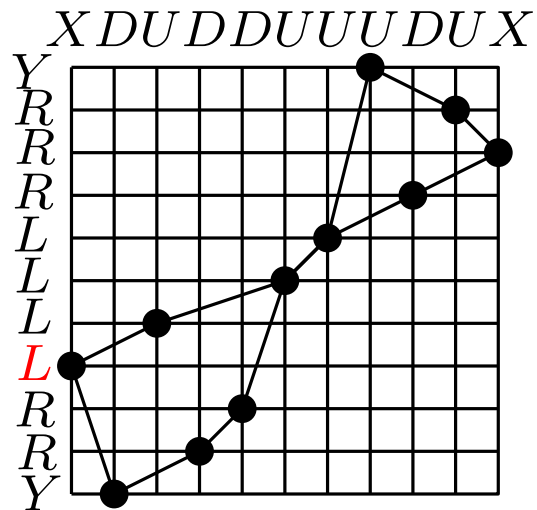
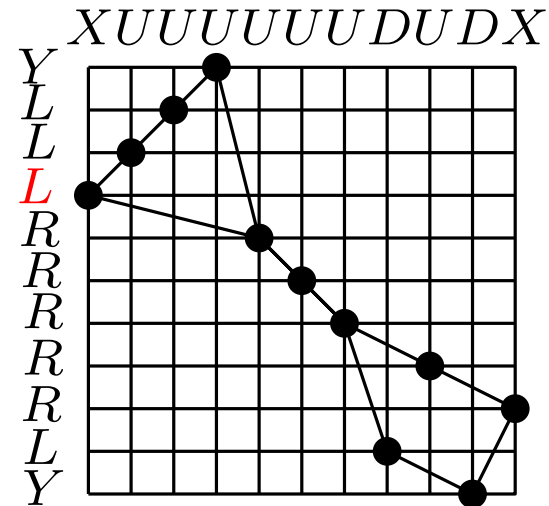
$u_i = X$ for extremal points in the vertical borders of the bounding box.

$$u_i = \begin{cases} U, & \text{if } (i, \sigma(i)) \text{ is an upper point} \\ D, & \text{otherwise} \end{cases}$$

$v_i = Y$ for extremal points in the horizontal borders of the bounding box.

$$v_i = \begin{cases} L, & \text{if } (i, \sigma(i)) \text{ is left point which is not an upper right one} \\ R, & \text{otherwise} \end{cases}$$

A code for square permutations



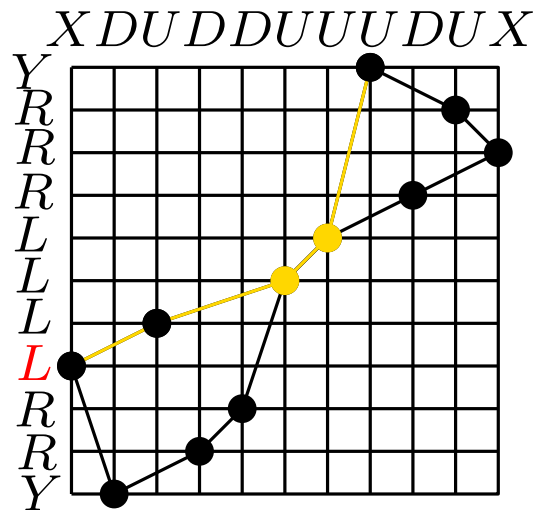
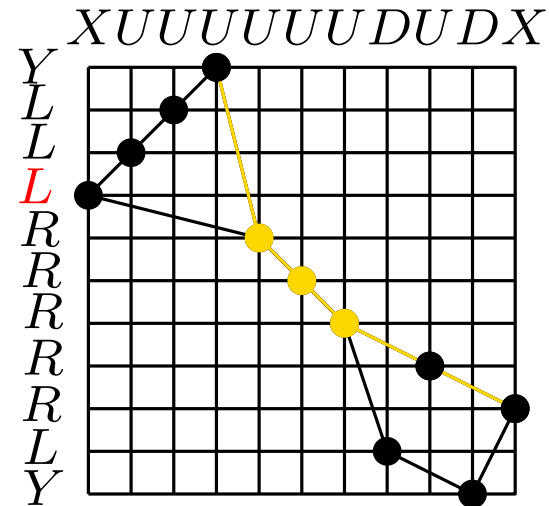
$u_i = X$ for extremal points in the vertical borders of the bounding box.

$$u_i = \begin{cases} U, & \text{if } (i, \sigma(i)) \text{ is an upper point} \\ D, & \text{otherwise} \end{cases}$$

$v_i = Y$ for extremal points in the horizontal borders of the bounding box.

$$v_i = \begin{cases} L, & \text{if } (i, \sigma(i)) \text{ is left point which is not an upper right one} \\ R, & \text{otherwise} \end{cases}$$

A code for square permutations



$u_i = X$ for extremal points in the vertical borders of the bounding box.

$$u_i = \begin{cases} U, & \text{if } (i, \sigma(i)) \text{ is an upper point} \\ D, & \text{otherwise} \end{cases}$$

$v_i = Y$ for extremal points in the horizontal borders of the bounding box.

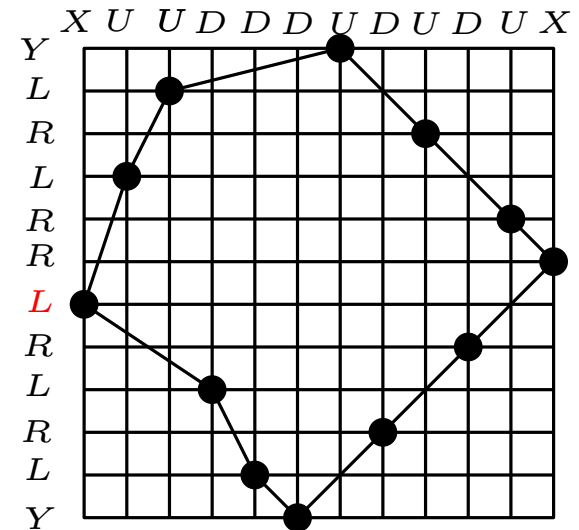
$$v_i = \begin{cases} L, & \text{if } (i, \sigma(i)) \text{ is left point which is not an upper right one} \\ R, & \text{otherwise} \end{cases}$$

A code for square permutations

Let $\mathcal{W} = \mathcal{A}^*$ the set of (bi)words on the alphabet $\mathcal{A} = \{U, D\} \times \{L, R\}$

Then our codes can be viewed as elements of the set $\mathcal{M}$ of marked biwords (w, m) of the form:

- $w = (X, Y) \cdot w' \cdot (X, Y)$ with $w' \in \mathcal{W}$
- $1 \leq m \leq n$ and $v_m = (L, Y)$



The set of marked words $\mathcal{M}$ gives an interpretation for rational part of the formula. Indeed

$$|\mathcal{M}_n| = (n - 2) \cdot 2^{n-3} \cdot 2^{n-2} + 2 \cdot 2^{n-2} \cdot 2^{n-2} = (n + 2) \cdot 2^{2n-5}$$

$$M(t) = \frac{t^2}{1-4t} \left(2 + \frac{2t}{1-4t} \right)$$

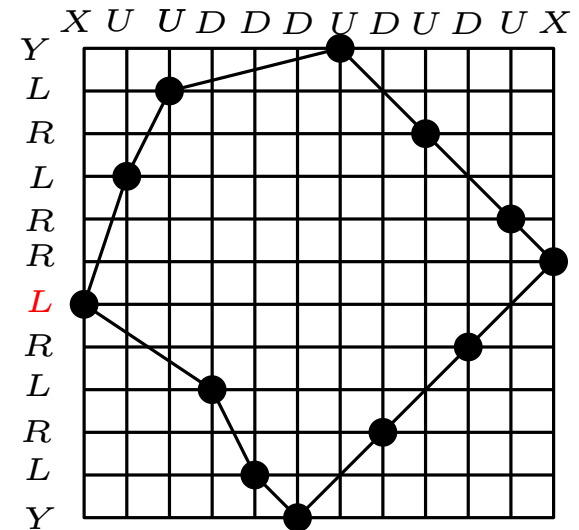
Observe for later purpose that one can easily generate an element of $|\mathcal{M}_n|$ uniformly at random using a random integer in $\{1, \dots, n + 2\}$ and $2n - 5$ random bits.

A code for square permutations

Let $\mathcal{W} = \mathcal{A}^*$ the set of (bi)words on the alphabet $\mathcal{A} = \{U, D\} \times \{L, R\}$

Then our codes can be viewed as elements of the set $\mathcal{M}$ of marked biwords (w, m) of the form:

- $w = (X, Y) \cdot w' \cdot (X, Y)$ with $w' \in \mathcal{W}$
- $1 \leq m \leq n$ and $v_m = (L, Y)$



The set of marked words $\mathcal{M}$ gives an interpretation for rational part of the formula. Indeed

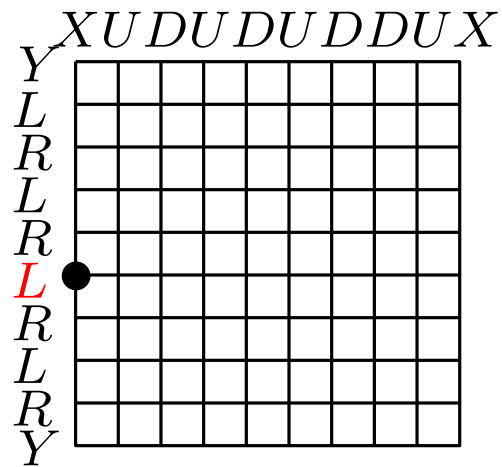
$$|\mathcal{M}_n| = (n-2) \cdot 2^{n-3} \cdot 2^{n-2} + 2 \cdot 2^{n-2} \cdot 2^{n-2} = (n+2) \cdot 2^{2n-5}$$

$$M(t) = \frac{t^2}{1-4t} \left(2 + \frac{2t}{1-4t} \right)$$

Observe for later purpose that one can easily generate an element of $|\mathcal{M}_n|$ uniformly at random using a random integer in $\{1, \dots, n+2\}$ and $2n-5$ random bits.

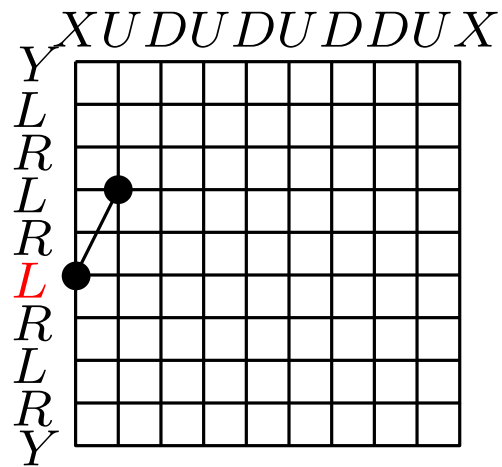
Decoding a word of $\mathcal{M}$

Consider a marked bi-word $(w, m) \in \mathcal{M}$



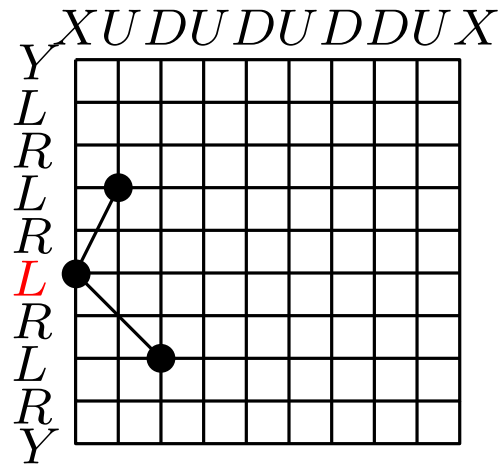
Decoding a word of $\mathcal{M}$

Consider a marked bi-word $(w, m) \in \mathcal{M}$



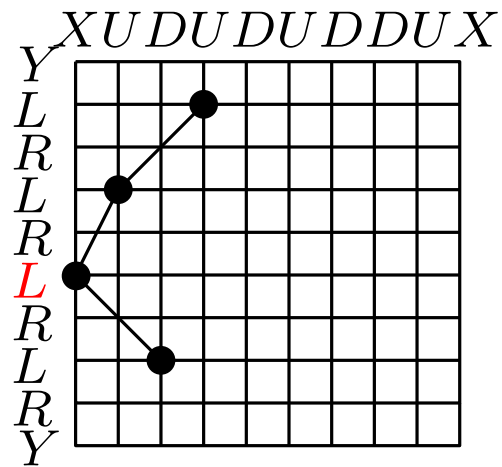
Decoding a word of $\mathcal{M}$

Consider a marked bi-word $(w, m) \in \mathcal{M}$



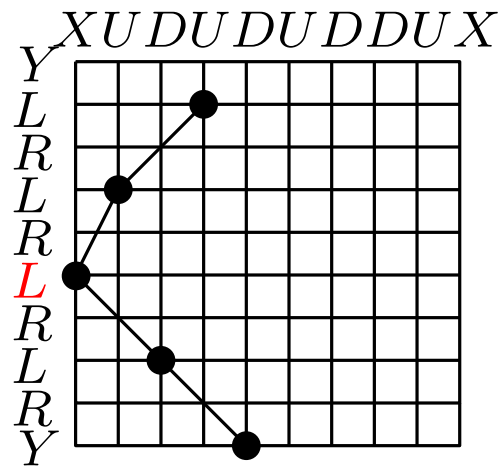
Decoding a word of $\mathcal{M}$

Consider a marked bi-word $(w, m) \in \mathcal{M}$



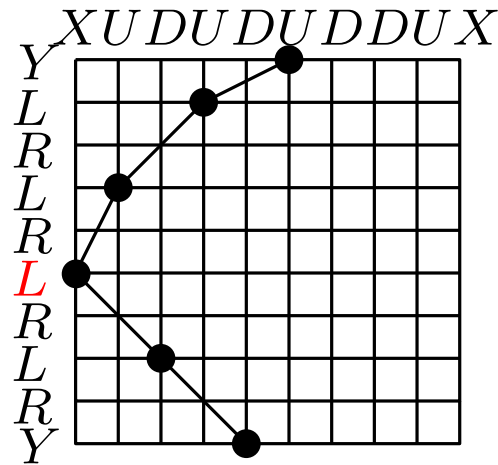
Decoding a word of $\mathcal{M}$

Consider a marked bi-word $(w, m) \in \mathcal{M}$



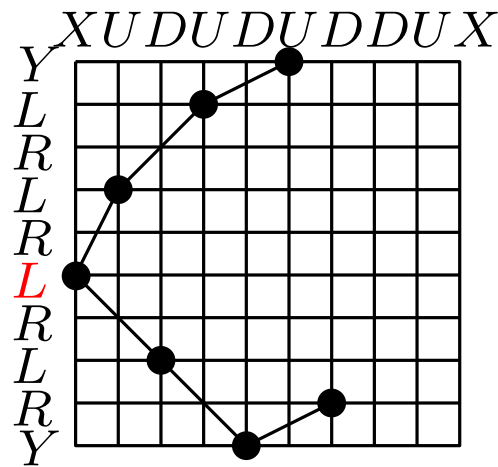
Decoding a word of $\mathcal{M}$

Consider a marked bi-word $(w, m) \in \mathcal{M}$



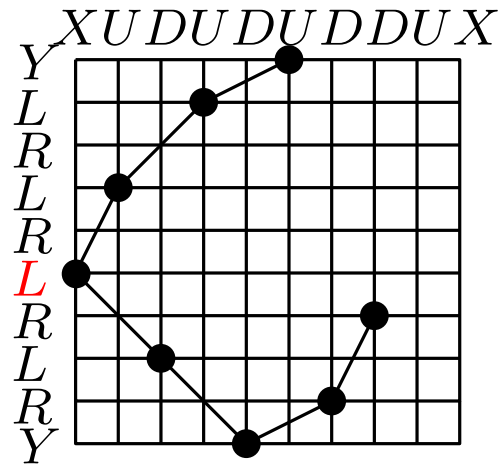
Decoding a word of $\mathcal{M}$

Consider a marked bi-word $(w, m) \in \mathcal{M}$



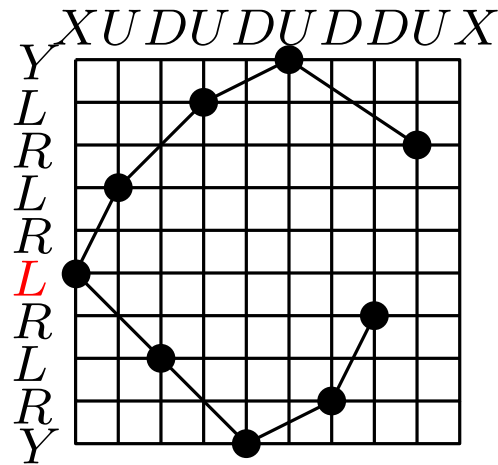
Decoding a word of $\mathcal{M}$

Consider a marked bi-word $(w, m) \in \mathcal{M}$



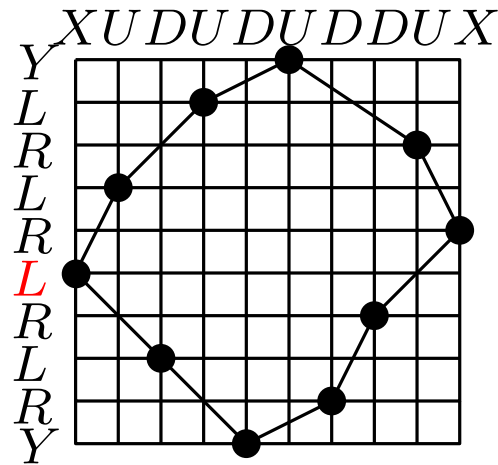
Decoding a word of $\mathcal{M}$

Consider a marked bi-word $(w, m) \in \mathcal{M}$



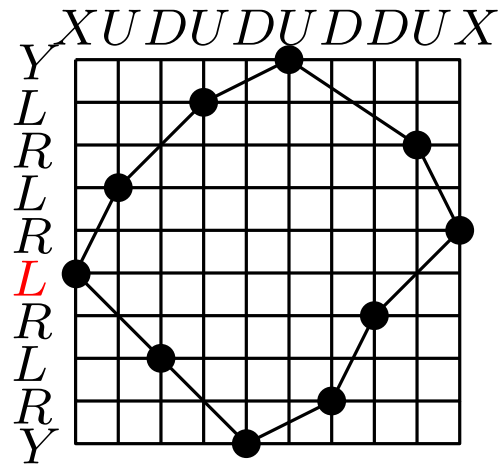
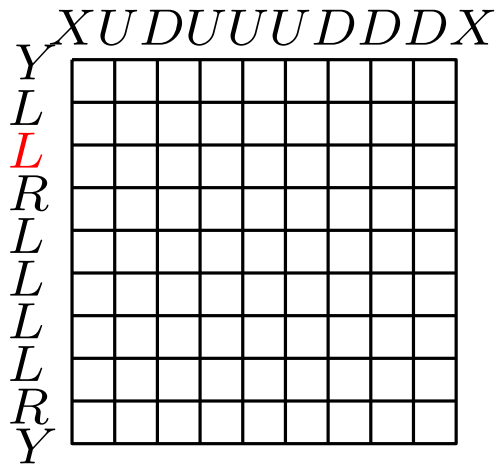
Decoding a word of $\mathcal{M}$

Consider a marked bi-word $(w, m) \in \mathcal{M}$



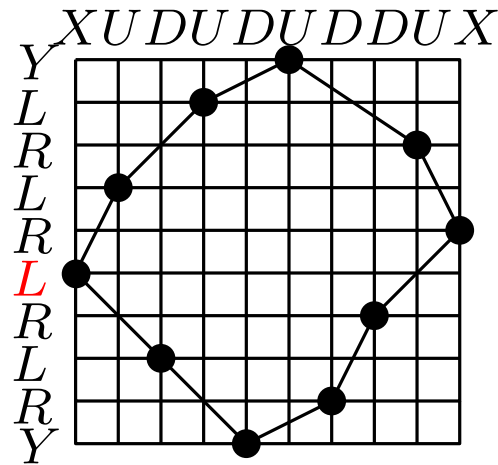
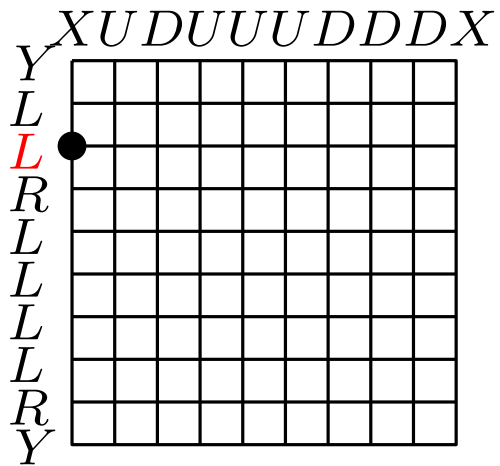
Decoding a word of $\mathcal{M}$

Consider a marked bi-word $(w, m) \in \mathcal{M}$



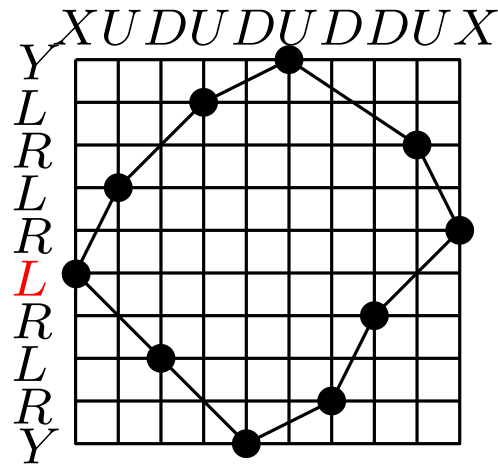
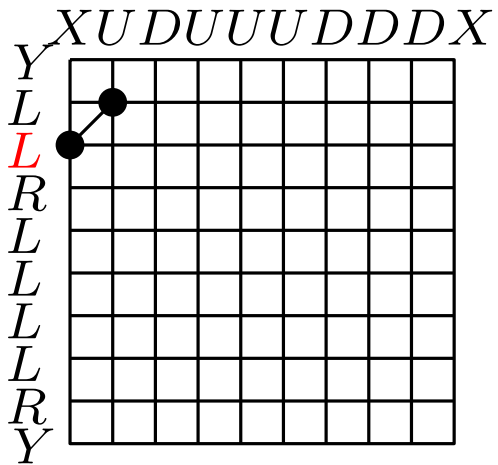
Decoding a word of $\mathcal{M}$

Consider a marked bi-word $(w, m) \in \mathcal{M}$



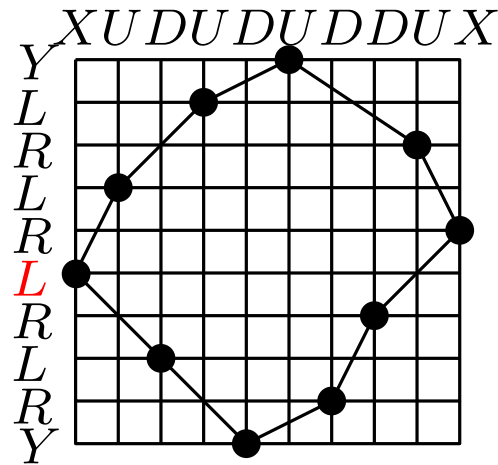
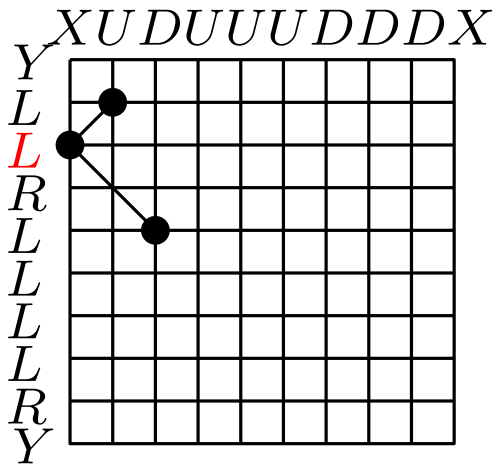
Decoding a word of $\mathcal{M}$

Consider a marked bi-word $(w, m) \in \mathcal{M}$



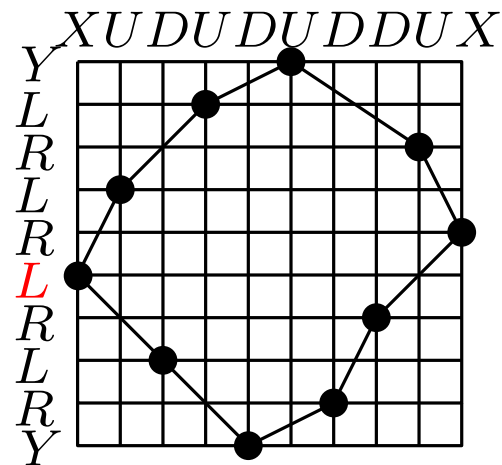
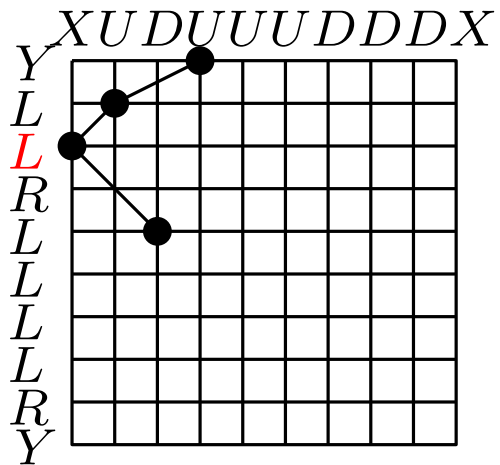
Decoding a word of $\mathcal{M}$

Consider a marked bi-word $(w, m) \in \mathcal{M}$



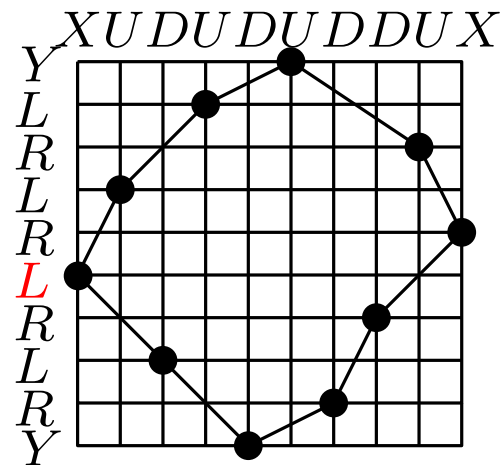
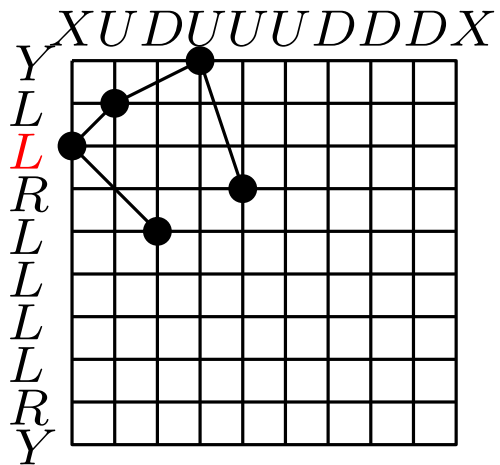
Decoding a word of $\mathcal{M}$

Consider a marked bi-word $(w, m) \in \mathcal{M}$



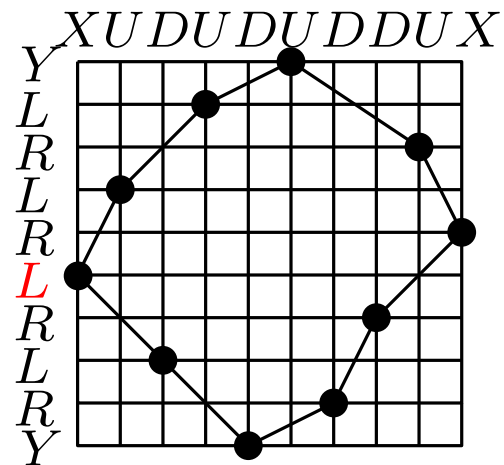
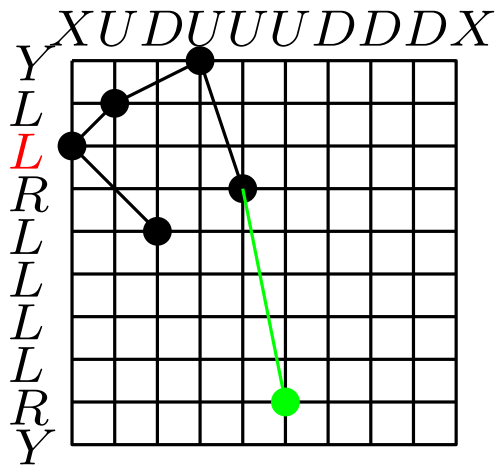
Decoding a word of $\mathcal{M}$

Consider a marked bi-word $(w, m) \in \mathcal{M}$



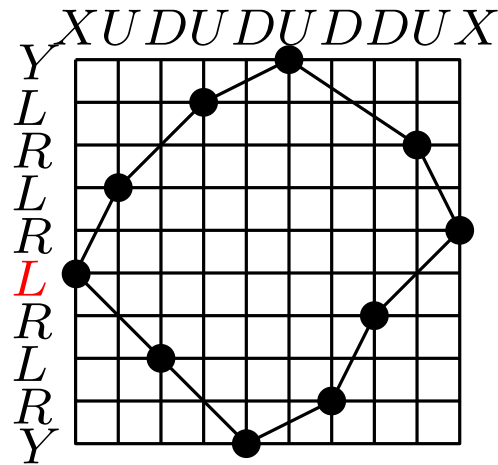
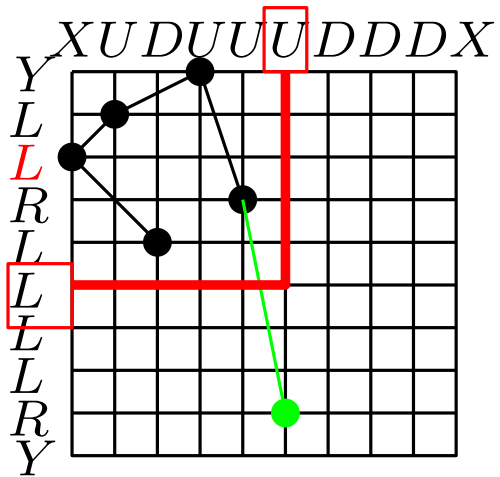
Decoding a word of $\mathcal{M}$

Consider a marked bi-word $(w, m) \in \mathcal{M}$



Decoding a word of $\mathcal{M}$

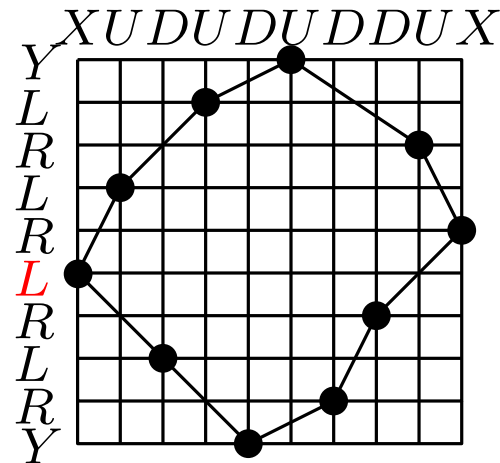
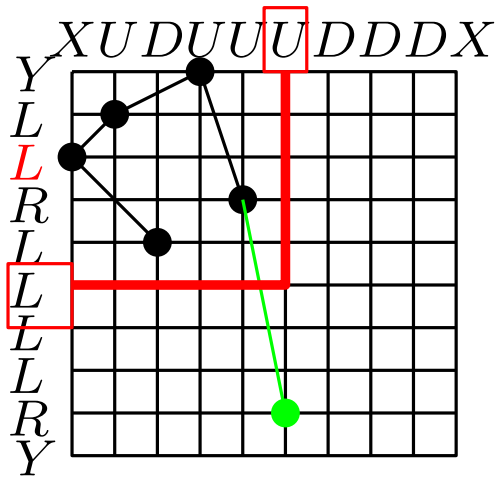
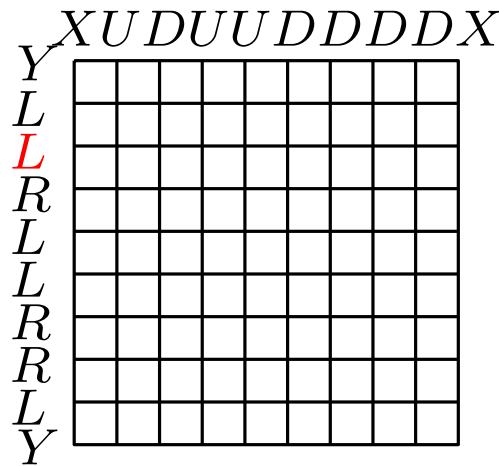
Consider a marked bi-word $(w, m) \in \mathcal{M}$



$$\mathcal{T}^{\nwarrow} \cdot (U, L) \cdot W \cdot (X, Y)$$

Decoding a word of $\mathcal{M}$

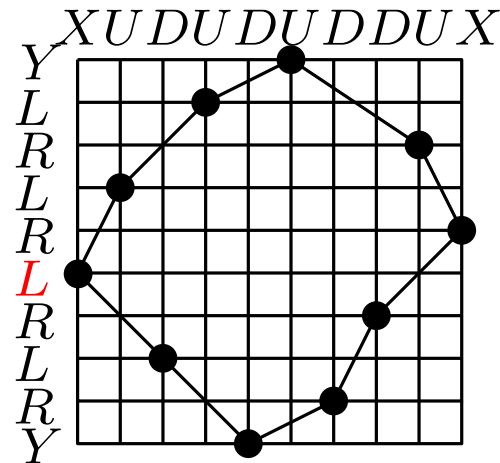
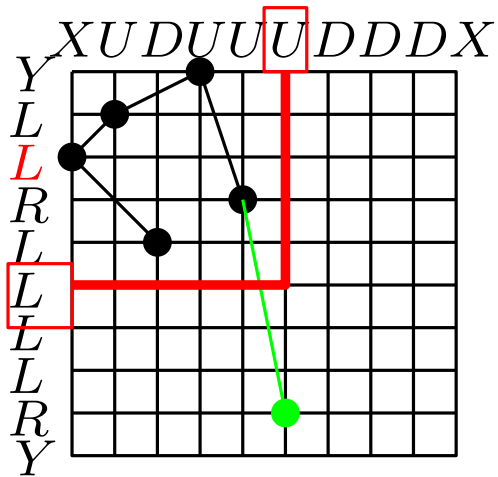
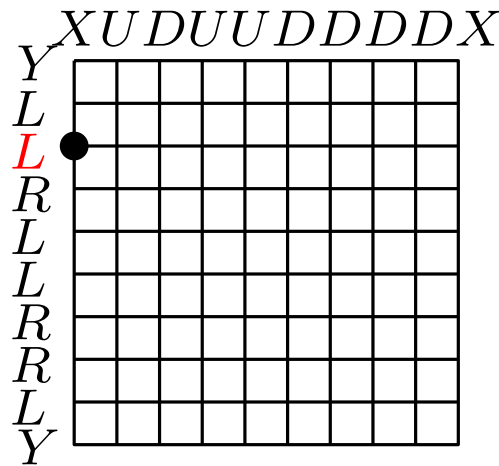
Consider a marked bi-word $(w, m) \in \mathcal{M}$



$$\mathcal{T}^{\nwarrow} \cdot (U, L) \cdot W \cdot (X, Y)$$

Decoding a word of $\mathcal{M}$

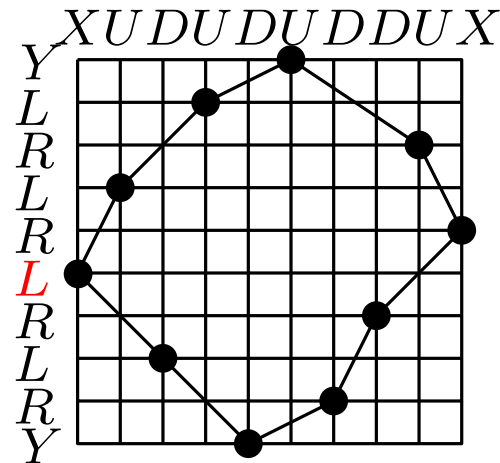
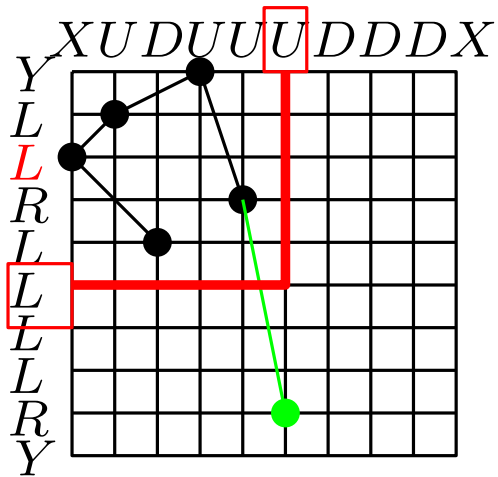
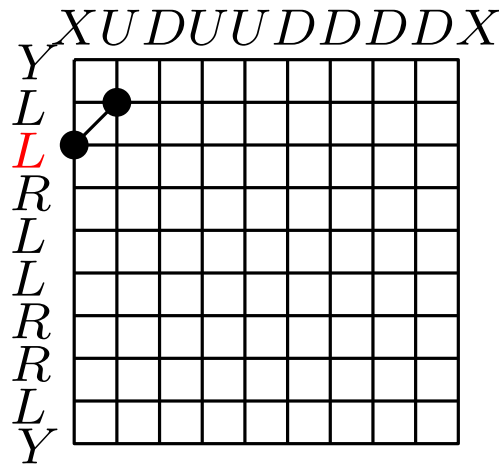
Consider a marked bi-word $(w, m) \in \mathcal{M}$



$$\mathcal{T}^{\nwarrow} \cdot (U, L) \cdot W \cdot (X, Y)$$

Decoding a word of $\mathcal{M}$

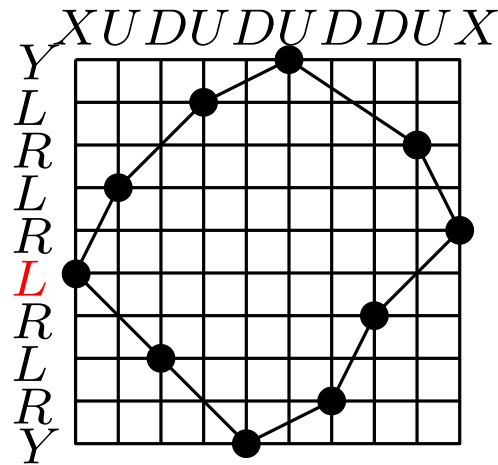
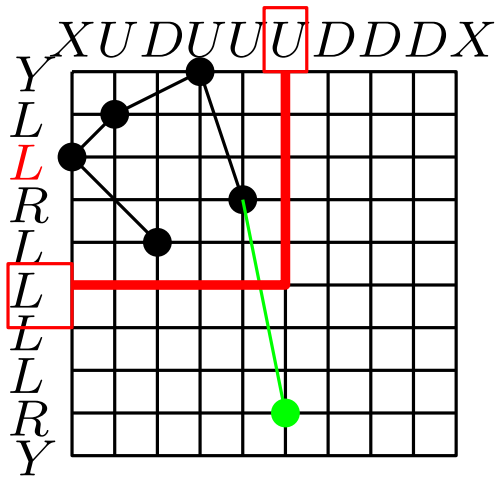
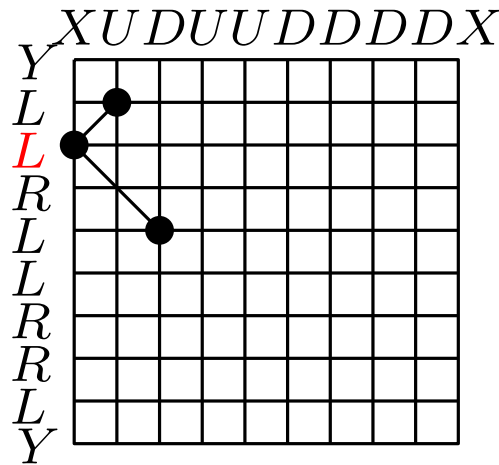
Consider a marked bi-word $(w, m) \in \mathcal{M}$



$$\mathcal{T}^{\nwarrow} \cdot (U, L) \cdot W \cdot (X, Y)$$

Decoding a word of $\mathcal{M}$

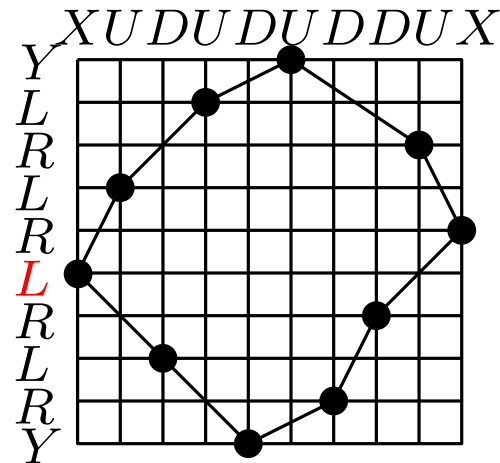
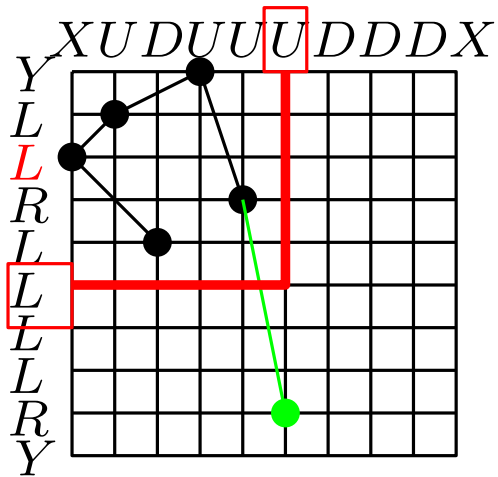
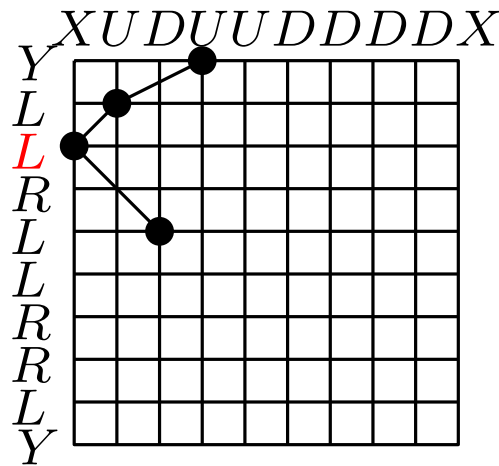
Consider a marked bi-word $(w, m) \in \mathcal{M}$



$$\mathcal{T}^{\nwarrow} \cdot (U, L) \cdot W \cdot (X, Y)$$

Decoding a word of $\mathcal{M}$

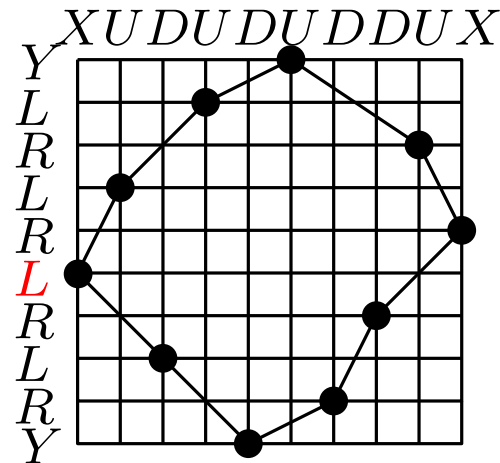
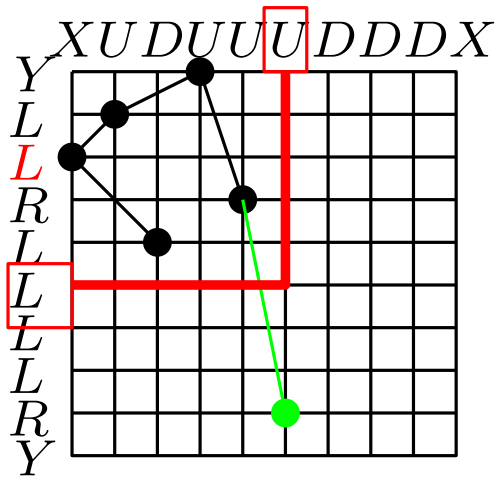
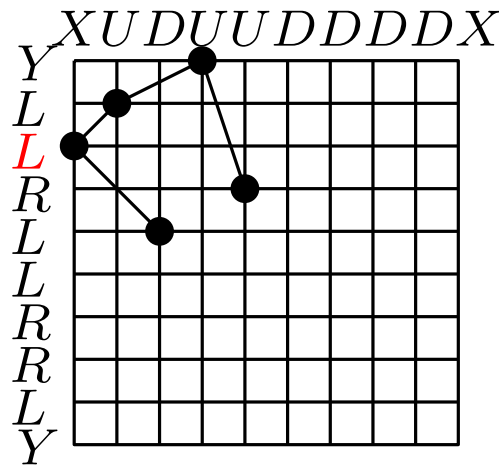
Consider a marked bi-word $(w, m) \in \mathcal{M}$



$$\mathcal{T}^{\nwarrow} \cdot (U, L) \cdot W \cdot (X, Y)$$

Decoding a word of $\mathcal{M}$

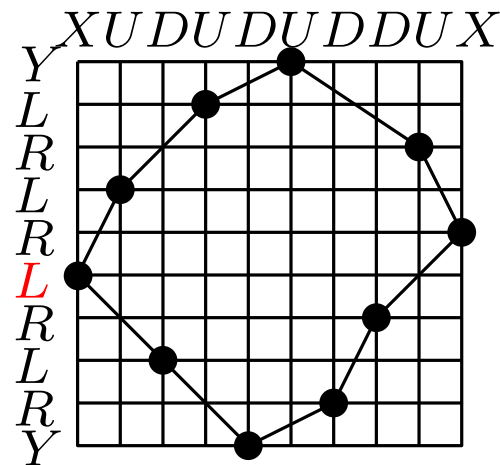
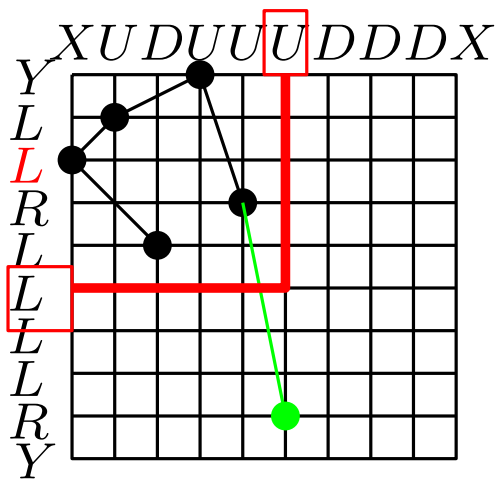
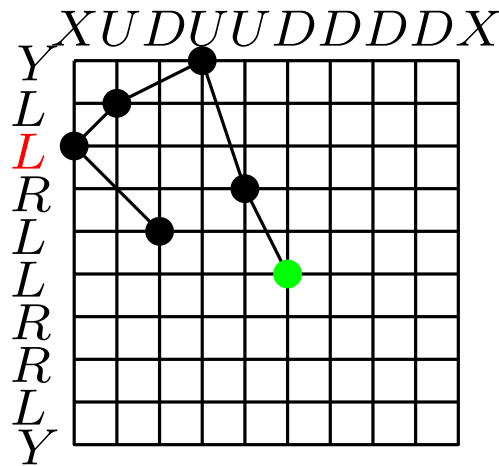
Consider a marked bi-word $(w, m) \in \mathcal{M}$



$$\mathcal{T}^{\nwarrow} \cdot (U, L) \cdot W \cdot (X, Y)$$

Decoding a word of $\mathcal{M}$

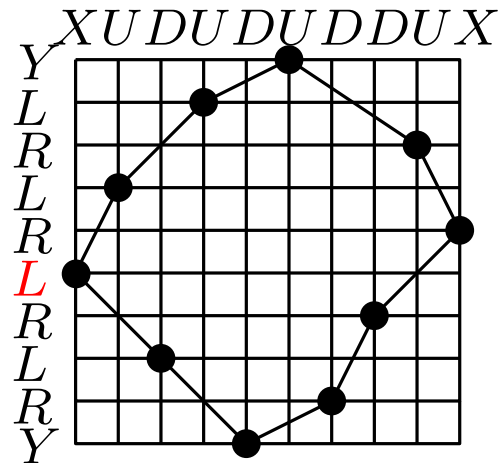
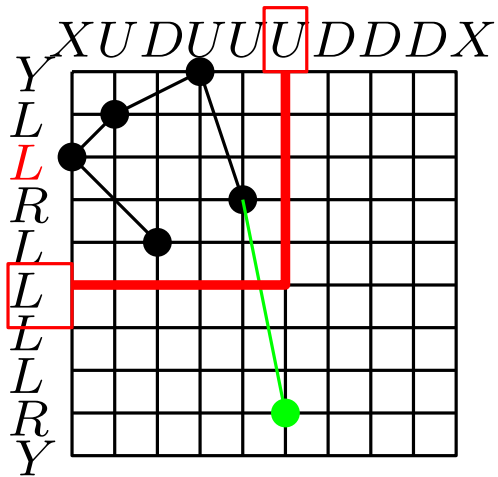
Consider a marked bi-word $(w, m) \in \mathcal{M}$



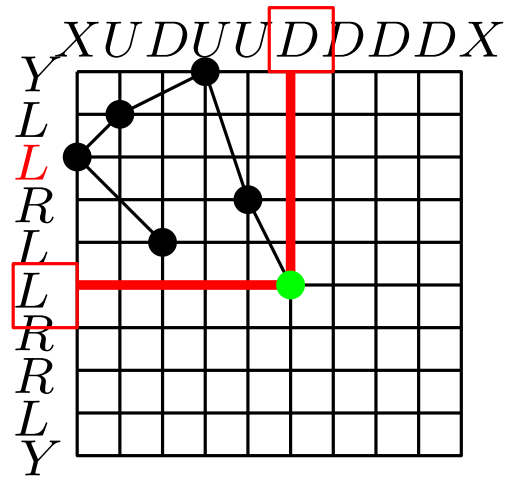
$$\mathcal{T}^{\nwarrow} \cdot (U, L) \cdot W \cdot (X, Y)$$

Decoding a word of $\mathcal{M}$

Consider a marked bi-word $(w, m) \in \mathcal{M}$



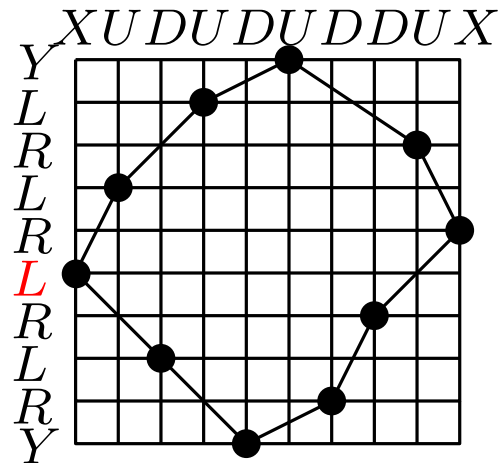
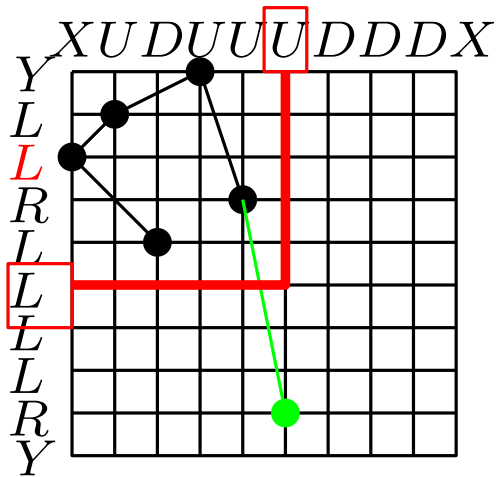
$$\mathcal{T}^{\nwarrow} \cdot (U, L) \cdot W \cdot (X, Y)$$



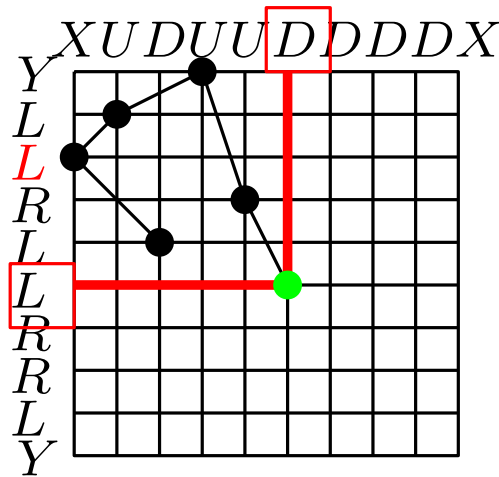
$$\mathcal{T}^{\nwarrow} \cdot (D, L) \cdot W \cdot (X, Y)$$

Decoding a word of $\mathcal{M}$

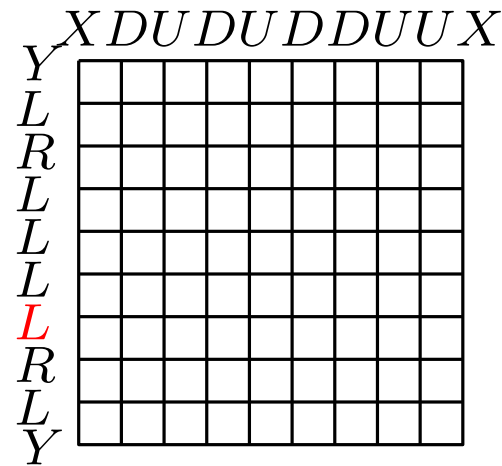
Consider a marked bi-word $(w, m) \in \mathcal{M}$



$$\mathcal{T}^{\nwarrow} \cdot (U, L) \cdot W \cdot (X, Y)$$

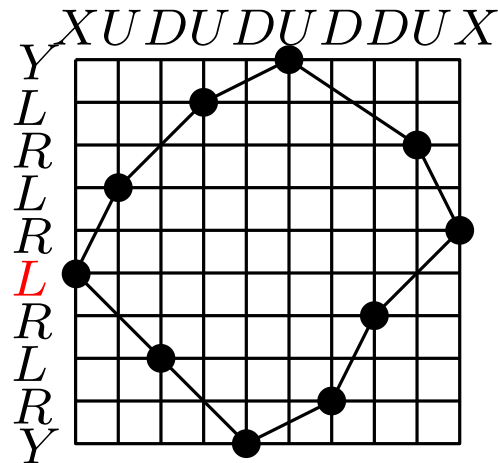
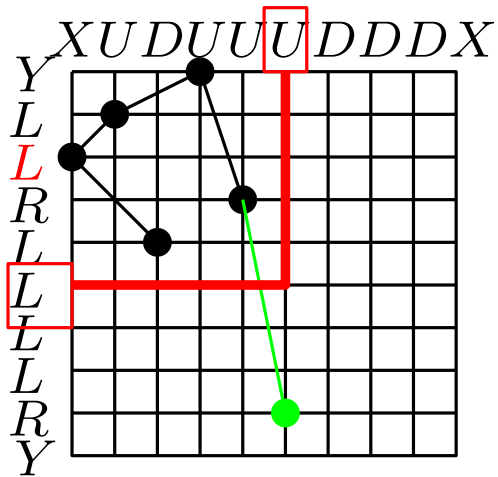


$$\mathcal{T}^{\nwarrow} \cdot (D, L) \cdot W \cdot (X, Y)$$

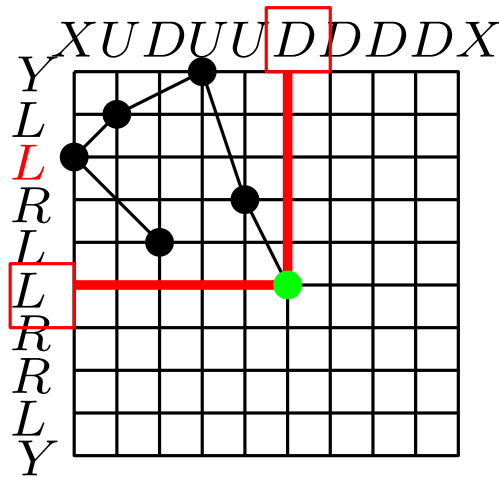


Decoding a word of $\mathcal{M}$

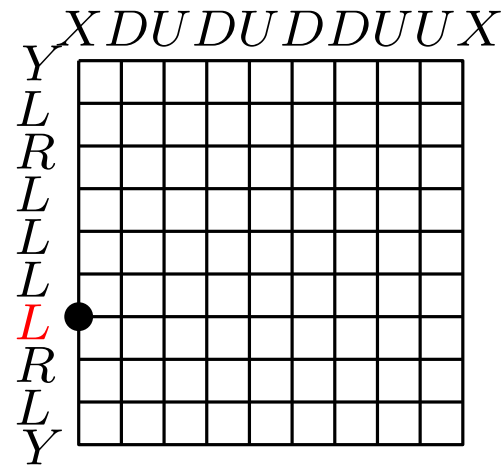
Consider a marked bi-word $(w, m) \in \mathcal{M}$



$$\mathcal{T}^{\nwarrow} \cdot (U, L) \cdot W \cdot (X, Y)$$

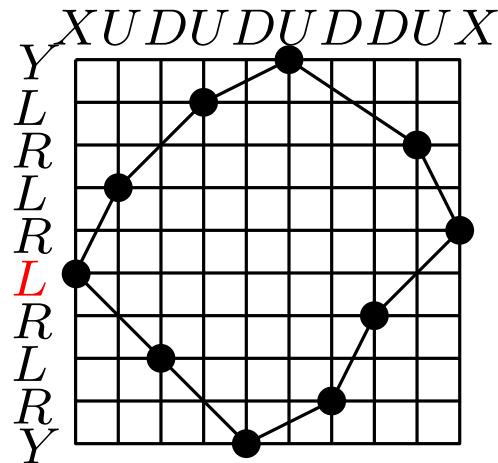
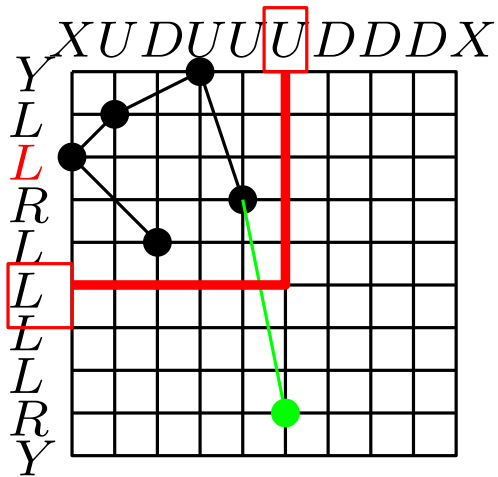
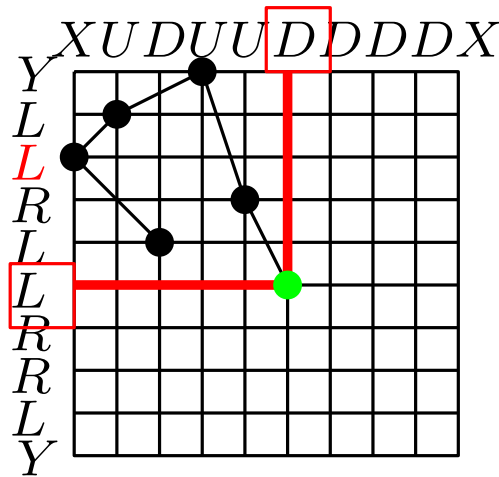
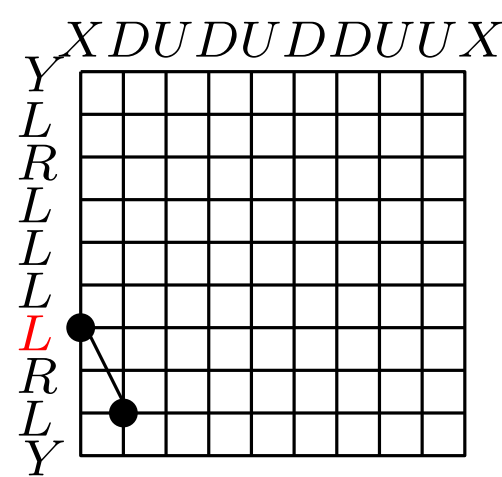


$$\mathcal{T}^{\nwarrow} \cdot (D, L) \cdot W \cdot (X, Y)$$



Decoding a word of $\mathcal{M}$

Consider a marked bi-word $(w, m) \in \mathcal{M}$

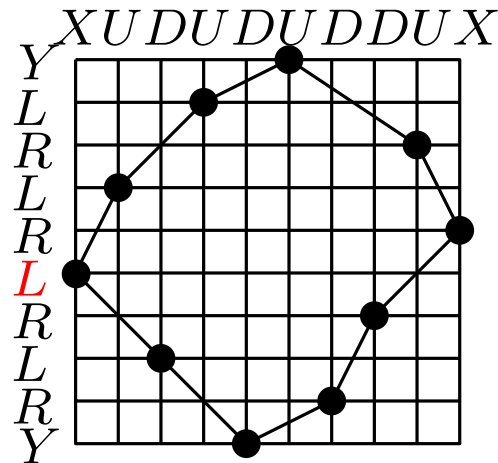
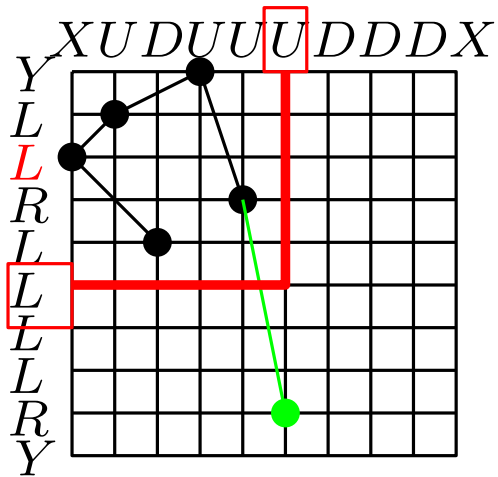


$$\mathcal{T}^{\nwarrow} \cdot (U, L) \cdot W \cdot (X, Y)$$

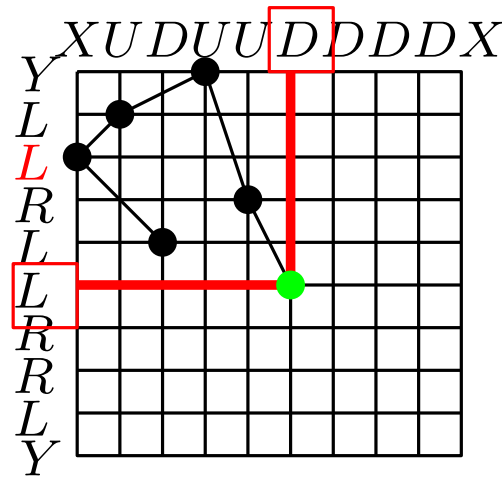
$$\mathcal{T}^{\nwarrow} \cdot (D, L) \cdot W \cdot (X, Y)$$

Decoding a word of $\mathcal{M}$

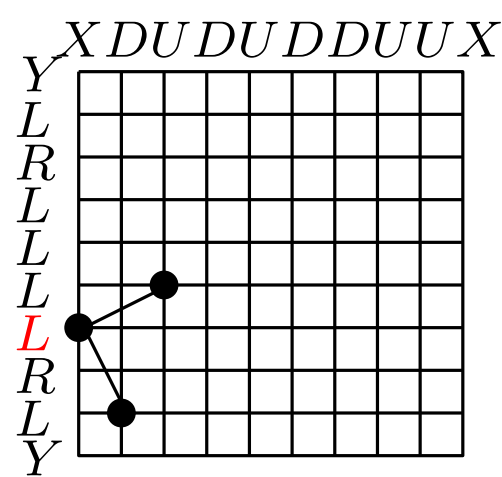
Consider a marked bi-word $(w, m) \in \mathcal{M}$



$$\mathcal{T}^{\nwarrow} \cdot (U, L) \cdot W \cdot (X, Y)$$

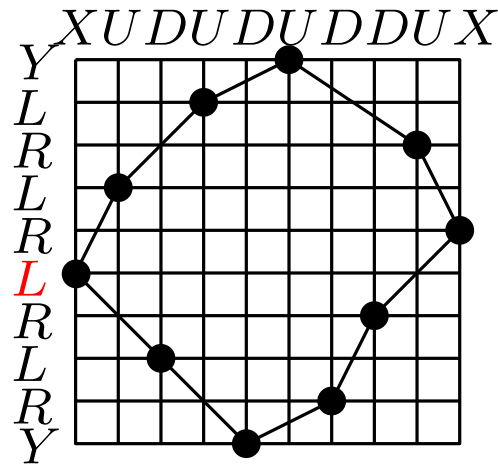
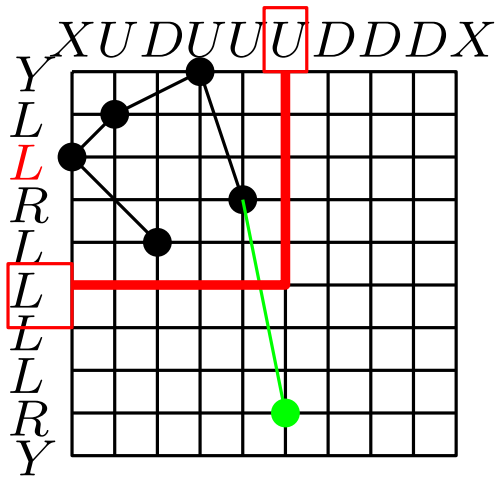
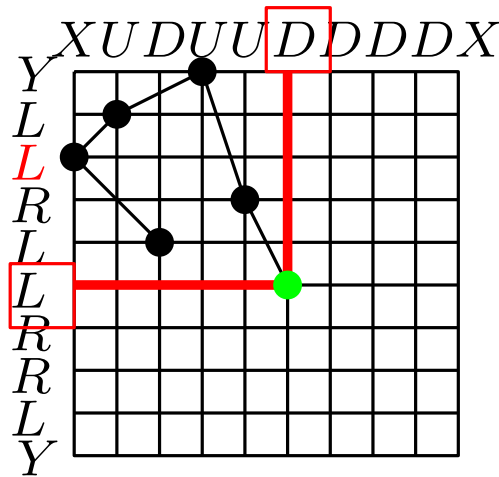
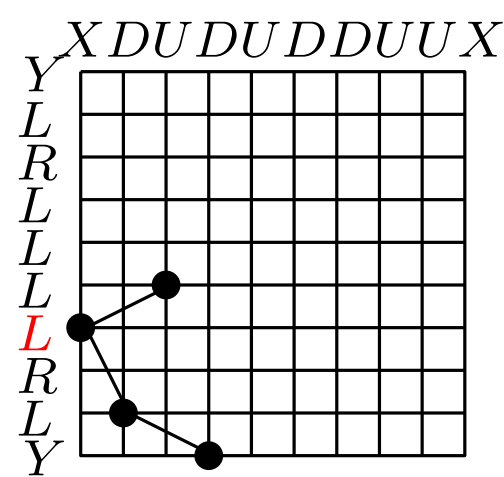


$$\mathcal{T}^{\nwarrow} \cdot (D, L) \cdot W \cdot (X, Y)$$



Decoding a word of $\mathcal{M}$

Consider a marked bi-word $(w, m) \in \mathcal{M}$

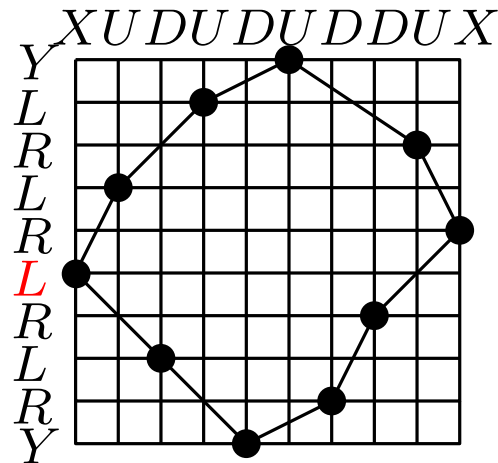
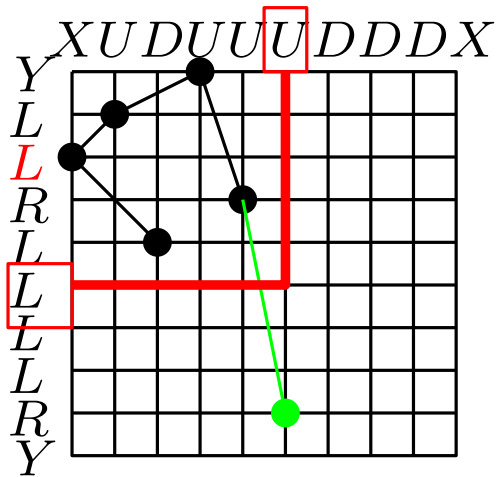
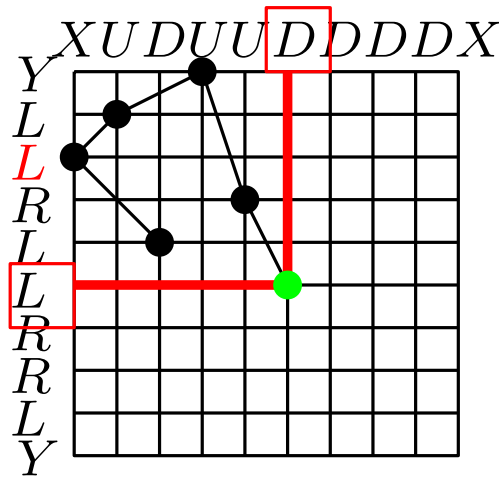
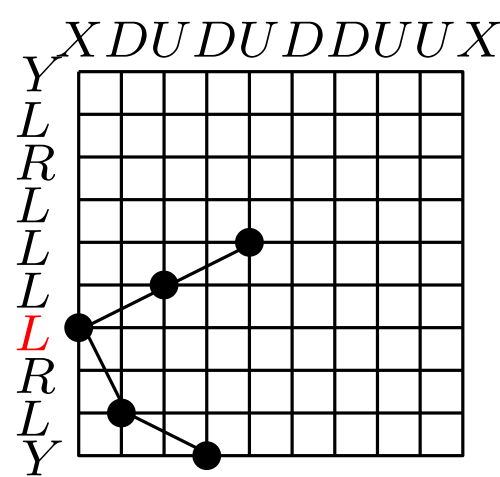


$$\mathcal{T}^{\nwarrow} \cdot (U, L) \cdot W \cdot (X, Y)$$

$$\mathcal{T}^{\nwarrow} \cdot (D, L) \cdot W \cdot (X, Y)$$

Decoding a word of $\mathcal{M}$

Consider a marked bi-word $(w, m) \in \mathcal{M}$

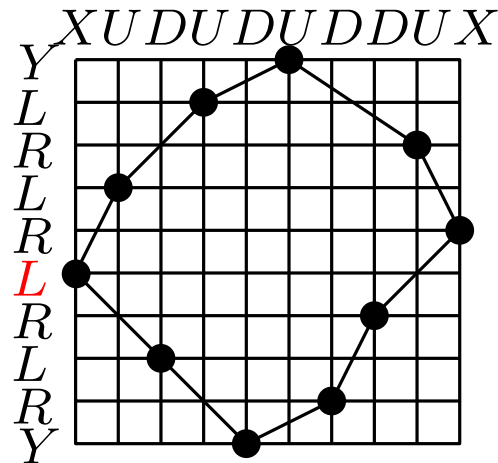
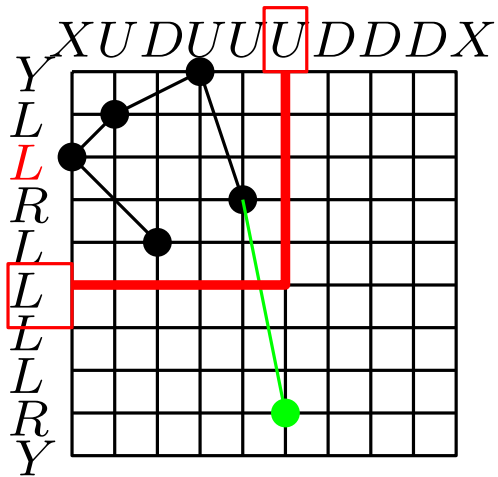
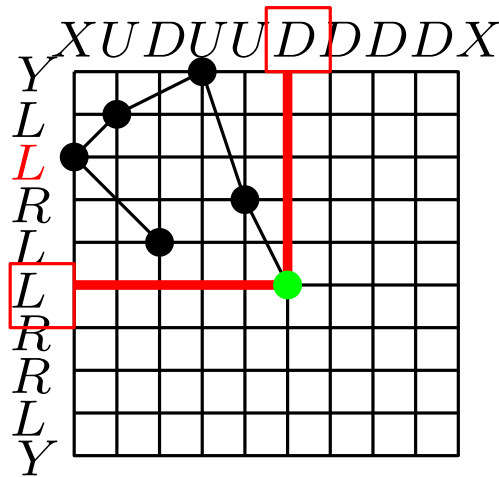
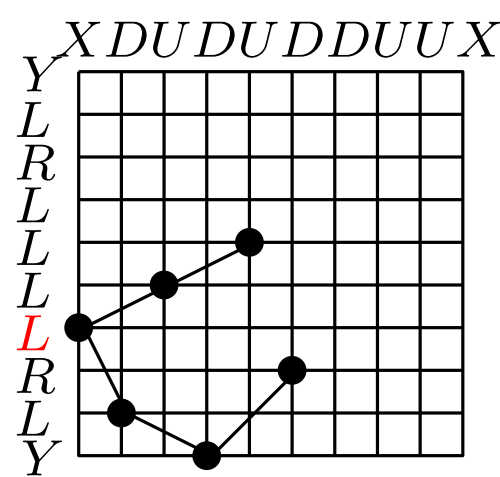


$$\mathcal{T}^{\nwarrow} \cdot (U, L) \cdot W \cdot (X, Y)$$

$$\mathcal{T}^{\nwarrow} \cdot (D, L) \cdot W \cdot (X, Y)$$

Decoding a word of $\mathcal{M}$

Consider a marked bi-word $(w, m) \in \mathcal{M}$

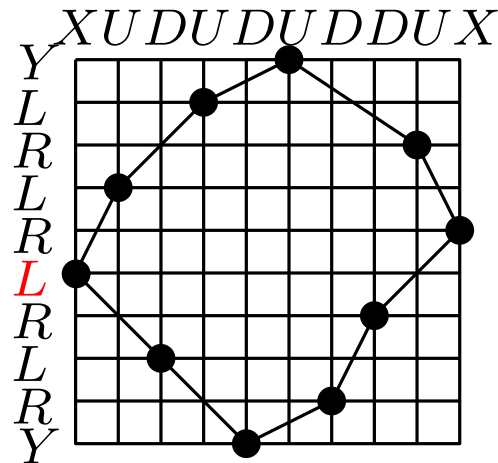
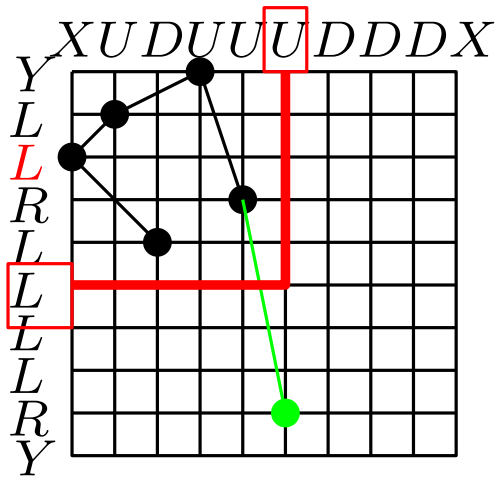


$$\mathcal{T}^{\nwarrow} \cdot (U, L) \cdot W \cdot (X, Y)$$

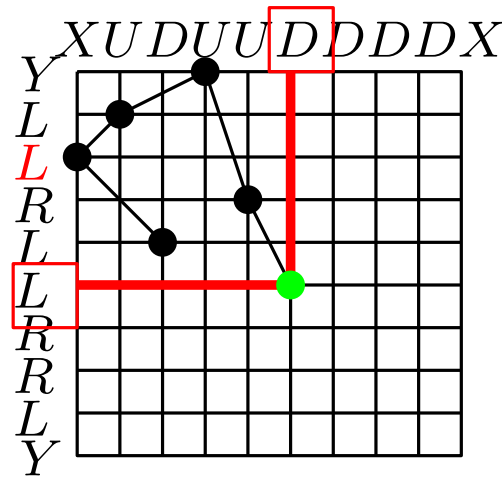
$$\mathcal{T}^{\nwarrow} \cdot (D, L) \cdot W \cdot (X, Y)$$

Decoding a word of $\mathcal{M}$

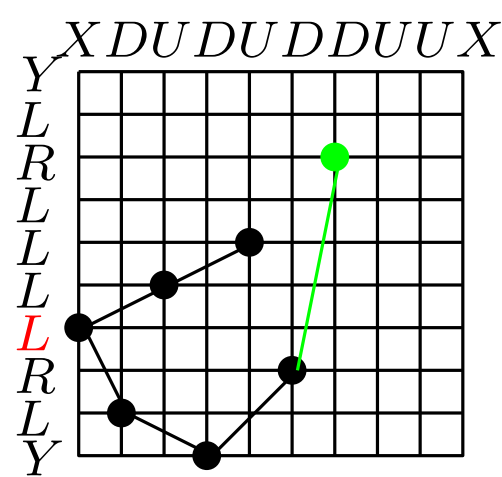
Consider a marked bi-word $(w, m) \in \mathcal{M}$



$$\mathcal{T}^{\nwarrow} \cdot (U, L) \cdot W \cdot (X, Y)$$

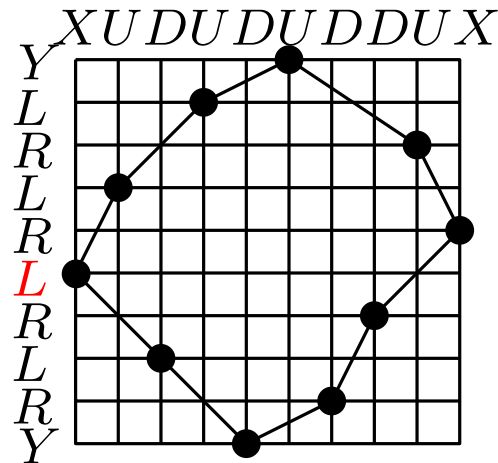
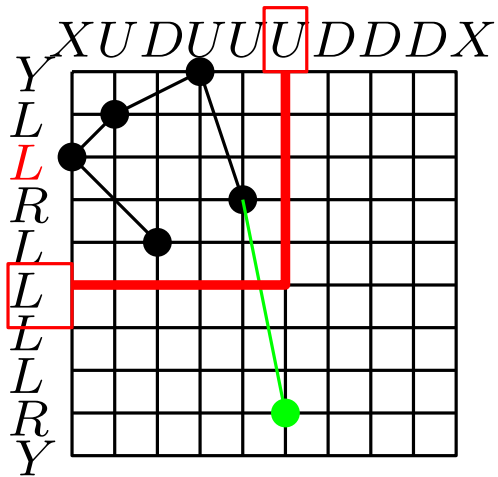


$$\mathcal{T}^{\nwarrow} \cdot (D, L) \cdot W \cdot (X, Y)$$

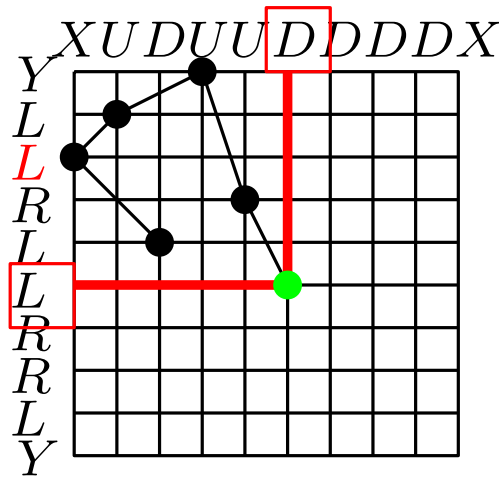


Decoding a word of $\mathcal{M}$

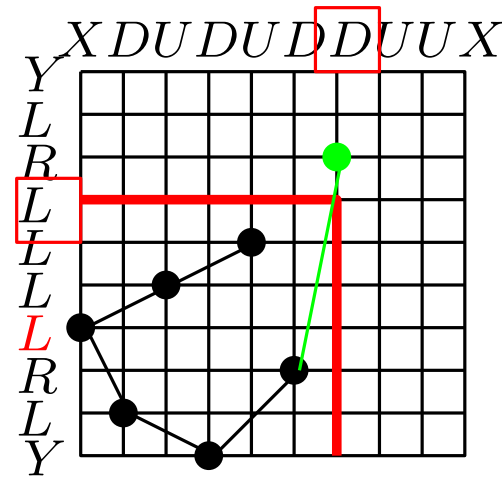
Consider a marked bi-word $(w, m) \in \mathcal{M}$



$$\mathcal{T}^{\nwarrow} \cdot (U, L) \cdot W \cdot (X, Y)$$



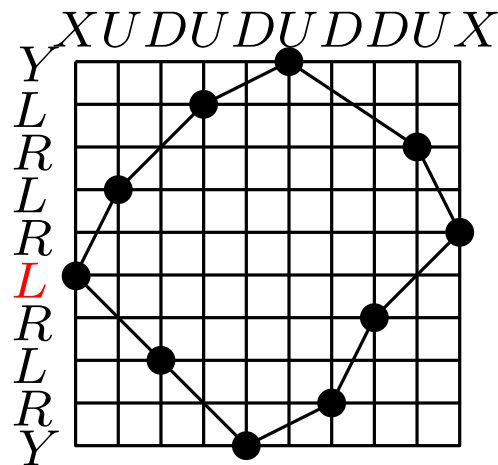
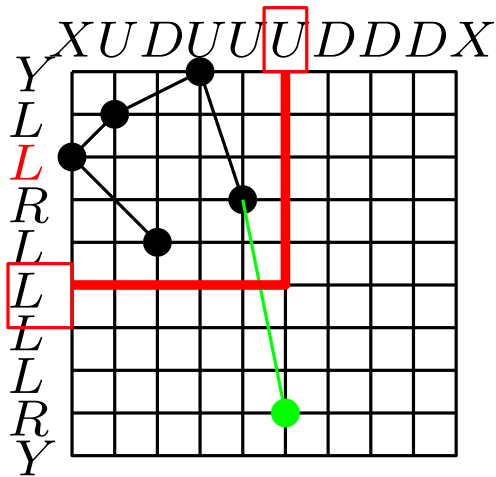
$$\mathcal{T}^{\nwarrow} \cdot (D, L) \cdot W \cdot (X, Y)$$



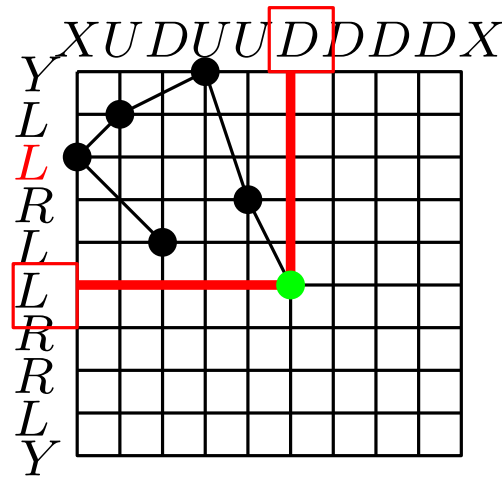
$$\mathcal{T}^{\swarrow} \cdot (D, L), \cdot W \cdot (X, Y)$$

Decoding a word of $\mathcal{M}$

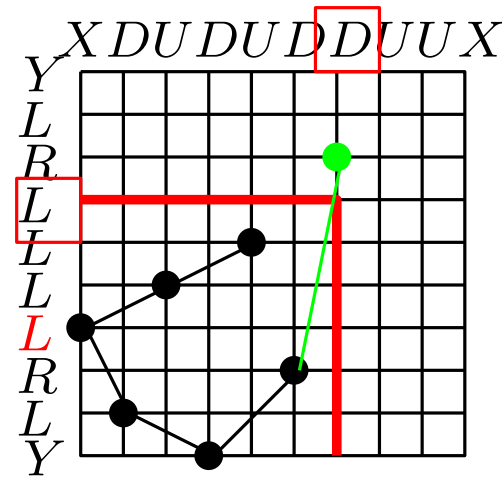
Consider a marked bi-word $(w, m) \in \mathcal{M}$



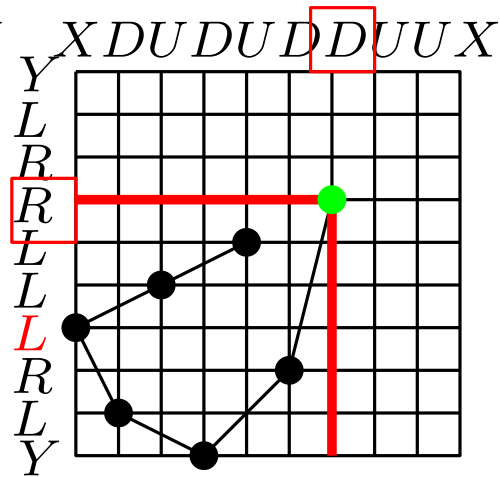
$$\mathcal{T}^{\nwarrow} \cdot (U, L) \cdot W \cdot (X, Y)$$



$$\mathcal{T}^{\nwarrow} \cdot (D, L) \cdot W \cdot (X, Y)$$



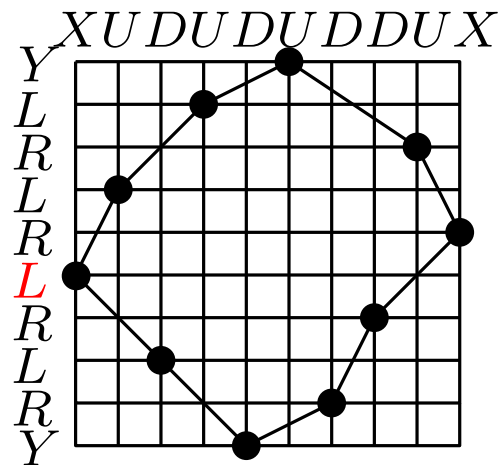
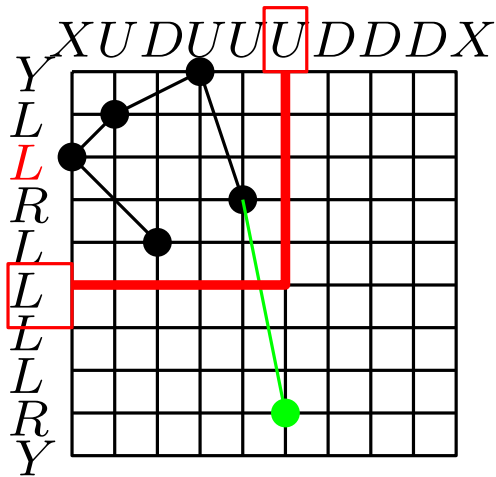
$$\mathcal{T}^{\swarrow} \cdot (D, L), \cdot W \cdot (X, Y)$$



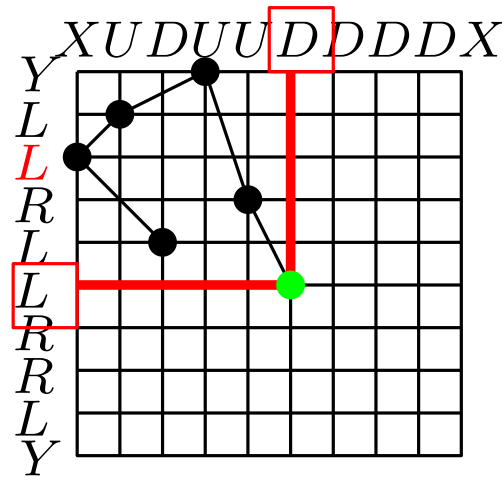
$$\mathcal{T}^{\swarrow} \cdot (D, R), \cdot W \cdot (X, Y)$$

Decoding a word of $\mathcal{M}$

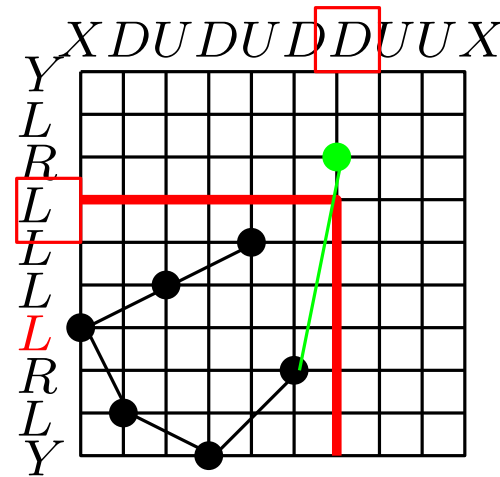
Consider a marked bi-word $(w, m) \in \mathcal{M}$



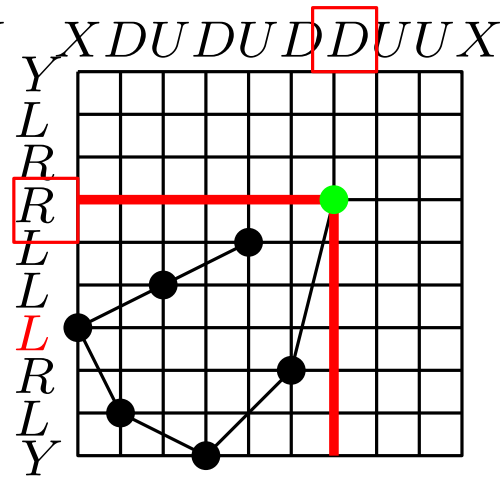
$$\mathcal{T}^{\nwarrow} \cdot (U, L) \cdot W \cdot (X, Y)$$



$$\mathcal{T}^{\nwarrow} \cdot (D, L) \cdot W \cdot (X, Y)$$



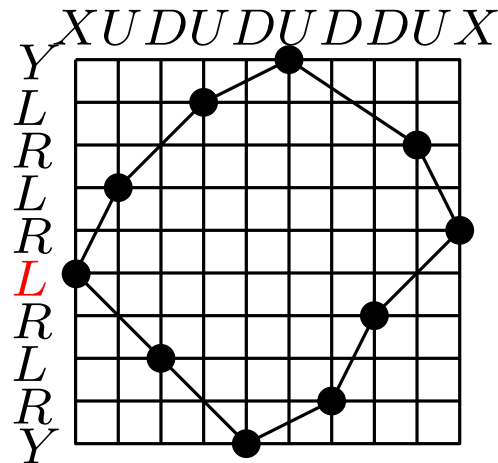
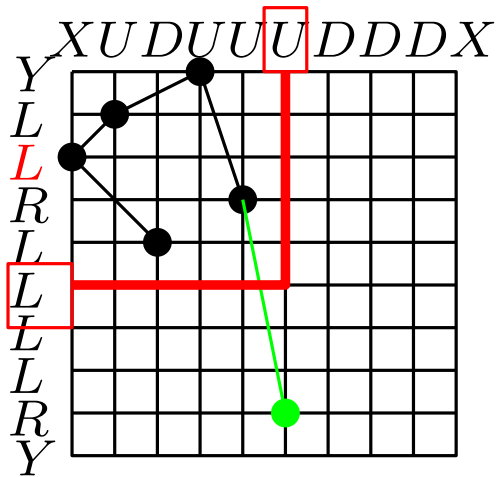
$$\mathcal{T}^{\swarrow} \cdot (D, L), \cdot W \cdot (X, Y)$$



$$\mathcal{T}^{\swarrow} \cdot (D, R), \cdot W \cdot (X, Y)$$

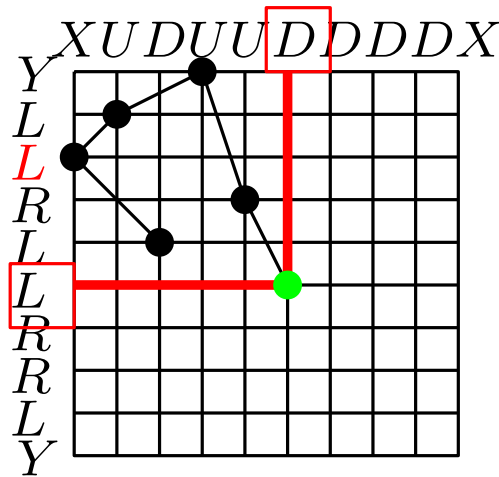
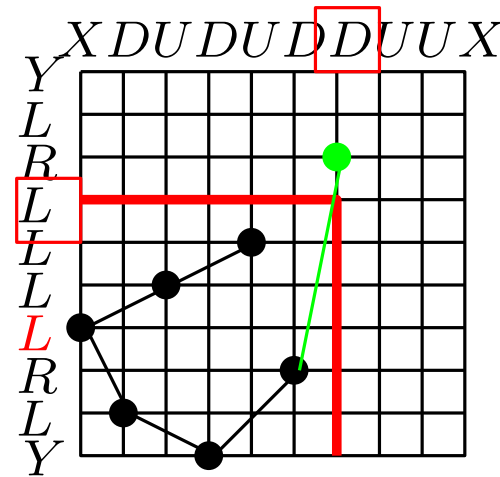
Decoding a word of $\mathcal{M}$

Consider a marked bi-word $(w, m) \in \mathcal{M}$

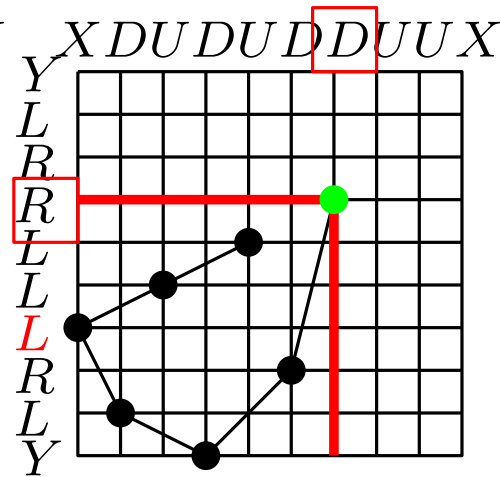


$$\mathcal{T}^{\nwarrow} \cdot (U, L) \cdot W \cdot (X, Y)$$

$$\mathcal{T}^{\nwarrow} \cdot (D, L) \cdot W \cdot (X, Y)$$



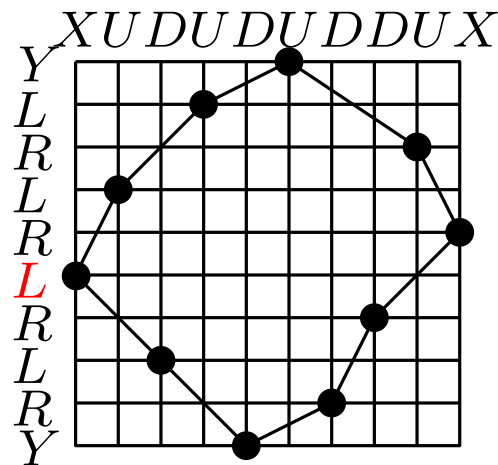
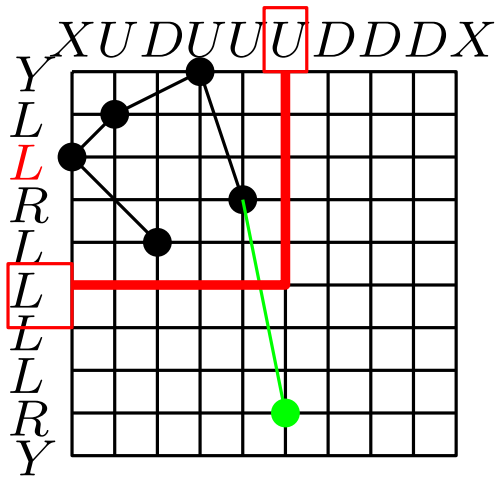
$$\mathcal{T}^{\swarrow} \cdot (D, L), \cdot W \cdot (X, Y)$$



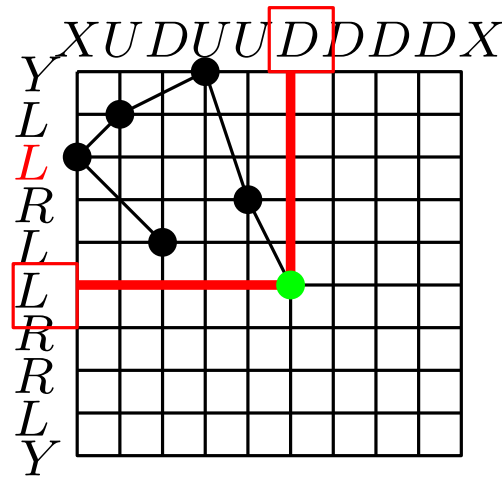
$$\mathcal{T}^{\swarrow} \cdot (D, R), \cdot W \cdot (X, Y)$$

Decoding a word of $\mathcal{M}$

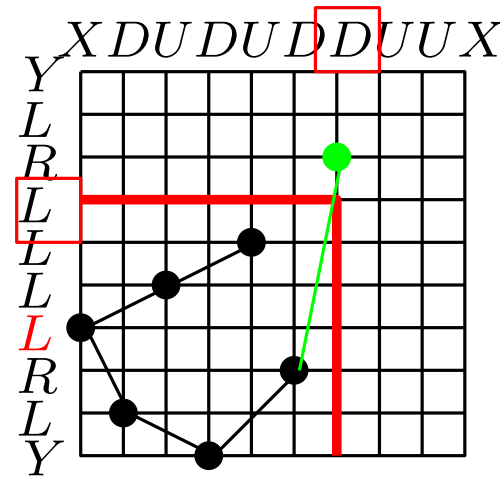
Consider a marked bi-word $(w, m) \in \mathcal{M}$



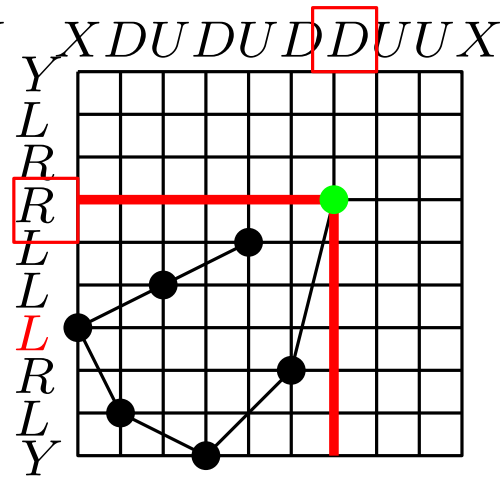
$$\mathcal{T}^{\nwarrow} \cdot (U, L) \cdot W \cdot (X, Y)$$



$$\mathcal{T}^{\nwarrow} \cdot (D, L) \cdot W \cdot (X, Y)$$



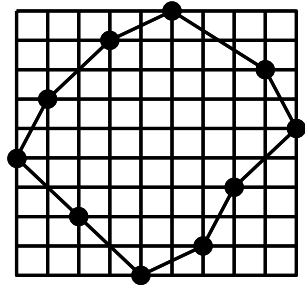
$$\mathcal{T}^{\swarrow} \cdot (D, L), \cdot W \cdot (X, Y)$$



$$\mathcal{T}^{\swarrow} \cdot (D, R), \cdot W \cdot (X, Y)$$

$$\mathcal{S} = \mathcal{M} - \mathcal{T}^{\nwarrow} \cdot \{(U, L), (D, L)\} \cdot W \cdot (X, Y) - \mathcal{T}^{\swarrow} \cdot \{(D, L), (D, R)\} \cdot W \cdot (X, Y)$$

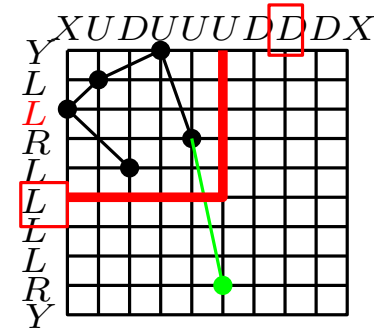
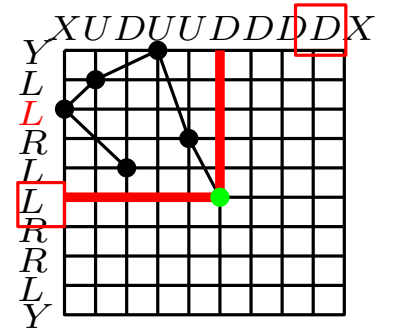
Interpretation of the difference



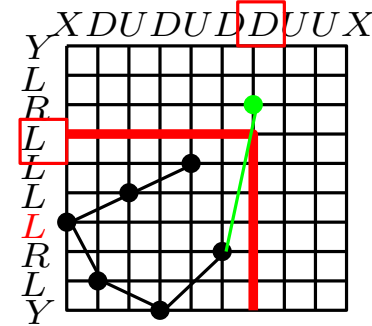
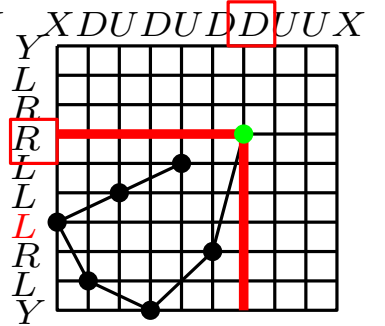
=

XUDUDUDDUX
YRLRLRLRLY

—

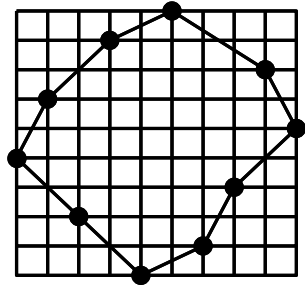


—



$$\begin{aligned} \mathcal{S} &= \mathcal{M} - \mathcal{T}^{\nwarrow} \cdot \{(U, L), (D, L)\} \cdot W \cdot (X, Y) \\ &- \mathcal{T}^{\swarrow} \cdot \{(D, L), (D, R)\} \cdot W \cdot (X, Y) \end{aligned}$$

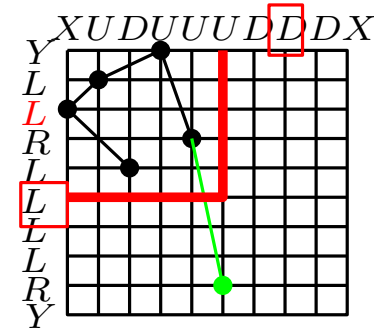
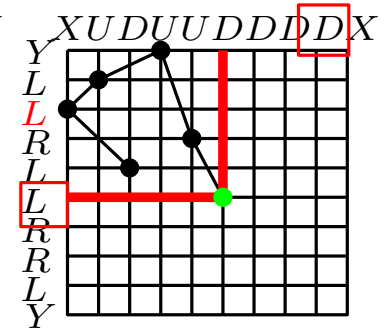
Interpretation of the difference



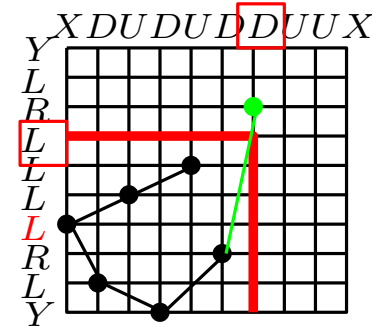
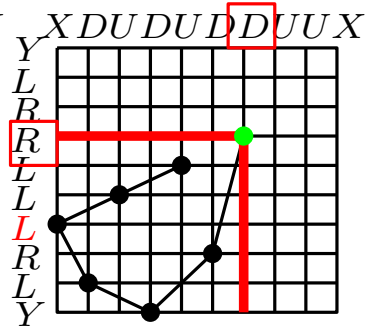
=

$XUDUDUDDUX$
 $YRLRLRLRLY$

—

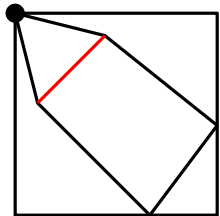


—



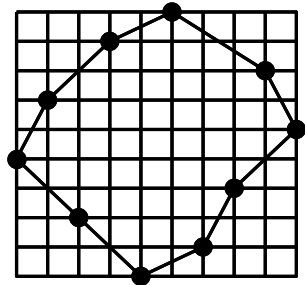
$$\mathcal{S} = \mathcal{M} - \mathcal{T}^{\nwarrow} \cdot \{(U, L), (D, L)\} \cdot W \cdot (X, Y)$$

$$- \mathcal{T}^{\swarrow} \cdot \{(D, L), (D, R)\} \cdot W \cdot (X, Y)$$



$$\equiv \bullet \mathcal{T}^{\searrow} = \{(w, \textcolor{red}{n})\} - \bullet \mathcal{P} \cdot \{(U, L), (D, L)\} \cdot W \cdot (X, Y)$$

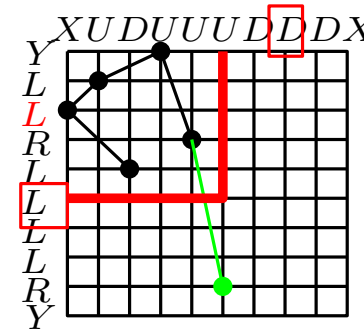
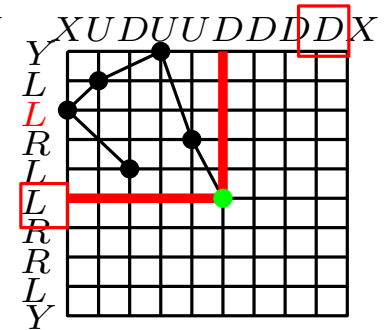
Interpretation of the difference



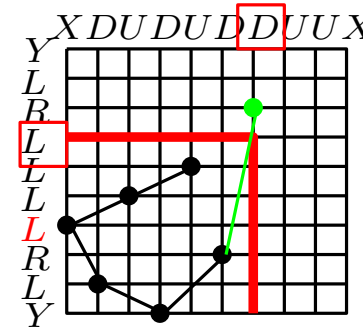
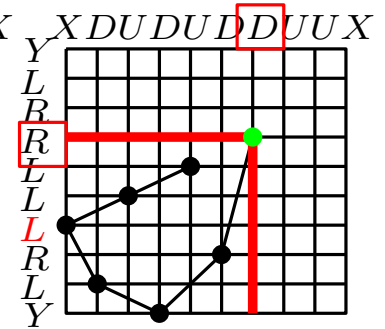
=

$XUDUDUDDUX$
 $YRLRLRLRLY$

—

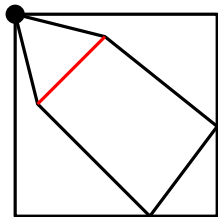


—



$$\mathcal{S} = \mathcal{M} - \mathcal{T}^{\nwarrow} \cdot \{(U, L), (D, L)\} \cdot W \cdot (X, Y)$$

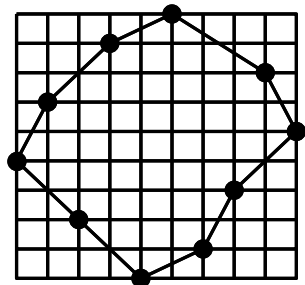
$$- \mathcal{T}^{\swarrow} \cdot \{(D, L), (D, R)\} \cdot W \cdot (X, Y)$$



$$\equiv \bullet \mathcal{T}^{\searrow} = \{(w, \textcolor{red}{n})\} - \bullet \mathcal{P} \cdot \{(U, L), (D, L)\} \cdot W \cdot (X, Y)$$

$\mathcal{P}$ are counted by Catalan numbers

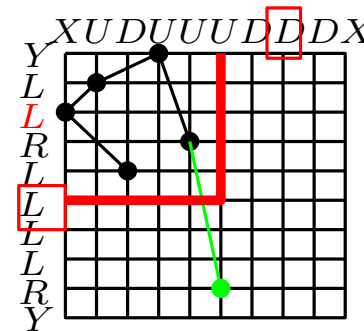
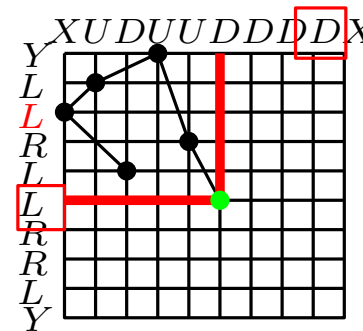
Interpretation of the difference



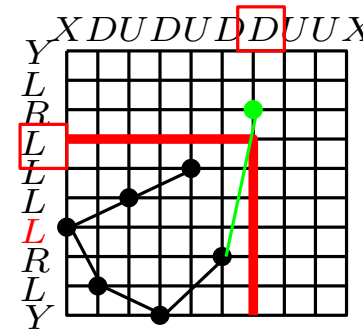
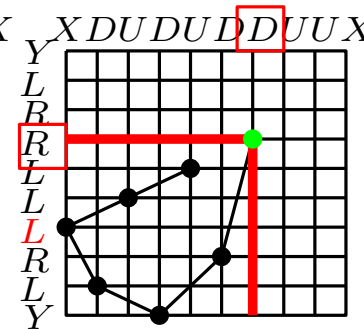
=

$XUDUDUDDUX$
 $YRLRLRLRLY$

—

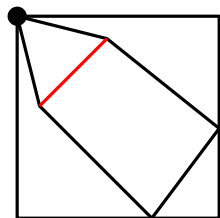


—



$$\mathcal{S} = \mathcal{M} - \mathcal{T}^{\nwarrow} \cdot \{(U, L), (D, L)\} \cdot W \cdot (X, Y)$$

$$- \mathcal{T}^{\swarrow} \cdot \{(D, L), (D, R)\} \cdot W \cdot (X, Y)$$



$$\equiv \bullet \mathcal{T}^{\searrow} = \{(w, n)\} - \bullet \mathcal{P} \cdot \{(U, L), (D, L)\} \cdot W \cdot (X, Y)$$

$\mathcal{P}$ are counted by Catalan numbers

rational gf

algebraic gf

Translating into equation for generating functions

Recall that $\mathcal{W}(t) = \sum_{\bar{w} \in \mathcal{W}} t^{n(\bar{w})} = \frac{1}{1-4t}$ and $\mathcal{M}(t) = \frac{2t^2}{1-4t} + \frac{2t^3}{(1-4t)^2}$

while $\mathcal{P}^{\nwarrow}(t) = \mathcal{P}^{\nearrow}(t) = \frac{1-2t-\sqrt{1-4t}}{2t}$ and $\mathcal{T}^{\nwarrow}(t) = \mathcal{T}^{\nearrow}(t) = \mathcal{T}^{\searrow}(t)$

so that the combinatorial interpretation

$$\begin{aligned} \mathcal{S} &= \mathcal{M} - \mathcal{T}^{\nwarrow} \cdot \{(U, L), (D, L)\} \cdot W \cdot (X, Y) \\ &\quad - \mathcal{T}^{\swarrow} \cdot \{(D, L), (D, R)\} \cdot W \cdot (X, Y) \\ \bullet \mathcal{T}^{\searrow} &= \{(w, \textcolor{red}{n})\} - \bullet \mathcal{P} \cdot \{(U, L), (D, L)\} \cdot W \cdot (X, Y) \end{aligned}$$

translates into the following expressions for generating functions:

$$t\mathcal{T}^{\searrow}(t) = t^2\mathcal{W}(t) - 2t^3\mathcal{P}^{\nwarrow}(t)\mathcal{W}(t)$$

and

$$\begin{aligned} \mathcal{S}(t) &= \mathcal{M}(t) - 4t^2\mathcal{T}^{\searrow}(t)\mathcal{W}(t) \\ &= \frac{2t^2}{1-4t} + \frac{2t}{(1-4t)^2} - \frac{4t^3}{(1-4t)^{3/2}} \end{aligned}$$

Random sampling algorithms

Recall that elements of $\mathcal{M}_n$ can easily be generated uniformly at random.

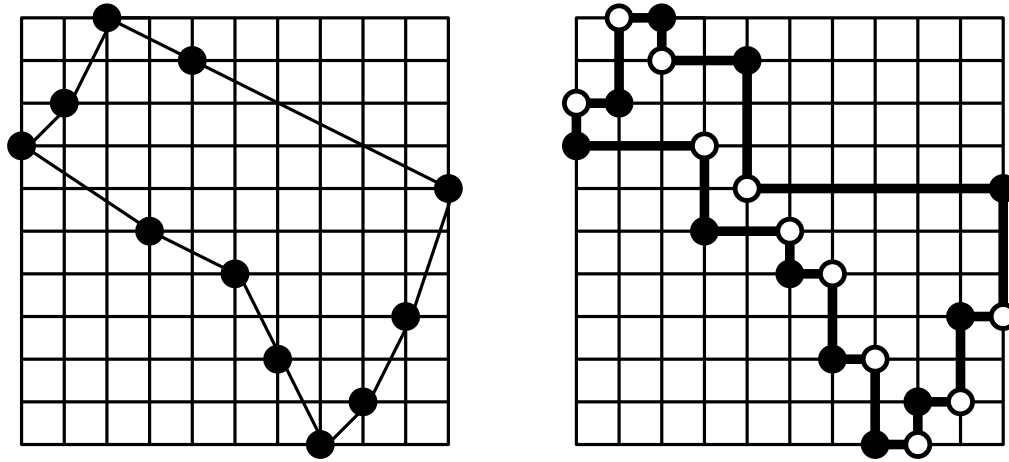
The decoding procedure immediately yields a rejection sampling algorithm:

it consists in generating and trying to decode random elements of $\mathcal{M}_n$ until a valid code is obtained.

The probability of success tends to 1 as n goes to infinity, and decoding can be implemented in linear time so that the algorithm has linear expected complexity.

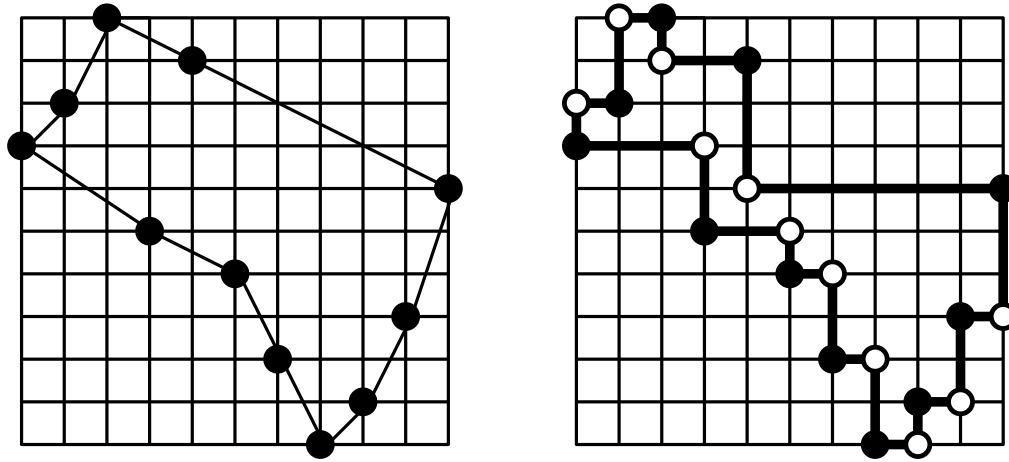
An extended code for convex permutominoes

A bijection between co-indecomposable colored square permutations and convex permutominoes. (Bernini *et al.*, 2007)



An extended code for convex permutominoes

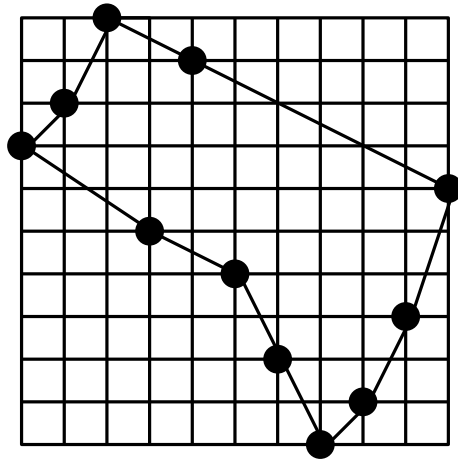
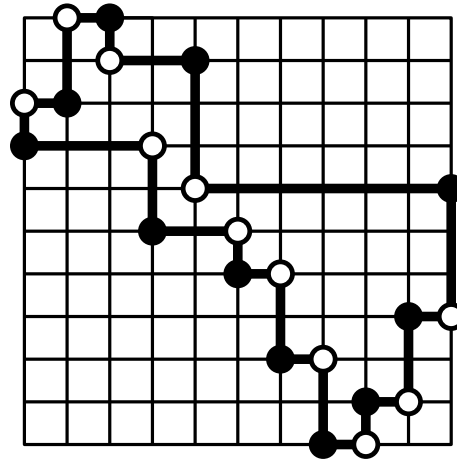
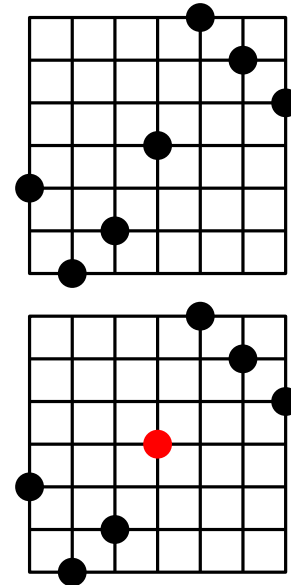
A bijection between co-indecomposable colored square permutations and convex permutominoes. (Bernini *et al.*, 2007)



The square permutation corresponds to the black permutation in the associated convex permutomino.

An extended code for convex permutominoes

A bijection between co-indecomposable colored square permutations and convex permutominoes. (Bernini *et al.*, 2007)

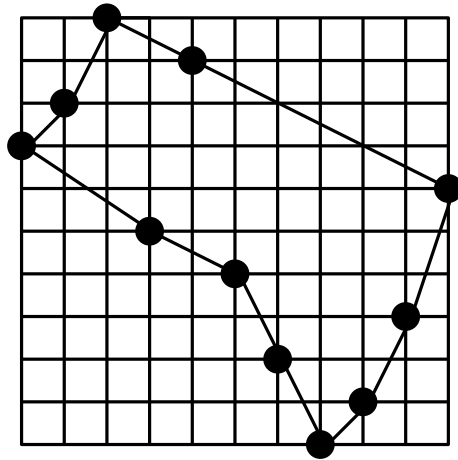
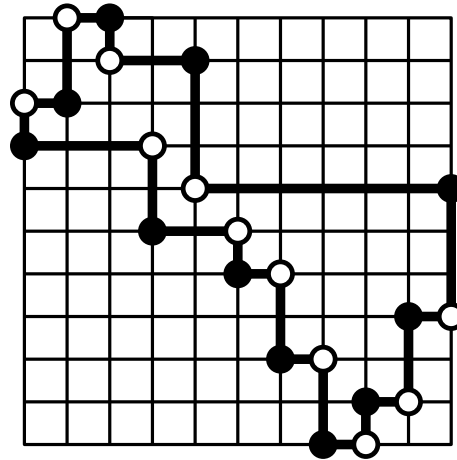


The square permutation corresponds to the black permutation in the associated convex permutomino.

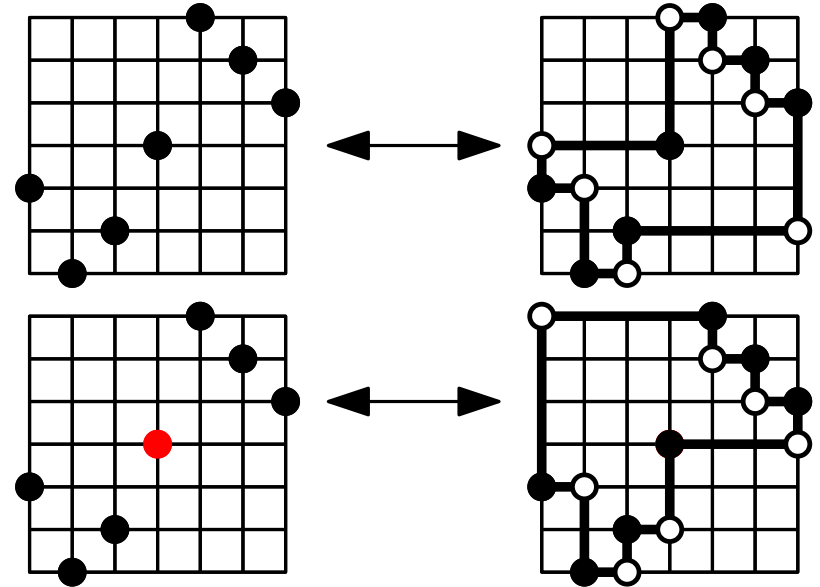
What happens if we have fix points?

An extended code for convex permutominoes

A bijection between co-indecomposable colored square permutations and convex permutominoes. (Bernini *et al.*, 2007)



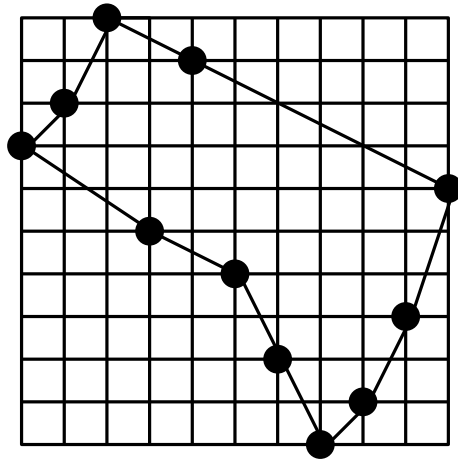
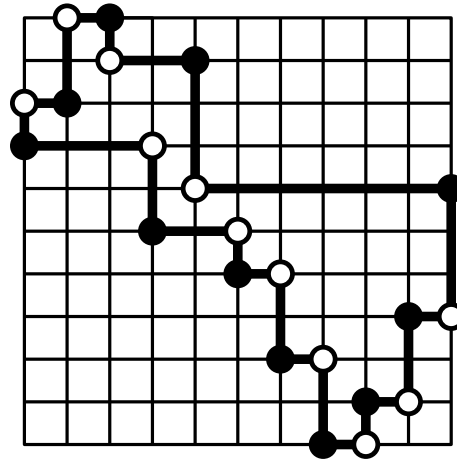
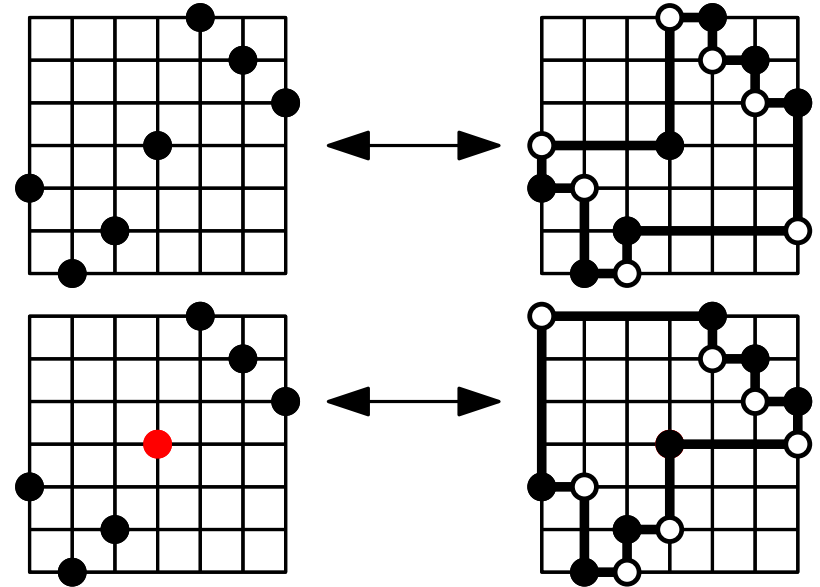
The square permutation corresponds to the black permutation in the associated convex permutomino.



What happens if we have fix points?
It depends if they are colored or not.

An extended code for convex permutominoes

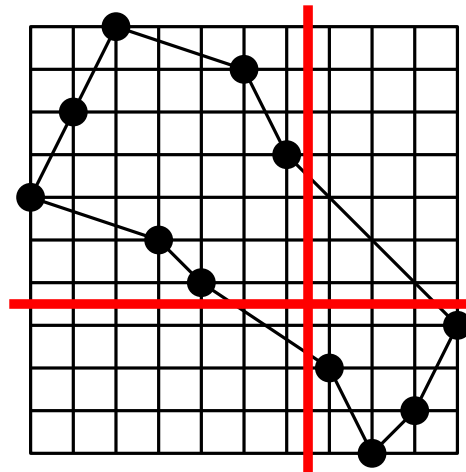
A bijection between co-indecomposable colored square permutations and convex permutominoes. (Bernini *et al.*, 2007)



The square permutation corresponds to the black permutation in the associated convex permutomino.

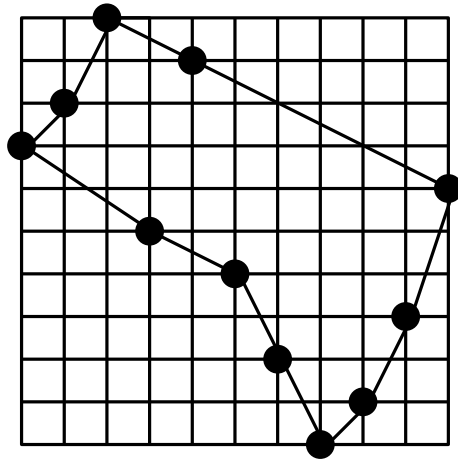
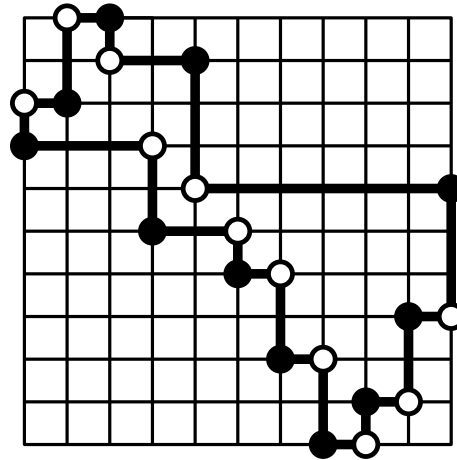
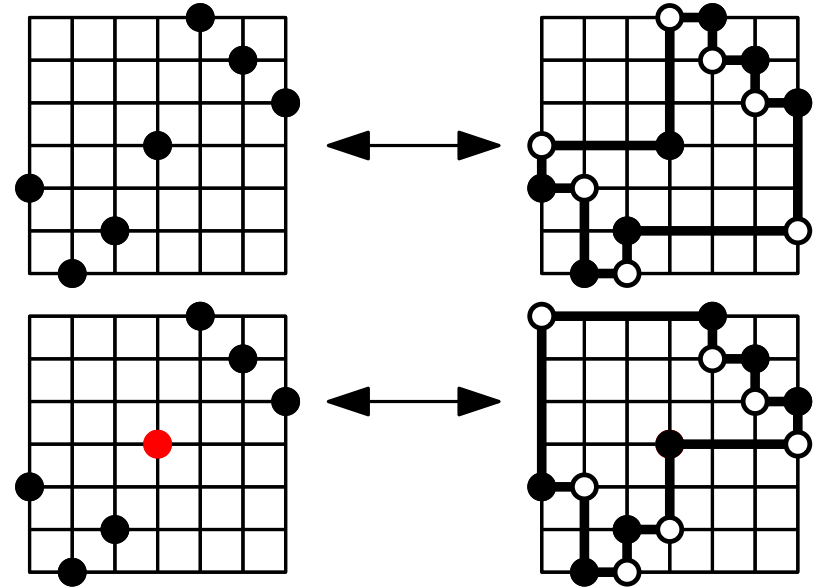
What happens if we have fix points?
It depends if they are colored or not.

Why are co-decomposable permutation forbidden?



An extended code for convex permutominoes

A bijection between co-indecomposable colored square permutations and convex permutominoes. (Bernini *et al.*, 2007)

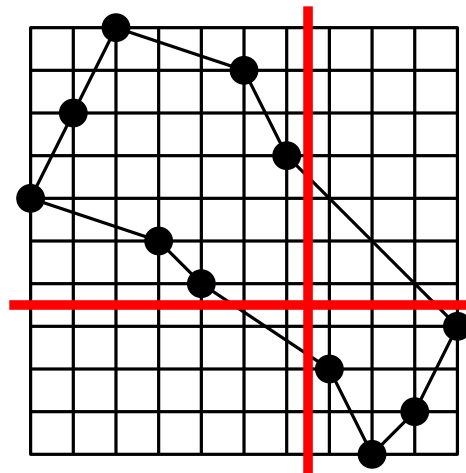
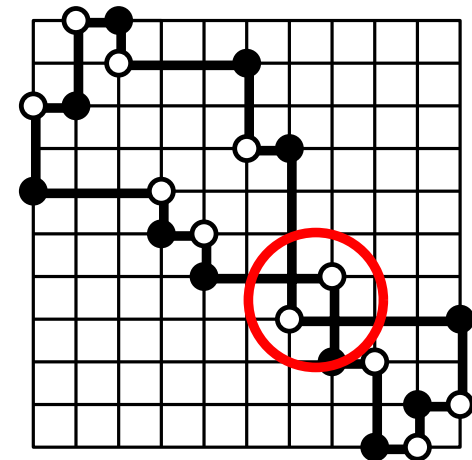


The square permutation corresponds to the black permutation in the associated convex permutomino.

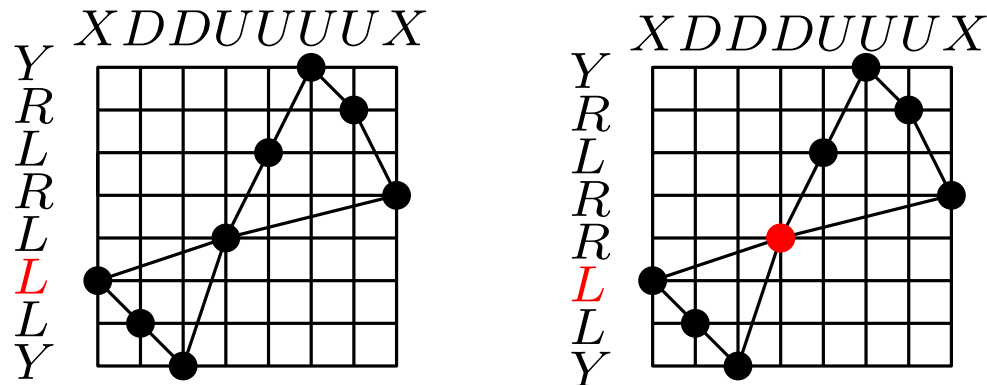
What happens if we have fix points?
It depends if they are colored or not.

Why are co-decomposable permutation forbidden?

They lead to self intersecting permutominoes.



An extended code for convex permutominoes



$u_i = X$ for extremal points in the vertical borders of the bounding box.

$$u_i = \begin{cases} U, & \text{if } (i, \sigma(i)) \text{ is an upper point that is not colored} \\ D, & \text{otherwise} \end{cases}$$

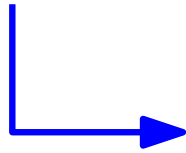
$v_i = Y$ for extremal points in the horizontal borders of the bounding box.

$$v_i = \begin{cases} L, & \text{if } (i, \sigma(i)) \text{ is left point which is not an upper right one or is not colored} \\ R, & \text{otherwise} \end{cases}$$

What's next?

Next talk!

Typical shape of a large square permutation



J. Borga and E. Slivken

Ongoing project with Borga, Slivken

Asymptotic of square permutations with few internal points

Polyominoes and permutominoes with few internal points?

Thank you !