Reconfiguration par 2-adjacence dans la grille carrée

Florian Galliot, Sylvain Gravier, Isabelle Sivignon

23 novembre 2023

Journées Graphes et Algorithmes, Lyon

Description du problème

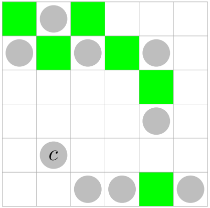
- Reconfiguration dans la grille carrée (Demaine et. al., 2002).

Description du problème

- Reconfiguration dans la grille carrée (Demaine et. al., 2002).
- *Configuration* : un ensemble fini de *pièces* indistinguables, au plus une par case.
- *Mouvement* autorisé : déplacer une pièce d'une case occupée vers une case inoccupée, de sorte que cette pièce soit **voisine d'au moins deux autres pièces** (contrainte de *2-adjacence*).
- Voisinage = haut, bas, gauche, droite.
- Remarque : les mouvements ne sont pas réversibles en général...

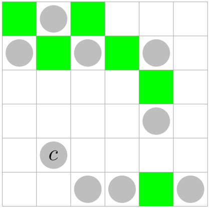
Description du problème

- Reconfiguration dans la grille carrée (Demaine et. al., 2002).
- *Configuration* : un ensemble fini de *pièces* indistinguables, au plus une par case.
- *Mouvement* autorisé : déplacer une pièce d'une case occupée vers une case inoccupée, de sorte que cette pièce soit **voisine d'au moins deux autres pièces** (contrainte de *2-adjacence*).
- Voisinage = haut, bas, gauche, droite.
- Remarque : les mouvements ne sont pas réversibles en général...



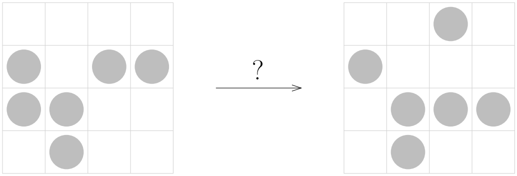
Description du problème

- Reconfiguration dans la grille carrée (Demaine et. al., 2002).
- *Configuration* : un ensemble fini de *pièces* indistinguables, au plus une par case.
- *Mouvement* autorisé : déplacer une pièce d'une case occupée vers une case inoccupée, de sorte que cette pièce soit **voisine d'au moins deux autres pièces** (contrainte de *2-adjacence*).
- Voisinage = haut, bas, gauche, droite.
- Remarque : les mouvements ne sont pas réversibles en général...

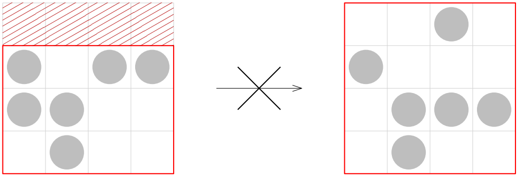


Etant donné un *puzzle* $A \xrightarrow{?} B$ i.e. une configuration de départ A et une configuration d'arrivée B telles que $|A| = |B|$, a-t-on $A \rightarrow B$?

Span d’une configuration



Span d'une configuration



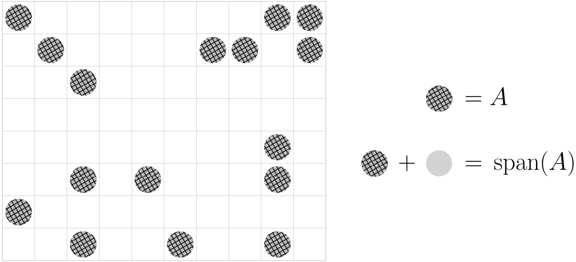
Certaines positions sont hors d'atteinte à jamais !

Span d'une configuration

- *Span* d'une configuration A : l'ensemble span(A) obtenu en ajoutant à A les positions respectant la condition de 2-adjacence et en itérant ce processus jusqu'à stationnement.

Span d'une configuration

- *Span* d'une configuration A : l'ensemble span(A) obtenu en ajoutant à A les positions respectant la condition de 2-adjacence et en itérant ce processus jusqu'à stationnement.



Span d'une configuration

- *Span* d'une configuration A : l'ensemble $\text{span}(A)$ obtenu en ajoutant à A les positions respectant la condition de 2-adjacence et en itérant ce processus jusqu'à stationnement.
- Union de rectangles à distance au moins 3 les uns des autres.
- Toute position hors de $\text{span}(A)$ est **inatteignable** depuis A.

Span d'une configuration

- *Span* d'une configuration A : l'ensemble span(A) obtenu en ajoutant à A les positions respectant la condition de 2-adjacence et en itérant ce processus jusqu'à stationnement.
- Union de rectangles à distance au moins 3 les uns des autres.
- Toute position hors de span(A) est **inatteignable** depuis A.
- **Le span ne fait que diminuer (au sens large de l'inclusion) au cours des mouvements !**

Span d'une configuration

- *Span* d'une configuration A : l'ensemble span(A) obtenu en ajoutant à A les positions respectant la condition de 2-adjacence et en itérant ce processus jusqu'à stationnement.
- Union de rectangles à distance au moins 3 les uns des autres.
- Toute position hors de span(A) est **inatteignable** depuis A.
- **Le span ne fait que diminuer (au sens large de l'inclusion) au cours des mouvements !**

Condition nécessaire de résolubilité

Si $A \to B$ alors span$(A) \supseteq$ span(B).

Span d'une configuration

- *Span* d'une configuration A : l'ensemble span(A) obtenu en ajoutant à A les positions respectant la condition de 2-adjacence et en itérant ce processus jusqu'à stationnement.
- Union de rectangles à distance au moins 3 les uns des autres.
- Toute position hors de span(A) est **inatteignable** depuis A.
- **Le span ne fait que diminuer (au sens large de l'inclusion) au cours des mouvements !**

Condition nécessaire de résolubilité

Si $A \to B$ alors span(A) $\supseteq$ span(B).

- *Pièces "bonus"* d'un puzzle $A \stackrel{?}{\to} B$: un ensemble $E \subseteq A$ tel que span($A \setminus E$) $\supseteq$ span(B).

Span d'une configuration

- *Span* d'une configuration A : l'ensemble span(A) obtenu en ajoutant à A les positions respectant la condition de 2-adjacence et en itérant ce processus jusqu'à stationnement.
- Union de rectangles à distance au moins 3 les uns des autres.
- Toute position hors de span(A) est **inatteignable** depuis A.
- **Le span ne fait que diminuer (au sens large de l'inclusion) au cours des mouvements !**

Condition nécessaire de résolubilité

Si $A \to B$ alors span(A) $\supseteq$ span(B).

- *Pièces "bonus"* d'un puzzle $A \xrightarrow{?} B$: un ensemble $E \subseteq A$ tel que span($A \setminus E$) $\supseteq$ span(B).
- Etant donné un puzzle, **combien peut-on trouver de pièces bonus ?**
  - 0 pièce bonus : pas de solution, sauf si $A = B$...
  - 1 pièce bonus ? (cas "+1")
  - ≥ 2 pièces bonus ? (cas "+2")

Span d'une configuration

- *Span* d'une configuration A : l'ensemble span(A) obtenu en ajoutant à A les positions respectant la condition de 2-adjacence et en itérant ce processus jusqu'à stationnement.
- Union de rectangles à distance au moins 3 les uns des autres.
- Toute position hors de span(A) est **inatteignable** depuis A.
- **Le span ne fait que diminuer (au sens large de l'inclusion) au cours des mouvements !**

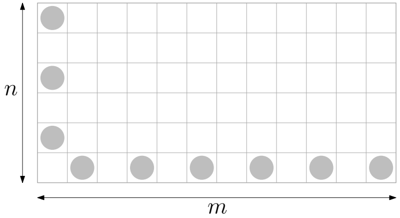
Condition nécessaire de résolubilité

Si $A \to B$ alors span(A) $\supseteq$ span(B).

- *Pièces "bonus"* d'un puzzle $A \xrightarrow{?} B$: un ensemble $E \subseteq A$ tel que span($A \setminus E$) $\supseteq$ span(B).
- Etant donné un puzzle, **combien peut-on trouver de pièces bonus ?**
  - 0 pièce bonus : pas de solution, sauf si $A = B$...
  - 1 pièce bonus ? (cas "+1")
  - **$\geq$ 2 pièces bonus ? (cas "+2")**

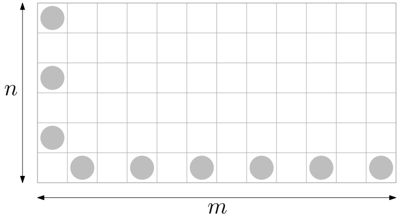
Transformations de 'L's

Un 'L' de span $m \times n$ contient $\lceil \frac{m+n}{2} \rceil$ pièces :



Transformations de 'L's

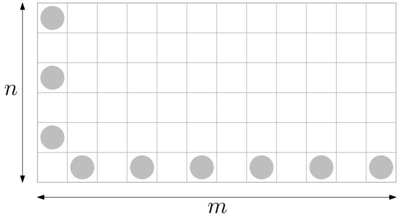
Un 'L' de span $m \times n$ contient $\lceil \frac{m+n}{2} \rceil$ pièces :



- **La clé :** avec 2 pièces bonus, on peut transformer un 'L' de taille $m \times n$ en n'importe quel autre 'L' de même span, en $O(mn)$ mouvements.

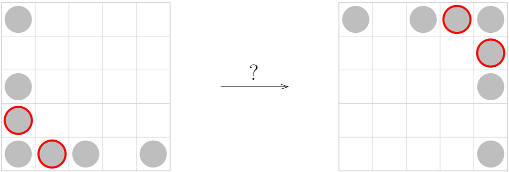
Transformations de 'L's

Un 'L' de span $m \times n$ contient $\lceil \frac{m+n}{2} \rceil$ pièces :

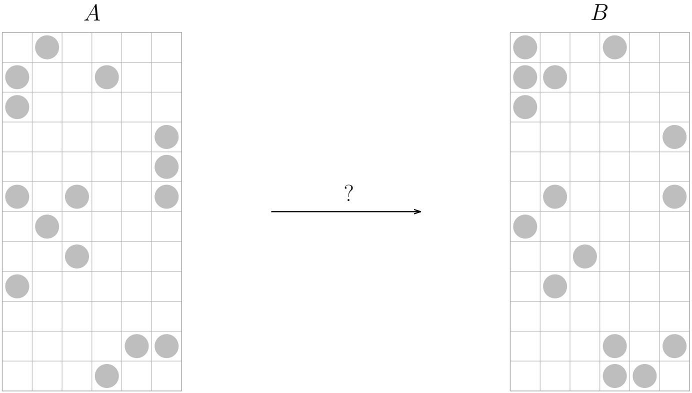


- **La clé :** avec 2 pièces bonus, on peut transformer un 'L' de taille $m \times n$ en n'importe quel autre 'L' de même span, en $O(mn)$ mouvements.

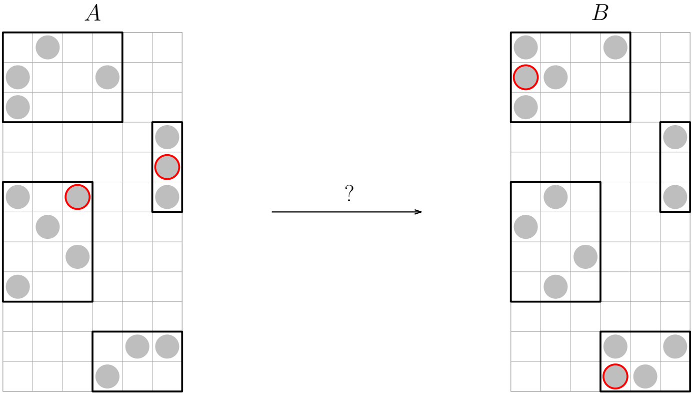
Exemple : *flip* par "vague" de pièces



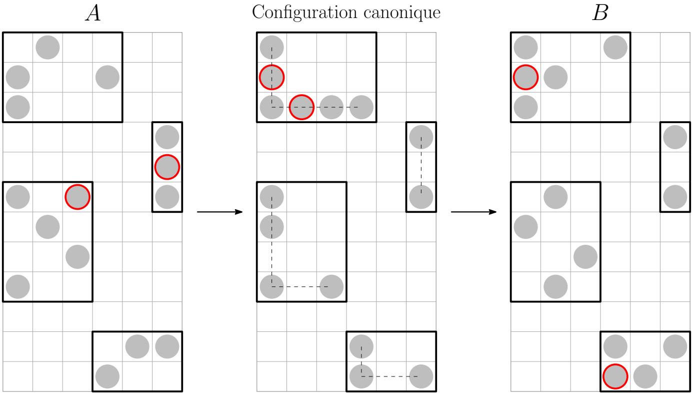
Une méthode avec 2 pièces bonus



Une méthode avec 2 pièces bonus

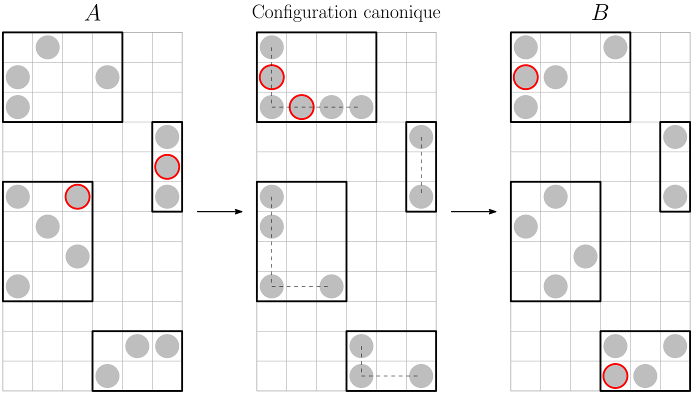


Une méthode avec 2 pièces bonus



- On transite par une *configuration canonique* avec des 'L's.

Une méthode avec 2 pièces bonus



- On transite par une *configuration canonique* avec des 'L's.

Lemme [Demaine et. al. (2002)]

Avec 2 pièces bonus, on peut transformer toute configuration C en sa configuration canonique associée L_C, **de manière réversible**.

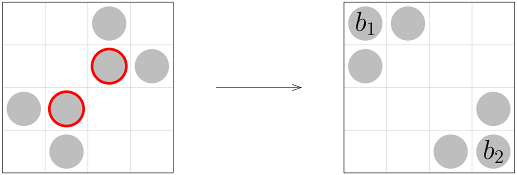
Condition suffisante avec 2 pièces bonus

Theorème [Demaine et. al. (2002)]

Soit $A \xrightarrow{?} B$ un puzzle tel que :

- Il y a deux pièces bonus $\{a_1, a_2\}$.
- Il existe $b_1 \neq b_2$ dans B tels que b_1 a au moins deux voisins dans B et b_2 a au moins deux voisins dans $B \setminus \{b_1\}$.

Alors $A \to B$, en $O(N^3)$ mouvements où $N = |A| = |B|$.



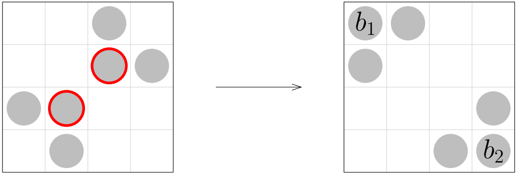
Condition suffisante avec 2 pièces bonus

Theorème [Demaine et. al. (2002)]

Soit $A \xrightarrow{?} B$ un puzzle tel que :

- Il y a deux pièces bonus $\{a_1, a_2\}$, telles que $\text{span}(A \setminus \{a_1, a_2\}) = \text{span}(B)$.
- Il existe $b_1 \neq b_2$ dans B tels que b_1 a au moins deux voisins dans B et b_2 a au moins deux voisins dans $B \setminus \{b_1\}$.

Alors $A \rightarrow B$, en $O(N^3)$ mouvements où $N = |A| = |B|$.



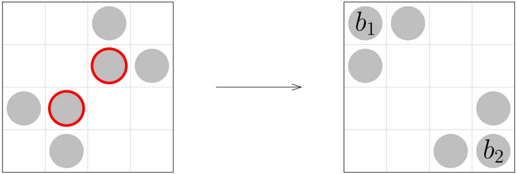
Condition suffisante avec 2 pièces bonus

Theorème [Demaine et. al. (2002)]

Soit $A \xrightarrow{?} B$ un puzzle tel que :

- Il y a deux pièces bonus $\{a_1, a_2\}$, telles que $\mathrm{span}(A \setminus \{a_1, a_2\}) = \mathrm{span}(B)$.
- Il existe $b_1 \neq b_2$ dans B tels que b_1 a au moins deux voisins dans B et b_2 a au moins deux voisins dans $B \setminus \{b_1\}$.

Alors $A \rightarrow B$, en $O(N^3)$ mouvements où $N = |A| = |B|$.



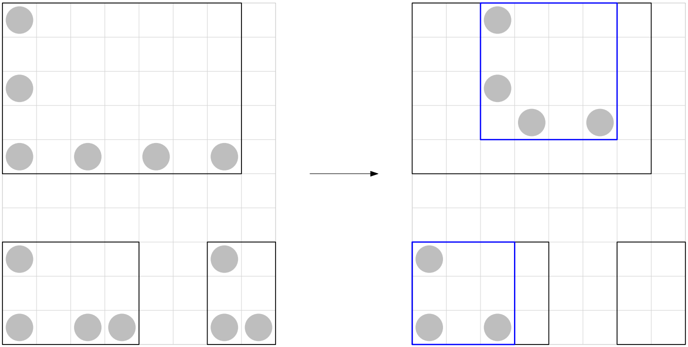
- Si $\mathrm{span}(A \setminus \{a_1, a_2\}) \supsetneq \mathrm{span}(B)$: $A \xrightarrow{\text{OK}} L_A \xrightarrow{?!?} L_B \xrightarrow{\text{OK}} B$

Cas où $\text{span}(A \setminus \{a_1, a_2\}) \supsetneq \text{span}(B)$

- Si chaque composante de $\text{span}(A \setminus \{a_1, a_2\})$ contient au plus une composante de $\text{span}(B)$?

Cas où span$(A \setminus \{a_1, a_2\}) \supsetneq$ span(B)

- Si chaque composante de span$(A \setminus \{a_1, a_2\})$ contient au plus une composante de span(B) ?



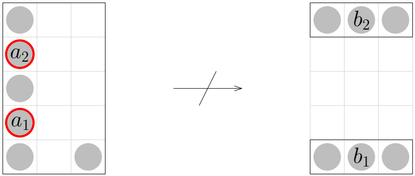
OK : avec 2 pièces bonus, on peut rétrécir des 'L'.

Cas où $\mathrm{span}(A \setminus \{a_1, a_2\}) \supsetneq \mathrm{span}(B)$

- Si une composante de $\mathrm{span}(A \setminus \{a_1, a_2\})$ contient plusieurs composantes de $\mathrm{span}(B)$?

Cas où span$(A \setminus \{a_1, a_2\}) \supsetneq$ span(B)

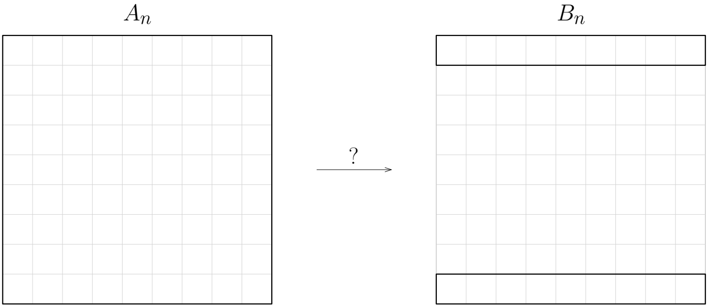
- Si une composante de span$(A \setminus \{a_1, a_2\})$ contient plusieurs composantes de span(B) ?



Pas OK : le puzzle ci-dessus n'est pas résoluble !

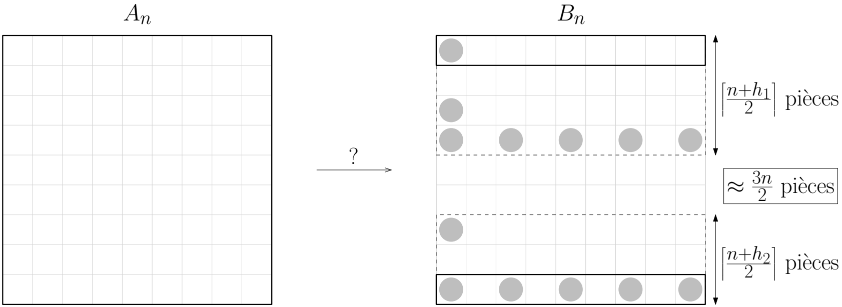
Cas où span$(A \setminus \{a_1, a_2\}) \supsetneq$ span(B)

Contre-exemple plus général en taille $n \times n$:



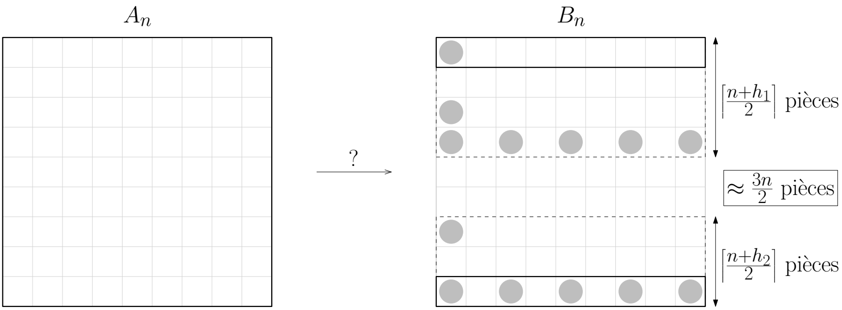
Cas où $\mathrm{span}(A \setminus \{a_1, a_2\}) \supsetneq \mathrm{span}(B)$

Contre-exemple plus général en taille $n \times n$:



Cas où span($A \setminus \{a_1, a_2\}$) $\supsetneq$ span(B)

Contre-exemple plus général en taille $n \times n$:



Il existe des contre-exemples en taille $n \times n$ avec $\lfloor \frac{n}{2} \rfloor - 1$ pièces bonus.

- Aucune quantité $O(1)$ de pièces bonus n'est suffisante !

Un algorithme de résolution

On souhaiterait un algorithme permettant de résoudre certains puzzles où le span d'arrivée est fragmenté.

Un algorithme de résolution

On souhaiterait un algorithme permettant de résoudre certains puzzles où le span d'arrivée est fragmenté.

Au départ : un 'L' de taille $m \times n$ + k pièces "en main".

Objectif : une configuration C inscrite dans le même rectangle.

Un algorithme de résolution

On souhaiterait un algorithme permettant de résoudre certains puzzles où le span d'arrivée est fragmenté.

Au départ : un 'L' de taille $m \times n$ + k pièces "en main".

Objectif : une configuration C inscrite dans le même rectangle.

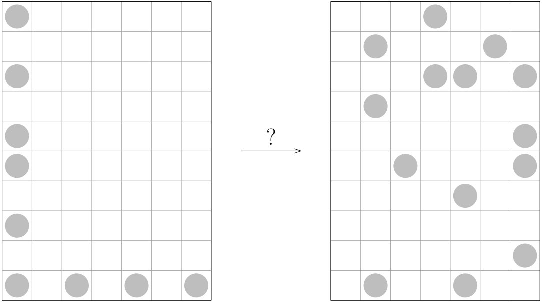
- Idée : faire une vague de pièces qui balaie le rectangle (flip du 'L') en déposant des pièces aux bons endroits pour former C.

Un algorithme de résolution

On souhaiterait un algorithme permettant de résoudre certains puzzles où le span d'arrivée est fragmenté.

Au départ : un 'L' de taille $m \times n$ + k pièces "en main".
Objectif : une configuration C inscrite dans le même rectangle.

- Idée : faire une vague de pièces qui balaie le rectangle (flip du 'L') en déposant des pièces aux bons endroits pour former C.



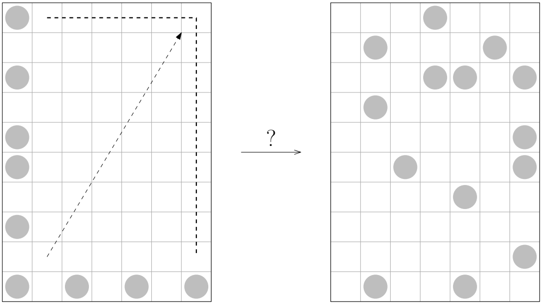
Un algorithme de résolution

On souhaiterait un algorithme permettant de résoudre certains puzzles où le span d'arrivée est fragmenté.

Au départ : un 'L' de taille $m \times n$ + k pièces "en main".

Objectif : une configuration C inscrite dans le même rectangle.

- Idée : faire une vague de pièces qui balaie le rectangle (flip du 'L') en déposant des pièces aux bons endroits pour former C.



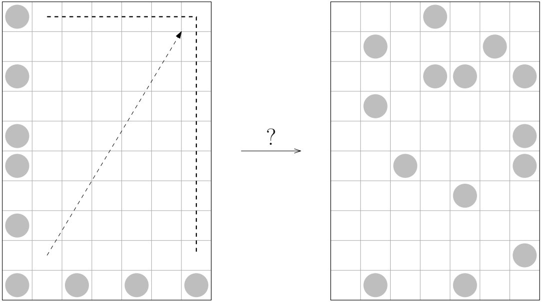
Un algorithme de résolution

On souhaiterait un algorithme permettant de résoudre certains puzzles où le span d'arrivée est fragmenté.

Au départ : un 'L' de taille $m \times n$ + k pièces "en main".

Objectif : une configuration C inscrite dans le même rectangle.

- Idée : faire une vague de pièces qui balaie le rectangle (flip du 'L') en déposant des pièces aux bons endroits pour former C.



Trivial si $k \geq 2 + |C|$... on veut faire mieux.

Un algorithme de résolution

On souhaiterait un algorithme permettant de résoudre certains puzzles où le span d'arrivée est fragmenté.

Au départ : un 'L' de taille $m \times n$ + k pièces "en main".
Objectif : une configuration C inscrite dans le même rectangle.

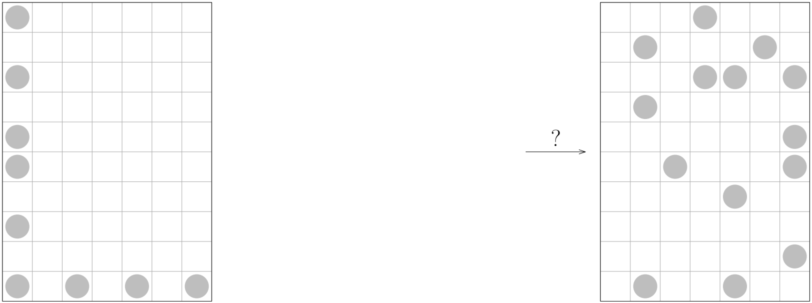
- Idée : faire une vague de pièces qui balaie le rectangle (flip du 'L') en déposant des pièces aux bons endroits pour former C.
- Récurrence : on construit d'abord la moitié C_1 de C qui contient le moins de pièces, puis appel récursif sur l'autre moitié.

Un algorithme de résolution

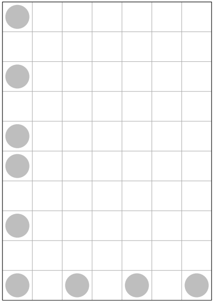
On souhaiterait un algorithme permettant de résoudre certains puzzles où le span d'arrivée est fragmenté.
Au départ : un 'L' de taille $m \times n$ + k pièces "en main".
Objectif : une configuration C inscrite dans le même rectangle.

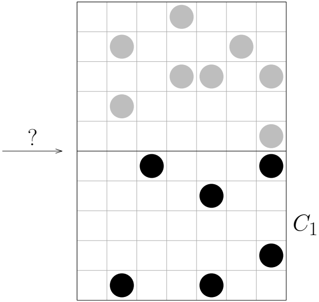
- Idée : faire une vague de pièces qui balaie le rectangle (flip du 'L') en déposant des pièces aux bons endroits pour former C.
- Récurrence : on construit d'abord la moitié C_1 de C qui contient le moins de pièces, puis appel récursif sur l'autre moitié.
- Utilisation de nos k pièces en main :
  - 2 pièces de support pour les transformations de 'L'
  - les autres = pièces de construction pour former C_1

Un algorithme de résolution

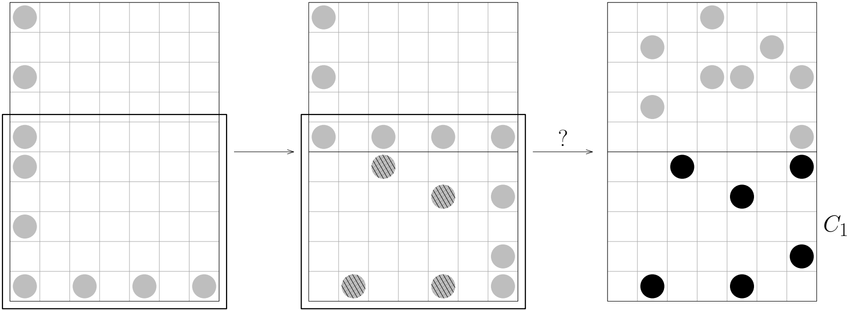


Un algorithme de résolution



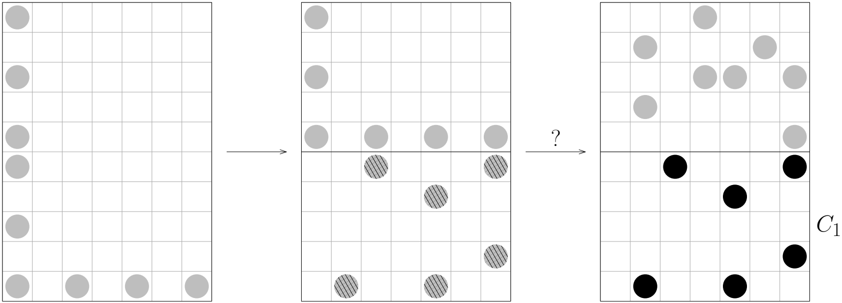


Un algorithme de résolution



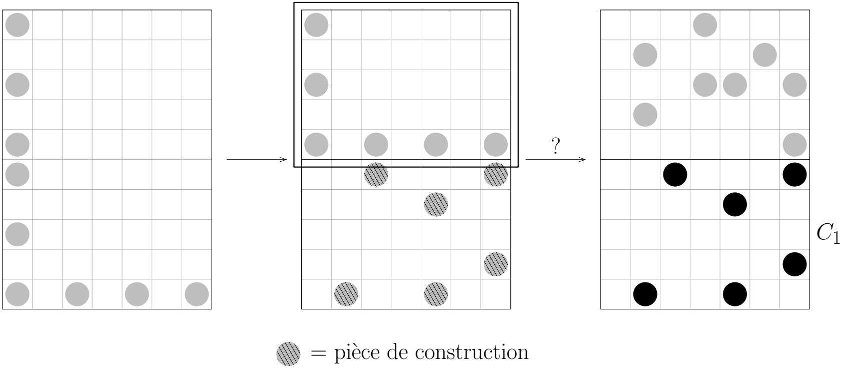
= pièce de construction

Un algorithme de résolution



$=$ pièce de construction

Un algorithme de résolution



$=$ pièce de construction

Un algorithme de résolution

On souhaiterait un algorithme permettant de résoudre certains puzzles où le span d'arrivée est fragmenté.

Au départ : un 'L' de taille $m \times n$ + k pièces "en main".
Objectif : une configuration C inscrite dans le même rectangle.

- Idée : faire une vague de pièces qui balaie le rectangle (flip du 'L') en déposant des pièces aux bons endroits pour former C.
- Récurrence : on construit d'abord la moitié C_1 de C qui contient le moins de pièces, puis appel récursif sur l'autre moitié.
- Utilisation de nos k pièces en main :
  - 2 pièces de support pour les transformations de 'L'
  - les autres = pièces de construction pour former C_1
- Ne fonctionne que si on a suffisamment de pièces en main...

Un algorithme de résolution

On souhaiterait un algorithme permettant de résoudre certains puzzles où le span d'arrivée est fragmenté.
Au départ : un 'L' de taille $m \times n$ + k pièces "en main".
Objectif : une configuration C inscrite dans le même rectangle.

- Idée : faire une vague de pièces qui balaie le rectangle (flip du 'L') en déposant des pièces aux bons endroits pour former C.
- Récurrence : on construit d'abord la moitié C_1 de C qui contient le moins de pièces, puis appel récursif sur l'autre moitié.
- Utilisation de nos k pièces en main :
  - 2 pièces de support pour les transformations de 'L'
  - les autres = pièces de construction pour former C_1
- Ne fonctionne que si on a suffisamment de pièces en main...
  - Pour avoir assez de pièces de construction i.e. $k \geq 2 + |C_1|$: il suffit que $k > \frac{|C|}{2} + 1$.

Un algorithme de résolution

On souhaiterait un algorithme permettant de résoudre certains puzzles où le span d'arrivée est fragmenté.

Au départ : un 'L' de taille $m \times n$ + k pièces "en main".
Objectif : une configuration C inscrite dans le même rectangle.

- Idée : faire une vague de pièces qui balaie le rectangle (flip du 'L') en déposant des pièces aux bons endroits pour former C.
- Récurrence : on construit d'abord la moitié C_1 de C qui contient le moins de pièces, puis appel récursif sur l'autre moitié.
- Utilisation de nos k pièces en main :
  - 2 pièces de support pour les transformations de 'L'
  - les autres = pièces de construction pour former C_1
- Ne fonctionne que si on a suffisamment de pièces en main...
  - Pour avoir assez de pièces de construction i.e. $k \geq 2 + |C_1|$: il suffit que $k > \frac{|C|}{2} + 1$.
  - Pour faire marcher l'hérédité : il suffit que $k \geq |C| - \left\lceil \frac{m+n}{2} \right\rceil + \left\lceil \frac{\min(m,n)}{2} \right\rceil + 2$.

Résultat principal

Pour une configuration C, on note $\min_C$ le nombre minimum de pièces d'une configuration ayant un span égal à $\text{span}(C)$.

Résultat principal

Pour une configuration C, on note $\min_C$ le nombre minimum de pièces d'une configuration ayant un span égal à $\mathrm{span}(C)$.

Theorème [G., Gravier, Sivignon (2021)]

Soit $A \xrightarrow{?} B$ un puzzle à N pièces tel que :

- Il y a au moins deux pièces bonus $\{a_1, a_2\}$, avec $\mathrm{span}(A \setminus \{a_1, a_2\})$ en un seul morceau.
- Il existe $b_1 \neq b_2$ dans B tels que b_1 a au moins deux voisins dans B et b_2 a au moins deux voisins dans $B \setminus \{b_1\}$.
- $N \geq \frac{3}{2} \max(\min_A, \min_B) + 2$.

Alors $A \rightarrow B$, en $O(N^3)$ mouvements.

Méthode : $A \xrightarrow[O(N^3)]{\text{algo Demaine}} L_A \xrightarrow[O(N^2)]{\text{algo GGS}} L_B \xrightarrow[O(N^3)]{\text{algo Demaine}} B$.

Résultat principal

Pour une configuration C, on note $\min_C$ le nombre minimum de pièces d'une configuration ayant un span égal à $\mathrm{span}(C)$.

Theorème [G., Gravier, Sivignon (2021)]

Soit $A \xrightarrow{?} B$ un puzzle à N pièces tel que :

- Il y a au moins deux pièces bonus $\{a_1, a_2\}$, avec $\mathrm{span}(A \setminus \{a_1, a_2\})$ en un seul morceau.
- Il existe $b_1 \neq b_2$ dans B tels que b_1 a au moins deux voisins dans B et b_2 a au moins deux voisins dans $B \setminus \{b_1\}$.
- $N \geq \frac{3}{2}\max(\min_A, \min_B) + 2$.

Alors $A \rightarrow B$, en $O(N^3)$ mouvements.

Méthode : $A \xrightarrow[O(N^3)]{\text{algo Demaine}} L_A \xrightarrow[O(N^2)]{\text{algo GGS}} L_B \xrightarrow[O(N^3)]{\text{algo Demaine}} B$.

- Le contre-exemple vu précédemment vérifie $N = \frac{3}{2}\max(\min_A, \min_B) - 3$: pour un span rectangulaire, **notre borne est serrée à un $O(1)$ additif près.**

Conclusion et perspectives

- La qualité des pièces ne fait pas tout : leur quantité est importante.

Conclusion et perspectives

- La qualité des pièces ne fait pas tout : leur quantité est importante.
- Borne serrée pour un span carré... mais pour un span rectangulaire ?

Conclusion et perspectives

- La qualité des pièces ne fait pas tout : leur quantité est importante.
- Borne serrée pour un span carré... mais pour un span rectangulaire ?
- Cas avec 1 pièce bonus ? Difficile !

Conclusion et perspectives

- La qualité des pièces ne fait pas tout : leur quantité est importante.
- Borne serrée pour un span carré... mais pour un span rectangulaire ?
- Cas avec 1 pièce bonus ? Difficile !
  $\rightarrow$ Puzzles "minimum+1"

Conclusion et perspectives

- La qualité des pièces ne fait pas tout : leur quantité est importante.
- Borne serrée pour un span carré... mais pour un span rectangulaire ?
- Cas avec 1 pièce bonus ? Difficile !
  $\rightarrow$ Puzzles "minimum+1"
- Complexité algorithmique ? Dans les graphes généraux ?