ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Projet de mathématiques
semestre d'hiver 2005-2006
Chaire : ROSE

Collecte d'objets par ordre décroissant de leur taille

Fanny Gilliéron

Professeur : D. de Werra
Assistante : T. Ekim

9 février 2006

Table des matières

Chapitre 1

Introduction au problème

1.1 Quelques éléments théoriques

Définition 1. *S est un **ensemble stable** de $G = (V, E)$ si $\forall x, y \in S$, $[x, y] \notin E$.*
*Par abus de langage, nous appellerons S un **stable**.*

Définition 2. *K est une **clique** de $G = (V, E)$ si $\forall x, y \in K$, $[x, y] \in E$.*

Définition 3. *Un **graphe scindé** (split-graph) est un graphe $G = (V, E)$ dont l'ensemble des sommets admet une partition (S, K), où S est un stable et K est une clique (voir figure 1.1).*

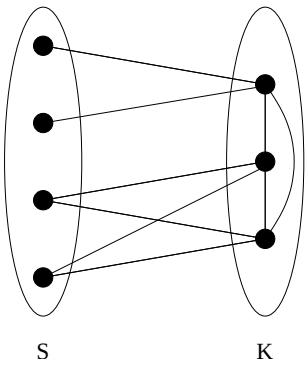


FIG. 1.1 – Graphe scindé

Définition 4. *Un graphe G est dit **(p,k)-colorable** si l'ensemble des sommets V peut être partitionné en p cliques et k stables.*

Notons $\chi_S(G)$ le nombre $\min_{(p,k)-coloration}(\max(p, k))$.

$\chi_S(G)$ *est appelé* ***nombre split-chromatique*** *car la recherche de ce nombre revient à partitionner l'ensemble des sommets en un nombre minimal de graphes scindés.*

Définition 5. *Un* ***graphe à seuil*** *est un graphe scindé dans lequel les voisinages des sommets du stable sont imbriqués (voir figure 1.2).*

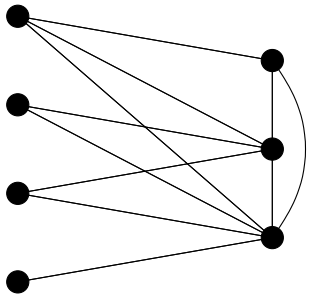


FIG. 1.2 – Graphe à seuil

Définition 6. *Soit* $\pi(N)$ *une permutation de* $N = \{1, 2, ..., n\}$. *Le* ***graphe de permutation associé à*** π *est un graphe dans lequel chaque nombre* $\{1, 2, ..., n\}$ *est représenté par un sommet, et dans lequel une arête relie* i *et* j *si* $i > j$ *et* $\pi(i) < \pi(j)$.

Exemple

Considérons la permutation 3 1 6 2 5 4 7. Le graphe associé à cette permutation est le suivant :

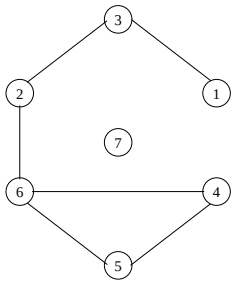


FIG. 1.3 – Graphe de permutation

1.2 Application au problème

1.2.1 Description du problème

Dans notre problème, nous considérons une permutation $\pi(N)$ de $N = \{1, 2, ..., n\}$.

Imaginons que notre permutation π représente une suite d'objets alignés dans un couloir, chaque objet ayant une taille inversément proportionnelle à la valeur de la permutation qui lui est associée.

Nous voulons alors collecter tous ces objets à l'aide de robots. Nous appellerons **robot-gauche** un robot partant de la gauche du couloir, et **robot-droite** un robot partant de la droite. Ces robots ne peuvent collecter les objets que durant un aller-retour, et doivent toujours les ramasser par ordre décroissant de taille, pour des raisons d'équilibre de la pile.

Nous nous intéressons au nombre de robots nécessaires à ramasser tous les objets. Nous noterons $\rho_g(\pi)$ le nombre minimum de robots-gauche nécessaires à collecter la permutation π, $\rho_d(\pi)$ le nombre minimum de robots-droite, et $\rho(\pi)$ le nombre minimum de robots si l'on autorise des robots partant des deux côtés.

Dans ce travail, nous nous pencherons sur les permutations dites **parfaites**, pour lesquelles $\rho(\pi') = \min(\rho_g(\pi'), \rho_d(\pi'))$, i.e. les permutations peuvent être ramassées par un nombre minimal de robots partant du même côté.

1.2.2 Interprétation en termes de graphes

Reprenons la permutation de l'exemple 1.1 :

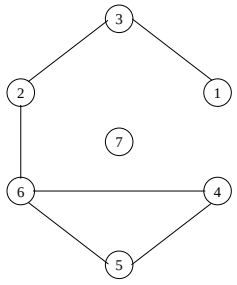


FIG. 1.4 – $\pi = (3\ 1\ 6\ 2\ 5\ 4\ 7)$

Dans un graphe de permutation, un stable (comme $\{1, 2, 4, 7\}$ dans notre exemple) représente une sous-permutation croissante dans π et une clique (comme $\{4, 5, 6\}$) représente une sous-permutation décroissante dans π.
Ces conditions sur les robots impliquent qu'un robot-gauche ramassera pendant son trajet un stable suivi d'une clique, et un robot-droite ramassera une clique suivie d'un stable.

Minimiser le nombre de robots nécessaires revient donc à minimiser le nombre de graphes scindés, sous la contrainte supplémentaire que dans le cas d'un robot-gauche (resp. droite), le maximum des sommets du stable (resp. de la clique) est inférieur au minimum des sommets de la clique (resp. du stable). Avec cette contrainte supplémentaire, nos graphes scindés sont en fait des graphes à seuils.

Ainsi, le problème de la recherche du nombre minimal de robots est en réalité un cas particulier de split-coloration ; on peut montrer que la split-coloration dans les graphes de permutations[1] ainsi que ce cas particulier[2] sont NP-difficiles[3].

1.2.3 Interprétation en termes de "shuffle-product"

Nous appellerons **voyage légal** l'ensemble des nombres ramassables par un robot durant un aller-retour. Nous avons vu précédemment qu'un voyage légal pour un robot-gauche est constitué d'un stable et d'une clique, avec $\max\{\textit{sommets du stable}\} < \min\{\textit{sommets de la clique}\}$, et qu'un voyage légal pour un robot-droite est constitué d'une clique et d'un stable, avec $\max\{\textit{sommets de la clique}\} < \min\{\textit{sommets du stable}\}$.

Appelons **produit du type** 1 un "shuffle-product"[4] de la forme

$$[1\ 2\ ...\ p] \sqcup\!\sqcup [n\ (n-1)\ ...\ (p+1)]$$

et **produit du type 2** un "shuffle-product" de la forme

$$[(p+1)\ ...\ n] \sqcup\!\sqcup [p\ ...\ 1]$$

avec $1 < p < n$.
Un voyage légal pour un robot-gauche s'écrit donc comme un produit du type 1, et un voyage légal pour un robot-droite comme un produit du type 2.

[1] Démonstration : (1)
[2] Démonstration : (2)
[3] Pour des détails sur la complexité et les problèmes NP-difficiles, se référer à (3)
[4] Le "shuffle-product" est défini dans (4)

Il s'agit alors de décomposer π en un nombre minimal de produits du type 1 et 2 pour trouver le nombre minimal de robots nécessaires à ramasser la permutation.

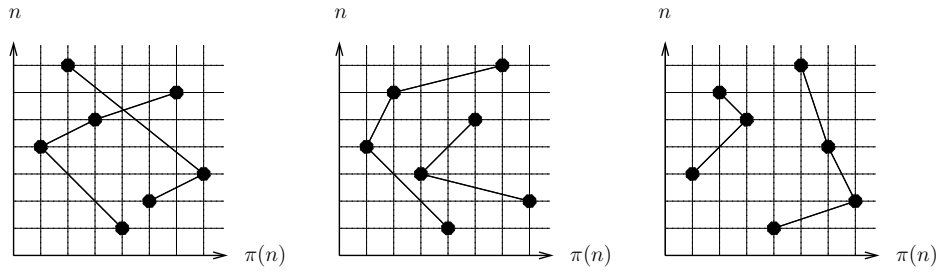
1.2.4 Notre objectif

Dans ce qui suit, nous appellerons **obstruction** une permutation π pour laquelle $\rho(\pi) < min(\rho_g(\pi), \rho_d(\pi))$, i.e. une permutation qui n'est pas parfaite.
Remarquons que $\forall \pi,\ \rho_(\pi) \leq \min(\rho_g(\pi), \rho_d(\pi))$; ainsi, toute permutation est soit parfaite, soit une obstruction.

Nous sommes intéressés à trouver une borne inférieure, et, dans le cas où elle existe, une borne supérieure pour la taille de telles obstructions dites minimales, c'est-à-dire pour lesquelles $\rho(\pi) < \min(\rho_g(\pi), \rho_d(\pi))$, et toutes les sous-permutations $\pi' \subset \pi$ sont parfaites, i.e. ont la propriété $\rho(\pi') = \min(\rho_g(\pi'), \rho_d(\pi'))$.

1.2.5 Illustrations

Dans ce travail, nous n'avons pas travaillé sur les graphes, mais directement sur les permutations. Une représentation efficace de la permutation et des trajets des robots est proposée dans la figure 1.5.



Permutation ramassable par 2 robots, l'un partant de gauche, l'autre de droite.

Permutation ramassable par 2 robots-droite

Permutation ramassable par 2 robots-gauche

FIG. 1.5 – Représentation d'une permutation

Chapitre 2

Etude du problème

2.1 Description des cas étudiés

Nous allons commencer par montrer dans le paragraphe 2.2 que toutes les permutations de taille 6 ou inférieure sont parfaites.

Ensuite, nous allons montrer que toutes les permutations de taille 7 sont ramassables par deux robots, puis vérifier grâce à un programme informatique qu'elles sont également parfaites (paragraphe 2.3).

Toujours à l'aide de programmes informatiques, nous allons ensuite étudier les permutations de taille 8 pour arriver à la conclusion qu'il existe des obstructions de taille 8, et donc que 8 est une borne inférieure pour la taille des obstructions minimales (paragraphe 2.4).

Nous allons encore étudier dans le paragraphe 2.5 les permutations de taille 9 par des méthodes similaires, et obtenir comme résultat qu'aucune obstruction de taille 9 n'est minimale.

Enfin, nous allons utiliser les résultats obtenus pour proposer des bornes sur $\rho(\pi)$, $\rho_g(\pi)$ et $\rho_d(\pi)$ et proposer comme déduction du résultat obtenu sur les permutations de taille 9 qu'il n'existe pas d'obstruction minimale de taille supérieure à 8 avec $\rho(\pi) = 2$ (conjecture 2.5.4 paragraphe 2.5.3).

2.2 Etude de la taille 6

Nous voulons trouver une borne inférieure pour la taille des obstructions minimales. Nous allons commencer par étudier les permutations de taille 6.

2.2.1 Résultats théoriques

Proposition 1. *Il n'existe pas d'obstruction de taille* 6.

Démonstration. Si π est une permutation de taille 3, alors $\rho(\pi) = 1$ $\Rightarrow \min\{\rho_g(\pi), \rho_d(\pi)\} = 1$.

En effet, si $\pi^{-1}(1) < \pi^{-1}(2)$, alors π est du type $[1\ 2] \sqcup [3]$ (type 1) ; c'est donc un voyage légal pour un robot-gauche $\Rightarrow\ \rho_g(\pi) = 1$.
Inversément, si $\pi^{-1}(1) > \pi^{-1}(2)$, alors π est du type $[3] \sqcup [2\ 1]$ (type 2) : c'est donc un voyage légal pour un robot-droite $\Rightarrow\ \rho_d(\pi) = 1$.

Conséquence :
Pour π une permutation de taille 6, $\rho(\pi) \leq 2$.

En effet, il suffit de décomposer π en deux permutations π' et π'' de taille 3. Ainsi, on obtient $\rho(\pi) \leq \rho(\pi') + \rho(\pi'') = 1 + 1$.

Remarque :
Si π est une permutation monotone de taille 3 , i.e. si on a $\pi^{-1}(1) < \pi^{-1}(2) < \pi^{-1}(3)$ ou $\pi^{-1}(3) < \pi^{-1}(2) < \pi^{-1}(1)$, alors π est du type 1 et 2.
En effet, on peut écrire π de la façon suivante :

$$[1\ 2\ 3] \sqcup [\ ] \Longleftrightarrow [\ ] \sqcup [1\ 2\ 3]$$

et de même si π est décroissante.

Nous remarquons au passage que cet argument reste valable pour une permutation monotone de taille quelconque.

Montrons maintenant que pour toute permutation π de taille 6,
$\min\{\rho_g(\pi), \rho_d(\pi)\} = \rho(\pi)$:

On a

$$\rho(\pi) \leq \min\{\rho_g(\pi), \rho_d(\pi)\} \leq \rho(\pi) + 1$$

La première inégalité est évidente, et la seconde découle du fait que

$$\underbrace{[1\ 2\ ...\ p] \sqcup [n\ (n-1)\ ...\ (p+1)]}_{type\ 1} = \underbrace{[\ ] \sqcup [n\ (n-1)\ ...\ (p+1)]}_{type\ 2} \sqcup \underbrace{[1\ 2\ ...\ p] \sqcup [\ ]}_{type\ 2}$$

$$\underbrace{[(p+1)\ ...\ n] \sqcup [p\ ...\ 1]}_{type\ 2} = \underbrace{[(p+1)\ ...\ n] \sqcup [\,]}_{type\ 1} \sqcup \underbrace{[\,] \sqcup [p\ ...\ 1]}_{type\ 1}$$

Si $\rho(\pi) = 1$, $\min\{\rho_g(\pi), \rho_d(\pi)\} = 1$, et le résultat est démontré.
Supposons que $\rho(\pi) = 2$.
Si on montre que $\exists \pi' \subset \pi$, avec π' de taille 3 t.q. π' est du type 1 et 2, on aura terminé.

Pour cela, on va montrer que si $\nexists \pi'$ comme ci-dessus contenant 1, alors $\exists \pi'$ ne contenant pas 1 :

On a l'un des deux cas suivants :

- Si $\pi^{-1}(1) \in \{1, 2, 3\}$, on a $\pi'^{-1}(4) > \pi'^{-1}(5) > \pi'^{-1}(6)$, sans quoi on obtiendrait une sous-permutation du type $\pi'^{-1}(1) < \pi'^{-1}(x) < \pi'^{-1}(y)$ avec $x, y \in \{4, 5, 6\}$ et $x < y$. Mais on a alors que $\pi'^{-1}(4) > \pi'^{-1}(5) > \pi'^{-1}(6)$, sous-permutation monotone.
- Si $\pi^{-1}(1) \in \{4, 5, 6\}$, on a $\pi'^{-1}(1) < \pi'^{-1}(2) > \pi'^{-1}(3)$, sans quoi on obtiendrait une sous-permutation du type $\pi'^{-1}(x) > \pi'^{-1}(y) > \pi'^{-1}(1)$ avec $x, y \in \{1, 2, 3\}$ et $x < y$. Mais on a alors que $\pi'^{-1}(1) < \pi'^{-1}(2) < \pi'^{-1}(3)$, sous-permutation monotone.

Ainsi, $\forall \pi$ permutation de taille 6, $\exists \pi'$, π'' des permutations de taille 3 t.q. $\pi = \pi' \sqcup \pi''$, avec π' du type 1 ou 2 et π'' du type 1 et 2.
$\Longrightarrow \min(\rho_g(\pi), \rho_d(\pi)) = 2$.

□

2.2.2 Remarque

Ce résultat nous assure que toute permutation de taille inférieure ou égale à 6 n'est pas une obstruction ; en effet, si $\pi' \subset \pi$, alors $\min(\rho_g(\pi'), \rho_d(\pi')) \leq \min(\rho_g(\pi), \rho_d(\pi))$, et comme les obstructions ne peuvent exister qu'à partir de $\min(\rho_g(\pi), \rho_d(\pi)) = 2$, il ne peut en exister de taille inférieure ou égale à 6.

Nous allons toutefois montrer un autre résultat qui sera utile par la suite :

Proposition 2. *$\forall \pi$ de taille 5, $\exists \pi' \subset \pi$ de taille 3 du type 1 et 2.*

Démonstration. Nous utilisons le même genre d'arguments que pour la preuve de la proposition précédente :

Supposons qu'$\nexists \pi' \subset \pi$ monotone de taille 3 contenant 1. Alors on est dans un des deux cas suivants :

- Si $\pi^{-1}(1) \in \{1,2,3\}$, on a $\pi^{-1}(4) > \pi^{-1}(5)$ par hypothèse. De plus, si $\pi^{-1}(4) < \pi^{-1}(1)$, $\pi^{-1}(2)$ ou $\pi^{-1}(3)$, on a formé une sous-suite décroissante de taille 3.
  Supposons que ce n'est pas le cas, et donc que $\pi^{-1}(4) > \pi^{-1}(1)$, $\pi^{-1}(2)$ et $\pi^{-1}(3)$; on a donc forcément que $\pi^{-1}(4) = 5$.
  De plus, par hypothèse, on sait que π ne commence pas par une suite décroissante $\Longrightarrow$ on a une suite croissante de taille 3 contenant le 4.

- Le cas où $\pi^{-1}(1) \in \{4,5\}$ se traite de façon symétrique.

□

2.3 Etude de la taille 7

Nous cherchons toujours à borner inférieurement la taille des obstructions minimales. Pour cela, nous devons savoir s'il existe des obstructions de taille 7.

2.3.1 Un premier résultat théorique

Avant de chercher des obstructions, il serait utile d'obtenir des informations sur $\rho(\pi)$; nous avons donc commencé par vérifier que toute permutation de taille 7 était ramassable par deux robots.

Proposition 3. *$\forall \pi$ permutation de taille 7, $\rho(\pi) \leq 2$. Autrement dit, toute permutation de taille 7 est ramassable par 2 robots.*

Démonstration. Dans la preuve de la Proposition 1, nous avons montré que toute permutation π de taille 6 pouvait être décomposée en deux permutations π' et π'' de taille 3, t.q. $\pi = \pi' \sqcup\!\sqcup \pi''$ avec π' de type 1 et 2, et π'' de type 1 ou 2. Soit π'_i le i^e élément de π'.

π' étant de type 1 et 2, elle est monotone, donc ramassable en un aller simple depuis la droite ou la gauche. Pour fixer les idées, supposons depuis la gauche. On peut donc écrire $\pi' \sqcup\!\sqcup [7]$ comme $[\pi'_1\ \pi'_2\ \pi'_3] \sqcup\!\sqcup [7]$ avec $\pi_1 < \pi_2 < \pi_3$, ce qui représente un voyage légal pour un robot-gauche.

Un raisonnement similaire s'applique si on suppose π' ramassable en un aller simple depuis la droite.

□

Il nous faut maintenant voir s'il existe des permutations de taille 7 pour lesquelles $\min(\rho_g(\pi), \rho_d(\pi)) = 3$, car dans ce cas, et avec le résultat démontré ci-dessus, nous aurions trouvé des obstructions.

Les arguments utilisés pour la taille 6 deviennent fastidieux à utiliser ici, car cela revient à une énumération d'un grand nombre de types de permutations. Nous allons donc créer un programme informatique qui testera, pour chaque permutation, si elle représente une obstruction.

2.3.2 Considérations préliminaires pour l'algorithme

Pour éviter d'utiliser un algorithme trop simpliste qui prendrait beaucoup de temps, nous essayons de trouver quelques critères qui nous assurent qu'une permutation est parfaite, sans pour autant rechercher tous les voyages légaux.

- Le premier argument utilisé est que si π contient une sous-permutation π' monotone de taille 4, alors $\min(\rho_g(\pi), \rho_d(\pi)) = 2$, car π' est du type 1 et 2, et peut donc être ramassée par un robot-gauche et un robot-droite, et on a vu précédemment[1] que la permutation de taille 3 qu'il reste est toujours ramassable en un trajet (utilisé en 2.(a)).

- Nous remarquons également que si la permutation est ramassable par deux robots-droite (resp. gauche), alors l'un des deux robots ramassera $\pi^{-1}(1)$ (resp. $\pi^{-1}(7)$). Ceci nous amène à utiliser la méthode d'énumération des pics :

  Considérons une permutation quelconque de taille 7 :

  Nous appelons **pic** l'élément marquant la transition entre l'aller et le retour d'un robot. La méthode utilisée est la suivante : Poser $(1, \pi^{-1}(1))$ comme **premier pic**, et choisir un **deuxième pic** (de la forme $(x, \pi^{-1}(x))$ avec $x > 1$) parmi les nombres restants. Vérifier que tous les nombres venant avant le deuxième pic peuvent être ramassés par le premier robot, puis essayer d'ajouter tous les nombres à l'un des deux robots (illustration dans la figure 2.1).

[1] preuve de la proposition 1

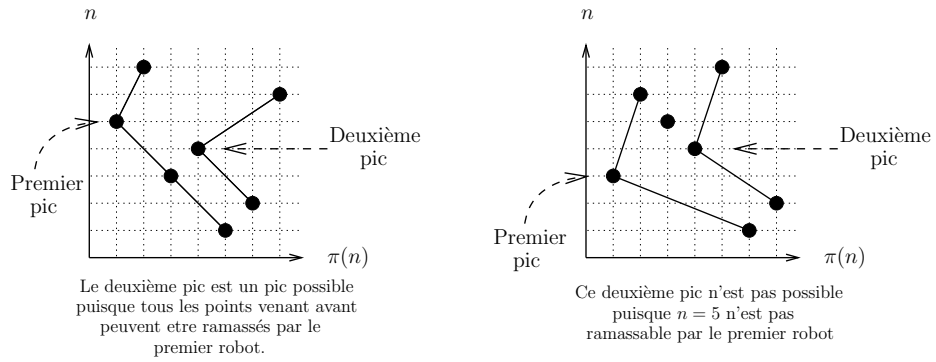


FIG. 2.1 – Méthode d'énumération des pics

2.3.3 Algorithme utilisé

L'algorithme que nous avons finalement implémenté[2] est le suivant :

1. Générer toutes les permutations de taille 7.
2. Pour chaque permutation :
   (a) Si la permutation ne contient pas de sous-permutation monotone de taille 4, garder la permutation.
3. Pour chaque permutation restante :
   (a) Pour chaque deuxième pic :
      i. Tester s'il est possible d'ajouter au trajet du premier robot tous les points précédant le deuxième pic.
      ii. Si c'est le cas, tester s'il est possible d'ajouter aux deux robots considérés tous les points restants.
      iii. Si c'est le cas, s'arrêter, la permutation est ramassable par 2 robots-droite, donc parfaite.
   (b) Si on n'a trouvé aucune manière de tout ramasser avec deux robots-droite, inverser la permutation et retourner au point 3.(a).
   (c) Si on ne trouve aucune manière de tout ramasser après inversion, stocker la permutation ; elle représente une obstruction.
4. Retourner les obstructions trouvées.

[2] fait dans algo7.cpp, disponible sur le CD.

2.3.4 Résultats

Après utilisation de cet algorithme, nous apprenons que toutes les permutations de taille 7 sont ramassables par deux robots-droite, ou deux robots-gauche (l'algorithme nous retourne "aucune obstruction trouvée"). Nous savons à présent que la borne minimale pour les obstructions est supérieure ou égale à 8.

2.4 Etude de la taille 8

Pour traiter le cas des permutations de taille 8, nous allons procéder en deux étapes : tout d'abord, nous allons vérifier que toutes les permutations de taille 8 sont ramassables par deux robots à l'aide d'un programme informatique ; ensuite nous allons rechercher des obstructions à l'aide d'un autre programme.

2.4.1 Quelques remarques utiles

Commençons tout d'abord par remarquer que pour toute permutation π de taille 8, on a $\rho(\pi) \leq 3$.

En effet, nous avons vu dans le paragraphe précédent que toutes les permutations de taille 7 étaient ramassables par deux robots. Ainsi, on voit aisément qu'un troisième robot suffirait à ramasser le 8, dans le cas où il n'aurait pas pu l'être par l'un des deux robots déjà utilisés. Ceci nous amène à créer un programme qui vérifierait pour toute permutation de taille 8 si elle est bien ramassable par deux robots.

Avant de donner l'algorithme que nous avons utilisé, il est utile de préciser quelques résultats employés :

Le nombre de permutations de taille 8 est très élevé ($8! = 40320$) ; il serait relativement long de toutes les générer, puis de toutes les tester. Nous avons donc choisi de générer les permutations de taille 7, puis d'éliminer toutes celles qui seraient trivialement ramassables par deux robots, une fois le 8^e élément ajouté.

- Si une permutation de taille 7 contient une sous-permutation monotone de taille 4, alors un aller simple sera suffisant pour ramasser cette sous-permutation, et une fois le 8^e élément ajouté, et quelle que soit sa place, il pourra être pris durant le retour de ce même robot.

- Si une permutation de taille 7 est ramassable par deux robots, et si l'un des deux robots ne prend que deux éléments, alors comme vu précédemment,

il pourra en ramasser un troisième (le 8^e) dans un même trajet, puisque toute permutation de taille 3 est ramassable en un trajet (cf preuve 1).

- Enfin, si une permutation de taille 7 est ramassable par deux robots, et si l'un des deux robots ramasse une sous-permutation monotone de taille 3, alors par le même argument que pour la sous-permutation de taille 4, il sera ramassable en un aller simple, et pourra donc prendre le 8^e élément au retour.

2.4.2 Algorithme utilisé pour $\rho(\pi)$

Voici l'algorithme[3] que nous avons utilisé pour vérifier que toute permutation π de taille 8 vérifie $\rho(\pi) = 2$:

1. Générer toutes les permutations de taille 7.
2. Pour chaque permutation :
   (a) Si la permutation ne contient pas de sous-permutation monotone de taille 4, garder la permutation.
3. Pour chaque permutation restante :
   (a) Tester si il est possible de la ramasser avec deux robots dont l'un ne ramasse que deux éléments.
   Si c'est le cas, passer à la permutation suivante.
   (b) Si ce n'est pas le cas, tester s'il est possible de la ramasser avec deux robots dont l'un ramasse une sous-permutation monotone de taille 3.
   Si c'est le cas, passer à la permutation suivante.
   (c) Si ce n'est pas le cas, inverser la permutation et retourner au point 3.(a).
   (d) Si ce n'est toujours pas le cas, stocker la permutation ; elle a un $\rho(\pi)$ encore inconnu.
4. Retourner les permutations avec $\rho(\pi)$ inconnu.

[3] fait dans algo8.cpp, disponible sur le CD.

2.4.3 Résultats

L'algorithme nous retourne zéro permutation avec $\rho(\pi)$ inconnu, ce qui nous assure que toutes les permutations de taille 8 sont ramassables par deux robots.

Il est nécessaires de préciser deux points :

1. Aux points 3.(a) et 3.(b), nous utilisons le même principe que dans l'algorithme 2.3.3, avec les tests sur chaque deuxième pic ; par souci de clarté, nous avons choisi de ne mentionner ici que les idées utilisées, et non les opérations exactes de l'algorithme[4].

2. L'algorithme nous retourne zéro permutation ; il faut tout de même préciser que s'il nous avait retourné des permutations, nous n'aurions pas pu en conclure qu'il existait des permutations de taille 8 qui n'étaient pas ramassables par deux robots. Il aurait fallu faire d'autres tests, car rien ne nous assure que les conditions testées par l'algorithme excluent tous les cas.

2.4.4 Quelques considérations pour la recherche d'obstructions

Ayant appris que toutes les permutations de taille 8 sont ramassables par deux robots, nous voulons maintenant tester s'il existe des permutations pour lesquelles $\min(\rho_g(\pi), \rho_d(\pi)) = 3$. Pour cela, nous avons écrit un programme similaire à l'algorithme 2.3.3, dans lequel nous avons utilisé les arguments suivants :

- Si la permutation de taille 8 contient une sous-permutation monotone de taille 5, elle sera ramassable par deux robots-gauche ou deux robots-droite, pour les mêmes raisons que celles développées pour la taille 7 (2.3.2).

- Pour tester si une permutation est ramassable par deux robots-droite, nous pouvons à nouveau utiliser la méthode d'énumération des pics (voir 2.3.2).

[4]Pour des détails sur l'implémentation, se référer au code source algo8.cpp sur le CD.

2.4.5 Algorithme utilisé pour la recherche d'obstructions

Voici l'algorithme de détection d'obstruction adapté pour la taille 8[5].

1. Générer toutes les permutations de taille 8.
2. Pour chaque permutation :
   (a) Si la permutation ne contient pas de sous-permutation monotone de taille 5, garder la permutation.
3. Pour chaque permutation restante :
   (a) Pour chaque deuxième pic :
      i. Tester s'il est possible d'ajouter au trajet du premier robot tous les points précédant le deuxième pic.
      ii. Si c'est le cas, tester s'il est possible d'ajouter aux deux robots considérés tous les points restants.
      iii. Si c'est le cas, s'arrêter, la permutation est ramassable par 2 robots-droite, donc parfaite.
   (b) Si rien n'a été trouvé, inverser la permutation et retourner au point 3.(a).
   (c) Si rien n'a été trouvé après inversion, stocker la permutation ; c'est une obstruction.
4. Retourner les obstructions trouvées.

2.4.6 Résultats

Si l'algorithme est similaire à celui utilisé pour l'étude de la taille 7, les résultats, eux, sont différents ; en effet, nous obtenons par cet algorithme que sur les 40320 permutations de taille 8, 1166 d'entre elles sont des obstructions.

A ce stade, nous avons donc obtenu notre borne inférieure pour la taille des obstructions minimales, puisque nous avons vérifié d'une part que toutes les permutations de taille inférieure à 8 étaient parfaites, et d'autre part qu'il existait des obstructions de taille 8.

[5] fait dans obstructions8.cpp, disponible sur le CD.

2.4.7 Un exemple d'obstruction

La permutation (1 8 4 3 6 2 7 5) est un exemple d'obstruction de taille 8. Dans la figure 2.2, nous donnons une façon de ramasser la permutation avec un robot-gauche et un robot-droite. Nous pouvons nous amuser à essayer de trouver une manière de tout ramasser par deux robots-gauche ou deux robots-droite ; nous remarquerons assez rapidement que ceci n'est bel et bien pas possible !

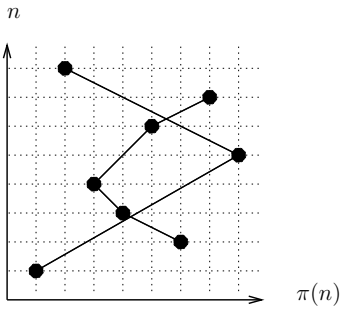


FIG. 2.2 – Obstruction de taille 8

2.5 Etude de la taille 9

Après avoir obtenu une borne inférieure, nous nous intéressons à une potentielle borne supérieure. Nous allons donc étudier les permutations de taille 9, pour voir s'il existe des obstructions minimales de taille 9, ou si toutes les obstructions contiennent en fait une sous-permutation de taille 8 qui fait partie des obstructions trouvées au paragraphe précédent.

2.5.1 Présentation des différentes étapes de l'algorithme

Pour la taille 9, nous avons choisi d'implémenter un seul algorithme qui chercherait les obstructions, puis testerait si elles sont minimales. Voici les arguments utilisés dans cet algorithme :

1. Vu le grand nombre de permutations de taille 9, nous commençons par éliminer certaines permutations de taille 8 avant de générer les permutations de taille 9 :

   - Si une permutation de taille 9 contient une sous-permutation de taille 8 ramassable par deux robots-droite (resp. gauche) dont l'un ne

travaille que sur un aller simple (donc ramasse une sous-permutation décroissante (resp.croissante)), nous pourrons attribuer à ce robot le 9^e élément lors de son retour. Ainsi, nous pouvons déjà supprimer toutes les permutations de taille 8 ayant cette propriété, puisque nous aurons forcément $\min(\rho_g(\pi), \rho_d(\pi)) \leq \rho_d(\pi) = 2$ et donc π sera parfaite (utilisé en 2.(a)).

- Si une permutation de taille 9 contient une obstruction de taille 8, alors dans le cas où la permutation serait une obstruction, elle ne serait pas minimale, et donc ne nous intéresserait pas. Nous pouvons donc encore supprimer toutes les obstructions de taille 8 (2.(b)).

Après ces deux élagages, nous avons déjà réduit le nombre de permutations de taille 8 de $8! = 40320$ à 13102.

2. Notons π_k pour une permutation de taille k.
Toujours sur des permutations de taille 8, nous essayons d'obtenir des informations sur $\rho(\pi_9)$, en utilisant les mêmes critères qu'en 2.4.1, adaptés pour la taille 8 :

   - Les permutations de taille 9 contenant une sous-permutation de taille 8 contenant elle-même une sous-permutation monotone de taille 5 vérifient $\rho(\pi) = 2$ (utilisé en 2.(c)).

   - Si une sous-permutation de taille 8 est ramassable par deux robots-droite (ou gauche) dont l'un ne ramasse que deux objets, nous aurons aussi que $\rho(\pi_9) = 2$ (2.(d).i).

   - Enfin, si une sous-permutation de taille 8 est ramassable par deux robots-droite (ou gauche) dont l'un ramasse une sous-permutation monotone de taille 3, $\rho(\pi_9) = 2$ (2.(d).ii).

   Après ces tests, on obtient les quantités suivantes : sur les 13102 permutations, 11316 entrent dans l'un des cas traités, et sont donc ramassables par deux robots, et 1786 ont un $\rho(\pi)$ inconnu qu'il faudra tester.

3. Nous allons générer les permutations de taille 9 issues des 1786 permutations avec $\rho(\pi)$ inconnu, et tester si elles sont ramassables par deux robots. Si c'est le cas, elles pourraient être une obstruction.

   - Pour générer les permutations de taille 9, il suffit d'inclure le 9 à chaque position possible dans la permutation (3.(a)).

- Pour tester si la permutation est ramassable par deux robots, nous pouvons à nouveau utiliser la méthode des pics, mais cette fois, comme nous ne connaissons pas le premier pic (dépendant du sens du robot), nous devons tester chaque paire de pic, et chaque sens pour chaque pic (c'est-à-dire si le robot sera un robot-gauche ou un robot-droite).

Sur les $1786 \cdot 9 = 16074$ permutations de taille 9 testées, 14547 sont ramassables en deux trajets, et donc des obstructions potentielles qu'il faut tester. En effet, pour π une permutation de taille 9, on a toujours $\min(\rho_g(\pi), \rho_d(\pi)) \leq 3$. Ce résultat sera expliqué au paragraphe 2.6.

4. Pour toutes les permutations satisfaisant $\rho(\pi) = 2$, nous devons tester si $\min(\rho_g(\pi), \rho_d(\pi)) = 2$. Pour ceci, nous utilisons la même méthode que dans l'algorithme 2.4.5 (utilisé en 5.(a)).

5. Enfin, pour toutes les obstructions trouvées, nous devons encore vérifier qu'elles ne contiennent pas de sous-permutation représentant déjà une obstruction. Nous avons déjà éliminé les potentielles obstructions ne contenant pas le 9, mais nous devons tester les autres. Ainsi, nous saurons s'il existe des obstructions minimales de taille 9 (5.(b).i).

2.5.2 Algorithme

Voilà donc l'algorithme implémenté[6] :

1. Générer toutes les permutations de taille 8.
2. Pour chaque permutation :
   (a) Tester si la permutation est ramassable par deux robots dont l'un n'effectue qu'un aller. Si ce n'est pas le cas, garder la permutation.
   (b) Tester si la permutation est une obstruction. Si ce n'est pas le cas, garder la permutation.
   (c) Tester si la permutation contient une sous-permutation monotone de taille 5. Si c'est le cas, stocker dans Liste 1, et passer au point 3.
   (d) Si ce n'est pas le cas :
      i. Tester s'il est possible de tout ramasser avec deux robots dont l'un ne ramasse qu'un ou deux objets. Si c'est le cas, stocker dans Liste 1, et passer au point 3.

[6] fait dans algo9.cpp, disponible sur le CD.

ii. Si ce n'est pas le cas, tester s'il est possible de tout ramasser avec deux robots dont l'un ramasse une sous-suite monotone de taille 3 ou 4. Si c'est le cas, stocker dans Liste 1, et passer au point 3.

iii. Si ce n'est pas le cas, stocker dans Liste 2.

3. Pour chaque permutation de la Liste 1 :
   (a) Générer les permutations de taille 9, et les stocker dans Liste 3.
4. Pour chaque permutation de la Liste 2 :
   (a) Générer les permutations de taille 9.
   (b) Pour chaque permutation générée :
      i. Tester si la permutation est ramassable par deux robots. Si c'est le cas, stocker dans Liste 3.
5. Pour chaque permutation de la Liste 3 :
   (a) Tester si elle est ramassable par deux robots-droite ou deux robots-gauche.
   (b) Si ce n'est pas le cas, c'est une obstruction. Pour toutes les sous-permutations de taille 8 de cette obstruction :
      i. Tester si c'est une obstruction. Si ce n'est pas le cas, stocker la permutation : c'est une obstruction minimale.
6. Retourner les obstructions minimales.

2.5.3 Résultats

Par cet algorithme, nous apprenons qu'aucune obstruction de taille 9 n'est minimale. Suite à cette information et à des essais sur les obstructions trouvées, nous proposons la conjecture suivante :

Conjecture 2.5.4. *Il n'existe pas d'obstruction minimale de taille supérieure à 8 satisfaisant* $\rho(\pi) = 2$.

Nous n'avons pas réussi à montrer ce résultat, mais nous avons développé des arguments valables dans certains cas particuliers qu'il faudrait généraliser.

Nous allons considérer que les permutations de taille N sont générées à partir de permutations de taille $N-1$ en ajoutant l'élément N à toutes les positions possibles. Nous appellerons par la suite π' la permutation de taille $N-1$ générant π.

Nous avons étudié les permutations satisfaisant la condition suivante :

Condition $(\star)$ **:** π est une obstruction de taille N avec $\rho(\pi) = 2$, et la permutation π' depuis laquelle π a été générée est ramassable par deux robots-gauche ou deux robots-droite dont l'un n'effectue qu'un retour. Cette condition est illustrée dans la figure 2.3.

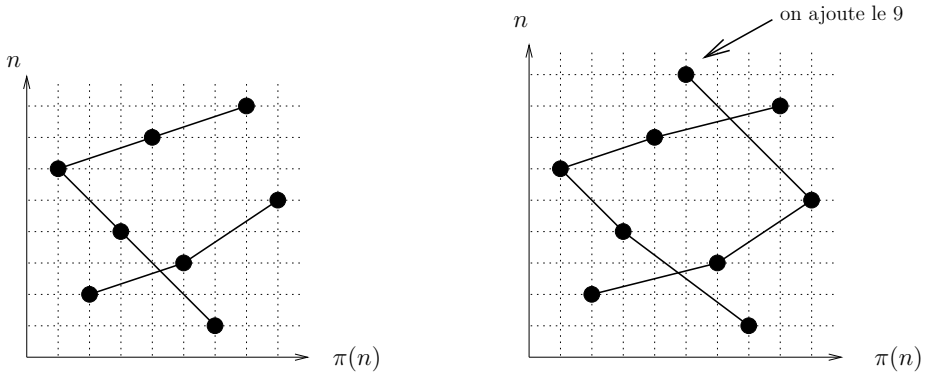


FIG. 2.3 – Permutation vérifiant la condition $(\star)$

Cette condition n'est pas vérifiée pour toutes les obstructions de taille N ; cependant, il est possible de montrer qu'elle est satisfaite pour toutes les obstructions π dont le pic du robot-gauche est supérieur au pic du robot-droite.

Pour fixer les idées, plaçons-nous dans le cas où π' est ramassable par deux robots-droite - le cas où π' est ramassable par deux robots-gauche se traitant de manière symétrique. π est une obstruction, donc ramassable avec un robot-gauche et un robot-droite ; de plus, nous savons que l'élément N est ramassé par le robot-gauche, sans quoi π serait ramassable par deux robots-droite, et ne serait donc pas une obstruction.

Appelons x le pic du robot-gauche, et y le pic du robot-droite considérés lors du ramassage de π. Considérons encore x_p le premier point ramassé par le robot-gauche, y_p le premier point ramassé par le robot-droite et y_d le dernier point ramassé par le robot-droite (le robot-gauche ramassant N en dernier). La figure 2.4 illustre cette notation et définit 6 zones dans le graphe, que nous utiliserons plus tard.

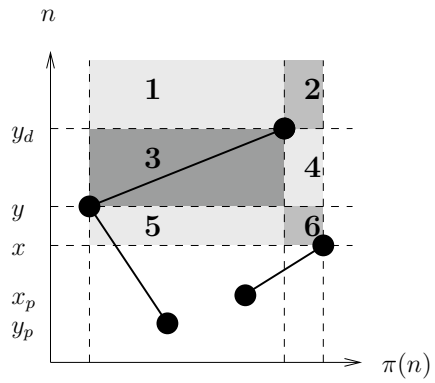


FIG. 2.4 – Définition des zones

Il y a deux cas possibles : ou le robot-gauche ne ramasse durant son retour que l'élément N, ou il ramasse au moins un autre élément, appelé k par la suite.

Dans le cas où le robot-gauche ne ramasse durant son retour que l'élément N, on voit aisément que π satisfait la condition $(\star)$.

Supposons maintenant que, quel que soit les voyages légaux considérés, le robot-gauche ramasse durant son retour un autre élément que l'élément N, appelé k, i.e. que π ne peut pas satisfaire la condition $(\star)$. k se trouve dans l'une des 6 zones proposées dans la figure 2.4, car il est par hypothèse en haut à gauche de x, et on peut supposer qu'il se trouve à droite de y, puisque π' est ramassable par deux robots-droite, et donc l'un des deux robots a pour pic le point le plus à gauche. Nous pouvons ainsi raisonnablement supposer que le robot-droite de π' restant un robot-droite pour π est celui le plus à gauche.

Nous allons montrer que la condition $(\star)$ est satisfaite si $x > y$, c'est-à-dire si les zones 5 et 6 n'existent pas. Dans le cas contraire, cette condition n'est pas toujours vérifiée, et il faudrait employer d'autres méthodes (ou d'autres arguments similaires adaptés) pour prouver la conjecture.

Pour montrer que la condition $(\star)$ est satisfaite pour $x > y$, nous allons montrer qu'une telle permutation contient toujours ce que nous appellerons un **triangle problématique** : un triple (x, y, z) tel que $x < y < z$ et $\pi(y) > \{\pi(x), \pi(z)\}$, et n'appartenant pas entièrement au trajet d'un des deux robots-droite nécessaires à ramasser la permutation π' (figure 2.5).

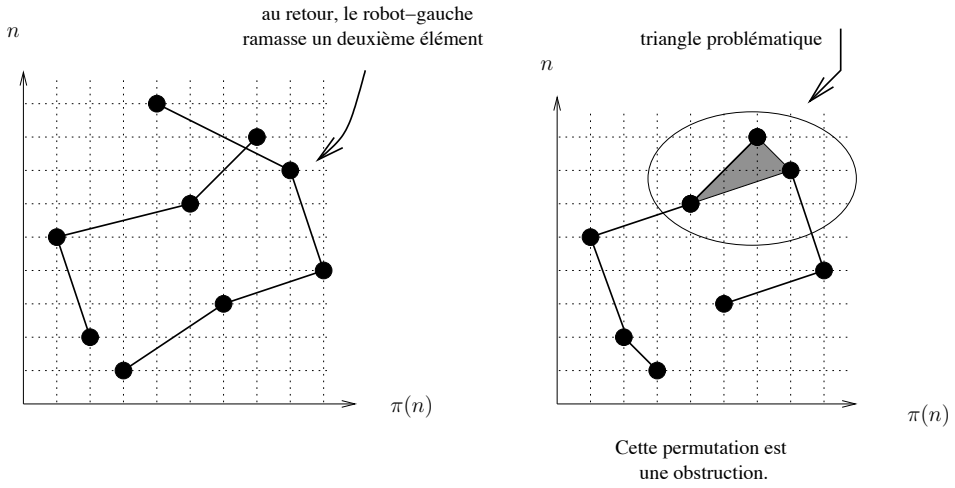


FIG. 2.5 – Triangle problématique

Remarquons que nous avons représenté les pics comme étant les valeurs extrêmes de la permutation, ce qui n'est pas forcément le cas ; mais dans la suite, nous ne considérerons que la partie de la permutation délimitée par ces pics.

Triangles problématiques et obstruction :

Nous allons à présent étudier chaque zone séparément et montrer que, soit k fait partie d'un triangle problématique, soit k peut être intégré au trajet du robot-droite, et donc π' satisfait la condition $(\star)$. Nous avons étudié toutes les zones, même si par la suite nous nous limiterons au cas où $x > y$, car ces triangles problématiques pourraient peut être servir pour démontrer la conjecture dans les cas manquants.

Zone 1 **:**
Si k se trouve dans la zone 1, alors le triple (y, y_d, k) est un triangle problématique.

Zone 2 **:**
Si k se trouve dans la zone 2, il peut être ajouté à la fin du trajet du robot-droite.

Zone 3 **:**
Si k se trouve dans la zone 3, il y a deux cas à considérer :

1. Il existe $x < y$ dans le retour du robot-droite avec $\pi(x) < \pi(k) < \pi(y)$ et $k > y$. Alors (x, y, k) est un triangle problématique.
2. Il n'existe pas de tels x et y dans le retour du robot-droite, donc il existe $x < y$ avec $\pi(x) < \pi(k) < \pi(y)$ et $x < k < y$; ainsi k peut être

intégré au retour du robot-droite.

Zone 4 **:** Si k se trouve dans la zone 4, (y, k, y_d) est un triangle problématique.

Zone 5 **:** Si k se trouve dans la zone 5, il y a à nouveau deux cas à considérer :

1. Il existe $x < y$ dans l'aller du robot-droite avec $x < k < y$ et $\pi(k) > \pi(x)$. Alors (x, k, y) est un triangle problématique.
2. Il n'existe pas de tels x et y dans l'aller du robot-droite, donc k peut être intégré à l'aller du robot-droite, soit avant le premier, soit au milieu s'il existe $x < y$ avec $x < k < y$ et $\pi(x) > \pi(k) > \pi(y)$.

Zone 6 **:** Enfin, si k se trouve dans la zone 6, (y_p, k, y) est un triangle problématique.

Nous allons maintenant montrer que si $x > y$ et k fait partie d'un triangle problématique, i.e. si π' ne satisfait pas la condition $(\star)$, alors π' est une obstruction. Ceci nous permettra de conclure, toujours sous notre hypothèse, que si π est une obstruction minimale, alors π' est ramassable par deux robots-gauche ou deux robots-droite dont l'un ne fait qu'un retour.

Supposons que k fait partie d'un triangle problématique, et que π est une obstruction. Appelons D_1 et D_2 deux robots-droite qui doivent ramasser π', D et G le robot-droite et le robot-gauche qui ramassent π selon notre hypothèse. Nous allons montrer que D_1 et D_2 ne peuvent pas ramasser π'.

Soit (x, y, z) un triangle problématique ; les points x, y et z ne peuvent pas tous être ramassés par un seul robot-droite. Ainsi, D_1 et D_2 ramasseront chacun au moins l'un de ces trois points.

L'un des deux robots, disons D_1, doit ramasser x ; il ne pourra donc ramasser aucun des éléments faisant partie de l'aller de G. Il faut donc que tous les points inférieurs à x forment un voyage légal pour D_2.

Ceci implique que D peut ramasser tous les éléments de l'aller de G, et donc que G peut être vu comme un robot-droite, i.e. π n'est pas une obstruction.

Ainsi, si π est une obstruction, alors π' satisfait la condition $(\star)$. Remarquons que ce résultat n'est plus valable si $y > x$: un contre-exemple est donné dans la figure 2.6.

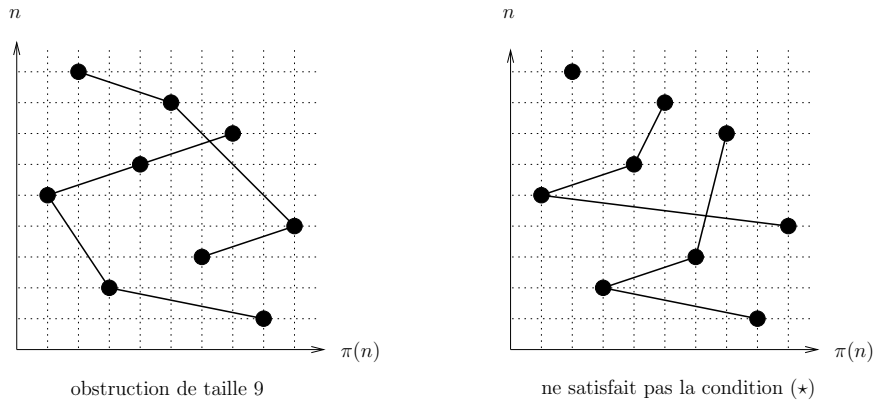


FIG. 2.6 – contre-exemple : $\pi = (5\ 9\ 2\ 6\ 8\ 3\ 7\ 1\ 4)$

Reprenons maintenant l'algorithme utilisé pour tester s'il existe des obstructions minimales de taille 9 (2.5.2) : nous avons considéré toutes les permutations parfaites de taille 8, donc en particulier toutes les permutations de taille 8 satisfaisant la condition $(\star)$. Ensuite, nous avons généré les permutations de taille 9 ; nous avons donc considéré en particulier toutes les potentielles obstructions de taille 9.

Nous pouvons maintenant nous convaincre que toute permutation de taille $N-1$ satisfaisant la condition $(\star)$ contient une sous-permutation de taille 8 satisfaisant également cette condition. Or, nous avons appris par l'algorithme 2.5.2 que toutes les obstructions de taille 9 contenaient une obstruction de taille 8 ; ceci implique que toute obstruction de taille N avec $\rho(\pi) = 2$, donc entrant dans le schéma étudié, contient également, dans le cas où $x > y$, une obstruction de taille 8, et n'est donc pas minimale. Il suffit de considérer la sous-permutation de taille 8 satisfaisant la condition $(\star)$ à laquelle on ajoute le N^e élément ; nous avons alors une permutation de taille 9 qui, d'après l'algorithme, contient une obstruction.

Ces arguments ne constituent évidemment pas une preuve complète ; nous avons fait des hypothèses qu'il faudrait relâcher par la suite pour terminer la démonstration. Toutefois, ces résultats fonctionnent dans un grand nombre de cas, ce qui nous pousse à croire que cette conjecture est vraie.

2.6 A propos des bornes supérieures

Hormis ce résultat, nous pouvons déduire de l'étude des tailles inférieures à 9 quelques résultats à propos des valeurs de $\rho(\pi)$ et de $\min(\rho_g(\pi), \rho_d(\pi))$:

- Nous avons vu dans la preuve de la proposition 1 que toute permutation de taille 3 était ramassable par un robot. Nous pouvons donc écrire, pour $\pi(N)$:

  $$\rho(\pi) \leq \left\lceil \frac{N}{3} \right\rceil$$

  Cette borne n'est pas d'un très grand intérêt, mais l'idée peut s'appliquer aux autres résultats obtenus.

- Nous avons vu dans le paragraphe 2 que dans toute permutation de taille 5, il y avait une sous-permutation monotone de taille 3, donc ramassable par un robot-gauche et un robot-droite. Ceci nous amène à l'inégalité suivante :

  $$\min(\rho_g(\pi), \rho_d(\pi)) \leq \left\lceil \frac{N}{3} \right\rceil$$

  En effet, tant qu'il reste plus de 4 éléments, on peut toujours trouver une sous-permutation de taille 3 ramassable par un robot-gauche ou un robot-droite ; ainsi, dans le cas où N est multiple de 3, on peut réduire la permutation en enlevant des sous-permutations monotone de taille 3, et il ne restera à la fin qu'une permutation de taille 3 qui déterminera le sens des robots. Dans le cas où N n'est pas multiple de 3, on a clairement

  $$\min(\rho_g(\pi(3k+1), \rho_d(\pi(3k+1))) \leq \min(\rho_g(\pi(3(k+1)), \rho_d(\pi(3(k+1)))))$$

  ou

  $$\min(\rho_g(\pi(3k+2), \rho_d(\pi(3k+2))) \leq \min(\rho_g(\pi(3(k+1)), \rho_d(\pi(3(k+1)))))$$

  ce qui montre l'inégalité.

- Appliquons le même raisonnement au résultat obtenu pour les permutations de taille 7 (paragraphe 2.3). Comme $\min(\rho_g(\pi), \rho_d(\pi)) \leq 2$ pour toute permutation de taille 7, nous obtenons, pour toute permutation π de taille N :

  $$\min(\rho_g(\pi), \rho_d(\pi)) \leq 2 \left\lceil \frac{N}{7} \right\rceil$$

- D'une manière similaire, nous pouvons déduire du paragraphe 2.4 la borne suivante :

$$\rho(\pi) \leq 2 \left\lceil \frac{N}{8} \right\rceil$$

2.7 Améliorations et extensions

Les résultats obtenus jusqu'à présent nous poussent à nous poser plusieurs questions qu'il serait intéressant d'étudier dans une extension de ce travail :

- Les bornes proposées au paragraphe (2.6) peuvent en fait être améliorées au fur et à mesure de l'obtention de résultats sur des permutations de plus grandes tailles ; cependant, elles ne répondent pas à la question principale que nous nous posons, à savoir est-il possible de borner la taille des obstructions minimales ; en effet, elles ne bornent que le nombre de robots, mais ne nous donnent rien sur les sous-permutations, donc sur la minimalité d'une obstruction. Existe-t-il d'autres sortes de bornes ?

- La conjecture 2.5.4 nous donne un résultat sur les obstructions minimales ; il faudrait évidemment achever la démonstration. Le cas échéant, il serait bon de savoir si elle peut être généralisée à des obstructions pour lesquelles $\rho(\pi)$ est supérieur à 2. Nous pourrions alors penser que la taille des obstructions minimales peut augmenter à l'infini, mais que lorsque ρ est fixé, il existe toujours une seule taille pour de telles obstructions. il faudrait encore savoir si ρ peut effectivement augmenter autant que nous le souhaitons.

- A cela s'ajouterait ensuite le problème de savoir s'il peut exister des obstructions avec $\rho(\pi)$ aussi grand que l'on veut ; dans le cas où $\rho(\pi)$ prend une valeur quelconque, et en supposant que la proposition 2.5.4 puisse être généralisée, comment déterminer la taille de telles obstructions ?

- Enfin, une question que nous n'avons pas du tout abordée est de savoir s'il existe des permutations π avec $\min(\rho_g(\pi), \rho_d(\pi)) - \rho(\pi) \geq 2$; en effet, dans les cas que nous avons étudié, nous avions toujours $\rho(\pi) = 2$ si π était une obstruction, et nous avions une taille suffisamment petite pour toujours avoir $\min(\rho_g(\pi), \rho_d(\pi)) \leq 3$. Mais lorsque la taille des permutations augmente, il n'est a priori pas exclu que la différence soit plus grande.

Conclusion

Nous avions pour but d'étudier les permutations parfaites et les obstructions de taille N et d'essayer de borner la taille des obstructions minimales. Nous sommes parvenus, grâce à des résultats théoriques, mais également à l'aide de moyens informatiques, à vérifier qu'il n'existe pas d'obstruction de taille inférieure à 8, ce qui résoud déjà le problème de la borne inférieure.

Nous avons, de plus, étudié la taille 9, et vérifié qu'il n'existe pas d'obstruction minimale de taille 9 ; ce résultat nous a permi d'aboutir sur une conjecture plus générale, à savoir qu'il n'existe pas d'obstruction minimale de taille supérieure à 8 ramassable par deux robots. Nous avons également développé certains arguments théoriques qui pourraient entrer dans la démonstration.

Ces résultats sont certes intéressants, mais nous ouvrent également la porte sur de nouvelles questions à propos de ces permutations : peut-on généraliser nos résultats ? Existe-t-il une borne supérieure ? Est-il possible que le nombre minimal de robot nécessaire à tout ramasser diffère de plus de 1 selon que l'on autorise, ou non, des départs des deux côtés ?

Autant de questions qui mériteraient d'être approfondies...

Bibliographie

[1] T. Ekim et D. de Werra (2005), *J. of Combinatorial Optimization*. v.10, p.211–225.

[2] G. Di Stefano, S. Krause, M. E. Luebbecke et U. T. Zimmermann (October 2005), *On Minimum k-Modal Partitions of Permutations* (soumis).

[3] M. R. Garey et D. S. Johnson (1979), *Computers and Intractability - A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company. San Francisco.

[4] M. C. Golumbic (1980), *Algorithmic graph theory and perfect graphs*. Computer Science and Applied Mathematics. Academic Press.