Rapport du projet de programmation

Quentin MALLEGOL et Kwame MBOBDA-KUATE

10 mars 2024

1 Introduction

Ce rapport décrit les choix algorithmiques pris et les résultats obtenus dans le cadre du projet de programmation 2024. Il présente d'abord les heuristiques et solveurs employés puis les analyse et les compare.

2 Définition

Dans toute la suite du rapport, n et m désignent deux entiers naturels non nuls et une grille est un élément de l'ensemble G_{mn} des applications bijectives de $[\![0,\, mn-1]\!]$ dans lui-même. Soit $S^h_{m,n} = \{(i \quad i+1) \mid i \in [\![0,\, mn-1]\!],\ i \,\%\, n \neq n-1\}$ l'ensemble des échanges horizontaux possibles et $S^v_{m,n} = \{(i \quad i+n) \mid i \in [\![0,\, mn-1-n]\!]\}$ l'ensemble des échanges verticaux possibles puis $S_{m,n} = S^h_{m,n} \cup S^v_{m,n}$. Le problème du *swap puzzle* consiste à, étant donnée une grille $\sigma \in G_{mn}$ à trouver une décomposition de σ^{-1} en un nombre minimal d'éléments de $S_{m,n}$.

3 Heuristiques

3.1 Heuristiques basiques

Nous aborderons dans un premier temps les heuristiques « naïves », c'est-à-dire celles qui ne demandent pas de précalcul.

3.1.1 Distance de Manhattan

Definition 3.1. La distance de Manhattan est la fonction MD qui à un couple de grilles $(\sigma, \tau) \in G^2_{mn}$ associe

$$\mathrm{MD}(\sigma,\tau) = \sum_{k=0}^{mn-1} \left(\left| \left\lfloor \frac{\tau^{-1}(\sigma(k))}{n} \right\rfloor - \left\lfloor \frac{k}{n} \right\rfloor \right| + \left| (\tau^{-1}(\sigma(k)) \,\%\, n) - (k \,\%\, n) \right| \right)$$

.

Lemme 3.1. *La distance de Manhattan par rapport à une grille varie au plus de deux au cours d'un échange :*

$$\forall(\sigma,\tau)\in G_{mn}^2, \forall s\in S_{m,n}, |MD(\sigma,\tau)-MD(\sigma\circ s,\tau)|\leq 2$$

Démonstration. Soient $(\sigma,\tau)\in G_{mn}^2$ et $s\in S_{m,n}$. Il existe des entiers i et j avec $i<j$ tels que $s=(i\quad j)$.

$$\begin{aligned}\mathrm{MD}(\sigma,\tau)-\mathrm{MD}(\sigma\circ s,\tau) &= \left|\left\lfloor\frac{\tau^{-1}(\sigma(i))}{n}\right\rfloor-\left\lfloor\frac{j}{n}\right\rfloor\right|+\left|(\tau^{-1}(\sigma(i))\,\%\,n)-(j\,\%\,n)\right|\\ &+\left|\left\lfloor\frac{\tau^{-1}(\sigma(j))}{n}\right\rfloor-\left\lfloor\frac{i}{n}\right\rfloor\right|+\left|(\tau^{-1}(\sigma(j))\,\%\,n)-(i\,\%\,n)\right|\\ &-\left|\left\lfloor\frac{\tau^{-1}(\sigma(i))}{n}\right\rfloor-\left\lfloor\frac{i}{n}\right\rfloor\right|+\left|(\tau^{-1}(\sigma(i))\,\%\,n)-(i\,\%\,n)\right|\\ &-\left|\left\lfloor\frac{\tau^{-1}(\sigma(j))}{n}\right\rfloor-\left\lfloor\frac{j}{n}\right\rfloor\right|+\left|(\tau^{-1}(\sigma(j))\,\%\,n)-(j\,\%\,n)\right|\end{aligned}$$

Si $s\in S_{m,n}^h$, $j=i+1$ et on a $\lfloor\frac{j}{n}\rfloor=\lfloor\frac{i}{n}\rfloor$ donc

$$\begin{aligned}\mathrm{MD}(\sigma,\tau)-\mathrm{MD}(\sigma\circ s,\tau) &= \left|(\tau^{-1}(\sigma(i))\,\%\,n)-(i\,\%\,n)-1\right|-\left|(\tau^{-1}(\sigma(i))\,\%\,n)-(i\,\%\,n)\right|\\ &+\left|(\tau^{-1}(\sigma(i+1))\,\%\,n)-(i\,\%\,n)\right|-\left|(\tau^{-1}(\sigma(i+1))\,\%\,n)-(i\,\%\,n)-1\right|\end{aligned}$$

D'après l'inégalité triangulaire,

$$\begin{aligned}|\mathrm{MD}(\sigma,\tau)-\mathrm{MD}(\sigma\circ s,\tau)| &\leq \left|\left|(\tau^{-1}(\sigma(i))\,\%\,n)-(i\,\%\,n)-1\right|-\left|(\tau^{-1}(\sigma(i))\,\%\,n)-(i\,\%\,n)\right|\right|\\ &+\left|\left|(\tau^{-1}(\sigma(i+1))\,\%\,n)-(i\,\%\,n)\right|-\left|(\tau^{-1}(\sigma(i+1))\,\%\,n)-(i\,\%\,n)-1\right|\right|\\ &\leq \left|(\tau^{-1}(\sigma(i))\,\%\,n)-(i\,\%\,n)-1-(\tau^{-1}(\sigma(i))\,\%\,n)+(i\,\%\,n)\right|\\ &+\left|(\tau^{-1}(\sigma(i+1))\,\%\,n)-(i\,\%\,n)-(\tau^{-1}(\sigma(i+1))\,\%\,n)+(i\,\%\,n)-1\right|\\ &=2.\end{aligned}$$

Le cas où $s\in S_{m,n}^v$ se traite de manière analogue en remarquant qu'on a alors $j=i+n$ et on a $(i\,\%n)=(j\,\%n)$. □

Théorème 3.2. *Soit $\tau\in G_{m,n}$. Pour la recherche du plus court chemin jusqu'à τ, la fonction $\tau\in G_{m,n}\to\dfrac{MD(\sigma,\tau)}{2}$ est monotone (donc admissible) et $\tau\in G_{m,n}\to\left\lceil\dfrac{MD(\sigma,\tau)}{2}\right\rceil$ est admissible.*

Démonstration. La monotonie découle directement du lemme précédent et du fait que $\mathrm{MD}(\tau,\tau)=0$. Le coût d'un chemin étant toujours entier, $\tau\in G_{m,n}\to\left\lceil\dfrac{\mathrm{MD}(\sigma,\tau)}{2}\right\rceil$ est admissible. □

Conjecture 3.3. *Soit σ la permutation de $[\![0,\,mn-1]\!]$ définie par $\forall k\in[\![0,\,mn-1]\!], \sigma(k)=mn-1-k$. Alors $\forall\tau\in G_{mn}, MD(\tau,id)\leq MD(\sigma,id)$.*

Remarque. Le résultat a été vérifié numériquement pour $n\leq 3$ et $m\leq 3$ et est utilisé pour générer des grilles de difficulté contrôlée.

3.2 Distance d'inversion

Definition 3.2. Soit $r \in G_{mn}$ la réflexion par rapport à la diagonale principale définie par $\forall (i,j) \in [\![0,\, m-1]\!] \times [\![0,\, n-1]\!]$, $r(ni+j) = mi+j$. Soit $\sigma \in G_{mn}$. On appelle inversion tout couple $(i,j) \in [\![0,\, mn-1]\!]^2$ tel que $i \leq j$ et $\sigma(i) \geq \sigma(j)$. On note $\mathrm{N}_h(\sigma)$ le nombre d'inversions de σ et $\mathrm{N}_v(\sigma)$ celui de $r \circ \sigma \circ r$.

Lemme 3.4. *Effectuer un échange crée ou résout au plus* $2n-1$ *inversions horizontales et* $2m-1$ *inversions verticales :*

$$\forall \sigma \in G_{mn}, \forall s \in S^h_{m,n} | N_h(\sigma \circ s) - N_h(\sigma)| \leq 1$$

$$\forall \sigma \in G_{mn}, \forall s \in S^v_{m,n} | N_v(\sigma \circ s) - N_v(\sigma)| \leq 1$$

$$\forall \sigma \in G_{mn}, \forall s \in S^v_{m,n} | N_h(\sigma \circ s) - N_h(\sigma)| \leq 2n-1$$

$$\forall \sigma \in G_{mn}, \forall s \in S^h_{m,n} | N_v(\sigma \circ s) - N_v(\sigma)| \leq 2m-1$$

.

Démonstration. Par symétrie, nous ne traiterons que le cas horizontal. Soit $\sigma \in G_{mn}$ et $\forall s \in S_{m,n}$. Nous utilisons le résultat suivant : le nombre d'inversions de σ est l'entier minimal k tel qu'il existe une décomposition de σ en k transpositions de la forme $(j \quad j+1)$. Si $s \in S^h_{m,n}$ alors il existe un entier i tel que $s = (i \quad i+1)$ donc $N_h(\sigma \circ s) \leq N_h(\sigma)+1$. Sinon, il existe un entier i tel que $s = (i \quad i+n)$. Or, $(i \quad i+n) = (i \quad i+1)(i+1 \quad i+2)...(i+n-2 \quad i+n-1)(i+n-1 \quad i+n)(i+n-2 \quad i+n-1)...(i+1 \quad i)$ et comporte $2n-1$ termes donc $N_h(\sigma \circ s) \leq N_h(\sigma)+2n-1$. Le même raisonnement appliqué à $\sigma \circ s$ donne $N_h(\sigma) \leq N_h(\sigma \circ s) + 2n - 1$ si $s \in S^v_{m,n}$ et $N_h(\sigma) \leq N_h(\sigma \circ s) + 1$ $s \in S^h_{m,n}$ (puisque $s^2 = \mathrm{id}$) ; d'où le résultat. □

Théorème 3.5. *Pour la recherche du plus court chemin jusqu'à id, la fonction* $\dfrac{N_h + N_v}{2\max(n,m)}$ *est monotone (donc admissible) et* $\left\lceil \dfrac{N_h + N_v}{2\max(n,m)} \right\rceil$ *est admissible.*

Démonstration. $\mathrm{N}_h(\mathrm{id}) = \mathrm{N}_v(\mathrm{id}) = 0$ et, pour $\sigma \in G_{mn}$

$$\begin{aligned}\forall s \in S^h_{m,n}, |\mathrm{N}_h(\sigma \circ s) - \mathrm{N}_h(\sigma) + \mathrm{N}_v(\sigma \circ s) - \mathrm{N}_v(\sigma)| &\leq |\mathrm{N}_h(\sigma \circ s) - \mathrm{N}_h(\sigma)| + |\mathrm{N}_v(\sigma \circ s) - \mathrm{N}_v(\sigma)| \\ &\leq 2n - 1 + 1 \leq 2\max(n,m)\end{aligned}$$

et aussi

$$\begin{aligned}\forall s \in S^v_{m,n}, |\mathrm{N}_h(\sigma \circ s) - \mathrm{N}_h(\sigma) + \mathrm{N}_v(\sigma \circ s) - \mathrm{N}_v(\sigma)| &\leq |\mathrm{N}_h(\sigma \circ s) - \mathrm{N}_h(\sigma)| + |\mathrm{N}_v(\sigma \circ s) - \mathrm{N}_v(\sigma)| \\ &\leq 1 + 2m - 1 \leq 2\max(n,m)\end{aligned}$$

donc $\dfrac{\mathrm{N}_h + \mathrm{N}_v}{2\max(n,m)}$ est monotone. La coût d'un chemin étant toujours entier, $\left\lceil \dfrac{\mathrm{N}_h + \mathrm{N}_v}{2\max(n,m)} \right\rceil$ est admissible. □

3.3 Heuristiques précalculées

Les heuristiques présentées ici sont tirées de travaux sur le taquin et fonctionnent en résolvant des problèmes simplifiés puis en stockant le nombre d'échanges nécessaire.

3.3.1 Walking Distance

Cette heuristique a été développée par Ken'ichiro Takahashi (takaken). Définissons les relations d'équivalence $\sim_h$ et $\sim_v$ sur G_{mn} par $\forall(\sigma,\tau) \in G_{m,n}^2$

$$\sigma \sim_h \tau \iff \exists s \in S_{n,m}^h, \sigma = \tau \circ s$$
$$\sigma \sim_v \tau \iff \exists s \in S_{n,m}^v, \sigma = \tau \circ s$$

Considérons les ensembles quotients $G_{m,n}/\sim_h$ et $G_{m,n}/\sim_v$ puis

$$A_{m,n}^h = \bigcup_{(\sigma,s)\in G_{mn}\times S_{m,n}} \{([\sigma]_h, [\sigma \circ s]_h\}$$

et

$$A_{m,n}^v = \bigcup_{(\sigma,s)\in G_{mn}\times S_{m,n}} \{([\sigma]_v, [\sigma \circ s]_v\}$$

où pour tout $\sigma \in G_{mn}$, $[\sigma]_h$ désigne un représentant de σ pour $\sim_h$. Enfin, pour toute permutation $\sigma \in G_{mn}$, soient $\mathrm{WD}_h(\sigma)$ la longueur du plus court chemin de $[\sigma]_h$ à id dans le graphe $(G_{m,n}/\sim_h, A_{m,n}^h)$ et $\mathrm{WD}_v(\sigma)$ la longueur du plus court chemin de $[\sigma]_v$ à id dans le graphe $(G_{m,n}/\sim_v, A_{m,n}^v)$. Enfin, l'heuristique *walking distance* est alors définie par $\forall\sigma \in G_{mn}, \mathrm{WD}(\sigma) = \mathrm{WD}_v(\sigma) + \mathrm{WD}_v(\sigma)$. Informellement, pour $\sim_h$ deux grilles sont égales si et seulement si elles ont les mêmes chiffres dans chaque ligne. De même, deux grilles sont égales pour $\sim_v$ si et seulement si elles ont les mêmes chiffres dans chaque colonne. $\mathrm{WD}_v(\sigma)$ désigne alors le nombre minimal d'échanges à faire pour mettre chaque nombre dans sa ligne, et idem pour $\mathrm{WD}_h(\sigma)$.

Théorème 3.6. *La walking distance est admissible et plus informée que la distance de Manhattan :*

$$\forall\sigma \in S_{mn}, d(\sigma, id) \geq WD(\sigma) \geq \frac{MD(\sigma, id)}{2}$$

.

Démonstration. Considérons un chemin σ à id et notons V le nombre de d'échanges horizontaux effectués et H le nombre d'échanges verticaux. Par définition, un échange vertical ne modifie pas WD_v et un échange horizontal ne modifie pas WD_h. En outre, un échange horizontal fait varier WD_v de 1, -1 ou 0 et un échange vertical fait varier WD_h de 1, -1 ou 0. Ainsi on a $H \geq \mathrm{WD}_h(\sigma)$ et $V \geq \mathrm{WD}_v(\sigma)$ donc $d(\sigma, id) \geq \mathrm{WD}(\sigma)$. Considérons

$$\mathrm{MD}_h : [\tau]_h \in G_{m,n}/\sim_h \to \frac{1}{2}\sum_{k=0}^{mn-1} |(\tau(k)\,\%\,n) - (k\,\%\,n)|$$

$$\mathrm{MD}_v : [\tau]_v \in G_{m,n}/\sim_v \to \frac{1}{2}\sum_{k=0}^{mn-1}\left|\left\lfloor\frac{\tau(k)}{n}\right\rfloor - \left\lfloor\frac{k}{n}\right\rfloor\right|$$

Alors MD_h et MD_v sont bien définies (elles ne dépendent pas du choix du représentant) et un raisonnement similaire à celui fait en 3.1 montrerait que ce sont, respectivement, des heuristiques admissibles pour la recherche d'un plus court chemin de $[\tau]_h$ à $[id]_h$ dans $(G_{m,n}/\sim_h, A^h_{m,n})$ et de $[\sigma]_v$ à $[id]_v$ dans $(G_{m,n}/\sim_v, A^v_{m,n})$ pour tout $\tau \in G_{mn}$. Or, $\mathrm{WD}_v(\sigma)$ et $\mathrm{WD}_h(\sigma)$ représentent justement les distances du plus court chemin dans ces deux graphes. On a donc $\mathrm{WD}_v(\sigma) \geq \mathrm{MD}_v(\sigma)$ et $\mathrm{WD}_h(\sigma) \geq \mathrm{MD}_h(\sigma)$ d'où $\mathrm{WD}(\sigma) \geq \dfrac{\mathrm{MD}(\sigma)}{2}$. □

3.3.2 Heuristique de base de données

Korf & Felner ont développé deux méthodes pour calculer des heuristiques. La première, dite paritionnée statiquement, consiste à partitionner $[\![0, mn-1]\!]$ en ensembles $(S_i)_{i\in I}$. Pour chacun des S_i, on considère la relation d'équivalence $\sigma \iff {}_i\tau \forall k \in S_i, \tau^{-1}(k) = \sigma^{-1}(k)$ puis $A_i = \{([\sigma]_i, [\tau]_i \,|\, (\sigma,\tau) \in A^2_{m,n}\}$. Autrement dit, deux grilles sont égales pour $\iff {}_i$ si et seulement si les éléments de S_i y occupent la même position. La valeur de l'heurisitique est alors la demi-somme pour $i \in I$ des distances de $[\sigma]_i$ à $[\mathrm{id}]_i$ dans le graphe $(S_{mn}/\sim_i, A_i)$.

La seconde, dite partitionnée dynamiquement, demande d'abord de choisir un entier $k \in [\![0, mn-1]\!]$ puis de résoudre toutes les grilles partielles à k nombres. Dans une telle grille, il y a k vrais nombres et les autres sont remplacés par des symboles génériques dont la position finale importe peu. Étant donné une grille σ et un ensemble d'entiers I on considère l'ensemble des grilles partielles à k nombres (avec $k \in I$) extraites. On peut représenter cet ensemble par un hypergraphe pondéré où les sommets sont les couples $(i, \sigma(i))$, $i \in [\![0, mn-1]\!]$ et les hyperarêtes tous les k-uplets de sommets. Chaque hyperarête est pondérée par la moitié du nombre minimal d'échanges nécessaire pour résoudre la configuration associée aux sommets inclus. La valeur de l'heuristique est alors la somme des poids d'un couplage.

4 Solveurs

4.1 Solveurs naïfs

4.1.1 Solveur glouton

Considérons l'application $f : (\sigma, k) \in S_{nm} \times [\![0,\ nm-1]\!] \to \sigma\circ(\sigma^{-1}(k)\sigma^{-1}(k)-1)(\sigma^{-1}(k)\sigma^{-1}(k)-1)$ Soit $\sigma \in S_{nm}$ et construisons σ^-1 comme produit d'éléments de S_{nm} par récurrence sur $k \in [\![0,\ nm-1]\!]$:

4.2 Parcours en largeur

Nous avons implémenté 4 algorithmes de parcours en largeur :

— Le parcours en largeur de la question de la question 7 qui demande de construire le graphe $(S_{nm}, A_{n,m})$. $\text{card}(S_{nm}) = (nm)!$ et on peut faire $nm - 1 - (m - 1) = n(m - 1)$ échanges horizontaux distincts et $nm-2-(n-1) = (n-1)m$ échanges verticaux distincts donc $2nm-m-n$ échanges au total. D'après le lemme des poignées de main, $\text{card}(A_{nm}) = \frac{1}{2}\sum_{\sigma\in S_{nm}} \deg(\sigma) = \frac{1}{2}\sum_{\sigma\in S_{nm}}(2nm-m-n) = (2nm-m-n)(mn)!$. L'algorithme a donc des complexités temporelles et spatiales $O(nm(nm)!)$.
— Le parcours en largeur de la question 8 qui construit au fur et à mesure le graphe $(S_{nm}, A_{n,m})$. Les complexités temporelles et spatiales dans le pire cas sont respectivement $O(nm(nm)!)$ et $O((nm)!)$ où encore $O(((2nm))^d)$ en temps et mémoire pour trouver un nœud à distance d de id.
— Le parcours en largeur bidirectionnel. Il consiste à effectuer un parcours en largeur depuis la grille qu'on cherche à résoudre et un autre depuis la grille finale. Comme le parcours en largeur considère tous les noeuds à distance d avant les noeuds à distance $d+1$, on a la garantie que le premier chemin formé par un noeud visité les deux parcours appartient à un plus court chemin entre la grille considérée et la grille finale. Il suffit alors de remonter de chaque côté pour en déduire un chemin. Les complexités dans le pire cas sont les mêmes que celles de la version unidirectionnelle mais, pour trouver un nœud à distance d, la complexité en temps et mémoire est de $O(\sqrt{((2nm))^d})$.
— Le parcours en largeur pseudo-bidirectionnel. Il consiste à simuler un parcours en largeur en bidirectionnel en ne faisant en réalité qu'un parcours unidirectionnel. Si l'on connaît un chemin de id à τ, il ne reste plus qu'à trouver un chemin de τ de σ ce qui revient à un chemin de $\sigma^{-1}\circ = \text{id}$. Toutefois, la réunion de deux plus courts chemins n'a aucune raison d'être un plus court chemin aussi donc cet algorithme n'est pas exact (contrairement à ceux du dessus).

4.3 Solveur avec heuristique

Nous avons implémenté l'algorithme A*. Dans le pire cas, ses complexités en mémoire et en temps sont respectivement $O(nm(nm)!\log((nm)!) = O((nm)^2(nm)!\log(nm))$ (d'après la formule de Stirling) et $O((nm)!)$ respectivement.

5 Détails d'implémentation

— Les distances de Manhattan et d'inversion sont implémentées dans le fichier `utils.py` sous les noms respectifs `half_manhattan_distance` et `inversion_distance`. L'implémentation de la distance d'inversion utilise l'algorithme naïf quadratique.
— La *walking distance* est implémentée dans le fichier `wd.py`. La résolution des grilles lors du précalcul se fait avec un parcours en largeur.
— L'heuristique de base de données statique est implémentée dans `apdb.py`. Deux implémentations sont proposées : une utilisant un dictionnaire et

l'autre un tableau numpy. L'heuristique de base de données dynamique est implémentée dans `gadb.py`. Au vu de la taille exponentielle des base de données en jeu, un tableau numpy est utilisé ici. Contrairement à l'implémentation de l'heurisitque statique où, pour stocker $\frac{(nm)!}{(nm-k!)}$ éléments on crée un tableau k dimensionnel de côté nm (soit $(nm)^k$ emplacements au total), on alloue ici exactement la taille demandée. Pour ce faire, on stocke une bijection de l'ensemble des k-arrangements de $[\![0,\, nm-1]\!]$ dans $\left[\!\left[1,\, \frac{(nm)!}{(nm-k)!}\right]\!\right]$ et une autre des parties à k éléments de $[\![0,\, nm-1]\!]$ dans $\left[\!\left[1,\, \binom{nm}{k}\right]\!\right]$. Puisque l'ensemble des grilles à k nombres s'identifie au produit cartésien entre les k-arrangements de $[\![0,\, nm-1]\!]$ et les parties à k éléments de $[\![0,\, nm-1]\!]$, on peut simplement créer un tableau bidimensionnel $\frac{(nm)!}{(nm-k)!} \times \binom{nm}{k}$. Le stockage des bijections f et g sous forme de dictionnaire est largement contrebalancé par le gain de place. Le calcul de l'heuristique dans le cas statique se fait avec un algorithme glouton : les hyperarêtes sont considérées par poids décroissant et sélectionnées si elles ne partagent pas de sommet avec celles déjà prises.

— Une implémentation d'A* utilise un tas binaire pour la recherche du minimum et une autre une *bucket queue* (implémentée dans `solver_utils.py`). Entre deux nœuds qui ont la même distance estimée f, on privilégie celui avec la plus grande distance g. Une implémentation spéciale d'A* calcule la distance de Manhattan de manière incrémentale. En effet, la variation de la distance de Manhattan au cours d'un échange ne dépend que des nombres échangées et de leurs positions donc on peut précalculer rapidement cette quantité. Au moment d'un échange il ne reste plus qu'à la mettre à jour.

6 Résultats

6.1 Résultats sur les heuristiques

Heuristique	Temps de calcul (en s)	Taille dans la RAM (en MB)	Taille sur disque (en MB)
GADB 4 × 4, 3	3,27	1,9	0,4
GADB 4 × 4, 4	241	79,4	16,7
APDB 3 × 3 (1, 2, 3, 4, 5, 6, 7, 8, 9)	16,6	387,4	1,2
APDB 4 × 4 (1, 2, 3, 5, 6)	24	1,0	0,2
APDB 4 × 4 (1, 2, 3, 4, 5, 6, 7)	3652	268,4	26,3
APDB 4 × 4 (9, 10, 13, 14, 15, 16)	313	16,8	2,3