rapport : Two Algorithmic Results for the Traveling Salesman Problem

Fanny Dufossé

7 janvier 2007

1 Introduction

Cet article présente plusieurs algorithmes concernant le calcul du hamiltonien. Toutes les opérations peuvent se faire sur des rationnels si les entrées sont rationnelles.
Les premiers comptent le nombre de circuits hamiltoniens d'un graphe. On calcule le hamiltonien en temps $O(n^4 \cdot 2^n)$, et celui d'une matrice dont le rang est borné par r en temps polynomial en n, le degrés du polynôme étant linéaire en r^2 . Cela permet de construire un circuit hamiltonien en temps $O(n^6 \cdot 2^n)$.
Les suivants construisent un chemin hamiltonien d'un graphe approximant le plus long chemin du graphe. On approxime ainsi le poids du plus long chemin d'un graphe G avec une erreur relative ϵ en temps polynomial en n et en ϵ^{-1}. On peut calculer également calculer avec une erreur relative inférieure à ϵ la longueur maximale d'un circuit hamiltonien pour un ensemble de points d'un espace euclidien en temps polynomial en n et ϵ^{-1}, pour toute norme de cet espace.

2 Notations

On notera $\mathcal{P}_n = \mathbb{R}[x_1, \ldots, x_n]$ l'ensemble des polynômes réels à n variables. Les polynômes de $\mathcal{P}_n$ seront présenté sous cette forme :

$$P(x) = \sum_{\alpha} p_\alpha \cdot x^\alpha$$

avec x et α vecteurs de $\mathbb{R}_n$, pour tout α $\alpha_i \geq 0$
et $x^\alpha = x_1^{\alpha_1} \cdots x_n^{\alpha_n}$.

Soit G un graphe. Pour un chemin $\pi = \{i_1 \to \cdots \to i_m\}$, on notera $i_1 = head(\pi)$ et $i_m = tail(\pi)$. On dira que $\Pi = \{\pi_1, \ldots, \pi_t\}$ est une collection de chemin de G si les chemins sont deux à deux disjoints et si tout sommet de G est dans Π. On définit le poids d'une collection Π de chemins , noté w_Π comme la somme des poids des chemins : $w_\Pi = w_{\pi_1} + \ldots + w_{\pi_t}$.

3 Calcul du hamiltonien

3.1 Méthode

Pour cet algorithme, on utilise le produit interne suivant pour $\mathcal{P}_n$:
pour $P = \sum_\alpha p_\alpha \cdot x^\alpha$ et $Q = \sum_\alpha q_\alpha \cdot x^\alpha$
On pose : $\langle P, Q \rangle = \sum_\alpha \alpha_1! \cdots \alpha_n! \cdot p_\alpha \cdot q_\alpha$

Ce produit interne présente l'intérêt de vérifier certaines propriétés :
Soit $P, Q \in \mathcal{P}_n$ et G fonction linéaire de $\mathbb{R}_n$.
On définit $G(P)$ ainsi : $G(P)(x) = P(G^*(x))$ avec G^* étant le conjugué de G
($\langle x, G(y) \rangle = \langle G^*(x), y \rangle$).

On a donc la propriété suivante : $\langle G(P), Q \rangle = \langle P, G^*(Q) \rangle$

On a également cette propriété : pour $P \in \mathcal{P}_n$ et $Q(x) = x_1 \cdots x_n$

$$\langle P, Q\rangle = (-1)^n \sum_{\omega} (-1)^{|\omega|} P(x_\omega)$$

avec $\omega \subset \{1, \ldots, n\}$ et $(x_\omega)_i = 1$ si $i \in \omega$, et 0 sinon.

3.2 Calcul du hamiltonien d'une matrice

On notera $hamp\ A = \sum_{g \in H_n} \prod_{i=1}^{n} a_{i,g(i)}$

L'algorithme de calcul de $hamp\ A$ utilisera la propriété suivante :
Soit A une matrice carrée de taille $n \times n$ et $x = (x_1, \ldots, x_n)$.
Soit $D = diag\{x_1, \ldots, x_n\}$. On pose $P(x) = Tr((D(x) \cdot A)^n)$ et $Q(x) = x_1 \times \ldots \times x_n$.
Alors, $hamp\ A = \frac{1}{n} \cdot \langle P, Q\rangle$.

On obtient ainsi l'algorithme suivant sur l'entrée A, matrice carrée de taille $n \times n$:

- Poser $hamp\ A := 0$
- Pour tout les $\omega \in 1, \ldots, n$, on définit $A(\omega) = (a_{ij}(\omega))$ ainsi :

$$a_{ij}(\omega) = \begin{cases} a_{ij}, & \text{si } i \in \omega \\ 0, & \text{sinon} \end{cases}$$

  puis, calculer $hamp\ A := hamp\ A + (-1)^{|\omega|} \cdot Tr\ A(\omega)^n$
- renvoyer : $hamp\ A := \frac{(-1)^n}{n} \cdot hamp\ A$

Cet algorithme calcule le hamiltonien d'une matrice en temps $O(n^4 \cdot 2^n)$.

3.3 hamiltonien d'une matrice de rang borné

Voici un algorithme pour calculer le hamiltonien d'une matrice carrée A de taille $n \times n$ et de rang borné par r. On supposera que $n \geq r^2$.

- Calculer une matrice carrée inversible U de taille $n \times n$ telle que les $n - r$ dernières colonnes de AU sont nulles. Calculer U^{-1}.
- Définir $D(x) = diag\{x_1, \ldots, x_n\}$ et $S(x) = U^{-1}D(x)AU$. On calcule les $\sigma_{i,j} \in \mathbb{R}^n$, pour tout $i, j \in \{1, \ldots, n\}$, définis ainsi : $s_{i,j} = \langle \sigma_{i,j}, x\rangle$.
- Poser $m = r^2$. Construire une bijectionϕ de l'ensemble $\{(i, j) : 1 \leq i, j \leq r\}$ dans l'ensemble $\{1, \ldots, m\}$.
- Construire la matrice carrée $F(x)$ de taille $r \times r$ telle que $f_{i,j} = x_{\phi(i,j)}$. Calculer $L(x) = Tr(F(x)^n)$ sous la forme $L(x) = \sum_\alpha \lambda_\alpha \cdot x^\alpha$.
- Construire la matrice carrée $G(x)$ de taille $r \times r$ telle que la j-ième colonne de G soit égale au vecteur $s_{\phi^{-1}(j)}$ complétée par des zéros. Construire $R(x) = \sum_{i=1}^{n} \prod_{j=1}^{m} g_{ij} \cdot x_j$ sous la forme $R(x) = \sum_\alpha \rho_\alpha \cdot x^\alpha$
- Renvoyer $hamp\ A = \frac{1}{n} \cdot \langle L, R\rangle$

Cet algorithme calcule le hamiltonien d'une matrice carrée de taille $n \times n$ de degrés borné par r en temps polynomial en n, le degrés du polynôme étant linéaire en r^2.

3.4 Nombre de circuits hamiltoniens d'un graphe

Ces deux algorithmes calcule le nombre de circuits hamiltoniens d'un graphe représenté par sa matrice d'adjacence non pondérée.
Voyons maintenant comment construire un circuit hamiltonien dans un graphe G représenté par une matrice d'adjacence A telle que $hamp\ A < 0$. Voici un algorithme simple découlant des algorithmes de calcul de $hamp\ A$.

Pour toute arête e de G, retirer l'arête e de G, et recalculer *hamp* A.
– Si il est non nul, passer à l'arête suivante ;
– sinon, remettre e dans g et passer à l'arête suivante.

Cet algorithme construit un circuit hamiltonien en temps $O(n^6 \cdot 2^n)$ puisqu'il calcule *hampA* autant de fois qu'il y a d'arête dans G, donc moins de n^2 fois.

4 Calcul du plus long chemin

On va ici essayer de calculer le plus long chemin hamiltonien d'un graphe G pondéré par une matrice de poids W. Plus formellement, on va calculer $c(W) = \max\{\sum_{i=1}^{n} w_{i,g(i)} : g \in H_n\}$

4.1 Rang combiné

Pour approximer $c(W)$, on va utiliser le rang combiné des matrices : Pour une matrice réelle carrée W de taille $n \times n$, on dit que le rang combiné de W, noté *comr* W est inférieur à r si il existe $2r$ vecteurs de $\mathbb{R}^n$ $h^1, \ldots, h^r, v^1, \ldots, v^r$ tels que : $\forall i, j \in \{1, \ldots, n\}$

$$w_{ij} = \max_s\{h_i^s + v_j^s\}$$

Le rang combiné de toute matrice carrée de taille $n \times n$ est inférieur ou égal à n. Une matrice de rang inférieur ou égal à r sera présenté sous forme de $2r$ vecteurs. On dit que cette représentation est bien centrée si : $\forall i, j \in \{1, \ldots, n\}$

$$\max_s\{h_i^s + v_j^s\} = \max_s\{|h_i^s + v_j^s|\}$$

Certains algorithmes nécessitent une représentation bien centrée. Si une matrice est formulée sous une représentation qui ne vérifie pas cette propriété, on peut obtenir une représentation bien centrée en ajoutant une constante C suffisamment grande à l'ensemble des poids. Ainsi, le poids de tous les chemins hamiltoniens du graphe sera augmenté de la même valeur :$C \times n$ Cependant, en faisant cela, on peut perdre l'approximation relative quand il faudra soustraire $C \times n$ au poids du plus long chemin.

4.2 Méthode

Étant donné un graphe G représenté par une matrice de poids W et une valeur ϵ, on veut calculer le poids du plus long chemin de G avec une approximation relative ϵ.
Pour cela, on va supposer que W est donné sous une représentation bien centrée. On a donc $r \geq comr\ W$ et $2 \cdot r$ vecteurs de $\mathbb{R}^n$.

On définit la matrice carrée $A(t) = (a_{ij}(t)$ de taille $n \times n$ ainsi :

$$a_{ij}(t) = \sum_{s=1}^{r} \exp\{t \cdot h_i^s\} \exp\{t \cdot v_j^s\}$$

On choisit m tel que $m > \frac{\log((n-1)!) + n \log r}{\log(1+\epsilon)}$ et on pose

$$c_m = \left(\frac{1}{r^n (n-1)!} \frac{d^m}{dt^m} hamp\ A(t) \Big|_{t=0} \right)^{\frac{1}{m}}$$

Alors, c_m calcule $c(W)$ avec l'approximation relative ϵ. On a la relation :
$c_m \leq c(W) \leq (1+\epsilon) \cdot c_m$
On va se servir de cette relation pour approximer $c(W)$. Mais si les valeurs en entrée sont rationnelles, les valeurs en sorties le sont également, et on ne veut pas avoir des irrationnels dans le calcul. On va donc remplacer $A(t)$ par $\widetilde{A}(t)$ avec

$$\tilde{a}_{ij}(t) = \sum_{s=1}^{r} \left(1 + t \cdot h_i^s + \cdots + \frac{(t \cdot h_i^s)^m}{m!}\right) \left(1 + t \cdot v_j^s + \cdots + \frac{(t \cdot v_j^s)^m}{m!}\right)$$

En effet, pour tout $1 \leq i, j \leq m$ a_{ij} et $\tilde{a}_{ij}$ ont la même valeur en zéro et il en va de même pour toute leurs dérivées jusqu'à la dérivée m-ième.

4.3 Calcul du poids d'un long chemin hamiltonien

Voici un algorithme pour calculer le poids du plus long chemin hamiltonien d'un graphe G avec pour matrice de poids W, avec l'approximation relative ϵ. On suppose $\epsilon < \frac{1}{2}$.

- Poser $m = \lfloor 4 \cdot \epsilon^{-1} \cdot n^2 \rfloor + 1$
- Calculer $\widetilde{A}(t)$ et $hamp\ \widetilde{A}(t)$
- Pour $0 \leq k \leq 2mn$ calculer les α_k tels que $hamp\ \widetilde{A}(t) = \sum_{k=0}^{2mn} \alpha_k \cdot t^k$
- Calculer
$$d = \frac{m!}{r^n(n-1)!} \cdot \alpha_m$$
- Renvoyer
$$c_m = d^{\frac{1}{m}}$$

Cet algorithme calcule c_m en temps polynomial en n et en ϵ^{-1}. Le dernier calcul peut générer un irrationnel même si les entrées sont rationnelles. Pour éviter cela, on peut doubler m dans la première étape, puis approximer $d^{\frac{1}{m}}$ par un rationnel à $\frac{\epsilon}{2}$ prés. Cette première étape permet simplement de choisir un m de taille polynomiale en n et en ϵ^{-1} tout en vérifiant la relation : $m > \frac{\log((n-1)!)+n\log r}{\log(1+\epsilon)}$.

Autre version de cet algorithme : on renvoie (d, m) tels que $d^{\frac{1}{m}}$ approxime $c(W)$ à ϵ prés et m est un entier tel que $m \leq 4\epsilon^{-1}n^2 + 2$.

4.4 Construction d'un long chemin hamiltonien

Dans un graphe G, pour construire un chemin dont le poids appproxime $c(W)$ à ϵ prés, on va partir d'une collection de chemin réduits à un sommet, et on va ajouter les arêtes les unes après les autres. On notera $c(W, \Pi)$ le poids du plus long chemin hamiltonien de G contenant les chemins π_i.

On va donc définir $\overline{W}$ tel que $\overline{w}_{ij} = w_{ab}$ avec $a = tail(\pi_i)$ et $b = head(\pi_j)$ et si W est donné sous sa représentation bien centrée : $w_{ij} = \max_s\{h_i^s + v_j^s\}$, $\overline{w}_{ij} = \max_s\{\overline{h}_i^s + \overline{v}_j^s\}$ avec $\overline{h}_i^s = \overline{h}_a^s$ et $\overline{h}_i^s = \overline{h}_b^s$. On a donc d'après l'article $c(W, \Pi) = w_\Pi + c(\overline{W})$. Mais cette relation ne semble pas justifiée si il n'existe pas de chemin hamiltonien contenant les π_i.

Selon l'article, on peut donc construire un chemin de longueur approximant l'optimal à ϵ prés sur un graphe G de matrice de poids W ainsi : On supposera $n > 2$ et $\epsilon < 1$.

- On pose $\epsilon_1 = \frac{\epsilon}{2n}$, $t = n$ et $\Pi = \{\pi_1, \ldots, \pi_n\}$ avec pour tout i $\pi_i = \{i\}$
- Tant que $t > 1$: Pour k allant de 1 à t faire :
  Construire le chemin $\tilde{\pi}_k$ en concaténant les chemins π_k et π_t (on ajoute l'arête entre $tail(\pi_k)$ et $head(\pi_t)$). Soit $\Pi_k = \{\pi_1, \ldots, \pi_{k-1}, \tilde{\pi}_k, \pi_{k+1}, \ldots, \pi_{t-1}\}$. Calculer $c(\overline{W})$ à ϵ_1 prés, puis $c_k = w_{\Pi_k} + c(\overline{W})$. fin Pour fin Tant que.
  Choisir k tel que $c_k = \max_{1 \leq j < t}\{c_j\}$. Redéfinir Π : $\Pi = \Pi_k$.
- Renvoyer l'unique circuit hamiltonien g contenant le chemin π_1.

On remarquera que dans cet algorithme, on ne tient aucun compte on ne considère pas qu'une matrice de poids 0 n'existe pas. Cet algorithme est en temps polynomial en n et ϵ^{-1}. D'après l'article, il renvoie un circuit hamiltonien dont le poids est une approximation relative à ϵ prés l'optimal.

4.5 Dans un espace euclidien

Dans l'espace $\mathbb{R}^d$, on dit qu'une norme est polyédrique si sa boule unité ($\mathbf{B} = \{x \in \mathbb{R}^d : \|x\| \leq 1\}$) est un polytope. Pour une telle norme, il existe donc $b^1, \ldots, b^r$ tels que $\mathbf{B} = \{x \in \mathbb{R}^d : \langle b^s, x\rangle \leq 1, s = 1, \ldots, r\}$
. On dit que $\mathbf{B}$ a r facettes.

Soit $\|\cdot\|$ norme polyédrique de $\mathbb{R}^d$. Soit $P_1, \ldots, P_n \in \mathbb{R}^d$. On pose W tel que $w_{ij} = \|P_i - P_j\|$. Alors, le rang combiné de W est borné par r.

Pour approximer le plus long chemin hamiltonien d'un ensemble de points $P_1, \ldots, P_n \in \mathbb{R}^d$ à ϵ prés selon la norme polyédrique $\|\cdot\|$, on calcule $h_i^s = \langle P_i, b^s\rangle$ et $v_i^s = -\langle P_i, b^s\rangle$. On définit ainsi W par sa représentation bien centrée. On peut ensuite appliquer l'algorithme vu précédemment pour W. On sait que *comr* $W \leq r$.

Pour une norme quelconque $\|\cdot\|$, on va construire un polytope à symétrie centrale en 0 $\mathbf{B_1}$ tel que

$$(1 - \epsilon/3) \cdot \mathbf{B} \subset \mathbf{B_1} \subset (1 + \epsilon/3) \cdot \mathbf{B}$$

On effectue l'algorithme vu précédemment avec une erreur relative inférieure à $\epsilon/3$.

5 Conclusion

On obtient ainsi des algorithmes rapides pour résoudre des problèmes NP-complets. Cependant, l'un des algorithmes semble présenter un défaut majeur. L'auteur présente ces algorithmes comme les meilleurs connus actuellement.