# Algorithmique avancée: Approximation

Serge Haddad

LMF, ENS Paris-Saclay & CNRS & INRIA

L3

# Optimisation et approximation

Une instance d'un problème de *minimisation* consiste en :

- un ensemble de solutions admissibles $\{sol\}_{sol \in \mathcal{S}}$ ;
- le *coût* strictement positif $C(sol) > 0$ d'une solution $sol$ ;
- l'existence de $sol^*$ telle que $C(sol^*) \overset{\text{def}}{=} C^* = \min(C(sol) \mid sol \in \mathcal{S})$.

Les problèmes de *maximisation* se définissent avec des *récompenses* à maximiser.

Un *algorithme d'approximation* $\rho(n)$

- prend en entrée une instance de taille $n$, renvoie une solution $sol$ ;
- et fournit une *garantie de performance* $\rho(n)$ : $\frac{C(sol)}{C^*} \leq \rho(n)$.

Un *schéma d'approximation* est un algorithme qui prend en entrée une instance et $\varepsilon > 0$ et garantit une performance $1 + \varepsilon$.

- Il est *polynomial* s'il s'exécute en temps polynomial par rapport à $n$ ;
- Il est *entièrement polynomial* s'il s'exécute en temps polynomial par rapport à $n$ et $\frac{1}{\varepsilon}$.

# Plan

1. La couverture de sommets

Le voyageur de commerce

Différents couplages

Le sac à dos

# La couverture de sommets

**Le problème de décision**

- Soit $G = (V, E)$ un graphe et $k \in \mathbb{N}$.
- Existe-t-il $V'$ tel que $|V'| \leq k$ et pour tout $\{u, v\} \in E$, $\{u, v\} \cap V' \neq \emptyset$ ?

**Le problème de minimisation**

- Soit $G = (V, E)$ un graphe.
- Renvoyer $V'$ tel que $|V'|$ soit minimal
  et pour tout $\{u, v\} \in E$, $\{u, v\} \cap V' \neq \emptyset$ ?

Le problème de décision est NP-complet.

Par conséquent, sauf si P $=$ NP, il n'existe pas d'algorithme en temps polynomial pour résoudre le problème de minimisation.

# Un algorithme glouton

$V' \leftarrow \emptyset$
$E' \leftarrow E$
**While** $E' \neq \emptyset$ **do**
  $\{u, v\} \leftarrow \mathbf{Extraire}(E')$
  $V' \leftarrow V' \cup \{u, v\}$
  $\mathbf{Supprimer}(E', u)$
  $\mathbf{Supprimer}(E', v)$
**Return**$(V')$

**Complexité.**

Avec un tableau des listes d'arêtes par sommet pointant sur (et pointées par) une liste globale, la suppression d'une arête s'effectue en $O(1)$.

Par conséquent, l'algorithme opère en $O(|E|)$.

# Analyse de l'algorithme

**Correction.**

L'invariant de boucle est le suivant :

$$\text{Pour tout } \{u,v\} \in E \setminus E',\ \{u,v\} \cap V' \neq \emptyset.$$

D'où en sortie de boucle :

$$\text{Pour tout } \{u,v\} \in E,\ \{u,v\} \cap V' \neq \emptyset.$$

**Observation.** Soit $\{u_1, v_1\}, \ldots \{u_k, v_k\}$,
les arêtes extraites en début de chaque itération.

- Pour tout $i \neq j$, $\{u_i, v_i\} \cap \{u_j, v_j\} = \emptyset$ ;
- $|V'| = 2k$.

**Garantie de performance.**

Soit $V^*$, une solution minimale.

Pour tout $i \leq k$, $\{u_i, v_i\} \cap V^* \neq \emptyset$.

Par conséquent $|V^*| \geq k$ et $\frac{|V'|}{|V^*|} \leq 2$.

# Plan

# Le voyageur de commerce

**Le problème de décision** *(le cycle hamiltonien)*

- Soit $G = (V, E)$ un graphe avec $|V| = n$.
- Existe-t-il une énumération $V = (v_0, \dots, v_{n-1})$ telle que pour tout $i < n$, $\{v_i, v_{i+1\%n}\} \in E$ ?

**Le problème de minimisation**

- Soit $V$ un ensemble de sommets avec $|V| = n$ et $d : V \times V \to \mathbb{R}_+$.
- Renvoyer une énumération $V = (v_0, \dots, v_{n-1})$ telle que $\sum_{i<n} d(v_i, v_{i+1\%n})$ soit minimal.

Le problème de décision est NP-complet.

Sauf si P = NP, il n'existe pas d'algorithme d'approximation en temps polynomial avec une garantie de performance constante.

# Preuve d'inapproximabilité

**Par l'absurde.**

Supposons qu'il existe un algorithme $\mathcal{A}$ en temps polynomial
avec une garantie de performance constante $\rho$.

Soit $G = (V, E)$, on construit en temps linéaire le problème suivant :

- $V' = V$ ;
- Pour tout $(u, v)$, si $\{u, v\} \in E$ alors $d(u, v) = 1$ sinon $d(u, v) = \rho|V| + 1$.

Soit une énumération $sol \stackrel{\text{def}}{=} (v_0, \ldots, v_{n-1})$ avec $\{v_i, v_{i+1\%n}\} \notin E$ alors :

$$C(sol) \geq \rho|V| + 1 + |V| - 1 > \rho|V|$$

- Si $G$ possède un cycle hamiltonien alors $C^* = |V|$
  et $\mathcal{A}$ renvoie une énumération $sol$ avec $C(sol) \leq \rho|V|$.
- Si $G$ ne possède pas de cycle hamiltonien alors pour toute $sol$, $C(sol) > \rho|V|$
  et $\mathcal{A}$ renvoie une énumération $sol$ avec $C(sol) > \rho|V|$.

Selon la réponse de $\mathcal{A}$, on décide si $G$ possède un cycle hamiltonien.

# Qu'arrive-t-il si $d$ est une distance ?

**Distance euclidienne.** *(prix Gödel 2010)*

Il existe un schéma d'approximation (non entièrement) polynomial en dimension quelconque (mais fixe), établi indépendamment par Sanjeev Arora et Joseph S.B. Mitchell (très difficile).

**Distance quelconque.**

- Il existe un algorithme en temps polynomial avec garantie de performance 2 basé sur un arbre couvrant de poids minimal (facile).
- Il existe un algorithme en temps polynomial avec garantie de performance $\frac{3}{2}$ basé sur un couplage parfait de poids minimal (difficile).

**Observation.** Sanjeev Arora a aussi obtenu le prix Gödel en 2001 pour un résultat en théorie de la complexité $\mathsf{NP} = \mathsf{PCP}[O(\log n), O(1)]$.

# Construction d'une énumération

Soit $V$ un ensemble de sommets et $d : V \times V \to \mathbb{R}_+$ une distance.

On considère le graphe complet $G = (V, \{u, v\}_{u,v \in V})$ et les poids $d$.

On construit un arbre couvrant de poids minimal.

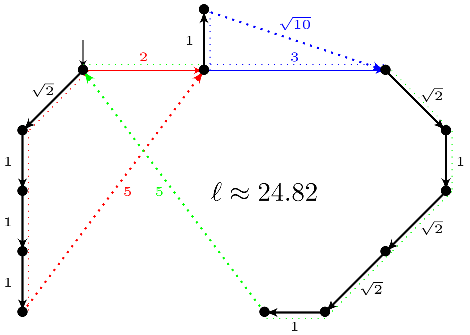
On choisit une racine et on parcourt en profondeur l'arbre.

L'énumération est la suite des sommets visités.

**Complexité.**

Toutes les opérations s'effectuent en temps polynomial.

# Illustration




Pour cet exemple, on choisit la distance euclidienne.

En plein, le parcours de l'arbre couvrant

En pointillés fins, les retours des appels récursifs

En gras, le parcours du voyageur de commerce

En gras et pointillés, les raccourcis effectués par le voyageur de commerce

# Analyse de la garantie de performance

Le parcours de l'arbre $\mathcal{T}$ en profondeur « visite » deux fois chaque arête $(u, v)$ orientée par le choix de la racine :

- lors de l'exploration à partir de $u$ par un appel récursif en $v$ ;
- lors du retour de l'appel récursif en $v$.

Un « arc » $(u, v)$ de l'énumération $sol$ correspond dans le parcours de l'arbre :

- si $u$ n'est pas une feuille, à un arc $(u, v)$ de l'arbre ;
- si $u$ est une feuille qui n'est pas la dernière visitée, à une suite de retours d'appel $u, u_1, u_1, u_2, u_{k-1}$ suivi d'un appel en $v$ à partir de $u_k$ avec $d(u, v) \leq d(u, u_1) + \ldots + d(u_{k-1}, u_k) + d(u_k, v)$.
- si $u$ est la dernière feuille visitée, à une suite de retours d'appel $u, u_1, u_1, u_2, u_{k-1}, v$ où $v$ est la racine avec $d(u, v) \leq d(u, u_1) + \ldots + d(u_{k-1}, v)$.

Soit $C(\mathcal{T})$ le poids de $\mathcal{T}$. Alors $C(sol) \leq 2C(\mathcal{T})$.
Soit $sol^*$ la solution optimale et $(u, v) \in sol^*$. $sol^* \setminus (u, v)$ est un arbre couvrant.
Donc $C(sol^*) \geq C(\mathcal{T})$ et $\frac{C(sol)}{C(sol^*)} \leq 2$.

# Plan

# Des couplages optimaux

- Soit $G = (V, E)$ un graphe.

  Un *couplage* $M \subseteq E$ est un ensemble d'arêtes disjointes.

  Recherche d'un couplage $M$ avec $|M|$ maximal.

  - en temps polynomial dans les graphes bipartis (vu en Algorithmique 1) ...
  - et dans les graphes quelconques (plus difficile).

- Soit $V$ un ensemble de sommets et $C : V \times V \to \mathbb{R}_+$ avec $C(v, u) = C(u, v)$.

  Un *couplage* $M$ est un ensemble de paires disjointes.

  Recherche d'un couplage $M$ de poids $C(M) \stackrel{\text{def}}{=} \sum_{\{u,v\} \in M} C(u, v)$ maximal.

- Si $|V|$ est pair, un *couplage parfait* est une partition de $V$ en paires disjointes.

  Recherche d'un couplage parfait $M$ de poids minimal.

# Chemin $M$-augmentant

Un sommet est *couvert* par $M$, couplage de $G$, s'il appartient à une paire de $M$.

Un chemin élémentaire $v_0 \dots v_{2t+1}$ est *$M$-augmentant* si

- pour tout $0 < i \leq t$, $\{v_{2i-1}, v_{2i}\} \in M$ ;
- $v_0$ et $v_{2t+1}$ ne sont pas couverts par $M$.

$$\boxed{M \text{ est maximal ssi il n'existe pas de chemin } M\text{-augmentant.}}$$

**Preuve.** Soit $v_0 \dots v_{2t+1}$ un chemin $M$-augmentant. Alors :
$M' = M \cup \{\{v_{2i}, v_{2i+1}\}\}_{0 \leq i \leq t} \setminus \{\{v_{2i-1}, v_{2i}\}\}_{0 < i \leq t}$ est un couplage et $|M'| = |M| + 1$.

Soit $M$ et $M'$ des couplages avec $|M'| > |M|$.

Considérons le graphe constitué des sommets et des arêtes de $M$ et $M'$.

Tout sommet est adjacent à une ou deux arêtes.

Une composante connexe est :

- soit un sommet isolé, soit une arête commune à $M$ et $M'$ ;
- soit alterne les arêtes de $M$ et $M'$ formant
  un cycle de longueur paire ou un chemin.

Puisque $|M'| > |M|$, l'une des composantes est un chemin de longueur impaire commençant et finissant par des arêtes de $M'$, soit un chemin $M$-augmentant.

# Calcul d'un couplage maximal

$M \leftarrow \emptyset$

**While** $(\rho = \mathbf{ChercheAug}(G, M)) \neq \emptyset$ **do** $M \leftarrow \mathbf{Aug}(M, \rho)$

$\mathbf{Return}(M)$

**Complexité.**

Au plus $\frac{|V|}{2}$ appels à **ChercheAug**.

**Observation.**

Lors d'un appel à **ChercheAug**, $M$ peut être modifié *sans changer sa taille*.

# $M$-Fleur

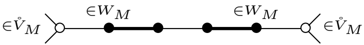
Soit $M$ un couplage.

- $\mathring{V}_M$ est l'ensemble des sommets non couverts par $M$.
- $W_M = \{u \in V \mid \exists \{u, v\} \in E \land v \in \mathring{V}_M\}$, est l'ensemble des voisins de $\mathring{V}_M$.

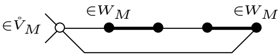
Un cycle élémentaire $v_0 \dots v_{2t+1} = v_0$ est une $M$-fleur si :

- pour tout $0 < i \leq t$, $\{v_{2i-1}, v_{2i}\} \in M$ ;
- $v_0 = v_{2t+1}$ n'est pas couvert par $M$.

un chemin $M$-augmentant



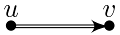

une $M$-fleur

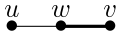


# Double graphe et plus court chemin

Le (double) graphe orienté $D(G,M) = (V,A)$ est défini par :

$$(u,v) \in A \text{ si et seulement si il existe } \{u,w\} \in E \setminus M \text{ et } \{w,v\} \in M.$$




Soit $u_0 u_2 \dots u_{2t}$ un plus court chemin de $\mathring{V}_M$ à $W_M$ dans $D(G,M)$.

Par définition de $D(G,M)$, $\mathring{V}_M$ et $W_M$, on obtient dans $G$ un chemin $\rho = u_0 u_1 u_2 \dots u_{2t} u_{2t+1}$ avec :

- $u_0, u_{2t+1} \in \mathring{V}_M$ ;
- pour tout $0 < i \leq t$, $\{u_{2i-1}, u_{2i}\} \in M$.

D'où s'il n'existe pas de plus court chemin,

alors il n'existe pas de chemin $M$-augmentant.

# Etude de cas

① Si $\rho$ est élémentaire, $\rho$ est un chemin $M$-augmentant.

Sinon soit $j = \min(k \mid \exists i < k \wedge u_k = u_i)$ et $i < j$ tel que $u_i = u_j$.

② Si $j = 2t+1$ alors $i = 0$ et $\rho$ est une $M$-fleur ;

③ Si $j < 2t+1$ alors $0 < i$ et :

- $j - i$ est impair car sinon le « repliage » de $u_0 \ldots u_{i-1} u_j \ldots u_{2t}$ dans $D(G, M)$ fournirait un chemin plus court.
- $i$ est pair car sinon $\{u_i u_{i+1}\}, \{u_{j-1} u_j\} \in M$ impliquant $u_{i+1} = u_{j-1}$ contredisant la définition de $j$.
- Soit le couplage $M' = M \setminus \{\{u_{2\ell-1}, u_{2\ell}\}\}_{0<2\ell\leq i} \cup \{\{u_{2\ell}, u_{2\ell+1}\}\}_{0\leq 2\ell<i}$ $|M'| = |M|$ et $u_i \ldots u_j$ est une $M'$-fleur.

# Graphe et couplage quotient

Soit $M$ un couplage et $\rho = u_0 u_1 \dots u_{2t+1}$ une $M$-fleur.

$G_{/\rho} = (V_{/\rho}, E_{/\rho})$ est défini par :

- $V_{/\rho} = V \setminus \{u_0, \dots, u_{2t}\} \cup \{u_\rho\}$ ;
- $E_{/\rho} = \{\{u,v\} \in E \mid \{u,v\} \cap \{u_0, \dots, u_{2t}\} = \emptyset\} \cup \{\{u, u_\rho\} \mid \{u,v\} \in E \wedge u \notin \{u_0, \dots, u_{2t}\} \wedge v \in \{u_0, \dots, u_{2t}\}\}$.

**Observation.** $|V_{/\rho}| \leq |V| - 2$.

$M_{/\rho}$, un couplage de $G_{/\rho}$, est défini par :

$$M_{/\rho} = \{\{u,v\} \in M \mid \{u,v\} \cap \{u_0, \dots, u_{2t}\} = \emptyset\}$$

**Observation.** $u_\rho$ n'est pas couvert par $M_{/\rho}$.

Il existe un chemin $M$-augmentant de $G$
ssi il existe un chemin $M_{/\rho}$-augmentant de $G_{/\rho}$.

# Une propriété des quotients

**Preuve.**

• Soit $\sigma = w_1 \dots w_n$ un chemin $M$-augmentant de $G$.

Sans perte de généralité, $w_1 \neq u_0$.

Si $\{w_1, \dots, w_n\} \cap \{u_0, \dots, u_{2t}\} = \emptyset$ alors $\sigma$ est un chemin $M_{/\rho}$-augmentant.

Sinon soit $j = \min(k \mid w_k \in \{u_0, \dots, u_{2t}\})$.

Alors $w_0 \dots w_{j-1} u_\rho$ est un chemin $M_{/\rho}$-augmentant.

• Soit $\sigma = w_1 \dots w_n$ un chemin $M_{/\rho}$-augmentant de $G_{/\rho}$.

Sans perte de généralité, $w_1 \neq u_\rho$.

Si $u_\rho \notin \{w_1, \dots, w_n\}$ alors $\sigma$ est un chemin $M$-augmentant.

S'il existe $j$ tel que $w_j = u_\rho$ alors $j = n$.

L'arête $\{w_{n-1}, u_\rho\} \in E_{/\rho}$ correspond à une arête $\{w_{n-1}, u_i\} \in E$.

Si $i$ est impair alors $w_0 \dots w_{n-1} u_i u_{i+1} \dots u_{2t+1}$ est un chemin $M$-augmentant

Si $i$ est pair alors $w_0 \dots w_{n-1} u_i u_{i-1} \dots u_0$ est un chemin $M$-augmentant.

**Observation.** En marquant les arêtes $\{u, u_\rho\}$ par le sommet $u_i$ associé la transformation du chemin $M_{/\rho}$-augmentant s'effectue en temps linéaire.

# Recherche d'un chemin augmentant

**ChercheAug**$(G, M)$

$D \leftarrow$ **ConstDouble**$(G, M)$

$\rho \leftarrow$ **PlusCourtChemin**$(D, \mathring{V}_M, W_M)$

**If** $\rho = \emptyset$ **then Return**$(\emptyset)$

**Case** $\rho$ **in**

① **Return**$(\rho)$

② $\rho' \leftarrow$ **ChercheAug**$(G_{/\rho}, M_{/\rho})$

**If** $\rho' = \emptyset$ **then Return**$(\emptyset)$ **else Return**(**Develop**$(\rho', M_{/\rho}, \rho, M)$)

③ $M \leftarrow M \setminus \{\{u_{2\ell-1}, u_{2\ell}\}\}_{0<2\ell\leq i} \cup \{\{u_{2\ell}, u_{2\ell+1}\}\}_{0\leq 2\ell<i}$

$\rho' \leftarrow$ **ChercheAug**$(G_{/\rho}, M_{/\rho})$

**If** $\rho' = \emptyset$ **then Return**$(\emptyset)$ **else Return**(**Develop**$(\rho', M_{/\rho}, \rho, M)$)

**Complexité.** Au plus $\frac{|V|}{2}$ appels récursifs.

# Réduction de couplages

Soit $V$ un ensemble de sommets et $c : V \times V \to \mathbb{R}_+$ avec $c(v, u) = c(u, v)$.

Si $|V|$ est impair alors sans changer la résolution du problème :

- On ajoute un sommet $v_0$ à $V$ ;
- avec pour tout $v \in V$, $c(v, v_0) = 0$.

Le problème du couplage de poids maximal se réduit donc
au problème du couplage parfait de poids maximal.

Soit $C = \max(c(u, v) \mid u, v \in V)$.

On définit un problème de couplage parfait de poids minimal sur $V$ avec :

$$c'(u, v) = C - c(u, v)$$

Le problème du couplage parfait de poids maximal se réduit donc
au problème du couplage parfait de poids minimal et vice-versa.

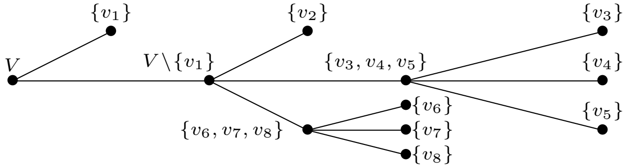
Résolu par Jack Edmonds en 1965 (John von Neumann Theory Prize 1985).

# Partitionnement récursif

Un*partitionnement récursif* de $V$ est un arbre étiqueté par la fonction $\lambda$ :

- tel que pour tout sommet $x$, $\lambda(x) \subseteq V$ ;
- dont la racine $r$ vérifie $\lambda(r) = V$ ;
- tel que pour tout sommet $x \neq r$, $|\lambda(x)|$ est impair ;
- et que pour sommet interne $x$ de fils $y_1, \ldots y_k$,
  on a $k \geq 2$ et $\lambda(x) = \biguplus_{i \leq k} \lambda(y_i)$.

**Illustration.**




Le partitionnement initial consiste en une racine
dont les fils sont des feuilles étiquetées par les singletons $\{v\}$ pour $v \in V$.

# Quelques propriétés

- $\lambda$ est injectif (évident). On note :
  - $\Omega = \{\lambda(x) \mid x \neq r\}$ ;
  - $\Omega^{\max} = \{\lambda(x) \mid x$ fils de $r\}$.
- $|\Omega| \leq 2|V|$.

**Preuve.** *(par récurrence sur $|V|$)*

Si $\Omega$ est l'ensemble des singletons alors $|\Omega| = |V|$.

Par conséquent, $|V| = 2$ implique $|\Omega| = |V|$.

Si $\Omega$ contient des ensembles non singletons.

Soit $W = \{v_1, \ldots, v_k\}$ (avec $k \geq 3$) un tel ensemble de cardinalité minimale.

$W$ a pour fils $\{v_1\}, \ldots, \{v_k\}$.

Considérons $V' = V \setminus \{v_1, \ldots, v_k\} \cup \{W\}$ et $\Omega' = \Omega \setminus \{\{v_1\}, \ldots, \{v_k\}\}$

$|\Omega'| \leq 2|V'| \Leftrightarrow |\Omega| - k \leq 2(|V| - k + 1) \Leftrightarrow |\Omega| \leq 2|V| - k + 2 \leq 2|V|$.

# Potentiel et partitionnement

Soit $U \subseteq V$, la frontière de $U$ est définie par $\delta(U) = \{\{u,v\} \mid u \in U \wedge v \notin U\}$.

La fonction $\pi : \Omega \to \mathbb{R}$ est un potentiel si :

- Pour tout $U \in \Omega$, $|U| \geq 3$ implique $\pi(U) \geq 0$ ;
- Pour tout $u, v \in V$, $\sum_{U \in \Omega \mid \{u,v\} \in \delta(U)} \pi(U) \leq c(u,v)$.

Le potentiel initial du partitionnement initial est nul.

**Observation.** Soit $M$ un couplage parfait alors pour tout $U \in \Omega$ et tout $u \in V$, $|U| \geq 3$ implique $|M \cap \delta(U)| \geq 1$ et $|M \cap \delta(\{u\})| = 1$.

Soit $M$ un ensemble de paires de $V$, $c(M) \stackrel{\text{def}}{=} \sum_{\{u,v\} \in M} c(u,v)$.

$$\boxed{\text{Pour tout couplage parfait } M\text{, tout } \Omega \text{ et } \pi : c(M) \geq \sum_{U \in \Omega} \pi(U).}$$

**Preuve.**

$$\begin{aligned}
c(M) &\geq \sum_{\{u,v\} \in M} \sum_{U \in \Omega, \{u,v\} \in \delta(U)} \pi(U) \text{ \textit{(par définition de } \pi\text{)}} \\
&= \sum_{U \in \Omega} \pi(U) |M \cap \delta(U)| \text{ \textit{(par inversion des sommes)}} \\
&\geq \sum_{U \in \Omega} \pi(U) \text{ \textit{(par décomposition entre les singletons et les autres ensembles)}}
\end{aligned}$$

$$\boxed{\text{S'il existe } M, \Omega \text{ et } \pi \text{ tels que } c(M) = \sum_{U \in \Omega} \pi(U) \text{ alors } M \text{ a un poids minimal.}}$$

# Poids relatif au potentiel

Le poids relatif d'une paire $\{u, v\}$ est défini par :

$$c_\pi(u, v) \stackrel{\text{def}}{=} c(u, v) - \sum_{U \in \Omega \mid \{u,v\} \in \delta(U)} \pi(U) \geq 0$$

Soit $x$ un sommet interne de $\Omega$ de fils $\{y_0, \dots y_{k-1}\}$. Pour chaque $0 \leq i \neq j < k$,

$$c_\pi(y_i, y_j) \stackrel{\text{def}}{=} \{c_\pi(v_i, v_j) \mid v_i \in \lambda(y_i) \wedge v_j \in \lambda(y_j)\}$$

L'algorithme maintient pour tout $x \neq r$, sommet interne,
une énumération des fils $(y_{\alpha(0)}, \dots y_{\alpha(k-1)})$ telle que pour tout $0 \leq i < k$,

$$0 \in c_\pi(y_{\alpha(i)}, y_{\alpha(i+1\%k)})$$

C'est vrai initialement car le seul sommet interne est la racine.

# Couplage et forêt

L'algorithme maintient un couplage $M$ entre les fils de la racine $\{y_0, \ldots, y_{k-1}\}$.

$M$ est initialement vide.

L'algorithme maintient une $M$-*forêt* $F$ sur les sommets $\{y_0, \ldots, y_{k-1}\}$ qui vérifie :

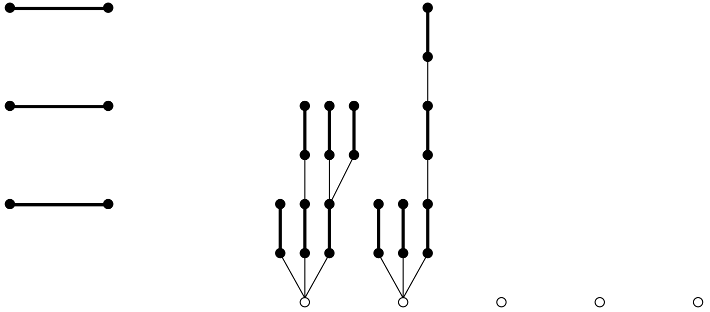
- Pour toute arête $\{y_i, y_j\} \in F$, $0 \in c_\pi(y_i, y_j)$ ;
- Pour tout $\{y_i, y_j\} \in M$, $\{y_i, y_j\} \in F$ ;
- Les composantes connexes de $F$ sont soit :
  - une paire de $M$ ;
  - un arbre dont un seul sommet (la racine) n'est pas couvert par $M$ ;
  - et chaque branche alterne des arêtes de $M$ et hors de $M$.

$F$ est initialement constituée des $k$ arbres singletons $\{y_0, \ldots, y_{k-1}\}$.

$Pair(F)$ est l'ensemble des sommets des arbres à distance paire de la racine.

$Impair(F)$ est l'ensemble des sommets des arbres à distance impaire de la racine.

# Illustration

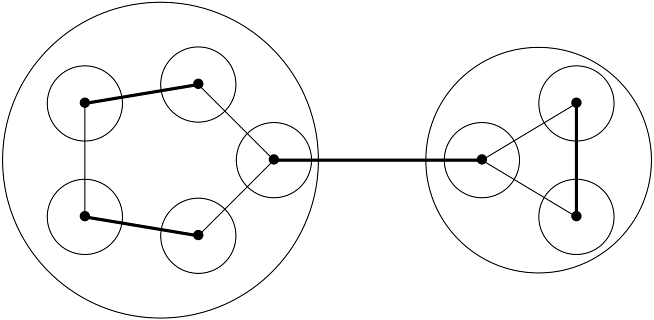


En résumé, l'algorithme maintient :

- $\Omega$, une partition récursive de $V$ initialement définie par les singletons ;
- $\pi$, un potentiel sur $\Omega$ initialement nul ;
- Pour tout $U \in \Omega$ avec $|U| > 1$
  une énumération $U = (y_0, \ldots, y_{k-1})$ telle que $0 \in c_\pi(y_i, y_{i+1\%k})$ ;
- $M$, un couplage sur $\Omega^{\max}$ initialement vide ;
- $F$, une $M$-forêt sur $\Omega^{\max}$ initialement constituée des arbres-singletons.

# Terminaison de l'algorithme

L'algorithme se termine lorsque $M$ est un couplage parfait de $\Omega^{\max}$
qu'on complète en un couplage parfait de $V$
en descendant le long de $\Omega$ de la façon suivante :



$M$ est un couplage de poids minimal car :

- toutes les paires $\{u, v\}$ de $M$ vérifient $c_\pi(u, v) = 0$ ;
- Pour tout $x \neq r$, $|\delta(\lambda(x)) \cap M| = 1$.

# Une itération de l'algorithme

**Première étape.** Mise à jour de $\pi$ :

- Pour tout $x \in Impair(F)$, $\pi(x) \leftarrow \pi(x) - \alpha$ ;
- Pour tout $x \in Pair(F)$, $\pi(x) \leftarrow \pi(x) + \alpha$ ;
- avec $\alpha$ la plus grande valeur possible qui garantit que $\pi$ soit un potentiel.

$\alpha$ est bien définie puisque :

- $\sum_{x \in \Omega} \pi(x) \leq c(N)$ avec $N$ couplage parfait quelconque ;
- $|Pair(F)| > |Impair(F)|$.

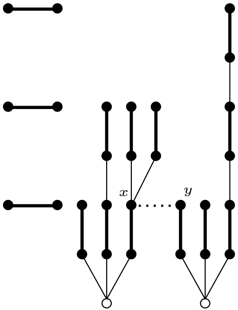
$\alpha$ est la plus petite valeur parmi :

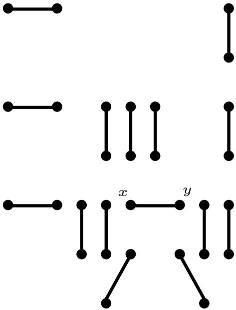
- $\min(\frac{c_\pi(u,v)}{2} \mid \exists x, y \in Pair(F)\ u \in \lambda(x) \wedge v \in \lambda(y))$ ;
- $\min(c_\pi(u,v) \mid \exists x \in Pair(F)\ \exists y \in \Omega^{\max} \setminus (Pair(F) \cup Impair(F))\ u \in \lambda(x) \wedge v \in \lambda(y))$.
- $\min(\pi(x) \mid |\lambda(x)| \geq 3 \wedge x \in Impair(F))$ ;

**Deuxième étape.** Mise à jour des autres structures de données selon le choix de $\alpha$.

# Cas n°1 : chemin augmentant

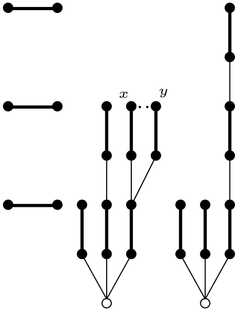
$\exists u, v\ \exists x, y \in Pair(F)\ u \in \lambda(x) \wedge v \in \lambda(y) \wedge c_\pi(u, v) = 0 \wedge x, y$ non connectés

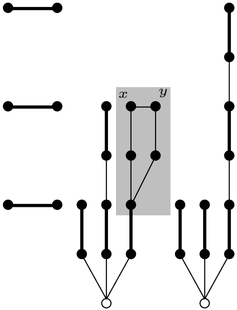


# Cas n°2 : création d'un sommet de $\Omega$

$\exists u, v\ \exists x, y \in Pair(F)\ u \in \lambda(x) \wedge v \in \lambda(y) \wedge c_\pi(u, v) = 0 \wedge x, y$ connectés



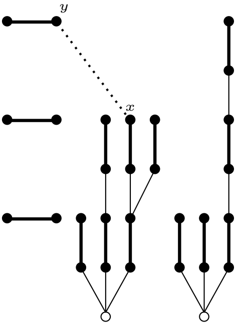



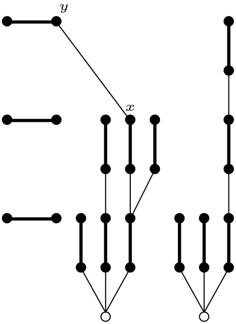

Le potentiel de ce nouveau sommet est nul.

**Observation.** Le nouveau sommet appartient à $Pair(F)$.

# Cas n°3 : extension d'arbre

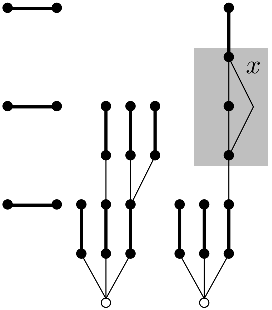
$$\exists u, v\ \exists x \in Pair(F)\ \exists y \notin Pair(F) \cup Impair(F)\ u \in \lambda(x) \wedge v \in \lambda(y) \wedge c_\pi(u,v) = 0$$

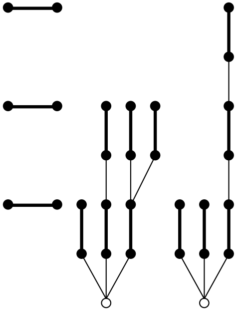


# Cas n°4 : disparition d'un sommet de $\Omega$

$$\exists x \in Impair(F)\ \pi(x) = 0 \wedge |\lambda(x)| \geq 3$$



# Terminaison et complexité

- Le nombre de racines des arbres i.e. $|\{x \in \Omega^{\max} \mid \nexists\{u,v\} \in M \wedge u \in \lambda(x)\}|$ décroît de deux unités pour le cas n° 1 et reste constant pour les autres cas. Il y a donc au plus $\frac{|V|}{2}$ occurrences de cas n° 1.

- Soit $V_{pair} = \{v \in V \mid \exists x \in Pair(F) \wedge v \in \lambda(x)\}$ et $\Omega_0 = \Omega^{\max} \setminus Pair(F)$. Considérons la quantité $q = 2|V_{pair}| + |\Omega_0|$ :
  - Après une occurrence du cas n° 2, $q$ croît d'au moins le nombre de sommets de $Impair(F)$ qui ont disparu.
  - Après une occurrence du cas n° 3, $q$ croît d'au moins une unité.
  - Après une occurrence du cas n° 4, $q$ croît d'au moins une unité.

  Il y a donc au plus $2|V|$ occurrences des cas n° 2, 3 et 4 entre deux occurrences du cas n° 1 car $|V_{pair}| + |\Omega_0| \leq |V|$.

$$\boxed{\text{Cet algorithme est polynomial.}}$$

# Le voyageur de commerce revisité

① Calcul d'un arbre couvrant de poids minimal $\mathcal{T}$

Soit $V_{odd}$, l'ensemble des sommets de degré impair de $\mathcal{T}$, $|V_{odd}|$ est pair.

② Calcul d'un couplage parfait de poids minimal $\mathcal{M}$ pour $V_{odd}$

Soit le multi-graphe sur $V = \{v_1, \ldots, v_n\}$ défini par $\mathcal{T} \cup \mathcal{M}$ :

- Ce multi-graphe est connexe ;
- Tous les sommets ont un degré pair.

③ Calcul (en temps polynomial) d'un cycle eulérien
$\mathcal{C} = (v_{\alpha(1)} \ldots v_{\alpha(2)} \ldots v_{\alpha(n)} \ldots v_{\alpha(1)})$
où $\alpha$ est la permutation associée à la première occurrence des sommets.

④ L'algorithme renvoie $\mathcal{E} = (v_{\alpha(1)} v_{\alpha(2)} \ldots v_{\alpha(n)} v_{\alpha(1)})$

# Garantie de performance

Par l'inégalité triangulaire,

$$c(\mathcal{E}) \leq c(\mathcal{C}) = c(\mathcal{T}) + c(\mathcal{M})$$

Soit $sol^*$ une solution optimale. On a déjà montré que :

$$c(sol^*) \geq c(\mathcal{T})$$

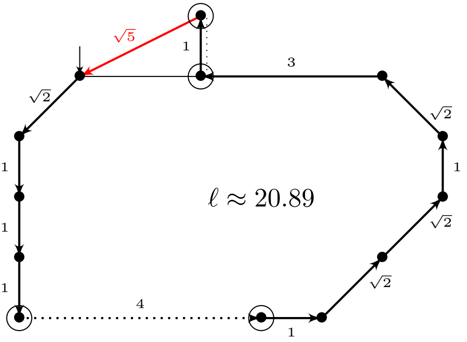
Posons $sol^* = (v_0 \dots v_1 \dots v_{2m-1} \dots v_0)$ où $V_{odd} = \{v_0, v_1, \dots, v_{2m-1}\}$.
Par l'inégalité triangulaire,

$$c(sol^*) \geq \sum_{i<m} d(v_{2i}, v_{2i+1}) + \sum_{i<m} d(v_{2i+1}, v_{2i+2\%2m}) \geq 2c(\mathcal{M})$$

D'où :

$$\boxed{\frac{c(\mathcal{E})}{c(sol^*)} \leq \frac{3}{2}}$$

# Illustration



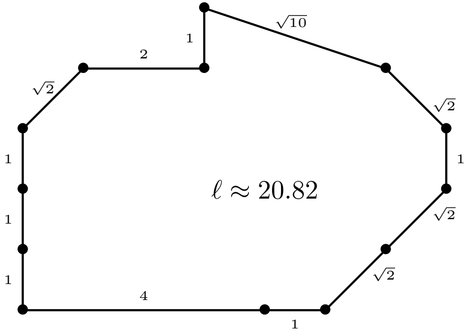

En plein noir, l'arbre couvrant

En pointillés, le couplage

En gras, le parcours du voyageur de commerce

En rouge, le raccourci effectué par le voyageur de commerce

# La tournée minimale



# Plan

La couverture de sommets

Le voyageur de commerce

Différents couplages

4 Le sac à dos

# Les problèmes de sac à dos

**Un problème de décision.**

Soit $I = \{1, \ldots, n\}$, $(v_i)_{i \in I} \in \mathbb{N}^I$ et $w \in \mathbb{N}$.

$$\text{Existe-il } J \subseteq I \text{ tel que } \textstyle\sum_{i \in J} v_i = w\,?$$

Ce problème est NP-complet.

**Un problème de maximisation.**

Soit $I = \{1, \ldots, n\}$, $(v_i)_{i \in I} \in \mathbb{N}^I$ et $w \in \mathbb{N}$.

$$\text{Renvoyer } J^* \in \arg\max(\textstyle\sum_{i \in J} v_i \mid J \subseteq I \wedge \sum_{i \in J} v_i \leq w)$$

Sauf si P=NP,

il n'existe pas d'algorithme en temps polynomial pour résoudre ce problème.

# Un algorithme exponentiel

$L \leftarrow \{(\emptyset, 0)\}$
**For** $i$ **from** $1$ **to** $n$ **do**
  $L' \leftarrow \emptyset$
  **For** $(J, v) \in L$ **do**
    $J \leftarrow J \cup \{i\}$
    $v_J \leftarrow v + v_i$
    **If** $v_J \leq w$ **then** $L' \leftarrow L' \cdot (J, v_J)$
  $L \leftarrow \textbf{Fusion}(L, L')$
**Return**$(\textbf{Last}(L))$

Par induction, $L$ et $L'$ sont triées par valeurs croissantes.

Chaque itération est linéaire en la taille de $L$.

**Idée** : Conserver une taille polynomiale.

# Un algorithme avec épuration

$L \leftarrow \{(\emptyset, 0)\}$
**For** $i$ **from** 1 **to** $n$ **do**
  $L' \leftarrow \emptyset$
  **For** $J \in L$ **do**
    $J \leftarrow J \cup \{i\}$
    $v_J \leftarrow \sum_{i \in J} v_i$
    **If** $v_J \leq w$ **then** $L \leftarrow L' \cdot (J, v_j)$
  $L \leftarrow \mathbf{Fusion}(L, L')$
  $L \leftarrow \mathbf{Epuration}(L, \delta)$
**Return**$(\mathbf{Last}(L))$

L'épuration consiste à éliminer un élément $(J, v_J)$ si $v_J < (1+\delta)v'$

où $v'$ est la valeur de l'élément précédent.

Soit $k$ la taille de la liste alors $(1+\delta)^{k-2} \leq w$.

Par conséquent $k \leq 2 + \frac{\log(w)}{\log(1+\delta)}$.

# Choix de $\delta$

Soit $\varepsilon \leq 1$. Alors $\delta = \frac{\varepsilon}{2n}$ garantit une performance de $1+\varepsilon$.

**Preuve.** Par récurrence sur $i$ démontrons qu'à la fin de l'itération $i$, pour tout $J \subseteq \{1, \dots, i\}$ avec $v_J \leq w$, il existe un $(J', v_{J'}) \in L$ tel que $\frac{v_J}{(1+\frac{\varepsilon}{2n})^i} \leq v_{J'} \leq v_J$.

Soit $J \subseteq \{1, \dots, i+1\}$ avec $v_J \leq w$ et $J'' = J \setminus \{i+1\}$.
Il existe $(J^*, v_{J^*}) \in L$ à la fin de l'itération $i$ tel que $\frac{v_{J''}}{(1+\frac{\varepsilon}{2n})^i} \leq v_{J^*} \leq v_{J''}$.

Soit $J^+ = J^* \cup \{i+1\}$ si $i+1 \in J$ et $J^+ = J^*$ sinon.

**Cas $J^+ = J^* \cup \{i+1\}$.**

$$\frac{v_J}{(1+\frac{\varepsilon}{2n})^i} = \frac{v_{J''}+v_{i+1}}{(1+\frac{\varepsilon}{2n})^i} \leq \frac{v_{J''}}{(1+\frac{\varepsilon}{2n})^i} + v_{i+1} \leq v_{J^*} + v_{i+1} = v_{J^+} \leq v_{J''} + v_{i+1} = v_J$$

**Cas $J^+ = J^*$.** L'encadrement ci-dessus est l'hypothèse de récurrence.
Après épuration il existe un $J'$ tel que : $\frac{v_J}{(1+\frac{\varepsilon}{2n})^{i+1}} \leq \frac{v_{J^+}}{(1+\frac{\varepsilon}{2n})} \leq v_{J'} \leq v_{J^+} \leq v_J$.

Pour tout $J \subseteq I$ tel que $v_J \leq w$, il existe $(J', v_{J'}) \in L$ tel que $\frac{v_J}{(1+\frac{\varepsilon}{2n})^n} \leq v_{J'} \leq v_J$.

Or $n \log(1+\frac{\varepsilon}{2n}) \leq \frac{\varepsilon}{2}$. Donc $(1+\frac{\varepsilon}{2n})^n \leq e^{\frac{\varepsilon}{2}}$. Pour $\varepsilon \leq 1$, $e^{\frac{\varepsilon}{2}} \leq 1+\varepsilon$.

# Complexité de l'algorithme

L'algorithme a une complexité en $\Theta(n|L|)$.

$$\begin{aligned}
|L| &\leq \left\lfloor \frac{\log(w)}{\log(1+\frac{\varepsilon}{2n})} \right\rfloor + 2 \\
&\leq \frac{\log(w)}{\log(1+\frac{\varepsilon}{2n})} + 2 \\
&\leq \frac{2n(1+\frac{\varepsilon}{2n})\log(w)}{\varepsilon} + 2 \text{ (car } \log(1+x) \geq \tfrac{x}{1+x} \text{ pour } x \geq 0) \\
&\leq \frac{4n\log(w)}{\varepsilon} + 2 \text{ (dès que } \varepsilon \leq 1)
\end{aligned}$$

On a donc affaire à un schéma d'approximation complètement polynomial.