Problème du voyageur de commerce

Un représentant de commerce ou VRP souhaite faire une tournée en visitant n villes, chacune exactement une fois, et en terminant sa tournée dans la ville de départ.

World Challenge : 1.904.711 villes
(meilleure solution en 2009 : $7,5.10^6$km (LKH))

Problème du voyageur de commerce symétrique

Définition

Soit $G = (S, A)$ *un graphe non-orienté simple et complet pondéré par une* distance $c(u, v)$

On a $|S| = n$ *et* $|A| = \frac{n(n-1)}{2}$.

Cycle hamiltonien *: cycle élémentaire de* G *qui contient chaque sommet de* S.

Objectif : Trouver un cycle hamiltonien ou tour *de longueur minimum.*

Le problème du voyageur de commerce est NP-difficile.

Remarque : tester l'existence d'un cycle hamiltonien dans un graphe quelconque est NP-complet

Remarque : tester l'existence d'un *cycle eulerien* (cycle simple de G qui contient chaque arête de A) est polynômial (sommets tous de degré pair)

Algorithme de séparation et évaluation

Fonction *Branch-and-Bound(VRP* : (G, c)*)*

Construire un tour H approché;
$U \leftarrow c(H)$ /* majorant initial */;
$Q \leftarrow \{G\}$;
tant que $Q \neq \varnothing$ **faire**
  $G_i \leftarrow$ Extrait(Q);
  /* Résout une relaxation du sous-problème G_i */;
  $L_i =$ optimum d'une relaxation de G_i;
  if $L_i \geq U$ **then continue** /* Abandon de G_i */;
  if $L_i < U$ *et la solution relaxée est un tour* **then**
    $U \leftarrow L_i$;
  /* Règle de branchement (partition de problèmes plus contraints en restreignant les arêtes possibles de A_i) */;
  $Q \leftarrow \{G_{i_1}, \ldots, G_{i_q}\}$;
retourner *optimum* U;

Relaxation : problème d'affectation

Définition

Soit $(G = (S, A), c)$ *un problème VRP de longueur minimum* V^*.

Le problème d'affectation pour le graphe biparti $S_1 = S$ *et* $S_2 = S$ *pondéré par* c $(c(u, u) = +\infty)$ *a une solution optimale composée uniquement de cycles. Son optimum est donc toujours inférieur ou égal à* V^*. *C'est une (bonne)* relaxation *du VRP.*

Il y a $n!$ solutions au problème d'affectation et $(n-1)!$ sont des tours.

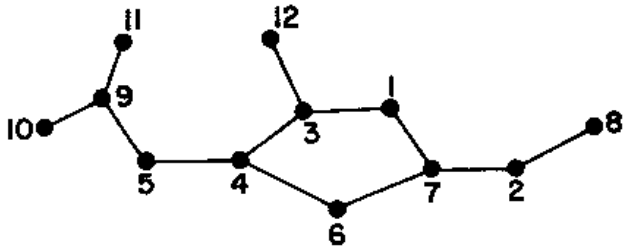
Relaxation d'Held-Karp

Propriété

Le problème d'arbre couvrant de longueur minimum plus une arête supplémentaire de longueur minimum est une (mauvaise) relaxation du VRP.

Définition

Soit $(G = (S, A), c)$ *un VRP et* $u \in A$. *Le problème d'arbre couvrant de longueur minimum sur* $S \setminus \{u\}$ *plus deux arêtes de longueur minimum entre* u *et* $S \setminus \{u\}$ *est appelé* 1−arbre. *C'est aussi une relaxation du VRP.*



Relaxation d'Held-Karp

Idée :ajout pénalité λ_u sur chaque sommet en fonction de son degré

Propriété

Le problème de maximiser selon les paramètres λ un $1-$arbre T de longueur minimum $\sum_{(u,v)\in T} c(u,v) + \lambda_u + \lambda_v$ est une (bonne, meilleure que le problème d'affectation) relaxation du VRP.

Procedure *Subgradient-Optimization*(G, c, U)

$\lambda \leftarrow 0$; $\alpha \leftarrow 2$; $k \leftarrow 0$;

répéter

Trouve un $1-$arbre T de longueur modifiée (λ) minimum ;

pour chaque $u \in S$ **faire**

$\lambda_u \leftarrow \lambda_u + \alpha.(degre(u) - 2).\frac{(U - c(T))}{\sum_{v\in S}(degre(u)-2)^2}$;

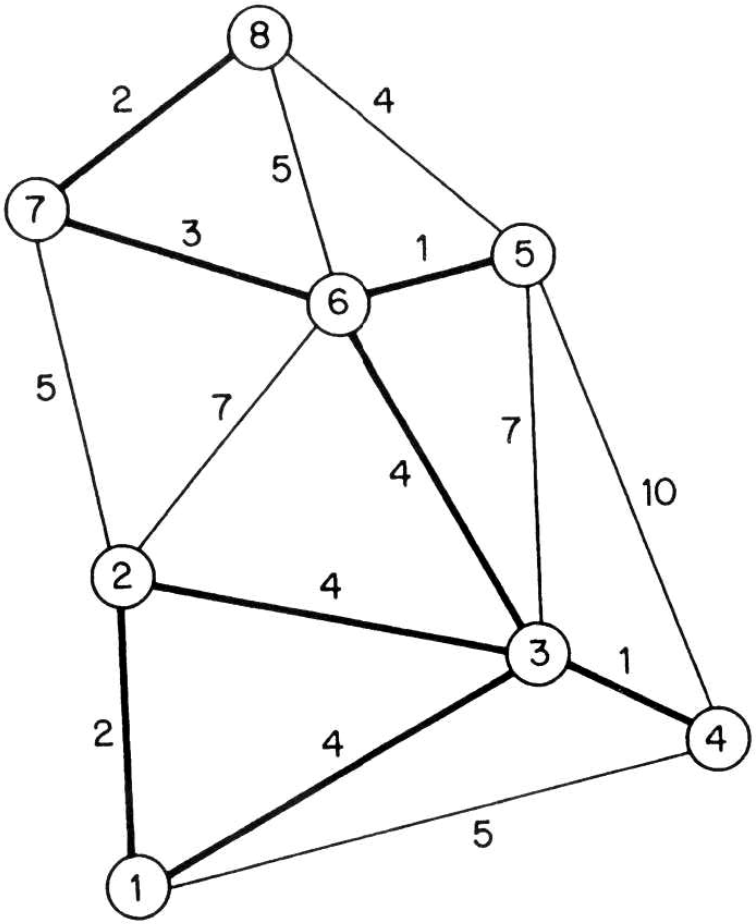
$k \leftarrow k + 1$;

if k `modulo` $Periode = 0$ **then** $\alpha \leftarrow \alpha - \epsilon$;

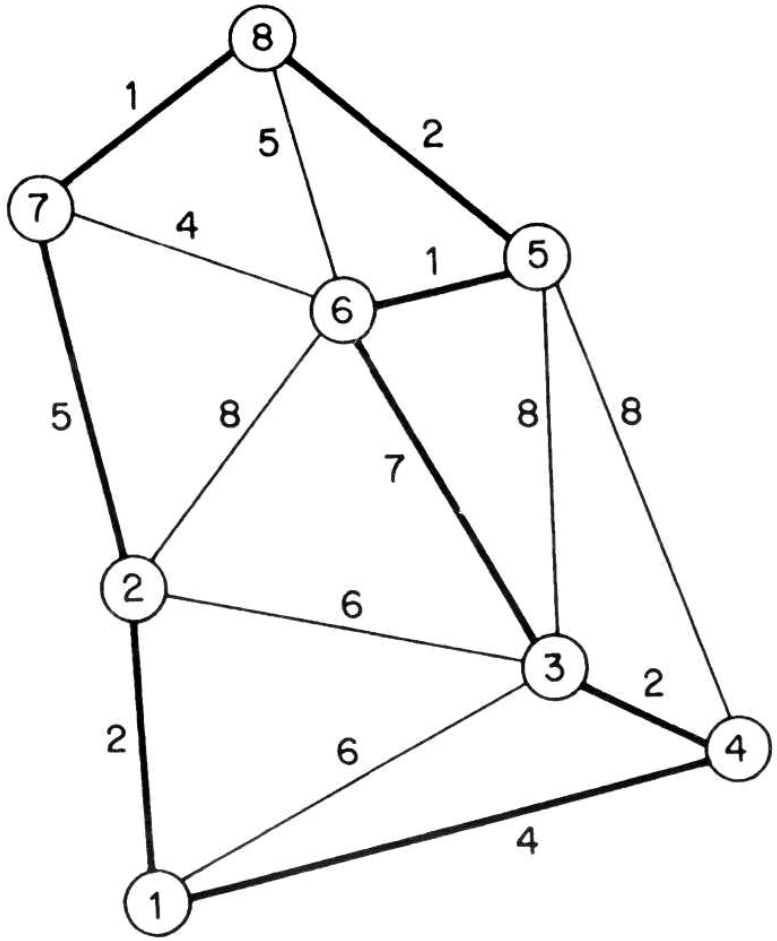
jusqu'à $k < k_{max}$ *ou* T *est un tour ou* $c(T) \geq U$;

Relaxation d'Held-Karp

1−arbre initial (a) et 1−arbre amélioré (b) :



(a) $L = 21$ $(U = 25, \lambda = 0, \alpha = 2)$



(b) $L = 24$ $(\lambda = (0, 0, 2, -1, -1, 1, 0, -1))$

Algorithme d'approximation

Objectif : construire un tour approché.

Fonction *Approximation-Tour-VRP(* G, c *)*

Choix d'une racine $r \in S$;

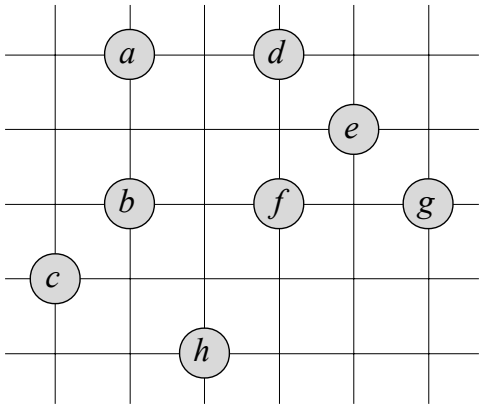
Construire un arbre couvrant de longueur minimum T avec MST-Prim(G, c, r);

$L \leftarrow$ Tri-Topologique(T, r), liste des sommets visités lors d'un parcours préfixe (en profondeur d'abord) de T;

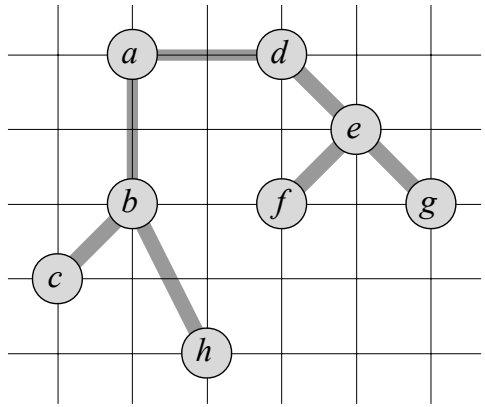
retourner *cycle hamiltonien visitant les sommets dans l'ordre L*

Complexité en $O(n^2)$

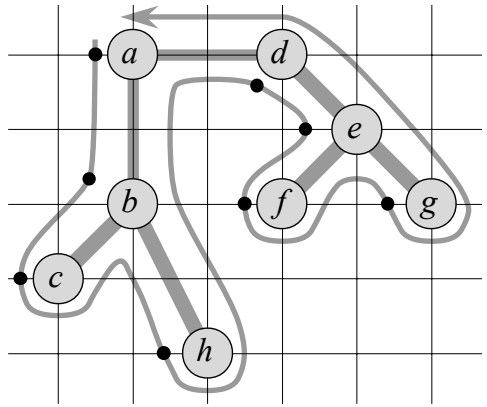
Algorithme d'approximation



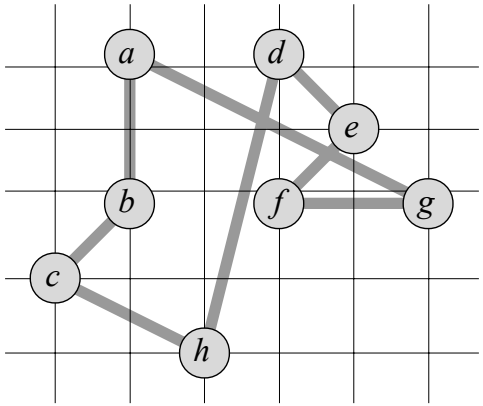
(a)



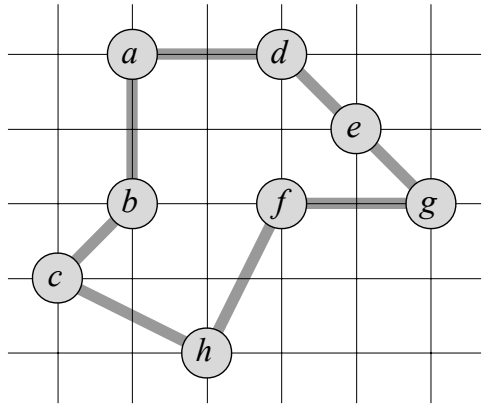
(b)



(c)



(d)



(e)

Algorithme d'approximation

Définition

Un algorithme d'approximation *est un algorithme qui retourne une solution presque optimale à un problème donné.*

Un algorithme d'approximation a une borne $\rho(n)$ *si pour tout problème de taille n, la solution produite a une valeur V éloignée d'un facteur* $\rho(n)$ *de celle* V^* *d'une solution optimale :*
$\max(\frac{V}{V^*}, \frac{V^*}{V}) \leq \rho(n)$

Théorème

*L'algorithme Approximation-Tour-VRP est un algorithme d'approximation ayant une borne égale à 2 pour un problème vérifiant l'*inégalité triangulaire $(c(u, v) \leq c(u, w) + c(w, v))$.

Preuve de la borne

Soit H^* une solution optimale et H celle retournée par l'algorithme.

Preuve : on veut prouver que $c(H) \leq 2.c(H^*)$.

En supprimant une arête de H^*, on obtient un arbre couvrant.
Donc $c(T) \leq c(H^*)$ pour tout arbre couvrant minimum T.

Soit un *parcours complet* de T : un chemin W visitant deux fois chaque arête de T (cf. chemin de la Figure (c) précédente).
On a $c(W) = 2.c(T)$. Donc $c(W) \leq 2.c(H^*)$.

Il est possible d'extraire un cycle hamiltonien de W en supprimant les sommets déjà visités dans l'ordre W et sans augmenter la longueur totale grâce à l'inégalité triangulaire. On obtient le cycle H retourné par l'algorithme. On en déduit $c(H) \leq c(W) \leq 2.c(H^*)$.

Algorithme d'approximation de Christofides

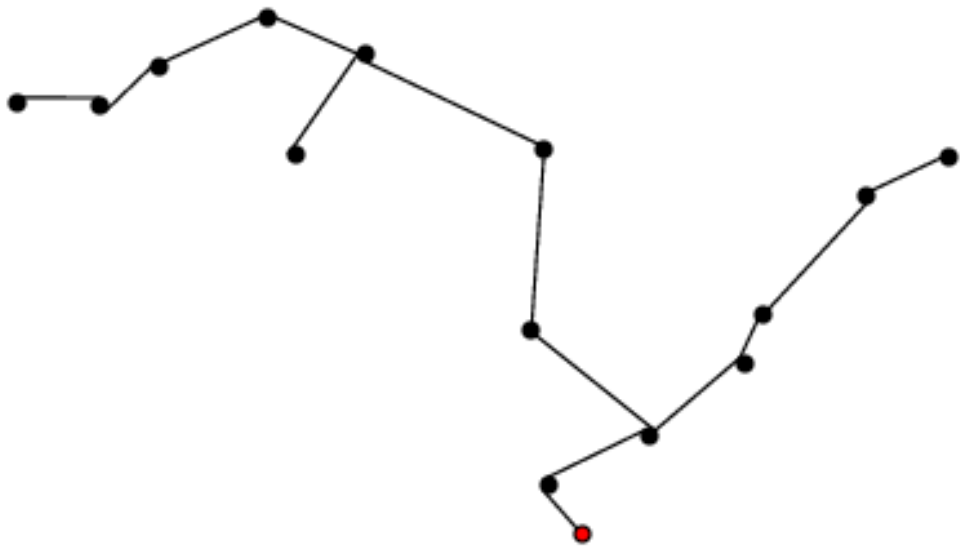
Idée : construire un graphe de degrés tous pairs, extraire un cycle eulerien, puis à l'aide de l'inégalité triangulaire en extraire un cycle hamiltonien.

Fonction *Christofides*(G, c)

Choix d'une racine $r \in S$;
Construire un arbre couvrant de longueur minimum $T = (S, A')$ avec MST-Prim(G, c, r);
Construire un couplage parfait M de longueur minimum pour les m sommets de degré impair de T (m pair);
Extraire un cycle eulerien E du graphe $G' = (S, A' \cup M)$ (degrés tous pairs);
retourner *cycle hamiltonien visitant les sommets (la première fois) dans l'ordre E*

Complexité en $O(n^3)$

Algorithme d'approximation de Christofides

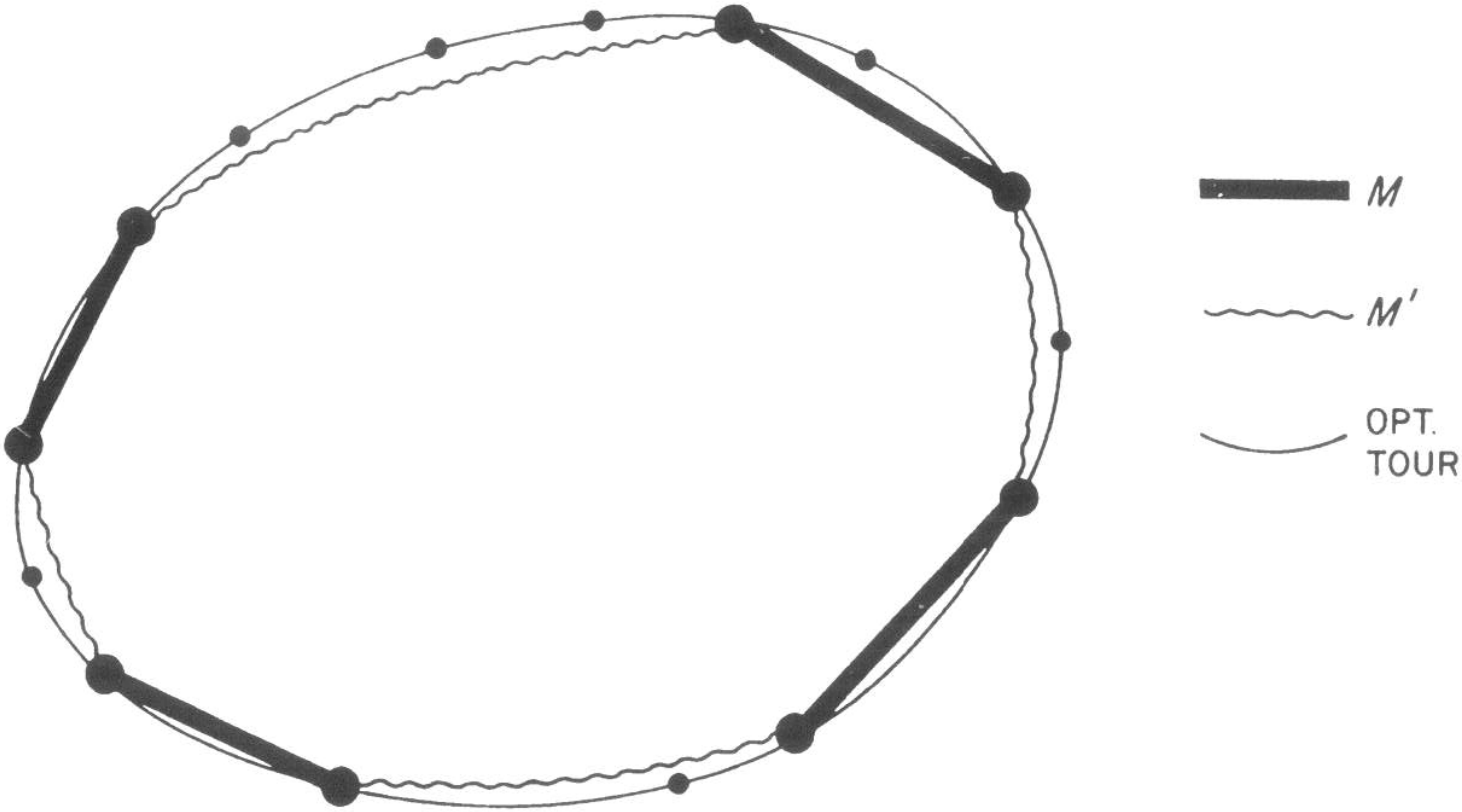


Heuristique de Christofides (1976)

Preuve de la borne

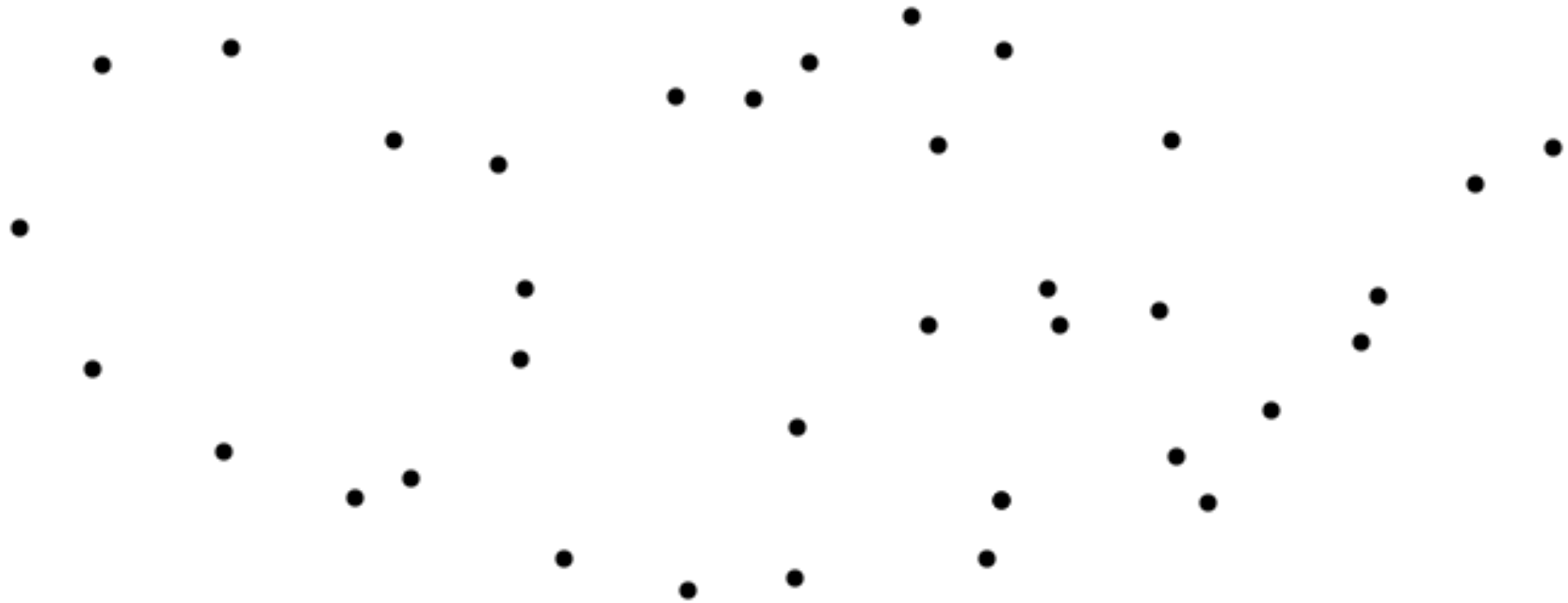
L'algorithme Christofides a une borne $\rho(n) = \frac{3}{2}$

On a $c(T) \leq c(H^*)$

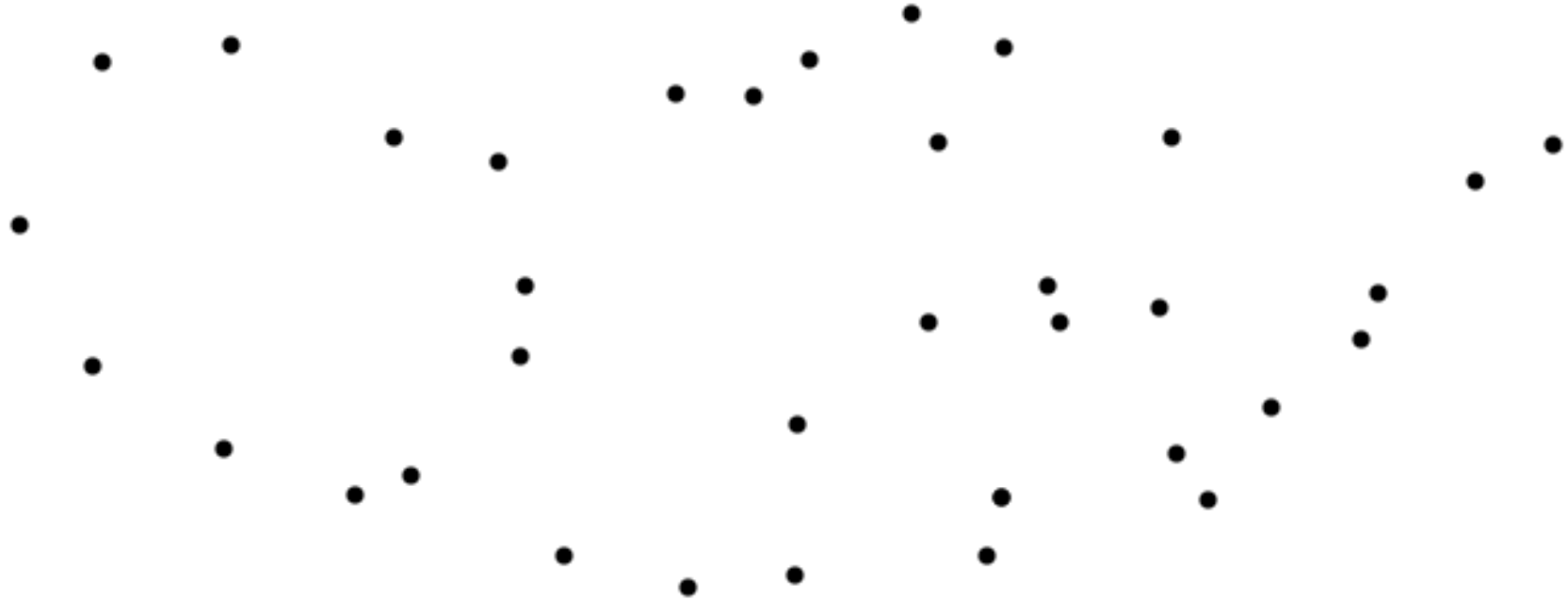


Inégalité triangulaire $c(M) + c(M') \leq c(H^*) \implies \min c(M), c(M') \leq \frac{1}{2} c(H^*)$

Algorithme d'approximation du plus proche voisin

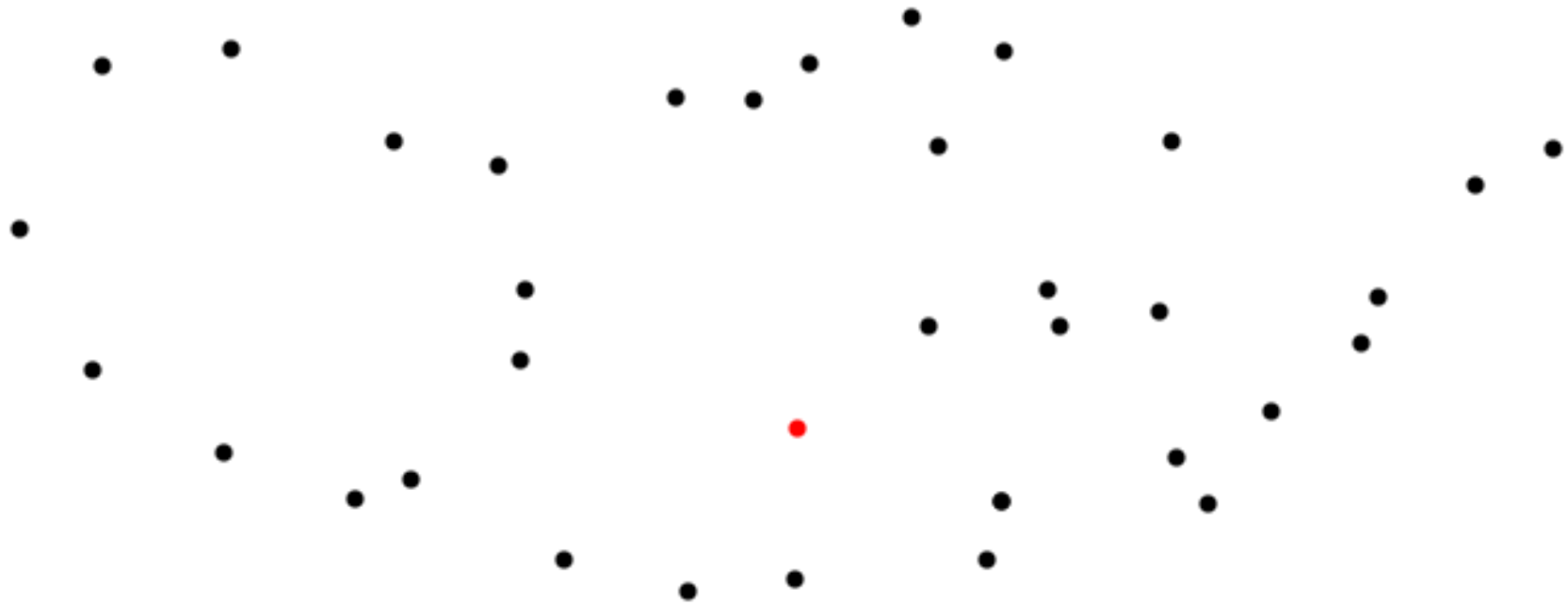


Algorithme d'approximation de la plus petite distance

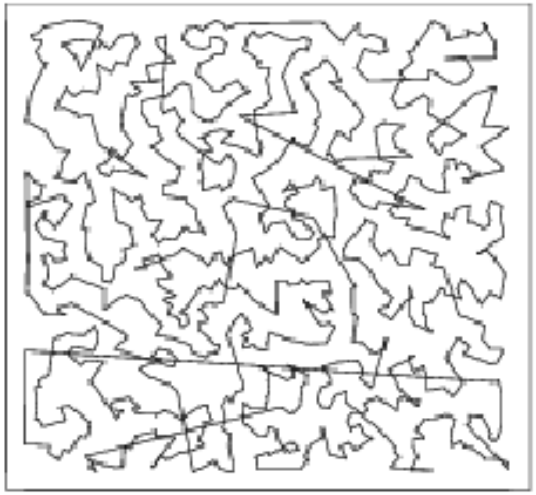


Algorithme d'approximation de Clarke-Wright

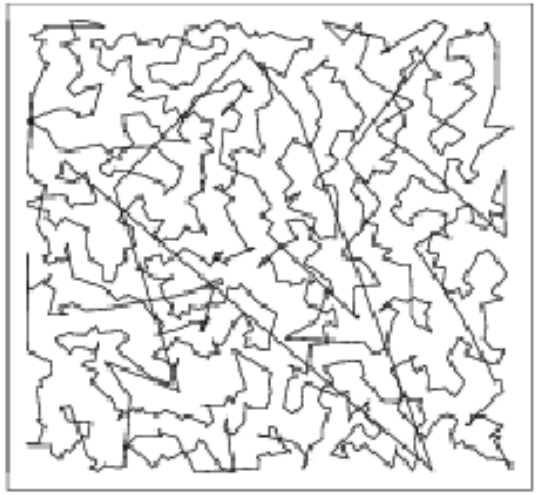
Choix hub h et sélection paire (u,v) maximisant $c(u,h)+c(h,v)-c(u,v)$



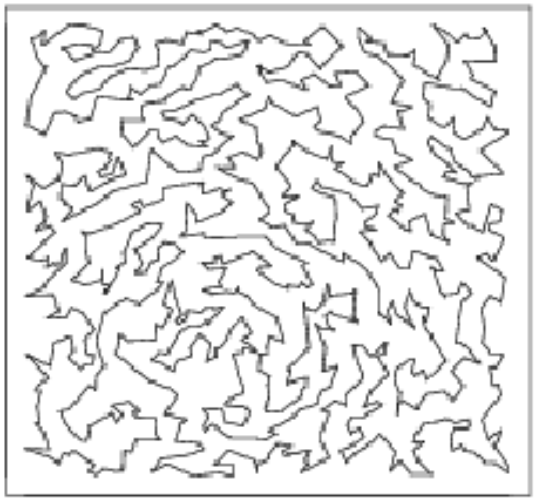
Algorithmes d'approximations



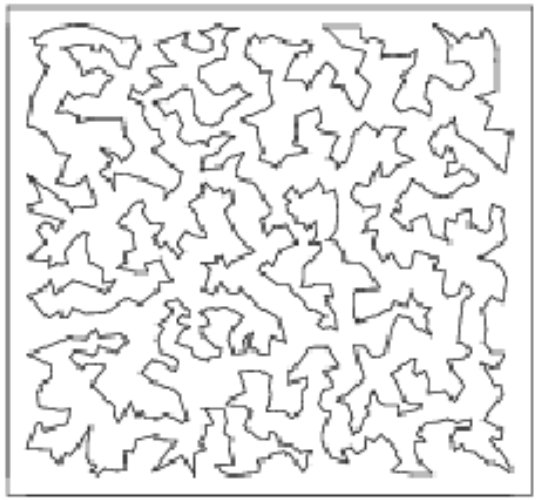
Greedy Tour



Nearest Neighbor Tour

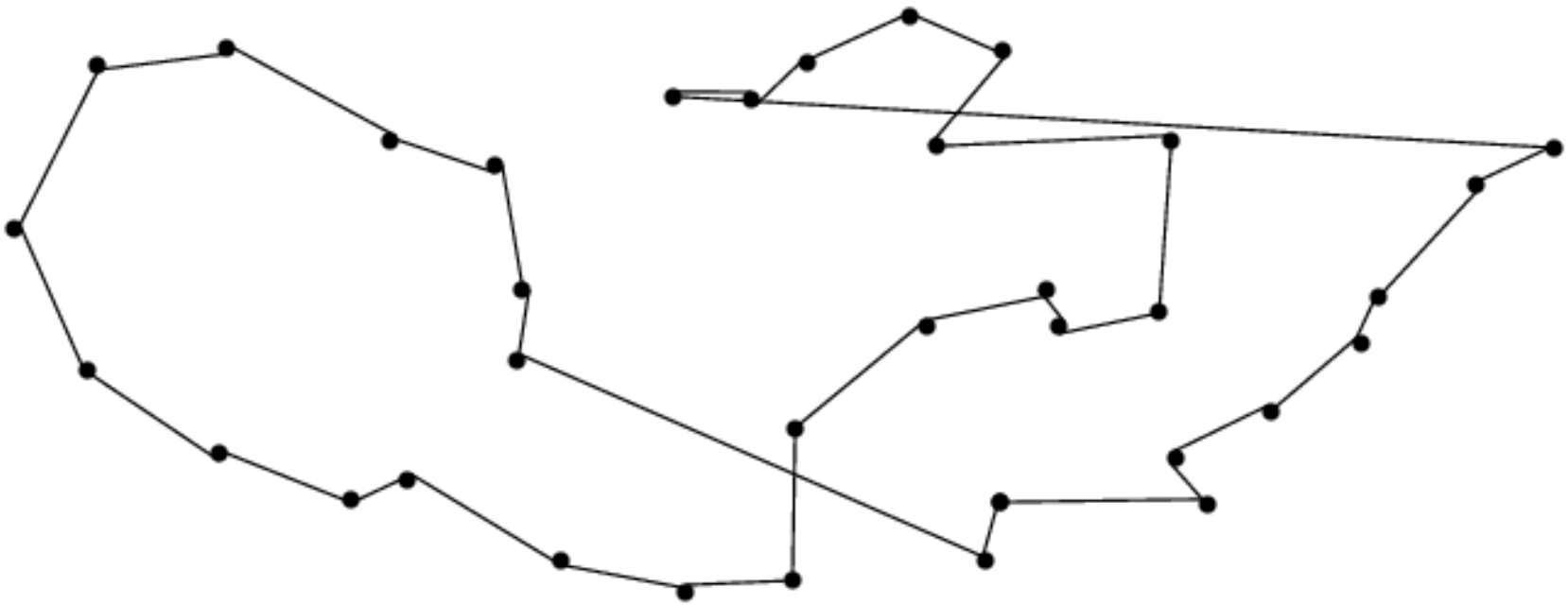


Savings Tour



Optimal Tour

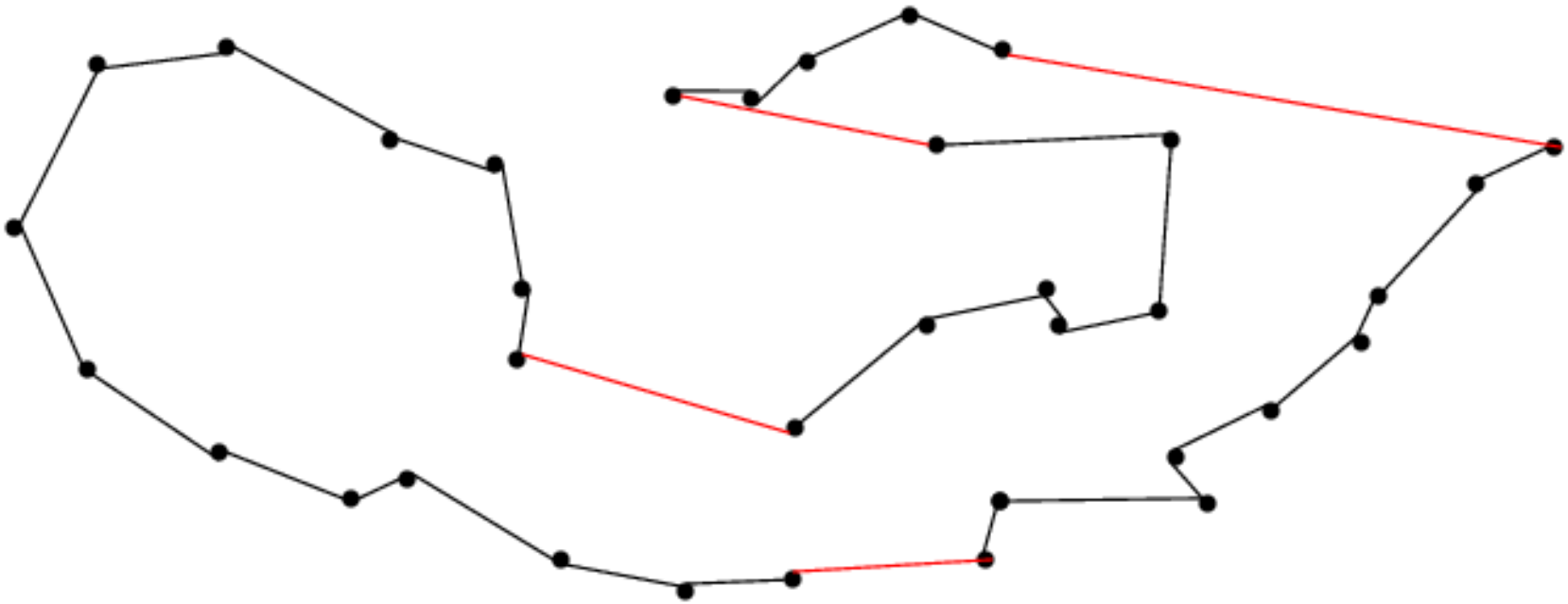
Recherche locale 2-opt



Quelle modification locale peut-on faire pour améliorer ce tour ?

Recherche locale 2-opt

Idée : échanger les sommets de deux arêtes du tour et répéter l'opération tant que cela améliore le tour.



Qualité des algorithmes d'approximation

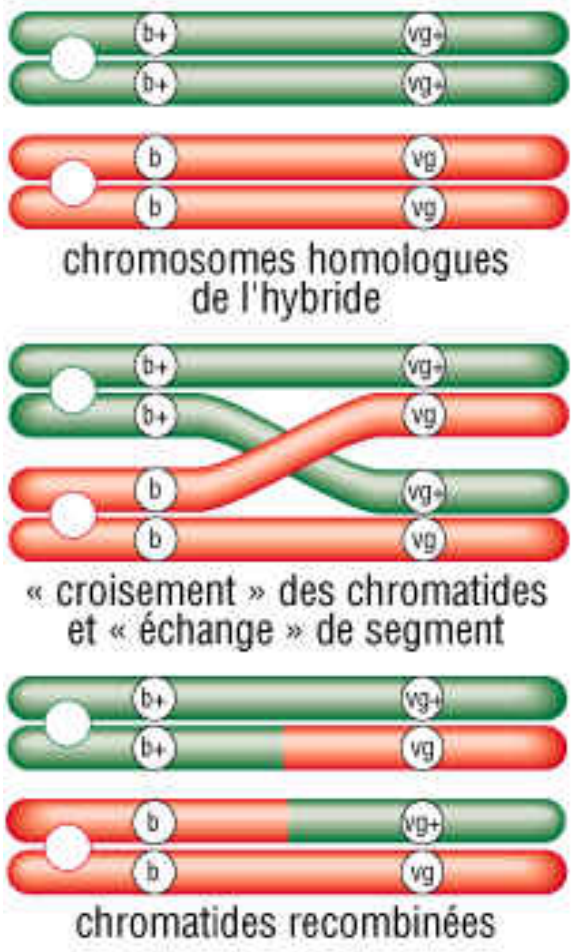
Distances moyennes à la borne inférieure fournie par Held-Karp

- Plus proche voisin : 26%
- Plus petite distance : 17%
- Clarke-Wright : $11,3\%$
- Christofides : $9,7\%$
- 2-opt : $4,9\%$
- 3-opt : $3,1\%$
- LKH (k-opt bridé) : 2%

(distances euclidiennes aléatoires pour n=1000 villes)

Exercice : cartographie génétique

On dispose de balises M (*marqueurs génétiques*) sur les chromosomes d'un organisme donné. On établit un protocole expérimental mesurant la recombinaison génétique sur les descendants de deux grands-parents homozygotes (les balises de la mère ont la valeur 0 et celles du père la valeur 0 ou 1). Sachant que la recombinaison génétique est un événement rare, proposer une formulation du problème de localisation des balises sur les chromosomes comme un VRP. Comment trouver $M' \subseteq M$ tel qu'il n'existe pas de carte aussi vraisemblable ?



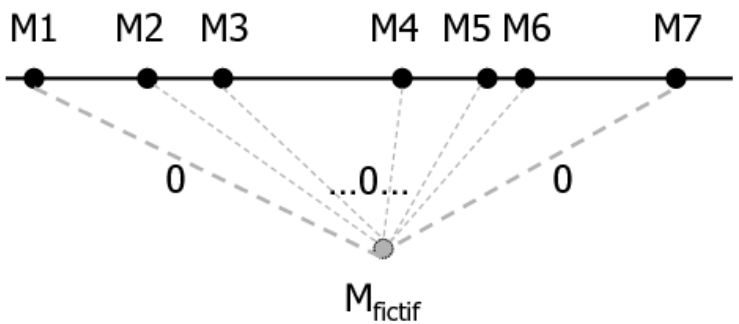
Solution : cartographie génétique

Soit n le nombre d'enfants mesurés et $n_{i,j}$, le nombre de recombinaisons observées entre deux marqueurs M_i et M_j (cas $M_i = 0, M_j = 1$ ou cas $M_i = 1, M_j = 0$).

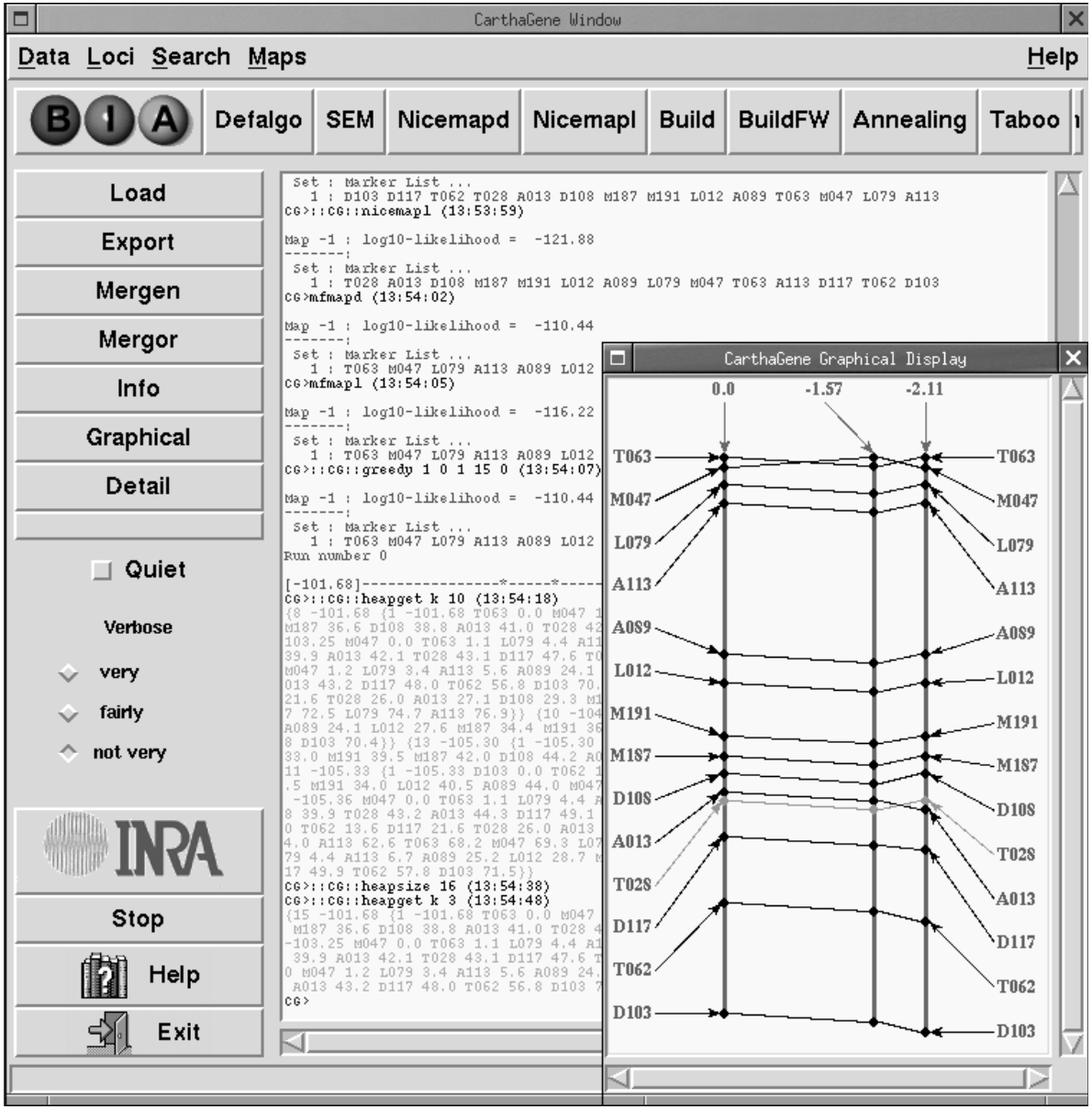
Matrice de distance : $\delta(M_i, M_j) = n_{i,j}$ (critère de parcimonie).

Seuillage de la matrice $(n/2)$ et recherche des composantes connexes.

Pour chaque composante C, définition d'un VRP $(G = (C', C' \times C'), \delta)$ avec $C' = C \cup \{M_f\}$ et M_f un marqueur fictif tel que $\delta(M_f, M) = \delta(M, M_f) = 0 \quad \forall M \in C$.



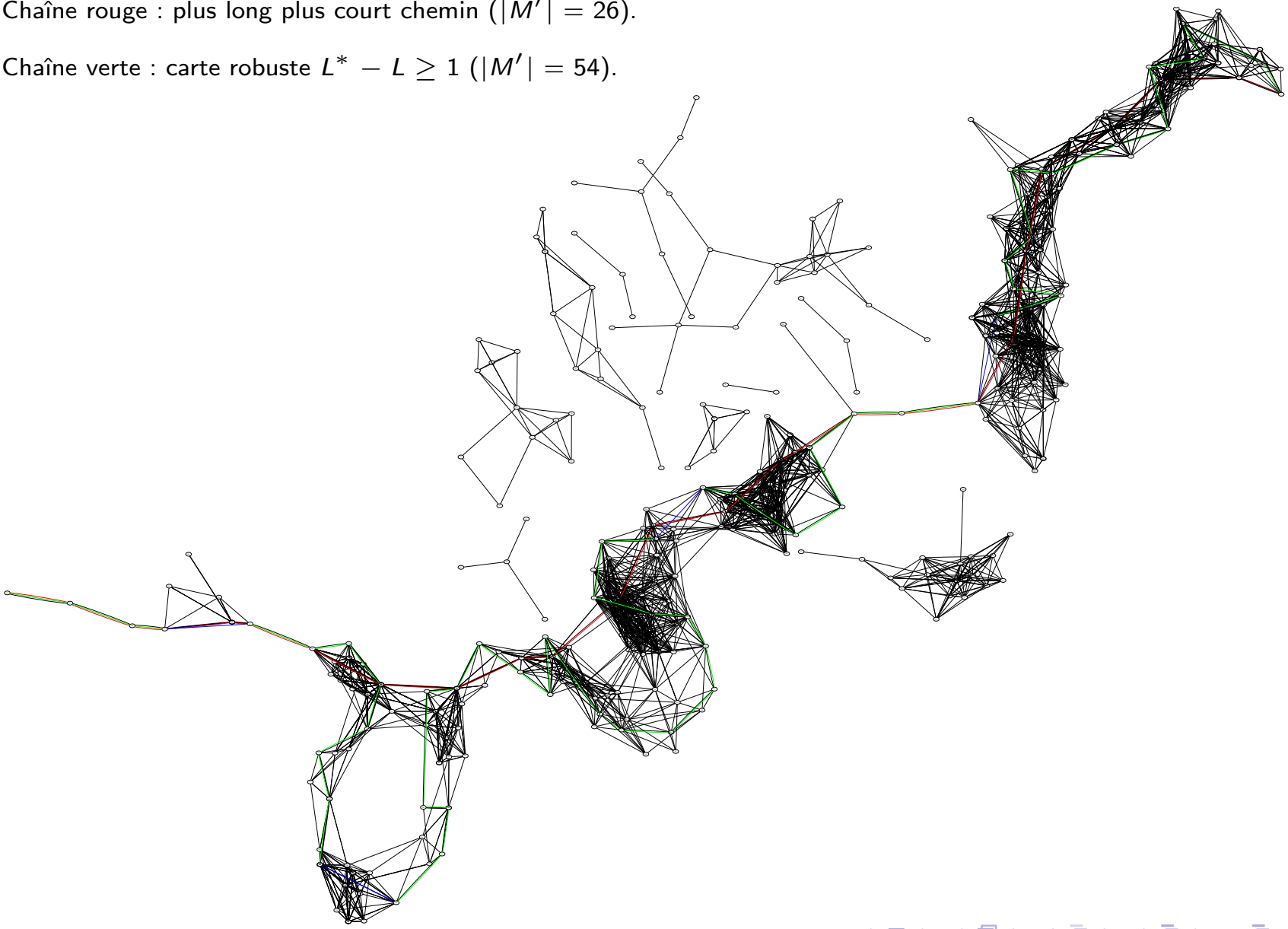
Solution : cartographie génétique



Solution : cartographie génétique

Matrice de log-vraisemblance normalisée 2-point avec un seuil à 9 sur le chromosome 2 du chien ($|M| = 427$).
Chaîne rouge : plus long plus court chemin ($|M'| = 26$).

Chaîne verte : carte robuste $L^* - L \geq 1$ ($|M'| = 54$).



Partionnement de graphe et maximisation de la modularité

Objectif : regrouper les sommets d'un graphe en modules homogènes et bien séparés.

Définition

(Newman, 2002)
Soit un graphe non-orienté $G = (S, A)$.
Trouver une partition de S *en modules qui maximise le nombre d'arêtes* internes *moins le nombre attendu d'arêtes sous l'hypothèse d'un tirage aléatoire des arêtes ayant la même distribution des degrés que dans* G.

Maximisation de la modularité

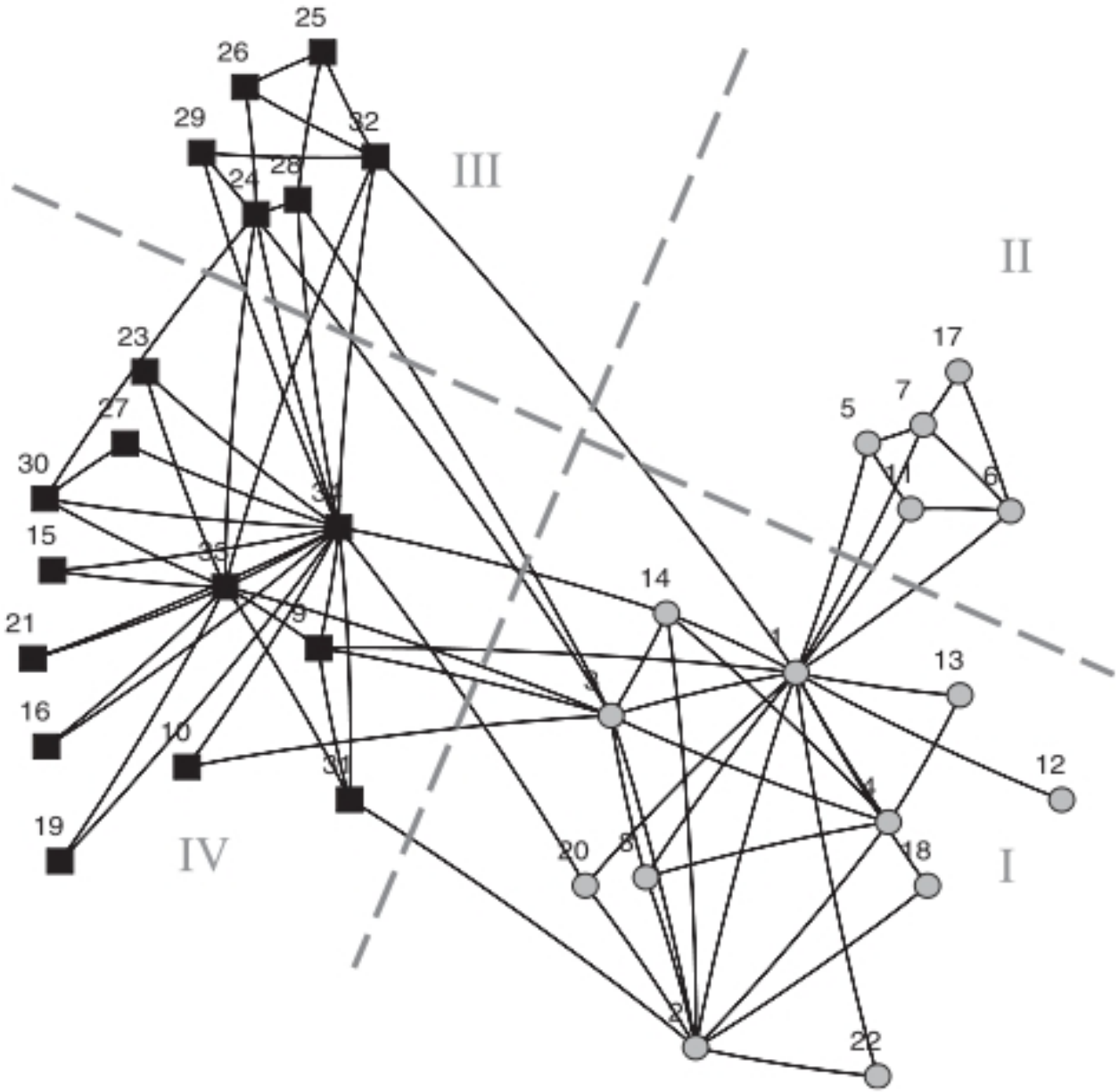
Définition

(Newman, 2004)
Soit un graphe non-orienté $G = (S, A)$.
Maximiser $Q = \sum_m [a_m - e_m]$
avec a_m, *le nombre d'arêtes de* A *dans le module* m
et e_m *le nombre attendu.*

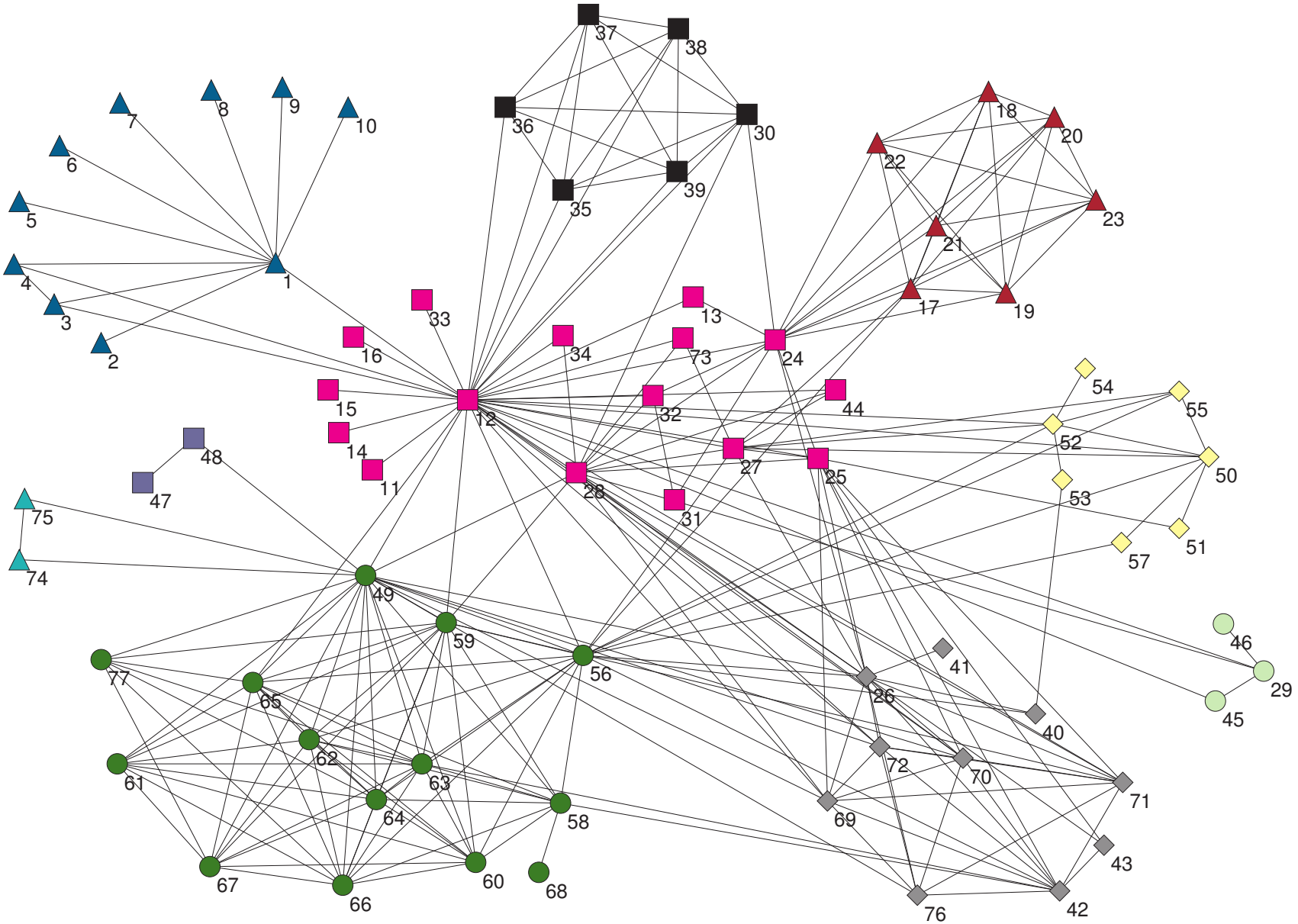
$Q = 0$ *: graphe aléatoire.*
$Q = 1$ *: graphe ayant une structure en fortes* communautés.

Le problème de maximisation de la modularité est NP-difficile.

Modularité : karaté club de Zachary $Q = 0.42$



Modularité : Les Misérables de Victor Hugo $Q = 0.56$



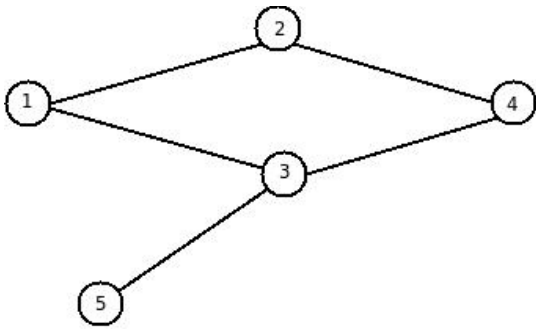
Reformulation en partionnement de cliques

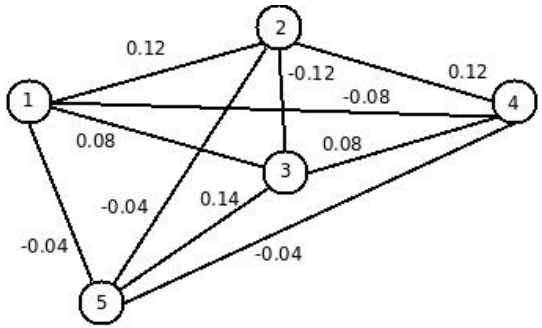
Définition

Soit un graphe non-orienté $G = (S, A)$ *pondéré par*
$w(u, v) = \frac{1}{|A|}(e_{u,v} - \frac{degre(u).degre(v)}{2|A|})$
avec $e_{u,v} = 1$ *si* $(u, v) \in A$ *et* $e_{u,v} = 0$ *sinon.*

On introduit des variables Booléennes $x_{u,v} = 1$ *si les sommets* u *et* v *sont dans le même module et* $x_{u,v} = 0$ *sinon.*

Maximiser $Q = \sum_{u<v \in S} w(u, v).x_{u,v}$
Tel que $x_{u,v} + x_{u,w} - x_{v,w} \leq 1 \qquad \forall u, v, w \in S$
et $x_{u,v} \in \{0, 1\} \qquad \forall u, v \in S$





Résolution via la *programmation linéaire en nombres entiers*.