Introduction au partitionnement de graphe

Charles-Edmond Bichot

Novembre 2007

Plan du cours

Introduction

Le problème du partitionnement de graphe

Différentes méthodes de partitionnement

Outils et comparaisons entre méthodes

Introduction

Définition de la partition d'un ensemble

Definition (Partition)

Soit un ensemble S quelconque. Un ensemble P de sous-ensembles de S est appelé une partition de S si :

1. Aucun élément de P n'est vide ;
2. L'union des éléments de P est égal à S ;
3. Les éléments de P sont deux à deux disjoints.

Les éléments de P sont appelés les parties de la partition P.

Définition de la partition d'un ensemble

Remarque :

Soit un graphe $G = (S, A)$. On peut chercher :

- une partition de l'ensemble des sommets S
- une partition de l'ensemble des arêtes A

En générale, on entend par partition d'un graphe la partition de l'ensemble de ses sommets.

Définition de la partition d'un ensemble

Vocabulaire et notations

Bissection d'un graphe $G = (S, A)$:

Il s'agit d'une partition de S en 2 parties

k-partition d'un graphe $G = (S, A)$:

Il s'agit d'une partition de S en k parties

Notation :

Soit un graphe $G = (S, A)$.

- $n_S = card(S)$
- $n_A = card(A)$

Notations et rappels sur les graphes

Graphe simple, poids d'une arête et poids d'un sommet

Definition (Graphe simple)

Un graphe est dit simple s'il n'a ni boucle ni arête multiple.

Definition (Graphe valué)

Soit un graphe $G = (S, A)$. On dit que le graphe est valué si à chaque élément $a \in A$ est associé une valeur $poids(a) \in \mathbb{N}^*$, appelée poids de a.

- Par extension, on considère qu'un couple de sommets $(s, s') \in S^2$, tel que $(s, s') \notin A$, possède un poids nul : $poids(s, s') = 0$.
- Le poids d'un sous-ensemble X d'éléments de A, est la somme des poids des éléments de X : $poids(X) = \sum_{a \in X} poids(a)$.
- De même, on associe parfois aux éléments s de S un entier strictement positif appelé le poids de s et noté $poids(s)$.

Domaines d'application du partitionnement de graphe

Recherche et ingénierie.

Nombreuses applications :

- répartition de charge dans les machines parallèles ;
- conception de circuits intégrés électroniques ;
- segmentation d'image ;
- exploration de données ;
- trafic aérien ;
- *etc.*

Exemples d'application du partitionnement de graphe

La répartition de charge dans les machines parallèles

But :

Répartir la charge de travail entre plusieurs processeurs.

Exemple :

Résoudre un système linéaire $Ax = b$ avec A une matrice creuse.

Problème de partitionnement le plus étudié, littérature abondante.

Exemples d'application du partitionnement de graphe

La répartition de charge dans les machines parallèles

Constat :

Difficile d'augmenter la fréquence des microprocesseurs

$\Downarrow$

Solution utilisée pour augmenter la puissance des ordinateurs :

Utilisation de processeurs multi-cœurs

$\Downarrow$

Nouveau problème pour les ordinateurs personnels :

Répartir la charge de travail entre les processeurs

Exemples d'application du partitionnement de graphe

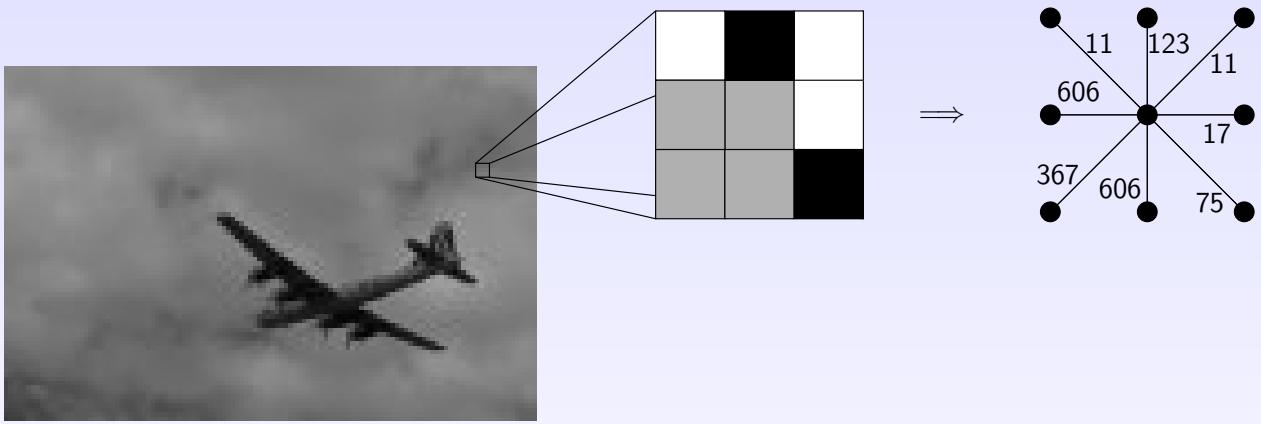
La segmentation d'image

But :

Isoler les différentes régions d'une image.

Principe :

Convertir une image en un graphe pour transformer la segmentation en partitionnement.



- À chaque pixel de l'image est associé un sommet ;
- Les pixels « proches » sont liés par des arêtes dont le poids dépend de leurs distances et de leurs différences d'intensité lumineuse.

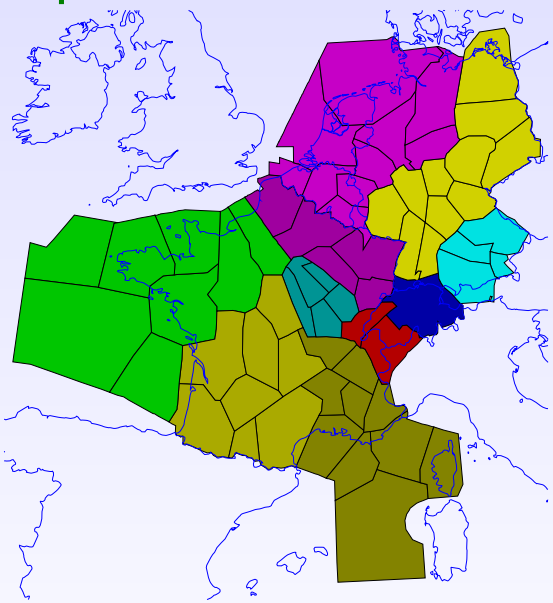
Exemples d'application du partitionnement de graphe

Le découpage de l'espace aérien

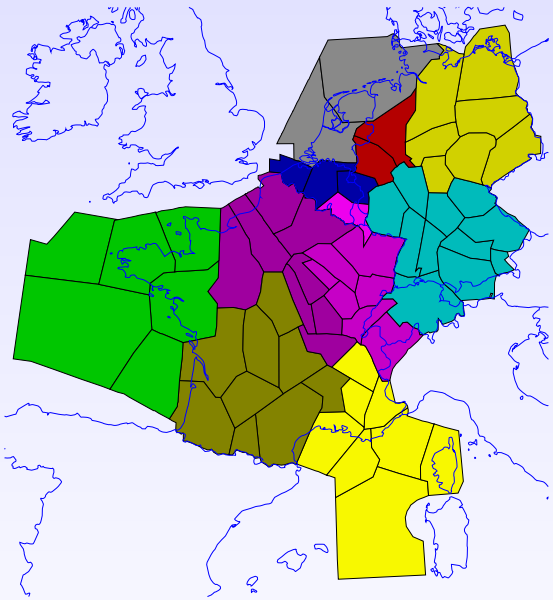
But :

Redécouper les régions de l'espace aérien actuel afin de le rendre plus capacitif.

Exemple :



Découpage réel à 36 000 pieds



Découpage trouvé à 36 000 pieds

Le problème du partitionnement de graphe

Le problème du partitionnement de graphe

Le partitionnement de graphe possède de nombreuses applications.

$\Downarrow$

Existe-t-il une formulation unique de ce problème ?

Deux catégories de problèmes de partitionnement

Deux problèmes :

La littérature sur le partitionnement de graphe fait apparaître deux catégories de problèmes de partitionnement :

- le partitionnement contraint ;
- le partitionnement non contraint.

Difficulté :

Problèmes de la classe NP-difficile.

Deux catégories de problèmes de partitionnement

Équilibre d'une partition

Balance de partitionnement

Soit une partition $P_k = \{S_1, \ldots, S_k\}$ de S en k parties.

$$poids_{moy} = \left\lceil \frac{poids(S)}{k} \right\rceil$$

$$balance(P_k) = \frac{\max_i poids(S_i)}{poids_{moy}}$$

Intuitivement :

Le poids maximal d'une partie sera au plus égal à $balance(P_k)$ fois le poids moyen d'une partie.

Partitionnement contraint

Description du problème

Le problème du partitionnement contraint consiste à trouver une partition dont :

- les parties sont de tailles similaires ($balance \leq 1.05$) ;
- le coût de coupe est minimisé.

Problème de la répartition de charge dans les machines parallèles.

Partitionnement contraint

Coût de coupe

Coût de coupe d'une partition

Soit une partition $P_k = \{S_1, \ldots, S_k\}$ de S en k parties.

$$coupe(S_1, S_2) = \sum_{s_1 \in S_1, s_2 \in S_2} poids(s_1, s_2)$$

$$coupe(P_k) = \sum_{i<j} coupe(S_i, S_j)$$

Intuitivement :

Minimise le « nombre » d'arêtes coupée entre les différentes parties.

Partitionnement non contraint

Description du problème

- Trouver une partition qui minimise une fonction objectif tenant compte des différences de tailles entre les parties ;
- La contrainte de balance de partitionnement est facultative.

Partitionnement non contraint

Fonctions objectifs pour le partitionnement non contraint

Différentes fonctions objectifs :

Soit une partition $P_k = \{S_1, \ldots, S_k\}$ de S en k parties.

- le ratio de coupe :

$$ratio(P_k) = \sum_{i=1}^{k} \frac{coupe(S_i, S - S_i)}{poids(S_i)}$$

- la coupe normalisée :

$$norm(P_k) = \sum_{i=1}^{k} \frac{coupe(S_i, S - S_i)}{coupe(S_i, S)}$$

- *etc.*

Différentes méthodes de partitionnement

Énumération de méthodes pour le partitionnement de graphe

En fonction du problème à résoudre :

- Méthodes inertielles
- Classification hiérarchique
- Expansion de région
- Méthode spectrale
- Multi-niveaux
- Variantes de l'algorithme de Kernighan-Lin
- Analogie avec la mécanique du fluide : diffusion et percolation
- Métaheuristiques
- Méthodes hybrides
- *etc.*

Les méthodes inertielles

Spécificité :

Méthodes de classification

Différences entre classification et partitionnement :

- Classification : on connaît les distances entre chaque couple de sommets ;
- Partitionnement : la distance entre 2 sommets quelconques n'est *a priori* pas connue.

Les méthodes inertielles

La méthode des centres mobiles

Principe :

- Méthode de *clustering*
- Chaque partie de la partition possède un centre de gravité dépendant de la disposition de ses sommets
- k centres de gravité permettent de créer une k-partition
- Algorithme itératif qui déplace progressivement les centres de gravité
- L'algorithme converge vers la partition qui minimise l'inertie intraclasse :

$$\sum_{i=1}^{k} \sum_{s \in S_i} d(s, g_i)^2$$

Les méthodes inertielles

La méthode des centres mobiles

Algorithme :

- Initialisation : k points sont pris au hasard pour former les centres de gravité de la partition initiale
- Tant que les centres de gravité ne sont pas fixes, faire :
  - Les points les plus proches d'un centre de gravité sont assemblés dans la partie de la partition correspondante
  - Les nouveaux centres de gravité des parties sont calculés

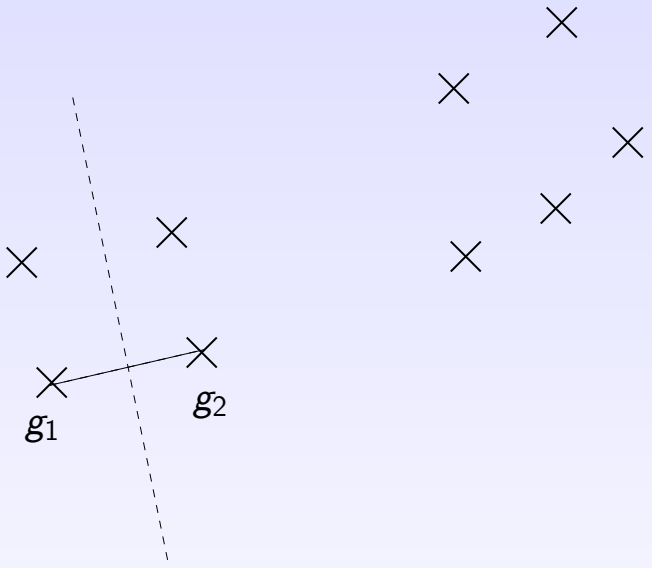
Particularités :

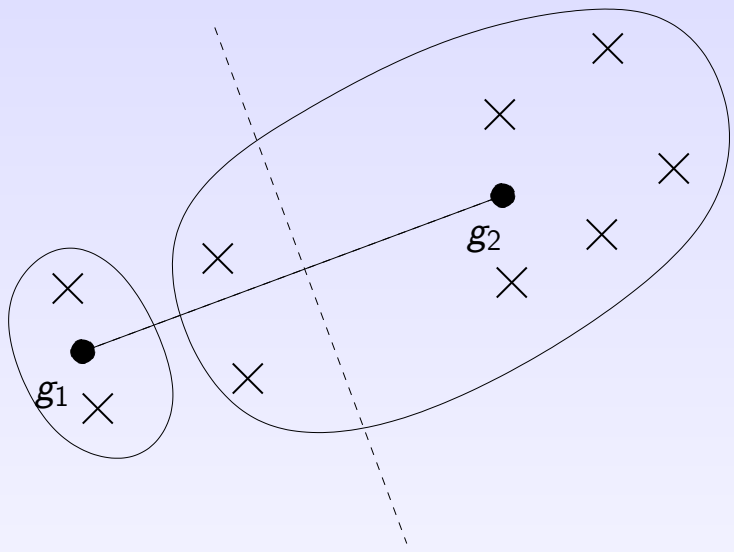
- Convergence rapide de l'algorithme
- Repose sur une distance calculable entre points et avec des points fictifs
- Si pendant l'itération, une classe se vide, il est possible de tirer au hasard un nouveau centre

Les méthodes inertielles

La méthode des centres mobiles

Exemple :





Équivalence entre classification non supervisée et partitionnement de graphe

Il a été montré que :

Les objectifs classiques de la classification sont équivalents à ceux du partitionnement de graphe.

Dans [1] :

Inderjit S. Dhillon, Yuqiang Guan, and Brian Kulis.
Weighted graph cuts without eigenvectors : A multilevel approach.
IEEE Transactions on Pattern Analysis and Machine Intelligence, 2007.

Les méthodes d'expansion de région

Principe :

- Partir de k sommets constituant la partition initiale du graphe
- Ajouter les sommets les plus proches et/ou fortement liés à chaque partie dans celles-ci jusqu'à ce qu'il n'y ait plus de sommet libre.

Analyse :

- Fonctionne par voisinage
- Très rapide
- Souvent peu performant, sauf dans le cas des petits graphes

La méthode spectrale

Introduction

Fondements :

- Méthode mathématique
- Son nom provient du théorème spectrale de l'algèbre linéaire qui permet d'affirmer la diagonalisation des matrices symétriques réelles.

Principe :

Utiliser des résultats d'algèbre linéaire afin de trouver une partition d'un graphe

Definition (Matrice Laplacienne)

Soit un graphe $G = (S, A)$. La matrice

$$M_{Lap} = \begin{cases} \sum_{k=1}^{n_S} poids(i,k) & \text{si } i = j \\ -poids(i,j) & \text{sinon} \end{cases}$$

est appelée matrice Laplacienne de G.

Propriété

La matrice Laplacienne d'un graphe non-orienté et positivement pondéré, est semi-définie positive.
$\Rightarrow$ ses valeurs propres sont positives

La méthode spectrale

Principes

- Minimiser le coût de coupe d'une bissection revient à trouver un vecteur $x \in \{-1, 1\}^{ns}$ qui minimise $x^T M_{Lap} x$
- Ce qui consiste à résoudre le système linéaire :

$$M_{Lap} x = \lambda x$$

Méthodes utilisées pour résoudre ce système linéaire :

- algorithme itératif de Lanczos
- méthode itérative du quotient de Rayleigh
- algorithme de la décomposition QR

Ces méthodes permettent de trouver des familles de vecteurs propres de M_{Lap} (les vecteurs de Fiedler), que l'on peut classer en fonction des valeurs propres respectives.

La méthode spectrale

Principes

Problème :

Les vecteurs de Fiedler sont dans $\mathbb{R}^{ns}$

Solution :

Discrétiser les vecteurs de Fiedler de $\mathbb{R}^{ns}$ vers $\{-1, 1\}^{ns}$ en tenant compte du problème à résoudre (contraint ou non)

Au final :

Le second vecteur de Fiedler va permettre de trouver une bissection du graphe

Littérature [8, 3] :

-  Alex Pothen, Horst D. Simon, and Kang-Pu Liou.
  Partitioning sparse matrices with eigenvectors of graphs.
  SIAM Journal on Matrix Analysis and Applications, 11(3) :430–452, 1990.

-  Bruce Hendrickson and Robert Leland.
  An improved spectral graph partitioning algorithm for mapping parallel computations.
  SIAM Journal on Scientific Computing, 16(2) :452–469, 1995.

La méthode spectrale

Importance de la méthode de résolution du système linéaire

Temps et mémoire utilisés pour trouver la partition d'un graphe en fonction de la méthode de résolution :

Graphes		Lanczos			Quotient de Rayleigh		
Nom	n_S	coût	tps(s)	Mo	coût	tps(s)	Mo
bcsstk32	44 609	7779	2,56	155	7647	0,41	28
t60k	60 005	186	1,11	165	171	0,20	17
wing	62 032	1324	1,20	136	1386	0,32	19
fe_tooth	78 136	4396	2,39	222	4372	0,3	30
finan512	74 752	206	2,68	317	498	0,34	29
fe_ocean	143 437	2222	2,32	301	659	0,43	42

- Méthode de Lanczos plus performante pour les petits graphes
- Méthode itérative du quotient de Rayleigh plus performante pour les grands graphes

La méthode spectrale

Bilan

Avantages de la méthode spectrale :

- Permet de trouver des solutions au problème du partitionnement contraint et non contraint
- Permet de trouver une borne inférieure au coût de coupe d'une partition d'un graphe non pondéré

Inconvénients de la méthode spectrale :

- Mieux adapté à la bissection qu'au k-partitionnement

Technique de la bissection récursive

Procédure BissectionRec($G = (S, A)$, k)
  $P_k \leftarrow \{S_1 = \emptyset, \ldots, S_k = \emptyset\}$
  Procédure Itérer(S', k', num)
    si $k' > 1$ **alors**
      $k'_1 \leftarrow \lfloor \frac{k'}{2} \rfloor$
      $k'_2 \leftarrow k' - k'_1$
      $S'_1, S'_2 \leftarrow bissection(S', \frac{k'_1}{k'}, \frac{k'_2}{k'})$
      Itérer(S'_1, k'_1, num)
      Itérer(S'_2, k'_2, $num + k'_1$)
    sinon
      $S_{num} \leftarrow S'$
    fin si
  fin Procédure
begin
  Itérer(S, k, 1)
  retourner P_k
fin Procédure

Utilité :

De nombreuses méthode de partitionnement ne s'intéressent qu'à la bissection

Complexité :

$O(log(k))$

Attention :

Ne permet de faire que des 2^i-partitions.

Algorithmes d'affinage basés sur celui de Kernighan-Lin

Principe de l'algorithme de Kernighan-Lin

But :

Affiner une partition, c.-à-d. minimiser localement la fonction objectif ; ici, le coût de coupe.

Idée :

Algorithme itératif par voisinage qui cherche à minimiser le coût de coupe en échangeant des ensembles de sommets entre 2 parties.

Article fondateur [7] :

B. W. Kernighan and S. Lin.
An efficient heuristic procedure for partitioning graphs.
Bell System Technical Journal, 49(2) :291–307, 1970.

Algorithmes d'affinage basés sur celui de Kernighan-Lin

La notion de gain introduite par Kernighan-Lin

Notion de gain :

Gain de coût de coupe résultant de l'échange du sommet $s \in S_1$ avec la partie S_2.

$$gain(s) = \sum_{s' \notin S_1} poids(s, s') - \sum_{s' \in S_1} poids(s, s')$$

Si $gain(s) > 0$ alors le coût de coupe de la partition diminue.

Algorithmes d'affinage basés sur celui de Kernighan-Lin

Algorithme de Kernighan-Lin

Procédure KL($G = (S, A)$, P_2)
Initialisation : calculer le gain de chaque sommet
tant que il existe des sommets non marqués **faire**
Sélectionner le couple de sommets dont l'échange constitue le plus grand gain
Échanger et marquer ces deux sommets
Mettre à jour les gains des voisins de 2 sommets
fin tant que
Échanger réellement dans P_2 les 2 ensembles de sommets dont la somme des gains est maximale.
fin Procédure

Problème :

- N'est adapté qu'à la bissection de graphe parfaitement équilibrée !
- Complexité : $O(n_A^2 log(n_A))$

Algorithmes d'affinage basés sur celui de Kernighan-Lin

Algorithme de Fiduccia-Mattheyses

Algorithme :

Procédure FM($G = (S, A)$)
    Initialisation : calculer le gain de chaque sommet
    tant que il existe des sommets non marqués dans la partie la plus grosse avec un gain ≥ 0 **faire**
        Sélectionner le sommet de plus grand gain
        Déplacer et marquer ce sommet
        Mettre à jour le gain des voisins de ce sommet
    fin tant que
fin Procédure

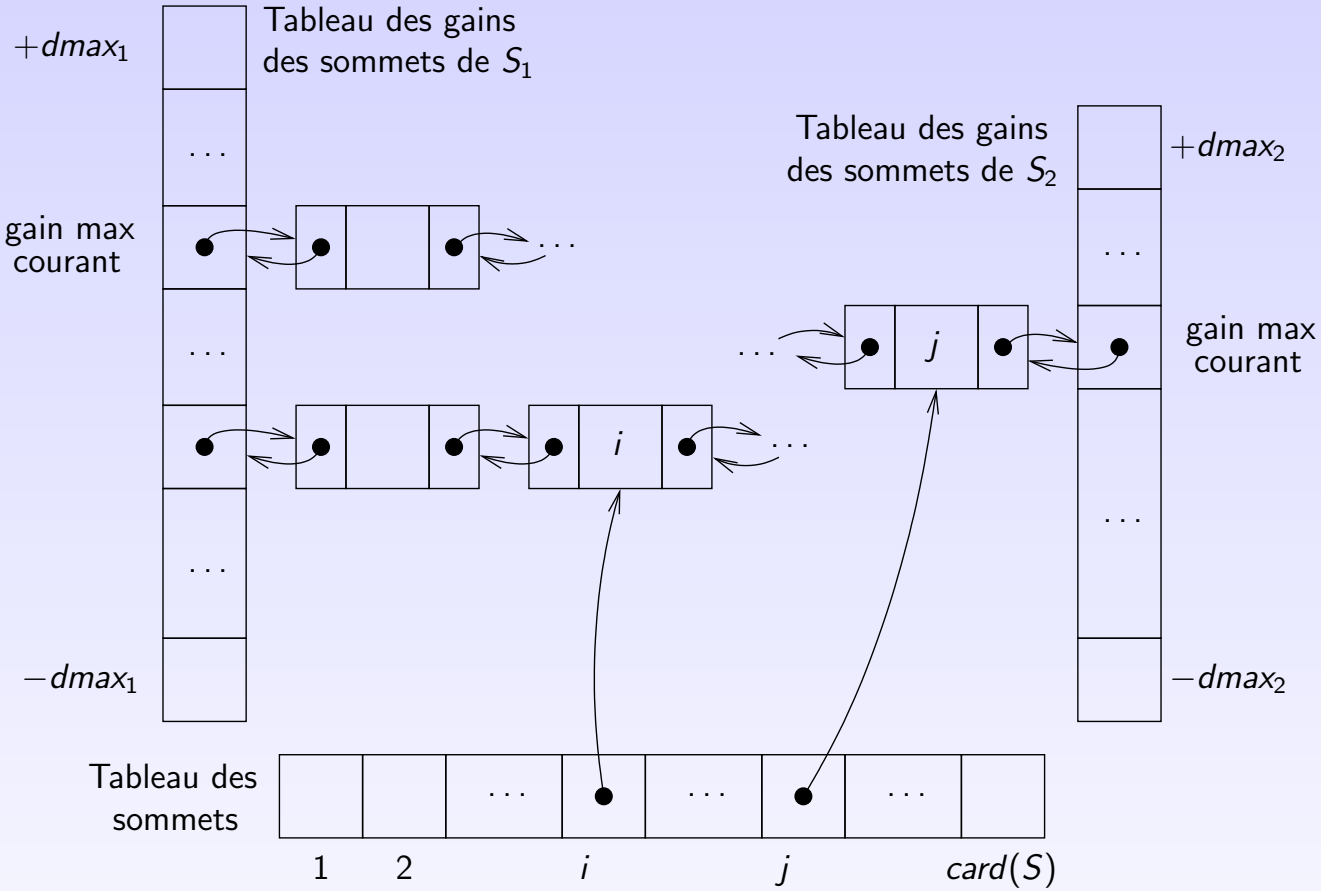
Complexité : $O(n_A)$

Présenté dans [2] :

C. M. Fiduccia and R. M. Mattheyses.
A linear-time heuristic for improving network partitions.
In *Proceedings of 19th ACM/IEEE Design Automation Conference*, pages 175–181, 1982.

Algorithmes d'affinage basés sur celui de Kernighan-Lin

Structure de contrôle des gains de Fiduccia-Mattheyses



- Accès à un sommet en temps constant
- Déplacement des gains en temps constant

Algorithmes d'affinage basés sur celui de Kernighan-Lin

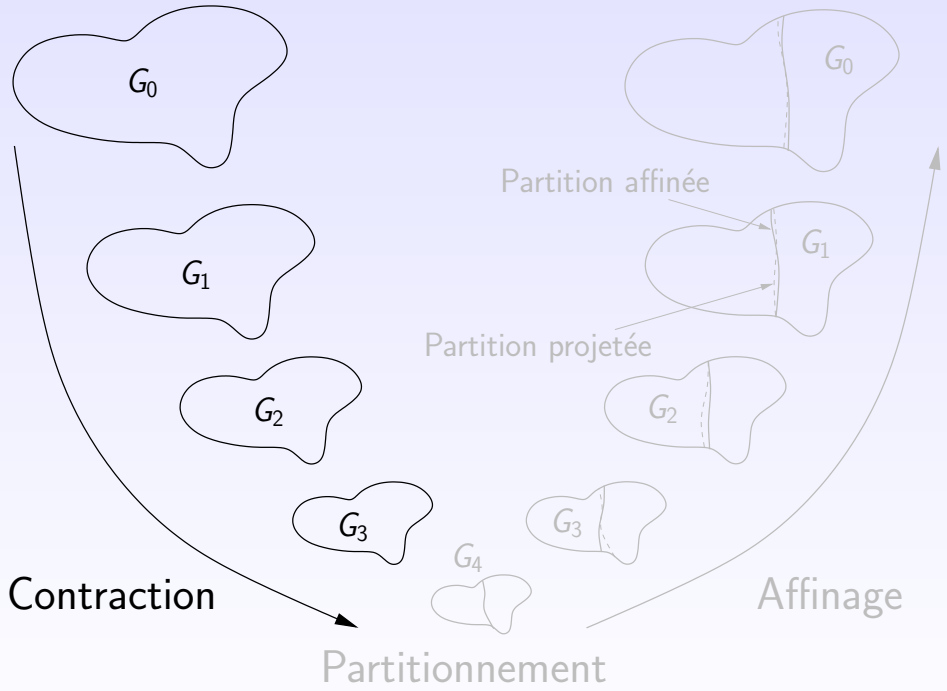
Autres améliorations de l'algorithme de Kernighan-Lin

- k-partitionnement direct [4, 6] :
  -  Bruce Hendrickson and Robert W. Leland.
    A multilevel algorithm for partitioning graphs.
    In *Proceedings of Supercomputing*, 1995.
  -  George Karypis and Vipin Kumar.
    Multilevel k-way partitioning scheme for irregular graphs.
    Journal of Parallel and Distributed Computing, 48(1) :96–129, 1998.
- afin de s'adapter à la balance de partitionnement demandée, on peut équilibrer les parties entre elles [9] :
  -  Chris Walshaw and M. Cross.
    Mesh partitioning : A multilevel balancing and refinement algorithm.
    SIAM Journal on Scientific Computing, 22 :63–80, 2000.

La méthode multi-niveaux

Méthode en 3 temps :

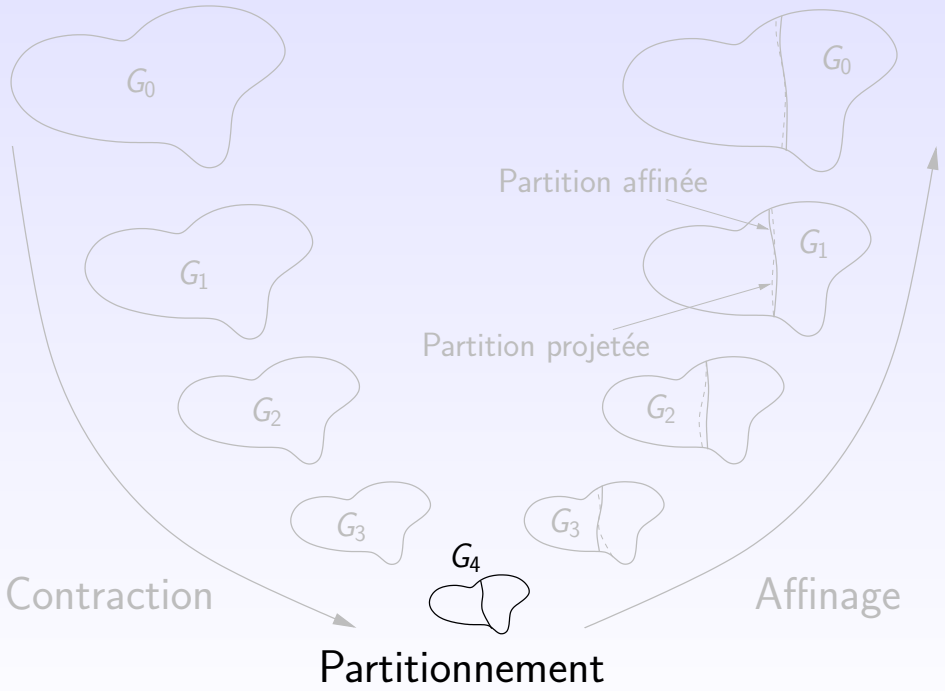
1. réduction de la taille du graphe par agrégation des sommets ;
2. partition du graphe réduit ;
3. retour progressif à la taille originelle avec améliorations locales de la partition.



La méthode multi-niveaux

Méthode en 3 temps :

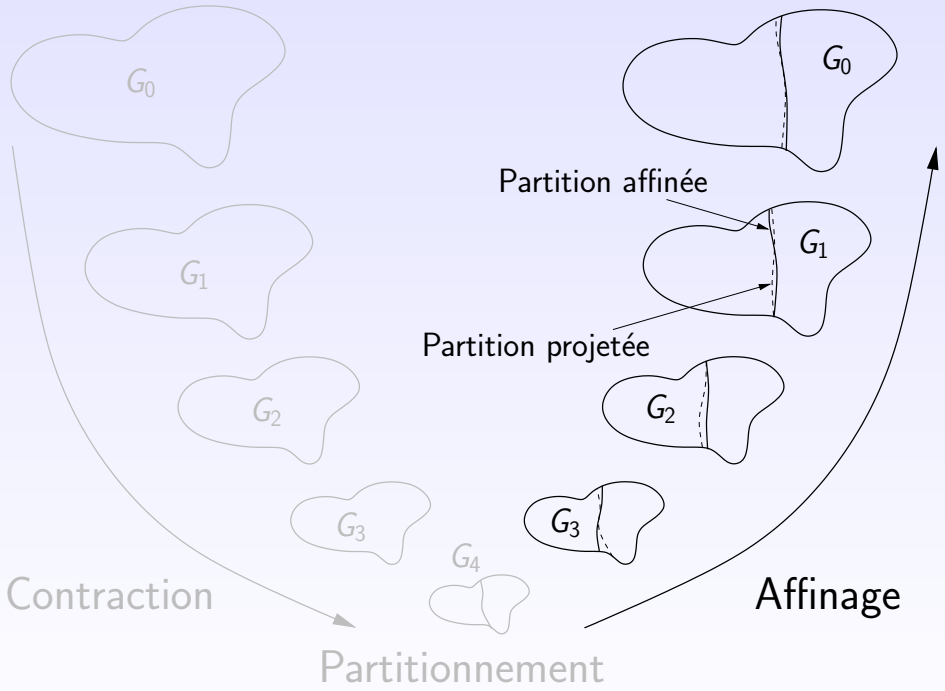
1. réduction de la taille du graphe par agrégation des sommets ;
2. partition du graphe réduit ;
3. retour progressif à la taille originelle avec améliorations locales de la partition.



La méthode multi-niveaux

Méthode en 3 temps :

1. réduction de la taille du graphe par agrégation des sommets ;
2. partition du graphe réduit ;
3. retour progressif à la taille originelle avec améliorations locales de la partition.



La méthode multi-niveaux

La phase de contraction

Objectif :

Obtenir une séquence de graphes $(G = G_1, \ldots, G_l)$ de plus en plus petits : $|S_1| > \cdots > |S_l|$.

Idée :

Construire G_{i+1} à partir de G_i en cherchant un *appariement maximal* $M_i \subset A_i$ et en contractant ensemble les paires de sommets incidents à chaque arête de l'appariement.

Definition (Appariement)

Soit un graphe $G = (S, A)$. L'appariement M du graphe G est un ensemble d'arêtes non-adjacentes deux à deux.
On dit que l'appariement M est maximal lorsque toute arête du graphe possède une intersection non vide avec au moins une arête de M.

L'algorithme *Random Matching*

Visite des sommets du graphe dans un ordre aléatoire et sélection aléatoire d'un sommet adjacente parmi ceux non sélectionnés.

Procédure RandomMatching($G_1 = (S_1, A_1)$, *taille*)
  $i \leftarrow 1$
  tant que $|S_i| >$ *taille* **faire**
    $App \leftarrow \{\}$
    tant que il existe des sommets non marqués **faire**
      Sélection aléatoire de $s_1 \in S_i$, non marqué
      Sélection aléatoire de $s_2 \in S_i$, non marqué et voisin de s_1
      Marquer s_1 et s_2, et ajouter l'arrête (s_1, s_2) à App
    fin tant que
    Construire G_{i+1} en contractant les couples de sommets de App
    $i \leftarrow i + 1$
  fin tant que
  retourner $(G_1, \ldots, G_i)$
fin Procédure

L'algorithme *Heavy Edge Matching*

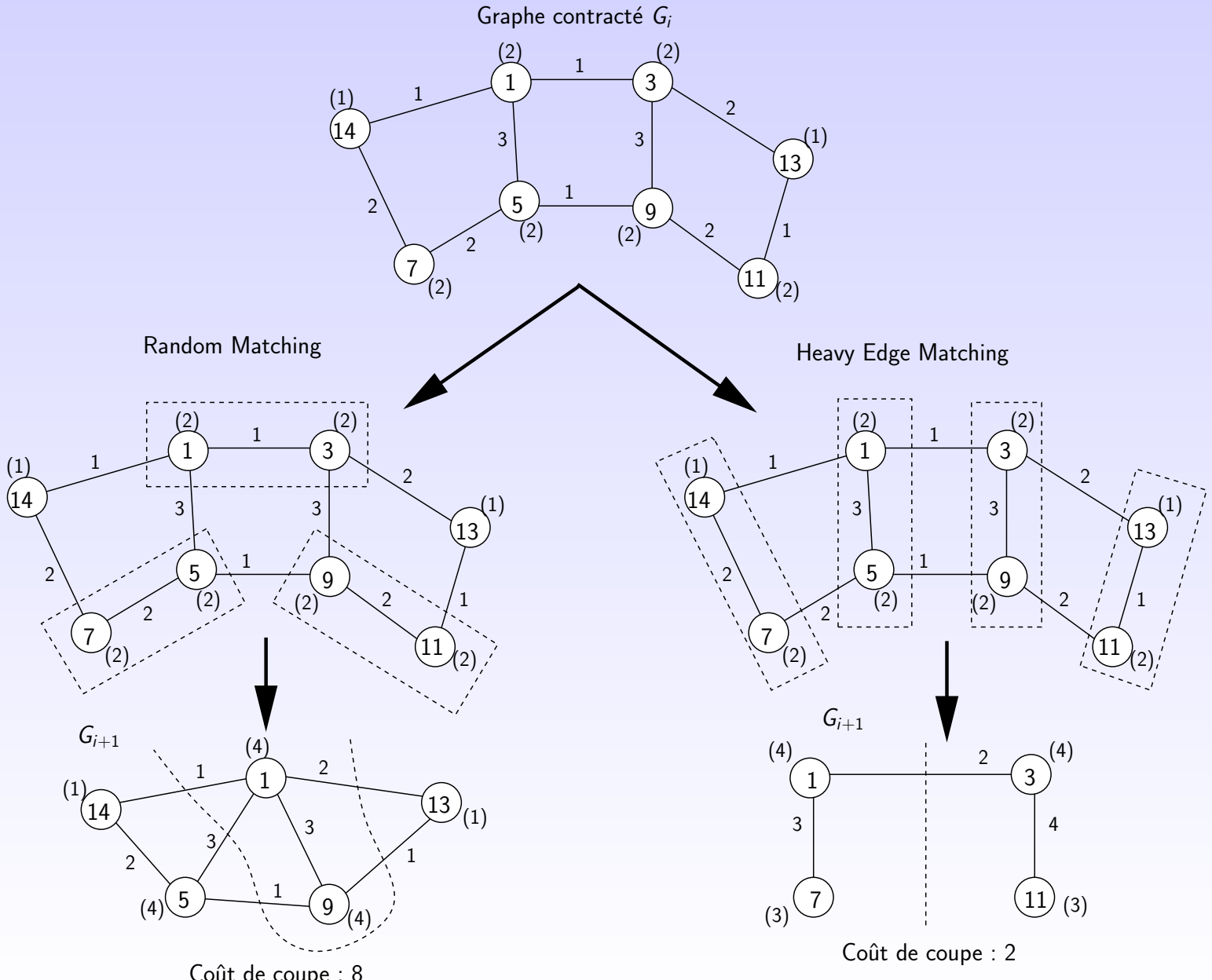
- Objectif : minimiser la somme du poids des arêtes du graphe contracté
- L'appariement est construit de manière à ce que le poids des arêtes qui le constitue soit élevé
- Les sommets sont visités dans un ordre aléatoire
- Un sommet est contracté avec le sommet non marqué qui lui est lié par l'arête de poids maximal

Présenté dans [5] :

George Karypis and Vipin Kumar.
A fast and high quality multilevel scheme for partitioning irregular graphs.
SIAM Journal of Scientific Computing, 20(1) :359–392, 1998.

La méthode multi-niveaux

La phase de contraction



La méthode multi-niveaux

Partition du graphe contracté

Méthodes souvent utilisées :

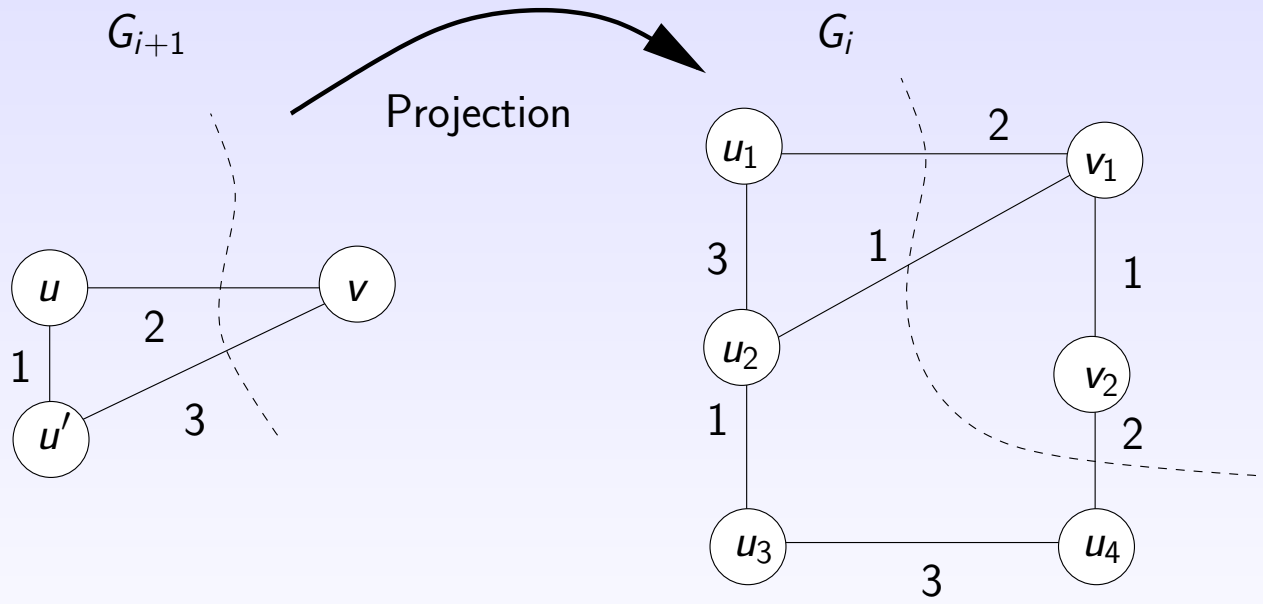
- Contracter le graphe jusqu'à obtenir k sommets :
  - Difficile et long d'arriver à k sommets
  - Partition peu équilibrée
- Expansion de région :
  - Graphe contracté petit (100 sommets)
  - Plusieurs partitions créées (5 à 20), la meilleure est choisie
  - Rapide
  - Partition plus équilibrée

La méthode multi-niveaux

Projection de la partition et affinage

La partition de graphe contracté G_l est projeté progressivement sur G en affinant successivement les partitions intermédiaires grâce à l'utilisation d'une variante de l'algorithme de Kernighan-Lin

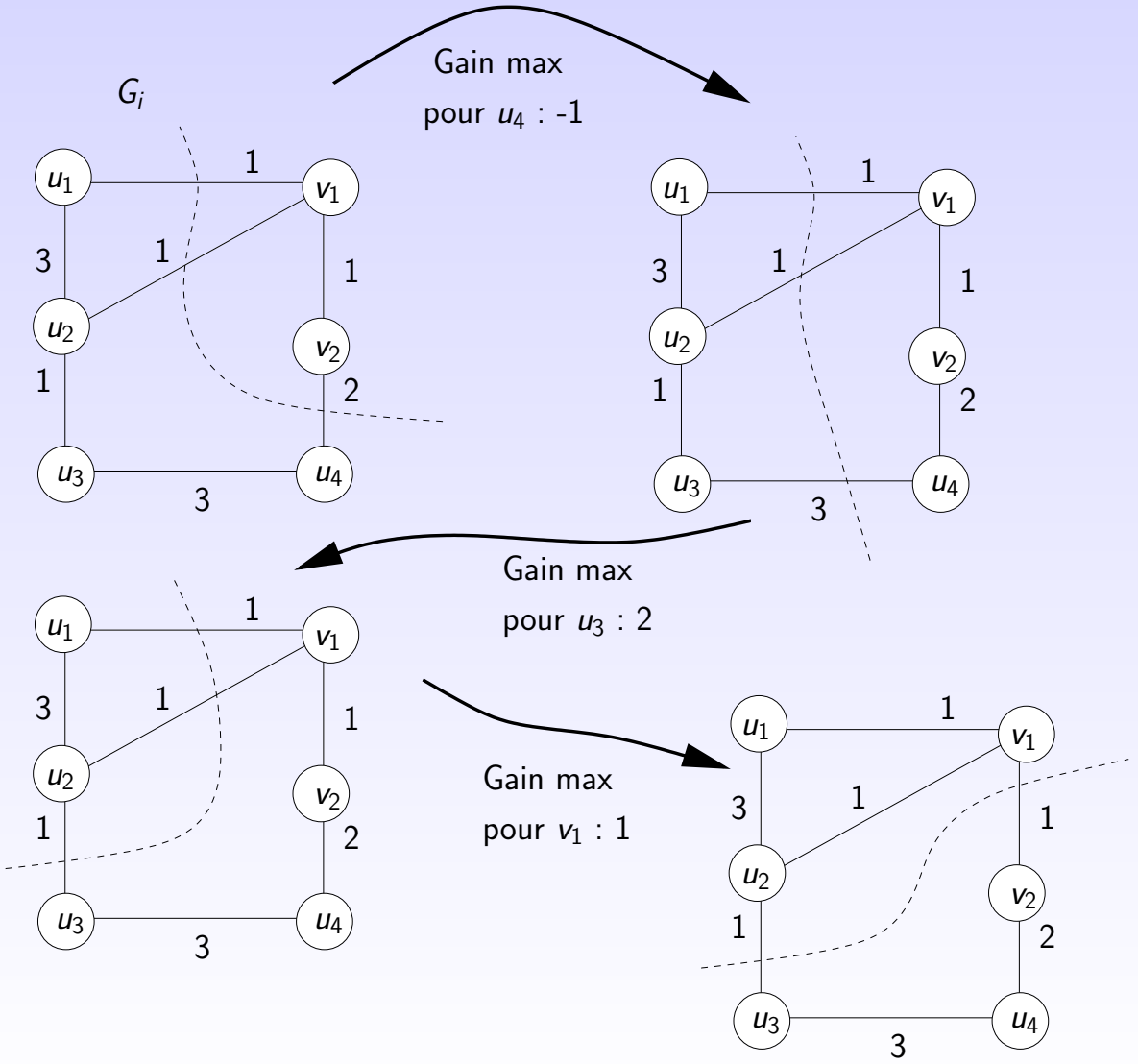
Exemple :



La méthode multi-niveaux

Affinage de Fiduccia-Mattheyses

Exemple suite :



Outils et comparaisons entre méthodes

Outils de partitionnement actuellement disponibles

Pour le partitionnement contraint :

- CHACO (*Sandia National Laboratories*, USA)
- METIS (*University of Minnesota*, USA)
- SCOTCH (INRIA/LaBRI, Bordeaux, France)
- JOSTLE (*University of Greenwich*, UK)
- PARTY (Université de Paderborn, Allemagne)

Pour le partitionnement non contraint :

- GRACLUS (*University of Texas*, USA)

Outils de partitionnement parallèle :

- ParMETIS (*University of Minnesota*, USA)
- PT-SCOTCH (INRIA/LaBRI, Bordeaux, France)
- JOSTLE parallel (*University of Greenwich*, UK)

Outils de partitionnement actuellement disponibles

The graph partition archive

The graph partition archive de Chris Walshaw :

- Importante base de données sur le partitionnement de graphe contraint
- 34 graphes
- Partitions en : 2, 4, 8, 16, 32 et 64 parties
- Partitions pour des balances : 1,00, 1,01, 1,03 et 1,05
- 816 partitions consultables

http ://staffweb.cms.gre.ac.uk/~wc06/partition/

Algorithmes actuellement les plus performants :

- Jostle Evolutionnary
- Iterated Jostle
- Fusion-Fission

Outils de partitionnement actuellement disponibles

Attention :

Dans la réalité, les algorithmes les plus performants ne sont pas forcément utilisés !

- très long
- pas forcément adapté au problème

Les outils classiques sont très rapides et performants :

CHACO, METIS, SCOTCH

Ce sont eux qui sont utilisés pour le problème de la répartition de charge dans les machines parallèles

Bibliographie I


Inderjit S. Dhillon, Yuqiang Guan, and Brian Kulis.
Weighted graph cuts without eigenvectors : A multilevel approach.
IEEE Transactions on Pattern Analysis and Machine Intelligence, 2007.
To appear.


C. M. Fiduccia and R. M. Mattheyses.
A linear-time heuristic for improving network partitions.
In *Proceedings of 19th ACM/IEEE Design Automation Conference*, pages 175–181, 1982.


Bruce Hendrickson and Robert Leland.
An improved spectral graph partitioning algorithm for mapping parallel computations.
SIAM Journal on Scientific Computing, 16(2) :452–469, 1995.


Bruce Hendrickson and Robert W. Leland.
A multilevel algorithm for partitioning graphs.
In *Proceedings of Supercomputing*, 1995.

Bibliographie II


George Karypis and Vipin Kumar.
A fast and high quality multilevel scheme for partitioning irregular graphs.
SIAM Journal of Scientific Computing, 20(1) :359–392, 1998.


George Karypis and Vipin Kumar.
Multilevel k-way partitioning scheme for irregular graphs.
Journal of Parallel and Distributed Computing, 48(1) :96–129, 1998.


B. W. Kernighan and S. Lin.
An efficient heuristic procedure for partitioning graphs.
Bell System Technical Journal, 49(2) :291–307, 1970.


Alex Pothen, Horst D. Simon, and Kang-Pu Liou.
Partitioning sparse matrices with eigenvectors of graphs.
SIAM Journal on Matrix Analysis and Applications, 11(3) :430–452, 1990.


Chris Walshaw and M. Cross.
Mesh partitioning : A multilevel balancing and refinement algorithm.
SIAM Journal on Scientific Computing, 22 :63–80, 2000.