Optimisation de profil

RAKOTOARIVONY Rado Fitiavana
M1 MISA

10 Avril 2024

1 Algorithme de Cuthill Mc Kee

Soit A une matrice symétrique définie positive et creuse. On veut arranger A de sorte que les éléments non nuls soient disposés autour de la diagonale et donc de réduire le nombre d'éléments nuls autour de la diagonale.
On représente la matrice A sous forme de graphe non orienté et sans boucle. Le but est de réorganiser la numérotation des noeuds du graphe, ce qui équivaut à réorganiser les lignes de A et donne naissance à une matrice de permutation P. On obtient alors une nouvelle matrice A' donnée par $A' = PAP^T$ qui répond à nos besoins.

1.1 Recherche du noeud d'initialisation

Dans le processus de renumérotation des noeuds, il faut choisir un noeud de départ. Le meilleur noeud de départ est le noeud le plus exentrique, il faudrait alors calculer l'exentricité de tous les noeuds ce qui est très coûteux.
Il faut alors trouver un algorithme moins coûteux mais qui permet de trouver un noeud quasi optimal pour l'initialisation de l'algorithme de renumérotation.

Algorithme (1):

* Initialiser un entier N au nombre de noeuds du graphe
* Choisir un noeud quelconque s
(a) Établir la structure en niveau de s, récupérer l'exentricité de s dans ex_{cur} et récupérer un noeud au choix dans le dernier niveau non vide et le stocké dans $next$
(b) Stockée la taille du dernier niveau avant le vide dans t
* N:=N-1
* start:=s
* s:=next
* Tant que $N \geq 1$

$$\begin{cases} ex_{prev} := ex_{cur} \\ N := N - t \\ \text{refaire } (a) \\ \text{refaire } (b) \\ \text{Si } ex_{cur} \; > \; ex_{prev} \text{ alors } start := s \\ s := next \end{cases}$$

* retourner *start*

1.2 Renumérotation des noeuds par l'algorithme de Cuthill Mc Kee

Algorithme (2):

* Initialiser une liste de listes de voisins L telle que $L[i]$ soit le voisin du noeud i
* Initialiser une liste vide *nodes*
* Obtenir le noeud d'initialisation *start* par l'algorithme (1) précédent
* Ajouter *start* à *nodes*
(1) $curNei := L[start]$
(2) Trier $curNei$ dans l'ordre croissant du nombre de vosins
* Initialiser un entier $i := 1$
* Tant que la taille de *nodes* < nombre de sommets du graphe:

$$\begin{cases} \text{Élargir } nodes \text{ avec } curNei \\ \text{Pour chaque } k,\ L[k] := L[k] - nodes \\ start := nodes[i] \\ \text{Refaire (1) et (2)} \end{cases}$$

* retourner *nodes*

2 Résolution d'un GSM

On souhaite résoudre $Ax = b$ avec A symétrique, définie positive et creuse.
On peut alors appliquer l'algorithme de Cuthill Mc Kee pour optimiser le profil de A avant de la stockée en profil et passer à la résolution.
On a vu que l'algorithme de Cuthill Mc Kee génère une matrice de permutation P telle que la matrice A' dont le profil est optimisé est donné par: $A' = PAP^T$
Le systeme précédent devient alors: $A'x' = b'$ avec $x' = Px$ et $b' = Pb$
On résoud donc $A'x' = b'$ puis on obtient la solution finale $x = P^T x'$ par permuation inverse

Algorithme (3):

* Obtenir la permutation $nodes$ par l'algorithme de Cuthill Mc Kee
* Obtenir A' et b' en utilisant les permutations définies par $nodes$ sur A et b
* Stocker en profil A'
* Effectuer en profil la factorisation LDL^T de A' puis la résolution toujours en profil de $LDL^T x' = b'$
* Obtenir la solution finale x en appliquant la permuation inverse $nodes^{-1}$ à x'
* retourner x