Révisions – Algorithmes et architectures parallèles

F. Bouchez, V. Sonigo, A. Benoit

Examen janvier 2008 (l'année dernière…)

1 Deux amusettes pour gagner des points

1. Quelles différences voyez-vous entre un tore 3D de taille 4x4x4 et un hypercube de dimension 6 ?
2. On veut calculer le produit scalaire de deux vecteurs de taille n distribués sur un anneau orienté de p processeurs. A la fin, tous les processeurs doivent connaître le résultat. Ecrire le pseudo programme distribué correspondant.

2 Anomalies de l'ordonnancement de liste

$$\begin{tikzcd}
 & & E/7 \arrow[dr] & \\
 & A/8 \arrow[ur] \arrow[dr] & & G/18 \\
 & & F/7 \arrow[ur] & \\
B/2 \arrow[r] & D/3 \arrow[ur] & &
\end{tikzcd}
\qquad
\begin{tikzcd}
 & & I/8 \\
C/3 \arrow[r] & H/2 \arrow[ur] \arrow[dr] & \\
 & & J/8
\end{tikzcd}$$

Considérons le DAG ci-dessus, où une paire X/w signifie que la tâche X a un poids w. Par exemple, la tâche A a un poids 8.

1. Quel est le makespan obtenu par l'ordonnancement basé sur le chemin critique avec 2 processeurs ? Est-il optimal ?
2. On diminue maintenant le poids de chaque tâche d'une unité : la tâche X a un poids $w-1$ (A : poids 7, ...). Montrer que le makespan calculé par le même algorithme augmente. Prouver que le makespan de n'importe quel ordonnancement de liste augmente (surprenant, n'est-ce pas ?)
3. On revient aux poids de départ w, et on suppose maintenant disposer de 3 processeurs. Montrer que le makespan de l'ordonnancement basé sur le chemin critique augmente, ainsi que le makespan de tout algorithme de liste (encore un fait surprenant !)

3 Analyses de dépendances

1. Effectuer une analyse de dépendances sur le bout de programme suivant :

$$\begin{array}{l}
S_1 : A \leftarrow A + C \\
S_2 : B \leftarrow E + F + C \\
S_3 : D \leftarrow A + B \\
S_4 : B \leftarrow B + D + A \\
S_5 : E \leftarrow 1
\end{array}$$

2. On considère le nid de boucles suivant :

$$\begin{array}{l}
\textsf{for } i = 1 \textsf{ to } N \textsf{ do} \\
\quad t(i) = t(i) + 1 \\
\quad \textsf{for } j = 2 \textsf{ to } M \textsf{ do} \\
\quad\quad a(i,j) = a(i+2,j) + a(i,j-2) \times t(i) \\
\quad \textsf{endfor} \\
\textsf{endfor}
\end{array}$$

Donner toutes les dépendances, et dessiner le graphe de dépendances développé et le graphe de dépendances réduit. Appliquer l'algorithme d'Allen et Kennedy sur le code. Peut on extraire plus de parallélisme à l'aide de transformations de boucles ?

3. Effectuer une analyse de dépendances sur le nid de boucles suivant :

```
for i = 1 to N do
  for j = 1 to N do
    S1 : a(i+1, j-1) ← b(i, j+3) × c(i, j-3) + 2
    S2 : b(i-1, j-1) ← a(i, j) - 1
    S3 : c(i+2, j) ← a(i, j-2) - b(i, j-1)
  endfor
endfor
```

4. Quelles contraintes doit satisfaire un vecteur de Lamport $\Pi = (a, b)$ pour ordonnancer le nid de boucles précédent (exécuter l'itération $I = (i, j)$ au temps $\pi.I$) ? Déterminer celui qui minimise le temps d'exécution, et ré-écrire ce nid de boucles en faisant apparaître du parallélisme.
5. Donner un nid de boucle parfait avec 3 instructions S_1, S_2 et S_3, dont les dépendances sont les suivantes :
   - De S_1 à S_2 : dépendance anti de vecteur de distance $(1, -2)$
   - De S_1 à S_3 : dépendance de flot de vecteur de distance $(1, -1)$
   - De S_2 à S_1 : dépendance anti de vecteur de distance $(1, 2)$
   - De S_2 à S_3 : dépendance de flot de vecteur de distance $(1, 4)$
   - De S_3 à S_1 : dépendance anti de vecteur de distance $(0, 1)$
   - De S_3 à S_2 : dépendance de flot de vecteur de distance $(0, 0)$

4 Plongement

Soit $G = (V, E)$ un réseau d'interconnexion arbitraire (mais connexe) à n sommets. On veut le configurer en anneau, i.e. plonger un cycle à n sommets dans le réseau. On utilise l'algorithme suivant :
- Construire un arbre couvrant de G
- Partitionner les noeuds de l'arbre en sommets pairs (de niveau pair dans l'arbre) et impairs
- Parcourir l'arbre en profondeur, ajouter un sommet pair à l'anneau si c'est sa première visite dans le parcours, et ajouter un sommet impair si c'est sa dernière visite dans le parcours

1. Traiter un petit exemple
2. Dilatation : quelle est la distance maximale dans le réseau de deux sommets consécutifs dans l'anneau ?
3. Partage : combien de fois un lien donné du réseau est-il partagé dans l'anneau ?

5 Ordonnancement avec communication

On considère le graphe de tâches composé de $n(n+1)/2$ tâches $T_{k,j}$, $1 \leq k \leq j \leq n$. Les contraintes de dépendance sont $T_{k,k} \rightarrow T_{k,j}$ pour $j > k$ et $T_{k,j} \rightarrow T_{k+1,j}$ pour $j > k$.

Question 1 Dans cette question, le poids de chaque tâche et de chaque arête de dépendance est égal à 1.

1. *(bonus)* A quel algorithme pourrait correspondre cette structure de graphe ?
2. Quel est le temps d'exécution optimal $T_{opt}(n)$ à ressources non bornées ? Combien de processeurs utilisez-vous pour obtenir ce temps d'exécution ?
3. Faire tourner l'heuristique du chemin critique modifié pour $n = 5$ et $p = 3$. Même question pour $n = 5$ et $p = 2$. On rappelle que la règle du chemin critique modifié consiste à allouer une tâche libre au processeur qui permet de débuter son exécution au plus tôt, compte tenu des décisions déjà prises. Les tâches libres sont triées suivant les niveaux inférieurs (*bottom-level*).

Question 2 Dans cette question, le poids de la tâche $T_{k,j}$ est $n - k + 1$, pour tout $j \geq k$. Le poids d'une arête $T_{k,k} \rightarrow T_{k,j}$ ou $T_{k,j} \rightarrow T_{k+1,j}$ est aussi égal à $n - k + 1$.

1. *(bonus)* A quel algorithme pourrait correspondre un tel graphe pondéré ?
2. Quel est le temps d'exécution optimal $T_{opt}(n)$ à ressources non bornées ? Combien de processeurs utilisez-vous pour obtenir ce temps d'exécution ?