Exercices pour le cours ALPA

P. Quinton, T. Risset

5 janvier 2000

Les exercices difficiles sont indiqués par une étoile ($\star$)

1 algorithmique systolique

PRÉREQUIS : ALGORITHMIQUE SYSTOLIQUE.

1.1 Convolution généralisée (15 mn)

- Décrire l'architecture systolique pour la convolution généralisée

$$y_i = \sum_{k=0}^{3} a_k x_{i-k} + \sum_{k=0}^{3} w_k y_{i-k+1}.$$

- Expliquer pourquoi on est naturellement conduit à utiliser des réseaux de convolution bidirectionnels.
- Expliquer ce qu'on peut faire pour utiliser, malgré tout, un réseau de type unidirectionnel pour la partie non récursive de la convolution.

1.2 Produit de matrices (15 mn)

- Décrire l'architecture systolique (réseau carré) pour le produit de matrices carrées $C = AB$.
- Expliquer comment modifier ce réseau et son utilisation, de façon que les coefficients de C soient évacués vers la gauche, afin de reboucler les résultats.
- Examiner aussi le cas ou les résultats C sont rebouclés, lorsqu'ils sortent (normalement) par la droite du réseau. Donner le délai nécessaire entre l'entrée de deux matrices successives, et expliquer quel dispositif de mémorisation doit être utilisé.

1.3 Synthèse par équations: transformée en ondelettes (45 mn) $\star$

PRÉ-REQUIS : SYNTHÈSE AUTOMATIQUE DE RÉSEAU SYSTOLIQUES

La transformée en ondelettes à récemment beaucoup été étudiée. Elle permet de faire un passage dans le domaine des fréquences de manière hiérarchique. Le calcul se fait en plusieurs *niveaux*, chaque niveau rendant successivement une information *de moins en moins précise*. À chaque niveau, on peut modéliser le calcul effectué lors d'une transformée en ondelettes unidimensionnelle (à N coefficients et par un facteur de deux) par la formule suivante:

$$y(i) = \sum_{k=1}^{N} w(k) * x(2i - k) \tag{1}$$

dans laquelle, les $x(j)$ sont des coefficients donnés en entrée, les $y(i)$ sont les résultats, les $w(k)$ sont N coefficients fixés. On va rechercher une architecture systolique pour ce problème. On ne vous demandera pas d'écrire les équations récurrentes au cours de la dérivation de l'architecture, mais d'expliquer informellement sur le graphe de dépendance.

1. De quelle manière peut-on sérialiser la réduction de l'équation (1). Choisissez une de ces possibilités et représentez le graphe de dépendance étendu pour $N = 3$, et $M = 5$ (M est le nombre de coefficient y que l'on calcule).
2. énumérez les possibilités pour pipeliner les variables x et w dans ce graphe. Quelles sont, selon vous, les cas à éviter. Choisissez une de ces possibilité et représenter le graphe de dépendance étendu uniformisé pour $N = 3$, et $M = 5$.
3. Donnez les contraintes que doit vérifier une fonction de temps pour le graphe trouvé à la question 2. Choisissez une fonction de temps respectant ces contraintes. Donnez une fonction d'allocation des calculs compatible avec la fonction de temps choisie.
4. Décrivez l'architecture obtenue avec les fonctions de temps et d'allocation trouvées à la question précédente. On pourra faire un schéma de l'architecture.
5. Quelle sont: la période du flot de y, la latence (temps total pour l'exécution d'une instance du problème), la macro période (temps entre l'exécution de deux instances successives de problème)?

2 Arithmétique entière

PRÉREQUIS : ADDITION AVEC ANTICIPATION DE RETENUE SUR UN ARBRE

2.1 addition (15 mn)

Une architecture est composée d'un arbre binaire complet de processeurs (N feuilles). Chaque processeur feuille possède $Log(N)$ bits d'un nombre de $NLog(N)$ bits. Peut-on avec cette architecture additionner deux nombres de $Nlog(N)$ bits en $\mathcal{O}(Log(N))$ étapes?

2.2 Calcul de préfixe (30 mn) $\star$

On rappelle le calcul du préfixe pour une opération $\otimes$ sur une suite x_i ($0 \leq i \leq N$): calcul de $\otimes_{i=1,j} x_i$ pour tout j ($0 \leq j \leq N$).

1. Le calcul de la retenue dans l'algorithme d'addition avec anticipation de retenue peut être vu comme un calcul de préfixe. Pour cela quelle opération préfixons nous?
2. En se basant sur l'algorithme d'addition avec propagation de la retenue, proposer un algorithme pour le calcul de préfixe.

3 Réseaux

PRÉ-REQUIS : HYPERCUBE, COMPLEXITÉ DES OPÉRATIONS DE COMMUNICATIONS DE BASE

3.1 plongement d'un arbre binaire complet dans un hypercube

1. Montrer que l'hypercube de N noeud ne contient pas un arbre complet binaire de $N-1$ noeud ($\star$).
2. Montrer que l'on peut plonger dans un hypercube à N noeud le "arbre à double racine" de N noeud, c'est à dire un arbre dans lequel la racine à été dédoublée, chaque "demi-racine" n'ayant qu'un seul fils.

3.2 Procédure SPMD pour le One-to-all sur un hypercube

1. On souhaite écrire une procédure qui implémente la communication one-to-all sur un hypercube. On suppose que l'on dispose de communications élémentaires `send` et `receive` qui prennent comme arguments un message et un numéro de processeur (qui doit être un des voisins du processeur exécutant l'instruction). On veut donc une procédure qui prend en argument la dimension de l'hypercube, le numéro du processeur qui l'exécute et le numéro du processeur qui initie le one-to-all (et le message à diffuser bien sur).

   Remarque: On pourra d'abord réfléchir à la procédure spécifique du one-to-all initié par le processeur 0, puis proposer une généralisation.

3.3 One to All dans un arbre

On dispose d'une machine ayant la topologie suivante: un arbre binaire équilibré (toutes les feuilles sont à la même profondeur) dont les p feuilles sont des processeurs et dont tous les autres noeuds (noeuds internes à l'arbre) sont des noeuds d'interconnection. On suppose qu'un noeud ne peu router qu'un seul message à la fois et que un message de taille m prend le temps $t_s + t_w m + t_h l$ pour parcourir un chemin comportant $l-1$ noeuds d'interconnection. Montrer que la diffusion simple (one to all) d'un message de taille m peut être exécutée en temps: $(t_s + t_w m + t_h(Log(p)+1))Log(p)$, donner l'algorithme.