Calcul du volume de données utilisées dans un nid de boucles : Quantité de mémoire locale utilisée pour l'exécution d'une itération d'une tâche.

Nicolas MUSEUX

18 octobre 1999

Résumé

Ce document présente une méthode pour calculer le volume de données nécessaire à un bloc de calcul d'une tâche. On ne cherche pas ici à modéliser le volume mémoire occupé par une application, mais juste à définir la quantité de mémoire nécessaire à un bloc de calcul d'une tâche donnée.

Table des matières

1 Introduction

La méthode exposée ci-après fait suite aux travaux décrits dans [Gue97] et [Jea96] sur la définition d'un modèle capacitif de la mémoire. Plus précisément, nous définissons un modèle de mémoire locale pour une itération (dans le temps) d'une tâche. Nous commencerons par décrire ce que nous appellerons *le volume de données de référence*, puis nous l'exploiterons pour donner l'expression du volume total de mémoire nécessaire à l'exécution d'un nid de boucle partitionné. Ce dernier est une sur approximation de la quantité réellement nécessaire.

2 Volume de données de référence

2.1 Définition du problème.

Considérons une tâche $\mathcal{T}$ constituée de n_e boucles externes, n_i boucles internes parfaitement imbriquées et dont le vecteur d'itérations externes (resp. internes) est $\mathbf{i} \in \left([0,\, up\,(\mathbf{i}_j)]_{\cap\mathbb{N}}\right)^{n_e}$ (resp. $\mathbf{m} \in \left([0,\, up\,(\mathbf{m}_j)]_{\cap\mathbb{N}}\right)^{n_i}$). Son partitionnement est donné par les matrices P et L et le vecteur $up\,(\mathbf{c})$.

$$P = \begin{pmatrix} P_{11} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & P_{n_e n_e} \end{pmatrix} \quad L = \begin{pmatrix} L_{11} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & L_{n_e n_e} \end{pmatrix}$$

Soit un tableau A utilisé par $\mathcal{T}$. Ses matrices de pavage et d'ajustage sont [1] :

$$\Omega_e^A = \begin{pmatrix} \omega_{11}^e & \cdots & \omega_{1n_e}^e \\ \vdots & \ddots & \vdots \\ \omega_{n1}^e & \cdots & \omega_{nn_e}^e \end{pmatrix} \quad \text{avec} \quad \begin{cases} \forall k \in \{1,\ldots,n\},\, \exists_1 j \in \{1,\ldots,n_e\},\, \omega_{kj}^e \neq 0 \\ \forall j \in \{1,\ldots,n_e\},\, \exists_1 k \in \{1,\ldots,n\},\, \omega_{kj}^e \neq 0 \end{cases} \tag{1}$$

$$\Omega_i^A = \begin{pmatrix} \omega_{11}^i & \cdots & \omega_{1n_i}^i \\ \vdots & \ddots & \vdots \\ \omega_{n1}^i & \cdots & \omega_{nn_i}^i \end{pmatrix} \quad \text{avec} \quad \begin{cases} \forall k \in \{1,\ldots,n\},\, \exists_1 j \in \{1,\ldots,n_i\},\, \omega_{kj}^i \neq 0 \\ \forall j \in \{1,\ldots,n_i\},\, \exists_1 k \in \{1,\ldots,n\},\, \omega_{kj}^i \neq 0 \end{cases} \tag{2}$$

Elles définissent la fonction d'accès aux données [2] d'un tableau. Pour le problème qui nous interresse, la forme exacte de cette fonction d'accès est $\varphi = \left(\Omega_e^A \,\vdots\, \Omega_i^A\right) \begin{pmatrix} \mathbf{i} \\ \mathbf{m} \end{pmatrix} \bmod \mathbf{k} = \Omega_e^A.\mathbf{i} + \Omega_i^A.\mathbf{m} \bmod \mathbf{k}$, où φ est le vecteur d'index permettant de repérer une donnée dans un tableau [3]. Dans la suite de ce document, nous considérerons qu'il n'y a pas de modulo, i.e.,

$$\varphi = \Omega_e^A.\mathbf{i} + \Omega_i^A.\mathbf{m} \tag{3}$$

Le partitionnement 3D [Gue97] donne, pour une affectation $(\mathbf{c},\, \mathbf{p},\, \mathbf{l})$ (i.e., un partitionnement bloc/cycle) :

1. Par abus de langage, nous dirons que les matrices de pavage et d'ajustage sont toutes deux diagonales à une permutation près, pour signifier le fait qu'il n'y a au plus qu'un seul terme non nul par ligne et par colonne.

2. Le terme *donnée* désigne ici non pas la valeur de la donnée mais l'emplacement mémoire qu'elle occupe.

3. Typiquement, une donnée du tableau $\mathcal{A}$ indexée par φ est obtenue par $\mathcal{A}[\varphi]$.

$$\varphi = \Omega_e^A . (L.P.\mathbf{c} + L.\mathbf{p} + \mathbf{l}) + \Omega_i^A .\mathbf{m}$$

La relation liant une donnée d'un tableau avec le vecteur d'index permettant d'accéder à celle-ci est une bijection : à une donnée correspond un unique vecteur d'index, à un vecteur d'index correspond une unique donnée. Aussi, évaluer le volume de données nécessaire à un calcul revient à évaluer le nombre de vecteurs d'index distincts générés pour ce calcul.

Par conséquent, le volume de données du tableau A nécessaire à l'exécution de la tâche $\mathcal{T}$ à l'instant logique **c** sur la partition **p** est donné par

$$\begin{aligned}
\mathcal{V}_{ref}(\mathbf{c},\, \mathbf{p}) &= card\left(\left\{\varphi \mid \varphi \in \Phi,\, \exists \mathbf{l} \in \mathcal{D}_l,\, \exists \mathbf{m} \in \mathcal{D}_m,\, \varphi = \Omega_e^A . (L.P.\mathbf{c} + L.\mathbf{p} + \mathbf{l}) + \Omega_i^A .\mathbf{m}\right\}\right) \\
\text{avec } \mathcal{D}_l &= [0,\, L_{11} - 1]_{\cap \mathbb{N}} \times \cdots \times [0,\, L_{n_e n_e} - 1]_{\cap \mathbb{N}} \\
\mathcal{D}_m &= [0,\, up(\mathbf{m}_1) - 1]_{\cap \mathbb{N}} \times \cdots \times [0,\, up(\mathbf{m}_{n_i}) - 1]_{\cap \mathbb{N}}
\end{aligned}$$

Le problème est de calculer ce cardinal.

2.2 De l'injectivité de la fonction d'accès

Comme à chaque instant tous les processeurs exécutent le même calcul[4], la quantité de données utiles à une tâche à l'instant logique 0 sur la partition 0 est la même que celle utilisée par la même tâche sur une autre partition à un autre moment. D'où $\mathcal{V}_{ref}(\mathbf{c},\, \mathbf{p}) = \mathcal{V}_{ref}(0,\, 0)$, soit :

$$\mathcal{V}_{ref}(0,\, 0) = card\left(\left\{\varphi \mid \varphi \in \Phi,\, \exists \mathbf{l} \in \mathcal{D}_l,\, \exists \mathbf{m} \in \mathcal{D}_m,\, \varphi = \Omega_e^A .\mathbf{l} + \Omega_i^A .\mathbf{m}\right\}\right)$$

L'espace d'itération $D_l \times D_m$ définit un hyper-parallèlogramme. Comme à chaque itération on effectue un calcul, la première idée qui vienne à l'esprit pour calculer $V_{ref}(0,\, 0)$ est de calculer le volume de cet hyper-parallèlogramme, à savoir $\prod_{k=1}^{n_e} L_{kk} . \prod_{j=1}^{n_i} up(\mathbf{m}_j)$. Ce calcul ne convient pas car il ne tient pas compte des éventuels doublons (i.e., des références redondantes à une donnée). Par exemple, considérons deux boucles imbriquées, dont les indices sont respectivement i et m, et un tableau T à une dimension. Supposons que la fonction d'accès aux données de T soit $i + m$ et que le partitionnement donne $P = (1)$, $L = (up(i))$, alors si $i \in [0,\, 2]_{\cap N}$ et $m \in [0,\, 1]_{\cap \mathbb{N}}$ la formule précédente calcule $(2 + 1) . (1 + 1) = 6$ données, ce qui est faux puisque l'on n'accède qu'à 4 données ($T[0]$, $T[1]$, $T[2]$, $T[3]$). L'erreur est de supposer qu'à chaque itération d'un calcul élémentaire on utilise de nouvelles données. Ce n'est vrai que si Ω^A est injective, ce qui n'arrive presque jamais.

À quelles conditions Ω^A n'est-elle pas injective ? Par définition de l'injectivité, on a :

$$\left(\Omega^A \text{ n'est pas injective}\right) \Leftrightarrow \left(\begin{array}{c} \exists \varphi \in \Phi,\, \exists \left((\mathbf{l}_a,\, \mathbf{m}_a),\, (\mathbf{l}_b,\, \mathbf{m}_b)\right) \in (\mathcal{D}_l \times \mathcal{D}_m)^2,\, (\mathbf{l}_a,\, \mathbf{m}_a) \neq (\mathbf{l}_b,\, \mathbf{m}_b), \\ \Omega^A . \begin{pmatrix} \mathbf{l}_a \\ \mathbf{m}_a \end{pmatrix} = \Omega^A . \begin{pmatrix} \mathbf{l}_b \\ \mathbf{m}_b \end{pmatrix} = \varphi \end{array}\right)$$

Soit à résoudre

$$\Omega^A . \begin{pmatrix} \mathbf{l}_a - \mathbf{l}_b \\ \mathbf{m}_a - \mathbf{m}_b \end{pmatrix} = 0$$

4. À une partition globale (**p**) est associé un processeur. Si bien que parler de *processeur* revient à parler de *partition*.

Ce qui donne le système d'équations suivant à résoudre (les termes ω_{j,k_j} désignent les éléments de la diagonale de la matrice correspondante, à la permutation près) :

$$\left\{\begin{array}{c} \omega^e_{1,k_1} \times \left(\mathbf{l}_{a_{k_1}} - \mathbf{l}_{b_{k_1}}\right) + \omega^i_{1,k'_1} \times \left(\mathbf{m}_{a_{k'_1}} - \mathbf{m}_{b_{k'_1}}\right) = 0 \\ \omega^e_{2,k_2} \times \left(\mathbf{l}_{a_{k_2}} - \mathbf{l}_{b_{k_2}}\right) + \omega^i_{2,k'_2} \times \left(\mathbf{m}_{a_{k'_2}} - \mathbf{m}_{b_{k'_1}}\right) = 0 \\ \vdots \\ \omega^e_{n,k_n} \times \left(\mathbf{l}_{a_{k_n}} - \mathbf{l}_{b_{k_n}}\right) + \omega^i_{n,k'_n} \times \left(\mathbf{m}_{a_{k'_n}} - \mathbf{m}_{b_{k'_1}}\right) = 0 \end{array}\right. \quad (4)$$

$$\text{avec } \forall j \in [1,\, n]_{\cap \mathbb{N}} \ k_j \in [1,\, n_e]_{\cap \mathbb{N}} \ \text{ et } k'_j \in [1,\, n_i]_{\cap \mathbb{N}}$$

Ces équations sont linéairement indépendantes (Ω^A_e et Ω^A_i sont diagonales à une permutation près) et toutes de la forme $\alpha.x + \beta.y = 0$. On peut, par conséquent, réduire le problème à la résolution d'une seule de ces équations. Considérons alors la j^{ieme} telle que $\left(\mathbf{l}_{a_{k_j}},\, \mathbf{m}_{a_{k'_j}}\right)$ et $\left(\mathbf{l}_{b_{k_j}},\, \mathbf{m}_{b_{k'_j}}\right)$ sont distincts. On a donc :

$$\omega^e_{j,k_j}.\left(\mathbf{l}_{a_{k_j}} - \mathbf{l}_{b_{k_j}}\right) + \omega^i_{j,k'_j}.\left(\mathbf{m}_{a_{k'_j}} - \mathbf{m}_{b_{k'_j}}\right) = 0$$

Or,

- le cas $\left(\left(\mathbf{l}_{a_{k_j}} - \mathbf{l}_{b_{k_j}} = 0\right) \wedge \left(\mathbf{m}_{a_{k'_j}} - \mathbf{m}_{b_{k'_j}} = 0\right)\right)$ n'est pas possible, puisque les couples $\left(\mathbf{l}_{a_{k_j}},\, \mathbf{m}_{a_{k'_j}}\right)$ et $\left(\mathbf{l}_{b_{k_j}},\, \mathbf{m}_{b_{k'_j}}\right)$ sont distincts ;
- $\left(\omega^i_{j,k'_j} = 0 \wedge \omega^e_{j,k_j}. = 0\right)$ signifie (puisqu'il y a autant d'équations que le tableau a de dimensions) que la j^{ieme} dimension du tableau *A* n'est référencée que par une unique valeur qui est 0. La redondance sur cette dimension est donc nulle et n'influence alors aucunement la redondance globale pour l'accès au tableau.

Nous ne considérerons dans le reste du document que le cas o $\omega^e_{j,k_j}. \neq 0$ et $\omega^i_{j,k'_j}. \neq 0$, car, puisque nous avons supposé qu'il n'y avait pas d'accès modulo, si l'un de ces deux termes est nul alors il n'y a pas de redondance sur cette dimension. Par conséquent, là non plus, la redondance globale pour l'accès au tableau n'est pas augmentée. De plus on a alors :

- si $\omega^e_{j,k_j}. = 0$ alors le nombre d'accès distincts sur la dimension j vaut $up\left(\mathbf{m}_{k'_j}\right)$;
- si $\omega^i_{j,k'_j}. = 0$ alors le nombre d'accès distincts sur la dimension j vaut $L_{k_j k_j}$.

2.3 Caractérisation des accès redondants à une donnée dans un tableau.

Considérons la droite δ d'équation $y = -\frac{\omega^e_{j,k_j}}{\omega^i_{j,k'_j}}x$. Les points de δ qui nous interressent sont les points $(x,\, y)$ tels que $|x| \in \mathbb{N}$ et $|y| \in \mathbb{N}$. Une solution triviale est $x_s = \omega^i_{j,k'_j}$ et $y_s = -\omega^e_{j,k_j}$. Si x_s et y_s ne sont pas premier entre eux, considérons[5] alors $x'_s = \frac{\omega^i_{j,k'_j}}{pgcd\left(\omega^e_{j,k_j},\omega^i_{j,k'_j}\right)}$ et $y'_s =$

5. On prend le point le plus proche de l'origine car il est générateur des autres. En effet, en multipliant ses coordonnées par $k \in \mathbb{Z}^*$ jusqu'à sortir du domaine, on trouve tous les autres points.

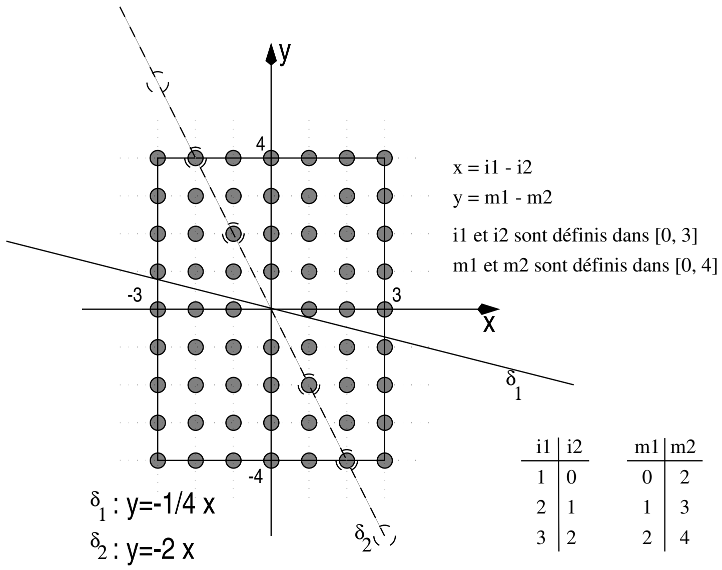
$-\dfrac{\omega^e_{j,k_j}}{pgcd\left(\omega^e_{j,k_j},\omega^i_{j,k'_j}\right)}$. Deux cas de figures se présentent alors à nous (Fig.1(a) p.5) :

1. $(x'_s,\, y'_s) \notin \left[-L_{k_jk_j}+1,\, L_{k_jk_j}-1\right]_{\cap\mathbb{Z}} \times \left[-up\left(\mathbf{m}_{k'_j}\right)+1,\, up\left(\mathbf{m}_{k'_j}\right)-1\right]_{\cap\mathbb{Z}}$
2. $(x'_s,\, y'_s) \in \left[-L_{k_jk_j}+1,\, L_{k_jk_j}-1\right]_{\cap\mathbb{Z}} \times \left[-up\left(\mathbf{m}_{k'_j}\right)+1,\, up\left(\mathbf{m}_{k'_j}\right)-1\right]_{\cap\mathbb{Z}}$

Le premier cas assure qu'il n'y a pas de solution entière dans $\left[-L_{k_jk_j}+1,\, L_{k_jk_j}-1\right]_{\cap\mathbb{Z}} \times \left[-up\left(\mathbf{m}_{k'_j}\right)+1,\, up\left(\mathbf{m}_{k'_j}\right)-1\right]_{\cap\mathbb{Z}}$ et par conséquent le nombre d'accès distincts au tableau A sur la dimension j a pour valeur :

$$L_{k_jk_j} \times up\left(\mathbf{m}_{k'_j}\right) \tag{5}$$

Le second cas indique qu'il existe des valeurs pour lesquelles deux itérations distinctes accèdent à la même donnée. Essayons de calculer le nombres de ces accès récurrents.



Du point de coordonnées (1, -2), on déduit les accès redondants: (i1=1, m1= 0) et (i2 = 0, m2 = 2) accèdent à la même donnée référencée par la valeur 2. Ainsi, tous les couples que l'on peut faire tels que x=1 et y=-2 sont des accès redondants. Dans cet exemple il y en a 9.

(a) Exemple pour deux fonctions d'accès distinctes à un même tableau : $\Omega_1 = (1 \quad 4)$ et $\Omega_2 = (2 \quad 1)$.

On a :

$$\left((x_s,\, y_s) \in \left[-L_{k_jk_j}+1,\, L_{k_jk_j}-1\right]_{\cap\mathbb{Z}} \times \left[-up\left(\mathbf{m}_{k'_j}\right)+1,\, up\left(\mathbf{m}_{k'_j}\right)-1\right]_{\cap\mathbb{Z}}\right)$$
$$\Updownarrow$$
$$\left(\begin{array}{c} \exists\left(\left(\mathbf{l}_{a_{k_j}},\, \mathbf{m}_{a_{k'_j}}\right),\, \left(\mathbf{l}_{b_{k_j}},\, \mathbf{m}_{b_{k'_j}}\right)\right) \in \left(\left[0,\, L_{k_jk_j}-1\right]_{\cap\mathbb{N}} \times \left[0,\, up\left(\mathbf{m}_{k'_j}\right)-1\right]_{\cap\mathbb{N}}\right)^2, \\ \omega^e_{j,k_j}\left(\mathbf{l}_{a_{k_j}}-\mathbf{l}_{b_{k_j}}\right)+\omega^i_{j,k'_j}\left(\mathbf{m}_{a_{k_j}}-\mathbf{m}_{b_{k_j}}\right)=0 \end{array}\right)$$

On cherche donc à calculer

$$\mathcal{C}\left(x'_s,\, y'_s\right) = card\left(\left\{\begin{array}{c} \left(\left(\mathbf{l}_a,\, \mathbf{m}_a\right),\, \left(l_b,\, m_b\right)\right) \in \left(\left[0,\, L_{k_jk_j}-1\right]_{\cap \mathbb{N}} \times \left[0,\, up\left(\mathbf{m}_{k'_j}\right)-1\right]_{\cap \mathbb{N}}\right)^2 \mid \\ \mathbf{l}_{a_{k_j}} - \mathbf{l}_{b_{k_j}} = x'_s,\, \mathbf{m}_{a_{k'_j}} - \mathbf{m}_{b_{k'_j}} = y'_s \end{array}\right\}\right)$$

Or on sait que $\forall d \in \mathbb{N},\, card\left(\left\{(a,b) \in \left([0,\, X]_{\cap \mathbb{N}}\right)^2 \mid a-b=d\right\}\right) = \max\left((X-0+1)-|d|\,,\, 0\right)$, d'où

$$\begin{aligned} \mathcal{C}\left(x'_s,\, y'_s\right) &= card\left(\left\{\begin{array}{c} \left(\mathbf{l}_a,\, \mathbf{l}_b\right) \in \left(\left[0,\, L_{k_jk_j}-1\right]_{\cap \mathbb{N}}\right)^2 \mid \\ \mathbf{l}_{a_{k_j}} - \mathbf{l}_{b_{k_j}} = x'_s \end{array}\right\}\right).card\left(\left\{\begin{array}{c} \left(\mathbf{m}_a,\, \mathbf{m}_b\right) \in \left(\left[0,\, up\left(\mathbf{m}_{k'_j}\right)-1\right]_{\cap \mathbb{N}}\right)^2 \mid \\ \mathbf{m}_{a_{k'_j}} - \mathbf{m}_{b_{k'_j}} = y'_s \end{array}\right\}\right) \\ &= \left(L_{k_jk_j} - \left|x'_s\right|\right).\left(up\left(\mathbf{m}_{k'_j}\right) - \left|y'_s\right|\right) \end{aligned}$$

Par conséquent le nombre d'accès distincts à la j^{ieme} dimension du tableau $\mathcal{A}$, noté $\mathcal{N}_j^{\mathcal{A}}$, a pour valeur :

$\forall j \in [1,\, n]_{\cap \mathbb{N}}$,

$$\begin{aligned} \mathcal{N}_j^{\mathcal{A}} &= L_{k_jk_j}.up\left(\mathbf{m}_{k'_j}\right) - \left(L_{k_jk_j} - \frac{\omega^i_{j,k'_j}}{pgcd\left(\omega^e_{j,k_j},\, \omega^i_{j,k'_j}\right)}\right).\left(up\left(\mathbf{m}_{k'_j}\right) - \frac{\omega^e_{j,k_j}}{pgcd\left(\omega^e_{j,k_j},\, \omega^i_{j,k'_j}\right)}\right) && (6) \\ &= \frac{\omega^i_{j,k'_j}.up\left(\mathbf{m}_{k'_j}\right)}{pgcd\left(\omega^e_{j,k_j},\, \omega^i_{j,k'_j}\right)} + \frac{\omega^e_{j,k_j}.L_{k_jk_j}}{pgcd\left(\omega^e_{j,k_j},\, \omega^i_{j,k'_j}\right)} - \left(\frac{\omega^i_{j,k'_j}.\omega^e_{j,k_j}}{pgcd\left(\omega^e_{j,k_j},\, \omega^i_{j,k'_j}\right).pgcd\left(\omega^e_{j,k_j},\, \omega^i_{j,k'_j}\right)}\right) && (7) \\ &= \frac{\omega^e_{j,k_j}.L_{k_jk_j}}{pgcd\left(\omega^e_{j,k_j},\, \omega^i_{j,k'_j}\right)} + \frac{\omega^i_{j,k'_j}.\left(up\left(\mathbf{m}_{k'_j}\right) - \frac{\omega^e_{j,k_j}}{pgcd\left(\omega^e_{j,k_j},\, \omega^i_{j,k'_j}\right)}\right)}{pgcd\left(\omega^e_{j,k_j},\, \omega^i_{j,k'_j}\right)} && (8) \end{aligned}$$

Des équations (5) et (8), nous déduisons l'expression du volume de données de référence pour le tableau $\mathcal{A}$:

$$\boxed{\forall\,(c,\, p)\ \mathcal{V}_{ref}\,(c,\, p) = \prod_{j=1}^{n} \min\left(\frac{\omega^e_{j,k_j}.L_{k_jk_j}}{pgcd\left(\omega^e_{j,k_j},\omega^i_{j,k'_j}\right)} + \frac{\omega^i_{j,k'_j}.\left(up\left(\mathbf{m}_{k'_j}\right) - \frac{\omega^e_{j,k_j}}{pgcd\left(\omega^e_{j,k_j},\omega^i_{j,k'_j}\right)}\right)}{pgcd\left(\omega^e_{j,k_j},\omega^i_{j,k'_j}\right)},\, L_{k_jk_j}.up\left(\mathbf{m}_{k'_j}\right)\right)^{\mathcal{A}}}$$

3 Volume de données total nécessaire à l'exécution d'un bloc de calcul

3.1 Volume de données consommées

Notre tâche $\mathcal{T}$ lit ses données dans N tableaux. Par conséquent, le volume total de données en lecture est la somme des volumes de données accédées en lecture par $\mathcal{T}$ par tableau, i.e., :

$$\boxed{\mathcal{V}_{lec}^{\mathcal{T}} = \sum_{s=1}^{N} \mathcal{V}_{ref}^{\mathcal{A}_s}}$$

Ce volume n'est exacte que si l'on accède à un tableau grâce à une et une seule fonction d'accès . En effet, dans le cas contraire, le volume de donnèes consommées calculé est une sur approximation du volume réel puisqu'alors ce type d'accès redondants n'est pas détecté.

3.2 Volume de données produites

Le volume de données en écriture est plus simple du fait de l'hypothèse de travailler sur des tableaux à assignation unique. Par conséquent, à une itération correspond une donnée. En d'autres mots, il n'y a pas de doublon, et le volume est obtenu par :

$$\boxed{\mathcal{V}_{ecr}^{\mathcal{T}} = \prod_{j=1}^{n} L_{k_j k_j}.up\left(\mathbf{m}_{k'_j}\right)}$$

3.3 Volume de données total

On en déduit alors le volume total (consommé et produit) nécessaire à l'exécution d'un bloc de calcul comme étant la somme du volume de données consommées et du volume de données produites :

$$\mathcal{V}_{total}^{T} = \mathcal{V}_{lec}^{\mathcal{T}} + \mathcal{V}_{ecr}^{\mathcal{T}}$$

4 Notations utilisées

Symbol	**Signification**
n_e	Nombre de boucles constituant le nid de boucles externe.
n_i	Nombre de boucles constituant le nid de boucles interne.
n	Dimension du tableau référencé.
$\mathbf{k}$	Vecteur pour lequel chaque composante k_j est la taille de la j^{ieme} dimension du tableau référencé. Sa dimension est n.
$\mathbf{i}$	Vecteur de boucle externe. Sa dimension est n_e .
$\mathbf{m}$	Vecteur de boucle interne. Sa dimension est n_i .
Ω_e^A	Matrice de pavage *diagonale à une permutation près* (i.e., il existe au plus un terme non nul sur chaque ligne et chaque colonne) du tableau A. Sa dimension est $n \times n_e$.
Ω_i^A	Matrice d'ajustage *diagonale à une permutation près* (i.e., il existe au plus un terme non nul sur chaque ligne et chaque colonne) du tableau A. Sa dimension est $n \times n_i$.
Ω^A	Matrice d'accès aux données du tableau A. Elle est de la forme $\left(\Omega_e^A \; \vdots \; \Omega_i^A \right)$. Transformation linéaire des vecteurs de boucles donnant un vecteur d'index. Sa dimension est $n \times (n_e + n_i)$. Implicitement on a $n \leq n_e + n_i \leq 2n$.
Φ	Espace d'index d'accès au tableau référencé. Sa dimension est n.

Symbol	Signification
φ	Vecteur d'index d'accès au tableau référencé. Sa dimension est n. On a $\varphi = \Omega^A.\begin{pmatrix} \mathbf{i} \\ \mathbf{m} \end{pmatrix} \bmod \mathbf{k}$.
$up\,(x)$	Désigne le plus petit majorant de x tel que $x < up\,(x)$. Si x est un vecteur de dimension n, alors $up\,(x) = \begin{pmatrix} up\,(x_1) \\ \vdots \\ up\,(x_n) \end{pmatrix}$.
P	Matrice de partitionnement global. Son déterminant est non nul. Sa dimension est $n_e \times n_e$.
L	Matrice de partitionnement local. Son déterminant est non nul. Sa dimension est $n_e \times n_e$.
$\mathbf{c}$	Composante vectorielle temporelle du partitionnment 3D. Sa dimension est n_e.
$\mathbf{p}_{ii}$	Composante vectorielle de partition du partitionnment 3D. Sa dimension est n_e. $\mathbf{p}_{ii}$ vérifie $0 \leq P^{-1}.\mathbf{p}_{ii} < 1$
$\mathbf{l}_{ii}$	Composante vectorielle de localité du partitionnment 3D. Sa dimension est n_e. $\mathbf{l}_{ii}$ vérifie $0 \leq L^{-1}.\mathbf{l}_{ii} < 1$
V_{ref}	Volume de données de référence.
D_x	Domaine de définition de x.
$[x_1,\, x_2]_{\cap \mathbb{N}}$	Notation utilisée pour représenter $\{x \in \mathbb{N}\, \lvert x_1 \leq x \leq x_2\}$
$\exists_k x,\, Q\,(x)$	*Il existe au plus k x tels que $Q\,(x)$.*

Références

[Gue97] C. GUETTIER. *Optimisation globale et placement d'applications de traitement du signal sur architectures parallèles utilisant la programmation logique avec contraintes.* PhD thesis, ENSMP, Novembre 1997

[Jea96] B. JEANNET. *Découpage de données et placement de tâches sur machines massivement parallèles.* Rapport de stage, École polytechnique, Août 1996