Minimisation de buffers avec des contraintes énergétiques pour les systèmes de vision embarquée

Khadija Hadj Salem, Tifenn Rault, Alexis Robbes

Minimisation de buffers avec des contraintes énergétiques pour les systèmes de vision embarquée

Khadija HADJ SALEM[1], Tifenn RAULT[1] and Alexis ROBBES[1]

Université de Tours, LIFAT EA 6300, CNRS, ROOT ERL CNRS 7002,
64 avenue Jean Portalis, 37200 Tours
{khadija.hadj-salem, tifenn.rault, alexis.robbes}@univ-tours.fr

Mots-clés : *Vision embarquée, Buffers, Job Sequencing and tool Switching Problem, Programmation Linéaire en Nombres Entiers, Programmation par contraintes*

1 Introduction

La conception des systèmes de vision embarquée engendre plusieurs problèmes d'optimisation, tels que la planification du traitement des images et des vidéos. En effet, les ressources limitées des circuits électroniques utilisés impliquent de réduire considérablement le temps de calcul, la consommation d'énergie et le coût mémoire.

Dans Hadj Salem et al. [2], le problème de planification du traitement des images a été abordé et modélisé comme une variante du problème d'outillage *Job Sequencing and tool Switching Problem* (SSP) [1], un problème $\mathcal{NP}$-difficile très étudié dans la littérature scientifique. Les premiers résultats concernant la minimisation du temps total de traitement ainsi que le nombre total de préchargements (qui représente le trafic depuis la mémoire centrale et la consommation d'énergie du circuit) ont été présentés. Cependant, la minimisation de la quantité de mémoire interne (buffers) a été uniquement mentionnée et elle sera abordée dans cette étude.

Dans ce papier, nous considérons le problème *Prefetch-Constrained Minimum Buffers Problem* (P-C-MBP) dont le critère d'optimisation considéré reflète les enjeux électroniques : encombrement et coût de production.

2 Problème de minimisation de buffers : P-C-MBP

Le problème P-C-MBP peut être formulé ainsi : on considère un jeu $\mathcal{X}$ de tuiles d'entrée qui doivent être préchargées de la mémoire externe vers des buffers, un ensemble $\mathcal{Y}$ de tuiles de sortie à calculer et un nombre de préchargements N fixé. Le calcul d'une tuile de sortie nécessite en général plusieurs tuiles de l'image d'entrée noté $\mathcal{R}_y$, où $\mathcal{R}_y \subseteq \mathcal{X}, \forall y \in \mathcal{Y}$. Il est impératif que ces tuiles soient accessibles dans les buffers internes du système au moment de ce calcul.

Le problème P-C-MBP consiste à déterminer un séquencement des préchargements de tuiles d'entrée et leurs emplacements dans les mémoires locales (buffers), et un séquencement des calculs des tuiles de sortie de manière à minimiser la quantité de buffers Z.

Dans le contexte du problème SSP, cela revient à chercher à minimiser la capacité du magasin étant donné un nombre maximal de changements d'outils sur la machine. Nous pouvons donc prouver que P-C-BMP est $\mathcal{NP}$-difficile en montrant que si un algorithme existait pour le résoudre, alors il pourrait être appelé en un nombre polynomial de fois pour résoudre le problème SSP. Par conséquent, si P-C-BMP était polynomial, le SSP le serait également.

Le problème P-C-BMP a été modélisé en Programme Linéaire en Nombres Entiers (PLNE), où nous avons considéré plusieurs versions, auxquelles nous avons introduits des contraintes de rupture de symétrie.

Nous avons également proposé un modèle de Programmation Par Contraintes (PPC). Ce choix ayant été motivé par le fait que peu de travaux se sont intéressés à la PPC pour résoudre le problème SSP.

3 Expérimentations

Nous avons mené des expérimentations numériques, sur un ensemble d'instances de la littérature de SSP (disponible à cet adresse http ://www.unet.edu.ve/∼jedgar/ToSP/ToSP.htm) ainsi qu'un ensemble d'instances réelles générées par Mancini et al. [3] et Hadj Salem et al. [2], afin de jauger l'efficacité des approches proposées. Les instances réelles ont été réduites lors d'un pré-traitement en se basant sur une propriété de dominance qui supprime chaque tuile de sortie nécessitant un sous-ensemble de tuiles d'entrée déjà utilisé au moins une fois par une autre tuile de sortie.

Nous avons utilisé le solveur IBM ILOG Cplex et CP Optimizer pour la résolution du PLNE et du modèle PPC avec un temps limite de 300 secondes. Pour comparer l'efficacité des approches proposées, le Gap(%) est calculé comme suit : $100 * (Z_{\text{PLNE}} - Z_{\text{PPC}})/Z_{\text{PPC}}$.

Les tests préliminaires sur les instances de la littérature de SSP, où $|\mathcal{X}| \in [9, 60]$ et $|\mathcal{Y}| \in [10, 50]$, montrent que les deux modèles sont capables de résoudre dans le temps imparti les instances, avec $|\mathcal{X}| < 15$ et $|\mathcal{Y}| < 10$. Pour les autres, les deux modèles donnent des résultats un peu similaires.

Instances	$\lvert\mathcal{X}\rvert$	$\lvert\mathcal{Y}\rvert$	$Z_{\textbf{PLNE}}$	$\textbf{CPU}_{\textbf{PLNE}}$	$Z_{\textbf{PPC}}$	$\textbf{CPU}_{\textbf{PPC}}$	**Gap(%)**
test_2D_0	256	64	**4**	101.39	**4**	0.09	**0**
test_2D_1	64	256	**1**	2,5	**1**	0,11	**0**
test_polaire_0	146	90	146	300	68	300	114.7
test_polaire_1	80	60	80	–	46	–	73.91
test_polaire_2	244	82	244	–	142	–	71.83
test_fisheye_0	176	103	176	–	109	–	61.46
test_fisheye_1	224	103	224	–	139	–	61.15
test_fisheye_2	360	103	360	–	230	–	56.52
test_fd_0	429	300	429	–	349	–	22.92
test_fd_1	2272	206	2272	–	2081	–	9.17

TAB. 1 – Comportement de PPC versus PLNE sur les instances réelles [3]

A partir de la Table 1, nous pouvons observer que la PPC produit des résultats de meilleurs qualités que ceux obtenus par la résolution du PLNE. Néanmoins, des tests approfondis, plusieurs améliorations et des comparaisons avec d'autres approches doivent être envisagés.

Références

[1] Catanzaro, D. and Gouveia, L. and Labbé, M. *Improved integer linear programming formulations for the job Sequencing and tool Switching Problem. European Journal of Operational Research*, 244 : 766–777, 2015.

[2] Hadj Salem, K. and Kieffer, Y. and Mancini, M. *Meeting the Challenges of Optimized Memory Management in Embedded Vision Systems Using Operations Research. Recent Advances in Computational Optimization : Results of the Workshop on Computational Optimization WCO 2016* : 177–205, 2018.

[3] Mancini, S. and Rousseau, F. *Enhancing non-linear kernels by an opti-mized memory hierarchy in a high level synthesis flow. Conference on Design, Automation and Test in Europeh*,1130–1133, 2012.