THALES

Techniques for Dynamic Workload Partitioning in High-Performance Heterogeneous Computing Platforms

WILDER LOPES

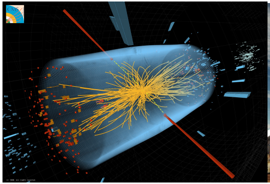
20 October 2016

The research leading to these results has received funding from the People Programme (Marie Curie Actions) of the European Union's Seventh Framework Programme FP7/2007-2013/ under REA grant agreement No. (317446) INFIERI "INtelligent Fast Interconnected and Efficient Devices for Frontier Exploitation in Research and Industry".

wilderlopes@gmail.com

Introduction

- INFIERI project – research network supported by an European FP7 (Marie Curie Actions)
- Thales: expertise in high performance embedded computing (Duhem's talk)
- WP 4: Massive Parallel Computing
  WP 6: Test Platforms / Benches
- Dynamic allocation of data-parallel kernels in heterogeneous architectures



Réf.: TRT-Fr/STI/LCHP/WL,16/0021
Thales Research & Technology France
wilderlopes@gmail.com

Preliminaries

- A pool of devices:
  - Central Processing Unit (CPU)
  - Graphics Processing Unit (GPU)
  - Field-Programmable Gate Array (FPGA)
- Best way to perform the partitioning? Several criteria:
  - Execution time
  - Data transfer
  - Power consumption

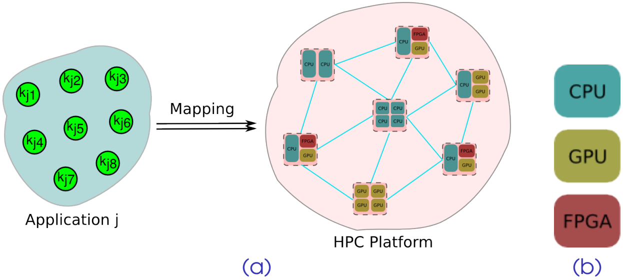


Figure : (a) Kernels of application j that must be mapped onto the pool of computational resources. (b) The correspondent colors for each device.

Réf.: TRT-Fr/STI/LCHP/WL,16/0021
Thales Research & Technology France
wilderlopes@gmail.com

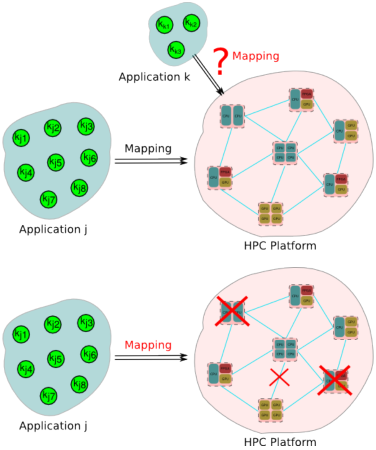


Figure : Changes at runtime

- Changes at runtime
  - Type and amount of devices
  - Amount of kernels
  - Applications requirements – Quality of Service (QoS)
- In the literature:

  Graph partitioning (1)
  Machine-learning techniques (2, 3)
  Programming heuristics (4, 5, 6)
- However, they are targeted at static partitioning

Réf.: TRT-Fr/STI/LCHP/WL,16/0021
Thales Research & Technology France
wilderlopes@gmail.com

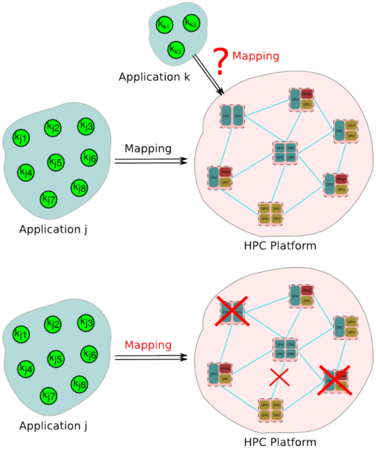


Figure : Changes at runtime

GOAL

Develop a system manager able to sense and react at runtime to variations in the High-Performance Heterogeneous Computing platform as well as in the QoS requirements

Réf.: TRT-Fr/STI/LCHP/WL,16/0021
Thales Research & Technology France
wilderlopes@gmail.com

Problem Formulation

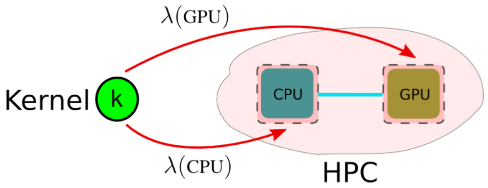


Figure : A simple workload partitioning scenario.

$$\lambda_k = [\lambda(\text{CPU})\ \lambda(\text{GPU})],$$
$$\lambda(j) \in [0,1] \text{ and } \sum_{j}^{N} \lambda(j) = 1$$

- Simple Scenario: single-kernel data-parallel application
- Only two devices: CPU and GPU
- Workload partitioning ⇔ Data partitioning
- $\lambda(j)$ is the amount of data that should be allocated to each device $j = \{CPU, GPU\}$

Réf.: TRT-Fr/STI/LCHP/WL,16/0021
Thales Research & Technology France
wilderlopes@gmail.com

Design of the System Manager

- Strategy: minimize the discrepancy between the required and profiled performances of the data-parallel application
- It is assumed a performance profile of the kernel is available
- QoS requirements: application dependent
- Feedback from the HPC: Measurement of devices performances in order to update profiles
- Metric: kernel execution time

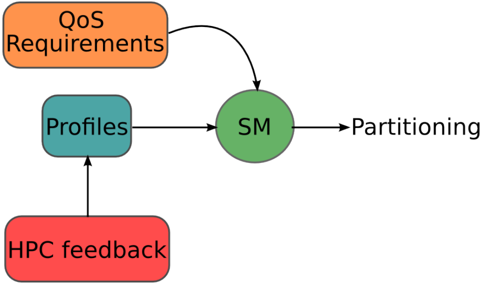
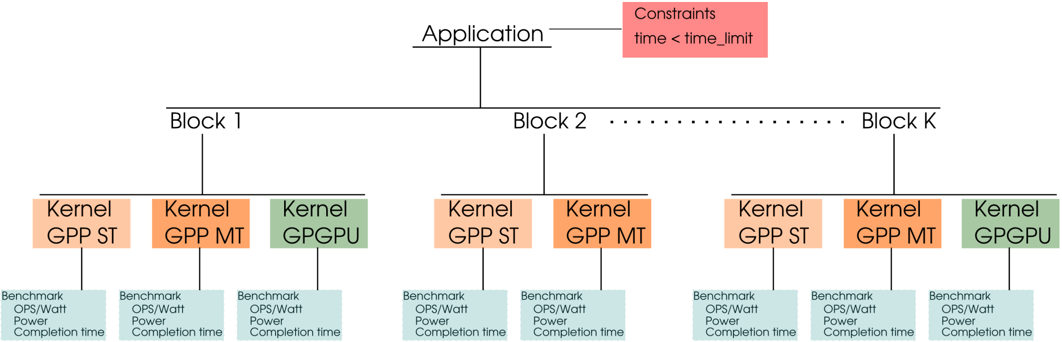


Figure : Strategy for dynamic partitioning

wilderlopes@gmail.com

Application Profiling



Réf.: TRT-Fr/STI/LCHP/WL,16/0021
Thales Research & Technology France
wilderlopes@gmail.com

Mathematical Formulation

- Error e_k: measure of how well the resources can be combined to achieve the QoS requirements

$$e_k = u\left(p_k^o - \sum_j^N \lambda(j) r_j\right) = u(p_k^o - R\lambda_k) \qquad (1)$$

- List of requirements p_k^o
- Matrix R: profiles of the application at each device.

wilderlopes@gmail.com

Mathematical Formulation

- Error e_k: measure of how well the resources can be combined to achieve the QoS requirements

$$e_k = u\left(p_k^o - \sum_j^N \lambda(j) r_j\right) = u\left(p_k^o - R\lambda_k\right) \qquad (1)$$

- List of requirements p_k^o
- Matrix R: profiles of the application at each device.

- Minimization problem $\Rightarrow$ Mean-square criteria

$$\boxed{\begin{aligned} &\min_\lambda J(\lambda) = \mathsf{E}|e_k|^2 = \mathsf{E}\left|\boldsymbol{u}\left(\boldsymbol{p}_k^o - R\lambda_k\right)\right|^2, \\ &\text{subject to} \quad \lambda(j) \geq 0, j \in [1, N], \\ &\qquad\qquad\quad \mathbb{1}^T \lambda_k = \sum_{j=1}^N \lambda(j) = 1 \end{aligned}} \qquad (2)$$

Réf.: TRT-Fr/STI/LCHP/WL,16/0021
Thales Research & Technology France
wilderlopes@gmail.com

The Intelligence Embedded in the System Manager

- Adaptive Filters: naturally fit for real-time estimation without previous training (as opposed to machine learning-based techniques)
- Able to track variations in the HPC resources (matrix R) and in the QoS requirements (vector p_k^o)
- Easy to be scaled up towards several applications (kernels) and computing devices

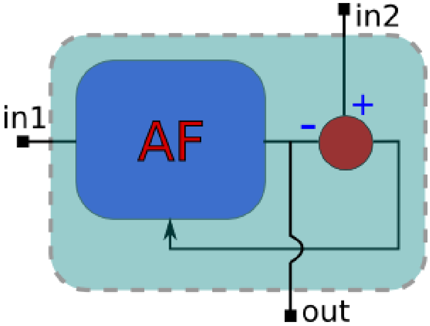


Figure : Schematics of an Adaptive Filter

The Intelligence Embedded in the System Manager

Constrained Least-Mean-Square Filter (7, 8)

$$\boxed{\lambda_{k,i} = P[\lambda_{k,i-1} + \mu R^{-1} D_{R\lambda} u_i^T e_k(i)] + F} \quad (3)$$

$$\begin{aligned} P &= I - \frac{1}{N}\mathbb{1}\mathbb{1}^T \\ F &= \frac{1}{N}\mathbb{1}, \end{aligned} \quad (4)$$

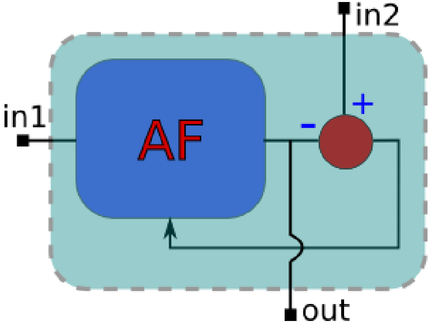


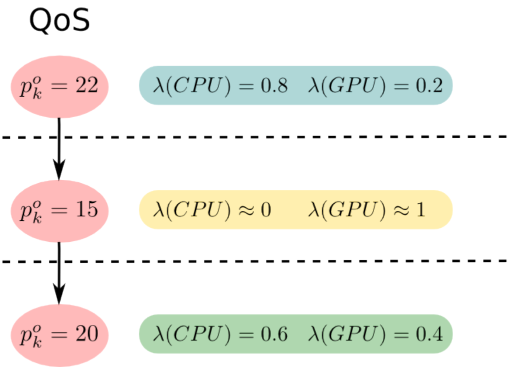
Figure : Schematics of an Adaptive Filter

Réf.: TRT-Fr/STI/LCHP/WL,16/0021
Thales Research & Technology France
wilderlopes@gmail.com

Experiments

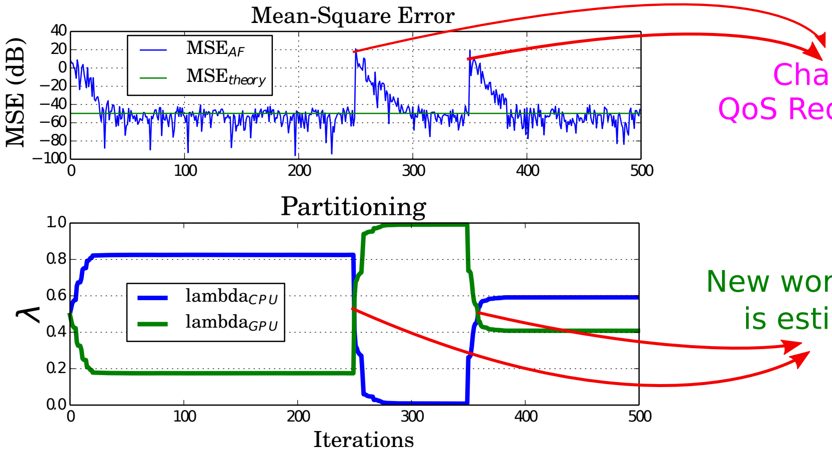
- Test: variation in the QoS requirements

  p_k^o changes while R remains fix
- Profile:
  $R = [23.5$ (CPU) 14.93 (GPU) $]$ seconds
- The adaptive filter is able to provide a new workload partitioning at runtime



Réf.: TRT-Fr/STI/LCHP/WL,16/0021
Thales Research & Technology France
wilderlopes@gmail.com

Experiments



Change in
QoS Requirement

New workload partitioning
is estimated at runtime

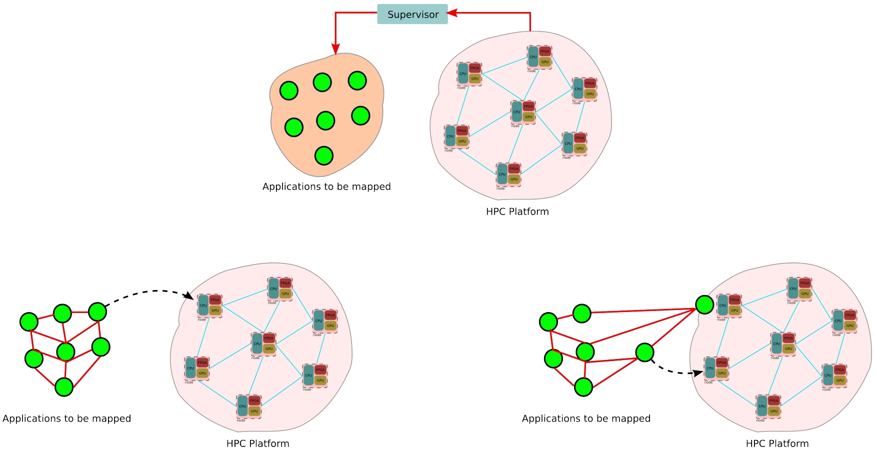
Réf.: TRT-Fr/STI/LCHP/WL,16/0021
Thales Research & Technology France
wilderlopes@gmail.com

Conclusions

- The strategy successfully tracks variations in p_k^o (QoS requirements)
- Next 1: Track variations in the computing devices and update R (HPC feedback)
- Next 2: Perform exhaustive tests in a real scenario – video processing application

Réf.: TRT-Fr/STI/LCHP/WL,16/0021
Thales Research & Technology France
wilderlopes@gmail.com

Distributed System Manager - Near Future



Réf.: TRT-Fr/STI/LCHP/WL,16/0021
Thales Research & Technology France

wilderlopes@gmail.com

References I

Irene Moulitsas and George Karypis,
Architecture Aware Partitioning Algorithms, pp. 42–53,
Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

Dominik Grewe and Michael F. P. O'Boyle,
"A static task partitioning approach for heterogeneous systems using opencl,"
in *Proceedings of the 20th International Conference on Compiler Construction: Part of the Joint European Conferences on Theory and Practice of Software*, Berlin, Heidelberg, 2011, CC'11/ETAPS'11, pp. 286–305, Springer-Verlag.

Klaus Kofler, Ivan Grasso, Biagio Cosenza, and Thomas Fahringer,
"An automatic input-sensitive approach for heterogeneous task partitioning,"
in *Proceedings of the 27th International ACM Conference on International Conference on Supercomputing*, New York, NY, USA, 2013, ICS '13, pp. 149–160, ACM.

Michael D. Linderman, Jamison D. Collins, Hong Wang, and Teresa H. Meng,
"Merge: A programming model for heterogeneous multi-core systems,"
SIGPLAN Not., vol. 43, no. 3, pp. 287–296, Mar. 2008.

Chi-Keung Luk, Sunpyo Hong, and Hyesoon Kim,
"Qilin: Exploiting parallelism on heterogeneous multiprocessors with adaptive mapping,"
in *Proceedings of the 42Nd Annual IEEE/ACM International Symposium on Microarchitecture*, New York, NY, USA, 2009, MICRO 42, pp. 45–55, ACM.

References II

J. Shen, A. L. Varbanescu, Y. Lu, P. Zou, and H. Sips,
"Workload partitioning for accelerating applications on heterogeneous platforms,"
IEEE Transactions on Parallel and Distributed Systems, vol. 27, no. 9, pp. 2766–2780, Sept 2016.

J. Chen, C. Richard, J. C. M. Bermudez, and P. Honeine,
"Nonnegative least-mean-square algorithm,"
IEEE Transactions on Signal Processing, vol. 59, no. 11, pp. 5225–5235, Nov 2011.

Marcello L. R. de Campos, , Stefan Werner, and José A. Apolinário,
Constrained Adaptive Filters, pp. 46–64,
Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.