23th International Conference on Architecture of Computing Systems 2010

ARCS '10

Workshop Proceedings

February 22 – 23, 2010
Hannover, Germany

Michael Beigl, Francisco J. Cazorla-Almeida (Editors)
PARMA: Workshop on Parallel Programming and Run-time Management Techniques for Many-core Architectures

**SESSION 1: Programming models and languages, compilers and virtualization techniques**

**Improved Programming of GPU Architectures through Automated Data Allocation and Loop Restructuring** ................................................................. 271
Andrea Di Biagio, Giovanni Agosta, Politecnico di Milano

**Extending a Light-weight Runtime System by Dynamic Instrumentation for Performance Evaluation** ................................................................. 279
Mario Kicherer, Fabian Nowak, Rainer Buchty, Wolfgang Karl, Karlsruhe Institute of Technology

**Parallelism and Retargetability in the ILDJIT Dynamic Compiler** ................................................................. 285
Michele Tartara, Simone Campanoni, Giovanni Agosta, Stefano Crespi Reghizzi, Politecnico di Milano

**SESSION 2: Runtime management, power management and memory management**

**CPM: A Cross-Layer Framework to Efficiently Support Distributed Resources Management** ........ 293
Patrick Bellasi, Stefano Bosisio, Matteo Carnevali, William Fornaciari Politecnico di Milano, David Siorpaes, STMicroelectronics

**An Efficient Run-Time Management Methodology for a Stereo Matching Application** ............... 299
Giovanni Mariani, University of Lugano, Chantal Ykman-Couvreur, Ke Zhang, Lu Zhang, Gauthier Lafruit IMEC

**Dynamic Frequency Scaling for MPSoCs-based on Chaotic Workload Analysis** .......................... 305
Nikolaos Zompakis, National Technical University of Athens, Vasileios Tsoutsouras, Democritus University of Thrace, Alexandros Bartzas, National Technical University of Athens, Dimitrios Soudris, National Technical University of Athens, Georgios Pavlos, Democritus University of Thrace

**SESSION 3: Design space exploration and many-core architecture customization**

**Modeling Separate Memory Spaces in Native Co-simulation with SystemC for Design Space Exploration** ......................................................... 313
Hector Posadas, University of Cantabria, Santander, Eugenio Villar, University of Cantabria, Santander

**Fast Design Space Exploration Environment Applied on NoC’s for 3D-Stacked MPSoC’s** ........ 319
Alienor Richard, Dragomir Milojevic, Frederic Robert, Bio Electro and Mechanical Systems, Université Libre de Bruxelles, Alexandros Bartzas, Antonis Papanikolaou, Kostas Siozios, Dimitrios Soudris, National Technical University of Athens

**Multicube Explorer: An Open Source Framework for Design Space Exploration of Chip Multi-Processors** ......................................................... 325
V. Zaccaria, G. Palermo, F. Castro, C. Silvano, Politecnico di Milano, G. Mariani, University of Lugano
Multicube Explorer: An Open Source Framework for Design Space Exploration of Chip Multi-Processors

V. Zaccaria, G. Palermo, F. Castro, C. Silvano
Dipartimento di Elettronica e Informazione - Politecnico di Milano
Milano, Italy

G. Mariani
AlaRI - University of Lugano
Lugano, Switzerland

Abstract

Given the increasing complexity of Chip Multi-Processors (CMPs), a wide range of architecture parameters must be explored at design time to find the best trade-off in terms of multiple competing objectives (such as energy, delay, bandwidth, area, etc.). The design space of the target architecture is huge because of the need to consider all possible combinations of each parameter (number of processors, processor issue width, L1 and L2 cache sizes, etc.). In this complex scenario, the multi-objective exploration of the huge design space of next generation CMPs cannot be anymore based on intuition and past experience of the design architects. An Automatic Design Space Exploration methodology is needed to support systematically the exploration and the quantitative comparison of the design alternatives in terms of multiple competing objectives (Pareto analysis). An overall design space exploration framework is needed to combine all optimizations into a global search space with a common interface to the simulation and evaluation tools. Our work focuses on the definition of an automatic multi-objective Design Space Exploration (DSE) framework for tuning Chip Multi-Processor architectures by evaluating a set of metrics (such as energy and delay) for the next generation embedded computing platforms.

Multicube Explorer is an interactive open-source framework to enable the designer to automatically explore a design space of configurations for a parameterized architecture for which an executable model (use case simulator) exists. Multicube Explorer is an advanced multi-objective optimization framework which is entirely command-line/script driven and can be re-targeted to any configurable platform by writing a suitable XML design space definition file and providing a configurable simulator.

1 Introduction

Multi-Processor Systems-on-Chip (MPSoCs) and Chip Multi-Processors (CMPs) represent the current de facto standard in both the embedded and general-purpose worlds. Customizable multi-core architectures supported by parallel programming paradigms have become the dominant computing paradigm for application-specific processors. For these multi-core and many-core architectures, the platform-based design approach [7] has been widely used to meet time-to-market constraints. In this approach, configurable simulation models have been used to tune the on-chip architectures and to meet the target application requirements in terms of performance, battery lifetime and area.

In this scenario, the Design Space Exploration (DSE) phase is used to tune the configurable system parameters and it generally consists of a multi-objective optimization problem. The DSE problem consists of exploring a large design space consisting of several parameters at system and micro-architectural levels. Although several heuristic techniques have been proposed so far to address this problem, they are all characterized by low efficiency to identify the Pareto front of feasible solutions. Among these heuristics, evolutionary or sensitivity-based algorithms represent the most notable, state-of-the art techniques.

The emphasis on the efficiency of the DSE algorithms is due to the fact that in computer architecture research and development, simulation still represent the main tool to predict performance of alternative architectural design points. If we consider cycle-accurate system-level simulation, it requires a lot of simulation time and the exploration of the design alternatives can exceed practical limits. Furthermore, the growing trend towards chip multi-processor architectures amplifies this problem because the simulation speed linearly decreases by increasing the number of cores to be simulated. While from the simulation point of view, the use of statistical sampling techniques seems to represent the best solution [11], from the design space exploration point of view, efficient solutions that reduce the number of architectural alternatives to be analyzed are still missing.

Although several tools exist to optimize each single phase of designing of multi-processor systems, an overall design space exploration framework is needed at system-level to combine all optimization steps into a global search strategy exploiting a common interface for the optimization and performance evaluation tools. To fill this gap,
the main contribution of this work consists of introducing Multicube Explorer, an automatic design space exploration framework and a related methodology which leverages traditional Design of Experiments and Response Surface Modeling techniques.

Design of Experiments (DoE) [13] techniques have been used to identify the design of an information-gathering experimentation plan where a set of tunable parameters can vary. Design of experiments is a discipline that has been widely used in the past across natural and social sciences and encompasses a set of techniques whose main goal is the screening and analysis of the system behavior with a small number of experiments. Each DoE technique differs in terms of the layout which specifies how to select the design points in the design space.

Response Surface Modeling (RSMs) techniques describe the analytical relation between several design parameters and one or more response variables. The main idea is to use a set of experiments generated by DoE to obtain a response model of the system's behavior. An RSM-based design flow usually includes a training phase, in which a set of known data (or training set) is used to tune the RSM configuration, and a prediction phase in which the RSM is used to predict unknown system response.

To the best of our knowledge, while there have already been some other applications of DoEs and RSM techniques to the field of performance analysis and optimization. Multicube Explorer represents the first comprehensive application of DoE and RSM techniques to the field of multi-objective design space exploration of on-chip multiprocessors [10].

The present paper focuses on the description of an automatic multi-objective Design Space Exploration (DSE) framework for tuning Chip Multi-Processor architectures by evaluating a set of metrics (such as energy and delay) for the next generation embedded computing platforms. Multicube Explorer is an interactive open-source framework to support the automatic exploration of several design space of configurations for a target architecture for which an executable and configurable model is provided. Multicube Explorer is an advanced multi-objective optimization framework which is entirely command-line/script driven and can be re-targeted to any configurable platform by writing a suitable XML design space definition file and providing a configurable simulator.

The paper is organized as follows. Section 2 introduces some theoretical background and the state-of-the-art related to design space exploration, while Section 3 introduces the architecture of the Multicube Explorer design space exploration tool. Section 4 reports some experimental results of the exploration of an MPEG2 decoder platform obtained by using Multicube Explorer. Finally, some concluding remarks have been outlined in Section 5.

2 Introduction to design space exploration

Design Space Exploration (DSE) consists of an optimization process which takes into account a typical set of architecture parameters mainly associated with the memory hierarchy (e.g., cache size), the parallelism of the processor (e.g., number of cores processors and issue width) and the on-chip interconnect configuration. The optimization problem involves the minimization (maximization) of multiple objectives (such as latency, energy, area, etc.) making the definition of optimality not unique [3]. In fact, a system which is the best from the performance point of view, can be the worst in terms of power consumption and vice-versa.

In our context, the multi-objective optimization targets a set of \( n \) system configuration parameters grouped on a configuration vector:

\[
x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \in X
\]

where \( X \) is usually a finite, discrete domain (subset of \( N^n \)). The multi-objective optimization problem is defined as a set of \( m \) objective functions to be minimized (or maximized):

\[
\min_{x \in X} f(x) = \left[ \begin{array}{c} f_1(x) \\ \vdots \\ f_m(x) \end{array} \right] \in \mathbb{R}^m
\]

subject to a set of \( k \) constraints which, without loss of generality, can be expressed as:

\[
e(x) = \left[ \begin{array}{c} e_1(x) \\ \vdots \\ e_k(x) \end{array} \right] \leq \left[ \begin{array}{c} 0 \\ \vdots \\ 0 \end{array} \right]
\]

The set of all the admissible or feasible solutions of an optimization problem is called feasible region and it is defined as:

\[
\Phi = \{ x \in X \mid e_i(x) \leq 0, 1 \leq i \leq k \}
\]

In single-objective optimization problems, the feasible set is totally ordered according to the objective function \( f(x) \), thus a single exact solution \( x \) does exist. However, when several objectives are involved, the feasible set is partially ordered by a dominance relation, thus multiple exact solutions exist. The multiple solution set is called the exact Pareto set of the problem. In order to formally define the concept of Pareto set, we need to introduce the dominance relation. We say that \( f \) dominates \( g \) when:

\[
f \prec g = \left\{ f_i \leq g_i, \forall i = 1, \ldots, m \text{ and } f_i < g_i \text{ for at least one } i \right\}
\]

Given a subset of feasible configurations \( \Omega \subseteq X \), we define the Pareto set \( \Phi \) associated to \( \Omega \) as:

\[
\Phi(\Omega) = \left\{ x \mid \left( \Phi \cap \Omega \right) \wedge \exists y \in \Phi(\Omega) \text{ s.t. } f(y) < f(x) \right\}
\]
The Pareto set $\Psi(\Phi)$ is the exact Pareto set of the problem. Based on set theory, the projection of $\Psi(\Phi)$ in the objective space is called the exact Pareto front.

Whenever the considered solution set $\Omega$ is a subset of the feasible solution space $\Phi$, the Pareto set $\Psi(\Omega)$ is called an approximate Pareto set of the problem.

![Diagram](image)

**Figure 1:** The concept of Pareto Dominance, $\Omega = \{x_a, x_b, x_c\}$. Point $x_c$ is dominated by point $x_b$.

Figure 1 shows a feasible set of solutions $\Omega = \{x_a, x_b, x_c\}$ for an unconstrained minimization problem for $[f_1, f_2]$. Point $x_c$ is dominated by point $x_b$ since both $f_1(x_c)$ and $f_2(x_c)$ are greater than $f_1(x_b)$ and $f_2(x_b)$. In this case we thus have that $\Psi(\Omega) = \{x_a, x_b\}$.

2.1 Previous Work

Several methods have been recently proposed in literature to reduce the design space exploration complexity by using conventional statistic techniques and advanced exploration algorithms.

Among the most recent heuristics to support power/performance architecture exploration, we can find [1, 9]. In [9], the authors compare Pareto Simulated Annealing, Pareto Reactive Taboo Search and Random Search exploration to identify energy-performance trade-offs for a parametric super-scalar architecture executing a set of multimedia kernels. In [1], a combined Genetic-Fuzzy system approach is proposed. The technique is applied to a highly parametrized SoC platform based on a VLIW processor in order to optimize both power dissipation and execution time. The technique is based on a Strength Pareto Evolutionary Algorithm coupled with fuzzy system rules in order to speedup the evaluation of the system configurations.

State-of-the-art, system performance optimization has been presented in [4–6, 8]. A common trend among those methods is the combined usage of Response Surface Modeling and Design of Experiments techniques.

In [5], a Radial Basis Function has been used to estimate the performance of a super-scalar architecture; the approach is critically combined with an initial training sample set that is representative of the whole design space, in order to obtain a good estimation accuracy. The authors propose to use a variant of the Latin Hypercube method in order to derive an optimal, initial set. In [6, 8] linear regression has been used for the performance prediction and assessment. The authors analyze the main effects and the interaction effects among the processor architectural parameters. In both cases, random sampling has been used to derive an initial set of points to train the linear model. A different approach is proposed in [4], where the authors tackle performance prediction by using an Artificial Neural Network paradigm to estimate the system performance of a Chip-Multiprocessor.

In [14–16], a Plackett-Burman design of experiments is applied to the system architecture to identify the key input parameters. The exploration is then reduced by exploiting this information. The approach shown in [14] is directed towards the optimization of an FPGA, soft-core-based design, while in [16] the target problem is more oriented to a single, super-scalar processor.

3 Architecture of Multicube Explorer

The overall goal of the Multicube Explorer framework is to provide an open source re-targetable tool to drive the designer towards near-optimal solutions to the architectural exploration problem, given multiple constraints. The final product of the framework is a Pareto curve of configurations within the design evaluation space of the given architecture.

![Diagram](image)

**Figure 2:** Structure of Multicube Explorer

The tool provides a command line interface to the exploration kernel that enables the construction of automated exploration strategies. Those strategies are implemented by means of command scripts interpreted by the tool without the need of any manual intervention. This structure can easily support the batch execution of complex strategies that are less prone to human intervention, due to their execution time. The tool is portable across a wide range of systems. This goal is achieved by not sacrificing the efficiency of the overall exploration engine. The standard ANSI C++ programming language is used for developing the open source framework. The Standard Template Library as well as other open source libraries are used during the development process.
Table 1: Optimizers, design of experiments (DoEs) and response surface models (RSMs) available in Multicube Explorer

<table>
<thead>
<tr>
<th>Optimizers</th>
<th>DoEs</th>
<th>RSMs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adaptive window Pareto Random Search</td>
<td>Full Search</td>
<td>Linear regression</td>
</tr>
<tr>
<td>Multi-Objective Simulated Annealing</td>
<td>Random</td>
<td>Spline</td>
</tr>
<tr>
<td>Multi-Objective Particle Swarm</td>
<td>Two Level Full Factorial</td>
<td>Radial Basis Functions</td>
</tr>
<tr>
<td>Non-dominated Sorting Genetic Alg.</td>
<td>Two Level Full Factorial Extended</td>
<td>Shepard</td>
</tr>
<tr>
<td>Simple Evolutionary Multi-Obj. Opt.</td>
<td>Scrambled</td>
<td>Neural Network</td>
</tr>
<tr>
<td>Pareto DoE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

One of the strength of the open-source tool is the modularity of its components. Simulator, optimization algorithms and other design space exploration components are dynamically linked at run-time, without the need of recompiling the entire code base. This will be supported by well-defined interfaces between the drivers supporting the simulation and the optimization algorithm. This will strongly support the introduction of new modules for both academic and industrial purposes. Given the modular decomposition, a single optimization algorithm can be used for every use case simulator. Moreover, a single use case architecture can be optimized with a wide range of optimization algorithms.

The tool (Figure 3) is basically composed of an exploration kernel which orchestrates the functional behavior of the design of experiments and optimization algorithms.

![Figure 3: Architecture of the tool](image)

The kernel is responsible for reading in the design space definition file (in XML format) and accepting commands from the shell interface (or the corresponding script). It then exposes the parameters of the design space to the modules involved in the optimization process (DoE, Optimization Algorithms) by means of a core design space representation.

The core design space representation provides a set of abstract operations, that are mapped on the specific use case under analysis. The abstract operations are represented by iterators over the feasible design space, among which we can find:

- Full search iterators.
- Random search iterators, (global and neighborhood).
- Factorial iterators (two-level, two-level + center point).

The core design space representation also provides some services for validating architectural choices at the optimizer level and evaluating the associated objective functions. The objective functions are defined as a subset of the use case system level metrics and can be user manipulat- ed by interacting with Multicube Explorer.

The implemented plug-in modules (optimizers, design of experiments and response surface models) are shown in Table 1.

Concerning the interaction with the simulator, the design space exploration is executed by using the simulation abstraction layer exported by the XML driver to the optimizer plug-ins. In principle, the optimizer instantiates a set of architectural configurations throughout the design space iterators, and it passes the corresponding representation to the XML driver which will execute the simulator (see Figure 4). Information about simulator runs will be displayed directly on the Multicube Explorer shell.

![Figure 4: Interaction of Multicube Explorer with the Use Case Simulator](image)

Then, Multicube Explorer generates a specific directory to execute each instance of the simulator. In this directory, a valid system parameters file is created before starting the simulator. A system metrics file is expected to be obtained as the output of the simulator execution.

4 Design Space Exploration with Multicube Explorer: The MPEG Decoder Case Study

In this section, we present some of the analysis features of Multicube Explorer and some experimental results obtained by using Multicube Explorer to customize a CMP architecture for the execution of an MPEG2 decoder application. The target architecture is a shared-memory multiprocessor with private L2 cache. We focused our analysis
on the architectural parameters listed in Table 4, where the minimum and maximum values have been reported. Globally, the resulting design space consists of $2^{17}$ (131 072) alternative configurations.

### Table 2: Design space for the shared-memory multiprocessor platform

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td># Processors</td>
<td>2</td>
<td>16</td>
</tr>
<tr>
<td>Processor core width</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>L1 instruction cache size</td>
<td>2K</td>
<td>16K</td>
</tr>
<tr>
<td>L1 data cache size</td>
<td>2K</td>
<td>16K</td>
</tr>
<tr>
<td>L2 cache size</td>
<td>32K</td>
<td>256K</td>
</tr>
<tr>
<td>L1 instruction cache assoc.</td>
<td>1w</td>
<td>8w</td>
</tr>
<tr>
<td>L1 data cache assoc</td>
<td>1w</td>
<td>8w</td>
</tr>
<tr>
<td>L2 cache assoc</td>
<td>1w</td>
<td>8w</td>
</tr>
<tr>
<td>L2 block size</td>
<td>16</td>
<td>32</td>
</tr>
</tbody>
</table>

To carry out the system metrics evaluation, we leveraged the SESC [12] simulation tool, a fast simulator for chip-multiprocessor architectures with out-of-order processors that is able to provide energy and performance results for a given application. Within SESC, the energy consumption computation for the memory hierarchy is supported by CACT [15], while the energy consumption computation due to the core logic is based on the WATTCH models [2]. The exploration flow is composed of a full-search design of experiments over the entire design design space of the use case. We selected a subset of the system metrics to be used as objective functions for the analysis. For this preliminary design space exploration, we selected the following, relevant system metrics:

- **cycles**: Actual number of cycles of execution of the target application on the architecture.
- **energy**: Energy consumption (in Joule) associated with the architecture when executing the target application.

In this section, we discuss some experimental results derived from the exploration of the MPEG Decoder use case with the overall purpose to explain what kind of exploration analysis could be carried out by using **Multicube Explorer**. The output of the tool is a **HTML** file format from which the designer can browse the exploration results in terms of the objective design space, the Pareto curve, parameters main effects and parameter interactions.

### Objective Space Analysis

Figure 5 shows the scatter plot of the objective functions (Cycles and Energy) associated with the architecture design points by focusing on the parameter **pn** number of processors (**Multicube Explorer** can generate an objective space analysis file focused on each parameter of the design space). As can be seen, a higher number of processors corresponds to a lower number of cycles, but it could also imply an increment of energy in some cases.

From the multi-objective space, **Multicube Explorer** can derive the Pareto curve corresponding to each pair of objective functions. Figure 6 shows a snapshot of the HTML file generated by **Multicube Explorer** representing the Pareto curve to analyze performance and energy trade-offs for the MPEG use case.

![Figure 5: Objective space analysis: Cycles and Energy associated with the design points focusing on the number of processors.](image-url)
**Main effects analysis.** By using Multicube Explorer, the overall impact of the design parameters on the objective functions can be evaluated. For the given case study, the impact of the parameters on the average values of the system level cycles and energy is shown in Figure 7 and Figure 8 respectively. The graphs have been generated by considering the average difference on the objective function by passing from the minimum(-) to the maximum (+) value for each parameter averaged over all the other parameters. We can definitely identify a strong dependence of the execution cycles with respect to the number of processors, the size of the level 2 cache and the issue width. The influence of the remaining set of parameters is relatively weaker. Finally, we can note (Figure 8) that the energy consumption increases with the data cache size and the issue width, but it decreases with the associativity of the data cache.

**Figure 6:** Snapshot of the html file generated by Multicube Explorer showing the Cycles and Energy Pareto curve corresponding to the overall objective space points

**Figure 7:** Analysis of parameters main effects on cycles.

**Figure 8:** Analysis of parameters main effects on energy.

**5 Summary**

In this paper, we presented Multicube Explorer, an interactive framework for design space exploration of parameterized CMP architectures based on a target executable model (use case simulator). Multicube Explorer is a multi-objective optimization framework which is entirely command-line/script driven and can be re-targeted to any configurable platform by writing a suitable XML design space definition file and providing a configurable simulator. In particular, we have presented some of the main characteristics of the analysis tools which are available in Multicube Explorer.

**References**

[1] G. Ascia, V. Catania, A. G. Di Nuovo, M. Palesi, and D. Patti. Efficient design space exploration for appli-


