Design Space and Application Autotuning for Runtime Adaptivity in Multicore Architectures

Cristina Silvano
Politecnico di Milano
cristina.silvano@polimi.it
Energy efficiency underlies all markets

- **Energy efficiency** is of paramount importance for all application markets (automotive, consumer, mobile, healthcare and beyond) and target systems spanning from sensors, cyber-physical systems, embedded systems up to servers and HPC systems.
The design space in the age of multi/many-cores

- A wide range of architecture parameters must be tuned to find the best system configuration.
- Design space of the target architecture $A$ should consider all possible configurations of each parameters $p_i$:
  \[ A = S_{p1} \times S_{p2} \times \ldots \times S_{pn} \]

<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>#Threads</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>L1 I$ size</td>
<td>2K</td>
<td>16K</td>
</tr>
<tr>
<td>L1 D$ size</td>
<td>2K</td>
<td>16K</td>
</tr>
<tr>
<td>L1 I$ associativity</td>
<td>1w</td>
<td>8w</td>
</tr>
<tr>
<td>L1 D$ associativity</td>
<td>1w</td>
<td>8w</td>
</tr>
<tr>
<td>L2 $ size</td>
<td>32K</td>
<td>256K</td>
</tr>
<tr>
<td>L2 $ associativity</td>
<td>1w</td>
<td>8w</td>
</tr>
</tbody>
</table>

Large design space: $2^{18}$ (262 144) design points
FULL SEARCH
Can become quickly unfeasible

~10 minutes per simulation*

262,144 design points
×
8 data sets
=
2,097,152 simulations

10 Years

* Using a cycle-accurate simulator
The multi-objective optimisation problem

- **Objective function:** To minimize both energy $\varepsilon(x)$ and execution time $\delta(x)$ of the target application on system configurations $x$:

$$
\min_{x \in X} \omega(x), \quad \omega(x) = \begin{bmatrix} \varepsilon(x) \\ \delta(x) \end{bmatrix}
$$

where $X$ is the design space.

- The solution is a set of tradeoff configurations $X_p \subseteq X$ known as Pareto set.
Multi-objective exploration: Pareto Points

Multi-Objective Exploration: **best designs are not unique**. **Pareto points** provide tradeoffs with respect to the multiple objectives.
1. Design of Experiments (DoEs):
   To identify the experimentation plan: how to select the
design points in the design space to be simulated:
random, full factorial, central composite design.

2. Optimisation Algorithms:
   Meta-heuristics methods inspired by analogies with
physics, or with biology to solve multi-objective
optimization problems: simulated annealing, genetic
algorithms, evolutionary strategies, etc.

3. Response Surface Modeling (RSM):
   To use the set of simulated points to obtain an
analytical model of the system behavior: linear
regression, spline interpolation, artificial neural
network, etc.
How to combine DoEs and Response Surface Models?

- ReSPIR: Response Surface-based Pareto Iterative Refinement

APPLICATION AUTOTUNING
Key idea is that most of the applications are configurable at design-time in terms of a set of tunable parameters, code transformations and code variants (application knobs) to trade-off accuracy and latency.
Tunable Applications: Software Knobs

- Application Space
  - Application Parameters
  - Source to Source Code Transformations
  - Compiler Flags

- Target Independent Space

- Target Dependent Space
  (e.g. Intel Xeon & Xeon Phi)

- App

- .bin
Quality of Results: Pixel Disparity Error

Left camera  Right camera

stereo-matching

Reference disparity map

Application Knobs

Disparity Error

1 2 3
Design-time optimization of OpenCL applications


mARGOt Application Autotuning

- **mARGOt** is a **light-weight application autotuning framework** for many-core platforms in an adaptive multi-application environment.

  - Combination of **design-time** and **run-time** techniques to create an effective way of “self-aware” computing with **limited runtime overhead**
  - **Adaptivity** to runtime adjust the application “behavior” (configuration) to the changing operating conditions, usage contexts and resource availability
  - **Approximate computing** to trading off accuracy and throughput while the output just needs to be “good enough”

---


Runtime feedback loop

- **Application autotuning** enables self-optimization capabilities based on Monitor-Analyze-Plan-Execute (MAPE) feedback loop*

APPLICATION AUTOTUNING AND RUNTIME RESOURCE MANAGEMENT
Orthogonality Concept: App Autotuning & RTRM

Orthogonality Concept: App Autotuning & RTRM

 Requests

 Resources

 Platform OS

 Target HW Platform

 Resource Availability
Orthogonality Concept: App Autotuning & RTRM
Autotuning aNd adaptivity AppRoach for Energy efficient eXascale HPC systems

Prof. Cristina Silvano, Politecnico di Milano
cristina.silvano@polimi.it
Context:

• To meet DARPA’s target of **20MW** of Exascale supercomputers
  • Current supercomputers must achieve an energy efficiency “quantum leap”
  • **10x** energy efficiency from 5 to 50 GFlops/W
• Next generation supercomputers need to be coupled with a radically **new software stack** capable of exploiting the benefits offered by heterogeneity to meet the scalability and energy efficiency required by the Exascale era.
ANTAREX Motivations

Context:

ANTAREX wants to provide a breakthrough approach

- to express by a Domain Specific Language the application self-adaptivity and non-functional requirements
- to runtime monitor, manage and autotune applications for green HPC systems up to the Exascale level

benefits offered by heterogeneity to meet the scalability and energy efficiency required by the Exascale era.
LARA DSL for Extra Functional Specifications

- To enable separation of concerns: functional and extra-functional descriptions are decoupled.
- Useful to express strategies for instrumentation and profiling.
- Support for explore compiler optimization space, according to code and target architectures.
- To enable more advanced control than using pragmas/directives/switches (e.g. application parameters and mapping control).

C/C++ w/ OpenMP, MPI, OpenCL
ANTAREX Runtime Framework

- Compile Time
- Deploy Time
- Run Time

Job Scheduler

Allocator

COff

COn

COff

COn

COff

COn

A

Autotuner

A

Collector

A

Power Manager

Node1

Node2

J1

JN

Job

Scheduler

Allocate

AM

Offline Compiler

Online Compiler

Hardware Monitor

Application Monitor

Compile Time

Deploy Time

Run Time

Cooling

DB

Nodei

NodeN

Rack1

RackK
Autotuning framework implements a **MAPE-K loop** to make the application behaviour self-aware.

ANTAREX DSL is used to instrument the application for monitoring and to support autotuning.
ExaMon: Monitoring System & Power Management

Flexible & scalable: Integrated with Linux Perf and MQTT
High-precision and low overhead (programmable events):

- 480 PMU metrics for node @ 2s : < 0.2% of overhead
- High level metrics: top-down analysis (TMAM), roofline

Suitable to be integrated in a production environment:
- Running on 64 nodes of CINECA Galileo ~ 15k metrics/s
Use Case 1: HPC Accelerated Drug Discovery System
Use Case 1: HPC Accelerated Drug Discovery System

Personalized Medicine will enable to “treat the right patient with the right drug at the right dose at the right time.” [FDA]

Need of HPC in Drug Discovery: HPC Molecular Simulations
LiGen in Antarex

Developing energy and resource efficient algorithms
Using self-functionalities to adapt and scale-out the application

LiGen: Exascale-ready HPC Application

Exascale HPC Virtual Screening

- Tangible Chemical Space: 300 Bio
- Peptide library: 120 Mio
- Commercial libraries: 6 Mio
- SafeInMan library: 9 K
Molecular docking is a method to calculate the preferred position and shape of one molecule w.r.t a second one when bound to each other.

- **Geometric Docking**
  - **Shape complementarity:** Geometric matching search to find out compatible pairs and most suitable poses

- **Pharmacophoric Docking**
  - Molecular simulation: Exploration of a large energy landscape determined by chemical and physical interactions
Autotuning with mARGOtt

Low-Precision Step

- #Cores/Nodes
- time_to_solution
- Ligand-DB MetaData

High-Precision Step

- #Iterations
- Size Threshold

MINIAPP

Ligand-DB

Score degradation [%]

Throughput per process [atoms/s]
Use Case 2: Self-adaptive Navigation System
**Use Case 2: Self-adaptive Navigation System**

**Sygic Company** develops the world's most popular navigation application & provides professional navigation software for business solutions.

Exploit synergies between client-side and server-side:
- Many drivers – many routing requests to HPC system
- Traffic status data sources
- Continuous update of traffic flow calculation
- **Smart City Challenge**
Autotuning with mARGO\textit{t} (User & Data-aware)

- **Data Feature Extraction**
- **Profiling Knowledge**
- **OP List**
- **Application Requirements**
- **#Samples**
- **User type**
- **Path**
- **Time + Path + Speed Profiles**
- **MCS**
- **Travel-Time Probability Distribution**
http://www.antarex-project.eu/
Research Collaborations