



MULTICUBE: Multi-Objective Design Space Exploration of Multi-Core Architectures

Cristina Silvano

Politecnico di Milano, Milano (ITALY)

Dipartimento di Elettronica e Informazione

`cristina.silvano@polimi.it`

`http://home.dei.polimi.it/silvano/`

MULTICUBE Project

MULTI-OBJECTIVE DESIGN SPACE EXPLORATION OF MULTI-PROCESSOR SOC ARCHITECTURES FOR EMBEDDED MULTIMEDIA APPLICATIONS

www.multicube.eu

Project Duration: from January 2008 to June 2010



Politecnico di Milano (POLIMI) – Italy **(Project Coordinator)**



DS2 – Spain



Università della Svizzera Italiana
(ALaRI) - CH



IMEC - Belgium



University of Cantabria - Spain



STMicroelectronics - Italy



STMicroelectronics - China



ESTECO - Italy



Institute of Computing Technology (ICT)
China

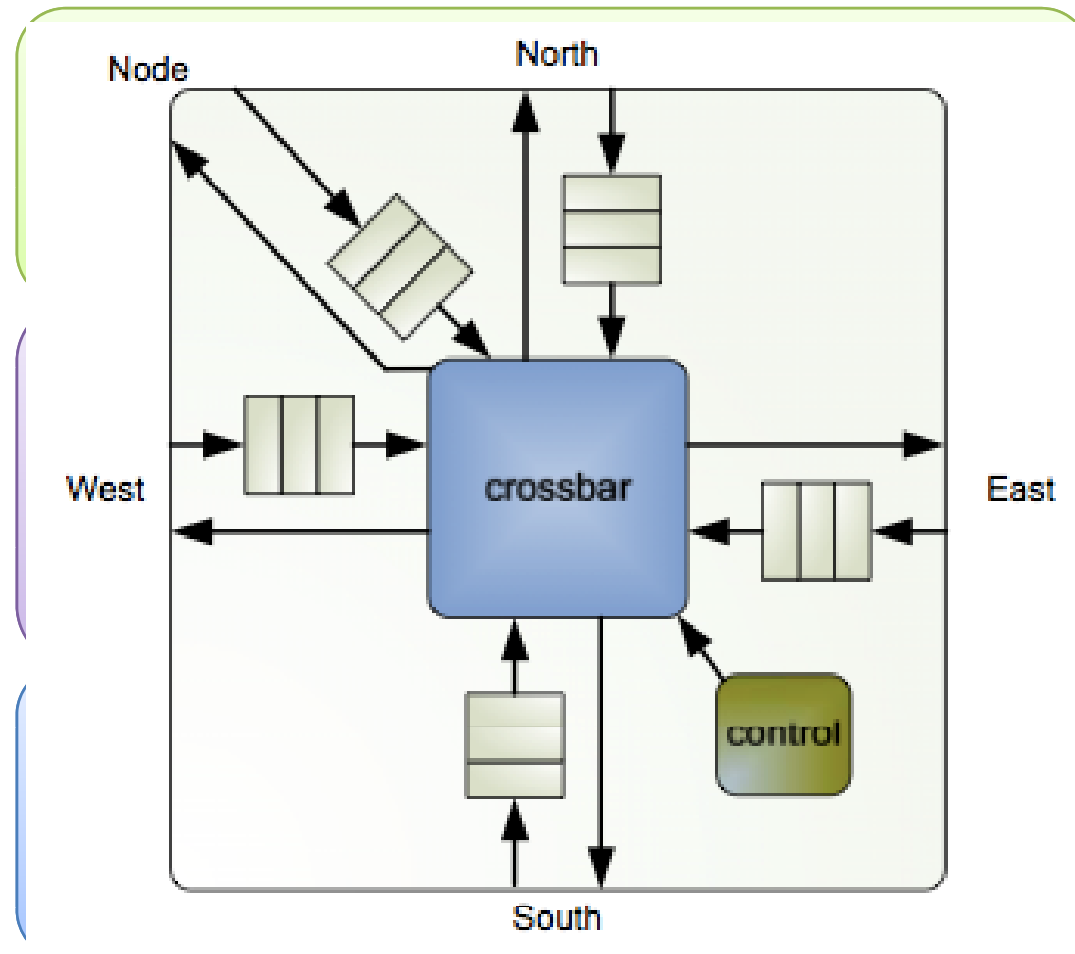
Why Automatic Design Space Exploration?



Multi-core architecture



Institute of Computing Technology,
Chinese Academy of Sciences
中科院计算所



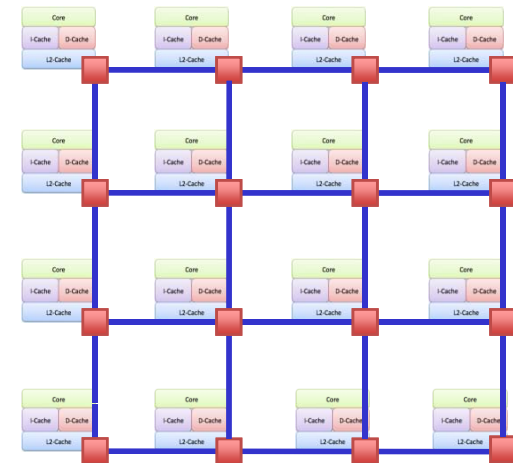
The design space

- In the age of multi/many-core, 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 :

$$A = S_{p1} \times S_{p2} \times \dots \times S_{pn}$$

- Example:

Parameter	Min.	Max.
# Processors	2	16
Processor issue width.	1	8
L1 instruction cache size	2K	16K
L1 data cache size	2K	16K
L2 private cache size	32K	256K
L1 instruction cache assoc.	1w	8w
L1 data cache assoc.	1w	8w
L2 private cache assoc.	1w	8w
I/D/L2 block size	16	32



⇒ Large design space composed of 2^{17} (131,072) system configurations

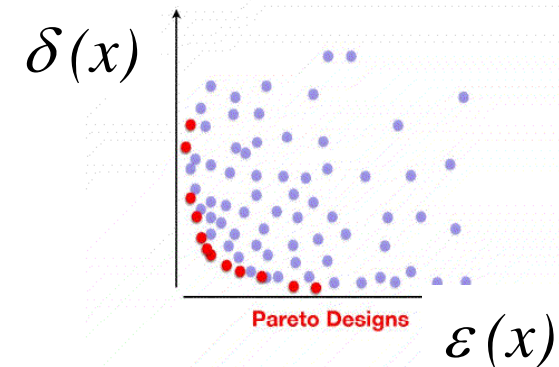
The multi-objective optimisation problem

- **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 **trade-off** configurations $X_p \subseteq X$ known as Pareto set



The concepts behind the automatic DSE

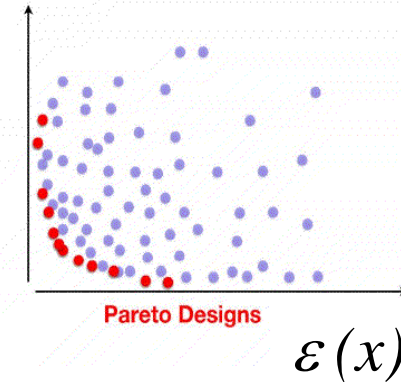
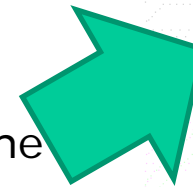
Output Variables define the objective space

The black box generates the output values accordingly to the inputs.

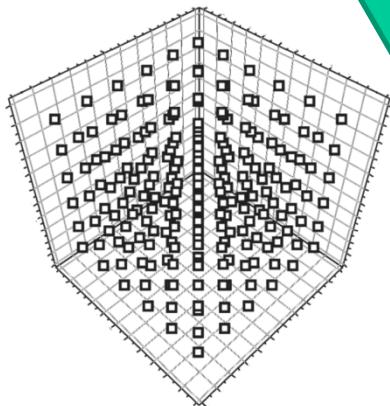
$\delta(x)$

The black box can be:

- 1) A simulator that models the system behavior and generates output values
- 2) A set of solvers that models the system behavior and estimates output values

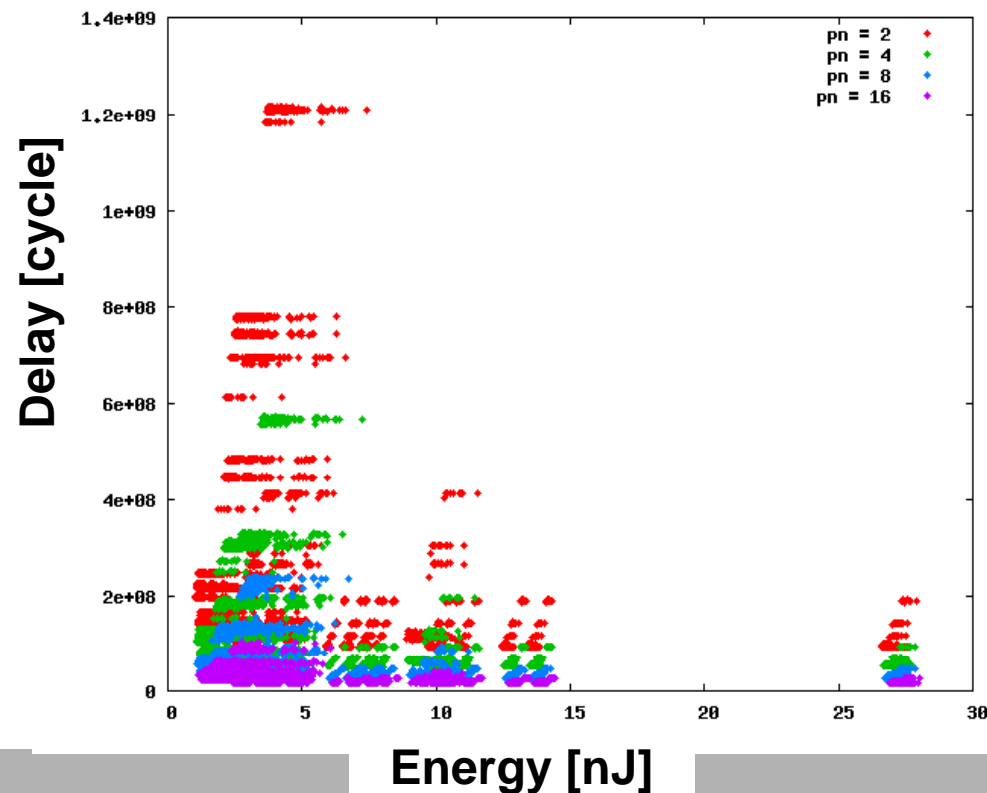


Input Variables: architecture parameters that define the design space



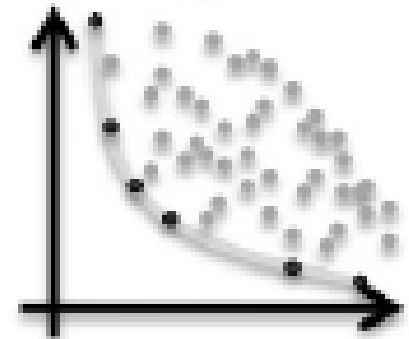
Full Search Design Space Exploration

- **Full-search exploration** is often unfeasible because it takes too much time
- Example: Design space composed of $2^{17} = 131,072$ system configurations. If simulation of the target application for each configuration requires 5 min \Rightarrow \sim 3 months on 5 parallel machines for full-search exploration of the target application



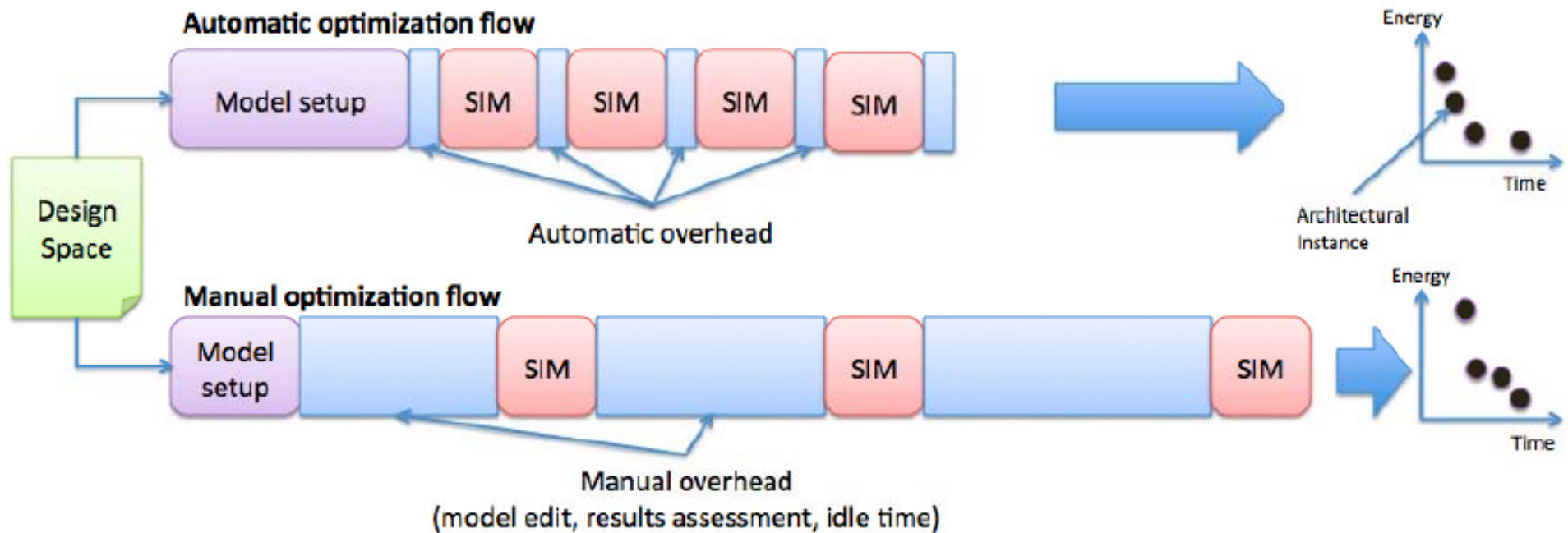
Why Automatic Design Space Exploration?

- Given the increasing complexity of **Chip Multi-Processors**, a wide range of **architecture parameters** must be explored to find the best trade-off in terms of **multiple objectives** (energy, delay, bandwidth, area, etc.)
- **Multi-Objective Exploration** of the huge design space of next generation CMPs cannot be anymore a manual optimisation process based on intuition and past experience of the designer
- **Need for Automatic Design Space Exploration** to support systematically the exploration and the quantitative comparison in terms of multiple competing objectives (**trade-offs analysis**)



Motivations

- Why Automatic Design Space Exploration?
 1. Faster exploration time
 2. Better quality of results



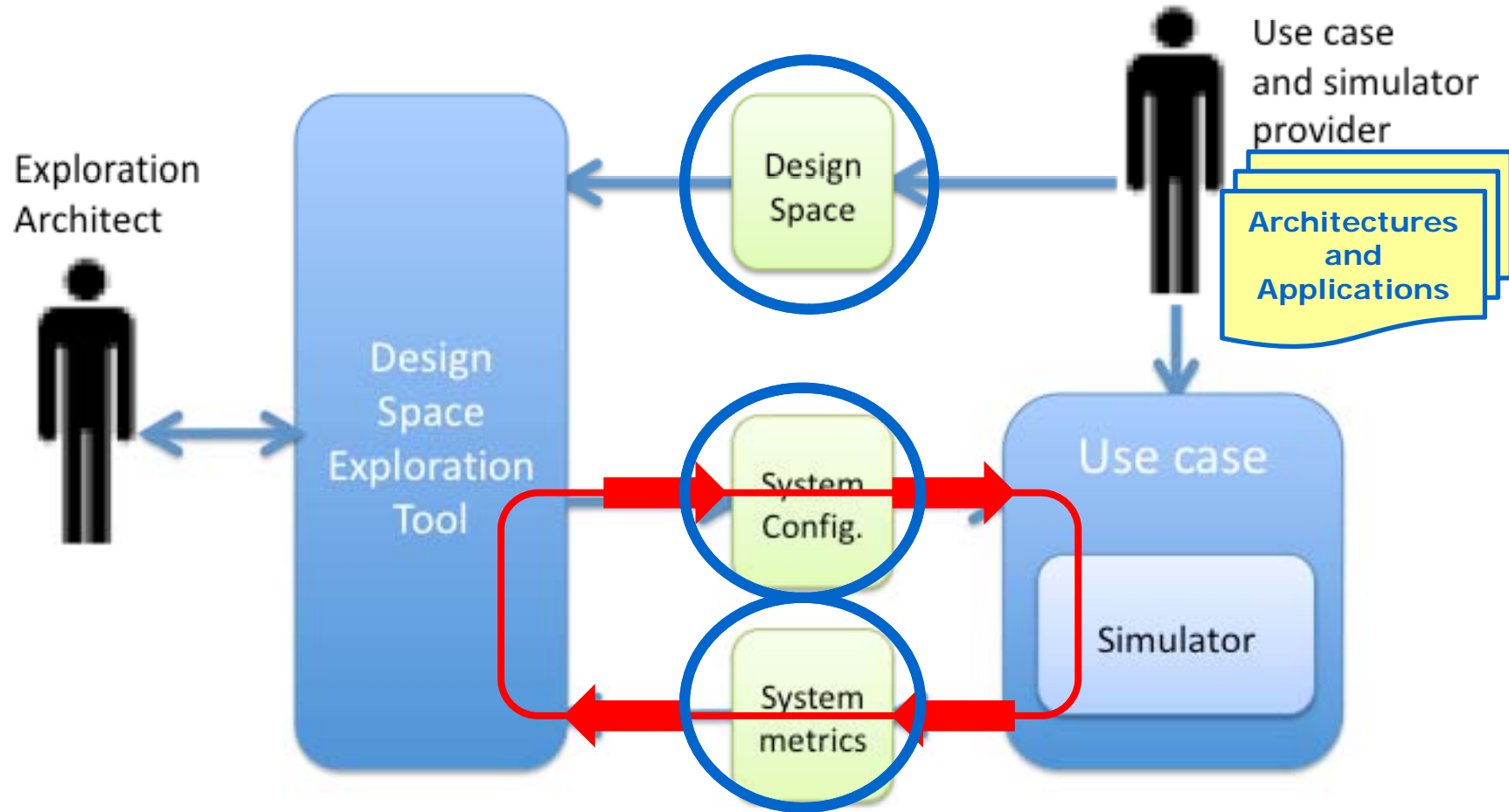
MULTICUBE Project

- An **overall design space exploration framework** is needed to combine simulation and optimization techniques into a global search space with a common interface to the simulation and optimisation tools.
- **MULTICUBE FP7-ICT Project** focuses on the definition of an **automatic multi-objective Design Space Exploration (DSE) framework** to be used to tune Chip Multi-Processor architectures evaluating a set of metrics (such as energy and delay) for the next generation embedded computing platforms.

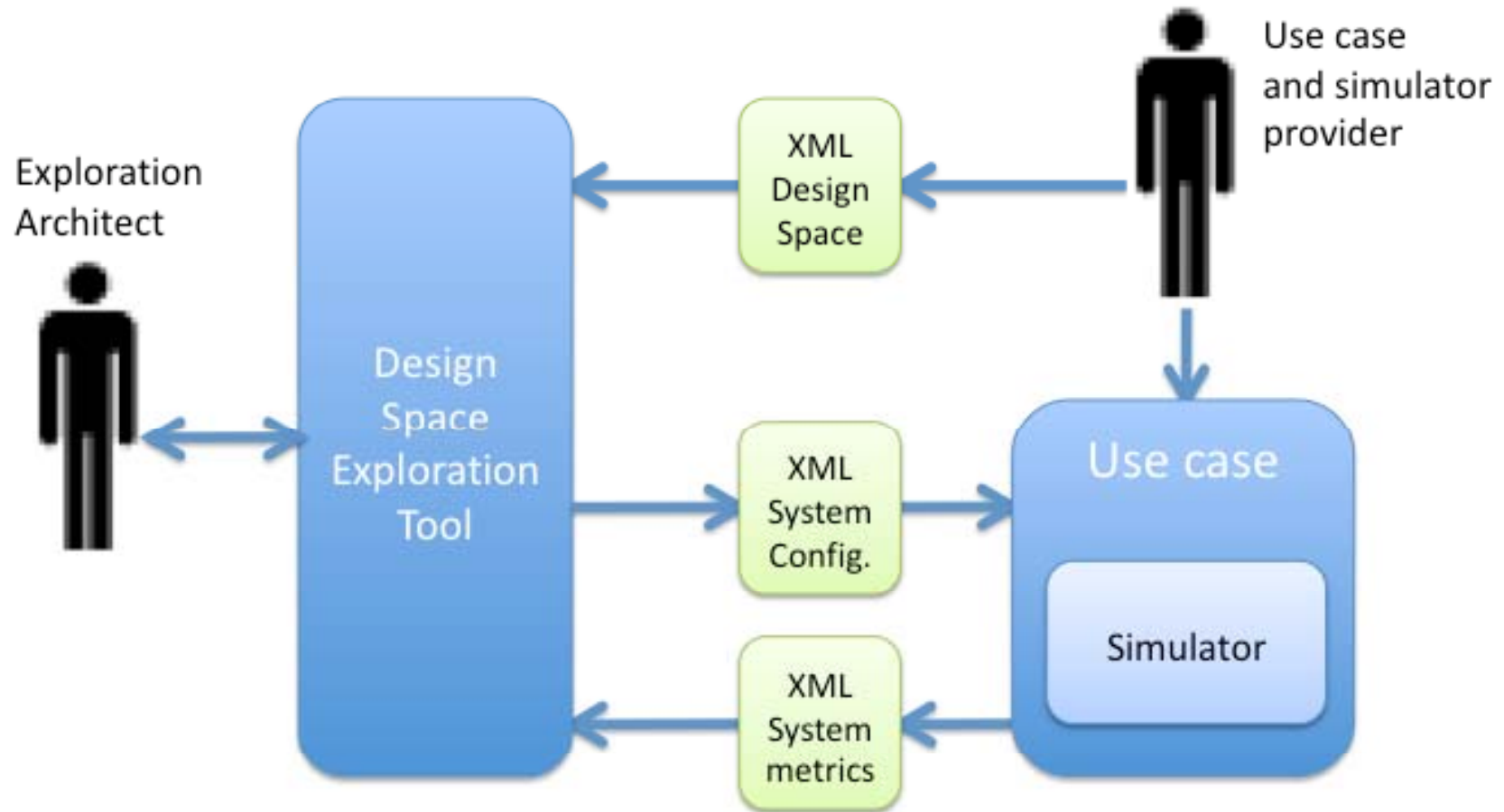
www.multicube.eu



Exploration framework: an overview

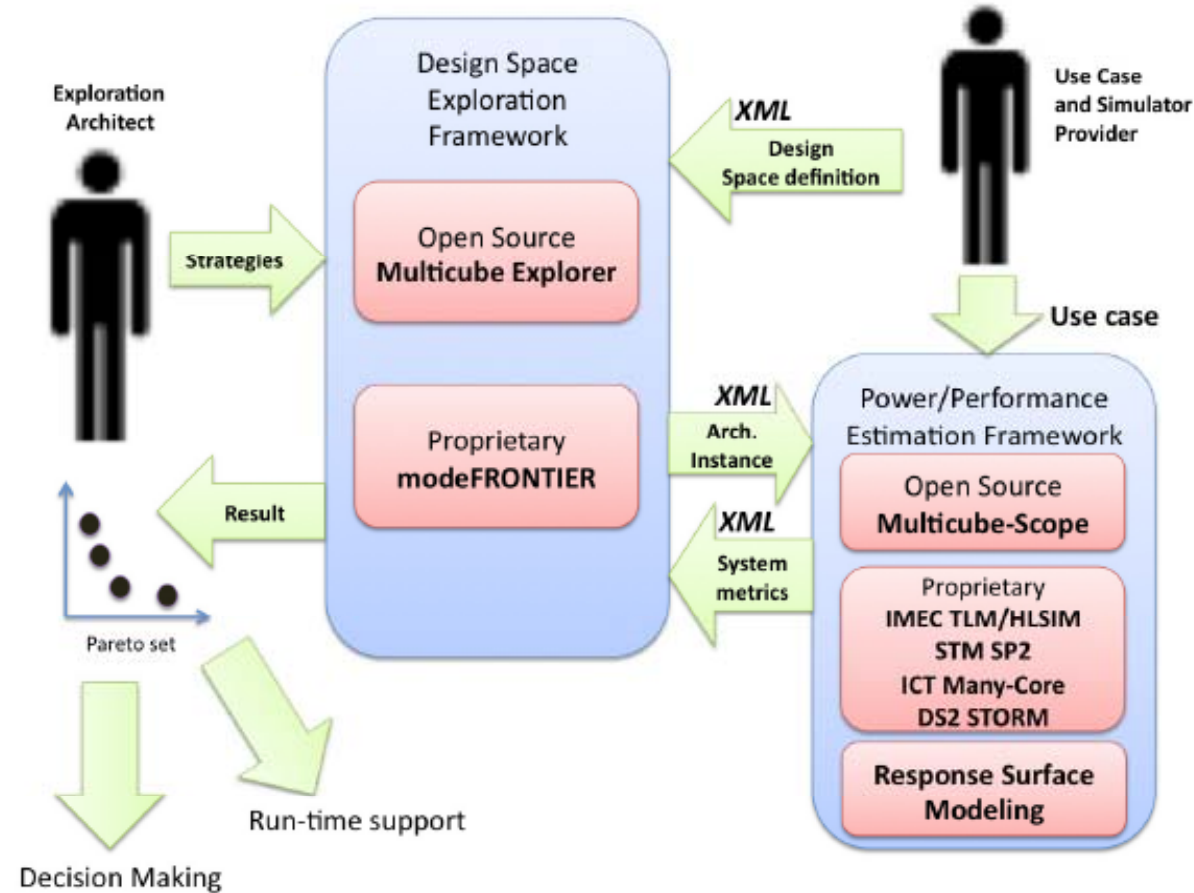


XML-based Tool Interface



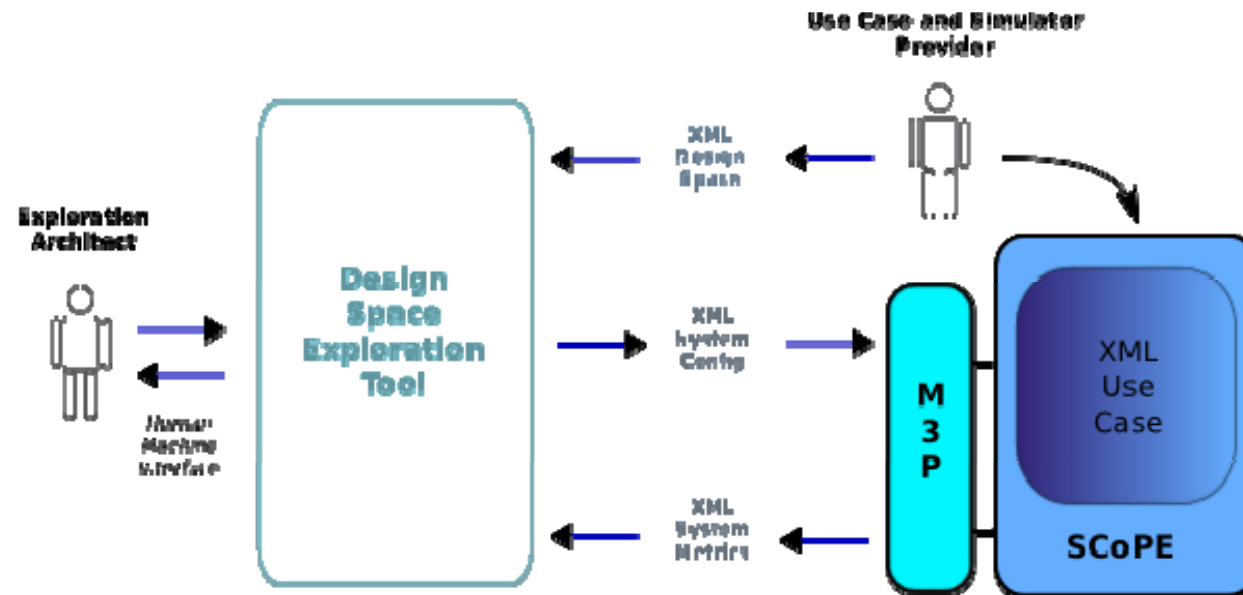
Flexibility: MULTICUBE Explorer is a design optimization and exploration framework where you can easily plug-in your own simulator by defining the XML interface

MULTICUBE Design Flow

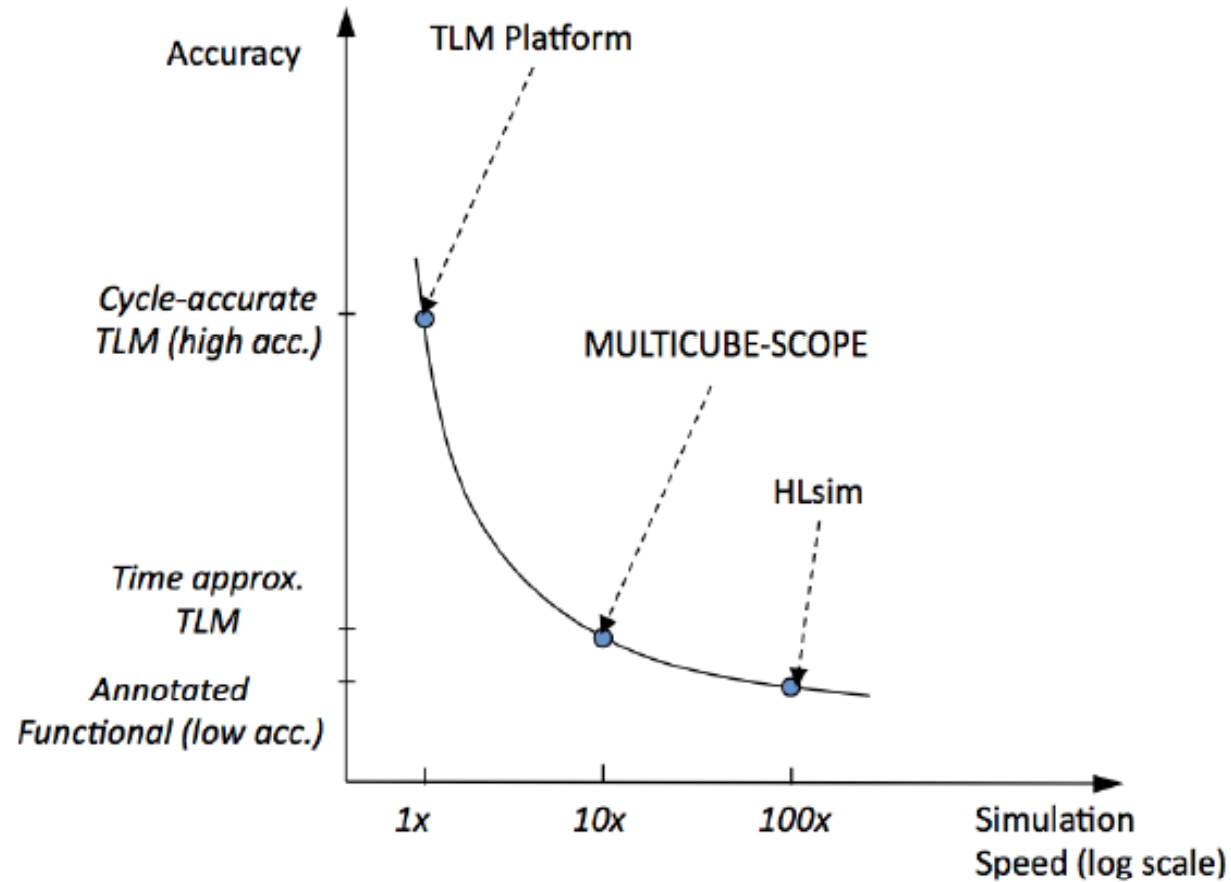


MULTICUBE-SCoPE

- **Open-source prototype tool (MULTICUBE-SCoPE) for performance and power estimation.** The tool can dynamically generate system models corresponding to the configuration parameters received from the DSE tools and can feed back the required system metrics to guide the exploration process.



High-Level Performance Estimation by using Simulators at Multiple Abstraction Levels

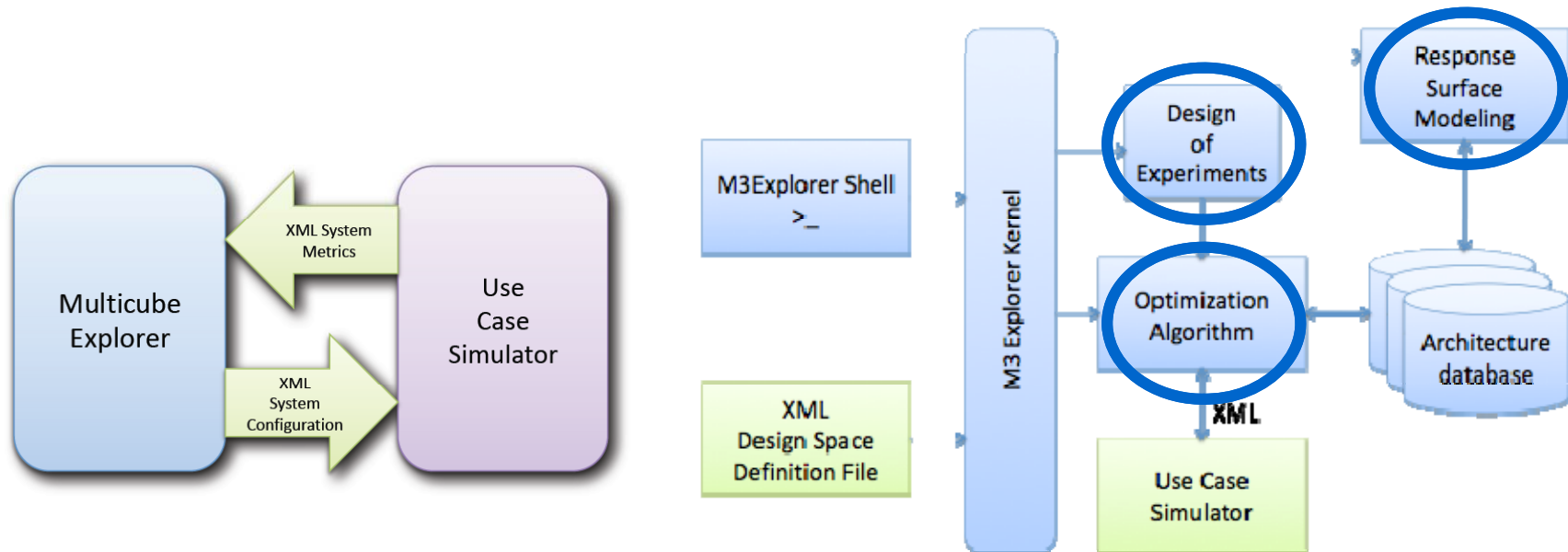


Automatic Design Space Exploration: Open Issues

- Efficiency of Automatic DSE process can be improved by:
 1. Minimizing the numbers of simulations to be executed by using **exploration heuristics** such as state-of-art evolutionary algorithms
 2. Speeding up simulations
 3. Simulating at higher abstraction levels
 4. Defining an **analytical response model** of the system behavior based on a subset of simulations to predict the unknown system response

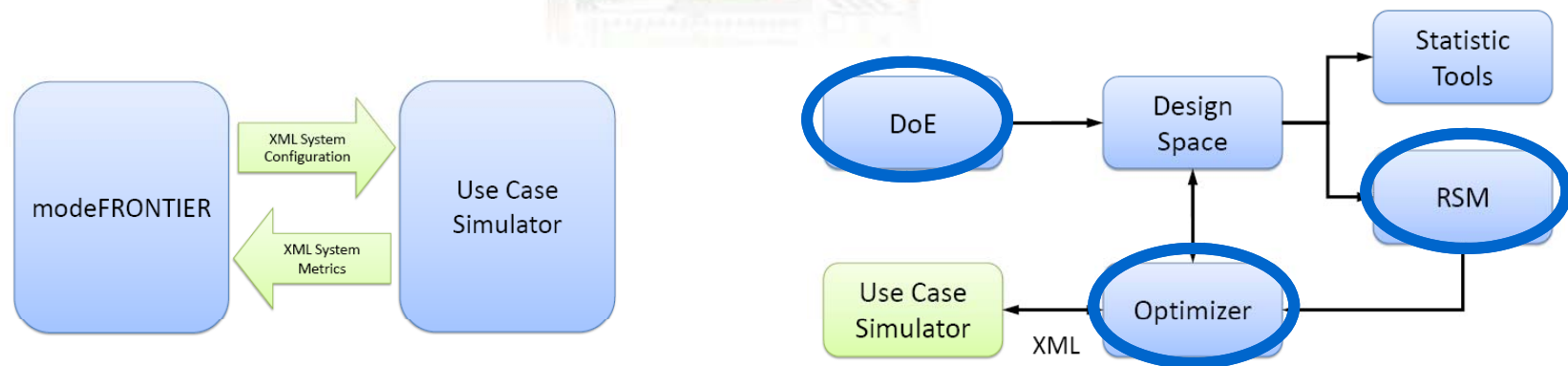
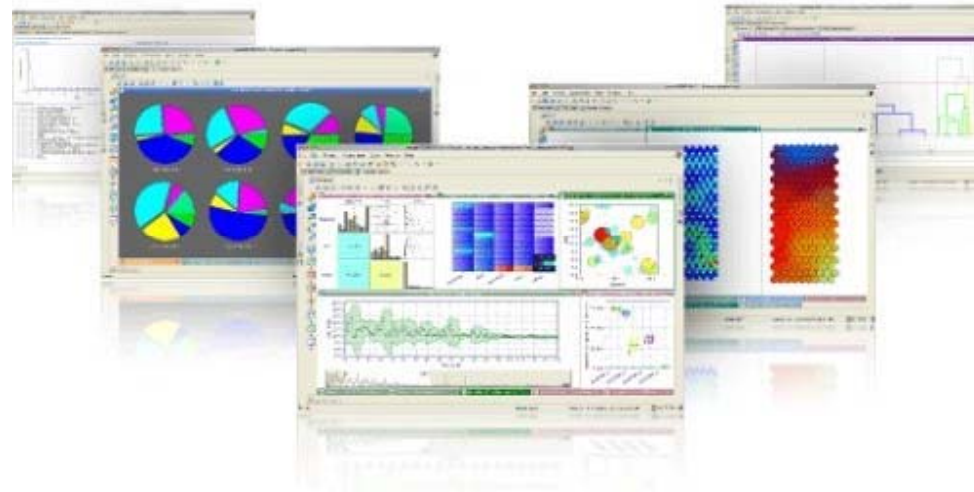
MULTICUBE Explorer

- **Open-source prototype exploration framework (MULTICUBE Explorer):**
The tool enables a fast automatic optimization of parameterized system architectures towards a set of multiple objective functions.



Retargeting of modeFRONTIER tool

- Retargeting of modeFRONTIER optimization tool from ESTECO to the discrete domain used in the embedded systems field

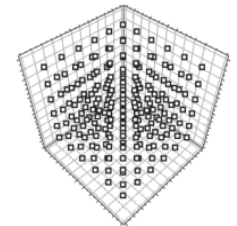


Design Space Exploration

- The Multi-Objective DSE frameworks are composed of:

1. Design of Experiments (DoEs):

To identify the experimentation plan: how to select the design points in the design space to be simulated



2. Optimisation Algorithms:

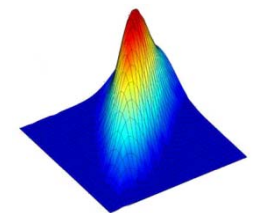
Metaheuristics methods inspired by analogies with physics, or with biology to solve multi-objective optimization problems.

This class of methods includes between the others: simulated annealing, genetic algorithms, evolutionary strategies.



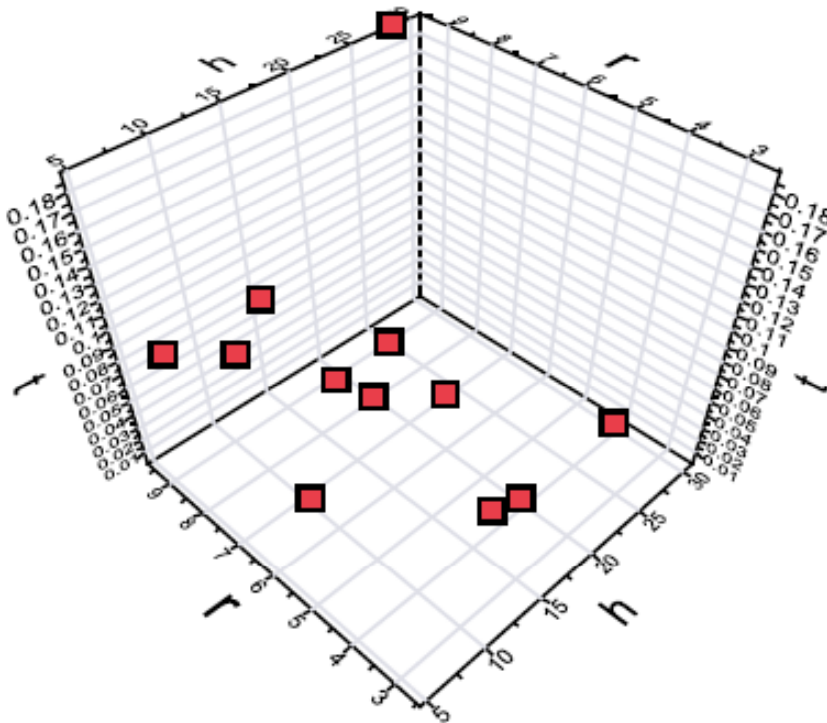
3. Response Surface Modeling (RSM):

To use the set of simulated points to obtain a response surface of the system behavior

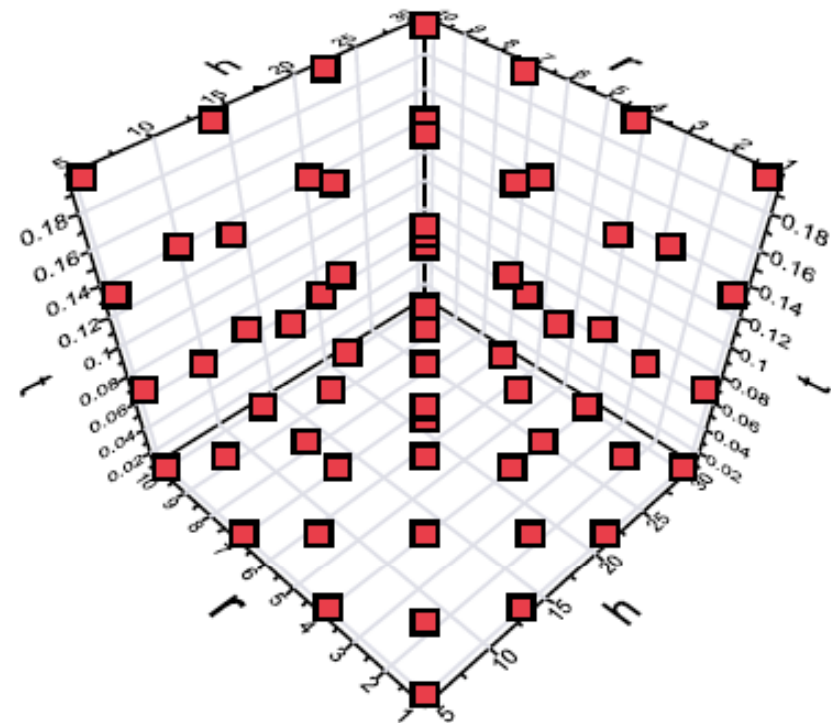


How to explore the design space?

- **Design of Experiments:** to identify the planning of experimentation campaign where the set of tunable design parameters can vary
- To specify the **layout:** how to select the design points in the design space



Random DOE, 12 Entries



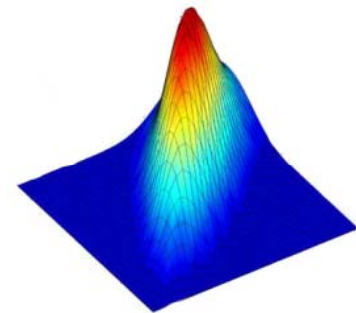
Full Factorial DOE, 64 Entries

Optimisation Algorithms

- Several algorithms can be selected for solving different problems:
 - NSGA-II: Non-dominated Sorting Genetic Algorithm II
 - MOGA-II: Multi-Objective Genetic Algorithm with elitism
 - MOSA: Multi-Objective Simulated Annealing
 - ES: Evolution Strategies
 - MOPSO: Multi-Objective Particle Swarm Optimizer
 - MFGA: Magnifying Front Genetic Algorithm
 - APRS: Adaptive windows Pareto Random Search
- All these metaheuristics are not mutually exclusive. It is often hard to predict with certainty the efficiency of a method when it is applied to a problem. This statement is confirmed by the well-known "*no-free-lunch theorem*"

Response Surface Modeling

- RSM techniques are used to define an analytical dependence between design parameters and one or more response variables.
- RSM based on two main phases:
 - During the **training phase**, known data (or training set) defined by Design of Experiments are used for tuning the RSM.
 - During the **prediction phase**, the RSM is used to predict the unknown system response.
- Several RSM techniques:
 - Linear Regression
 - Spline Interpolation
 - Shepard's Interpolation
 - Artificial Neural Networks (3-layer fully-connected feed-forward ANNs)
 - Radial Basis Functions
 - Kriging Interpolation (recently added)





Experimental Results

Target Architecture

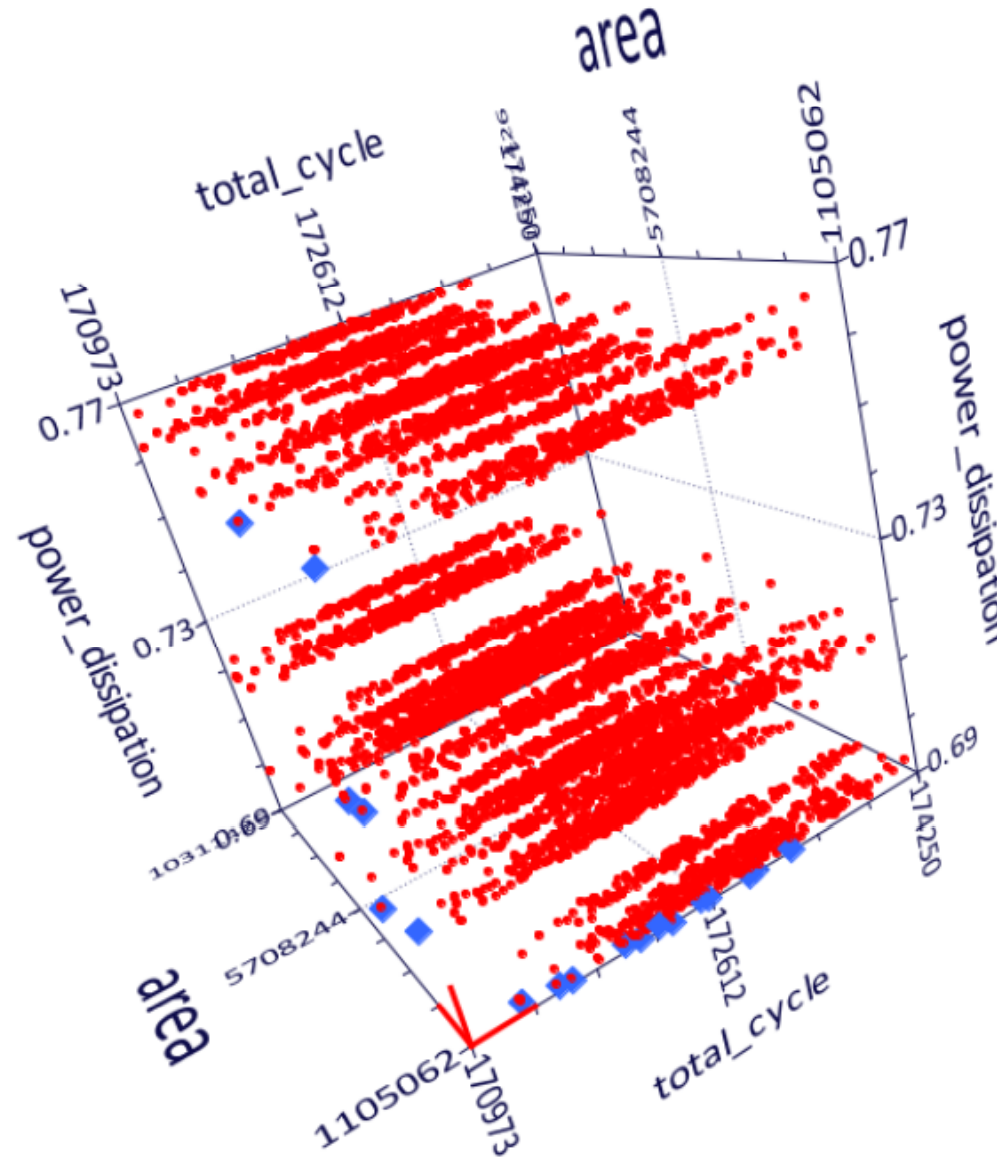
- SP2 Low-Power Processor use case provided by STMicroelectronics China
- **sp2sim** executable model of SP2 processor
- Target application: 164.gzip

category	parameter	description	values
out of order execution	rob_depth	reorder buffer depth	32, 48, 64, 80, 96, 112, 128
	mreg_cnt	rename register number	16, 32, 48, 64
	iw_width	instruction window width	4, 8, 16, 24, 32
cache system	icache_size	instruction cache size	16, 32, 64
	dcache_size	data cache size	16, 32, 64
	scache_size	secondary cache size	0, 256, 512, 1024
	lq_size	load queue size	16, 24, 32
	sq_size	store queue size	16, 24, 32
	mshr_size	miss holding register size	4, 8
branch prediction	bht_size	branch history table size	512, 1024, 2048, 4096
	btb_size	branch target buffer size	16, 32, 64, 128

- Design space composed of **1,161,216** design points reduced to **9,216** design points based on a statistical analysis (significance of each parameter) executed by picking up 5000 points randomly

Optimisation Algorithms: Comparison Results

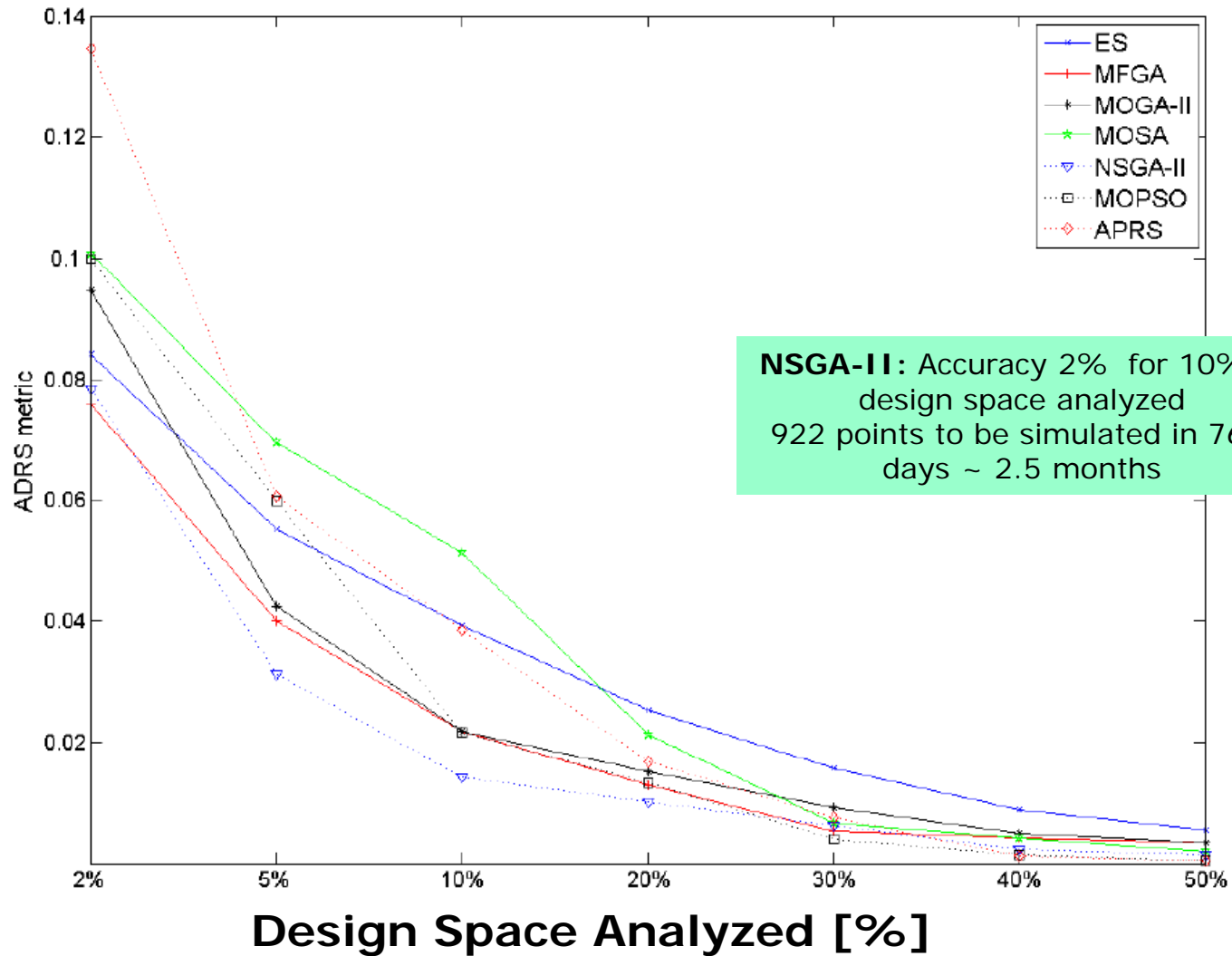
- Target **total_area**
- The execution time is **9,216**
 - The execution time is 2 h
- Accuracy measurement approach – **Lo**



ion of
 'area'
 oloration of
 ration takes
 illel machines
ence Set to
 id

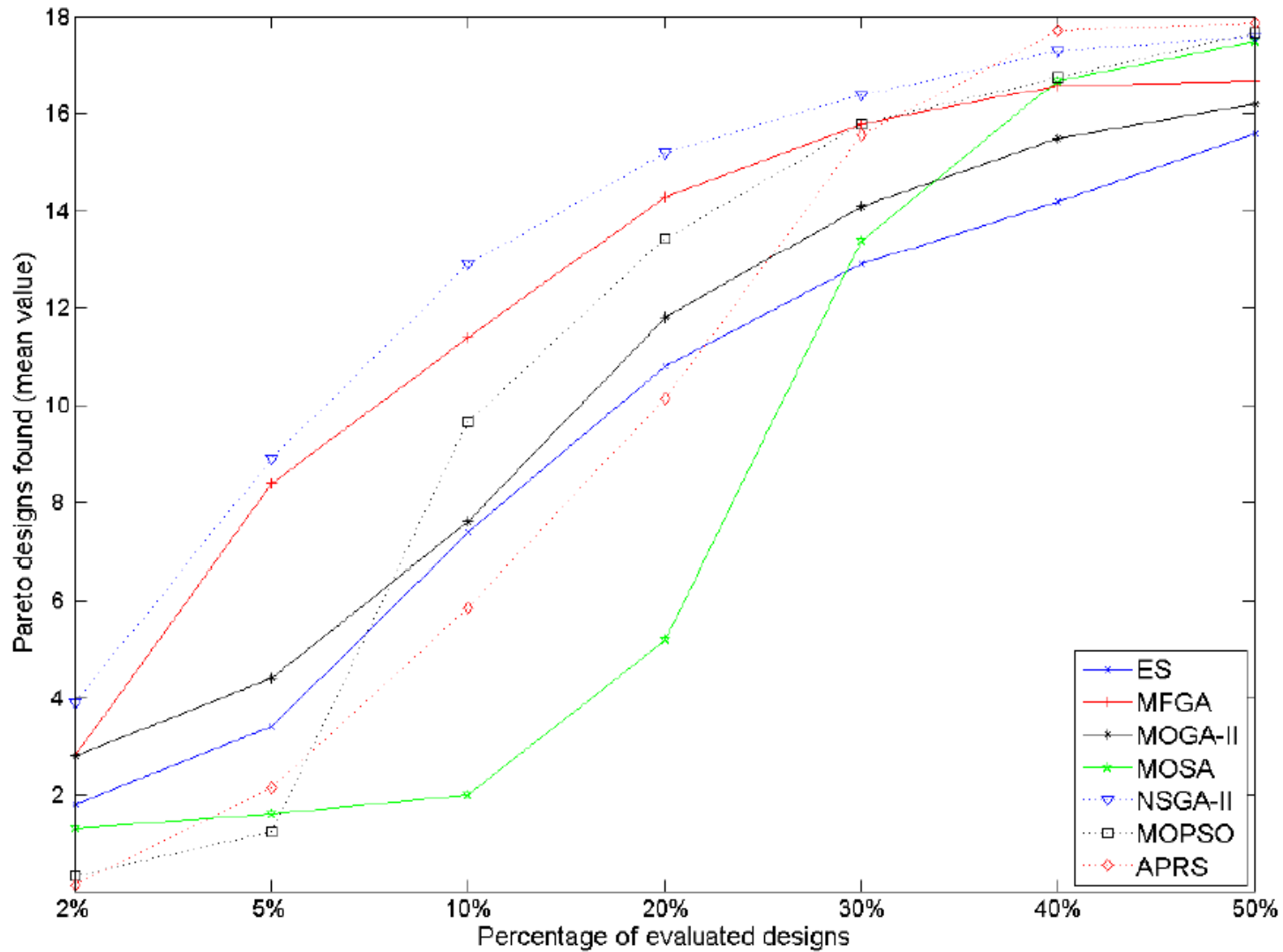
Accuracy vs. Design Space Analyzed [%]

ADRS



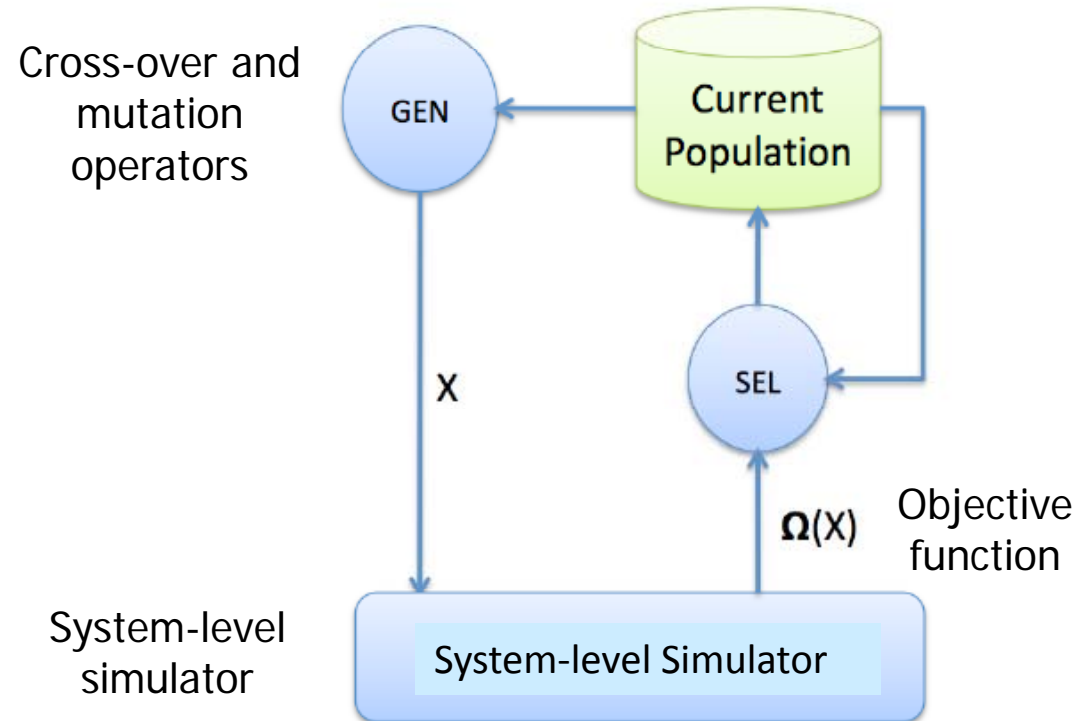
NSGA-II: Accuracy 2% for 10% of design space analyzed
 922 points to be simulated in 76.8 days ~ 2.5 months

Convergency vs. Design Space Analyzed [%]



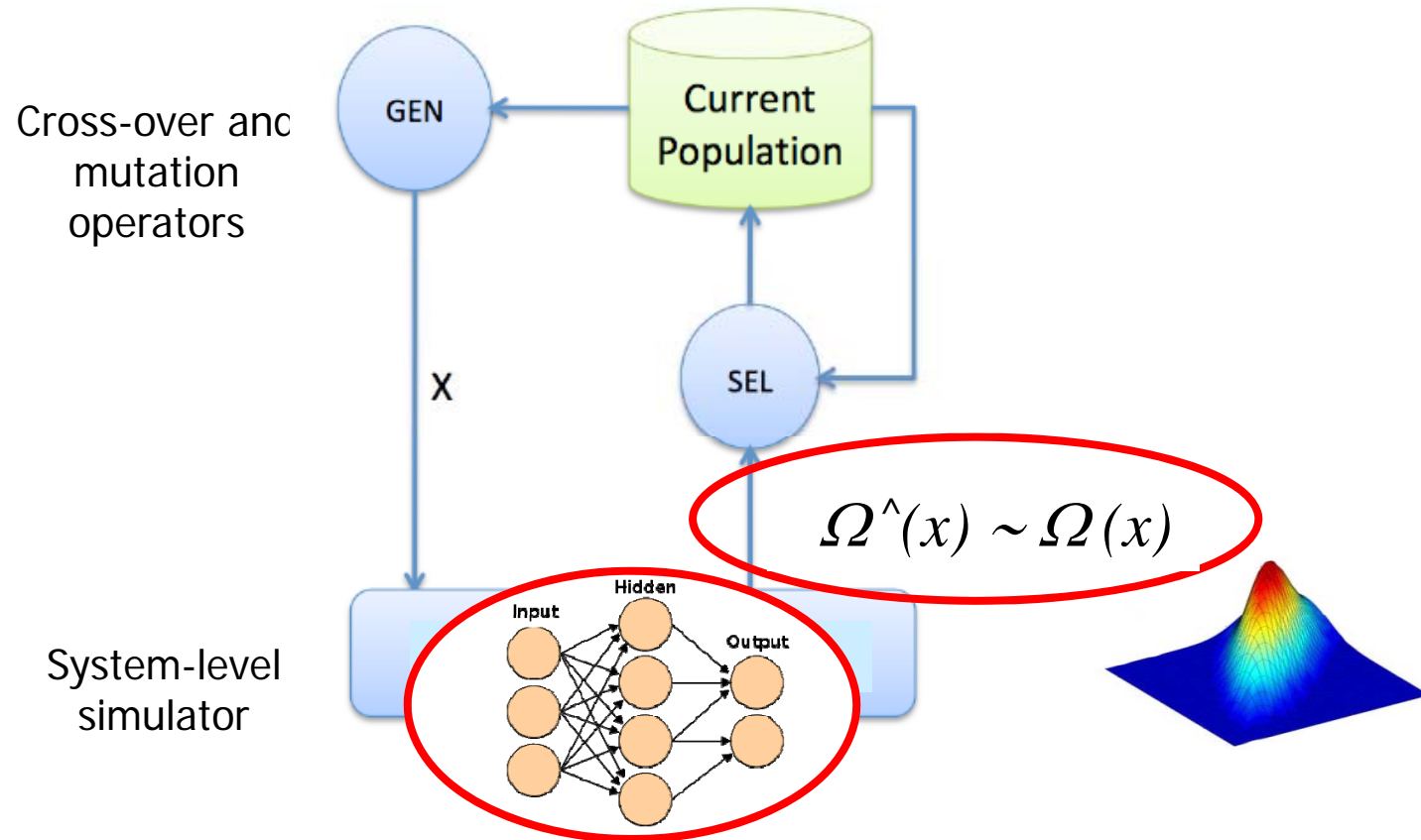
**How to combine optimisation
heuristics and response surface
models to further reduce the
exploration time?**

NSGA-II Exploration Flow



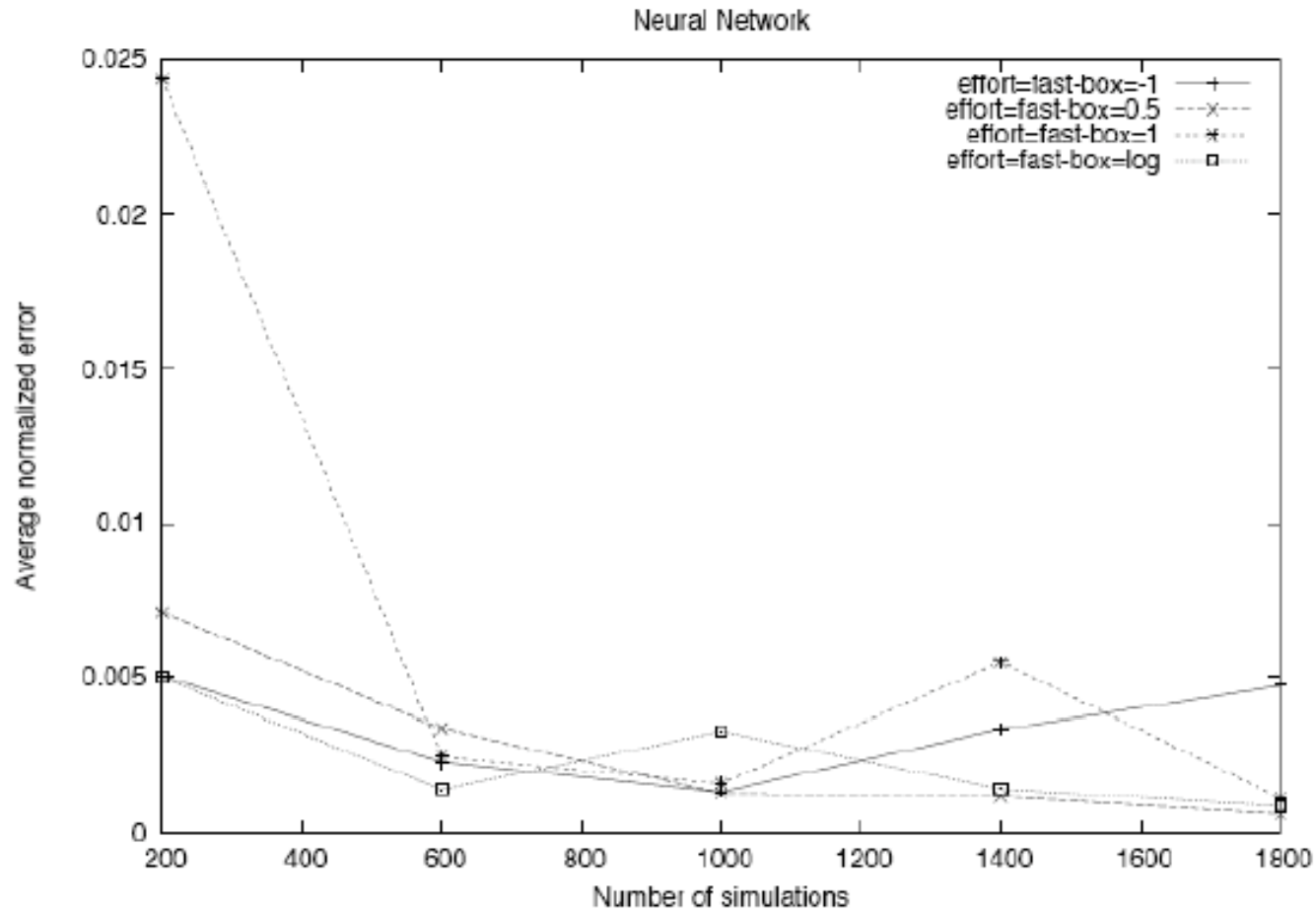
Main problem: Very long simulation time required to evaluate the system-level objective function $\Omega(x)$

NSGA-II Exploration Flow combined to ANN model



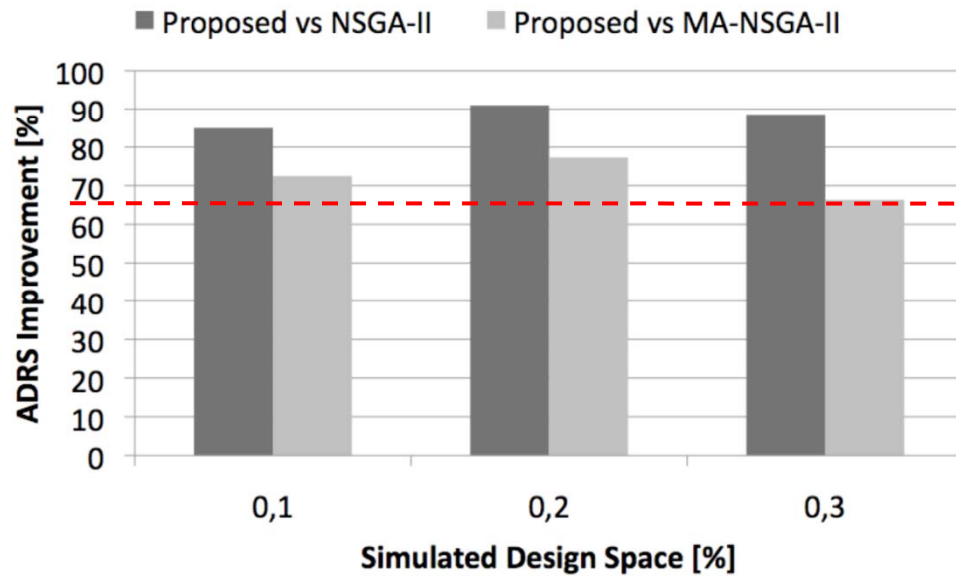
Proposed solution: NSGA-II combined to an Artificial Neural Network to predict the system-level objective function $\Omega(x) = [\varepsilon(x), \delta(x)]$

Average normalized error for Artificial Neural Network technique used as RSM

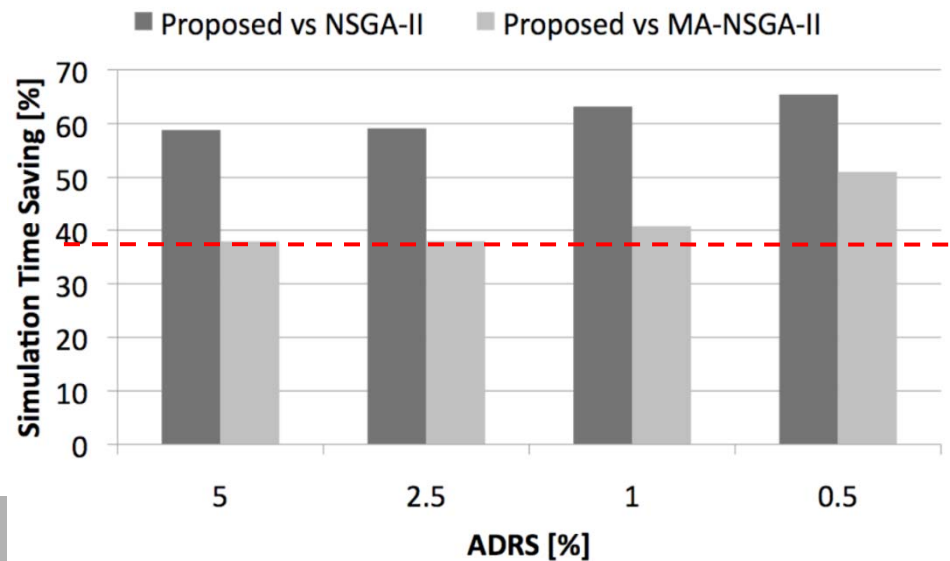




Comparison Results vs. NSGA-II [DAC2010]



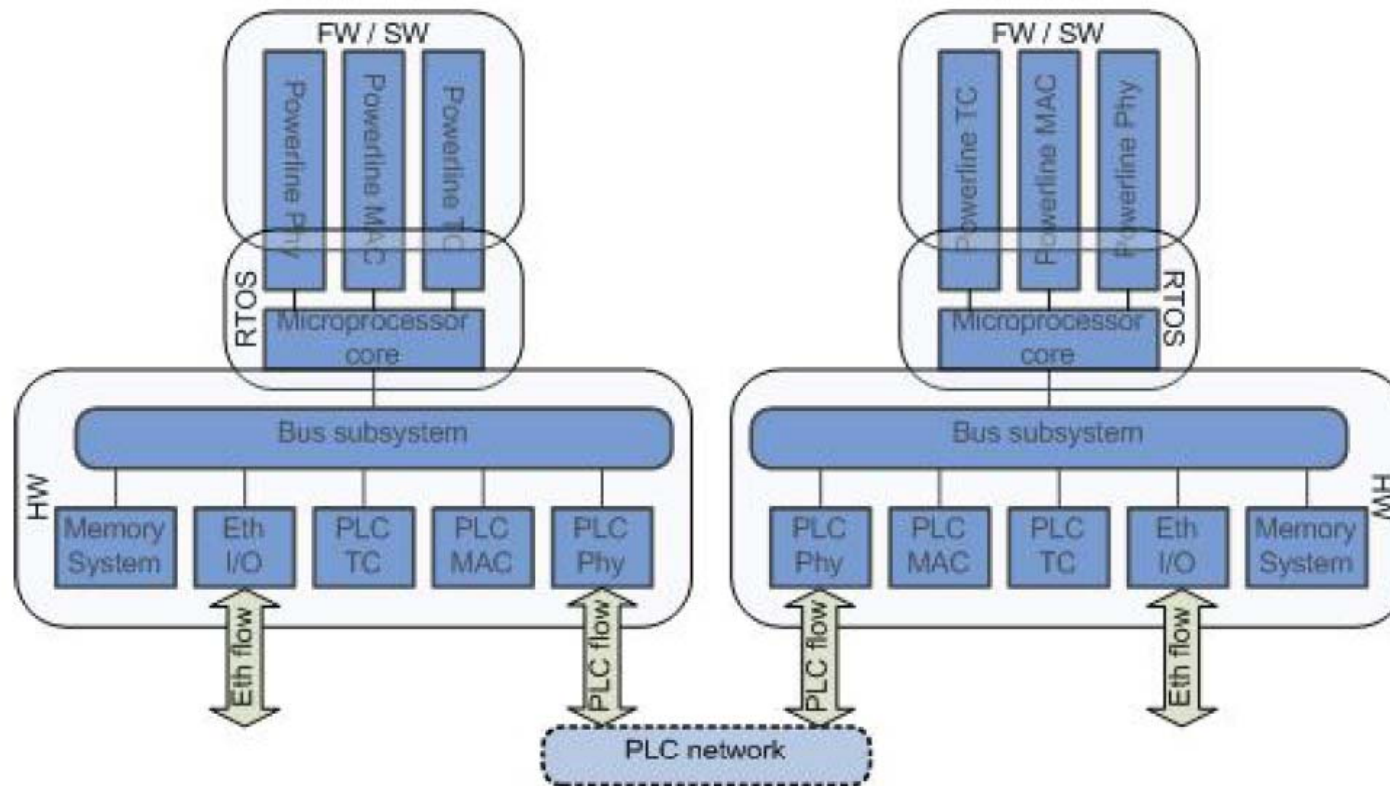
Quality of the solution set improved by 65%



Exploration time reduced by 37%

Use Case: Powerline Communication (DS2)

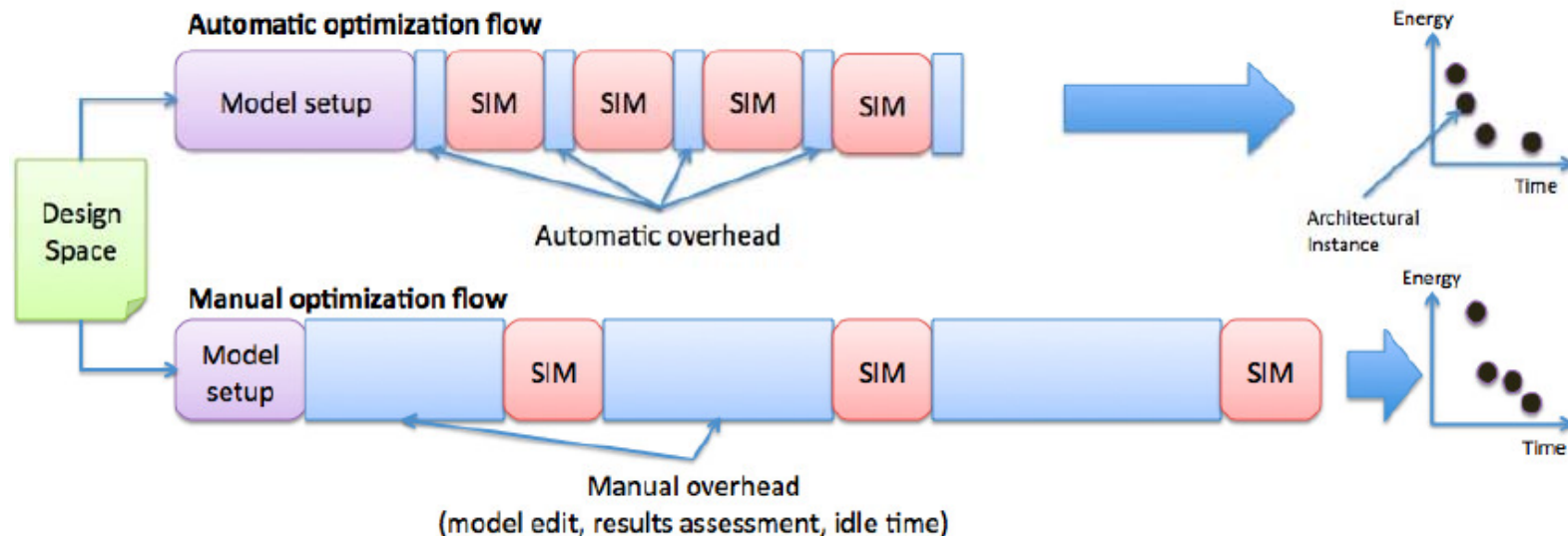
- **Powerline communication:** an advanced telecommunication gateway to transfer audio, video and data streaming through the domestic power line.



Industrial assessment done by DS2 on Powerline

According to DS2, main advantages of automatic DSE with respect to manual DSE:

- *“Faster convergence towards the Pareto points”*
 - Save up to 80% of designer time
- *“Higher number of design points”*
 - Up to 10 times more possible combinations
- *“Less complexity of the design flow”*



Conclusions

- An automatic design space exploration methodology has been proposed leveraging Design of Experiments and Response Surface Modeling techniques
- The proposed framework makes automatic exploration of multi-core architectures more feasible
- Future work: Run-time management of system resources and dynamic compilation support
- This work is part of the ICT-FP7 EU project MULTICUBE

www.multicube.eu

