



Some Key Issues in Embedded System Design

Maurizio Palesi

Outline

- Design Space Exploration of Parameterized Systems
- Network on Chip Architectures
- Instruction Level Power Estimation

Outline

- **Design Space Exploration of Parameterized Systems**
- Network on Chip Architectures
- Instruction Level Power Estimation

Motivation

■ Design trends

- Growing demand for portable devices
- Growing demand for low power design
- Increased application complexity
- Shrinking time-to-market windows

■ Technology trends

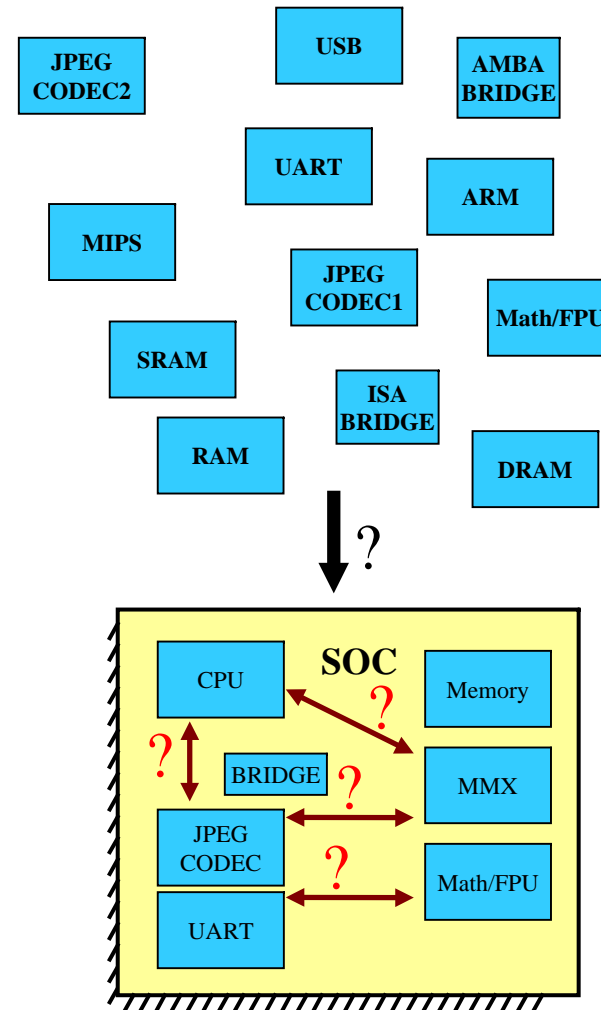
- Increased chip capacity
- Increased I/O pins
- Improved on-chip integration techniques (storage, digital, analog, digital, ...)
- SOC era

Need for greater design productivity!

Motivation

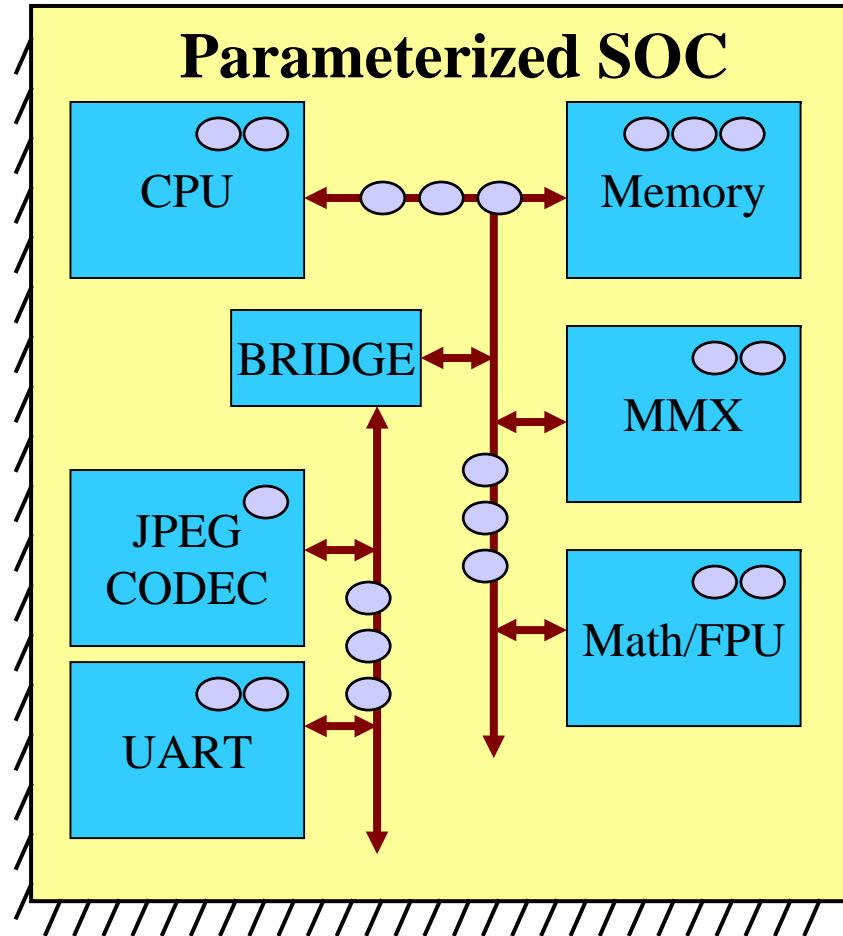
■ One approach: reuse of existing IP

- IP selection?
- IP integration?
- SOC verification?
- Multi-source IP licensing
- ...



Motivation

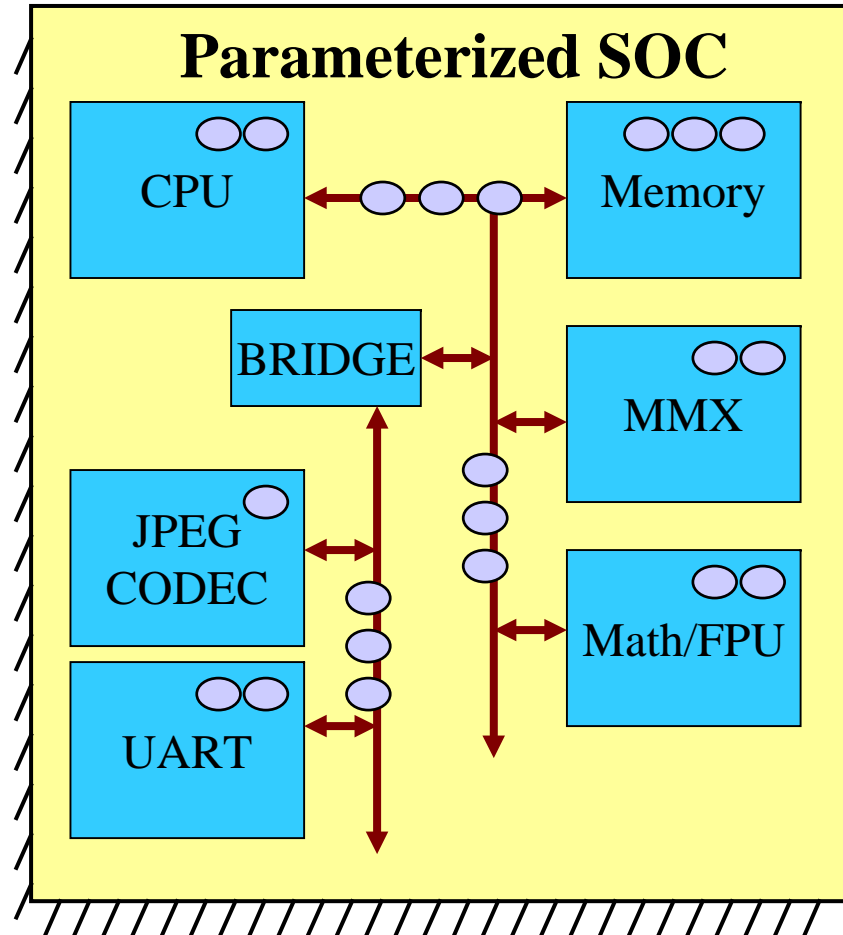
- Alternate approach: reuse of SOC
 - Designed, integrated, tested
 - Domain specific
 - Parameterized
- Designed by firms specializing in SOC
- User: map application, then, “configure-and-execute”



Motivation

- Composed of 100s of cores
- Cores are “configurable”
- Configurations impact power/performance
- Large number of total configurations!

Architecture is otherwise fixed!



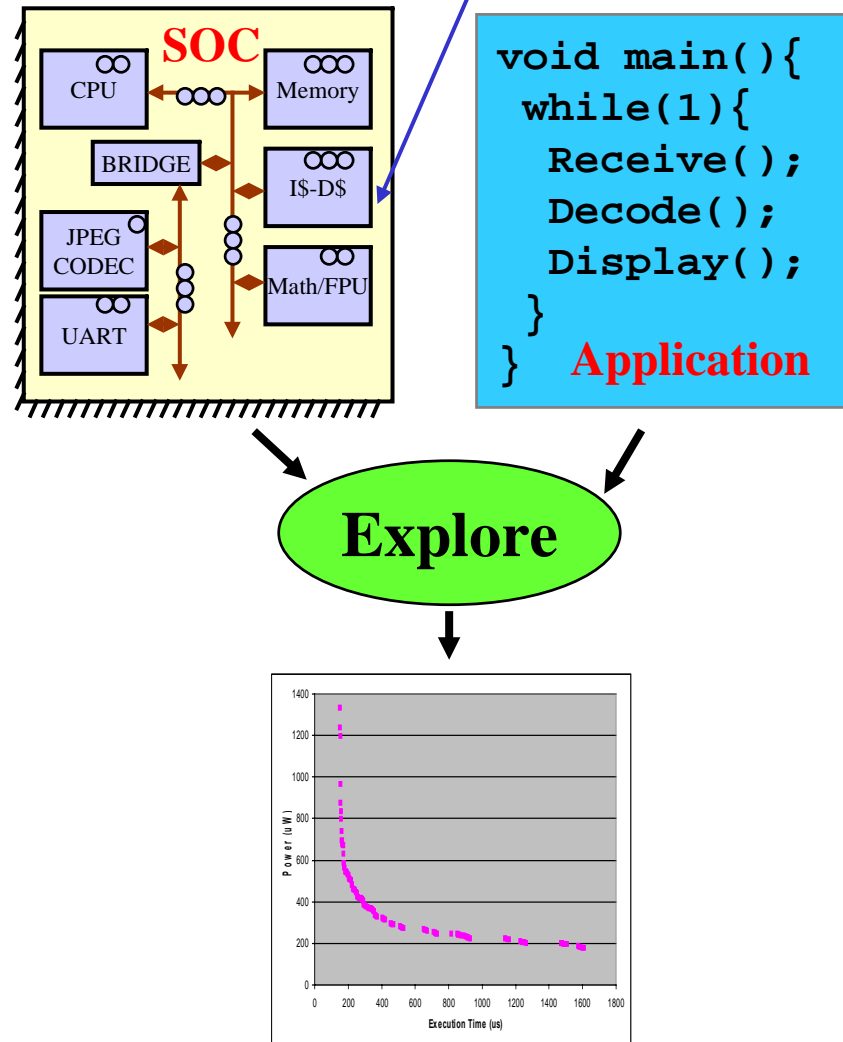
Motivation

- ATI Technologies – **XILLEON™ 220** SOC for Digital Set-top Box Market
- Tensilica – **Xtensa™ 1040** configurable processor cores
- Philips Semiconductors – **Nexperia™** SOC platforms
- Adelante Technologies – offers complete SOC customizable platforms for DSP domains
- More...

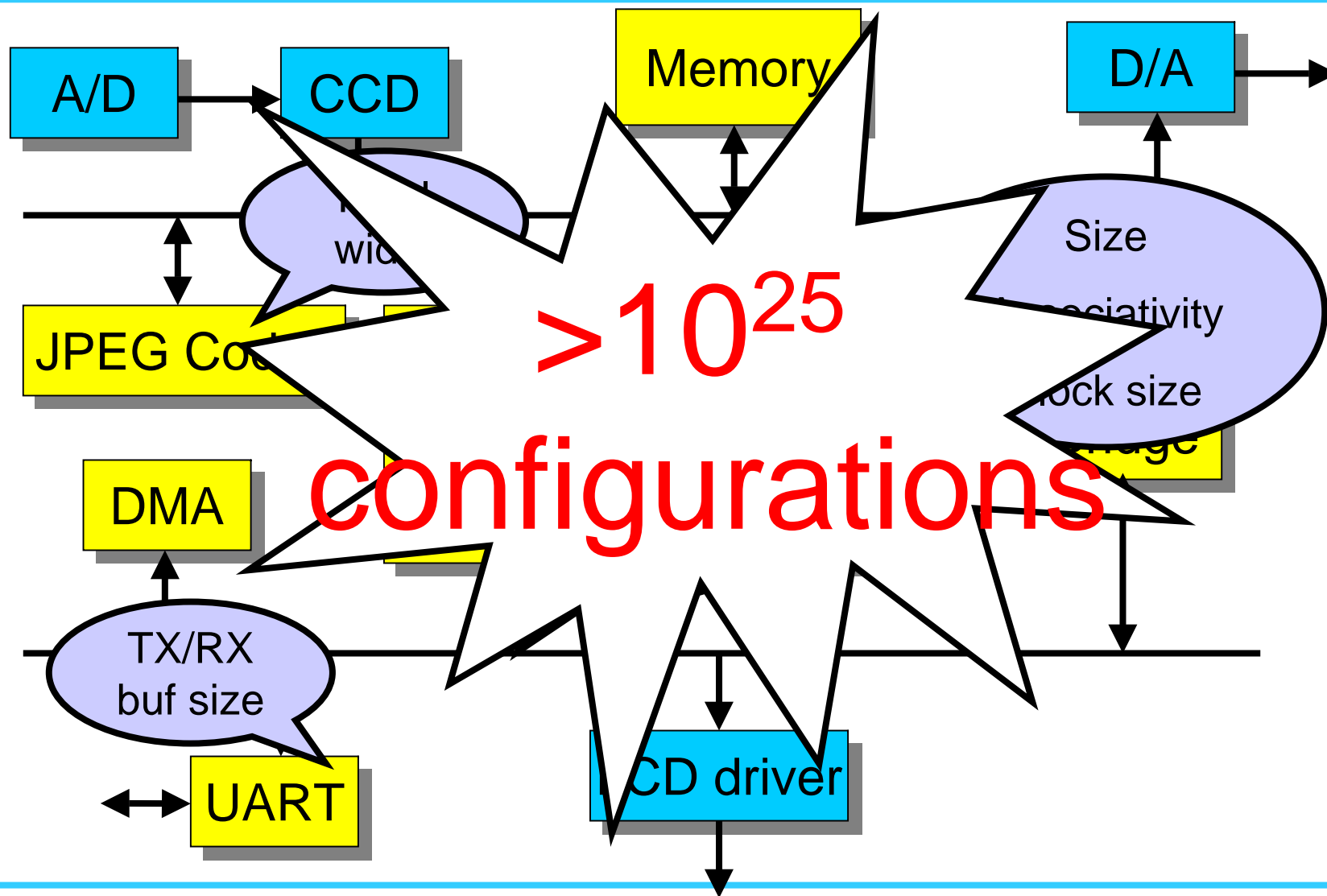
Overview

- Given:
 - ➔ Parameterized SOC architecture
 - ➔ Fixed application
- Automatically explore the design space
- Find optimal points w/respect to power and performance

- ❖ Size = {1K, 4K, 8K}
- ❖ Line = {4, 8, 16}
- ❖ Assoc = {1, 2, 4}



Sample SOC Platform for Digital Camera

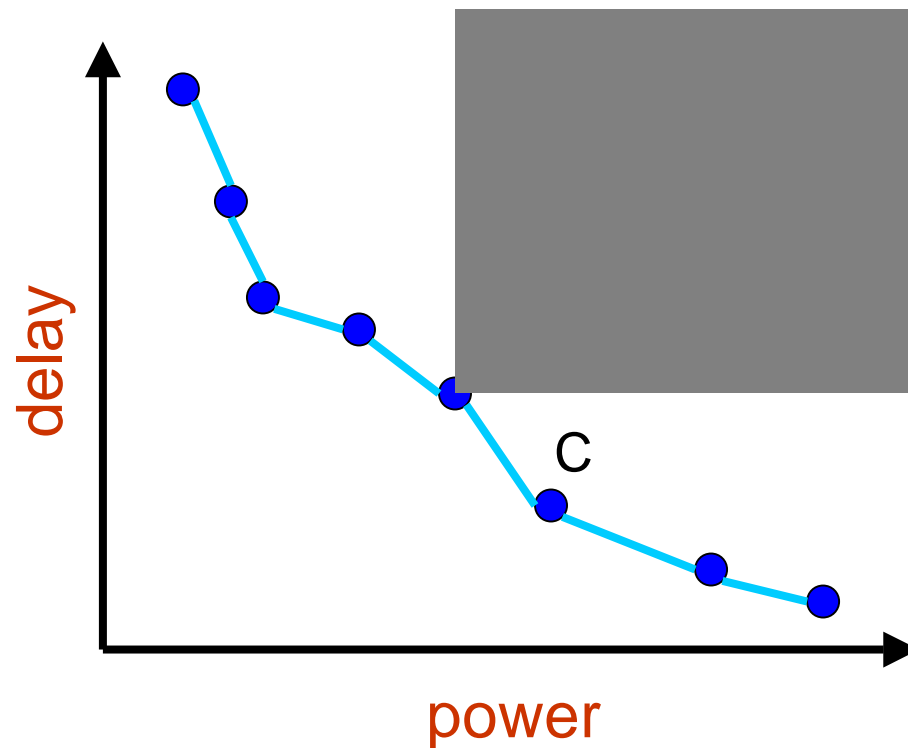


Design Space Exploration (DSE)

- Defining strategies for tuning the parameters so as to obtain the Pareto-optimal set of configurations that provide multi-criteria optimisation
- Criteria (a.k.a. objectives)
 - Power dissipation
 - Performance (delay, execution time, ...)
 - Area (cost, complexity)
 - Energy
 - ...

Pareto's Concept

- A new notion of optimality is required in the presence of objective conflicts



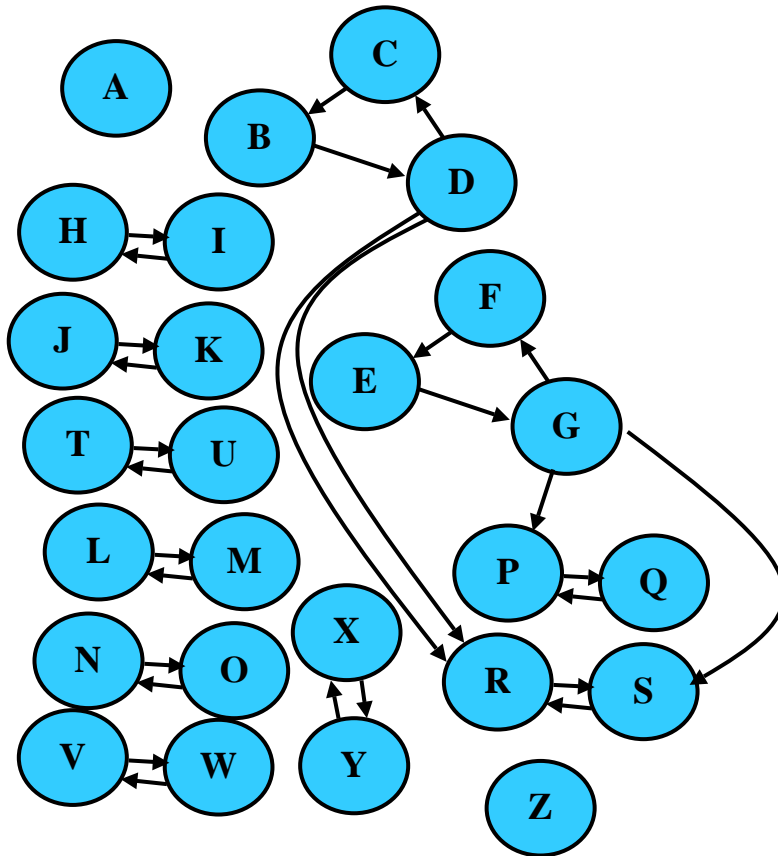
DSE Approaches

- Dependency analysis (dep) [Givargis *et al.* TVLSI02]
- GA-based DSE (ga) [Palesi *et al.* TCAD05]
- Sensitivity Analysis [Zaccaria *et al.* DAES02]
 - Pareto-based Sensitivity Analysis (pbsa) [Palesi *et al.* IWSOC02]
- Dependency + GA (depga) [Palesi, Givargis CODES02]
- Sensitivity + GA (saga) [Palesi *et al.* ISCS02]

Parameter Dependency

- $A \rightarrow B$: Pareto-optimal configurations of B should be calculated after Pareto-optimal configurations of nodes along the path $A \rightarrow B$ are calculated
- $A \rightarrow B \rightarrow A$, (cycle): Pareto-optimal configurations of all the parameters on the cycle need to be calculated simultaneously
- A : Pareto-optimal configurations calculated in isolation

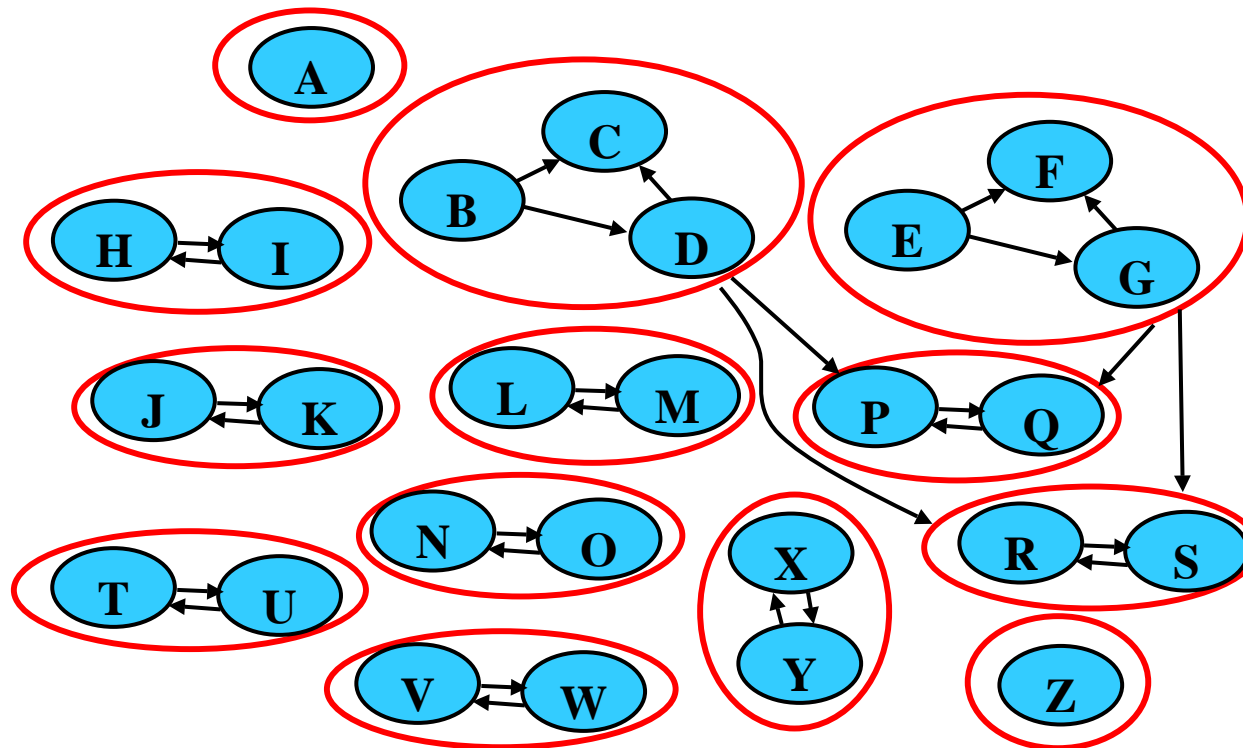
Dependency Graph



| Node | Core | Parameter | Node | Core | Parameter |
|------|-------------------|-------------------|------|---------------------|---------------------|
| A | MIP S | Voltage scale | L | CPU D\$ bus | Data bus width |
| B | I\$ | Total size | M | | Data bus code |
| C | | Line size | N | | Addr bus width |
| D | | Associativity | O | | Addr bus code |
| E | D\$ | Total size | P | I/D\$ Mem bus | Data bus width |
| F | | Line size | Q | | Data bus code |
| G | | Associativity | R | | Addr bus width |
| H | CPU I\$ bus | Data bus width | S | | Addr bus code |
| I | | Data bus code | T | | Periphe ral bus |
| J | | Addr bus width | U | Data bus code | |
| K | | Addr bus code | V | Addr bus width | |
| X | UAR T | Tx buffer size | W | Addr bus code | |
| Y | | Rx buffer size | Z | DCT CODE C | Pixel resolution |

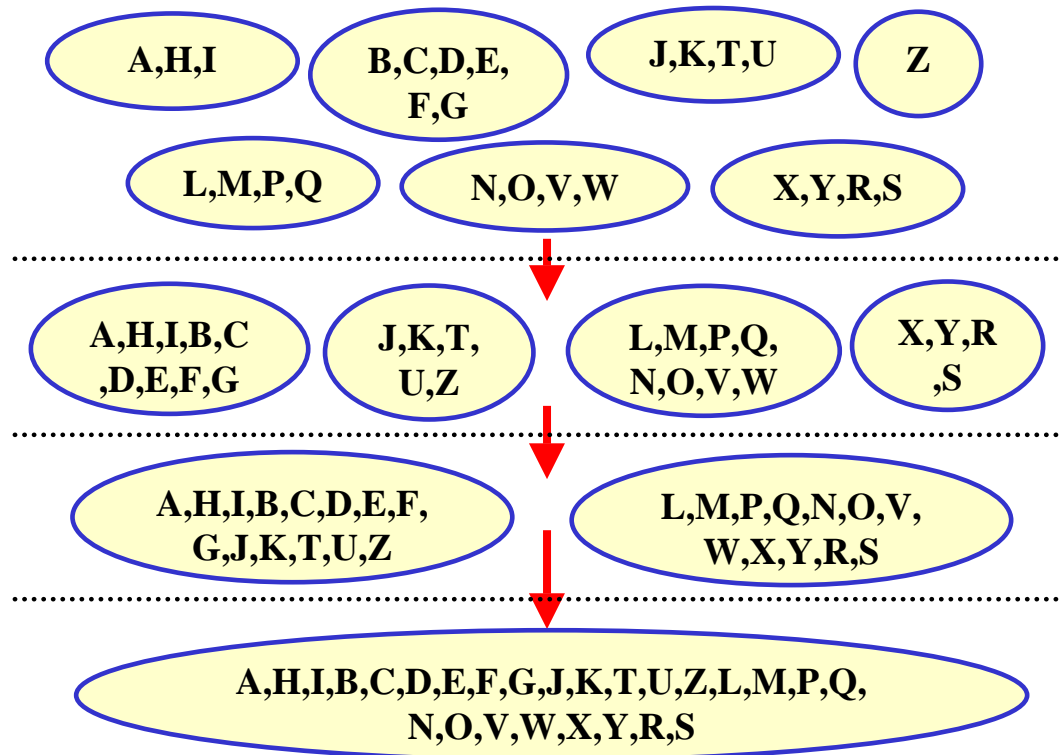
dep: Exploration Algorithm

Step 1: Clustering followed by simulation



dep: Exploration Algorithm

Step 2: Pair-wise merge followed by simulation



dep: Problems

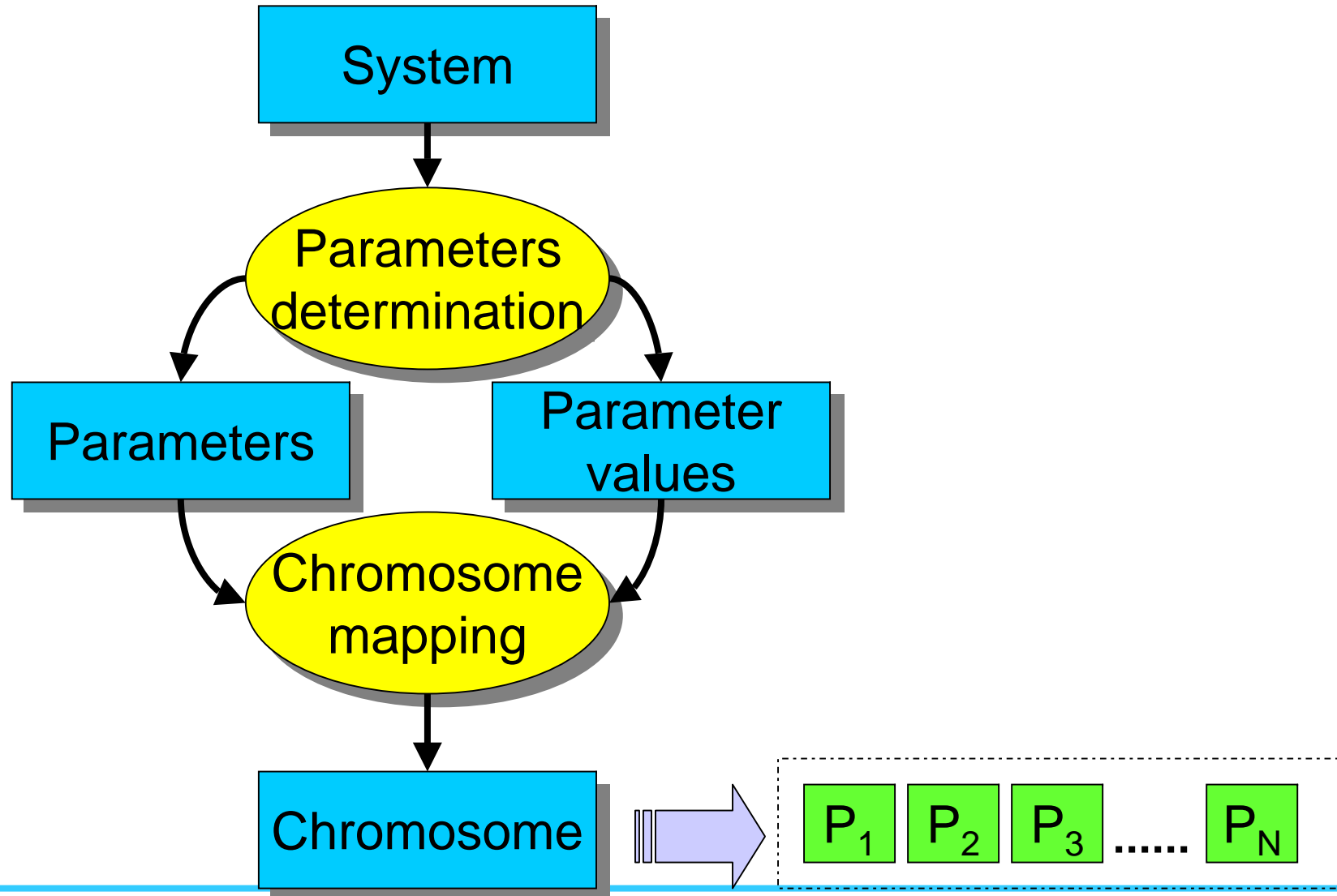
- Dependency graph
 - Based on designer knowledge
- If the parameters are heavily interdependent
 - Big cluster
 - Local configuration space too large
 - The approach becomes infeasible

GA: Application

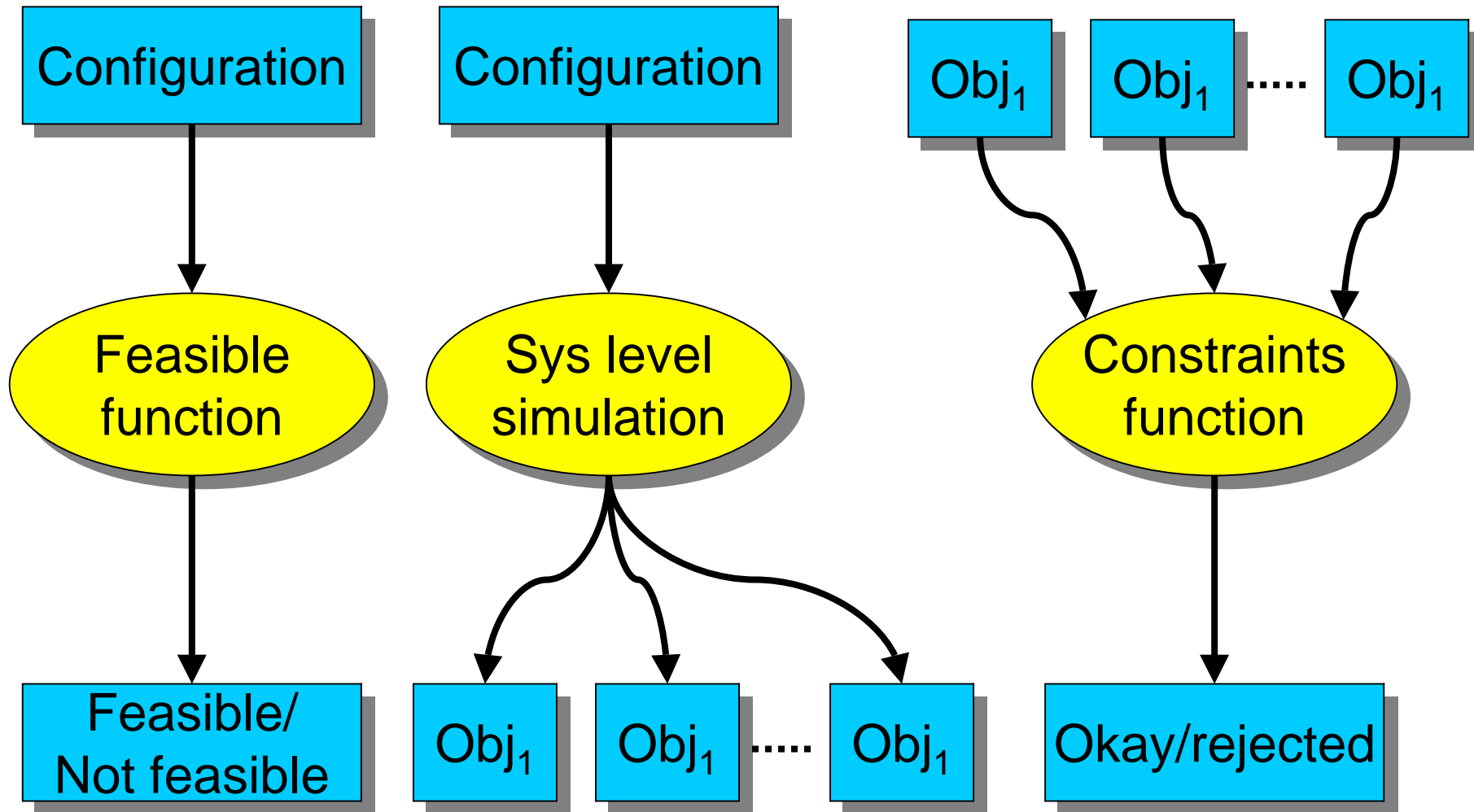
■ 5 items

- Configuration representation
- Feasible function
- Cost/Objective functions
- Constraint functions
- Convergency criteria

Configuration Representation

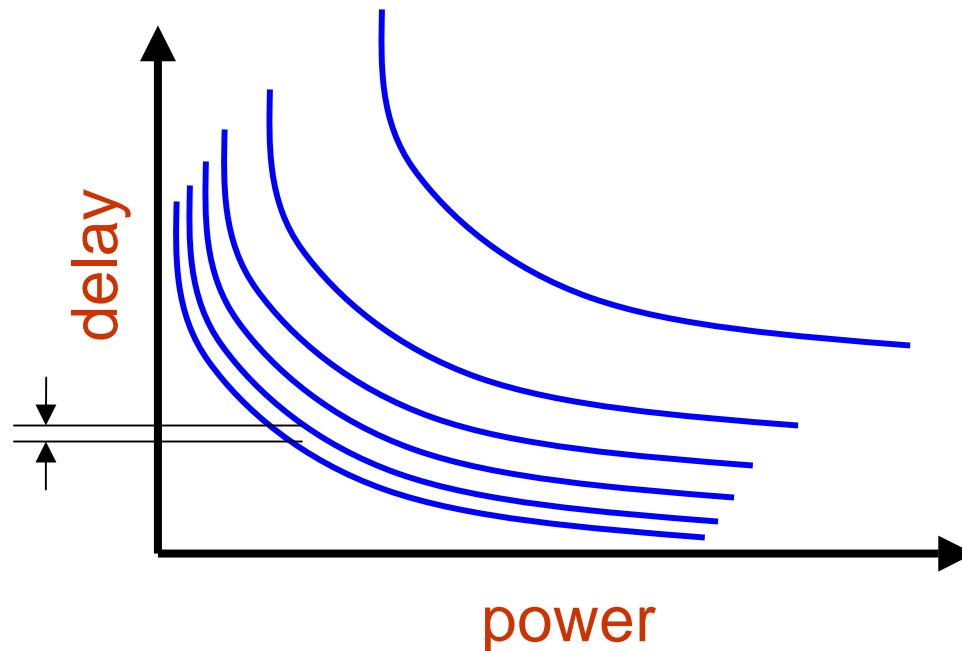


Feasible/Cost/Constraints Functions



How Many Generations?

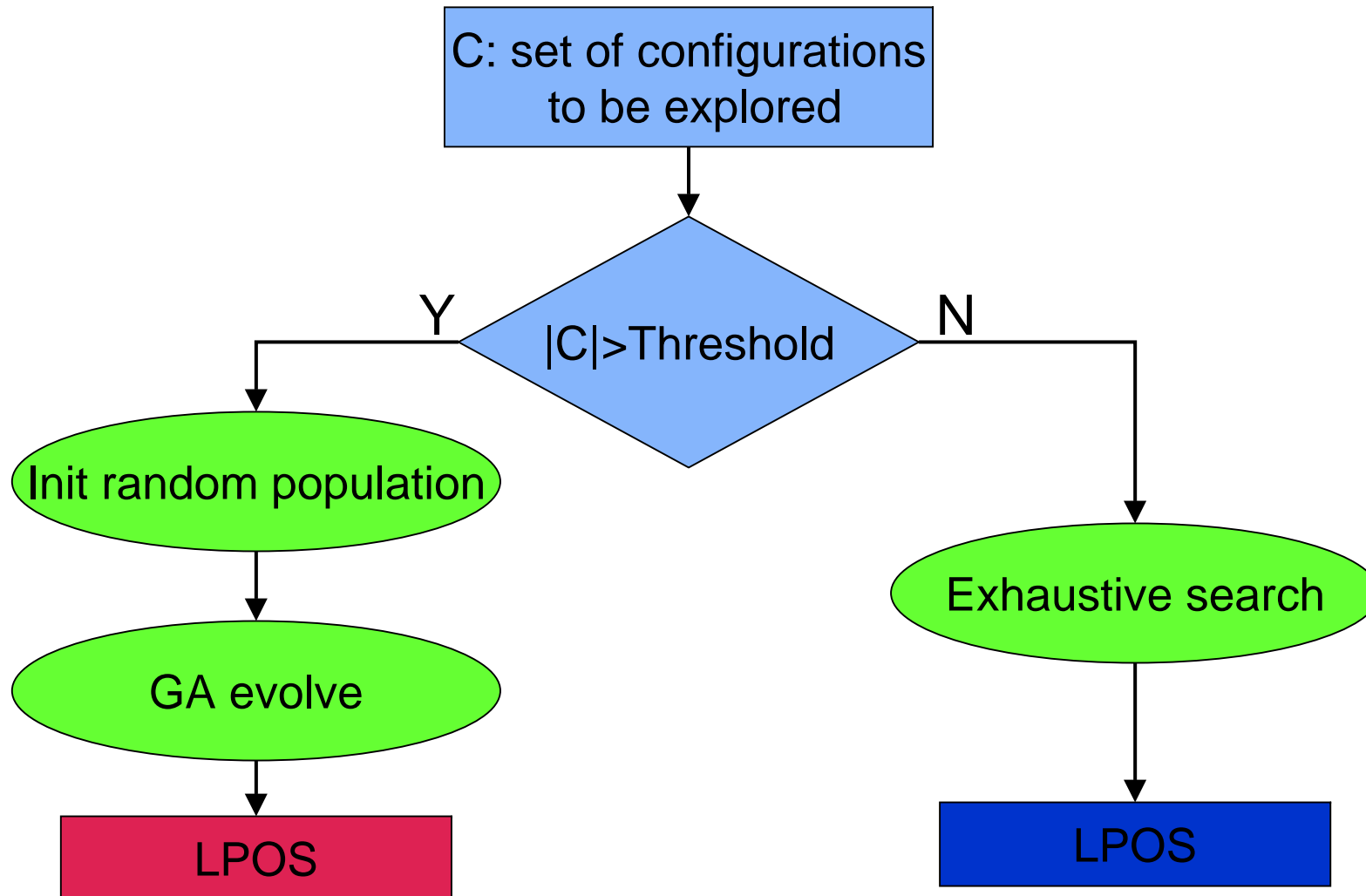
- Fixed number of generations
- Autostop criteria
 - Based on convergency



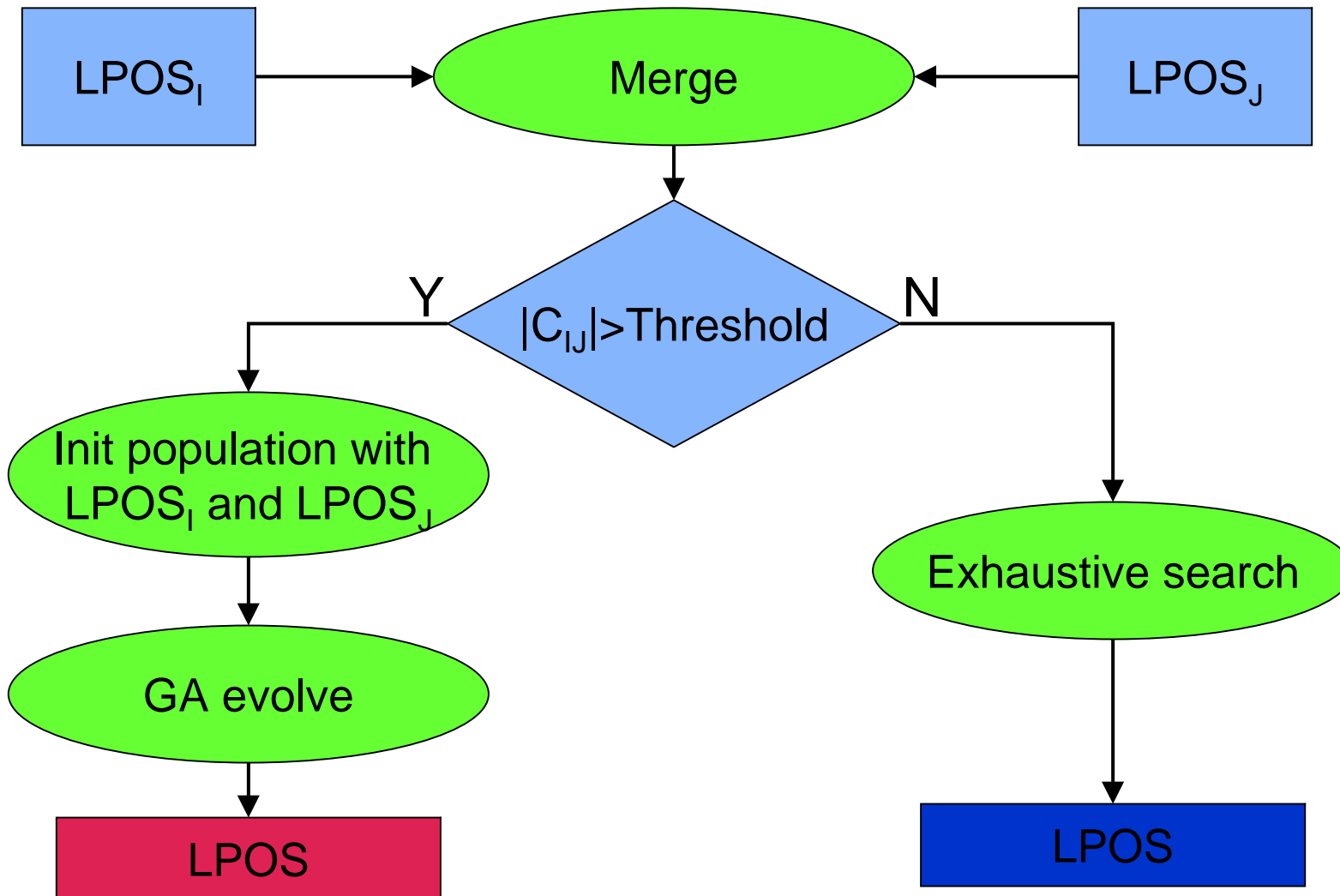
dep: Problems & Solutions

- If the parameters are heavily interdependent
 - Big cluster
 - Local configuration space too large
 - The approach becomes infeasible
- Solution
 - Substitute a GA based approach in place of the exhaustive search

depga: 1st phase

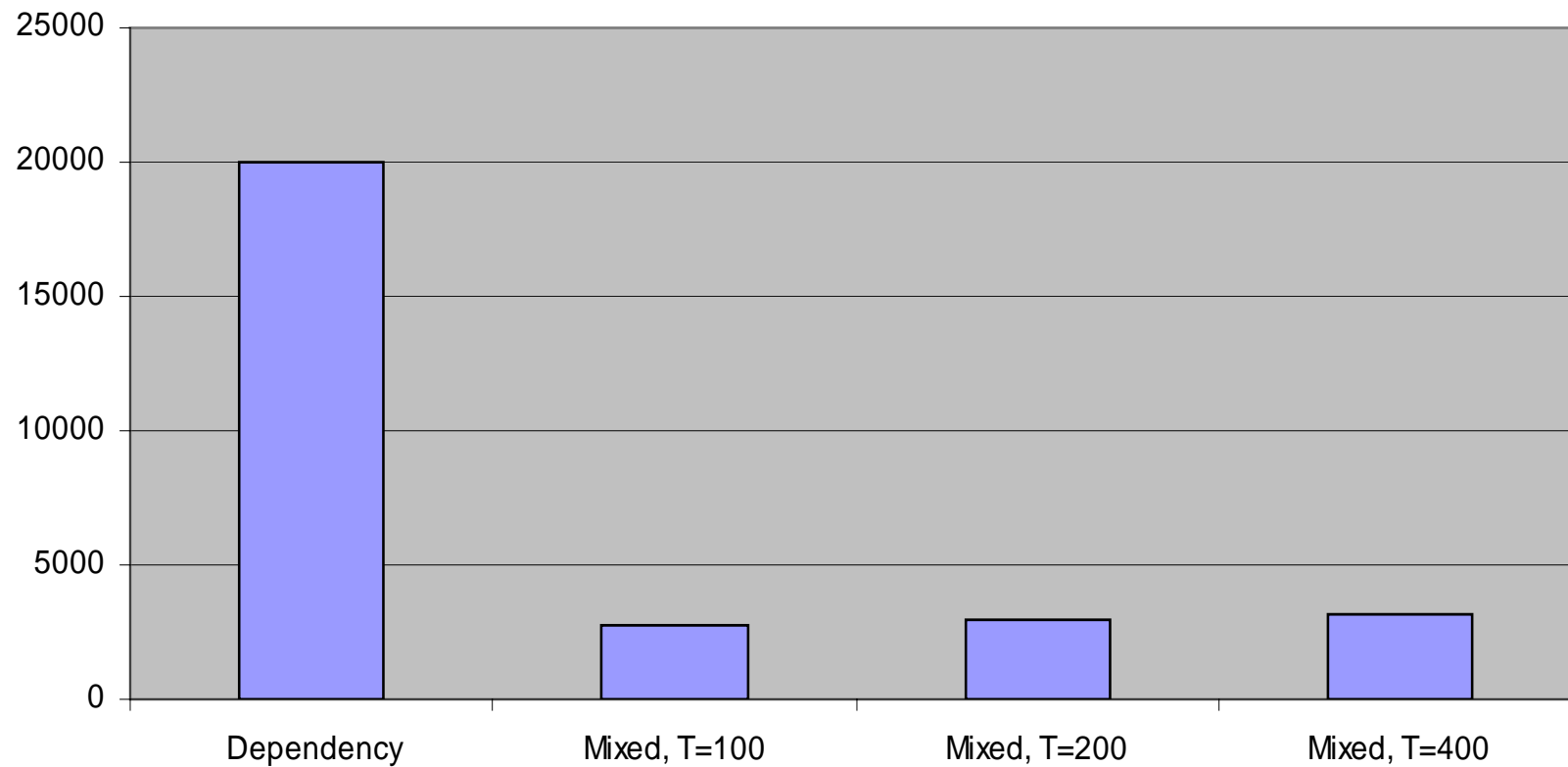


depga: 2nd phase

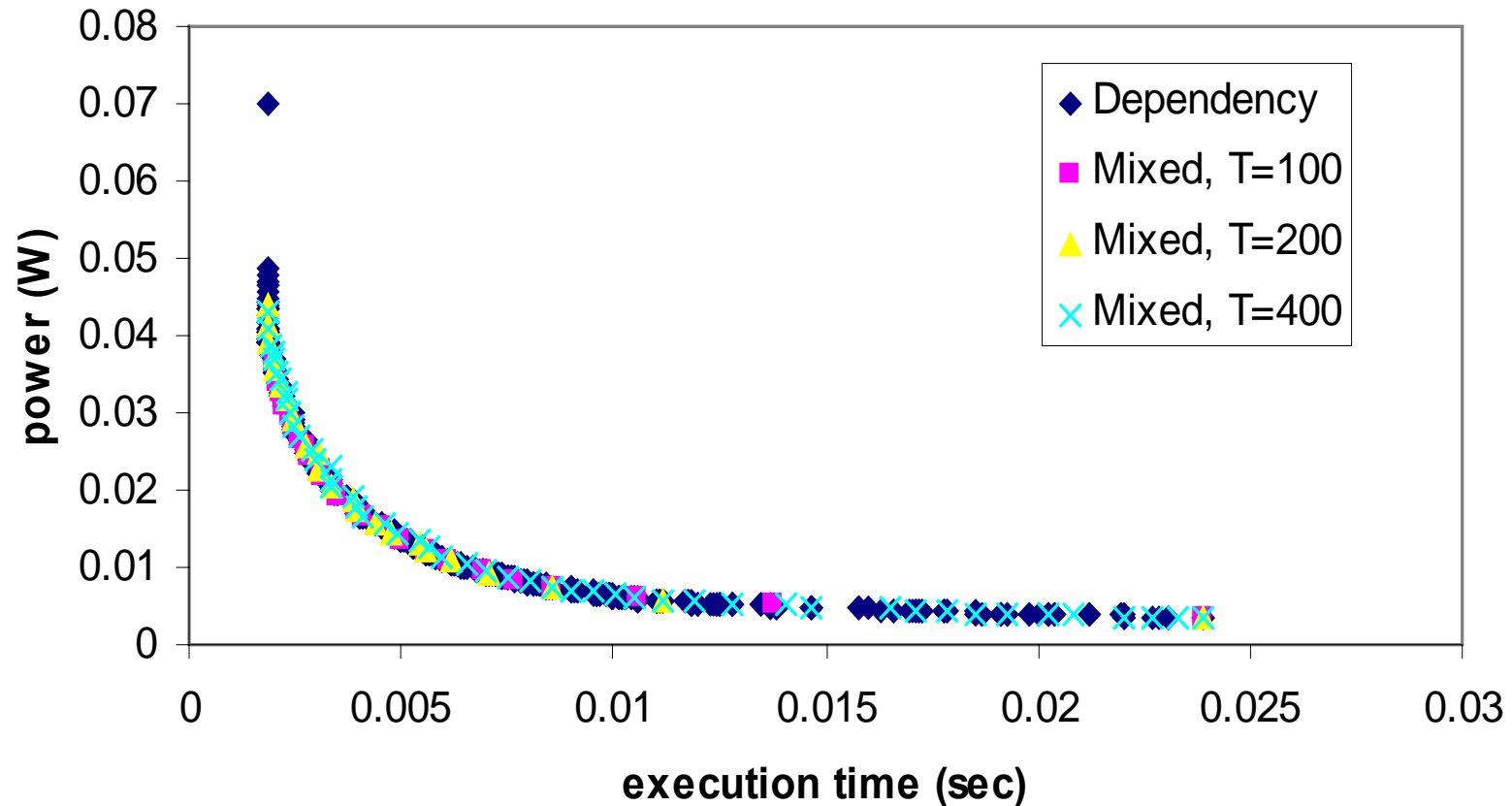


Efficiency

Number of Configuration Visited



depga: Experiments



Sensitivity Analysis

- Let us consider N parameters P_1, P_2, \dots, P_N each of them can assume respectively $\{V_1\}, \{V_2\}, \dots, \{V_N\}$ values
 - The number of possible configurations will be $|V_1| \times |V_2| \times \dots \times |V_N|$
 - Sensitivity analysis allows to test only $|V_1| + |V_2| + \dots + |V_N|$ configurations
- Two phases
 - Parameters sorted according to their sensitivity
 - Exhaustive search in the individuated subspace

Sensitivity Analysis (example)

■ Minimization power-delay (PD) of a cache

→ A configuration is a triple $c = \langle s, b, a \rangle$ (size, bsize, assoc)

→ Sensitivity analysis

✓ Fix b and a and let s variable $\Rightarrow Pd_{min}^s, Pd_{max}^s$

✓ Fix s and a and let b variable $\Rightarrow Pd_{min}^b, Pd_{max}^b$

✓ Fix s and b and let a variable $\Rightarrow Pd_{min}^a, Pd_{max}^a$

✓ Sorting starting from the most sensitive to the less one (e.g. s, a, b)

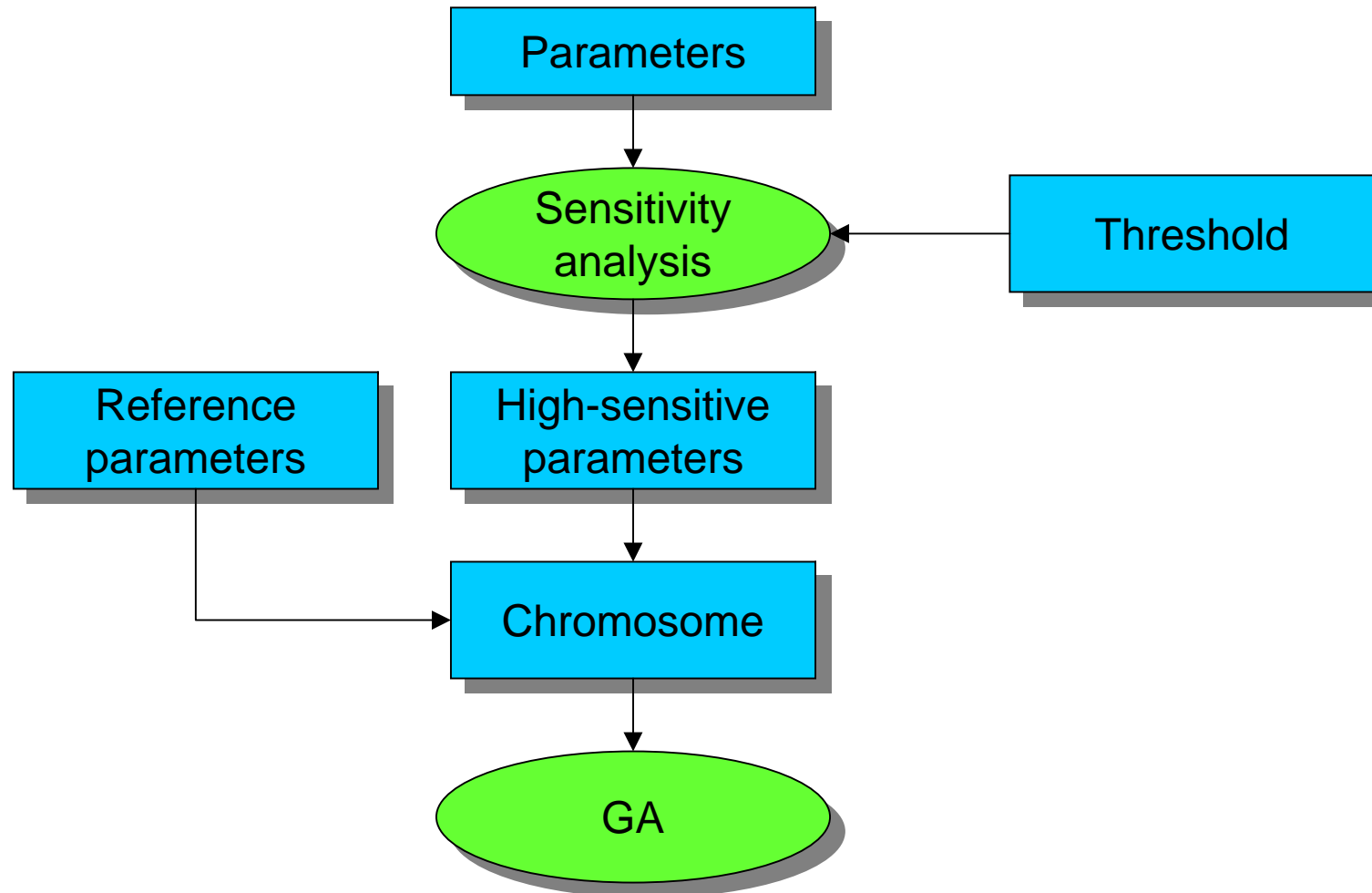
→ Exploration

✓ Fix $a = a_0, b = b_0$ and make s variable $\Rightarrow s_{opt}$

✓ Fix $s = s_{opt}, b = b_0$ and make a variable $\Rightarrow a_{opt}$

✓ Fix $s = s_{opt}, a = a_{opt}$ and make b variable $\Rightarrow b_{opt}$

SA + GA = SAGA



Overall Results

| Benchmark | dep | ga | | depga | | pbsa | | saga | |
|-----------|-------|--------|-------|--------|-------|--------|-------|--------|-------|
| | sims | % dist | % sav | % dist | % sav | % dist | % sav | % dist | % sav |
| adpcm | 14694 | 0.04 | 89.21 | 0.02 | 78.75 | 0.00 | 80.53 | 0.03 | 91.40 |
| bcnt | 22642 | 0.02 | 93.01 | 0.01 | 89.67 | 0.00 | 83.99 | 0.03 | 95.42 |
| blit | 37787 | 0.04 | 95.82 | 0.00 | 90.99 | 0.02 | 93.09 | 0.05 | 97.03 |
| compress | 14035 | 0.03 | 89.02 | 0.01 | 75.24 | 0.01 | 60.41 | 0.03 | 91.12 |
| crc | 21205 | 0.04 | 92.94 | 0.01 | 83.75 | 0.01 | 93.78 | 0.03 | 96.42 |
| des | 21970 | 0.02 | 93.00 | 0.01 | 82.60 | 0.00 | 58.77 | 0.03 | 93.18 |
| engine | 15532 | 0.03 | 90.36 | 0.02 | 78.44 | 0.00 | 70.35 | 0.03 | 91.93 |
| fir | 16832 | 0.03 | 91.23 | 0.01 | 82.60 | 0.00 | 67.77 | 0.03 | 92.43 |
| g3fax | 27382 | 0.04 | 94.26 | 0.02 | 86.20 | 0.01 | 93.09 | 0.02 | 96.83 |
| jpeg | 15996 | 0.04 | 90.61 | 0.01 | 82.15 | 0.00 | 57.48 | 0.03 | 91.87 |
| pocsag | 19102 | 0.04 | 91.49 | 0.01 | 83.93 | 0.00 | 71.49 | 0.03 | 94.66 |
| qurt | 13231 | 0.03 | 88.68 | 0.00 | 77.58 | 0.00 | 73.92 | 0.02 | 91.63 |
| ucbqsort | 19102 | 0.03 | 91.82 | 0.01 | 84.60 | 0.00 | 84.21 | 0.03 | 94.66 |
| Average | | 0.03 | 91.65 | 0.01 | 82.81 | 0.01 | 76.07 | 0.03 | 93.74 |

VLIW Embedded System

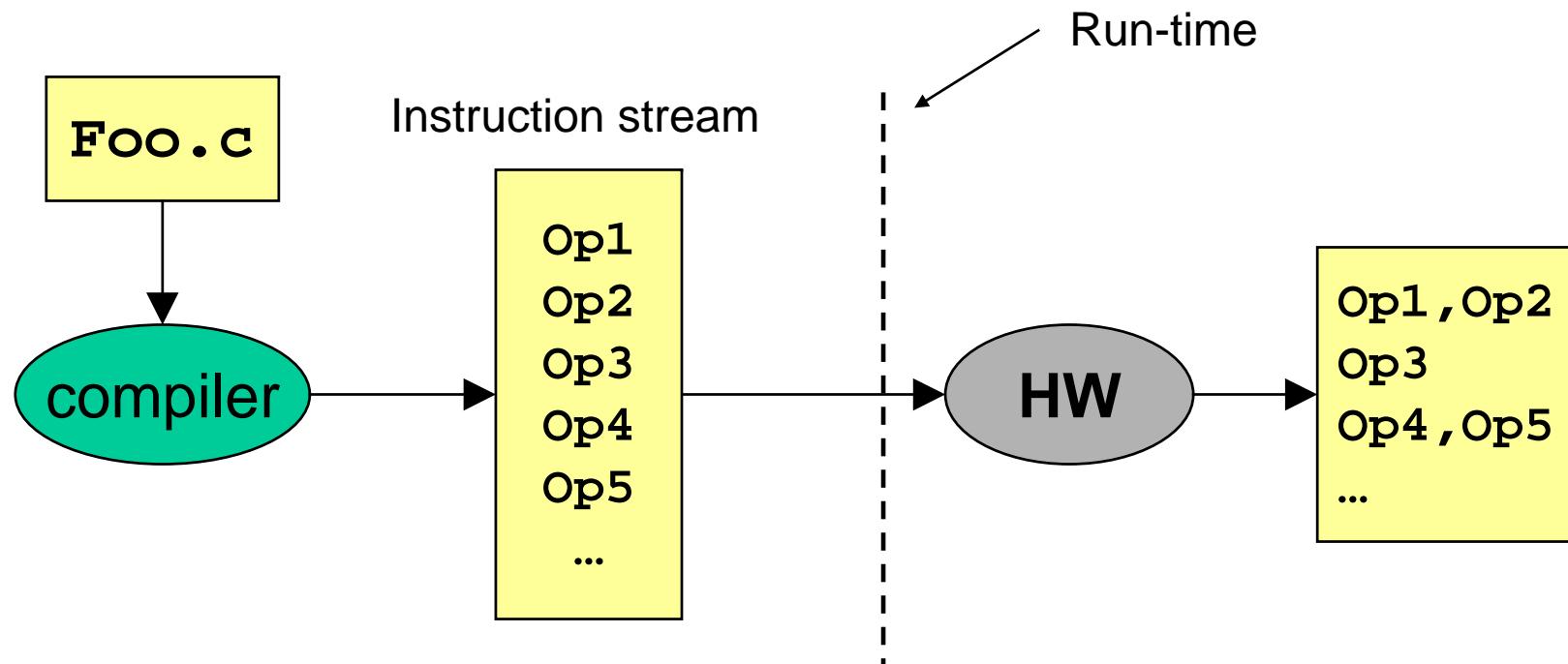
- Simultaneous interaction between
 - Hardware parameters
 - Software parameters
- Analysis of the tradeoffs
 - Power/Performance
 - Energy/Performance

Instruction Level Parallelism

- High performance processors in the 1980s:
maximize ILP
 - Issue more than one single instruction in a given clock cycle
 - Who decides which instructions can be executed in parallel?
- Two different philosophies
 - Superscalar
 - Very Long Instruction Word (VLIW)

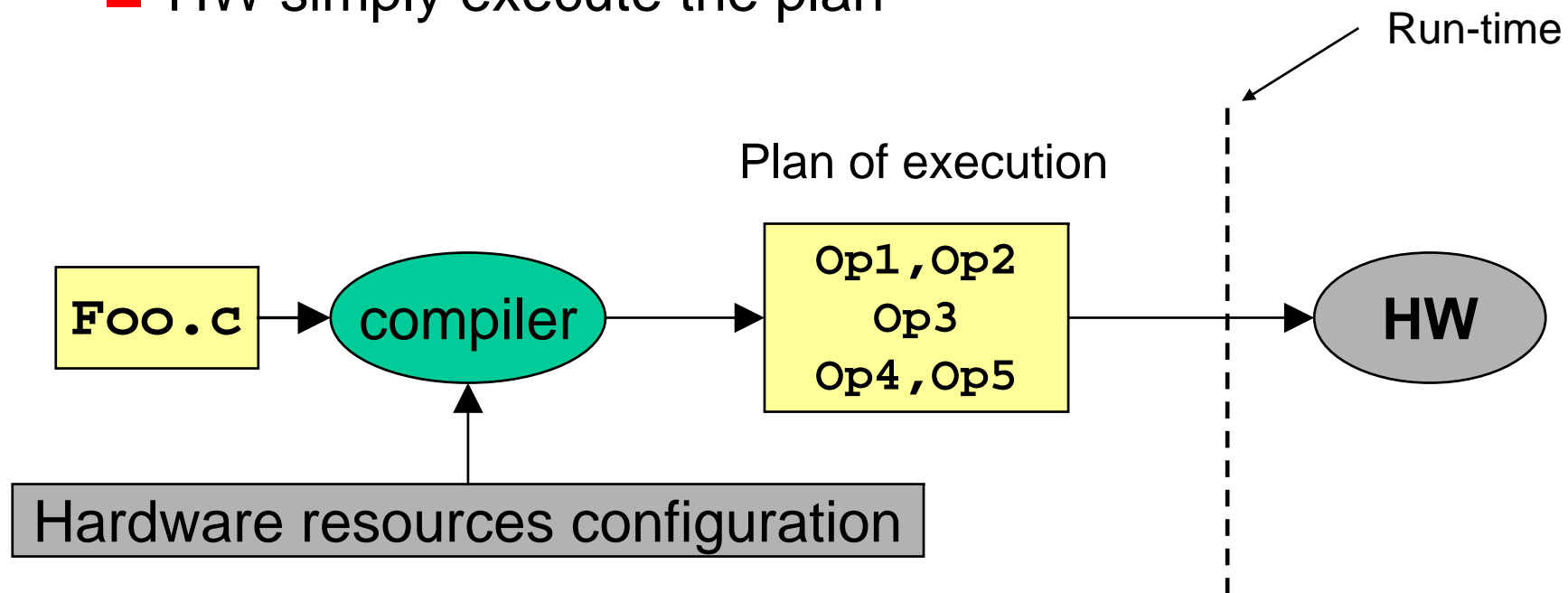
ILP philosophy: Superscalar

- Hide the process of finding ILP
- ILP is discovered *dynamically* at run-time by the control hardware of the processor



ILP philosophy: VLIW

- Hardware resources are *architecturally visible* to the compiler
- Compiler can create a sequence of **Very Long Instructions** that defines the *plan of execution*
- HW simply execute the plan



VLIW past & future

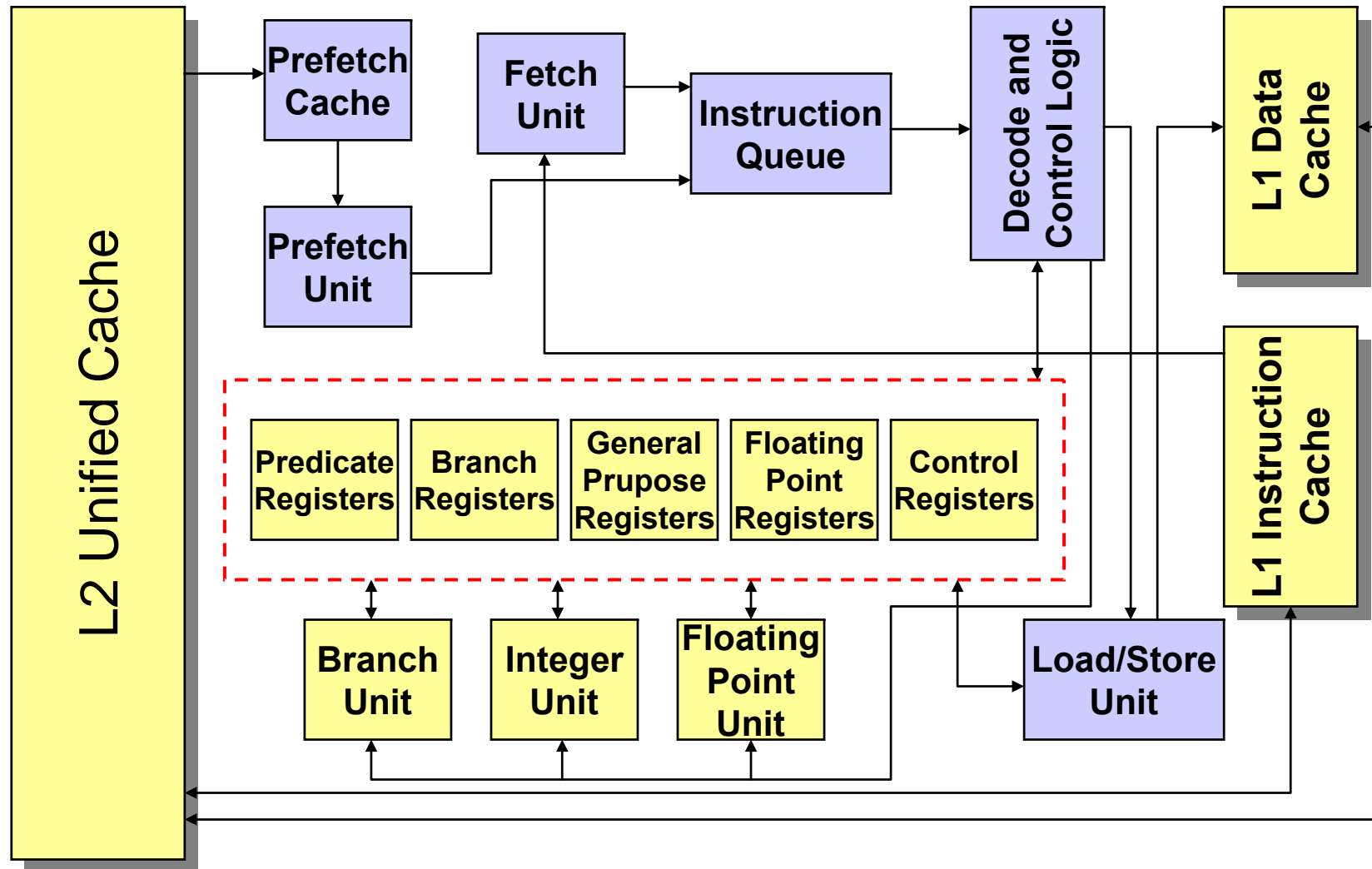
- Decline of VLIWs for general purpose systems
 - Couldn't be integrated in a single chip
 - Binary compatibility between implementations

- Rediscovery of VLIW in embedded
 - No more integrability issues
 - Binary incompatibility not relevant
 - ✓ SW is very specific and supplied with the HW
 - Advantages of VLIW
 - ✓ Simplified hardware
 - ✓ Optimize ad-hoc the architecture to achieve ILP

ASIP VLIW in Embedded System

- ASIP based on VLIW architectures
- Examples are
 - TI C6x family
 - Agere/Motorola StarCore architecture
 - Sun MAJC
 - Fujitsu FR-V
 - ST HP/ST Lx
 - Philips TriMedia
 - ...

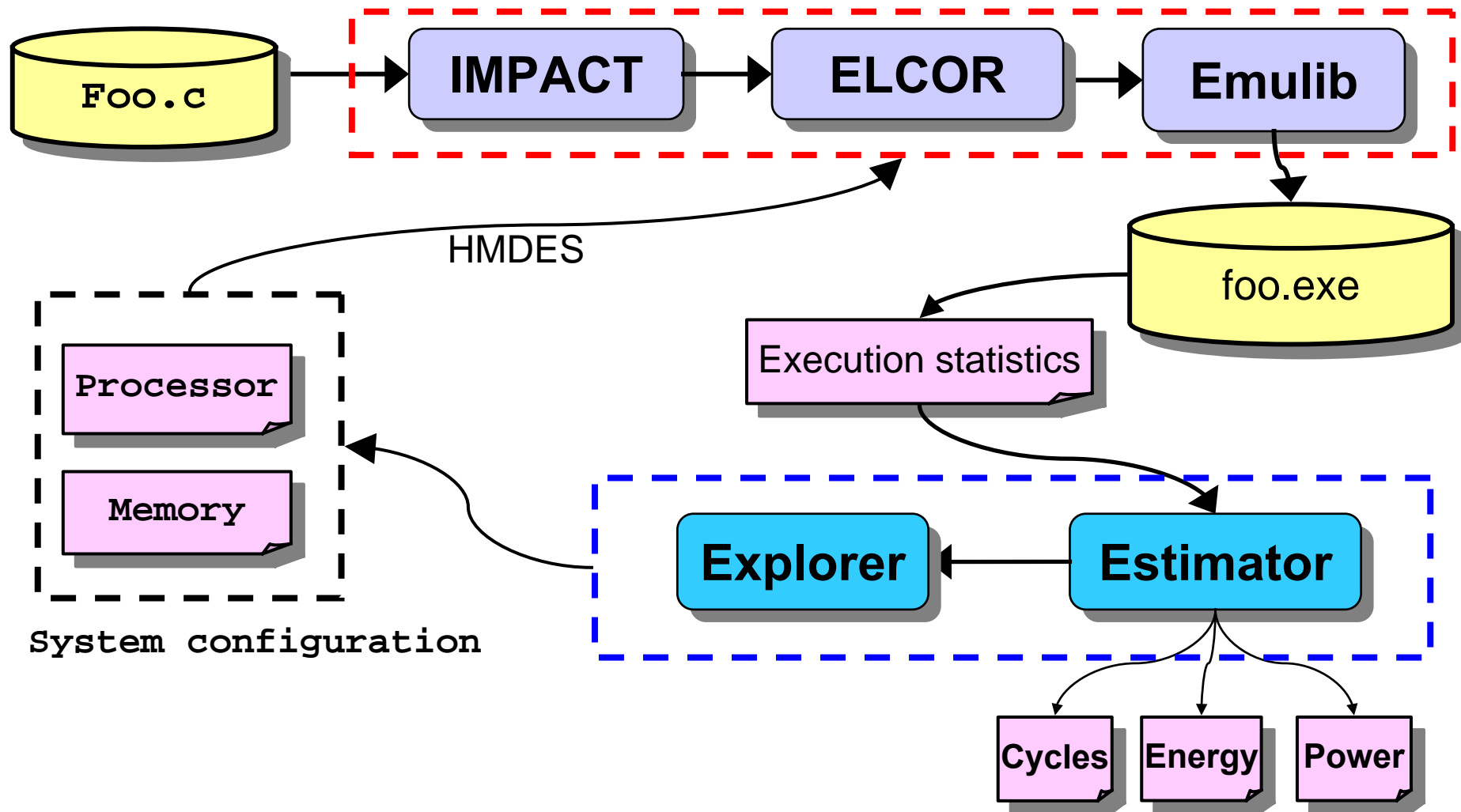
Reference architecture (HPL-PD)



An Open Platform: EPIC Explorer

- Interfacing to the **Trimaran** framework that provide VLIW **compiler** and **simulator** for dynamic statistics
- **Estimator** component implementing high level models
- **Explorer** component implementing multi-objective design space exploration algorithms
- **EPICExplorer**
 - ➔ <http://epic-explorer.sourceforge.net>

The Exploration Data Flow



Energy estimation

- Subdivide architecture in *Functional Block Unit* (FBU)
 - Instruction decode logic, Integer units, floating point units, register files
- For each FBU (from ST Microelectronics LX)
 - **Active power**: average power dissipated when the FBU is used
 - **Inactive power**: average power dissipated when the FBU is not used
- From the execution statistic, we know how many cycles each FBU has been active/inactive
 - $E_{\text{FBU}} = (P_{\text{active}} \times \text{cycles}_{\text{active}} + P_{\text{inactive}} \times \text{cycles}_{\text{inactive}}) \times T_{\text{clock}}$
- Discrete degree of accuracy (about 25%)
 - investigate relative power savings between designs

Configuration Space

Three main parameter categories:

- **VLIW core:**

- Number of Registers in each register file (from 16 to 256)
- Number of instances for Functional Units of each type (from 1 to 6)

- **Mem Hierarchy:**

- Size, Blocksize, Associativity for each of the caches (L1 Instruction, L1 Data, L2)

- **Compiler:**

- Conservative compilation strategy (basic blocks)
- Aggressive ILP oriented compilation strategy (hyperblocks)

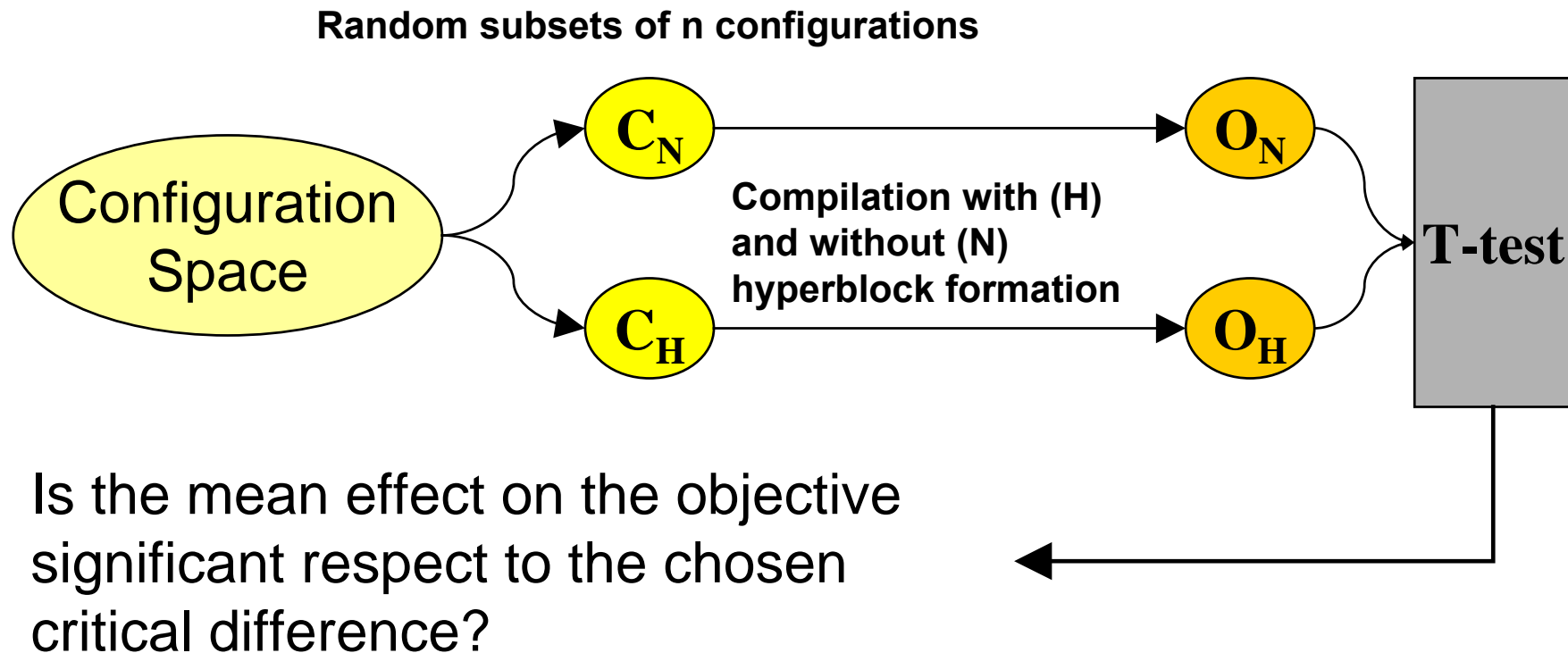
Total space size: **1.47 x 10¹³ configurations!**

Exploration Methodology

- Preliminary analysis of compilation
 - Impact of ILP oriented code transformations
 - Predict the right compilation strategy
 - ✓ Basic Blocks (conservative)
 - ✓ Hyper Blocks (aggressive, ILP-oriented)
- Multi-objective Design Space Exploration
 - Extract Pareto Set

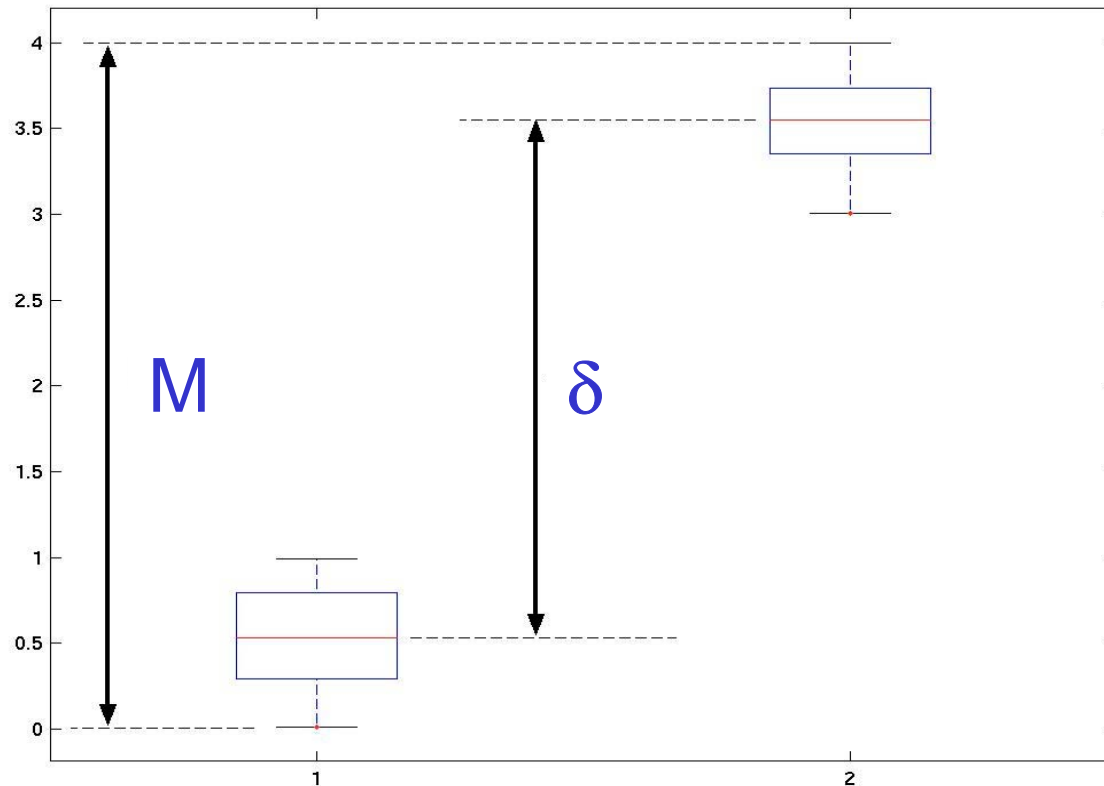
Preliminary Analysis (1/3)

- For each objective, *Unpaired two sample t-test* allows to estimate the average effect of hyperblock formation



Preliminary Analysis (2/3)

- Example of a metric for critical difference in means: $\delta > 50\% M$

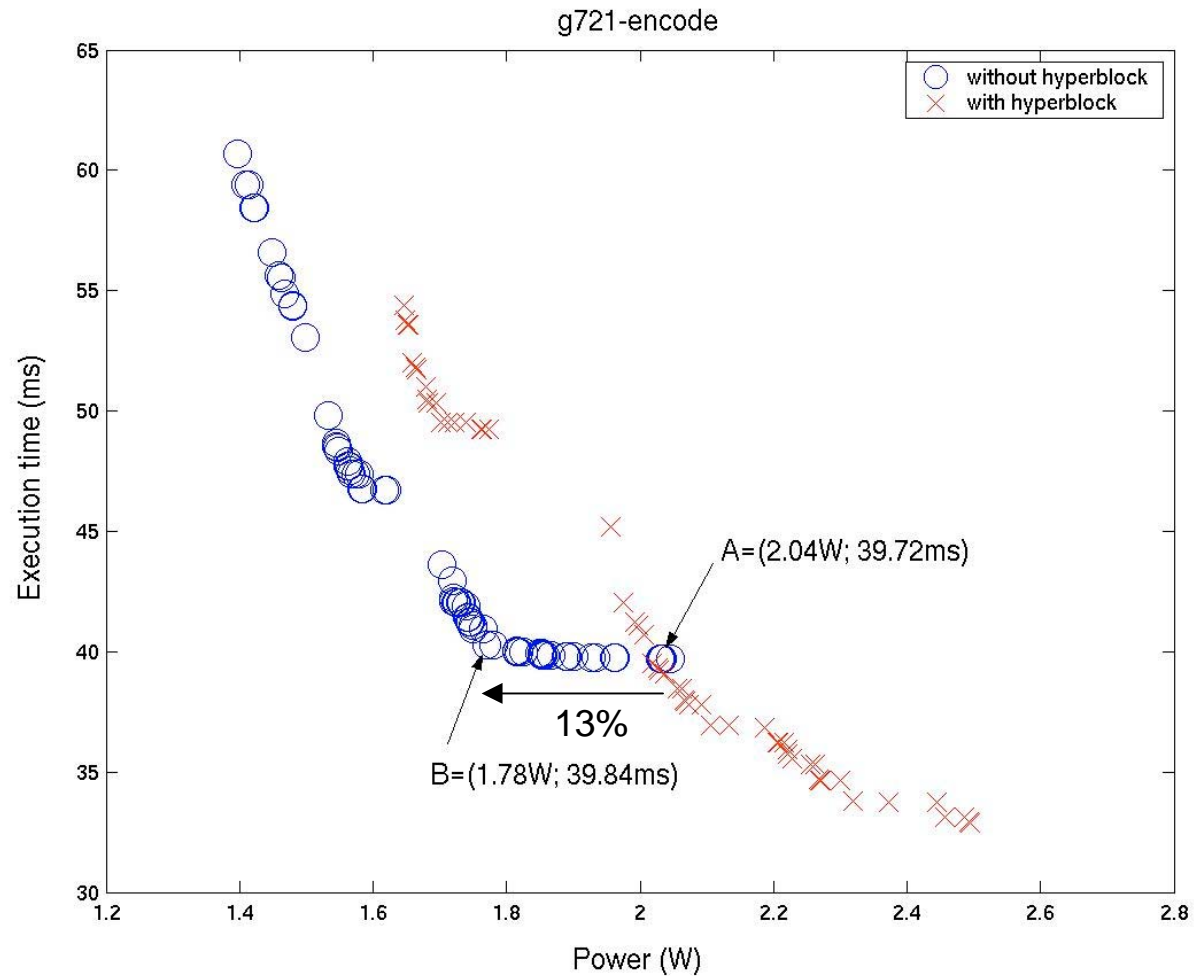


Preliminary Analysis (3/3)

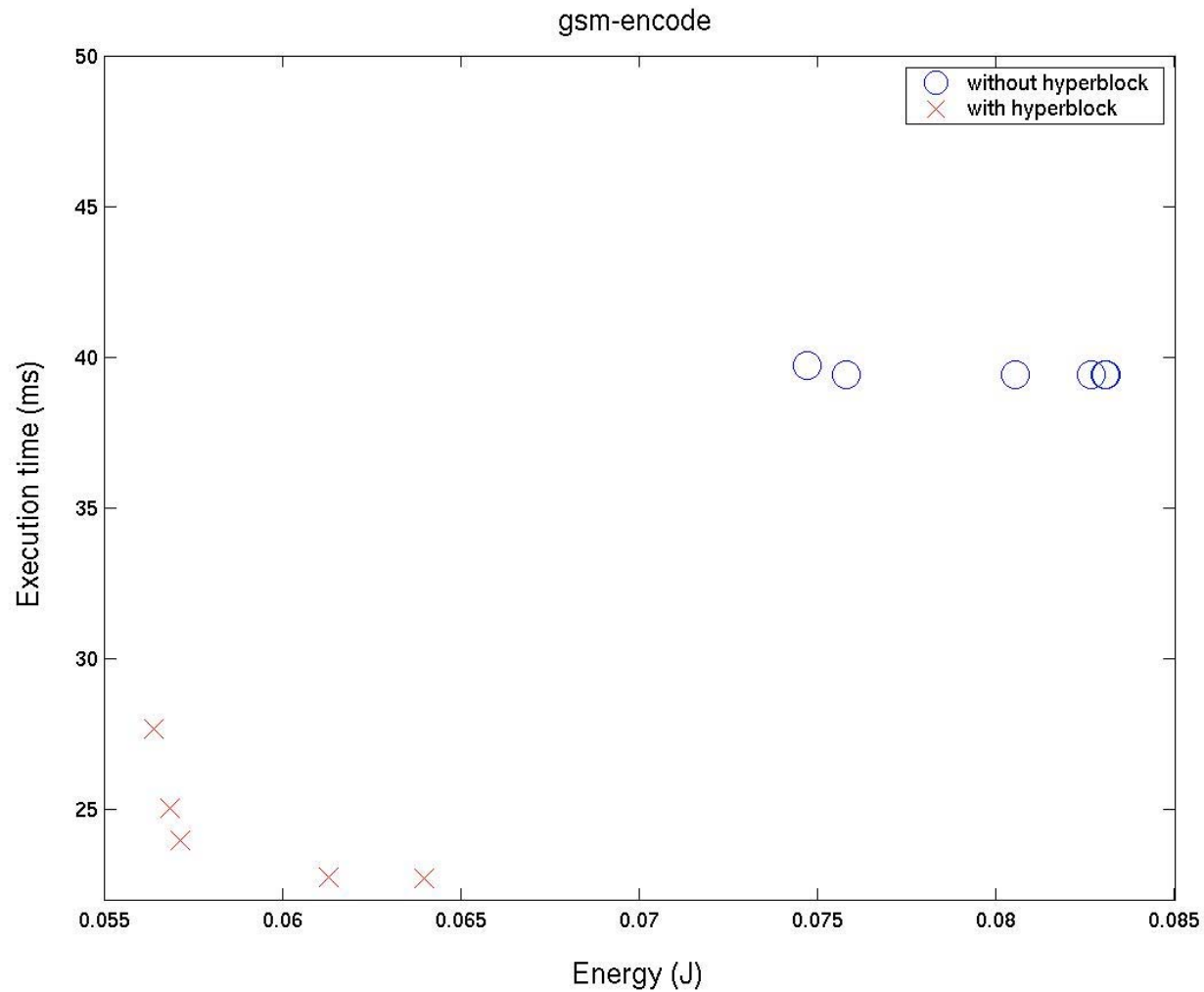
ILP-oriented compilation impact (positive, negative)

| Application | Time (ms) | | Power (W) | | Energy (mJ) | |
|-------------|--------------|--------------------|-------------|-------------------|--------------|-------------------|
| | Δ | $\mu_N - \mu_H$ | Δ | $\mu_N - \mu_H$ | Δ | $\mu_N - \mu_H$ |
| ieee810 | 16.64 | 6.76+1.84 | 1.64 | 0.38+0.16 | 49.01 | 30.82+4.55 |
| gsm-enc | 36.62 | 33.25+4.79 | 0.88 | -0.48+0.14 | 79.28 | 55.84+9.82 |
| jpeg | 4.07 | -0.97+0.51 | 0.89 | -0.07+0.09 | 9.72 | -2.31+1.01 |
| adpcm-enc | 15.8 | 8.17+2.2 | 1.25 | -0.89+0.14 | 46.12 | -8.56+3.73 |
| mpeg-dec | 33.39 | -5.28+4.85 | 0.88 | 0.25+0.16 | 62.50 | -3.48+9.88 |
| g721-enc | 22.76 | -7.23+2.95 | 0.76 | -0.39+0.08 | 65.53 | -32.4+5.9 |
| adpcm-dec | 24.2 | -6.19+3.31 | 1.02 | -0.5+0.12 | 58.54 | -27.74+7.3 |
| fir | 0.68 | -0.26+0.08 | 0.79 | -0.27+0.09 | 1.40 | -0.97+0.12 |
| gsm-dec | 21.55 | -23.83+2.58 | 0.54 | -0.24+0.09 | 59.60 | -56.6+6.43 |

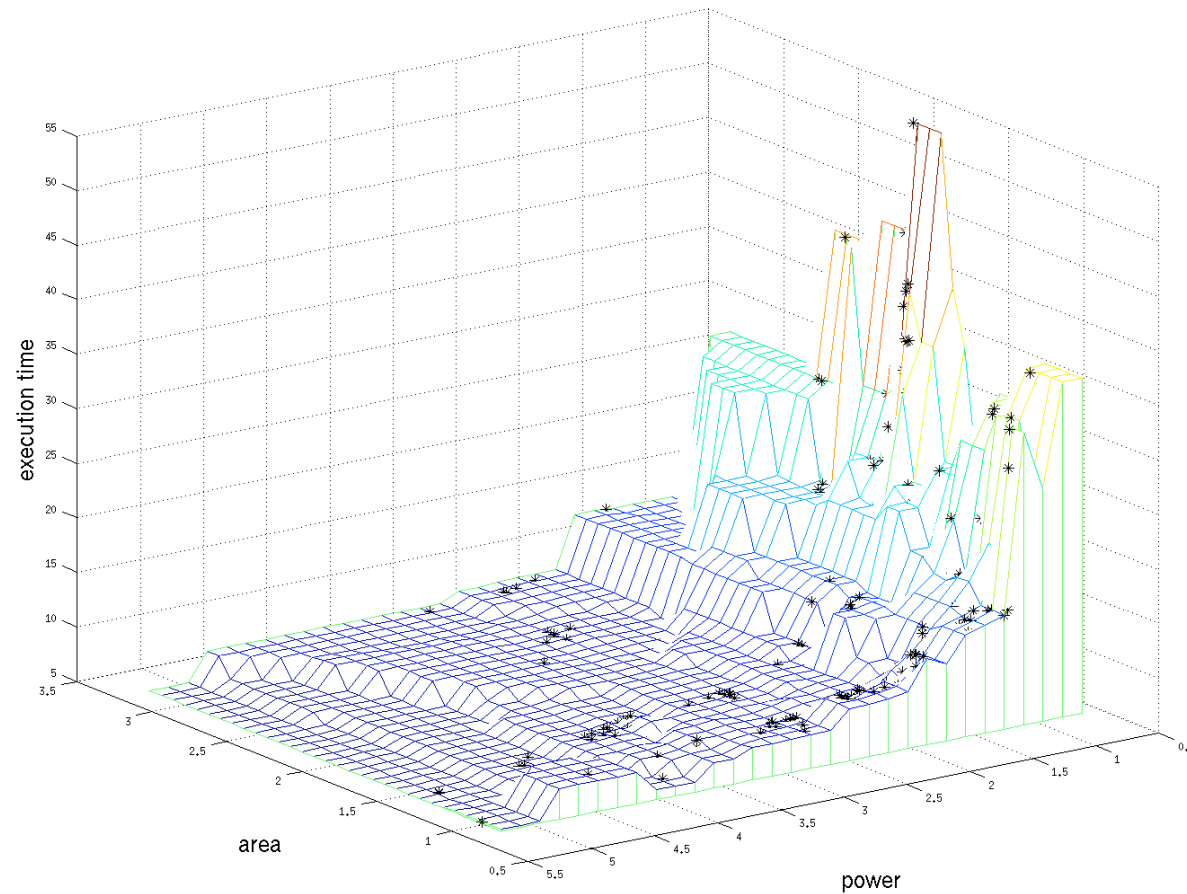
Pareto Set (G721 encode)



Pareto Set (GSM-encode)



Pareto Surface



Outline

- Design Space Exploration of Parameterized Systems
- **Network on Chip Architectures**
- Instruction Level Power Estimation

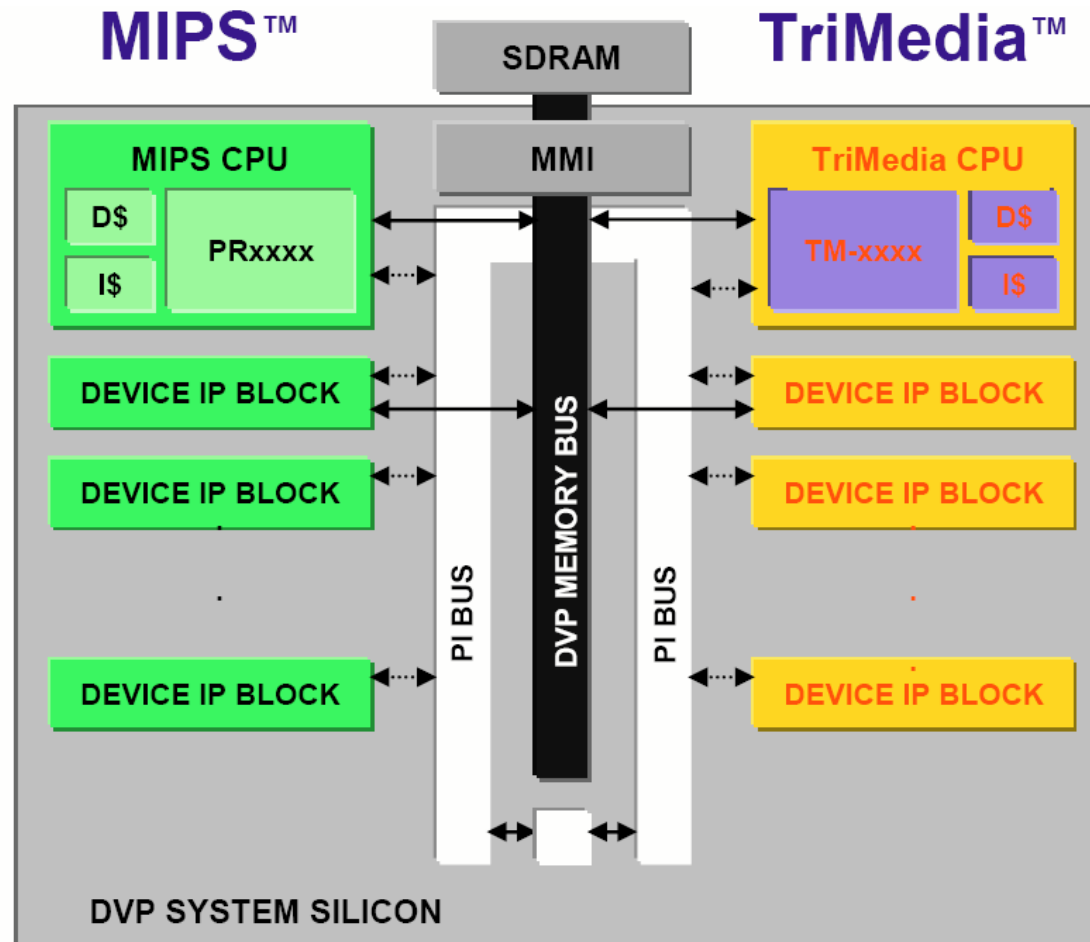
The Evolution of SoC Platforms

General-purpose Scalable RISC Processor

- 50 to 300+ MHz
- 32-bit or 64-bit

Library of Device IP Blocks

- Image coprocessors
- DSPs
- UART
- 1394
- USB



Scalable VLIW Media Processor:

- 100 to 300+ MHz
- 32-bit or 64-bit

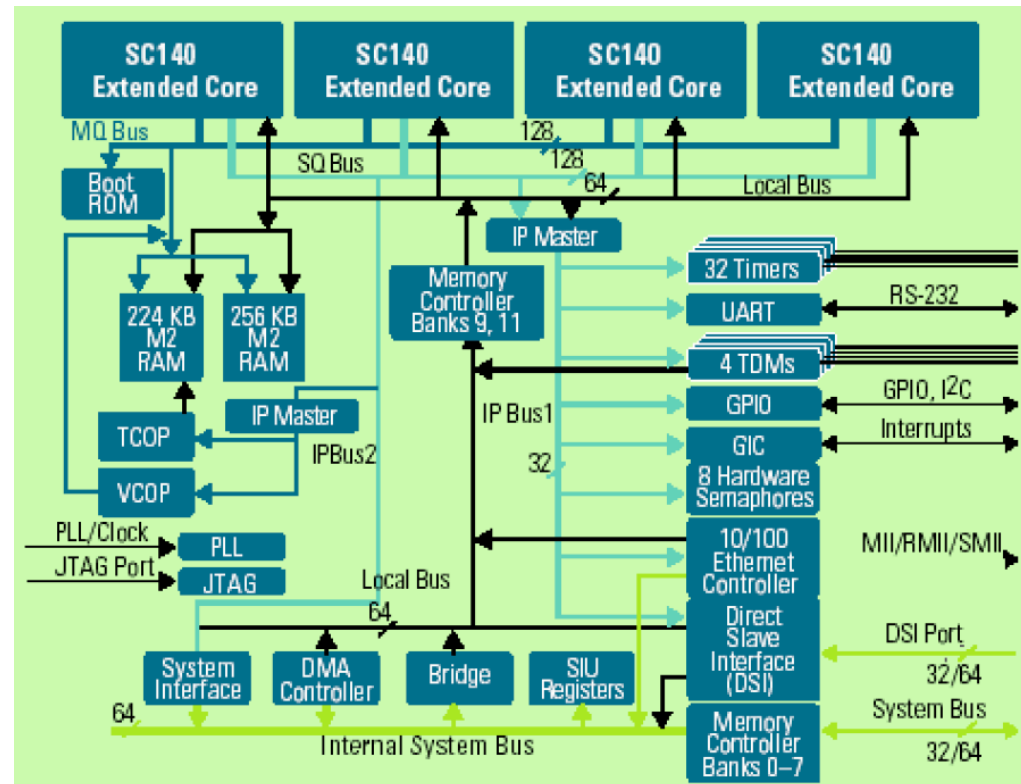
Nexperia™ System Buses

- 32-128 bit

2 Cores: Philips' Nexperia PNX8850 SoC platform for High-end digital video (2001)

Running Forward...

- Four 350/400 MHz StarCore SC140 DSP extended cores
- 16 ALUs: 5600/6400 MMACS
- 1436 KB of internal SRAM & multi-level memory hierarchy
- Internal DMA controller supports 16 TDM unidirectional channels,
- Two internal coprocessors (TCOP and VCOP) to provide special-purpose processing capability in parallel with the core processors

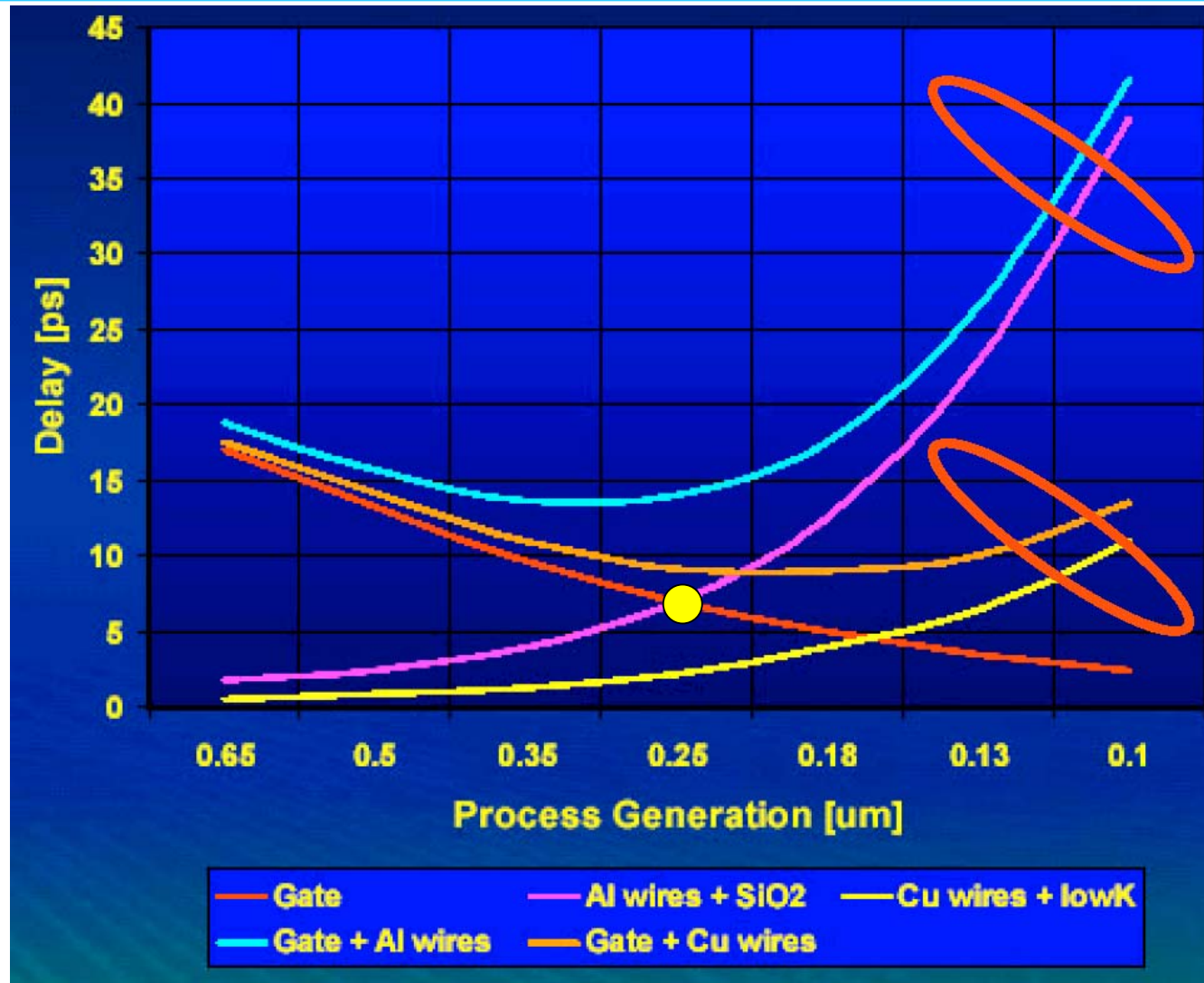


6 Cores: Motorola's MSC8126 SoC platform for 3G base stations (late 2003)

What's Happening in SoCs?

- Technology: no slow-down in sight!
 - Faster and smaller transistors: 90 → 65 → 45 nm
 - ... but slower wires, lower voltage, more noise!
 - ✓ 80% or more of the delay of critical paths will be due to interconnects
- Design complexity: from 2 to 10 to 100 cores!
 - Design reuse is essential
 - ...but differentiation/innovation is key for winning on the market!
- Performance and power: GOPS for MWs!
 - Performance requirements keep going up
 - ...but power budgets don't!

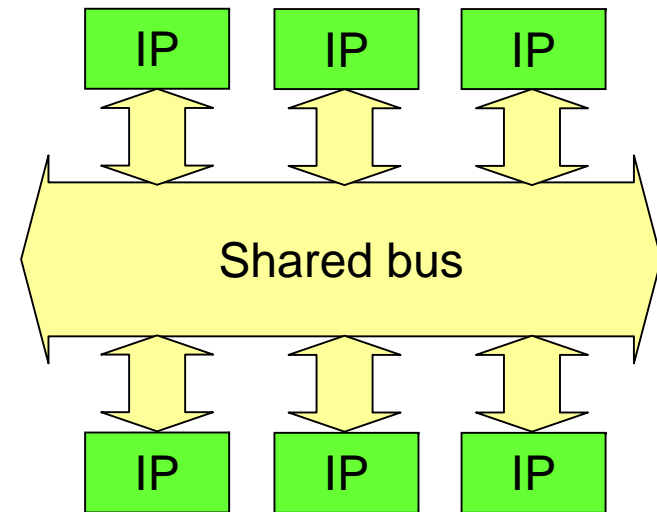
The Deep Submicron Effects



Communication Architectures

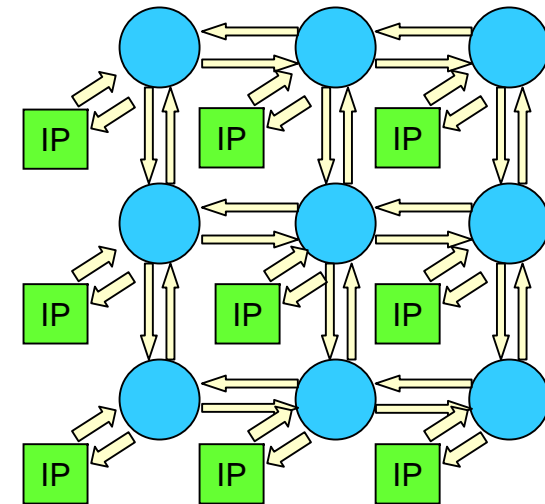
■ Shared bus

- Low area
- Poor scalability
- High energy consumption

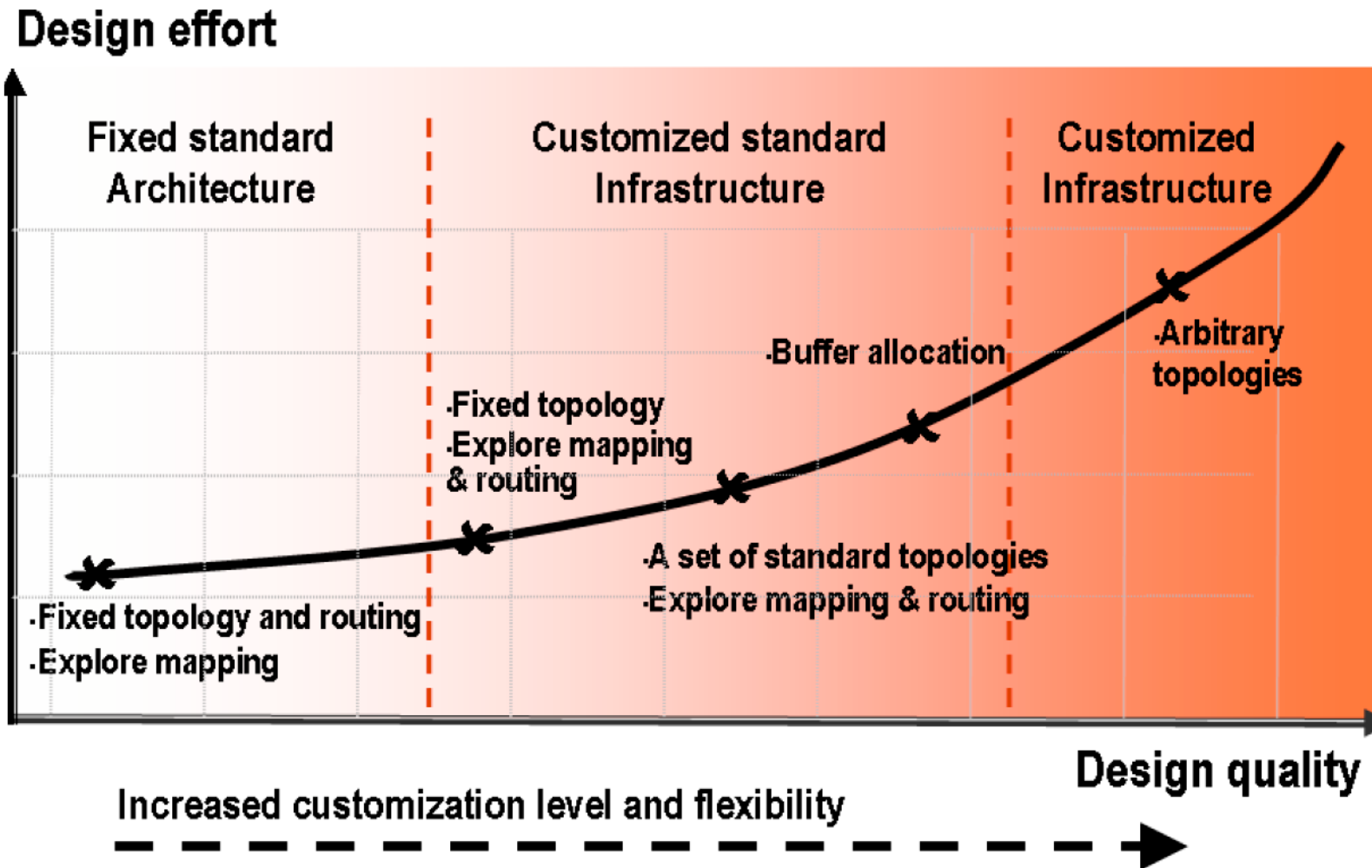


■ Network-on-Chip

- Scalability and modularity
- Low energy consumption
- Increase of design complexity



Design Space Exploration for NoC



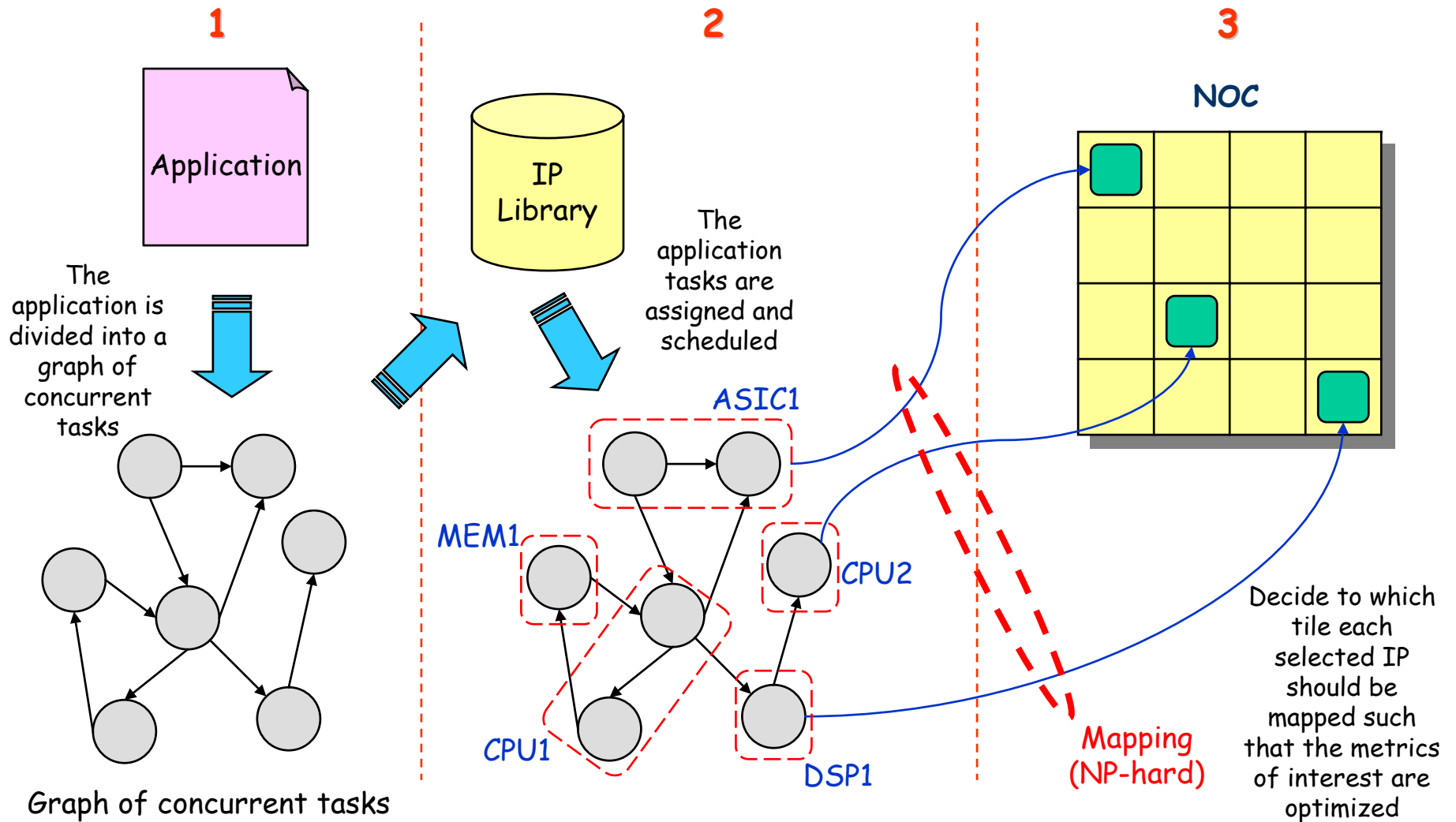
Ogras *et al.*, ASAP'05

NOC Research Topics

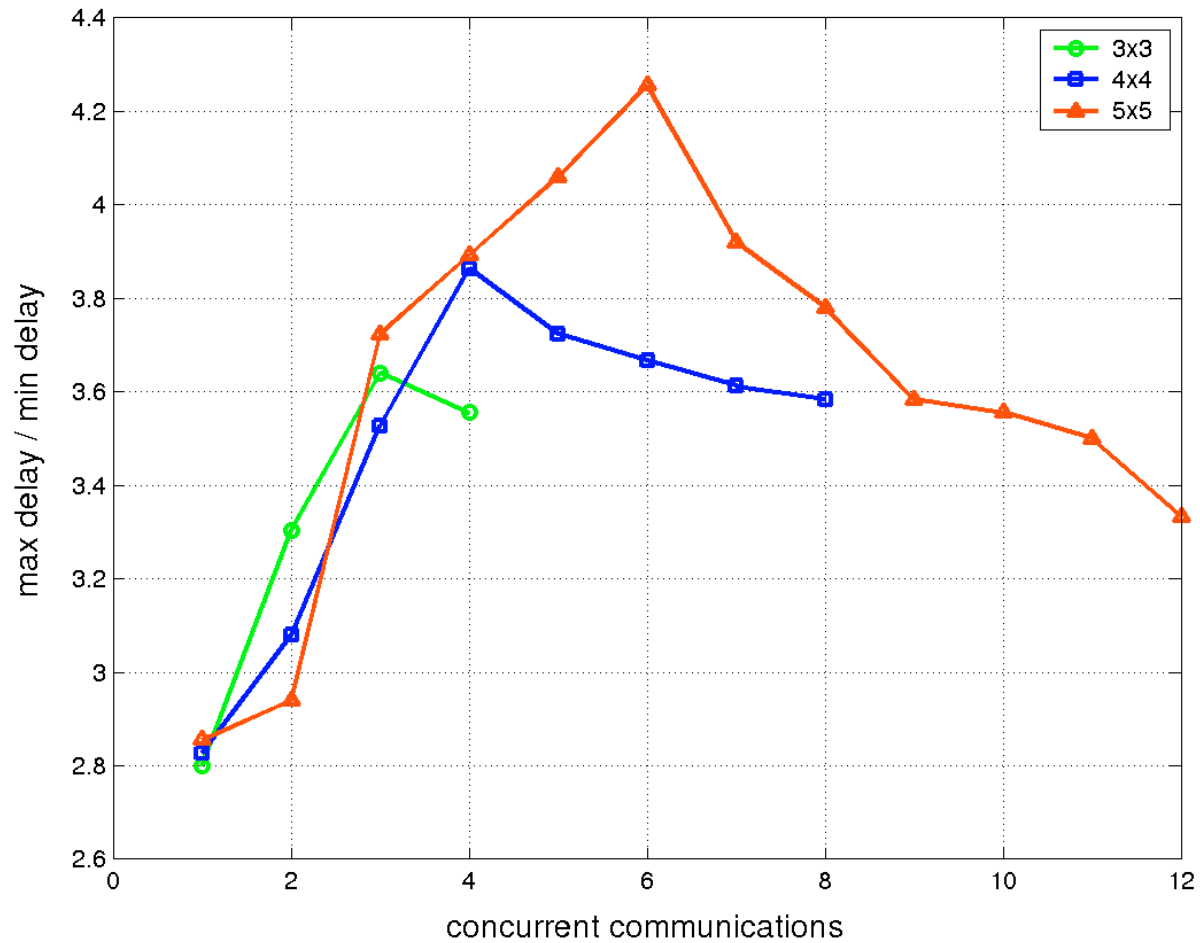
- Topological Mapping Problem
- Routing Strategies
 - Routing Function
 - Selection Function

Topological Mapping

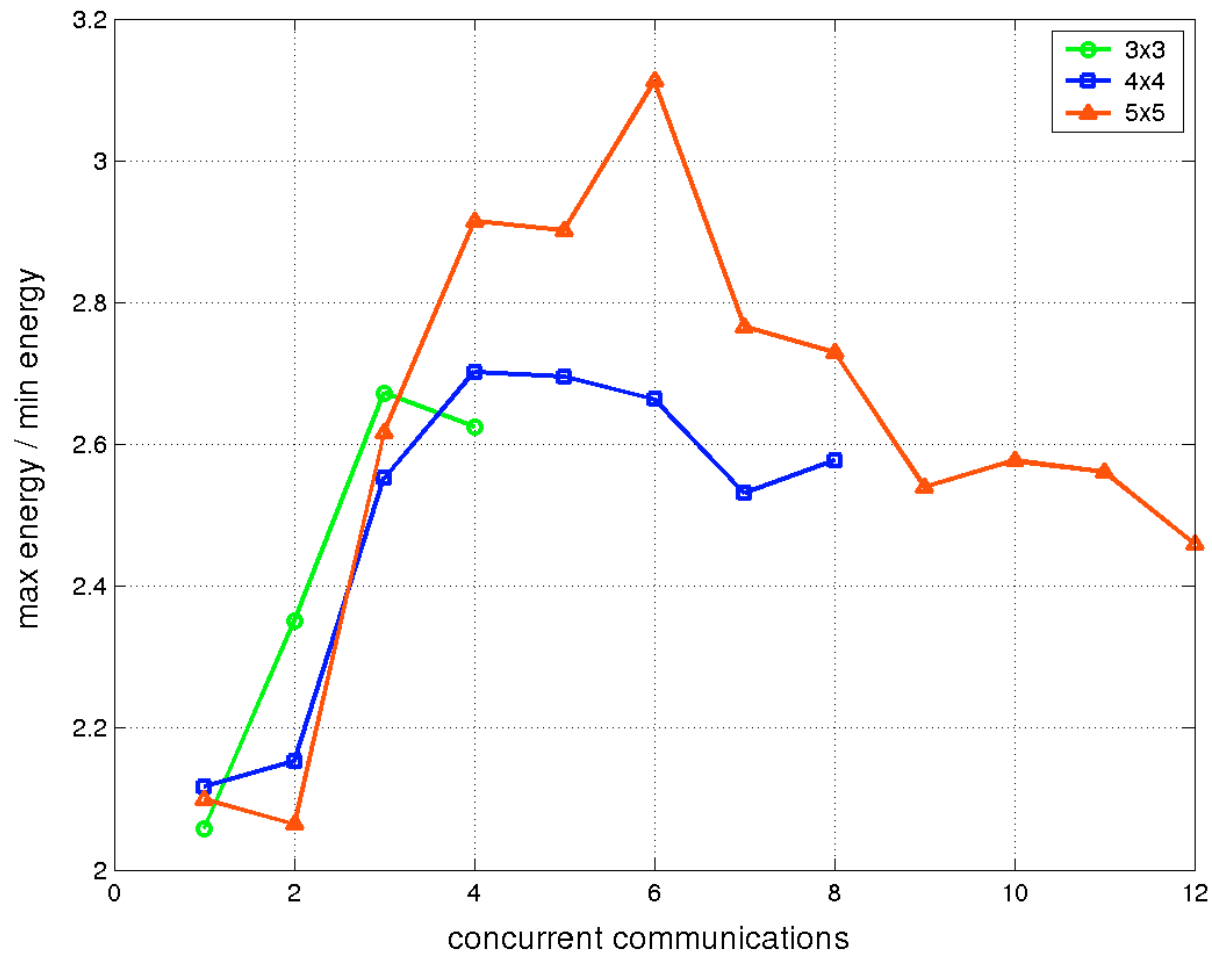
The Mapping Problem



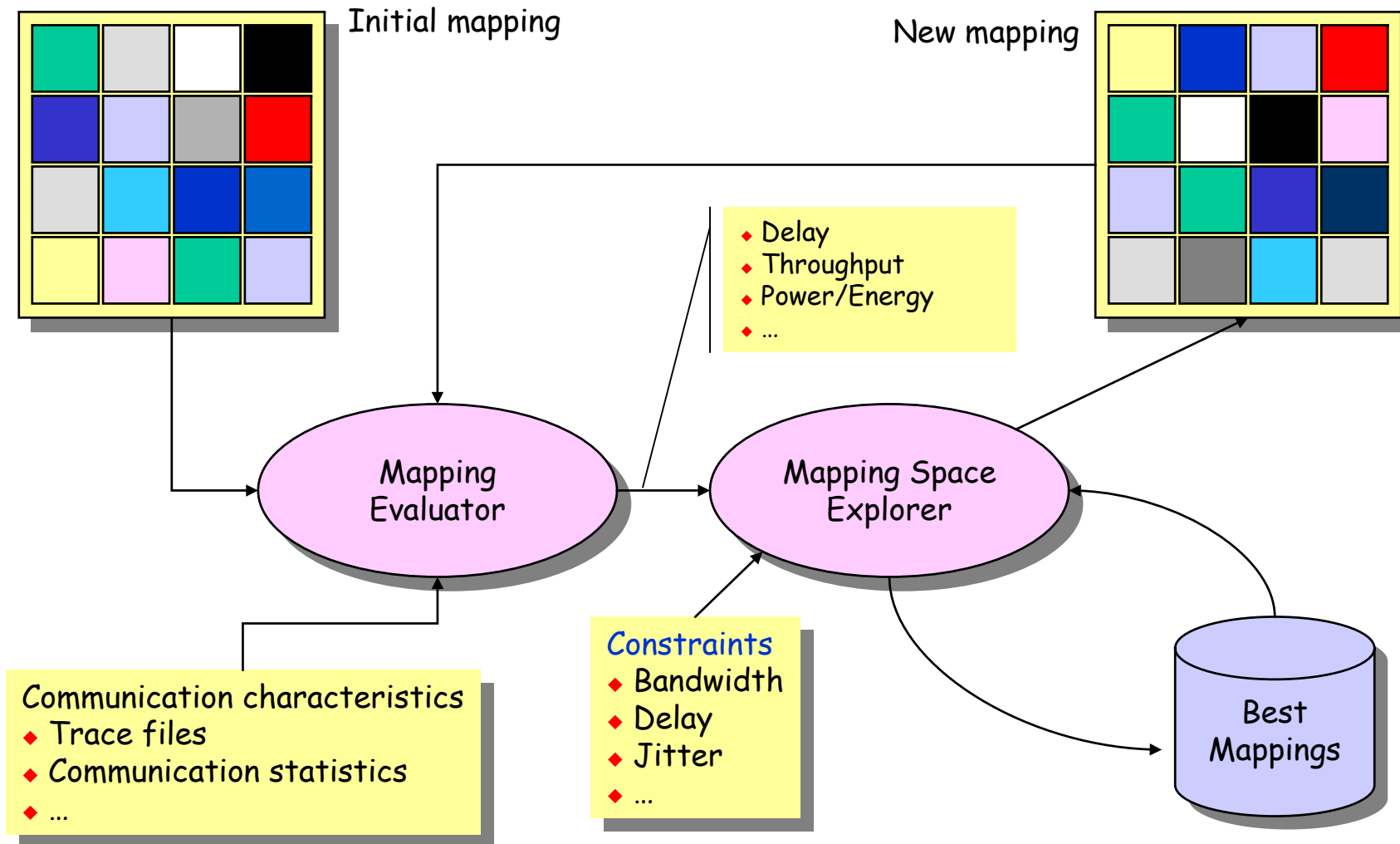
Impact on Performance



Impact on Energy



Design Space Exploration



Exploration Engines

■ **GAMAP**

→ Multi-objective mapping based on genetic algorithms

■ **PBBB**

→ Pareto-based Branch & Bound approach

- ✓ Adaptation of the approach proposed by Hu and Marculescu [ASPDAC03] for a multi-objective exploration of the mapping space

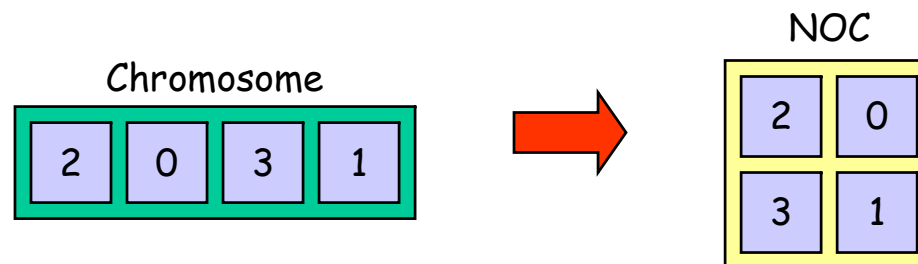
■ **PBNMAP**

→ Pareto-based NMAP approach

- ✓ Adaptation of the approach proposed by Murali and De Micheli [DATE04] for a multi-objective exploration of the mapping space
 - For the XY routing

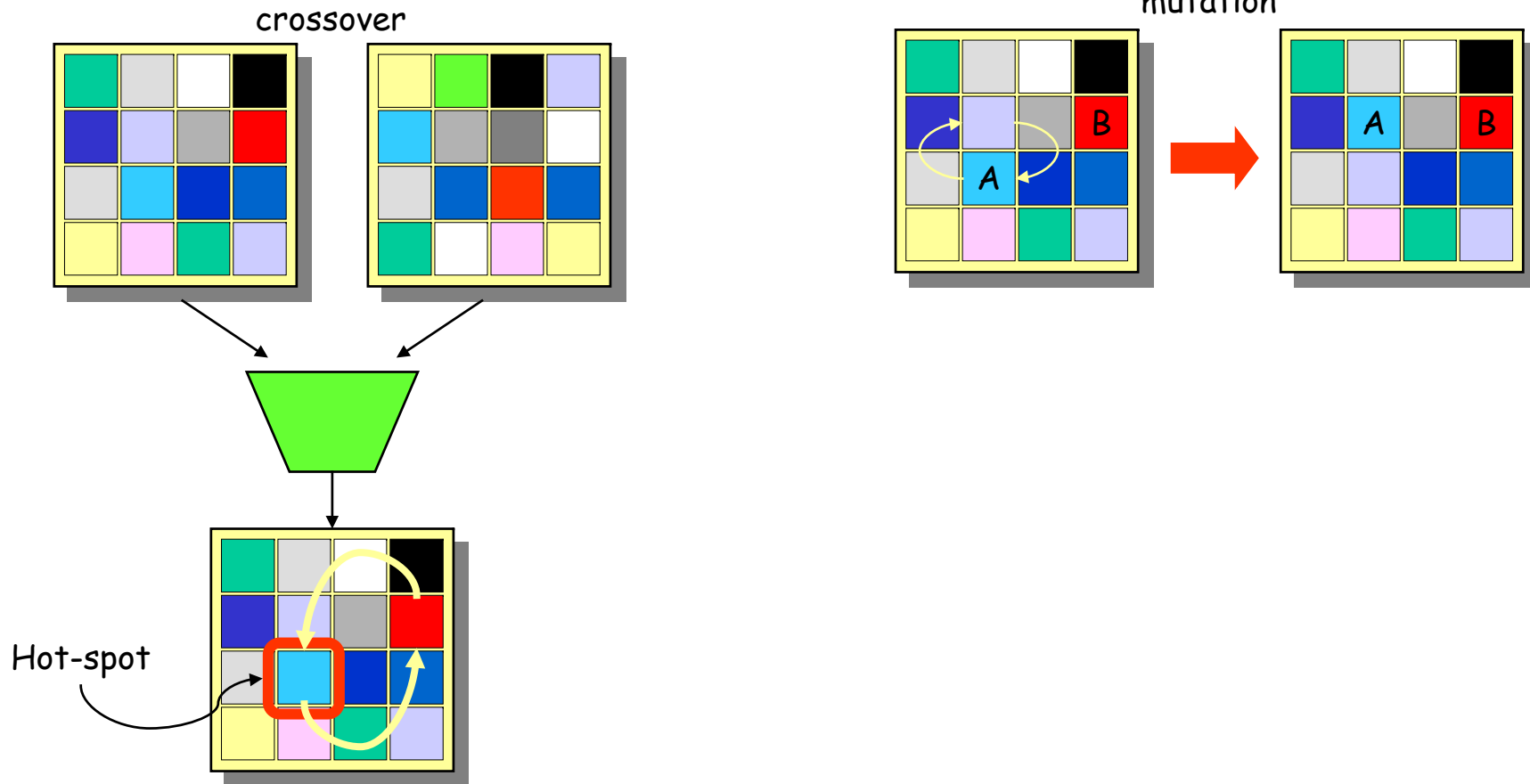
GAMAP

- **Chromosome:** A representation of the solution to the problem (the mapping)
 - Each tile in the mesh has an associated gene which encodes the identifier of the core mapped in the tile
 - In an $n \times m$ mesh
 - ✓ The chromosome is formed by $n \times m$ genes
 - ✓ The i -th gene encodes the identifier of the core in the tile in row i/n and column $i \% m$



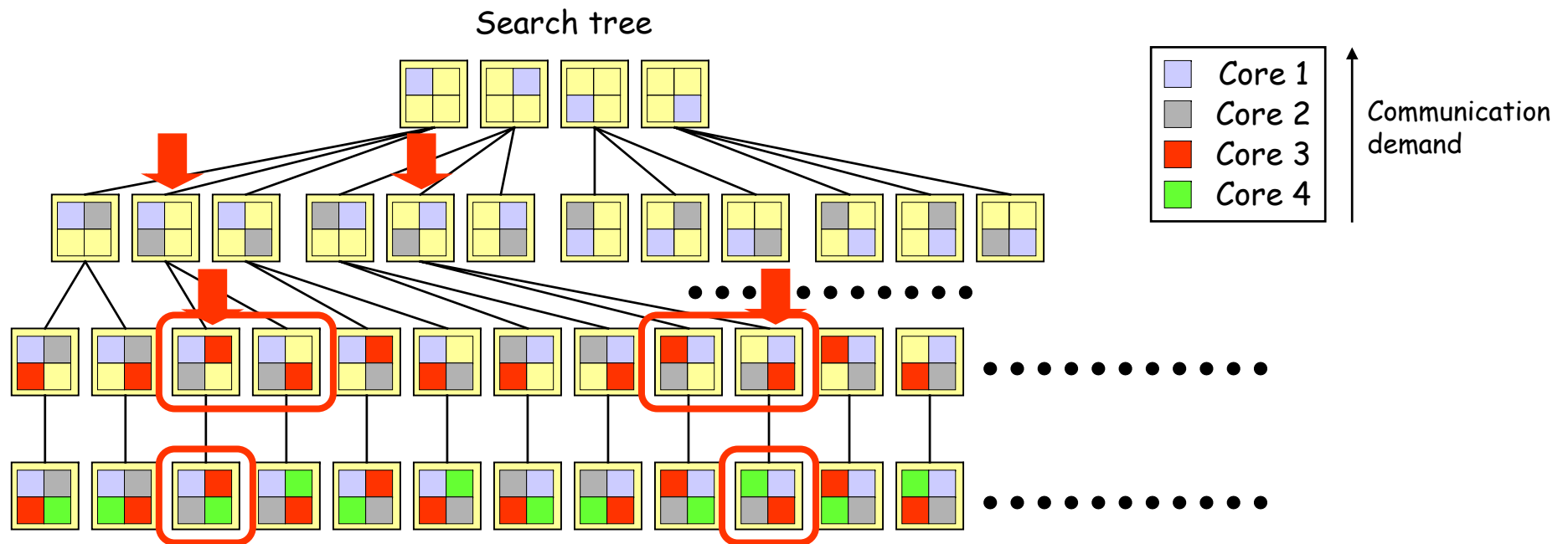
GAMAP

■ Genetic operators



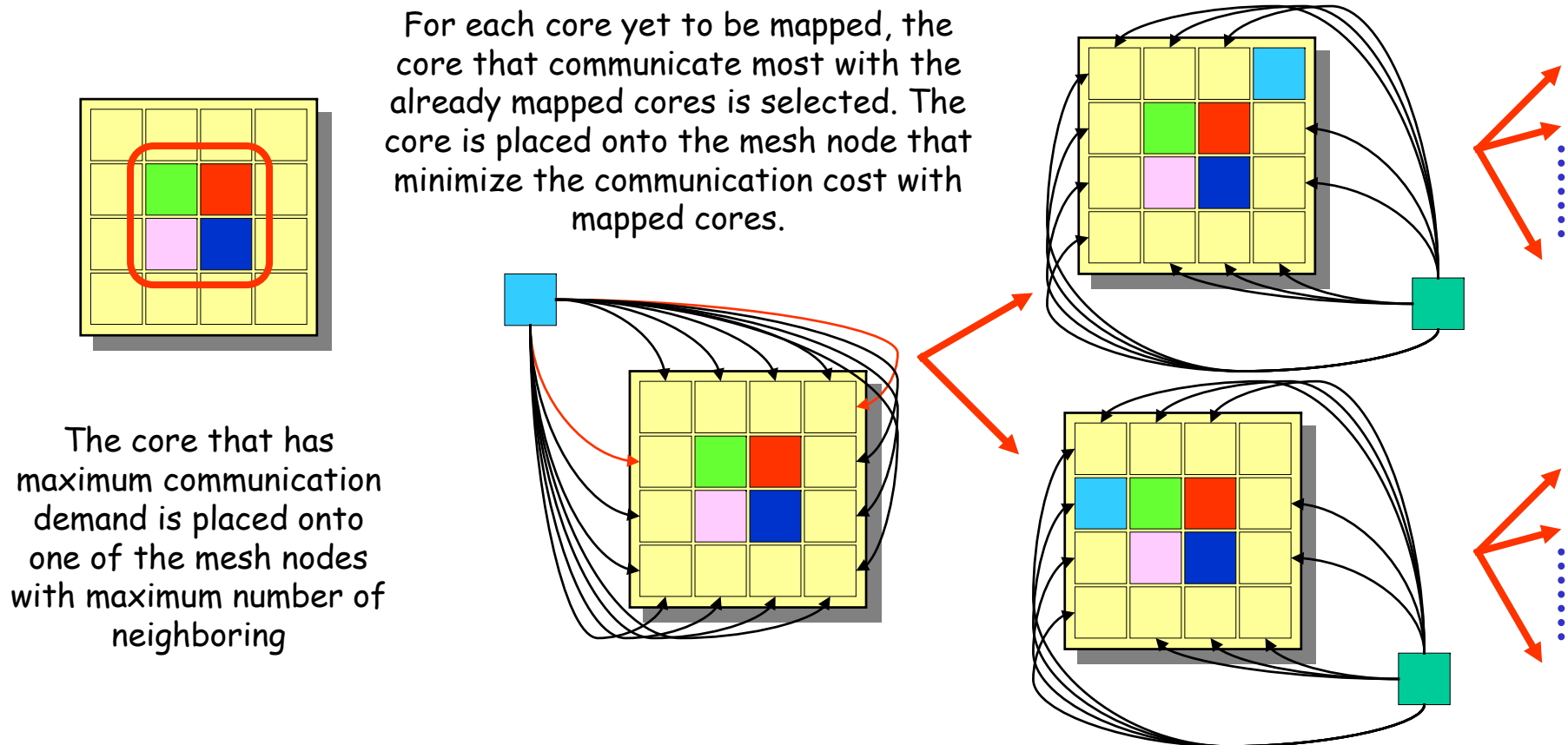
PBBB

- Multi-objective extension of the Branch-and-Bound algorithm



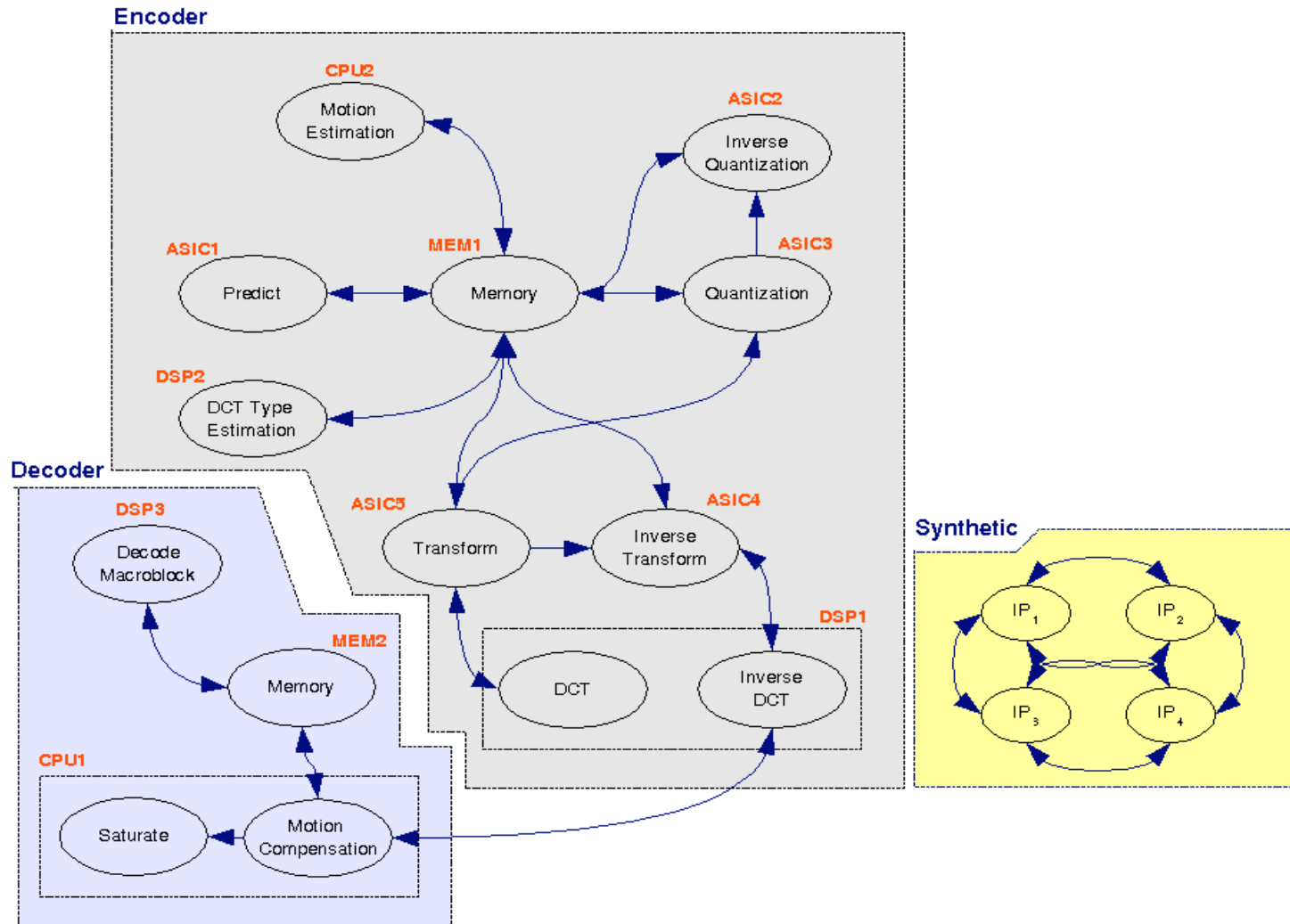
PBNMAP

- Multi-objective extension of the NMAP algorithm proposed by Murali and De Micheli in [DATE04] for the single XY routing



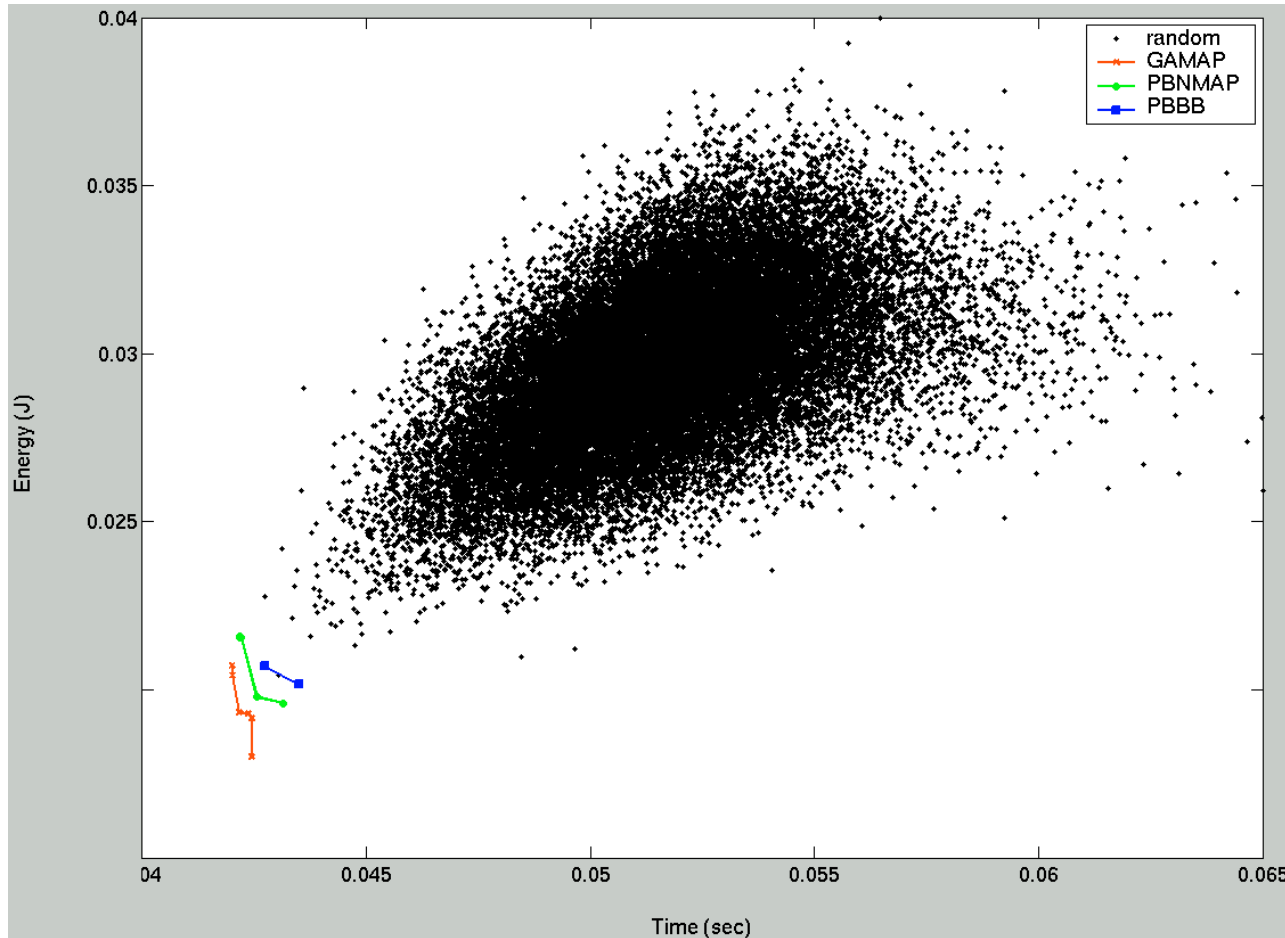
Experiments

Real Traffic (MPEG-2)



Experiments

Real Traffic (MPEG-2)



High performance mapping

| | 0 | 1 | 2 | 3 |
|---|-------|-------|------|-------|
| 0 | ASIC5 | ASIC4 | DSP2 | CPU2 |
| 1 | CPU1 | ASIC3 | MEM1 | ASIC1 |
| 2 | DSP1 | ASIC2 | IP3 | IP1 |
| 3 | MEM2 | DSP3 | IP2 | IP4 |

Execution time 41.0 ms
Energy: 20.8 mJ

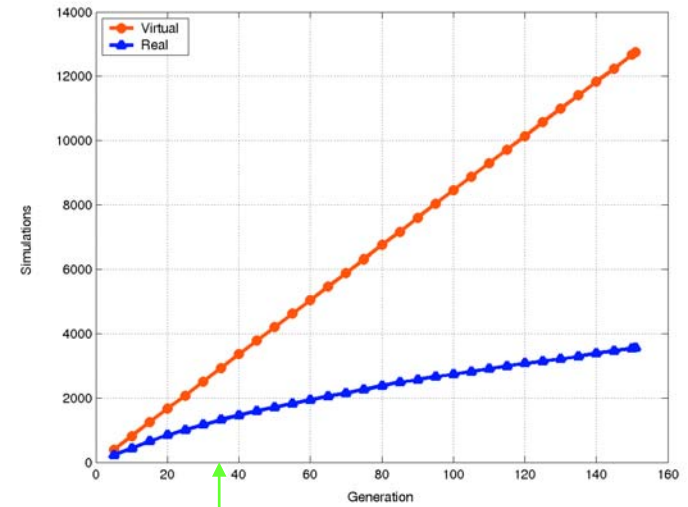
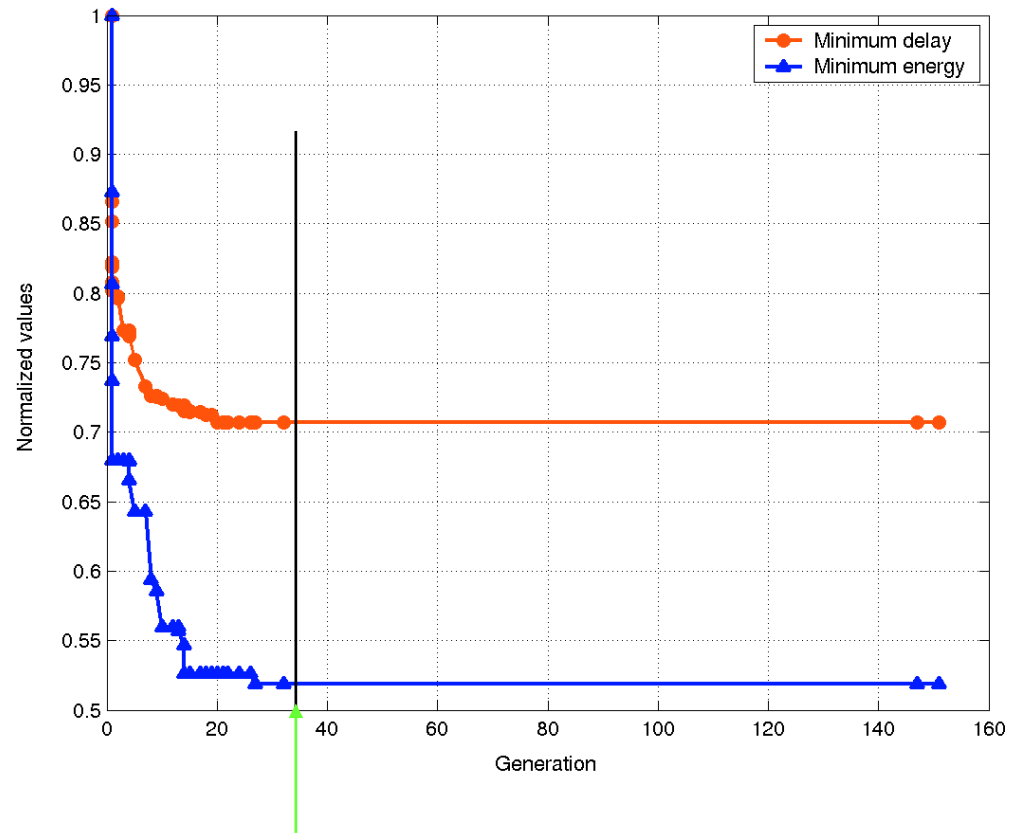
Low energy mapping

| | 0 | 1 | 2 | 3 |
|---|-------|-------|-------|------|
| 0 | ASIC5 | ASIC3 | ASIC1 | DSP2 |
| 1 | DSP1 | ASIC4 | MEM1 | CPU2 |
| 2 | CPU1 | ASIC2 | IP3 | IP1 |
| 3 | MEM2 | DSP3 | IP2 | IP4 |

Execution time 42.4 ms
Energy: 17.8 mJ

Experiments

Efficiency

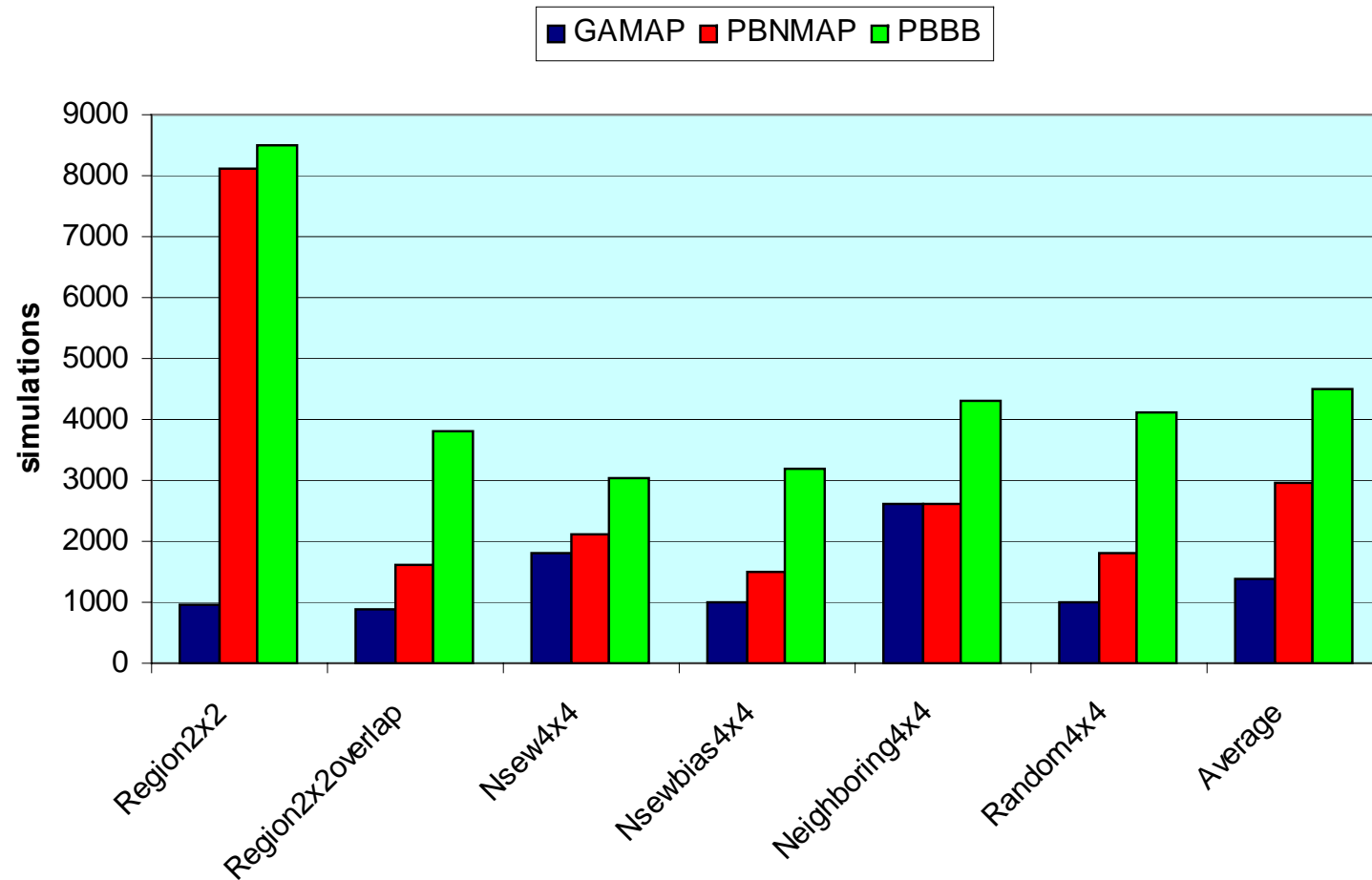


GAMAP 840 sims
PBBB 7238 sims
PBNMAP 2670 sims

$Speedup_{PBBB} = 8.6$
 $Speedup_{PBNMAP} = 3.2$

Experiments

Efficiency



Routing

Routing

■ Deterministic

→ Path completely determined by the source and the destination address

■ Adaptive

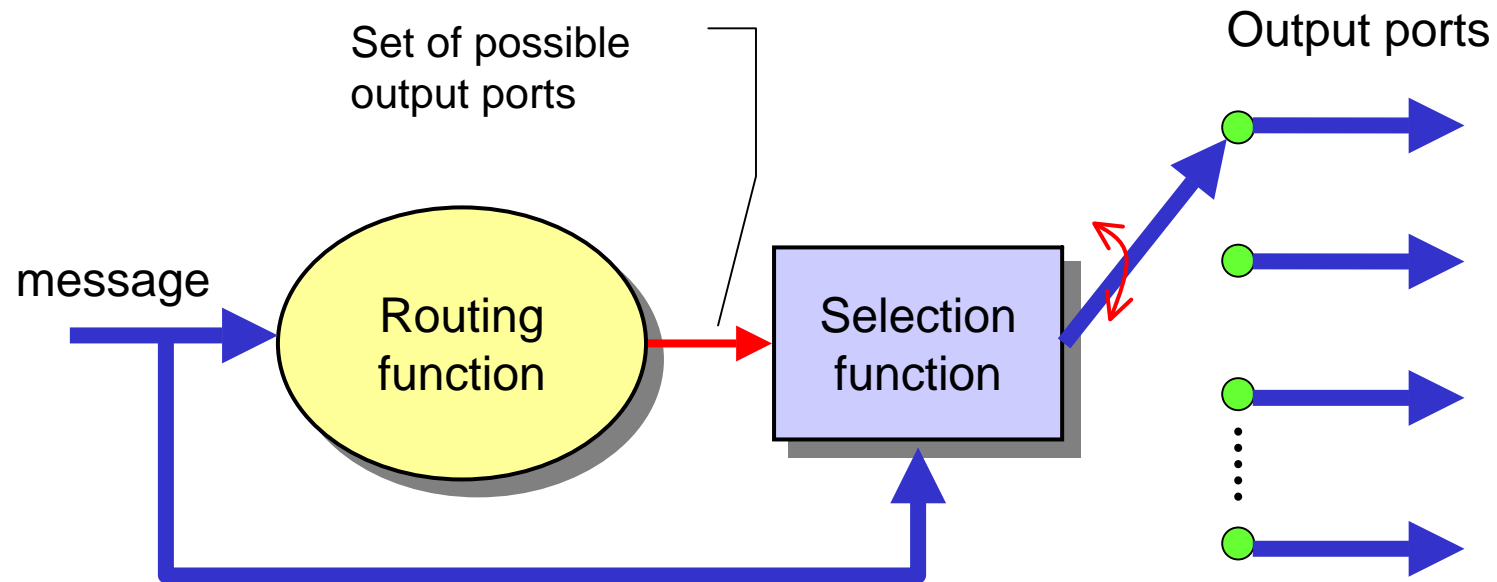
→ The path taken by a particular packet depends on dynamic network conditions

- ✓ Fault,

- ✓ Congestion, ...

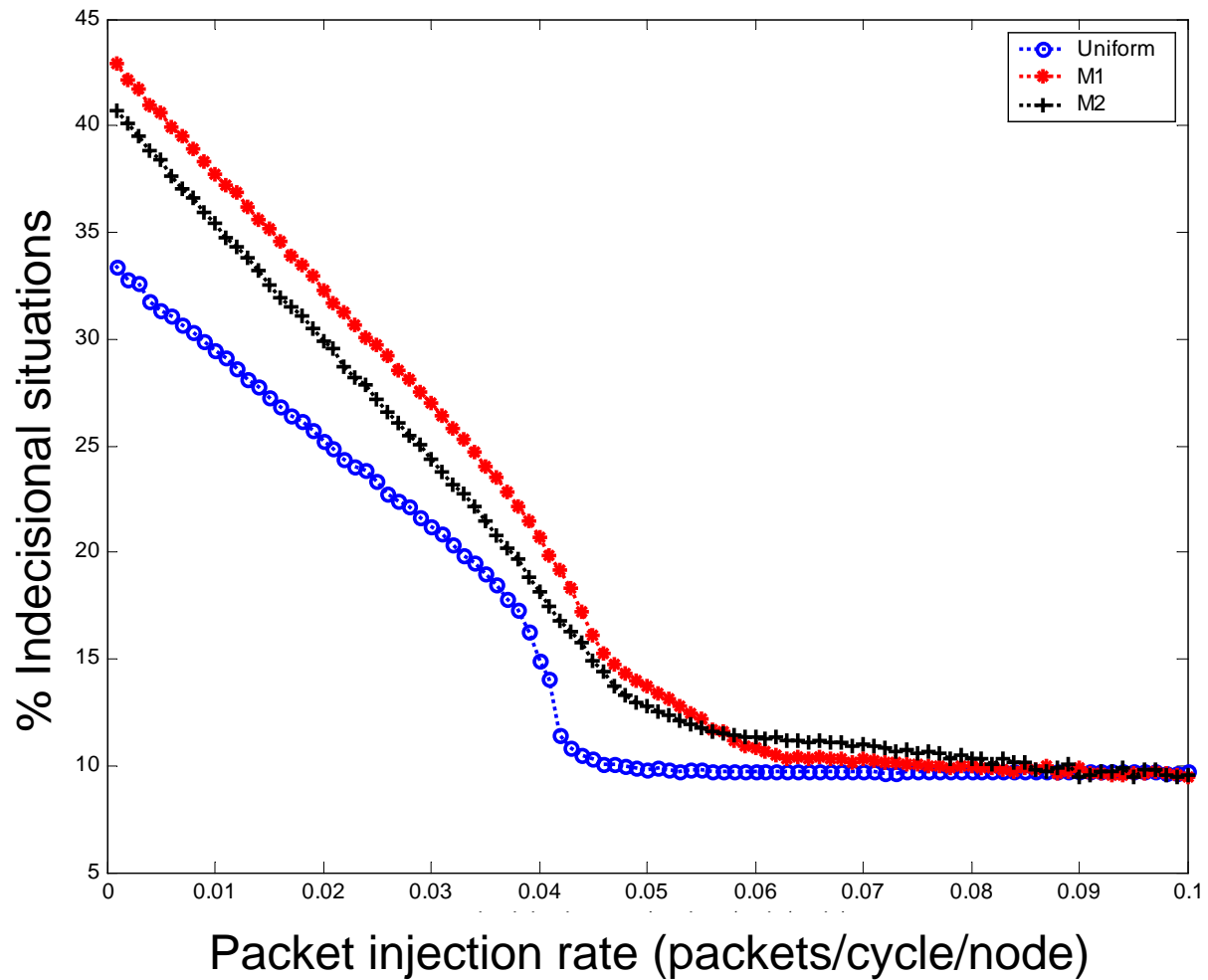
→ Routing packets along alternative paths in order to avoid congested areas

Routing and Selection

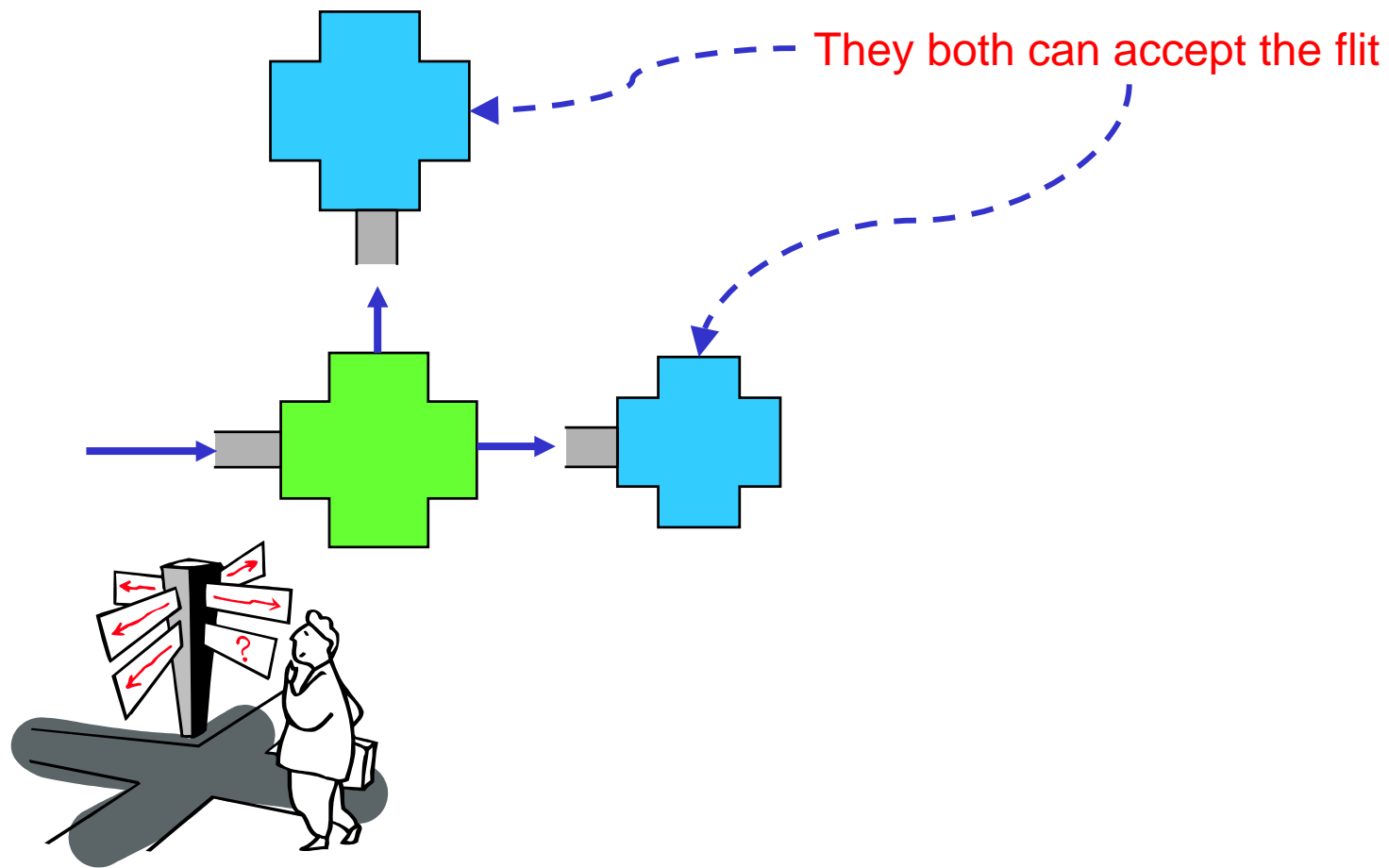


Neighbors on Path Congestion-Aware Routing

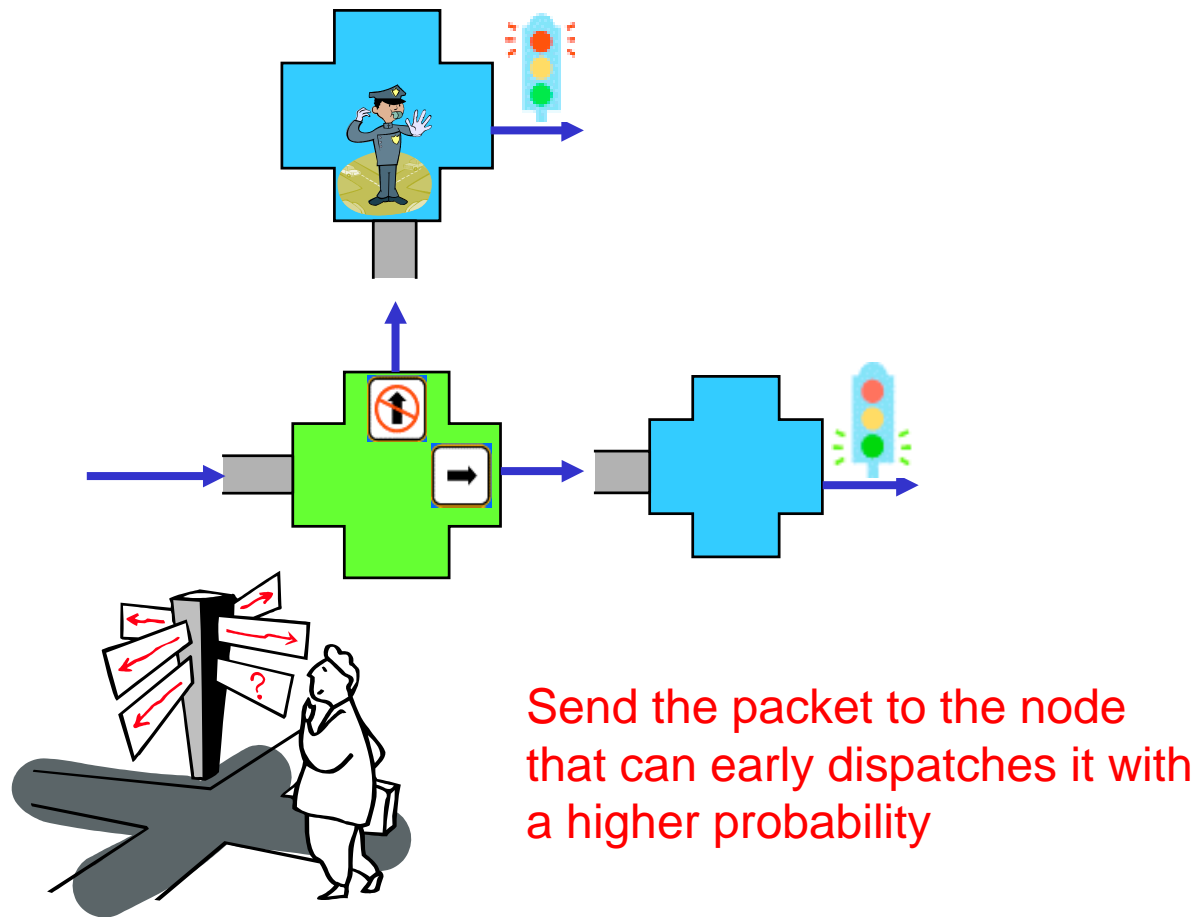
Situations of Indecision



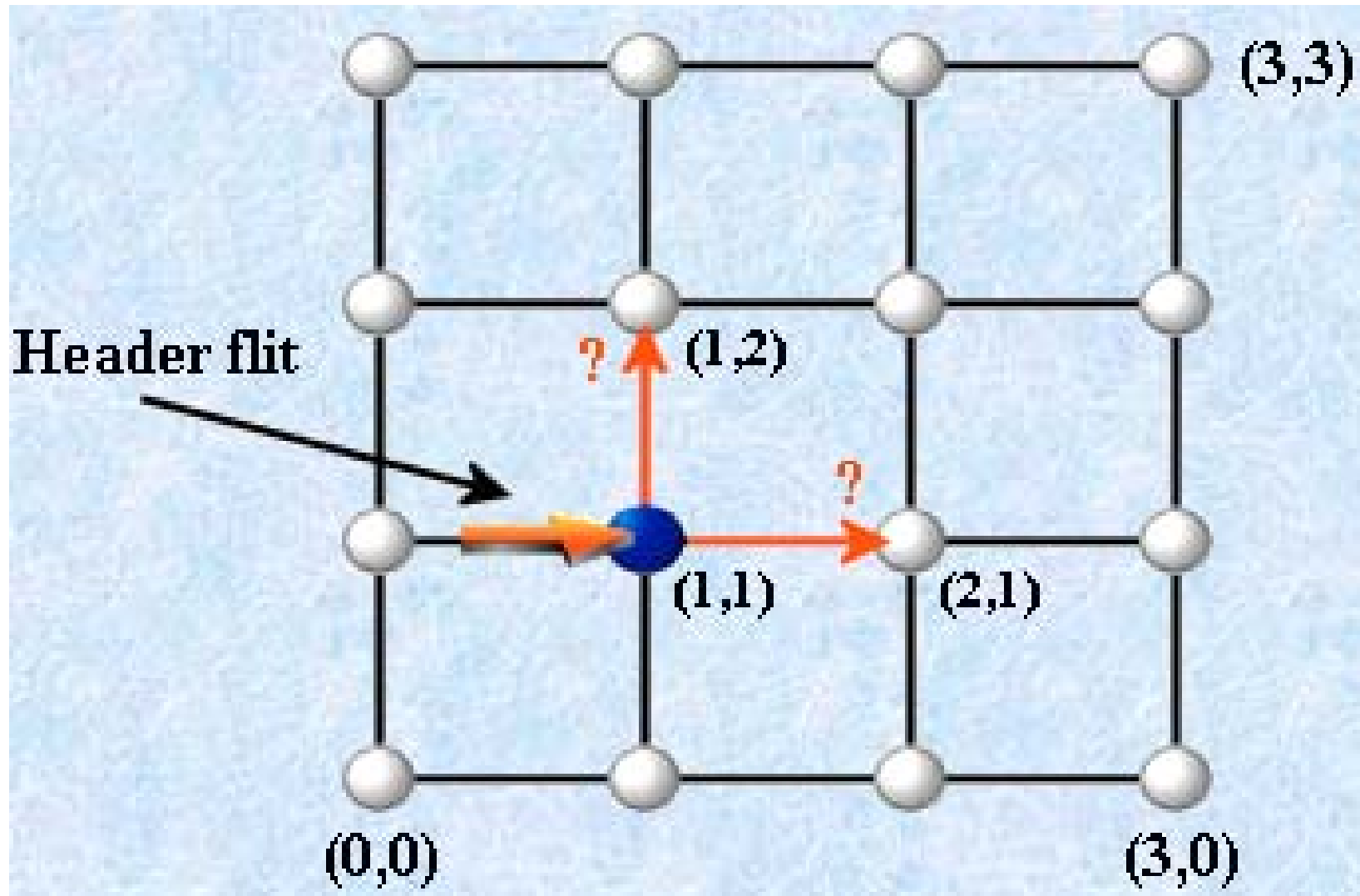
Selection Problem



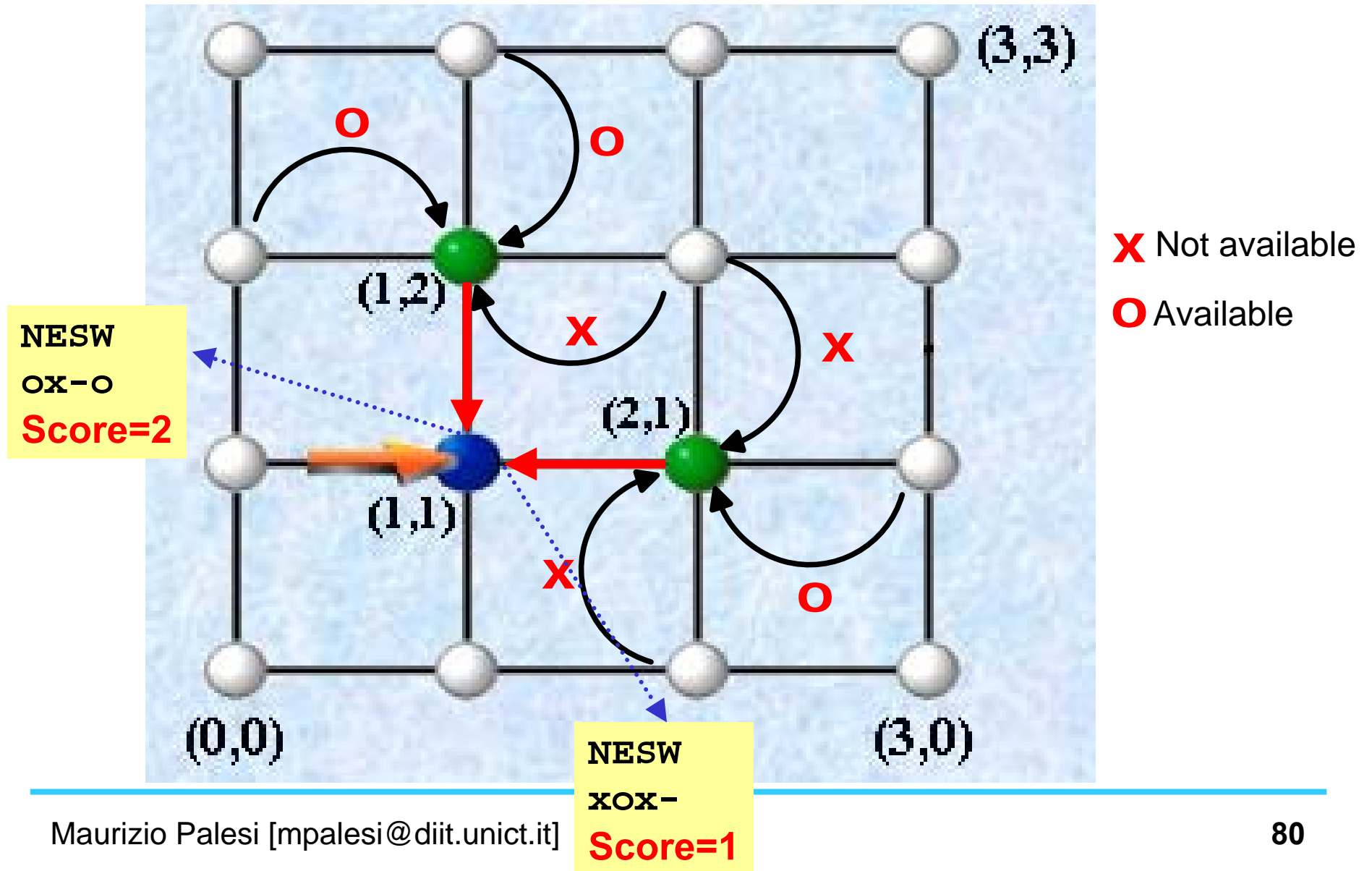
Selection Problem



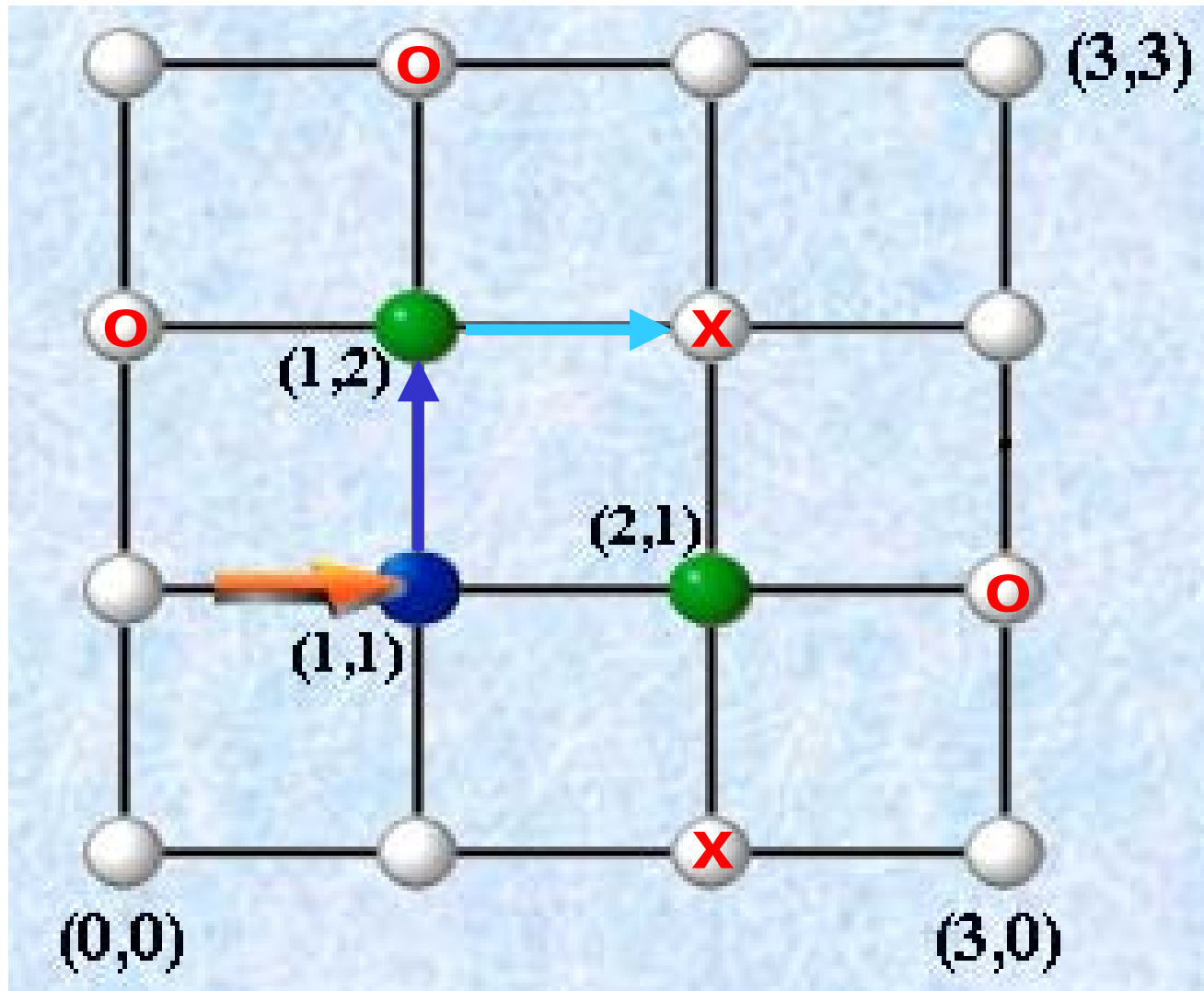
NoPCAR Algorithm



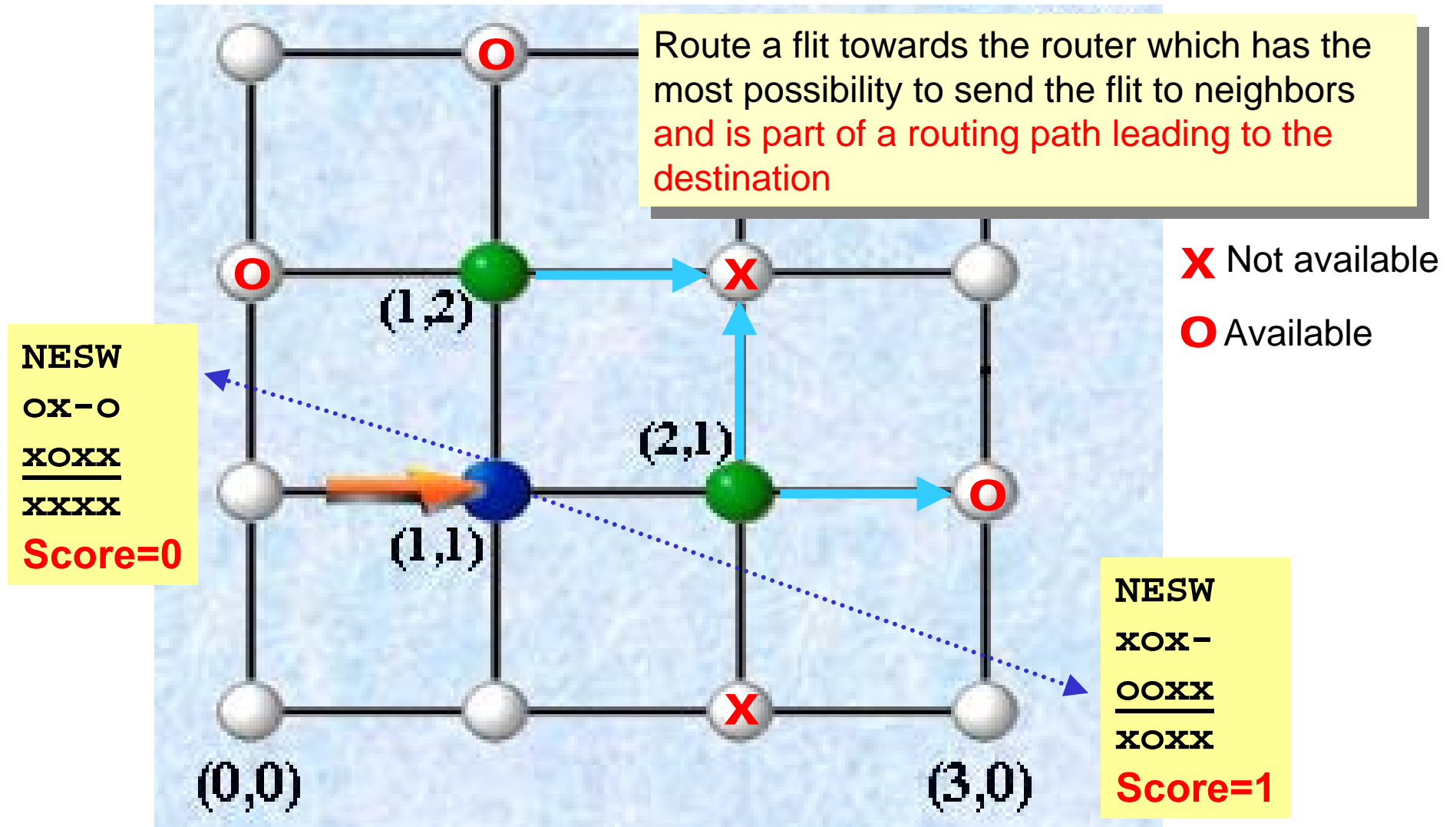
NoPCAR Algorithm



NoPCAR Algorithm



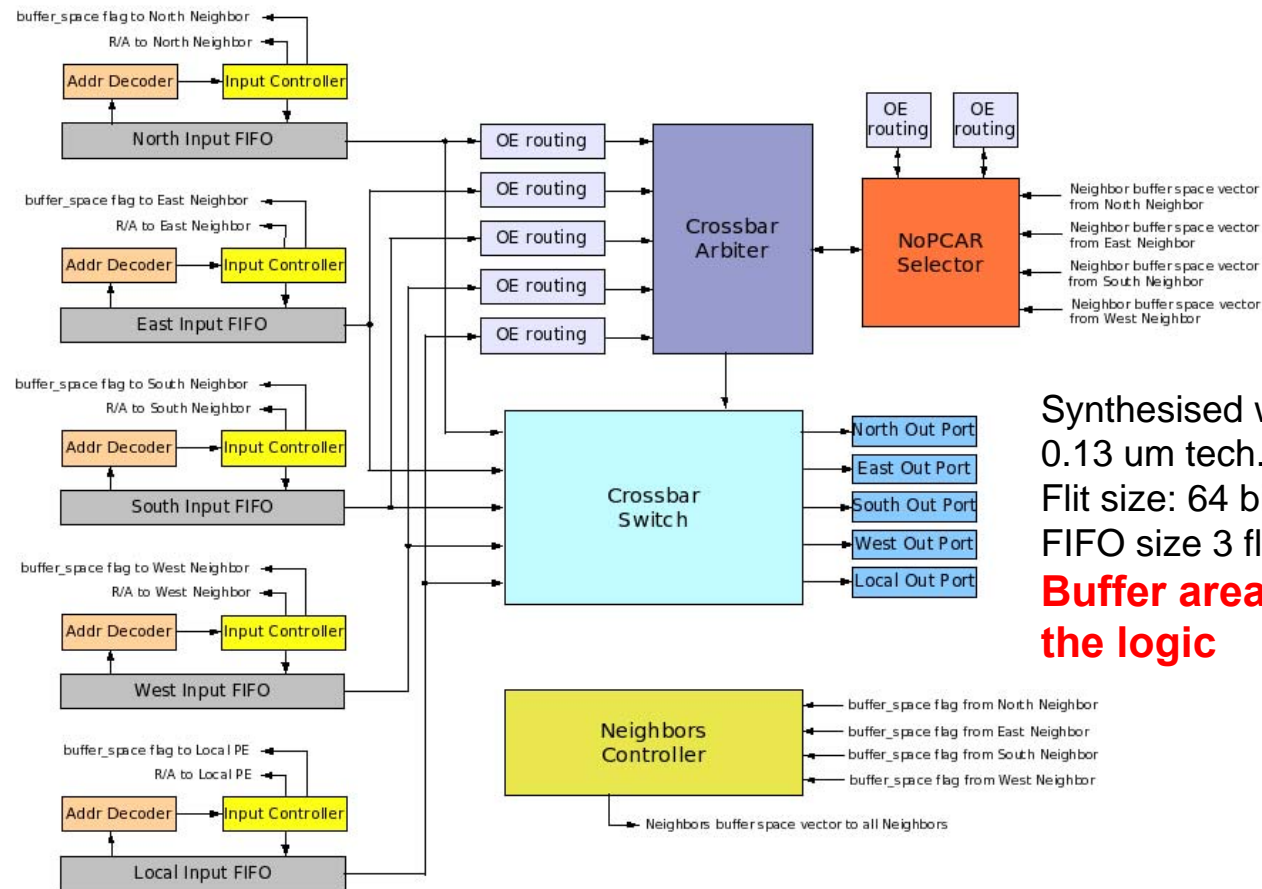
NoPCAR Algorithm



NoPCAR algorithm

```
NoPCAR_Selection(in AC: set of admissible output channels,  
                nd: destination node,  
                out SC: selected output channel)  
{  
    credits[] = 0;  
    foreach channel1 in AC {  
        node1 = target(channel1);  
        foreach channel2 in R(node1, nd) {  
            node2 = target(channel2);  
            if bsv[node1][node2] is available then  
                credits[channel1]++;  
        }  
    }  
    sc = ch s.t. credits[ch] = max(credits[]);  
}
```

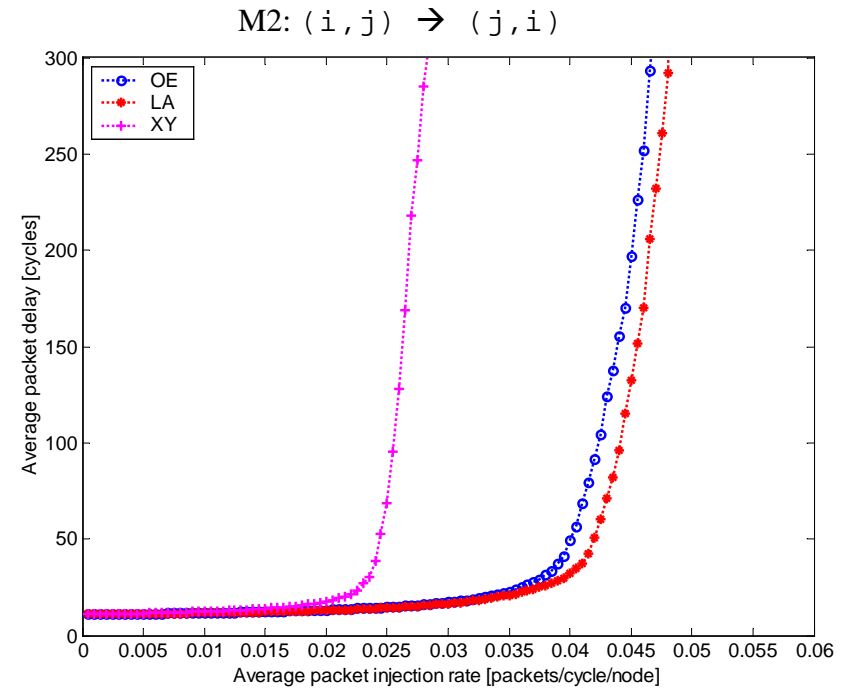
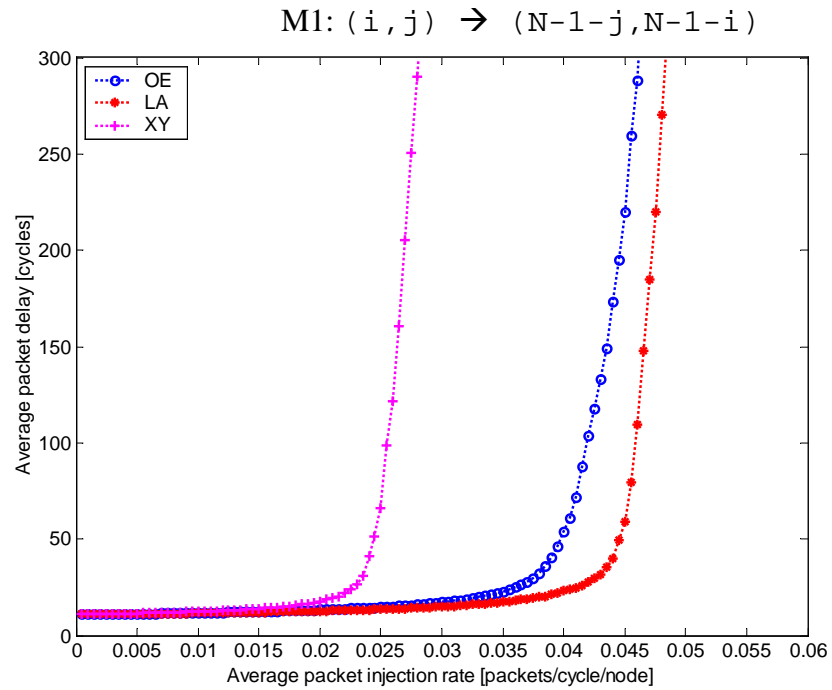
Block Diagram of the Router



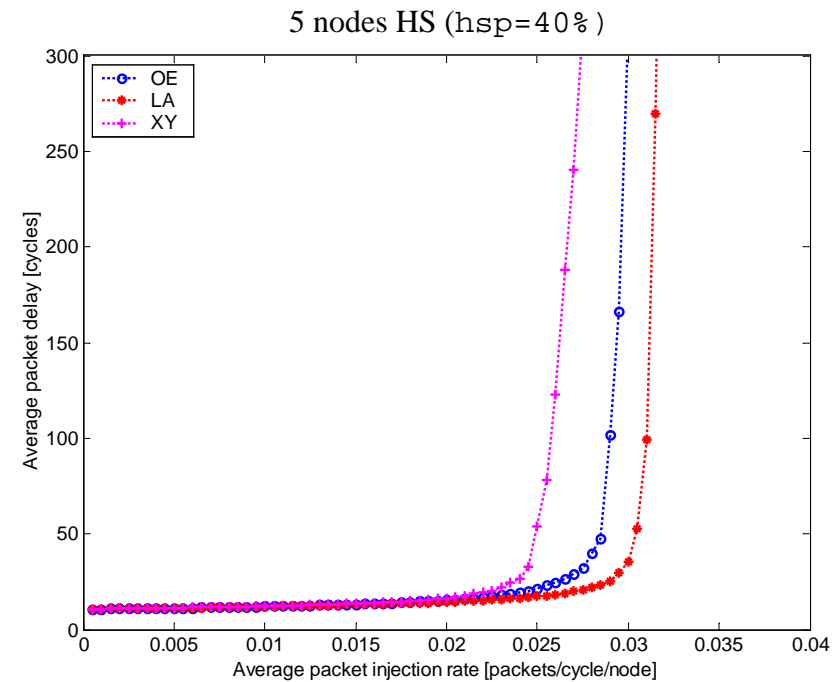
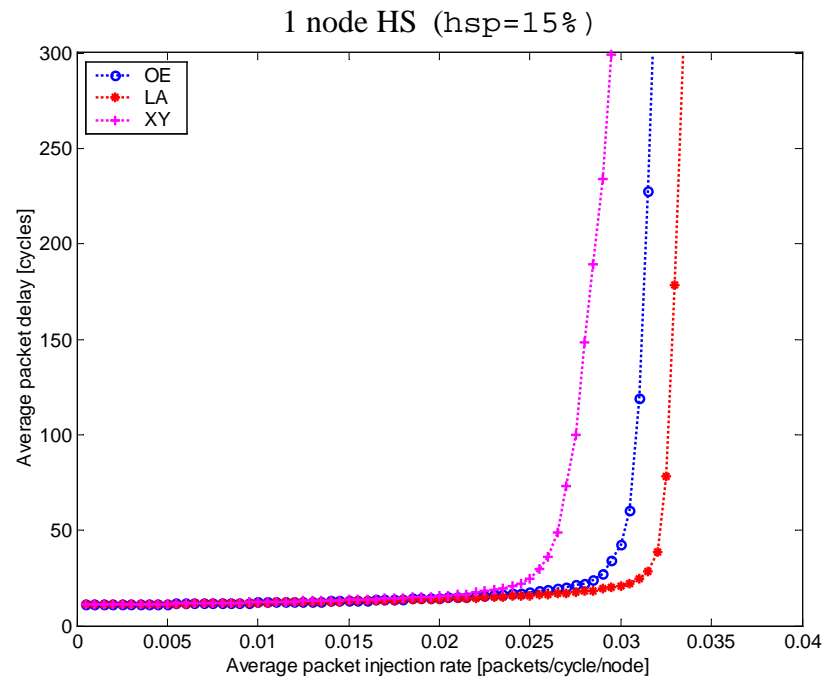
Synthesised with Synopsys Design Compiler
 0.13 um tech. Library from Virtual Silicon
 Flit size: 64 bit
 FIFO size 3 flits
**Buffer area significantly dominates
 the logic**

Area overhead due to NoPCAR is less then 6%

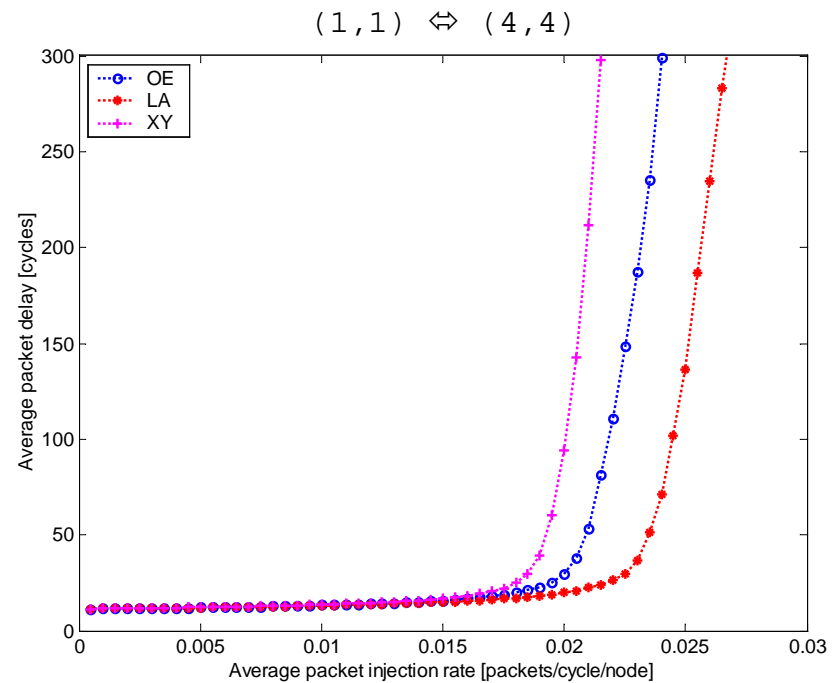
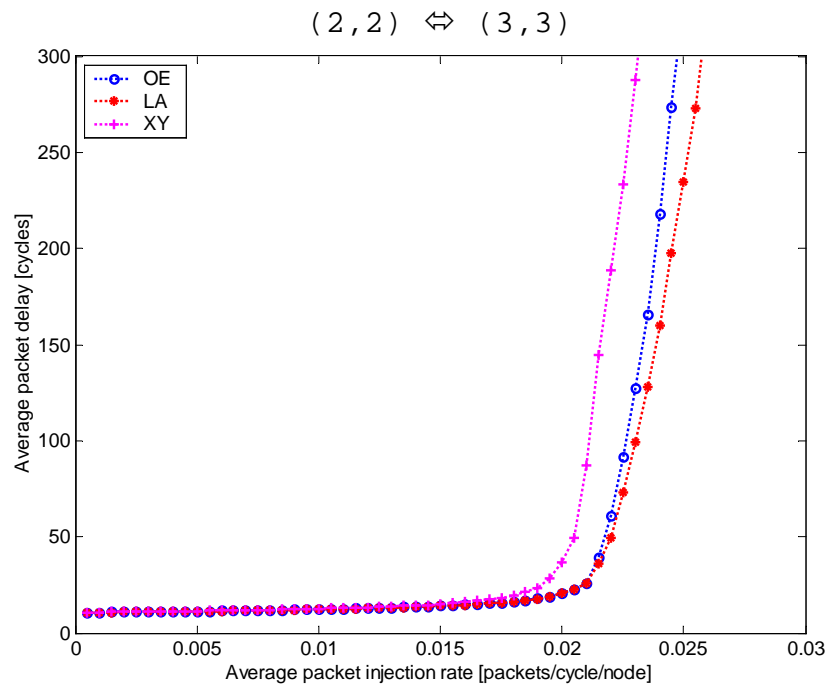
M1/M2 Traffic



Hot-Spot Traffic (1/2)



Hot-Spot Traffic (2/2)



Outline

- Design Space Exploration of Parameterized Systems
- Network on Chip Architectures
- **Instruction Level Power Estimation**

Why Low Power Design?

■ Mobile applications

- Large & heavy devices
- Battery life

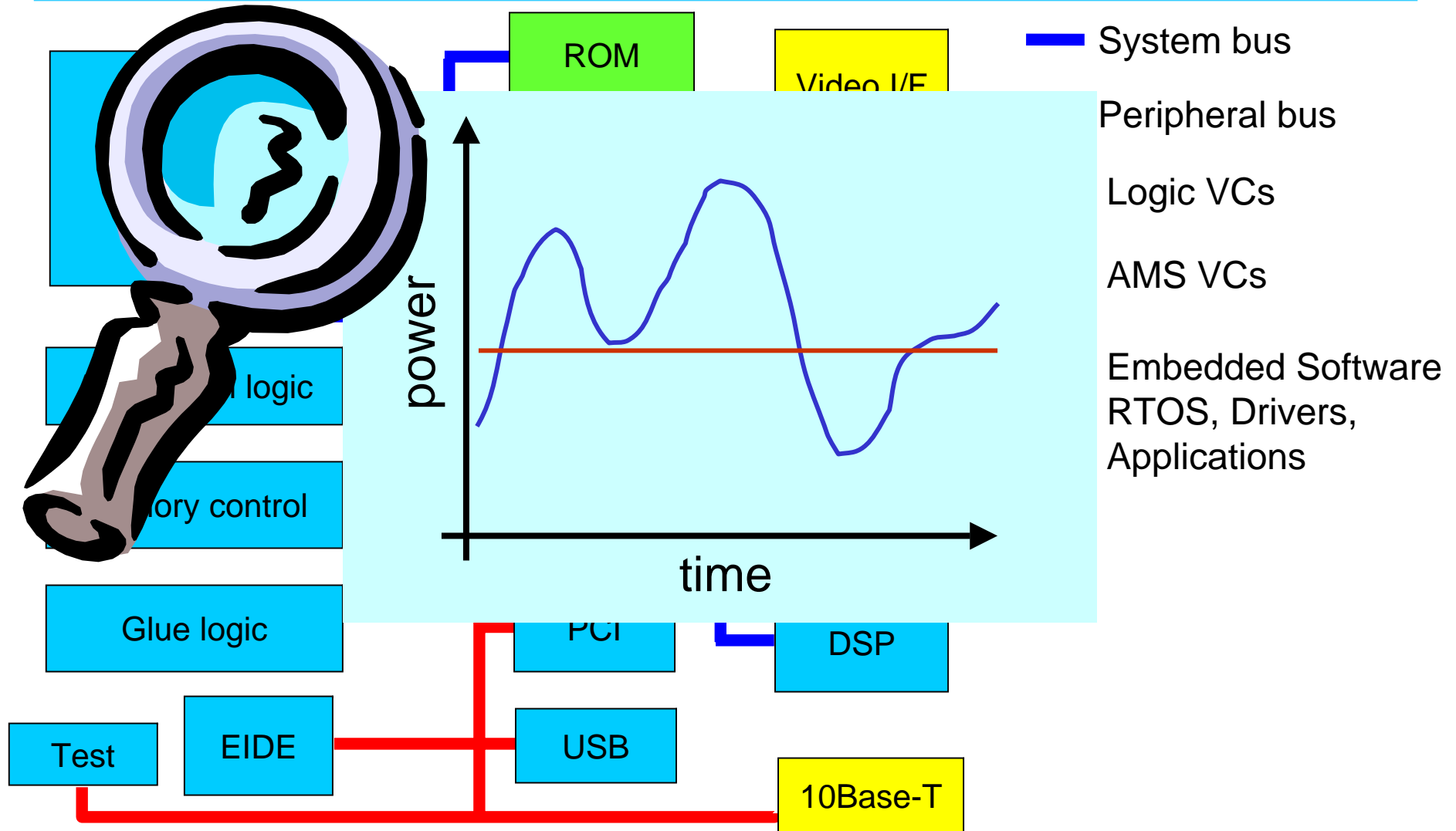
■ High-performance devices

- Heat dissipation problems
- Expensive packaging

■ Reliability

- Fault probability doubling for every 10°C increase in temperature

SoC Architecture: an example



The Average Cost Model

"By measuring the power absorbed by a processor X repeating certain instructions or short sequences of instructions it is possible to obtain much of the information needed to calculate the power dissipated when the processor X executes the instructions of any program Y ".

Tiwari *et. al* '94

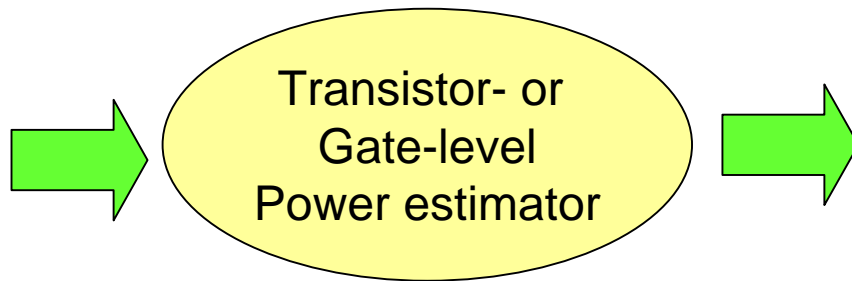
Case Study

- STMicroelectronics ST20 C2P
 - Stack based architecture
 - Strong data dependency
- STMicroelectronics LX
 - VLIW architecture

Work supported by CNR (Project MADESS II) and STMicroelectronics

The Average Cost Model

```
add r1, r2, r3
add r1, r2, r3
add r1, r2, r3
add r1, r2, r3
add r1, r2, r3
add r1, r2, r3
add r1, r2, r3
... ..
add r1, r2, r3
add r1, r2, r3
add r1, r2, r3
add r1, r2, r3
```



| Instruction | Cost |
|--------------------|-------------|
| add | 1.5 |
| sub | 1.9 |
| lw | 4.2 |
| sw | 7.9 |
| ... | |

Inter-instruction Effects

| Instr. | Cost | Incr. |
|--------|---------|-------|
| Ldnlp | 25.885 | |
| Stl | 16.5719 | 85% |
| Stl | 9.6162 | 0% |
| Stl | 58.223 | 353% |
| Add | 15.3052 | |

Code Optimization

```
...  
ldl    0  
ldl    1  
add  
ldl    2  
add  
ldl    3  
add  
ldl    4  
  
...  
ldl    28  
add  
ldl    29
```

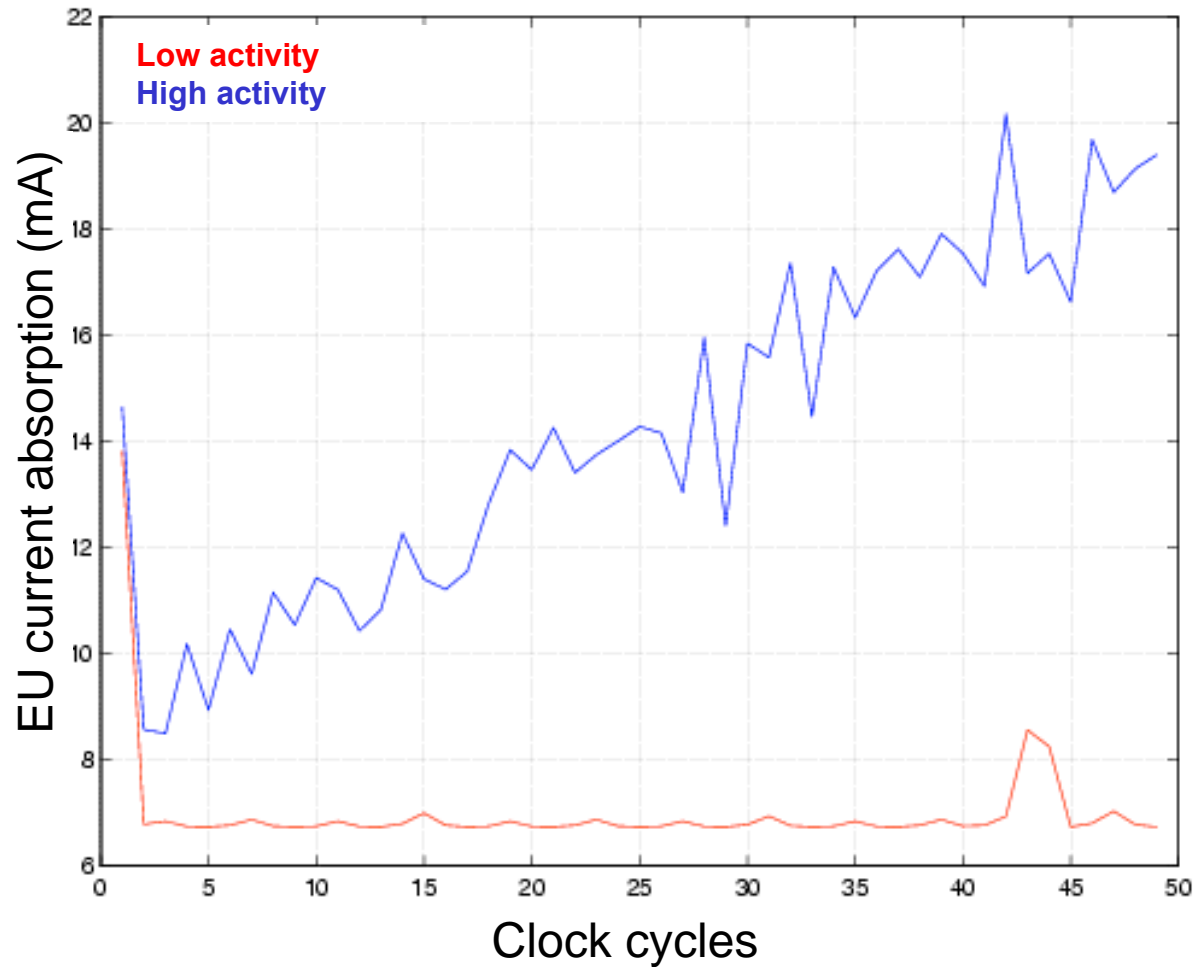
13 μ J

```
...  
ldl    0  
ldl    1  
ldl    2  
add  
add  
ldl    3  
ldl    4  
add  
  
...  
ldl    28  
ldl    29  
add
```

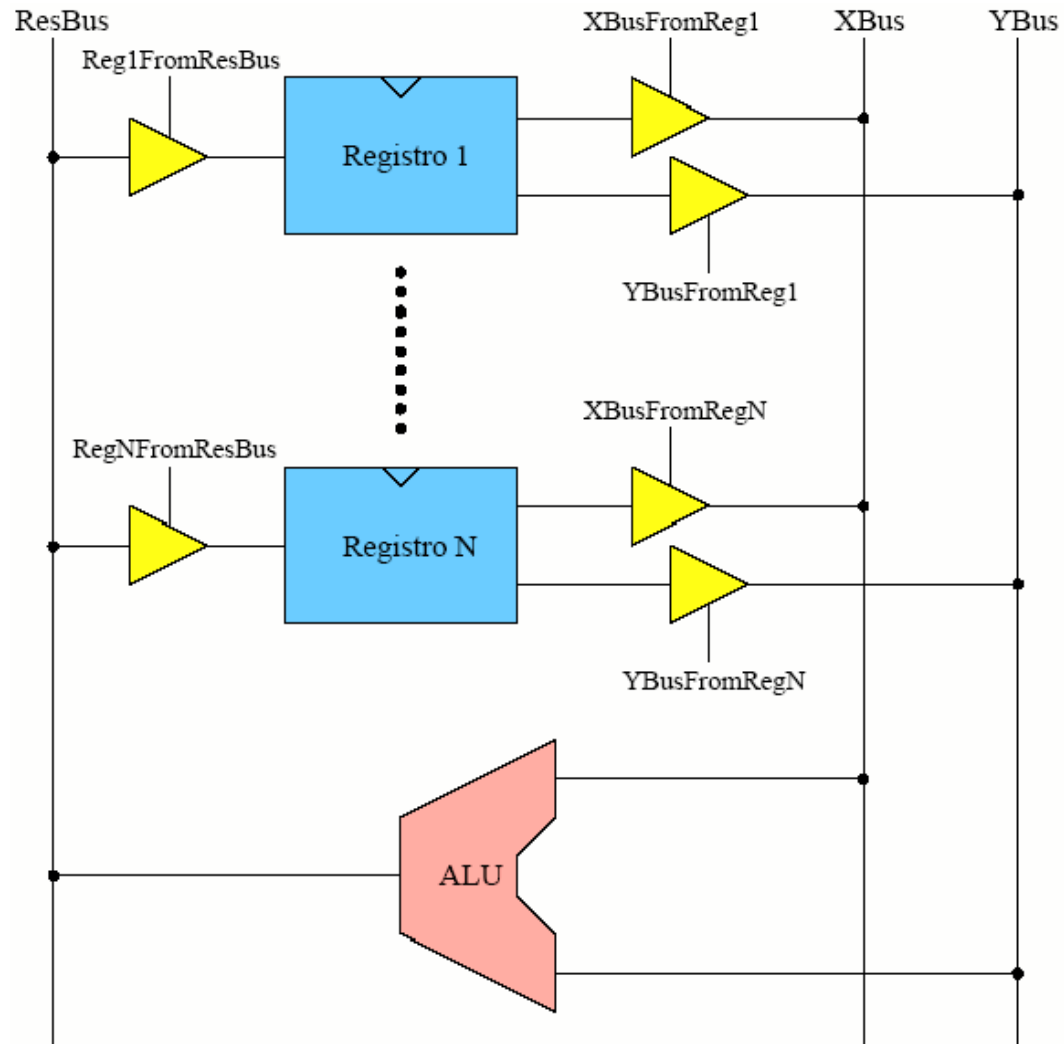
7.3 μ J

44% saving!

Data Dependency

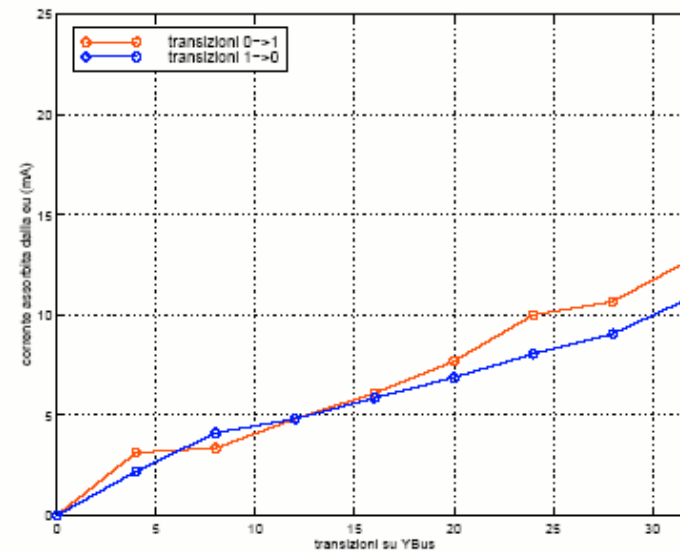
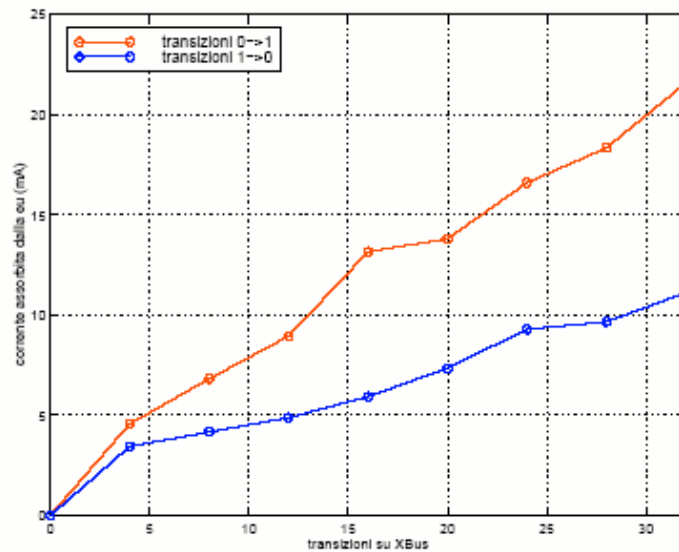


Data Dependency



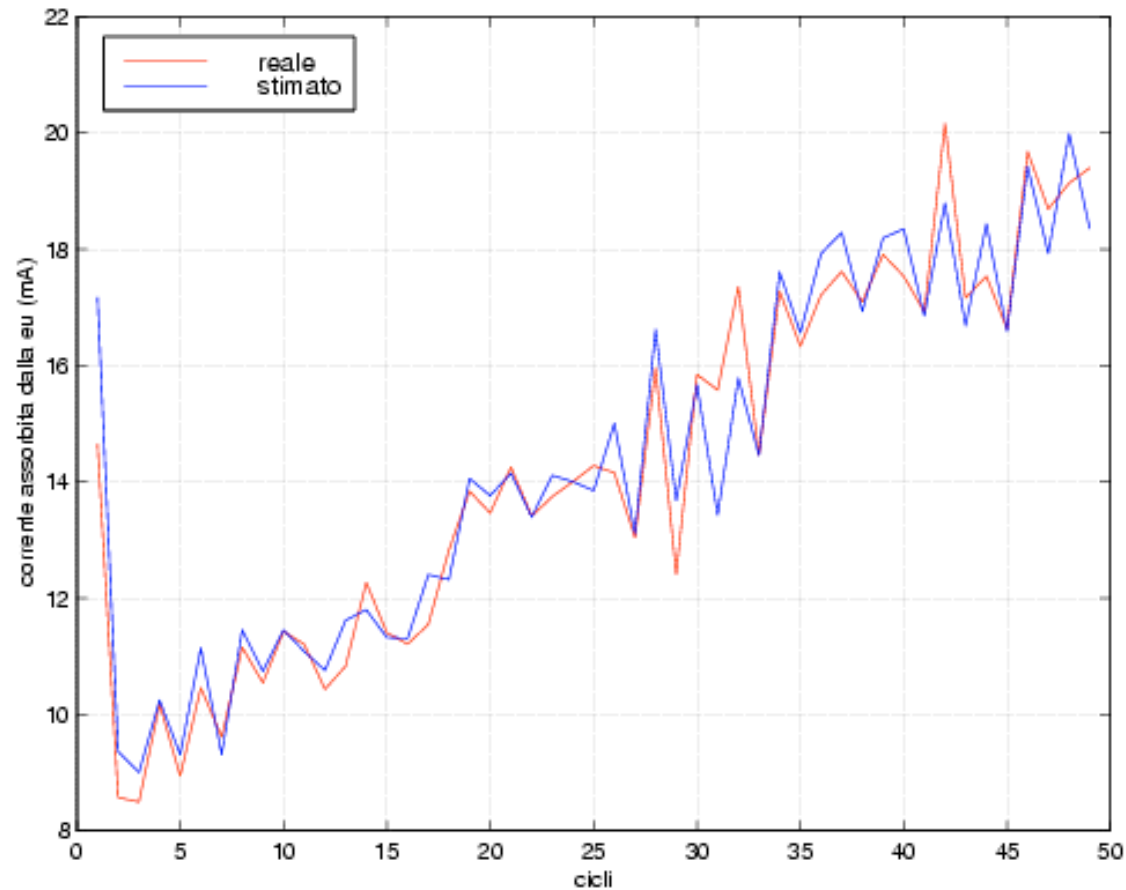
Data Dependency

$$\begin{aligned} f(\text{data}) = & \text{Hamming}^{0 \rightarrow 1}(x\text{bus}_{i-1}, x\text{bus}_i) \times \text{weight}_{x\text{bus}}^{0 \rightarrow 1} + \\ & + \text{Hamming}^{1 \rightarrow 0}(x\text{bus}_{i-1}, x\text{bus}_i) \times \text{weight}_{x\text{bus}}^{1 \rightarrow 0} + \\ & + \text{Hamming}^{0 \rightarrow 1}(y\text{bus}_{i-1}, y\text{bus}_i) \times \text{weight}_{y\text{bus}}^{0 \rightarrow 1} + \\ & + \text{Hamming}^{1 \rightarrow 0}(y\text{bus}_{i-1}, y\text{bus}_i) \times \text{weight}_{y\text{bus}}^{1 \rightarrow 0} + \\ & + \text{Hamming}(\text{memaddr}_{i-1}, \text{memaddr}_i) \times \text{weight}_{\text{memaddr}} + \dots \end{aligned}$$

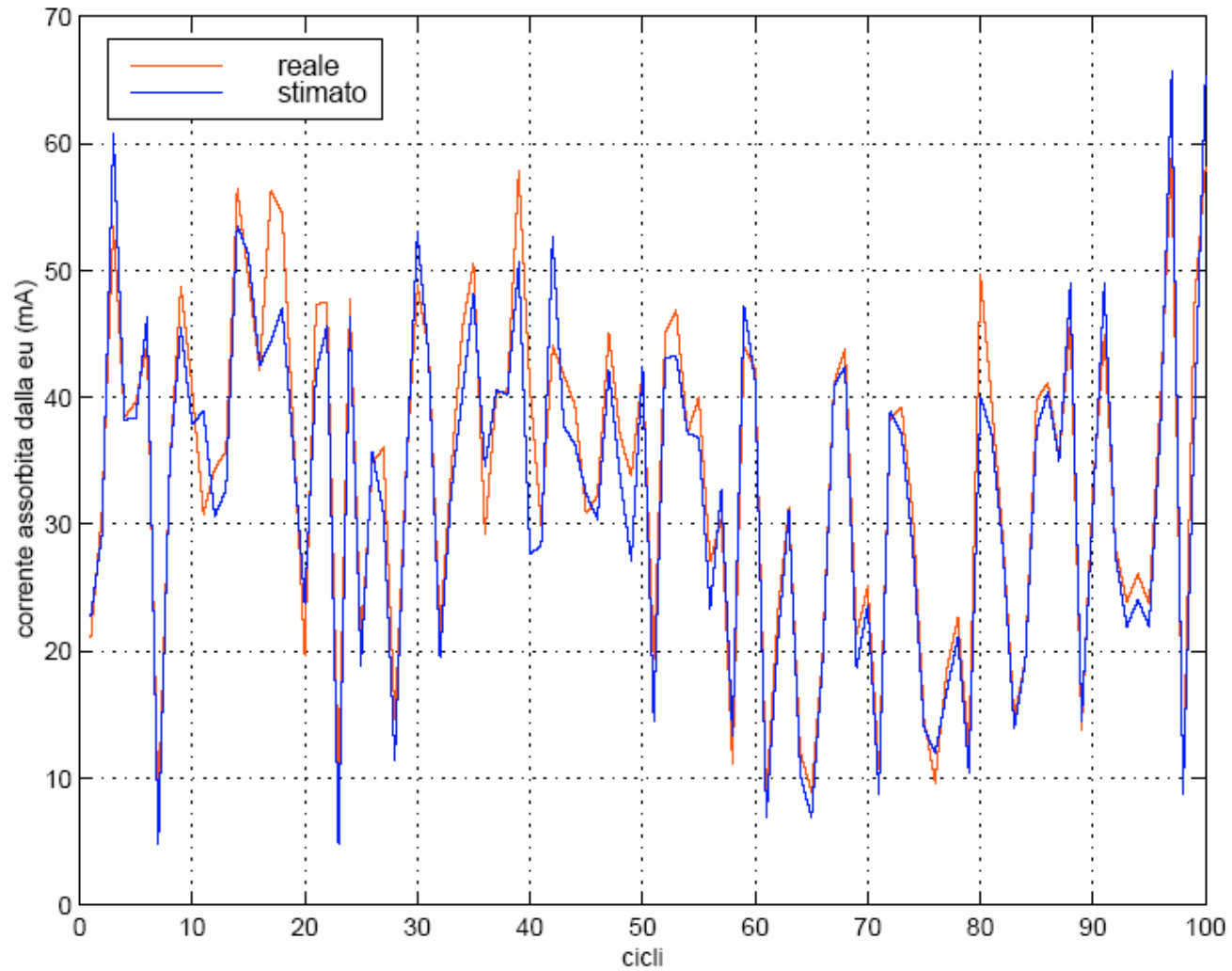


Data Dependent Model

$$E(I_N) = \text{base}(I_N) + \text{Int}(I_N, I_{N-1}) + \text{Data}(I_N)$$



FFT



Experiments

| Benchmark | Real (mA) | ACM (mA) | DDM (mA) | Relative Error | | Absolute Error | |
|-----------|--------------|-------------|-------------|----------------|--------------|----------------|--------------|
| | | | | ACM | DDM | ACM | DDM |
| sum_low | 6,79 | 13,44 | 6,8 | 97,77% | 0,12% | 97,79% | 0,86% |
| sum_high | 14,24 | 13,44 | 14,31 | 5,58% | 0,50% | 20,37% | 3,47% |
| rand_inst | 40,62 | 48,14 | 40,12 | 18,51% | 1,24% | 31,37% | 6,44% |
| mtx_sum | 31,97 | 37,83 | 30,97 | 18,32% | 3,13% | 31,44% | 6,81% |
| mtx_mul | 36,01 | 41,26 | 34,02 | 7,07% | 5,53% | 27,21% | 8,61% |
| dft | 32,07 | 28,57 | 30,92 | 10,91% | 3,54% | 15,13% | 8,94% |
| Average | | | | 26,36% | 2,34% | 37,22% | 5,86% |

Outline

- Design Space Exploration of Parameterized Systems
- Network on Chip Architectures
- Instruction Level Power Estimation