Università di Catania
Dipartimento di Ingegneria Informatica e delle Telecomunicazioni

Some Key Issues in Embedded System Design

Maurizio Palesi

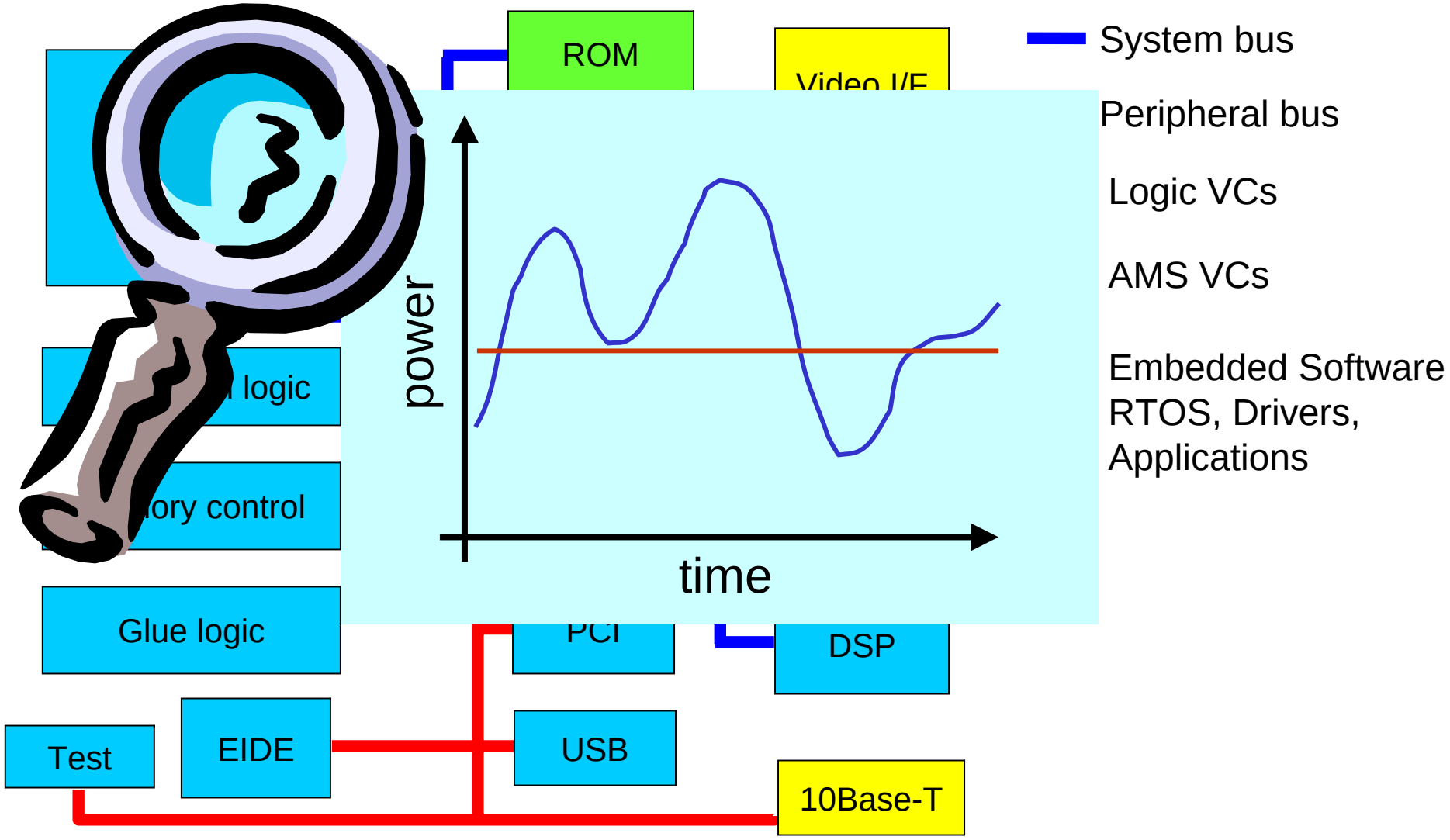
Outline

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

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

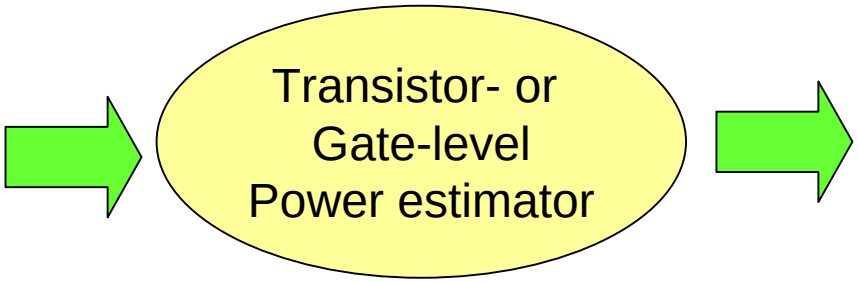
Case Study

- 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
```

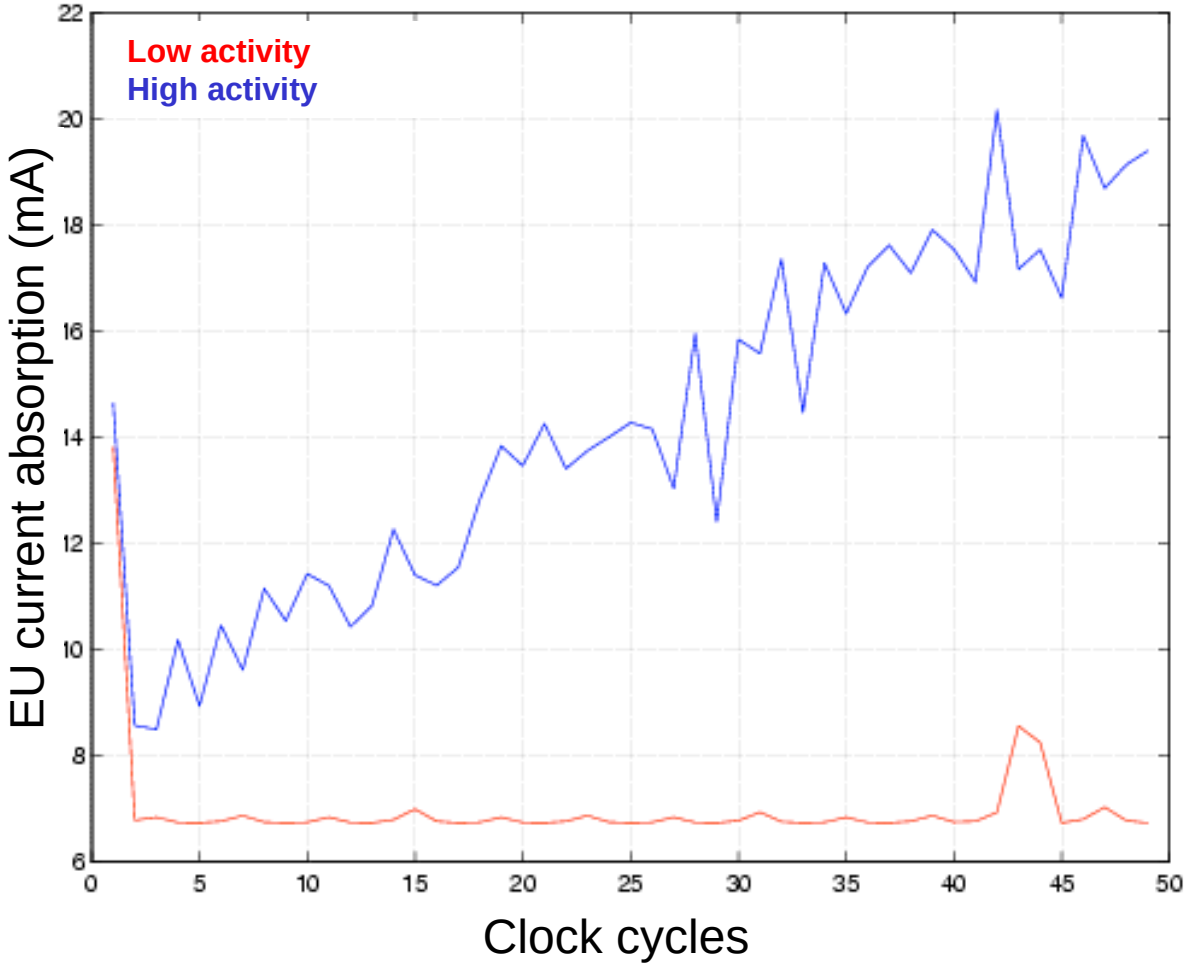
13 μJ

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

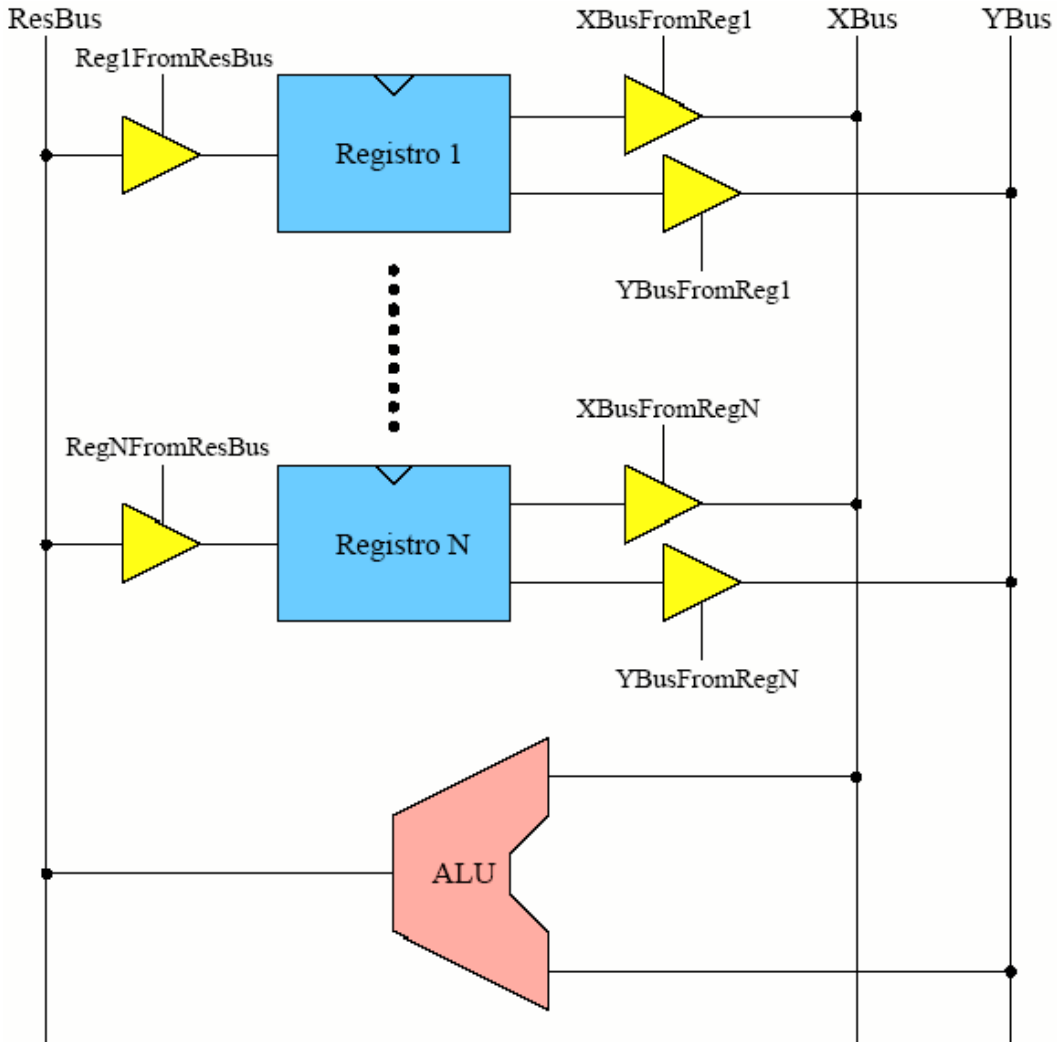
7.3 μJ

44% saving!

Data Dependency

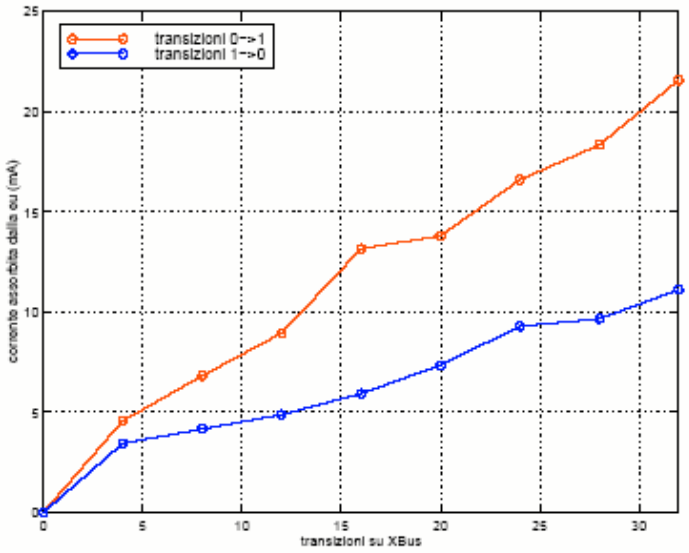


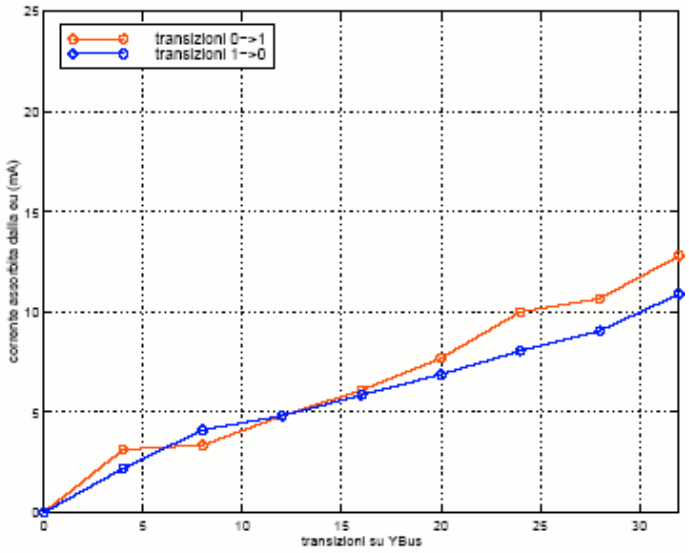
Data Dependency



Data Dependency

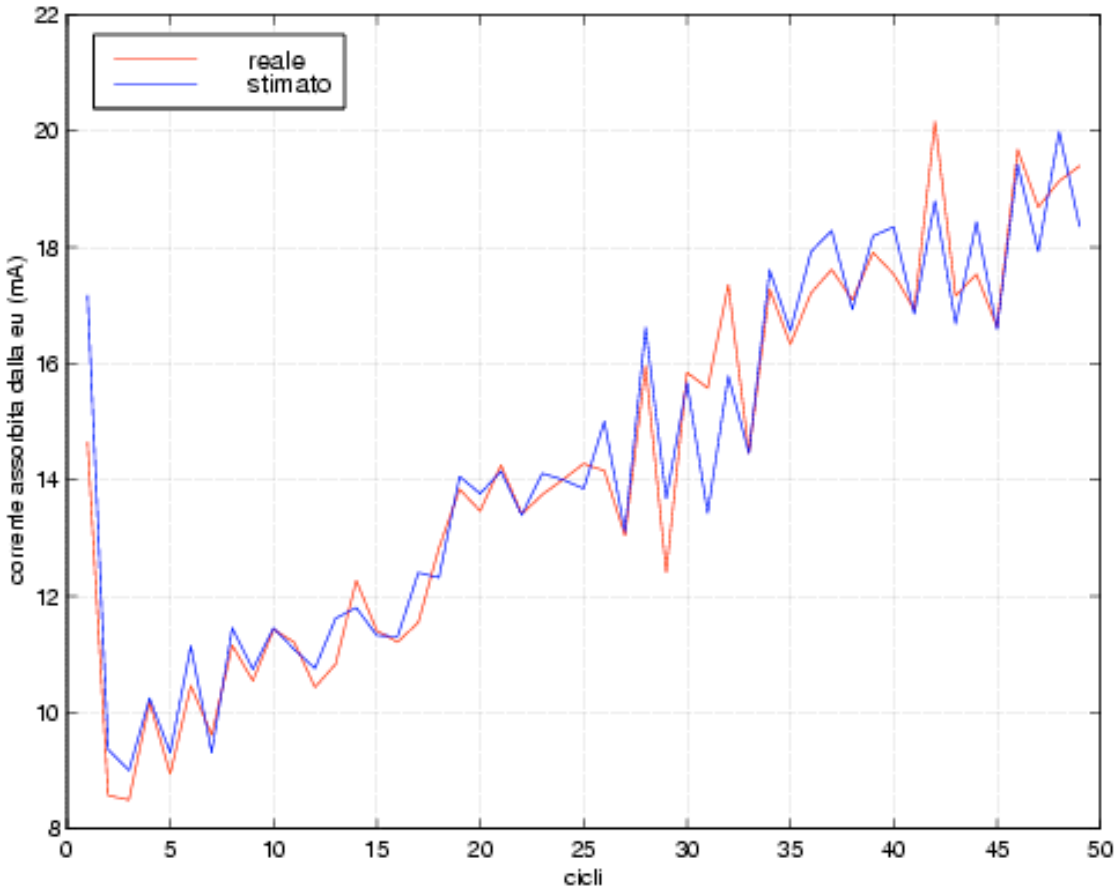
$$f(data) = Hamming^{0\rightarrow 1}(xbus_{i-1}, xbus_i) \times weight^{0\rightarrow 1}_{xbus} +$$
$$+Hamming^{1\rightarrow 0}(xbus_{i\ \ 1}, xbus_i) \times weight^{1\rightarrow 0}_{xbus} +$$
$$+Hamming^{0\rightarrow 1}(ybus_{i\ \ 1}, ybus_i) \times weight^{0\rightarrow 1}_{ybus} +$$
$$+Hamming^{1\rightarrow 0}(ybus_{i-1}, ybus_i) \times weight^{1\rightarrow 0}_{ybus} +$$
$$+Hamming(memaddr_{i-1}, memaddr_i) \times weight_{memaddr} + \ldots$$



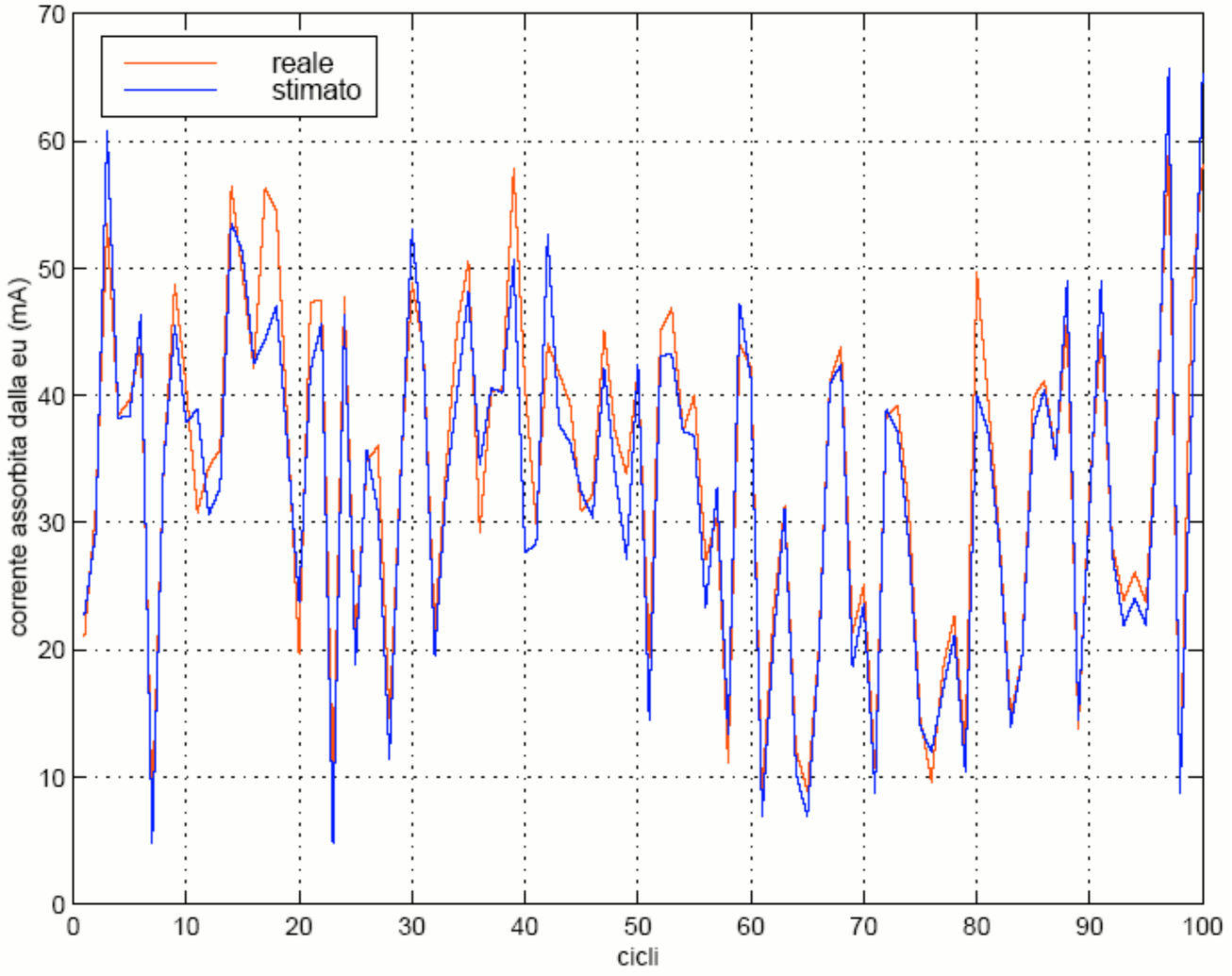


Data Dependent Model

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



FFT


reale
stimato
corrente assorbita dalla eu (mA)
cicli
0
10
20
30
40
50
60
70
80
90
100

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%**

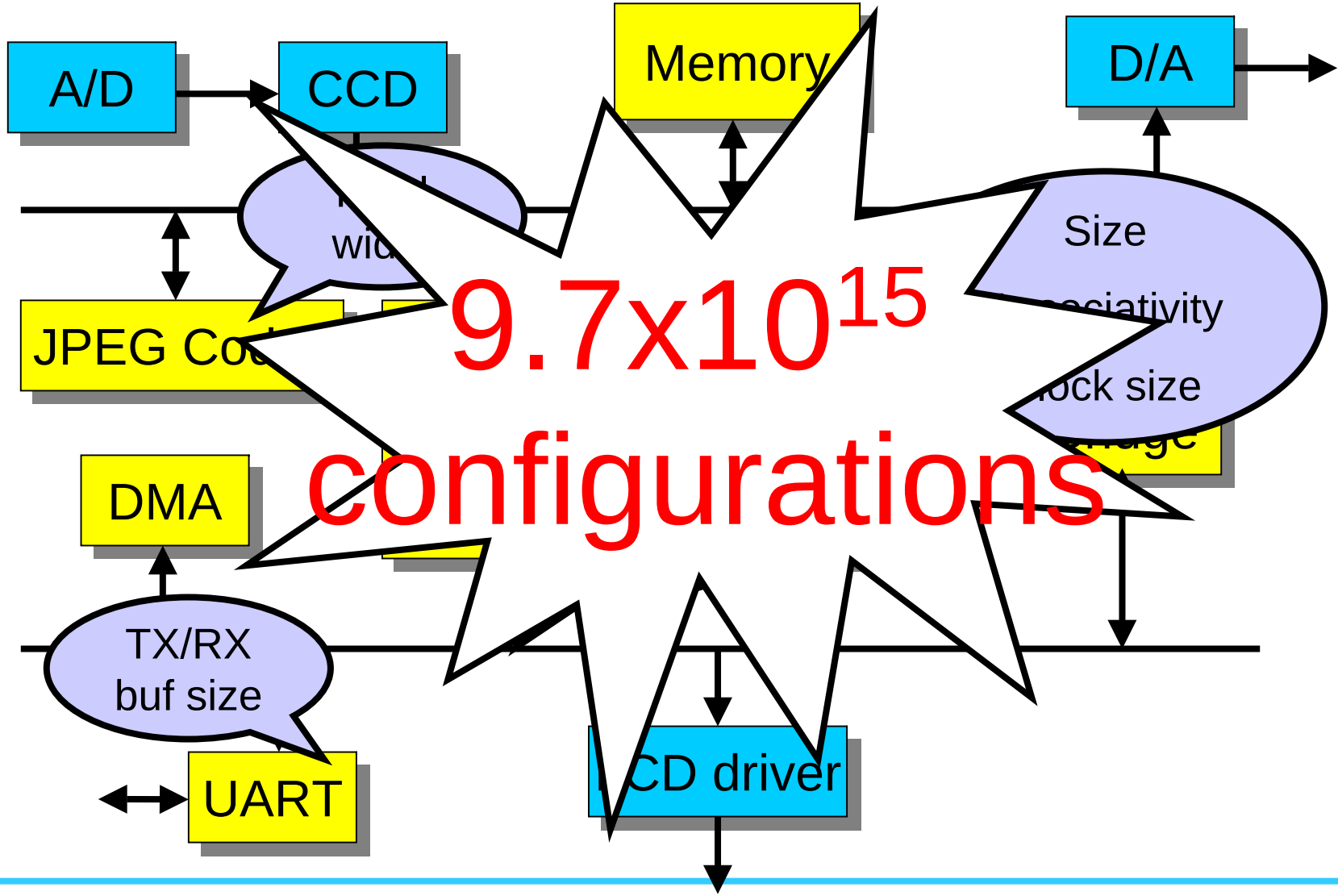
Outline

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

Parameterized Systems

- IP-based design in which a highly parametric pre-designed SoC is configured according to the application it will have to execute
  - ➔ Microprocessors, microcontrollers, DSP processors, coprocessors, memory, FPGA, peripherals, …
  - ➔ Bus parameters, cache parameters, DMA parameters, core-dependent parameters, algorithm options…

Case Study

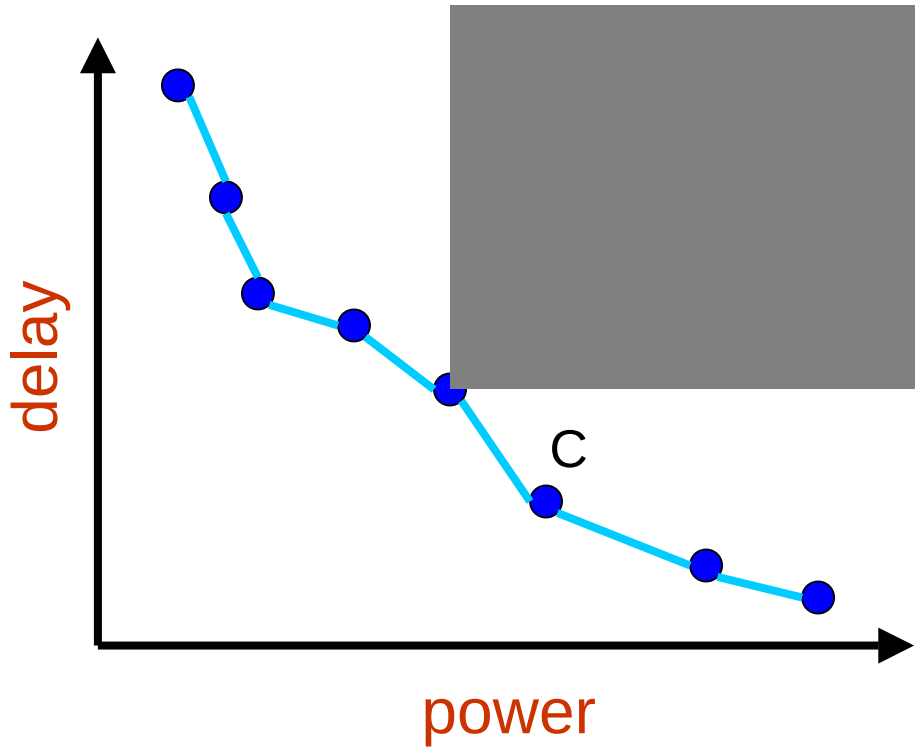

A/D
CCD
Memory
D/A
Size
JPEG Co
DMA
TX/RX
buf size
UART
CD driver
9.7x10^15
configurations

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



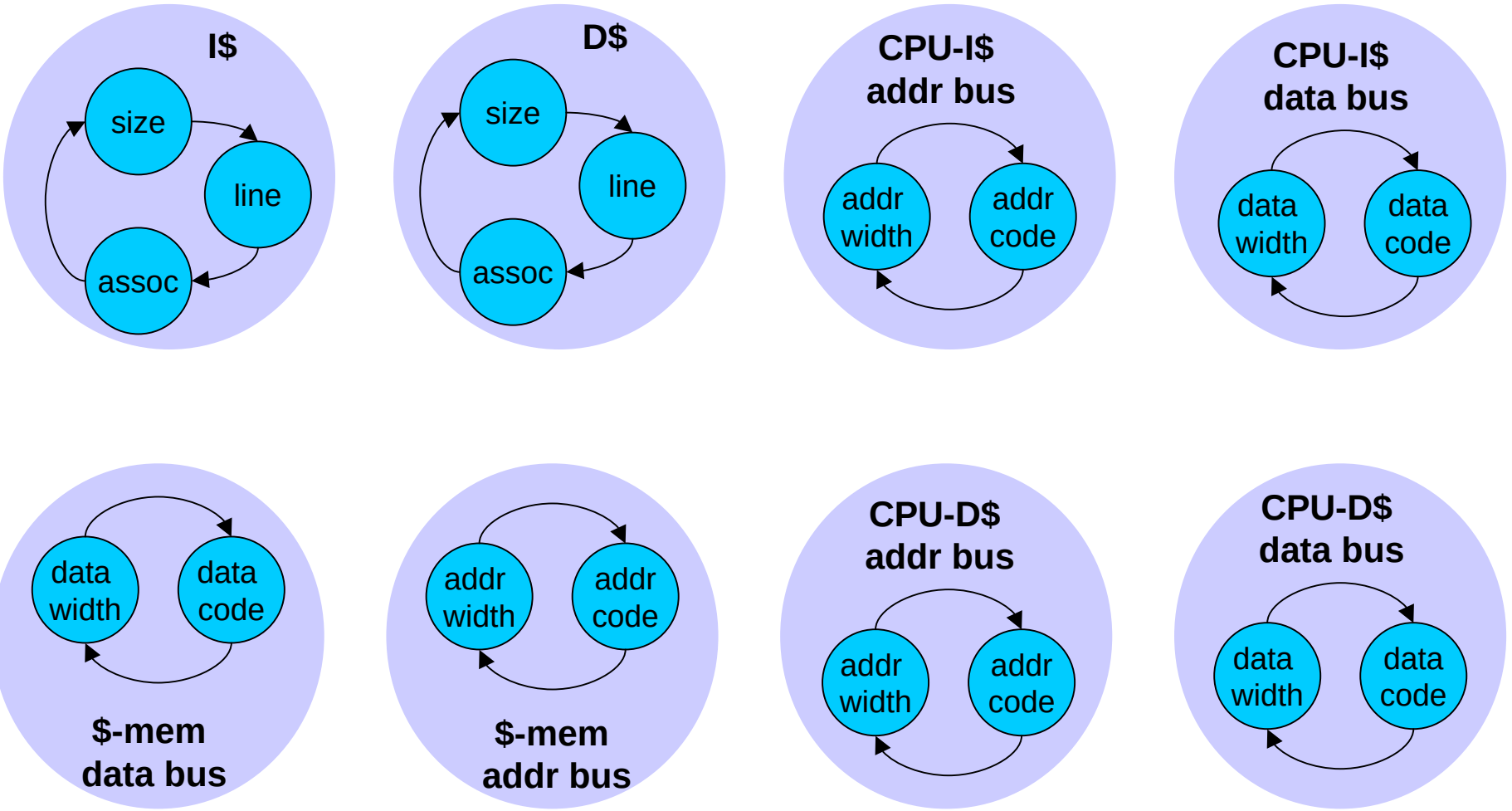
DSE Approaches

- 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]

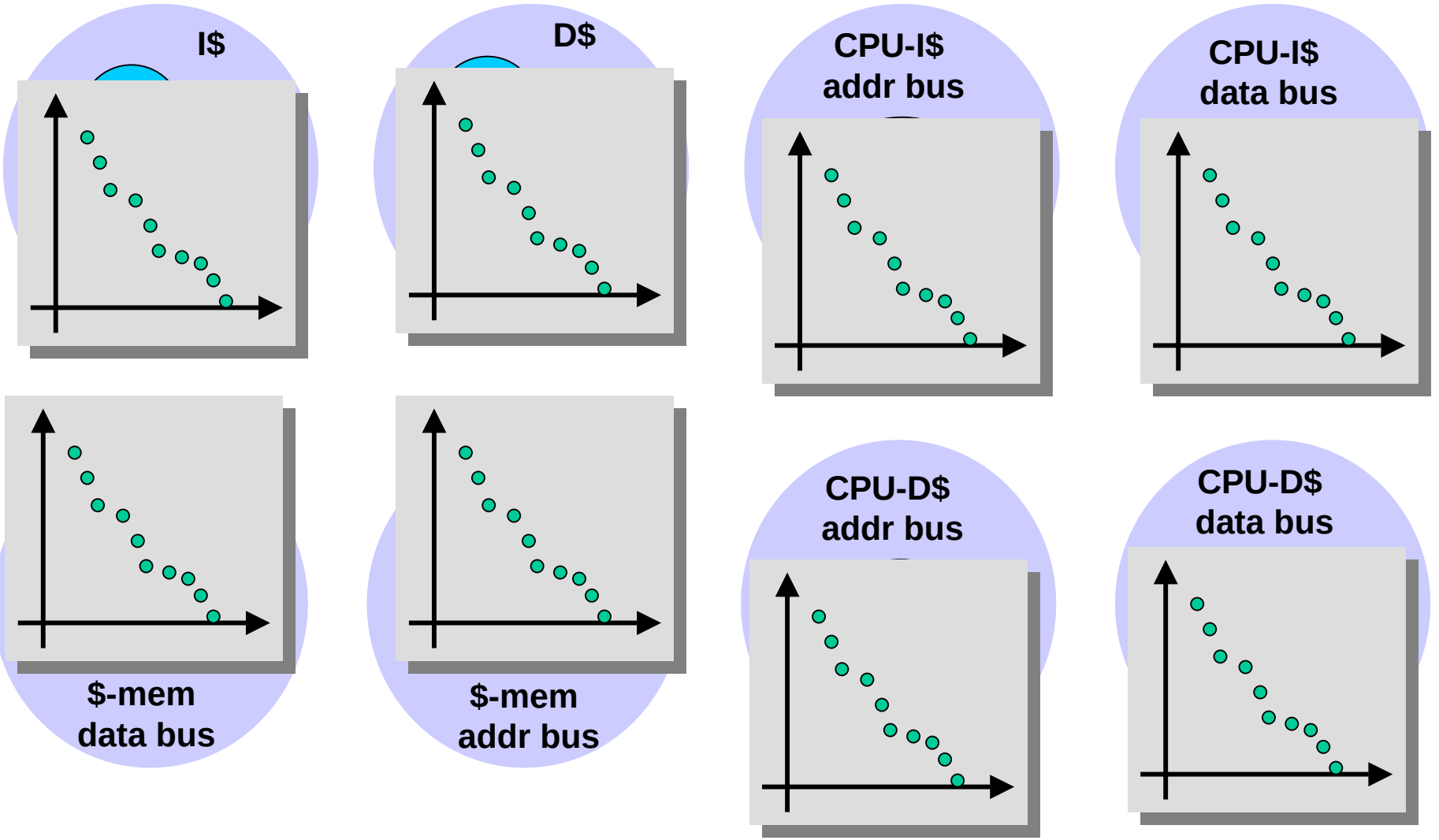
dep: How It Works

- Interdependent parameters are grouped in clusters
- 1st phase
  - ➔ Clusters are exhaustively explored with the aim to compute the local Pareto-optimal set (LPOS)
- 2nd phase
  - ➔ The LPOSs are merged and exhaustively searched to find the global Pareto-optimal set (GPOS)

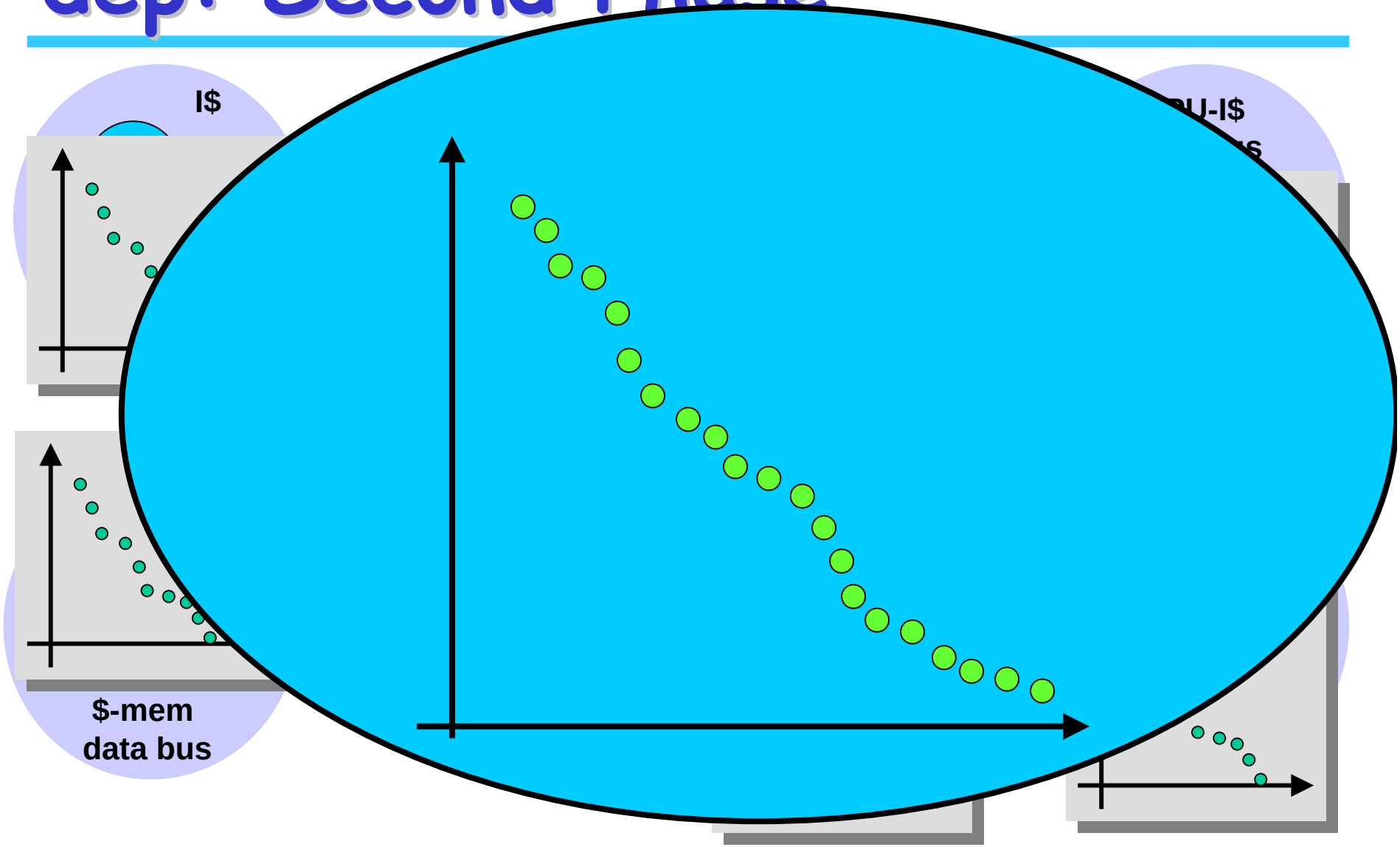
dep: Dependency Graph



dep: First Phase



dep: Second Phase



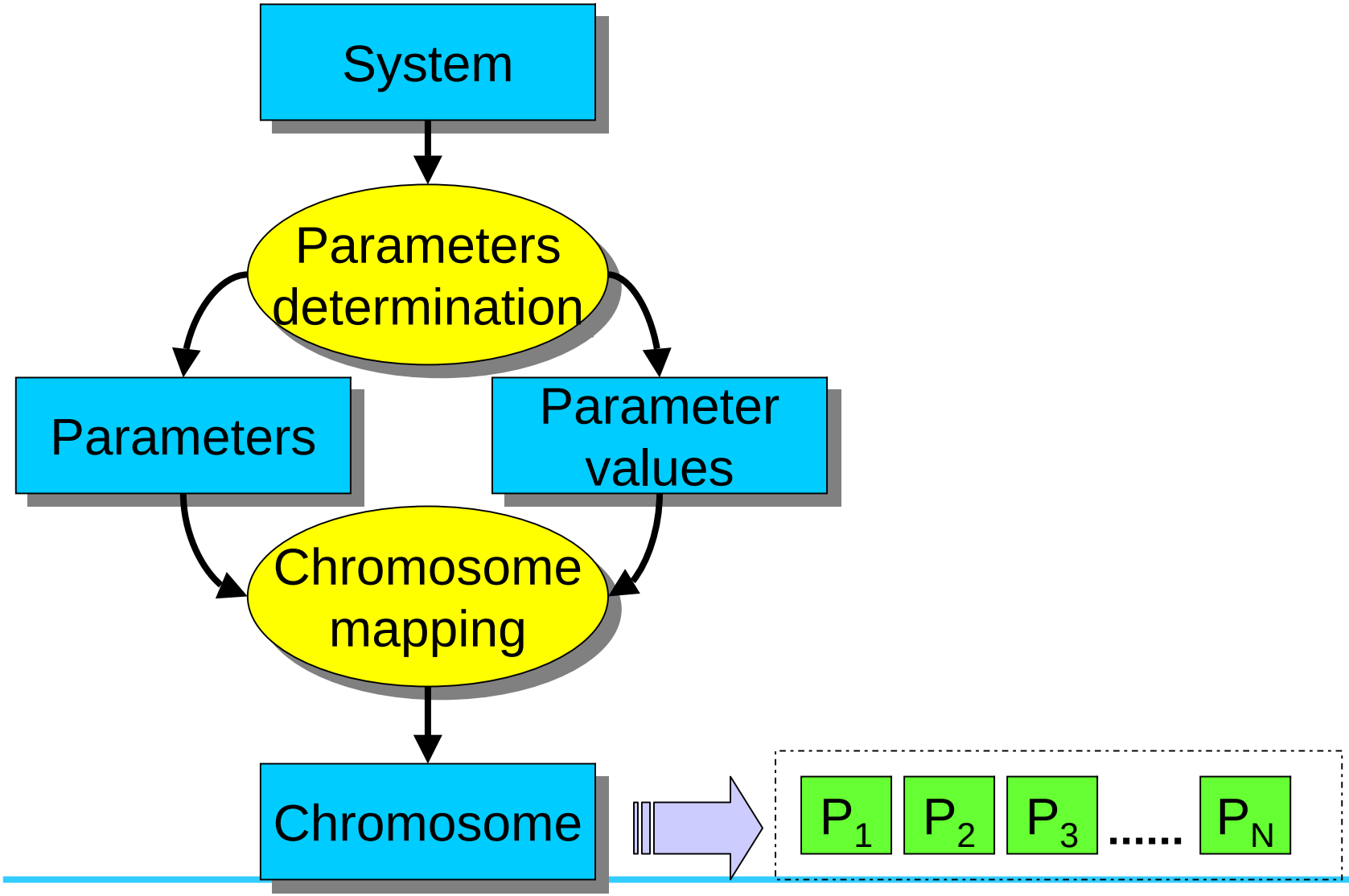
dep: Problems

- If the parameters are heavily interdependent
  - ➔ Big cluster
  - ➔ Local configuration space too large
  - ➔ The approach becomes infeasible

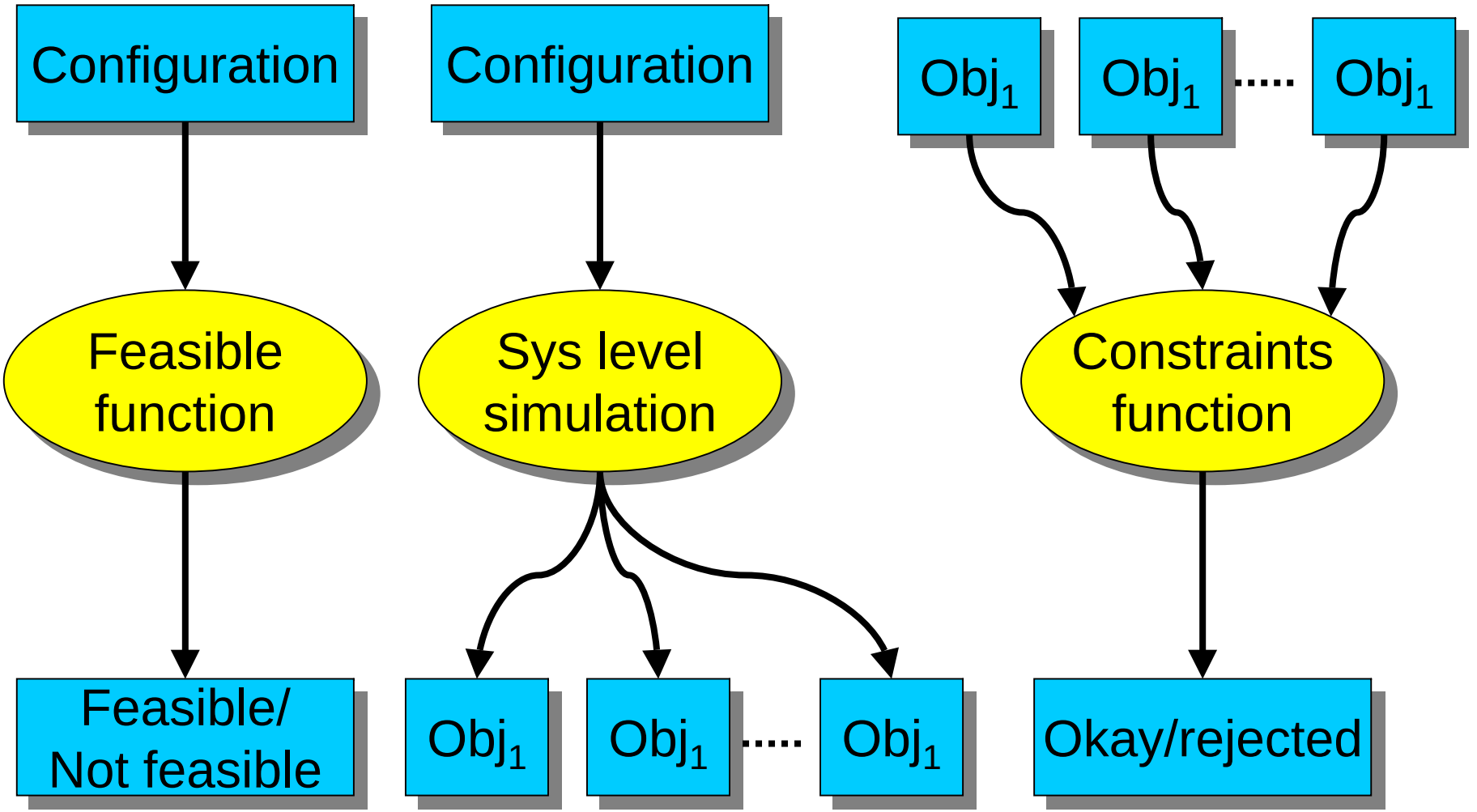
GA: Application

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

Configuration Representation

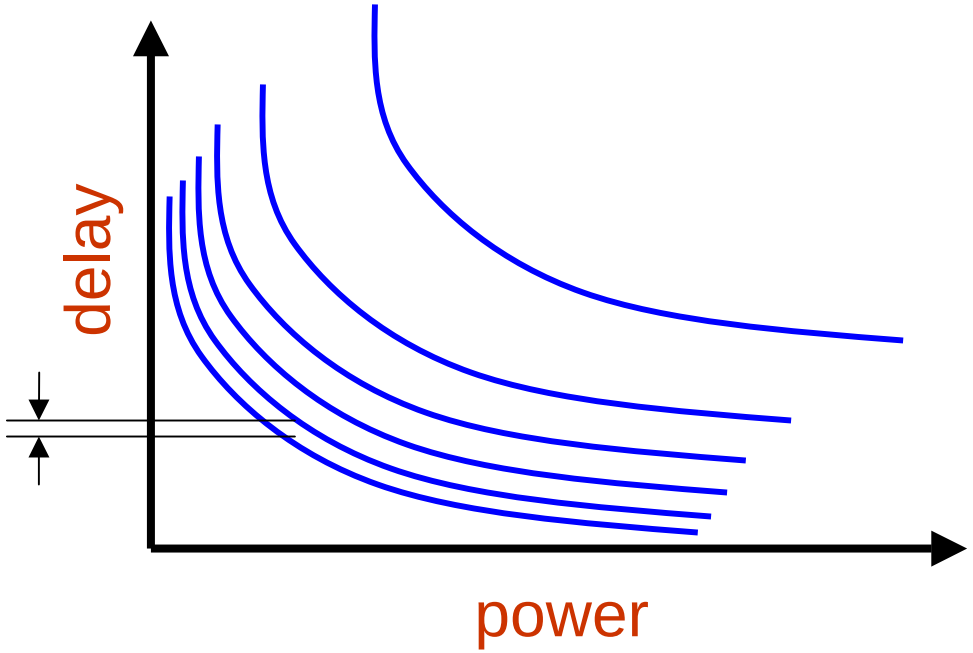


Feasible/Cost/Constraints Functions



How Many Generations?

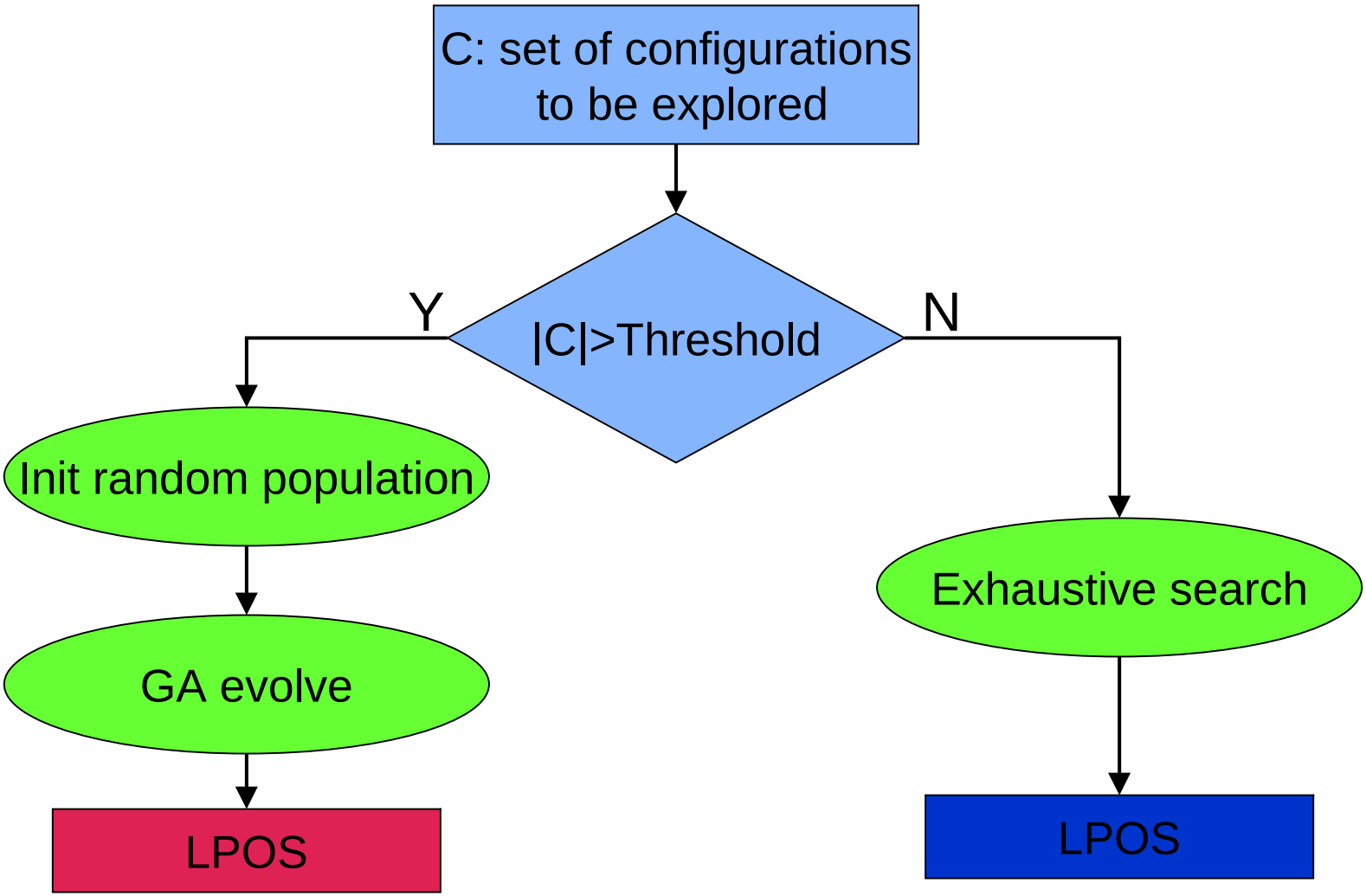
- Fixed number of generations
- Autostop criteria
  - ➔ Based on convergency



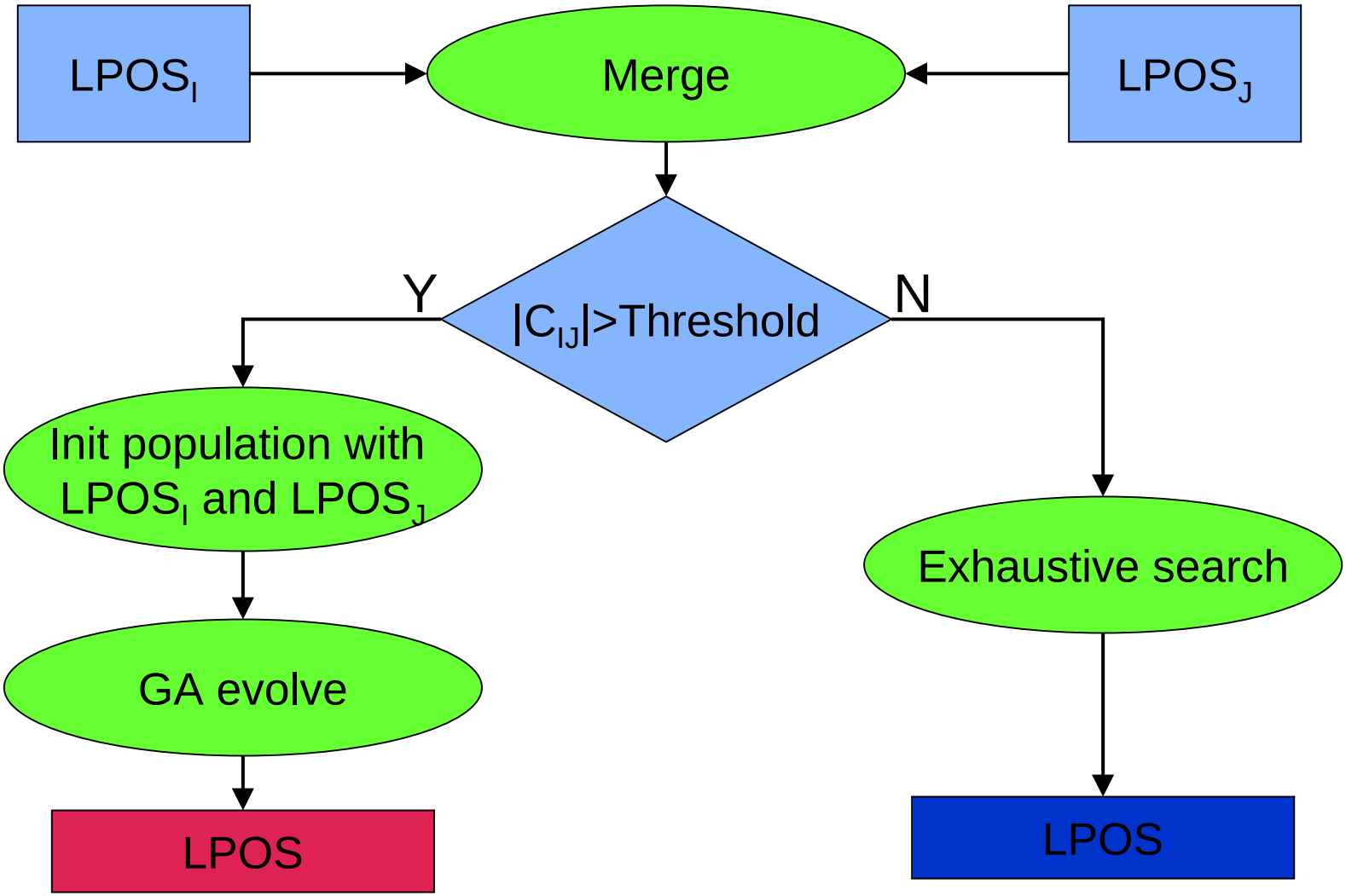
dep: Problems & Solutions

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

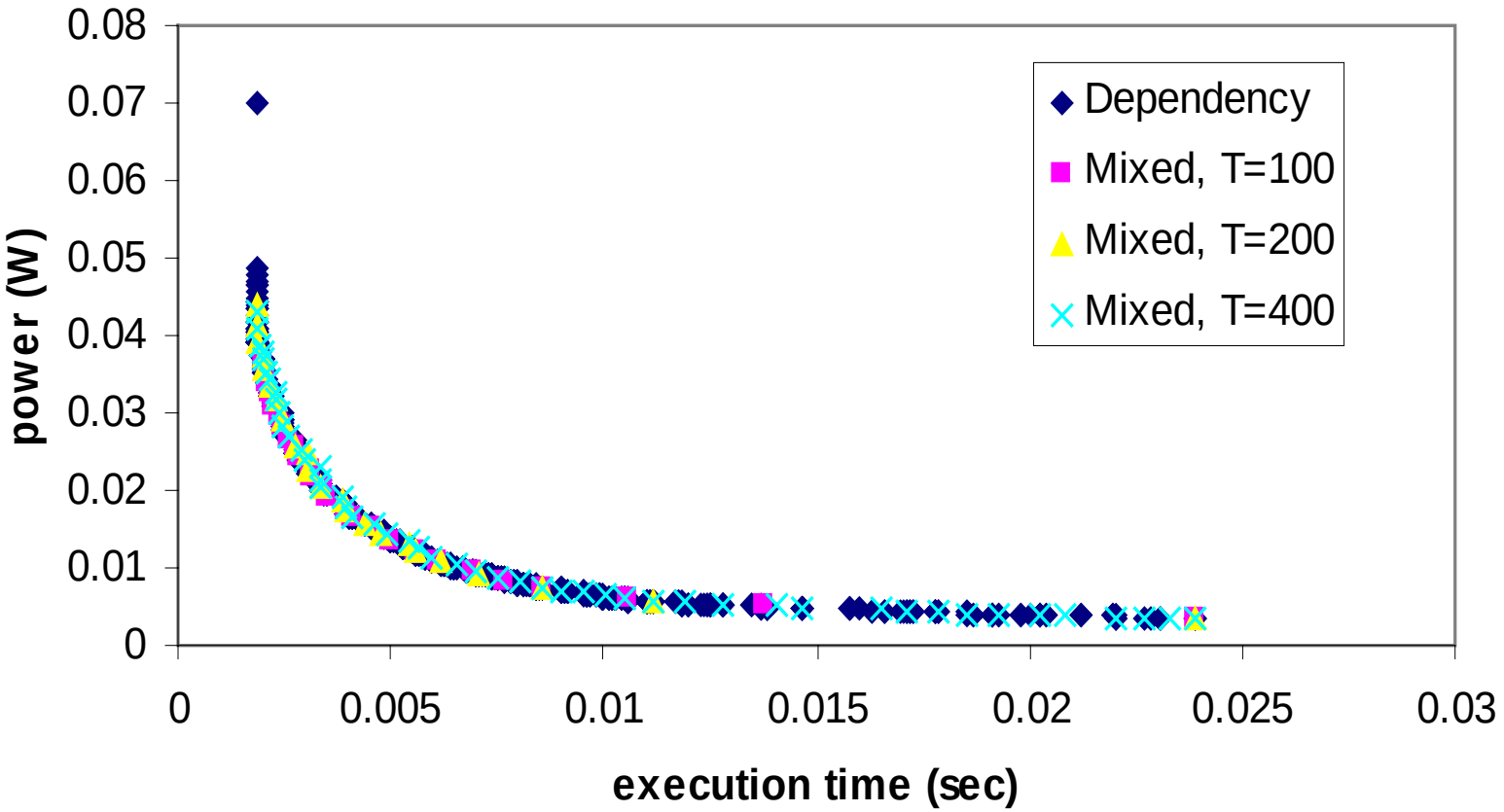
depga: 1st phase



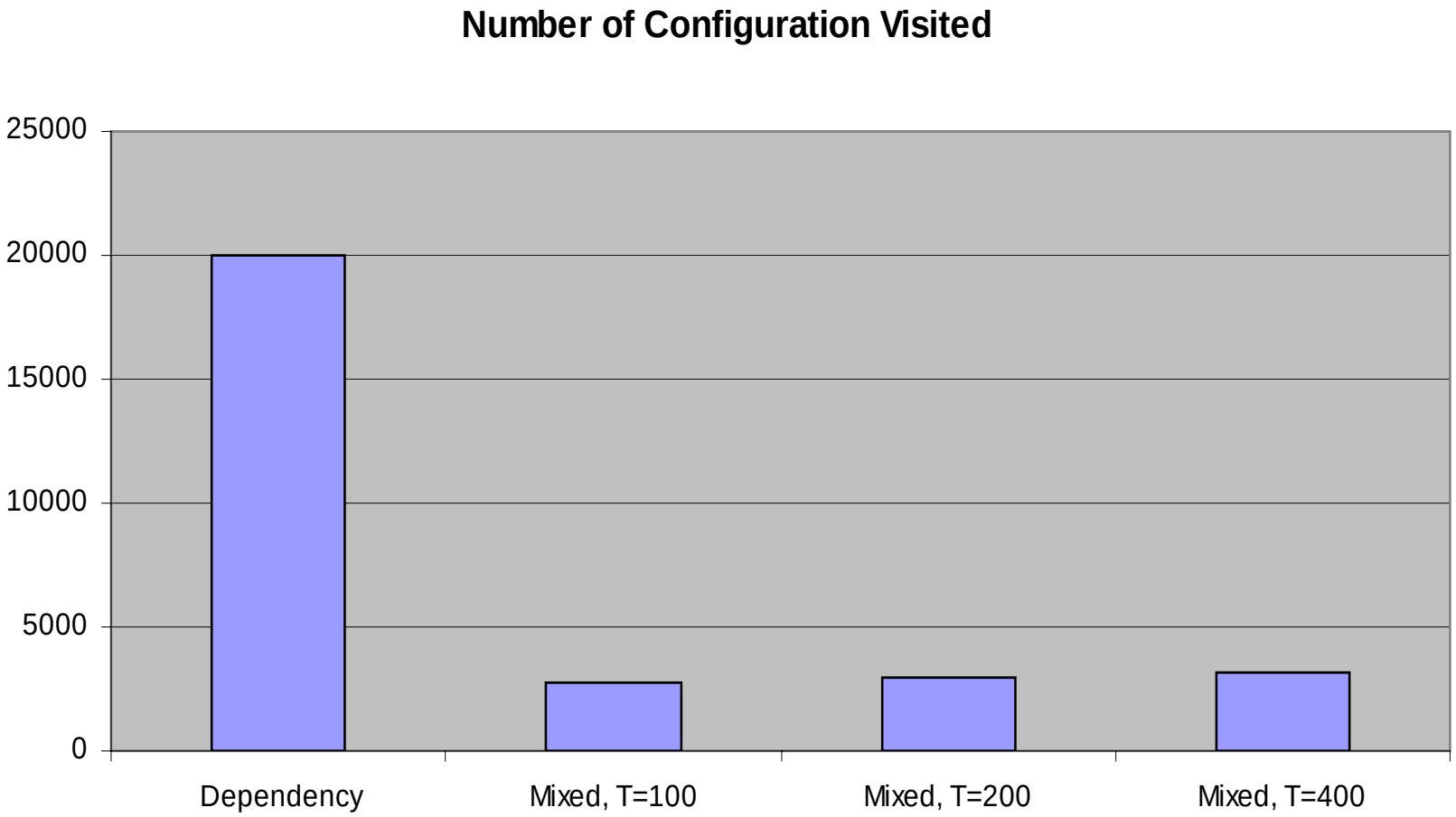
depga: 2nd phase



depga: Experiments



Efficiency


Number of Configuration Visited
25000
20000
15000
10000
5000
0
Dependency
Mixed, T=100
Mixed, T=200
Mixed, T=400

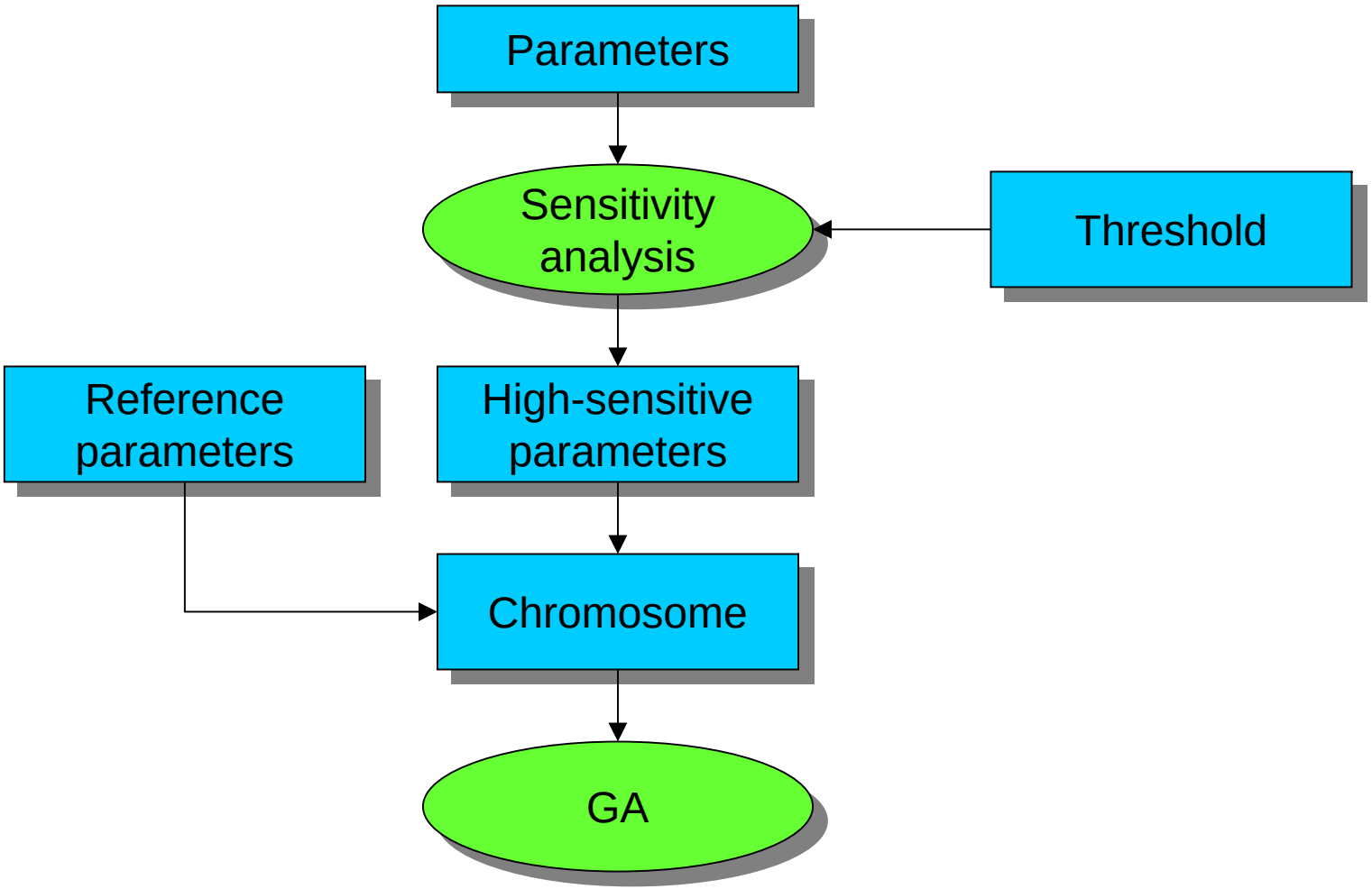
Sensitivity Analysis

- Let us consider N parameters $P_1, P_2, \ldots, P_N$ each of them can assume respectively $\{V_1\}$, $\{V_2\}, \ldots, \{V_N\}$ values
  - ➔ The number of possible configurations will be $|V_1| \times |V_2| \times, \ldots, \times |V_N|$
  - ➔ Sensitivity analysis allows to test only $|V_1| + |V_2| +, \ldots, + |V_N|$ 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 $c=<s,b,a>$ (size,bsize,assoc)
  - Sensitivity analysis
    - Fix b and a and let s variable $\Rightarrow Pd^{s}_{min}$, Pd^{s}_{max}
    - Fix s and a and let b variable $\Rightarrow Pd^{b}_{min}$, Pd^{b}_{max}
    - Fix s and b and let a variable $\Rightarrow Pd^{a}_{min}$, Pd^{a}_{max}
    - 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


Parameters
Sensitivity analysis
Threshold
Reference parameters
High-sensitive parameters
Chromosome
GA

Overall Results

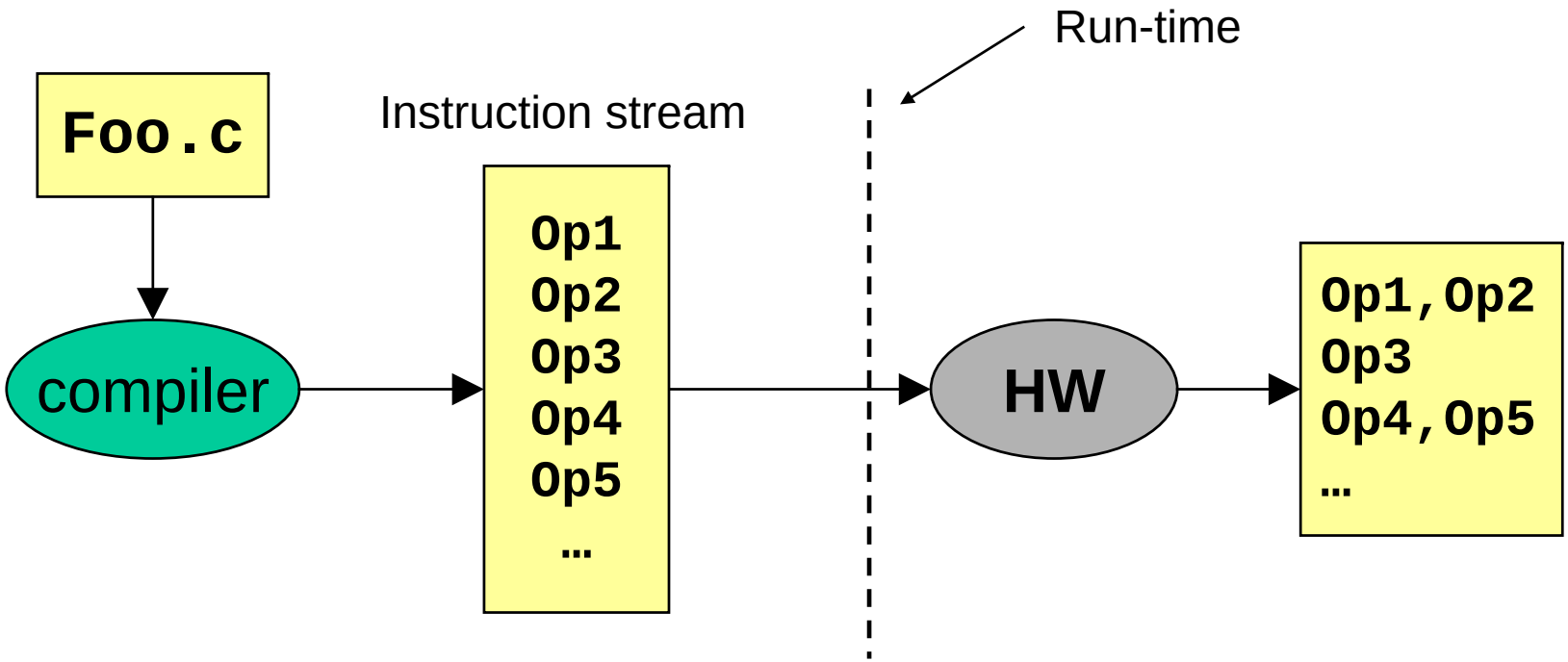
Benchmark	dep	ga		depga		pbsa		saga	
	sims	% dist	% sav	% dist	% sav	% dist	% sav	% dist	% sav
adpcm	14694	0.04	89.21	0.02	78.75	0.00	80.53	0.03	91.40
bcnt	22642	0.02	93.01	0.01	89.67	0.00	83.99	0.03	95.42
blit	37787	0.04	95.82	0.00	90.99	0.02	93.09	0.05	97.03
compress	14035	0.03	89.02	0.01	75.24	0.01	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

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)

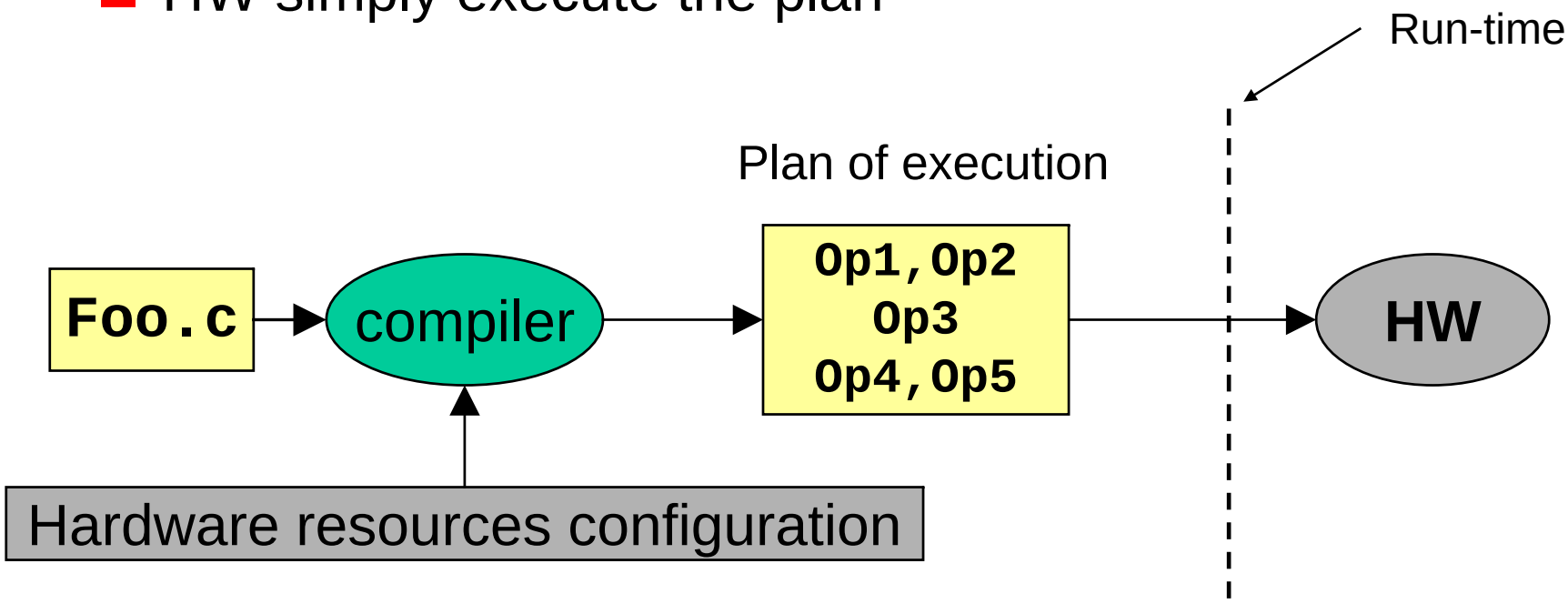
ILP philosophy: Superscalar

- 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

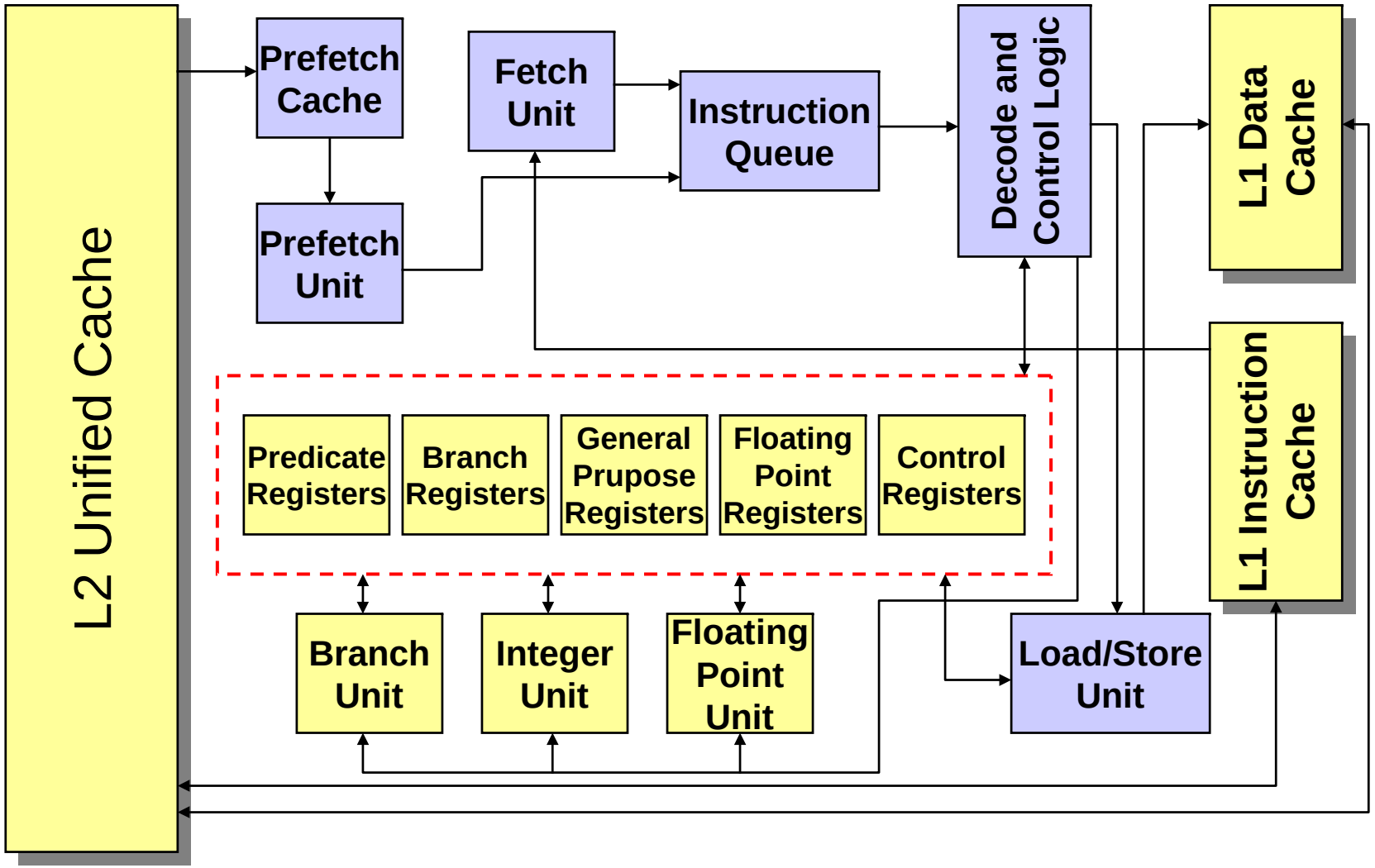


VLIW past & future

- 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
  - ➔ Advanteges of VLIW
    - ✓ Simplified hardware
    - ✓ Optimize ad-hoc the architecture to achieve ILP

Reference architecture (HPL-PD)

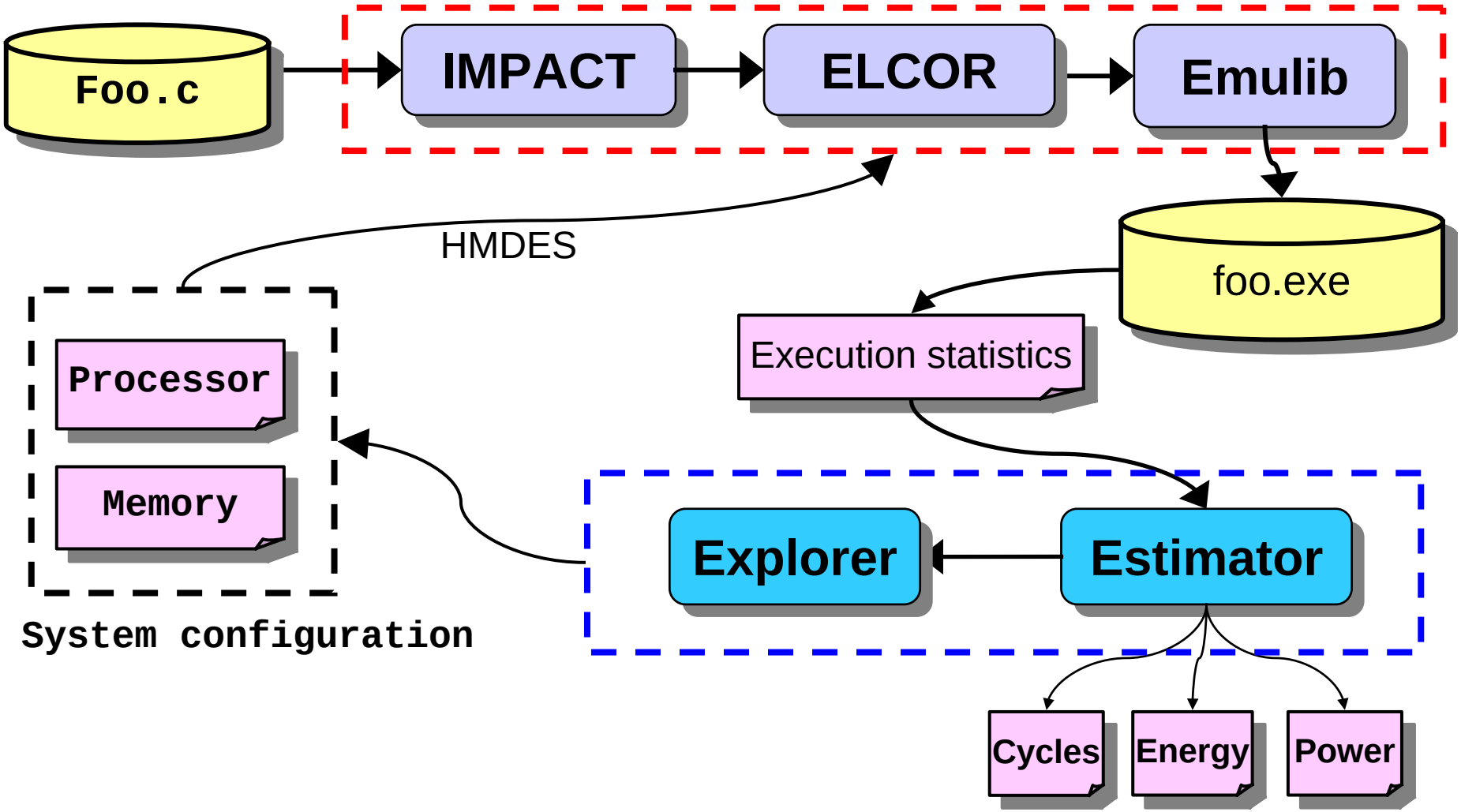


An Open Platform: EPIC Explorer

- 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
  - ➔ $E_{FBU}=(P_{active} \times cycles_{active} + P_{inactive} \times cycles_{inactive}) \times T_{clock}$
- Discrete degree of accuracy (about 25%)
  - ➔ investigate relative power savings beetween designs

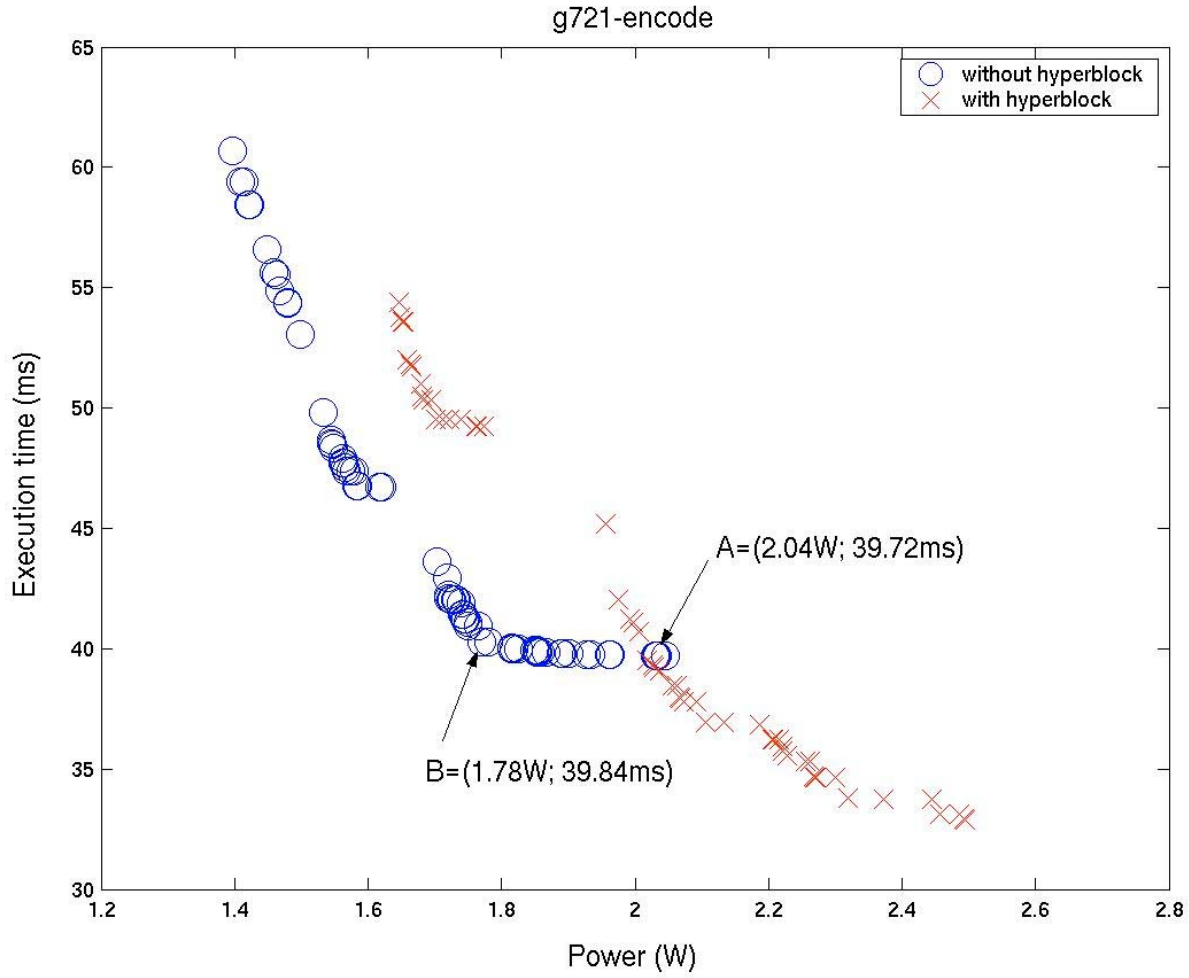
Configuration Space

Three main parameter categories:

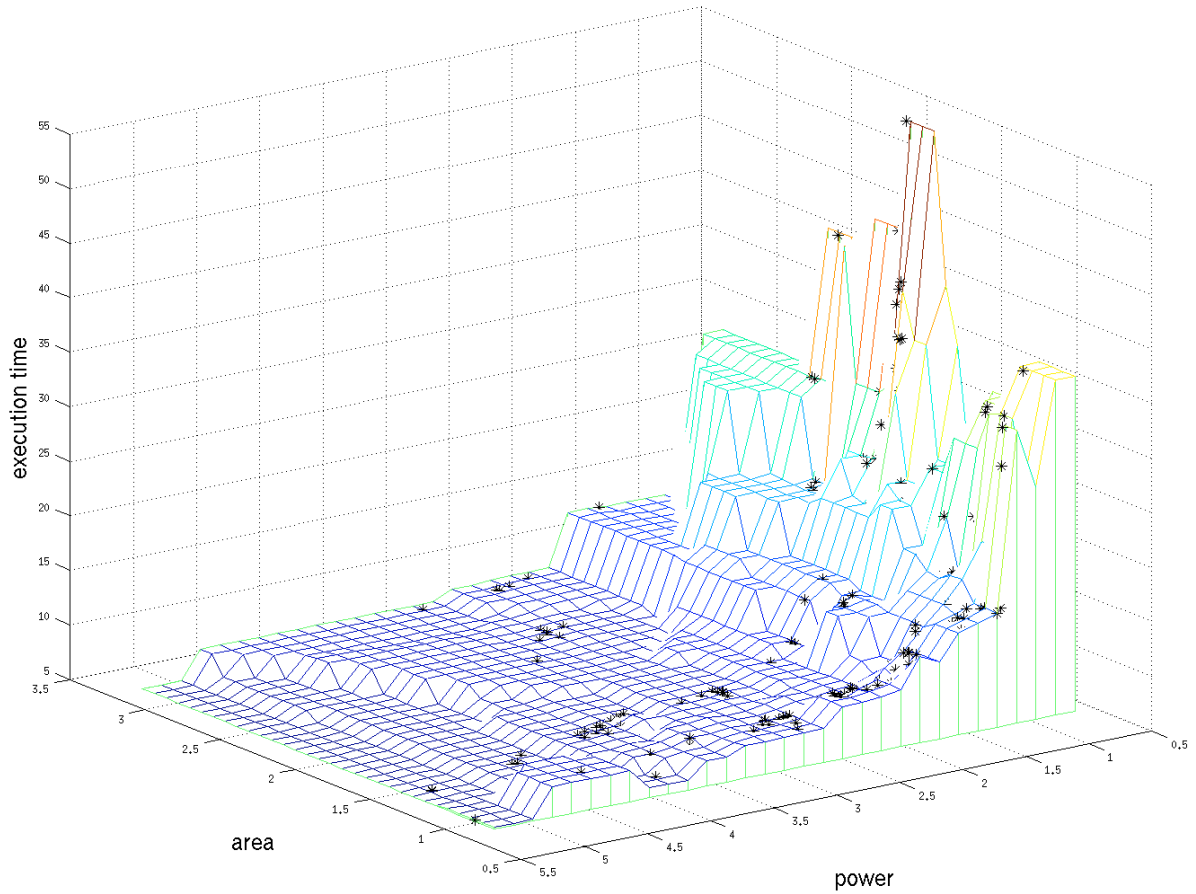
- **VLIW core:**
  - ➔ Number of Registers in each register file (from 16 to 256)
  - ➔ Number of istancies for Functional Units of each type (from 1 to 6)
- **Mem Hierarchy:**
  - ➔ Size, Blocksize, Associativity for each of the caches (L1 Instruction, L1 Data, L2)
- **Compiler**:
  - ➔ Conservative compilation strategy (basic blocks)
  - ➔ Aggressive ILP oriented compilation strategy (hyperblocks)

Total space size: **1.47×10^{13}** configurations!

Pareto Set (G721 encode)



Pareto Surface



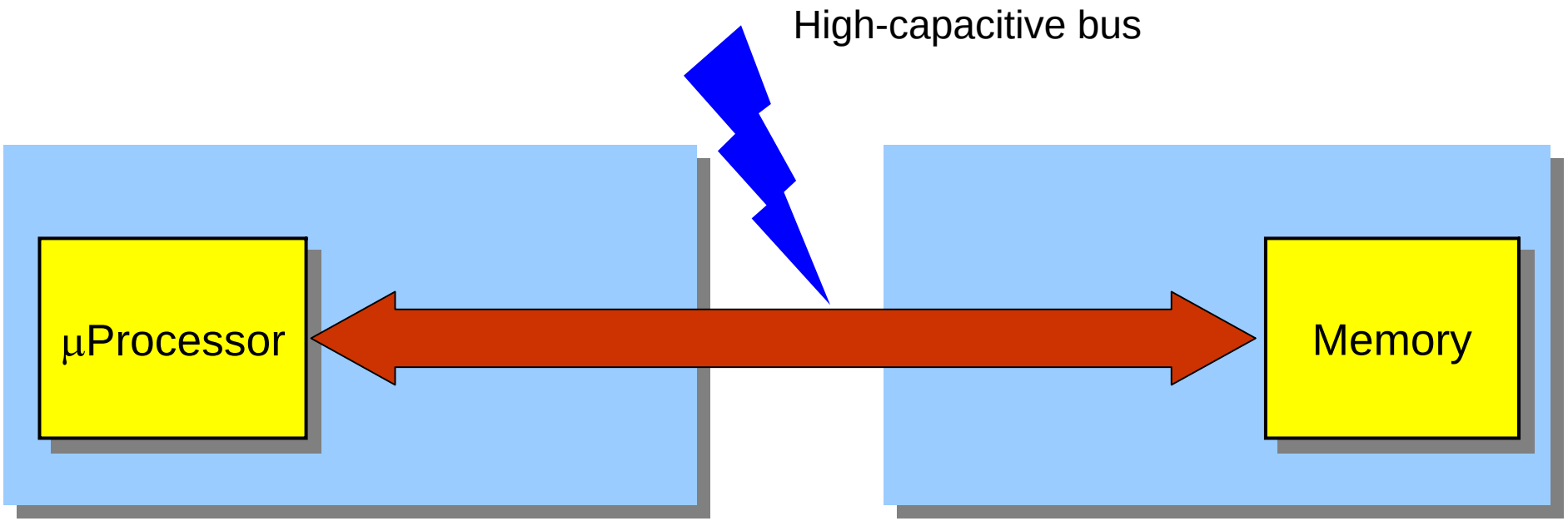
Outline

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

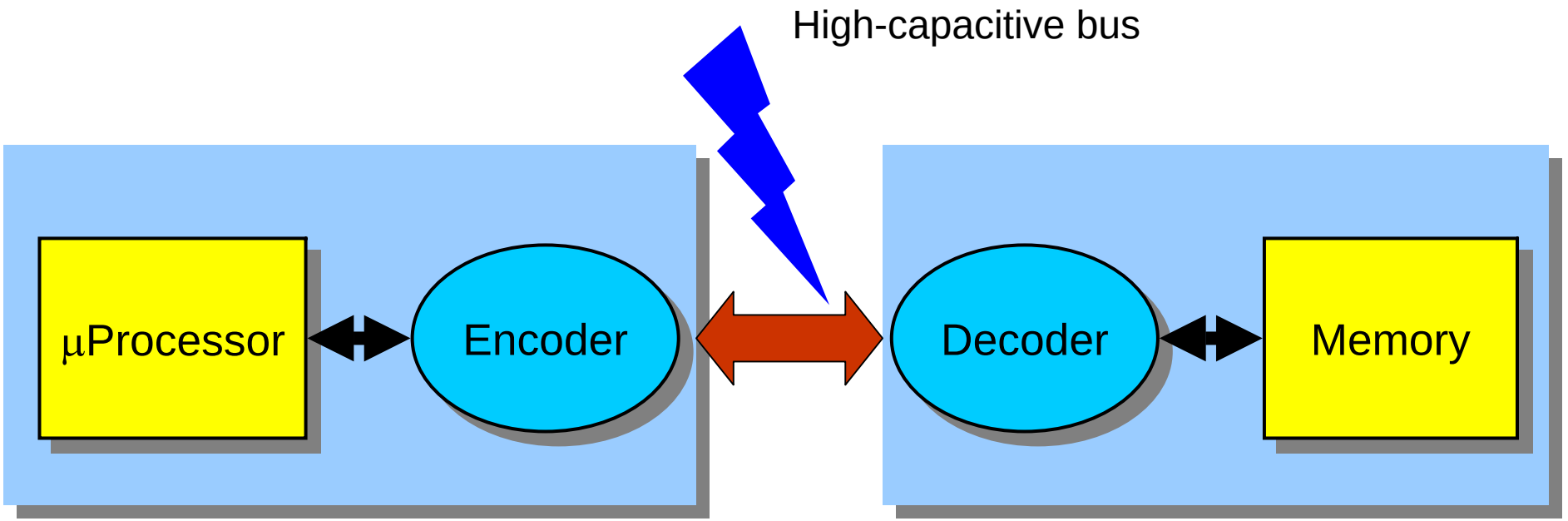
Motivation

- Off-chip accesses are a major source of power consumption
- Increase the bandwidth of the data transfers
  - ➔ PowerPC 620: 128 bit data bus (D$)
- Modern SW applications span a very large address space
  - ➔ DEC Alpha AXP 64 bit addr bus
- Growing in wire-to-gate capacitance ratios

Bus Encoding



Bus Encoding



Formulation of the Problem

- Let U(w) be the universe of discourse for word of w bits
  - ➔ $|U(w)| = 2^w$
- An encoder E is a function $E:U(w) \rightarrow U(w)$ such that

  $\forall\ \alpha, \beta \in U(w), \alpha \neq \beta \Rightarrow E(\alpha) \neq E(\beta)$
  - ➔ **2^w!** different encoders are possible
  - ➔ 8 bit bus → over **10^{506}** encoders

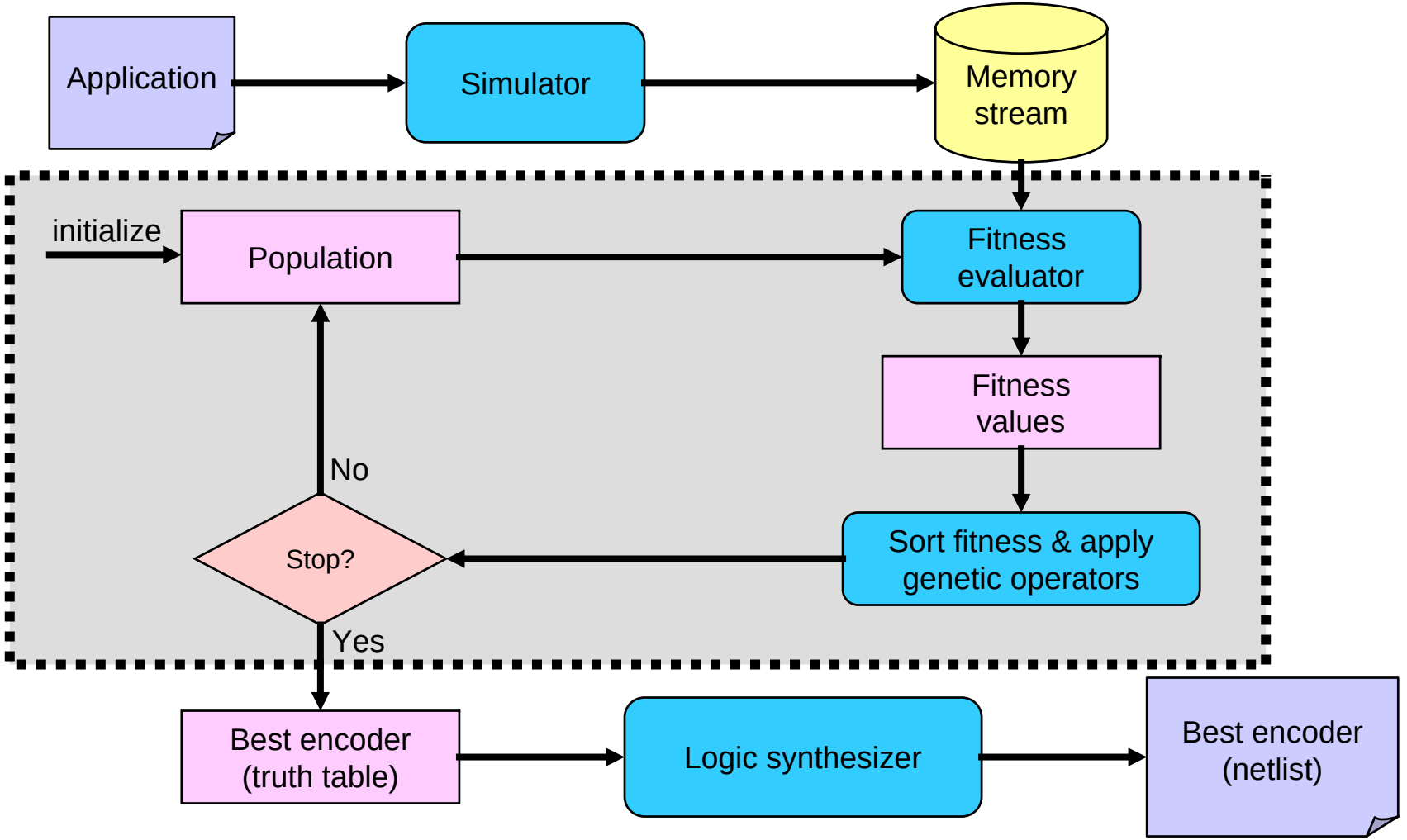
Formulation of the Problem

- Let S be the reference stream
  - ➔ $S = \{s_0, s_1, s_2, \ldots, s_{N-1}\}$
- Let $T_{woe}(S)$ be the number of transitions on the bus due to S without encoding
  - ➔ $T_{woe}(S) = \sum_{i=1}^{N-1} H(s_{i-1}, s_i)$
- Let Twe(S,E) be the number of transitions on the bus due to S with encoding
  - ➔ $T_{we}(S,E) = \sum_{i=1}^{N-1} H(E(s_{i-1}), E(s_i))$

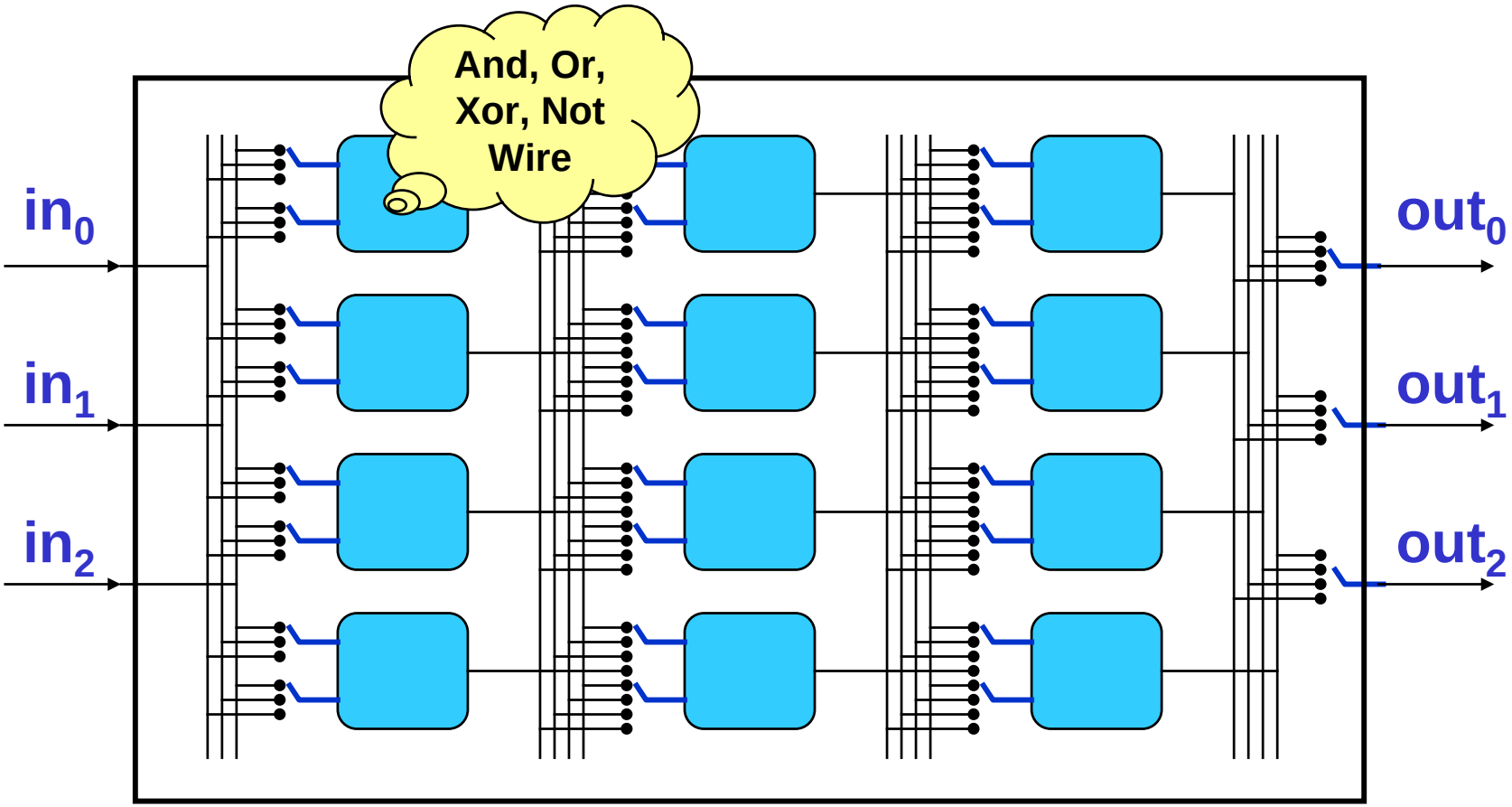
Formulation of the Problem

- Goal
  - ➔ Maximize $\dfrac{T_{woe}(S) - T_{we}(S,E)}{T_{woe}(S)}$

GEG: Genetic Encoder Generator



GNEG



GNEG: Fitness function

- Let C be a logic circuit (with n inputs and n outputs)
- Let us indicate as $O(C)$ the number of pairs of input words that generate the same output
  $O(C)=|\{(x_1,x_2) : x_1 \neq x_2 \wedge C(x_1) \neq C(x_2)\}|$
  - ➔ C is an encoder iff $O(C)=0$
- We define the fitness function as follows

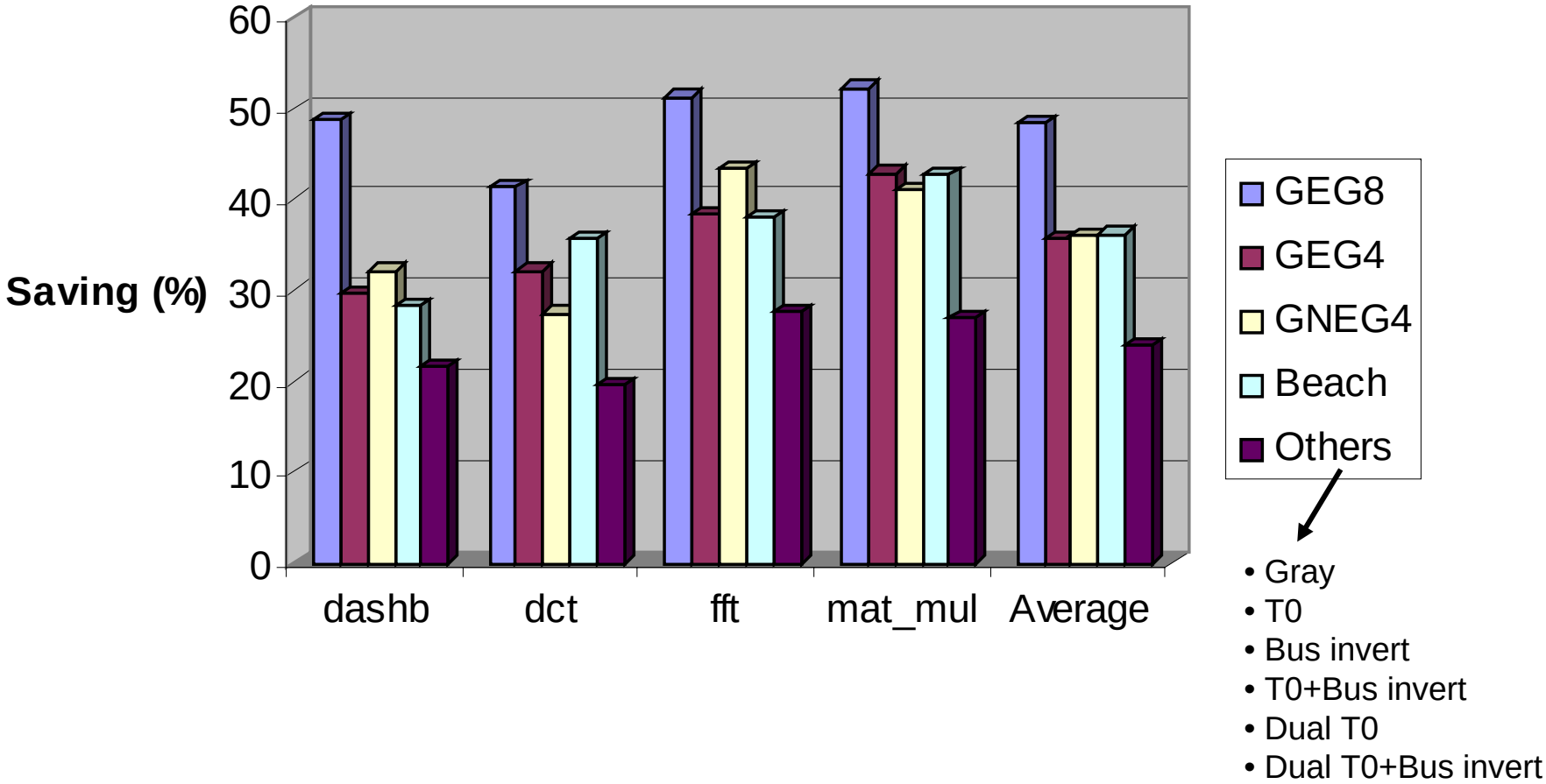
$$f(C) = \begin{cases} 2^n - O(C) & \text{if } O(C) \neq 0 \\ 2^n + s(C) + w(C) & \text{otherwise} \end{cases}$$

Fraction of transitions saved by using the encoder *C* as compared with the amount required when no coding is used

Number of wires in the circuit *C*

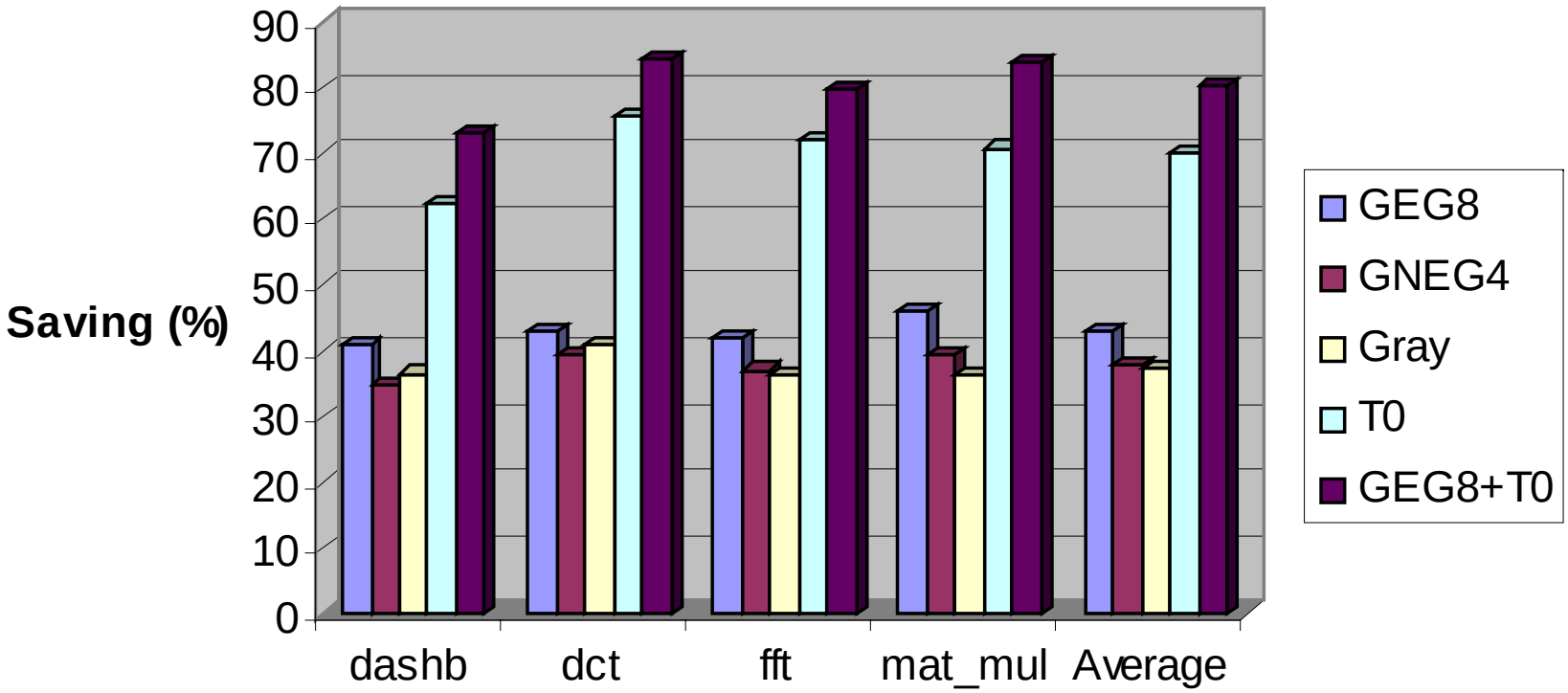
Experiments

- Fetch + Load/Store

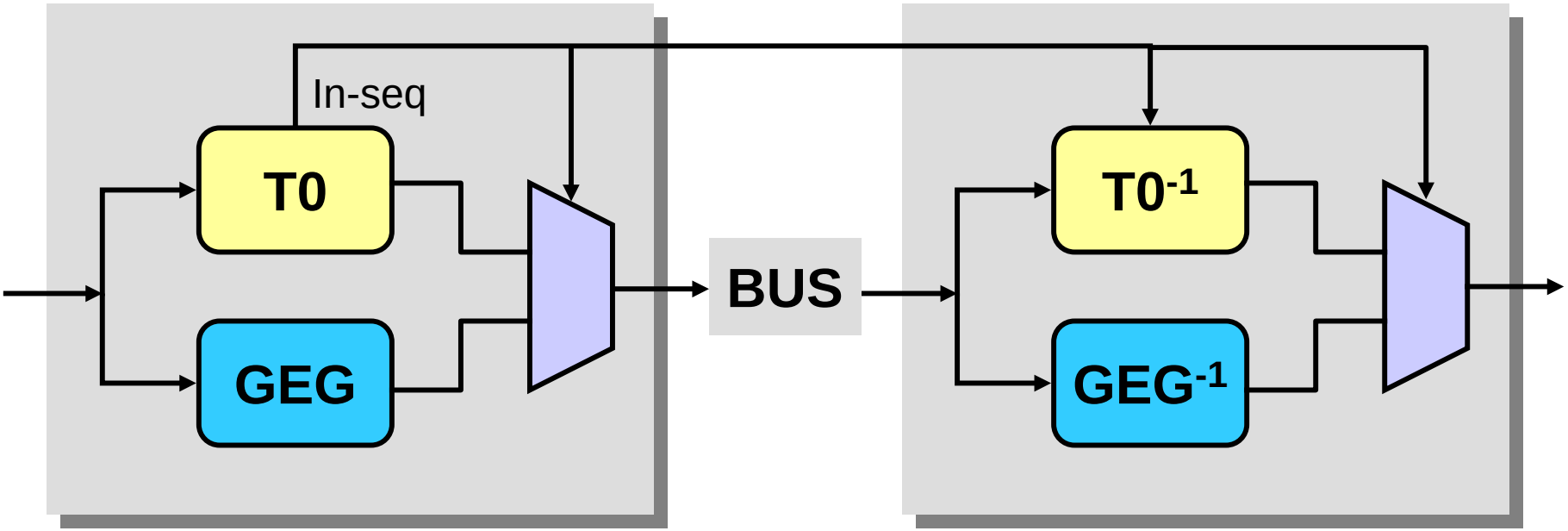


Experiments

- Fetch



GEG+T0

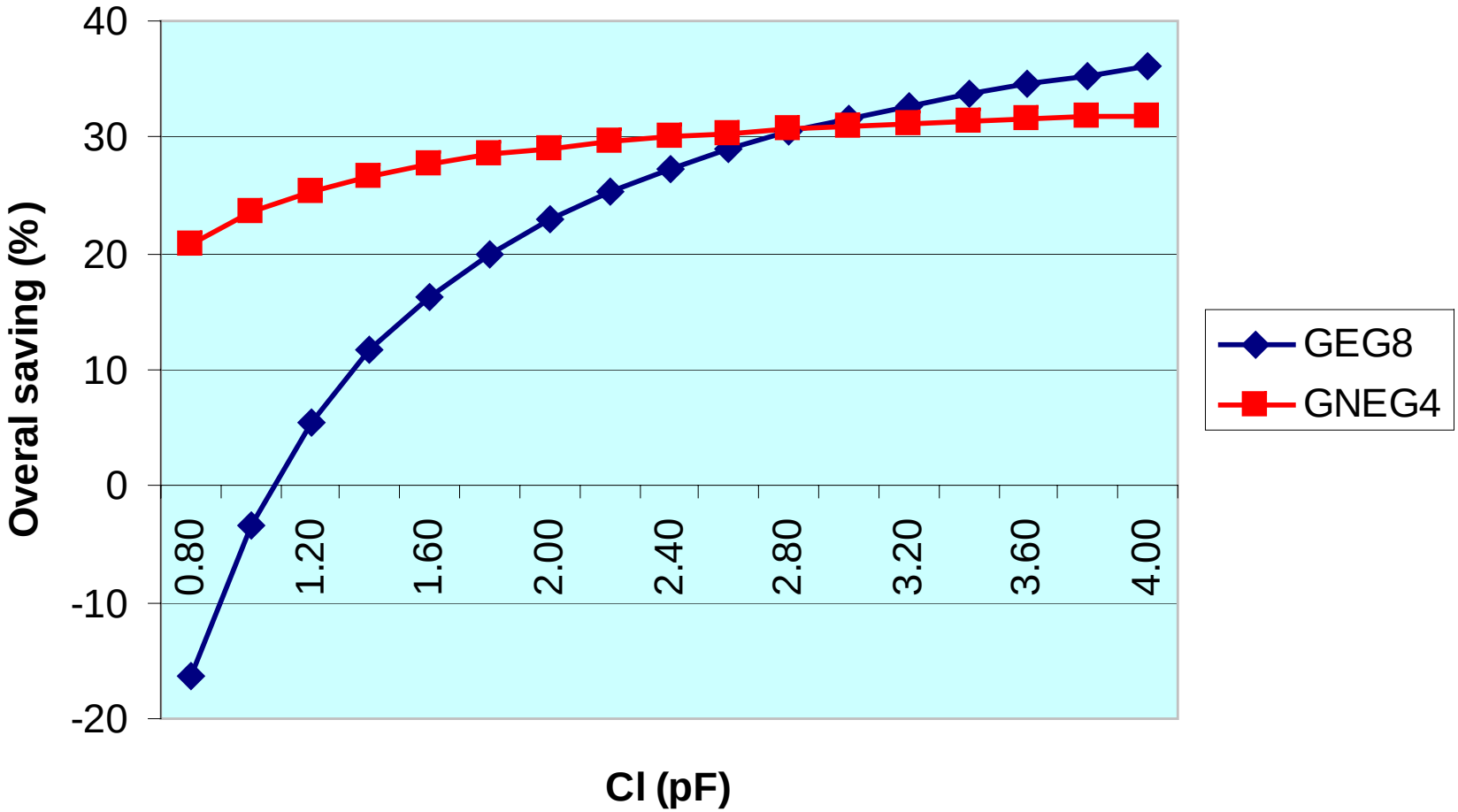


Encoder/Decoder

Benchmark	Area (um^2)		Delay (ns)		Power (mW)	
	Enc	Dec	Enc	Dec	Enc	Dec
GEG8						
dashb	10806	10397	1.50	1.42	0.46	0.47
dct	10642	10352	1.67	1.38	0.45	0.44
fft	10753	10397	1.49	1.38	0.46	0.46
mat_mul	8916	8729	1.53	1.51	0.40	0.42
Average	*10279*	*9969*	*1.55*	*1.42*	*0.44*	*0.45*
GNEG4						
dashb	265	273	0.38	0.42	0.09	0.11
dct	255	275	0.39	0.43	0.09	0.11
fft	298	301	0.41	0.48	0.10	0.10
mat_mul	272	280	0.39	0.41	0.08	0.09
Average	*273*	*282*	*0.39*	*0.44*	*0.09*	*0.10*

0.13μm, 1.2V from Virtual Silicon

Overal saving



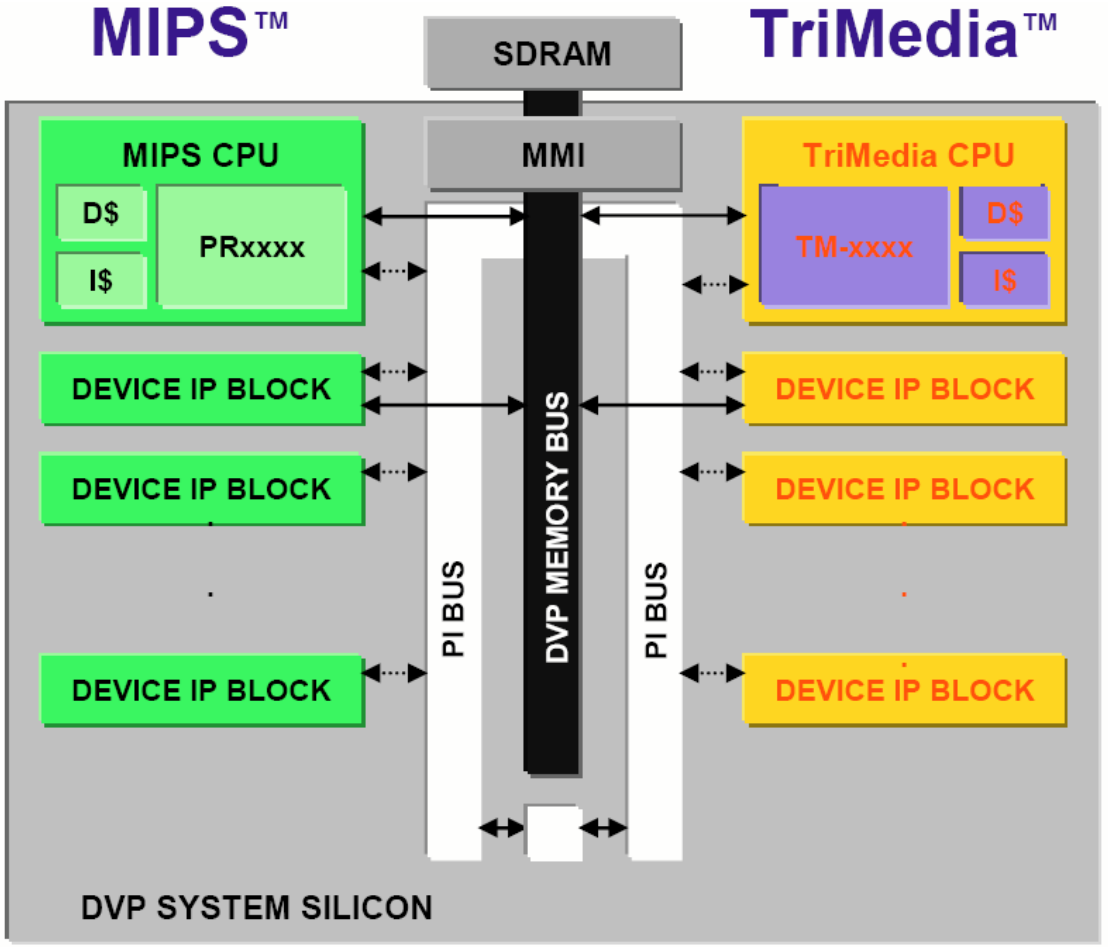
Outline

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

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**



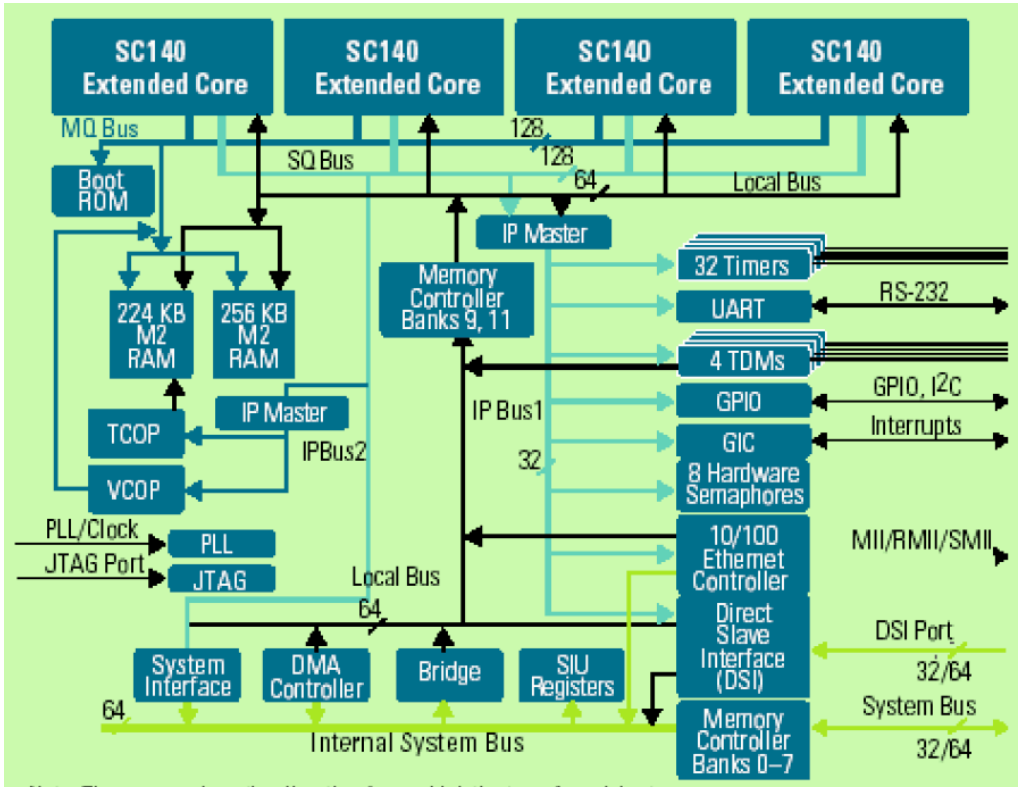
Scalable VLIW Media Processor:
- **100 to 300+ MHz**
- **32-bit or 64-bit**

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 coprocesssors (TCOP and VCOP) to provide special-purpose processing capability in parallel with the core processors

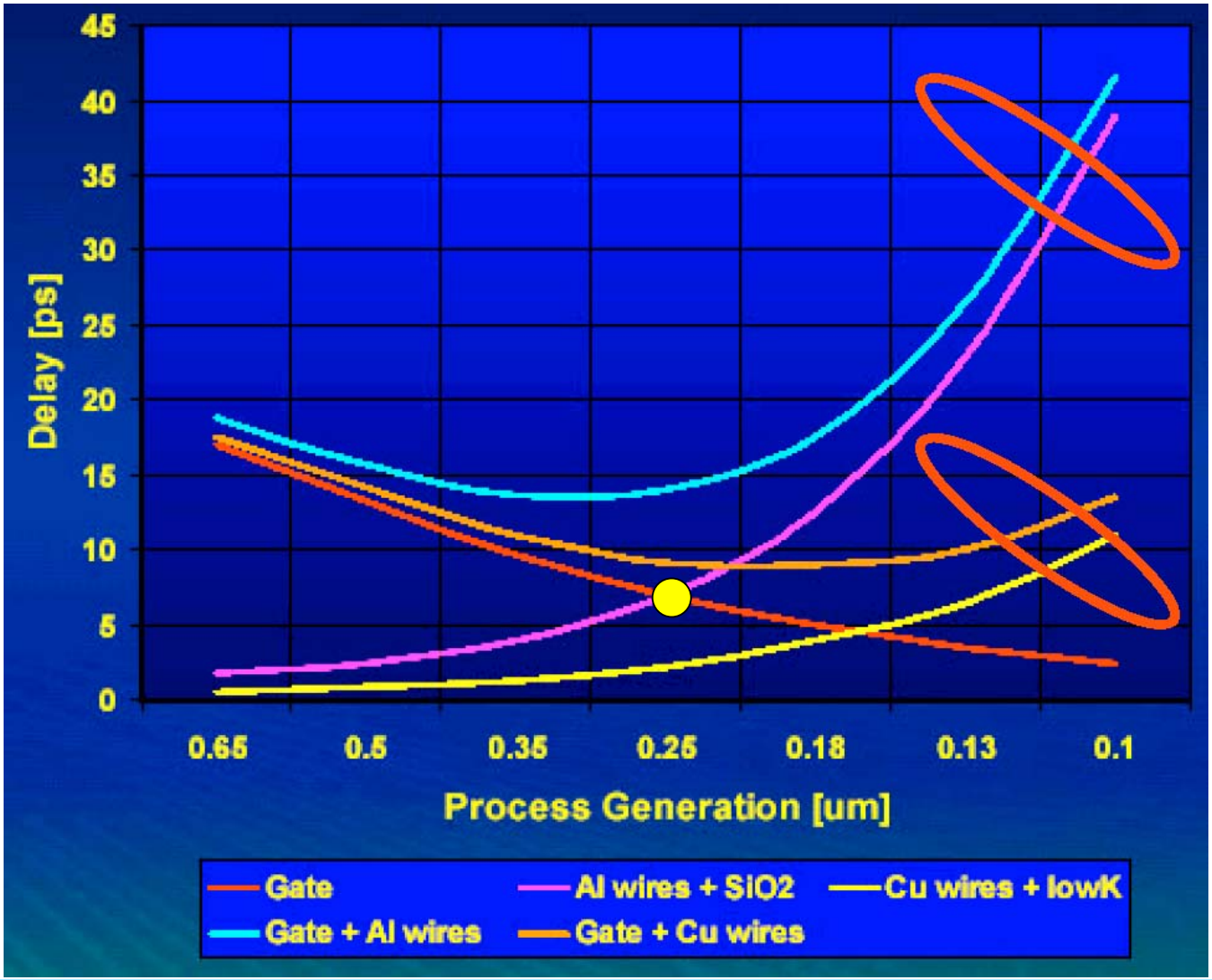


6 Cores: Motorola's MSC8126 SoC platform for 3G base stations (late 2003)

What´s Happening in SoCs?

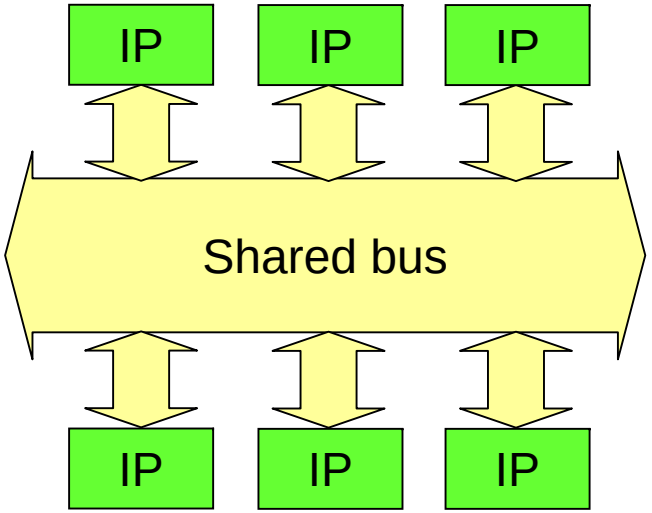
- Technology: no slow-down in sight!
  - ➔Faster and smaller transistors: 90 → 65 → 45 nm
  - ➔… but slower wires, lower voltage, more noise!
    - ✓80% or more of the delay of critical paths will be due to interconnects
- Design complexity: from 2 to 10 to 100 cores!
  - ➔Design reuse is essential
  - ➔…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

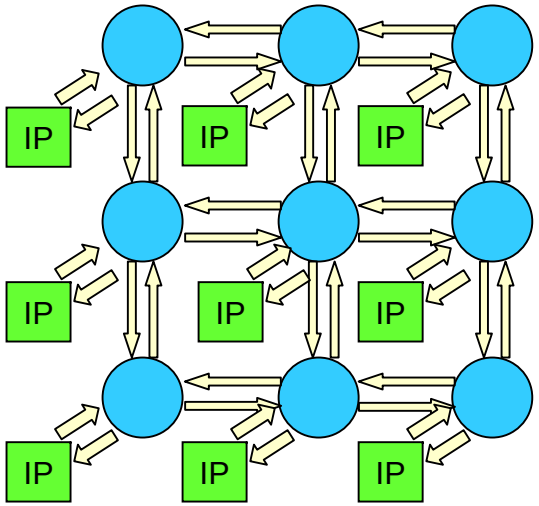


Communication Architectures

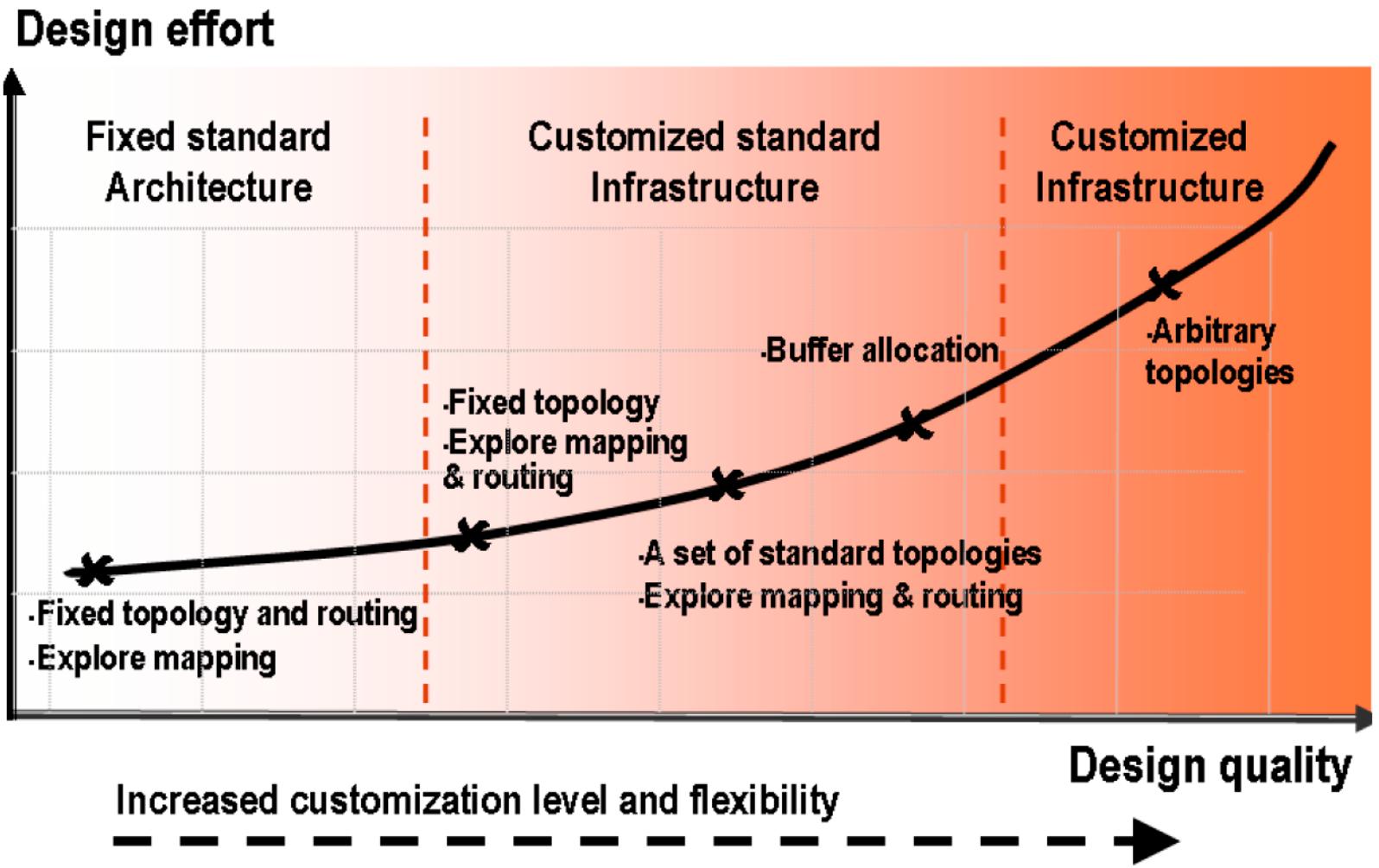
- Shared bus
  - Low area
  - Poor scalability
  - High energy consumption



- Network-on-Chip
  - Scalability and modularity
  - Low energy consumption
  - Increase of design complexity



Design Space Exploration for NoC



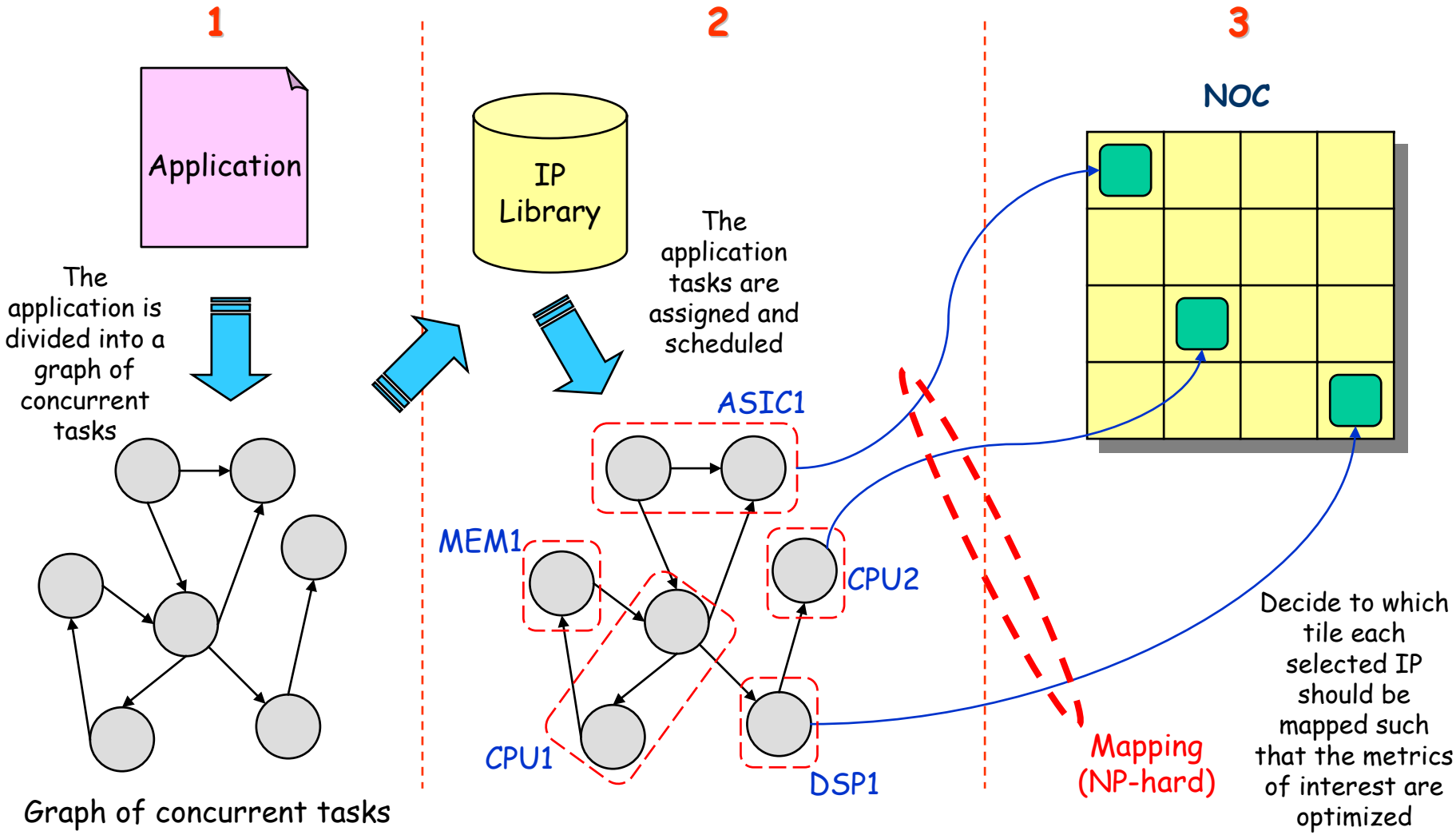
Ogras *et al.*, ASAP'05

NOC Research Topics

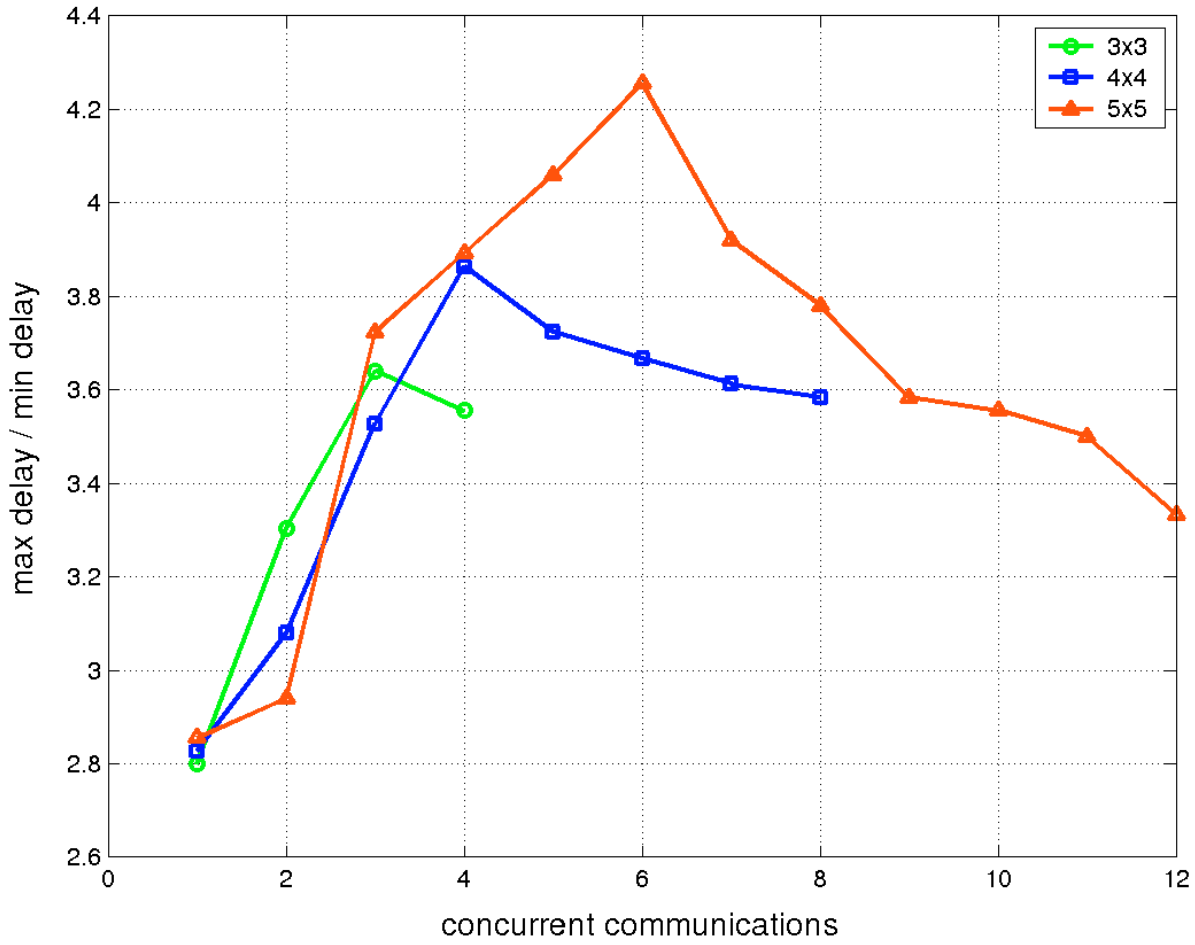
- Topological Mapping Problem
- Routing Strategies
  - ➔ Routing Function
  - ➔ Selection Function

Topological Mapping

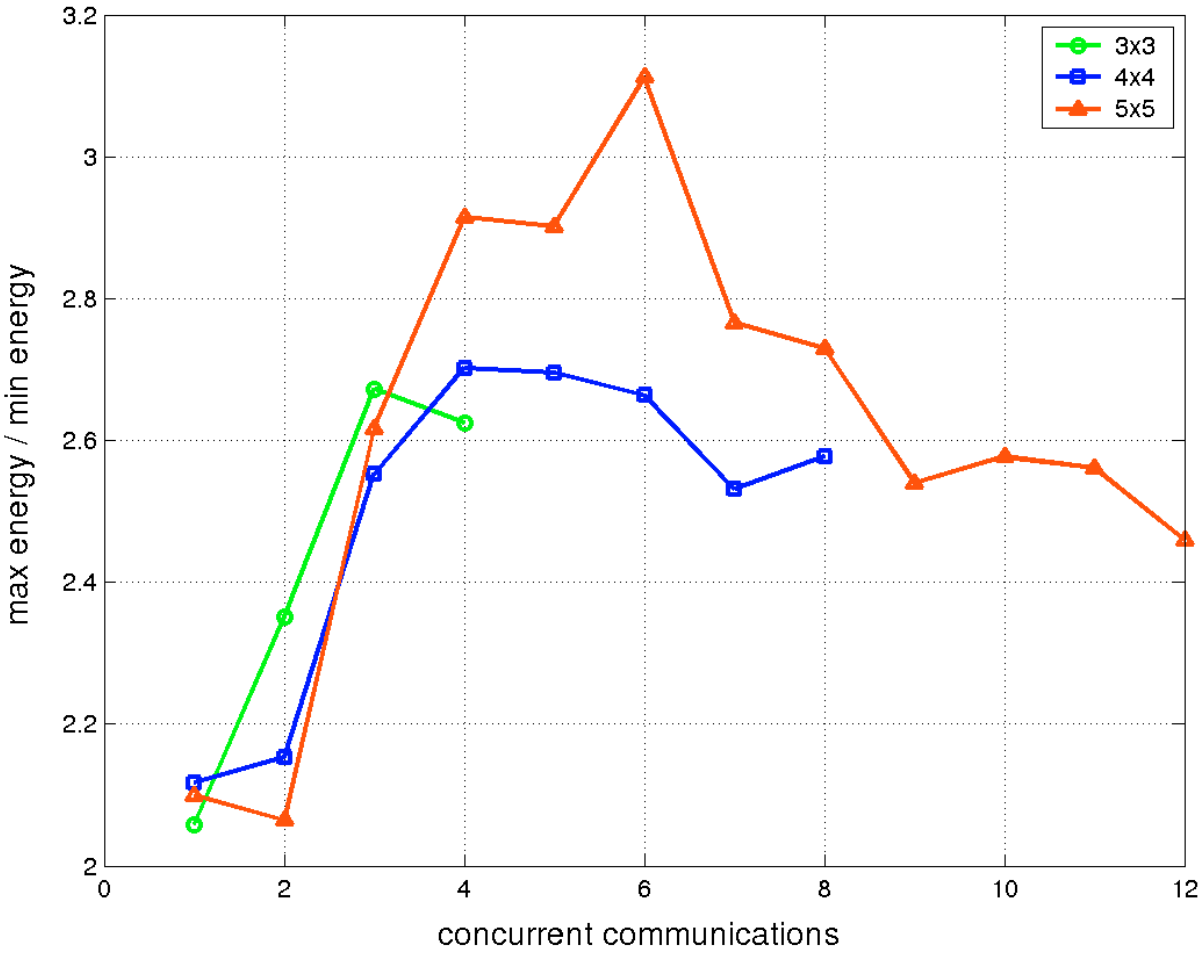
The Mapping Problem



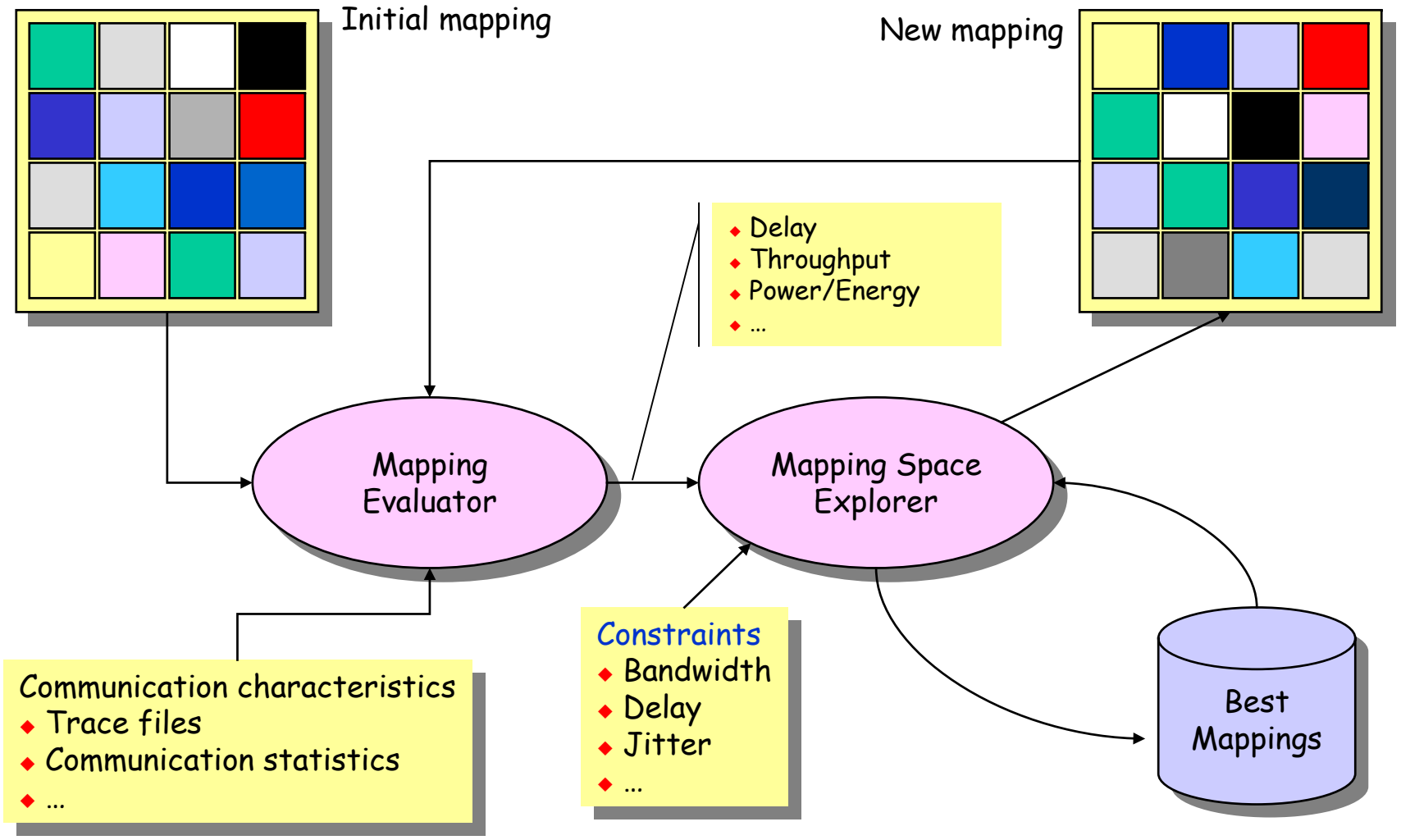
Impact on Performance



Impact on Energy



Design Space Exploration

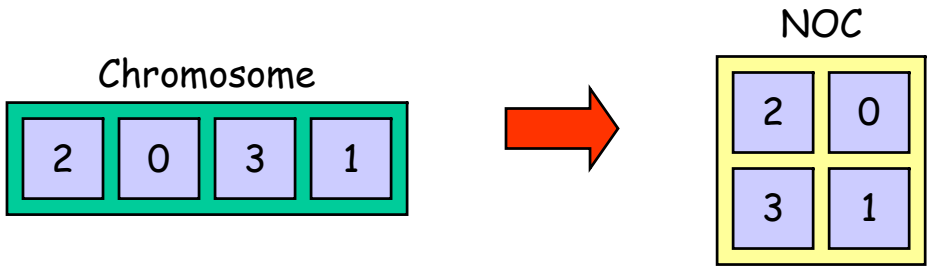


Exploration Engines

- **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

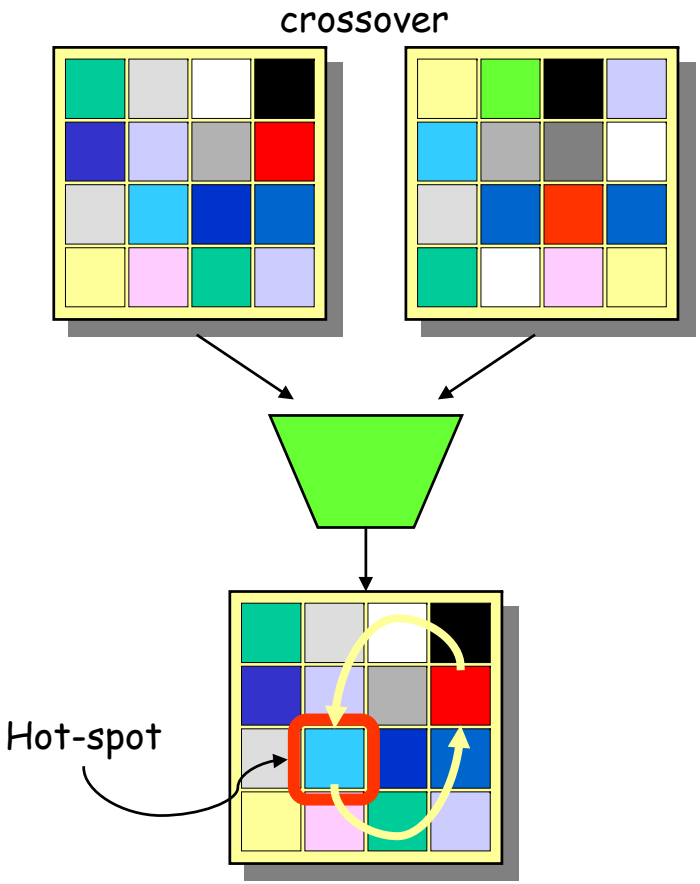
GAMAP

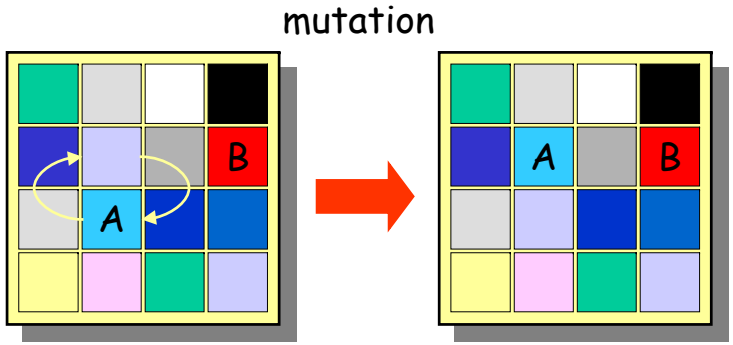
- **Chromosome**: A representation of the solution to the problem (the mapping)
  - ➔ Each tile in the mesh has an associated gene which encodes the identifier of the core mapped in the tile
  - ➔ In an $n \times m$ mesh
    - ✓ The chromosome is formed by $n \times m$ genes
    - ✓ The i-th gene encodes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



GAMAP

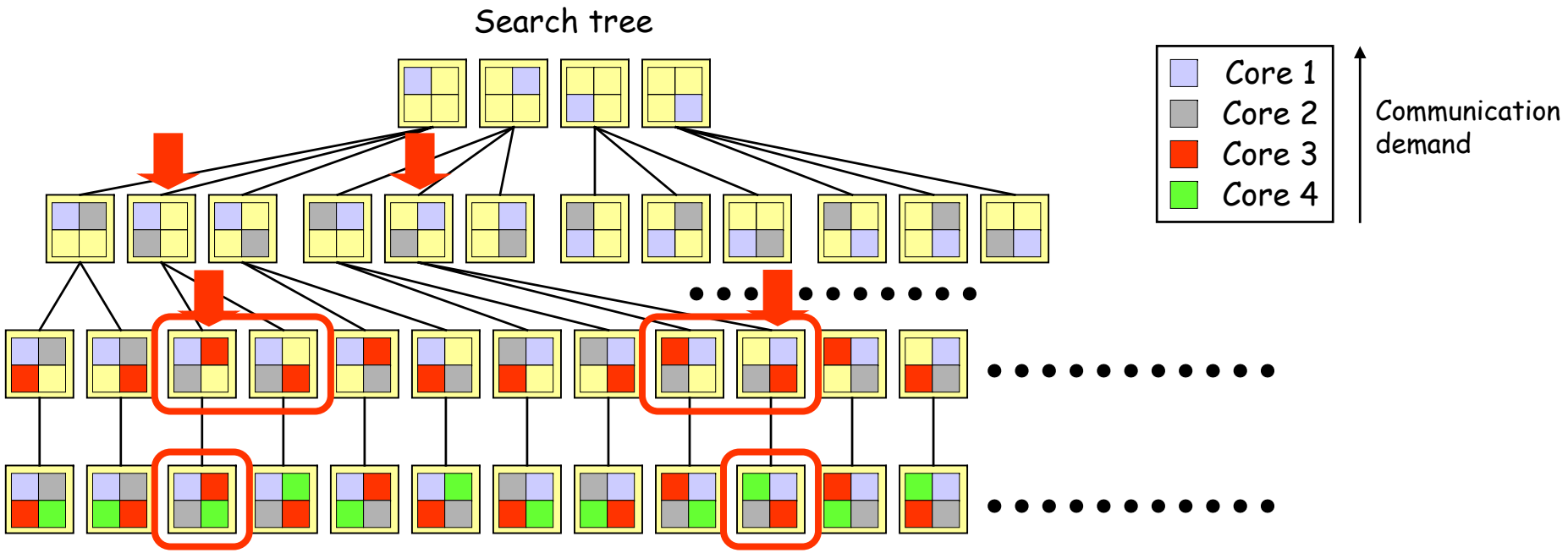
- Genetic operators





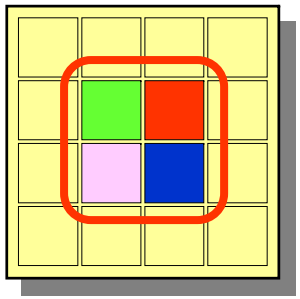
PBBB

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



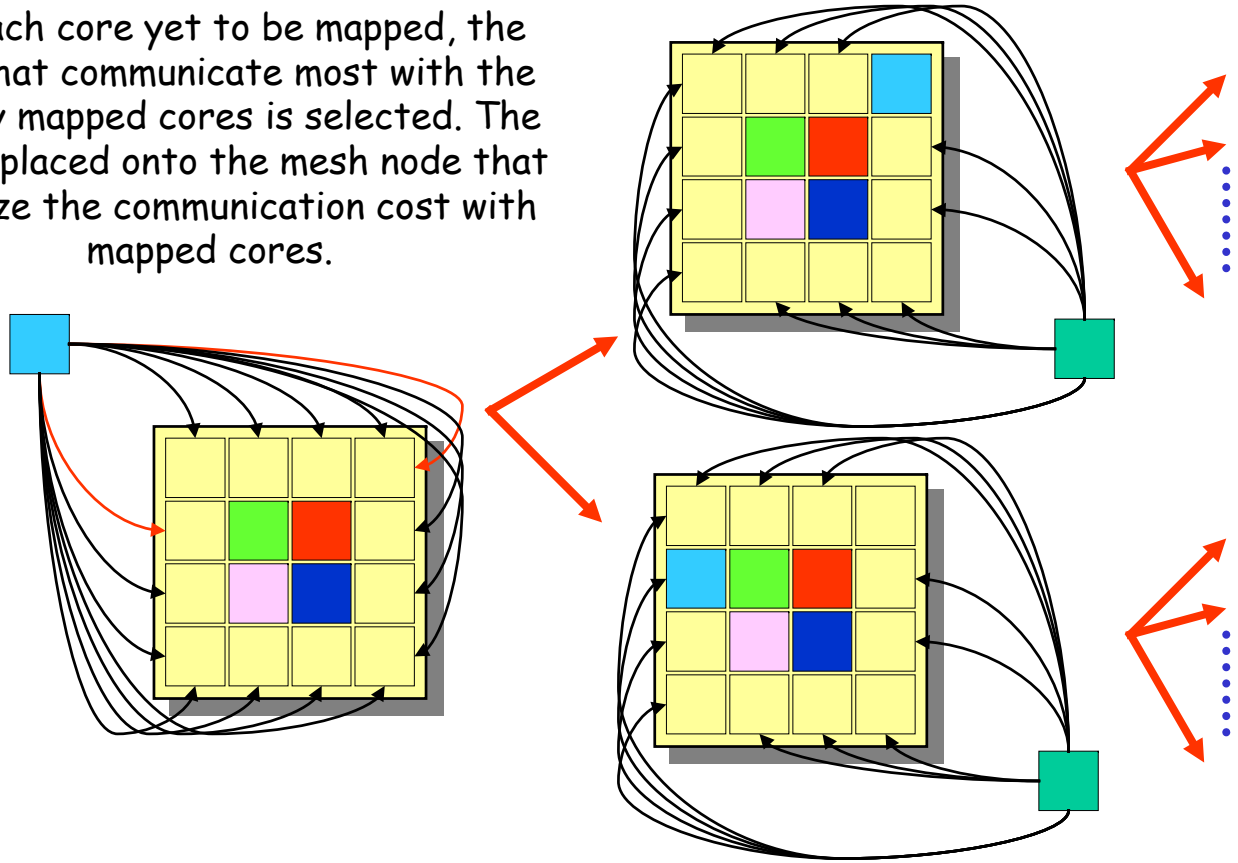
PBNMAP

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

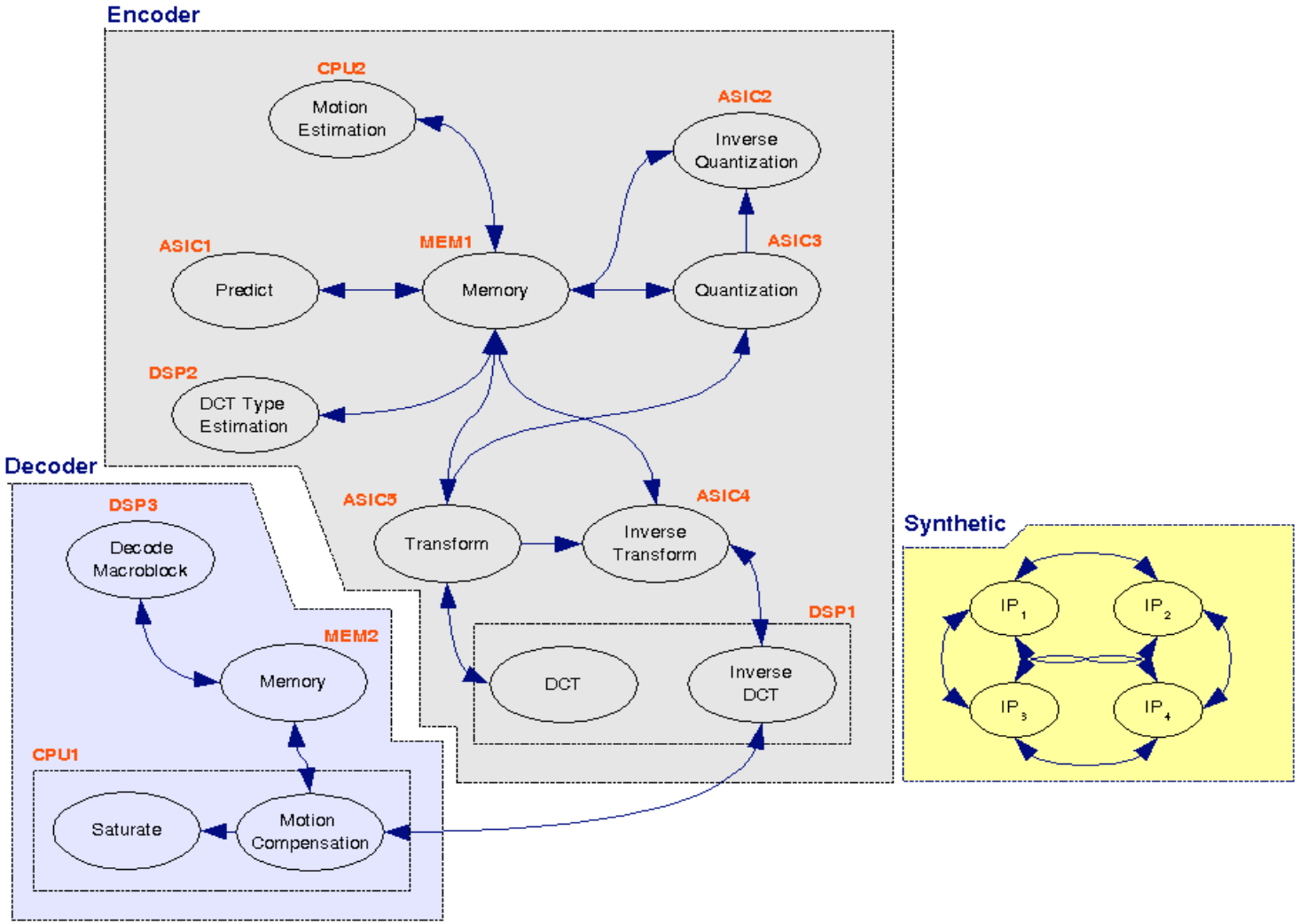


The core that has maximum communication demand is placed onto one of the mesh nodes with maximum number of neighboring

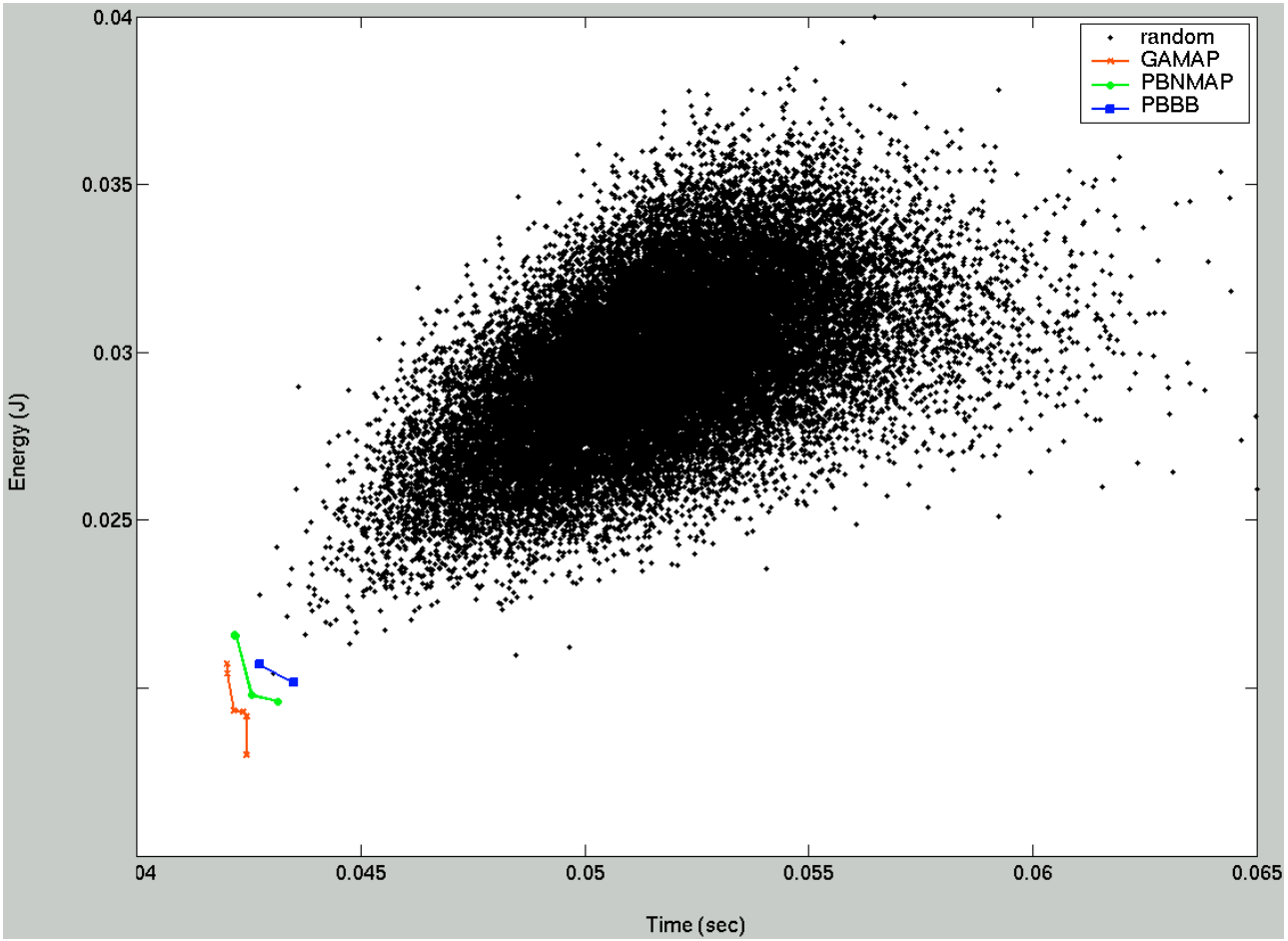
For each core yet to be mapped, the core that communicate most with the already mapped cores is selected. The core is placed onto the mesh node that minimize the communication cost with mapped cores.



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

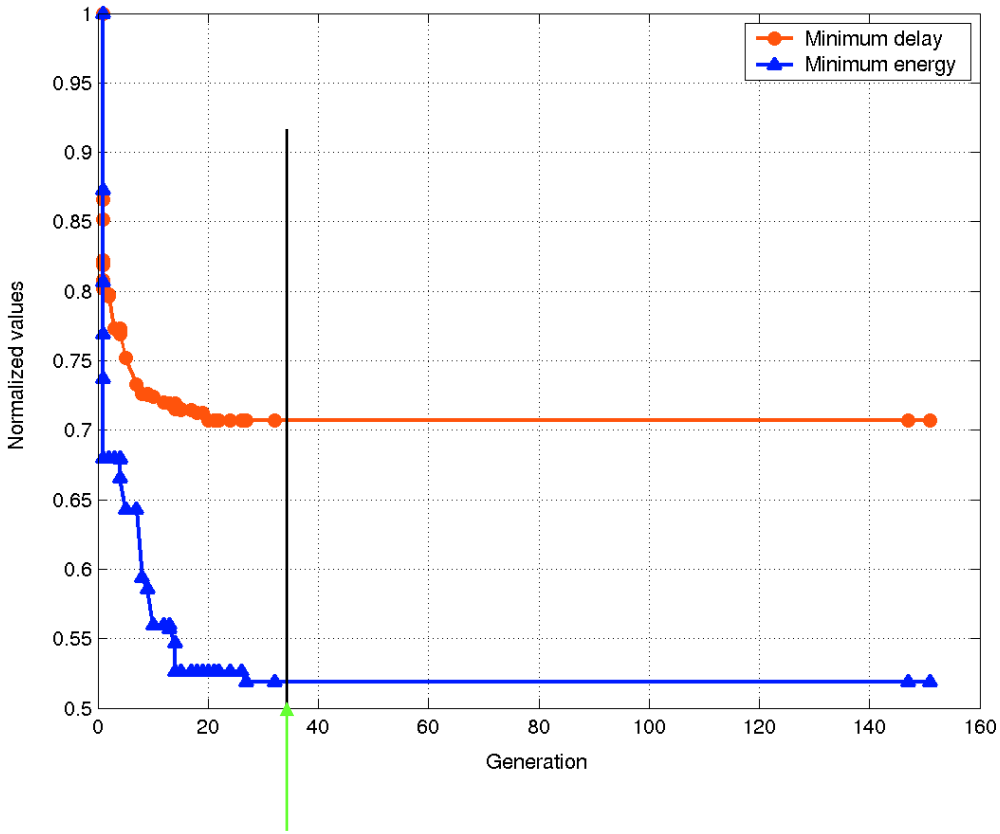
Execution time 41.0 ms
Energy: 20.8 mJ

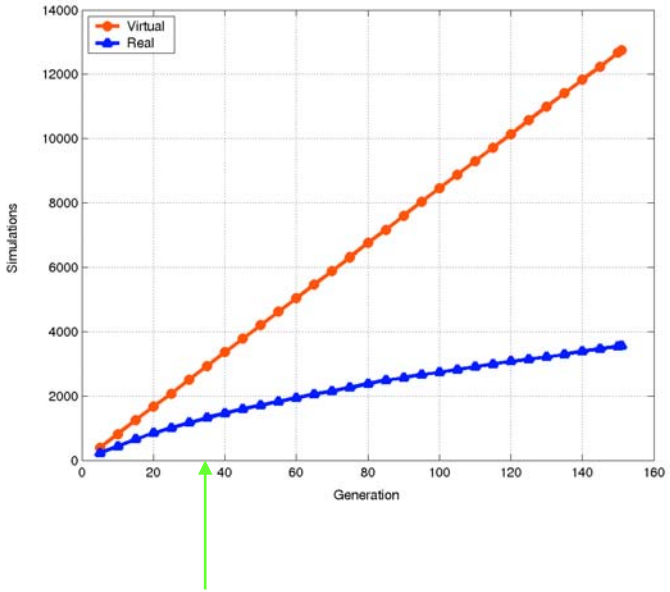
Low energy mapping

	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

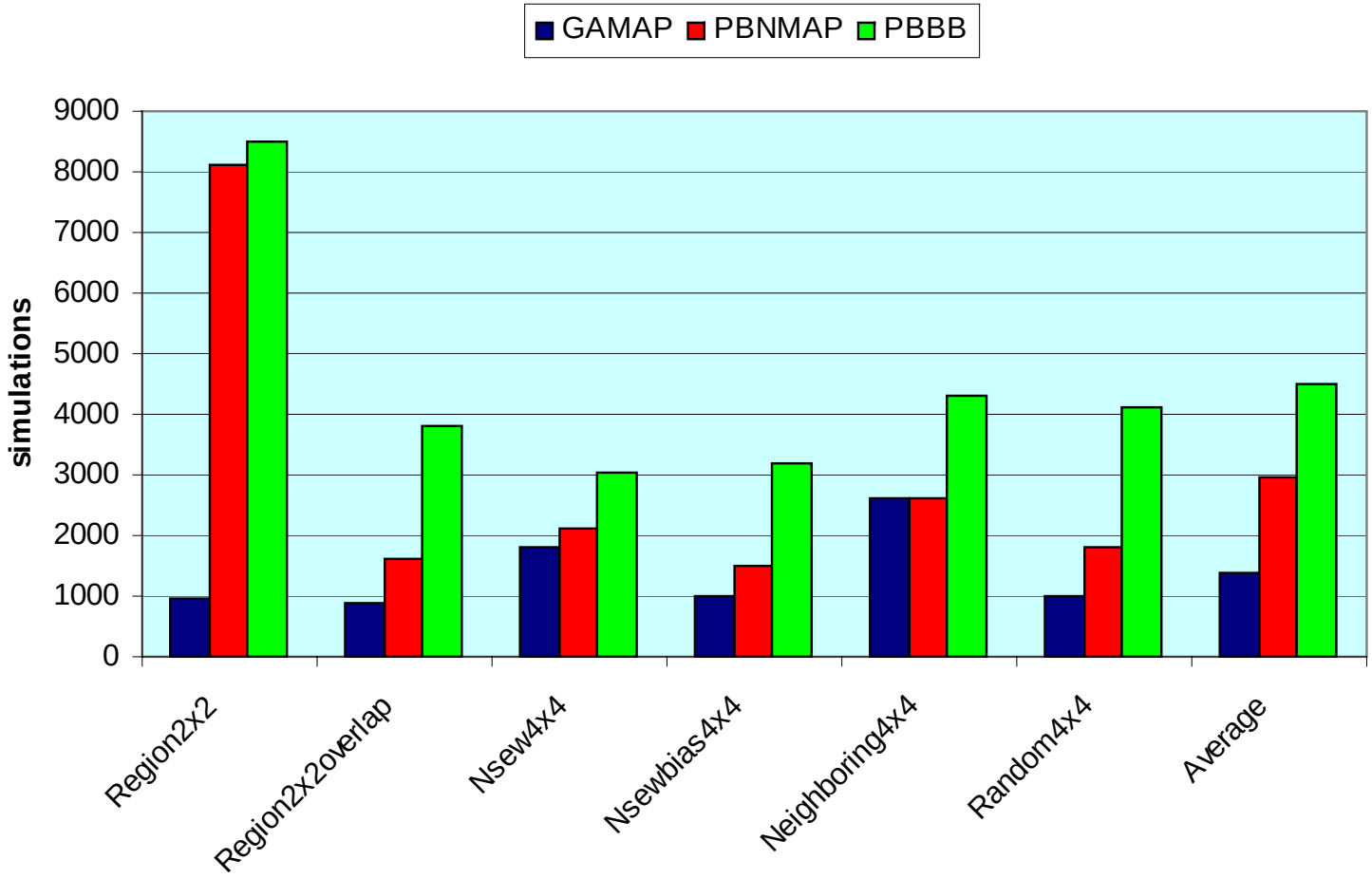




GAMAP	840 sims
PBBB	7238 sims
PBNMAP	2670 sims

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

Experiments
Efficiency

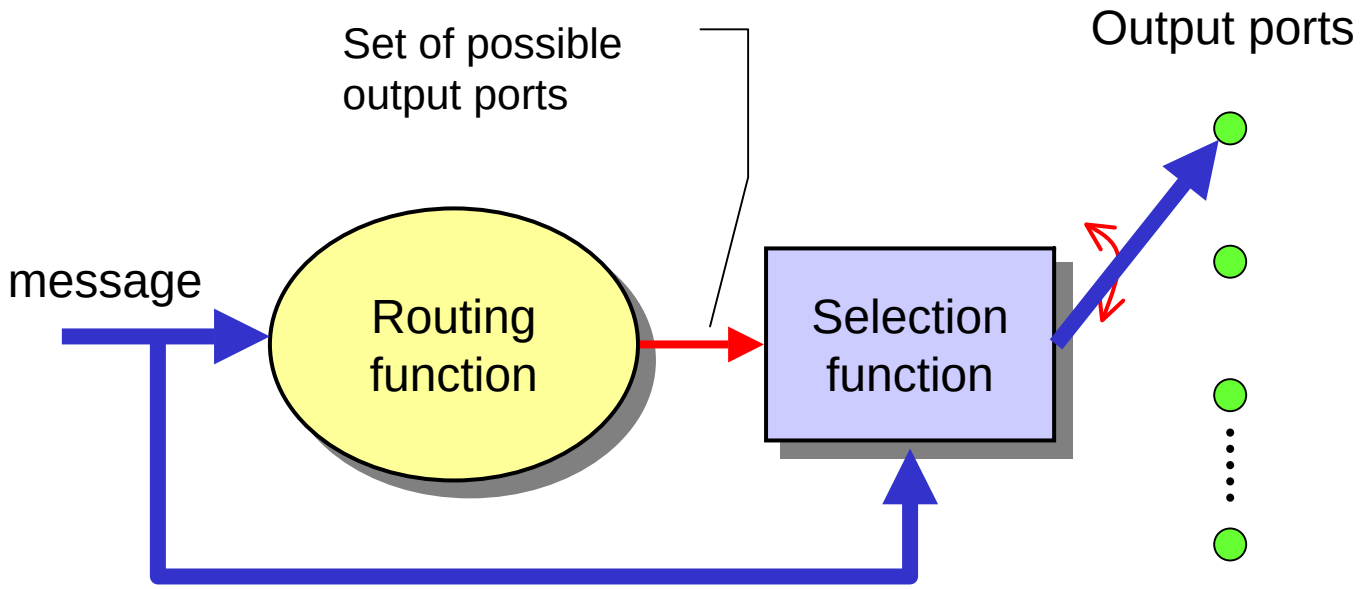


Routing

Routing

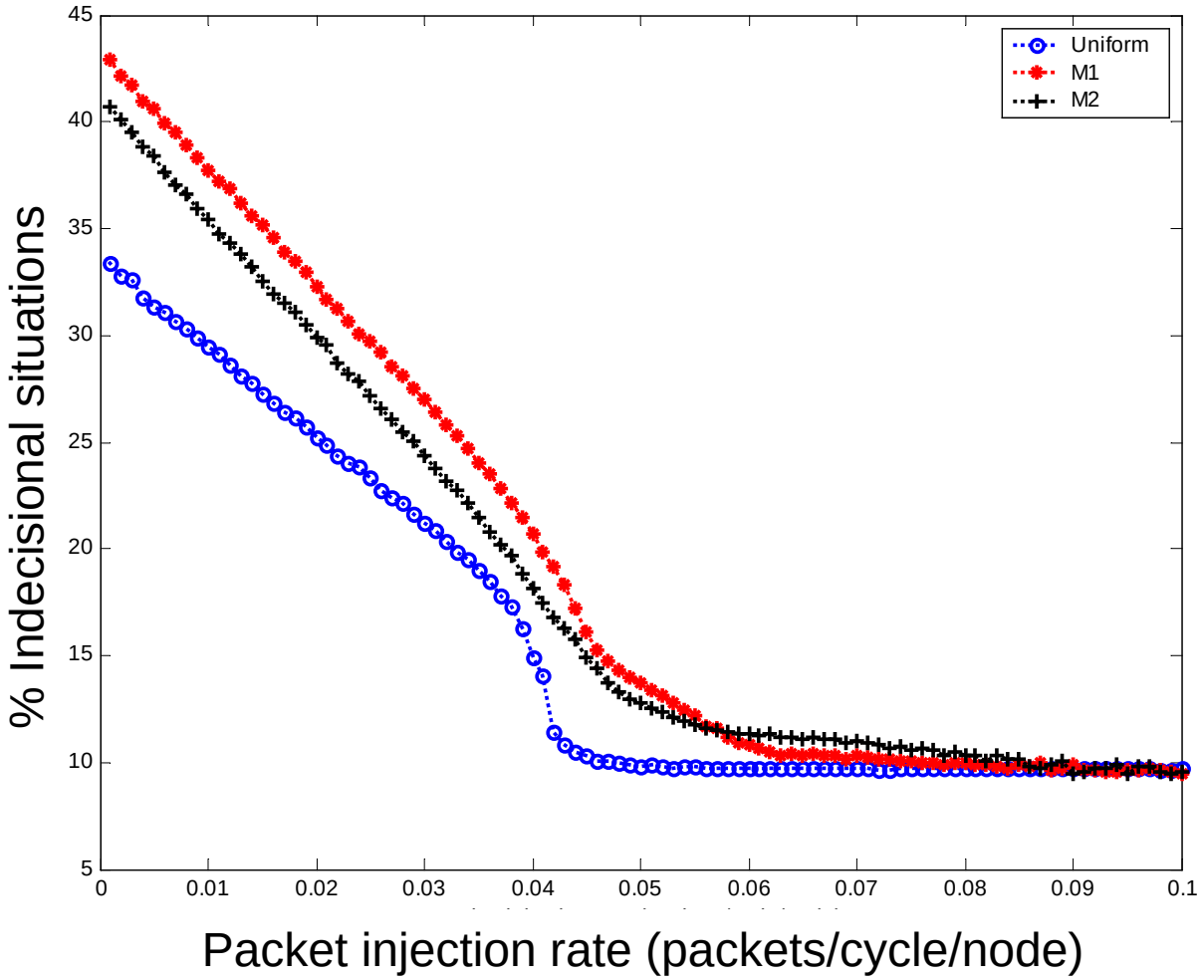
- **Deterministic**
  - ➔ Path completely determined by the source and the destination address
- **Adaptive**
  - ➔ The path taken by a particular packet depends on dynamic network conditions
    - ✓ Fault,
    - ✓ Congestion, …
  - ➔ Routing packets along alternative paths in order to avoid congested areas

Routing and Selection

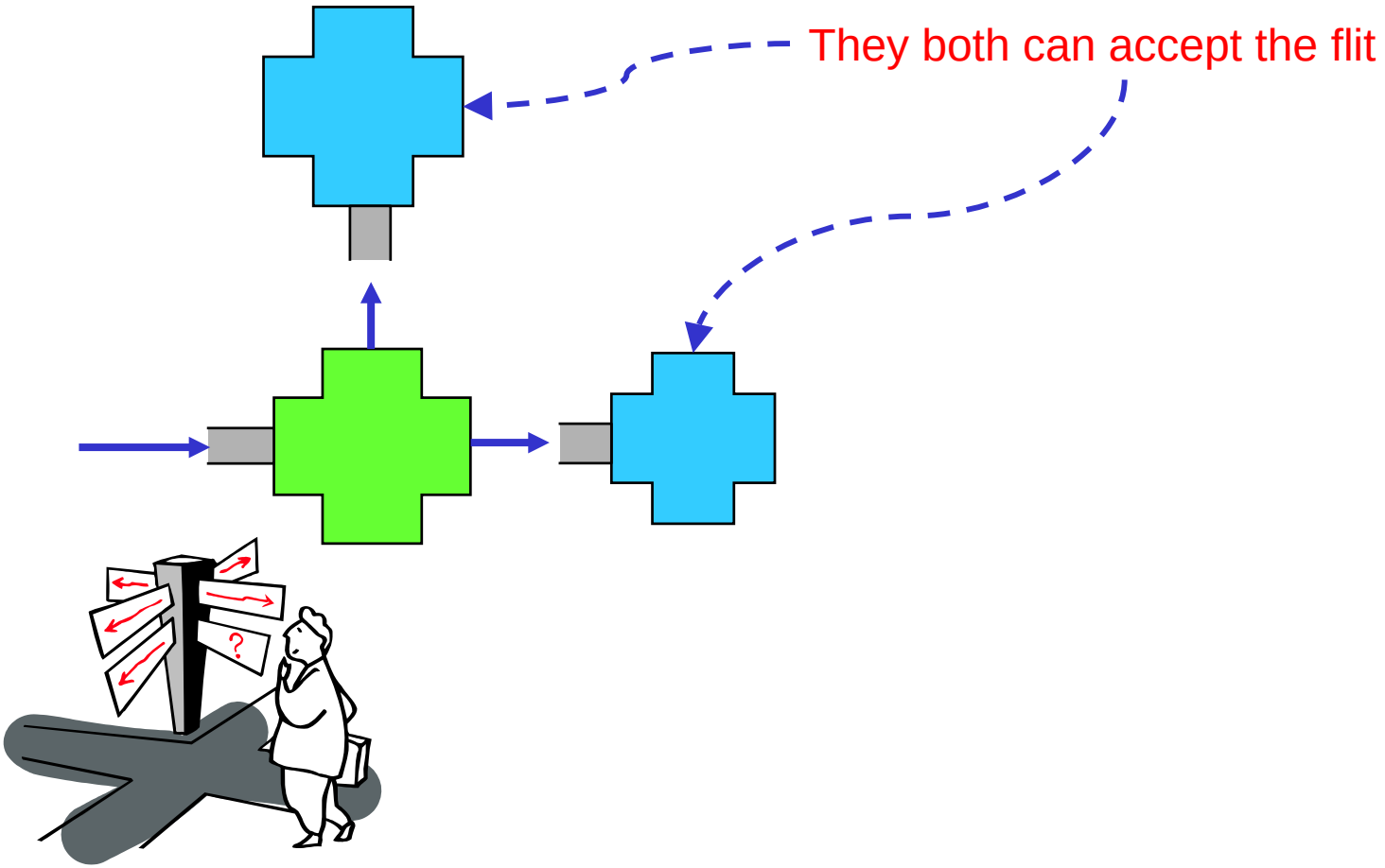


Neighbors on Path Congestion-Aware Routing

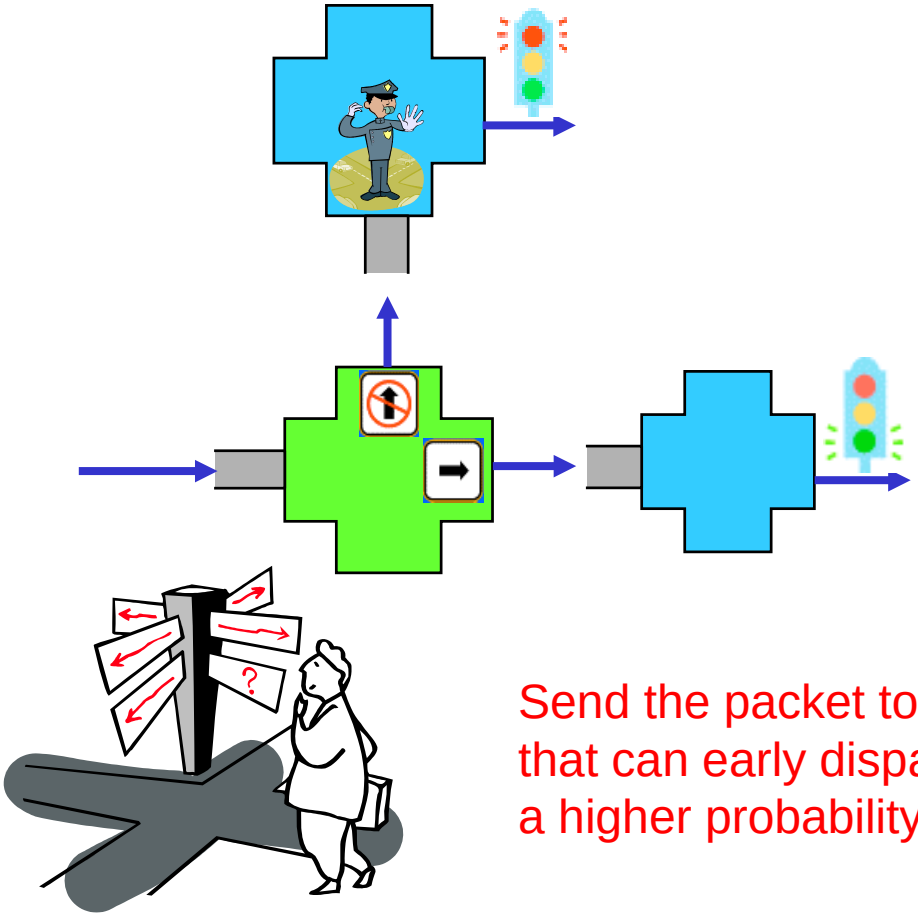
Situations of Indecision



Selection Problem

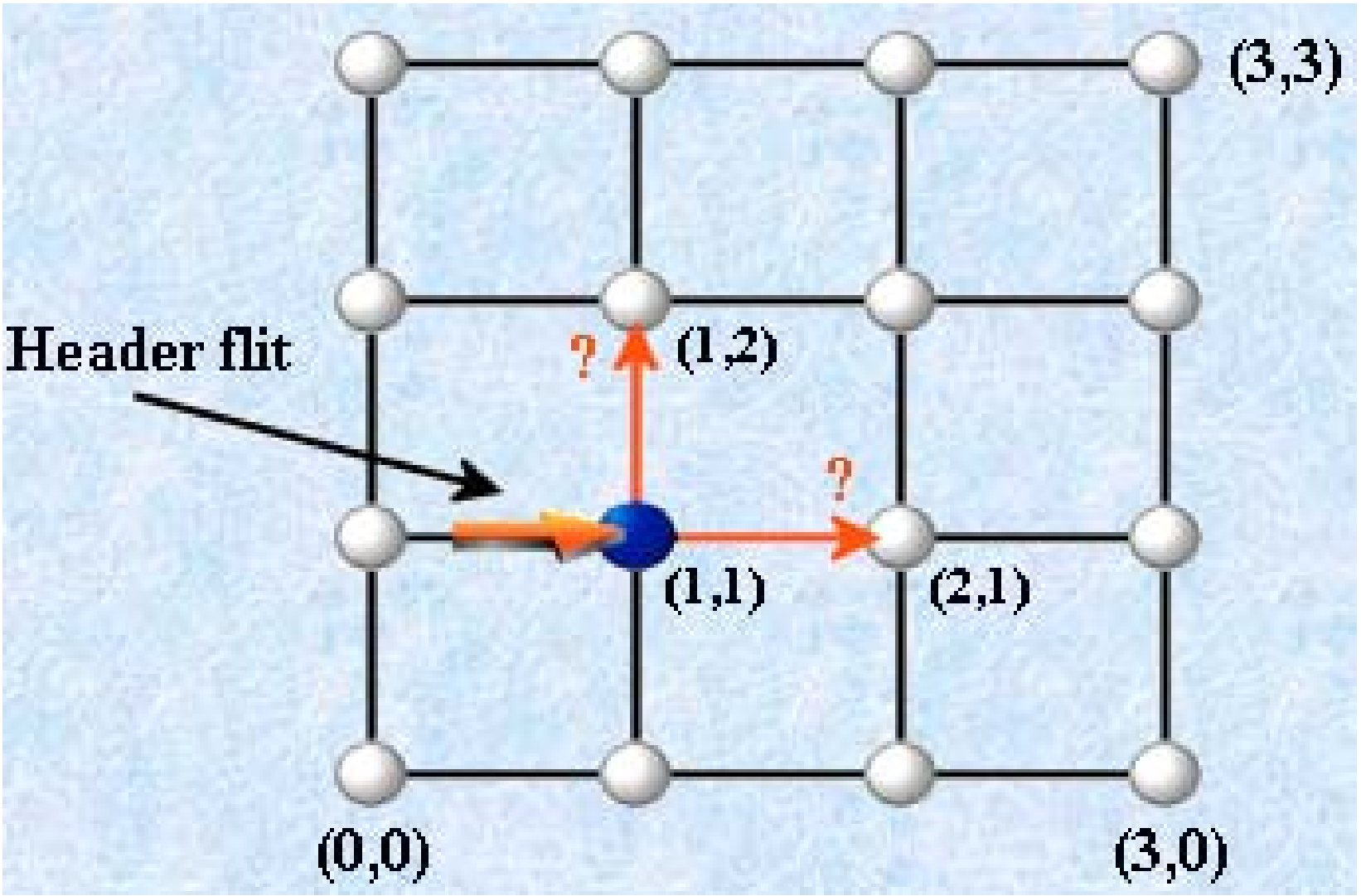


Selection Problem

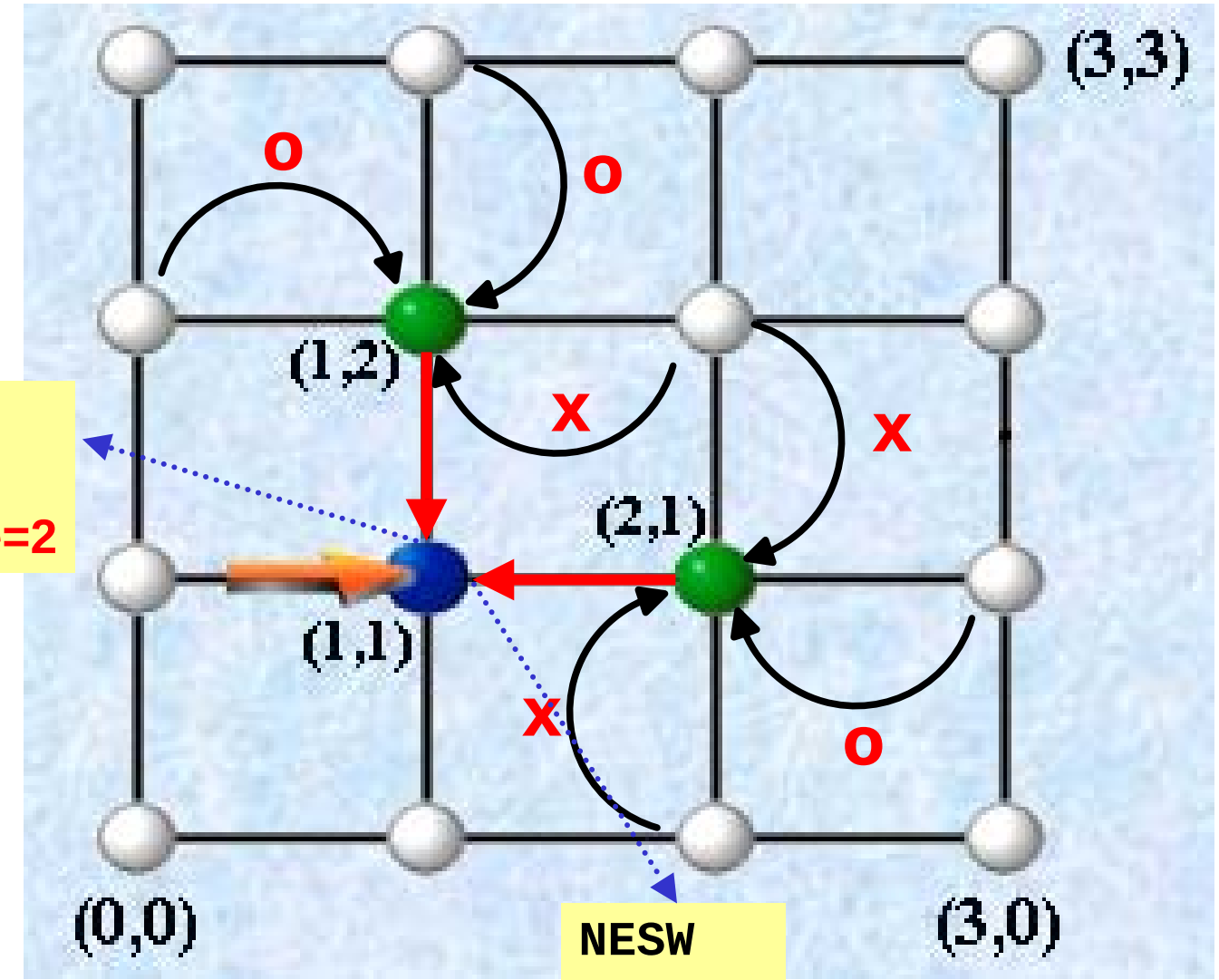


Send the packet to the node
that can early dispatches it with
a higher probability

NoPCAR Algorithm



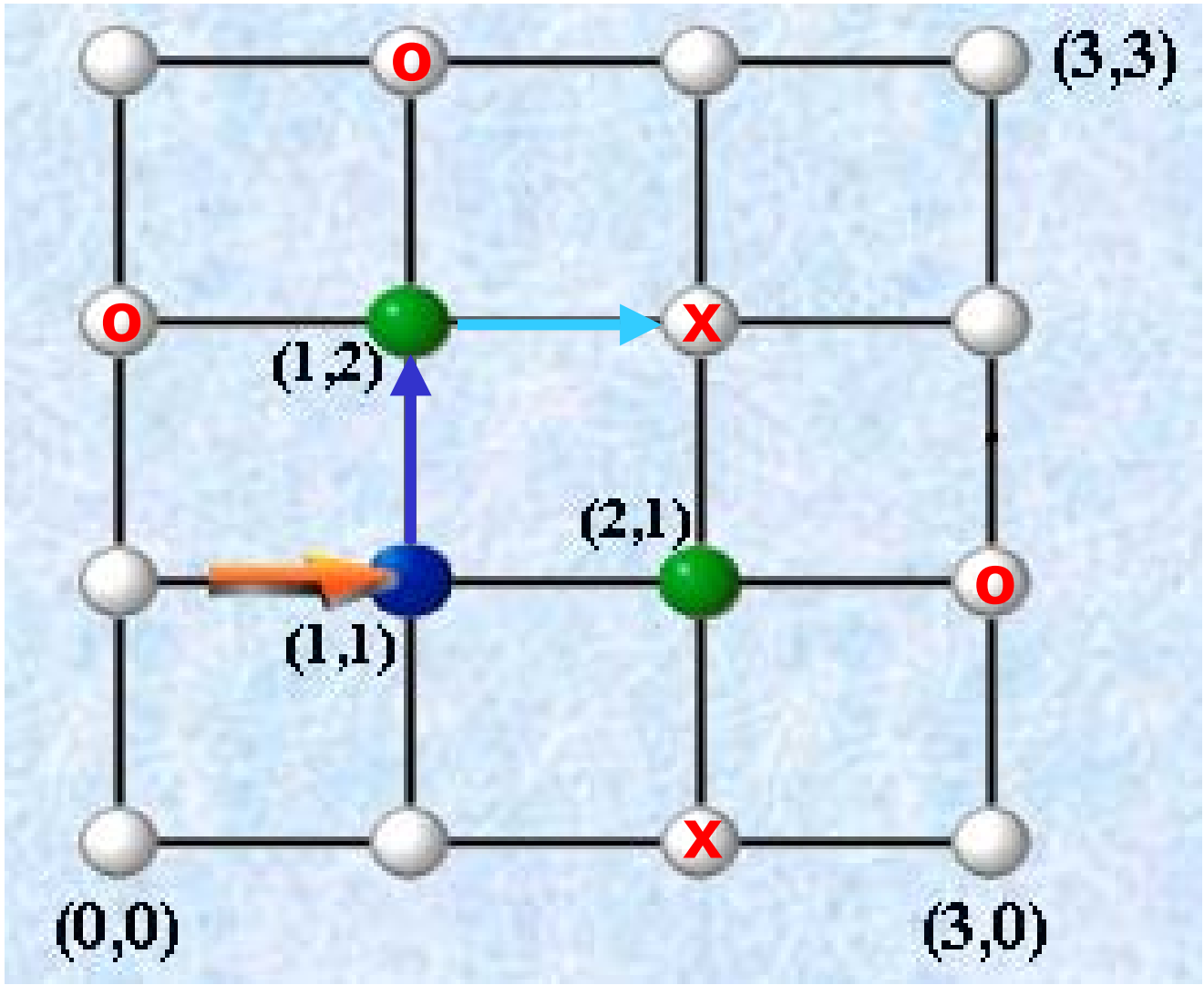
NoPCAR Algorithm



x Not available

o Available

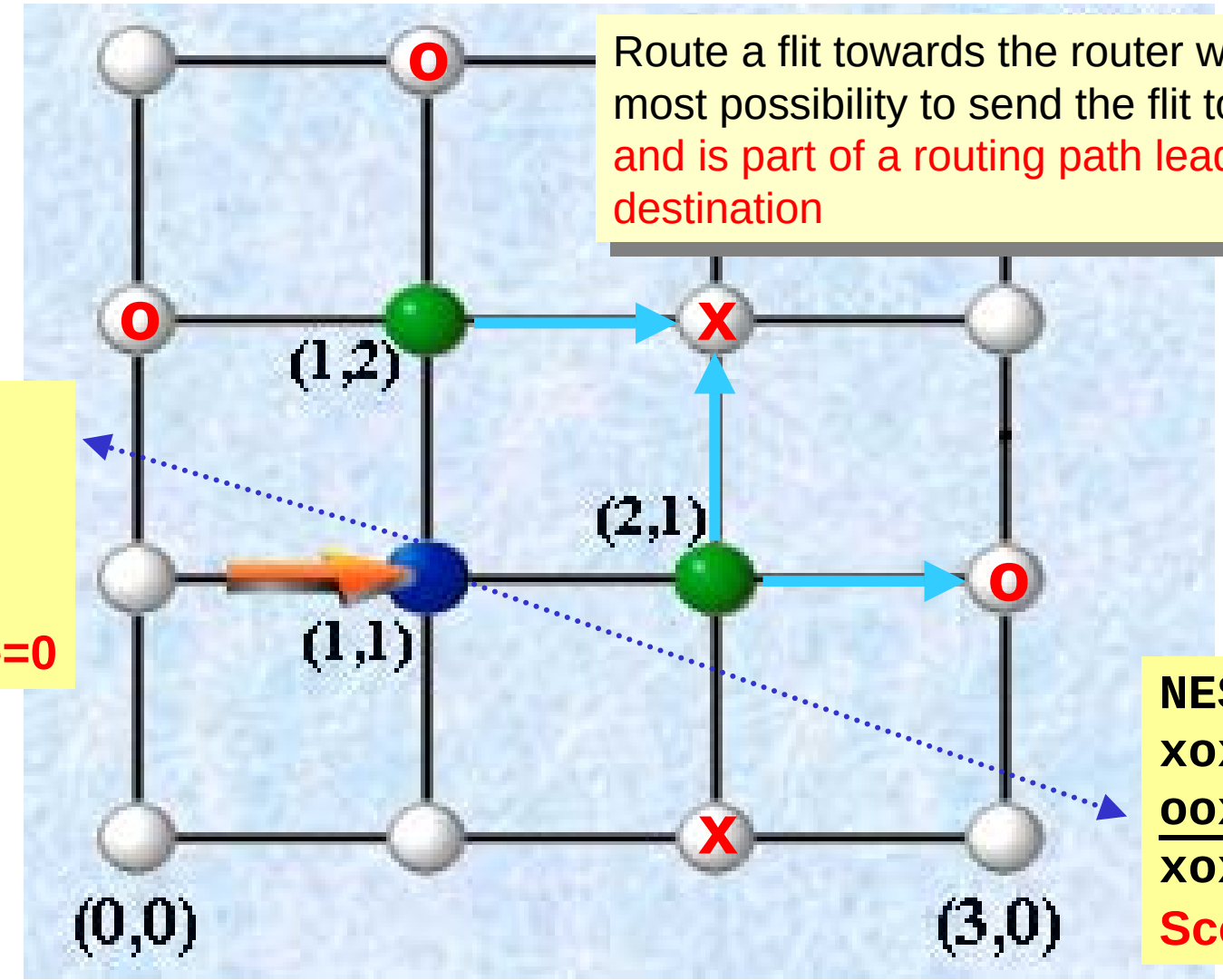
NoPCAR Algorithm



x Not available

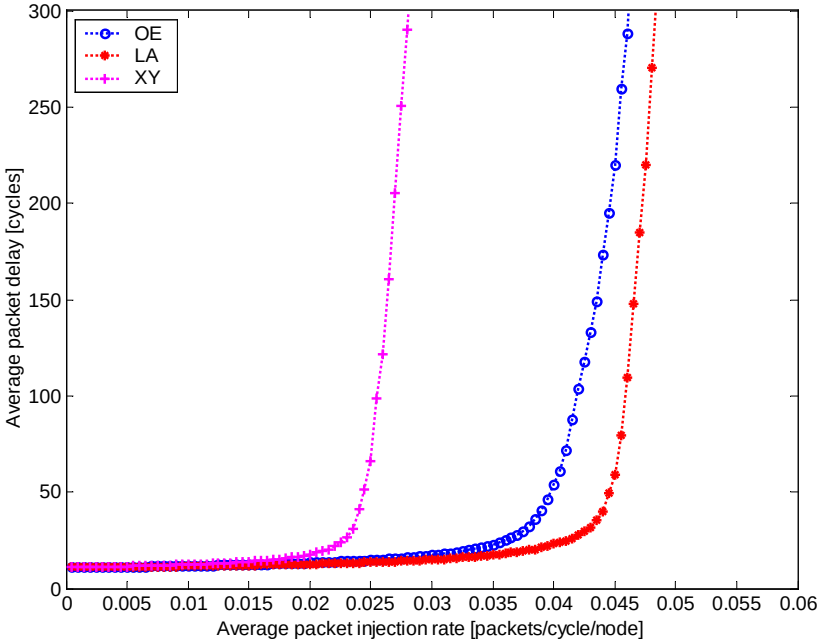
o Available

NoPCAR Algorithm

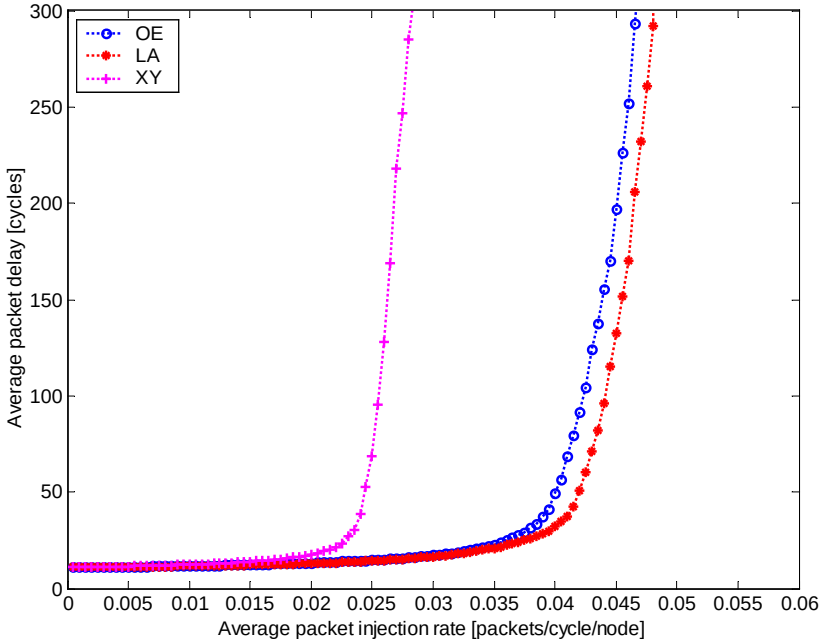


M1/M2 Traffic

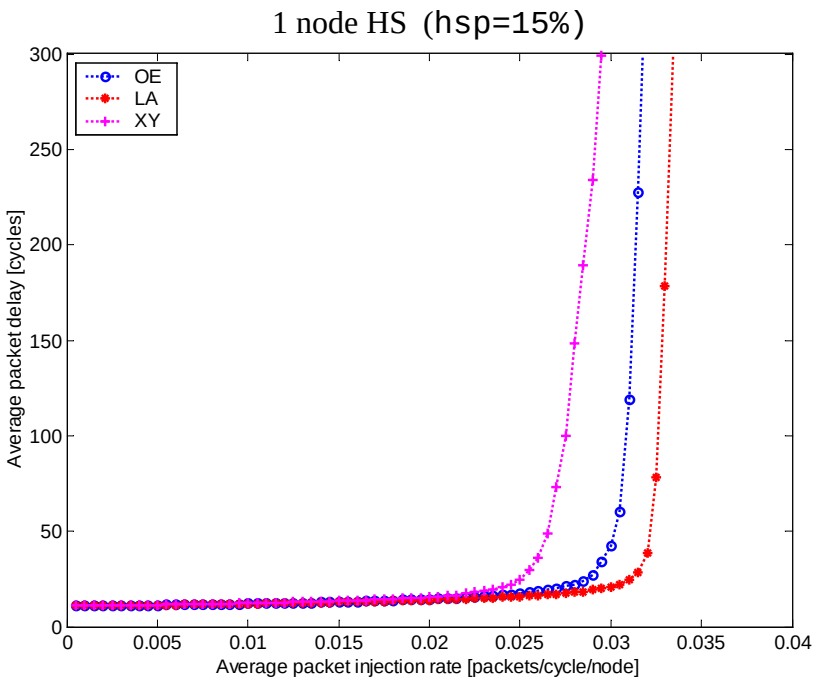
M1: (i,j) → (N-1-j,N-1-i)

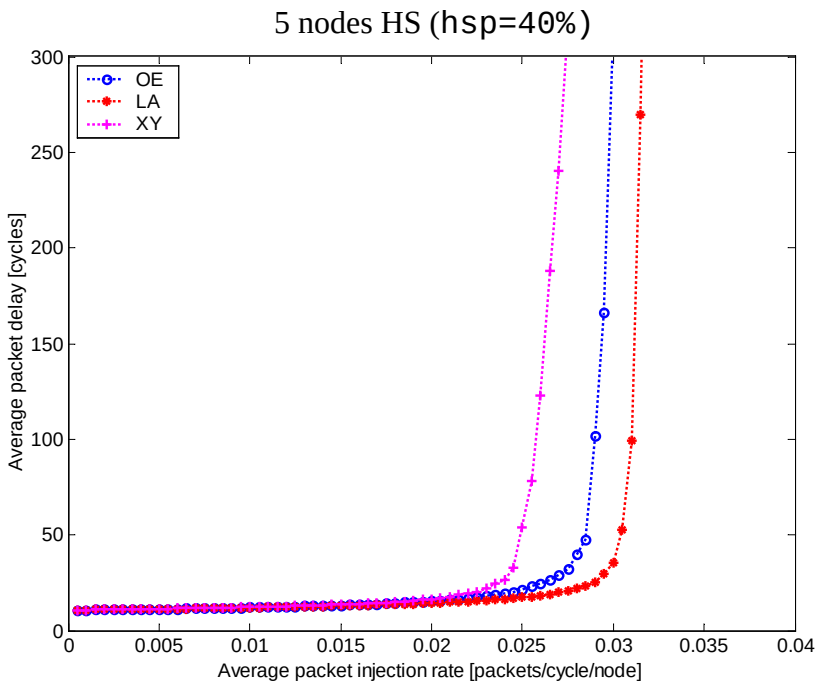


M2: (i,j) → (j,i)



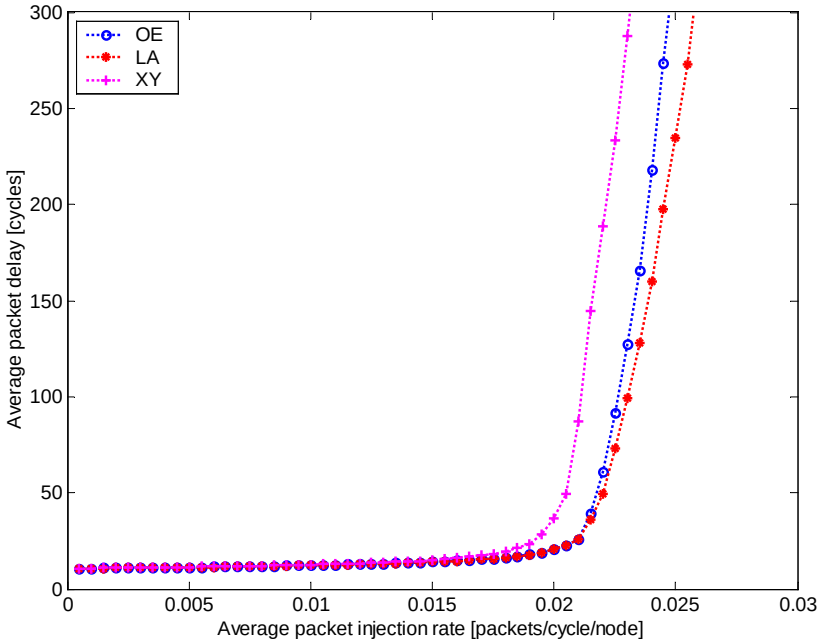
Hot-Spot Traffic (1/2)



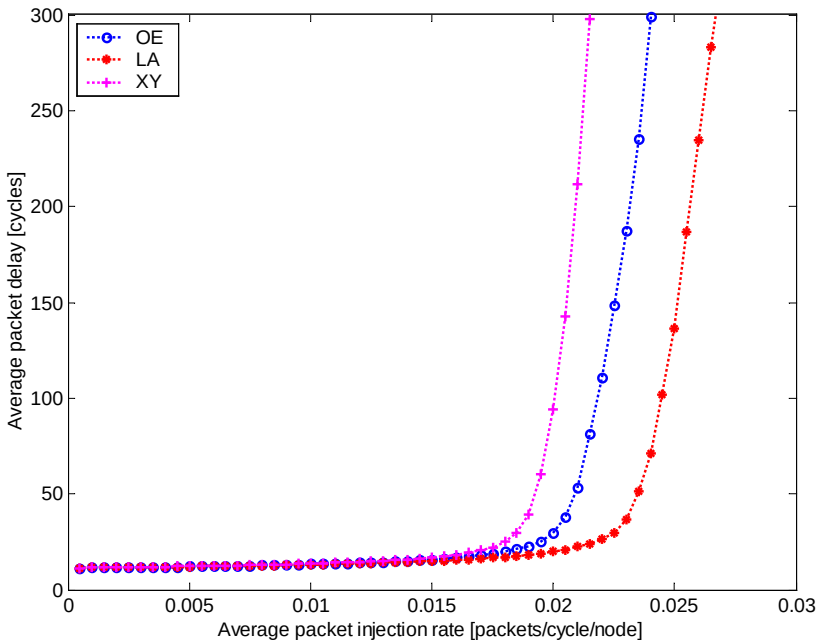


Hot-Spot Traffic (2/2)

(2,2) ⇔ (3,3)

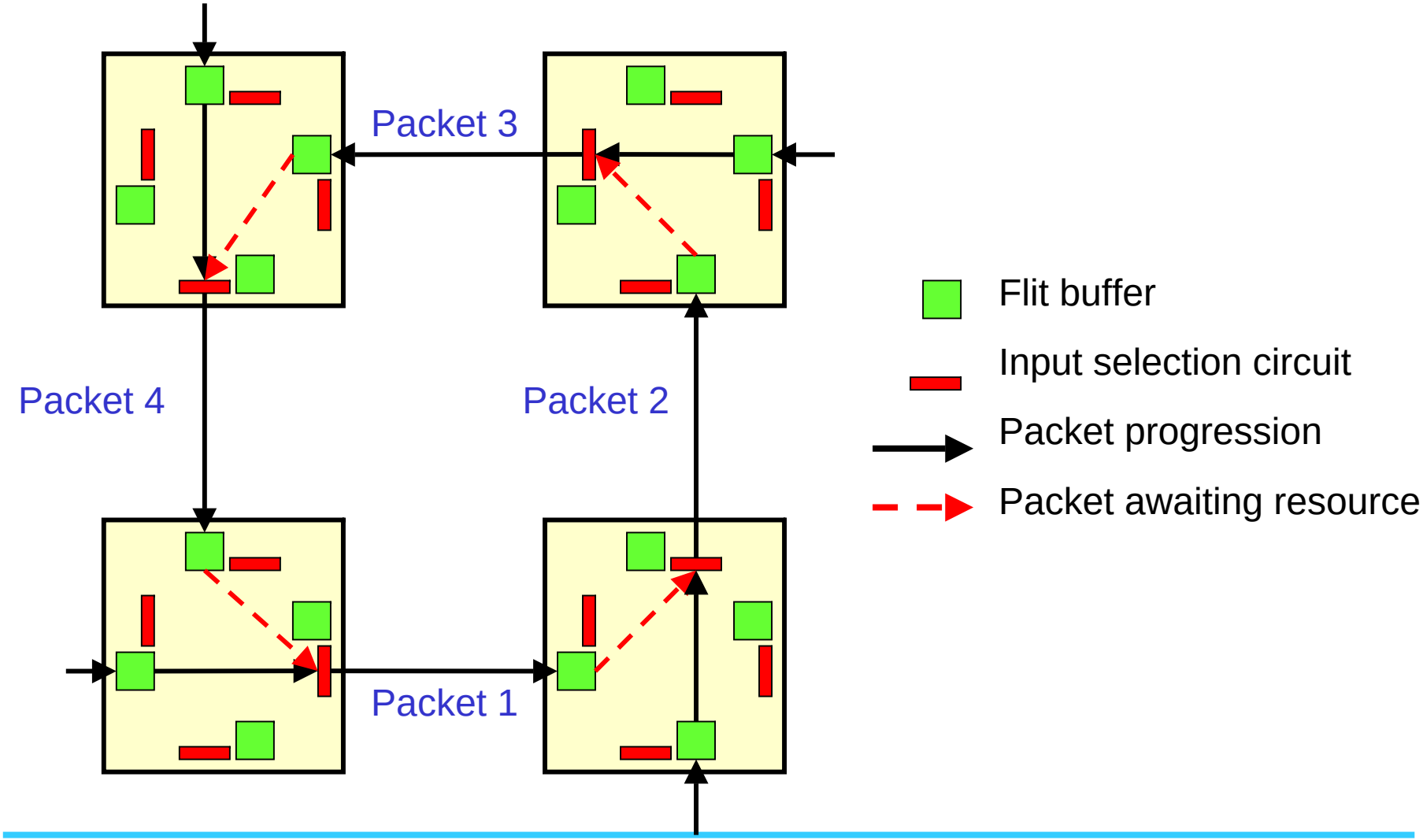


(1,1) ⇔ (4,4)



Application Specific Routing

Wormhole Routing & Deadlock

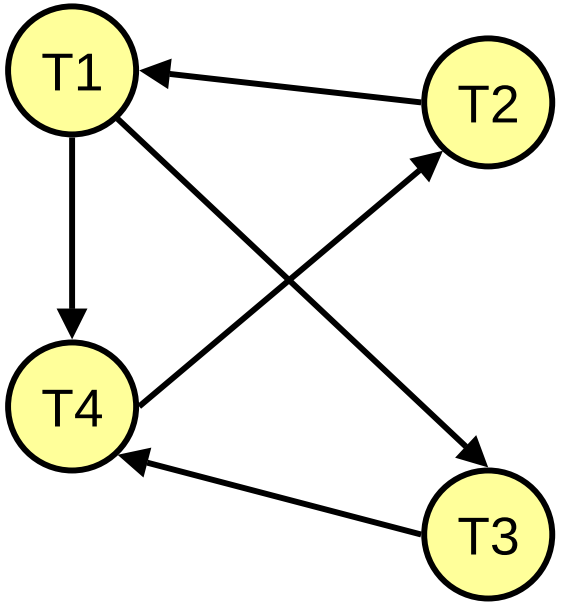


Duato's Theorem

- A routing function R is *deadlock-free* if there are no cycles in its channel dependency graph
  - ➔ There is a *Direct Dependency* from l_i to l_j if l_j can be used immediately after l_i by messages destined to some node *n*

Motivation

- A routing function must guarantees that
  - ➔Every pair of nodes can communicate
- Knowlege of the communication traffic
  - ➔Not all cores communicate directly
  - ➔Why do not use this information?
  - ➔A routing function must guarantees that every pair of communicating nodes can communicate

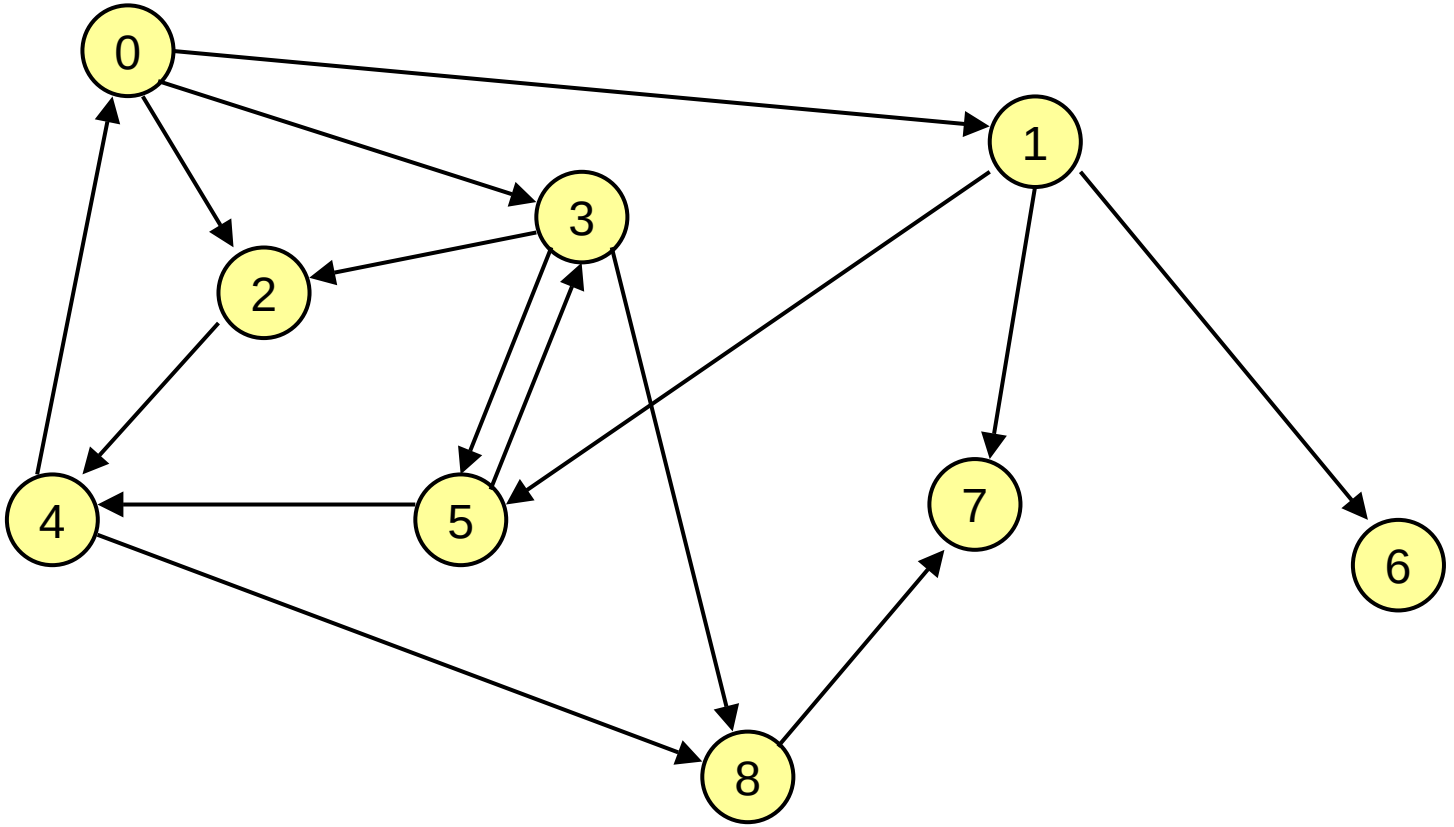


Theorem

- A routing function R is *deadlock-free* if there are no cycles in its application specific channel dependency graph
  - ➔ There is a *Application Specific Direct Dependency* from l_i to l_j if l_j can be used immediately after l_i by messages destined to some node *n* **AND** exist at least one communication that uses l_i and l_j

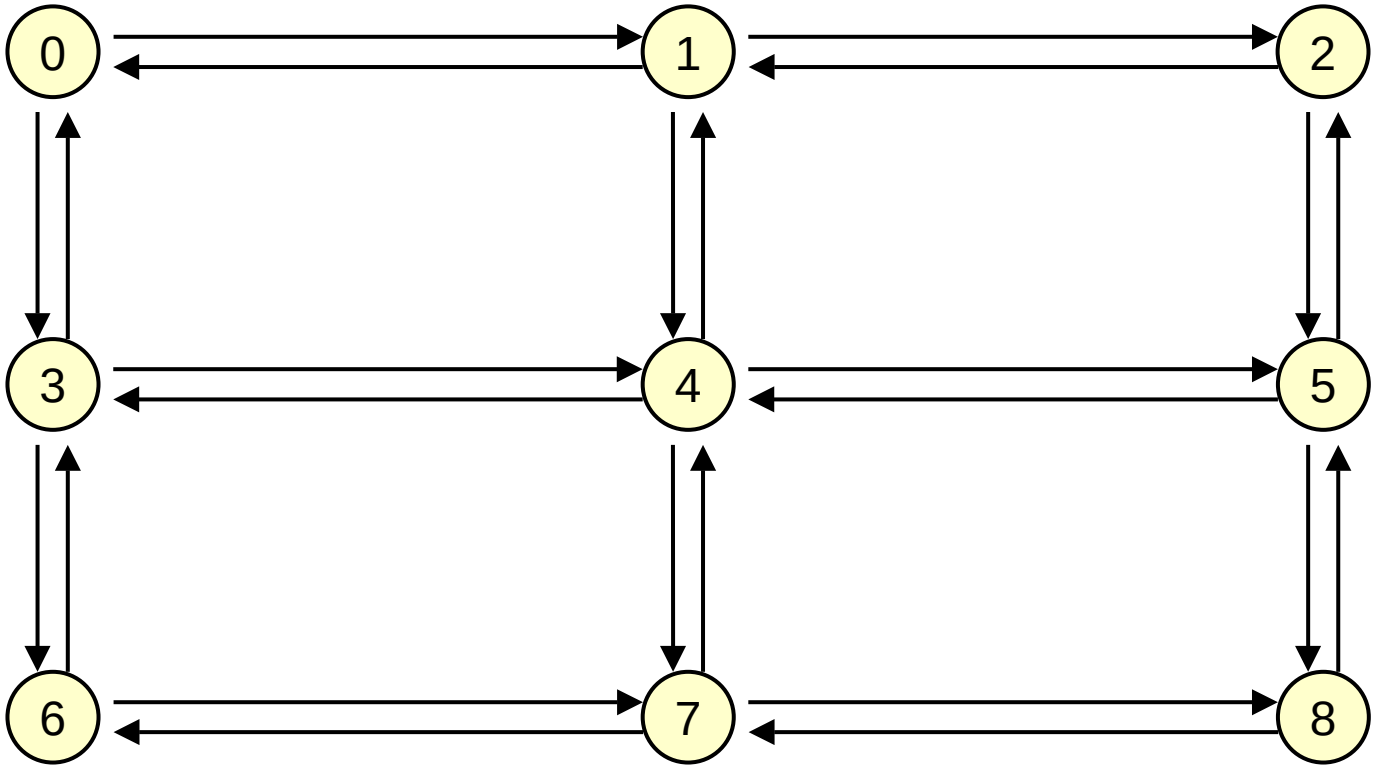
Example

Communication Graph

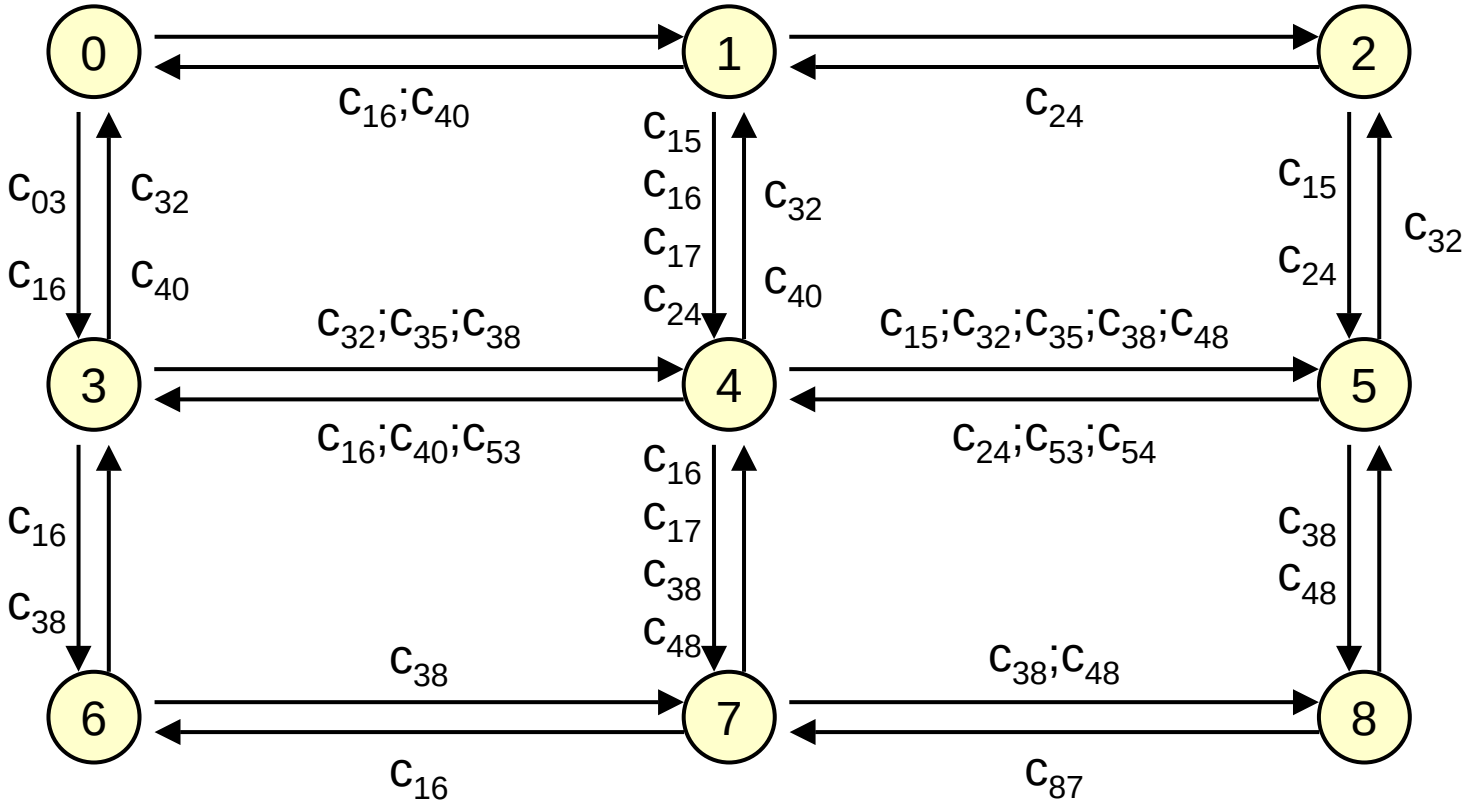


Example (cnt'd)

Topology Graph

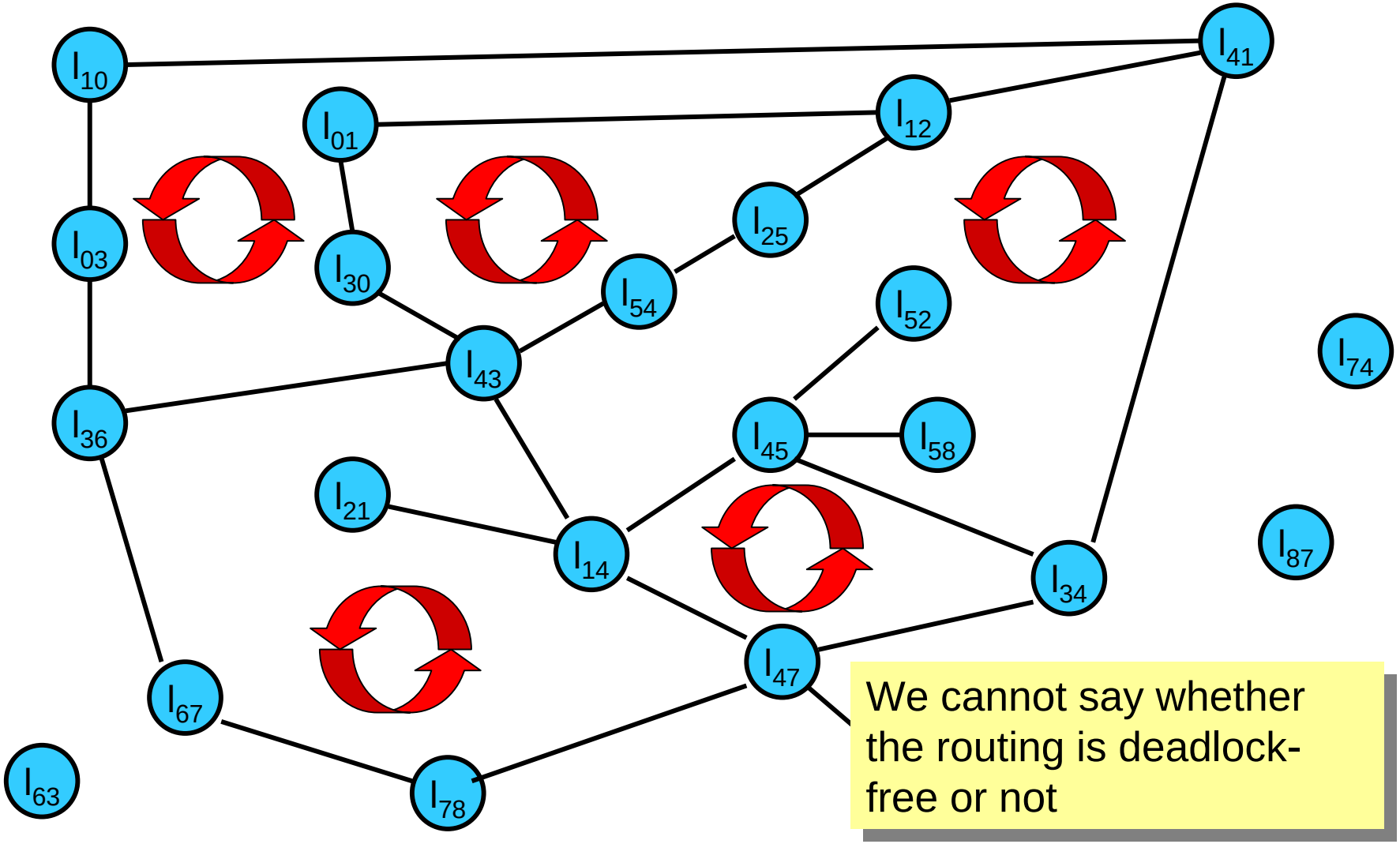


Example (cnt'd)



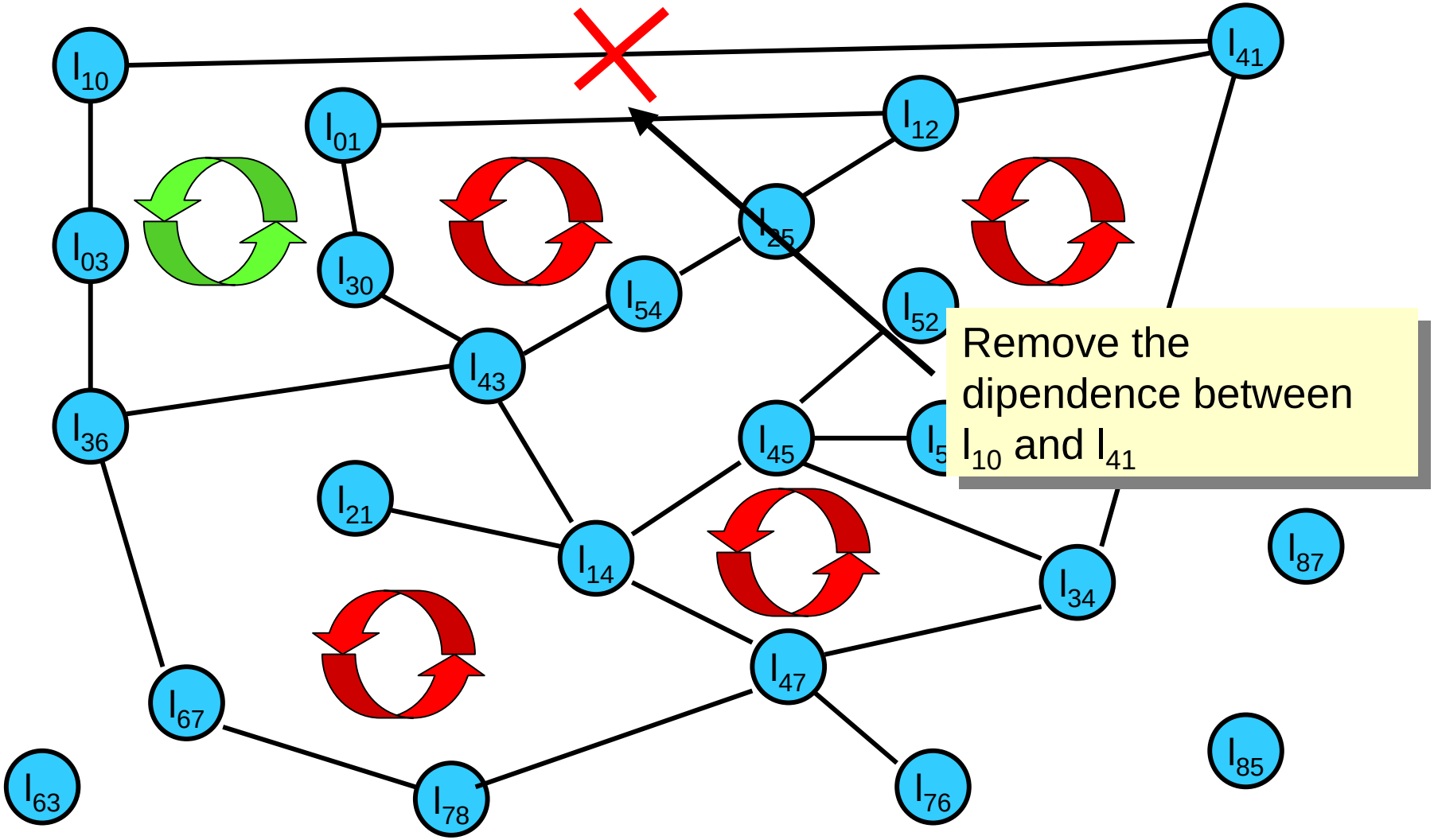
Pass through functionale for a minimal fully adaptive routing

Example (cnt'd)



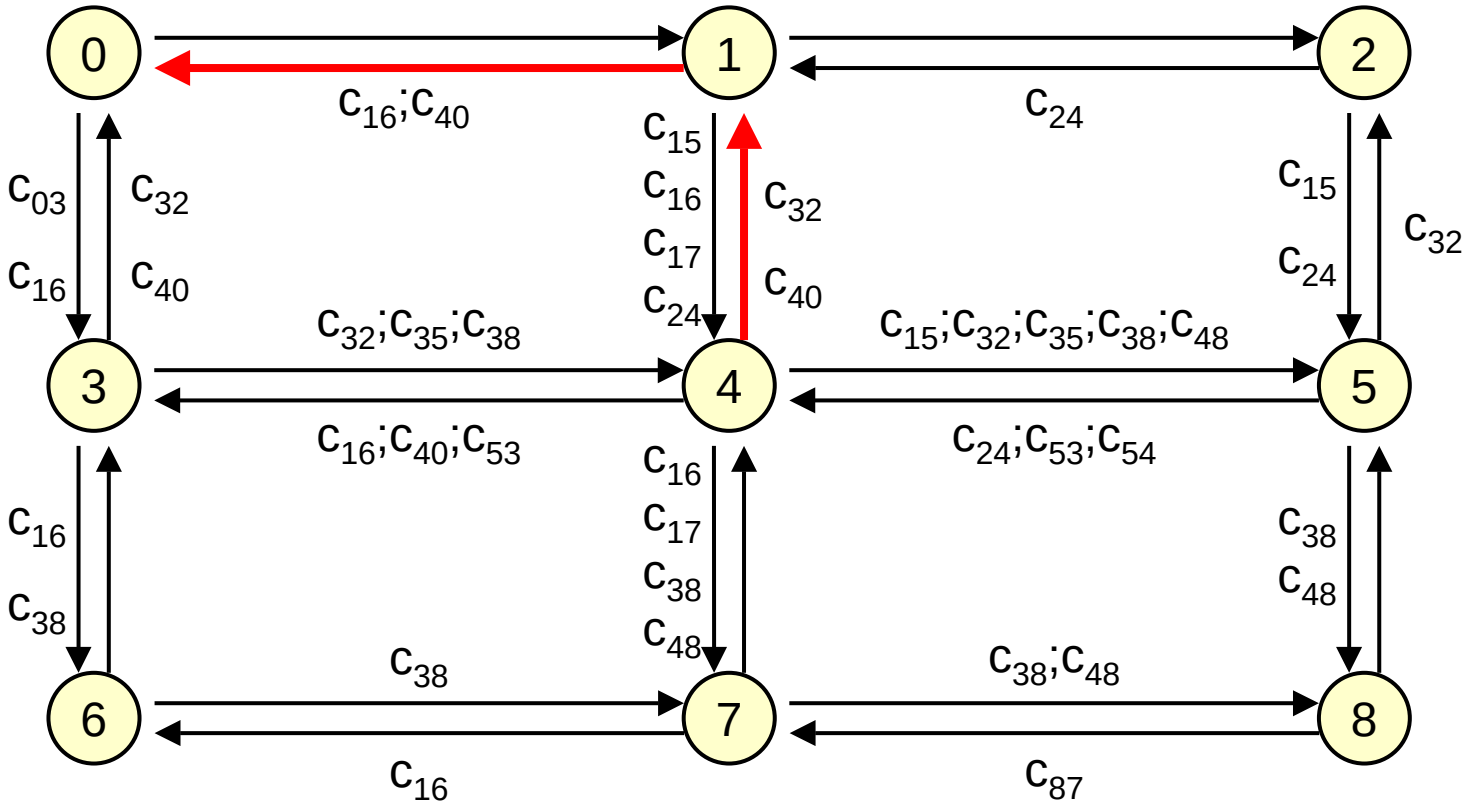
We cannot say whether the routing is deadlock-free or not

Example (cnt'd)



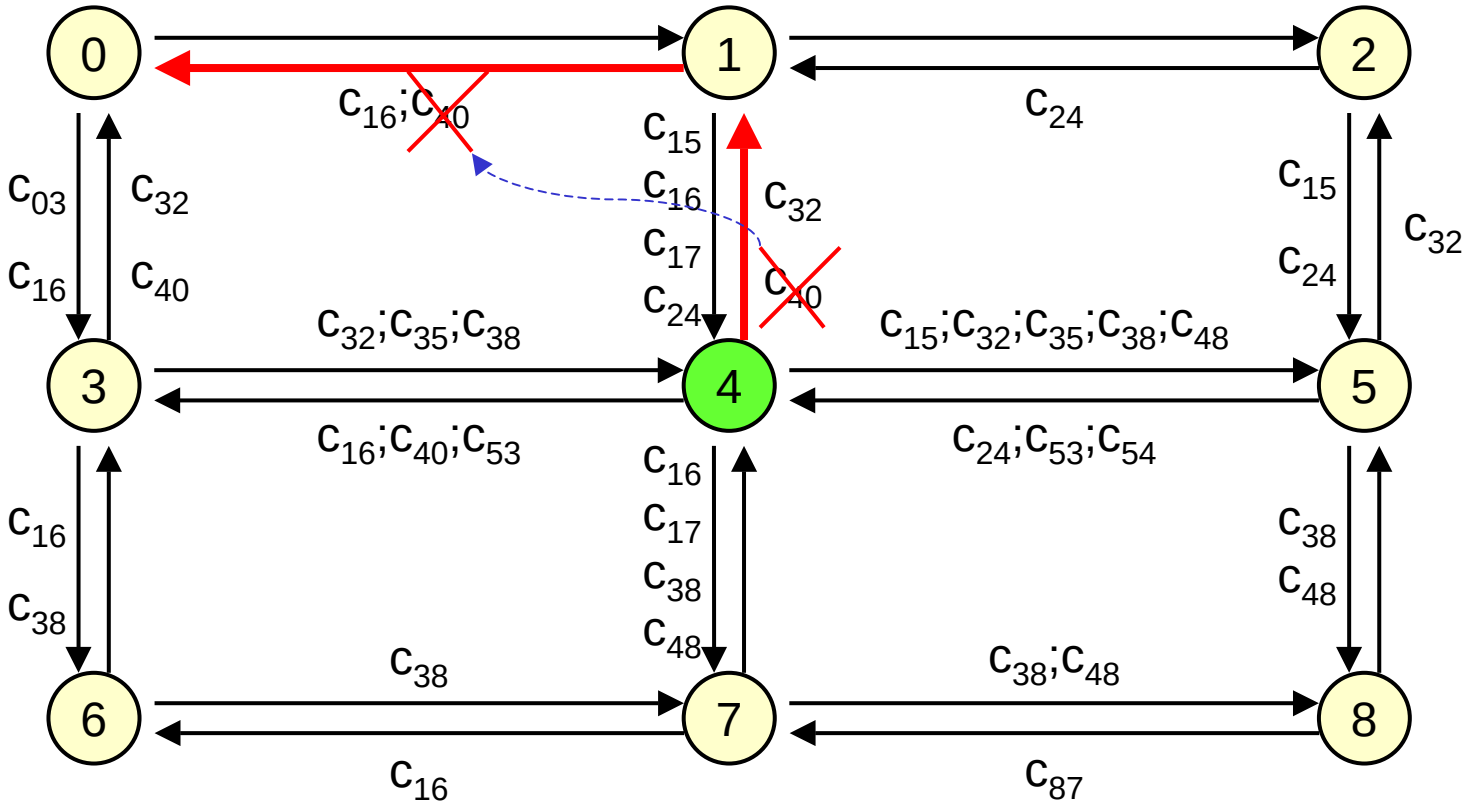
Example (cnt'd)

The dependence between l_{10} and l_{41} is due exclusively to c_{40} (communication from PE 4 to PE 0)



Example (cnt'd)

We can remove such a dependency simply by restricting the routing function for node 4 ⇒ Negative impact on adaptivity.



Observations

- Topological mapping has a great impact on deadlock-freeness
  - ➔ We can remove application specific dependencies simply mapping the PEs differently
    - ✓ No impact on adaptivity
- How to break the cycles of the ASDG?
  - ➔ Optimization metrics
    - ✓ Degree of adaptivity
    - ✓ Size of routing tables
    - ✓ …
  - ➔ Respect of the CG

Outline

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