Communication-Based Mapping Using Shared Pages

Matthias Diener, Eduardo H. M. Cruz, Philippe O. A. Navaux

Federal University of Rio Grande do Sul, Porto Alegre, Brazil
{*mdiener,ehmcruz,navaux*}*@inf.ufrgs.br*

`http://inf.ufrgs.br/~mdiener`

IPDPS 2013 – Boston, MA
May 22, 2013

INSTITUTO DE INFORMÁTICA UFRGS

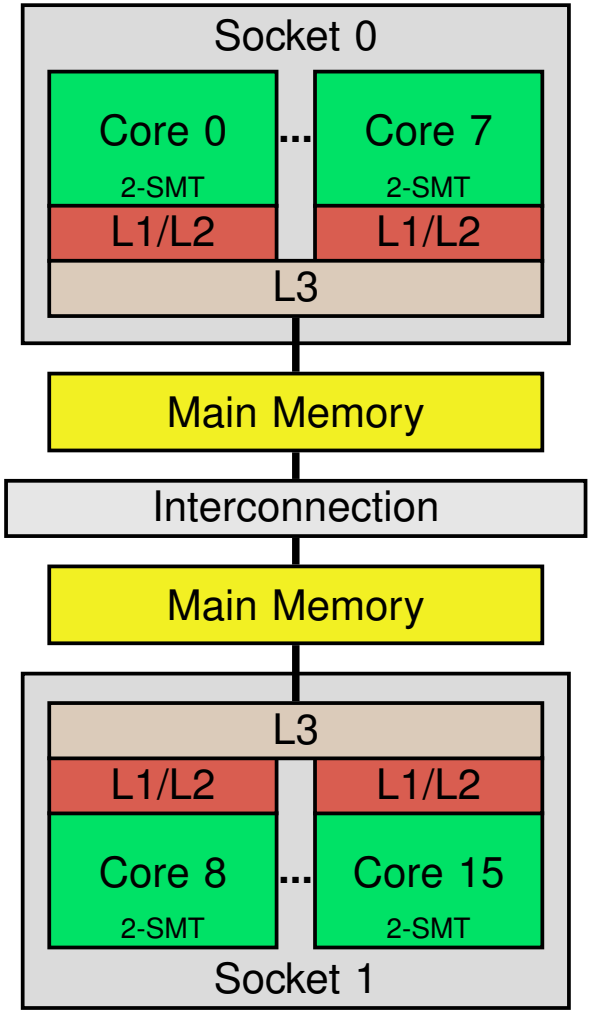
WHAT IS COMMUNICATION-BASED MAPPING?

WHAT IS COMMUNICATION-BASED MAPPING?

- Computer systems have complex memory hierarchies, influence communication efficiency

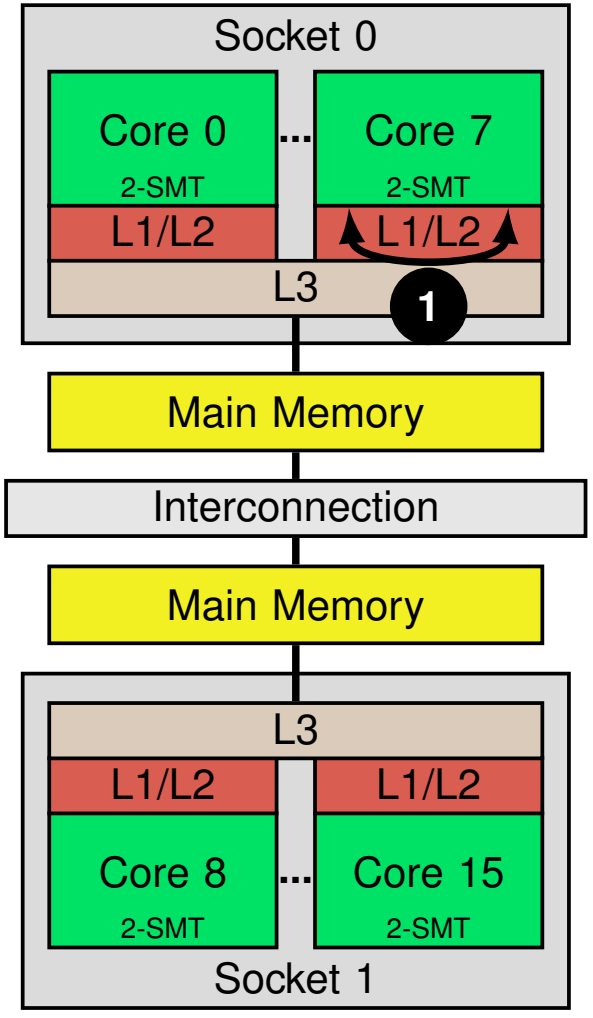
WHAT IS COMMUNICATION-BASED MAPPING?

- Computer systems have complex memory hierarchies, influence communication efficiency



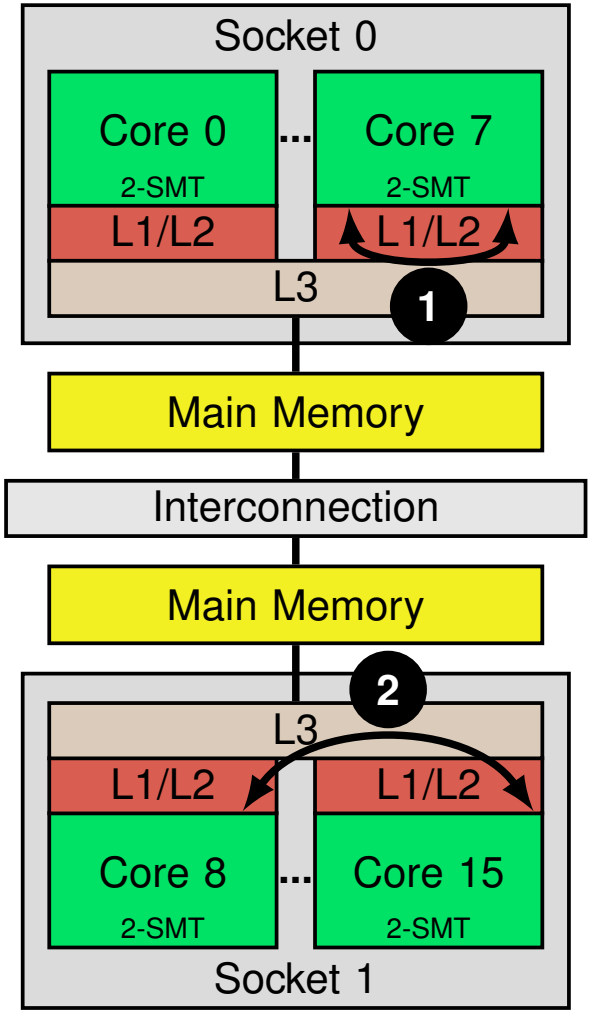
WHAT IS COMMUNICATION-BASED MAPPING?

- Computer systems have complex memory hierarchies, influence communication efficiency



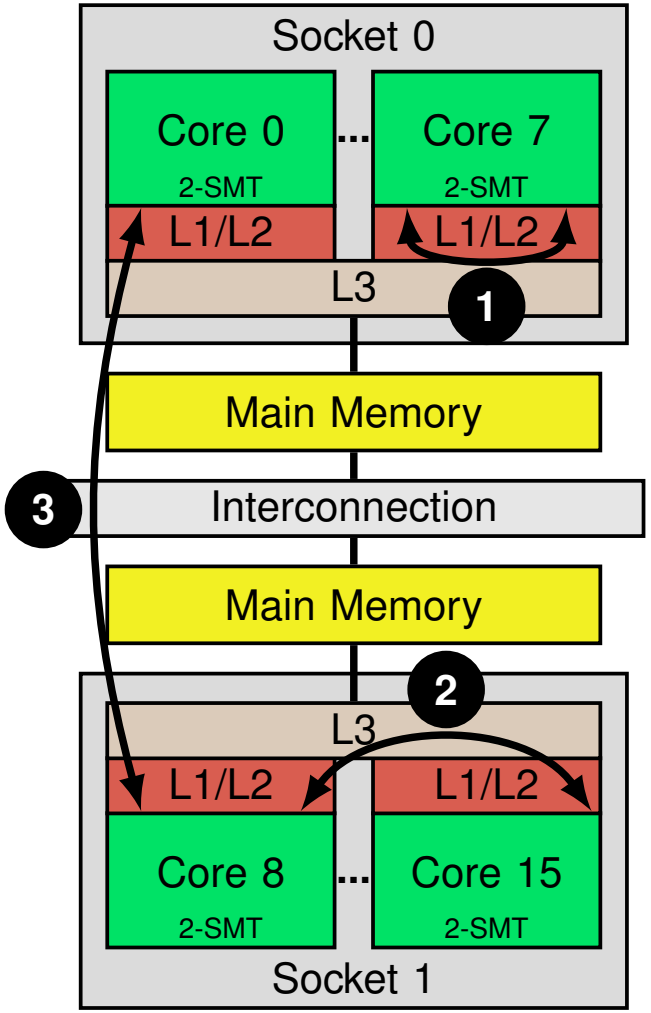
WHAT IS COMMUNICATION-BASED MAPPING?

- Computer systems have complex memory hierarchies, influence communication efficiency



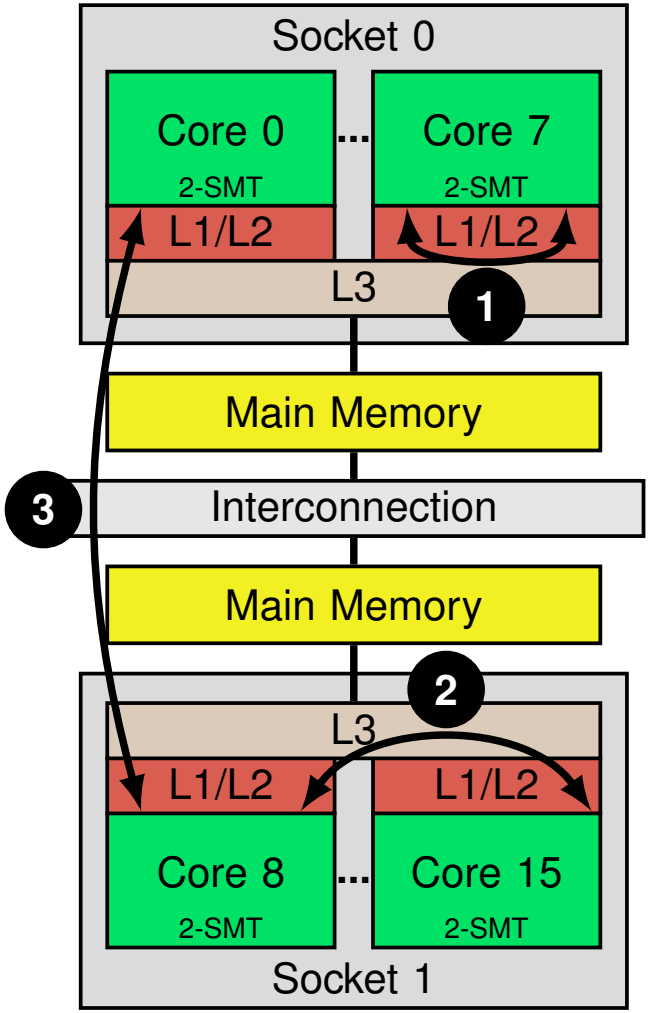
WHAT IS COMMUNICATION-BASED MAPPING?

- Computer systems have complex memory hierarchies, influence communication efficiency



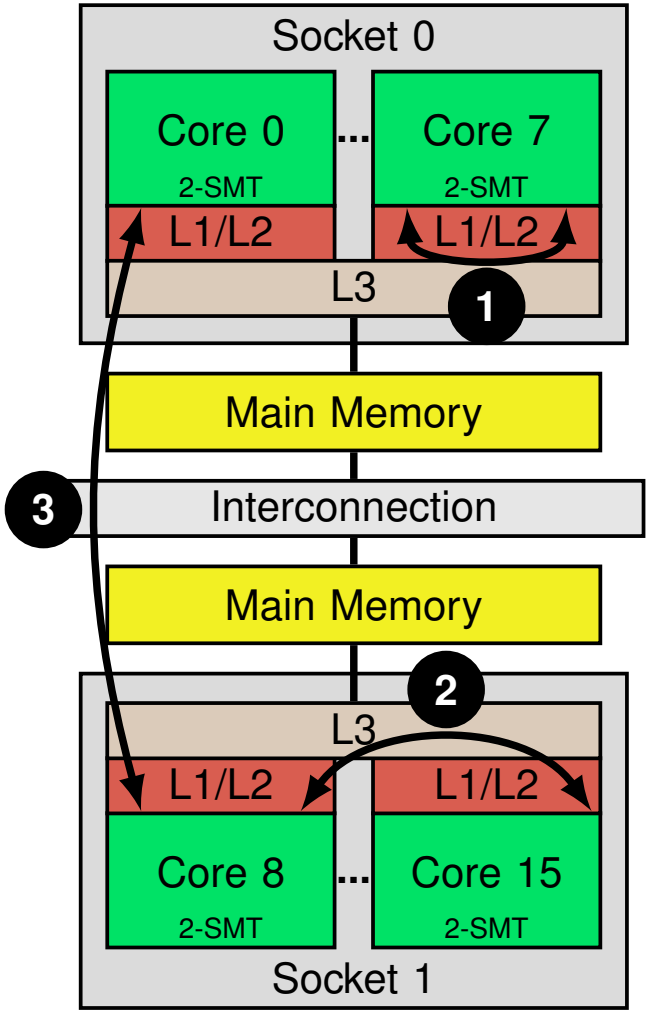
WHAT IS COMMUNICATION-BASED MAPPING?

- Computer systems have complex memory hierarchies, influence communication efficiency
- **Map threads to the hardware topology according to the communication between them**



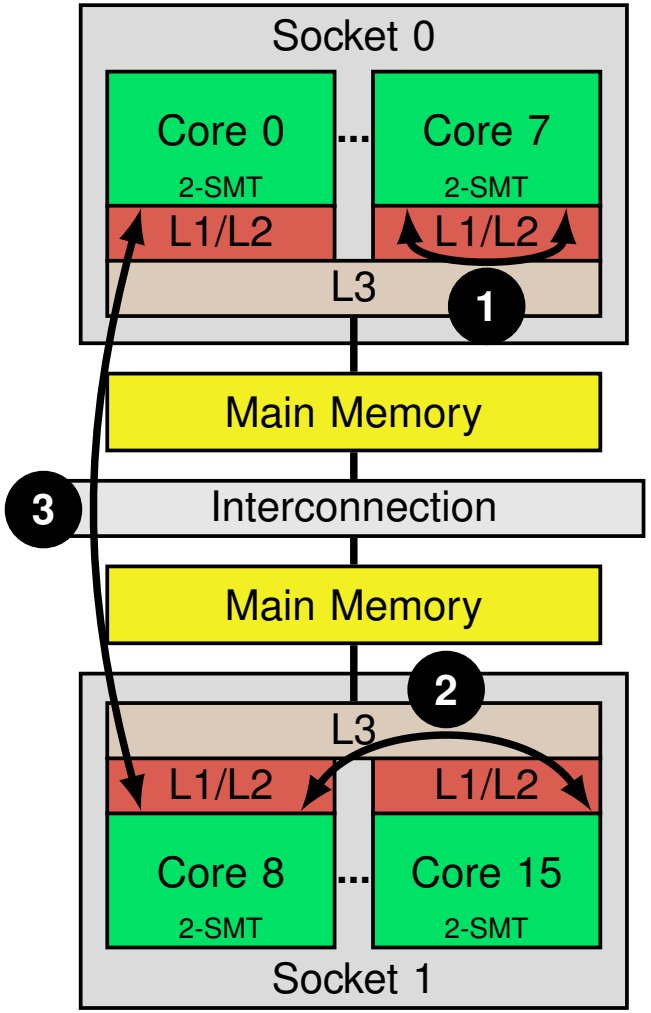
WHAT IS COMMUNICATION-BASED MAPPING?

- Computer systems have complex memory hierarchies, influence communication efficiency
- **Map threads to the hardware topology according to the communication between them**
- Goal: Improve performance and energy efficiency



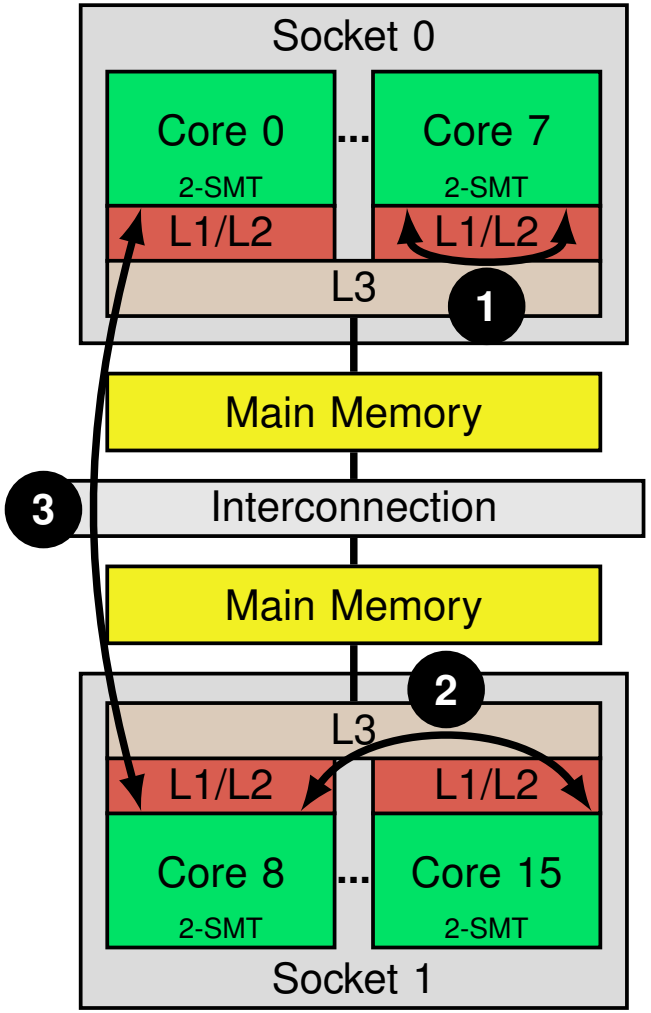
WHAT IS COMMUNICATION-BASED MAPPING?

- Computer systems have complex memory hierarchies, influence communication efficiency
- **Map threads to the hardware topology according to the communication between them**
- Goal: Improve performance and energy efficiency
  - Reduce cache misses



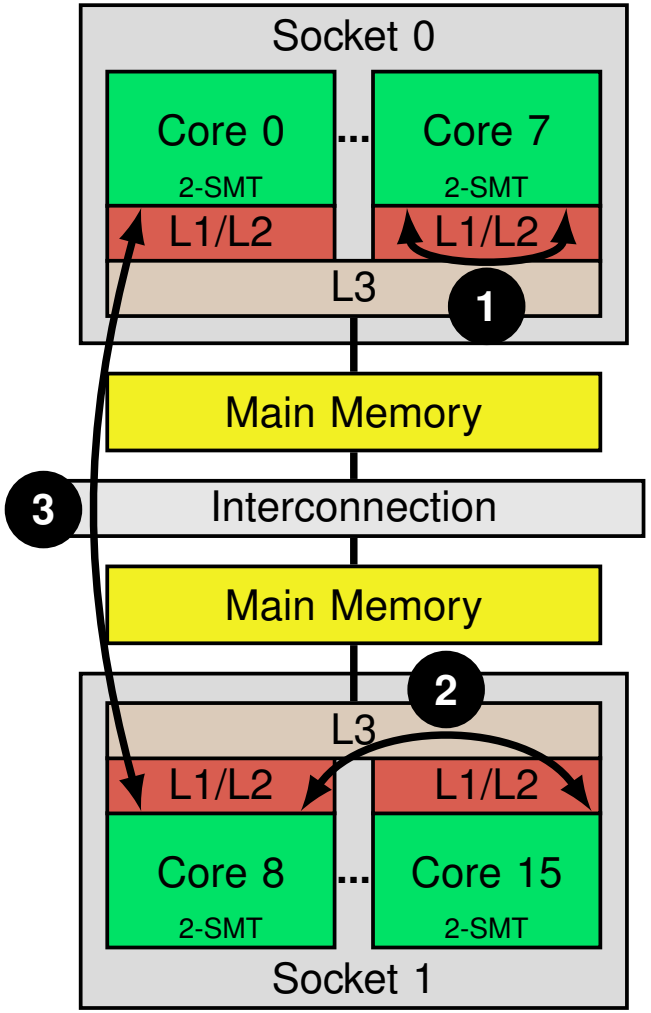
WHAT IS COMMUNICATION-BASED MAPPING?

- Computer systems have complex memory hierarchies, influence communication efficiency
- **Map threads to the hardware topology according to the communication between them**
- Goal: Improve performance and energy efficiency
  - Reduce cache misses
  - Improve interconnections usage



WHAT IS COMMUNICATION-BASED MAPPING?

- Computer systems have complex memory hierarchies, influence communication efficiency
- **Map threads to the hardware topology according to the communication between them**
- Goal: Improve performance and energy efficiency
  - Reduce cache misses
  - Improve interconnections usage
  - Increase locality of memory accesses



COMMUNICATION IN SHARED MEMORY

- Focus on parallel applications that use **shared memory**, inter-thread communication

COMMUNICATION IN SHARED MEMORY

- Focus on parallel applications that use **shared memory**, inter-thread communication

Main challenge

Communication is **implicit**
$\Rightarrow$ need to analyze memory access behavior

COMMUNICATION IN SHARED MEMORY

- Focus on parallel applications that use **shared memory**, inter-thread communication

Main challenge

Communication is **implicit**
$\Rightarrow$ need to analyze memory access behavior

- Previous work:
  - Memory traces (high overhead)

COMMUNICATION IN SHARED MEMORY

- Focus on parallel applications that use **shared memory**, inter-thread communication

Main challenge

Communication is **implicit**
$\Rightarrow$ need to analyze memory access behavior

- Previous work:
  - Memory traces (high overhead)
  - Hardware statistics (imprecise, hardware dependent)

COMMUNICATION IN SHARED MEMORY

- Focus on parallel applications that use **shared memory**, inter-thread communication

Main challenge

Communication is **implicit**
$\Rightarrow$ need to analyze memory access behavior

- Previous work:
  - Memory traces (high overhead)
  - Hardware statistics (imprecise, hardware dependent)
  - Modifying runtime libraries (only for a subset of applications)

Our proposal:

SPCD: Shared Pages Communication Detection

Use virtual memory for detection

SPCD: BACKGROUND

- Use virtual memory implementation of the operating system

SPCD: BACKGROUND

- Use virtual memory implementation of the operating system
  - Threads of a parallel application share the virtual address space, common page table

SPCD: BACKGROUND

- Use virtual memory implementation of the operating system
  - Threads of a parallel application share the virtual address space, common page table
  - On memory access to a page that is not in the page table, application causes page fault

SPCD: BACKGROUND

- Use virtual memory implementation of the operating system
  - Threads of a parallel application share the virtual address space, common page table
  - On memory access to a page that is not in the page table, application causes page fault
  - Page fault is resolved by the OS

SPCD: BACKGROUND

- Use virtual memory implementation of the operating system
  - Threads of a parallel application share the virtual address space, common page table
  - On memory access to a page that is not in the page table, application causes page fault
  - Page fault is resolved by the OS
- Two parts:

SPCD: BACKGROUND

- Use virtual memory implementation of the operating system
  - Threads of a parallel application share the virtual address space, common page table
  - On memory access to a page that is not in the page table, application causes page fault
  - Page fault is resolved by the OS
- Two parts:
  1. **Detect Communication**

SPCD: BACKGROUND

- Use virtual memory implementation of the operating system
  - Threads of a parallel application share the virtual address space, common page table
  - On memory access to a page that is not in the page table, application causes page fault
  - Page fault is resolved by the OS
- Two parts:
  1. **Detect Communication**
  2. Map threads to hardware topology

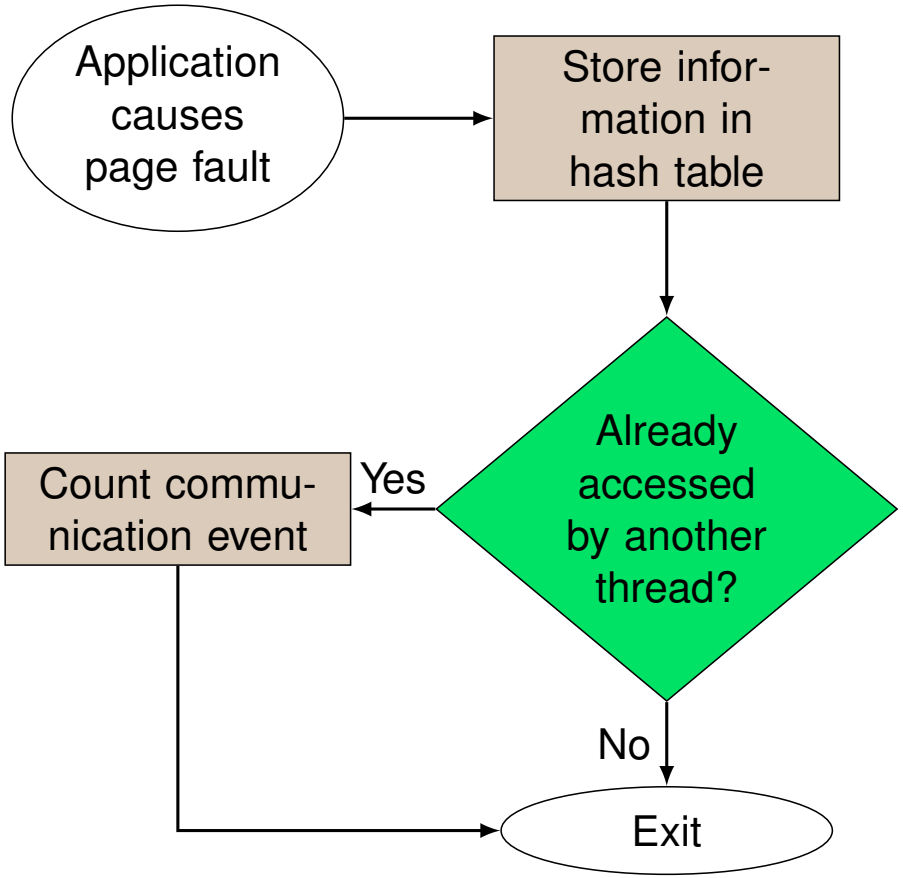
SPCD: ANALYZING MEMORY ACCESS BEHAVIOR

SPCD: ANALYZING MEMORY ACCESS BEHAVIOR

- **Analyze memory access behavior by observing page faults**

SPCD: ANALYZING MEMORY ACCESS BEHAVIOR

- **Analyze memory access behavior by observing page faults**
  - Store information about them (address, thread ID)
  - Page faults with the same address: *communication event*
- Granularity of detection is configurable
  - Divide address space into *memory regions*



SPCD: ADDITIONAL PAGE FAULTS

Problem:

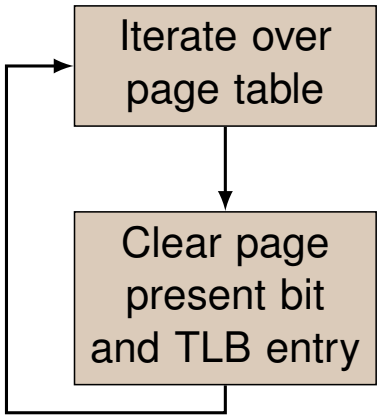
Only 1 page fault per page

SPCD: ADDITIONAL PAGE FAULTS

Problem:

Only 1 page fault per page

- **Insert additional page faults in the application**

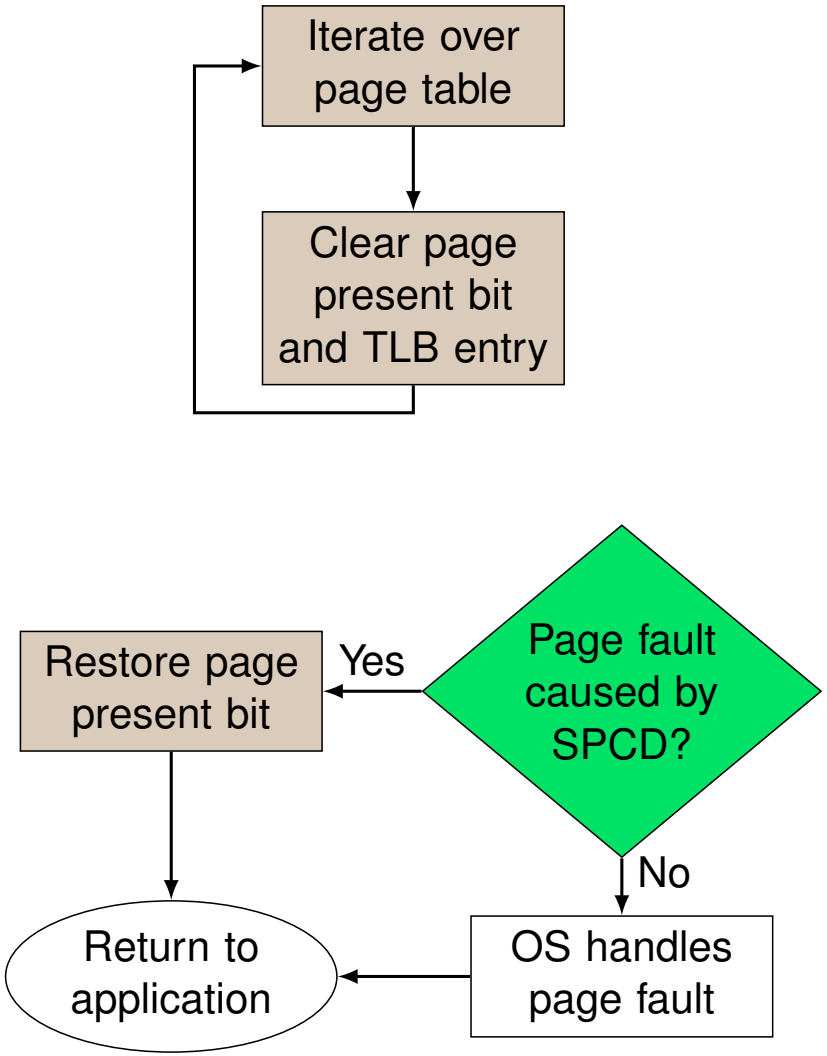


SPCD: ADDITIONAL PAGE FAULTS

Problem:

Only 1 page fault per page

- **Insert additional page faults in the application**
- Page faults can be resolved with a low overhead

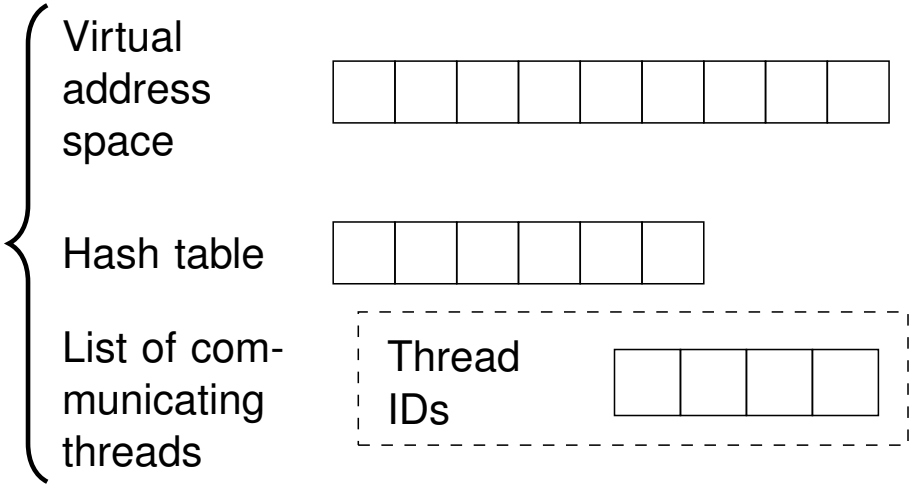


SPCD: IMPLEMENTATION

- Implemented as kernel module for Linux

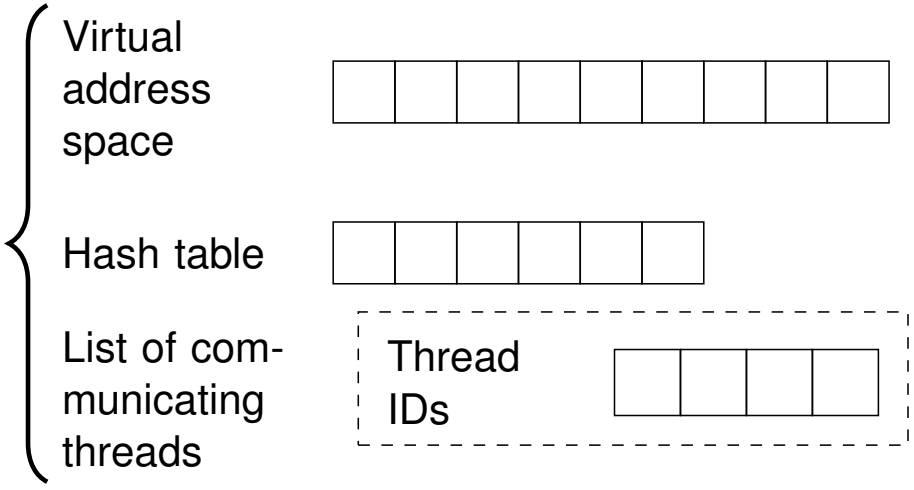
SPCD: IMPLEMENTATION

- Implemented as kernel module for Linux
- Store memory accesses in a hash table, indexed by the virtual address



SPCD: IMPLEMENTATION

- Implemented as kernel module for Linux
- Store memory accesses in a hash table, indexed by the virtual address

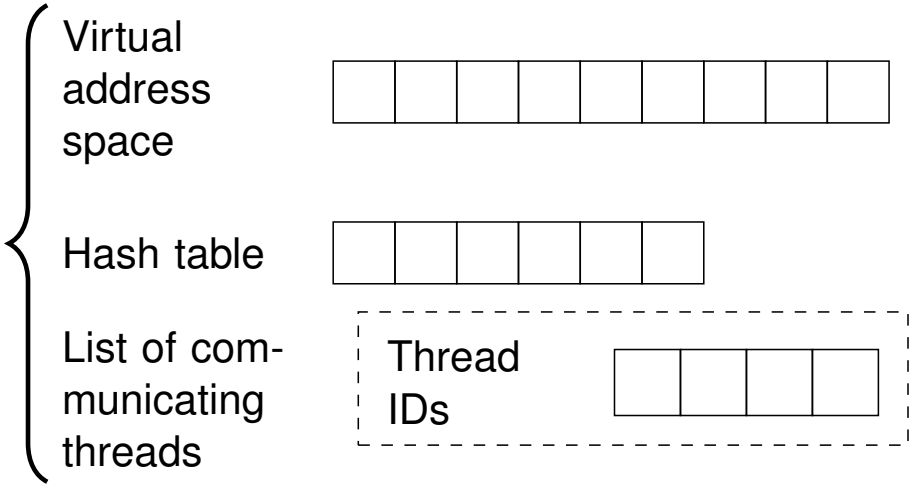


- Store communication events in communication matrix

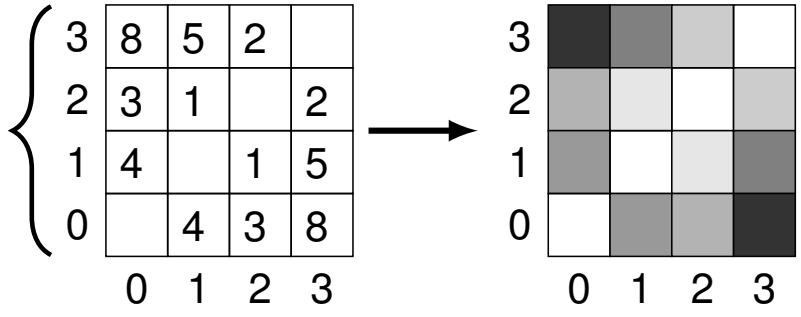
3	8	5	2	
2	3	1		2
1	4		1	5
0		4	3	8
	0	1	2	3

SPCD: IMPLEMENTATION

- Implemented as kernel module for Linux
- Store memory accesses in a hash table, indexed by the virtual address

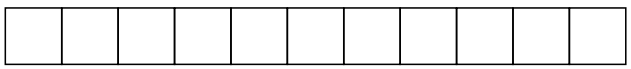


- Store communication events in communication matrix

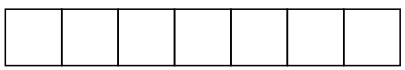


SPCD: EXAMPLE

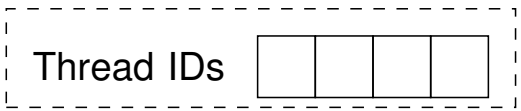
Virtual address space



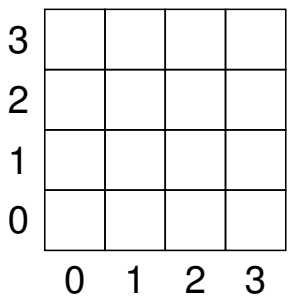
Hash table



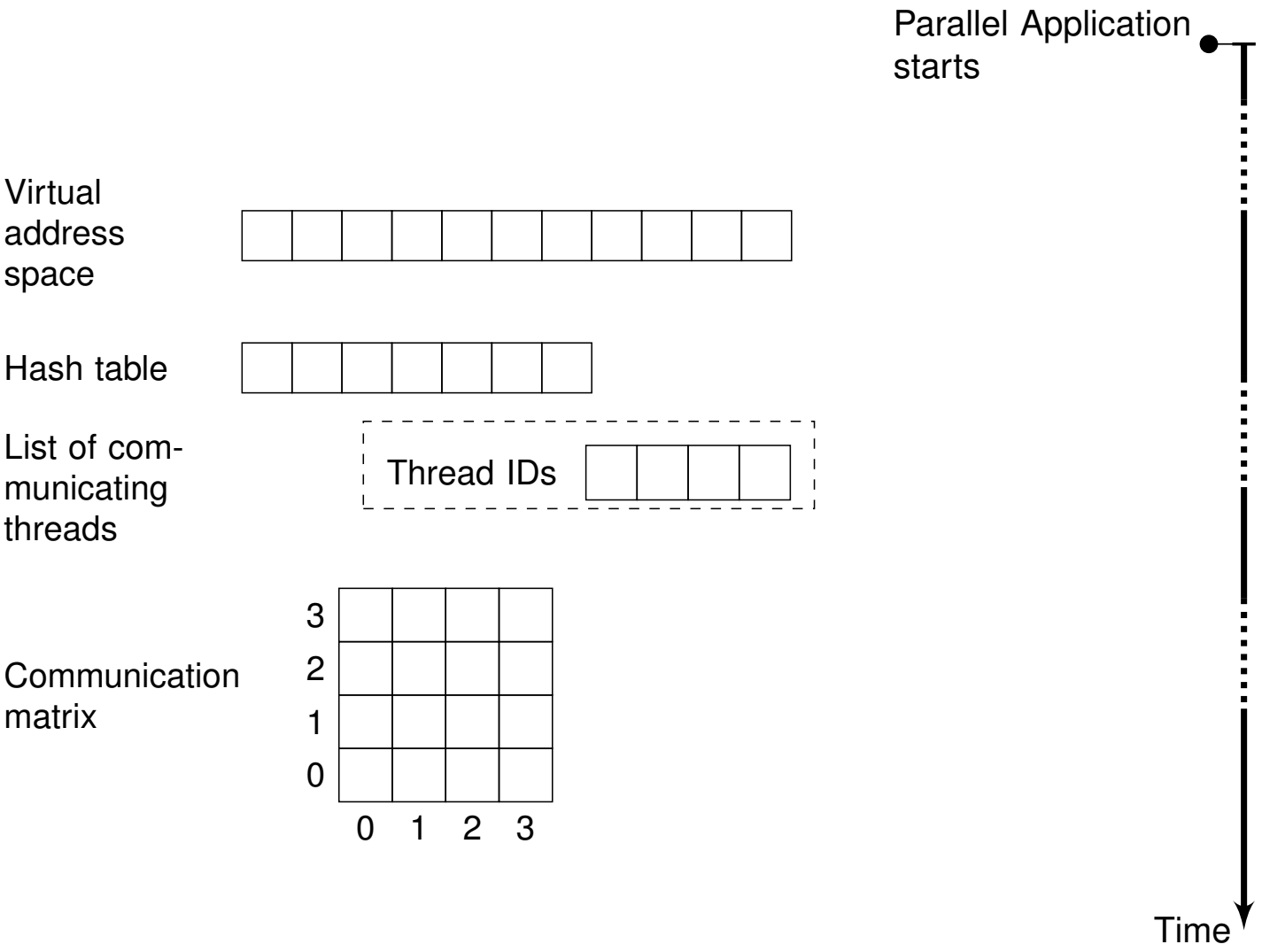
List of communicating threads



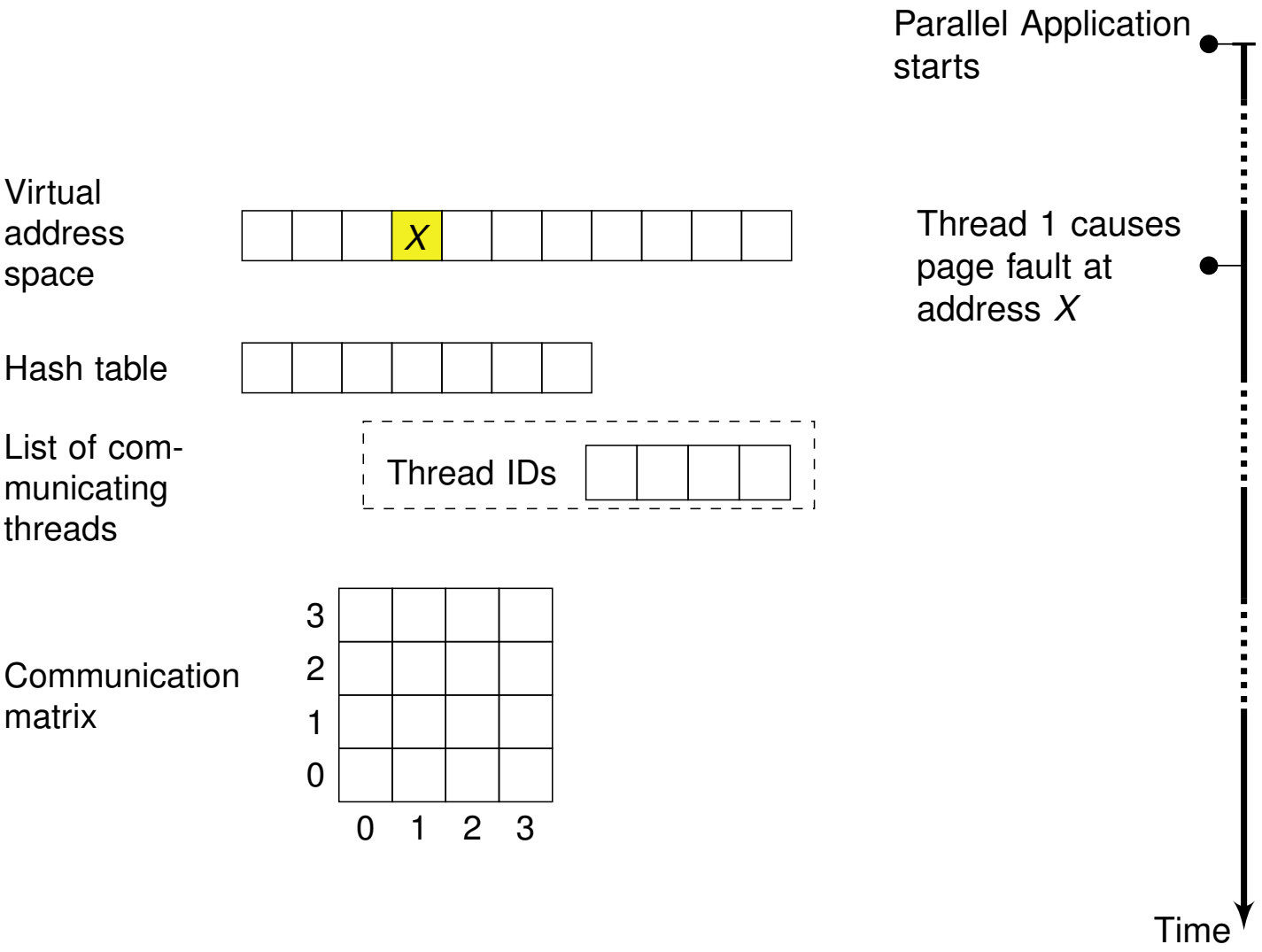
Communication matrix



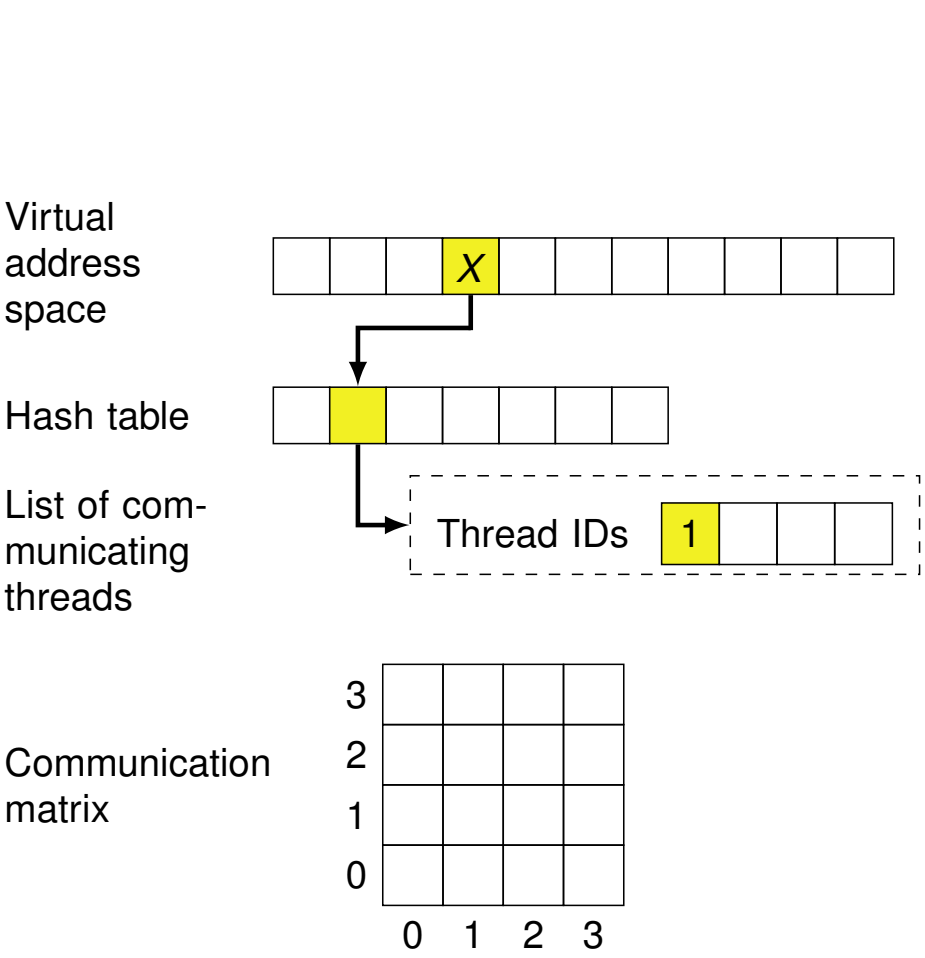
SPCD: EXAMPLE

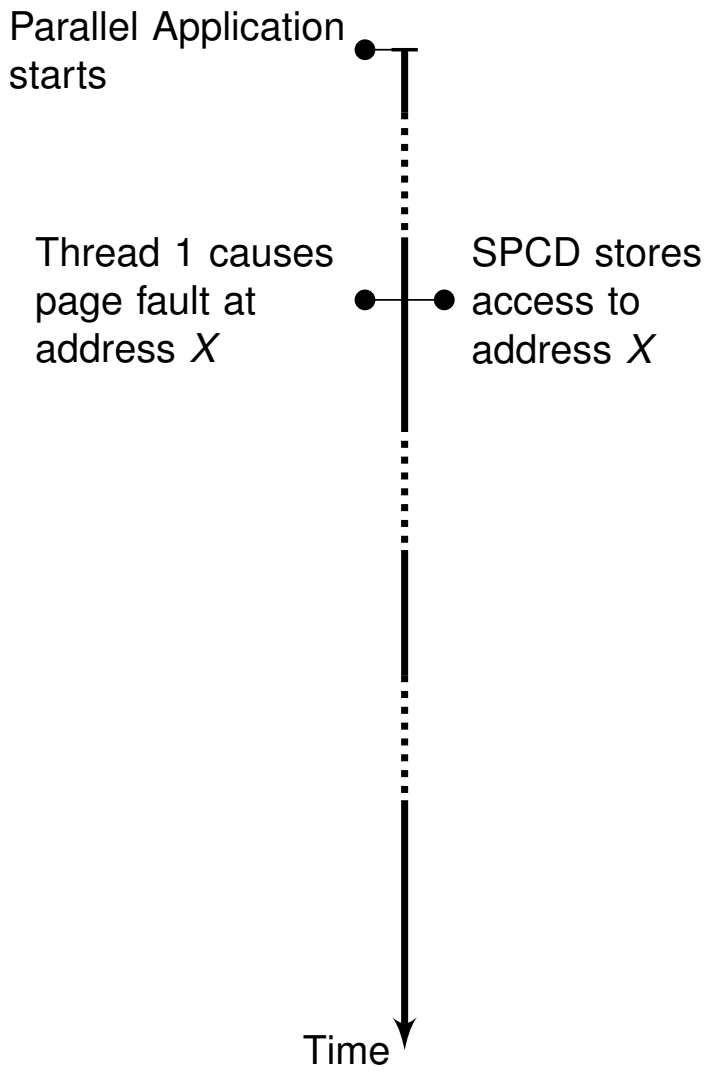


SPCD: EXAMPLE

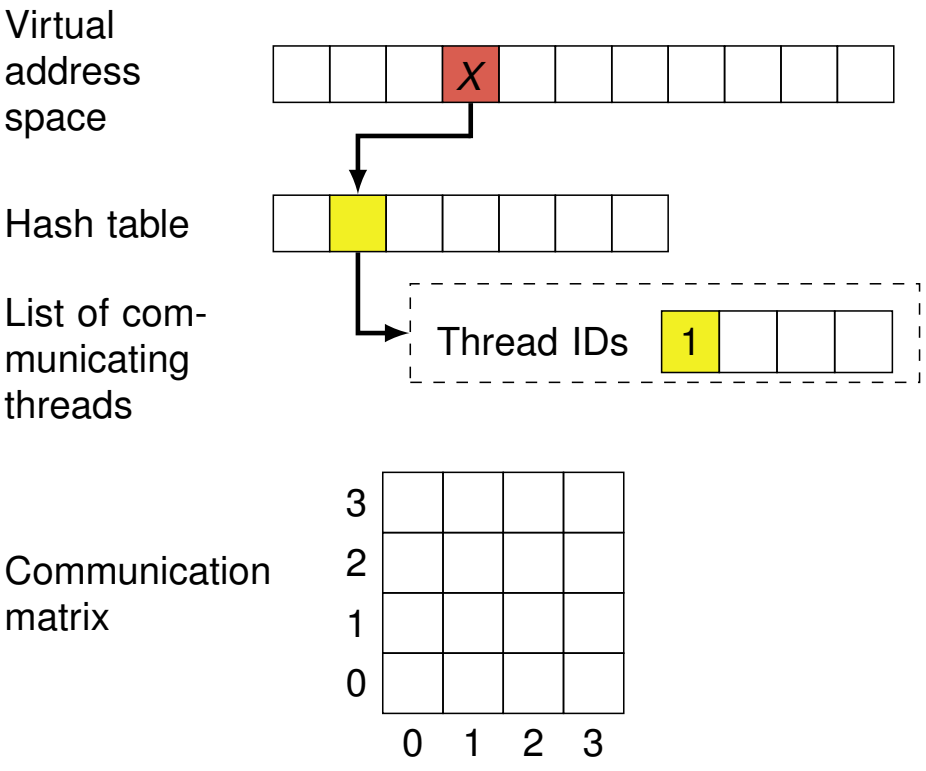


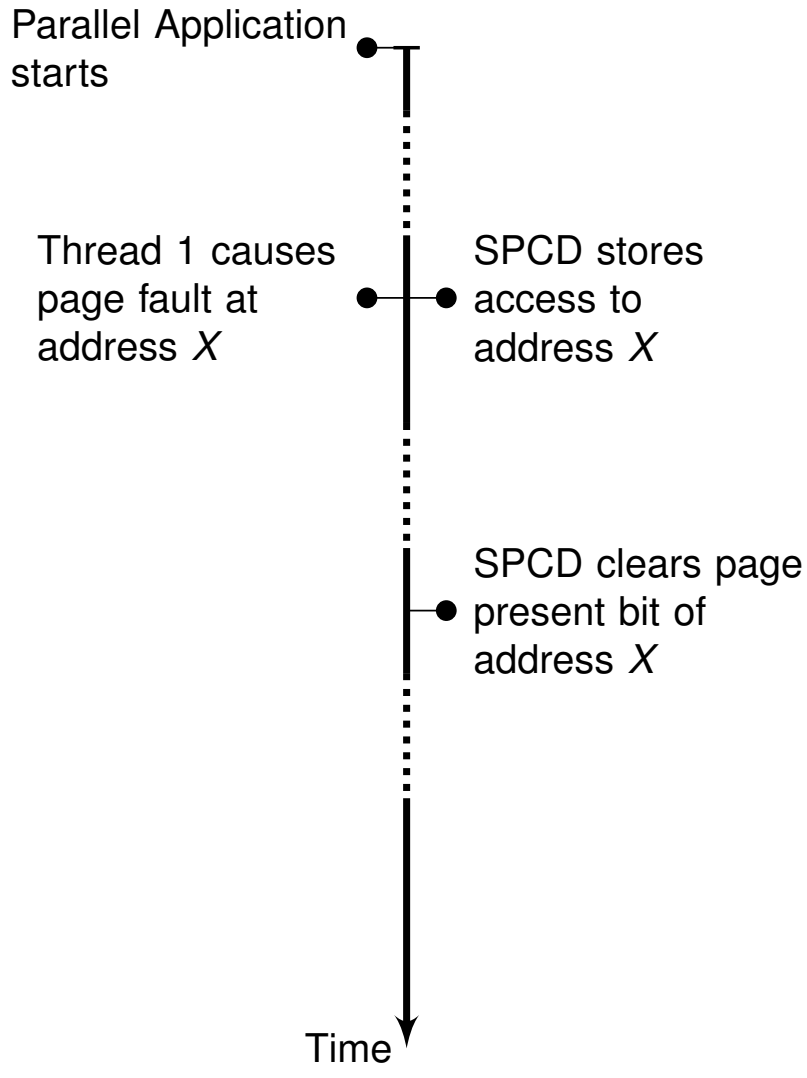
SPCD: EXAMPLE



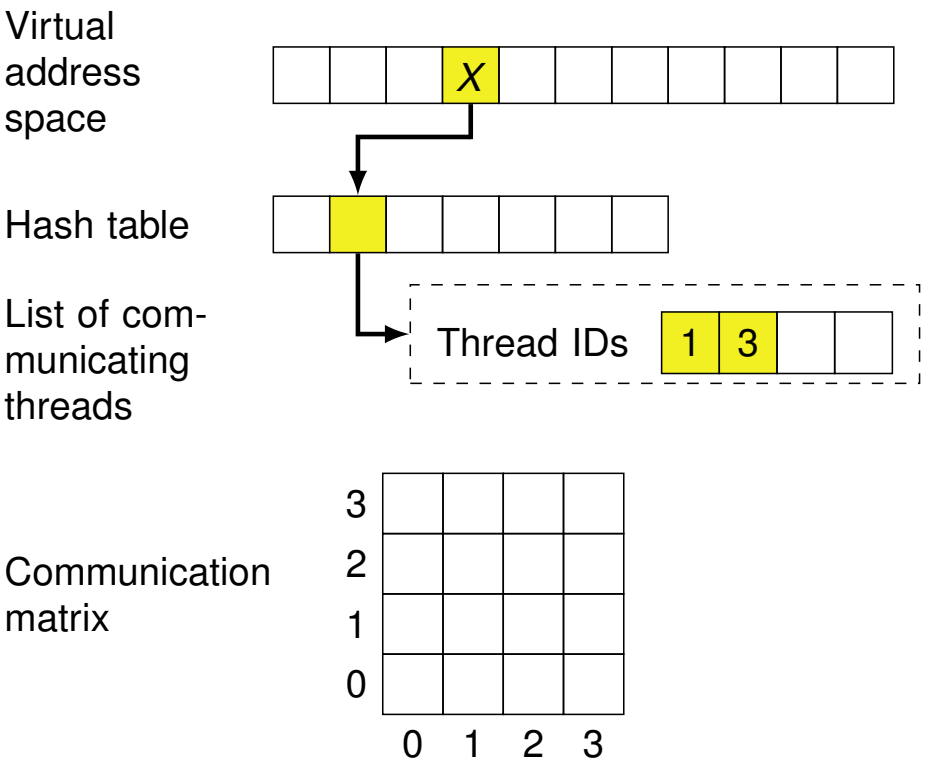


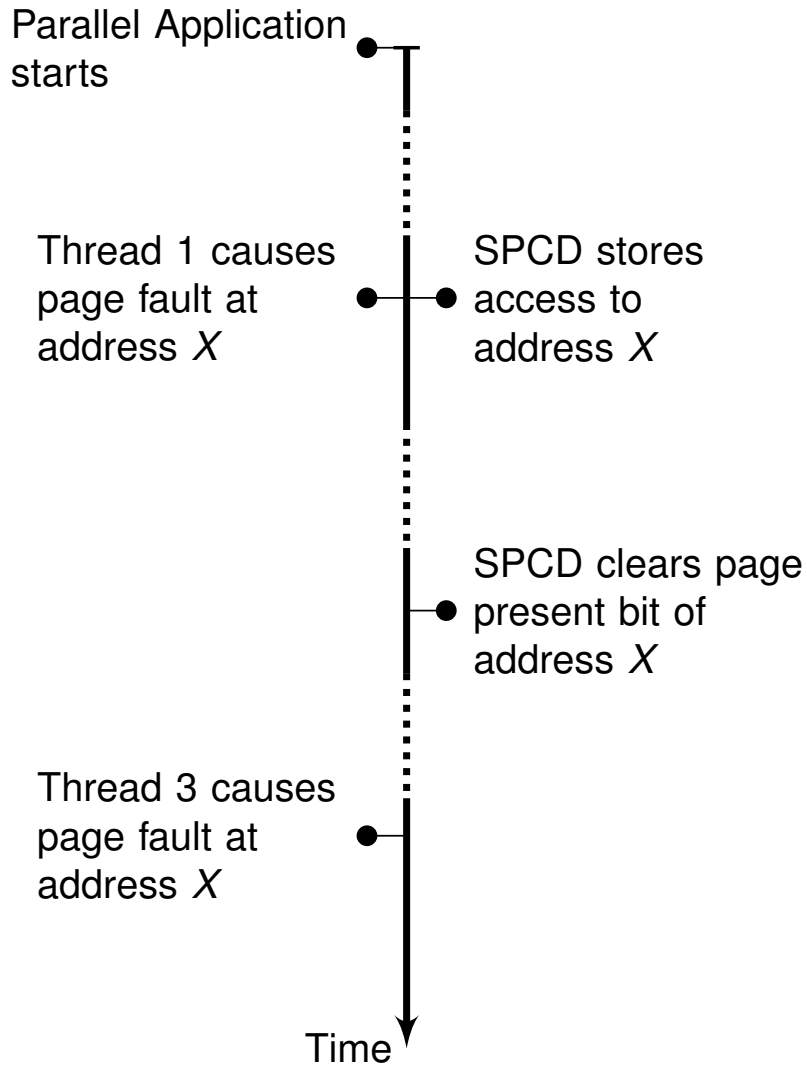
SPCD: EXAMPLE



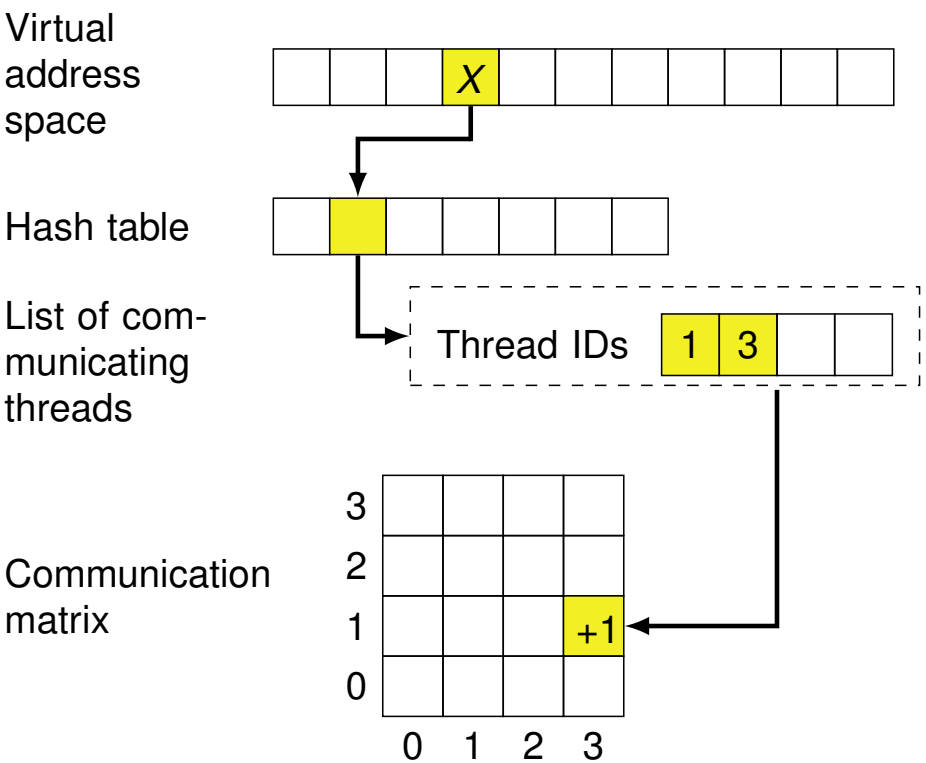


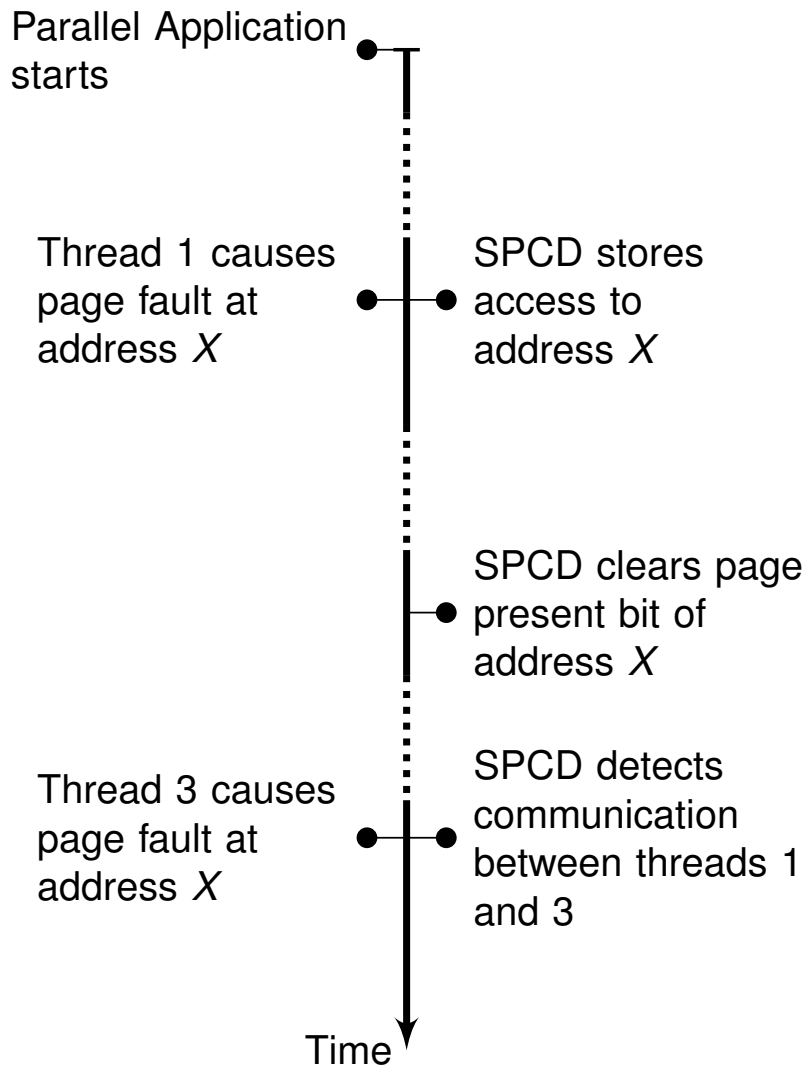
SPCD: EXAMPLE





SPCD: EXAMPLE





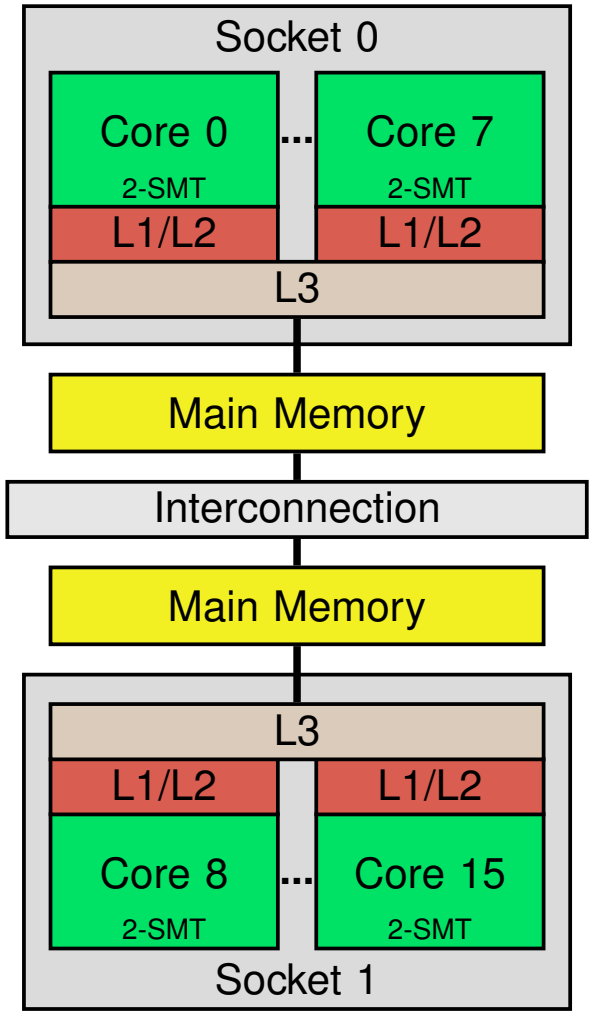
Evaluation

METHODOLOGY: COMMUNICATION PATTERNS

- Experiments with NAS Parallel Benchmarks (OpenMP, 32 threads)

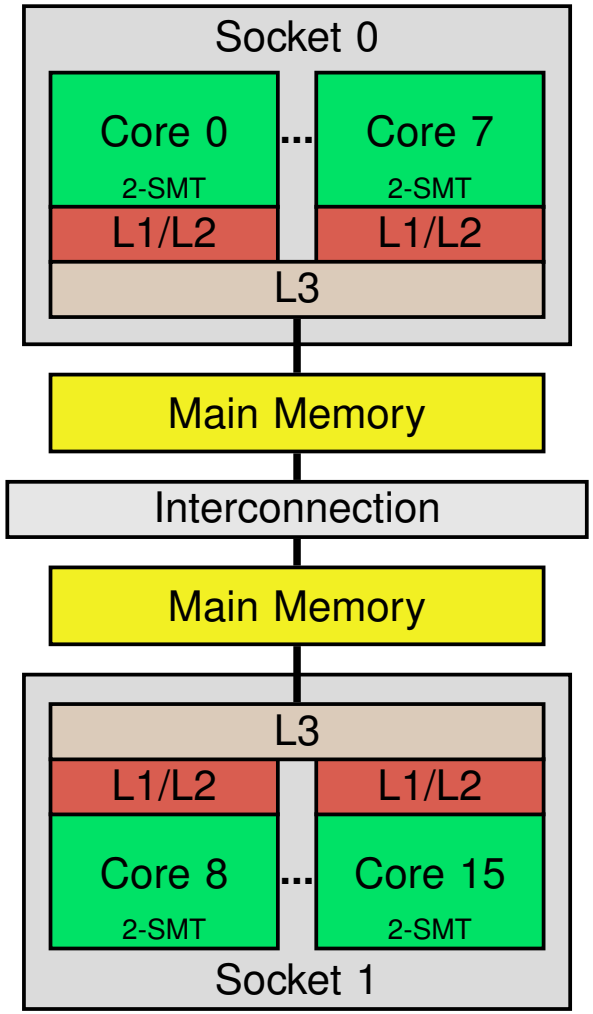
METHODOLOGY: COMMUNICATION PATTERNS

- Experiments with NAS Parallel Benchmarks (OpenMP, 32 threads)
- Host machine:
  2x Intel Xeon E5-2650, 2.0 GHz
  (8x 2-way SMT cores each)



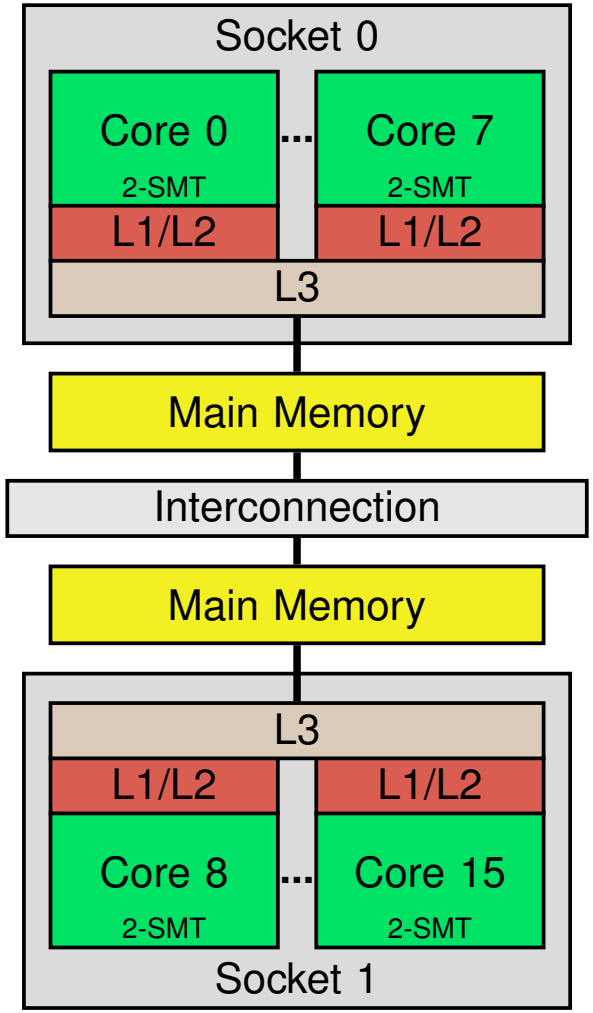
METHODOLOGY: COMMUNICATION PATTERNS

- Experiments with NAS Parallel Benchmarks (OpenMP, 32 threads)
- Host machine:
  2x Intel Xeon E5-2650, 2.0 GHz
  (8x 2-way SMT cores each)
- SPCD



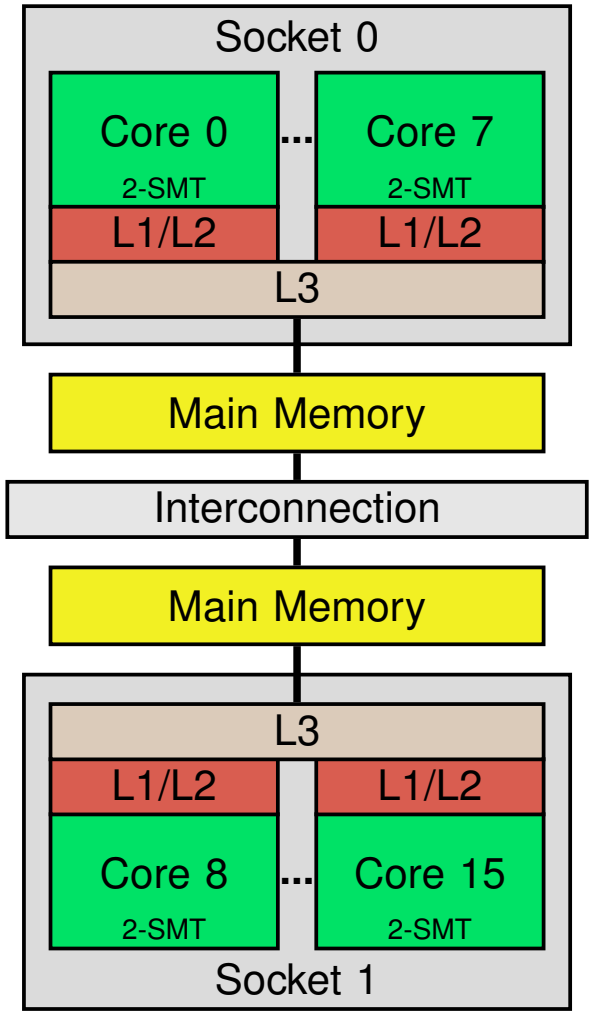
METHODOLOGY: COMMUNICATION PATTERNS

- Experiments with NAS Parallel Benchmarks (OpenMP, 32 threads)
- Host machine:
  2x Intel Xeon E5-2650, 2.0 GHz
  (8x 2-way SMT cores each)
- SPCD
  - Granularity: 512 Bytes

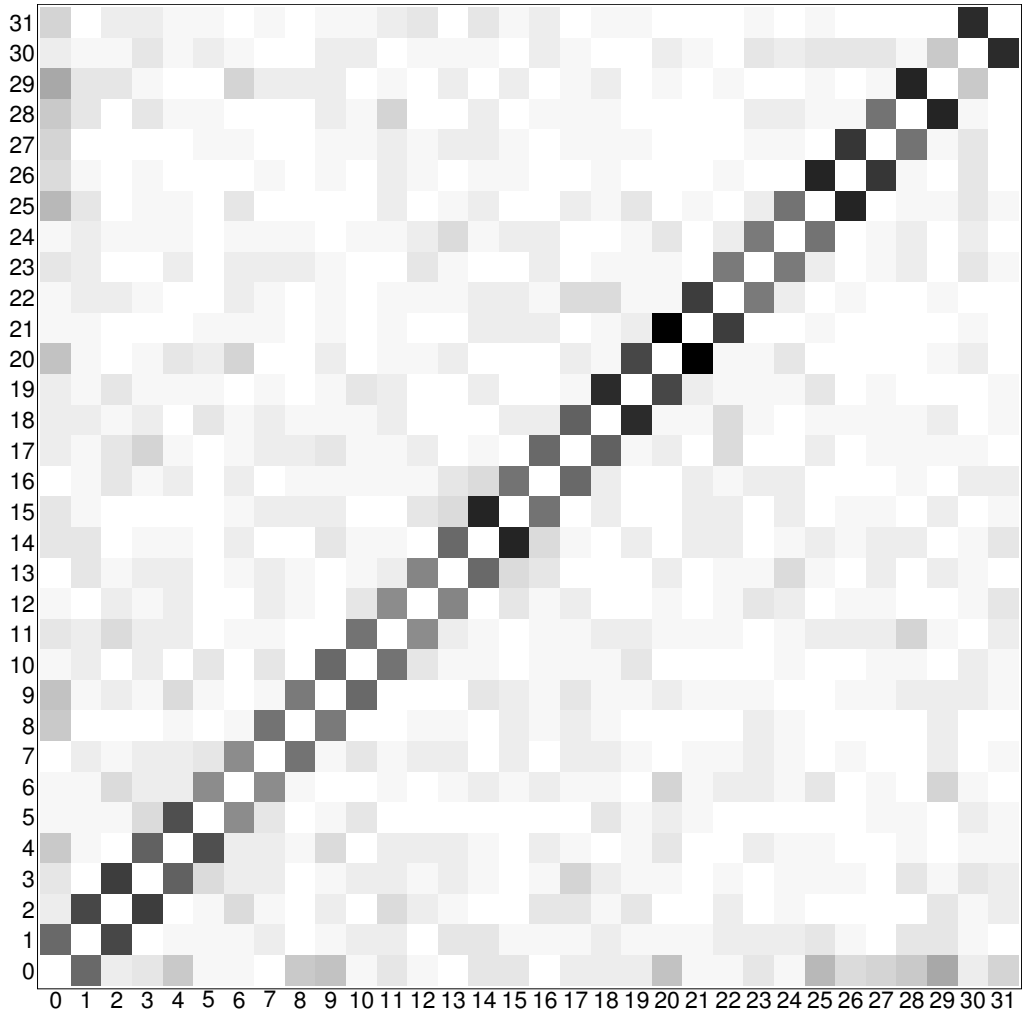


METHODOLOGY: COMMUNICATION PATTERNS

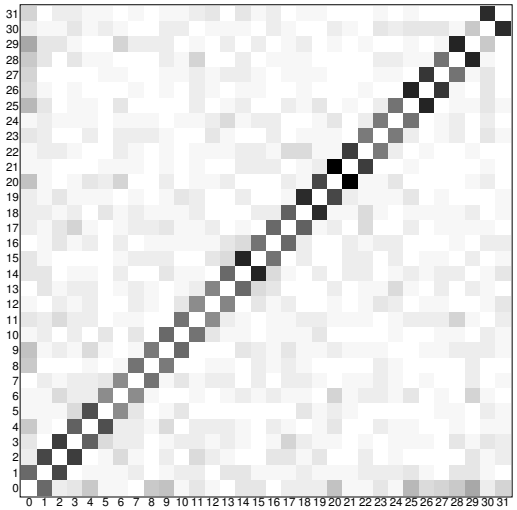
- Experiments with NAS Parallel Benchmarks (OpenMP, 32 threads)
- Host machine:
  2x Intel Xeon E5-2650, 2.0 GHz
  (8x 2-way SMT cores each)
- SPCD
  - Granularity: 512 Bytes
  - Additional page faults: ~10%



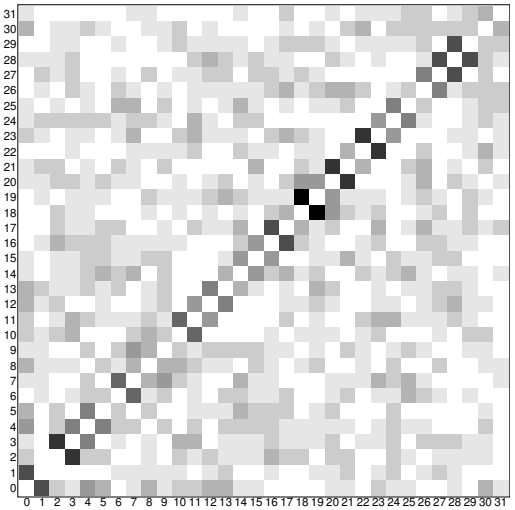
COMMUNICATION PATTERN – BT



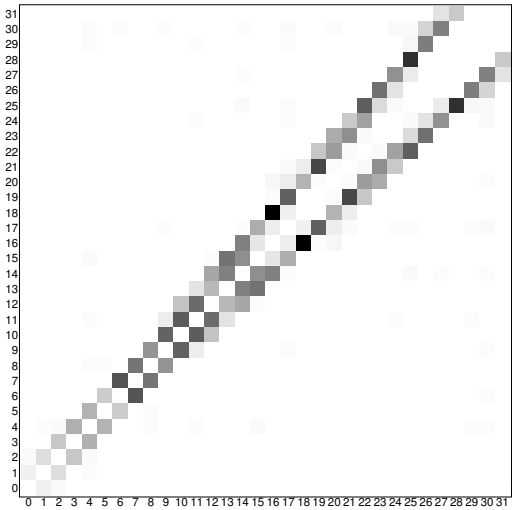
COMMUNICATION PATTERNS – BT, LU, MG, SP, UA



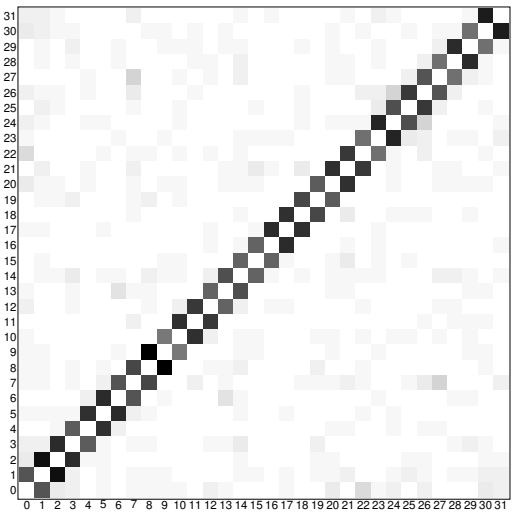
BT



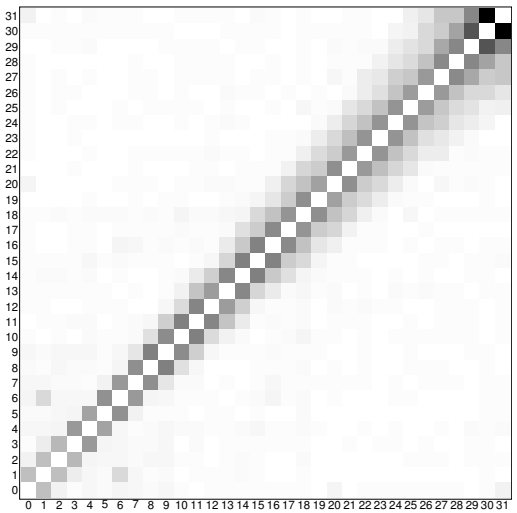
LU



MG

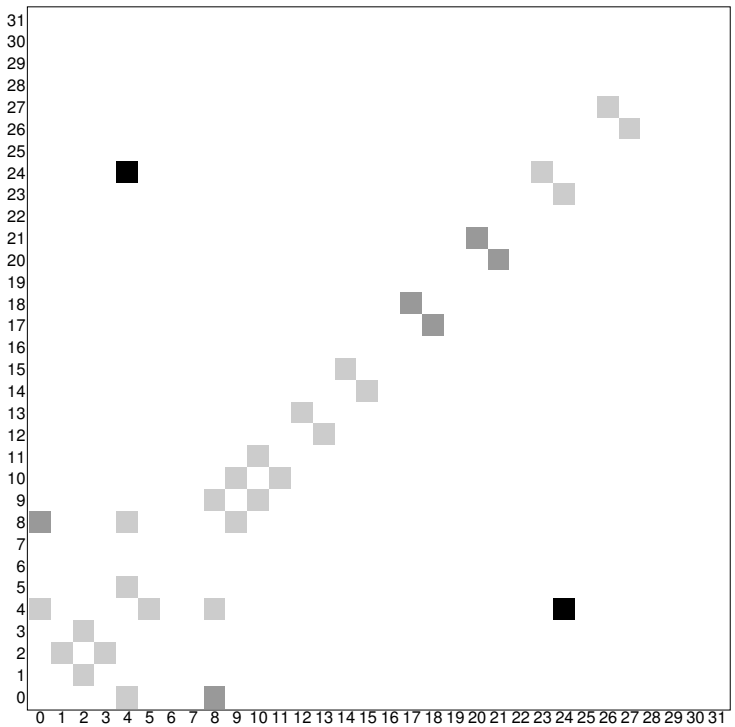


SP

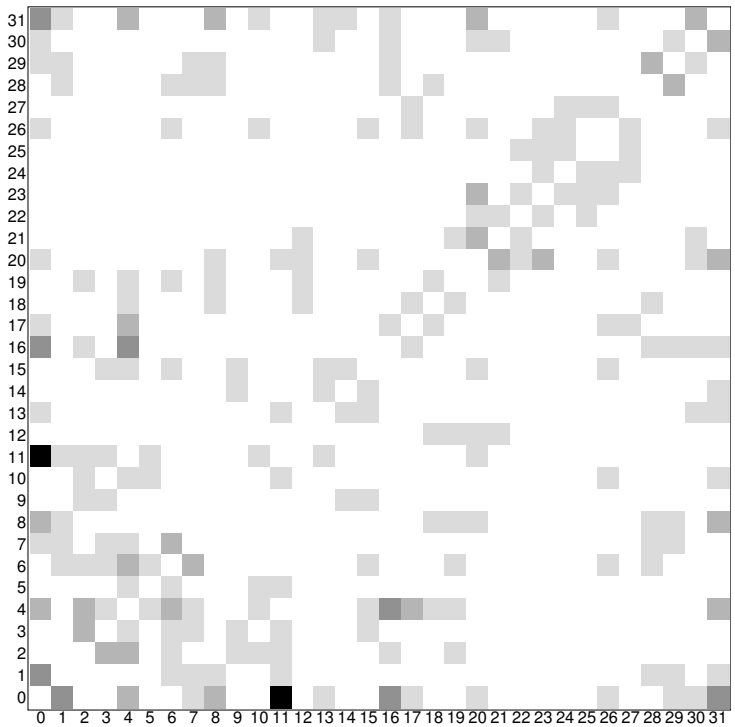


UA

COMMUNICATION PATTERNS – CG, DC

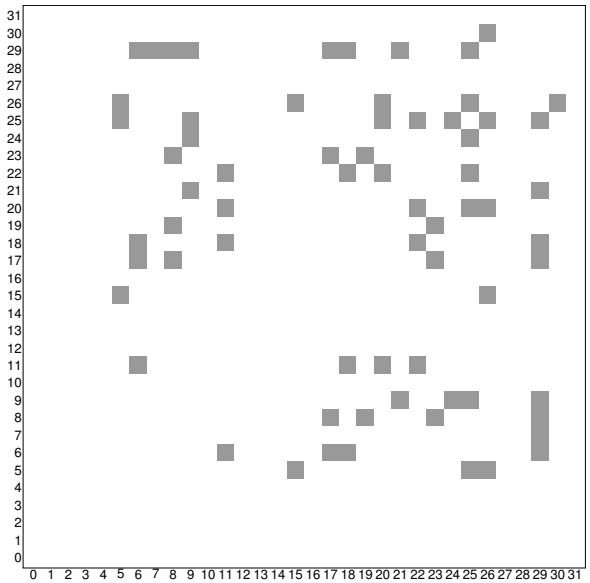


CG

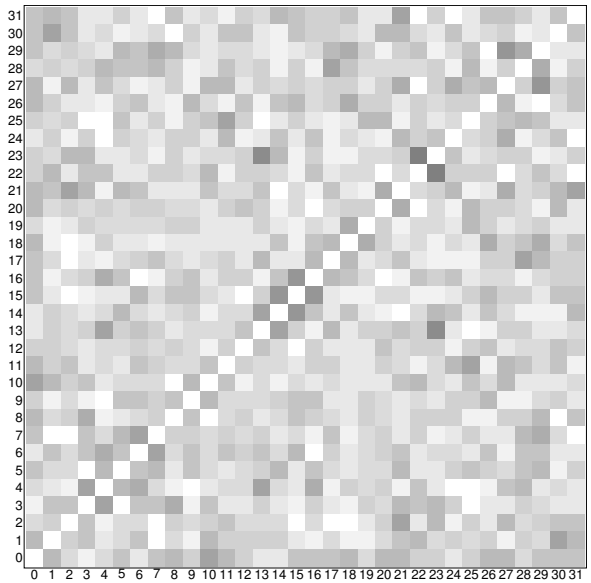


DC

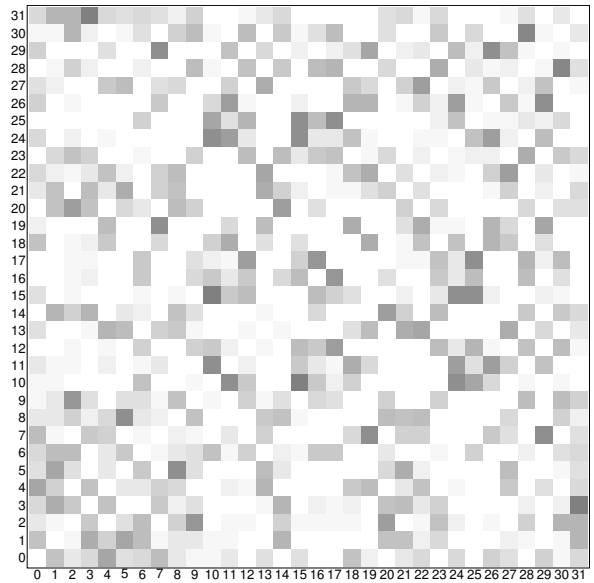
COMMUNICATION PATTERNS – EP, FT, IS



EP



FT



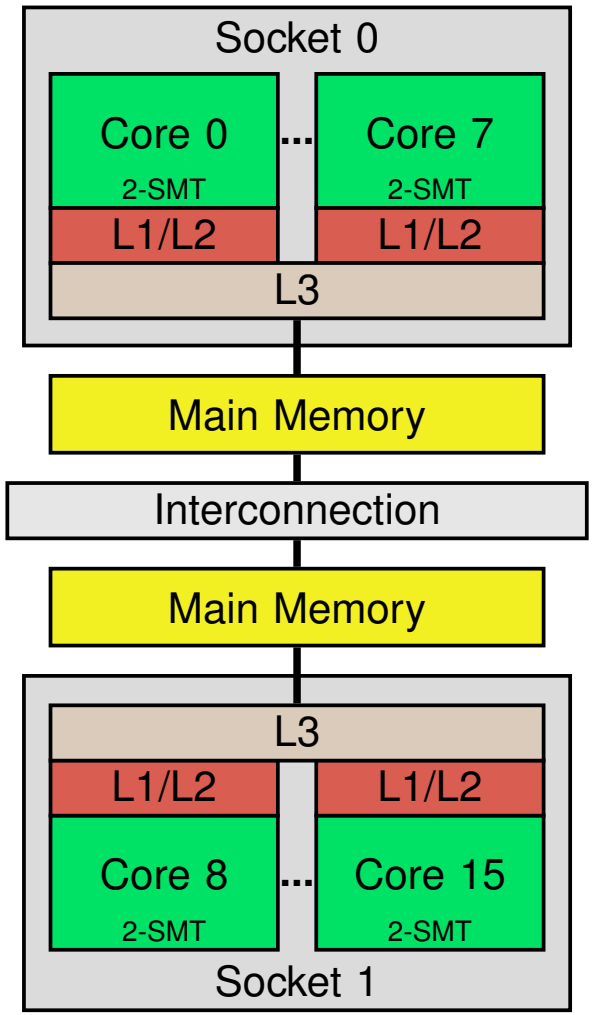
IS

COMMUNICATION PATTERNS – SUMMARY

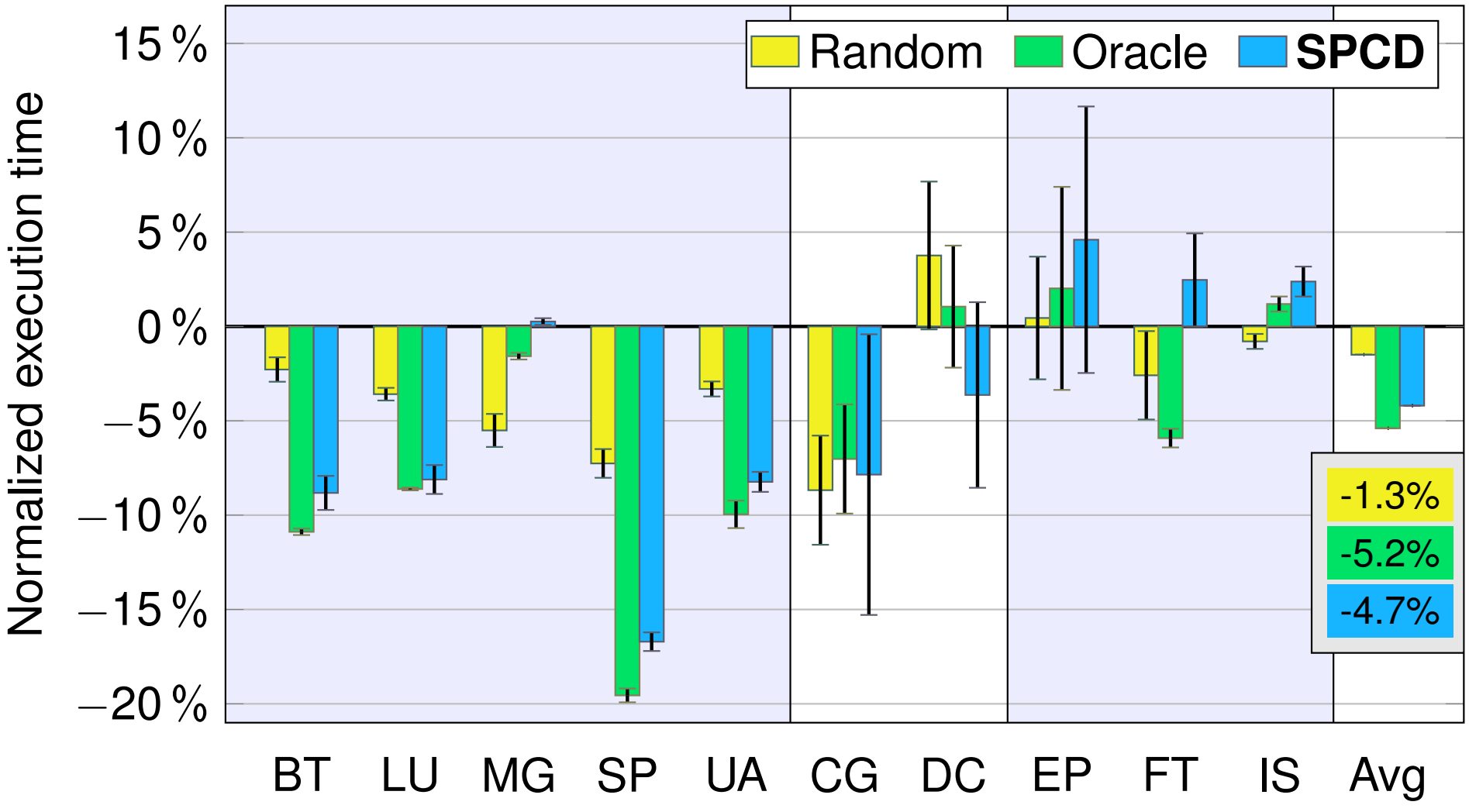
- All patterns detected correctly
- 3 groups
  1. Structured pattern, nearest neighbors: BT, LU, MG, SP, UA
  2. Slightly structured pattern, nearest neighbors: CG, DC
  3. Unstructured pattern, all-to-all: EP, FT, IS
- Expect performance improvements for structured patterns

METHODOLOGY: PERFORMANCE AND ENERGY

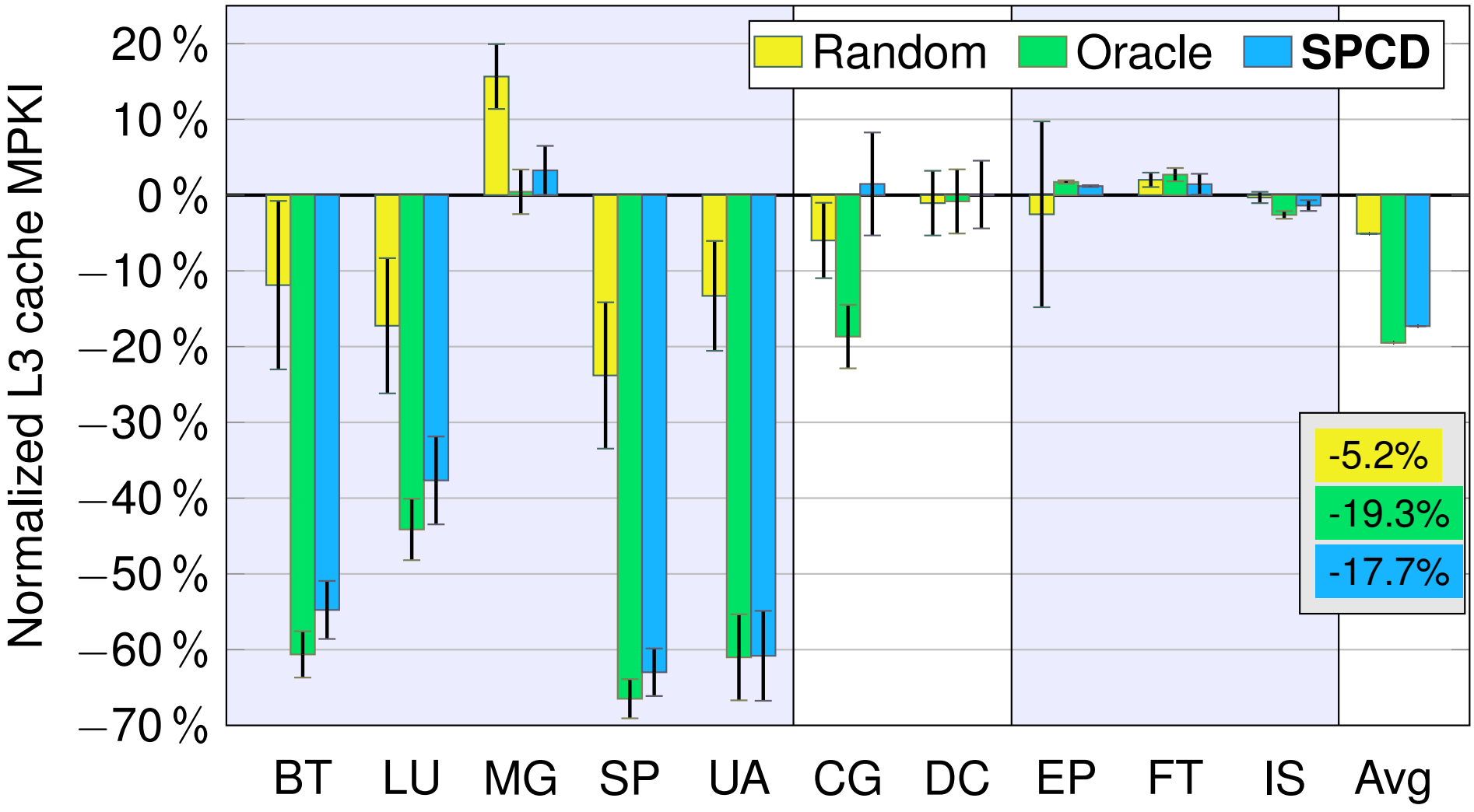
- Same machine, benchmarks, SPCD configuration as before
- **Online mapping of threads to cores using Edmonds' graph matching, minimizing communication costs**
- Compare SPCD to
  - Operating system (baseline)
  - Random static mapping
  - Oracle mapping
- Measure execution time, cache misses and processor energy consumption (using SandyBridge RAPL counters)



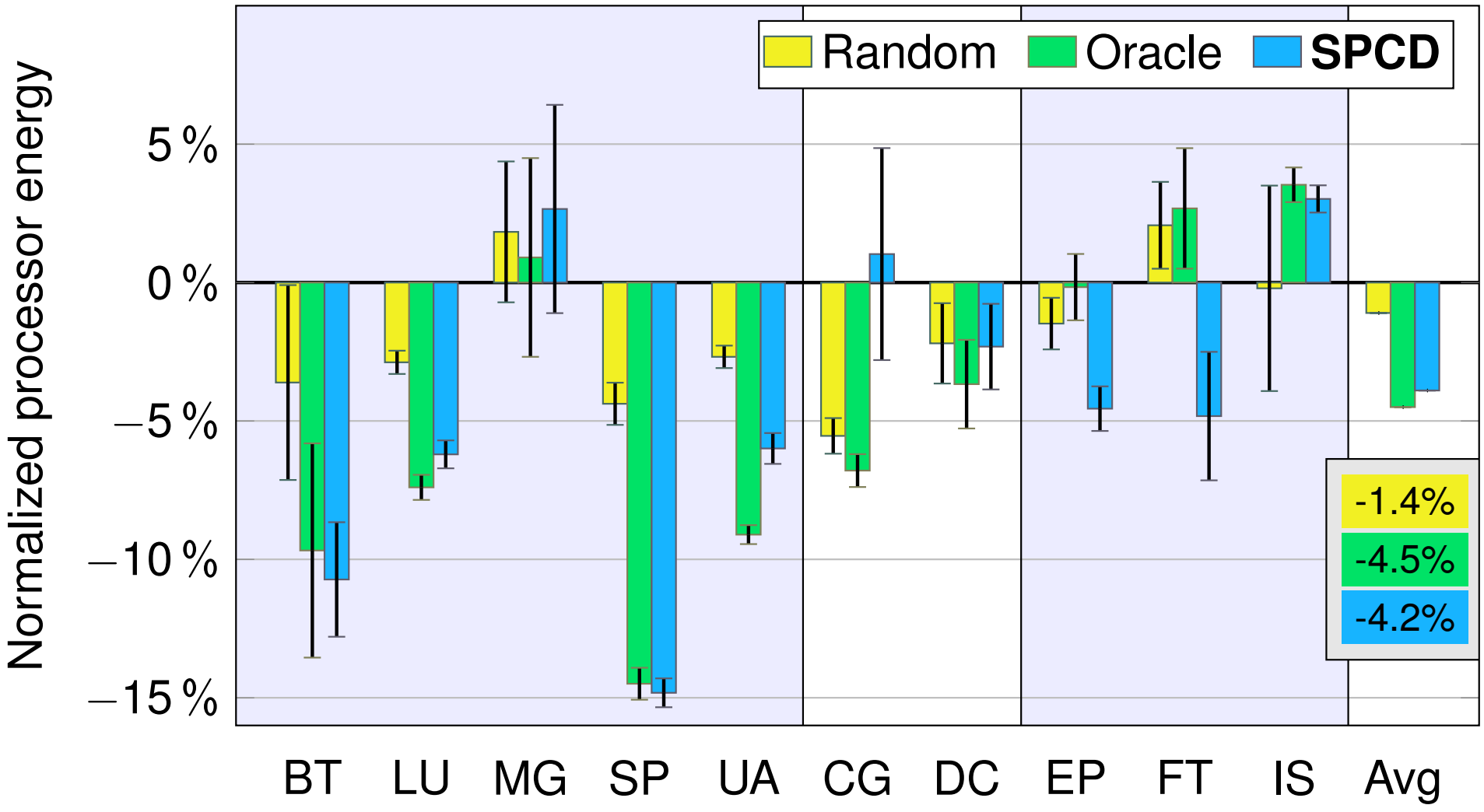
RESULTS – EXECUTION TIME



RESULTS – L3 CACHE MPKI



RESULTS – PROCESSOR ENERGY PER INSTRUCTION

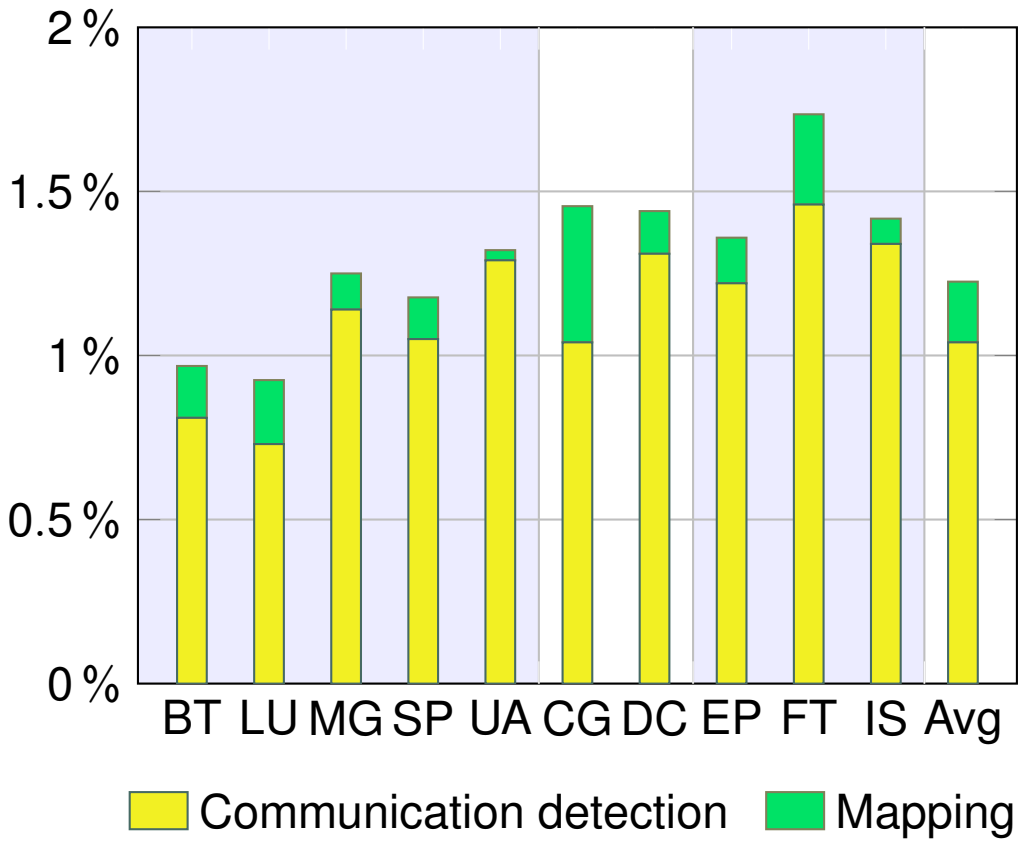


RESULTS – OVERHEAD

- Sources of overhead:
  - Communication detection (+ additional page faults)
  - Mapping

RESULTS – OVERHEAD

- Sources of overhead:
  - Communication detection (+ additional page faults)
  - Mapping



CONCLUSIONS

- **Page faults of a parallel application can be used to analyze its memory access behavior**
- SPCD: kernel-based mechanism to detect inter-thread communication
  - Low overhead
  - No need to modify applications or support libraries
  - Provide detection and migration during execution of the application, no prior knowledge required
- Results
  - Improve performance by up to 16.7% (avg 4.7%)
  - Reduce energy consumption by up to 14.7% (avg 5.2%)

Download: `http://inf.ufrgs.br/~mdiener`

Communication-Based Mapping Using Shared Pages

Matthias Diener, Eduardo H. M. Cruz, Philippe O. A. Navaux

Federal University of Rio Grande do Sul, Porto Alegre, Brazil
{*mdiener,ehmcruz,navaux*}*@inf.ufrgs.br*

http://inf.ufrgs.br/~mdiener

IPDPS 2013 – Boston, MA
May 22, 2013

CAPES
CNPq Conselho Nacional de Desenvolvimento Científico e Tecnológico
INF INSTITUTO DE INFORMÁTICA UFRGS