Locality vs. Balance: Exploring Data Mapping Policies on NUMA Systems

Matthias Diener, Eduardo Cruz, Philippe Navaux

Federal University of Rio Grande do Sul (UFRGS), Porto Alegre, Brazil

PDP 2015 – Turku, Finland

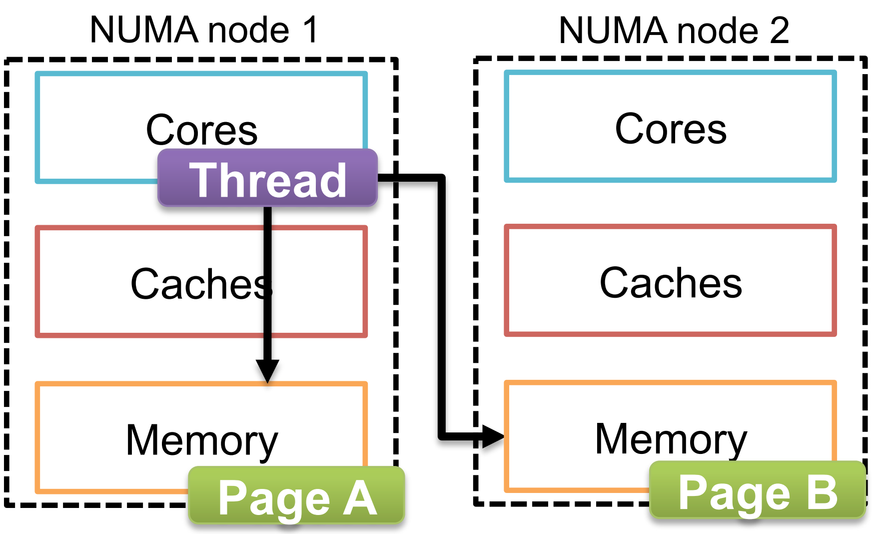
March 4th, 2015

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

INSTITUTO DE INFORMÁTICA UFRGS

The data mapping problem

- Shared-memory architectures with **multiple memory controllers (MC)**
  - Non-uniform memory access (**NUMA**) behavior
  - Each controller forms a NUMA node, can access its own part of the main memory



Data mapping: map memory pages to NUMA nodes to improve memory access performance

Locality and balance

- Basic data mapping policy: **first-touch**
  - Map page to **first** node that accesses it

- Types of improved policies
  1. **Locality-based** mapping
     - Traditional policy to improve data mapping
     - Minimize remote memory accesses
     - Map pages to the node with the **most** accesses
  2. **Balance-based** mapping
     - Recent policies for modern NUMA systems
     - Avoid overloading controllers
     - Map pages in such a way that the load is balanced

1. Which memory access behavior can be improved by mapping?
2. Which improvements can be achieved with different policies?

Contributions

1. **Metrics** to describe memory access behavior of parallel applications.
2. **Mapping policies** that improve locality, balance or both.
3. **Evaluation** of a large set of parallel applications on several NUMA platforms.

Describing the memory access behavior of parallel applications

Memory access metrics

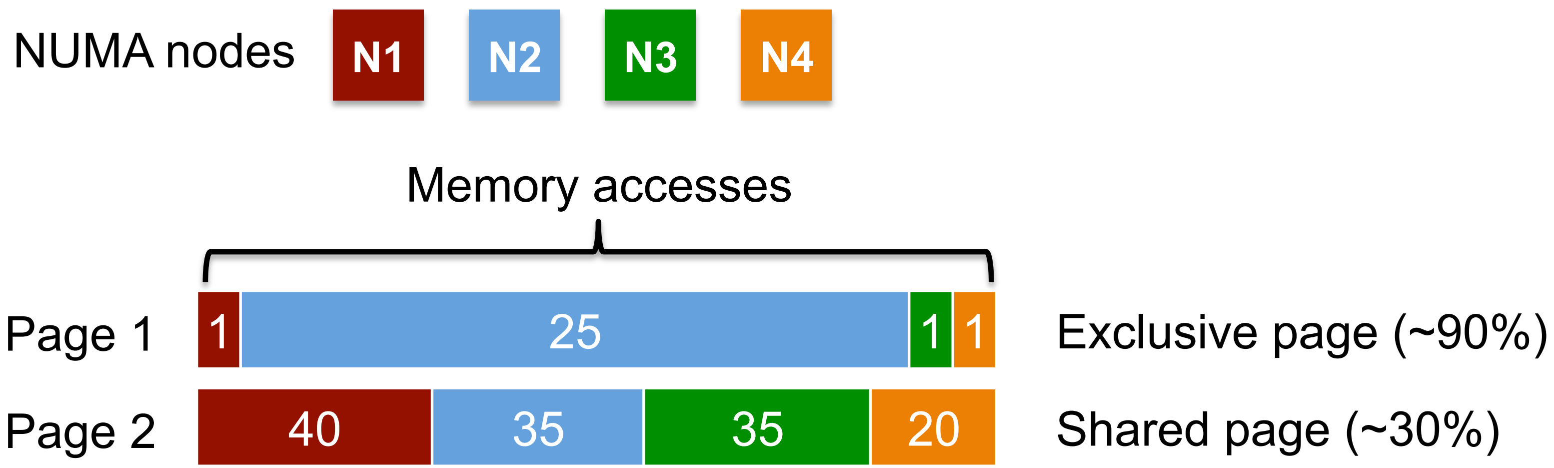
- Two sets of metrics:
  - **Exclusivity** (locality-based mapping)
    - Property of application
  - **Balance** (balance-based mapping)
    - Property of application and data mapping policy

- Describe per-node memory access behavior

- Granularity: memory page size
  - Currently: several KByte (4 KByte on x86 by default, our baseline)
  - Future systems may have larger pages

Locality metrics – Exclusivity

- **Page Exclusivity:** highest % of memory accesses from same NUMA node

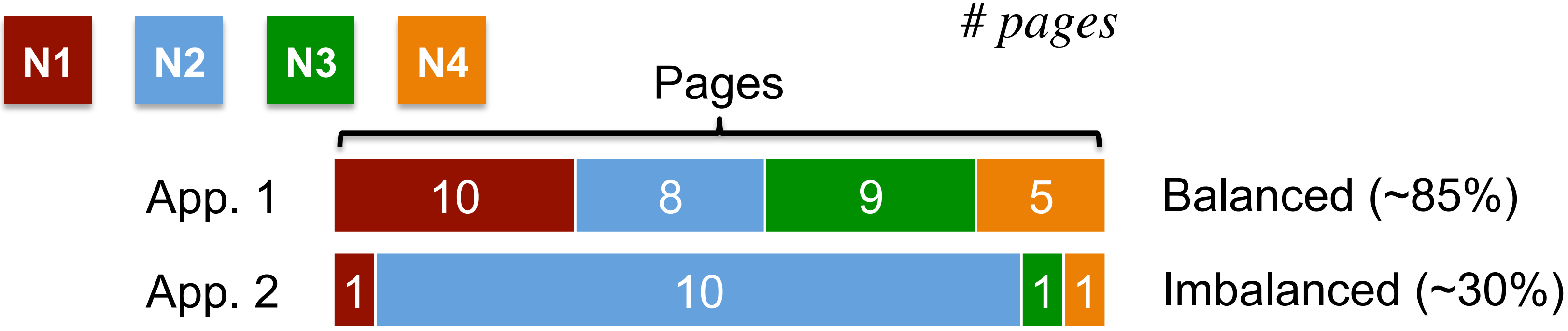


- **Application Exclusivity:** Calculate weighted exclusivity for all pages:
  Page 1: 28 accesses, Page 2: 130 accesses
  (28*0.9 + 130*0.3)/2 = **64%**

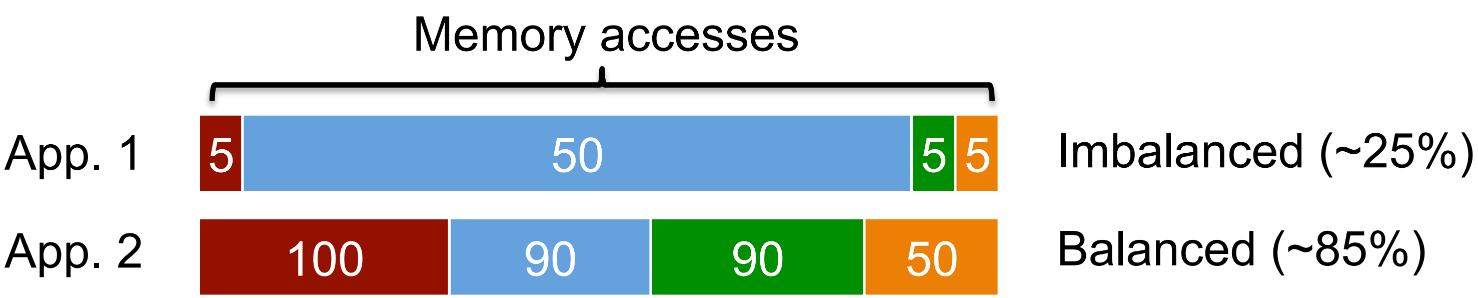
- Min: 1/#nodes (fully shared), Max: 100% (fully exclusive)
- Expect higher improvements from locality-based mapping with high exclusivity

Balance metrics

- **Page balance** $1 - \frac{\max(\#\, pages\_node) - \min(\#\, pages\_node)}{\#\, pages}$



- **Memory access balance:** calculate weighted balance



- Min: 0% (all pages/accesses to single node), Max: 100% (fully balanced)
- Expect more improvements from balanced-based policy with low balance

Mapping policies

Mapping policies

- Policies
  - **First-touch**
  - **Random**, **Interleave** (independent of memory access behavior)
  - **Locality**, **Balanced**, **Mixed** (depend on memory access behavior)

- First-touch
  - Page is mapped to NUMA node with first memory access to page
  - Default policy of most OS (baseline)
- Random
  - Assign pages to nodes randomly
  - Validate importance of mapping
- Interleave
  - Traditional policy to balance pages between nodes
  - node[p] = address(p) mod #nodes

Mapping policies (ctd.)

- Locality
  - Traditional policy to maximize memory access locality
  - node[p] = node_with_most_accesses(p)

- Balanced
  - Balance number of memory accesses per NUMA node
  - Algorithm
    - Sort pages by number of memory accesses
    - Place each page on the node with the most accesses to it, that is not overloaded yet
  - Complementary policy to Locality
    - Favor balance over locality vs. favor locality over balance

Mapping policies (ctd.)

- Mixed
  - Apply locality policy for highly exclusive pages
  - Shared pages: distribute with Interleave policy

$$\text{node[p]} = \begin{cases} \text{node\_with\_most\_accesses(p)}, & \text{if exclusivity} > \text{threshold} \\ \text{node[p] = address(p) mod \#nodes}, & \text{otherwise} \end{cases}$$

  - Trade-off between Locality and Balance
  - Evaluated various thresholds, 90% gave highest improvements

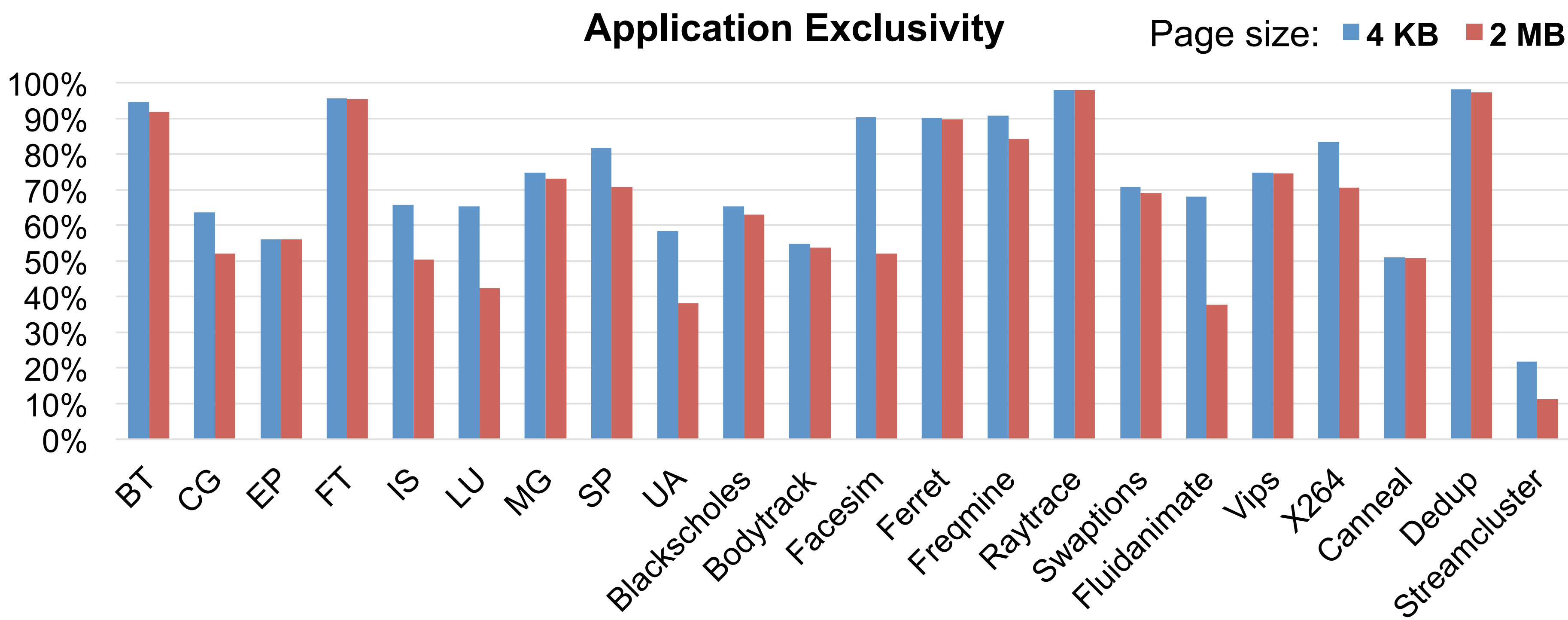
Benchmark metrics

Methodology

- Evaluate memory access behavior with memory tracing tool built with Pin
- Benchmark suites
  – NAS Parallel Benchmarks, OpenMP version (*C* input size)
  – PARSEC (*native* input size)
- 64 threads; 4 NUMA nodes for balance metrics
- Behavior is constant with same input data/number of threads

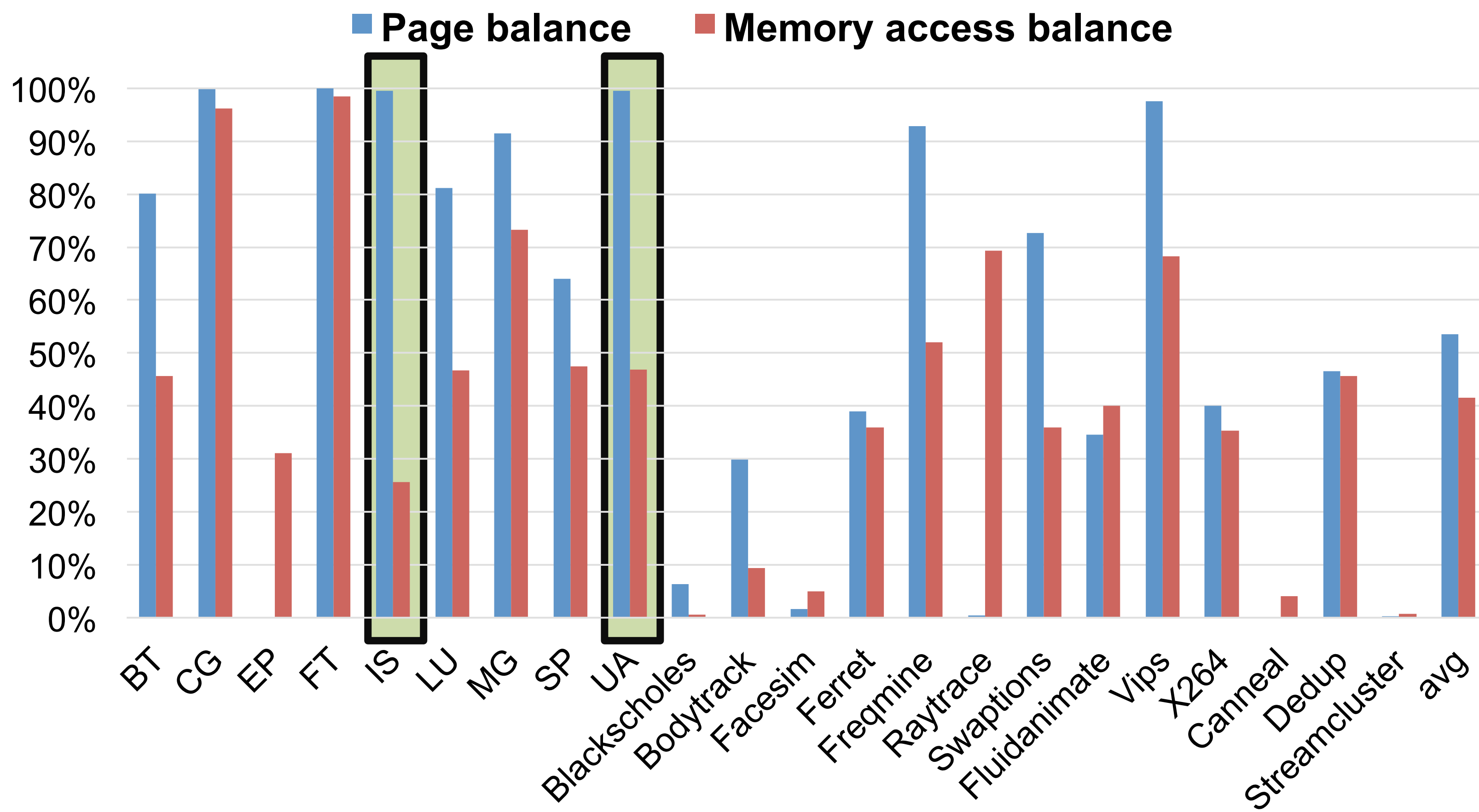
- Measure
  – Application exclusivity for 2 page sizes
  – Page and memory access balance for all mapping policies

Application Exclusivity

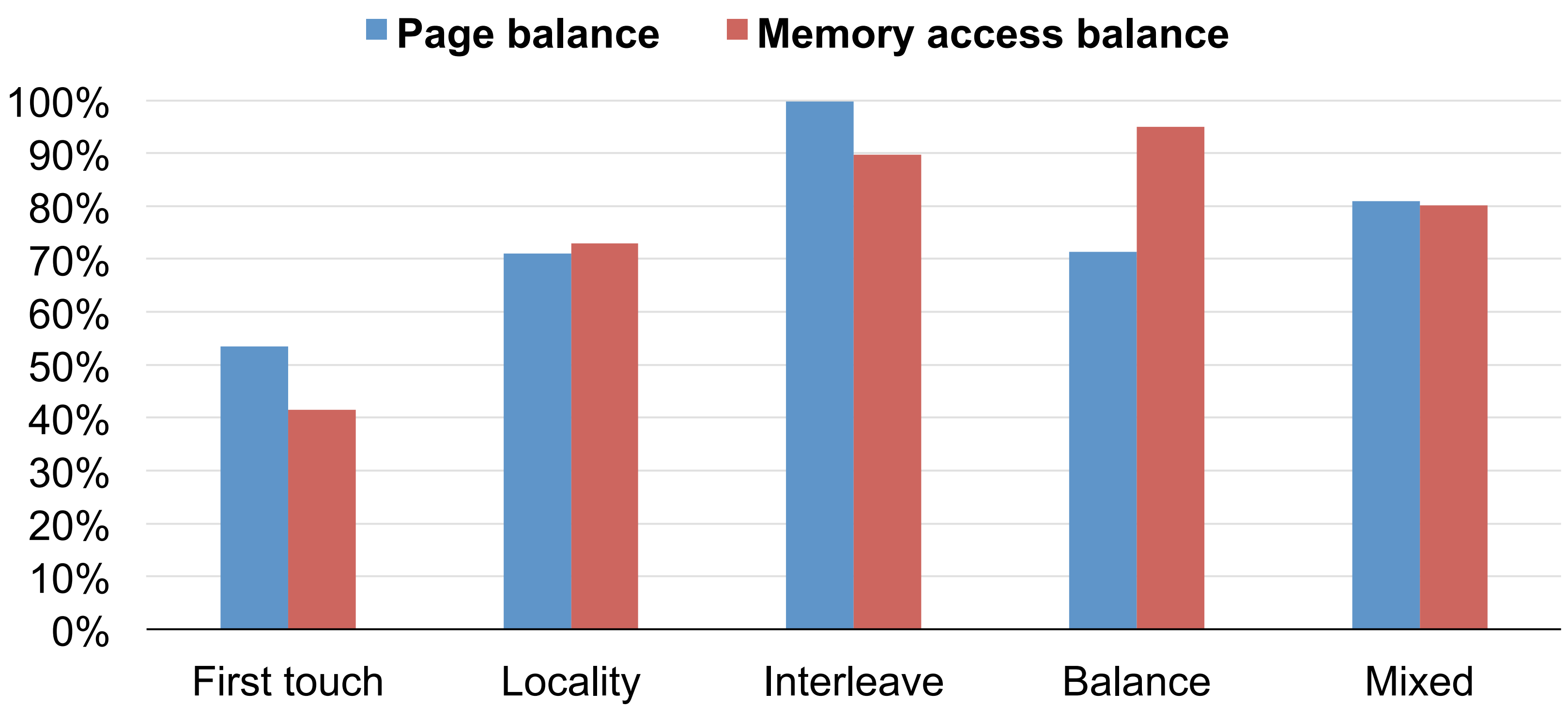


Average: 73% (4KB), 65% (2MB)

Balance – First touch



Benchmark metrics – Balance



Summary of benchmark behavior

- Exclusivity high for most applications
  - Suitable for locality-based data mapping
  - Decreases only slightly for larger pages

- Most applications imbalanced
  - First-touch from master thread
  - Suitable for balance-based data mapping
  - Locality already balances better than first-touch

Performance Evaluation

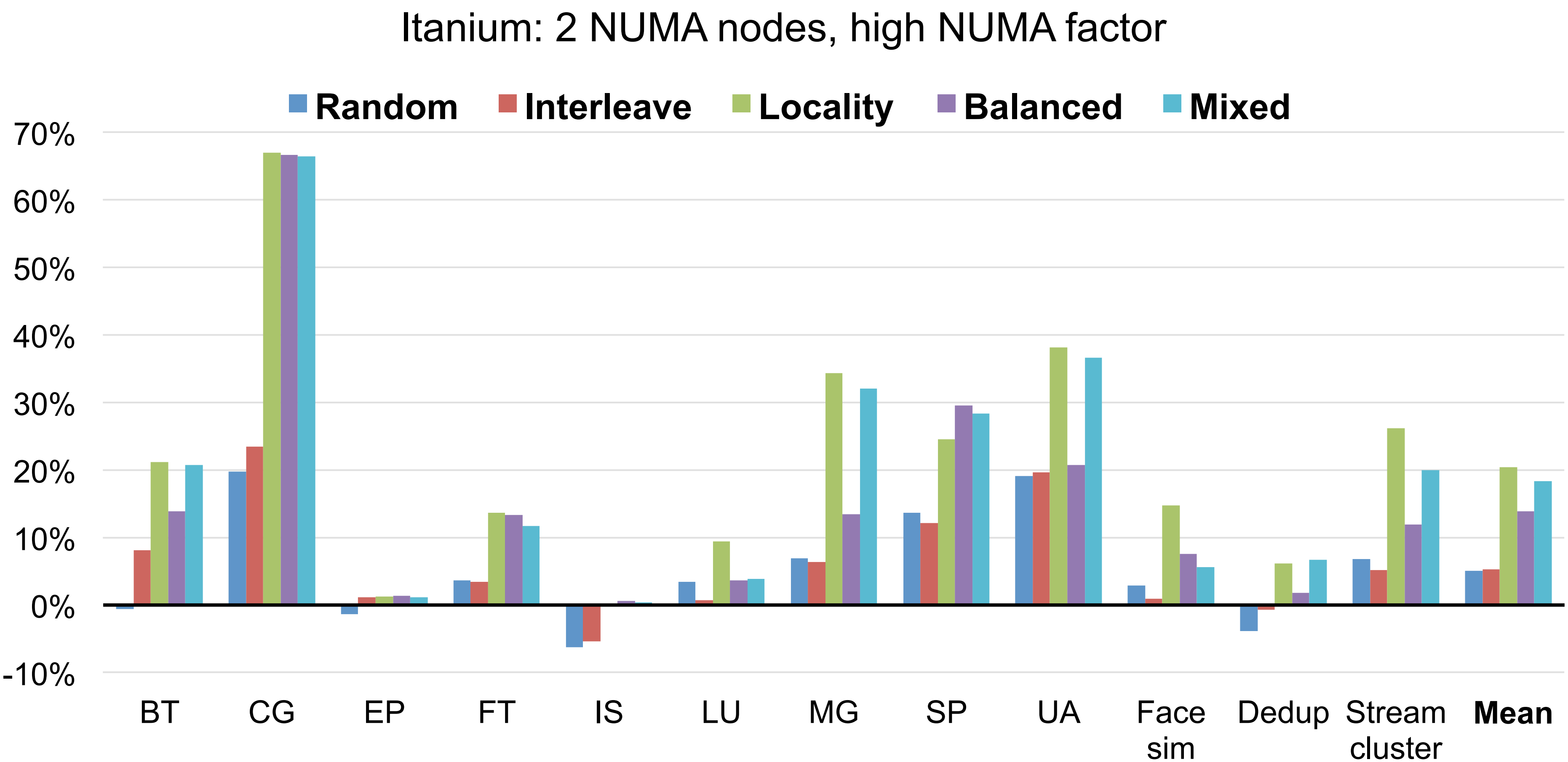
Methodology

- 3 NUMA machines: Itanium, Xeon, Opteron
  - NUMA factor: latency(remote access) / latency(local access)

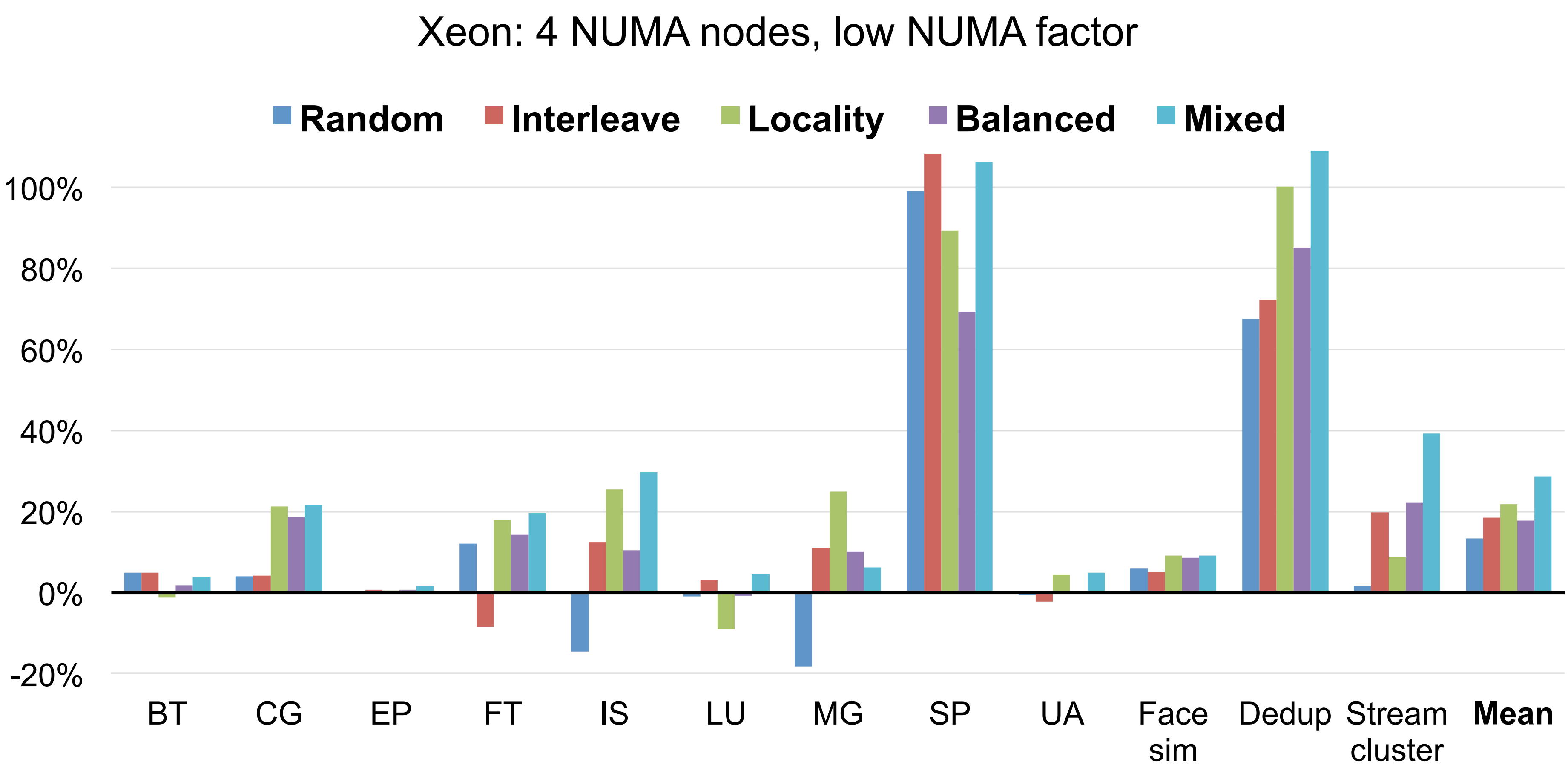
Machine	# NUMA nodes	# PUs	Processors	Page size	NUMA factor
Itanium	2	8	4x Itanium 9030, 2 cores	16 KB	2.1
Xeon	4	64	4x Xeon X7550, 8 cores, SMT	4 KB	1.5
Opteron	8	64	4x Opteron 6386, 8 cores, SMT	4 KB	2.8

- Same benchmarks/input sizes as before
  - Use memory tracer + offline data mapping algorithm
  - No benchmarks whose memory addresses change between executions
- Execute with 8/64/64 threads
- Compare all mapping policies
  - Performance improvements to first-touch mapping

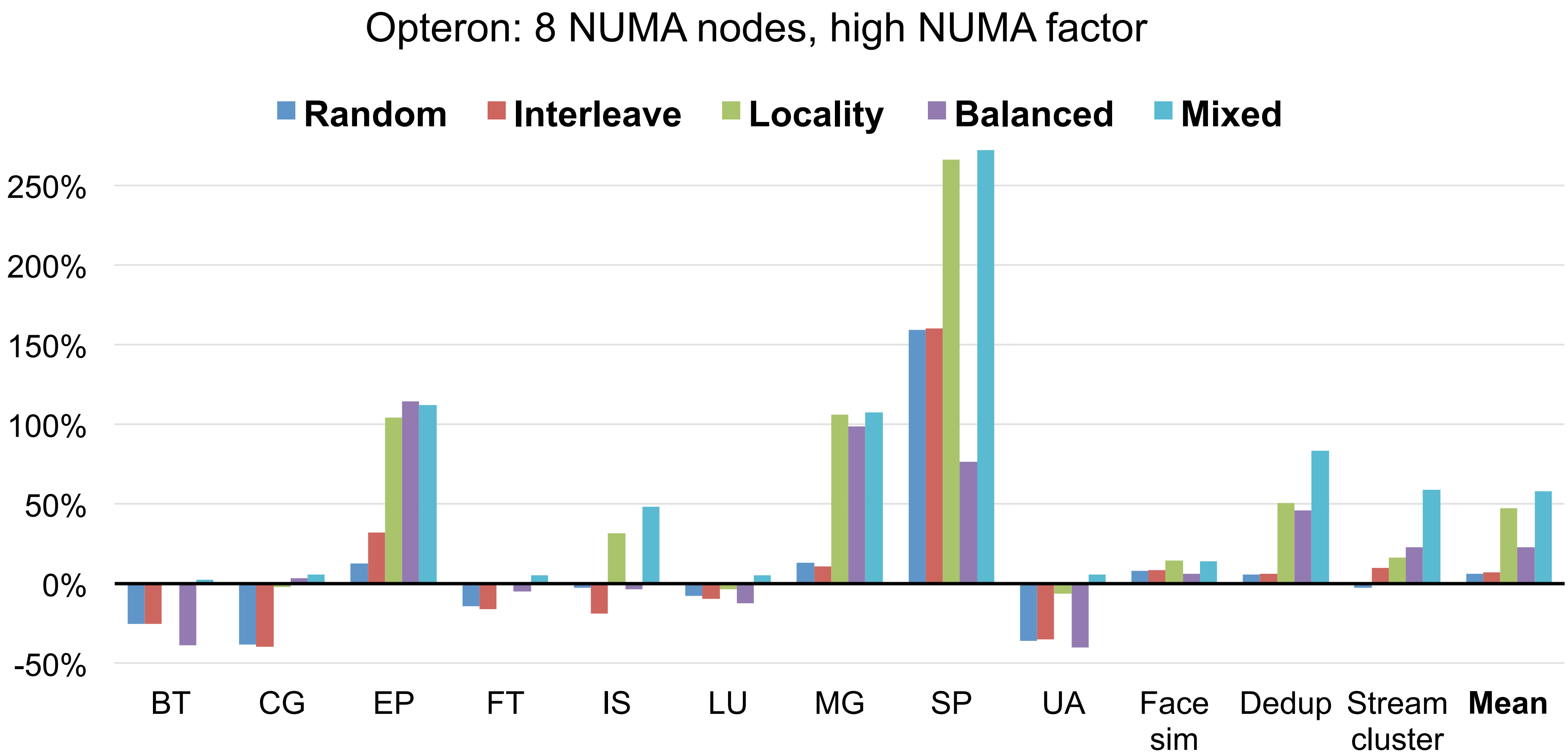
Performance improvements – Itanium



Performance improvements – Xeon



Performance improvements – Opteron



Summary of results

- First-touch has a negative impact on performance
  - Even random page assignment is better
- Locality is still more important than balance
  - Results of Locality > Balanced
- Mixed policies can achieve the highest improvements
- What can be done?
  - Application developers
    - Prepare application for first-touch
    - Access pages only from single thread to increase exclusivity
  - OS developers
    - Use mechanisms to refine data mapping during execution
      - Basics already done (NUMA balance for Linux, …)

Conclusions

Conclusions

- Data mapping has a substantial influence on the performance of NUMA machines
  - Gains expected to rise in the future
  - Need to choose correct policy

- Memory access locality is most important metric to improve
  - Even on current NUMA architectures

- Policies that combine Locality and Balance can result in the highest improvements

- Future work
  - Apply policies to online mechanisms (no need for profiling step before execution)

Locality vs. Balance: Exploring Data Mapping Policies on NUMA Systems

Matthias Diener, Eduardo Cruz, Philippe Navaux

Federal University of Rio Grande do Sul (UFRGS), Porto Alegre, Brazil

`mdiener@inf.ufrgs.br`

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

PDP 2015 – Turku, Finland

March 4th, 2015

.inf INSTITUTO DE INFORMÁTICA UFRGS

Dynamic behavior