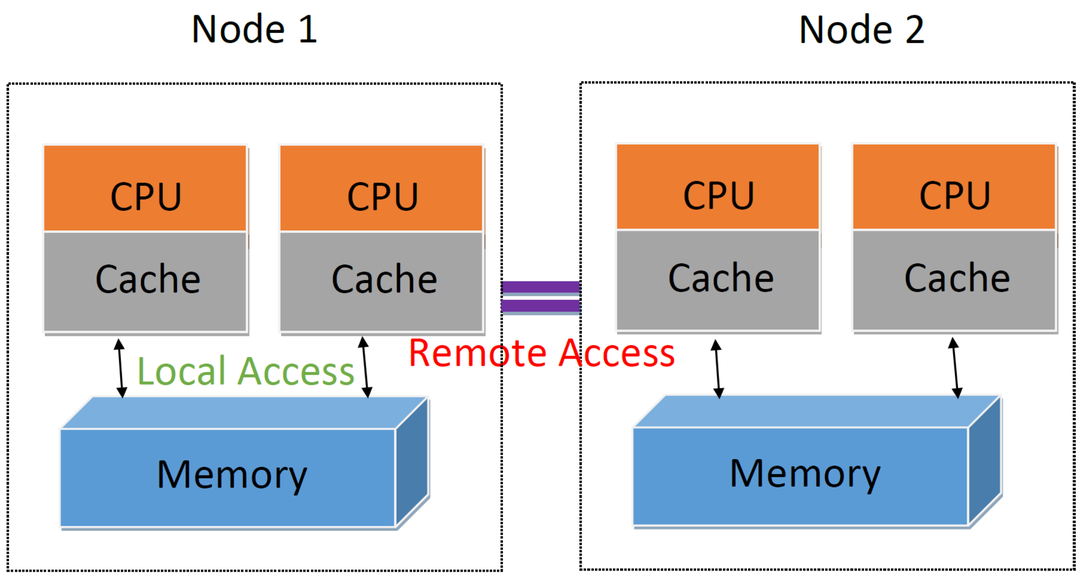
NUMA Awareness
Improving Memory and Thread Management in the JVM

Maria Patrou, Kenneth B. Kent, Gerhard Dueck, Charlie Gracie
University of New Brunswick, IBM Canada
Faculty of Computer Science
{Maria.Patrou, ken, gdueck}@unb.ca, charlie_gracie@ca.ibm.com

Non Uniform Memory Access - NUMA

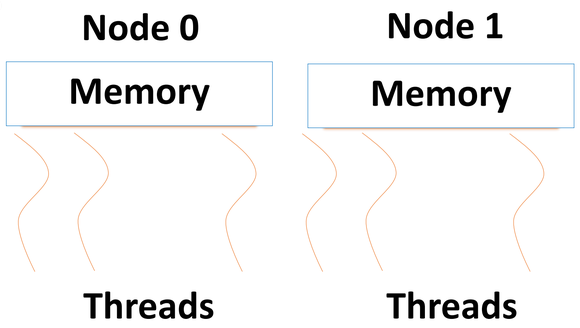


But, why should we use NUMA?

- The main benefit of NUMA over traditional UMA is scalability.

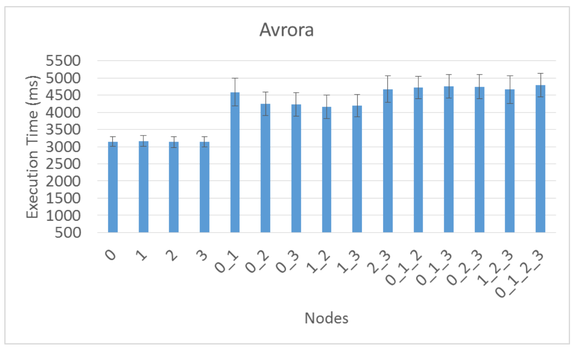
NUMA Design in Language Runtimes

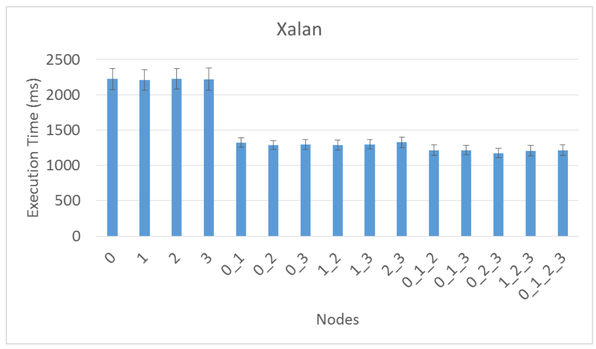
- Java Virtual Machine (JVM): memory from physical nodes is used.
- Threads are pinned to different Nodes, using memory from them.
- During Garbage Collection, memory is reclaimed by the GC threads

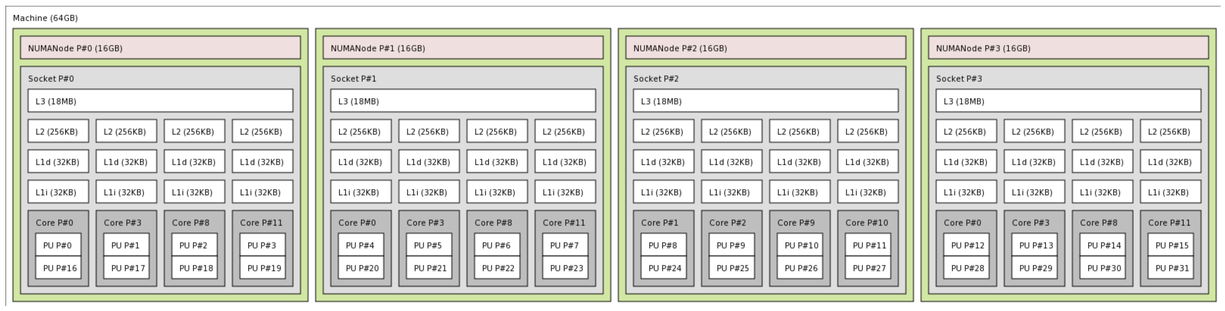


Performance

- Degradation of performance during Garbage Collection.
- Proof of Concept Experiments (DaCapo):







Proposed Work

Goal

- Minimize Remote Accesses
- Without causing work overload on Nodes
- Based on the architecture of each Node

How?

- Memory Organization: keep memory closer to threads
- Thread Affinity: different policies based on the server hardware characteristics

Devise a NUMA topology aware algorithm, that will take into consideration the hardware in Automated Memory Management Operations