Shoal: self-tuning for memory allocation and access
based on access patterns and hardware → helps to tackle hardware complexity

Problem

multicores: complex memory subsystems
• non-uniform memory access (NUMA)
• complex interconnects
• hardware features (DMA, large/huge page)
future:
• global address space?
• cache coherence?
→ Hard to program

Results

Scalability

Memory controller throughput

Copy using DMA engine

Comparison array backends

Ideas

Auto-Tune Memory Layout
• memory abstraction
• annotate memory allocation
• memory access patterns
  • from high-level languages
  • manually specified
• use hardware characteristics
• memory abstraction (arrays)
• specialized implementations (replication, distribution)

Auto-Tune Memory Layout
• memory abstraction
• annotate memory allocation
• memory access patterns
  • from high-level languages
  • manually specified
• use hardware characteristics
• memory abstraction (arrays)
• specialized implementations (replication, distribution)

Abstraction

• decouples memory from program logic
→ exchange array implementation online
• high-level operations (copy, initialize..) → transparent use of large page sizes, DMA engines

Conclusion

• automatic scalability
• good performance across various NUMA machines
• no programmer efforts

Future work

• synchronization
• scheduling
• accelerators (Xeon Phi)
• multiple address spaces