Research at Scalable Computing Software Laboratory

Data Centric Computing

Xian-He Sun
Department of Computer Science
Illinois Institute of Technology
The Scalable Computing Software Lab

www.cs.iit.edu/~scs/

Personnel
2 Faculty, 2 Post-docs, 2 Pre-doc, 2 Visitors

Facility
Clusters, Parallel Machine, Grid

Distributed Optical Testbed (Grid)

I-WIRE OMNI
High Performance Computing at SCS

- Data intensive computing
- Faulty tolerant
- Energy saving
- Reduce data access delay

Scalable Supercomputer

Cloud Computing

Currently Support
- NSF (5)
- DoE (SciDAC)
- Microsoft, Argonne, Fermi
Processor-memory performance gap

- Processor performance increases rapidly
  - Uni-processor: ~52% until 2004, ~25% since then
  - New trend: multi-core/many-core architecture
    - Intel TeraFlops chip, 2007
  - Aggregate processor performance much higher
- Memory: ~9% per year
- Processor-memory speed gap keeps increasing

Source: Intel
Source: OCZ
Data-Centric Computing

- Data access is the bottleneck needs attention
- Need to rethinking of system design to reflect the fact

**Our Solutions**
- Data Prefetching
- Data Layout
- Data-centric Scheduling
- Data-centric Architecture
- Integrated optimization
- Understanding memory system
- Understanding design trade-off

Dynamic Application-aware Data-Centric Optimization

10/9/2011
Prefetching

- Prefetch data as close as possible to the processor in the memory hierarchy
- Key to prefetching
  - What data should be prefetched?
  - When should prefetching occur?
- Limitation of current Prefetching
  - Conservative and limited to static prediction strategies
  - Only works for simple access patterns with locality
What Data: Data Access History Cache and Dynamic hardware prefetch
**When**: Timing in Multi-streaming Prefetching

- L2 cache miss stream is fed to prefetcher.
- The global stream is localized into local streams by PC.
- The local streams are chained according to their last accessed time.
- Adds time information to the local streams.
Software Solution: *Server Push I/O Architecture*

- Dynamic I/O architecture
  - Optimize I/O architecture for each application
- Use a dedicated data server for
  - Finding data access signature
  - Data prefetching
  - Data layout
  - Optimization and coordination
- Carry the data access service via
  - Enhanced parallel I/O file system
  - Special designed parallel cache system
- Explore various strategies and adaptive support
- Combine merits of prediction, post analysis, and pre-execution

10/9/2011
Result: Two approaches for I/O prefatching
Result: Smart Data Layout

Round-robin may not be the best for parallel I/O
Smart three dimension data layout
Optimization: Coordination & Load Balance

- Stripe size also affects I/O workload on multiple file servers
- Scheduling issues

I/O Request

<table>
<thead>
<tr>
<th></th>
<th>4K</th>
<th>12K</th>
</tr>
</thead>
<tbody>
<tr>
<td>I/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Servers</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Balanced

Imbalance

I/O Servers

Strip Size 1KB

Strip Size 4KB

Strip Size 8KB

Strip Size 16KB
Core-aware memory scheduling
Integrated Optimization: A system approach
Data-Centric Computing

- Data access is a complex matter
- Dynamic, Application-aware
- System and algorithm re-design

Big Data, Big Deal, Big Opportunity

Operation of Memory Hierarchy

Layers of parallel I/O

Application  MPI  forwarding  PFS