Introduction to Data Center Computing

Derek Murray
What we’ll cover

• Techniques for handling “big data”
  – Distributed storage
  – Distributed computation

• Focus on recent papers describing real systems
Example: web search

Crawling → Indexing → Querying

WWW
A system architecture?
Distributed storage

- High volume of data
- High volume of read/write requests
- Fault tolerance
Brewer’s CAP theorem

- Consistency
- Availability
- Partition Tolerance
The Google file system

Client

GFS Master

Chunk server

Chunk server

Chunk server
Distributed computation

• Parallel distributed processing
• Single Program, Multiple Data (SPMD)
• Fault tolerance
• Applications
Task farming
MapReduce

Input → Map → Shuffle → Reduce → Output
Dryad

- Arbitrary task graph
- Vertices and channels
- Topological ordering
DryadLINQ

• Language Integrated Query (LINQ)

```csharp
var table = PartitionedTable.Get<int>("...");

var result = from x in table
              select x * x;

int sumSquares = result.Sum();
```
Scheduling issues

• Heterogeneous performance

• Sharing a cluster fairly

• Data locality
References

- **Storage**
  - DeCandia *et al.*, “Dynamo: Amazon’s Highly-Available Key-value Store”, *Proceedings of SOSP 2007*
  - Burrows, “The Chubby lock service for loosely-coupled distributed systems”, *Proceedings of OSDI 2006*

- **Computation**
  - Dean and Ghemawat, “MapReduce: Simplified Data Processing on Large Clusters”, *Proceedings of OSDI 2004*
  - Isard *et al.*, “Dryad: Distributed Data-Parallel Programs from Sequential Building Blocks”, *Proceedings of EuroSys 2007*
  - Yu *et al.*, “DryadLINQ: A System for General-Purpose Distributed Data-Parallel Computing Using a High-Level Language”, *Proceedings of OSDI 2008*
  - Olston *et al.*, “Pig Latin: A Not-So-Foreign Language for Data Processing”, *Proceedings of SIGMOD 2008*

- **Scheduling**
  - Zaharia *et al.*, “Improving MapReduce Performance in Heterogeneous Environments”, *Proceedings of OSDI 2008*
  - Isard *et al.*, “Quincy: Fair Scheduling for Distributed Computing Clusters”, *Proceedings of SOSP 2009*
Conclusions

• Data centers achieve high performance with commodity parts

• Efficient storage requires application-specific trade-offs

• Data-parallelism simplifies distributed computation on the data