Failure-Awareness and Dynamic Adaptation in Data Scheduling

Mehmet Balman
MS Thesis
Department of Computer Science
Louisiana State University

Research Goal

"Reliability and Efficiency" for wide-area Data Access

Outline

- The Data Placement Challenge
- Lessons Learned from Computer Architecture
- Adaptive Data Scheduling
- Failure-Aware Data Placement
- Conclusion

Large Scale Applications

- **Science**
  - Astronomy - SuperNova, LSST(Large Synoptic Survey Telescope)
  - Biology (bimolecular computing)
  - Climate research
  - High Energy Physics (Cern)
- **Business**
- Credit Card Fraud detection
  - (historical data, analyze transactions)
- Data mining for brokerage and customer services
- Oil and electronic design companies
  - (long term batch processes)
- Medical institutions
  - (computational network, large image transfers)

Data Deluge

- Scientific and Business applications becoming more data-intensive
- Huge Computational requirements
- Immense data sets (real time processing of data)

Data-intensive Computing

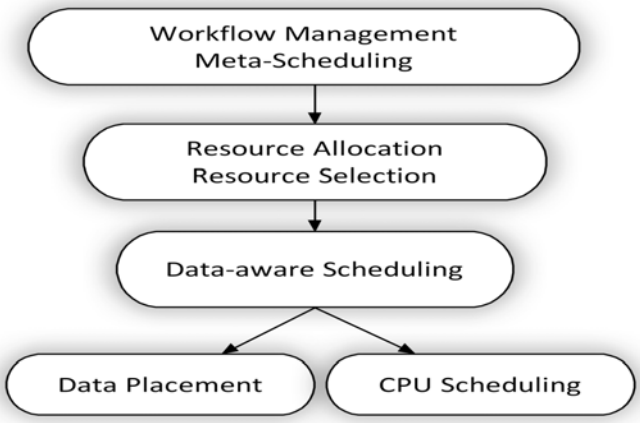
- Using Distributed Resources to satisfy excessive computation requirements
- Data to be shared between geographically distributed sites
- Complex workflow characteristics
- High capacity, fast storage systems

Data Scheduling

- Make data placement a first **class citizen**
- Orchestrating data placement jobs

Stork www.storkproject.org

Data-Aware System Model



Key Attributes affecting Data Placement Performance

	In Single Host	Between a Pair of Hosts	Multiple Servers to Single Server	Between Distributed Servers
Available Storage Space	✓	✓	✓	✓
CPU Load and Memory Usage	✓	✓	✓	✓
Transfer Protocol Performance		✓	✓	✓
Number of Parallel Connections		✓	✓	✓
Network Bandwidth and Latency			✓	✓
Number of Concurrent Operations			✓	✓
Ordering of Data Placement Tasks				✓

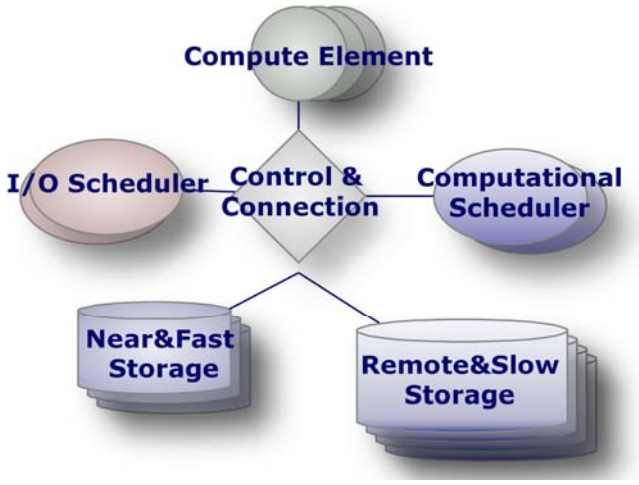
Contribution

- Failure-Aware Data Placement Paradigm for increased Fault-Tolerance
- Adaptive Scheduling of Data Placement Tasks

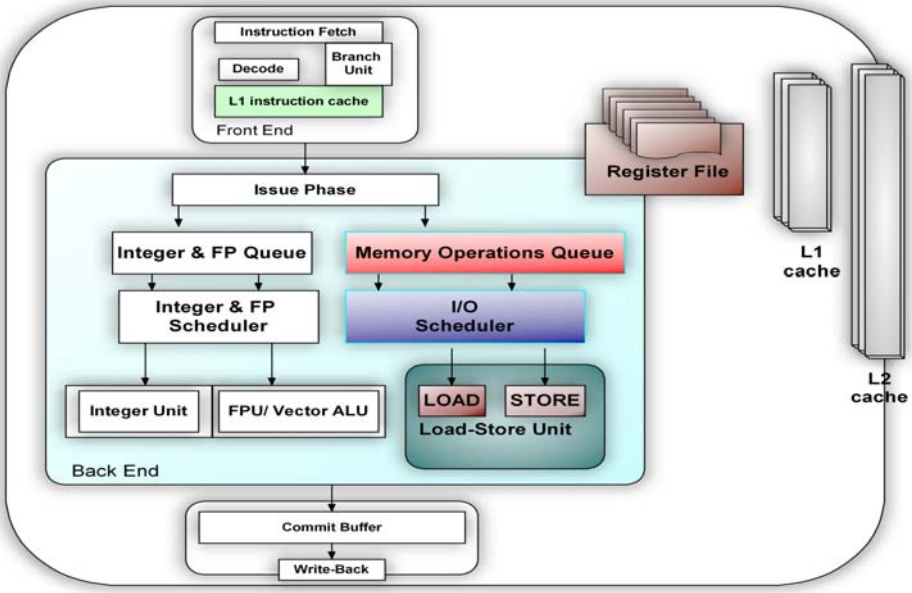
Outline

- The Data Placement Challenge
- Lessons Learned from Computer Architecture
- Adaptive Data Scheduling
- Failure-Aware Data Placement
- Conclusion

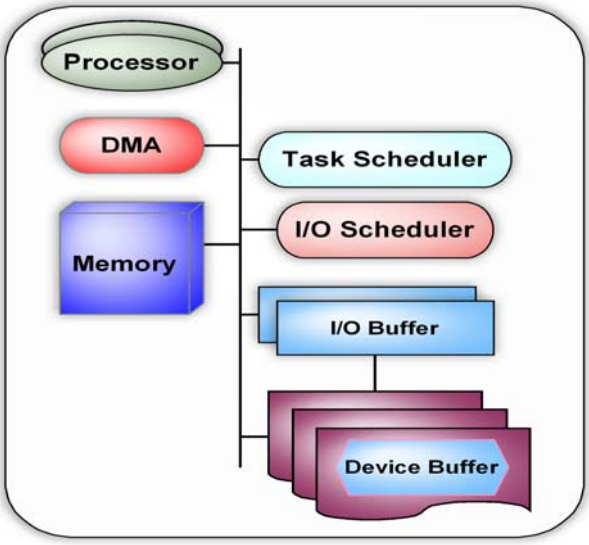
Generic Model



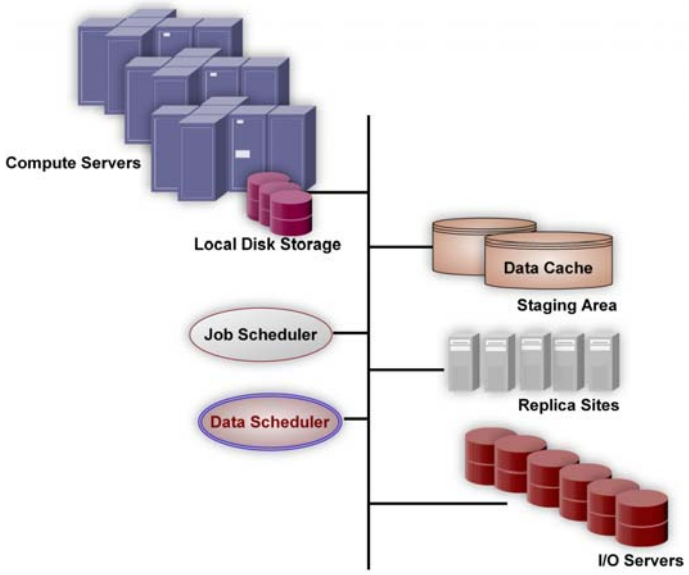
Microprocessor



Operating System



Distributed Systems



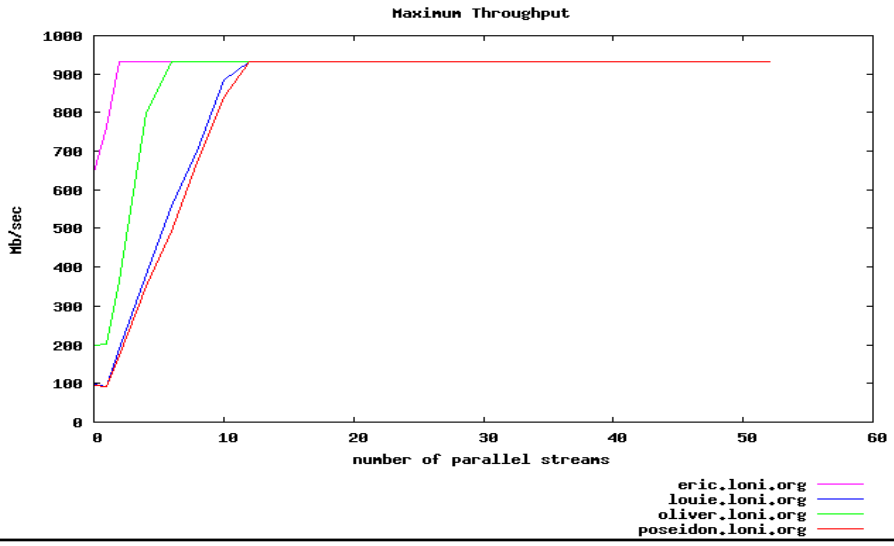
Outline

- The Data Placement Challenge
- Lessons Learned from Computer Architecture
- Adaptive Data Scheduling
- Failure-Aware Data Placement
- Conclusion

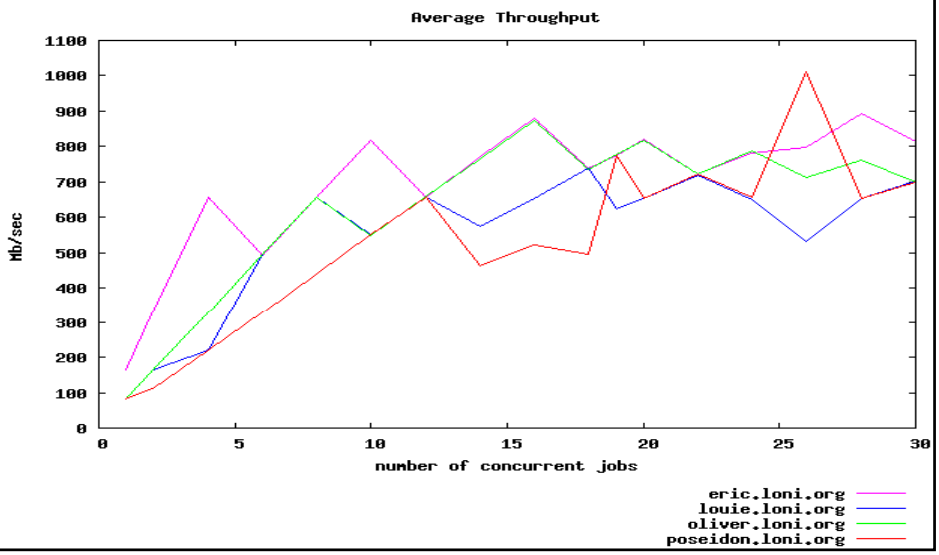
Adaptive Scheduling

- Dynamic Parameter Tuning
  - Parallel Stream
    - Aggregate TCP connections
  - Concurrent Jobs
- Aggregation of Data Placement Job
    - Source/Destination pair

Impact of Parallelism



Concurrent Jobs

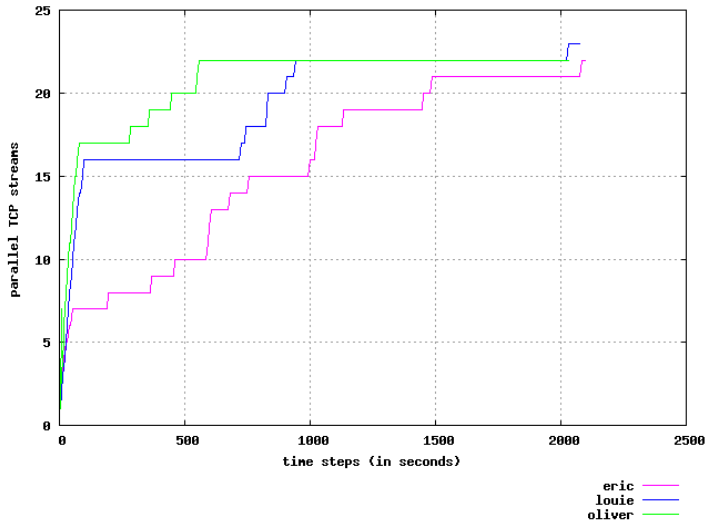


Dynamic Parameter Setting

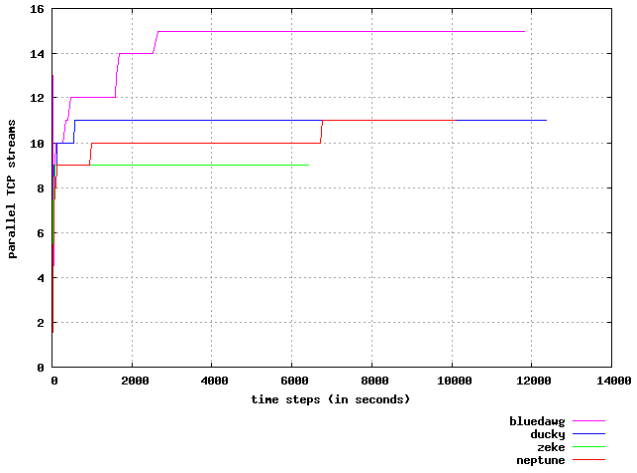
- Low integration cost (no external profilers)
- Adapt to changing network conditions
- No high level predictors

- Increase level of parallelism gradually
  - Can we set the number of parallel streams while transfer is in progress?

Adaptive Tuning of Parallel Streams



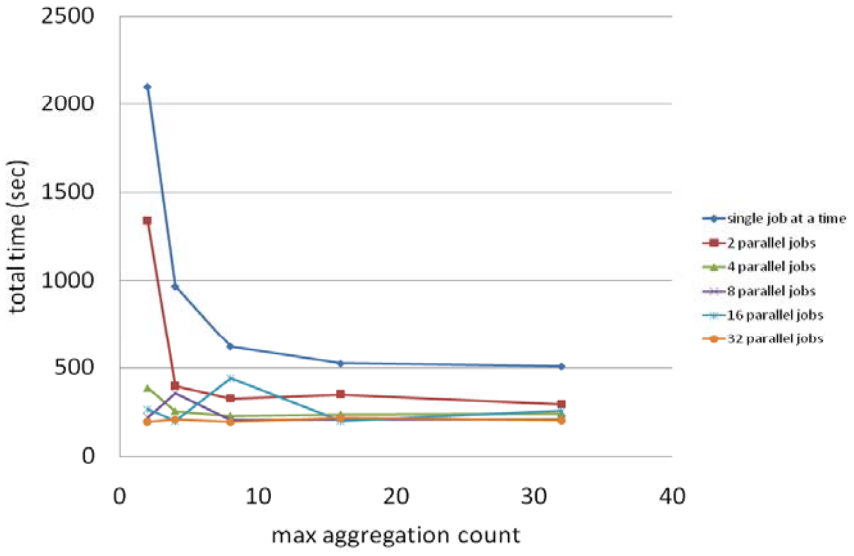
Adaptive Tuning of Parallel Streams



Job Aggregation

- Aggregate data transfer jobs into a single job
- Eliminate the cost of connection for each transfer
- Major performance improvement
  - Especially with small files

Job Aggregation



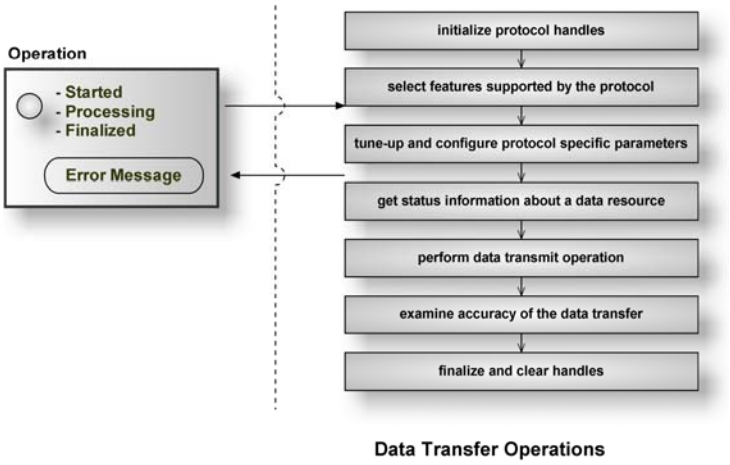
Outline

- The Data Placement Challenge
- Lessons Learned from Computer Architecture
- Adaptive Data Scheduling
- Failure-Aware Data Placement
- Conclusion

Failure-Awareness

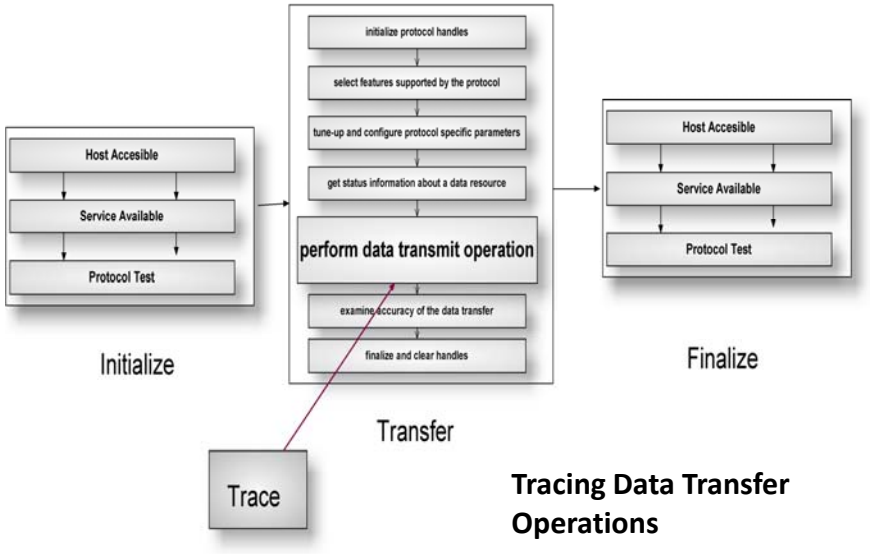
- Early Error Detection
  - Network Exploration
- Error Classification and Reporting
- Adapt to Failures (Retry?)

Error Reporting Framework



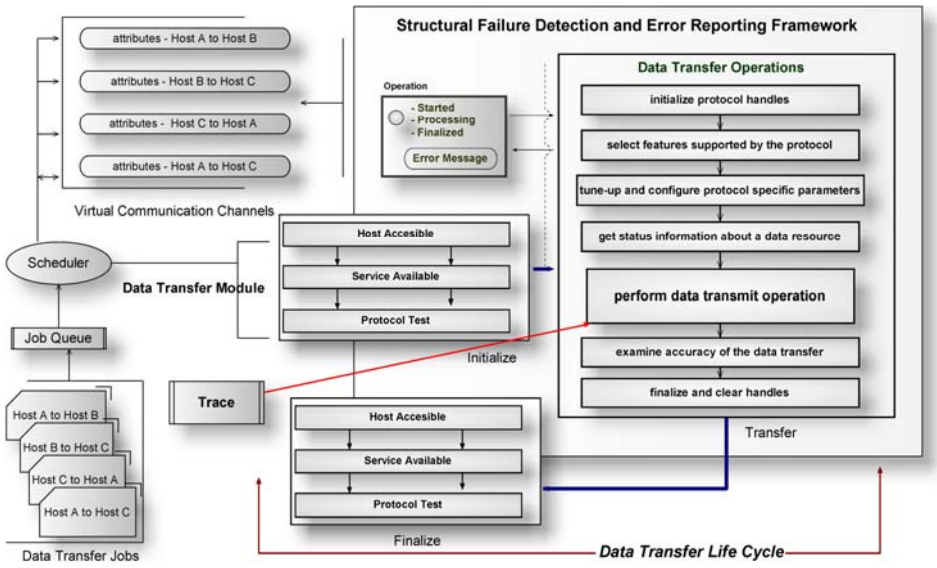
Data Transfer Operations

Data Transfer Life Cycle

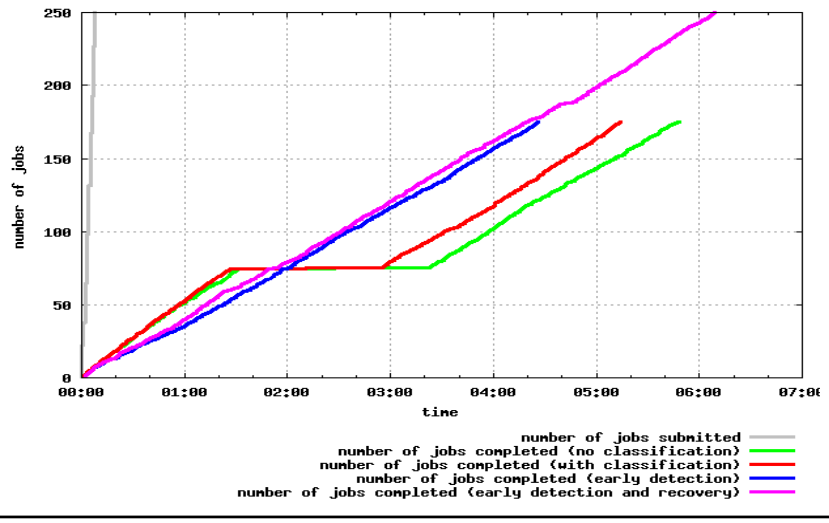


Tracing Data Transfer Operations

Integration



Failure-Awareness



Outline

- The Data Placement Challenge
- Lessons Learned from Computer Architecture
- Adaptive Data Scheduling
- Failure-Aware Data Placement
- Conclusion

Conclusion

- An Adaptive Approach for Parameter Tuning
- Early Error Detection and Error Classification
- Failure-Awareness in Scheduling
- Aggregation of Data Placement Jobs

Broader Impact

- Stork
  - http://www.storkproject.org/
- Petashare (petaFS & petaShell)
  - http://www.petashare.org/
- I/O aggregation
  - ***IRODS*** FUSE and IRODS ***Parrot*** clients
    - 3-fold performance increase
- Stork.globus-url-copy
  - Extending **globus-url-copy**
    - New features:
      - Checkpointing (rescue file for restart)
      - Network explorations
      - Checksum verification

      Auto Tuning the number of Parallel Streams

Future Research Problems

- Semantic Compression
  - For better end-to-end performance
- Utilizing Replicated Data
- Distributed Scheduling
  - Job delegation