Interactive Data Processing at Massive Scale

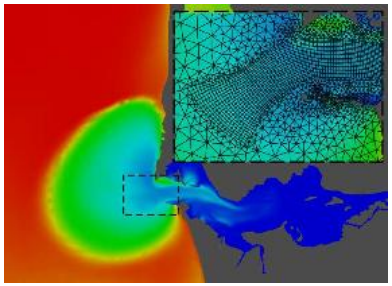
Magdalena Balazinska
University of Washington
http://www.cs.washington.edu/homes/magda

Microsoft Research

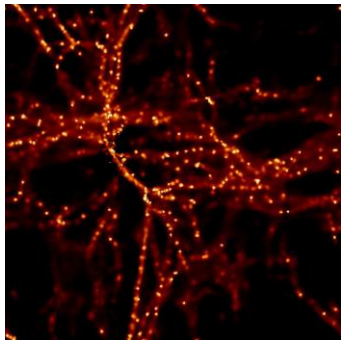
Nuage Project

- Science is becoming a data management problem
- Existing database management systems are insufficient
  - Wrong data model, wrong features, insufficient scalability

- Nuage project goals (http://nuage.cs.washington.edu/)
  - Focus on scientific applications
  - Massive-scale parallel query processing
  - Cloud computing: DBMS as a service for science
- Current collaborators/applications:



Oceanography
Bill Howe, UW eScience



Astronomy
Jeff Gardner
Andrew Connolly

Astronomy Simulation Use Case

- Evolution of large scale structure in the universe
  - Universe is a set of particles (gas, dark matter, stars)
  - Particles interact through gravity and hydrodynamics
  - Output snapshot every few simulation timesteps

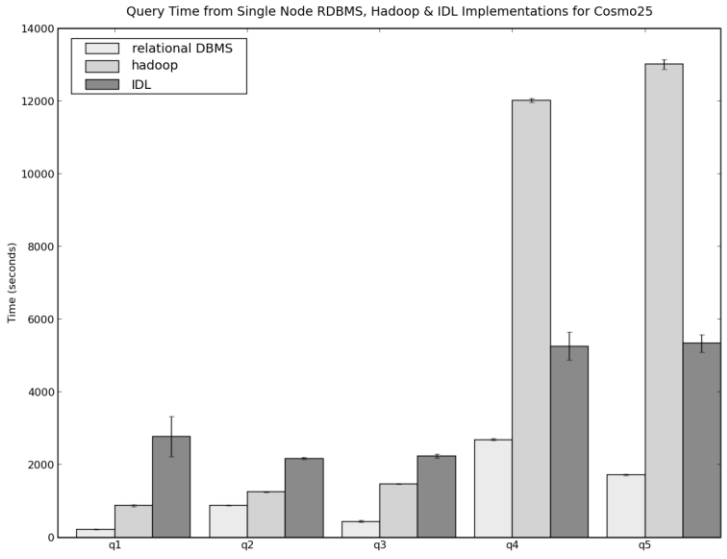
Few dozen to few hundred snapshots per run

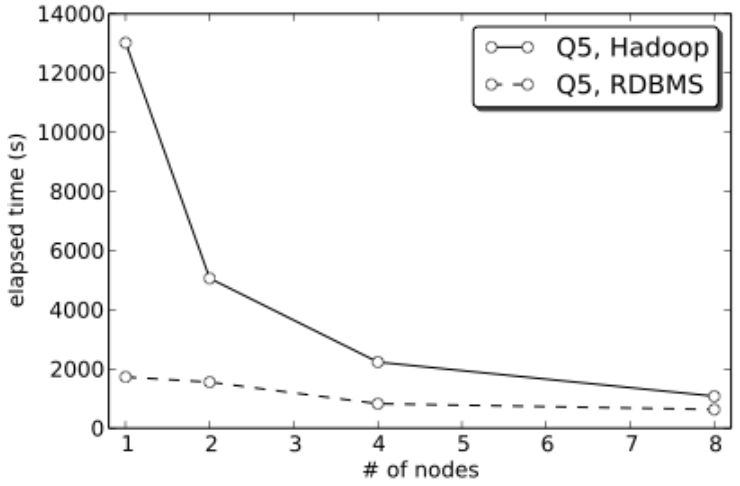
Simulation	No. Particles	Snapshot Size
dbtest128g	4.2 million	169MB
cosmo50	33.6 million	1.4 GB
cosmo25	916.8 million	36 GB

- Analysis needs:
  - Select-project-join (SPJ) queries over snapshot data
  - Data clustering within snapshot
  - SPJ and recursive queries over clustered data

Astronomy Simulation Use Case

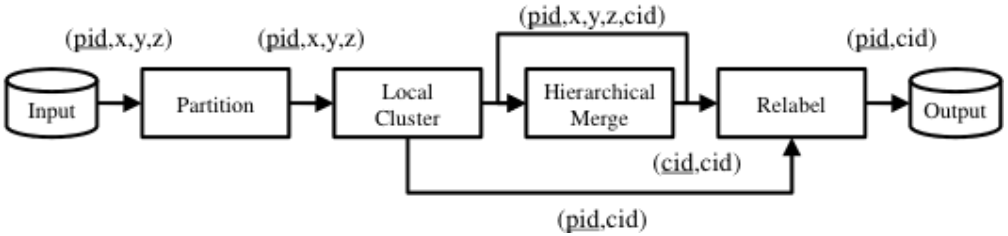
- Implemented SPJ queries over raw data
  - Relational DBMS (single site and distributed)
  - Pig/Hadoop
  - IDL: State-of-the-art in astronomy



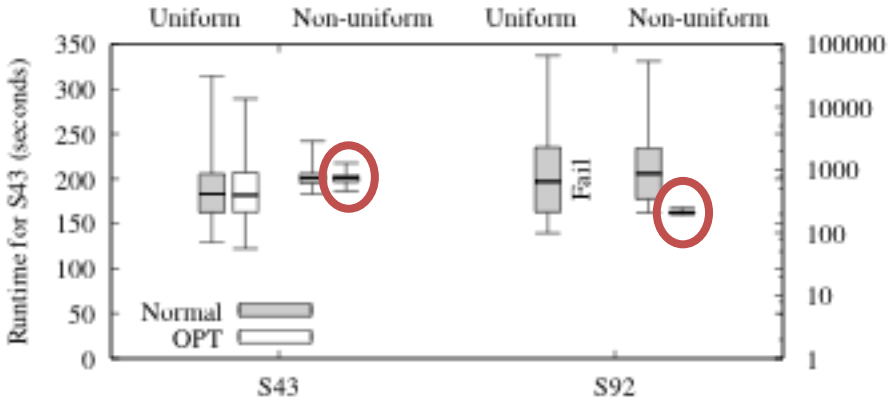


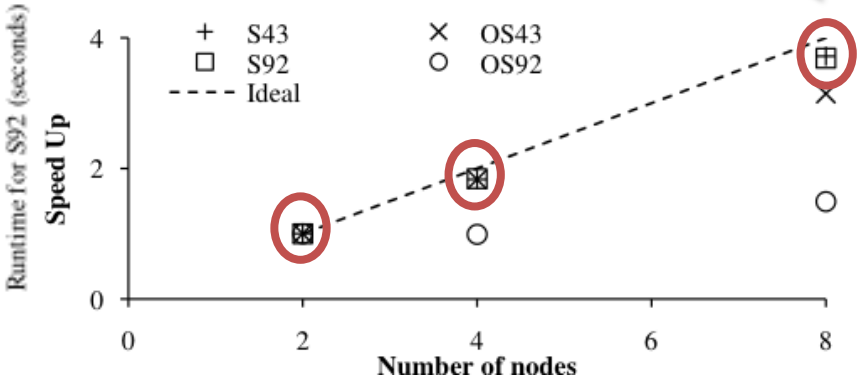
Friends-of-Friends Clustering

- Efficient clustering algorithm
- Implemented in Pig/Hadoop and Dryad/DryadLINQ



Best **total** runtime was 70 min





Problem Statement

- Given magnitude of data and queries
- Need more than efficient query processing

- Users need tools for managing queries at runtime:
  - Accurate, time-based progress indicators
  - The ability to see representative partial results
  - The ability to suspend and resume queries
  - Intra-query fault-tolerance
  - Agile query scheduling and resource management

- All this without too much runtime overhead

Parallax: Progress Indicator for Parallel Queries

- Accurate time-remaining estimates for parallel queries

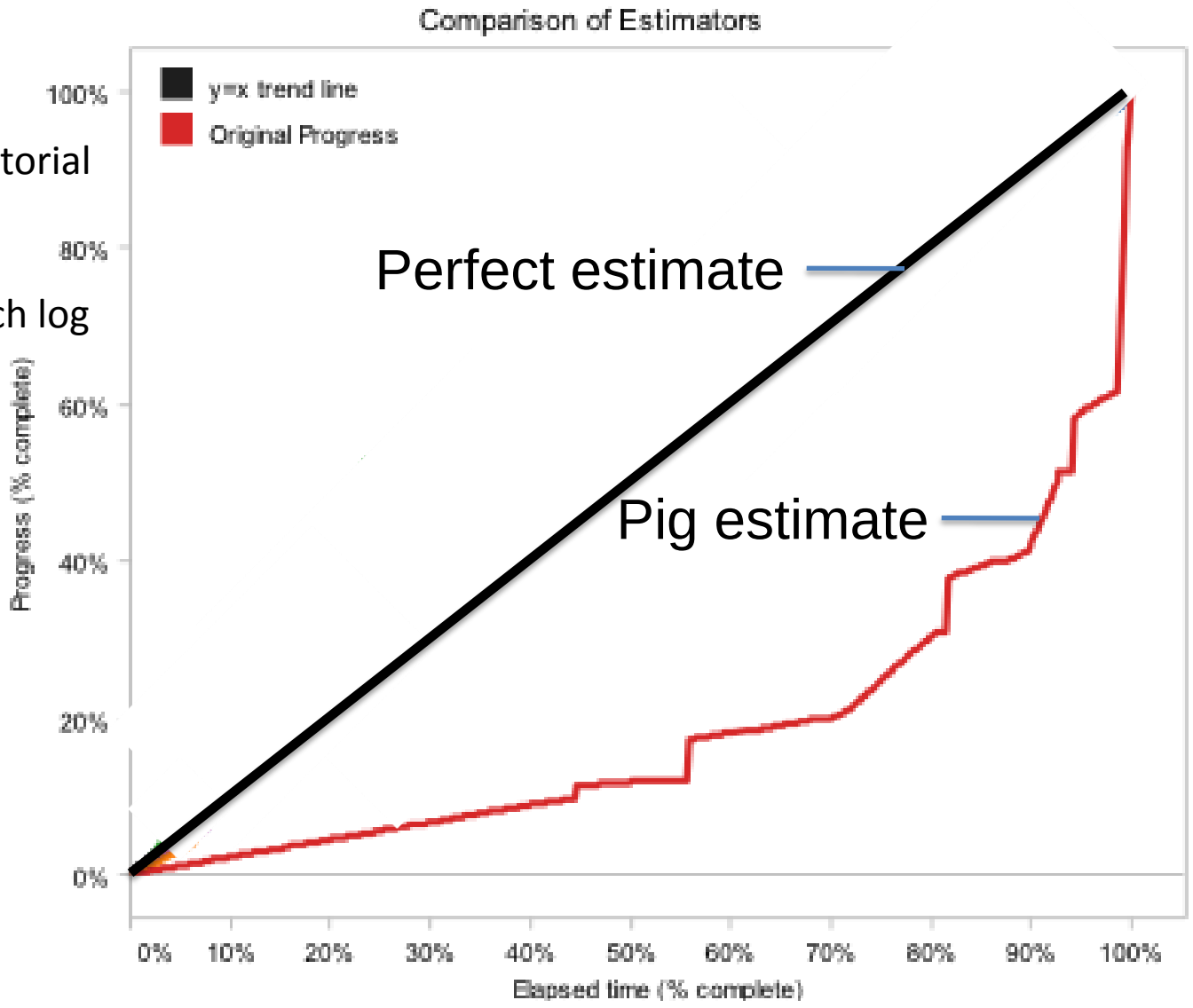
- Why is accurate progress important?
  - Users need to plan their time
  - Users need to know when to stop queries

- Implementation: MapReduce DAGs in Pig
  - Pig scripts that translate into MapReduce DAGs

Accuracy is a Challenge

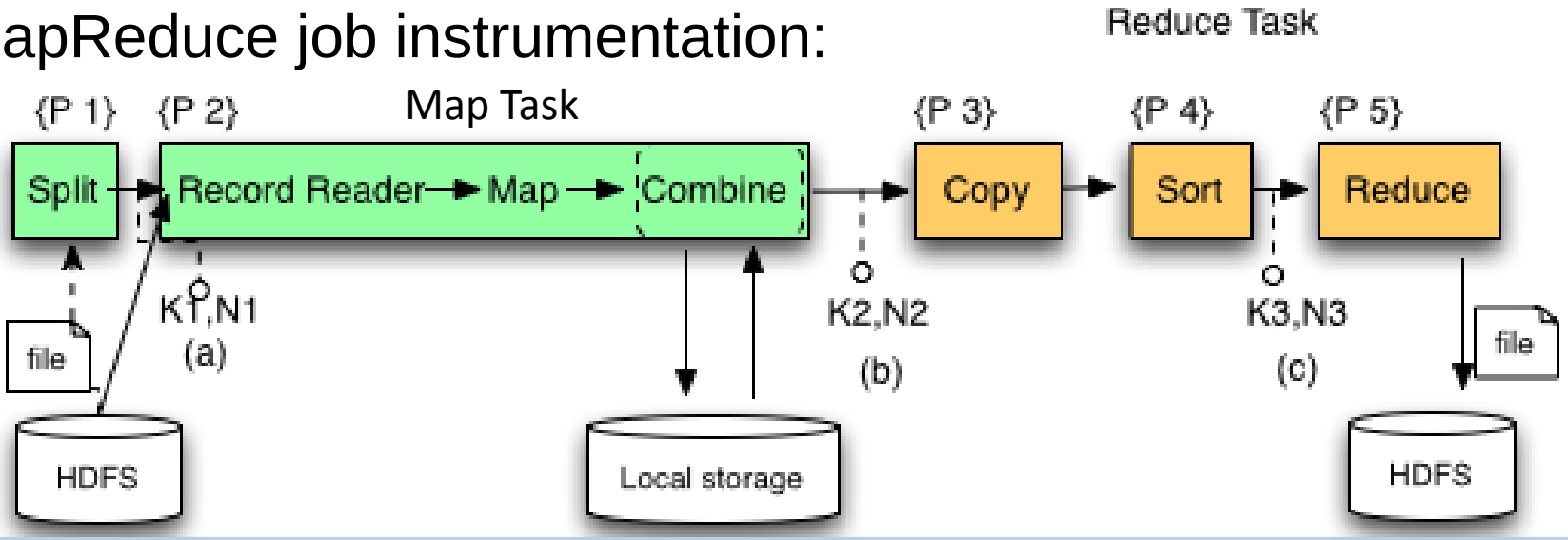
Query: Script1 from Pig tutorial
Translates into 5 MR jobs

Input data: 5X excite search log
210 MB of data



Parallax Approach

MapReduce job instrumentation:



Expected processing speed
Slowdown factor

Tuples remaining

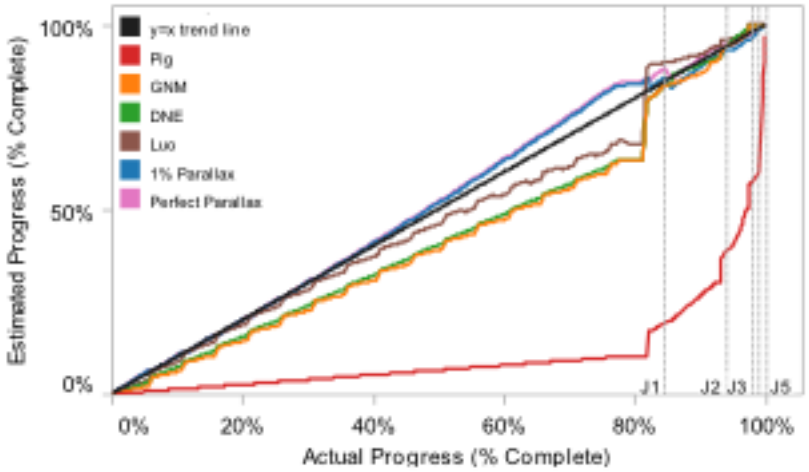
$$T_{\text{remaining}} = \text{SetupOverhead}_{\text{remaining}} + \sum_{j \in J} \sum_{p \in P} \frac{s'_{jp} \alpha_{jp} (N_{jp} - K_{jp})}{\text{pipeline\_width}_{jp}}$$

For all pipelines in all jobs

Parallelism accounting for skew and variations

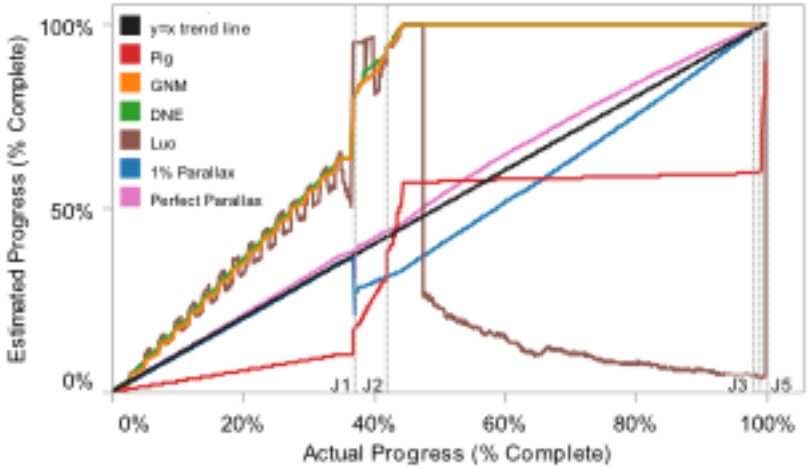
Experimental Results

Script 1 - Serial

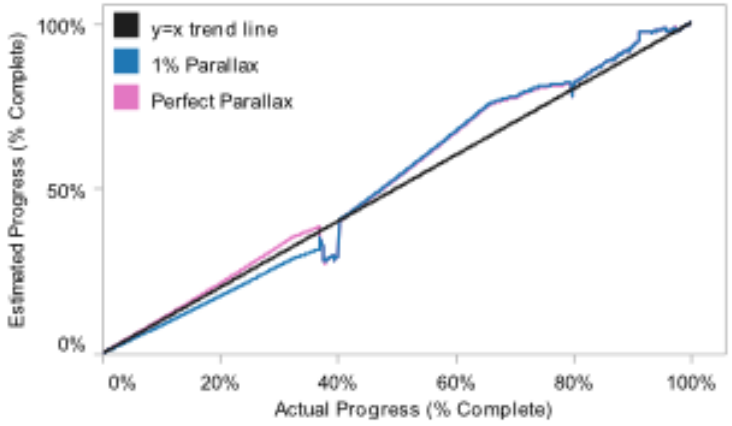


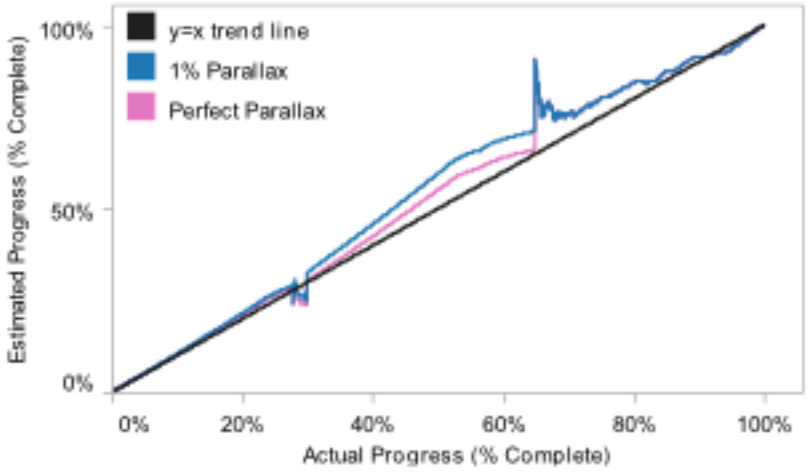
8 nodes, 32 maps, 17 reduces, uniform

Script 1 + UDF - Serial



8 nodes, 32 maps, 32 reduces, zipf





Problem Statement

- Given magnitude of data and queries
- Need more than efficient query processing

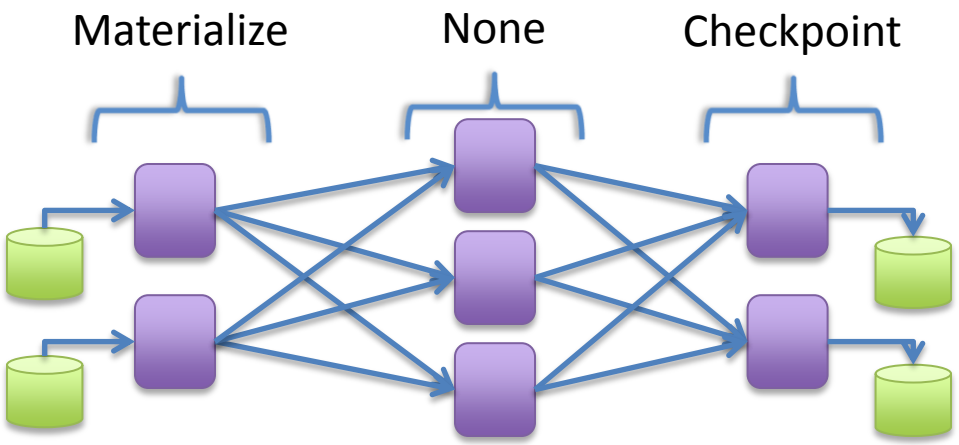
- Users need tools for managing queries at runtime:
  - Accurate, time-based progress indicators
  - The ability to see representative partial results
  - The ability to suspend and resume queries
  - Intra-query fault-tolerance
  - Agile query scheduling and resource management

- All this without too much runtime overhead

Intra-Query Fault Tolerance

- Existing intra-query fault-tolerance methods are limited
  - Parallel DBMSs restart queries when failures occur
  - MapReduce-style systems materialize all intermediate results
  - Result: either high-runtime overhead or costly failure recovery!

- FTOpt: We have developed a fault-tolerance optimizer
  - Automatically picks the best fault-tolerance strategy **per operator**



Problem Statement

- Given magnitude of data and queries
- Need more than efficient query processing

- Users need tools for managing queries at runtime:
  - Accurate, time-based progress indicators
  - The ability to see representative partial results
  - The ability to suspend and resume queries
  - Intra-query fault-tolerance
  - Agile query scheduling and resource management

- All this without too much runtime overhead

Nuage Project

- Nuage project goals (http://nuage.cs.washington.edu/)
  - Massive-scale parallel query processing
  - With focus on scientific applications
  - Cloud computing: DBMS as a service for science

DBMS As a Service for Science

- SciFlex: A Cross-scale Cross-domain Scientific Data Management Service
  - Schema recommendation & data upload utilities
  - Query, archive, and visualization services
  - Data intensive computing!
  - Data, schema, and tool sharing + tool recommendation
  - Annotations, tagging, disagreement, discussions
  - Security: need to share safely
  - SLAs for science
- Interesting systems issues involved in building SciFlex
- In collaboration with Microsoft Research

Conclusion

- Sciences are increasingly data rich
- Need efficient, large-scale query processing
- Need other data management services too
- Nuage/SciFlex project strives to address these needs

Acknowledgments

- Students: Nodira Khoussainova, YongChul Kwon, Kristi Morton, Emad Soroush, and Prasang Upadhyaya

- Collaborators: Jeff Gardner, Dan Grossman, Bill Howe, Dan Suciu, and the SciDB team

Acknowledgments

- This research is partially supported by
  - NSF CAREER award IIS-0845397
  - NSF CRI grant CNS-0454425
  - An HP Labs Innovation Research Award
  - Gifts from Microsoft Research
  - Balazinska's Microsoft Research Faculty Fellowship