Adapting to Changing Resource Performance in Grid Query Processing

Anastasios Gounaris\textsuperscript{1}, Jim Smith\textsuperscript{2}, Norman W. Paton\textsuperscript{1}, Rizos Sakellariou\textsuperscript{1}, Alvaro A.A. Fernandes\textsuperscript{1}, Paul Watson\textsuperscript{2}

\textsuperscript{1}University of Manchester, UK
\textsuperscript{2}University of Newcastle, UK
Outline of Talk

- Query Processing on the Grid with OGSA-DQP
- A generic framework for adaptive query processing (AQP)
- Adaptations to Changing Resources in Grid query processing
  - Design
  - Evaluation
Selecting Resources in OGSA-DQP
Evaluating Queries in OGSA-DQP
Grid Distributed Query Services (GDQSs) that:
- parse, compile, partition and schedule the query execution over a union of distributed data sources.
- Coordinates the GQESs into executing the plan

Grid Query Evaluation Services (GQESs) that:
- implement the physical query algebra;
- run a partition of a query execution plan generated by a GDQS;
- interact with other GQESs/GDSs/WSs
Limitations of existing AQP systems:

- They are too specific in terms of the problem they address and are designed in isolation.
  - No interoperability.
  - Many different, non-compatible solutions for the same problem.


The framework is a step toward:

- Development of accepted abstractions.
- Generic design methodology for AQP systems.
- Reusable adaptivity components.

Current techniques tend to group together an approach to monitoring, a means of assessment, and a form of response.
Outline of Talk

- Query Processing on the Grid with OGSA-DQP
- A generic framework for adaptive query processing (AQP)
- Adaptations to Changing Resources in Grid query processing
  - Design
  - Evaluation
Adapting to workload imbalance: Architectural Components
Instantiating the framework: a dynamic balancing example

Evaluator
Detector

Evaluator
Detector

Diagnoser
Responder

Evaluator
Detector

Project
(Blast)

Exchange

Table_scan
(protein)

Site A, B

Site A

Site B

Site C

Subscribe
Instantiating the framework: a dynamic balancing example

Evaluators pass on monitoring info to detectors

Evaluator → Detector
Evaluator → Detector

Evaluator passes monitoring info to detectors

Diagnoser → Responder

Coordinator

Site A
Evaluator
Detector
Evaluator
Detector

Site B
Evaluator
Detector
Evaluator
Detector

Site A, B
Op_call
(Blast)

Site C
Table_scan
(protein)
Instantiating the framework: a dynamic balancing example

Evaluator
Detector

Evaluator
Detector

Diagnoser
Responder

Detectors notify the diagnoser of the actual behaviour

Site C

Evaluator
Detector

Project

Blast

Exchange

Table scan
(protein)

Site A, B

Site A

Site B

Coordinator
Instantiating the framework: a dynamic balancing example

**Site A**
- Evaluator
- Detector

**Site B**
- Evaluator
- Detector

**Coordinator**
- Diagnoser
- Responder

**Site C**
- Project
- Op_call (Blast)
- Exchange
- Table_scan (protein)

Diagnoser notifies responder
Instantiating the framework: a dynamic balancing example

Evaluator
Detector

Evaluator
Detector

Diagnoser
Responder

Responder checks progress

Site A

Site B

Site C

Evaluator
Detector

Evaluator
Detector

5

project

op_call
(Blast)

exchange

table_scan
(protein)
Instantiating the framework: a dynamic balancing example

Evaluator
Detector

Diagnoser
Responder

6 Responder enforces redistribution

Site C

Project
Site A, B
Op_call (Blast)
Exchange
Table_scan (protein)

Site C

Coordinator
Categories of Monitoring

- **M1**: Notifications on the cost of processing a tuple in a partition (tuple processing cost, selectivity, time waiting for input).

- **M2**: Notifications about communication costs (cost of sending a buffer, recipient of buffer, size of buffer).
Monitoring: Parameters

- **M1** notifications:
  - Events produced by evaluator every 10 tuples;
  - Averages are calculated for last 25 events;
  - Threshold for sending notification is a 20% change in cost per tuple.

- **M2** notifications are for every buffer.
Assessment

- The assessment component seeks to identify where there is a workload imbalance.
  - Existing Balance Vector $W = (w_1, \ldots, w_n)$
  - Proposed Balance Vector $W' = (w'_1, \ldots, w'_n)$
- Notify response component if:
  - $\left(\frac{|w_i - w'_i|}{w_i}\right) > \text{threshold (20%)}$ for some $i$.
- A1 assessment considers only $M_1$ notifications.
- A2 assessment considers both $M_1$ and $M_2$ notifications.
Response

- **If** evaluation not close to completion and **If** progress since last adaptation > x%  
  **then** apply new distribution vector $W'$
- **R1** response is *retrospective*: resends tuples to consumer based on $W'$.
- **R2** response is *prospective*: no tuples are resent, but future tuples are sent to consumers based on $W'$.
- Note that **R1** can be applied in more cases than **R2**.
Normalised Results

Q1: call a remote WS, Q2: join two remote databases

R1: retrospective adaptations, R2: prospective adaptations

A Gounaris
DMG Workshop 20
Changing the number of perturbed machines

![Graph showing the relationship between the number of perturbed machines and normalized response time. The graph compares two cases: one with no adaptivity and one with adaptivity. As the number of perturbed machines increases, the normalized response time also increases.](image-url)
Varying Imbalance Dynamically

- Q1: M1 monitoring, A1 assessment.
- Lessons:
  - Retrospective generally better at this level of perturbation.
  - Neither prospective nor retrospective thrown by varying imbalance.
Lessons Learnt

- AQP on the Grid can be based upon asynchronous communication of loosely coupled components, which support a pub/sub interface.
- Experiments with imbalances show that the approach can yield significant performance improvements in practice.
- It is relatively easy to tune and calibrate the monitoring-assessment-response policies.
Future Work

- More adaptations
  - Adapting to changes in the resource pool
  - Adapting to changing network bandwidth
- Better adaptations
- Experimentation in a more Grid-like environment (e.g., PlanetLab)
where to find out more

Polar*
Grid-enabled Adaptive Query Processing
www.ncl.ac.uk/polarstar

Complete list of publications
www.cs.man.ac.uk/~gounaris

OGSA-DQP
Grid middleware to query distributed data sources
www.ogsadai.org.uk/dqp