A Distributed Architecture for Data Mining and Integration

Malcolm Atkinson
Jano van Hemert
Liangxiu Han
Ally Hume
Chee Sun Liew

www.admire-project.eu
ADMIRE – Framework 7 ICT 215024
Introduction

- Motivation
- Mission & Principal Innovations

Proposed Architecture

- High-level overview of the architecture
- Components of the architecture
- DMIL
- Users communities and interaction with the system
- The path to DMI enactment

Feasibility Study

- Use case - EURExpressII
- System walkthrough
- Research Question

ADMIRE Project
A Revolution in Science

http://www.us-vo.org

http://www.geongrid.org

http://www.neuropsygrid.org

http://nctr.pmel.noaa.gov/Dart

http://lhc.web.cern.ch/lhc

http://esdis.eosdis.nasa.gov

http://www.sinapse.ac.uk

...making data-mining easier
"... continuing leadership in science relies increasingly on effective and reliable access to digital scientific data ..."

"... allow the users to identify and access spatial or geographical information from a wide range of sources, ..., in an interoperable way for a variety of uses ...

...making data-mining easier"
Combinatorial Complexity

• Data integration
  – precursor to Data Mining from multiple sources

• Data mining
  – key to learning from today’s wealth of data

• Growing opportunity and challenge
  – growing number of distributed data
  – growing content and complexity per data source
  – growing number of users
Our Mission

• Radically improve enactment of Data Mining and data Integration (DMI) processes across heterogeneous and distributed data resources and data mining services.
Principal Innovations

• De-coupling of the enactment technology from the tools used to prepare data mining and integration (DMI) processes

• Accommodate independent DMI enactment services, some of which may be tightly coupled with curated data
Hypothesis: By enforcing logical decoupling, both the tools development and the platform engineering will proceed rapidly and independently.
High-level Architecture

![Diagram showing a high-level architecture with components labeled as A DMI Tool, DMI Gateway, Enactment Platform, and Hidden diverse autonomous services. Arrows indicate direct bulk data delivery and dialogue about enactment status.]

...making data-mining easier
Components of the Architecture

...making data-mining easier
DMI Language (DMIL)

• notation for all DMI requests to a gateway
• encodes the following:
  – Requests for information about the services, data resources, data collections, defined components and libraries supported by the gateway.
  – Definition, redefinition and withdrawal of any of the above.
  – Submission of requests to enact a specified data mining and integration process.
User communities

- **Domain Experts**
  - I recognise gene expression

- **DMI Experts**
  - I know DMI algorithms

- **DADC Engineers**
  - I can implement and support

...making data-mining easier
User interaction with DMI systems

- Domain Experts
- DMI Experts
- DADC Engineers

Custom data cleansing
Data source 1
Data source 2
Custom data integration
Data source 3
Gateway
Gateway
Gateway
Gateway
Portal
DMI Workbench
DMI Experts

...making data-mining easier
The path to DMI enactment

...making data-mining easier
Use case: EURExpressII

Deployment phase

Manual Annotations

Images

Automatic annotations

Apply classifier

Image integration

Testing phase

Image processing

Feature generation

Feature selection/extraction

Prediction evaluation

Training phase

Image processing

Feature generation

Feature selection/extraction

Classifier construction

...making data-mining easier
Walkthrough: Processing of a DMI Request

- Decide gateway
- Validate request
- Organise computation
- Coordinate and Monitor
- Initiate enactment
- Terminate the enactment

ADMIRE @ DADC’09, Munich, Germany - June 9, 2009

...making data-mining easier
Walkthrough: Request in DMIL

/* import components */
use dmi.rdb.SQLQuery;
use dmi.samplers.ListRandomSample;
use dmi.image.ImageRescale; ...
use dmi.classifiers.nFoldValidation;
use dmi.classifiers.LDAClassifier;
/* set up and identify instances of the PE */
SQLQuery sqlQuery = new SQLQuery;
ListRandomSample listSample = new ListRandomSample;
TupleProjection tupleProj = new TupleProjection;
GetFile getFile = new GetFile;
ImageRescale imageRescale = new ImageRescale;
MedianFilter medianFilter = new MedianFilter;
WaveletDecomp wavelet = new WaveletDecomp;
TupleMerge tupleMerge = new TupleMerge;
ViaStatus deliver = new ViaStatus;
String query = “SELECT leName, . . .
FROM EURExpress.images, . . .
WHERE . . .”;

...making data-mining easier
Walkthrough: Request in DMIL

/* the literal “query" gets connected to sqlQuery's input “expression"*/
pipe (| query -> expression -> sqlQuery |

/* sqlQuery's output “data" gets connected to listSample’s input “dataIn" */
sqlQuery->data => dataIn->listSample;
pipe (| 0.01 -> fraction -> listSample |

Connection c1; listSample->dataOut => c1;
c1 => filename->getFile;
c1 => data->tupleProj;
pipe (| ["date", "assay#", . . . ] -> columnIds -> tupleProj |

getFile->data => dataIn->imageRescale;
imageRescale->dataOut => dataIn->medianFilter;
pipe (| repeat enough < 300, 200 -> size -> medianFilter |

medianFilter->dataOut => dataIn->wavelet;
wavelet->dataOut => dataIn[0]->tupleMerge;
tupleProj->result => dataIn[1]->tupleMerge;
Validation val = nFoldValidation (10, LDAClassifier);
tupleMerge->dataOut => data->val;
val->results => data->deliver;
Walkthrough: Decide Gateway

...making data-mining easier
**ADMIRE**

- Framework 7
- ICT 215024

---

**Walkthrough: Validate & Compile**

/* import non-universal components from the computational environment */
import uk.org.ogsadai.SQLQuery; // get definition of SQLQuery
import uk.org.ogsadai.TupleToWebRowSetCharArrays; // serialisation
import uk.org.ogsadai.DeliverToRequestStatus;

/* construct and identify instances of the PE */
SQLQuery query = new SQLQuery();
TupleToWebRowSetCharArrays wrs = new TupleToWebRowSetCharArrays();
DeliverToRequestStatus del = new DeliverToRequestStatus();

/* form connection c1 with 
expression as its source 
and query as its destination */
String q1 = "SELECT * FROM weather";
| expression => query;
String resourceID = "MySQLResource";
| resourceID => query;
query => data => data => wrs;
wrs => result => input => del;

---

**Produces**

**DMIL**

**JAVA**

**OGSA-DAI**

---

**Compile**

**...making data-mining easier**
Walkthrough: DMIL Processor

- DMIL request
  - Parses DMIL
  - Produces DMIL
- DMIL Compiler
  - DMIL Graph
  - Java
    - Java Compiler
      - Produces Java source code
  - Compilation and execution of Java to produce a DMIL graph
- Registry
  - Produces OGSA-DAI client toolkit request from a DMIL graph with one to one mapping
- Data
  - OGSA-DAI
  - OGSA-DAI Workflow
  - OGSA-DAI Client Toolkit
    - Produces DMIL Graph to OGSA-DAI

...making data-mining easier
function nFoldCrossValidationPattern(\$)PE{
    Integer k;
    PE (Connection dataIn: [...]) → (Connection output: Classifier) buildClassifier,
    PE (Connection classifier: Classifier; dataIn: [...]) → (Connection score: Real) evaluator
    PE(Connection inputData: [...]) →
    (Connection results: [(score: Real; classifier: Classifier)])
}
ListRandomSplitNWays lrs = new ListRandomSplitNWays;
Connection inputData;  // shouldn't be necessary
inputData => dataIn->lrs;
UnlimitedBuffer buffer = new UnlimitedBuffer[n];
TupleProjection[\$] projectInputVariables = new TupleProjection[k];
TupleProjection[\$] projectOutputVariables = new TupleProjection[k];
Classify[\$] classify = new Classify[k];
ListMerge[\$] listMerge = new ListMerge[k];
TupleMaker tupleMaker = new TupleMaker;
Connection[\$] splits = new Connection[k];
for (i = 1; i <= k; i++) {
    for (j = 1; j <= k; j++) {
        if (i == j) {
            /* connect test set */
            lrs->dataOut[\$] => dataIn->buffer[i];
        } else {
            /* connect training set */
            lrs->dataOut[\$] => dataIn[j]->listMerge[i];
        }
    }
}
/* training phase */
listMerge[i]->dataOut => dataIn->buildClassifier[i];
inputVariables[i] => columnIds->projectInputVariables[i];
buffer[i]->dataOut => dataIn->projectInputVariables[i];
buildClassifier[i]->classifier => classifier->classify[i];
projectInputVariables[i]->result => dataIn->classify[i];
/* testing phase */
classify[i]->class => proposedClass->evaluator[i];
buffer[i]->dataOut => dataIn->projectOutputVariables[i];
outputVariables[i] => columnIds->projectOutputVariables[i];
projectOutputVariables[i]->result => desiredClass->evaluator[i];
buildClassifier[i]->classifier => element[1]->tupleMaker;
evaluator[i]->score => element[2]->tupleMaker;
}
tupleMaker->result => [(score: classifier)];
/* form and return a PE comprising this DMI process subgraph */
return new PE(
    Integer k; PE (dataIn: [...]) → output: Classifier) buildClassifier,
    PE (classifier: Classifier; dataIn: [...]) → score: Real) evaluator
    PE(inputData: [...]) → [(score: Real; classifier: Classifier)]
)
Walkthrough: Enactment
**Walkthrough: Coordination**

Original workflow:

1. Image Scaling
2. Noise Reduction

Modified workflow with Observer and Gatherer:

1. Image Scaling
2. Observer
3. Noise Reduction
4. Observer
5. Gatherer
6. Database
Walkthrough: Monitoring

Timeline of processes execution in EURExpressII workflow (single machine with 800 images)

- preTupleSplit
- preReadFromFile
- preMedianFilter
- preFeatureGeneration
- preFisherRatio
- postFisherRatio
- potFeatureExtraction

Execution Time (second)

Number Images

Comparison of EURExpressII workflow with Matlab scripts vs pure OGSA-DAI implementation

Run-time Comparison of EURExpressII workflow (image preprocessing and feature generation stage)

- with Matlab scripts
- pure OGSA-DAI implementation

Execution Time (second)

Number Images

...making data-mining easier
Walkthrough: Terminate

• Final stage, still under construction, includes:
  – Resource-lifetime management
  – Provenance and audit
Outstanding Issues

• DMIL
  – One language -> full gamut of DMI processes?
• Process optimisation
  – What can we optimise in DMI processes?
• Pipeline streaming model
  – Automatically insertion of unlimited buffer?
• Large amount of data movement
  – How to store, partition, cache & move data?
Conclusion

• Introduce a separation of concerns between data mining and integration (DMI) process development and the mapping, optimisation and enactment of these processes.
• Postulate this separation of concerns will allow handling separately the user and application diversity and the system diversity and complexity issues simultaneously.
• Introduce an architecture, which as a principal element defines gateways as the point where these two concerns meet.
• Validate our hypothesis of separation of concerns with a feasibility study that comprises building prototypes of the architecture.
ADMIRE Goals

- Accelerate access to and increase the benefits from data exploitation;
- Deliver consistent and easy to use technology for extracting information and knowledge;
- Cope with complexity, distribution, change and heterogeneity of services, data, and processes, through abstract view of data mining and integration; and
- Provide power to users and developers of data mining and integration processes.

...making data-mining easier
ADMIRE Structure

– WP1: High-Level Model and Language Research
  • Incremental development of models and languages

– WP2: Architecture Research
  • Incremental development of a flexible, scalable, open DMI arch.

– WP3: Platform Support & Delivery
  • Deliver robust service platforms, support users

– WP4: Service Infrastructure Development and Enhancement
  • Develop technology and services to enhance the DMI service infra.

– WP5: DMI Tools Development
  • Develop and integrate tools that make the technology easier to use

– WP6: Integrated Applications
  • Demonstration of validation and performance

– WP7: Project Management

...making data-mining easier
ADMIRE Partners

- Partners:
  - University of Edinburgh, UK (Coordinator)
  - Fujitsu Laboratories of Europe, UK
  - University of Vienna, Austria
  - Universidad Politécnica de Madrid, Spain
  - Institute of Informatics, Slovak Academy of Sciences, Slovakia
  - ComArch S.A., Poland

- Finance:
  - €4.3 Million in costs, €3 Million in EC funding (EU FP7 ICT 215024)
Further Information

http://www.admire-project.eu/