Web Data Integration: Exercise 3, Data Fusion
Exercise 3, Data Fusion

• Agenda
  – Exercise overview
  – Prepare the inputs
    – Add provenance to your data
  – Merge your data
    – Intermediate representation
    – Density and consistency
  – Prepare your gold standard
  – Fuse your data
    – Conflict resolution functions
    – Accuracy with respect to gold standard
  – Final report and final exam

• Timing: November 13 – 28
Exercise Overview

• In this exercise you will experiment with
  – defining provenance metadata
  – conflict resolution functions
  – evaluation metrics

• Your input is the output of Exercises 1 and 2

• Your task is to extend a template Eclipse Java project

• …but first, look at your data.
Do you know you data?

• Your input is the output of Exercises 1 and 2
  – Vocabularies are aligned, unique IDs are in place, identity resolution is done

• Are there data to fuse?
  – Instance intersection is big enough: select **at least 3 datasets** for which
    – Property (attribute) intersection is big enough
      – You should be able to fuse data for **>= 5 attributes** per class
    – Quality of identity resolution is good enough

• Note: template project **does not** work with list (multiple-valued) attributes
  – …unless you extend it. To do it, implement conflict resolution, consistency and accuracy calculation for lists.
  – Examples: list of actors for a movie, post codes of a city, etc.
Do you know you data?

• What conflict resolution functions to use for your use case?
  – Conflict resolution functions are attribute-specific
    – *Maximum* for city population, *most frequent* for country capital
  – Conflict resolution functions may use provenance metadata
    – Creation date, data source rank/trust/size/author, etc.

• The first step of your work with the template Java project – *merging your data* – does not require knowing which conflict resolution functions to use
  – Postpone thinking about conflict resolution functions till after you create the gold standard
Add provenance to your datasets

- Add **one** provenance element per dataset
- Two fields should be present: **source** and **date**
  - If you change their names, change them also in MostRecent and MostTrusted classes
- Date format is fixed to "d MMM yyyy HH:mm:ss Z"
  - If you change it, change it also in MostRecent class
- You can add more fields
  - And implement new resolution functions based on these new provenance properties

```xml
<?xml version="1.0" encoding="UTF-8"?>
<movies>
  <movie>
    <id>2-9178</id>
    <title>127 Hours</title>
    ...
  </movie>
  ...
  <provenance>
    <id>prov02</id>
    <source>movie_list.xml</source>
    <date>08 Mar 2009 00:00:00 GMT</date>
  </provenance>
</movies>
Start with the Template Project

• Download the .zip of the project from the course page

• Unzip it and look at the sample input files in \resources\movies\n  • .xml input datasets in datasets folder
  • .csv output of exercise 2 in duplicates folder
  • gold.xml (based on merged.xml produced by the project itself)

• Open the project in Eclipse
Start with the Template Project

- Download the .zip of the project from the course page
- Unzip it and look at the sample input files in `\resources\movies\`
  - .xml input datasets in `datasets` folder
  - .csv output of exercise 2 in `duplicates` folder
  - gold.xml (based on merged.xml produced by the project itself)
- Open the project in Eclipse

- We have implemented for you
  - …lots of things: loading and processing the inputs, computing evaluation metrics: density, consistency, accuracy; merging data, basic conflict resolution functions; fusing data; comparing results to gold standard
  - The only part you need to extend are conflict resolution functions

Christian Bizer, Volha Bryl
Template Project Structure

- **Input**
  - Datasets (xml), duplicates (csv) and gold (xml)
- Main class : start from here
- Conflict resolution functions
Template Project Structure

- Input
  - Datasets (xml), duplicates (csv) and gold (xml)
- **Main class : start from here**
- Conflict resolution functions
Define your inputs

In DataFusion.main() you should define:

- **(A)** Paths to your .xml datasets
- **(B)** Paths to the .csv results of the Exercise 2
- Xpaths to the **(C)** data and **(D)** provenance elements ID
- **(E)** Xpath to the root element of the document
- **(F)** Path to a merged dataset representation → output
- **(G)** Path to the fusion report → output

```java
String rootElementName = "movies"; (E)
String idDataPath = "/movies/movie/id"; (C)
String idProvPath = "/movies/provenance/id"; (D)
Set<String> filesDuplicates = new HashSet<String>(); (B)
filesDuplicates.add("resources/movies/duplicates/matched-1-2.txt");

Set<String> filesData = new HashSet<String>(); (A)
filesData.add("resources/movies/datasets/movie_list1.xml");

String unionFn = "resources/movies/merged.xml"; (F)
String fnFusionReport = "resources/movies/fusion-report.txt"; (G)
runDataUnion(rootElementName, idDataPath, idProvPath,
filesDuplicates, filesData, unionFn, fnFusionReport);
```
Generate the Merged Dataset

• Call to DataFusion.runDataUnion(...) generates the merged representation of the input datasets
  • provenance elements from all datasets are copied
  • all values of all the properties are added as “value” nodes with "provenance" attribute with the provenance ID

• The merged representation is the starting point for data fusion
Get Density and Consistency Metrics

- DataFusion.runDataUnion(...) also calculates and writes to the fusion report a number of evaluation metrics
  - Density (% of non-null values) of each dataset, overall and per property
  - Density and consistency (% of non-conflicting values) of the merged dataset, overall and per property
  - Density and consistency of ID elements is not interesting and always the same
    - By definition ID density is ??, consistency is ??

```
fusion report

dataset prov01:
dataset density : 0.762
average number of non-null attributes per object : 5.333
density, attribute genre : 0.3333333333333333
density, attribute studio : 0.667
...

Merged dataset:
dataset density : 0.952
average number of non-null attributes per object : 6.667
dataset consistency : 0.095
density, attribute genre : 0.667
consistency, attribute genre : 0.0
```
Template Project Structure

- **Input**
  - Datasets (xml), duplicates (csv) and gold (xml)
- **Main class**: start from here
- **Conflict resolution functions**
Define Conflict Resolution Functions

• A resolution function defines how values of the specific attribute are combined

• We have implemented several resolution functions for you, see uma.wdi.fusion.resolution package
  • ...and next slides

• You may need (and you do!) more conflict resolution functions
  • E.g. depending on the value type – select minimum date or shortest string; or use additional provenance data, different from date and source

• Your resolution functions should
  • Extend AbstractResolutionFunction class
  • Override applyStrategy method

java, DataFusion.main()

Map<String, AbstractResolutionFunction> rf =
    new HashMap<String, AbstractResolutionFunction>();
rf.put("budget", new YourResolutionFunction());
Resolution Functions: Take Maximum

- Takes the maximum value
- In case there are more than one maximum value, keeps all provenance IDs
- If at least one value is not numeric, just copies all the values to the fused dataset

```java
java, DataFusion.main()
rf.put("budget", new Maximum());
```

**merged dataset**

```xml
<budget>
  <value provenance="prov01">100</value>
  <value provenance="prov02">150</value>
  <value provenance="prov03">100</value>
  <value provenance="prov04">200</value>
  <value provenance="prov05">200</value>
</budget>
```

**fused dataset**

```xml
<budget>
  <value provenance="prov04,prov05">200</value>
</budget>
```
Resolution Functions: Take Average

- Takes the average value
- Add a (made up) provenance ID “average”
- If at least one value is not numeric, just copies all the values to the fused dataset

```java
DataFusion.main()
rf.put("budget", new Average());
```

**merged dataset**

```xml
<budget>
  <value provenance="prov01">100</value>
  <value provenance="prov02">150</value>
  <value provenance="prov03">100</value>
  <value provenance="prov04">200</value>
  <value provenance="prov05">200</value>
</budget>
```

**fused dataset**

```xml
<budget>
  <value provenance="average">150</value>
</budget>
```
Resolution Functions: Take Most Recent

- Takes the value(s) for which the value of “provenance/date” is maximal
- In case there are more than one most recent value, by default
  - If values are the same, keeps all provenance IDs
  - If values are different, copies all the values to the fused dataset
- Do not change data format and the name of the date field!

```java
java, DataFusion.main()
rf.put("budget", new MostRecent());
```

**merged dataset**

```xml
<budget>
  <value provenance="prov01">100</value>
  <value provenance="prov02">150</value>
  <value provenance="prov03">100</value>
  <value provenance="prov04">200</value>
  <value provenance="prov05">200</value>
</budget>
```

**prov01**: `<date>01 Jan 2010 00:00:00 GMT</date>`
**prov02**: `<date>10 Mar 2009 00:00:00 GMT</date>`
**prov03**: `<date>01 Jan 2010 00:00:00 GMT</date>`
**prov04**: `<date>15 Sep 2005 00:00:00 GMT</date>`
**prov05**: `<date>01 Jan 2010 00:00:00 GMT</date>`

**fused dataset**

```xml
<budget>
  <value provenance="prov01,prov03">100</value>
  <value provenance="prov05">200</value>
</budget>
```
Resolution Functions: Take One of Most Recent

- Takes the value(s) for which the value of "provenance/date" is maximal
- In case there are more than one most recent value, takes one of them
  - Not specified which one
- Do not change data format and the name of the date field!

```
java, DataFusion.main()
rf.put("budget", new MostRecent(true));
```

**merged dataset**

```xml
<budget>
  <value provenance="prov01">100</value>
  <value provenance="prov02">150</value>
  <value provenance="prov03">100</value>
  <value provenance="prov04">200</value>
  <value provenance="prov05">200</value>
</budget>
```

- prov01: <date>01 Jan 2010 00:00:00 GMT</date>
- prov02: <date>10 Mar 2009 00:00:00 GMT</date>
- prov03: <date>01 Jan 2010 00:00:00 GMT</date>
- prov04: <date>15 Sep 2005 00:00:00 GMT</date>
- prov05: <date>01 Jan 2010 00:00:00 GMT</date>

**fused dataset**

```xml
<budget>
  <value provenance="prov01,prov03">100</value>
</budget>
```
Resolution Functions: Take Most Trusted

- Takes the value(s) for which the numerical value associated to “provenance/source” is maximal
- In case there are more than one most trusted value, by default
  - If values are the same, keeps all provenance IDs
  - If values are different, copies all the values to the fused dataset
- Do not change the name of the source field!

java, DataFusion.main()

```java
rf.put("budget", new MostTrusted(trust));
```

merged dataset

```xml
  <budget>
    <value provenance="prov02">150</value>
    <value provenance="prov04">200</value>
  </budget>
```

fused dataset

```xml
  <budget>
    <value provenance="prov02">150</value>
    <value provenance="prov04">200</value>
  </budget>
```
To specify the trust preferences, you should define a map from source names to double trust values.

The greater the value, the more you trust the data source.

If `true` is passed as a second argument to the constructor, takes one of the most trusted values.

```java
Map<String,Double> trust =
    new HashMap<String,Double>()
trust.put("prov01", 1.0);
trust.put("prov02", 4.0);
trust.put("prov03", 3.0);
trust.put("prov04", 4.0);
trust.put("prov05", 2.0);
```
Resolution Functions: Take Most Frequent (Vote)

- Takes the value(s) which occur most frequently
- In case there are more than one most frequent values, by default
  - If values are the same, keeps all provenance IDs
  - If values are different, copies all the values to the fused dataset
- If true is passed as an argument to the constructor, takes one of the most frequent values

```java
java, DataFusion.main()
rf.put("budget", new Vote());
```

**merged dataset**

```xml
<budget>
  <value provenance="prov01">100</value>
  <value provenance="prov02">150</value>
  <value provenance="prov03">100</value>
  <value provenance="prov04">200</value>
  <value provenance="prov05">200</value>
</budget>
```

**fused dataset**

```xml
<budget>
  <value provenance="prov01,prov03">100</value>
  <value provenance="prov04,prov05">200</value>
</budget>
```
Prepare Gold Standard

- Best way to prepare it: copy and edit the merged dataset
  - Replace the values
  - Keep the same format and the same IDs!
- Use external trusted data source find out the true values of the properties
- Your gold standard should contain
  - >= 5 entities
  - >= 5 attributes per entity

<movies>
  ...
  <movie>
    <id>
      <value provenance="prov02">2-9179</value>
      <value provenance="prov01">1-9179</value>
      <value provenance="prov03">3-9179</value>
    </id>
    <budget>
      <value provenance="prov01">35</value>
      <value provenance="prov02">37</value>
      <value provenance="gold">38</value>
    </budget>
  </movie>
  ...
</movies>
Fuse and Evaluate the Results

- DataFusion.runDataFusion() method
  - produces the final fused dataset
  - calculates the accuracy evaluation metrics (% of correct values), overall and per property

```java
String fnOutput = "resources/movies/fused.xml";
runDataFusion(rf, idDataPath, fnGold, fnOutput, fnFusionReport);
```

**fusion report**

Fused dataset:
- overall accuracy : 0.524
- accuracy, attribute genre : 0.333
- accuracy, attribute studio : 0.667
- accuracy, attribute budget : 0.0
- accuracy, attribute title : 1.0
...

Christian Bizer, Volha Bryl

11/12/2014
What you can add to the project

• If you have enough time and programming skills, and want a better mark, you can
  • Add resolution functions for list attributes
    • Call DataSet.calculateDatasetConsistency() with the full list of properties
  • Build on density calculation to assess dataset intersection
  • Change consistency calculation to account for approximate matches
  • Extend consistency calculation to detect outliers
  • Improve the fusion report layout
  • Extend the code so that it works with multiple provenance elements per dataset (but that’d be a lot of work)

• …and then, please, share your code with us
Final Report: Requirements

- **12 pages** (sharp!) – without title and table of content pages
  - Long appendixes or overly long summaries reduce your mark
- Due to **1 December 2014, 23:59**
  - Send by email to Chris and Volha
- You must use the **DWS master thesis layout**
- Please **cite sources properly** if you use any
  - Preferred citation style [Author, year]
- Also **submit your code and** (a subset) of your **data**
Your final report should contain:

- Description of your datasets
- Results of Exercise 1
- Results of Exercise 2
- Results of Exercise 3
Schema Matching in the Final Report

- Results of Exercise 1 should be part of your final report
- Your report should include
  - Your input data profile
    - number, types, min/max/avg, ... of attributes
    - size, estimated overlap, based on sample
  - Your consolidated schema and how you created it
  - Which transformations you used and why
    - if there was any information you could not transform
Identity Resolution in the Final Report

• Results of Exercise 2 should be part of your final report
• Your report should include
  • How your created your gold standard
  • What matchers and similarity metrics you added and tried
    • What happens with P/R/F1? And with runtime?
  • What blocking functions you tried
    • What happened with runtime and number of matches?
    • How do P/R/F1 change, and why?
  • What similarity metrics you combined and how
  • Whether you have learned new matching rules, input and results
Data Fusion in the Final Report

- Results of Exercise 3 should be part of your final report
- Your report should include
  - Which datasets you selected for fusion
  - What kind of provenance data you added
  - What the density of your input and the merged datasets were
  - What resolution functions you used for each attribute
  - Whether you define your own conflict resolution functions
    - Strongly recommended!
  - What the density and consistency were, how they changed with changing the set of conflict resolution functions
  - How you created the gold standard
  - What the accuracy was, how it changed with changing the set of conflict resolution functions
Final Report: Important

• Balance your content between 3 exercises
  • Not “10 pages on identity resolution and 2 pages on the rest”

• If you have done something cool – write about it!
  • It is highly unlikely we dig it out of your code ourselves
• Final report is due to **Monday, 1 December 2014, 23:59**
  • Send by **email to Chris and Volha**
Final Exam

• Date and Time of WDI Exam
  • 15 December 2014, 9:00, A5 B243

• Format
  • 5-6 open questions that show that you have understood the (theory) slides
...and now

- Add provenance to your dataset
- Get the template project and
  - Define your inputs
  - Experiment with creating the merged dataset, and density and consistency evaluation metrics
  - Define your conflict resolution functions
  - Define your gold standard
  - Experiment with data fusion and accuracy evaluation metrics
- Write your final report
- Be ready for the final exam