COMP9321

Data Services Engineering

Term 1, 2019

Week 2 Lecture 2
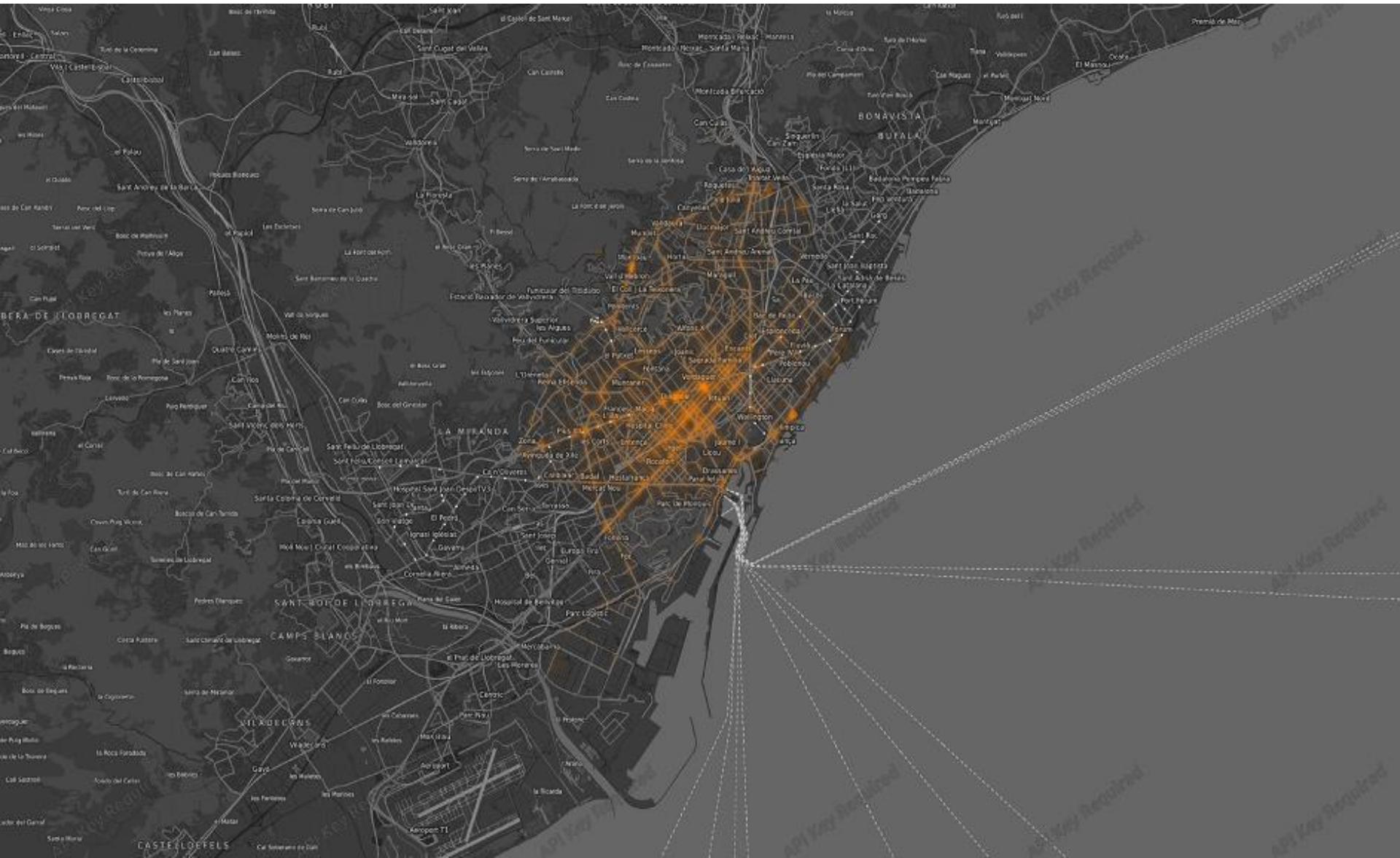
Data Cleansing and Integration

COMP9321 2019T1

UNSW
SYDNEY

Data Cleaning Activities

1. Extraction from sources

☐ Technical and syntactic obstacles

2. Transformation

☐ Schematic obstacles

3. Standardization

☐ Syntactic and semantic obstacles

4. Duplicate detection

☐ Similarity functions

☐ Algorithms

5. Data fusion / consolidation /integration

☐ Semantic obstacles

6. Loading into warehouse / presenting to user

Data Standardization / Normalization

Transformation

- Applying a function to each point z in the data: $y_i = f(z_i)$

Scaling/Normalization

- Converting data to a different scale (like Celsius and Fahrenheit). Typically linear: y = ax + b
- Converting data to a same scale

Formula   (0°C × 9/5) + 32 = 32°F

Standardization (e.g., Z-score normalization)

- Either applying a transformation so that you transformed data is roughly normally distributed
- Somecases, it can also mean putting different variables on a common scale (in this case it is a.k.a scaling)

Reasons to normalize and transformation

- Easy comparison of values

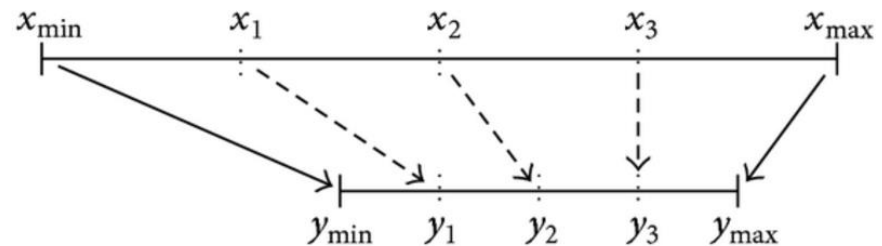- In some algorithms, objective functions will not work properly (or quickly) without it
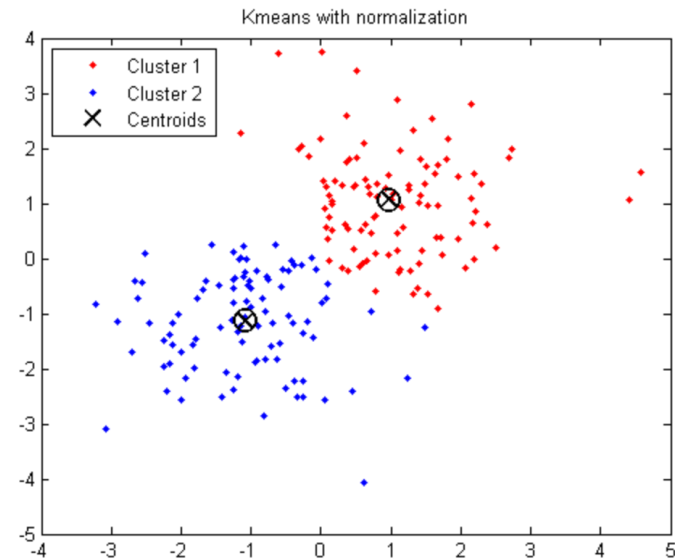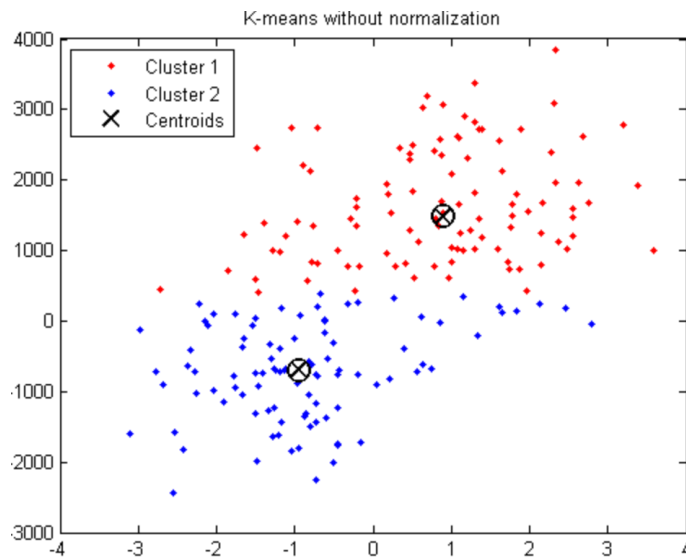
- Can create complex features that may improve the model (or make it non-linear, e.g., kernel functions)

Min-Max normalization
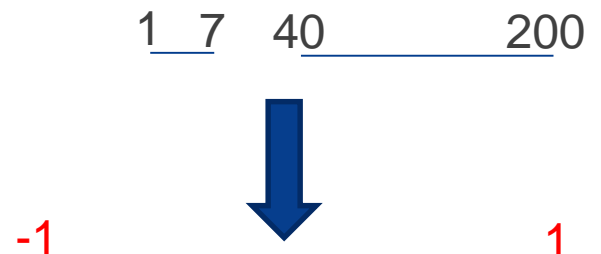
- Min/Max normalization to [0,1]

$$X_{i, 0 \text{ to } 1} = \frac{X_i - X_{Min}}{X_{Max} - X_{Min}}$$

- Min/Max normalization to [-1,1]
  (if we want 0 to be the central point)

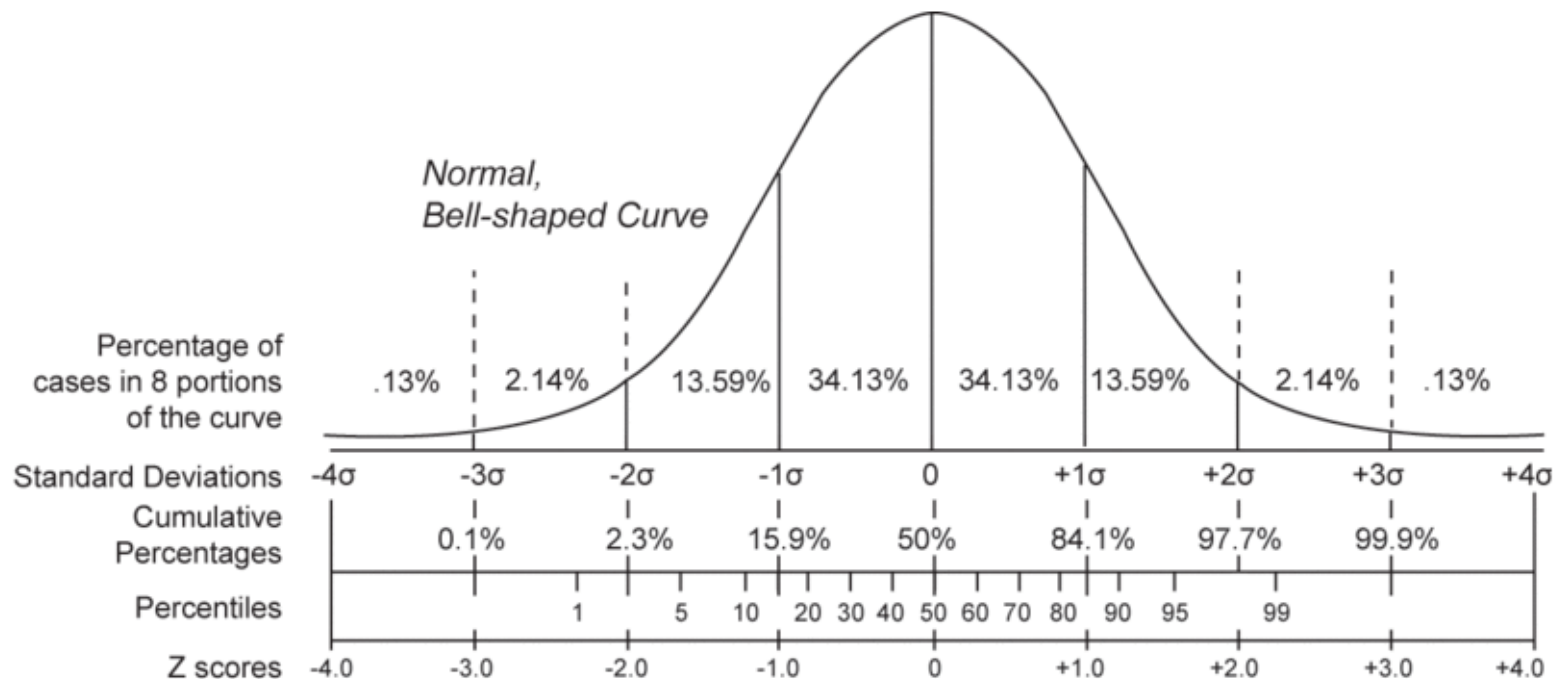$$X_{i, -1 \text{ to } 1} = \frac{2X_i - X_{Min} - X_{Max}}{X_{Max} - X_{Min}}$$

1   7   40   200

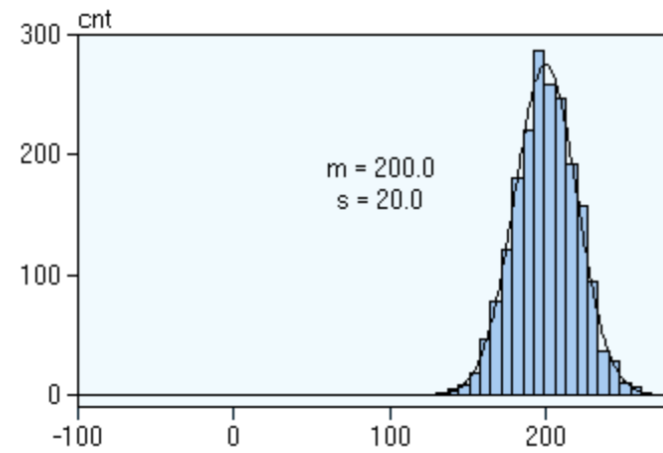-1   1

Intel – Advanced Analytics

Standardization: Z-normalization

- Subtracting the mean and dividing by the standard deviation

$$X_{i, 1\sigma} = \frac{X_i - \bar{X}_S}{\sigma_{X, S}}$$

Normal, Bell-shaped Curve

Percentage of cases in 8 portions of the curve	.13%	2.14%	13.59%	34.13%	34.13%	13.59%	2.14%	.13%

Standard Deviations	-4σ	-3σ	-2σ	-1σ	0	+1σ	+2σ	+3σ	+4σ
Cumulative Percentages		0.1%	2.3%	15.9%	50%	84.1%	97.7%	99.9%	
Percentiles			1	5	10 20 30 40 50 60 70 80	90 95	99		
Z scores	-4.0	-3.0	-2.0	-1.0	0	+1.0	+2.0	+3.0	+4.0

Intel – Advanced Analytics
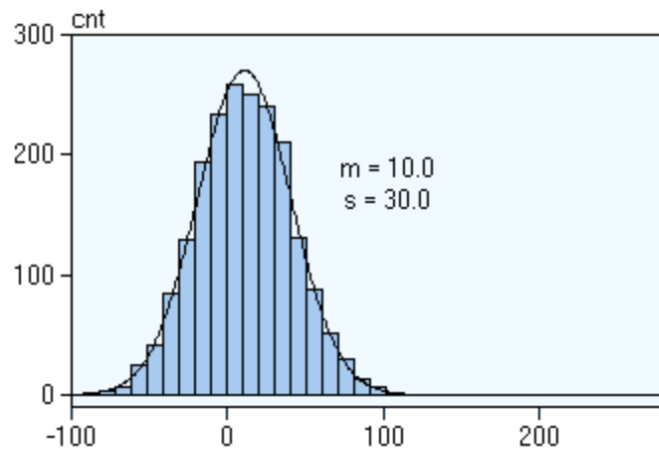
Standardisation

Standardisation
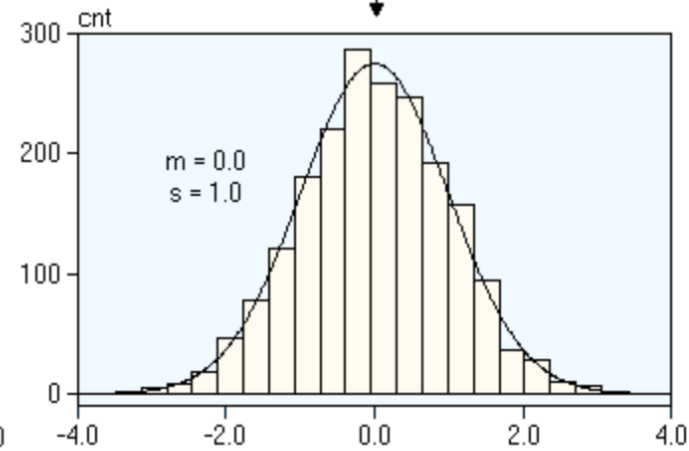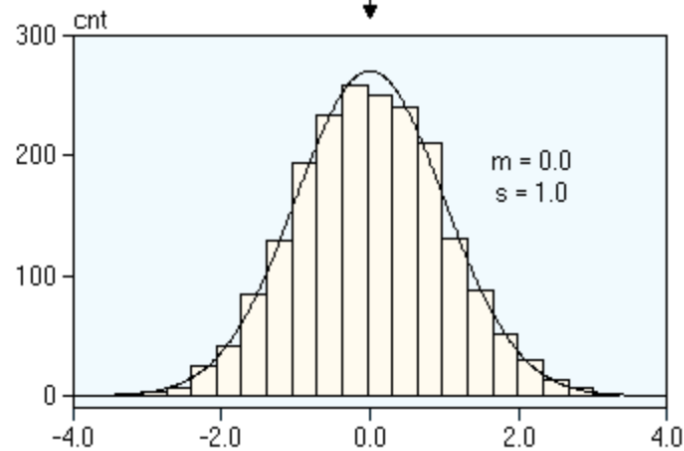
comparable distributions
(m = 0.0, s = 1.0)

Log normalization

Used when values are ranged over several orders

of magnitude.

$$X' = a*\log_b(X)$$

Choosing normalization method

Large range of data: (i.e. $4 to $120,000,000)
• Log transformation is often good

Skewed data (often large range)
• Log transformation is often good

If entropy is high → usually normalizing to [-1,1] is good

If entropy is low → often z-standardization is good

If near normally distributed → z-standardization is good

Intel – Advanced Analytics

Transformations for increasing model complexity

- Can transform features in various ways
  - $\log(X)$, $1/X$, X^2 etc....

- Can lead us to non-linear models

- The more complex your model, the higher the likelihood of overfitting
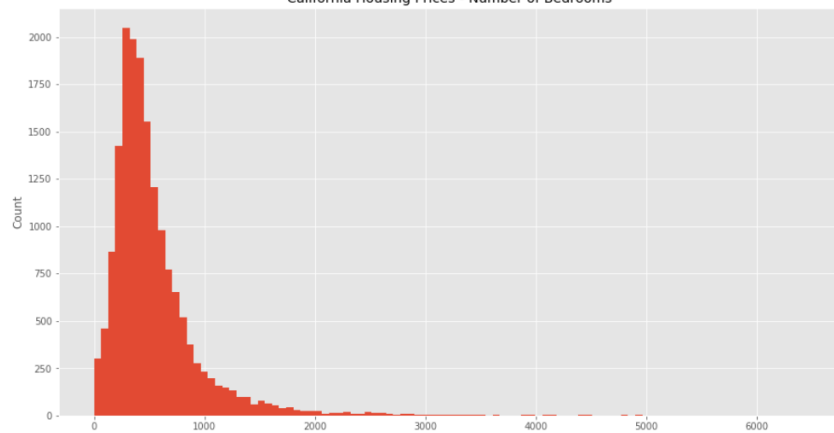
Intel – Advanced Analytics

UNSW
SYDNEY
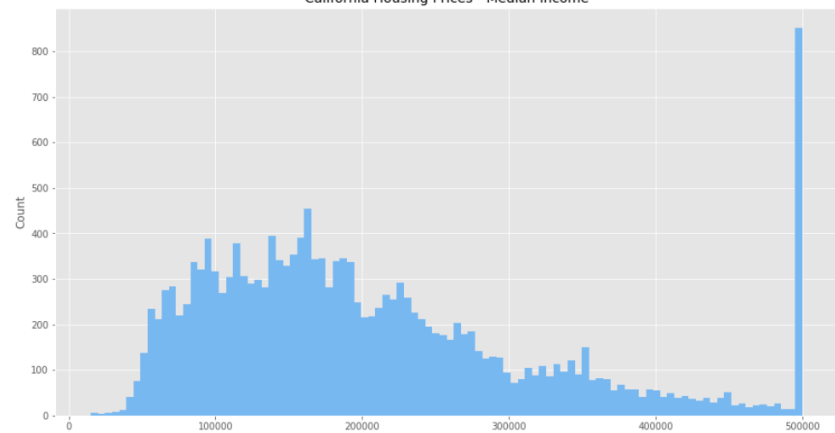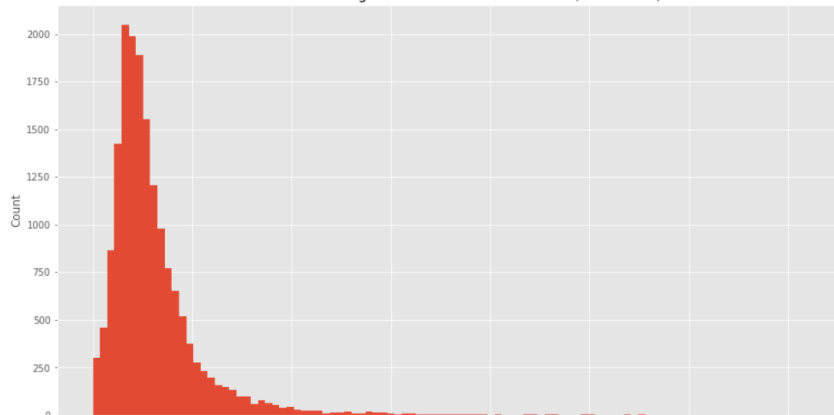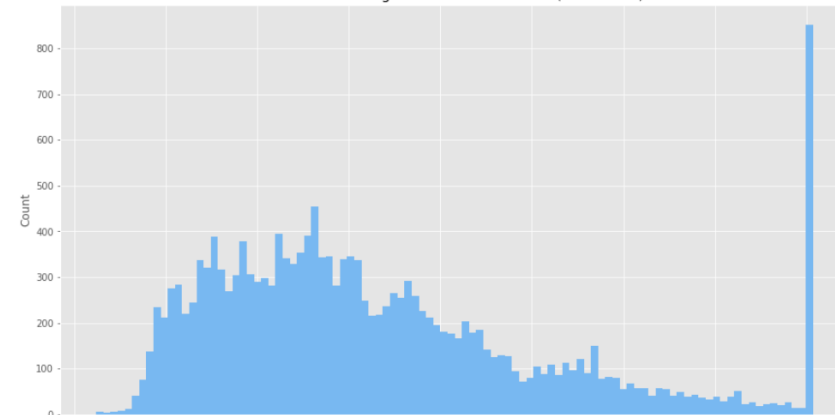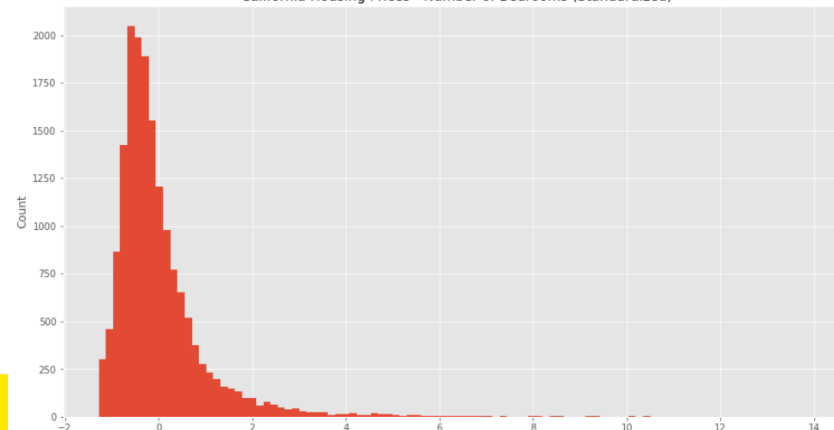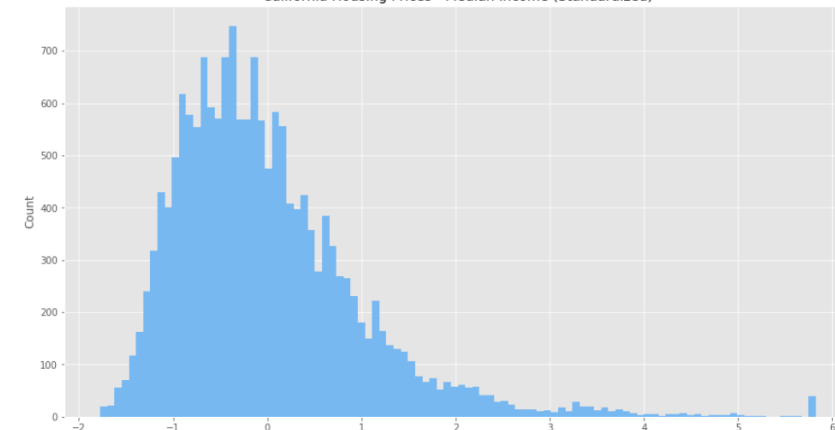
Now, for nature languages…

- Abstract concepts are **difficult to represent**

- **"Countless" combinations** of subtle, abstract relationships among concepts

- **Many ways** to represent similar concepts

For nature languages…

For example…

- **Sant Martí** and **Sant Marti**

- **Meridiana** and **Av Meridiana**

- **Spaceship** and **Spacecraft**

- **International Business Machines** and **IBM**

We need a more consistent representation

The complex story…

NLP   Inf. Retrieval   SW / Web2.0   ML/Text-Mining

Sharing of ideas, intuitions, methods and data

Represent.   Tasks

Techniques
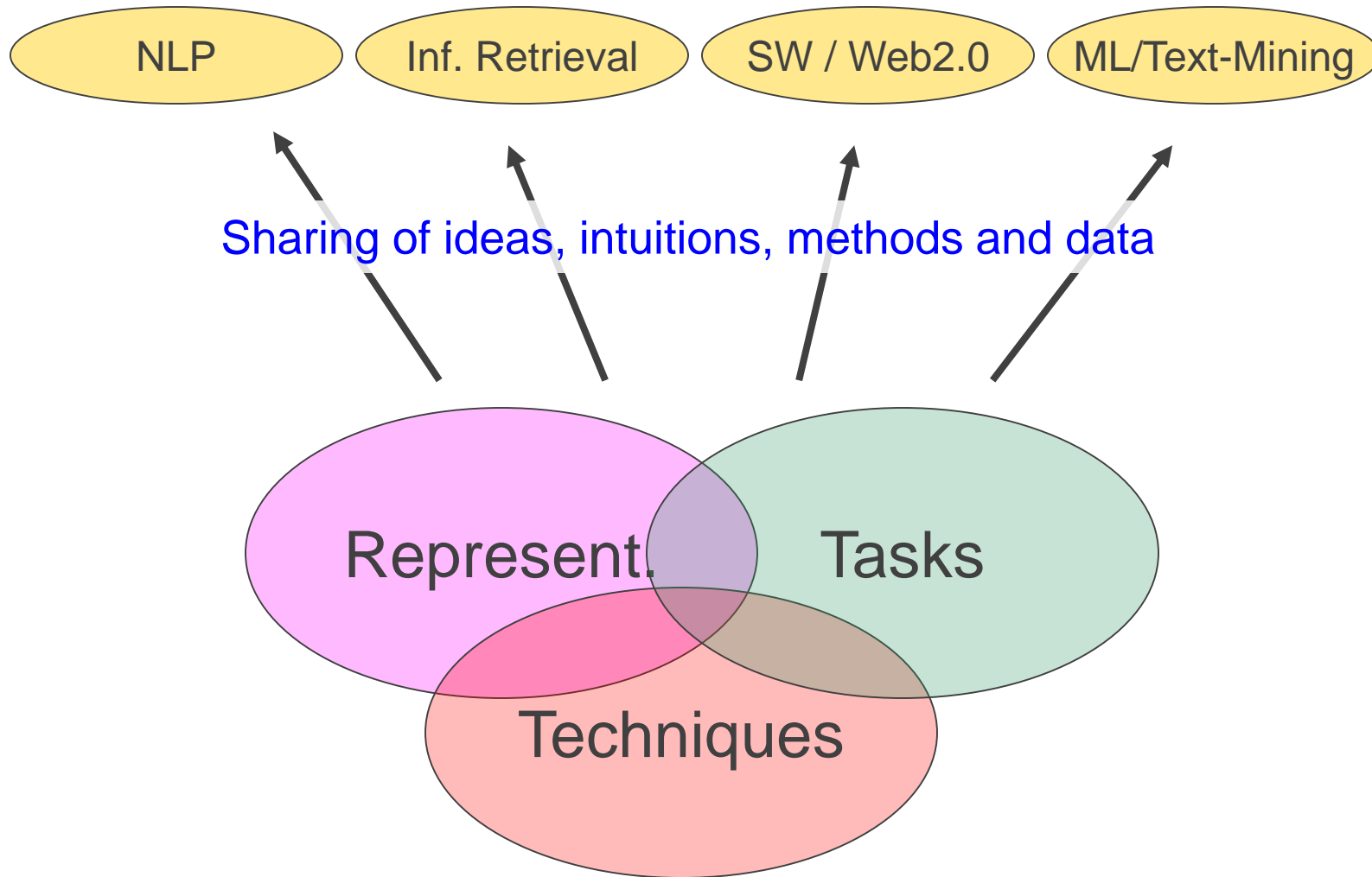
Levels of text representations

Character (character n-grams and sequences)

Words (stop-words, stemming, lemmatization)

Phrases (word n-grams, proximity features)

Part-of-speech tags

Taxonomies / thesauri

Lexical

Vector-space model

Language models

Full-parsing

Cross-modality

Syntactic

Collaborative tagging / Web2.0

Templates / Frames

Ontologies / First order theories

Semantic

UNSW
SYDNEY

Tokenization

The protein is activated by IL2.

The   protein   is   activated   by   IL2   .

Tokenizing general English sentences is relatively straightforward.

Use spaces as the boundaries

Use some heuristics to handle exceptions

Tokenization

The protein is activated by IL2.

The protein is activated by IL2 .

Convert a sentence into a sequence of tokens

Why do we tokenize?

Because we do not want to treat a sentence as a sequence of characters!

Tokenisation issues

Separate possessive endings or abbreviated forms from preceding words:

- Mary's → Mary 's
  Mary's → Mary is
  Mary's → Mary has

separate punctuation marks and quotes from words :

- Mary. → Mary .

- "new" → " new "

Tokenization

Tokenizer.sed: a simple script in sed

–http://www.cis.upenn.edu/~treebank/tokenization.html

Undesirable tokenization

- original: "1,25(OH)2D3"

- tokenized: "1 , 25 (OH) 2D3"

Tokenization for biomedical text

- Not straight-forward

- Needs domain knowledge? Machine learning?

Normalization

Need to "normalize" terms

- Information Retrieval: indexed text & query terms must have same form.
  - We want to match **_U.S.A._** and **_USA_**

We implicitly define equivalence classes of terms

- e.g., deleting periods in a term

Alternative: asymmetric expansion:

- Enter: **_window_** Search: **_window, windows_**
- Enter: **_windows_**   Search: **_Windows, windows, window_**
- Enter: **_Windows_**   Search: **_Windows_**

Potentially more powerful, but less efficient

Case folding

Applications like IR: reduce all letters to lower case

- Since users tend to use lower case
- Possible exception: upper case in mid-sentence?
  - e.g., **General Motors**
  - **Fed** vs. **fed**
  - **SAIL** vs. **sail**

For sentiment analysis, MT, Information extraction

- Case is helpful (**US** versus **us** is important)

Lemmatization

Reduce inflections or variant forms to base form

- *am, are, is $\rightarrow$ be*

- *car, cars, car's, cars' $\rightarrow$ car*

the boy's cars are different colors $\rightarrow$ the boy car be different color

Lemmatization: have to find correct dictionary headword form

Machine translation

- Spanish quiero ('I want'), quieres ('you want') same lemma as querer 'want'

Morphology

Morphemes:

- The small meaningful units that make up words
- **Stems**: The core meaning-bearing units
- **Affixes**: Bits and pieces that adhere to stems
  - Often with grammatical functions

Stemming

Reduce terms to their stems in information retrieval

Stemming is crude chopping of affixes

- language dependent
- e.g., **automate(s), automatic, automation** all reduced to **automat**.

for example compressed and compression are both accepted as equivalent to compress.

for exampl compress and compress ar both accept as equival to compress

Example: Network Analysis
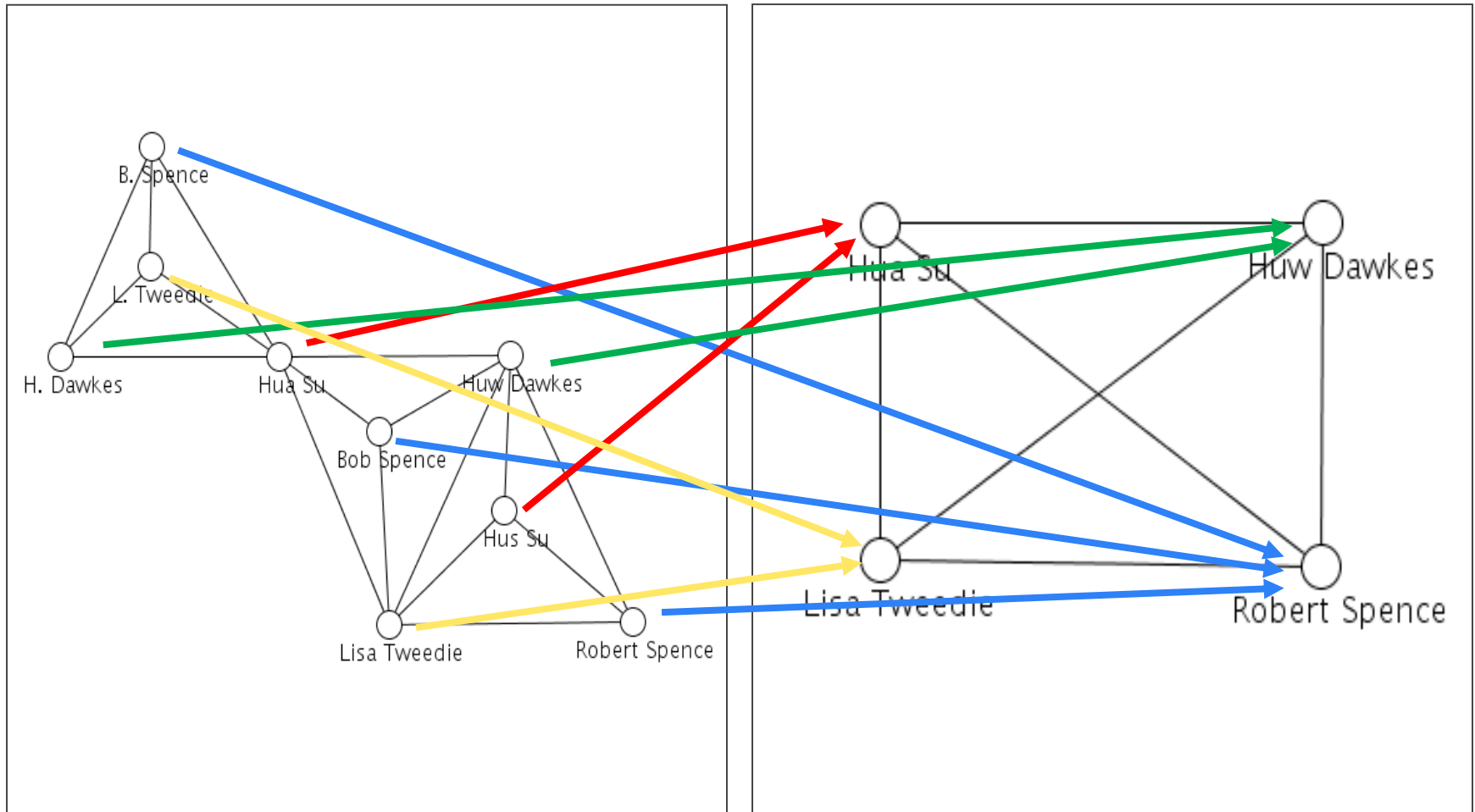
before   after

What is Duplicates Detection?

Problem of identifying and linking/grouping different manifestations of the same real world object.

Examples of manifestations and objects:

Different ways of addressing (names, email addresses, FaceBook accounts) the same person in text.

Web pages with differing descriptions of the same business.

Different photos of the same object.

…

Has many duplicate names
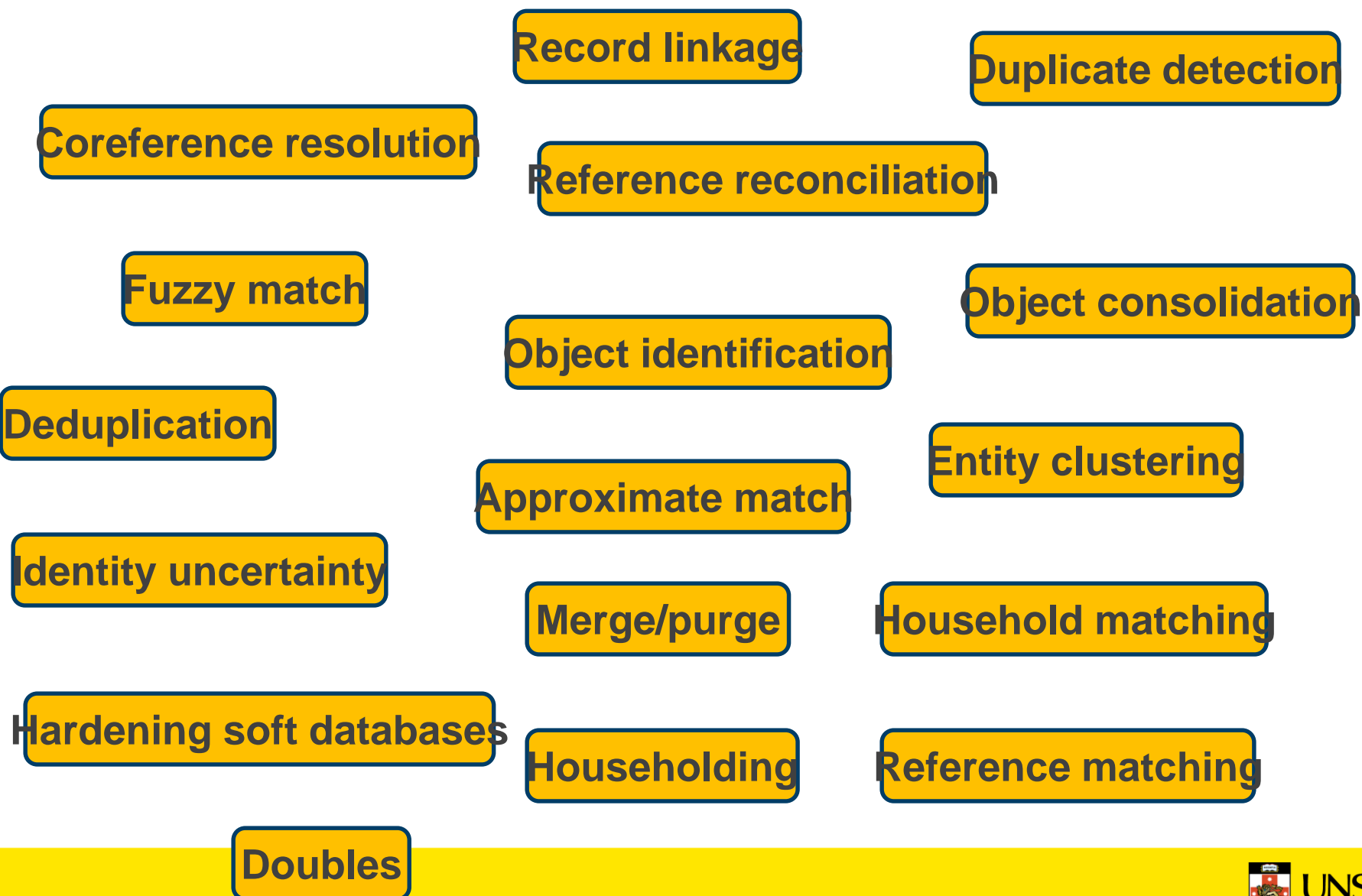
Record linkage

Duplicate detection

Coreference resolution

Reference reconciliation

Fuzzy match

Object consolidation

Object identification

Deduplication

Entity clustering

Approximate match

Identity uncertainty

Merge/purge

Household matching

Hardening soft databases

Householding

Reference matching

Doubles

Common Methods

- Pairwise Comparison Algorithms
- Algorithms for Complex Relationships
- Clustering

Common Methods

- Pairwise Comparison Algorithms
- Algorithms for Complex Relationships
- Clustering

Pairwise Match Score

Problem: Given a vector of component-wise similarities for a pair of records (x,y), compute P(x and y match).

Solutions:

1. Weighted sum or average of component-wise similarity scores. Threshold determines match or non-match.

   – 0.5*1st-author-match-score + 0.2*venue-match-score + 0.3*paper-match-score.

   – Hard to pick weights.

     • Match on last name match *more predictive* than login name.

     • Match on "Smith" *less predictive* than match on "Getoor" or "Machanavajjhala".

   – Hard to tune a threshold.

Remember Selection Bias?

Pairwise Match Score

Problem: Given a vector of component-wise similarities for a pair of records (x,y), compute P(x and y match).

Solutions:

1. Weighted sum or average of component-wise similarity scores. Threshold determines match or non-match.

2. Formulate rules about what constitutes a match.
   - (1st-author-match-score > 0.7 AND venue-match-score > 0.8) OR (paper-match-score > 0.9 AND venue-match-score > 0.9)
   - Manually formulating the right set of rules is hard.

Basic Machine Learning Rules

- $r = (x,y)$ is record pair, γ is comparison vector, M matches, U non-matches

- Decision rule $\qquad R = \dfrac{P(\gamma \mid r \in M)}{P(\gamma \mid r \in U)}$

$$R > t \;\Rightarrow r \rightarrow \text{Match}$$

$$R \leq t \;\Rightarrow r \rightarrow \text{Non-Match}$$

Data Integration
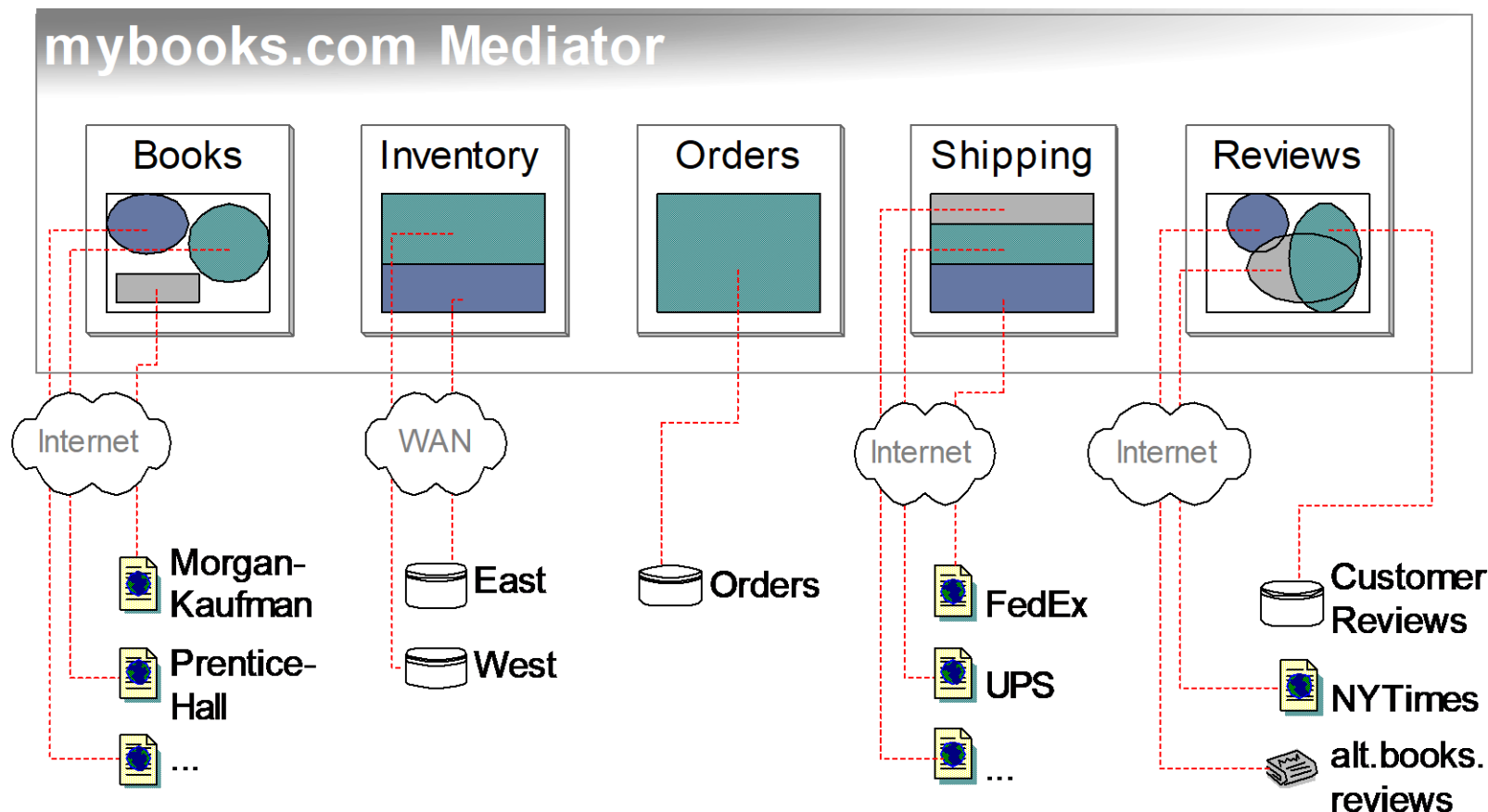
Provide **uniform access** to data available in **multiple**, **autonomous**, **heterogeneous** and **distributed** data sources

Goals of Data Integration

Provide

- Uniform (same query interface to all sources)
- Access to (queries; eventually updates too)
- Multiple (we want many, but 2 is hard too)
- Autonomous (DBA doesn't report to you)
- Heterogeneous (data models are different)
- Distributed (over LAN, WAN, Internet)
- Data Sources (not only databases).

UNSW
SYDNEY

Motivation

WWW

- Website construction
- Comparison shopping
- Portals integrating data from multiple sources
- B2B, electronic marketplaces

Science and culture

- Medical genetics: integrating genomic data
- Astrophysics: monitoring events in the sky.
- Culture: uniform access to all cultural databases produced by countries in Europe.

Current Solutions

Ad-hoc programming: Create custom solutions for each application.

Data Warehouse
• Extract all the data into a single data source

Query

Data Warehouse

Clean the data
Load Periodically
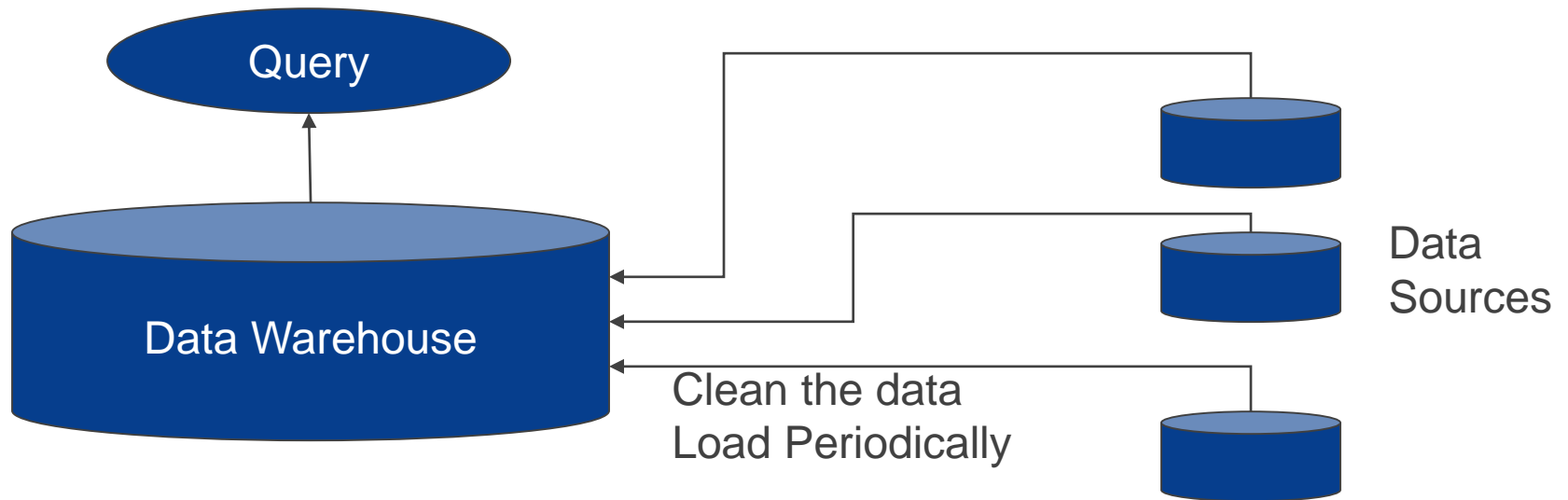
Data
Sources

Problems with DW Approach

Data has to be cleaned – different formats

Needs to store all the data in all the data sources that will ever be asked for

- Expensive due to data cleaning and space requirements

Data needs to be updated periodically

- Data sources are autonomous – content can change without notice
- Expensive because of the large quantities of data and data cleaning costs

Architecture Overview

Leave the data in the data sources

For every query over the mediated schema

• Find the data sources that have the data (probably more than one)

• Query the data sources

• Combine results from different sources if necessary

Architecture Overview
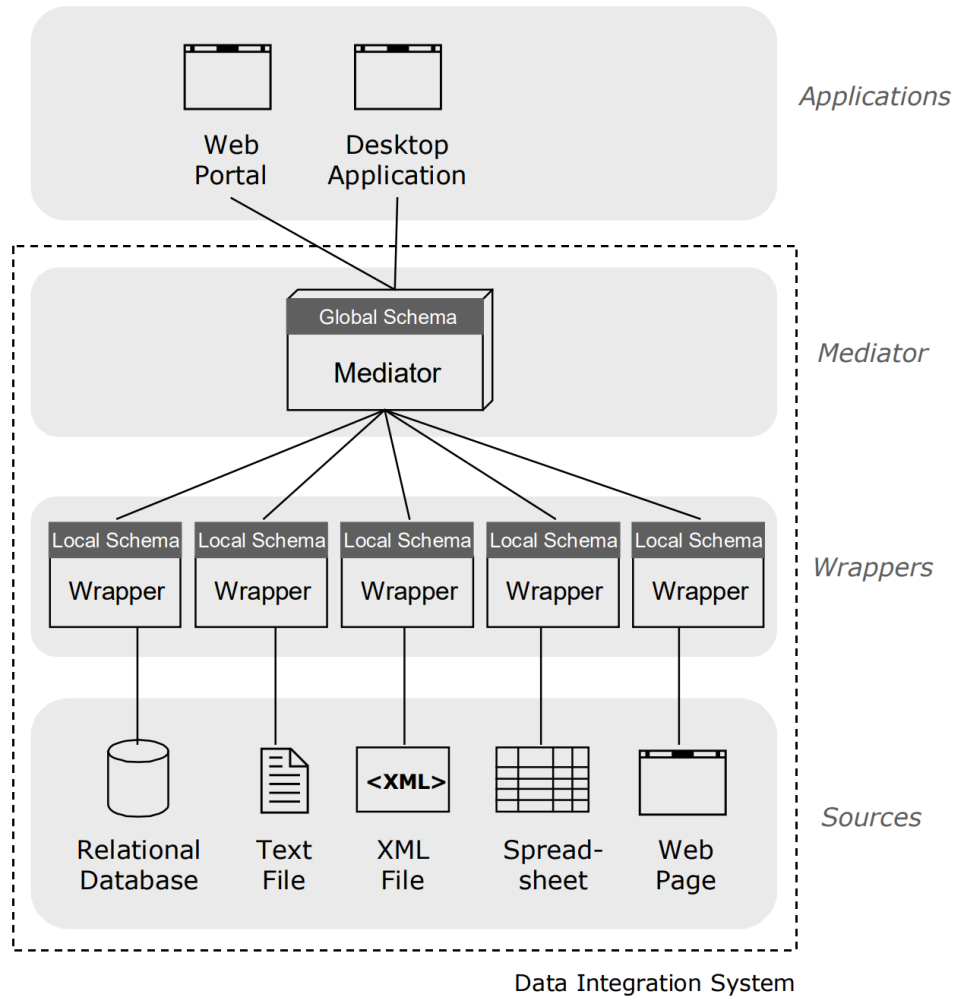
Figure 1: View-based Data Integration System (VDIS) Architecture

Virtual Integration

Mediator:

User Queries — Mediated schema

Reformulation engine

Optimizer

Execution engine — Data source catalog

Wrapper — Data Source

Wrapper — Data Source

Wrapper — Data Source
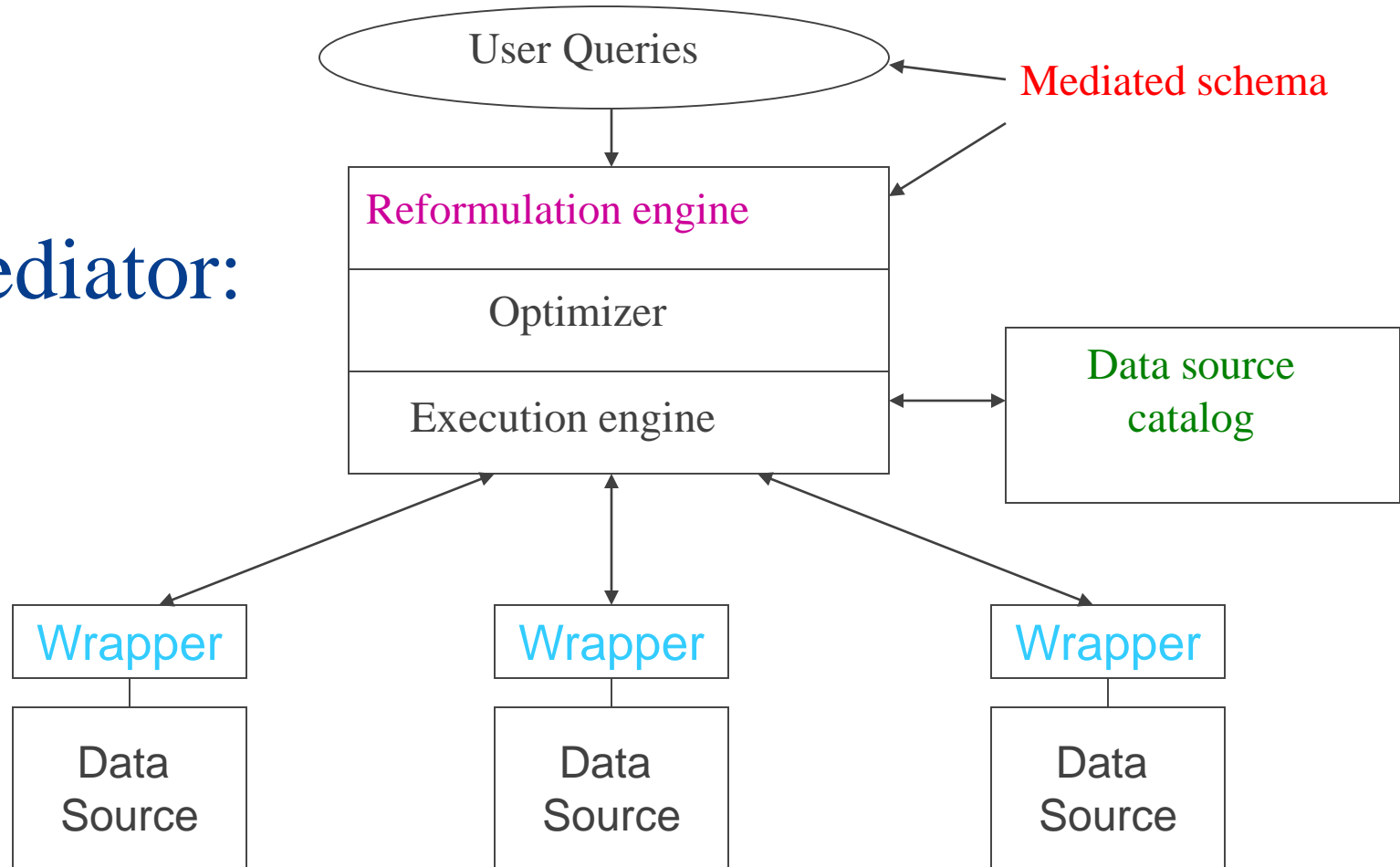
Breaks

COMP9321 2019T1

Challenges

Designing a single mediated schema
- Data sources might have different schemas, and might export data in different formats

Translation of queries over the mediated schema to queries over the source schemas

Query Optimization
- No/limited/stale statistics about data sources
- Cost model to include network communication cost
- Multiple data sources to choose from

UNSW
SYDNEY

Challenges (2)

Query Execution

- Network connections unreliable – inputs might stall, close, be delayed, be lost
- Query results can be cached – what can be cached?

Query Shipping

- Some data sources can execute queries – send them sub-queries
- Sources need to describe their query capability and also their cost models (for optimization)

Challenges (3)

Incomplete data sources

- Data at any source might be partial, overlap with others, or even conflict
- Do we query all the data sources? Or just a few? How many? In what order?

Wrappers

Sources export data in different formats

Wrappers are custom-built programs that transform data from the source native format to something acceptable to the mediator

HTML

XML

```
<b> Introduction to DB </b>
<i> Phil Bernstein </i>
<i> Eric Newcomer </i>
 Addison Wesley, 1999
```

→

```
<book>
<title> Introduction to DB </title>
<author> Phil Bernstein </author>
<author> Eric Newcomer </author>
<publisher> Addison Wesley </publisher>
<year> 1999 </year>
</book>
```

Wrappers(2)

Can be placed either at the source or at the mediator

Maintenance problems – have to change if source interface changes

Data Source Catalog

Contains meta-information about sources

- Logical source contents (books, new cars)
- Source capabilities (can answer SQL queries)
- Source completeness (has *all* books)
- Physical properties of source and network
- Statistics about the data (like in an RDBMS)
- Source reliability
- Mirror sources
- Update frequency

Schema Mediation

Users pose queries over the mediated schema

The data at a source is visible to the mediator is its local schema

Reformulation: Queries over the mediated schema have to be rewritten as queries over the source schemas

How would we do the reformulation?

Reformulation Problem

Given:

• A query Q posed over the mediated schema

• Descriptions of the data sources

Find:

• A query Q' over the data source relations, such that:
  – Q' provides only *correct answers* to Q, and
  – Q' provides *all* possible answers to Q given the sources.

Approaches to Specifying Source Descriptions

Global-as-view (GAV): express the mediated schema relations as a set of views over the data source relations

Local-as-view (LAV): express the source relations as views over the mediated schema.

Can be combined…?

Example Scenario

A mediator for movie databases

- Want to provide the information about movies and where they are playing

Global-as-View

Mediated schema as a view over the local schemas

Mediated Schema: Movie(title, dir, year, genre)

Data Sources and local schemas:

S1[Movie(title,dir,year,genre)]

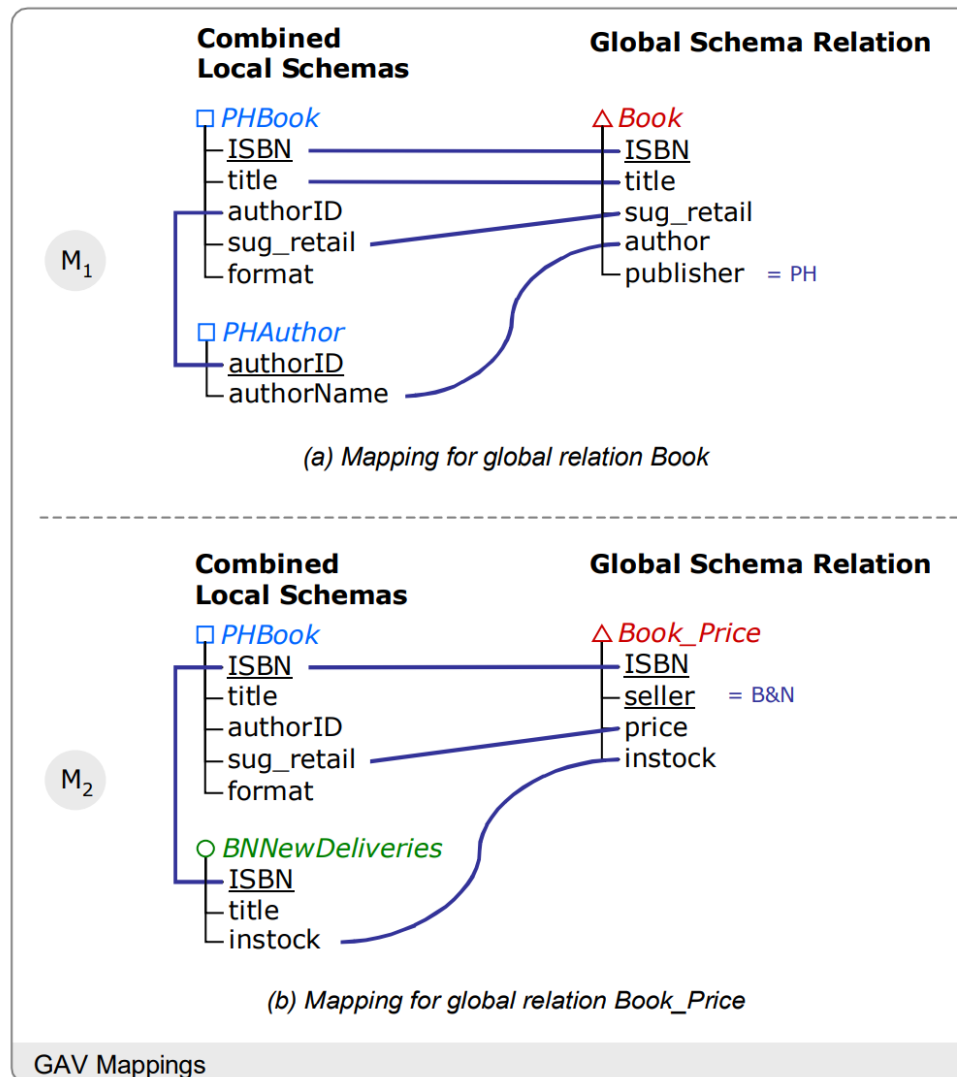S2[Director(title,dir), Movie(title,year,genre)]

Create View Movie As

Select * from S1.Movie

Union

Select * from S2.Director, S2.Movie

where S2.Director.title = S2.Movie.title

Global-as-View – A Running Example

(a) Mapping for global relation Book

(b) Mapping for global relation Book_Price

GAV Mappings

Yannis Katsis et. al, View-based Data Integration

UNSW SYDNEY

Global-as-View

Simply unfold the user query by substituting the view definition for mediated schema relations

Difficult to add new

- For instance, as observed in mapping M2 of the running example, a mapping has to explicitly specify how data from multiple sources are combined to form global relation tuples.

since the global schema is expressed in terms of the sources, global relations cannot model any information not present in at least one source.

- For instance, the Book relation in the example could not contain an attribute for the book weight, since no source currently provides it. In other words, the value of each global attribute has to be explicitly specified.

Local-as-View

LAV follows the opposite direction expressing each local schema as a function of the global schema.

LAV essentially corresponds to the "source owners view" of the system by describing which data of the global database are present in the source.

Local-as-View

Local schemas as views over the mediated schema

Mediated Schema: Movie(title, dir, year, genre)

Data Sources and local schemas:

S1[Movie(title,dir,year,genre)]

S2[Director(title,dir), Movie(title,year,genre)]
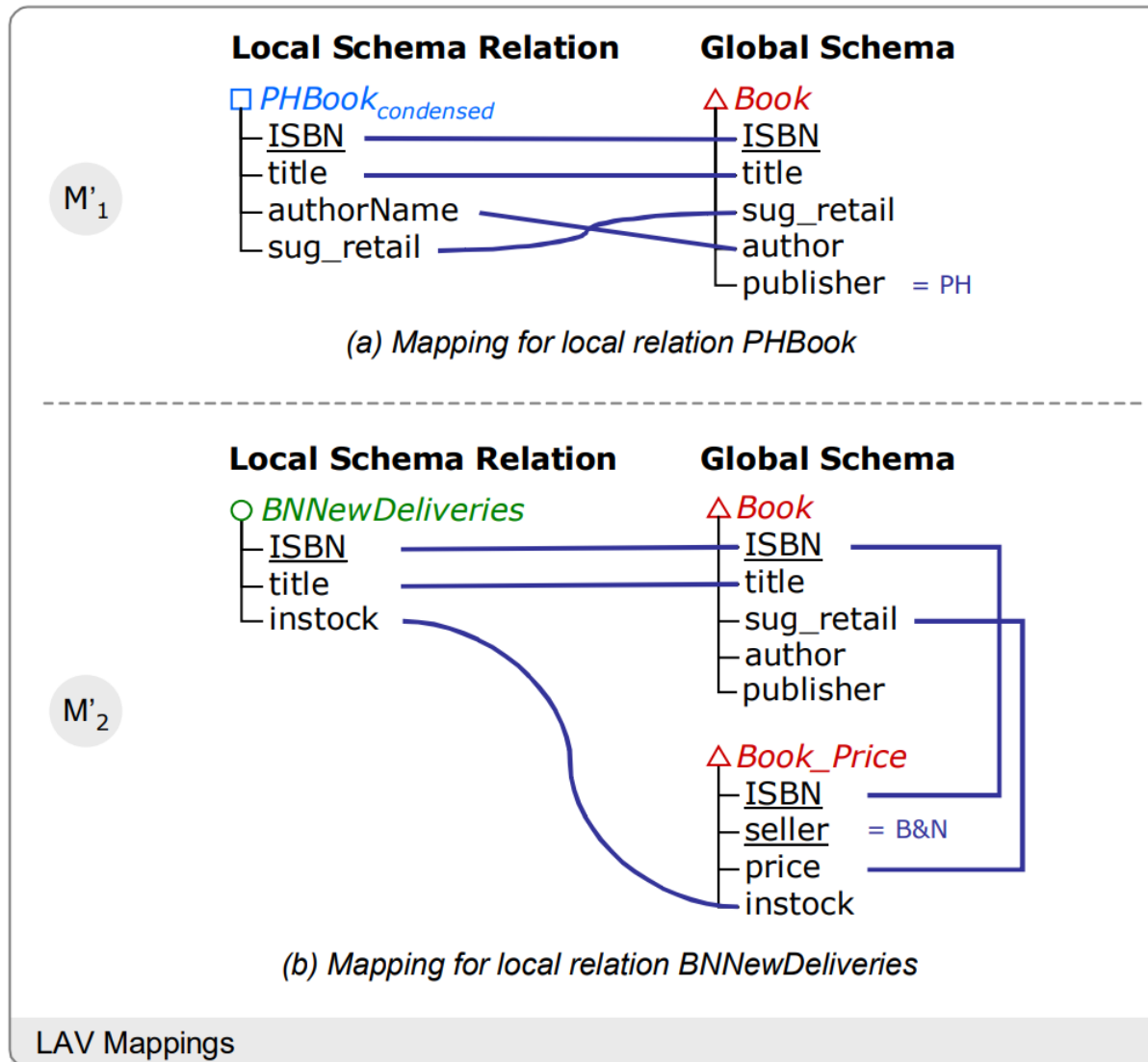
Create Source S1.Movie As

Select * from Movie

Create Source S2.Movie As

Select title, year, genre from Movie

Create Source S2.Director As

Select title,dir from Movie

Local-as-View – A Running Example

(a) Mapping for local relation PHBook

(b) Mapping for local relation BNNewDeliveries

LAV Mappings

Yannis Katsis et. al, View-based Data Integration

Local-as-View Summary

Very flexible. You have the power of the entire query language to define the contents of the source.

In contrast to GAV mappings, LAV mappings have a declarative flavor, since, instead of explaining how the global database can be created, they describe what information of the global database is contained in each local database. Adding sources is easy: they're independent of each other.

LAV addresses many of GAV problems with the most important being that sources can register independently of each other, since a source's mappings do not refer to other sources in the system.

Query reformulation: *answering queries using views!*

Query Optimization

Sources specify their capabilities if possible

- Transformation rules define the operations they can perform

- Sources might also specify cost models of their own

Cost model might be parametrized

- Mediator can estimate cost of transferring data by accumulating statistics from earlier transfers
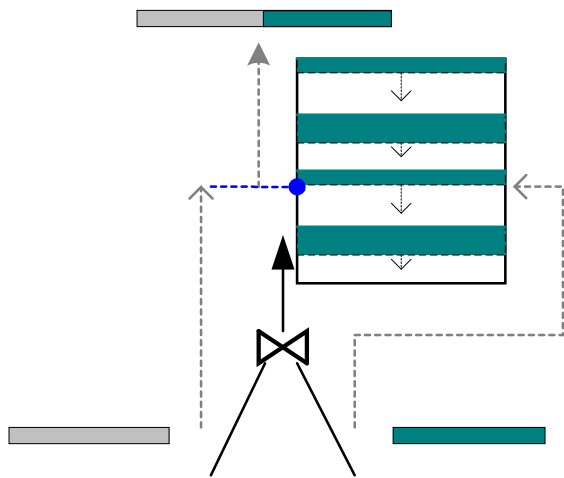
Adaptive Query Processing

Adaptive query operators

• Aware that network communication might fail

Interleave Query Optimization and Execution

• Optimize query once with available limited statistics

• Execute the query for some time, collect statistics

• Re-optimize query again with improved statistics

• Resume execution… repeat

Double Pipelined Hash Join

Hash Join

- Partially pipelined: no output until inner read
- Asymmetric (inner vs. outer) — optimization requires source behavior knowledge
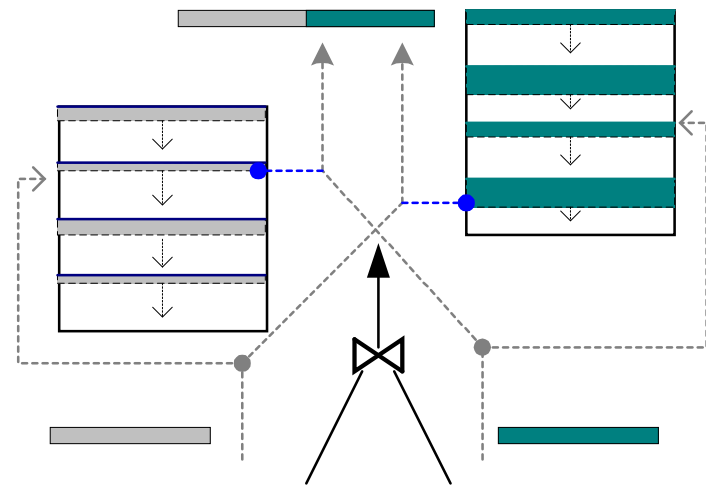
Double Pipelined Hash Join

- Outputs data immediately
- Symmetric — requires less source knowledge to optimize

Other Problems

Automatic Schema Matching

- How do I know what the view definitions are?

- Can I learn these definitions automatically?

Streaming Data

- The data is not stored in a data source, but is streaming across the network

- How do we query a data stream?

Other Problems(2)

Peer-to-Peer databases

- No clear mediator and data source distinction
- Each peer can both have data and fetch the rest of it
- Similar to Napster, Gnutella – except we want to share data that is not necessarily a single file

Examples - Cleaning

COMP9321 2019T1

~cs9321/19T1/dataCleaningSlides.html