

# Multi-column Substring Matching For Database Schema Translation (And other wild thoughts while shaving)

Robert H. Warren<sup>1</sup>    Dr. Frank Wm Tompa<sup>1</sup>

<sup>1</sup>rhwarren,fwtompa@uwaterloo.ca  
David R. Cheriton School of Computer Science  
University of Waterloo  
Waterloo, Canada

Bertinoro PhD School on Data and Service Integration,  
2006



# Where in the world am I from?



## Database Integration

- ...seen as a large, monolithic, one-off project.
- ...solved by database and domain experts with the time and motivation.

### *But!*

- The number, size and complexity of databases keeps growing. (+10,000 tables, +1,600 columns)
- Integration is an every day issue. (Semantic web, opportunistic data sources...)
- Multiple representation standards in use. (22 Locales)
- Standardized database access (JDBC, ODBC) possible.
- End user knows the data is available, can't access it and wants it right NOW!

## Database Integration

- ...seen as a large, monolithic, one-off project.
- ...solved by database and domain experts with the time and  
an ⇒ Need automation to deal with this problem.

### *But!*

- The number, size and complexity of databases keeps growing. (+10,000 tables, +1,600 columns)
- Integration is an every day issue. (Semantic web, opportunistic data sources...)
- Multiple representation standards in use. (22 Locales)
- Standardized database access (JDBC, ODBC) possible.
- End user knows the data is available, can't access it and wants it right NOW!

# Database schema matching and translation

## Objective

A generalisable method capable of resolving complex schema matches and the translation required to convert the instance data using substrings concatenation.

## Example

- Name(Warren, Rob) in database  $D \rightarrow$  First(Rob) + Last(Warren) in  $D'$
- 2005/05/29 in database  $D \rightarrow$  05-29-2005 in database  $D'$
- LastName(warner) + Birthdate(980102) in  $D \rightarrow$  Userid(warn98) in  $D'$ .
- PartNumber(04350306) in  $D \rightarrow$  Number(0435) + PlantId(03) + Year(2006) in  $D'$ .

## Problem formalization

### Definition

For a given target database table  $T_2$  with a target column  $A$  ...and a source table  $T_1$  with a set of likely source columns  $(B_1, B_2, \dots, B_n)$

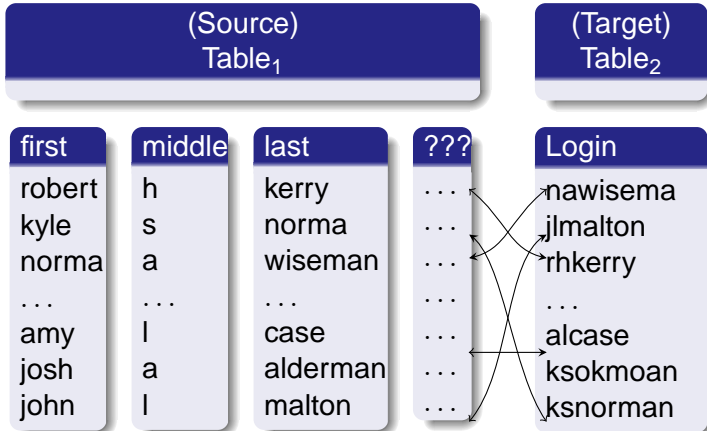
Find a transformation such that:

$A = \omega_1 + \omega_2 + \dots + \omega_\nu$  Where  $\omega_j$  represents a substring of column  $B_j$

### Translation model

$t' = t [\beta_1^{x_1 \dots y_1} + \beta_2^{x_2 \dots y_2} + \dots + \beta_\nu^{x_\nu \dots y_\nu}]$  (chars  $x_\nu \dots y_\nu$  of col  $B_\nu$ )

## Basic example



## Basic example

(Source)  
Table<sub>1</sub>

(Target)  
Table<sub>2</sub>

How to infer translation?

```
select substring(first from 1 for 1) ||  
substring(middle from 1 for 1) || last as login  
into target_table from ...
```

Solution:

Iteratively select substrings from “best-fit” columns while performing a simple form of record linkage.

## Basic example - Find initial column. (1)

(Source) Table <sub>1</sub>				(Target) Table <sub>2</sub>
first	middle	last	???	Login
robert	h	kerry	...	nawisema
kyle	s	norma	...	jmalton
norma	a	wiseman	?	rhkerry
...	...	...	...	...
amy	l	case	...	alcase
josh	a	alderman	...	ksokmoan
john	l	malton	...	ksnorman

## Basic example - Find a partial translation. (1)

(Source)  
Table  
Generate candidate translations

malton + wiseman  $\rightarrow$  %[1-2]%

malton + jmalton  $\rightarrow$  %[1-EOL]

malton + alcasa  $\rightarrow$  %[2-3]%

...

kerry + rhkerry  $\rightarrow$  %[1-EOL]

...

Highest occurrence: %last[1-EOL]

john  
john

l

malton  
malton

...

ksnorman  
ksnorman

Use string edit distance to create candidate translation.

## Basic example - Search for additional columns. (1)

(Source) Table <sub>1</sub>				(Target) Table <sub>2</sub>
first	middle	last	???	Login
robert	r	kerry	...	nawisema
kyle	s	norma	...	jmalton
norma	a	wiseman	...	rhkerry
...	...	...	...	...
amy	l	case	...	alcase
josh	a	alderman	...	ksokmoan
john	l	malton	...	ksnorman

Sample the tuples formed from translation formula.

## Basic example - Search for additional columns. (1)

(Source)		(Target)	
Table	Table	Table	Table
Generate candidate translations			
robert + rhkerry → first[1-1]%last[1-EOL]			
...			
...			
(Keep track of all candidates and their frequencies.)			
norman	a	wiseman	rhkerry
...	...	...	...
amy	l	case	alcase
josh	a	alderman	ksokmoan
john	l	malton	ksnorman

Sample the tuples formed from translation formula.

## Basic example - Search for additional columns. (2)

(Source) Table <sub>1</sub>				(Target) Table <sub>2</sub>
first	middle	last	???	Login
robert	h	kerry	...	nawisema
kyle	s	norma	...	jmalton
norma	a	wiseman	...	rhkerry
...	...	...	...	...
amy	l	case	...	alcase
josh	a	alderman	...	ksokmoan
john	l	malton	...	ksnorman

Sample the tuples formed from current translation formula

## Basic example - Search for additional columns. (2)

(Source)				(Target)
Table				Table
Generate candidate translations				
h + rhkerry → first[1-1]middle[1-1]last[1-EOL]				
...				
...				
(Keep track of all candidates and their frequencies.)				
norma	a	wiseman	...	rhkerry
...	...	...	...	...
amy	l	case	...	alcas
josh	a	alderman	...	ksokmoan
john	l	malton	...	ksnorman

Sample the tuples formed from current translation formula

## Basic example - Search for additional columns. (2)

(Source) Table		(Target) Table	
<b>Ending condition</b>			
No unknowns remain within: $t' = t [\beta_1^{x_1 \dots y_1} + \beta_2^{x_2 \dots y_2} + \dots + \beta_\nu^{x_\nu \dots y_\nu}]$ Login = first[1-1] + middle[1-1] + last[1-EOL]			
... amy josh john	... l a l	... case alderman malton	... alcase ksokmoan ksnorman

Sample the tuples formed from current translation formula

## Experimental setup - Noise column

Add and populate the following noise columns:

- A random RFC-2822 timestamp.
- A random street address.
- A random long integer.
- A random value, variable length string.
- *War and peace* by Leo Tolstoy.

### Definition

Simulate noisy matching environment and ensure proper algorithmic behavior.

## CiteSeer & DBLP Dataset

### CiteSeer

Extracted 526,000 records from OAI dump.  
Created Title, Year and Author (15) columns.  
Created Citation column from Title, Year and First Author.  
(Successfully matched at 1% sampling.)

### DBLP

Extracted 233,000 records from web dump.  
Created Title, Year and Author (15) columns.  
Created Citation column from Title, Year and First Author.  
(Successfully matched at 1% sampling.)

## Cross Citeseer and DBLP Dataset translation

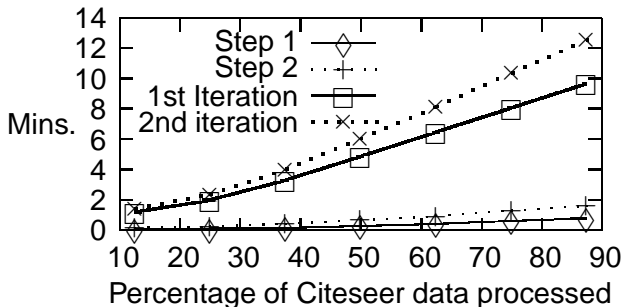
### Expected result

Match Citeseer Citation column to DBLP source table.  
Only 714 records match across Title, Year and First Author.

### Actual result

Citeseer Citation = DBLP Title + DBLP Year + DBLP Second Author.  
378 citations have their First and Second authors reversed!  
Returned mapping is “correct” according to the data.

## Incremental wall-clock performance



Estimated complexity

$$O(w * n * s_1 * s_2)$$

# Overall

- Previous approaches required specialized domain specific matchers to form both the match and the translation.
- This algorithm is a generalized algorithm for string-based concatenations matches.
- Meant to function as part of larger database integration framework.
- It is un-supervised, does not need examples or a known record overlap and can be implemented using basic SQL statement.

# Overall

- Pre ma **This is all old stuff!! Now what?** specific
  - This ma
    - Sampling extremely large tables.
  - This cor
    - Data-driven, machine readable descriptions of extremely large database.
  - Me
    - Questing for linear time data matching of attributes.
  - It is
    - Using ontologies as integration negotiation documents.
- statement. SQL

## Sampling extremely large tables

...or MY database is bigger than YOUR database.

- Currently two approaches: equidistant and random sampling.
- As the size of a table grows, a traversal can become almost impossible.
- Behavior is like that of a data stream or tape drive.
- Disk access can also be more efficient if we use it as a linear device.
- Can we use clever statistics to guess how deep into the table to go?

## Data-driven, machine readable descriptions of extremely large database.

Q:

How are we going to advertise and describe data in a format that automated integration system can actually use. (And maybe not lie about?)

# Questioning for linear (or better) time data matching of attributes.

- Currently two methods:  $q$ -gram or KL-divergence.
- What happens when we can't read the entire table?
- Are information theory methods useful? ...and faster?
- Can we use any clever sampling techniques?

# Using ontologies as integration negotiation documents.

...mostly used for hierarchy of concepts now.

- Ontological standards a good way to document the data exchange process (e.g.: constraints, dependencies, ...) in a machine readable way.
- What about 'pushing' database information rules to the outside world? (e.g.: Records are only accepted if they contain attributes (First, Middle and Last) or if the attributes (Name and DOB) are available.

# THE END