An Indexing Structure for Automatic Schema Matching

SMDB 2007, Istanbul

Fabien Duchateau, Zohra Bellahsène, Mathieu Roche

LIRMM, Université Montpellier 2

France

Mark Roantree

ISG, Dublin City University

Ireland

Outline

- Motivations
- BtreeMatch: an index structure for accelerating schema matching
  - Semantic Aspect
  - Performance Aspect
  - Some Performance Results
- Conclusion and Future Work

Motivations

- Finding semantic correspondences between 2 schemas still a challenging issue
- Available semi automatic matchers focus on the quality aspect of matching
- More and more large schemas, especially on the Web

Our Approach

An Index Structure for Automatic Schema Matching

- Semantic aspect
  - terminological (Levenhstein and 3grams)
  - structural (context based using cosine measure)
- Performance aspect
  - indexing structure (B-tree)

BtreeMatch: Semantic Aspect (1/4)

- Context of node *n*
  - represents the most *important* neighbour nodes of *n*
  - each of them is assigned a weight depending on the relationship with *n*

$$\omega_1(n_c, n_i) = 1 + \frac{K}{\Delta d + |lev(n_c) - lev(n_a)| + |lev(n_i) - lev(n_a)|}$$

- We define StringMatching as the average between
  - Levenhstein distance
  - n-grams

BtreeMatch: Semantic Aspect (2/4)

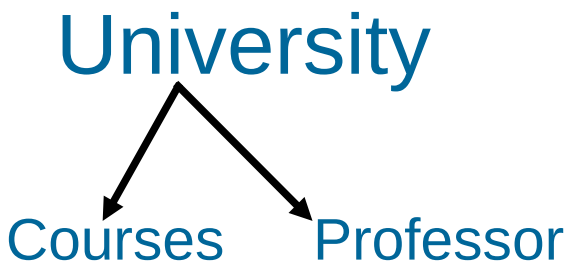
3grams(GradCourses, Courses) = 0.2

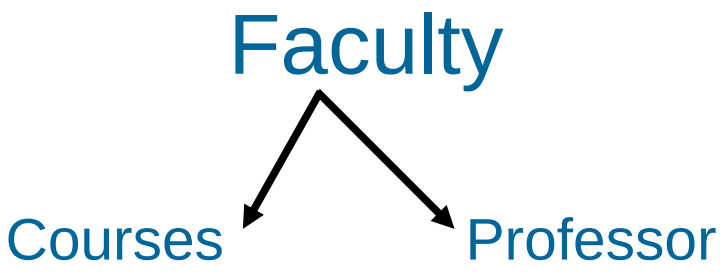
Lev(GradCourses, Courses) = 0.42

StringMatching(GradCourses, Courses) = 0.31

StringMatching(Professor, FullProfessor) = 0.38

BtreeMatch: Semantic Aspect (3/4)





StringMatching(Faculty, University) = 0.002

Context(University) = {University, Courses, Professor}

Context(Faculty) = {Faculty, Courses, Professor}

CosineMeasure(Context(University),Context(Faculty)) = 0.37

BtreeMatch: Semantic Aspect (4/4)

- Acceptable quality of matching
  - better than COMA++ in some scenarios

	Precision	Recall	F-measure
COMA++	1	0.56	0.72
BtreeMatch	0.62	0.89	0.73

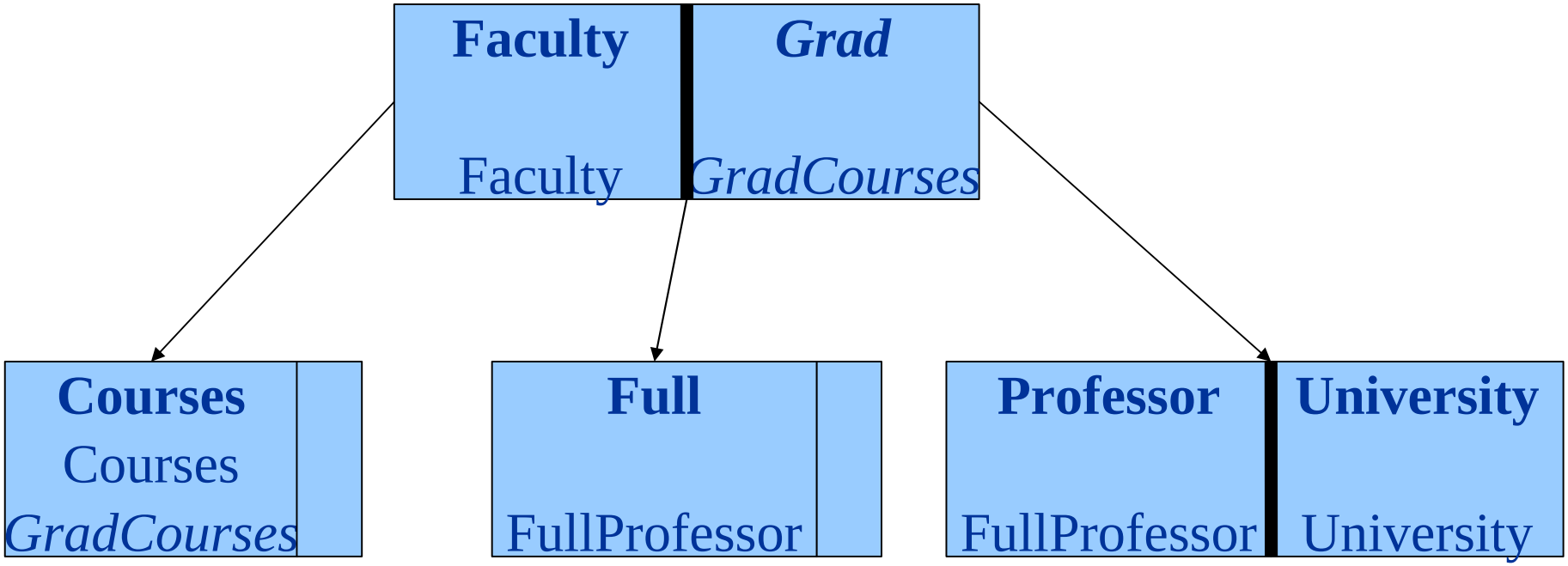
  - can be tuned
    - to restrict the context
    - to increase the similarity and replacement thresholds

BtreeMatch: Performance Aspect

- B-tree indexing structure to restrict the search space because "*most of similar labels share a common token*"

- Algorithm
  - Labels are divided into tokens
  - Each token is an index in the B-tree with references to all labels containing this token
  - Match search of a label is limited to the labels referenced by the common tokens
  - Else the whole B-tree may be searched using the cosine measure

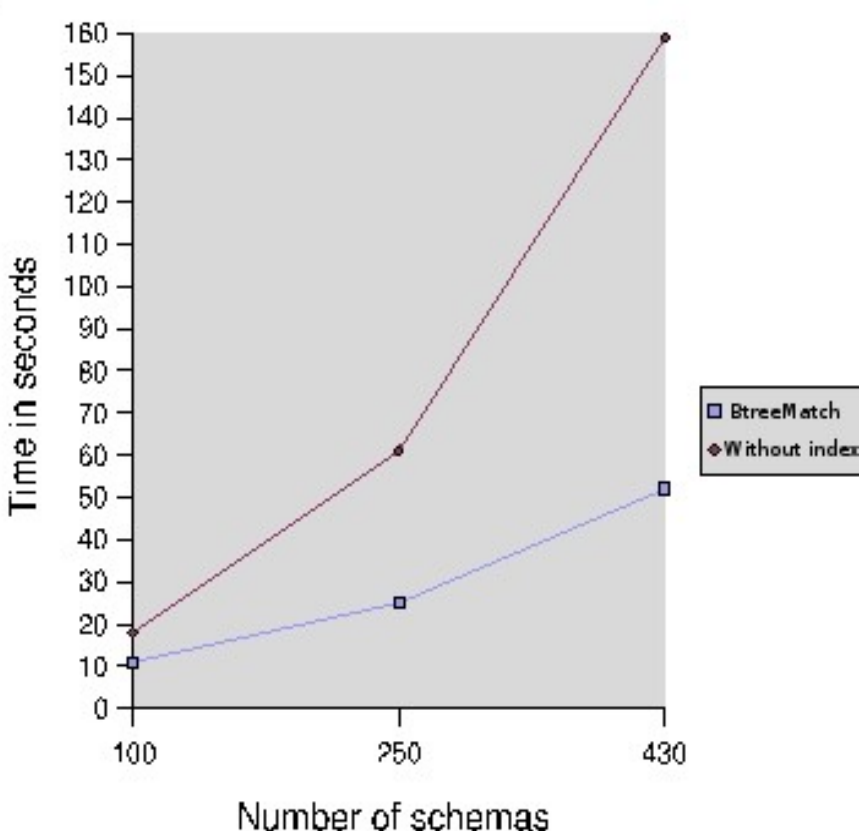
BtreeMatch: Performance Aspect



- Searching a match for *GradCourses* involves
  - creation of an index for ***Grad***
  - only evaluating and discovering a similarity between *GradCourses* and Courses due to their common token

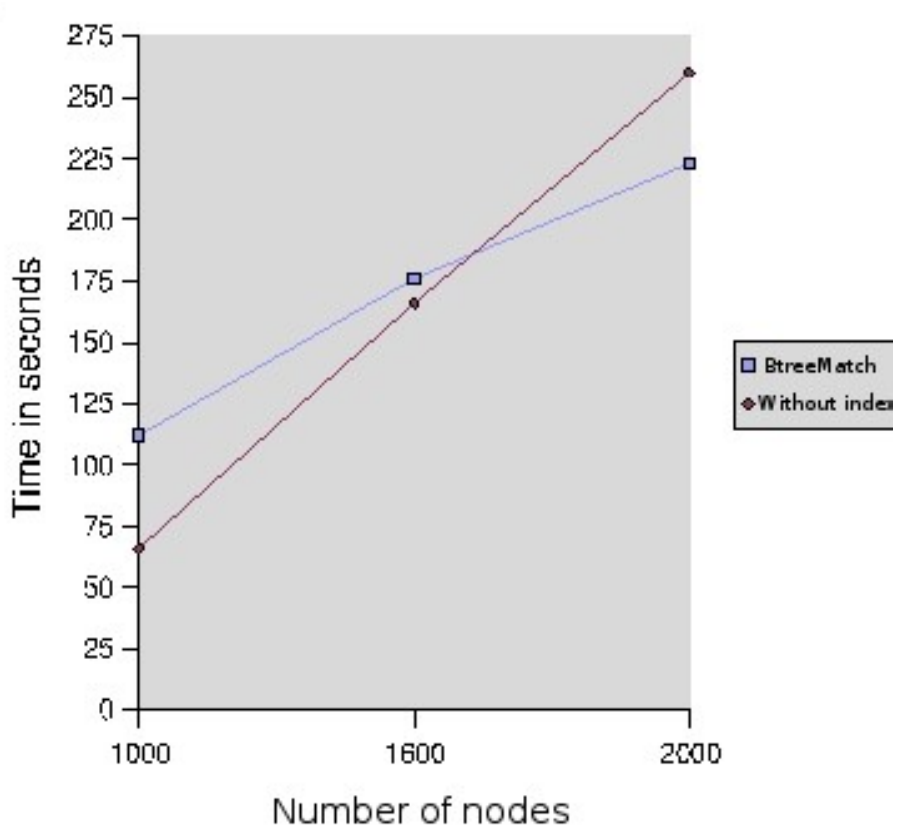
BtreeMatch: Performance Results

- Comparison of the performance with and without the indexing structure, depending on the number of schemas using XCBL and OASIS schemas



BtreeMatch: Performance Results

- Comparison of the performance with and without the indexing structure, depending on the size of the schemas using XCBL and OASIS schemas



Conclusion and Future Work

- An automatic schema matching tool that
  - handles many large schemas.
  - provides an acceptable quality of matching.
  - tuning is not automatic
- Discovering complex mappings
- Exploring other index structures (hashtables)