30th International Conference on Conceptual Modeling (ER 2011)

Brussels, Belgium

A Hidden Markov Model Approach to Keyword-based Search over Relational databases

Sonia Bergamaschi[1], **Francesco Guerra**[1], Silvia Rota[1], Yannis Velegrakis[2]

[1] University of Modena and Reggio Emilia, Italy
[2] University of Trento, Italy

Keyword based searching over RDBs

- Keyword-based searching is **an attractive alternative** to traditional SQL queries

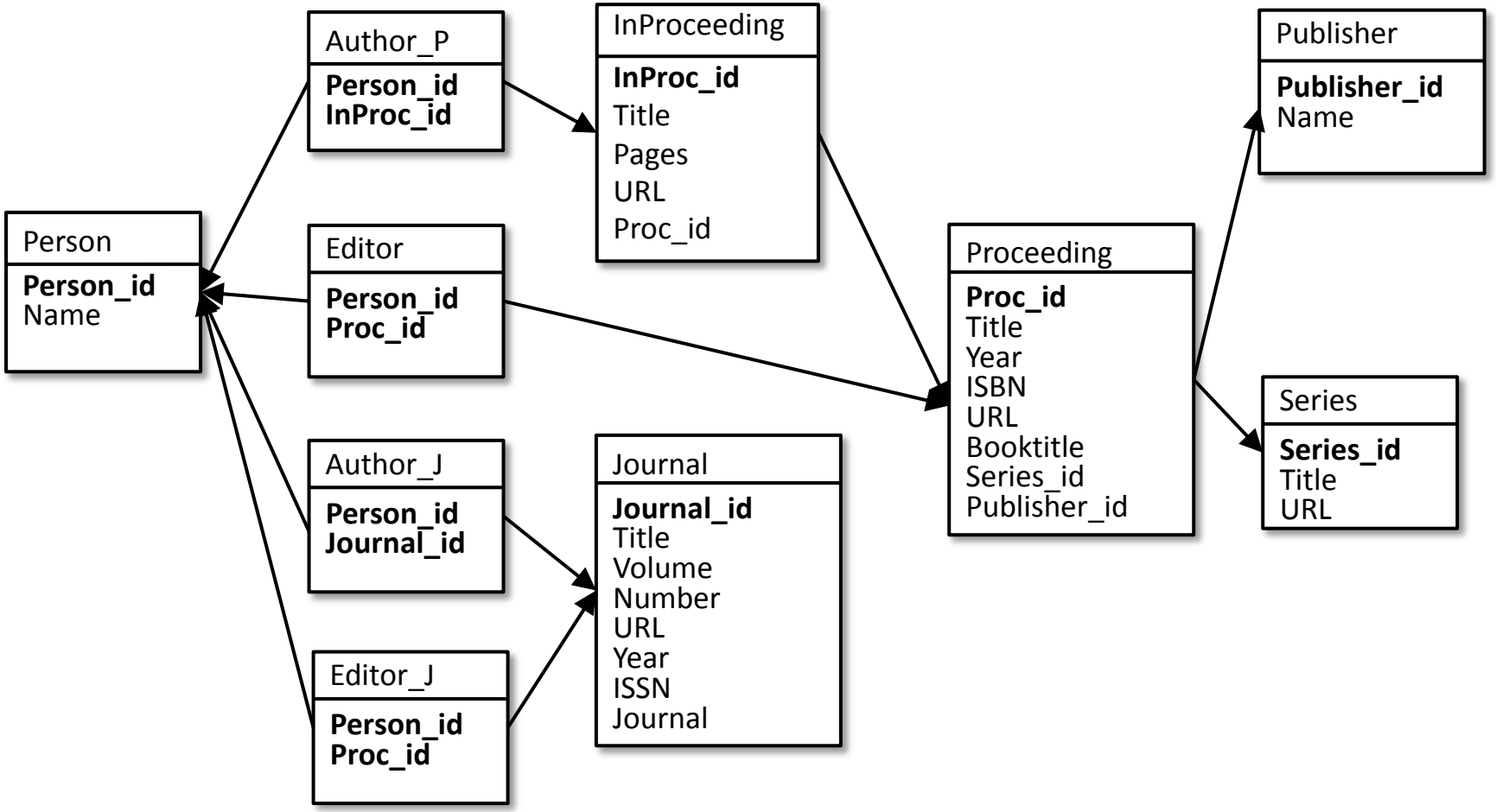
- The main challenges are
  - to discover the database structures that contain the keywords
  - to explore how these structures are inter-connected to form an answer

- Existing techniques typically suffer from two main limitations:
  1. they are **based on indexes** about the data
  2. no enough attention has been paid to the **inter-dependencies among the keywords**

Our approach

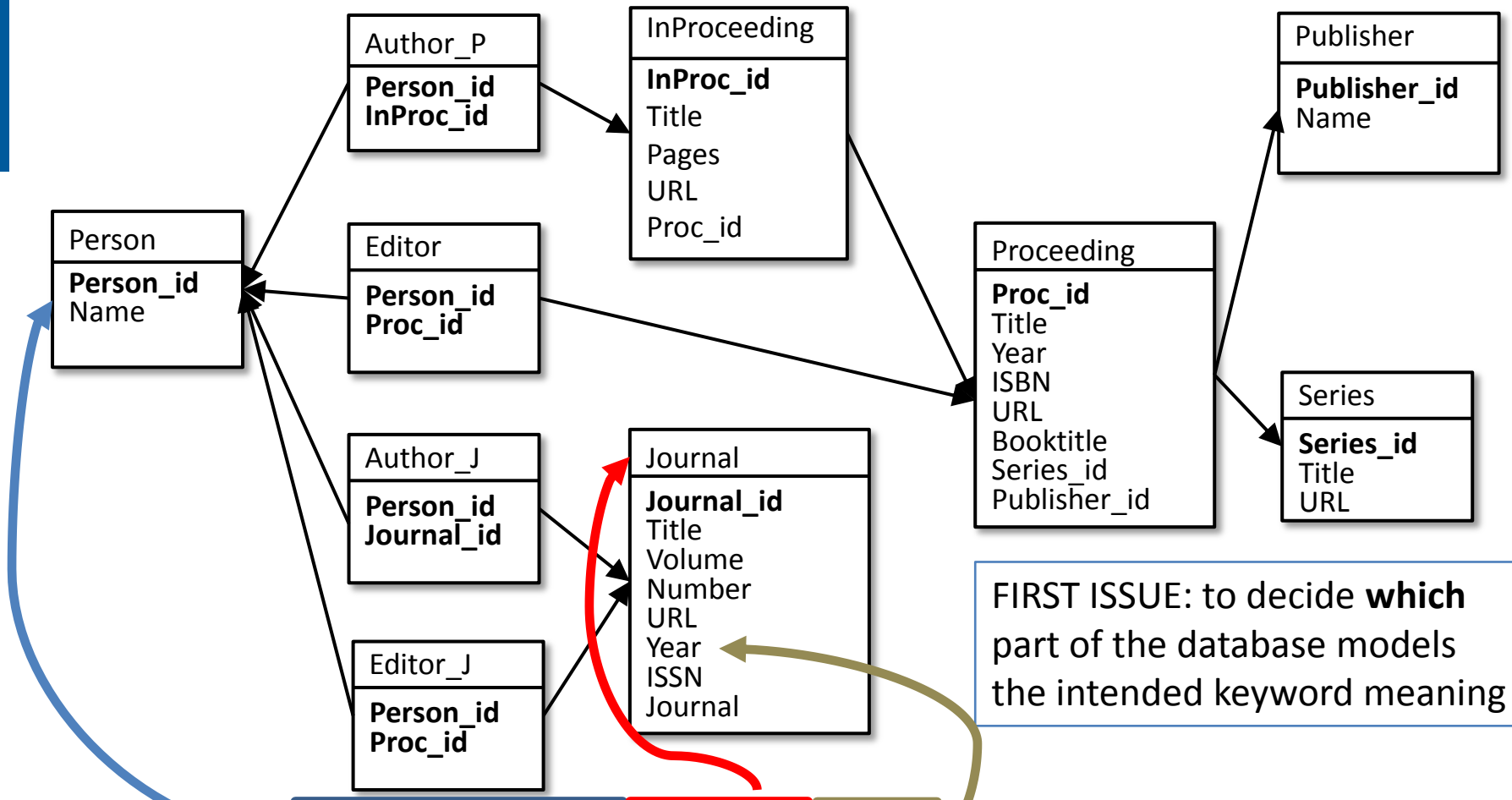
- We propose a keyword-based search engine that **does not rely on the knowledge of the database instance**
  - **configurations**, i.e., mappings between the keywords and the database terms
  - **interpretations** of the configurations , i.e., SQL queries for which every keyword corresponds to some database term
- Our approach is based on
  - a Hidden Markov Model for mapping the user keywords into database terms
  - a method for providing a parameter setting not relying on any training data
    - **heuristics** rules, **similarity** measures and a variation of the **HITS algorithm**

Motivating example

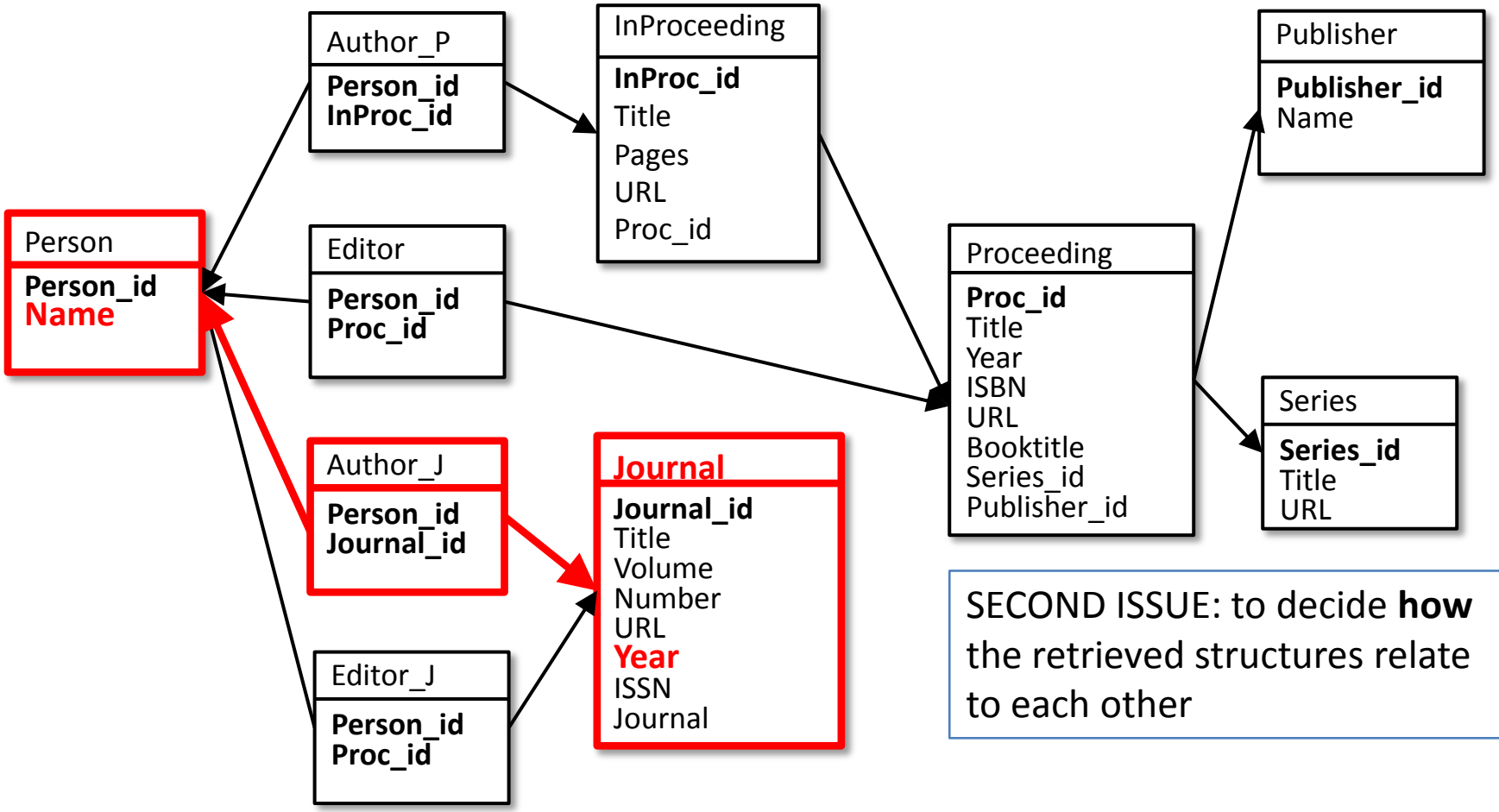


- User query «`Garcia-Molina Journal 2011`»

Motivating example

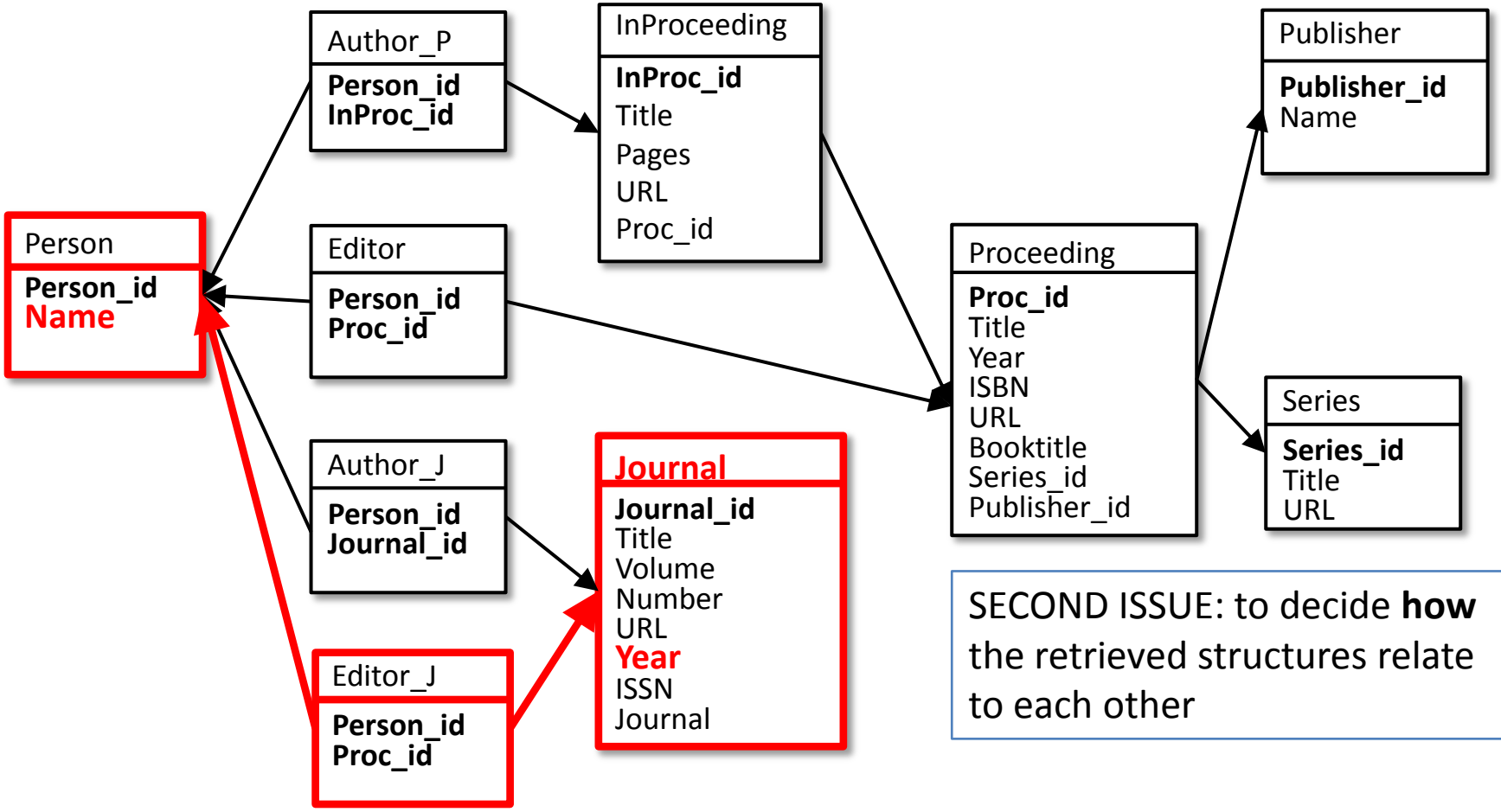


Motivating example



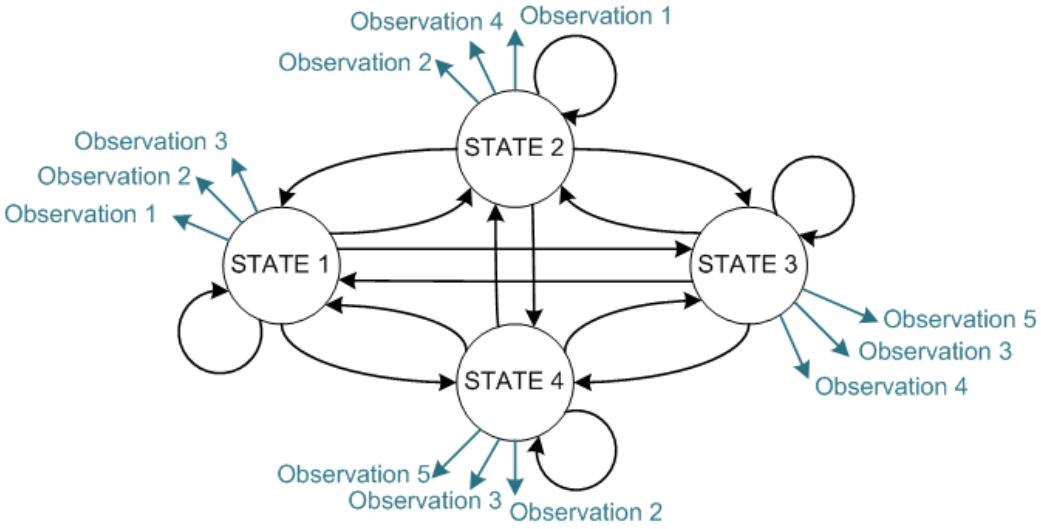
- User query «`Garcia-Molina Journal 2011`»
  - Journals where Garcia Molina was an author in 2011

Motivating example



- User query «`Garcia-Molina Journal 2011`»
  - Journals where Garcia Molina was an editor in 2011

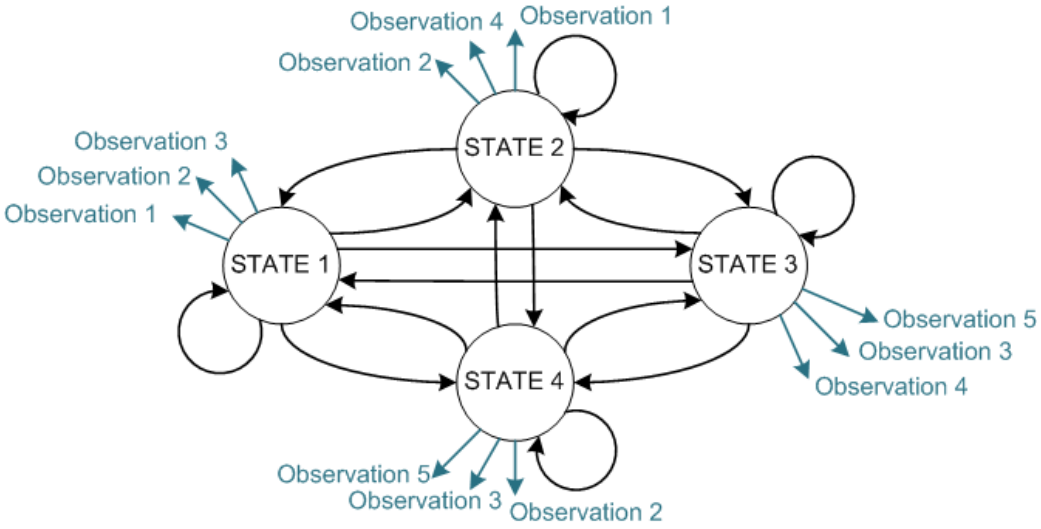
Hidden Markov Models



- A HMM can be used to address these problems:
  - **Prediction:** Given a Hidden Markov Model λ and a sequence of observations O, we would like to find out the state sequence which has the highest probability of generating O
  - ~~**Training:** Given a training set of observation sequences O, we would like to learn the model λ that maximizes the probability of generating O~~

Heuristic Rules

Modeling keyword search with a HMM



- HMMs are **parametric models**, and the parameters are:
  - N = number of states
  - Π = initial state probabilities
  - A = transition probability matrix N x N
  - B = emission probability for each state
- Mapping keywords to database elements:
  - keywords are the observable sequence O
  - the database elements are the hidden states to be inferred
- How do we calculate the HMM parameters?
  - N = || database vocabulary ||
  - Π → HITS algorithm
  - A → heuristic rules
  - B → similarity measures

Number of states

- N = || database vocabulary ||
  - Database vocabulary: the set of all the names of the tables, the attributes, and all the domains of the database.

Transition probability matrix A

- **Heuristic rules** based on the semantic relationships existing between the database terms (aggregation and generalization inferred by foreign key constraints)
  - the transition probability values decrease with the distance of the states.
  - Higher probabilities are associated to:
    - transitions to elements inside the same table
    - transitions between elements in tables connected through foreign keys

Emission probabilities

- The database vocabulary is composed of schema and domain elements.
- For schema elements:
  - Similarity measures
    - similarity value = P(schema element | keyword)
    - the Bayes theorem to calculate P(keyword | schema element), which is the emission probability
- For domain elements:
  - data types/ regular expressions/Google similarity to calculate the similarity of keywords and domains

Initial state probabilities

- We use an adaptation of the **HITS algorithm**.
  - the HITS algorithm is a link analysis algorithm used to rank web pages.
  - the algorithm calculates two rank values for each state: **hub** and **authority**.
    - a state is a good hub if it links to many good authority states, and a good authority is a state linked by many good hubs.
  - we take into account the number of attributes minus the number of foreign keys in a table

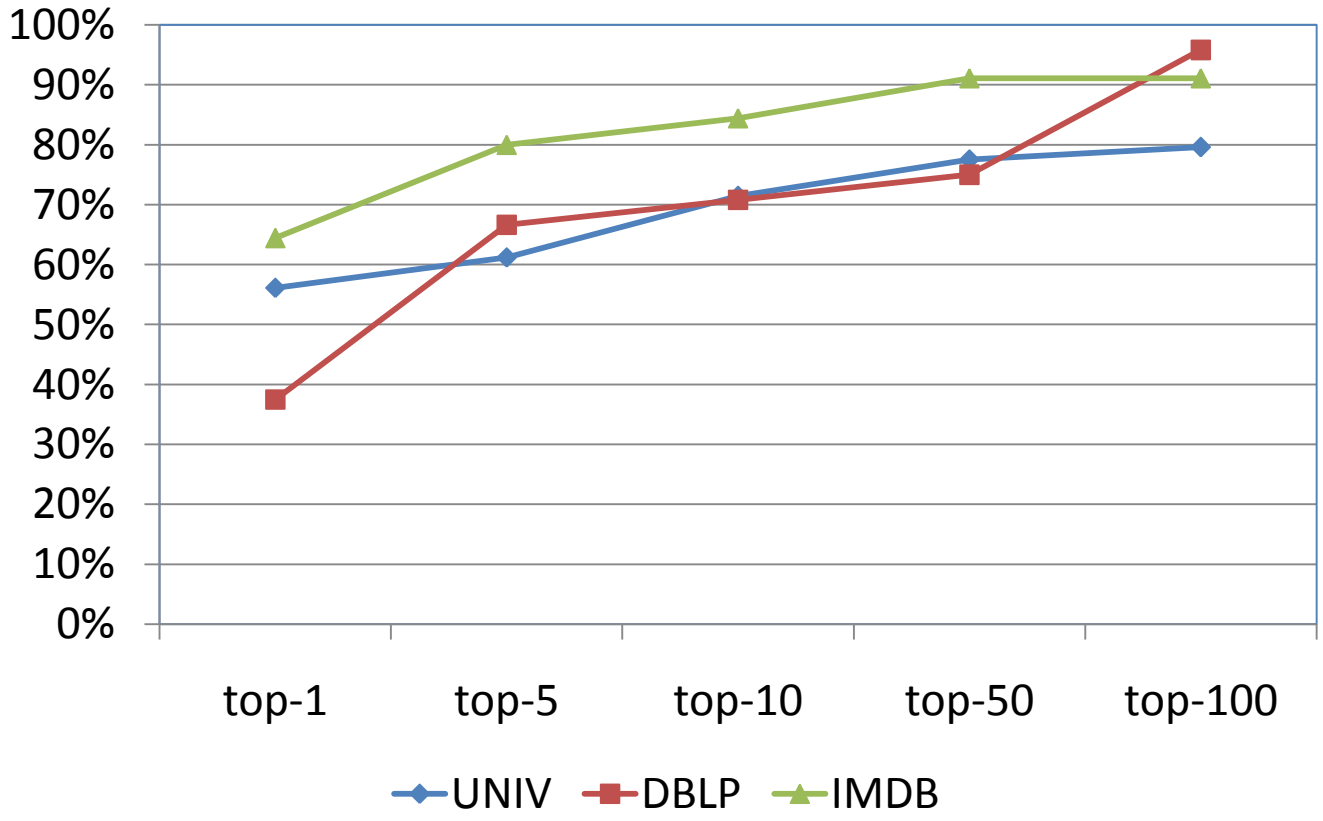
Prediction

- Given a HMM and a sequence of observations we use the **List Viterbi algorithm** to predict which are the best correspondent top-k state sequences, i.e. the database terms.
- The algorithm, which is a dynamic programming procedure, makes the following assumptions:
  - both the observations and the states must be in a sequence
  - a single element in the observation needs to correspond to exactly one state
  - computing the most likely state sequence up to a certain point *t* must depend only on the observed element at point *t*, and the most likely state at point *t* − *1*

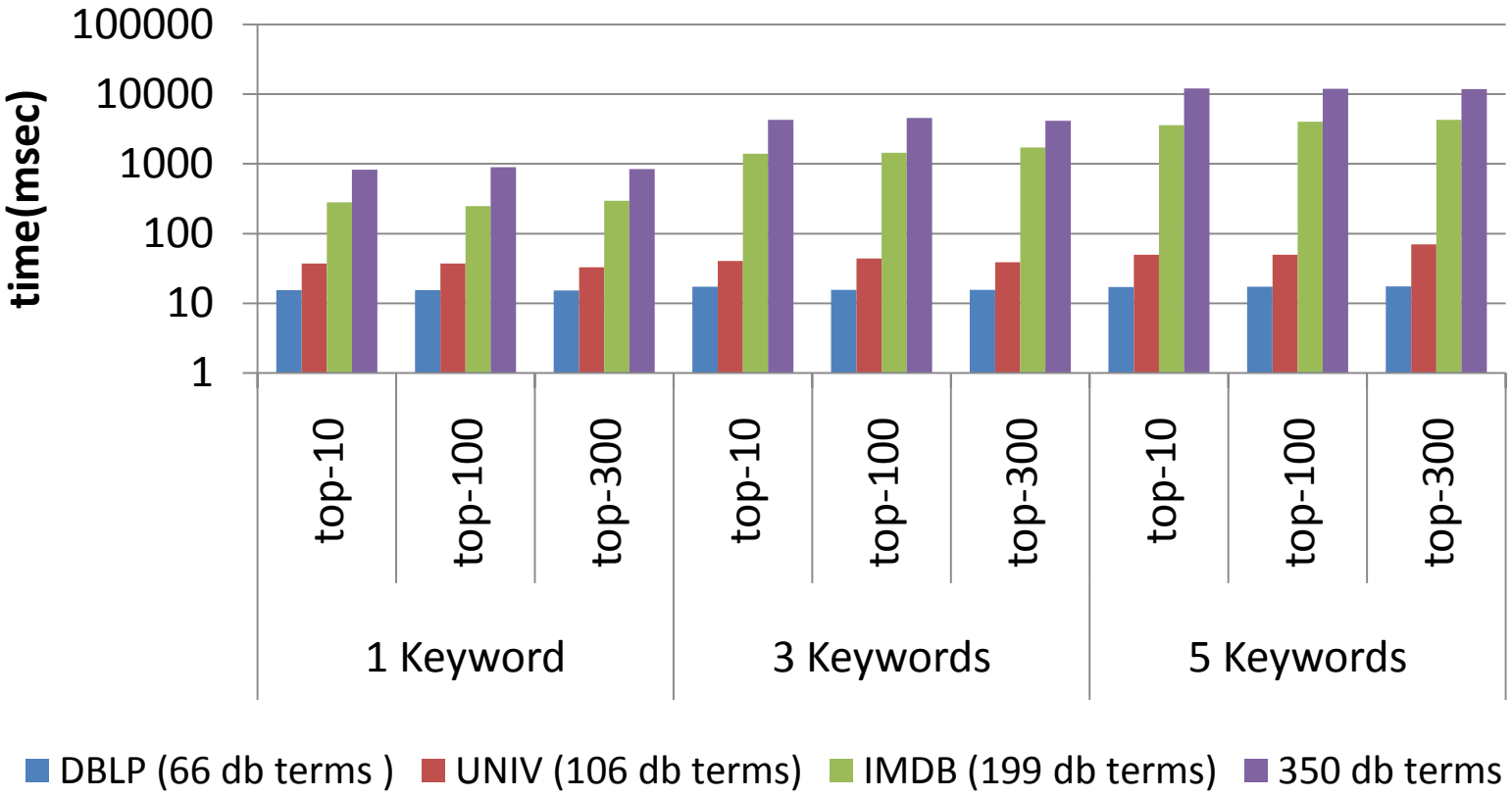
Evaluation

- We selected for our experiments three real data sets.
  - a university database
  - a fraction of the IMDB database
  - a fraction of the DBLP database
- 29 real users were asked to provide a set of keyword queries.
- A database expert translated each keyword query into a configuration.
- We used a total of 99, 44 and 30 queries for the university, the IMDB, and the DBLP database.

Evaluation - Effectiveness



Evaluation - Efficiency

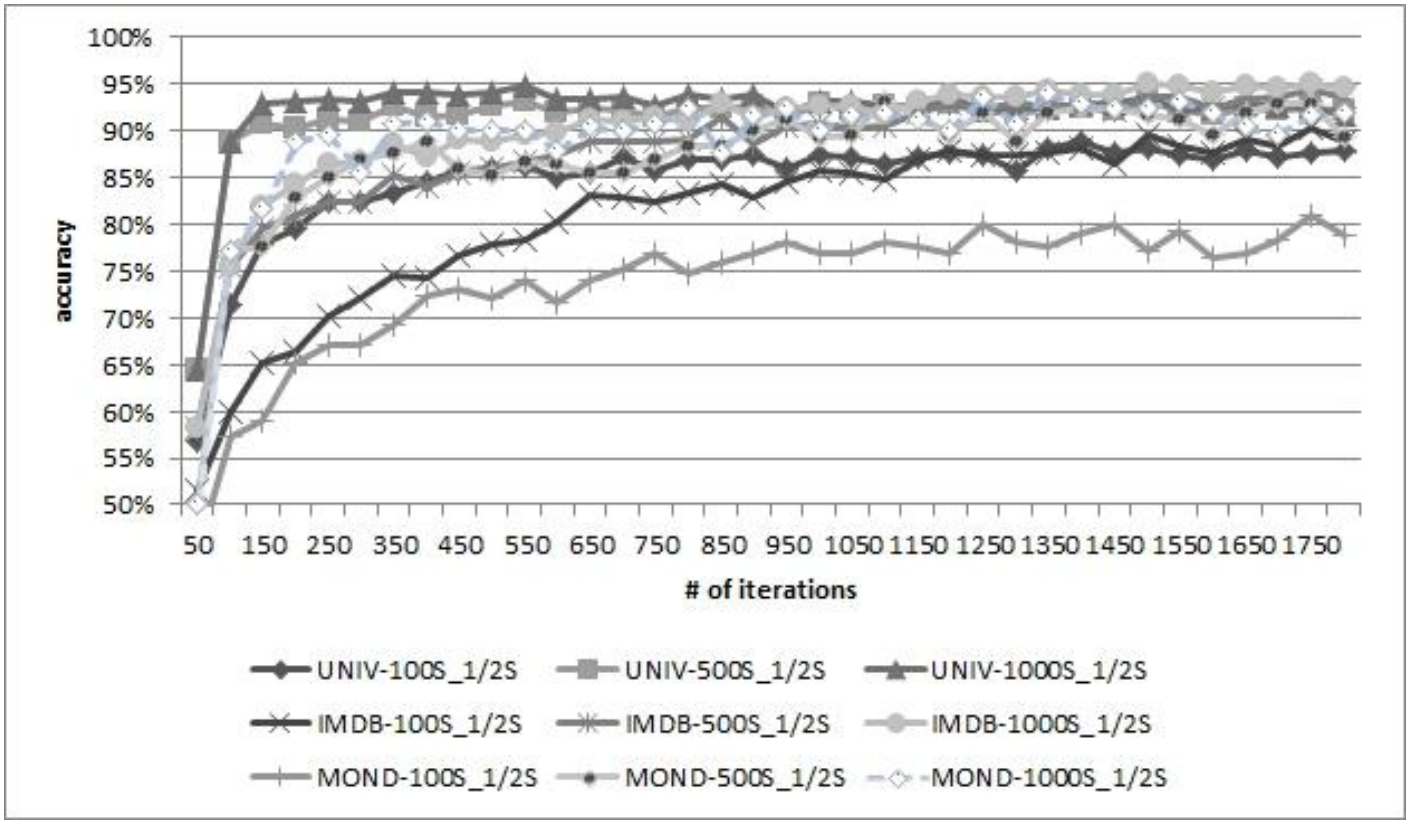


Conclusion

- We described a probabilistic keyword-based searching system for relational databases
  - based on a HMM for computing the top-K best mappings of the query keywords into the database terms
  - does not rely on any particular training
  - the approach has been implemented in a prototype (KEYRY) shown in the demo session
- Future work will be devoted to develop an effective solution for ranking the interpretations (i.e., the SQL queries).

Appendix

Convergence of the training algorithm



Effectiveness with a training dataset