STUDIA INFORMATICA 2012
Volume 33 Number 2B (106)

Dawid SKURZOK, Bartosz ZIÓŁKO
Akademia Górniczo-Hutnicza, Katedra Elektroniki
Aleksander POHL
Uniwersytet Jagielloński, Katedra Lingwistyki Komputerowej
Tomasz JADCZYK, Mariusz MĄSIOR
Akademia Górniczo-Hutnicza, Katedra Elektroniki

COMPARATIVE STUDY OF SQLITE AND BERKELEY DB IMPLEMENTATIONS OF N-GRAM MODEL OF POLISH LANGUAGE

Summary. Aspects of applying databases in computational linguistics are presented. An example of a dictionary and an n-gram model of the AGH automatic speech recognition system is depicted as well. An advantage of Berkeley DB, comparing to SQLite in time efficiency aspect is shown on this case.

Keywords: speech recognition, natural language processing, dictionary

PORÓWNAWCZE STUDIUM IMPLEMENTACJI MODELU N-GRAMOWEGO JĘZYKA POLSKIEGO W SQLITE I BERKELEY DB

Streszczenie. Przedstawiono zagadnienia dotyczące stosowania baz danych w lingwistyce komputerowej. Omówiono także przykład słownika i modelu n-gramowego systemu rozpoznawania mowy AGH. Pokazano na tym przykładzie znaczącą przewagę implementacji wykonanej w Berkeley DB nad implementacją SQLite w sensie wydajności czasowej.

Słowa kluczowe: rozpoznawanie mowy, przetwarzanie języka naturalnego, słownik

1. Introduction

Most of the progress in automatic speech recognition (ASR) [1] was done for English but is still below the level of human speech recognition capability. In case of Polish, there is no commercial large vocabulary continuous ASR software. Polish speech contains very high-frequency phones and the language is highly inflected and non-positional. The AGH ASR system (Fig.1) [2] allows to search the best path in a word hypotheses lattice by 2- and 3-gram model [3-5], collected from over 10 GB of text and semisupervisely corrected [6].

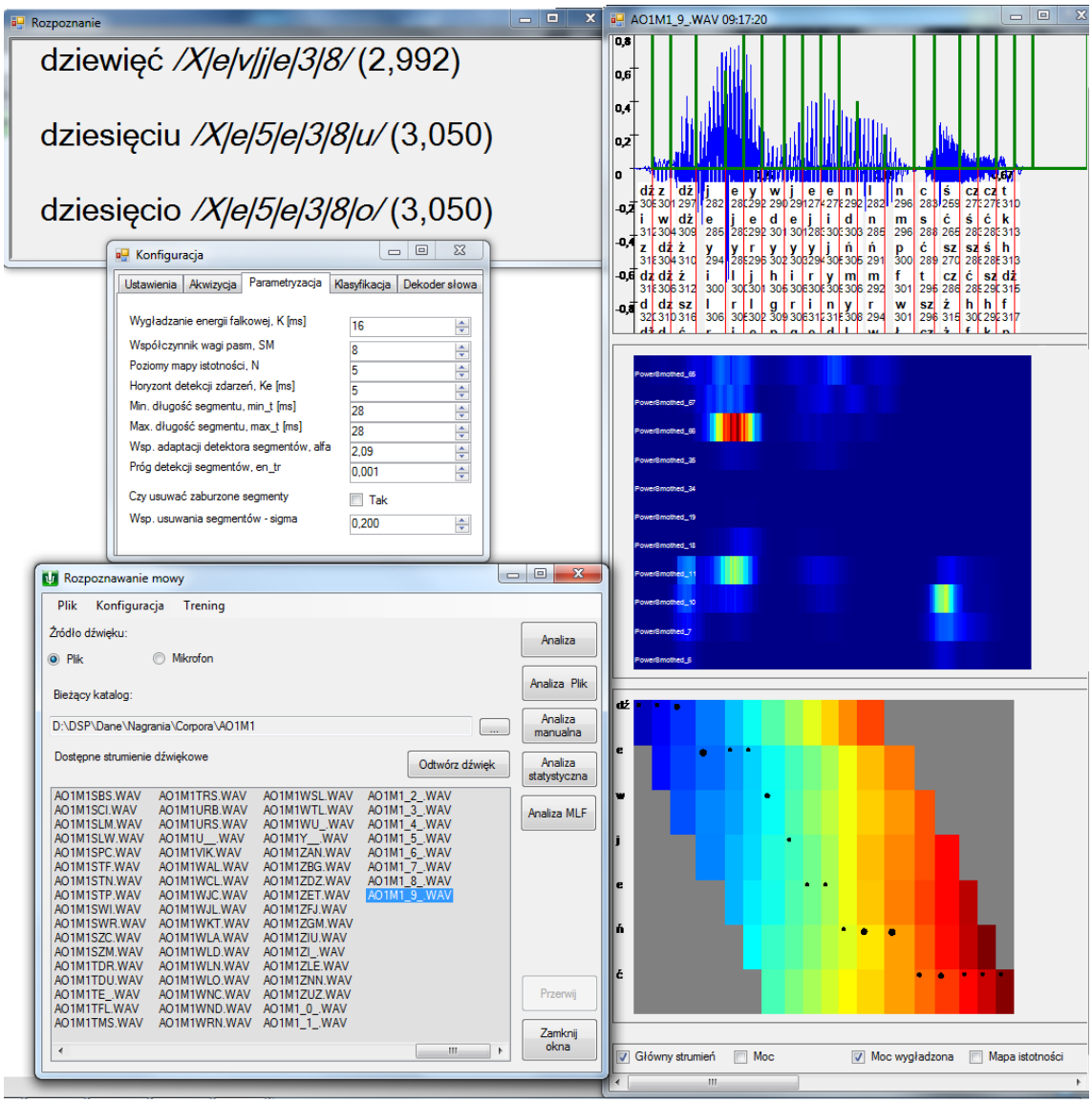

Fig. 1. AGH automatic speech recognition system
Rys. 1. Program AGH do rozpoznawania mowy

2. Databases in natural language processing

Recently in the context of databases there is a hot debate between advocates of the SQL vs. noSQL solutions [7-9]. The defenders of the well know standard SQL systems indicate, that these systems have many features indispensable for databases: the lack of anomalies characteristic for non-normalized data schemes, the support for well known SQL language and ACID transactions, many production ready implementations and a broad availability of supplementary tools (backup, recovery, etc.). On the other hand, the proponents of noSQL (expanded as "not only Structured Query Language") usually advocate for the new systems such as HBase, CouchDB or Cassandra [9] for other reasons. The primary argument is performance – e.g. by compromising the ACID trait or other features of typical SQL databases one can gain dramatic speed improvements. Among the other arguments one can find: a flexible data scheme (e.g. document-like, graph-like), larger scalability and higher availability.

However, in the context of Natural Language Processing (NLP) which is our primary field of research, the situation concerning data models for describing linguistic data as well as data stores for storing such data is different. Since the early Nineties when corpus based techniques started to gain popularity in NLP, researchers started to develop their own solutions for data storage and retrieval. In these years it was quite obvious that the traditional relational databases are not best suited for storing large amounts of text. The direct application of the relational model, where each word in a text (or even a part of a word) is treated as a separate datum would produce a very complex and inefficient storage model. As a result two solutions were developed. In the first the traditional database systems were extended with full text search capabilities especially useful in the context of web search engines [10]. In the second the relational model was replaced by various SGML based languages for describing linguistic phenomena, e.g. the TEI standard [11]. In the late Nineties another important application of SGML – XML started to be broadly adopted not only by researchers, but also by companies. As a result, database vendors started to provide support for XML document storage inside their database management systems, while the researchers transformed their SGML-based languages into derivatives of XML (e.g. the 4th version of TEI which appeared in 2002[1]). We can conclude that a need for non-relational solutions in NLP was observed long before the SQL vs. noSQL debate started.

As a result, many non-relational systems for storing and retrieving linguistic data were developed. First of all, finite state machines and finite state transducers are the primary means for obtaining taggings and lemmas from inflected word forms [12,13]. Instead of looking up a

[1] http://www.tei-c.org/Guidelines/P4/.

word form in a database and fetching its possible interpretations, which due to the word ambiguity phenomenon would require many table joins, a whole dictionary for a given language is transformed into finite state transducer, where each state transition corresponds to a letter in an analysed word form. The result of the analysis is a lemma or lemmas (in the case of ambiguity) of the word plus corresponding taggings. Since many word forms share some of the letter sequences, the information is much compressed and such a system works with a very high performance characteristic for finite state machines.

Another specific data stores used in NLP are engines built to store and efficiently query corpora, not only via key-words, but also via various features of the words. For example, Poliqarp – a corpus engine built by IPI PAN [14] – allows for storing large amounts of text and query them with custom query language by lemmas as well as by a specific part of speech or other morphosyntactic features such as gender, case or number. E.g. a user may ask for text excerpts containing segments with any of the inflected forms of word "kot" ("kot", "kota", "kotu", "kotem", "kocie",...) in singular by issuing the following query: [base=kot & number = sg]. Although it is feasible to obtain the similar result in a relational database with a full text trait, due to the expressiveness of the language it would require to store a lot of supplementary data, making the data model very complex and such a system would have much worse performance.

The last interesting example of non-relational data store used in NLP is the access layer built on top of PolNet – one of the two WordNets built for Polish. The architecture of the POLINT-112-SMS system and the reasons for building a custom query language on top of XML-based data store used to store the WordNet, which the POLINT system interacts with were described [15]. The author argues that a direct integration of WordNet with the system implementation language (Prolog is given as an example) would introduce high coupling between the NLP system and the storage system. This is the same reason why most of the applications use a separate database management system nowadays. But the author also indicates that the adoption of a generic solution such as SQL database, XML store or RDF store with SPARQL interface would yield a system which is less suited for NLP tasks, such as navigation over the WordNet structure or reasoning over the data – the queries would be much more verbose and less meaningful for the developers. So it would be harder to maintain the interoperability between the systems.

One could conclude that NLP tasks are so specific, that the traditional SQL databases are never used with NLP systems. However, such a conclusion will not be valid. The relational model assumed by SQL databases is so generic, that it would be strange if there was a research field within computer science, that could not be covered with it. It is true that a direct application of this model to text seems to be problematic, but NLP is not only about building text corpora: there are many other data types that have to be stored and retrieved

efficiently. Many NLP based applications use various kinds of knowledge sources, such as: inflectional and semantic dictionaries, as well as ontologies. The statistical data (about occurrences of words or their combinations) are also quite important, especially in the context of speech recognition. All these data are better suited for relational databases – the entity model is not as complex as in the case of a whole text, so relational databases still seem to be good choice for such data. What is more – considering text meta-data such as author, genre, number of paragraphs/words and the like, relational databases are still a very good choice.

So it is not a surprise that there are also many applications in NLP that use relational database management systems to store the linguistic data. The intermediate data store for the inflectional dictionary Morfeusz is a relational database [16]. A relational database was used to store the data of the second Polish WordNet – Slowosiec [17]. A relational database stores mappings between the Cyc ontology and its Polish lexicon [18]. So when it comes to building an NLP enabled system from scratch relational database seems to be a good first option for storing the linguistic data.

The decision to use a traditional relational database was also made for our n-gram model. It was obvious that various features of many of the available systems are not needed in this case. Especially the client – server architecture is not needed, since there is only one process modifying the database during its creation and the system would be slowed down by socket-based communication. As a result we decided to choose SQLite database [19], which seems to be well suited for such a setting. However, when we started to process large amounts of text, we have found out, that SQLite does not perform as well as it was supposed to. So we analysed the other options, especially the available noSQL databases. It turned out that Berkeley DB [20] seems to be a good choice: it is used in some NLP systems, e.g. DEBVisDic – a WordNet browsing and editing tool [21] uses Berkeley DB as a primary means for storing the data, it was also reported that this database performs much better than relational databases when it comes to store large amounts of links in a web crawler application [22]. This is why Berkeley DB was tested as an alternative to SQLite.

3. The Polish n-gram model

N-grams are very popular and effective language models in ASR systems [4, 5, 23]. Creating a large vocabulary statistical model of Polish is a difficult task because there are only a few Polish text corpora comparing to English. What is more, Polish is very inflected in contrast to English, what causes difficulties due to the data sparsity. Much more text data must be used for inflected languages than for positional ones to achieve the model of the

same efficiency [24]. Over a million words can be easily expected if all inflections are considered, and a few millions with proper names.

The NKJP Corpus [25] is the main corpus of Polish. However, there are several other large corpora. They are often not annotated and not publicly available. The Rzeczpospolita newspaper articles were used as one of our corpus. Several millions of Wikipedia articles in Polish made another one. Its smallest articles were removed from this corpus to avoid some patterns which could bias the statistics. Several thousand literature books were used as well, as transcripts from the Parliament, its committees and Solidarność meetings.

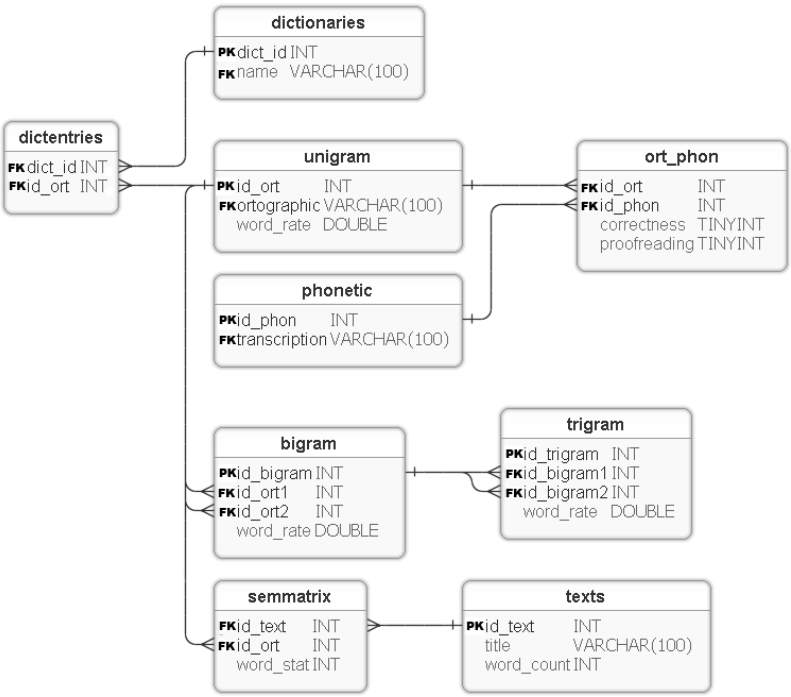

Fig. 2. Structure of the vocabulary database in speech recognition system for Polish
Rys. 2. Struktura bazy słownictwa w systemie rozpoznawania mowy polskiej

Our corpora perplexities are high comparing to English corpora. It is due to inflected nature of Polish and a significant number of proper names. Storing a large n-gram model is another issue to concern. 2- and 3-grams stored as strings would use a lot of disk space. This is why, each 1-gram has an *id*. The 2-grams are stored as two 1-gram *ids* (integer numbers). Each 2-gram has its *id* bigram, so 3-grams are stored as a pair of bigram *ids* (Fig. 2).

4. Comparison of SQLite and Berkeley DB implementations

Collecting n-grams is a time-consuming process. It takes dozens or hundreds of hours to process a few gigabytes corpora. The system based on SQLite was used to collect the n-grams. The schema of the database containing raw n-grams data is different than the

schema used in ASR system. It is much simpler and contains only three tables, separate for unigrams, bigrams and trigrams. The table for unigrams consists of *id* which is an integer primary key, a text field *word* with unicode representation of the word and integer field *count* representing the count of this word. Tables with 2- and 3-grams contain consecutively 2 and 3 fields with sequence of word *ids* and filed with count for this sequence. In the SQLite database none of the fields is marked as UNIQUE nor FOREIGN KEY because it forces the database engine to perform an additional operation to ensure data consistency which results in a decrease of the performance. It is not necessary, because database structure and operations on database are very simple. Berkeley DB contains three similar tables. A table for unigram contains *id*, *word* and its *count*, with the key set on *id*. A table for 2- and 3-grams contains a sequence of words *id*s as a key and its *count* as data.

Table 1

Sizes of the text corpora and their processing times by the implementations of the AGH n-gram model based on SQLite and Berkeley DB

	Rzeczpospolita	Literature
number of words	2 037 414	180 169 048
1-grams	128 366	1 413 296
2-grams	883 632	38 353 763
3-grams	1 429 187	106 363 548
SQLite	2m 25s	21h 22m 36s
Berkeley DB	0h 0m 30s	5h 16m 48s
Time efficiency improvement	**483%**	**405%**

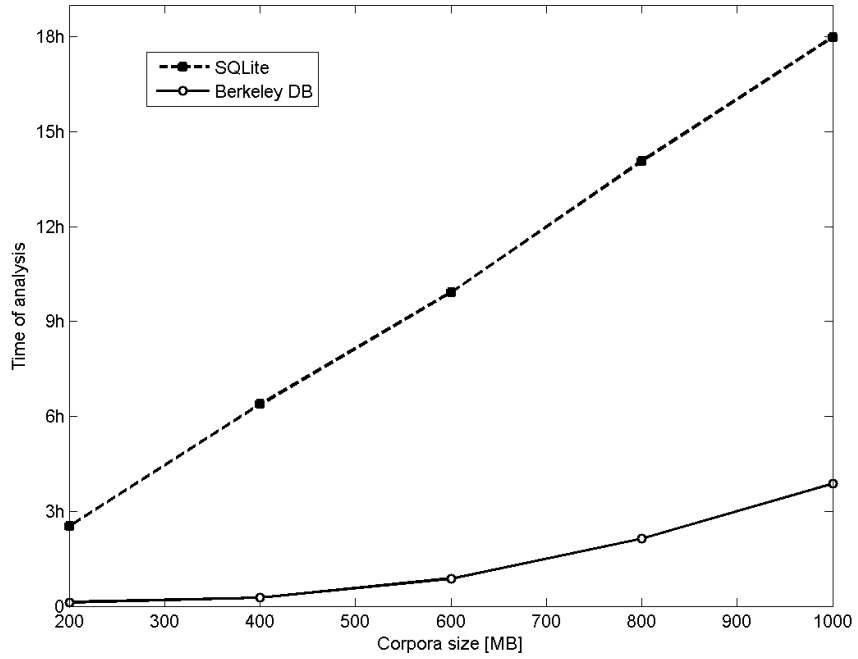

Fig. 3. The time of analysis in function of the corpora size
Rys. 3. Zależność czasu przetwarzania korpusu od jego wielkości

SQLite database has 4 indexes: one on *id* and one on *word* in a unigram table and on *id* sequences in 2- and 3-grams. Berkeley DB database contains additional 'secondary' table associated with unigram table. This table takes the role of the index on *word*, containing two fields: a text as a key and a word *id* as data. It is filled automatically by the database engine.

The cache sizes for both engines were enlarged to increase performance. Journaling and synchronisation between data in memory and data on a disk was turned off in SQLite engine. All SQL statements were in compiled forms. Additionally all tests were performed on SSD disk with IO operations latency below 0.1 ms.

The data filling process is a simple operation. After loading a word from a text file, it is checked if it already exists in the database. If it does, then a count for this word is increased, otherwise it is inserted. *Id* of a recently loaded word is put into a 3 items long buffer and *id* of the oldest word is taken out. This sequence is used to update information about 2- and 3-grams.

5. Conclusions

Several features implemented in SQL databases are not needed in the majority of computational linguistics applications. Some of them are quite time consuming. This is why dedicated database management system like Berkeley DB are used, showing much better efficiency in storing and processing linguistic data. In case of our implementations, the Berkeley DB version is 4-5 times faster then the SQLite version.

This work was supported by MNiSW resources for science as statutory activity.

BIBLIOGRAPHY

1. Ziółko B., Ziółko M.: Przetwarzanie mowy. Wydawnictwa AGH, Kraków 2011.
2. Ziółko M., Gałka J., Ziółko B., Jadczyk T., Skurzok D., Mąsior M.: Automatic Speech Recognition System Dedicated for Polish. Show and tell session, Interspeech, 2011.
3. Dijkstra E. W.: A Note on Two Problems in Connexion with Graphs. Numerische Mathematik, 1959.
4. Ziółko B., Skurzok D.: N-grams model for Polish. Speech and Language Technologies, Book 2, InTech Publisher, 2011.

5. Mąsior M., Ziółko B., Skurzok D., Jadczyk T.: Baza danych słownika języka polskiego ze statystykami słów dla systemu automatycznego rozpoznawania mowy (eng. A database of Polish dictionary with word statistics for automatic speech recognition). Studia Informatica, Vol. 32, No. 2B(97), Wydawnictwo Politechniki Śląskiej, Gliwice 2011, p. 349÷357.
6. Ziółko B., Skurzok D., Michalska M.: Polish n-grams and their correction process. The 4th International Conference on Multimedia and Ubiquitous Engineering, 2010.
7. Leavitt N.: Will NoSQL databases live up to their promise? Computer 43(2), 2010, p. 12÷14.
8. Stonebraker M.: SQL databases v. NoSQL databases. Communications of the ACM, 53(4), 2010, p. 10÷11.
9. Tudorica B., Bucur C.: A comparison between several NoSQL databases with comments and notes. 10th IEEE RoEduNet International Conference, 2011, p. 1÷5.
10. Pinkerton B.: Finding what people want: Experiences with the WebCrawler. The Second International World Wide Web Conference, Vol. 94, Chicago 1994, p. 17÷20.
11. Ide N., Veronis J.: Text encoding initiative: Background and contexts. Kluwer Academic Publishing, 1995.
12. Daciuk J.: Incremental Construction of Finite-State Automata and Transducers, and their Use in the Natural Language Processing, 1998.
13. Woliński M.: Morfeusz – a practical tool for the morphological analysis of Polish. Intelligent information processing and web mining, 2006, p. 511÷520.
14. Przepiórkowski A.: Korpus IPI PAN. Wersja wstępna. Instytut Podstaw Informatyki PAN, 2004.
15. Kubis M.: An Access Layer to PolNet-Polish WordNet. Human Language Technology. Challenges for Computer Science and Linguistics, 2011, p. 444÷455.
16. Woliński M.: A Relational Model of Polish Inflection in Grammatical Dictionary of Polish. Human Language Technology. Challenges of the Information Society, 2009, p. 96÷106.
17. Piasecki M., Szpakowicz S., Broda B.: A Wordnet from the Ground Up. Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław 2009.
18. Pohl A.: The Semi-automatic Construction of the Polish Cyc Lexicon. Investigationes Linguisticae, 21, 2010.
19. Owens M.: The denitive guide to SQLite. A Press, 2006.
20. Olson M., Bostic K., Seltzer M.: Berkeley DB. Proceedings of the FREENIX Track: 1999 USENIX Annual Technical Conference, 1999, p. 183÷192.

21. Horak A., Pala K., Rambousek A., Povolny M.: DEBVisDic – First Version of New Client-Server Wordnet Browsing and Editing Tool. Proceedings of the Third International WordNet Conference, 2006, p. 325÷328.
22. Dorosz K.: Usage of Dedicated Data Structures for URL Databases in a Large-scale Crawling. Computer Science, 10, 2009, p. 7÷17.
23. Hirsimaki T., Pylkkonen J., Kurimo M.: Importance of high-order n-gram models in morph-based speech recognition. IEEE Transactions on Audio, Speech and Language Processing ,17(4), 2009, p. 724÷732.
24. Whittaker E., Woodland P.: Language modelling for Russian and English using words and classes, Computer Speech and Language, 17, 2003, p. 87÷104.
25. Przepiórkowski A., Górski R. L., Łaziński M., Pęzik P.: Recent Developments in the National Corpus of Polish. Proceedings of LREC, 2010.

Wpłynęło do Redakcji 5 stycznia 2012 r.

Omówienie

W artykule przedstawiono zagadnienia dotyczące stosowania w przetwarzaniu języka naturalnego różnego typu baz danych. W szczególności omówiono niewielką przydatność relacyjnych i transakcyjnych baz danych, ze względu na mniejszą wydajność czasową, będącą efektem działania mechanizmów niemających zastosowania w przetwarzaniu i gromadzeniu danych językowych.

Przedstawiono przykład zastosowań w systemie rozpoznawania mowy AGH (rys.1). Jego częścią jest frekwencyjny słownik języka polskiego, rozszerzony do modelu n-gramowego, z planem wprowadzenia dodatkowych funkcji semantycznych (rys. 2). Model został rozbudowywany na podstawie automatycznej analizy plików tekstowych, do 10 GB danych, z możliwością ręcznych korekt automatycznie wykrywanych, problematycznych rekordów. Rozpoznawanie mowy jest skomplikowanym zadaniem obliczeniowym, którego duża część musi być wykonywana w czasie rzeczywistym. W związku z tym, w ciągu ostatnich miesięcy dokonano profilowania wydajności systemu, usprawniając znacząco jego działanie.

Kluczowym usprawnieniem była reimplementacja modelu w bazie danych Berkeley DB, rezygnując z wcześniejszej implementacji SQLite. Korzystając z doświadczeń budowania słownika polskiego systemu rozpoznawania mowy, przedstawiono przykład ukazujący znaczącą przewagę implementacji wykonanej w Berkeley DB nad implementacją SQLite w sensie wydajności czasowej (tab. 1).

Addresses

Dawid SKURZOK: Akademia Górniczo-Hutnicza, Katedra Elektroniki, al. Mickiewicza 30, 30-059 Kraków, Polska, skurzok@agh.edu.pl.

Bartosz ZIÓŁKO: Akademia Górniczo-Hutnicza, Katedra Elektroniki, al. Mickiewicza 30, 30-059 Kraków, Polska, bziolko@agh.edu.pl.

Aleksander POHL: Uniwersytet Jagielloński, Katedra Lingwistyki Komputerowej, ul. Łojasiewicza 4, 30-348 Kraków, Polska, apohllo@o2.pl.

Tomasz JADCZYK: Akademia Górniczo-Hutnicza, Katedra Elektroniki, al. Mickiewicza 30, 30-059 Kraków, Polska, jadczyk@agh.edu.pl.

Mariusz MĄSIOR: Akademia Górniczo-Hutnicza, Katedra Elektroniki, al. Mickiewicza 30, 30-059 Kraków, Polska, masior@agh.edu.pl.