Volume 19 Number 4

ACTA CYBERNETICA

Editor-in-Chief: János Csirik (Hungary)

Managing Editor: Csanád Imreh (Hungary)

Assistant to the Managing Editor: Attila Tanács (Hungary)

Associate Editors:

Luca Aceto (Iceland)
Mátyás Arató (Hungary)
Stephen L. Bloom (USA)
Hans L. Bodlaender (The Netherlands)
Lothar Budach (Germany)
Horst Bunke (Switzerland)
Bruno Courcelle (France)
Tibor Csendes (Hungary)
János Demetrovics (Hungary)
Bálint Dömölki (Hungary)
Zoltán Ésik (Hungary)
Zoltán Fülöp (Hungary)
Ferenc Gécseg (Hungary)
Jozef Gruska (Slovakia)
Tibor Gyimóthy (Hungary)
Helmut Jürgensen (Canada)
Zoltan Kato (Hungary)
Alice Kelemenová (Czech Republic)
László Lovász (Hungary)
Gheorghe Păun (Romania)
András Prékopa (Hungary)
Arto Salomaa (Finland)
László Varga (Hungary)
Heiko Vogler (Germany)
Gerhard J. Woeginger (The Netherlands)

Szeged, 2010

ACTA CYBERNETICA

Information for authors. Acta Cybernetica publishes only original papers in the field of Computer Science. Manuscripts must be written in good English. Contributions are accepted for review with the understanding that the same work has not been published elsewhere. Papers previously published in conference proceedings, digests, preprints are eligible for consideration provided that the author informs the Editor at the time of submission and that the papers have undergone substantial revision. If authors have used their own previously published material as a basis for a new submission, they are required to cite the previous work(s) and very clearly indicate how the new submission offers substantively novel or different contributions beyond those of the previously published work(s). Each submission is peer-reviewed by at least two referees. The length of the review process depends on many factors such as the availability of an Editor and the time it takes to locate qualified reviewers. Usually, a review process takes 6 months to be completed. There are no page charges. Fifty reprints are supplied for each article published.

Manuscript Formatting Requirements. All submissions must include a title page with the following elements:

- title of the paper
- author name(s) and affiliation
- name, address and email of the corresponding author
- An abstract clearly stating the nature and significance of the paper. Abstracts must not include mathematical expressions or bibliographic references.

References should appear in a separate bibliography at the end of the paper, with items in alphabetical order referred to by numerals in square brackets. Please prepare your submission as one single PostScript or PDF file including all elements of the manuscript (title page, main text, illustrations, bibliography, etc.). Manuscripts must be submitted by email as a single attachment to either the most competent Editor, the Managing Editor, or the Editor-in-Chief. In addition, your email has to contain the information appearing on the title page as plain ASCII text. When your paper is accepted for publication, you will be asked to send the complete electronic version of your manuscript to the Managing Editor. For technical reasons we can only accept files in LaTeX format.

Subscription Information. Acta Cybernetica is published by the Institute of Informatics, University of Szeged, Hungary. Each volume consists of four issues, two issues are published in a calendar year. Subscription rates for one issue are as follows: 5000 Ft within Hungary, €40 outside Hungary. Special rates for distributors and bulk orders are available upon request from the publisher. Printed issues are delivered by surface mail in Europe, and by air mail to overseas countries. Claims for missing issues are accepted within six months from the publication date. Please address all requests to:

Acta Cybernetica, Institute of Informatics, University of Szeged
P.O. Box 652, H-6701 Szeged, Hungary
Tel: +36 62 546 396, Fax: +36 62 546 397, Email: acta@inf.u-szeged.hu

Web access. The above informations along with the contents of past issues are available at the Acta Cybernetica homepage http://www.inf.u-szeged.hu/actacybernetica/ .

EDITORIAL BOARD

Editor-in-Chief: **János Csirik**
Department of Computer Algorithms
and Artificial Intelligence
University of Szeged
Szeged, Hungary
csirik@inf.u-szeged.hu

Managing Editor: **Csanád Imreh**
Department of Computer Algorithms
and Artificial Intelligence
University of Szeged
Szeged, Hungary
cimreh@inf.u-szeged.hu

Assistant to the Managing Editor:

Attila Tanács
Department of Image Processing
and Computer Graphics
University of Szeged, Szeged, Hungary
tanacs@inf.u-szeged.hu

Associate Editors:

Luca Aceto
School of Computer Science
Reykjavík University
Reykjavík, Iceland
luca@ru.is

Mátyás Arató
Faculty of Informatics
University of Debrecen
Debrecen, Hungary
arato@inf.unideb.hu

Stephen L. Bloom
Computer Science Department
Stevens Institute of Technology
New Jersey, USA
bloom@cs.stevens-tech.edu

Hans L. Bodlaender
Institute of Information and
Computing Sciences
Utrecht University
Utrect, The Netherlands
hansb@cs.uu.nl

Lothar Budach
Department of Computer Science
University of Potsdam
Potsdam, Germany
lbudach@haiti.cs.uni-potsdam.de

Horst Bunke
Institute of Computer Science and
Applied Mathematics
University of Bern
Bern, Switzerland
bunke@iam.unibe.ch

Bruno Courcelle
LaBRI
Talence Cedex, France
courcell@labri.u-bordeaux.fr

Tibor Csendes
Department of Applied Informatics
University of Szeged
Szeged, Hungary
csendes@inf.u-szeged.hu

János Demetrovics
MTA SZTAKI
Budapest, Hungary
demetrovics@sztaki.hu

Bálint Dömölki
IQSOFT
Budapest, Hungary
domolki@iqsoft.hu

Zoltán Ésik
Department of Foundations of
Computer Science
University of Szeged
Szeged, Hungary
ze@inf.u-szeged.hu

Zoltán Fülöp
Department of Foundations of
Computer Science
University of Szeged
Szeged, Hungary
fulop@inf.u-szeged.hu

Ferenc Gécseg
Department of Computer Algorithms
and Artificial Intelligence
University of Szeged
Szeged, Hungary
gecseg@inf.u-szeged.hu

Jozef Gruska
Institute of Informatics/Mathematics
Slovak Academy of Science
Bratislava, Slovakia
gruska@savba.sk

Tibor Gyimóthy
Department of Software Engineering
University of Szeged
Szeged, Hungary
gyimothy@inf.u-szeged.hu

Helmut Jürgensen
Department of Computer Science
Middlesex College
The University of Western Ontario
London, Canada
helmut@csd.uwo.ca

Zoltan Kato
Department of Image Processing
and Computer Graphics
Szeged, Hungary
kato@inf.u-szeged.hu

Alice Kelemenová
Institute of Computer Science
Silesian University at Opava
Opava, Czech Republic
Alica.Kelemenova@fpf.slu.cz

László Lovász
Department of Computer Science
Eötvös Loránd University
Budapest, Hungary
lovasz@cs.elte.hu

Gheorghe Păun
Institute of Mathematics of the
Romanian Academy
Bucharest, Romania
George.Paun@imar.ro

András Prékopa
Department of Operations Research
Eötvös Loránd University
Budapest, Hungary
prekopa@cs.elte.hu

Arto Salomaa
Department of Mathematics
University of Turku
Turku, Finland
asalomaa@utu.fi

László Varga
Department of Software Technology
and Methodology
Eötvös Loránd University
Budapest, Hungary
varga@ludens.elte.hu

Heiko Vogler
Department of Computer Science
Dresden University of Technology
Dresden, Germany
Heiko.Vogler@tu-dresden.de

Gerhard J. Woeginger
Department of Mathematics and
Computer Science
Eindhoven University of Technology
Eindhoven, The Netherlands
gwoegi@win.tue.nl

Conference on Hungarian Computational Linguistics

Guest Editor:

Zoltán Alexin

Department of Software Engineering
University of Szeged
Szeged, Hungary
alexin@inf.u-szeged.hu

Preface

This issue of *Acta Cybernetica* contains three papers whose preliminary versions appeared in Hungarian language in the proceedings of the Sixth Conference on Hungarian Computational Linguistics.

The conference was held in Szeged on December 3–4, 2009 and its aim was to provide a forum for researchers working on Hungarian computational linguistics and speech technology, see `http://www.inf.u-szeged.hu/mszny2009/`.

After the conference, the authors were invited to submit completed versions of their papers to *Acta Cybernetica.* All submitted papers were then subjected to the normal double refereeing process of the journal. Altogether three manuscripts were submitted, all of them have been finally accepted, one of them after major revision. I would like to thank the authors and the referees for their help in the preparation of this issue.

The Seventh Conference on Hungarian Computational Linguistics was held also in Szeged on December 2–3, 2010.

Zoltán Alexin

Guest Editor

Information Extraction from Wikipedia Using Pattern Learning

Márton Miháltz*

Abstract

In this paper we present solutions for the crucial task of extracting structured information from massive free-text resources, such as Wikipedia, for the sake of semantic databases serving upcoming Semantic Web technologies. We demonstrate both a verb frame-based approach using deep natural language processing techniques with extraction patterns developed by human knowledge experts and machine learning methods using shallow linguistic processing. We also propose a method for learning verb frame-based extraction patterns automatically from labeled data. We show that labeled training data can be produced with only minimal human effort by utilizing existing semantic resources and the special characteristics of Wikipedia. Custom solutions for named entity recognition are also possible in this scenario. We present evaluation and comparison of the different approaches for several different relations.

Keywords: natural language processing, information extraction, machine learning

1 Introduction

Today, in the world of the knowledge-hungry applications, there is an increased need for mass quantities of structured information that can enable searching technologies that go beyond simple character-based solutions. The construction of such Semantic Web technologies requires efforts to be made in the direction of automatically extracting semantic relations and properties of entities from available online textual resources. The work described here was carried out within the framework of the iGlue project[1], which aims to create a uniformly treated, semantically interlinked database of named entities such as persons, geographical names, institutions etc.

Recently, much attention was given to exploiting available large-scale online resources for information extraction, in particular, to using Wikipedia, the free-

*Péter Pázmány Catholic University, E-mail: `mihaltz@digitus.itk.ppke.hu`

[1] `http://iglue.com`

content online encyclopedia[2] ([3], [8], [14], [23], [26], [25]). The reasons we adopted Wikipedia for information extraction are that it has considerable coverage, the articles have good text quality making it possible to use state-of-the-art natural language processing algorithms, and finally because the redundancy between the structured and unstructured (free text) parts provides possibilities for generating annotated training data. Other special properties of Wikipedia pages (uniform encyclopedic structure, internal links, redirection pages etc.) provide further possibilities for enhancing information extraction.

Our research focuses on the development of a large-scale, reliable information extraction system, which mines for structured textual information, such as properties and relationships of entities, from the free text sections of the English pages of Wikipedia. Our system is therefore able to obtain information that is inaccessible from the structured sections (infoboxes, tables, category labels etc.) of the article pages. To leverage the task of processing free text, our system relies on natural language processing tools such as syntactic parsing, named entity recognition and coreference resolution.

Our system uses templates, or frames, consisting of slots that correspond to the entities that are in the given relation. The frame slots can be filled using extraction patterns, which are described using verb frame structures that have elements corresponding to the argument and modifier noun phrases of the main verbs in the input sentences.

In order to assess the performance of the linguistic analysis and to explore the potentials of the verb frame-based information extraction approach, we first created a system that uses hand-crafted extraction patterns. This system also served as a baseline for comparison to further research, in which we investigated a method to automatically learn extraction patterns. We also experimented with approaches that use less sophisticated linguistic analysis and machine learning.

The rest of this paper is organized as follows. In the next Section, we give an overview of related work. In Section 3, we describe the baseline system that uses manually crafted extraction patterns, the details of the linguistic analysis and the problems we encountered and solved. In section 4, we describe an alternative approach, using supervised machine learning. In section 5, we describe our pattern learning algorithm and compare its performance against both the baseline system and the machine learning approaches. Section 6 summarizes and discusses our results.

2 Related Work

Learning extraction patterns. Several authors have explored the possibilities of reducing the burden of manually authoring extraction patterns. Riloff [16] uses manual annotation only to categorize documents as relevant and irrelevant to generate extraction patterns. Another line of research was initiated by DIPRE [6], which relies on the redundancy of information on the web. Facts being expressed

[2] http://en.wikipedia.org/wiki/

in different forms enable bootstrapping from a small number of manually supplied seeds. The Snowball system [2] uses term vectors to represent the contexts containing the seed facts, which are clustered to generate new patterns. The patterns are given confidence scores, and the best performing ones are used to mine for new facts to retrain the system. Agichten et al. [1] improved Snowball by using sparse Markov transducers to represent contexts, enabling the coding of word order. StatSnowball [26] also built on Snowball but used Markov logic networks, supplying a measure of pattern confidence in a more natural way. KnowItAll [11] and LEILA [19] also applied bootstrapping, the latter incorporating both positive and negative examples as seeds.

Banko et al. [10] introduced Open Information Extraction, a scenario where the relations are not known in advance and the corpora involved are massive and heterogeneous. Textrunner, the first such system [10] uses a Naive Bayes classifier to predict whether tokens between two entities indicate a relationship or not. The authors continued this work [4] with a system that uses conditional random fields (CRF), trained by self-supervision: a small number of relation-independent heuristics are applied to generate labelled (positive and negative) examples.

Using Wikipedia. A number of projects have exploited Wikipedia for information extraction. DBpedia [3] and Yago [22] extract information from the structured parts (infoboxes, categories, lists etc.) Suchanek et al. [22] construct an ontology by mapping the extracted entities to WordNet [12].

Ruiz-casado et al. [17] proposed a method for automatic extraction and generalization of extraction patterns for semantic relationships (hyperonymy/hyponymy, holonymy/meronymy) from Simple English Wikipedia, using and extending WordNet. The extraction patterns are generalized with an algorithm using minimum edit distance, using a representation resembling the one proposed by this paper (see Section 5).

Culotta et al. [8] present a model to integrate information extraction and data mining, demonstrated on Wikipedia articles. They apply CRFs, using both contextual and relational features. Nguyen et al. [14] demonstrate relation extraction from Wikipedia article free texts using dependency tree mining and supervised machine learning with SVM classifiers. Similar to our system, they use a custom coreference resolution algorithm exploiting special characteristics of Wikipedia pages. They also use a custom named-entity type recognition relying on supervised classification of wikipedia pages corresponding to the entities in the relations.

Suchanek et al. [21] present PORE, an algorithm for situations involving only positive and unlabeled examples, applied to semi-automatic IE from free text in Wikipedia articles. The algorithm is based on an SVM classifier, and uses bootstrapping, strong negative identification and transductive inference. Just as in our approach, positive training examples are generated from infobox data. Entities are characterized by features from their respective pages.

The Kylin/KOG project [23] is a complex Open IE system that uses Wikipedia infoboxes to generate training data for document and sentence classifiers and CRF relation extractors that are run on the free text sections of the articles. Training data sparseness problem is solved by generating an ontology of Wikipedia infobox schemata and using the inheritance of relations. Training data is also augmented by Google searches for the Wikipedia page titles and finding sentences containing the infobox data. Extraction is also extended to web search results, increasing recall and precision.

The SOFIE system [20] presents an integrated approach involving 1st-order predicate logic representation and logical reasoning to solve pattern extraction, entity disambiguation and consistency checking together in one unified model. The system relies on knowledge in the Yago ontology and is able to extend it with information extracted from Wikipedia and web searches with high precision.

Weld et al. [24] propose a system for unsupervised relation extraction, the task of automatically discovering interesting relations between entities. Sentences in Wikipedia pages containing candidate entities (anchor texts linked to other Wikipedia pages) are clustered in two steps, using features from dependency parsing for high precision and additional surface patterns obtained from web searches for increasing coverage.

Using deep NLP methods. Several papers have proposed to use features from deep linguistic processing (parsing) for information extraction. Yan et al. [25] do relation extraction using a tree kernel defined over shallow parse tree representations of sentences. Culotta et al. [9] continue this work by defining a tree kernel over relation instances consisting of the smallest dependency tree containing the two entities of the relation. Bunescu et al. [7] improve this by using the shortest path between the entities in the dependency graph, while Nguyen et al. [14] use dependency subtrees. Yan et al [24] use the subpaths in the shortest path connecting the two entities in the dependency trees as features for their clustering algorithm.

Our system also uses a deep parser to leverage syntactic structures from the input text, however, it differs from these approaches in the respect that instead of dependency trees, our system uses phrase structures and verb frames derived from these (see Section 3).

3 Information Extraction with Verb Frames

The domain for the development of our baseline system was the *studies* relation, where a person's Wikipedia article is searched for fillers for the following slots: the name of the educational institution where the person studied; starting and ending times of studies; time of obtaining qualification; name of obtained qualification (i. e. type of degree: B.A., M.A., Ph.D etc.), and field of study. For example, the sentence

> In 1977, he graduated magna cum laude from Harvard University with a B.A. in mathematics.

would produce the following fillers for the studies template:

```
School name: Harvard University
Begin studies date: -
End studies date: -
Qualification date: 1977
Qualification: B.A.
Field(s): mathematics
```

3.1 Corpus Construction

The base of our Wikipedia corpus was the June 2008 version of the static dump of English Wikipedia pages[3], containing about 2.4 million articles. We used simple heuristics, such as checking for information about birth and death dates, etc. in order to identify about 100,000 autobiographical articles with high accuracy. These articles were processed to separate raw text content (containing only paragraph boundary information) from formatting and other page elements. Meta-information, such as page title and title variants (obtained by processing redirection page links), category labels, hyperlinks within the text etc. were retained in separate files to facilitate later processing.

3.2 Linguistic Analysis and Pattern Matching

The raw text paragraphs were processed by LingPipe's sentence segmentation tool[4], followed by parsing with Enju [18], an efficient and wide-coverage English parser using a probabilistic Head-Driven Phrase Structure (HPSG) grammar. Enju is capable of producing both phrase structures and predicate-argument structures. We identified the verb frame structures in the parses that would be matched by the IE patterns. The noun phrases in the verb frames were processed by special named entity recognizers. We will now describe these two steps in more detail.

In the parser's output, we first identified verb phrases (clauses) inside the sentence that carried relevant information: coordinate clauses, relative clauses, some prepositional phrases with a verb phrase (VP) complement having "before" or "after" for prepositions etc. We skipped VPs having negated or non-declarative main verbs.

Special care had to be taken when processing the noun phrases. The top-level NPs in Enjus grammar can have complex internal structures covering many terminals, of which not all would be necessary for information extraction (e.g. a PP at the end, as in "Juilliard School in New York City".) We therefore selected terminals in the noun phrases up to and including the head, plus any following tokens participating in structures analyzed as apposition or possession, both of which could be legitimate parts of the names (e.g. "Montana School of Mines", "December, 1988".) Determiners, possessive pronouns, prepositions etc. were removed from the front of the NPs. For each selected terminal in the noun phrases, we recorded its surface form, base form, part-of-speech tag and sentence position.

[3] `http://static.wikipedia.org/downloads/2008-06/`

[4] `http://alias-i.com/lingpipe/`

Coordinated phrases were split and we produced all possible combinations with the other sentence constituents. For example, the following complex sentence,

> After receiving a Bachelor's Degree in mathematics and physics at the University of Michigan, he went on to obtain a Ph.D. in electrical engineering at Harvard in 1998.

would produce the following structures after parsing and processing:

```
((Verb, "receive"), (Subj, "he"),
 (Obj, "Bachelor's Degree"), (PP-in, "mathematics"))
((Verb, "receive"), (Subj, "he"),
 (Obj, "Bachelor's Degree"), (PP-in, "physics"))
((Verb, "go on"), (Subj, "he"), (Verb2, "obtain"),
 (Obj2, "Ph.D."), (PP-in2, "electrical engineering"),
 (PP-at2, "Harvard"), (PP-in2, "1998"))
```

The recognition of frame slots (such as name of school, field of study etc.) is based on syntactic and semantic constraints. The syntactic constraints match the grammatical role of the given NP in the sentence. The semantic constraints ensure that only the right types of entities are matched. For instance, an extraction pattern would look like the following (in simplified form):

```
Subj(PERSON)+V('attain')+Obj(DEGREE)+PP-in(SCHOOL)+PP-in(DATE)
```

This means that the main verb must be (a form of) "attain", the subject NP must be of type PERSON, the object NP must be of type DEGREE, and prepositional phrases headed by "in" either designate the SCHOOL or the DATE slots of the relation. To check the semantic constraints, we applied simple, custom-made named-entity recognition using regular expressions and/or lexicons that were used to check the heads of the NPs. We had to employ custom NER solutions because most freely available NER-taggers can only recognize standard general categories such as *person, location, organization* etc. which are insufficient for the *studies* relation. We used several online sources, dictionaries, thesauri etc. to compile the lexicons as extensively as possible. The fields of study lexicon, for example, contains about 2,100 entries, while the educational institution names lexicon comprises more than 34,000 items.

In order to ensure that we were extracting information about the person in focus, we checked for references to the article title person in the input text. We checked for occurrences of the page title, its name variants obtained from the redirection links, its substrings (to account for further name variations), or personal pronouns.

The baseline system used about 20 extraction patterns that were constructed by human knowledge experts after several person days of time was spent on studying a sample of articles in the domain. We created a human-annotated development corpus in order to aid the construction of the baseline system. 200 "person" articles were randomly chosen and the relevant information slots were manually tagged by

2 annotators, while a 3rd annotator checked the results. We periodically performed automatic tests against this gold standard during development. As a design principle, the system was optimized for precision in the precision-recall tradeoff, since reliability was declared to be crucial, while a desired recall of at least 40% would still yield considerable amount of data given Wikipedia's vast coverage.

For the evaluation of the final baseline system, further 100 randomly chosen Wikipedia person articles were prepared by the human annotators. We calculated precision and recall of recognition of frame slots against this set (see Table 1).

Table 1: Evaluation of information extraction with manually developed patterns in the "studies" domain

Precision	Recall	F-measure
94.22%	60.33%	73.56%

3.3 Special Problems

During the development of the baseline system, we encountered several problems, which involved finding workarounds for several re-occurring errors in the output of Enju parser.

The first is the well-known prepositional phrase-attachment problem, when the parser attached the same type of PPs inconsistently to either the main VP of the clause or the final NP preceding the PP. For this reason, we ignored the structural relations proposed by the parser for the PPs, and used our own heuristics in order to identify the required dependencies (for example, in the case of time adverb PPs, rules based on relative sentential order.)

A second, very common problem was presented by the parser's failure to correctly identify the boundaries of multi-word proper names and other named entities, resulting in incorrect parses. To overcome this, we tried to recognize as many named entities as possible before parsing, using special characters to merge them into single tokens. These would be treated by Enju as single-word nouns in the input sentence thus producing the correct syntactic analyses. We used several simple but reliable methods that would not require the complex resources of a dedicated named-entity recognizer run on tens of thousands of documents. One such method was to search the original raw text for hyper-links (referring to other Wikipedia pages). If anchor texts within such links contained multi-word proper names, these were marked, as they would refer to proper name entities with a high probability. We also generated a list of potential names from all the multi-word capitalized page titles of all the Wikipedia pages in our static dump, and tried to recognize and mark these in the input articles.

Another, similar problem occurred when the parser incorrectly analyzed several common named entity types containing commas, such as dates ("April, 1996"), or school names ("University of Berkeley, California") as coordinated NPs. We applied recognition and marking before parsing for these categories as well. The dates were recognized by regulars expressions, while the recognition of school names used a combination of regular expressions and special lexicons, using the above-mentioned educational institution names directory in conjunction with 2.3 million geographical names.

4 Information Extraction Using Machine Learning

We conducted experiments with supervised machine learning methods, depending on shallow, less resource-intensive linguistic analysis, and compared it to the baseline method. The domain was the same as the baseline system's (*studies*). Training instances were generated automatically, by looking up instances of the Wikipedia "alumni" category labels, entries of the academic degree names lexicon described above, and simple regular expressions for dates in the articles' texts. These instances were then hand-checked, yielding a corpus of about 2000 annotated training examples. However, this method provided us with annotations for only 3 of the 6 slots in the *studies* scenario (name of educational institution, qualification name, year of qualification), since there was no redundant Wikipedia information available to automatically identify the other entities in the original relation presented in Section 3.

The training documents were only processed by sentence segmentation, tokenization and part-of-speech tagging. We trained the Mallet maximum entropy classifier[5]. The learning features were n-grams (n=1,2,3 before and n=1,2 after) and the base forms of the nearest verbs in the sentence.

We used the human-annotated evaluation set described in Section 3 to evaluate the performance of the classifier and to compare it with the baseline method that used deep parsing, special named entity recognizers and hand-crafted extraction patterns. We measured precision and recall for each of the 3 slots that were extracted. We were also interested in how the two methods complemented each other, so we also evaluated the union and the intersection of the results coming from the two systems (Table 2).

The machine learning approach came closest in precision to the pattern-based approach in the recognition of institution and academic degree names, while recall was significantly lower for both. The intersection of the two methods yielded 100% precision at a cost of very low recall. The union of the two methods, however, in the case of institution and degree names did not degrade the precision of the machine learning approach, but brought a significant increase in recall even in comparison to the better-performing pattern-based approach. This suggests that the two approaches tend to complement each-other, each working well on different types of instances. The resulting hybrid system could be used well for practical applications, since its precision remained above the critical 90% threshold while its recall was improved.

[5] http://mallet.cs.umass.edu

Table 2: Comparison of pattern-based (PB) and machine learning (ML) methods for 3 slots of the "studies" frame (precision, recall, F-measure) (decimals were retained to conserve space)

	Institution			*Date*			*Qualification*		
	P	R	F1	P	R	F1	P	R	F1
PB	92%	67%	78%	100%	55%	71%	94%	63%	75%
ML	91%	41%	57%	85%	47%	61%	92%	44%	60%
Union	91%	76%	83%	90%	75%	82%	94%	81%	87%
Intersection	100%	11	20%	100%	5%	10%	100%	2%	4%

The lower precision of qualification dates in the machine learning results could be explained by the fact that it had no knowledge that dates could participate in three different roles in these contexts (beginning and ending time of studies, time of obtaining qualification), in contrast to the baseline system.

5 Learning Extraction Patterns

We intended to develop a system that would be able to learn extraction patterns automatically from pre-annotated example sentences. Our goal was to create a general framework that could be adapted to new extraction domains in the shortest time possible, utilizing minimal human effort. A human knowledge expert would only be required to check, and if necessary, edit the extraction patterns that were automatically generated by the system. The annotator could also decide to mark certain patterns as negative, indicating constructions that are similar to real patterns, but produce incorrect results (for example, in the above-mentioned *studies* domain, one would want to exclude sentences about receiving honorary academic degrees, as opposed to real academic degrees.)

5.1 Generating Training Instances

In order to reduce the effort of creating annotated training instances, we relied on the assumption that a certain degree of redundancy could be expected between the structured and free text contents of Wikipedia. We used the results of the Yago project [22] in order to gain access to the information in the structured sections (tables, category labels) of the English Wikipedia pages. Part of Yago's knowledge is available in the form of binary relations between Wikipedia entities (Wikipedia entries.) We harvested free-text training examples automatically by looking up sentences contained in the articles about the 1st arguments that contained references to the 2nd arguments of the given Yago relations.

We used the *awards* relation for the development of the extraction pattern learning system, which holds between two entities (awarded person, award name).

A baseline system with manually developed extraction patterns, similar to the one described in Section 3 had been also developed for this domain and was available for comparison.

We used Yago's *hasWonPrize* relation, which holds between *person* and *award* Wikipedia entities. Looking up the award names in the persons' pages produced about 16,000 potential training sentences. Since the *hasWonPrize* relation was noisy not only award names, but titles of award-winning works like film and album titles were present as 2nd arguments, we used a lexicon of about 7,400 award names to filter out the misleading sentences, leaving about 13,000 for further processing.

The sentences were processed by Enju parser and the verb frame extraction process described in Section 3. We identified and annotated the two Yago arguments inside the identified constituent noun phrases. In order to recognize the 1st argument (person name), we applied simple rule-based coreference resolution. We looked for the page title, its variant extracted from the 1st sentence in the leading paragraph, or any token-substrings of these to cover other name variations. If none of these could be recognized under any NP, we looked for personal pronouns, taking into account the gender of the person corresponding to the page title. We counted the number of feminine and masculine 3rd person singular personal pronouns and assumed the gender of the title person to correspond with the gender having the higher count.

After the annotation of the Yago arguments, we excluded sentences that did not contain entities for both slots, leaving about 11,000 training sentences.

In order to facilitate the generalization of extraction patterns from the training sentences, we also annotated several named entity types that could be recognized easily using regular expressions (ordinals, cardinals, various date formats, month names, years, and numbers.)

5.2 Generating Extraction Patterns

Our goal was to generate extraction patterns from the annotated training sentences taking the following two criteria into account: 1) the number of generated patterns should be as low as possible to support human post-processing, but at the same time these patterns should cover as many as possible of the original training sentences, 2) the generated patterns should have a uniform syntax, and should be easy to read and edit by humans. Manual editing should mainly constitute deleting unapproved patterns or pattens elements.

The outline of our proposed pattern learning algorithm is the following:

1. Converting training instances to patterns

2. Creating pattern classes

3. Identification of marker tokens

4. Creating generalized patterns from the pattern classes

In the following, we describe each step in detail together with the results of our experiment with its application in the *awards* domain.

1. Converting training instances to patterns. All of the training sentences, annotated with syntactic constituents, Yago relation arguments and simple named entity categories were converted to patterns. A pattern is a list of ordered pairs *(G, S)*, where *G* is the name of a grammatical role (*Verb, Subj, Obj, ObjII* and *PP-xx*, *xx* being an English preposition), while *S* is a list of tokens inside the constituent having label *G*. *S* may consist of either meta-tokens (Yago argument or named entity labels), or simple sentence tokens (see examples below.)

2. Creating pattern classes. In the next step, we merged identical patterns that were found in different sentences, keeping pointers to the original containing sentences for later reference. Then we grouped the patterns into classes, assuming two patterns to be in the same class if: 1) both patterns had the same lexical value for the Verb constituents, 2) the two Yago arguments were located under the same constituent labels.

As a result, the original 11,000 training sentences were grouped into 376 different pattern classes. The classes were ranked according to the total number of training sentences the class members covered. We found that there were only 64 classes that contained at least 2 patterns, and that these covered about 97% of all the original training sentences.

In the following example, we show pattern classes ranked #1 and #4 and a few of their pattern elements along with the number of training sentences covered by each:

```
Class id: 8
Sentences covered by patterns in class: 1092
Patterns in class: 210
(('Verb', 'win'),
 ('Subj', '#PERSON#'), ('Obj', '#PRIZE#'))     548
(('Verb', 'win'),
 ('Subj', '#PERSON#'), ('Obj', '@CARDINAL@ #PRIZE#s')) 99
(('Verb', 'win'),
 ('Subj', '#PERSON#'), ('Obj', '@YEAR@ #PRIZE#'))      98
(('Verb', 'win'),
 ('Subj', '#PERSON#'), ('Obj', '@ORDINAL@ #PRIZE#'))   48
(('Verb', 'win'),
 ('Subj', '#PERSON#'), ('Obj', 'Daytime #PRIZE#'))     22
...

Class id: 5
Sentences covered by patterns in class: 406
Patterns in class: 258
(('Verb', 'be'),
 ('Subj', '#PERSON#'), ('Obj', '#PRIZE#'))      27
(('Verb', 'be'),
 ('Subj', '#PERSON#'),
 ('Obj', '#PRIZE# -winning American actor'))      18
(('Verb', 'be'),
 ('Subj', '#PERSON#'),
 ('Obj', '#PRIZE# -winning American actress'))    18
(('Verb', 'be'),
 ('Subj', '#PERSON#'),
 ('Obj', 'recipient of the #PRIZE#'))     10
(('Verb', 'be'),
 ('Subj', '#PERSON#'),
 ('Obj', 'American #PRIZE# -winning actor'))      8
...
```

In this example, pattern class with id "5" contains 258 patterns, covering 406 training sentences. All the patterns in this class have "be" as the main verb, the 1st Yago argument (person name) is in the subject position, while the 2nd argument (award name) is in the object position.

3. Identification of marker tokens. As it can be seen in the example patterns above, the pattern constituents may contain a number of different tokens around the award name (marked by the #PRIZE# meta-token.) Some of these tokens obviously indicate that the sentence corresponds to the event in question (e.g. "winner", "recipient" etc.), while others obviously not ("actor", "actress" etc.) Our goal was to attempt to automatically identify marker tokens that correlate with the extraction domain, and to set them apart from non-markers that are irrelevant and thus could be omitted from the suggested generalized patterns.

For the identification of marker tokens, we used Pearson's one-sided χ^2-test [15]. To achieve this, we needed negative examples, which, in contrast to the positive sentences described in the previous section, were not related to the relation under examination. These were produced by taking all the sentences in the persons' articles that did not include any of the positive sentences. Since these outnumbered the positives, we took a random sample to balance the two categories. Using the χ^2-test, we selected those tokens that did not show independence of the two categories. We used an empirically set threshold of 25.0 for the test (the critical value would have been 6.635 for $\alpha = 0.01$, but we chose a more strict threshold after some empirical tests.)

4. Creating generalized patterns from the classes. In the last step, we generated a single suggested general pattern for each pattern class, containing only the marker words identified by the χ^2-test. These suggested patterns were then given to a human knowledge expert for reviewing and categorizing into positive or negative patterns. The following generalized patterns were produced for the two pattern classes shown above:

```
Class id: 8
+(('Verb', 'win'),
 ('Subj', '* #PERSON# *'), ('Obj', '* #PRIZE# *'), ('ObjII', '*'))
```

```
-(('Verb', 'win'),
 ('Subj', '* #PERSON# *'),
 ('Obj', '* #PRIZE# -nomination|nomination'), ('ObjII', '*'))

Class id: 5
+(('Verb', 'be'),
 ('Subj', '* #PERSON# *'),
 ('Obj', '* #PRIZE# -winning|recipient|winner|winning'))
+(('Verb', 'be'),
 ('Subj', '* #PERSON# *'), ('Obj', 'recipient|winner #PRIZE# *'))
```

The S component of the (G, S) ordered pairs in the generalized patterns may contain disjunctive lists of tokens, separated by the "|" character, meaning that at least one of these items must be present in the given position for the pattern to match. The special character "*" means that any token can stand in that position. The "+" prefix indicates patterns marked positive, and the "-" prefix indicates patterns marked negative by the annotator (in the above example, sentences about award nominations are excluded by the patterns.)

5.3 Evaluation

In the *awards* domain, a human domain expert reviewed 43 suggested patterns that belonged to pattern classes covering at least 5 positive training sentences, covering 92% of all the positive training sentences. The work took about 1.5 hours, and produced 28 positive and 5 negative patterns.

Information extraction with the automatically extracted and human-approved patterns used the same parsing and named entity recognition methods that were described in Section 5.1. The annotated input text was first checked for negative patterns, then the remaining sentences were matched against the positive patterns.

We evaluated the results against a manually annotated corpus of 100 randomly selected person articles. Table 3 shows the precision and recall results of the evaluation of information extraction using both the automatically extracted patterns and the baseline system using completely manually developed extraction patterns.

Table 3: Evaluation of information extraction using automatically extracted (AE) and manually developed (PB) patterns in the "awards" domain

	Precision	Recall	F-measure
AE	91.66%	36.70%	52.41%
PB	93.97%	50.00%	65.27%

It can be seen from Table 3 that the precision of the automatically extracted patterns comes close to the manual system's. Recall, however, is significantly higher in the case of manually created patterns. It could likely be improved by adding more lower-ranked suggested patterns for human revision, which would require more time for revision but would exploit more information from the training sentences.

We were interested in how the pattern extraction framework would fare in other domains, with different types of relations and entities. We used Yago's *IsMarriedTo* relation, which holds between *person* entities (we will refer to this as the *spouse* domain), to generate a training corpus of about 1000 sentences, and used the framework to derive suggestions for extraction patterns. 31 patterns (25 positive and 6 negative) were approved by the knowledge expert in the end, requiring about 2 hours of work.

Preliminary tests showed that this domain would present challenges to a regular expression- and lexicon-based name recognizer, therefore we added the dedicated Stanford named-entity recognizer [13] to aid the correct recognition of person name boundaries in the input sentences. In addition, we also used simple, rule-based coreference-resolution [5] in order to track mentions of the proper names throughout the text and to be able to produce normalized forms of names in the output.

The system adapted to this domain was used to extract marriage relations inside 100 persons' Wikipedia pages, which were previously human-annotated with the correct answers. We were also curious about how it would compare to a machine-learning solution for this domain, so we trained the maximum entropy classifier on the 1000-sentence training set described above with the features described in Section 4. As before, we also carried out evaluation of the union and intersection combinations of the two methods to see how they complemented each other. The evaluation results can be seen in Table 4.

Table 4: Evaluation of information extraction using automatically extracted patterns (AE) and machine learning (ML) in the "spouse" domain

	Precision	Recall	F-measure
AE	89.97%	35.30%	50.71%
ML	90.43%	53.19%	66.98%
Union	90.36%	61.87%	73.45%
Intersection	100.00%	22.26%	36.41%

Machine learning outperformed the pattern-based approach in terms of recall, while the precision of the two approaches was nearly identical. A possible explanation could be hypothesized from the fact that Stanford NER's more general classifier is outperformed by the dedicated classifier that was trained on data that is more similar to the evaluation data (person names inside Wikipedia articles), leading to a higher coverage of recognized person names and thus a higher recall in the relation extraction. The combination tests revealed in this case, too, that the two approaches are likely to miss out in different situations. The union of the two result sets produced the same precision but a significantly higher recall than than either of the two methods.

6 Discussion and Conclusion

We have described several methods for the reliable extraction of massive numbers of facts from the Wikipedia online encyclopedia. In practice, for the *studies*, *awards* and *spouse* domains, we successfully applied these methods to extract more than 70,000 facts for about 35,000 persons in Wikipedia. In addition, we also created and applied solutions to learn properties of *award entities* (name of award, date it was first awarded, who is it awarded by, who are awarded, who was the award named after) and *geographical regions* (name of region, capital, containing and contained regions) using Wikipedia. This information was integrated into the semantically linked database of the iGlue project[6].

We have described a frame-based information extraction system that relies on deep natural language processing and manually crafted extraction patterns. This approach can be useful if no annotated training data is available, or when producing such annotations could be too costly (for example, when there are too many different slots in the extraction frame, as it is the case with the *studies* relation which has 6 slots (see Section 3)).

In our pattern-based approaches, the use of reliable, custom-made named entity recognizers is crucial. However, in many cases, this task can be overcome by resource-friendly methods such as regular expressions and/or lexicons, which can be complemented by simple but effective heuristics that utilize the special advantages of Wikipedia (hyperlinks, page titles, tables and category labels, special formatting etc.)

We have also proposed an approach for generating extraction patterns from labeled data, requiring only minimal amount of work on behalf of human knowledge experts. We have demonstrated that much of the burden of preparing training data can be reduced by utilizing existing semantic resources, such as the Yago Ontology, and taking advantage of both the redundancy and the volume of information found in an encyclopedia such as Wikipedia. The performance of information extraction using automatically generated patterns and the fraction of the human effort can be well compared to completely manually created systems, when precision is of prime importance.

We also compared the performance of the pattern-based approaches, using both manually and automatically generated patterns to the performance of machine learning solutions using state-of-the-art supervised algorithms and less resource-intensive natural language processing procedures. Our results indicated that while precision is comparable for the approaches, recall was favored by different approaches. In more detail, the experiments revealed that in terms of recall, manual patterns outperform machine learning (Table 2) and automatically extracted pattens (Table 3), while machine learning has better recall than automatically extracted patterns (Table 4). This suggests an order in the (F-measure) performance of the three approaches:

manual patterns (PB) > machine learning (ML) > automatically extracted

[6] `http:\\iglue.com`

patterns (AE).

While the superiority of the manually constructed patterns over the two other approaches is more obvious, the relationship between the performance of the machine learning and the pattern extraction methods raises new questions. In particular, it should initiate new experiments in order to tune the performance of our pattern learning algorithm.

The examination of the combination of the machine-learning and pattern-based approaches (using both manually and automatically generated patterns) revealed that they complement each other well, leading to a straightforward way of extending the performance of the pattern-based system. In the future it would be interesting to experiment with a deeper synergy of the two methods, for example using more sophisticated features available from deep parsing to train the classifiers.

References

[1] Agichtein, E., Eskin, E., and Gravano, L. Combining strategies for extracting relations from text collections. In *Proceedings of the 2000 ACM SIGMOD Workshop on Data Mining and Knowledge Discovery*, 2000.

[2] Agichtein, Eugene and Gravano, Luis. Snowball: Extracting relations from large plain-text collections. In *In Proceedings of the 5th ACM International Conference on Digital Libraries*, pages 85–94, 2000.

[3] Auer, Sören, Bizer, Christian, Kobilarov, Georgi, Lehmann, Jens, Cyganiak, Richard, and Ives, Zachary. Dbpedia: a nucleus for a web of open data. In *ISWC'07/ASWC'07: Proceedings of the 6th international The semantic web and 2nd Asian conference on Asian semantic web conference*, pages 722–735, Berlin, Heidelberg, 2007. Springer-Verlag.

[4] Banko, Michele and Etzioni, Oren. The tradeoffs between open and traditional relation extraction. In *Proceedings of ACL-08: HLT*, pages 28–36, Columbus, Ohio, June 2008. Association for Computational Linguistics.

[5] Bontcheva, Kalina, Dimitrov, Marin, Maynard, Diana, Tablan, Valentin, and Cunningham, Hamish. Shallow methods for named entity coreference resolution. In *TALN 2002*, June 2002.

[6] Brin, Sergey. Extracting patterns and relations from the world wide web. In *WebDB '98: Selected papers from the International Workshop on The World Wide Web and Databases*, pages 172–183, London, UK, 1999. Springer-Verlag.

[7] Bunescu, Razvan C. and Mooney, Raymond J. A shortest path dependency kernel for relation extraction. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 724–731, Morristown, NJ, USA, 2005. Association for Computational Linguistics.

[8] Culotta, Aron, McCallum, Andrew, and Betz, Jonathan. Integrating probabilistic extraction models and data mining to discover relations and patterns in text. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 296–303, Morristown, NJ, USA, 2006. Association for Computational Linguistics.

[9] Culotta, Aron and Sorensen, Jeffrey. Dependency tree kernels for relation extraction. In *ACL '04: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 423, Morristown, NJ, USA, 2004. Association for Computational Linguistics.

[10] Etzioni, Oren, Banko, Michele, Soderland, Stephen, and Weld, Daniel S. Open information extraction from the web. *Commun. ACM*, 51(12):68–74, December 2008.

[11] Etzioni, Oren, Cafarella, Michael, Downey, Doug, Kok, Stanley, Popescu, Ana-Maria, Shaked, Tal, Soderland, Stephen, Weld, Daniel S., and Yates, Alexander. Web-scale information extraction in knowitall. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 100–110, New York, NY, USA, 2004. ACM.

[12] Fellbaum, C., editor. *WordNet: An Electronic Lexical Database.* MIT Press, Cambridge, MA, 1998.

[13] Finkel, Jenny R., Grenager, Trond, and Manning, Christopher. Incorporating non-local information into information extraction systems by gibbs sampling. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 363–370, Morristown, NJ, USA, 2005. Association for Computational Linguistics.

[14] Nguyen, Dat P. T., Matsuo, Yutaka, and Ishizuka, Mitsuru. Relation extraction from wikipedia using subtree mining. In *AAAI'07: Proceedings of the 22nd national conference on Artificial intelligence*, pages 1414–1420. AAAI Press, 2007.

[15] Plackett, R. L. Karl pearson and the chi-squared test. *International Statistical Review / Revue Internationale de Statistique*, 51(1):59–72, 1983.

[16] Riloff, Ellen. Automatically generating extraction patterns from untagged text. In *AAAI/IAAI, Vol. 2*, pages 1044–1049, 1996.

[17] Ruiz-casado, Maria, Alfonseca, Enrique, and Castells, Pablo. Automatic assignment of wikipedia encyclopedic entries to wordnet synsets. In *Proceedings of the Atlantic Web Intelligence Conference, AWIC-2005. Volume 3528 of Lecture Notes in Computer Science*, pages 380–386. Springer Verlag, 2005.

[18] Sagae, Kenji and Miyao, Yusuke. Hpsg parsing with shallow dependency constraints. In *In Proc. ACL 2007*, 2007.

[19] Stevenson, M. and Greenwood, M. A. Dependency pattern models for information extraction. *Research on Language and Computation*, 7(1):13–39, 2009.

[20] Suchanek, Fabian M., Ifrim, Georgiana, and Weikum, Gerhard. Combining linguistic and statistical analysis to extract relations from web documents. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 712–717, New York, NY, USA, 2006. ACM Press.

[21] Suchanek, Fabian M., Kasneci, Gjergji, and Weikum, Gerhard. Yago: A large ontology from wikipedia and wordnet. *Web Semant.*, 6(3):203–217, 2008.

[22] Suchanek, Fabian M., Sozio, Mauro, and Weikum, Gerhard. Sofie: a self-organizing framework for information extraction. In *WWW '09: Proceedings of the 18th international conference on World wide web*, pages 631–640, New York, NY, USA, 2009. ACM.

[23] Wang, Gang, Yu, Yong, and Zhu, Haiping. Pore: positive-only relation extraction from wikipedia text. In *ISWC'07/ASWC'07: Proceedings of the 6th international The semantic web and 2nd Asian conference on Asian semantic web conference*, pages 580–594, Berlin, Heidelberg, 2007. Springer-Verlag.

[24] Weld, Daniel S., Hoffmann, Raphael, and Wu, Fei. Using wikipedia to bootstrap open information extraction. *SIGMOD Rec.*, 37(4):62–68, 2008.

[25] Yan, Yulan, Okazaki, Naoaki, Matsuo, Yutaka, Yang, Zhenglu, and Ishizuka, Mitsuru. Unsupervised relation extraction by mining wikipedia texts using information from the web. In *ACL-IJCNLP '09: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2*, pages 1021–1029, Morristown, NJ, USA, 2009. Association for Computational Linguistics.

[26] Zelenko, Dmitry, Aone, Chinatsu, and Richardella, Anthony. Kernel methods for relation extraction. *J. Mach. Learn. Res.*, 3:1083–1106, 2003.

Speech Recognition Experiments with Audiobooks*

László Tóth[†], Balázs Tarján[‡], Gellért Sárosi[‡] and Péter Mihajlik[‡§]

Abstract

Under real-life conditions several factors may be present that make the automatic recognition of speech difficult. The most obvious examples are background noise, peculiarities of the speaker's voice, sloppy articulation and strong emotional load. These all pose difficult problems for robust speech recognition, but it is not exactly clear how much each contributes to the difficulty of the task. In this paper we examine the abilities of our best recognition technologies under near-ideal conditions. The optimal conditions will be simulated by working with the sound material of an audiobook, in which most of the disturbing factors mentioned above are absent. Firstly pure phone recognition experiments will be performed, where neural net-based technologies will also be tried as well as the conventional Hidden Markov Models. Then we move on to large vocabulary recognition, where morph-based language models are applied to improve the performance of the standard word-based technology. The tests clearly justify our assertion that audiobooks pose a much easier recognition task than real-life databases. In both types of tasks we report the lowest error rates we have achieved so far in Hungarian continuous speech recognition.

Keywords: speech recognition, LVCSR, audiobooks

1 Introduction

Creating speech recognizers that operate reliably in practice requires the handling of several factors that may not arise under laboratory conditions, but are inevitable in real life. The pressure from industry towards creating robust recognizers forces researchers to focus more on issues like the handling of noisy and spontaneous speech. This is of course understandable, but from a scientific point of view it is unusual to move on to a more difficult task before solving the simple one. And even

*This research was partially supported by TÁMOP-4.2.2/08/1/2008-0008, TÁMOP-4.2.2-08/1/KMR-2008-0007, OM-00102/2007 and by KMOP-1.1.3-08/A-2009-0006.

[†] Research Group on Artificial Intelligence, Hungarian Academy of Sciences and University of Szeged, E-mail: `tothl@inf.u-szeged.hu`

[‡] Department of Telecommunications and Media Informatics, Budapest University of Technology and Economics, E-mail: `{tarjanb, sarosi, mihajlik}@tmit.bme.hu`

[§] THINKTech Research Center, Nonprofit LLC, Hungary

the recognition of clean and well-articulated speech has not yet been fully solved, and in particular for Hungarian we do not know how our recognizers would behave under such conditions. We think that the study of simpler recognition problems should not be abandoned: though the result of these are less applicable directly, they can shed light on how the various factors contribute to the difficulty of the recognition of real-life speech, and on how to handle these. Also, there might be applications where good quality speech can be assumed (for example, in a broadcast news captioning system both studio quality recording and good articulation are expected).

In this paper we test our current speech recognition systems on an audiobook. Our aim is to see how they perform under nearly ideal conditions. In Section 2 we explain why we regard the audiobook as providing 'ideal' speech by giving examples of the most important factors that are not present in the material of the audiobook. Then in Section 4 we present pure phone recognition experiments, which are useful for comparing the performance of various acoustic modeling techniques. In real life, however, one would expect a word-level output from a recognizer, so in Section 5 we create language models for the recognition task using both word- and morph-based techniques. In the last section we summarize our findings and draw some pertinent conclusions.

2 Factors Hindering Speech Recognition

In real-life situations several factors arise that degrade the performance of automatic speech recognizers. In this section we list the most important factors and try to assess their impact on the recognition rates in a real-life situation and also in the case of an audiobook.

It is hard to imagine such a real-life recognition environment where background noises could be perfectly excluded. One would find noises even in such high-quality recordings as broadcast news – e.g. paper being rustled and people taking breaths. And in fact there are quite a lot of applications that require speech recognition under a high level of noise, as in a car or cockpit. Many studies have been conducted to compare human and machine speech recognition under noisy conditions, and the results are usually disappointing [10]. The distorting effect of transfer media (most typically a phone line) is also counted as noise, and speech recognizers can be surprisingly sensitive to even a change in the microphone they are used with. Compared to the average background noise conditions and the quality and variability of telephone microphones and transfer lines, audiobooks contain practically no background noise and distortion, as they are normally recorded in sound-proof studios using professional equipment.

Speech recognizers can perform quite differently for different persons; that is, they are sensitive to the articulation characteristics and peculiarities of the speaker. We can demonstrate this by creating a histogram of the recognition accuracies for various speakers. The MTBA Hungarian Telephone Speech Corpus [23] is really suitable for such a test, as it contains recordings made from 500 people. Fig. 1

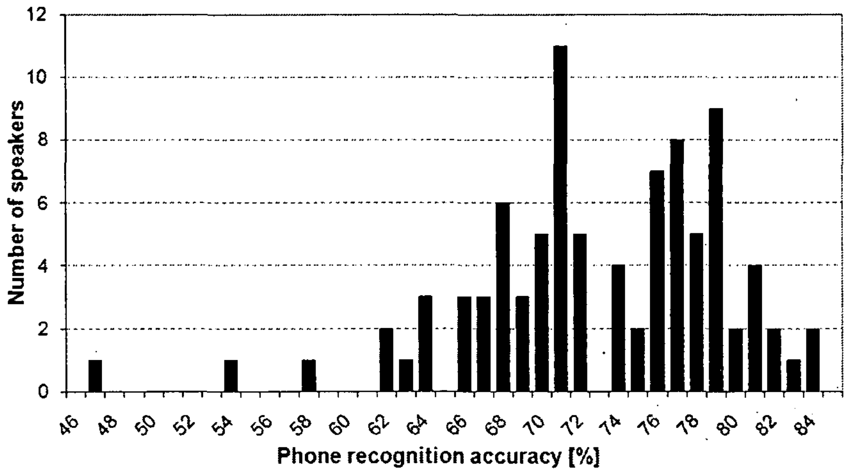

Figure 1: The distribution of phone recognition accuracy as a function of the speaker on the MTBA corpus

shows a histogram of phone recognition results obtained using this database that were presented in an earlier paper by the authors [21]. As can be seen, compared to the average value of around 74%, the results have quite a huge deviance in both directions. Though the recording conditions (phone line and background noise) also vary from speaker to speaker, we observed that the recordings are usually clean, so we think that the huge deviation in the scores can mostly be blamed on the sensitivity of the recognizer to the actual speaker's voice. Compared to the MTBA database, the sound material of the audiobook we used was presented by just one person, so the problems that could be caused by changing speakers were totally excluded.

The speech databases used to train speech recognizers were, for a long time, created by asking people to read aloud some written text. It was realized not long ago that there are huge differences between spontaneous articulation and careful pronunciation (e.g. reading aloud). Research has turned only recently towards studying real-life spontaneous speech. To assess the influence of spontaneous pronunciation on speech recognition, tests were made where people were asked to re-read their own passages that had been recorded at a meeting. The speech recognition error jumped by a factor of about two between the two recording (pronunciation) conditions. As regards Hungarian, Tarján et al. ran recognition tests on both planned and spontaneous speech using the same technology [20]. Although the results are not precisely comparable, as there were differences other than speaking style between the two recording situations, the error rate increase of a factor of two that they obtained accords well with the international findings. This drop in performance is so big that there are various applications where a trained speaker is applied to repeat the spontaneous parts – for example, in a Japanese system that uses speech recognition to close-caption broadcast TV news [27]. These problems cannot arise with an audiobook that is presented by an actor or actress. We think this is a guarantee for the cleanest possible articulation.

A further factor that may adversely affect the pronunciation is the presence of an emotional load. Similar to spontaneous articulation, the study of emotional speech has become popular only relatively recently. In Hungarian these studies have just been started [22], so we cannot give concrete examples of its influence on the recognition rates. In contrast with the above three factors, this one may be present to some degree in an audiobook, as the reading actor might use an emotional coloring of his voice to increase the expressive power of his speech. However, in the concrete recording we chose, such situations did not occur.

3 The Audiobook and its Preparation

Audiobooks have gained increasing popularity in the last couple of years. A lot of novels have been released in this 'talking book' format in Hungarian as well. We chose an audiobook for which the original novel is old enough so that its text is no longer copyrighted. Our choice fell on the short story collection by Gyula Krúdy entitled 'Sinbad's Voyages', presented by the actor Sándor Gáspár. The total duration of the audiobook was 212 minutes, and we converted it to Microsoft Wave mono 16 bit PCM format with a sample rate of 16kHz. The whole book was carefully listened to, looking for differences between the original text and the sound material. Only a minimal amount of such differences were found, consisting mostly of the skipping or the insertion of short interjections such as 'ah'. The music signals occurring at the end of each chapter were of course removed, and each file was segmented further into roughly two-minute long parts. Apart from these, no other modification was required.

For training and test purposes the recordings were divided into two parts. From the ten short stories eight were used for training (186 minutes) and two for testing (26 minutes).

4 Phone Recognition Experiments

In the first set of experiments the goal was pure phone recognition. Here no higher (word or morpheme-level) language model is applied, so the aim of recognition was to output a phonetic transcript that was as close to the signal's real phonetic content as possible. This set of experiments was motivated by our recent phone recognition results on the MTBA database [21]. That is, we intended to apply the same methodology as was used there, and by comparing the results we hoped to gain a good insight into how much the factors mentioned above hinder the recognition performance. To aid the comparison, the results obtained with the MTBA corpus will be repeated here (a detailed description of the corpus can be found in [23]). The acoustic modeling technique used was the same as that we apply and discuss here.

In addition to the conventional hidden Markov modeling scheme, in the acoustic modeling step we applied a neural net-based technology as well. Hence, after presenting the data processing steps, we first give a theoretical description of this approach, and move on to the details of the implementation only afterwards. We round off this section with the comparison of the phone recognition scores obtained using the various techniques.

4.1 Preprocessing of the Data

In the feature extraction step, the most conventional 39-dimensional mel-frequency cepstral (MFCC) feature set (including the delta and acceleration coefficients) was extracted from the recordings [8]. The MTBA database was represented by PLP features in the reference that served as our comparison, because we had to comply with an English system in that paper [21]. However, previous experience tells us that the recognition results obtained with the two representations rarely show any significant difference.

Training and testing a phonetic recognizer requires a phone-level transcription of both the train and the test data. However, for the audiobooks only the orthographic transcription was readily available, which we had to convert to its most probable phonetic manifestation. This is usually done by collecting the different word forms occurring in the corpus, and transcribing them one by one. The word-level transcript given for each sentence is then mapped to the most probable sequence of phonetic segments by so-called forced alignment [8]. We performed this alignment using a set of phone models trained on the MRBA corpus [24] in earlier experiments.

Though Hungarian writing is almost phonetic, there are several difficulties which are not trivial to handle in an automatic phonetic transcription system. The first one is the case of the two-character letters, which can be identified only by morphologic analysis (a classic example is the word *pácsó*). The second one is that certain consonant clusters may or may not undergo assimilation depending on the position of morpheme boundaries. Lastly, in some cases assimilation is optional, and more than one pronunciations might be correct. A typical example of this are the word boundaries, where the consecutive words may be pronounced with a short silence between, but coarticulation and even assimilation may also occur.

Due to the above, a precise phonetic transcriber is not easy to create. Hence we decided to construct the phonetic transcript at the level of syllables instead of words. This idea was motivated by the fact that with careful pronunciation the vowels are not discarded or reduced (apart from shortening and lengthening). Second, assimilation affects only the consonant clusters and does not spread across vowels. Thus instead of creating a pronunciation dictionary of words, we created it for consonant clusters (for simplicity we will refer to these as syllables, though they are different from the linguistic definition of the syllable). We should add that the space character was also treated as a special consonant, so our 'syllables' were allowed to spread across word boundaries. The text of the audiobook contained 7186 unique word forms, but only 809 different syllables. These units were then transcribed manually. This transcription contained all the possible pronunciation alternatives of the syllable, so for example the letter-sequence 'T SZ' has the pronunciations [t sil s], [t s] and [t͡s], where *sil* denotes silence. The drawback of this method is that it does not see the whole word during transcription, and so it offers pronunciation alternatives that are not correct in the given context (for example, [pa:t͡ʃo:] for the word *pácsó*). We hoped that in such cases the recognizer would be able to automatically choose the correct transcript during the forced alignment phase because of its better acoustic match. The phonetic label set used by the transcriber contained 52 symbols.

Besides the transcriber described above, an alternative phone-level transcription was also created using the full (language model-supported) recognizer configuration that will be presented in the next section. This resulted in a quite different series of symbols, because for those experiments a different phonetic transcriber was used. This was designed based on our earlier experimental findings that in large vocabulary recognition the explicit handling of the assimilations and similar phonological phenomena during the phonetic transcription of the dictionary is not necessary, provided that context-dependent phone models are applied at the phone level [14]. The explanation is that these models are implicitly aware of the phonetic context, and hence are sufficient to cope with most coarticulation artifacts. Therefore the phonetic transcriber used in these experiments performs only the most trivial grapheme-phoneme conversions, *and does not model such phonological processes as assimilation.* Also, the short and long consonants were not handled as separate labels by this converter, because fusing them causes only a negligible amount of confusions at the word level. Thus it operates with a label set of only 38 phonetic symbols.

4.2 Artificial Neural Net-Based Acoustic Modeling

The conventional Hidden Markov Modeling (HMM) technique approximates the probability distribution of the building blocks (model states) by fitting Gaussian mixtures (GMM) on the training data. In monophone modeling the states correspond to phone-thirds, while in the case of context-dependent triphone models the phonetic context is also reflected in the labeling of the building units (often referred to as senons [8]). This techniques raises the number of states from about one hundred to several thousands, but also brings considerable improvements in the performance.

Rather than refining the recognition units, an alternative approach is to improve the estimation of the probability distributions. One such solution is to use artificial neural nets (ANN) instead of Gaussian mixtures. The default training algorithm of neural nets is discriminative, and thanks to this it generally achieves slightly higher classification accuracies than the generatively trained GMMs. Secondly, the ANN-based systems are trained on several neighboring frames instead of just one, which again yields significantly better results. We should mention, however, that both discriminative training and the use of a longer observation context is possible with GMMs as well – it is just not part of the mainstream technology.

Under proper circumstances the output values of the neural net can be interpreted as probability estimates, and can be integrated into the conventional HMM scheme with some slight modifications to it. The resulting construct is known as the HMM/ANN hybrid model [2]. For smaller subtasks the hybrid was reported to outperform conventional HMMs. For example, the best phone recognition results on the TIMIT database were all obtained by applying neural nets [17, 15]. In larger systems, however, their applicability is much less obvious. First, their extension to context-dependent phone models is problematic, while on large (hundred hours) databases context-dependent modeling brings about a huge improvement for classic HMM models. Second, their combination with the standard language models seems to be less effective than that with conventional HMMs.

Most of these problems can be circumvented by using the so-called HMM/ANN tandem technology [6]. Here the neural net outputs are not interpreted as probability estimates, but as a non-linear transformation of the feature set. Hence, they can be used as the input for training a conventional HMM. With this trick only the acoustic preprocessor gets replaced and no other modification of the standard HMM recognizer is required. The only obvious drawback is that now there will be two training steps instead of one, and the evaluation will obviously also be slower.

Fig. 2 shows a schematic diagram of the processing steps involved for the conventional, the hybrid and the tandem methods.

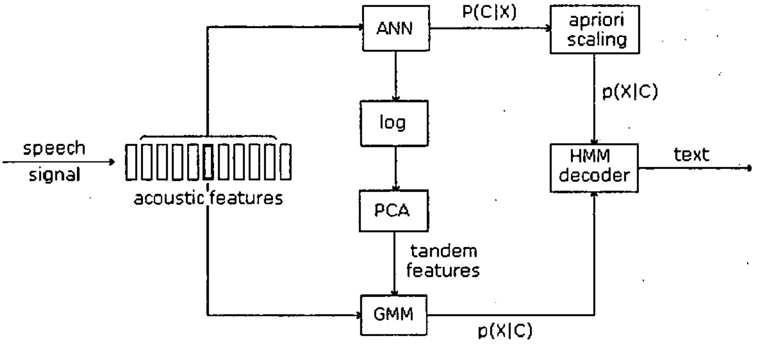

Figure 2: Schematics of the processing steps for the conventional HMM (lower path), the hybrid (upper path) and the tandem (middle path) models

4.3 Implementation Details

After the theoretical issues here we present the technical details of our implementation. The neural net we applied was a conventional feed-forward multilayer perceptron net with one hidden layer of 500 neurons. The number of outputs was the same as the number of phones, that is, 52 in the case of the syllable-based transcriber and 38 for the word-based transcriber. The output layer applied the softmax nonlinearity, while the hidden neurons utilized the sigmoid function. The number of inputs was 351, corresponding to 9 neighboring frames of MFCC vectors. The net was trained with backpropagation, and cross-validation on 10% of the training data served as the stopping criterion. The training targets were the phone labels obtained via forced alignment during the data preparation step. Before using the net outputs as features in the tandem, they were logarithmized and decorrelated using PCA. This eases the fitting of the data by Gaussian curves. A further trick we applied was to concatenate the tandem feature set with the MFCC feature set, and use them together. Though theoretically the two sets are highly redundant, the method still brings about a slight improvement in performance in practice.

As the implementation of the conventional HMM recognizer the well-known HTK package was used [26]. The phone models applied were standard 3-state left-to-right models with 9 Gaussians per state. In the case of the conventional HMM configuration both monophone and triphone models were trained. For the hybrid and tandem systems only monophone models were used. This is because efficient hybrid/tandem triphone modeling would require training the neural net with the senons as targets instead of the phones, and handling such a huge net is problematic. Extending ANN-based modeling to context-dependent cases is thus the most important topic of this area nowadays [1].

After the conventional maximum likelihood training the HMM models were further refined by the application of discriminative training, using the maximum mutual information (MMI) training criteria. Fortunately, the HTK package contains this algorithm.

As we mentioned previously, our goal here is pure phone recognition, so word-level (and morph-level) language models will be applied only in the next section. However, it is usual to support the recognition process at this low level by the application of phone bigrams. These were created from the force-aligned phonetic transcript of the training database.

4.4 Results and Discussion

Table 1 summarizes the phone accuracy scores obtained with the various acoustic modeling techniques, both for the syllable-based and the word-based automatic phonetic transcripts (the MTBA corpus was transcribed manually). For comparison the performance of the same models on the MTBA database is also shown, where the corresponding results were available (cf. [21]).

Comparing the results obtained with the syllable-based and the word-based transcripts, the former are consistently better for all but the simplest configuration. The accuracy obtained with the triphone model is, however, among the three highest, in spite of the fact that it was trained on the word-level transcript. This is in accordance with our earlier finding that context-dependent triphone modeling is able to handle not only the phonetic, but the phonological coarticulation effects as well, so it does not require the explicit modeling of the latter during phonetic transcription [14]. All the other models were trained with context-independent target labels, and they all show a degraded performance on the word-level transcripts.

Table 1: Phone recognition accuracies with the various acoustic models (the three highest scores are shown in boldface).

	MTBA (manual)	audiobook (syllabic)	audiobook (word-level)
HMM (monophone, no lang. model)	53.37%	72.18%	72.95%
HMM (monophone, phone bigram)	—	80.64%	76.85%
HMM (triphone, phone bigram)	—	—	**85.88%**
Tandem (monophone, no lang. model)	65.09%	79.49%	78.15%
Tandem (monophone, phone bigram)	69.67%	83.62%	80.93%
Tandem (mono., phone bg., discr. tr.)	73.93%	**86.26%**	82.84%
Hybrid (no language model)	—	84.84%	82.10%
Hybrid (phone bigram)	—	**86.60%**	82.69%

This indicates that monophone models do require the help of the phonetic transcriber. A further observation is that the phone-bigram consistently adds smaller improvements to the scores of the models trained with the word-level transcripts. A reasonable explanation is that the assimilation rules applied by the syllabic transcriber decrease the perplexity of the phone-bigram (consider, for example, the case of voiced-unvoiced consonant connections). This points towards the preference of full phonetic transcriptions when phone recognition output is required (but, of course, this argument has no importance when higher level language models are present).

As regards comparing the various modeling techniques, we can see that the tandem and hybrid models are indeed capable of the same (or even slightly better) performance than the conventional triphones, in spite of the fact that they use only monophone labels. Therefore, it would be very important to find such methods that are able to combine the advantages of the ANN-based and triphone technologies.

Now, comparing the audiobook results with those obtained with the MTBA corpus, one can find huge differences in each row of the table. These differences reflect how much easier it is to recognize the content of an audiobook than recordings made in real-life situations. The best result obtained on the MTBA is just slightly better than the worst score with the audiobook. We should also mention here that the situation could be even worse, because the MTBA corpus contains read speech (through phone lines). The recognition results of spontaneous phone calls would presumably be even worse. At the other end, the 86% accuracy score obtained on the audiobook is quite good: we made a test where a human subject was asked to read the output of the phone recognizer, and he reported that he was able to understand the whole story (one of the stories from the test set), apart from a couple of sentences. But, of course, an objective evaluation would require creating a language model, so that the whole recognition process would be automated. This is just what we shall do in the next section.

5 Large Vocabulary Speech Recognition Experiments

Phone recognition tests can be informative when we compare various acoustic models, but in a practical situation we of course expect a word-level output from a recognizer. In this section large vocabulary recognition test will be presented. On the acoustic modeling side only the conventional triphone HMM technique will be pursued, but in addition to the speaker-specific model training, comparisons will be made with models that have been trained on a larger, multi-speaker corpus. An adaptation of these models to the single speaker of the audiobook will also be attempted. On the language modeling side, we first present experiments with conventional word-level n-grams. In Hungarian the number of word forms is much higher than that in English, so the decomposition of the words into morphs can be beneficial. We have recently shown the usefulness of this technique on other domains [13], and we shall apply the same technology here. For the construction of a word- or a morph-based language model a large corpus of training text corpora is required. The various methods we applied for this and the parameters of the corpora assembled will also be presented. After, we will present our findings and discuss them.

5.1 Speaker-Independent, Speaker-Adapted and Speaker-Specific Acoustic Modeling

First a speaker independent state clustered cross-word triphone model was trained using ML (Maximum Likelihood) estimation [26]. Three-state left-to-right HMMs (Hidden Markov Models) were applied using GMMs (Gaussian Mixture Models) for each state. The model was trained on the MRBA database [24] augmented with 10 hours of transcribed press conference speech [20]. The feature type was MFCC (Mel Frequency Cepstral Coefficients) with delta and delta-delta parameters, which were calculated using blind channel equalization [11] at 8 kHz bandwidth. The resulting model contained 2100 HMM states with 7 Gaussians for each state. This speaker independent (SI) acoustic model will be referred to as **HMM (triphone, SI)**.

In the next step, speaker adaptation was applied using the unsupervised MLLR [9] technique. Speaker independent ASR was performed on the test and training set altogether, and the results were used as transcripts for a formally supervised adaptation. Thus the model parameters were retrained while the number of HMM states and the number of Gaussians per state were not changed. The speaker adapted (SA) acoustic model will be denoted by **HMM (triphone, SA)**.

In the third configuration the speaker independent model was used only for the phone labeling of the training set using forced alignment. Then an entirely new, speaker dependent (SD) model was trained from the MFCC features of the training data using the forced labels. This model had 1400 HMM states, each consisting a mixture of 10 Gaussians, and will be referred to as **HMM (triphone, SD)**. The same HMM-based triphone model was also used in Section 4 (third line of Table 1),

Table 2: Statistics of the language model training databases

	Size	Word Vocab.	Morph Vocab.	Word PPL	OOV Rate [%]
NM	17.6K	6.3K	2K	604	30.3
AM	1.4M	152K	18K	1792	2.9
SM	11.5M	590K	49k	2136	2.4

so we can make a fair comparison between the phone-level and the LVCSR (Large Vocabulary Continuous Speech Recognition) results.

5.2 Language Modeling

5.2.1 Collection and Preparation of the Text Corpus

The training transcript of the recorded literary novel contained fewer than 18K words. This transcription formed the smallest, **Novel Matched (NM)** corpus. Thereafter a separate training set – independent of the novel – was collected from the same author, Gyula Krúdy, which resulted in a 1.4M word size database. The source of this **Author Matched (AM)** corpus was the freely accessible Hungarian Electronic Library [28]. Finally, the corpus size was increased nearly ten times from the works of other authors from the early twentieth century, thus an 11.5M word count **Style Matched (SM)** corpus was created. This expansion includes texts from the Hungarian Electronic Library, the Digital Academy of Literature [29] and the Electronic Archive of Periodicals [30].

Following the text collection, the database had to be processed further. No extraordinary corpus preparation was required for word-based speech recognition, just the removal of any non-word characters, some number-to-text conversion and finally the conversion of all letters to lowercase. In contrast, the morph-based system required a special treatment of the given training text data. In our approach, first word boundary symbols <w> are placed into the text after each word, and are considered as separate morphs. (<w> symbols are required for the reconstruction of word boundaries in the decoder output [7]). Then segmentation is performed on the word dictionary by using the Morfessor Baseline (MB) algorithm [4]. MB is an unsupervised, language independent method for splitting words into morpheme-like lexical units called morphs. The method aims at the determination of the optimal lexicon and segmentation, that is, a set of morphs that is concise, and moreover gives a concise representation for the data. The corpus for a morph-based speech recognition system is obtained by replacing each word of the corpus by the corresponding morph sequence. For the statistical details of the training databases see Table 2 (PPL stands for perplexity and OOV denotes the out-of-vocabulary words).

5.2.2 Word- and Morph-Based Language Models

All the word- and morph-based n-gram language models were built on the corresponding database with *full vocabularies* applying the modified, interpolated Kneser-Ney smoothing technique [3] implemented via the SRILM toolkit [18]. Performance tests were run with several order of n-gram models – 2- to 4-grams – to find the optimal language model parameters. Based on these, full 3-gram language models were built for the words and full 4-gram models for the morphs (ignoring 3- and 4-grams found only once). No language model pruning was applied in our experiments.

5.2.3 Pronunciation and context dependency models

Simple grapheme-to-phoneme rules [19] and an exception list were used to generate word- and morph-to-phoneme mappings. The <w> symbols in the morph-based models were mapped to optional silences (similar to the 'short pause' (sp) model in [26]), while in the case of word-based models optional silences were added to the end of each word. The pronunciation models were further processed by applying triphone context expansion, as shown in equation (1) below. This includes not only the inter-word dependencies but the cross-word or cross-morph context dependencies as well, taking the optional inter-word silences into consideration. As was already mentioned in Section 4, phonological co-articulations were not considered explicitly by this pronunciation model.

5.3 The Recognition Network and the Decoder

The final step was the creation of a triphone level WFST (Weighted Finite State Transducer) [16] recognition network:

$$wred(fact(compact(C \circ S \circ compact(det(L \circ G)')))), \tag{1}$$

where the capital letters are transducers and the others are operators. This process commences with the composition and determinization of the language model (G) and the pronunciation model (L), then a suboptimal minimization process is applied. The optional silences are replaced with null transitions and silence models using the (S) transducer. Next, the context expansion is performed using the (C) transducer, then the network is minimized, factorized, and the weights are redistributed, resulting in a stochastic transducer suitable for a WFST decoder.

The above-described networks return word or morph sequences during the decoding process. Hence, a special operator has to be inserted in the computational process to obtain a phone sequence as output when the large vocabulary recognizer is used in phone recognition mode:

$$wred(fact(compact(C \circ proj(S \circ compact(det(L \circ G)'))))), \tag{2}$$

Table 3: Phone error rate (PER) results in [%]

	Training text					
Acoustic	NM (17.5K)		AM (1.4M)		SM (11.5M)	
Models	Word	Morph	Word	Morph	Word	Morph
triphone SI	26.9	23.9	13.7	14.2	13.7	14
triphone SA	19	13.2	6.7	6.5	5.8	6
triphone SD	14.6	7.4	3	3.1	2.5	2.7

where the projection operator copies the input labels of the silence model-substituted LG model in its output labels. All the operations were performed with "Mtool"* WFST building tool.

In the tests one-pass recognition was performed using the WFST decoder called VOXerver.* The tests were run on a Core 2 Quad processor at 2.67 GHz with 16 Gbytes of RAM. The RTF (Real Time Factor) of the morph- and the corresponding word-based system were adjusted so as to be nearly equal using standard pruning techniques. All tests ran in real-time (RTF<1) except for those that were performed with speaker independent acoustic models. In the following tests by the term relative improvement we mean the following:

$$Relative\,Improvement = \frac{ER_{reference} - ER_{new}}{ER_{reference}} * 100\% \qquad (3)$$

5.4 Results and Discussion

In this section, large vocabulary experimental results will be presented. First the phone error results (PERs) of section 4 are compared to PERs of large vocabulary recognizers, then word and letter error rates (WERs and LERs, respectively) are presented. After that, the morph-based improvements will be investigated. While WER and LER were calculated from the standard output of the large vocabulary decoder, special recognition networks had to be built to determine PER (see Section 5.3).

The primary aim of these experiments was to improve the phone recognition accuracy scores by utilizing higher level language models. As can be seen in Table 3, for most configurations we managed to outperform the phone-bigram models (the best error rates reported in Table 1 were around 14%). The improvement is especially good if there is a large training corpus available. By using large vocabulary language models even the distortion caused by poor acoustic modeling can be compensated for. Namely, the phone error rate that can be obtained with the combination of the largest language model and the general-purpose acoustic model is roughly the same as the scores of the best phone-bigram recognizers (~14%). The

*These tools were developed at AITIA International, Inc.

Table 4: Letter error rate (LER) results in [%]

Acoustic Models	Training text					
	NM (17.5K)		AM (1.4M)		SM (11.5M)	
	Word	Morph	Word	Morph	Word	Morph
triphone SI	29.1	25.5	14.3	14.4	14.3	14.4
triphone SA	21.7	15	7	6.3	6.1	5.8
triphone SD	17.7	9.9	3.3	2.8	2.5	2.4

Table 5: Word error rate (WER) results in [%]

Acoustic Models	Training text					
	NM (17.5K)		AM (1.4M)		SM (11.5M)	
	Word	Morph	Word	Morph	Word	Morph
triphone SI	68.3	61.9	37.1	37.4	36	36.8
triphone SA	65	50	24.9	22.6	21.6	20.8
triphone SD	61.6	41.6	17.5	14.2	13.4	12

only case where we got worse accuracies is when both the language and acoustic models were severely under-trained.

Large vocabulary speech recognizers are commonly characterized by their word and letter error rates. WER is the most widely applied way of evaluation, however, the LER provides a more realistic error measure in the case of morphologically rich languages. In our LER calculation the white spaces between words were modeled by a dedicated letter. Tables 4 and 5 summarize the word- and morph-based recognition results that were measured applying various acoustic models and training text corpora of various sizes.

The results clearly reveal the advantage of using a task-specific acoustic model. The best recognition results were attained with the model trained specially for the audiobook task. However, in many cases the available transcribed audio data is insufficient for training a new model. In such a case adapting the speaker independent model using the speaker-specific database – even in an unsupervised manner – provides a reasonable alternative.

A good match between the training and test sets is important, but it is not crucial for effective language modeling. As the results suggest, collecting large amounts of textual data is more rewarding than applying a well-matched but under-resourced database for the language model training. By combining the most elaborate models, the WER value was cut to 12%. To the best of our knowledge, this is the lowest WER reported on a Hungarian LVCSR task.

Having results with different language and acoustic models provides a great opportunity to investigate their impact on morph-based recognition improvements.

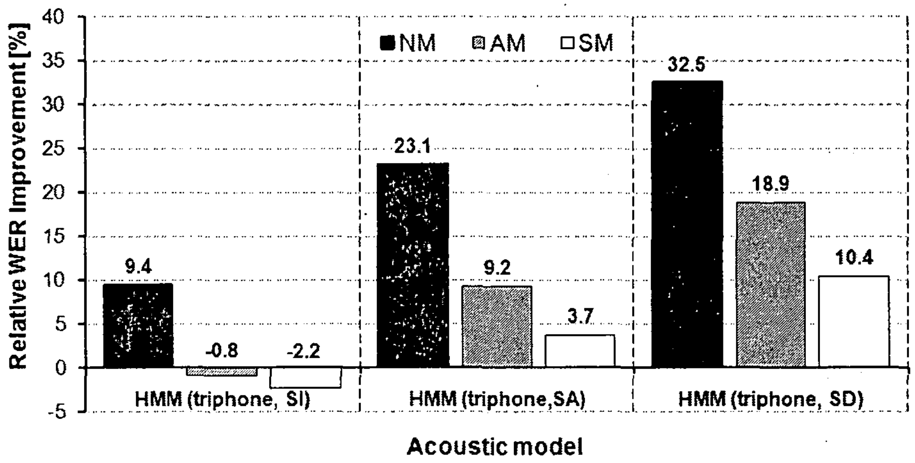

Figure 3: Relative improvements of WER due to change-over to morph-based language modeling. Improvement rates are measured using various acoustic models and training text corpora.

A commonly used metric for this is to measure the relative reduction of WER due to switching from word- to morph-based modeling (Fig 3). Looking at Figure 3 it can be seen that the fewer resources we have for training a language model the more beneficial is the morph-based approach. Furthermore, another useful conclusion is that by using a better matching acoustic model a higher improvement can be achieved. However, this also can turn into a deterioration in performance when the acoustic model is weakly-matched. These conclusions are in accord with the findings of [20].

6 Conclusions

In this paper we investigated the impact of various acoustical and language modeling techniques on the speech recognition accuracy measured on an audiobook recording. In such a near-ideal task the effect of disturbing acoustic factors (background noise, sloppy articulation, etc.) is almost negligible, thus the presented results can serve as baseline for the corresponding technologies in Hungarian continuous speech recognition. Despite the idealistic acoustic conditions, the task still represented various challenges in the language modeling step, due to the thematic and stylistic richness of the literary work the audiobook was based on.

First pure phone recognition experiments were presented, and the results clearly reflected that – at the acoustic level – the recognition of audiobooks is indeed much easier than that of a more realistic task. Our results with ANN-based acoustic models accord well with the similar international studies which indicate that these methods can attain similar or even better performance than the classic HMM-based triphone modeling technology. We also reinforced our earlier findings that triphone models do not require an explicit handling of phonological coarticulation during pronunciation modeling, while monophone models do.

In the large vocabulary recognition experiments we first measured phone recognition errors, but now using morph- and word-based language models instead of simple phone-bigrams. Since the phone-level output of our LVCSR system is not directly accessible, we had to build special recognition networks for this task. As expected, most of the large vocabulary supported recognition configurations outperformed the phone-based systems. We found that the larger the size of the training set, the higher the accuracy scores were. Comparing word- and morph-based phone error rates suggests that morphs are especially useful when the amount of the available training data is very limited. For instance, in the case of the novel-matched corpus the baseline phone-bigram phone error rate (~14%) obtained in Section 4 could be improved only by introducing morph-based language modeling. On the contrary, when a large corpus is available, well-trained word-based language models may outperform morph-based ones in terms of PER, due to the ambiguous phonetic transcriptions at morph boundaries.

Though word-based recognizers may have higher phone-level accuracies, the morph-based configurations made consistently fewer letter errors. The quality of the recognized word sequence is closely related to the letter error rate, hence a morph-based recognizer is usually a better choice for large vocabulary tasks in Hungarian. Despite the advantages of LER, the word error rate is a more widely accepted metric. Therefore we used the WER to express the benefit of switching from word- to morph-based recognition. This improvement is especially good if the acoustic model suitably matches the recognition task, or when only a small corpus is available for language model training.

Comparing the error rates in this study to some of our earlier experimental results [20] we can get an impression of how acoustic factors of speech can degrade recognition performance. On a spontaneous speech task, which was recorded in a noisy environment and was both trained and evaluated with multiple speakers, roughly 50% WER was measured. In the case of press conference speeches – thanks to the higher signal-to-noise ratio and more planned speech production – the error rate dropped to 30%. While in near-ideal conditions WER can come close to 10%, as we see in this paper. Although these results cannot be directly compared due to differences in language and acoustic model sets, the tendency clearly shows that still much research have to be done to overcome the issues of real-life recognition tasks.

Numerous technologies have been investigated in this study, but there is still space for further development. In the future we would like to combine discriminative training methods and triphone acoustic models, since both approaches resulted in a consistent phone error rate improvement. A further important research direction would be to find a way of combining the advantages of the neural net-based and the triphone technologies in acoustic modeling. For speaker adaptation, it would be worth trying a supervised MAP adaptation of the speaker-independent acoustic model, since a good quality transcription of the audiobook is available. We expect that with this technique the performance of the adapted acoustic model could get closer to the speaker-specifically trained one.

References

[1] Aradilla, G., Bourlard, H., Magimai-Doss, M. Using KL-based Acoustic Models in a Large Vocabulary Recognition Task. Proceedings of Interspeech 2008: 928-931.

[2] Bourlard, B., Morgan, N. Connectionist Speech Recognition - A Hybrid Approach. Kluwer Academic, 1994.

[3] Chen, S.F. and Goodman, J.T. An Empirical Study of Smoothing Techniques for Language Modeling. Technical Report TR-10-98, Computer Science Group, Harvard University, 1998.

[4] Creutz, M. and Lagus, K. Unsupervised Morpheme Segmentation and Morphology Induction from Text Corpora Using Morfessor 1.0. Comp. and Inf. Sci., report A81, HUT, March 2005.

[5] He, X., Deng, L. Discriminative Learning for Speech Recognition: Theory and Practice Morgan & Claypool, 2008.

[6] Hermansky, H., Ellis, D., Sharma, S. Tandem connectionist feature extraction for conventional HMM systems. Proceedings of ICASSP 2000: 1635-1638.

[7] Hirsimaki, T. and Kurimo, M. Decoder issues in unlimited Finnish speech recognition. Proceedings of the Nordic Signal Processing Symposium *NORSIG 2004*, Espoo, Finland, 2004.

[8] Huang, X., Acero, A., Hon, H.-W. Spoken Language Processing. Prentice Hall, 2001.

[9] Leggetter, C.J. and Woodland, P.C. Flexible Speaker Adaptation Using Maximum Likelihood Linear Regression. Proc. ARPA Spoken Language Technology Workshop, 1995.

[10] Lippmann, R. P. Speech Recognition by Machines and Humans. Speech Communication, 22(1): 1-15, 1997.

[11] Mauuary, L. Blind Equalization in the Cepstral Domain for robust Telephone based Speech Recognition. in Proc. of EUSPICO'98, Vol.1, pp. 359–363, 1998.

[12] Mihajlik P., Tatai P. Automatic phonetic transcription for Hungarian (In Hungarian). Beszédkutatás 2001: 172-185.

[13] Mihajlik P., Tüske Z., Tarján B., Németh B. and Fegyó T. Improved recognition of spontaneous Hungarian Speech - Morphological and Acoustic Modeling Techniques for a Less Resourced Task. in IEEE Transactions on Audio, Speech, and Language Processing Vol. 18, Issue 6, pp. 1588-1600, 2010.

[14] Mihajlik P. Coarticulation Modeling in Automatic Speech Recognition for Hungarian (In Hungarian). Proceedings of MSZNY 2006, pp. 231-242.

[15] Mohamed, A.-R., Dahl, G., Hinton, G. Deep Belief Networks for Phone Recognition. Proceedings of NIPS 2009.

[16] Mohri, M., Pereira, F. and Riley, M. Weighted Finite-State Transducers in speech Recognition. Computer Speech and Language, 16(1), pp. 69-88, 2002.

[17] Siniscalchi, S. M., Schwartz, P., Lee, C.-H. High-Accuracy Phone Recognition By Combining High-Performance Lattice Generation and Knowledge-Based Rescoring. Proceedings of ICASSP 2007, pp. 869-872.

[18] Stolcke, A. J.T. SRILM – an extensible language modeling toolkit. Proc. Intl. Conf. on Spoken Language Processing, pp. 901–904, Denver, 2002.

[19] Szarvas M., Fegyó T., Mihajlik P. and Tatai P. Automatic Recognition of Hungarian: Theory and Practice. International Journal of Speech Technology, 3:277-287, December 2000.

[20] Tarján B. and Mihajlik P. On Morph Based LVCSR Improvements. in Proc. of the 2nd Int. Workshop on Spoken Language Technologies for Under-resourced Languages, pp. 10–15, 2010.

[21] Tóth L., Frankel, J., Gosztolya G., King, S. Cross-lingual Portability of MLP-Based Tandem Features - A Case Study for English and Hungarian. Proceedings of Interspeech 2008: 2695-2698.

[22] Tóth Sz. L., Sztahó D., Vicsi K. Speech Emotion Perception by Human and Machine Proceedings of COST Action 2102 International Conference, Patras, Greece, October 29-31, 2007.

[23] Vicsi K., Tóth L., Kocsor A., Gordos G., Csirik J. MTBA - Hungarian Telephone Speech Database (In Hungarian). Híradástechnika, Vol. LVII, No.8, pp. 35-43, 2002.

[24] Vicsi K., Kocsor A., Teleki Cs., Tóth L. Speech Database for Office Computer Environments (In Hungarian) Proceedings of MSZNY 2004 (2004), pp. 315-318.

[25] Weintraub, M., Taussig, K., Hunicke-Smith, K., Snodgrass, A. Effect of speaking style on LVCSR performance. Proceedings of ICSLP 1996: 16-19.

[26] Young, S., Ollason, D., Valtchev, V. and Woodland, P. The HTK book. (for HTK version 3.4), March 2009. http://htk.eng.cam.ac.uk

[27] Zhao, Y. Speech-Recognition Technology in Health Care and Special-Needs Assistance. IEEE Signal Processing Magazine, 26(3): 87-90, 2009.

[28] Hungarian Electronic Library (Magyar Elektronikus Könyvtár). http://www.mek.oszk.hu

[29] Digital Academy of Literature (Digitális Irodalmi Akadémia). http://www.irodalmiakademia.hu

[30] Electronic Archive of Periodicals (Elektronikus Periodika Archívum és Adatbázis). http://epa.oszk.hu

Improvements of Hungarian Hidden Markov Model-based Text-to-Speech Synthesis*

Bálint Tóth[†] and Géza Németh[†]

Abstract

Statistical parametric, especially Hidden Markov Model-based, text-to-speech (TTS) synthesis has received much attention recently. The quality of HMM-based speech synthesis approaches that of the state-of-the-art unit selection systems and possesses numerous favorable features, e.g. small runtime footprint, speaker interpolation, speaker adaptation. This paper presents the improvements of a Hungarian HMM-based speech synthesis system, including speaker dependent and adaptive training, speech synthesis with pulse-noise and mixed excitation. Listening tests and their evaluation are also described.

Keywords: Hungarian HMM speech synthesis, speaker adaptation, pulse-noise excitation, mixed excitation

1 Introduction

Several TTS methods were created in the last decades, including rule based articulatory [1] and formant synthesis [2], which try to model the speech production mechanism; diphone, triphone based concatenative synthesis [3] and corpus-based unit selection synthesis [4], which are based on recordings from a speaker; and statistical parametric synthesis, which became a focused research are in the past few years.

The voice characteristics of automatic rule based articulatory and formant models can be widely modified, although the quality of these systems is not satisfactory, as the applied rules are not precise enough. Diphone and triphone based methods produce constant quality and the voice characteristics can be modified to some degree, but they still sound unnatural. Corpus-based unit selection systems produce high quality, natural sounding voice, but the quality is not constant, the voice

*The work was partially supported by the Hungarian National Office for Research and Technology (Teleauto project - OM-00102/2007, BelAmi project - ALAP2-00004/2005) and by ETOCOM project (TÁMOP-4.2.2-08/1/KMR-2008-0007) through the Hungarian National Development Agency in the framework of Social Renewal Operative Programme supported by EU and co-financed by the European Social Fund.

[†]Department of Telecommunications and Media Informatics, Budapest University of Technology and Economics, E-mail: {toth, nemeth}@tmit.bme.hu

characteristics cannot be modified and for the best quality large runtime databases are required.

In statistical parametric synthesis usually the hidden Markov Model paradigm is used [5]. It has numerous advantages compared to other methods: it has comparable voice quality to that of the state-of-the-art unit selection methods, the runtime database is small (2-10 MB) [6], the voice characteristics can be changed by speaker adaptation [7][8] and interpolation [9] and emotions can also be expressed [10].

HMM-based TTS is categorized as a kind of unit selection speech synthesis, although in this case the units are not waveform samples, but spectral and prosody parameters extracted from the waveform. HMMs are responsible for selecting those parameters which most precisely represent the text to be read and a vocoder generates the synthesized speech from these parameters. Different vocoder techniques can be applied, generally pulse-noise or mixed excitation is used (the latter has better quality, but its runtime database and computational cost is higher).

The first section of the current paper gives an overview about the architecture of HMM-based speech synthesis (that is the basis for our TTS system). It investigates the two basic training (speaker dependent, speaker adaptive) and the two basic synthesis methods (pulse-noise excitation, mixed excitation) that are applied in order to improve the systems quality. In the second part of the paper Hungarian specific solutions of the system are discussed and a listening test and its evaluation are carried out, which involves diphone-based, corpus-based unit selection and HMM-based Hungarian TTS systems.

2 HMM-based text-to-speech synthesis

Hidden Markov models are often used to simulate the behavior of physical processes based on observations. In speech technology HMMs can successfully model the behavior of human speech. Both in speech recognition and synthesis descriptive parameters of a speech corpus are used as observations, which is much more efficient than wave sample based observations. HMMs have already been applied in speech recognition for a long time [11]. In the last decade HMM-based speech synthesis became a focused research area. It differs from the method applied in speech recognition in three main parts:

- In case of speech synthesis at the last step instead of "pattern matching" "pattern selection" is executed, so the most likely parameters (e.g. spectral coefficients, pitch, state duration) are selected. Speech is generated by a vocoder from the selected parameters.
- Prosody is also modeled in speech synthesis, including pitch and phoneme durations.
- In speech synthesis a more complex acoustic model is used instead of triphones, which involves segmental and supra-segmental information. This is described by context dependent labels (see subsection 2.1.3).

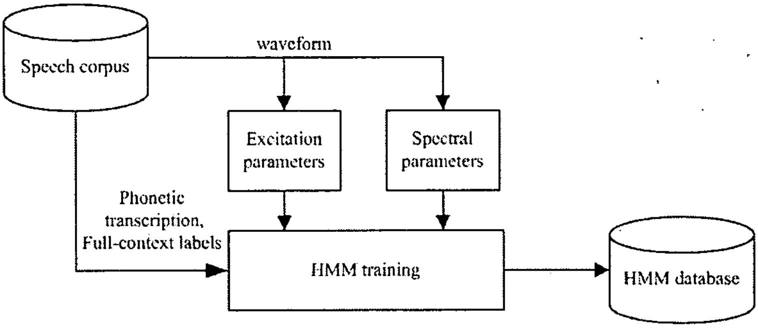

Figure 1: Block diagram of speaker dependent training.

HMM-based TTS consists of two main tasks: the training and the speech synthesis task. In the training task the HMM parameters are trained by a large, precisely labeled speech corpus. As a result a small HMM database is created, which includes the representative parameters of the speech corpus (training). From this database the best matching parameters of the text to be read are selected and the utterance is generated by a vocoder (synthesis).

2.1 Training

There are two main types of HMM training: the speaker dependent and the speaker adaptive training methods.

2.1.1 Speaker dependent training

For speaker dependent training (see Figure 1) a rather larger speech corpus (minimum 1-1.5 hours of speech) from a given speaker, the phonetic transcription and precise phoneme boundary labeling are required. The spectral parameters (e.g. features derived by linear prediction analysis), their first and second derivatives, the pitch, its first and second derivatives are extracted from the waveform.

As the next step phonetic transcriptions are extended to context dependent labels (see subsection 2.1.3.). When all these data are prepared, the training procedure is started. During training the HMMs learn the spectral and excitation parameters according to the context dependent labels of the given corpus. To be able to model parameters with varying dimensions multi-space distribution HMMs (MSD-HMMs) are used [12] (e.g. logF0 in case of voiced/unvoiced regions is modeled by 2 dimensional HMMs). To model the rhythm of the speech state duration densities are calculated for each phoneme. The set of state durations of each phoneme HMM is modeled by a multi-dimensional Gauss distribution.

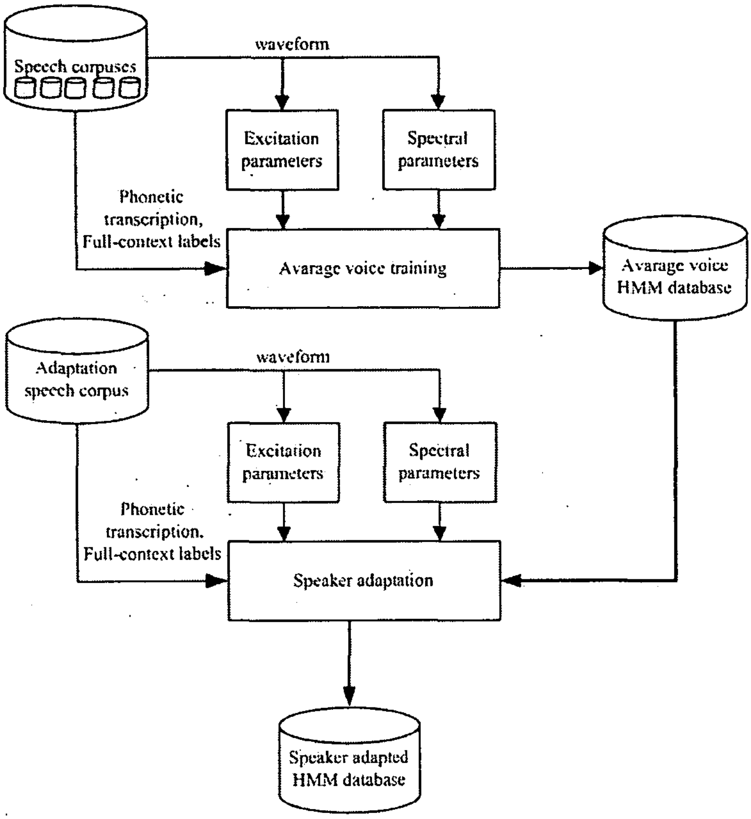

Figure 2: Block diagram of speaker adapted training.

2.1.2 Speaker adaptive training

Speaker adaptive training is divided into two parts (Figure 2). First the so called average voice must be constructed, then the average voice is adapted to the target speakers voice. For the average voice speech corpora (minimum 1-1.5 hours/speaker) from numerous speakers (minimum 4-5) is required. The excitation and spectral parameters and their first and second derivatives are extracted from the corpora. The average voice model is trained with this data and with the related phonetic transcriptions and context dependent labels. If an average voice is available, a much smaller speech corpus of 5-10 minutes is sufficient for adaptation. The same training data are extracted from the adaptation corpus for completeing the adaptation phase.

2.1.3 Context dependent labeling and decision trees

To describe the features of a phoneme in detail - to be able to select the most likely units in the synthesis phase - a number of phonetic features should be defined. These features are calculated for every sound. Labeling is done automatically, which may include errors (e.g. finding the accented syllables, defining the part of speech). This effect is likely not to influence the quality much, if the same algorithm is used in speech generation, thus the parameters are chosen by the HMMs consistently. In subsection 3.3. the features that were used in the Hungarian version of our HMM-based TTS system are described.

The combination of all possible context dependent features is a huge number. If only the possible variations of quintphones (this is a basic context dependent feature, see subsection 3.3.) are taken into account, that is over 160 million and this number increases exponentially if further context dependent features are included as well. Consequently it is impossible to design a speech corpus, which contains all combinations of context dependent features. To overcome this problem decision tree based clustering [13] is used. As different features influence the spectral parameters, the pitch values and the state durations, decision trees are separately handled for each. In subsection 3.3. the general questions used for building the decision trees in the Hungarian version of HMM-based synthesis are introduced.

2.2 Synthesis

The speech synthesis method is the same in the case of both training methods: the HMMs generate the most likely parameters (including pitch, state durations and spectral parameters) belonging to the text and then the speech is generated by a vocoder method. Depending on the type of the parameters, that were used during training, the vocoder may be a simple vocoder (e.g. LPC-10), although mixed excitation vocoders perform much better, as they significantly reduce the buzzyness of the speech. Certainly different vocoder techniques influence the choice of the parameters, that are to be extracted from the waveform, and they may also influence the training methods of the HMMs (e.g. pitch modeling requires MSD-HMMs).

In this study we have tested the two most commonly used vocoder techniques in HMM-based speech synthesis, the pulse-noise and mixed excitation vocoders.

2.2.1 Pulse-noise excitation vocoder

The pitch (voiced regions) or a binary flag (unvoiced regions), the spectral parameters and the state durations should be extracted from the speech corpus and trained for the HMMs in the pulse-noise excitation model. To be able to model voiced and unvoiced regions, MSD-HMMs are used. In the synthesis phase the excitation is modeled as periodic pulse trains at the rate of the pitch that was generated by the HMMs (voiced phonemes) or as white noise (unvoiced phonemes). This excitation signal is filtered by a Mel-Log Spectral Approximation (MLSA) filter [14] and the synthesized speech is generated (see Figure 3).

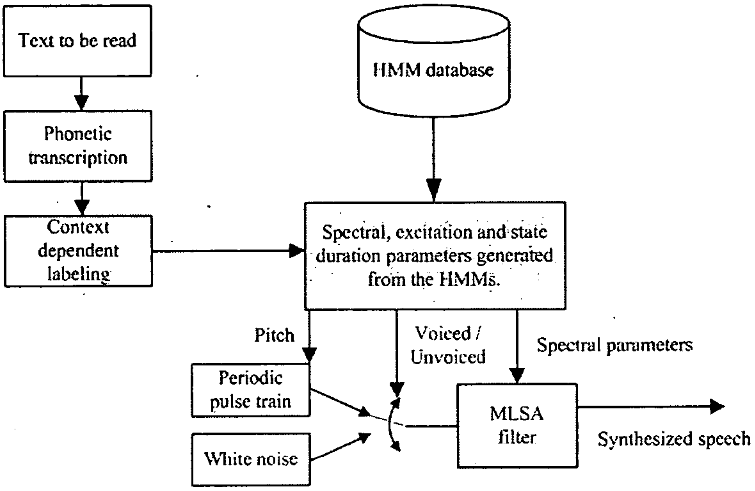

Figure 3: Block diagram of HMM TTS with pulse-noise excitation.

The advantage of pulse-noise excitation is the simplicity, furthermore a small footprint runtime database is enough and the computational cost mainly depends on the order of the MLSA filter. The main disadvantage is the buzzyness of the synthesized voice.

2.2.2 Mixed excitation vocoder

To make the synthesized voice more natural and to eliminate the buzzyness mixed-excitation vocoders were introduced [15]. In the mixed excitation model (see Figure 4) the pitch, the bandpass voicing strengths and spectral parameters are extracted and trained for the HMMs. In the synthesis phase the parameters of the bandpass filters for the periodic pulse train and for the white noise excitation are generated by the HMMs (bandpass voicing strengths). After the excitation signals passed through the bandpass filters, the results are summed and filtered by an MLSA filter. As a result the synthesized voice is generated.

The main advantage of using mixed excitation is the good, natural sounding quality, although more computational cost is required as the number and the order of filters increases. Further improvements in quality can be achieved by post filtering the synthesized voice [16].

3 Improvements of Hungarian HMM-based TTS

Several language specific steps are necessary to create a Hungarian HMM-based text-to-speech engine. The basics of a Hungarian HMM-based speaker dependent text-to-speech engine are described in [17]. In this chapter the most significant

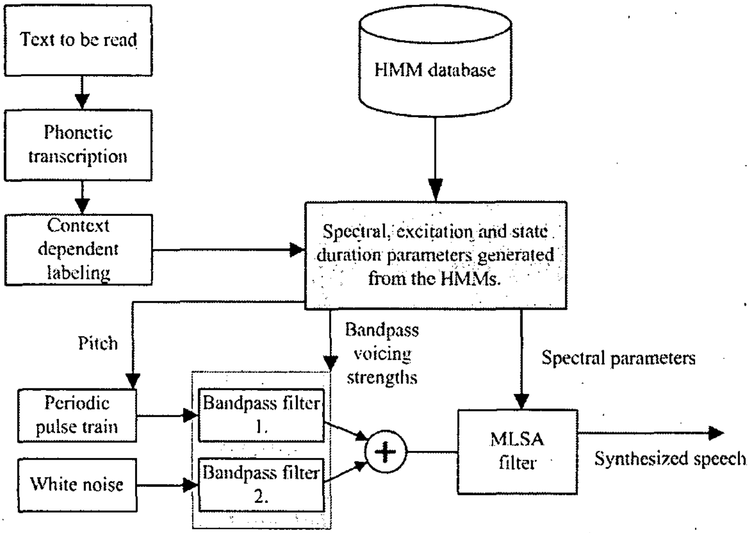

Figure 4: Block diagram of HMM TTS with mixed excitation.

issues of creating a Hungarian HMM-based speaker adapted text-to-speech system are investigated.

3.1 Speech databases

Five speech databases were recorded: four males and one female, for the average voice. The utterances are well designed, phonetically balanced sentences . The content of the utterances was manually verified. Phoneme boundaries were determined by forced alignment with a wide beam. The properties of the speech databases are shown in Table 1.

Table 1: Speech corpora for the average voice (44 kHz, 16 bit, mono format)

Speaker	Number of sentences	Duration	Size
1. male speaker	1941	170 minutes	857 MB
2. male speaker	1938	137 minutes	694 MB
3. male speaker	1944	191 minutes	966 MB
4. male speaker	1938	214 minutes	1082 MB
5. female speaker	1940	129 minutes	652 MB

For adaptation we used several different databases, including semi-spontaneous political speeches, weather forecasts, price list utterances (planned speech), and general, phonetically balanced utterances. The length of the adaptation speech databases was between 5-15 minutes. In the current paper adaptation with a general, phonetically balanced database is investigated (see section 4.) with the properties shown in Table 2.

Table 2: Speech corpus for adaptation (44 kHz, 16 bit, mono format)

Speaker	Number of sentences	Duration	Size
Female speaker	117	8 minutes	40 MB

The speech databases were resampled at a rate of 16 kHz on 16 bits and windowed by a 25 ms Hanning-window with 5 ms shift. The feature vectors consisted of 39 mel-cepstral coefficients (including the 0th coefficient), logF0, aperiodicity measures, and their dynamic and acceleration coefficients.

3.2 Adaptation technique

There are two main techniques of speaker adaptation in the HMM paradigm: maximum likelihood linear regression (MLLR) [7] and maximum a posteriori (MAP) estimation [8]. MLLR is applied when the amount of adaptation data is small, for MAP more data is required as the Gaussian distributions are updated individually.

The Hungarian version uses the MLLR adaptation method. MLLR modifies the parameters of the average voice to the target voice by linear transforms. In this case the state outputs are:

$$b_j(o_t) = \mathbb{N}(o_t; \hat{\mu}_j, \hat{\Sigma}_j) \tag{1}$$

$$\hat{\mu}_j = A_{\gamma(j)}\mu_j + b_{\gamma(j)} \tag{2}$$

$$\hat{\Sigma}_j = H^T_{r(j)}\Sigma_j H_{r(j)} \tag{3}$$

b_j corresponds to the output probability function, o_t is the observation vector, μ_j and Σ_j are the original mean vector and covariance matrix. $\hat{\mu}_j$ is the linearly transformed mean vector of the j-th state output distribution and $\hat{\Sigma}_j$ is the linearly transformed covariance matrix of the j-th state output distribution. The covariance matrix adaptation is performed after the mean vector adaptation. $A_{\gamma(j)}$, $b_{\gamma(j)}$ and $H_{r(j)}$correspond to the mean linear transformation matrix, to the bias vector and to the covariance linear transformation matrix for the r_j-th regression class.

Generally there are two types of MLLR adaptation. If A and H linear transformation matrices are the same, than we talk about constrained MLLR (CMLLR), otherwise it is unconstrained MLLR. We used CMLLR for adaptation. The state output distributions are clustered by regression class trees; in a given class we use the same transformation matrices and bias vectors. The linear transform is derived from the labeled adaptation data. In order to perform adaptation with less data, the context-dependent models with regression or decision trees are used. The complexity and generalization abilities of the adaptation can be controlled by adjusting the size of the regression-class / decision tree to the size of the adaptation data. CMLLR is the most commonly used adaptation technique, but other, more sophisticated schemes are available as well [18].

Classic speaker adaptation uses precise phonetic transcriptions, manually transcribed or automatically annotated segmentation and linguistic labels - this is called supervised speaker adaptation. In the unsupervised case the adaptation process does not require any manual interaction. The advantages of unsupervised adaptation are quite appealing: the creation of target voices becomes automatic which is favorable if several voices are required or if no pre-processing of the speech data is possible. There are some solutions for unsupervised speaker adaptation, which are introduced in [19], [20] and [21]. We also conducted some experiments of ASR transcription based unsupervised adaptation in Hungarian with promising results [22].

The gender of the average voice database speakers is an important question. If large speech corpora are available then creating gender dependent average voices is ideal. In practice only some speech corpora are available from both males and females, thus a mixed gender average voice is used often. [23] introduces a method, which causes minimal quality degradation in case of adapting a mixed gender average voice to male or to female voices, compared to the gender dependent case. As shown in Table 1 four male and one female speakers were used in our experiments for the average voice. According to some inner tests in our laboratory, there was no significant difference between adapting to male or to female voices from the average voice.

3.3 Context dependent labeling and decision trees

In 2.1.3. context dependent labels and decision trees were introduced in general. In this subsection we investigate their language specific features. Table 3 shows the context dependent labels, which were used in the Hungarian HMM-based TTS system. An example for a context dependent label looks like the following:

```
a^l-a1+bb=i@2_1/A:2_1/B:0-2@2-1&6-6$2-0;0-...
```

The questions for the decision tree building algorithm have been defined according to these features. Depending on the modeled parameter (spectral, pitch, duration) the most significant question varies, although generally the questions regarding to phonemes are dominant. These questions are determined by the behavior of the Hungarian phonemes [24]. Table 4 shows some important features that are used for the creation of the decision trees.

Figure 5 shows an example for decision trees in the case of spectral features. C_, L_ and R_ denote the central, left and right neighbouring phonemes that are under examination. The figure shows that the "Is the central phoneme in the quintphone a vowel?" was the most significant question in this case (it is on the "top" of the decision tree). On the next level there are the "Is the center phoneme a low

Table 3: The main features used by Hungarian context dependent labeling.

Sounds	The current and the two previous and the two following sounds/phonemes (quintphones). Pauses are also marked.
Syllables	Mark if the current / previous / next syllable is accented. The number of phonemes in the current / previous / next syllable. The number of syllables from / to the previous / next accented syllable. The vowel of the current syllable.
Word	The number of syllables in the current / previous / next word. The position of the current word in the current phrase (forward and backward).
Phrase	The number of syllables in the current / previous / next phrase. The position of the current phrase in the sentence (forward and backward).
Sentence	The number of syllables in the current sentence. The number of words in the current sentence. The number of phrases in the current sentence.

Table 4: The most important features used for building the decision tree.

Phonemes	Is it vowel or consonant? Is it short or long? Is it stop / fricative / affricative / liquid / nasal phoneme? Is it front / central / back vowel? Is it high / medium / low vowel? Is it rounded / unrounded vowel?
Syllable	Is it a stressed or a not stressed syllable? Numeric parameters (see Table 3).
Word	Numeric parameters (see Table 3).
Phrase	Numeric parameters (see Table 3).
Sentence	Numeric parameters (see Table 3).

vowel?" and the "Is the center phoneme unvoiced stop?" questions. The same idea is followed at lower levels.

4 Results

A modified version of the HTS framework with STRAIGHT [6] was applied for training and for generation. The speech corpora shown in Table 1 was processed to

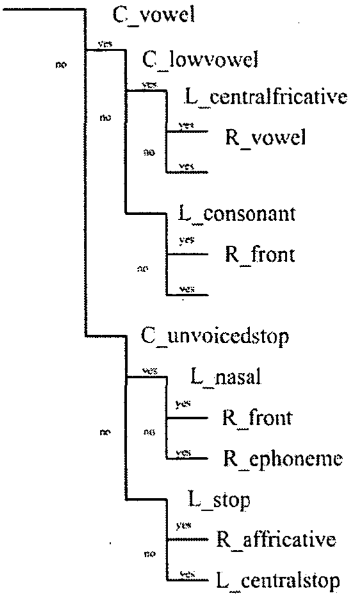

Figure 5: Example for decision trees (spectral features).

create the models of the average voice and the speech corpus in Table 2 was used for adaptation. A listening test was carried out to evaluate the quality of comparable Hungarian TTS systems.

4.1 Experimental conditions

Five TTS systems were involved in the listening test: a triphone based system (System A); a general domain corpus based unit selection system (System B); a domain specific corpus based unit selection system (System C); a HMM-based speaker adapted system with pulse-noise excitation (System D) and a HMM-based speaker adapted system with mixed excitation (System E). The original speaker (from whom the speech corpora were recorded) was the same in case of all TTS systems. The corpus based unit selection system had all the waveforms from one speaker in the runtime database. The HMM-based speech synthesis system had the waveforms in the training database from five (average voice) plus one speaker (adaptation), as it is described in 4.1. The language of the test was Hungarian. The properties of the different systems are shown in Table 5.

The listening test consisted of two parts:

- the first part was a Comparison Mean Opinion Score (CMOS) test;
- the second part was Mean Opinion Score (MOS) test.

In the first part test subjects had to decide on a five point scale from two synthesized samples which one sounds more natural. On the scale 3 meant the quality of the two samples are considered same, higher values meant that the second sample was considered more natural (4 more natural, 5 much more natural), lower values meant that the first sample was considered more natural (2 more natural, 5 much more natural). The text of the utterance in a pair was the same. In a pair different speech synthesis systems were used. Altogether 9 pairs were played; each pair was played twice (normal and inverted order). CMOS pair comparison as the first part of the test is favorable, as subjects get used to the synthetic voice and they will give consistent answers for the MOS tests in the next part. In the second section the test subjects had to mark on a five point scale the naturalness of 20 samples, 4 samples from each system. Lower values meant worse natüralness, higher values meant better naturalness. In the second section the text of the utterances was different.

We have chosen this order of the two main parts to minimize the chance that the test subjects memorize the different systems. The samples were selected from a larger set of sentences in order to get the desired information about the systems and not about the speech samples. Furthermore the samples were sorted in different pseudo-random orders for every test subject to avoid memory effects. The distribution of the samples and the systems was kept even.

The authors carried out a pre-test with five subjects to verify the effectiveness of the test design. The results of the pre-test were adequate, so the same design was kept and the results of the pre-test were also included in computing the final results.

Altogether 24 test subjects (7 female, 17 male) were involved in the test. All the test subjects were native Hungarian speakers with no known hearing loss. The test was internet-based, the average age was 32, and the youngest subject was 22, the oldest 67 years old. 7 test subjects were speech experts.

4.2 Analysis of the results

The results of the listening text are shown in Table 6 and Table 7. Table 6 contains the general preference scores of the CMOS test and the results of the MOS test. In Table 7 the particular values related to the HMM-based speech synthesis systems are shown. The results are represented in Figure 6 on boxplot diagrams according to the guidelines of [25]. On boxplot diagram systems can easily be compared by the median (black thick line), by the 1st and 3rd quartiles (bar), by the whiskers and outliers. The most significant information are the median, the 1st and the 3rd quartiles. As it was expected System A scored the worst in both parts of the test. Although the naturalness of System A is much worse then the naturalness of other systems, it has got a small footprint and its computational costs are very low, so it can be applied in low resource systems. The naturalness of System B was considered also quite low and its runtime database is large and the computational costs are also high.

System D achieved the third position. Its global preference score is almost

Table 5: Speech synthesis systems involved in the listening test

System	Technique	Training database	Runtime database
A	Triphone based unit selection	-	285 MB (triphones)
B	Corpus based unit selection (general domain)	-	4634 MB (one speaker, 44 kHz, 16 bit, mono waveforms)
C	Corpus based unit selection (domain specific)	-	3113 MB (one speaker, 44 kHz, 16 bit, mono waveforms)
D	HMM-based speech synthesis (speaker adapted, pulse-noise excitation)	4251 + 40 MB (five + one speakers, 44 kHz, 16 bit, mono waveforms)	2 MB (HMM parameters, decision trees)
E	HMM-based speech synthesis (speaker adapted, mixed excitation)	4251 + 40 MB (five + one speakers, 44 kHz, 16 bit, mono waveforms)	11 MB (HMM parameters, decision trees)

the same as the score of System B, but its general naturalness and CMOS score compared to System B are higher. In addition System B has a small runtime database.

System C and System E performed the best in the listening test. System C was considered better than System E in the pair comparison part (on Figure 6 they have the same median, but System C has higher 3rd quartile), in the general naturalness part System E was considered better. These differences are mostly not significant and the reason, why the two systems performed different in the two parts is that their naturalness is quite close to each other. The only significant difference is the median of the systems in the MOS test, where System E performs better (see Figure 6). In case of more test subjects the scores of systems C and E may get closer. However System C performed well only in a given domain with a large runtime database, System E performed the same quality on general sentences with a small runtime database.

5 Conclusions

In the current paper the basics of HMM-based speech synthesis are introduced, including speaker dependent and speaker adaptive training, furthermore two different speech generation techniques, the pulse-noise and mixed excitation based

Table 6: Results (mean ± variance) of the listening test. Higher numbers mean better naturalness.

	CMOS (Global preference score) Compared naturalness of speech synthesis systems	MOS General naturalness of the systems
System A	2.3 ± 1.14	2.1 ± 0.9
System B	2.8 ± 1.27	2.6 ± 1.1
System C	3.6 ± 1.3	3.2 ± 1.1
System D	2.9 ± 1.27	3.1 ± 1.2
System E	3.4 ± 1.22	3.5 ± 1.0

Table 7: CMOS pair comparison values for System D and System E (3 means identical naturalness, higher values mean that the system in the row was considered more natural, lower values mean that the system in the column was considered more natural)

System	A	B	C	D	E
D	3.3 ± 1.15	3.3 ± 1.3	2.7 ± 1.4	N/A	2.5 ± 1.0
E	3.9 ± 1.1	3.5 ± 1.3	2.7 ± 1.3	3.5 ± 1.0	N/A

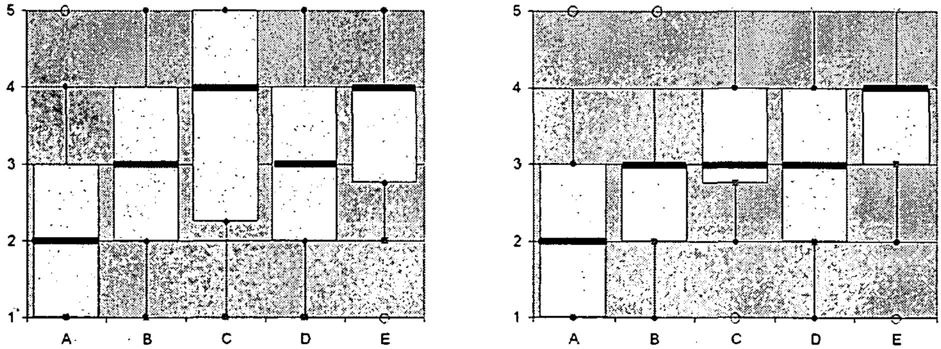

Figure 6: Boxplot showing compared naturalness of the speech synthesis systems (left) and general naturalness of the speech synthesis systems (right).

vocoders are described. The Hungarian version of a speaker adapted HMM-based speech synthesis engine was investigated, and the most important language specific features are shown. To measure the quality of the system a listening test was carried out with some Hungarian speech synthesis engines. The results showed that

HMM-based speech synthesis with mixed excitation performs with a small runtime database on general sentences like the state-of-the-art corpus-based unit selection system with a large runtime database on domain specific sentences.

In the future we plan further error corrections and more precise labeling of the training data, as it is likely to increase the quality of the synthesized voice. Additionally the solution will be optimized for embedded environments. Other voice coding algorithms will also be applied.

References

[1] P. Mermelstein. Articulatory model for the study of speech production. Journal of the Acoustical Society of America, Volume 53, 1973, pp. 1070-1082.

[2] D. H. Klatt, L. C. Klatt. Analysis, synthesis, and perception of voice quality variations among female and male talkers. The Journal of the Acoustical Society of America, Volume 87, Issue 2, February 1990, pp. 820-857

[3] D. O'Shaughnessy, L. Barbeau, D. Bernardi and D. Archambault. Diphone speech synthesis. Speech Communications, Volume 7, Issue 1, March 1988, pp. 55-65

[4] B. Möbius. Corpus-based speech synthesis: methods and challenges. Speech and Signals - Aspects of Speech Synthesis and Automatic Speech Recognition, 2000, pp. 7996

[5] A.W. Black, H. Zen, K. Tokuda. Statistical parametric speech synthesis. Proceedings of ICASSP, Apr. 2007, pp. 1229-1232

[6] J. Yamagishi, T. Nose, H. Zen, T. Toda, K. Tokuda. Performance evaluation of the speaker-independent HMM-based speech synthesis system "HTS-2007" for the Blizzard Challenge 2007. Proceedings of ICASSP 2008, Las Vegas, U.S.A, April 2008, pp. 3957-3960

[7] M. Tamura, T. Masuko, K. Tokuda, T. Kobayashi. Speaker adaptation for HMM-based speech synthesis system using MLLR. Proceedings of ESCA/COCOSDA Workshop on Speech Synthesis, November 1998, pp. 273-276

[8] K. Ogata, M. Tachibana, J. Yamagishi, T. Kobayashi. Acoustic model training based on linear transformation and MAP modification for HSMM-based speech synthesis. Proceedings of ICSLP 2006, September 2006, pp. 13281331.

[9] T. Yoshimura, T. Masuko, K. Tokuda, T. Kobayashi, T. Kitamura. Speaker interpolation for HMM-based speech synthesis system. Journal of the Acoustical Society of Japan (E), Volume 21, Issue 4, 2000, pp. 199-206

[10] T. Nose, M. Tachibana, T. Kobayashi. HMM-based style control for expressive speech synthesis with arbitrary speaker's voice using model adaptation. IEICE Trans. Inf. & Syst., Volume E92-D, Issue 3, Mar. 2009, pp. 489-497

[11] Lawrence R., Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the.IEEE, 1989, pp. 257286

[12] K. Tokuda, T. Masuko, N. Miyazaki, T. Kobayashi. Hidden markov models based on multi-space probability distribution for pitch pattern modeling. Proceedings of ICASSP-99, March 1999, pp. 229232

[13] Young, S., Ollason, D., Valtchev, V. and Woodland, P. The HTK book. (for HTK version 3.4), March 2009. http://htk.eng.cam.ac.uk

[14] S. Imai, K. Sumita, C. Furuichi. Mel log spectral approximation filter for speech synthesis. Trans. IECE, Volume J66-A, February 1983, pp. 122-129

[15] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, T. Kitamura. Mixed excitation for HMM-based speech Synthesis. Proceedings of Eurospeech, Sept. 2001, pp.2259-2262

[16] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, T. Kitamura. Incorporation of mixed excitation model and postfilter into HMM-based text-to-speechsynthesis. Systems and Computers in Japan, Volume 36, Issue 12, September 2005, pp. 4350

[17] B. Tóth, G. Németh. Hidden Markov model based speech synthesis system in Hungarian. Infocommunications Journal, Volume LXIII, no. 2008/7, 2008, pp. 3034

[18] J. Yamagishi, T. Kobayashi, Y. Nakano, K. Ogata, J. Isogai. Analysis of Speaker Adaptation Algorihms for HMM-based Speech Synthesis and a Constrained SMAPLR Adaptation Algorithm. IEEE Audio, Speech, and Language Processing, Volume 17 Issue 1, January 2009, pp. 66-83

[19] S. King, K. Tokuda, H. Zen, and J. Yamagishi. Unsupervised adaptation for HMM-based speech synthesis. Proceedings of Interspeech 2008, 2008, pp. 18691872

[20] M. Gibson, T. Hirshimaki, R. Karhila, M. Kurimo and W. Byrne. Unsupervised Cross-Lingual Speaker Adaptation for HMM-based Speech Synthesis Using Two-Pass Decision Tree Construction. Proceedings of ICASSP 2010, Dallas, USA

[21] K. Oura, K. Tokuda, J. Yamagishi, S. King, M. Wester. Unsupervised Cross-Lingual Speaker Adaptation for HMM-based Speech Synthesis. Proceedings of ICASSP 2010, Dallas, USA

[22] B. Tóth, T. Fegyó, G. Németh. Some aspects of ASR transcription based unsupervised speaker adaptation for HMM speech synthesis. 13th International Conference on Text, Speech and Dialogue, Brno, Czech Republic, September 2010

[23] J. Yamagishi, T. Kobayashi, S. Renals, S. King, H. Zen, T. Toda, K. Tokuda. Improved Average-Voice-based Speech Synthesis using Gender-Mixed Modeling and A Parameter Generation Algorithm considering GV. Proceedings of ISCA SSW6, Bonn, Germany, August 2007, pp. 125-130

[24] M. Gósy. Phonetics, the Science of Speech (in Hungarian). Budapest, Osiris, 2004, p. 350

[25] R. A. J. Clark, M. Podsiadlo, M. Fraser, C. Mayo, S. King. Statistical analysis of the Blizzard Challenge 2007 listening test results. Proceedings of Blizzard 2007 (in Proceedings of Sixth ISCA Workshop on Speech Synthesis), Bonn, Germany, August 2007, pp. 1-6

REGULAR PAPERS

Evaluating Dynamically Evolving Mobile-Based Social Networks

Péter Ekler[*], Tamás Lukovszki[†] and Hassan Charaf[‡]

Abstract

The increasing capabilities of mobile phones enable them to participate in different type of web-based systems. One of the most popular systems are social networks. The phonebooks of the mobile devices also represent social relationships of the owner. This can be used for discovering additional relations in social networks. Following this line of thought, mobile-based social networks can be created by enabling a synchronization mechanism between phonebooks of the users and the social network. This mechanism detects similarities between phonebook contacts and members of the network. Users can accept or ignore these similarities. After acceptance, identity links are formed. If a member changes her or his personal detail, it will be propagated automatically into the phonebooks, via identity links after considering privacy settings. Estimating the total number of these identity links is a key issue from scalability and performance point of view in such networks. We have implemented a mobile-based social network, called Phonebookmark and examined the structure of the network during a test period of the system. We have found, that the distribution of identity links of the users follows a power law. Based on this, we propose a model for estimating the total number of identity links in the dynamically evolving network. We verify the model by measurements and we also prove the accuracy of the model mathematically. For this we use the fact, that the number of identity links of each user (and thus, the value of the random variable modeling it) is bounded linearly by the number of members N_M of the network. Then we show, that the variance of the random variable is $\Theta(N_M^{3-\beta})$, where $2 < \beta \leq 3$ is the exponent of the bounded power law distribution, i.e. for constant $c > 0$, $\Pr[X = x] = c \cdot x^{-\beta}$, if $x \leq N_M$ and $\Pr[X = x] = 0$ otherwise. The model and the results can be used in general when the distribution shows similar behavior.

Keywords: networks, social networks, power law distribution, variance

[*] Department of Automation and Applied Informatics, Budapest University of Technology and Economics, E-mail: peter.ekler@aut.bme.hu. Supported by the project TÁMOP 4.2.1/B-09/1/KMR-2010-0002 of the National Development Plan.

[†] Faculty of Informatics, Eötvös Loránd University, E-mail: lukovszki@inf.elte.hu. Supported by the project TÁMOP 4.2.1/B-09/1/KMR-2010-0003 of the National Development Plan.

[‡] Department of Automation and Applied Informatics, Budapest University of Technology and Economics, E-mail: hassan.charaf@aut.bme.hu. Supported by the project TÁMOP 4.2.1/B-09/1/KMR-2010-0002 of the National Development Plan.

1 Introduction

In the last decade the Internet related technologies developed rapidly. One of the most popular solutions are social network sites. Since their introduction, social network sites such as Facebook, Myspace and LinkedIn have attracted millions of users.

According to new statistics [8] Facebook has more than 500 million active users, 50% of the active users log in to Facebook every day. Other statistics show that there are more than 65 million active users currently use their mobile devices for accessing Facebook. Those mobile users are almost 50% more active on Facebook than non-mobile users.

The fact, that the phonebook of the mobile device also describe social relationships of its owner, can be used for discovering additional relations in social networks. This is beneficial for sharing personal data or other content. Given an implementation that allows us to upload as well as download our contacts to and from the social networking application, we can completely keep our contacts synchronized. Besides that we can see all of our contacts on the mobile phone as well as on the web interface. In addition to that if the system detects that some of my private contacts in the phonebook is similar to another registered members of the social network (i.e. may identify the same person), it can discover and suggest social relationships automatically. Accepted similarities are called identities. In the rest of this paper we refer to this solution as a *mobile-based social network*.

If a member changes some of her or his detail, it should be propagated in every phonebook to which she or he is related. In addition to that, with the help of identity links, the system can keep the phonebooks always up-to-date. In this paper we show how to calculate the expected number of identities which is a key issue from scalability point of view and we propose a model which proves the accuracy of the calculation. The model is based on power law distribution and the results can be used in general cases as well. The results were applied in Phonebookmark project at Nokia Siemens Networks.

The rest of the paper is organized as follows. Section 2 summarizes related work in the field of dynamically evolving large networks and power law distribution. Section 3 defines mobile-based social networks. Section 4 proposes a model for estimating the total number of similarities. Section 5 proves the accuracy of the model and gives an estimation for the variance of power law distribution when the random variable has an upper bound. Finally Section 6 concludes the paper and proposes further research area.

2 Related work

Huge amount of papers and popular books, such as Barabási's Linked [2] study the structure and principles of dynamically evolving large scale networks like the Internet and networks of social interactions. In [9] the authors discuss about that nowadays social networking on mobile phones is not only a buzz term for today's enthusiasts but also provides real possibilities to the users. Many features of social processes and the Internet are governed by power law distributions. Following the terminology in [7] a nonnegative random variable X is said to have a power law distribution if $\Pr[X \geq x] = cx^{-\alpha}$, for constant $c > 0$ and $\alpha > 0$. In a power law distribution asymptotically the tails fall according to the power α, which leads to much heavier tails than other common models.

Distributions with an inverse polynomial tail have been first observed in 1896 by Pareto [12] (see. [13]), while describing the distribution of income in the population. Zipf observed similar statistical behavior in the distribution of inhabitants in cities [14].

In [4] the graph structure of the Web has been investigated and it was shown that the distribution of in- and out-degree of the web graph and the size of weekly and strongly connected components are well approximated by power law distributions. Nazir et al. [11] showed that the in- and out-degree distribution of the interaction graph of the studied social network applications also follow such distributions. Those distributions also approximate the degree distribution of the Gnutella network [13]. Crovella et al. [5] observed power law distributions in the sizes of files and transmission times in the Internet.

There has been a great deal of theoretical work on designing random graph models that result in a Web-like graph. Barabási and Albert [3] describe the preferential attachment model, where the graph grows continuously by inserting nodes, where new node establishes a link to an older node with a probability which is proportional to the current degree of the older node. Bollobás et al. [4] analyze this process rigorously and show that the degree distribution of the resulting graph follow a power law. Another model based on a local optimization process is described by Fabrikant et al. [7]. Mitzenmacher [10] gives an excellent survey on the history and generative models for power law distributions. Aiello et al. [1] studies random graphs with power law degree distribution and derives interesting structural properties in such graphs.

In our work power law distribution was also discovered in case of the number of identities. We have given a model for estimating the total number of identities and we have proven the accuracy of the model, where the variance of power law distribution was examined.

3 Mobile-based social network

Mobile-based social networks rely on the well-known social network sites, they have a similar web interface, but they add several major mobile phone-related functions to the system. Next we consider social networks as graphs. In case of general social networks, nodes are representing registered members and the edges between them represent the social relationships (e.g. friendship). Then we should notice that each member has a private mobile phone with a phonebook (Figure 1). In Figure 1, we can also observe that phonebook contacts are connected to the mobile devices "owned" by different members.

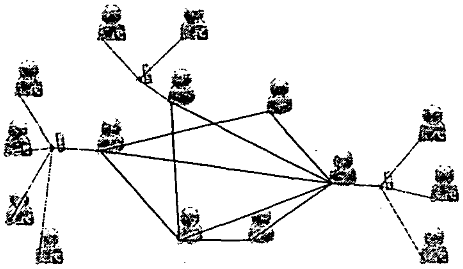

Figure 1: Basic structure of mobile-based social networks

One of the key advantages of mobile-based social networks is that they allow real synchronization between private phonebook contacts and the social network. For this a similarity detecting algorithm is needed. This algorithm is able to compare two person entries (members and private contacts, too) and determine whether they are likely similar, if so, it also proposes a probability to this detected similarity.

Figure 2 shows the graph structure when the similarity detecting algorithm has finished comparing the relevant person entries.

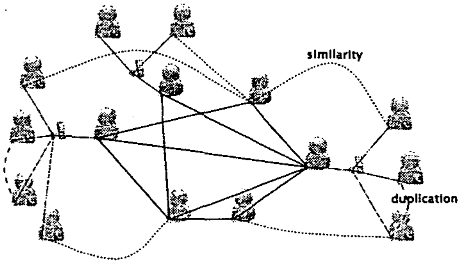

Figure 2: Detected similarities and duplications

In Figure 2 the dotted edges between members and private contacts represent detected similarities and broken lines between two private contacts illustrate possible duplications in the phonebooks. Duplications are detected as a positive side effect of the similarity detecting algorithm.

After the similarities and duplications are detected there is a semi-automatic step, the members who have phonebook contacts detected as similar to other members in the network have to decide whether detected similarities are the correct ones. In addition to that, members can also decide about the correctness of detected duplications in their phonebooks. Figure 3 shows the graph structure after some of the members have resolved the detected similarities and duplication. It can be observed that one of the private contacts of the most left member has been deleted. The other duplication link still remained on the right side because that member has not decided about it yet.

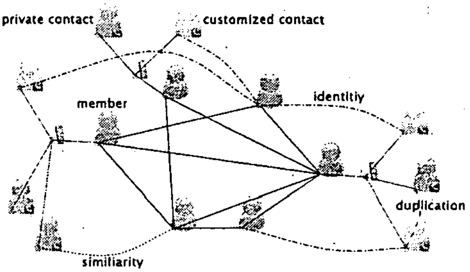

Figure 3: Resolving similarities and duplications

Figure 3 shows an example, where four of the five similarities were resolved (members found them correct) and there is still one in the system (the member has not decided yet). Resolving a similarity means that an identity link is being formed between the private contact in one's phonebook and the relevant member who represents the same person in the system. The private contacts that are linked to members via this type of identity links are called customized contacts. One of the key advantages of mobile-based social networks are these identity link, since if a member changes her or his personal detail on the web user interface (adds a new phone number, uploads a new image, changes the website address, etc), it will be automatically propagated to those phonebooks where there is a customized contact related to this member, after considering privacy issues. Additional important advantages of mobile-based social networks are:

- Private contacts can be managed (list, view, edit, delete, etc.) from a browser.
- Similarity detecting algorithm detects duplicate contacts in the phonebooks and warns about it.
- Private contacts are safely backed up in case the phone gets lost.
- Private contacts can be easily transferred to a new phone if the user replaces the old one.
- Phonebooks can be shared between multiple phones, if one happens to use more than one phone.
- It is not necessary to explicitly search for the friends in the service, because it notices if there are members similar to the private contacts in the phonebooks and warns about it.

The described mobile-based social network architecture was actually applied in the *Phonebookmark* project at Nokia Siemens Networks. Phonebookmark covered a wide range of mobile phones with the Symbian and Java ME clients. We took part in the implementation and before the public introduction it was available for a group of general users from April to December of 2008. It had 420 registered members with more than 72000 private contacts, which is a suitable number for analyzing the behavior of the network. During this period we have collected and measured different types of data related to the social network and its behavior.

We would like to highlight that the proposed mobile-based social network architecture extends the general social networks. Based on this, existing, large systems can be upgraded easier to involve mobile phones in their operation. This also indicates that it is important to examine such solutions from the performance and scalability point of view.

4 Expected value in power law based models

The additional resource requirements of mobile-based social networks depend at most from the number of identity links compared to general social networks, because the number of synchronizations depends on them. In this section we propose a model for calculating the total number of identities.

4.1 Distribution of similarities

Based on the database and database logs of Phonebookmark we managed to measure the distribution of similarities raised by a member during registration and phonebook synchronization.

Figure 4 shows the complementary cumulative distribution function of the number of similarities, where the x-axis is the number of similarities and the y-axis means how many people arises at least that amount of similarities when register and synchronize.

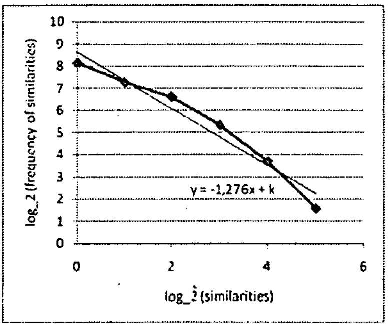

Figure 4: Distribution of similarities

We use logarithmically scaled x- and y-axis. We can see in Figure 4 that the points can be well approximated with a straight line by the least squares method, thus the distribution of similarities can be well approximated by a power law. The exponent of the power law distribution is $\alpha = 1.276$.

According to this measurement the distribution of similarities in our case can be well approximated as follows:

$$\Pr[X \geq x] = x^{-1.276} \tag{1}$$

The evidence that the distribution of similarities follows a power law has practical consequences. The expected number of members involving at least a certain number of similarities x can be estimated by $N_M \Pr[X \geq x] = N_M x^{-1.276}$, where N_M is the number of members in the network.

4.2 Calculating the expected value

Based on the previous measurement we can estimate the total number of identities.

Theorem 1. *The number of identities N_I in a mobile-based social network can be well approximated with $N_I = N_M \frac{\varsigma(\beta-1)}{\varsigma(\beta)} P_R$, where N_M represents the number of members in the system, P_R is the acceptance rate of the similarities by the users, $\varsigma(.)$ denotes the Riemann Zeta function, $\beta = \alpha + 1$ and $\alpha > 1$ is the parameter of the distribution.*

Remark 1. $P_R \approx 0.9$ is the accuracy rate of the similarity detecting algorithm discussed in [6]. The proof is based on measurements, the importance relies on that the number of similarities has not been modeled before. The trend can be seen very well on this data set.

Proof. (Theorem 1)

We model the number of similarities generated during a member registration by a random variable X. More precisely, X models the number of similarities related to a member. First we calculate the expected number of similarities:

$$E[X] = \sum_{x=1}^{\infty} x \Pr[X = x] \tag{2}$$

Note that x starts from one, because a new member registration involves at least one similarity, because the system allows registration only by invitation, therefore the new member is already in the phonebook of the inviting member.

Then the total number of identities N_I in a mobile-based social network can be calculated with the following formula:

$$N_I = N_M E[X] P_R. \tag{3}$$

In order to calculate $E[X]$, we need to determine $\Pr[X = x]$, which can be obtained from (1) by derivation:

$$\Pr[X = x] = c' \frac{1}{x^\beta}, \tag{4}$$

where $\beta = \alpha + 1$. In order to have a probability distribution,

$$\sum_{x=1}^{\infty} c' x^{-\beta} = 1. \tag{5}$$

Therefore,

$$c' = \frac{1}{\sum_{x=1}^{\infty} x^{-\beta}} = \frac{1}{\varsigma(\beta)}, \tag{6}$$

where $\varsigma(.)$ denotes the Riemann Zeta function.
Then the expected value can be calculated as:

$$\begin{aligned} E[X] &= \sum_{x=1}^{\infty} x \Pr[X = x] && (7) \\ &= \sum_{x=0}^{\infty} x \frac{1}{\varsigma(\beta)} x^{-\beta} && (8) \\ &= \frac{1}{\varsigma(\beta)} \sum_{x=1}^{\infty} x^{1-\beta} && (9) \\ &= \frac{\varsigma(\beta - 1)}{\varsigma(\beta)}. && (10) \end{aligned}$$

Finally, based on (3), the expected total number of identities N_I in a mobile related social network can be estimated with the following formula:

$$N_I = N_M \frac{\varsigma(\beta - 1)}{\varsigma(\beta)} P_R \tag{11}$$

For $\beta > 2$, $\varsigma(\beta - 1)/\varsigma(\beta)$ is a constant.

□

During the operational period of Phonebookmark 1088 identities were detected. By applying the identity estimation model in Theorem 1, for $\beta = 2.276$, we obtain that the expected total number of identities is $N_I = 2.9196 \cdot 420 \cdot 0.9 = 1103$, which is very close to the measured number.

Since mobile-based social networks are new type of social networks, we could not perform the measurements on other databases. However in the next section we prove the accuracy of the identity model mathematically. This result can be used widely in other similar cases, since power law distribution occurs often in social networks and the Web-graph.

5 Variance of power law distribution

In the identity estimation model in Section 4 we used a random variable X which represents the number of similarities raised by a member and we showed that X

follows a power law distribution. For $\alpha \leq 2$, a power law distribution has infinite variance. Thus, for $1 < \alpha \leq 2$, the accuracy of the estimation in Theorem 1 is an issue. The law of large numbers states that the total number of identities converges to their expected value. For this, an assumption of finite variance of the variables is not necessary. However, in order to obtain much faster convergence and error probability bound, finite variance is needed. In case of finite variance also the central limit theorem can be applied.

In this section we show that the random variable X has a relevant upper bound (in our case, this upper bound is linear in the number of the members of the network). This can be used to calculate an accurate variance value, if $1 < \alpha \leq 2$. After that the central limit theorem can be used in order to obtain, that the total number of identities will be close to their expected value.

Following we highlight that identities raised by a member have a relevant upper bound, then we propose and prove a general theorem to calculate the variance of upper bounded power law distributions. Finally, we show how to apply it in case of mobile-based social networks.

Fact: If the phonebooks do not contain duplicates then the number of similarities caused by a member is at most $2(N_M - 1)$.

With other words, in the interval $[1, 2(N_M - 1)]$ the distribution of similarities follows a power law and the probability of more similarities is zero. In order to see this, note that a member can be similar to at most one private contact of each of the other $N_M - 1$ members and, for each private contact, there is at most one similar member in the network.

We show that similarities resulting from this fact has a finite variance.

Theorem 2. *Let X be a random variable with* $\Pr[X = x] = c \cdot x^{-\beta}$ *if $x \leq n$ and* $\Pr[X = x] = 0$ *otherwise, where $2 < \beta \leq 3$ and $c > 0$ is a constant. In this case the variance $\sigma^2 X$ of X is $\sigma^2 X = \Theta\left(n^{3-\beta}\right)$.*

For the proof we use two lemmatas.

Lemma 1. *Let X be a random variable with* $\Pr[X = x] = c \cdot x^{-\beta}$ *if $x \leq n$ and* $\Pr[X = x] = 0$ *otherwise, where $2 < \beta \leq 3$ and $c > 0$ is a constant. In this case the variance is $\sigma^2 X = O\left(n^{3-\beta}\right)$.*

Proof. From the *Steiner formula*, the variance is calculated as $\sigma^2 X = E[X^2] - (E[X])^2$. $E[X]$ was defined previously. For $\beta > 2$, $E[X]$ is a finite constant, i.e. $E[X] = \Theta(1)$. Thus we only need to calculate $E[X^2]$. By definition:

$$
\begin{aligned}
E[X^2] &= \sum_{x=1}^{\infty} x^2 \Pr[X = x] && (12)\\
&= \sum_{x=1}^{n} x^2 \Pr[X = x] && (13)\\
&= \sum_{x=1}^{n} x^2 c x^{-\beta} && (14)\\
&= c \sum_{x=1}^{n} x^{2-\beta}, && (15)
\end{aligned}
$$

where $c = \frac{1}{\sum_{x=1}^{n} x^{-\beta}}$ and $\Pr[X = x] = cx^{-\beta}$. Then $\sum_{x=1}^{\infty} \Pr[X = x] = 1$.

Let $y = \frac{1}{c}E[X^2]$. Following we show an upper estimation for y.. In order to do so, we create an upper estimation model for the function of y by using the powers of 1/2. Let $z = 2^{\frac{1}{2-\beta}}$, then:

$$
\frac{1}{c} z^2 \Pr[x = z^i] = \frac{1}{\left(2^{i\frac{1}{2-\beta}}\right)^{\beta-2}} = \frac{1}{2^i} \qquad (16)
$$

Figure 5 illustrates how we performed the estimation, with the $f1$ function.

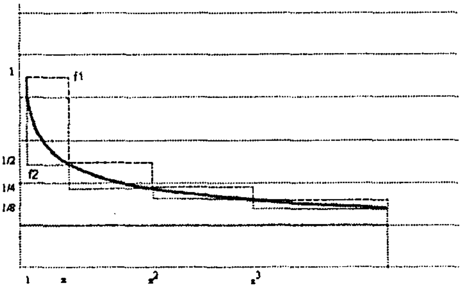

Figure 5: Staged estimation function

Now we are able to give an upper bound on y.

$$\begin{aligned}
y &\le \sum_{i=0}^{\log_z n} \left(z^{i+1} - z^i\right)2^{-i} &(17)\\
&= \sum_{i=0}^{\log_z n} (z-1)z^i 2^{-i} &(18)\\
&= (z-1)\sum\nolimits_{i=0}^{\log_z n} \left(\frac{z}{2}\right)^i &(19)\\
&= (z-1)\left(\frac{\left(\frac{z}{2}\right)^{\log_z n+1} - 1}{\frac{z}{2}-1}\right) &(20)\\
&= (z-1)\left(\frac{\frac{z}{2}n^{\frac{1}{\log_{z/2} z}} - 1}{\frac{z}{2}-1}\right) &(21)\\
&= (z-1)\left(\frac{\frac{z}{2}n^{\frac{1}{1+\log_{z/2} 2}} - 1}{\frac{z}{2}-1}\right). &(22)
\end{aligned}$$

The explanation to the last step:

$$\log_{z/2} z = \log_{z/2} 2\frac{z}{2} = 1 + \log_{z/2} 2 \tag{23}$$

To continue, first we have to check the following calculation. Remember that $z = 2^{\frac{1}{2-\beta}}$. Then

$$\log_{z/2} 2 = \frac{\log_2 2}{\log_2 z/2} = \frac{1}{\log_2\left(\frac{2^{\frac{1}{2-\beta}}}{2}\right)} = \frac{1}{\frac{1}{\beta-2} - 1} = \frac{\beta-2}{3-\beta} \tag{24}$$

Therefore,

$$n^{\frac{1}{1+\log_{z/2} 2}} = n^{\frac{1}{1+\frac{\beta-2}{3-\beta}}} = n^{3-\beta} \tag{25}$$

This way y looks as:

$$y \le (z-1)\left(\frac{\frac{z}{2}n^{\beta-3} - 1}{\frac{z}{2}-1}\right) \tag{26}$$

Next we show that the variance by applying the *Steiner formula* and the previous calculations is $O(n^{3-\beta})$.

$$\begin{aligned}
\sigma^2 X &= E[X^2] - (E[X])^2 && (27)\\
&\leq \frac{1}{c} y - \Theta(1) && (28)\\
&= \frac{1}{c}(z-1)\left(\frac{\frac{z}{2}n^{\beta-3}-1}{\frac{z}{2}-1}\right) - \Theta(1) && (29)\\
&\leq \frac{1}{c}\frac{(z-1)\,z}{z-2} n^{3-\beta} - \Theta(1) && (30)\\
&= O\left(n^{3-\beta}\right). && (31)
\end{aligned}$$

□

Lemma 2. *Let X be a random variable with $\Pr[X = x] = c \cdot x^{-\beta}$ if $x \leq n$ and $\Pr[X = x] = 0$ otherwise, where $2 < \beta \leq 3$ and $c > 0$ is a constant. In this case the variance is $\sigma^2 X = \Omega\left(n^{3-\beta}\right)$.*

Proof. We use the notations of the previous lemma. We give a lower bound on y using function $f2$ is shown on Figure 5.

$$y \geq \sum_{i=0}^{\log_z n} \left(z^{i+1} - z^i\right) 2^{-(i+1)}, \qquad (32)$$

which is the half of the upper bound given in (17). Then by following the steps of the proof of the previous lemma we obtain that

$$y \geq \frac{z-1}{2}\left(\frac{\frac{z}{2}n^{\beta-3}-1}{\frac{z}{2}-1}\right) = \Omega(n^{3-\beta}). \qquad (33)$$

Therefore,

$$\sigma^2 X = \Omega(n^{3-\beta}). \qquad (34)$$

□

Proof. (Theorem 2) The proof is straightforward by applying Lemma 1-2: $\sigma^2 X = \Theta\left(n^{3-\beta}\right)$, because $\sigma^2 X = O\left(n^{3-\beta}\right)$ and $\sigma^2 X = \Omega\left(n^{3-\beta}\right)$.

□

Theorem 2 can be applied in case of mobile-based social networks, when X represents the number of similarities raised by a member and the upper bound is $n = 2(N_M - 1)$.

6 Conclusion and future work

Social network sites are becoming more and more important in everyday life. Phonebook-centric social networks enable to manage online and mobile relationships within one system. The key mechanism of such networks is a similarity handling algorithm which detects similarities between members of the network and phonebook entries.

The number of identities is a key parameter from scalability point of view. In this paper we have shown how to calculate the expected number of identities and we have proven the accuracy of that calculation. However the results can be used generally in case of power law distributions where the random variable has an upper bound.

Further work includes additional measurements and extending the model with phonebook duplication handling.

References

[1] Aiello, W., Chung, F. R. K., and Lu, L. A random graph model for massive graphs. In *Proc. 32nd Symposium on Theory of Computing STOC*, pages 171–180, 2000.

[2] Barabási, A.-L. Linked: How everything is connected to everything else. *Perseus Publishing*, 2002.

[3] Barabási, A.-L. and Albert., R. Emergence and scaling in random networks. *Science*, 286:509–512, 1999.

[4] Bollobás, B., Riordan, O., Spencer, J., and Tusnady, G. Random structures and algorithms. *IEEE Internet Computing Journal*, 18:279–290, 2001.

[5] Crovella, M. E., Taqqu, M. S., and Bestavros, A. Heavy-tailed probability distributions in the world wide web. *In: R. J. Adler, R. E. Feldman, M. S. Taqqu (eds.), A Practical Guide To Heavy Tails 1.*, pages 3–26, 1998.

[6] Ekler, P. and Lukovszki, T. Similarity distribution in phonebook-centric social networks. In *Proceedings of 5th International Conference on Wireless and Mobile Communications (ICWMC 2009)*, Cannes, France, 2009.

[7] Fabrikant, A., Koutsoupias, E., and Papadimitriou, C. H. Heuristically optimized trade-offs: A new paradigm for power laws in the internet. In *Proceedings of the 29th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 110–122, 2002.

[8] Facebook statistics. http://www.facebook.com/press/info.php?statistics, August 2010.

[9] Forstner, B. and Kelnyi, I. *Mobile Peer to Peer A Tutorial Guide*, chapter Mobile Social networking - Beyond the Hype, pages 161–190. Number ISBN 978-0-470-69992-8. Wiley, 2009.

[10] Mitzenmacher, M. A brief history of generative models for power law and lognormal distributions. *Internet Mathematics*, 1:225–251, 2001.

[11] Nazir, Atif, Raza, Saqib, and nee Chuah, Chen. Unveiling facebook: A measurement study of social network based applications.

[12] Pareto, V. Course d'economie politique profess a l'universit de lausanne. 3, 1896.

[13] Ripeanu, M., Foster, I., and Iamnitch, A. Mapping the gnutella network: Properties of large-scale peer-to-peer systems and implications for system design. *IEEE Internet Computing Journal*, 6:50–57, 2002.

[14] Zipf, G. K. Human behavior and the principle of least effort. *Addison-Wesley*, 1949.

Received 13th August 2010

Acta Cybernetica 19 (2010) 749–764.

The Extended Analog Computer and Functions Computable in a Digital Sense

Monika Piekarz*

Abstract

In this paper we compare the computational power of the Extended Analog Computer (EAC) with partial recursive functions. We first give a survey of some part of computational theory in discrete and in real space. In the last section we show that the EAC can generate any partial recursive function defined over $\mathbb{N}$. Moreover we conclude that the classical halting problem for partial recursive functions is an equivalent of testing by EAC if sets are empty or not.

Keywords: analog computation, Extended Analog Computer, recursion theory, recursive functions

1 Introduction

People have always sought some models which could help us to explain and model various aspects of Nature. Hence, the need for machines which could simulate some aspects of the physical world in order to understand and model it appeared in a natural way. Several computational models seemed to perform this task, but nowadays discrete models play the main role. Nevertheless, computers need not to be digital. In fact, the first computers were analog computers where the internal states are continuous rather than discrete which is the case in digital computers. Studies of these machines began long ago with the machines of V. Bush (see [2]) which suited well to solve ordinary differential equations and were effectively used to solve many military problems during World War II (see [10]) and work of C. Shannon (see [21]). Later, in the 60s and 70s, this subject lost its importance, but from the 80s onwards a renewed interest of the study on analog computation can be observed. This stems partly from the search for new models which could provide an adequate notion of computation and from the complexity of the dynamical systems that are currently used to model the physical world.

Notwithstanding, the present knowledge about analog computation still leaves many unsolved questions. Although the study of analog computation began long

*Institute of Mathematics, University of Maria Curie-Sklodowska, Lublin, Poland, E-mail: monika.piekarz@poczta.umcs.lublin.pl

ago many fundamental questions have been analyzed relatively recently and many of them still remain unsolved.

Two different families of models of analog computation, that come from different behavior of computation processes in time, appear in the theory of analog computation. The first family contains models of computation on real numbers but in discrete time. These are, for example, the machines of Blum-Shub-Smale (see [1]) and Analog Recurrent Neural Network (see [22]). The second one are the models of computation on real numbers and in continuous time. One of the important model seems to be General Purpose Analog Computer proposed by C. Shannon in 1941 (see [21]), and took over by M. B. Pour-El (see [15]), L. Lipshitz and L. A. Rubel (see [11]). This model is able to generate all differentially algebraic functions, and it is built from some analog units connected together. Recently, many authors discussed the GPAC (see [3], [4], [7], [9]). There exists also an extension of this model called Extended Analog Computer (EAC). This model was defined by L. A. Rubel (see [20]) in 1993 and was proposed as a model of the human brain.

This work focuses on the comparison of a computational power of the EAC and recursive functions over $\mathbb{N}$. This subject is related to the Rubel's question ([20]) whether the EAC can be simulated by a digital computer and whether the digital computer can be simulated by an EAC. Solution of similar problems in the context of GPAC can be found in [18] and [8].

2 Preliminaries

The fundamental notion of partial recursive functions plays a significant role in the classical theory of computability (main notation taken from [14]). These are the functions which map $\mathbb{N}$ into $\mathbb{N}$ and which can be thought of as an equivalent to the class of functions computable in an intuitive sense. In fact, in the theory of computability it is shown that the recursive functions are precisely the functions that can be computed by Turing machines. First let us establish some notation. Lower letters: k, n, m will denote natural numbers, x, y, z real numbers and $\bar{k}, \bar{n}, \bar{m}$ sequences of natural numbers ($\bar{x}, \bar{y}, \bar{z}$ respectively sequences of real numbers).

The class $\mathcal{PRF}$ of partial recursive functions can be defined as follows.

Definition 2.1. *The class of partial recursive functions ($\mathcal{PRF}$) defined over $\mathbb{N}$ is the smallest class of functions:*

- *contains $\mathcal{O}(n) = 0$, $\mathcal{S}(n) = n + 1$, $\mathcal{I}_k^i(n_1, \ldots, n_k) = n_i$ $(1 \leq i \leq k)$*
- *closed under composition, i. e. the schema that for given $g_1, \ldots, g_l : \mathbb{N}^k \to \mathbb{N}, f : \mathbb{N}^l \to \mathbb{N}$, $(k, l > 0)$ produces*

$$h(\bar{n}) = f(g_1(\bar{n}), \ldots, g_l(\bar{n})), \quad h : \mathbb{N}^k \to \mathbb{N},$$

where $\bar{n} = n_1, \ldots, n_k$, and the left side is undefined when at least one of the values of $g_1, \ldots, g_l, f$ for the given arguments is undefined,

- *closed under primitive recursion, i. e. the schema that for given* $f : \mathbb{N}^{n+2} \to \mathbb{N}, g : \mathbb{N}^n \to \mathbb{N}$ *produces* $h : \mathbb{N}^{n+1} \to \mathbb{N}, (n > 0)$

$$h(\bar{n}, 0) = g(\bar{n}),$$

$$h(\bar{n}, m+1) = f(\bar{n}, m, h(\bar{n}, m)),$$

- *closed under unrestricted* μ*-recursion, i. e. the schema that for given* $f : \mathbb{N}^{n+1} \to \mathbb{N}$ *produces* $h : \mathbb{N}^n \to \mathbb{N}, (n > 0)$

$$h(\bar{n}) = \min_m((\forall j < m)(f(\bar{n}, j) \downarrow \wedge f(\bar{n}, j) \neq 0) \wedge f(\bar{n}, m) = 0),$$

  i. e. $h(\bar{n})$ *is the smallest* m *such that* $f(\bar{n}, m) = 0$ *and* $h(\bar{n})$ *is undefined if there is no such* m.

The operation of unrestricted μ-recursion is undefined when $(\forall m) f(\bar{n}, m) \neq 0$ or $(\exists m) f(\bar{n}, m) = 0$ but for certain $j < m$, $f(\bar{n}, j)$ is undefined. If $h(\bar{n})$ is defined by unrestricted μ - recursion we will simply write $h(\bar{n}) = \mu_m f(\bar{n}, m)$. To receive only total functions, the above definition should be modified. Within the operation of μ-recursion it is required for the function f to be regular which means f is total and $(\forall \bar{n})(\exists m) f(\bar{n}, m) = 0$. Such definition gives the class of total recursive functions. For simplicity, we will write "recursive function" instead of "total recursive function".

In the rest of the paper the notions of recursive sets and recursively enumerable sets will be used. It is said that a set S of natural numbers is recursive if its characteristic function is recursive. The characteristic function is understood as the function defined by:

$$c_S(\bar{n}) = \begin{cases} 0 & \bar{n} \in S \\ 1 & \bar{n} \in \neg S \end{cases}$$

where $\neg S$ denotes the complement of S.

It is said that a set S of natural numbers is recursively enumerable if S is the range of a partial recursive function.

Let us recall that the graph G_f of f is a set (a relation) defined in the following way: $G_f(\bar{n}, m) \Leftrightarrow f(\bar{n}) = m$.

Let us take a few useful results from [14].

Proposition 2.1. *Let* f *and* g *be, respectively, a partial and a total function. Then:*

- f *is partial recursive if and only if its graph is recursively enumerable,*
- g *is recursive if and only if its graph is recursive.*

Proposition 2.2. *A set* S *is recursive iff it is a recursively enumerable set and its complement is a recursively enumerable set too.*

Let $\mathcal{P}[\bar{n}, \bar{m}, \mathbb{Z}]$ be the ring of polynomials in the (infinite denumerable) set of unknowns with coefficients in the set $\mathbb{Z}$ of integers[1].

Let us present here some facts about recursive enumerable sets (taken from [14]).

Theorem 2.1. *Let $p(\bar{n}, \bar{m}) \in \mathcal{P}$ be a polynomial with the $(k+l)$ unknowns, $k, l \in \mathbb{N}$, where $n \in \mathbb{N}^k, m \in \mathbb{N}^l$. The set D (called a Diophantine set) defined in the following way* [2]

$$\langle \bar{n} \rangle \in D \Leftrightarrow (\exists(\bar{m}))p(\bar{n}, \bar{m}) = 0$$

is recursively enumerable and conversely every recursively enumerable set S is a Diophantine set.

More information about the above result can be found in [12] or [6]. It is worth mentioning here that Theorem 2.1 negatively solves Hilbert's Tenth Problem to decide effectively whether a given Diophantine equation: $p(\bar{n}, \bar{m}) = 0$ has a solution or not.

3 The General Purpose Analog Computer (GPAC)

In the two following sections some part of the theory of analog computation will be recalled. We start with the basic continuous-time model of analog computation known as General Purpose Analog Computer (GPAC). The GPAC was introduced in 1941 by C. Shannon (see [21]) as a mathematical model of an analog device called the Differential Analyzer (see [2]). Many variants of the Differential Analyzer was used from the 1930s to the early 60s to solve numerical problems. The Differential Analyzer may be seen as a circuit build of interconnected analog units. The units used by GPAC can be listed as follows:

- *Integrator*: a unit with a setting for initial condition: two constants a and t_0; two inputs: unary functions f, g; one output: the Riemann-Stieltjes integral $\int_{t_0}^{t} f(x)dg(x) + a$.

- *Constant multiplier*: a unit associated with a real number c with one input: function f; one output: cf.

- *Adder*: a unit with two inputs: functions f, g; one output: $f + g$.

- *Multiplier*: a unit with two inputs: functions f, g; one output: fg.

- *Constant function*: a unit with no input; one output: always equals 1.

[1] Sometimes for brevity we will write $\mathcal{P}$.

[2] Equation of the form $p(\bar{n}, \bar{m}) = 0$ is called Diophantine equation.

The GPAC model proposed by Shannon in [21] has further been refined in [11], [15], [7], [9]. Shannon, in his original paper, claimed that a unary function can be generated by GPAC if and only if the function is differentially algebraic, i. e. if it satisfies the condition the following definition:

Definition 3.1. *A unary function $f(x)$ is differentially algebraic ($\mathcal{DA}$) on the interval I if there exist some natural number n (where $n > 0$) and some $(n+2)$-ary nonzero polynomial P with real coefficients such that*

$$P(x, f(x), f'(x), \ldots, f^{(n)}(x)) = 0,$$

for every $x \in I$.

Shannon's definition of the GPAC was refined by M. B. Pour-El (see [15]) and L. Lipshitz, L. A. Rubel (see [11]). It is worth to notice that there are some functions such as the Euler Γ-function, $\Gamma(x) = \int_0^\infty t^{x-1}e^{-t}dt$, that cannot be generated by the GPAC because they are not differentially algebraic functions (compare [21]). However, it is a classical result in computable analysis that Γ is computable. So, it might seem that the GPAC is a less powerful model than computable analysis when "real time" computation is used for the GPAC. But in [9] D. S. Graça and J. F. Costa introduce some useful notion of the GPAC called a FF-GPAC. They refine the GPAC in terms of circuits to avoid problematic cases, showing that their model is equivalent to solutions of polynomial ODE's.

Definition 3.2. *Consider a GPAC U with n integrators $\overline{U}_1, \ldots, \overline{U}_n$. Suppose that to each integrator $\overline{U}_i$, $i = 1, \ldots, n$, we can associate two linear circuits* [3]*, A_i and B_i, with the property that the integrand and the variable of integration inputs of $\overline{U}_i$ are connected to the outputs of A_i and B_i respectively. Suppose also that each input of the linear circuits A_i and B_i is connected to one of the following: the output of an integrator or to an input unit. U is said to be a feedforward GPAC (FF-GPAC) iff there exists an enumeration of the integrators of U, $U_1, \ldots, U_n$, such that the variable of integration of k-th integrator can be expressed as*

$$c_k + \sum_{i=1}^{m} c_{ki}x_i + \sum_{i=1}^{k-1} \bar{c}_{ki}y_i, \quad \textit{for all } k = 1, \ldots, n,$$

where y_i is the output of U_i, for $i = 1, \ldots, n$, x_j is the input associated to the j-th input unit, and $c_k, c_{kj}, \bar{c}_{ki}$ are suitable constants, for all $k = 1, \ldots, n$, $j = 1, \ldots, m$ and $i = 1, \ldots, k-1$, where m is a number of input units used in U.

This definition describes the GPAC with some restrictions of the feedforward. So we get a model which is "well-behaved" when compared with the GPAC proposed by Pour-El.

[3] A limear circuit is an acyclic GPAC build only with adders, constant multipliers, and constant function units. If $x_1, \ldots, x_n$ are the inputs of a linear circuit, then the output of the circuit will be $y = c_0 + c_1x_1 + \cdots + c_nx_n$, where $c_0, c_1, \ldots, c_n$ are appropriate constants.

Other refinements of the notion of the GPAC and the GPAC-computability were made by D. S. Graça in [7]. He showed that if we do not compute in "real time" (which is usual way of computing for the GPAC) but in "limit time", then both the Euler Γ-function and Riemann ζ-function become computable. This result was generalized in [3] where it is shown that, on compact intervals, any function computable in the computable analysis sense can be computed by a GPAC using "limit time". In [7] Graça stated that, if f is generated by such GPAC, then f is computable by Rubel's Extended Analog Computer [20].

4 The Extended Analog Computer (EAC)

The topic of the section will be the mathematical model of analog computation proposed by L. A. Rubel in 1993 and called the Extended Analog Computer (EAC) (see [20]), which does not correspond to any existing device. Now, as a result of over a decade's of research we have some kind of implementation of Rubel's Extended Analog Computer model (see [13]) given by J. Mills. But this implementation is neither identical to the EAC model, nor is a complete implementation of the EAC model.

Rubel's EAC was introduced to expand the scope of the General Purpose Analog Computer (GPAC) (see [17]). The EAC works on a hierarchy of levels. It has no inputs from the outside, but it has a finite number of "settings", which are arbitrary real numbers (we don't put any restrictions of these real numbers, they have to be neither rational nor digitally computable i. e. approximable by Turing machine). At each level we have black boxes which produce real constants and independent variables $x_1, x_2, \ldots$. The outputs of the machine at level $(n-1)$ can be used as inputs at level n or any higher level. The inputs and outputs of levels are functions of a finite number of independent variables which are defined on some sets. Every function f produced by the EAC is associated with some set Λ on which it is defined, thus we have an ordered pair (f, Λ). At the lowest level 0, the EAC produces real polynomials of a finite number of real variables. However, at level 1 it produces all differentially algebraic functions of a finite number of real variables.

In general the EAC can generate a function on level n which is built of functions generated on level $(n-1)$ by the following operations: addition, multiplication, composition, inversion, differences, analytic continuation, solving a differential equation with boundary value conditions and limit taking.

The EAC can produce on a half of levels certain sets Ω in Euclidean space too. The sets can be obtained as non-negative or positive part of domain of some function f produced on the previous level. Moreover on the same half of levels unions, intersections or projections of such sets can be produced.

Before we give the definition of the EAC, let us recall some useful notation concerning analytic function. We say that f is in a class $C^{\omega}(\Lambda)$ and write $f \in C^{\omega}(\Lambda)$ if there is an extension $\tilde{f}$ of f to an open supersubset $\tilde{\Lambda}$ of Λ, and $\tilde{f}$ is real-analytic on $\tilde{\Lambda}$. By real-analytic functions we mean that these functions are locally sums of convergent power series.

Definition 4.1. [4] *The* EAC *machine can produce: function $f \in C^\omega(\Lambda)$ defined on Λ on the level n, $(f,\Lambda) \in \mathrm{EAC}_n$ where $n \in \mathbb{N}_0$ and $\Lambda \in \mathrm{EAC}_{n-\frac{1}{2}}$; or set $\Omega \subseteq \mathbb{R}^k$ on the level $n+\frac{1}{2}, \Omega \in \mathrm{EAC}_{n+\frac{1}{2}}$ where $n \in \mathbb{N}_0$ if the following conditions hold:*

1. *For $n = 0, (f, \mathbb{R}^k) \in \mathrm{EAC}_0$*

$$f(\bar{x}) = \sum_{(\alpha_1,\alpha_2,\ldots,\alpha_k)\in A} c_{\alpha_1\alpha_2\ldots\alpha_k} \prod_{i=1}^{k} x_i^{\alpha_i}$$

where finite $A \subset \mathbb{N}_0^k$ and $c_{\alpha_1\alpha_2\ldots\alpha_k}$ are fixed real constants.

2. *For finite $n > 0$, $(f,\Lambda) \in EAC_n$ is defined by one of the following methods:*

- *$f(\bar{x}) = g_1(\bar{x}) + g_2(\bar{x})$, where $(g_1,\Lambda),(g_2,\Lambda) \in \mathrm{EAC}_{n-1}$;*

- *$f(\bar{x}) = g_1(\bar{x})g_2(\bar{x})$, where $(g_1,\Lambda),(g_2,\Lambda) \in \mathrm{EAC}_{n-1}$;*

- *$f(\bar{x}) = h(g_1(\bar{x}),\ldots,g_l(\bar{x}))$, where $(h,\Omega),(g_1,\Lambda),\ldots,(g_l,\Lambda) \in \mathrm{EAC}_{n-1}$;*

- *$f(\bar{x}) = f_i(\bar{x})$ for some $i = 1,2,\ldots,l$, where $f_1(\bar{x}), f_2(\bar{x}),\ldots,f_l(\bar{x})$ [5] are the $C^\omega(\Lambda)$-functions which are the solutions of*

$$\begin{cases} g_1(\bar{x}, f_1, f_2, \ldots, f_l) = 0 \\ g_2(\bar{x}, f_1, f_2, \ldots, f_l) = 0 \\ \ldots \\ g_l(\bar{x}, f_1, f_2, \ldots, f_l) = 0 \end{cases}$$

where $(g_1,\Gamma),(g_2,\Gamma),\ldots,(g_l,\Gamma) \in \mathrm{EAC}_{n-1}$, $\Gamma \in EAC_{n-\frac{3}{2}}$;

- *$f(\bar{x}) = Dg(\bar{x})$, where $Dg = \frac{\partial^{\alpha_1+\alpha_2+\cdots+\alpha_k}}{\partial x_1^{\alpha_1}\partial x_2^{\alpha_2}\ldots\partial x_k^{\alpha_k}} g(\bar{x})$ are the partial derivatives and $(g,\Lambda) \in \mathrm{EAC}_{n-1}$;*

- *$f = \tilde{f}_{|\Lambda}$ [6] where $\Lambda \subset \tilde{\Lambda}$ and $(\tilde{f},\tilde{\Lambda}) \in \mathrm{EAC}_n$;*

- *$f(\bar{x}) = h(\bar{x})$ if $(h,\Lambda) \in \mathrm{EAC}_n$ is an analytic continuation [7] of $\tilde{h}$ from $\Lambda \cap \tilde{\Lambda}$ to all of Λ, where $(\tilde{h},\tilde{\Lambda}) \in \mathrm{EAC}_n$ is defined on $\tilde{\Lambda}$ and $\Lambda \cap \tilde{\Lambda} \neq \emptyset$;*

[4] This definition is the formalized version of the definition of the EAC proposed by L. A. Rubel in [20]

[5] It is required for these functions to be well-defined C^ω-functions on Λ. For example the equation $xy - 1 = 0$ has the solution $y = \frac{1}{x}$ which is not well-defined on $\mathbb{R}$ (because it is not defined for $x = 0$) but it is well-defined on the intervals $(-\infty, 0)$ and $(0, \infty)$. So $y = \frac{1}{x}$ is not EAC computable on $\mathbb{R}$ but is EAC computable on $(-\infty, 0)$ or on $(0, \infty)$.

[6] $f : \Lambda \to \mathbb{R}$ and $f(\bar{x}) = \tilde{f}(\bar{x})$ for all $\bar{x} \in \Lambda$.

[7] We understand the analytic continuation as in [23].

- *$f(\bar{x})$ is a solution of equations*

  $$F_i(\bar{x} : f, f^{(\alpha_1)}, f^{(\alpha_2)}, \dots, f^{(\alpha_l)}) = 0,$$

  for $i = 1, \dots, k$ on a set Λ which are subject to certain boundary values requirement [8] *where $F_i \in \mathrm{EAC}_{n-1}$ and $f^{(\alpha_1)}, f^{(\alpha_2)}, \dots, f^{(\alpha_l)}$ denote some partial derivatives of f;*

- *for all $\bar{x}_0 \in \Lambda$,*

  $$f(\bar{x}_0) = \lim_{\substack{\bar{x} \to \bar{x}_0 \\ \bar{x} \in \Gamma}} g(\bar{x})$$

  and

  $$\frac{\partial^{\alpha_1+\alpha_2+\cdots+\alpha_k}}{\partial x_1^{\alpha_1} \partial x_2^{\alpha_2} \dots \partial x_k^{\alpha_k}} f(\bar{x}_0) = \lim_{\substack{\bar{x} \to \bar{x}_0 \\ \bar{x} \in \Gamma}} \frac{\partial^{\alpha_1+\alpha_2+\cdots+\alpha_k}}{\partial x_1^{\alpha_1} \partial x_2^{\alpha_2} \dots \partial x_k^{\alpha_k}} g(\bar{x})$$

  where $(g, \Gamma) \in \mathrm{EAC}_{n-1}$ and Λ is a subset of $\partial\Gamma$ (is the edge of Γ);

3. *For $n + \frac{1}{2}, n \in \mathbb{N}_0, \Omega \in \mathrm{EAC}_{n+\frac{1}{2}}$*

   - *Ω is the set $\{\bar{x} \in \Lambda : f(\bar{x}) > 0\}$ or the set $\{\bar{x} \in \Lambda : f(\bar{x}) \geq 0\}$ where $(f, \Lambda) \in \mathrm{EAC}_n$;*

   - *$\Omega = \Omega_1 \cap \Omega_2$ or $\Omega = \Omega_1 \cup \Omega_2$ where $\Omega_1, \Omega_2 \in \mathrm{EAC}_{n+\frac{1}{2}}$;*

   - *$\Omega = \{\bar{x} : (\exists x \in \mathbb{R})(x, \bar{x}) \in \Omega_1\}$ where $\Omega_1 \in \mathrm{EAC}_{n+\frac{1}{2}}$.*

Figure 1 presents an example of how we can get an EAC on the level $n + 1$ where functions $g_1, \dots, g_l$ are not necessarily different from $f_1, \dots, f_m$. In this Figure we obtain functions $f_1, \dots, f_m$ on the level n, then from these functions we can compute a set on EAC level $n + 1/2$ by some operations O_s. Farther these functions and the set can be used as inputs for the level $n+1$ and by the operation O_f we obtain some function as output of EAC level $n + 1$. Where O_s denotes operation on sets described in point 3 of Definition 4.1 and O_f denotes operation on functions described in point 2 of Definition 4.1.

Moreover, there is an additional requirement for the EAC. The machine is required to produce unique outputs that are close on a compact set to the original

[8] For example: $f = f_0$ on a piece γ_0 of the boundary of Λ, and only functions $f_0 \in \mathrm{EAC}_{n-1}$ is used.

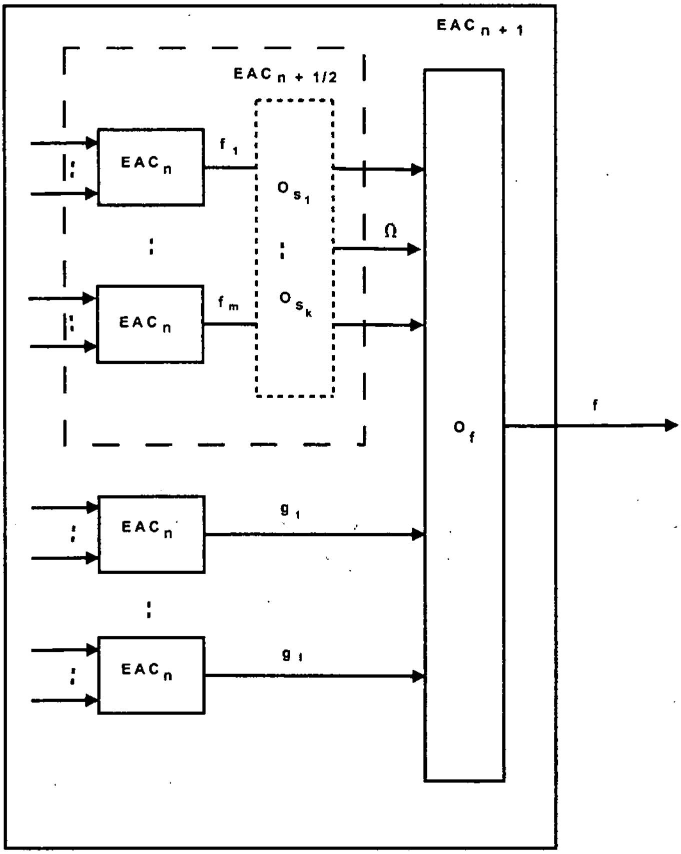

Figure 1: Extended Analog Computer — model of levels

unique output, in the case when the inputs are slightly deviated from the initial setting.

The Extend Analog Computer defined above has some interesting qualities. Some of them will be presented now. The results are taken from [20]. These machines can compute all the functions which can be computed by the GPAC. On level 1 the EAC can compute all differentially algebraic functions from C^ω. On the higher levels of the EAC such functions as the Euler Γ-function or the Riemman ζ-function can be obtained. We also know that a solution to Dirichlet problem for the Laplace's equation in the unit disc can be computed by the EAC.

Example 4.1 (Euler Γ-function). Let us consider the Euler Γ-function defined by:

$$\Gamma(x) = \int_0^\infty t^x e^{-t} \frac{dt}{t}.$$

To show how the EAC can generate this function we introduce the following function:

$$\Gamma^*(x) = \int_1^\infty t^x e^{-t} \frac{dt}{t}$$

The fact the EAC can compute $\Gamma^*(x)$ will be presented below and the other part $\int_0^1 t^x e^{-t} \frac{dt}{t}$ of the integral $\Gamma(x)$ can be handled similarly. Let

$$f(x,y) = \int_{t=1}^{t=y} t^x e^{-t} \frac{dt}{t},$$

$$g(x,y) = f(x, \frac{1}{y}),$$

$$h(x) = \lim_{y \to 0} g(x,y),$$

where limits of partial derivatives of g are well-behaved. Now we have to show that $h(x)$ is EAC computable. Since $t^{x-1}e^{-t}$ is differentially algebraic so it can be generated by the EAC. Since $g(x,y)$ is a solution of the following differential equation with boundary values:

$$g(x,1) = 0 \quad \text{for all} \quad x > 0, \quad \partial_y g(x,y) = y^{x-1} e^{-y}(-\tfrac{1}{y^2}),$$

it is an EAC computable function. Now we can put the limit to compute $h(x)$ by an EAC.

In a similar manner one can show the Riemman ζ-function can be computed by an EAC.

5 Richardson's results and the EAC

Let us start with selected Richardson's results from [16] and some results due to N. C. A. da Costa, F. A. Doria from [5]. We start with the definition and some lemmas from [16]. Let $\mathcal{A}[x_1, x_2, \ldots, \mathbb{R}]$ be an algebra of supplementary functions in the variables $x_1, x_2, \ldots$ over the real numbers $\mathbb{R}$ which corresponds to the set of expressions representing those functions. We construct algebra $\mathcal{A}[x_1, x_2, \ldots, \mathbb{R}]$ as follows:

1. All real numbers belong to $\mathcal{A}$,
2. $x_i, \sin(x_i), \exp(x_i) \in \mathcal{A}$,
3. If $f, g \in \mathcal{A}$ then $f + g \in \mathcal{A}$ and $fg \in \mathcal{A}$,
4. If $f, g \in \mathcal{A}$ then $(f \circ g) \in \mathcal{A}$, the symbol $\circ$ denotes the composition of functions,
5. $\mathcal{A}$ is the smallest algebra closed under the above conditions.

Let us present some useful results, see [16].

Lemma 5.1. *$\mathcal{A}$ is closed under partial derivatives.*

Lemma 5.2. *If $f \in \mathcal{A}$, then there is $g \in \mathcal{A}$ so that:*

$$(\forall \bar{x} \in \mathbb{R}^n) g(\bar{x}) > 1 \text{ and}$$

$$(\forall \bar{x} \in \mathbb{R}^n, \bar{\Delta} \in \mathbb{R}^n)(|\Delta_i| \leq 1) \rightarrow (g(\bar{x}) > |f(\bar{x} + \bar{\Delta})|).$$

As the conclusion from Lemma 5.1 and Lemma 5.2 we can obtain the following lemma.

Lemma 5.3. *There is a constructive procedure such that for a given expression for $p \in \mathcal{P}$ one can obtain the expressions for functions $k_i \in \mathcal{A}$ satisfying the following condition: if $|\Delta_i| \leq 1, i = 1, \ldots, n$, then*

$$k_i(\bar{m}, \bar{x}) > |\partial_i(p^2(\bar{m}, \bar{x} + \bar{\Delta}))|, \tag{1}$$

where $\bar{m} \in \mathbb{N}^k$ and $\bar{x} \in \mathbb{R}^n, \bar{\Delta} \in \mathbb{R}^n$, $k, n \in \mathbb{N}$.

Now we can add some observation regarding the algebra $\mathcal{A}$ in the context of the Extended Analog Computer. By using only the definition of the algebra $\mathcal{A}$,, we can give our lemma which connects the EAC with $\mathcal{A}$.

Lemma 5.4. *The* EAC *can compute all functions from $\mathcal{A}$.*

Proof. Elementary functions like $\exp(x)$ and $\sin(x)$ are differentially algebraic and therefore are EAC computable. By the definition, the EAC is closed under addition, multiplication and composition. So the EAC can generate all functions from $\mathcal{A}$. □

Directly from Lemma 5.3 and Lemma 5.4 we obtain the following fact.

Remark 5.1. All functions k_i from Lemma 5.3 are EAC computable functions.

Now let us recall another useful notion from [16].

Definition 5.1. *For given polynomial $p \in \mathcal{P}$, and k_i as in Lemma 5.3 let us define:*

$$f(\bar{m}, \bar{x}) = (n+1)^4 [p^2(\bar{m}, \bar{x}) + \sum_{i=1}^{n} (\sin^2 \pi x_i) k_i^4(\bar{m}, \bar{x})],$$

where $\bar{m} \in \mathbb{N}^k$ and $\bar{x} \in \mathbb{R}^n$, $k, n \in \mathbb{N}$.

It can be easily observed that $f(\bar{m}, \bar{x})$, as the composition of EAC computable functions, is the EAC computable, too. The following result is proved in [16].

Theorem 5.1. *For p and f defined as above, the following conditions are equivalent: for every $\bar{m} \in \mathbb{N}^k$:*

1. *There are natural numbers $x_1, \ldots, x_n$ such that $p(\bar{m}, x_1, \ldots, x_n) = 0$.*

2. *There are nonnegative real numbers* $x_1, \dots, x_n$ *such that* $f(\bar{m}, x_1, \dots, x_n) = 0$.
3. *There are nonnegative real numbers* $x_1, \dots, x_n$ *such that* $f(\bar{m}, x_1, \dots, x_n) \leq 1$.

The additional function $\hat{f}(\bar{m}, x_1, \dots, x_n) = f(\bar{m}, x_1^2, \dots, x_n^2)$ will be introduced and used. It is easy to obserwe $\hat{f}$ is EAC computable too.

For the purpose of creating one-argument functions the following construction is used in [16]. Let $r(y) = y\sin(y)$, and $s(y) = y\sin(y^3)$. Then for given $\hat{f}(\bar{m}, x_1, \dots, x_n)$, the following substitutions are made:

$$\begin{aligned} x_1 &= r(y), \\ x_2 &= (r \circ s)(y), \\ x_3 &= (r \circ s \circ s)(y), \\ &\vdots \\ x_{n-1} &= (r \circ \underbrace{s \circ \cdots \circ s}_{n-2})(y), \\ x_n &= (\underbrace{s \circ s \circ \cdots \circ s}_{n})(y). \end{aligned} \tag{2}$$

Finally, we obtain

$$g(\bar{m}, y) = \hat{f}(\bar{m}, r(y), r(s(y)), \dots, r(s(s(\dots s(y)) \dots)), s(s(s(\dots s(y)) \dots))),$$

where g is defined on $\mathbb{R}$ and with values in $\mathbb{R}$, $\bar{m} \in \mathbb{N}^k$.

The above functions s and r are the EAC computable, so this construction can be done by the EAC, as the composition of EAC computable functions.

Now as a consequence of Theorem 5.1 we can prove the most important corollary for the main results.

Corollary 5.1. *For every* $\bar{m} \in \mathbb{N}^k$ *the following conditions are equivalent:*

1. *There are natural numbers* $x_1, \dots, x_n$ *such that* $p(\bar{m}, x_1, \dots, x_n) = 0$.
2. *There is a real number* y *such that* $g(\bar{m}, y) \leq 1$.

Moreover, p *and* g *are* EAC *computable.*

Proof. From the introduction to Theorem Two presented in [16] it is known that for any real numbers $y_1, \dots, y_n$, and any $\delta > 0$, there is a real number y such that:

$$\begin{aligned} &|r(y) - y_1| < \delta \\ &|r(s(y)) - y_2| < \delta \\ &\vdots \\ &|r(\underbrace{s(s(\cdots s(y))\cdots))}_{n-2} - y_{n-1}| < \delta \\ &\underbrace{s(s(s(\cdots s(y))\cdots))}_{n} = y_n. \end{aligned} \tag{3}$$

We first consider the second condition of our corollary. So, by the equality

$$g(\bar{m}, y) = \hat{f}(\bar{m}, r(y), r(s(y))), \dots, r(s(s(\dots s(y))\dots), s(s(s(\dots s(y))\dots)),$$

and by the cited result (3) this condition holds iff there exist real numbers $y_1, \dots, y_n$ for which $\hat{f}(\bar{m}, y_1, \dots, y_n) \leq 1$.
Now $\hat{f}(\bar{m}, y_1, \dots, y_n)$ was defined as $f(\bar{m}, y_1^2, \dots, y_n^2)$, so there is the equivalent condition: there exist nonnegative real numbers $x_1, \dots, x_n$ for which $f(\bar{m}, x_1, \dots, x_n) \leq 1$. Next, it follows from Theorem 5.1 that the fact that there exist nonnegative real numbers $x_1, \dots, x_n$ for which $f(\bar{m}, x_1, \dots, x_n) \leq 1$ is equivalent to the statement that there exist natural numbers $x_1, \dots, x_n$ for which $p(\bar{m}, x_1, \dots, x_n) = 0$. So, we finally get:

$$(\exists y \in \mathbb{R}) g(\bar{m}, y) \leq 1 \equiv (\exists (x_1, \dots, x_n) \in \mathbb{N}^n) p(\bar{m}, x_1, \dots, x_n) = 0.$$

It is easy to observe that p ang g are EAC computable. Indeed, p is the polynomial and from the construction of g we see that g is EAC computable too. □

6 Main results

In this section we present our main results of this paper. We prove that the EAC can generate real functions which extend all partial recursive functions defined on $\mathbb{N}$.

The following theorem can be proved using facts quoted in the preliminaries and the previous two sections.

Theorem 6.1. *Every recursively enumerable set S can be generated by the* EAC.

Proof. Let D be recursively enumerable set. So D is also a Diophantine set. Then there exists a polynomial p for which the following condition holds:

$$x \in D \equiv (\exists z \in \mathbb{N}) p(x, z) = 0.$$

By Corollary 5.1, there exists an EAC computable function $g(x, y)$ such that

$$x \in S \equiv (\exists y \in \mathbb{R}) g(x, y) \leq 1,$$

where the set $S \subseteq \mathbb{R}$ has the following property of being identically with D on $\mathbb{N}$

$$(\forall n \in \mathbb{N})(n \in S \Leftrightarrow n \in D).$$

Let us present details of construction S by EAC. If

$$h(x, y) = 1 - g(x, y)$$

then the EAC can generate by Definition 4.1 the following set

$$S' = \{(x, y) \in \mathbb{R} : h(x, y) \geq 0\}$$

and also on the same level the set

$$S = \{x : (\exists y \in \mathbb{R})(x,y) \in S'\}.$$

Now we construct the set $\mathbb{N}$ of natural numbers as an intersection of N_1 and N_2, where

$$N_1 = \{x \in \mathbb{R}_+ : \sin(2x\pi) \geq 0\},$$
$$N_2 = \{x \in \mathbb{R}_+ : -\sin(2x\pi) \geq 0\}$$

and $\mathbb{R}_+ = \{x \in \mathbb{R} : x \geq 0\}$. The function $\sin(2x\pi)$ is the EAC computable. So by Definition 4.1, the set $\mathbb{N}$ can be obtained by the EAC. Finally, the set D is simply the intersection $S \cap \mathbb{N}$. □

Because every recursive set is recursively enumerable we get the following corollary.

Corollary 6.1. *Every recursive set S can be generated by the* EAC.

Theorem 6.2. *Let f be a partial recursive function defined over $\mathbb{N}$. Then there exists a family $(M_n)_{n\in\mathbb{N}}$ of* EAC*s such that for each $n \in \mathbb{N}$ M_n generates a singleton $\{f(n)\}$ if $f(n)\downarrow$ or $\emptyset$ if $f(n)\uparrow$.*

Proof. For every partial recursive function the set of pairs $\{(n, f(n)) : n \in \mathbb{N}, f(n)\downarrow\}$ is recursively enumerable. It follows from the proof of Theorem 6.1 that the EAC can construct set G_f of pairs of real numbers such that for all pairs of natural numbers if $(a,b) \in G_f$, then $b = f(a)$ and moreover $\{(n, f(n)) : n \in \mathbb{N}, f(n)\downarrow\} \subseteq G_f$.

Let n be a given natural number. We construct M_n which generates a singleton $\{f(n)\}$ in the following manner. First we build the functions $(x-n)$ and $(n-x)$ and next the sets:

$$\Omega_n^1 = \{(x,y) \in \mathbb{R}^2 : x - n \geq 0\}$$

and

$$\Omega_n^2 = \{(x,y) \in \mathbb{R}^2 : n - x \geq 0\}.$$

Then on the same level we obtain the set

$$\Omega_n = \{(x,y) \in \mathbb{R}^2 : x - n = 0\}$$

as an intersection $\Omega_n^1 \cap \Omega_n^2$. Finally, as the intersection of sets G_f and Ω_n we obtain the singleton $\{(n, f(n))\}$ if $f(n)\downarrow$ or empty set if $f(n)\uparrow$ and on the same level by projection of set $G_f \cap \Omega_n$ we can obtain in M_n set $\{f(n)\}$ if $f(n)\downarrow$ or empty set if $f(n)\uparrow$. □

The above theorem shows us that the halting problem for partial recursive functions (i. e. "is the partial recursive function defined for given n or is not defined") can be reduced in the EAC version to the question: "is the set empty?". However, we must remember that we discuss not the unique EAC but the whole family of EAC machines in Theorem 6.2, so the exact connection between halting problem and the emptiness of sets is a little more sophisticated.

Moreover we know now that for any partial recursive function f defined over $\mathbb{N}$ and for each $n \in \mathbb{N}$ we can obtain by EAC a singleton $\Lambda \doteq \{f(n)\}$ if $f(n) \downarrow$ or $\Lambda = \emptyset$ if $f(n) \uparrow$. Let i be the identity function defined over $\mathbb{R}$. So we can compute on EAC function $i_{|\Lambda}$ and obtain value $f(n)$ if $f(n) \downarrow$.

7 Conclusions

The above result implies that for any recursively enumerable sets there exists the EAC machine which generates it, and therefore the value of every partial recursive function at a given point can be obtained by the EAC (i. e. using the identity function on the singleton domain). It still remains unsolved whether for a partial recursive function f defined on $\mathbb{N}$ the EAC can generate a function $\tilde{f}$ defined on $\mathbb{R}$ such that $(\forall n \in \mathbb{N})\tilde{f}(n) \approx f(n)$. Moreover we have seen that classical halting problem is an equivalent to answer by EAC on the question: "is the set empty?".

References

[1] Blum, L., Shub, M., Smale, S.:*On a Theory of Computational and Complexity over the Real Numbers: NP-completeness, Recursive Functions and Universal Machines*, Bull. Amer. Math. Soc. (NS), Volume 21,1-49, 1989.

[2] Bush, V.: *The Differential Analyzer. A New Machine for Solving Differential Equations*, J. Franklin Institute, Volume 212, 447-488, 1931.

[3] Bournez, O., Campagnolo, M. L., Graça, D. S., Hainry, E.: *Polynomial Differential Equations Compute All Real Computable Functions on Computable Compact Intervals.* Journal of Complexity, 23(3), 317-335, 2007.

[4] Campagnolo, M., Moore, C., Costa, J. F., *Iteration, Inequalities, and Differentiabiity in Analog Computers.* Journal of complexity, 16(4), 642-660, 2000.

[5] da Costa, N. C. A., Doria, F. A.: *Undecidability and Incompleteness in Classical Mechanics*, International J. Theoret. Physics, Volume 30, 1041-1073, 1991.

[6] Davis, M., Matijasevich, Y. Robinson, J.: *Hilbert's Tenth Problem. Diophantine Equations: Positive Aspects of a Negative Solution*, Proc. Symp. Pure Math., 28, 323-378, 1976.

[7] Graça, D. S.:*Some Recent Developments on Shannon's General Purpose Analog Computer*, Math. Log Quart., Volume 50(4-5), 473-485, 2004.

[8] Graça, D. S., Campagnolo, M., Buescu, J.: *Computability with Polynomial Differential Equations.* Applied Mathematics, 40(3), 330-349, 2008.

[9] Graça, D. S., Costa, J. F.:*Analog Computers and Recursive Functions over the Reals*, J. Complexity, Volume 19(5), 644-664, 2003.

[10] Holst, P. A.: *Svein Rosseland and the Oslo Analyser*, IEEE Annals of History of Computing, Volume 18(4), 16-26, 1996.

[11] Lipshitz, L., Rubel, L. A.: *A Differentialy Algebraic Replacement Theorem, and Analog Computation.* Proceedings of the A.M.S., Volume 99(2), 367-372, 1987.

[12] Matijasevich, Y.: *Enumerable Sets are Diophantine.* Dokl. Acad. Nauk, 191, 279-282, 1970.

[13] Mills, J. W.:*The Nature of the Extended Analog Computer.* Physica, Nonlinear Phenomena, Volume 237, No 9, 1236-1256, 2008.

[14] Odifreddi, P.: *Classical Recursion Theory*, Elsevier, 1989.

[15] Pour-El, M. B.: *Abstract Computability and Its Relation to the General Purpose Analog Computer*, Trans. Am. Math. Soc.,199, 1-28, 1974.

[16] Richardson, D.: *Some Undecidable Problems Involving Elementary Functions of a Real Variable*, The Journal of Symbolic Logic, Volume 33, Number 4, 1968, 514-520.

[17] Rubel, L. A.: *Some Mathematical Limitations of the General-Purpose Analog Computer*, Advances in Applied Mathematics, Volume 9,22-34, 1988.

[18] Rubel, L., A.: *Digital Simulation of Analog Computation, and Church's Thesis.* J. Symbolic Logic, 54, 1011-1017, 1989.

[19] Rubel, L. A.:*A Survey of Transcendentally Transcendental Functions*, Amer. Math. Monthly, Volume 96(9), 777-788, 1989.

[20] Rubel, L. A.: *The Extended Analog Computer*, Advances in Applied Mathematics, Volume 14, 39-50, 1993.

[21] Shannon, C.: *Mathematical Theory of the Differential Anlyzer*, J. Math. Phys. MIT, 20, 337-354, 1941.

[22] Siegelmann, H. T.: *Neural Networks and Analog Computation: Beyond the Turing Limit*, Birkhäuser, 1999.

[23] Whittaker, E. T., Watson, G. N. *"The Process of Continuation." 5.5 in A Course in Modern Analysis*, 4th ed. Cambridge, England: Cambridge University Press, 96-98, 1990.

Received 5th July 2008

A Customised ASM Thesis for Database Transformations

Klaus-Dieter Schewe* and Qing Wang†

Abstract

In order to establish a theoretical foundation for database transformations, we search for a universal computation model as an umbrella for queries and updates. As updates are fundamentally distinct from queries in many respects, computation models for queries cannot be simply extended to database transformations. This motivates the question whether Abstract State Machines (ASMs) can be used to characterise database transformations in general. In this paper we start examining the differences between database transformations and algorithms, which give rise to the formalisation of five postulates for database transformations. Then a variant of ASMs called Database Abstract State Machines (DB-ASMs) is developed, and we prove that DB-ASMs capture database transformations, i.e. the main result of the paper is that every database transformation stipulated by the postulates can be behaviourally simulated by a DB-ASM.

Keywords: Abstract State Machine, database transformation, ASM thesis

1 Introduction

According to [2] a database transformation is a binary relation on database instances that encompass queries and updates. In general, a database transformation can be non-deterministic, but it must be recursively enumerable and generic in the sense that it preserves isomorphisms. The problem addressed in this article is to completely characterise the algorithms that transform input databases into output databases.

Abstract State Machines (ASMs) provide a universal computation model that formalises the notion of (sequential or parallel) algorithm [5, 10]. In his seminal work on the sequential ASM thesis Gurevich points out the difference between a computable function in the recursion-theoretic sense and an algorithm. Strictly speaking, many algorithms in numerical mathematics, e.g. Newton's algorithm for

*Software Competence Center Hagenberg, Hagenberg, Austria and Johannes-Kepler-University Linz, Research Institute for Applied Knowledge Processing, Linz, Austria, E-mail: kd.schewe@scch.at, kd.schewe@faw.at

†University of Otago, Dunedin, New Zealand, E-mail: qing.wang@otago.ac.nz

determining zeros of a differentiable function $f : \mathbb{R} \to \mathbb{R}$, do not define computable functions, as they deal with non-denumerable sets. Only if restricted to a countable subset – such as floating-point numbers instead of real numbers – and properly encoded can we get a computable function. Thus, we are actually aiming at a complete characterisation of database transformation algorithms, but nonetheless we stick to the commonly used term of database transformation.

1.1 Contributions

Analogous to the seminal work on the ASM thesis this article contains two major contributions. The first one is a characterisation of database transformations by a set of simple and intuitive postulates. These postulates should cover all database transformations, and thus leave sufficient latitude to specify the specific characteristics of data models such as the standard relational data model, object oriented data models, and databases based on the eXtensible Markup Language (XML). In forthcoming studies we will elaborate the details for these models – for XML this has been done in [16].

The second contribution is a variant of ASMs, which we call Database Abstract State Machines (DB-ASMs). We show that DB-ASMs can capture exactly database transformations stipulated by the postulates. That is, we first show that DB-ASMs satisfy the postulates, and then that any object satisfying the postulates can be simulated by a DB-ASM.

We start with examining database transformations in the light of the postulates for sequential algorithms[1] defined in [10]. Firstly, database transformations should terminate, which implies that a run will always be finite and reach a final state. Furthermore, in order to take into consideration not only deterministic but also non-deterministic database transformations the requirement of a one-step transition function has to be relaxed. We will permit a relation instead, leading to a slightly modified sequential time postulate. The necessity for non-determinism arises among others from the creation of objects [18].

According to the abstract state postulate for ASMs states are first-order structures, and the sets of states used for an algorithm are invariant under isomorphisms. This notion of state must capture databases in general. Therefore, we will customize the abstract state postulate requesting that a state is composed out of a finite database component and an arbitrary algorithmic component that are linked via bridge functions. This picks up a fundamental idea from meta-finite model theory [9].

Same as for the parallel ASM thesis we have to refer explicitly to the background of a computation, which contains everything that is needed to perform the computation, but is not yet captured by the state. For instance, truth values and their connectives, and a value $\perp$ to denote undefinedness constitute necessary elements in a background. Furthermore, for database transformations we have to capture constructs that are determined by the used data model, so we will have to

[1] In Gurevich's theory sequential algorithms still permit bounded parallelism, whereas parallel algorithms are understood to capture even unbounded parallelism.

deal with type constructors, and with functions defined on such types. This will lead us to the background postulate for database transformations.

The fourth postulate needed for the sequential ASM thesis, the bounded exploration postulate, requires that there is a finite set of terms called bounded exploration witness, and only these terms can be updated in a one-step transformation. The generalisation of this postulate to the case of parallel algorithms in the parallel ASM thesis [5] leads to several significantly more complex postulates. As database transformations are intrinsically parallel computations, though an implementation may be sequential, we have to adopt parts of these more complex postulates. The adoption will only be partial, as the parallelism in database transformations – excluding for now the area of parallel databases – is rather limited; it merely amounts to the same computation on different data.

Regarding the restricted form of parallelism needed for database transformations we capture this by location operators, which generalise aggregation functions and cumulative updates. With these location operators we actually deal with metafinite structures with multiset operations as defined in [9]. In doing so, we have to consider update multisets, which are reduced to update sets by means of the location operators. Furthermore, depending on the data model used and thus on the actual background signature we may use complex values, e.g. tree-structured values, which leads to the problem of partial updates [11], i.e. we have to ensure that parallel updates to different parts of a tree (or database object, in general) can be synchronised. Dealing with partial updates actually is subsumed in the notion of consistent update set. Taking these ingredients together we obtain a slightly modified bounded exploration postulate.

The fifth postulate addresses non-determinism. As we permit non-determinism, equivalence of substructures may indeed be destroyed, but the non-determinism postulate ensures that non-determinism is restricted by depending only on the database part and not on the algorithmic part. However, bridge functions need to be restricted accordingly.

We then define DB-ASMs. Naturally, the permitted non-determinism requires the presence of a choice construct, while the restricted parallelism leads to a let construct that binds locations to location operators and a forall construct that allows the creation of a finite number of parallel subcomputations. For DB-ASMs we first show that they satisfy the postulates for database transformations. Our main result then shows that DB-ASMs capture database transformations, i.e. every database transformation stipulated by the postulates can be behaviourally simulated by a DB-ASM.

1.2 Background and Previous Work

With the upcoming of query languages for object oriented databases [1] the view that a query transforms databases over an input database schema into databases over an output schema that is disjoint from the input schema had to be relaxed by considering queries as transformations from an input schema to an extended output schema that preserve the input. From here it is a very small step to consider database transformations in general [2]. Since then a lot of research has been undertaken aiming at a logical characterisation of database transformations, e.g. [1, 17, 20, 18].

As discussed in [1], database transformations should satisfy criteria such as well-typedness, effective computability, genericity and functionality. However, most research with respect to these properties was conducted only for queries. According to [19] extending these results to updates is by no means straightforward. So far, there is not yet a computation model that can serve as a theoretical foundation for database transformations in general. In this article we aim at such a general model exploiting the theory of ASMs.

This article extends and corrects a preliminary conference publication [21]. In our previous work we tried to focus on tree-based databases, and therefore described states by higher-order structures. In this article, however, we capture databases in general, and therefore stay with first-order structures, while everything needed to express the specific needs of a data model is subsumed by the inclusion of background structures, which among others provide the necessary type constructors. Furthermore, we took up the idea from [22] to exploit meta-finite states [9]. In our previous work we were not able to handle such bridge functions properly. Finally, we polished the postulate capturing non-determinism.

1.3 Organisation of the Article

The remainder of this article is organised as follows. We begin with an illustrative example in Section 2. Then in Section 3 we present in detail our five postulates for database transformations. We motivate the postulates with examples that highlight the specific problems of developing a customised ASM thesis for database transformations. In addition we discuss the differences to the postulates in the sequential and parallel ASM theses. In Section 4 we present the DB-ASM variant of ASMs, and show that DB-ASMs satisfy the postulates. While this is relatively easy to achieve, we prove the converse in Section 5, i.e. DB-ASMs capture all database transformations. We conclude in Section 6 with a summary and discussion of further research, for which our current result is only the basis.

2 Illustrative Example

In this section we will provide an example to illustrate how a database transformation can be characterised by the five postulates and to answer the question of what a simulating DB-ASM for a database transformation should look like. The intention is to help the understanding of formal definitions presented in Section 3 and Section 4.

Example 1. Let us consider the weighted graph shown in Figure 1 and the database transformation "find the cheapest costs to reach other cities from the city C", where a node in the graph denotes a city and an arrow between two nodes denotes the transportation cost from one city to another.

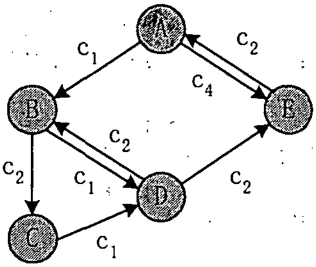

Figure 1: A weight graph with cities

Suppose that we choose the relational data model to represent the weighted graph in a database. That is, the relations City and Route in Figure 2 contain the information for cities and the transportation costs between two cities, respectively. Furthermore, we need the relation Visited to keep track of cities that have already been visited during the intermediate computation of a database transformation and the relation Result to store the final result. Now we discuss how to characterise this database transformation by using the five postulates.

- The sequential time postulate defines that this database transformation is a step-by-step computation which proceeds by one-step transitions from states to their successor states. A run of this database transformation is a finite sequence of states. If a run starts from an initial state that has the relations as shown in Figure 2, then it may terminate at a final state that has the relations as shown in Figure 3.

- The abstract state postulate defines that each state of this database transformation consists of the database part that contains four relations City, Route, Visited and Result, the algorithmic part that is an infinite structure $(\mathbb{R}, +, \cdot, max, min, \sum, \prod)$ and a bridge function $f_b : c_i \rightarrow i$ for $i = 0, ..., 5$. The purpose of the bridge function f_b is to interpret abstract elements of the attribute Cost in the relation Route of the database part with real numbers from the algorithmic part.

- The background postulate defines the background of this database transformation to reflect the choice of the relational data model and the use of relational algebra for query rewriting and optimisation at the implementation level. Therefore, each state should contain a background class defined by the background signature including at least type constructor symbols for the relational data model (i.e., finite tuple $(\cdot)$ and finite set $\{\cdot\}$) and relational algebraic operators (i.e.,σ (selection), π (projection), $\bowtie$ (join), $\cup$ (union), $-$ (difference), ϱ (renaming)), a set of base domains and a set of algebraic identities for rewriting query expressions.

- The bounded exploration postulate defines that, regardless of the chosen database language, etc, there must exist a fixed, finite set of access terms for this database transformation, which should include at least (CITY(x,y), true), (ROUTE(x,y,z), true), (VISITED(x), false), (RESULT(x,y,z), true) to access elements in the database part of a state. Furthermore, for any two different states, if they have the same interpretation for such a set of access terms, then the database transformation much create the same set of update sets over these two states at one-step transitions.

- The bounded exploration postulate defines that at any one-step transition the number of successor states to a state is finite, depending on access terms that can access elements from the database part of a state in a generic way.

CITY

Cid	Name
i_1	A
i_2	B
i_3	C
i_4	D
i_5	E

ROUTE

FromCid	ToCid	Cost
i_1	i_2	c_1
i_1	i_5	c_4
i_2	i_3	c_2
i_2	i_4	c_1
...	...	...

VISITED

Cid

RESULT

Cid	TotalCost	LastStop

Figure 2: The relations in an initial state

CITY

Cid	Name
i_1	A
i_2	B
i_3	C
i_4	D
i_5	E

ROUTE

FromCid	ToCid	Cost
i_1	i_2	c_1
i_1	i_5	c_4
i_2	i_3	c_2
i_2	i_4	c_1
...	...	...

VISITED

Cid
i_3
i_4
i_2
i_5
i_1

RESULT

Cid	TotalCost	LastStop
i_1	c_5	i_5
i_2	c_3	i_4
i_3	c_0	null
i_4	c_1	i_3
i_5	c_3	i_4

Figure 3: The relations in a final state

As one of important results in this paper is the development of DB-ASMs that can capture exactly all database transformations stipulated by the postulates, we implement Dijkstra's algorithm in a DB-ASM to simulate the database transformation discussed in Example 1.

Assume that, we have nullary function symbols @startcity, @infinity, initial, finished and mvalue in the state signature and @startcity=C, @infinity is a predefined number that should be big enough to be distinguished from the calculated cost numbers and initial=0 in every initial state of the database transformation. Then a simulating DB-ASM is presented in Figure 4.

```
par if initial=0 then seq
      forall x with City(x,y) do
         if x=@startcity then
            Result(x, 0, null):=true
         else
            Result(x,@infinity,null):=true
         endif enddo
      par initial:=1    finished:=0 endpar
   endseq endif
   if initial=1$\wedge$finished=0 then seq
      finished:=1
      let $\theta$(mvalue)=min in
         forall y with $\exists$ x,y. Result(x,y,z)$\wedge$ y< @infinity$\wedge\neg$ Visited(x) do
            par mvalue:=$f_b$(y)    finished:=0 endpar
         enddo
      endlet
      if finished=0 then
         choose x with $\exists$y,z.Result(x,y,z)$\wedge f_b$(y)=mvalue do seq
            Visited(x):=true
            forall y,y$'$,z$'$,z with Route(x,y,z)$\wedge$ Result(y,y$'$,z$'$)$\wedge$
                               mvalue+$f_b$(z)< $f_b$(y$'$)$\wedge\neg$ Visited(y) do seq
            f(y):=new()
               par
                  Result(y,f(y),x):=true
                  Result(y,y$'$,z$'$):=false
                  $f_b$(f(y)):=mvalue+$f_b$(z)
               endpar
            endseq enddo
         endseq enddo
      endif
   endseq endif
endpar
```

Figure 4: A simulating DB-ASM

3 Postulates for Database Transformations

In this section we will formally introduce the five postulates for database transformations: the *sequential time postulate*, the *abstract state postulate*, the *background postulate*, the *bounded exploration postulate*, and the *bounded non-determinism postulate*.

Definition 1. A *database transformation* is an object satisfying the sequential time postulate, the abstract state postulate, the background postulate, the bounded exploration postulate, and the bounded non-determinism postulate.

3.1 Sequential Time

As in [10] and [5] we assume that a database transformation same as any algorithm proceeds step-wise on a set of states. It starts somewhere, which gives us a set of initial states. However, while it makes perfect sense to consider non-terminating algorithms, we want to consider only terminating database transformations, for which we add a set of final states. As discussed in [10] this is more a technicality as far as the work in this paper is concerned, but we aim at embedding our results into a theory of database systems, in which many database transformations have to co-exist.

Furthermore, we deviate from the sequential time postulate in the sequential and parallel ASM theses by using a one-step transition relation on states as in bounded-choice sequential algorithms [12] instead of a transformation function. This introduces non-determinism into database transformations, which will be limited by further postulates. In fact we will only permit non-deterministic choice among the finite answer to a query. The major reason for the non-determinism in database transformations is the need for creating objects in some data models as discussed intensively in [19, 17, 18].

Postulate 1 (sequential time postulate). *A database transformation t is associated with a non-empty set of states $\mathcal{S}_t$ together with non-empty subsets $\mathcal{I}_t$ and $\mathcal{F}_t$ of initial and final states, respectively, and a one-step transition relation τ_t over $\mathcal{S}_t$, i.e. $\tau_t \subseteq \mathcal{S}_t \times \mathcal{S}_t$.*

The sequential time postulate allows us to define the notion of a *run* in analogy to sequential and parallel algorithms. As we require termination, a run must be finite ending in a final state, which should be the first final state that is reached. Nevertheless, we permit the initial state in a run to be also a final state. The motivation behind this is that database transformations, though treated in isolation in this paper, are associated with a database system, in which each run of a database transformation produces a state transition, and the final state of that transition becomes the initial state for another transition, etc. leading to an infinite sequence of states that results from running a set of database transformations in a serial (or serialisable) way. This view of database systems has been stressed in [14].

Definition 2. A *run* of a database transformation t is a finite sequence $S_0, \dots, S_f$ of states with $S_0 \in \mathcal{I}_t$, $S_f \in \mathcal{F}_t$, $S_i \notin \mathcal{F}_t$ for $0 < i < f$, and $(S_i, S_{i+1}) \in \tau_t$ for all $i = 0, \dots, f-1$.

We will consider database transformations only up to behavioural equivalence.

Definition 3. Database transformations t_1 and t_2 are *behaviourally equivalent* iff $\mathcal{S}_{t_1} = \mathcal{S}_{t_2}$, $\mathcal{I}_{t_1} = \mathcal{I}_{t_2}$, $\mathcal{F}_{t_1} = \mathcal{F}_{t_2}$ and $\tau_{t_1} = \tau_{t_2}$ hold.

Obviously, behaviourally equivalent database transformations have the same runs.

3.2 Abstract States

The abstract state postulate is an adaptation of the corresponding postulate for Abstract State Machines [10], according to which states are first-order structures, i.e. sets of (partial) functions, some of which may be marked as relational. These functions are interpretations of function symbols given by some signature. Following [10] it is assumed that each signature contains the equality sign, and nullary names *true*, *false*, *undef*, a unary name *Bool*, and the names of the usual Boolean operations. With the exception of *undef* all these logic names are relational. Equality, truth values and Boolean operations are interpreted in a fixed way in all states. In particular, partial functions are captured by the undefinedness value $\perp$ associated with *undef*.

Definition 4. A *signature* Σ is a set of function symbols, each associated with a fixed arity. A *structure* over Σ consists of a set B, called the *base set* of the structure together with interpretations of all function symbols in Σ, i.e. if $f \in \Sigma$ has arity k, then it will be interpreted by a function from B^k to B.

Let X be a structure over Σ. For each term $t \in \Sigma$ we use $val_X(t)$ to denote the interpretation of t in the structure X.

Definition 5. An *isomorphism* from structure X to structure Y is defined by a bijection $\sigma : B_X \to B_Y$ between the base sets that extends to functions by $\sigma(val_X(f(b_1, \dots, b_k))) = val_Y(f(\sigma(b_1), \dots, \sigma(b_k)))$. A *$Z$-isomorphism* for $Z \subseteq B_X \cap B_Y$ is an isomorphism σ from X to Y that fixes Z, i.e. $\sigma(b) = b$ for all $b \in Z$.

As the base set B contains a value $\perp$ representing undefinedness, partial functions are captured in the usual way. Furthermore, relations are captured by letting $val_X(f(a_1, \dots, a_k)) = true$ mean that $(a_1, \dots, a_k)$ is in the relation f of X, and $val_X(f(a_1, \dots, a_k)) = false$ mean that it is not.

Z-isomorphisms are needed when dealing with constants in the base set that are represented by 0-ary function symbols. However, an automorphism σ of a structure X fixes all values $a \in B$ that are represented by ground terms, i.e. if $a = val_X(t)$ holds for some ground term t, then $\sigma(a) = a$ holds for each automorphism σ of

X. Thus, Z-isomorphisms can be neglected, as we could always add syntactic surrogates, i.e. 0-ary function symbols for all elements of Z to the signature Σ.

Taking structures as states reflects common practice in mathematics, where almost all theories are based on first-order structures. Variables are special cases of function symbols of arity 0, and constants are the same, but unchangeable.

In the case of databases we have to take care of two specific problems that affect the definition of states. The first problem is the intrinsic finiteness of databases. For this it is tempting to adopt Finite Model Theory [8] and consequently require that states for database transformations are finite structures. This would, however, not capture the full picture. For instance, a simple query counting the number of tuples in a relation would require natural numbers in the base set, and any restriction to a finite set would already rule out some database transformations. Fortunately, in order to deal with this problem Grädel and Gurevich proposed the use of *meta-finite* structures [9], henceforth *meta-finite states*, in which we may consider actual database entries as being merely surrogates for real values. This permits the database to remain finite while adding functions that interpret database entries in possibly infinite domains, and at the same time generalises most of the result achieved in Finite Model Theory to meta-finite models. We adopt this model, and consequently there will be three kinds of function symbols, those representing the database, those representing everything outside the database, and bridge functions that map surrogates in the database to values outside the database in possibly infinite domains.

There is one little sublety here that we also have to take care of. In object-oriented databases we may make use of object identifiers, in tree-based databases we may require identifiers for tree nodes, and in both cases there is a need to create new identifiers. As discussed in [10] the creation of new values in principle is no problem, as we can assume an infinite set of reserve values existing in the background of a database transformation, from which such new values are taken, but it is not a priori clear how many such values will be needed. In the subsection on backgrounds we will further clarify this matter. Therefore, this problem can be circumvented by requiring that only the active database domain is finite, i.e. the set of values of the base set appearing in the database part of the structure.

The second database-specific problem is the presence of a data model that prescribes how a database should look like. In case of the relational model we would have to deal simply with relations, while in case of object-oriented and XML-based databases we need constructors for complex values such as finite sets, multisets, maps, arrays, union, trees, etc. We will deal with the consequences for states separately in the subsection on backgrounds.

Definition 6. The *signature* Σ of a meta-finite structure is composed as a disjoint union consisting of a sub-signature Σ_{db} (called *database part*), a sub-signature Σ_a (called *algorithmic part*) and a finite set of bridge function symbols each with a fixed arity, i.e. $\Sigma_t = \Sigma_{db} \cup \Sigma_a \cup \{f_1, \dots, f_\ell\}$. The *base set* of a meta-finite strcuture S is $B = B_{db}^{ext} \cup B_a$ with interpretation of function symbols in Σ_{db} and Σ_a over $B_{db} \subseteq B_{db}^{ext}$ and B_a, respectively, with B_{db} depending on S. The interpretation of a bridge function symbol of arity k defines a function from B_{db}^k to B_a. With respect to such states S the restriction to Σ_{db} is a finite structure, i.e. B_{db} is finite.

Postulate 2 (abstract state postulate). *All states $S \in \mathcal{S}_t$ of a database transformation t are meta-finite structures over the same signature Σ_t, and whenever $(S, S') \in \tau_t$ holds, the states S and S' have the same base set B. The sets $\mathcal{S}_t$, $\mathcal{I}_t$ and $\mathcal{F}_t$ are closed under isomorphisms, and for $(S_1, S_1') \in \tau_t$ each isomorphism σ from S_1 to S_2 is also an isomorphism from S_1' to $S_2' = \sigma(S_1')$ with $(S_2, S_2') \in \tau_t$.*

The abstract state postulate is an adaptation of the analogous postulate from [10, 5] to further consider states as being meta-finite structures, the presence of final states and the fact that one-step transition is a binary relation.

Example 2. Consider a database, in which we represent persons with their name and age. In the sub-signature Σ_{db} we would thus have a binary function symbol *person*. In addition, Σ_a could contain a unary function symbol *even*, and we would have two bridge functions f_{name} and f_{age} for the interpretation of the two components of persons.

In a structure, we would get $B_{db} = D_{\text{name}} \cup D_{\text{age}} \cup \{true, false\}$ as a union of three disjoint, finite sets, and $B_a = \mathbb{N} \cup A^*$ as the union of the set of non-negative integers and the set of character strings over some alphabet A, both infinite.

The function symbol *person* would be interpreted by a function $D_{\text{name}} \times D_{\text{age}} \rightarrow \{\text{true, false}\}$, and *even* would be interpreted by a function $\mathbb{N} \rightarrow \{\text{true, false}\}$. The bridge function symbols would be interpreted as functions $D_{\text{name}} \rightarrow A^*$ and $D_{\text{age}} \rightarrow \mathbb{N}$, respectively.

Using this representation of a finite relation of persons with name and age, a query such as "List the names of all persons with even age" would be possible, provided we add a unary function symbol *names* to the signature to pick up the result.

Similarly, a query such as "List the names of all persons who are older than average" would require 0-ary function symbols *age_sum*, *count*, and *average_age* as well as a binary function symbol $>$ in the algorithmic part of the signature.

In the abstract state postulate above we adopt the idea of meta-finite states, but we do not restrict the database part of the state to be relational, so we can capture data models other than the relational one as well.

Let us finally look at genericity as expressed by the preservation of isomorphisms in successor states. In the sequential ASM thesis sequential algorithms are deterministic, so a state S has a unique successor state $\tau(S)$. Then the abstract state postulate implies that an automorphism σ of S is also an automorphism of $\tau(S)$. In the abstract state postulate for database transformations (i.e., Postulate 2), however, there can be more than one successor state of S, as τ_t is a relation. Now, if σ is an automorphism of S, and $(S, S') \in \tau_t$ holds, we obtain an isomorphism σ from S' to $S'' = \sigma(S')$ with $(S, S'') \in \tau_t$. Thus, an automorphism of S induces a permutation of the successor states of S.

3.3 Updates

The definitions of locations, updates, update sets and update multisets are the same as for ASMs [6].

Definition 7. For a database transformation t let S be a state of t, f a dynamic function symbol of arity n in the state signature of t, and $a_1, ..., a_n, v$ be elements in the base set of S. Then $f(a_1, ..., a_n)$ is called a *location* of t. The interpretation of a location ℓ in S is called the *content* of ℓ in S, denoted by $val_S(\ell)$. An *update* of t is a pair (ℓ, v), where ℓ is a location and v is an update value. An *update set* is a set of updates; an *update multiset* is a multiset of updates.

An update is *trivial* in a state S if its location content in S is the same with its update value, while an update set is *trivial* if all of its updates are trivial.

An update set Δ is *consistent* if it does not contain conflicting updates, i.e. for all $(\ell, v), (\ell, v') \in \Delta$ we have $v = v'$.

Using a *location function* (denoted by θ) that assigns a location operator or $\perp$ to each location, an update multiset can be reduced to an update set. It is further possible to construct for each $(S, S') \in \tau_t$ a minimal update set $\Delta(t, S, S')$ such that applying this update set to the state S will produce the state S'. More precisely, if S is a state of the database transformation t and Δ is a consistent update set for the signature of t, then there exists a unique state $S' = S + \Delta$ resulting from updating S with Δ: we simply have

$$val_{S+\Delta}(\ell) \quad = \quad \begin{cases} v & \text{if } (\ell, v) \in \Delta \\ val_S(\ell) & \text{else} \end{cases}$$

If Δ is not consistent, we let $S + \Delta$ be undefined. Note that this last point is different from the treatment of inconsistent update sets in [10], but as discussed there the difference is a mere technicality as long as we concentrate on a single database transformation. Same as with final states the distinction only becomes necessary when placed into the context of persistence with several concurrent database transformations, and a serialisability request. In that case a computation that gets stuck and thus has to be aborted (this is the case, when $S + \Delta$ is undefined) has to be distinguished from a computation that produces the same state S over and over again (this is the case, if $S + \Delta$ is defined as S in case of Δ being inconsistent as in [10]).

Lemma 1. *Let $S, S' \in \mathcal{S}_t$ be states of the database transformation t with the same base set. Then there exists a unique, minimal consistent update set $\Delta(t, S, S')$ with $S' = S + \Delta(t, S, S')$.*

Note that the minimality of the update set implies the absence of trivial updates.

Proof. Let $Loc_\Delta = \{\ell \mid val_S(\ell) \neq val_{S'}(\ell)\}$ be the set of locations, on which the two states differ. Then the update set $\Delta(t, S, S') = \{(\ell, val_{S'}(\ell) \mid \ell \in Loc_\Delta\}$ is the one needed. □

Let us now look at the one-step transition relation τ_t of a database transformation t. As we permit non-determinism, i.e. there may be more than one successor state of a state S, we need a set of update sets. Therefore, define

$$\Delta(t, S) = \{\Delta(t, S, S') \mid (S, S') \in \tau_t\}$$

for a database transformation t and a state $S \in \mathcal{S}_t$.

Let us take a brief look at the effect of isomorphisms on update sets and sets of update sets. For this, any isomorphism σ can be extended to updates $(f(a_1, \dots, a_n), b)$ by defining $\sigma((f(a_1, \dots, a_n), b)) = (f(\sigma(a_1), \dots, \sigma(a_n)), \sigma(b))$, and to sets by defining $\sigma(\{u_1, \dots, u_k\}) = \{\sigma(u_1), \dots, \sigma(u_k)\}$.

Lemma 2. *Let S_1 be a state of a database transformation t and σ be an isomorphism from S_1 to S_2. Then $\Delta(t, S_2, \sigma(S_1')) = \sigma(\Delta(t, S_1, S_1'))$ for all $(S_1, S_1') \in \tau_t$, and consequently $\Delta(t, S_2) = \sigma(\Delta(t, S_1))$.*

Proof. According to the abstract state postulate all $(S_2, S_2') \in \tau_t$ have the form $(\sigma(S_1), \sigma(S_1'))$ with $(S_1, S_1') \in \tau_t$. Then $val_{\sigma(S_1)}\sigma(\ell) = \sigma(val_{S_1}(\ell))$ and analogously for S_1'. So

$$Loc_{\Delta_2} = \{\ell \mid val_{S_2}(\ell) \neq val_{S_2'}(\ell)\} = \{\sigma(\ell) \mid val_{S_1}(\ell) \neq val_{S_1'}(\ell)\} = \sigma(Loc_{\Delta_1}),$$

and

$$\begin{aligned}\Delta(t, S_2, S_2') =& \{(\ell, val_{S_2'}(\ell)) \mid \ell \in Loc_{\Delta_2}\} = \\ & \{(\sigma(\ell'), \sigma(val_{S_1'}(\ell'))) \mid \sigma(\ell') \in \sigma(Loc_{\Delta_1})\} = \sigma(\Delta(t, S_1, S_1')).\end{aligned}$$

□

3.4 Backgrounds

The postulates 1 and 2 are in line with the sequential and parallel ASM theses [10, 5], and with the exception of allowing non-determinism in the sequential time postulate and the reference to meta-finite structures in the abstract state postulate there is nothing in these postulates that makes a big difference to postulates for sequential algorithms. The next postulate, however, is less obvious, as it refers to the background of a computation, which contains everything that is needed to perform the computation, but is not yet captured by the state. For instance, truth values and their connectives, and a value $\perp$ to denote undefinedness constitute necessary elements in a background.

For database transformations, in particular, we have to capture constructs that are determined by the used data model, e.g. relational, object-oriented, object-relational or semi-structured, i.e. we will have to deal with type constructors, and with functions defined on such types. Furthermore, when we allow values, e.g. identifiers to be created non-deterministically, we would like to take these values out of an infinite set of reserve values. Once created, these values become active, and we can assume they can never be used again for this purpose.

Let us take the following example, which was used in [1] to illustrate a data model that generalises most of known complex object data models. In this model a distinction is made between abstract identifiers and constants. These elements stem from disjoint base domains, which together constitute the *base set* for database transformations. With the addition of constructors for records, sets, multisets, lists, etc. domains of arbitrarily nested complex values can be built upon the base domains. For instance, a domain $D_{(Int,\{String\})}$ over base domains *Int* and *String* would represent complex record values consisting of an integer and a set of strings.

Example 3. Suppose that the state of a database has a universe containing abstract identifiers from domains $I_1 = \{i_{eve}, i_{adam}\}$, $I_2 = \{i_{cain}, i_{abel}, i_{seth}, i_{other}\}$, $I_3 = \{i_{n_k} | k = 1, ..., 5\}$, $I_4 = \{i_{o_k} | k = 1, ..., 3\}$, $I_5 = \{i_{d_1}, i_{d_2}\}$ and constants from domains *String* and *Bool*. Furthermore, assume that we have the following constructors: finite sets $\{\cdot\}$ with unfixed arity, records $(\cdot)$ with arity up to 3, and union $\cup$ with arity 2.

Let the state signature contain function names *1st-generation*, *2nd-generation*, *name*, *occupation*, *descendant*, and relation names *founded-lineage*, *ancestor-of-celebrity* such that

- *1st-generation*: $I_1 \rightarrow \mathrm{D}_{(nam:I_3, spou:I_1, children:\{I_2\})}$,
- *2nd-generation*: $I_2 \rightarrow \mathrm{D}_{(nam:I_3, occu:I_4)}$,
- *founded-lineage*: 2nd-gen:$I_2 \rightarrow$ *Bool*,
- *ancestor-of-celebrity*: anc:$I_2\times$ desc:$I_5 \rightarrow$ *Bool*,
- *name*: $I_3 \rightarrow \mathrm{D}_{String}$,
- *occupation*: $I_4 \rightarrow \mathrm{D}_{\{String\}}$,
- *descendant*: $I_5 \rightarrow \mathrm{D}_{String\cup(spou:String)}$.

The interpretation of function and relation names in the state signature is as follows:

- for *1st-generation*, $i_{eve} \mapsto$ (nam: i_{n_1}, spou: i_{adam}, children: $\{i_{cain}, i_{abel}, i_{seth}, i_{other}\}$) and $i_{adam} \mapsto$ (nam:i_{n_2}, spou: i_{eve}, children:$\{i_{cain}, i_{abel}, i_{seth}, i_{other}\}$),
- for *2nd-generation*, $i_{cain} \mapsto$ (nam: i_{n_3}, occu: i_{o_1}), $i_{seth} \mapsto$ (nam: i_{n_4}, occu: i_{o_2}) and $i_{abel} \mapsto$ (nam: i_{n_5}, occu: i_{o_3}),
- for *founded-lineage*, it is {(2nd-gen: i_{cain}),(2nd-gen: i_{seth}),(2nd-gen: i_{other})},
- for *ancestor-of-celebrity*, it is {(anc: i_{seth}, desc: i_{d_1}), (anc: i_{cain}, desc: i_{d_2})},
- for *name*, $i_{n_1} \mapsto$ Eve, $i_{n_2} \mapsto$ Adam, $i_{n_3} \mapsto$ Cain, $i_{n_4} \mapsto$ Seth and $i_{n_5} \mapsto$ Abel,
- for *occupation*, $i_{o_1} \mapsto$ {Farmer, Nomad, Artisan}, $i_{o_2} \mapsto$ {} and $i_{o_3} \mapsto$ {Shepherd},

- for *descendant*, $i_{d_1} \mapsto$ Noah and $i_{d_2} \mapsto$ (spou: Ada).

That is, objects of *1st-generation* are described by a name, a reference to a spouse, and a set of references to children. Objects of *2nd-generation* are described by a name and a set of professions, and objects of *descendant* are described by a name only or a record with a name. The *founded-lineage* defines a subset of the second generation, and *ancestor-of-celebrity* is a simple binary relation.

Suppose, we want to create a new object with a new identifier in I_3. For this, we obtain a new identifier $i_3 \in I_3$, from the set of reserve values. We then set $name(i_3) :=$ Isaac and $ancestor\text{-}of\text{-}celebrity(i_{seth}, i_3) := true$ to update *name* and *ancestor-of-celebrity* (taking $false$ as the default value for all other cases). Similarly, with $founded\text{-}lineage(i_{other}) := false$ we would delete i_{other} from the *founded-lineage* relation.

Following [5] we use background classes to define backgrounds, which will then become part of states. Background classes themselves are determined by background signatures that consist of constructor symbols and function symbols. Function symbols are associated with a fixed arity as in Definition 4, but for constructor symbols we permit the arity to be unfixed or bounded.

Definition 8. Let $\mathcal{D}$ be a set of base domains and V_K a background signature, then a *background class* $\mathcal{K}$ with V_K over $\mathcal{D}$ is constituted by

- the universe $\mathcal{U} = \bigcup_{D \in \mathfrak{D}} D$ of elements, where $\mathfrak{D}$ is the smallest set with $\mathcal{D} \subseteq \mathfrak{D}$ satisfying the following properties for each constructor symbol $\llcorner\lrcorner \in V_K$:
  - If $\llcorner\lrcorner \in V_K$ has unfixed arity, then $\llcorner D \lrcorner \in \mathfrak{D}$ for all $D \in \mathfrak{D}$, and $\llcorner a_1, \dots, a_m \lrcorner \in \llcorner D \lrcorner$ for every $m \in \mathbb{N}$ and $a_1, \dots, a_m \in D$.
  - If $\llcorner\lrcorner \in V_K$ has unfixed arity, then $A_{\llcorner\lrcorner} \in \mathfrak{D}$ with $A_{\llcorner\lrcorner} = \bigcup_{\llcorner D \lrcorner \in \mathfrak{D}} \llcorner D \lrcorner$.
  - If $\llcorner\lrcorner \in V_K$ has bounded arity n, then $\llcorner D_1, \dots, D_m \lrcorner \in \mathfrak{D}$ for all $m \leq n$ and $D_i \in \mathfrak{D}$ $(1 \leq i \leq m)$, and $\llcorner a_1, \dots, a_m \lrcorner \in \llcorner D_1, \dots, D_m \lrcorner$ for every $m \in \mathbb{N}$ and $a_1, \dots, a_m \in D$.
  - If $\llcorner\lrcorner \in V_K$ has fixed arity n, then $\llcorner D_1, \dots, D_n \lrcorner \in \mathfrak{D}$ for all $D_i \in \mathfrak{D}$ $(1 \leq i \leq n)$, and $\llcorner a_1, \dots, a_n \lrcorner \in \llcorner D_1, \dots, D_n \lrcorner$ for all $a_1, \dots, a_n \in D$.
- and an interpretation of function symbols in V_K over $\mathcal{U}$.

Example 4. Let us consider the type system used in [1] with some slight modifications. Type expressions are defined as follows:

$$\tau = \lambda \mid D \mid P \mid (A_1 : \tau_1, \dots, A_k : \tau_k) \mid \{\tau\} \mid \tau_1 \sqcup \tau_2 \mid \tau_1 \sqcap \tau_2$$

The semantics of these type expressions, denoted as $\llbracket \tau \rrbracket$, is formally defined as follows:

$$\begin{aligned}
[\![\lambda]\!] &= \emptyset \\
[\![D]\!] &= \xi_1(D) \\
[\![P]\!] &= \xi_2(P) \\
[\![(A_1 : \tau_1, ..., A_k : \tau_k)]\!] &= \{(A_1 : v_1, ..., A_k : v_k) \mid v_i \in [\![\tau_i]\!], i = 1, ..., k\} \\
[\![\{\tau\}]\!] &= \{\{v_1, ..., v_j\} \mid j \geq 0 \text{ and } v_i \in [\![\tau]\!], i = 1, ..., j\} \\
[\![\tau_1 \sqcup \tau_2]\!] &= [\![\tau_1]\!] \cup [\![\tau_2]\!] \\
[\![\tau_1 \sqcap \tau_2]\!] &= [\![\tau_1]\!] \cap [\![\tau_2]\!]
\end{aligned}$$

So λ is a trivial type denoting the empty set $\emptyset$. D and P represent a base type for constants and a class type for objects, respectively, and ξ_1 and ξ_2 are functions mapping each base type to a possibly infinite set of constants, and each class type to a finite set of objects, respectively. In addition to these, there are constructor symbols – records $(\cdot)$ with bounded arity k, finite sets $\{\cdot\}$ with unfixed arity, as well as unions $\sqcup$ and intersections $\sqcap$, both of arity 2.

The types are associated with function symbols $\in$ of arity 2 denoting set membership, π_i $(1 \leq i \leq k)$ of arity k denoting projection functions on records, and $\cup$ and $\cap$, both of arity 2 denoting union and intersection, respectively.

For every database transformation, a binary tuple constructor $(,)$ is indispensable. This is due to the formalisation of update that is a pair of a location and an update value as defined in Definition 7. The type constructor for finite multisets also plays a critical role in database transformations since a database transformation may have many subcomputations running in parallel, which yield possibly identical updates. As we will introduce later, by assigning location operators to locations, identical updates yielded during a computation can be aggregated to form a final update in an update set yielded by a one-step transition. Therefore, we need the type constructor for finite multisets to collect all updates generated during parallel computations.

The following are several multiset operations [5]. We use the constructor symbol $\langle\cdot\rangle$ for finite multisets with unfixed arity. Let x and y be two multisets, and M be a set of multisets.

- $x \uplus y$ returns a multiset that has members from x and y, and the occurrence of each member is the sum of the occurrences of such a member in x and in y.
- $\biguplus M$ returns a multiset that has members from all elements of M, and the occurrence of each member is the sum of the occurrences of such a member in all elements of M.
- $AsSet(x)$ returns a set that has the same members as x, such that

$$AsSet(x) \quad = \quad \{a \mid a \in x\}$$

- $\mathrm{I}x$ is defined by

$$\mathrm{I}x \quad = \quad \begin{cases} a & \text{if } x = \langle a \rangle \\ \bot & \text{otherwise} \end{cases}$$

Postulate 3 (background postulate). *Each state of a database transformation t must contain*

- *an infinite set of reserve values,*
- *truth values and their connectives, the equality predicate, the undefinedness value* $\bot$*, and*
- *a background class* $\mathcal{K}$ *defined by a background signature* V_K *that contains at least a binary tuple constructor* $(\cdot)$*, a finite multiset constructor* $\langle\cdot\rangle$*, and function symbols for operations such as pairing and projection for pairs, and empty multiset* $\langle\rangle$*, singleton* $\langle x\rangle$*, binary multiset union* $\uplus$*, general multiset union* $\biguplus$*, AsSet, and* $\mathrm{I}x$ *on multisets.*

The minimum requirements in the background postulate are the same as for parallel algorithms [5], but we leave it open how many other constructors will be in a background class in order to capture any request in data models.

Given the base set of a state S, we can add truth values and $\bot$, and partition them into base domains. Then the background class $\mathcal{K}$ contained in S can be obtained by applying the construction provided in Definition 8 to get a much larger base set and to interpret functions symbols in V_K with respect to this enlarged base set.

3.5 Bounded Exploration

The bounded exploration postulate for sequential algorithms requests that only finitely many terms can be updated in an elementary step [10]. For parallel algorithms this postulate becomes significantly more complicated, as basic constituents not involving any parallelism (so-called "proclets") have to be considered [5].

For database transformations the problem lies somehow in between. Computations are intrinsically parallel, even though implementations may be sequential, but the parallelism is restricted in the sense that all branches execute de facto the same computation. We will capture this by means of location operators, which generalise aggregation functions as in [7] and cumulative updates.

The idea behind location operators is inspired by the synchronisation of parallel updates in [5]. First, updates generated by parallel computations define an update multiset, then all updates to the same location are merged by means of a location operator to reduce the update multiset to an update set.

Definition 9. Let $M(D)$ be the set of all non-empty multisets over a domain D, then a *location operator* ρ over $M(D)$ consists of a unary function $f_\alpha : D \to D$, a commutative and associative binary operation $\odot$ over D, and a unary function $f_\beta : D \to D$, which define $\rho(m) = f_\beta(f_\alpha(b_1) \odot \cdots \odot f_\alpha(b_n))$ for $m = \langle b_1, ..., b_n \rangle \in M(D)$.

If a database transformation uses location operators, they must be defined in the algorithmic part of states requested in Postulate 2.

Example 5. *sum* is a location operator with $f_\alpha(v) = v$, $v_1 \odot v_2 = v_1 + v_2$, and $f_\beta(v) = v$. Another example is *avg* with $f_\alpha(v) = (v, 1)$, $(v_1, w_1) \odot (v_2, w_2) = (v_1 + v_2, w_1 + w_2)$, and $f_\beta((v, w)) = v \div w$.

Location operators define operations on multisets, and as such form an important part of logics for meta-finite structures [9]. They permit to express in a simple way the restricted parallelism in many database aggregate functions such as building sums or average values over query results, selecting maximum or minimum, and even structural recursion on sets, multisets, lists or trees.

Example 6. Consider the evaluation of a Boolean formula $\forall x \in D_1 \exists y \in D_2 \varphi(x, y)$. Assume the evaluation result will be stored at location ℓ. Let the cardinalities of D_1 and D_2 be n_1 and n_2, respectively. Then there are two nested parallel computations involved in the evaluation. The inner parallel computation for a specific value $u_i \in D_1$ $(i \in [1, n_1])$ has n_2 parallel branches, each of which evaluating a term $\varphi(u_i, v_j)$ for the various values $v_j \in D_2$ $(j \in [1, n_2])$, thus producing an update multiset with n_2 updates $(\ell, true)$ or $(\ell, false)$. Using $\theta(\ell) = \bigvee$ (logical OR) as location operator – in this case f_α and f_β are the identity function, and $\odot$ is $\vee$ – evaluates the inner existentially quantified formula, thereby producing another update multiset with n_1 entries $(\ell, true)$ or $(\ell, false)$. Using the location operator $\theta(\ell) = \bigwedge$ (logical AND) reduces this update multiset to a set with a single update.

Depending on the data model used and thus on the actual background signature we may use complex values, e.g. tree-structured values. As a consequence we have to cope with the problem of partial updates [11], e.g. the synchronisation of updates to different parts of the same tree values, or more generally complex database objects. Since updates may produced at different levels of abstraction, overlapping locations can lead to clashes. However, the issues relating to inconsistent update sets are irrelevant for the proof of the characterization theorem in Section 5 as shown in [5, 10]. That is, the problem of partial updates is subsumed by the problem of providing consistent update sets, in which there cannot be pairs (ℓ, v_1) and (ℓ, v_2) with $v_1 \neq v_2$.

The bounded exploration postulate in [10] for sequential algorithms is motivated by the *sequential accessibility principle*, which could be phrased as the request that each location must be uniquely identifiable. Leaving aside the discussion how to deal logically with partially defined terms unique identifiability can be obtained by using terms of the form $\mathbf{I}x.\varphi(x)$ with a formula φ, in which x is the only free variable. Such terms have to be interpreted as "the unique x satisfying formula $\varphi(x)$", which of course may be undefined, if no such x exists or more than one exist. According to the modified abstract state postulate 2 for database transformations the sequential accessibility principle must be preserved for the algorithmic part of the structure.

In principle, the claim of unique identifiability also applies to databases, as emphasised by Beeri and Thalheim in [4]. More precisely, unique identifiability has to be claimed for the basic updatable units in a database, e.g. objects in [15]. Unique identifiability, however, does not necessarily apply to all elements in a database. Sets of logically indistinguishable locations may be updated simultaneously. Nevertheless, for databases only logical properties are relevant – this is the so-called "genericity principle" in database theory [3] – and therefore, it must still be possible to use terms to access elements and locations in the database part of a state. These terms, however, may be non-ground. If a non-ground term identifies more than one location in a state S, these locations will be called accessible in parallel.

Definition 10. Let S be a state of the database transformation t. An element a of S is *accessible* if there is a ground term α in the signature of S that is interpreted as a in S. A location $f(a_1, \dots, a_n)$ is *accessible* if the elements $a_1, \dots, a_n$ are all accessible. An update $(f(a_1, \dots, a_n), b)$ is *accessible* if the location $f(a_1, \dots, a_n)$ and the element b are accessible.

Locations $f(a_1^1, \dots, a_n^1), \dots, f(a_1^m, \dots, a_n^m)$ with $f \in \Sigma_{db}$ are *accessible in parallel* if there exists a term α and an accessible element b', such that the values for which α is interpreted by b' in S are $f(a_1^1, \dots, a_n^1), \dots, f(a_1^m, \dots, a_n^m)$.

Updates $(f(a_1^1, \dots, a_n^1), b), \dots, (f(a_1^m, \dots, a_n^m), b)$ with $f \in \Sigma_{db}$ are *accessible in parallel* iff $f(a_1^1, \dots, a_n^1), \dots, f(a_1^m, \dots, a_n^m)$ are accessible in parallel and b is accessible.

The first part of Definition 10 is exactly the same as defined in [10, Definition 5.3]. The second part formalises our discussion above.

Example 7. Take a database transformation t with a ternary predicate symbol R in its signature. Let the interpretation of R in a state S be $\{(a, a, b), (a, b, c), (b, b, a), (b, a, c)\}$. Then $R(a, a, b)$ and $R(b, b, a)$ are accessible in parallel using the term $R(x, x, y)$ and the accessible element **true**.

The bounded exploration postulate in the sequential ASM thesis in [10] uses a finite set of ground terms as bounded exploration witness in the sense that whenever states S_1 and S_2 coincide over this set of ground terms the update set produced by the sequential algorithm is the same in these states. The intuition behind the postulate is that only the part of a state that is given by means of the witness will actually be explored by the algorithm.

The fact that only finitely many locations can be explored remains the same for database transformations. However, permitting parallel accessibility within the database part of a state forces us to slightly change our view on the bounded exploration witness. For this we need access terms.

Definition 11. An *access term* is either a ground term α or a pair (β, α) of terms, the variables $x_1, \dots, x_n$ in which must be database variables, referring to the arguments of some dynamic function symbol $f \in \Sigma_{db} \cup \{f_1, ..., f_\ell\}$. The interpretation of (β, α) in a state S is the set of locations

$$\{f(a_1, \dots, a_n) \mid val_{S,\zeta}(\beta) = val_{S,\zeta}(\alpha) \text{ with } \zeta = \{x_1 \mapsto a_1, \dots, x_n \mapsto a_n\}\}.$$

Structures S_1 and S_2 *coincide* over a set T of access terms if the interpretation of each $\alpha \in T$ and each $(\beta, \alpha) \in T$ over S_1 and S_2 are equal.

Instead of writing (β, α) for an access term, we should in fact write (f, β, α), but for simplicity we drop the function symbol f and assume it is implicitly given.

Due to our request that the database part of a state is always finite there will be a maximum number m of elements that are accessible in parallel. Furthermore, there is always a number n such that n variables are sufficient to describe the updates of a database transformation, and n can be taken to be minimal. Then for each state S the upper boundary of exploration is $\mathcal{O}(m^n)$, where m depends on S. Taking these together we obtain our fourth postulate.

Postulate 4 (bounded exploration postulate). *For a database transformation t there exists a fixed, finite set T of access terms of t such that $\Delta(t, S_1) = \Delta(t, S_2)$ holds whenever the states S_1 and S_2 coincide over T.*

As in the sequential ASM thesis we continue calling the set T of access terms a *bounded exploration witness.* The only difference to the bounded exploration postulate for sequential algorithms in [10] is the use of access terms (β, α), whereas in the sequential ASM thesis only ground terms are considered. Access terms of the form (β, α) are actually equivalent to closed set comprehension terms $\{f(x_1, \dots, x_n) \mid \beta = \alpha\}$, i.e. they express first-order queries to the database similar to the relational calculus, and due to the fact that the database part of a state is a finite structure the set of locations defined by an access term is always finite. However, building terms on top of the state signature does not yet capture such terms. Access terms for the algorithmic part can still only be ground terms, otherwise finiteness cannot be guaranteed. Therefore, the modified Postulate 4 still expresses the same intention as the bounded exploration postulate for sequential algorithms does, i.e. only finitely many locations can be updated at a time, and these locations are determined by finitely many terms that appear in some way in the textual description of a database transformation.

3.6 Bounded Non-determinism

The last postulate addresses the question of how non-determinism is permitted in a database transformation. To handle this, we need to further clarify the relationship between access terms and states. As defined in the abstract state postulate, every state of a database transformation is a meta-finite structure consisting of two parts: the database part and algorithmic part, which are linked via a fixed, finite number of bridge functions. To restrict non-determinism in a database transformation t, we consider that *ground access terms* of t can access only the algorithmic part of a state, while *non-ground access terms* of t can access both the database and algorithmic parts of a state. Furthermore, variables in non-ground access terms are limited to range merely over the database part.

Example 8. Let us look back Example 2 again. We would have

- 2, 7.8, $+(3,9)$ and $\text{EVEN}(9)$ as ground access terms, and
- $(\text{PERSON}(x,y,z,z'),\mathbf{true})$, $(f_{num}(x),8)$, $(+(f_{num}(x),f_{num}(y)),20)$ and $(\text{EVEN}(f_{num}(x)),\mathbf{false})$ as non-ground access terms.

Given a meta-finite structure with the signature $\Sigma = \Sigma_{db} \cup \Sigma_a \cup \{f_1,...,f_\ell\}$ and the base set B, i.e., $B = B_{db} \cup B_a$, we now formally define access terms.

Definition 12. A *ground access term* is defined by the following rules:

- $\alpha \in B_a$ is a ground access term, and
- $f(\alpha_1,...,\alpha_n)$ for n-ary function symbol $f \in \Sigma_a$ and ground access terms $\alpha_1,...,\alpha_n$ is a ground access term.

A *non-ground access term* is a pair (β,α) of terms, in which at least one of them is a non-ground term inductively defined by applying function symbols from Σ over variables in accordance with the definition of a meta-finite structure as in Definition 6.

We define equivalent substructures in the following sense.

Definition 13. Given two structures S' and S of the same signature Σ, a structure S' is a *substructure* of the structure S (notation: $S' \preceq S$) if

- the base set B' of S' is a subset of the base set B of S, i.e., $B' \subseteq B$, and
- for each function symbol f of arity n in the signature Σ the restriction of $val_S(f(x_1,...,x_n))$ to B' results in $val_{S'}(f(x_1,...,x_n))$.

Substructures $S_1, S_2 \preceq S$ are *equivalent* (notation: $S_1 \equiv S_2$) if there exists an automorphism $\sigma \in Aut(S)$ with $\sigma(S_1) = S_2$. The *equivalence class* of a substructure S' in the structure S is the subset of all substructures of S which are equivalent to S'.

Example 9. Let us consider a simple ternary relation schema R. Suppose our database contains $R(a,a,b_1), R(b,b,a_1), R(c_1,c,c_2)$. Then $R(a,a,b_1)$ defines a substructure with base set $\{a,b_1\}$. This substructure is equivalent to the substructure $R(b,b,a_1)$, as the isomorphism defined by the permutation $(a,b)(b_1,a_1)$ just swaps the two substructures.

If, however, we have a second relation schema R', and the database contains only $R'(a,b_1)$ and $R'(c,c)$, then the restriction to $\{a,b_1\}$ defines a substructure containing $R(a,a,b_1)$ and $R'(a,b_1)$, whereas the restriction to $\{b,a_1\}$ defines a substructure $R(b,b,a_1)$ – these substructures are no longer equivalent.

If the database contained also $R'(b,a_1)$, the two restrictions would again define equivalent substructures. □

Now we need to discuss the relationship between access terms and states, and explain how non-determinism can be restricted in a database transformation via access terms. Let us start with the simple case that states have no bridge functions and thereby only elements in the database part of a state are accessible in parallel via the interpretation of non-ground access terms. As the abstract state postulate captures the genericity of database transformations (i.e., preserved under isomorphisms [3]), an isomorphism between two states gives rise to an isomorphism between their corresponding successor states. Consequently, whenever a substructure $S^{'}$ of the database part is preserved in some successor state, each substructure in the equivalence class of $S^{'}$ is preserved in some (possibly same) isomorphic successor state. This is because the automorphism that interchanges two equivalent substructures permutes successor states, according to Definition 13. Nevertheless, it is also possible that there exists a successor state, in which none of substructures in the equivalence class of $S^{'}$ is preserved.

Example 10. Let us consider the relation $R = \{(a,a,b_1),(b,b,a_1),(c_1,c,c_2)\}$ in Example 9 again.

- Suppose that we non-deterministically delete a tuple from R in a state S. Then there will be three successor states of S with $R = \{(b,b,a_1),(c_1,c,c_2)\}$, $R = \{(a,a,b_1),(c_1,c,c_2)\}$ or $R = \{(a,a,b_1),(b,b,a_1)\}$, respectively. It is clear to see that for the equivalent substructures $R(b,b,a_1)$ and $R(a,a,b_1)$, whenever one of them is preserved in some successor state, another one is preserved in some (possibly same) isomorphic successor state. Note that not all of the successor states are isomorphic.

- If we non-deterministically select two tuples from R in a state S and delete them. Then we will also get three successor states of S: $R = \{(b,b,a_1)\}$, $R = \{(a,a,b_1)\}$ or $R = \{(c_1,c,c_2)\}$. In this case, in terms of the equivalence class $\{(a,a,b_1),(b,b,a_1)\}$, none of the equivalent substructures are preserved in the successor state of S with $R = \{(c_1,c,c_2)\}$.

In the case that states have bridge functions, however, the situation becomes a bit tricky because bridge functions define substructures of the algorithmic part based on substructures of the database part. Thus non-determinism caused by non-deterministically selecting elements in the database part may also result in the non-deterministic changes on substructures of the algorithmic part. Nevertheless, the distinction between the database and algorithmic parts is that non-determinism cannot arise from the algorithmic part by selecting non-deterministically substructures in the algorithmic part of a state.

Example 11. For the relation $R = \{(a,a,b_1),(b,b,a_1),(c_1,c,c_2)\}$ in Example 9, we assume that there exists a bridge function $f_{num} = \{(a,4),(a_1,6),(b,3),(b_1,1)(c,5),(c_1,8),(c_2,7)\}$ and $\{\textsc{Even}, \textsc{Odd}, \textsc{Test}\} \subseteq \Sigma_a$. First we can use the non-ground access terms $(\textsc{Odd}(f_{num}(z)),\mathbf{true})$, $(\textsc{Even}(f_{num}(x)),\mathbf{true})$ and $(R(x,y,z),\mathbf{true})$ with the formula $R(x,y,z) \wedge \textsc{Even}(f_{num}(x)) \wedge \textsc{Odd}(f_{num}(z))$ to retrieve out the tuples (a,a,b_1) and (c_1,c,c_2) in R.

Then we can non-deterministically generate updates on function TEST by non-deterministically selecting one of these two tuples. For example, two update sets $\{(\text{TEST}(4), 1)\}$ and $\{(\text{TEST}(8), 7)\}$ may be created by using the access term $(\text{TEST}(f_{num}(x)), f_{num}(z))$ together with the formula $R(x, y, z) \wedge$ $\text{EVEN}(f_{num}(x))$ $\wedge \text{ODD}(f_{num}(z))$.

Therefore, for a state of database transformations, substructures of its algorithmic part may or may not be preserved in its successor states. This indeed can be explained under a broader view on equivalence classes, i.e., they are defined in terms of a state taking all of the database part, the algorithmic part and bridge functions into consideration. Then the rationale that whenever a substructure of a state is preserved in some successor state, each substructure in the equivalence class of that substructure is preserved in some (possibly same) isomorphic successor state can still be captured by the abstract state postulate in the same way as in the case without bridge functions.

Now we formalise the bounded non-determinism postulate to capture these ideas by properly defining the presence of non-ground access terms. In doing so, we put a severe restriction on the non-determinism in the transition relation τ_t.

Postulate 5 (bounded non-determinism postulate). *For a database transformation t, if there are states S_1, S_2 and $S_3 \in \mathcal{S}_t$ with $(S_1, S_2) \in \tau_t$, $(S_1, S_3) \in \tau_t$ and $S_2 \neq S_3$, then there exists a non-ground access term of the form (β, α) in the bounded exploration witness of t.*

According to this bounded non-determinism postulate, if a database transformation t over some state S_1 has non-determinism (i.e., $\Delta(t, S_1)$ contains more than one update set), then we must have a non-ground access term in the bounded exploration witness of t. Alternatively, if the bounded exploration witness of t contains only ground access terms, then t can access only the algorithmic part of a state and cannot have non-determinism. The bounded non-determinism postulate is motivated by the necessity of non-determinism in database queries and updates to permit identifier creation. This will become clear in the proof of our main result in Section 5, according to which the bounded non-determinism postulate enforces that only the bounded choice among database elements can be the source of non-determinism.

Remark 1. In [17, 18] Van den Bussche defined the notions of *determinacy* and *semi-determinism* - a determinate transformation preserves the input database, whereas a semi-deterministic transformation produces isomorphic outputs and thus preserves the input database up to an automorphism. In the sense of Van den Bussche an input database would define a substructure, and by the bounded non-determinism postulate preserving this substructure implies that each automorphism of the input database defines an isomorphism between the possible successor states, but not all of successor states will be isomorphic. Hence, database transformations characterised by five postulates subsume semi-deterministic transformations. In the same way they captures the insertion of new objects with a choice of identifiers as worked out for generic updates in [15].

3.7 Final Remarks

Naturally, for a database transformation the decisive part is the progression of the database part of states, whereas the algorithmic part could be understood as playing only a supporting role. Nonetheless, the postulates for database transformations in this section permit transformations, in which the major computation happens on the algorithmic part. In the extreme case we could even only manipulate the algorithmic part. This implies that our model actually subsumes all sequential algorithms. Furthermore, all extensions such as bounded non-determinism, meta-finite states, location operators and bounded exploration with non-ground terms only affect the database part. This will become more apparent in the next two sections, when we present a variant of ASMs capturing exactly database transformations as stipulated by the five postulates. On the other hand, our model of database transformations does not capture parallel algorithms, as the bounded exploration postulate excludes unbounded parallelism.

4 Database Abstract State Machines

In this section we define a variant of Abstract State Machines, called *Database Abstract State Machines*, and show that DB-ASMs satisfy the postulates of a database transformation. In the next section we will address the more challenging problem showing the converse of this result.

4.1 DB-ASM Rules and Update Sets Generated by Them

First we define DB-ASM rules r, and if S is a state, i.e. a Σ-structure for the signature Σ of r, we associate a set $\Delta(r, S)$ of update sets with r and S. For convenience, we also use the notation $\ddot{\Delta}(r, S)$ for a set of update multisets defined by r and S.

For the signature Σ we adopt the requirements of the abstract state postulate, i.e. it comprises a sub-signature Σ_{db} for the database part, a sub-signature Σ_a for the algorithmic part, and bridge functions $\{f_1, \dots, f_\ell\}$. For states we assume that the requirement in the abstract state postulate, according to which the restriction to Σ_{db} results in a finite structure, is satisfied. Furthermore, we assume a background in the sense of the background postulate being defined.

DB-ASM rules may involve variables, so in the following definition we also use the notations $\Delta(r, S, \zeta)$ for a set of update sets that depends on a variable assignment ζ, and analogously $\ddot{\Delta}(r, S, \zeta)$ for a set of update multisets. If ζ is a variable assignment, then $\zeta[x_1 \mapsto b_1, \dots, x_k \mapsto b_k]$ is another variable assignment defined by

$$\zeta[x_1 \mapsto b_1, \dots, x_k \mapsto b_k](x) = \begin{cases} b_i & \text{if } x = x_i (i = 1, \dots, k) \\ \zeta(x) & \text{else} \end{cases}$$

We refer to *database variables* as variables that must be interpreted by values in B_{db}. The notation $var(t)$ is used to denote the set of variables occurring in a term

t. Similar to free variables occurring in formulae we can define the set $fr(r)$ of free variables appearing in a DB-ASM rule r. A rule r is called *closed* if $fr(r) = \emptyset$.

Definition 14. The set $\mathcal{R}$ of *DB-ASM rules* over a signature $\Sigma = \Sigma_{db} \cup \Sigma_a \cup \{f_1, \dots, f_\ell\}$ and associated sets of update sets (with respect to states as in Postulate 2 with a background as in Postulate 3) are defined as follows:

- If $t_0, \dots, t_n$ are terms over Σ, and f is a n-ary dynamic function symbol in Σ, then $f(t_1, \dots, t_n) := t_0$ is a rule r in $\mathcal{R}$ called *assignment rule* with $fr(r) = \bigcup_{i=0}^{n} var(t_i)$. For a state S over Σ and a variable assignment ζ for $fr(r)$ we obtain
$$\Delta(r, S, \zeta) = \{\{(f(a_1, \dots, a_n), a_0)\}\}$$
with $a_i = val_{S,\zeta}(t_i)$ $(i = 0, \dots, n)$, and
$$\ddot{\Delta}(r, S, \zeta) = \{\langle (f(a_1, \dots, a_n), a_0)\rangle\}$$

- If φ is a Boolean term and $r' \in \mathcal{R}$ is a DB-ASM rule, then **if** φ **then** r' **endif** is a rule r in $\mathcal{R}$ called *conditional rule* with $fr(r) = fr(\varphi) \cup fr(r')$. For a state S over Σ and a variable assignment ζ for the variables in $fr(r)$, we obtain
$$\ddot{\Delta}(r, S, \zeta) = \begin{cases} \ddot{\Delta}(r', S, \zeta) & \text{if } val_{S,\zeta}(\varphi) = true \\ \emptyset & \text{else} \end{cases}$$
and
$$\Delta(r, S, \zeta) = \begin{cases} \Delta(r', S, \zeta) & \text{if } val_{S,\zeta}(\varphi) = true \\ \emptyset & \text{else} \end{cases}$$

- If φ is a Boolean term with only database variables, $\{x_1, \dots, x_k\} \subseteq fr(\varphi)$ and $r' \in \mathcal{R}$ is a DB-ASM rule, then **forall** $x_1, \dots, x_k$ **with** φ **do** r' **enddo** is a rule r in $\mathcal{R}$ called *forall rule* with $fr(r) = (fr(r') \cup fr(\varphi)) - \{x_1, \dots, x_k\}$. For a state S over Σ and a variable assignment ζ for the variables in $fr(r)$ let $\mathcal{B} = \{(b_1, \dots, b_k) \mid val_{S,\zeta[x_1 \mapsto b_1, \dots, x_k \mapsto b_k]}(\varphi) = true\}$ and $\mathcal{W}$ denote the set of mappings η from $\mathcal{B}$ to $\bigcup\{\Delta(r', S, \zeta[x_1 \mapsto b_1, \dots, x_k \mapsto b_k]) \mid (b_1, \dots, b_k) \in \mathcal{B}\}$ with $\eta(b_1, \dots, b_k) \in \Delta(r', S, \zeta[x_1 \mapsto b_1, \dots, x_k \mapsto b_k])$. Then each $\eta \in \mathcal{W}$ defines an update set $\Delta_\eta = \bigcup\{\eta(b_1, \dots, b_k) \mid (b_1, \dots, b_k) \in \mathcal{B}\}$, from which we obtain
$$\Delta(r, S, \zeta) = \{\Delta_\eta \mid \eta \in \mathcal{W}\}.$$
Analogously, let $\ddot{\mathcal{W}}$ denote the set of mappings $\ddot{\eta}$ from $\mathcal{B}$ to $\bigcup\{\ddot{\Delta}(r', S, \zeta[x_1 \mapsto b_1, \dots, x_k \mapsto b_k]) \mid (b_1, \dots, b_k) \in \mathcal{B}\}$ with $\ddot{\eta}(b_1, \dots, b_k) \in \ddot{\Delta}(r', S, \zeta[x_1 \mapsto b_1, \dots, x_k \mapsto b_k])$. Then each $\ddot{\eta} \in \ddot{\mathcal{W}}$ defines an update multiset $\ddot{\Delta}_{\ddot{\eta}} = \biguplus\{\ddot{\eta}(b_1, \dots, b_k) \mid (b_1, \dots, b_k) \in \mathcal{B}\}$, which finally gives
$$\ddot{\Delta}(r, S, \zeta) = \{\ddot{\Delta}_{\ddot{\eta}} \mid \ddot{\eta} \in \ddot{\mathcal{W}}\}.$$

- If $r_1, \dots, r_n$ are rules in $\mathcal{R}$, then the rule r defined as **par** $r_1 \dots r_n$ **endpar** is a rule in $\mathcal{R}$, called *parallel rule* with $fr(r) = \bigcup_{i=1}^{n} fr(r_i)$. For a state S over Σ and a variable assignment ζ for the variables in $fr(r)$ we obtain

  $$\Delta(r, S, \zeta) = \{\Delta_1 \cup \dots \cup \Delta_n \mid \Delta_i \in \Delta(r_i, S, \zeta) \text{ for } i = 1, \dots, n\}$$

  and

  $$\ddot{\Delta}(r, S, \zeta) = \{\ddot{\Delta}_1 \uplus \dots \uplus \ddot{\Delta}_n \mid \ddot{\Delta}_i \in \ddot{\Delta}(r_i, S, \zeta) \text{ for } i = 1, \dots, n\}.$$

- If φ is a Boolean term with only database variables, $\{x_1, \dots, x_k\} \subseteq fr(\varphi)$ and $r' \in \mathcal{R}$ is a DB-ASM rule, then **choose** $x_1, \dots, x_k$ **with** φ **do** r' **enddo** is a rule r in $\mathcal{R}$ called *choice rule* with $fr(r) = (fr(r') \cup fr(\varphi)) - \{x_1, \dots, x_k\}$. For a state S over Σ and a variable assignment ζ for the variables in $fr(r)$ let $\mathcal{B} = \{(b_1, \dots, b_k) \mid val_{S, \zeta[x_1 \mapsto b_1, \dots, x_k \mapsto b_k]}(\varphi) = true\}$. Then we obtain

  $$\Delta(r, S, \zeta) = \bigcup \{\Delta(r', S, \zeta[x_1 \mapsto b_1, \dots, x_k \mapsto b_k]) \mid (b_1, \dots, b_k) \in \mathcal{B}\}.$$

  and

  $$\ddot{\Delta}(r, S, \zeta) = \bigcup \{\ddot{\Delta}(r', S, \zeta[x_1 \mapsto b_1, \dots, x_k \mapsto b_k]) \mid (b_1, \dots, b_k) \in \mathcal{B}\}.$$

- If r_1, r_2 are rules in $\mathcal{R}$, then the rule r defined as **seq** r_1 r_2 **endseq** is a rule in $\mathcal{R}$, called *sequence rule* with $fr(r) = fr(r_1) \cup fr(r_2)$. For a state S over Σ and a variable assignment ζ for the variables in $fr(r)$ we obtain

  $$\Delta(r, S, \zeta) = \{\Delta_1 \oslash \Delta_2 \mid \Delta_1 \in \Delta(r_1, S, \zeta) \text{ and } \Delta_2 \in \Delta(r_2, S + \Delta_1, \zeta)\}$$

  with update sets defined as

  $$\Delta_1 \oslash \Delta_2 = \Delta_2 \cup \{(\ell, v) \in \Delta_1 \mid \neg \exists v'.(\ell, v') \in \Delta_2 \text{ and } v \neq v'\}$$

  and

  $$\ddot{\Delta}(r, S, \zeta) = \{\ddot{\Delta}_1 \oslash \ddot{\Delta}_2 \mid \ddot{\Delta}_1 \in \ddot{\Delta}(r_1, S, \zeta) \text{ and } \ddot{\Delta}_2 \in \ddot{\Delta}(r_2, S + AsSet(\ddot{\Delta}_1), \zeta)\}$$

  with update multisets defined as

  $$\ddot{\Delta}_1 \oslash \ddot{\Delta}_2 = \ddot{\Delta}_2 \uplus \langle (\ell, v) \in \ddot{\Delta}_1 \mid \neg \exists v'.(\ell, v') \in \ddot{\Delta}_2 \text{ and } v \neq v' \rangle.$$

- If $r' \in \mathcal{R}$ is a DB-ASM rule and θ is a location function that assigns location operators ρ to terms t with $var(t) \subseteq fr(r')$, then **let** $\theta(t) = \rho$ **in** r' **endlet** is a rule $r \in \mathcal{R}$ called *let rule* with $fr(r) = fr(r')$. For a state S over Σ and a variable assignment ζ for the variables in $fr(r)$ let $\ddot{\Delta}(r', S, \zeta) =$

$\{\ddot{\Delta}_1, \dots, \ddot{\Delta}_n\}$ with update multisets $\ddot{\Delta}_i = \ddot{\Delta}_i^{(t)} \uplus \ddot{\Delta}_i^-$ such that the first of these two multisubsets contains the updates to locations $val_{S,\zeta}(t)$, while the second one contains updates to all other locations. Define

$$\ddot{\Delta}_i^{(r)} = \langle (\ell, v) \mid \ell = val_{S,\zeta}(t), v = \rho(\langle v_1, \dots, v_k \mid (\ell, v_i) \in \ddot{\Delta}_i^{(t)}\rangle))\rangle \uplus \ddot{\Delta}_i^-$$

and

$$\Delta_i^{(r)} = \{(\ell, v) \mid \ell = val_{S,\zeta}(t), v = \rho(\langle v_1, \dots, v_k \mid (\ell, v_i) \in \ddot{\Delta}_i^{(t)}\rangle))\} \cup \Delta_i^-,$$

with $\Delta_i^- = \{(\ell, v) \mid (\ell, v) \in \ddot{\Delta}_i^-\}$. This finally gives

$$\ddot{\Delta}(r, S, \zeta) = \{\ddot{\Delta}_1^{(r)}, \dots, \ddot{\Delta}_n^{(r)}\} \quad \text{and} \quad \Delta(r, S, \zeta) = \{\Delta_1^{(r)}, \dots, \Delta_n^{(r)}\}.$$

Note that only assignment rules "create" updates in update sets and multisets, only choice rules introduce non-determinism, let rules reduce update multisets to update sets by letting updates to the same location collapse to a single update using assigned location operators, whereas all other rules just rearrange these updates into different sets and multisets, respectively. The sequence operator **seq** is associative, so we can also use more complex sequence rules **seq** $r_1 \dots r_n$ **endseq**.

Example 12. Consider the following DB-ASM rule

```
forall $x$ with $\exists z.R(x,x,z)$
do
    let $\theta(f(x)) = \text{sum}$ in
        forall $y$ with $R(x,x,y)$
        do
            $f(x) := 1$
        enddo
    endlet
enddo
```

using sum as a shortcut for the location operator $(id, +, id)$. If the state contains the tuples $R(a,a,b), R(a,a,b'), R(c,c,a'), R(b,b,c), R(b,b,a'), R(b,b,b)$, $R(b',a',a)$, then first the update multisets $\langle (a,1),(a,1)\rangle, \langle (c,1)\rangle, \langle (b,1),(b,1),(b,1)\rangle$ are produced by means of the forall rules, which are then collapsed to the update set $\{(a,2),(c,1),(b,3)\}$ using the sum-operator in the let rule. Thus, for x such that there are tuples $R(x,x,y)$ in the database, then number of such tuples is counted and assigned to $f(x)$.

Let us denote the inner and outer forall rules as r_1 and r_2, respectively. Then we have

- $fr(r_1) = (\{x\} \cup fr(R(x,x,y))) - \{y\} = \{x\}$ and
- $fr(r_2) = (\{x\} \cup fr(\exists z.R(x,x,z))) - \{x\} = \emptyset$.

Hence this DB-ASM rule is closed.

Lemma 3. *Let r be a DB-ASM rule and $\sigma : S_1 \to S_2$ be an isomorphism between states S_1 and S_2. Let $S_1' = S_1 + \Delta$ be a successor state of S_1 for some $\Delta \in \Delta(r, S_1)$. Then we have $\sigma(\Delta) \in \Delta(r, S_2)$, and $\sigma : S_1' \to S_2' = S_2 + \sigma(\Delta)$ is an isomorphism between the successor states S_1' and S_2'.*

Proof. We proceed by structural induction on the rule r. So we start with an assignment rule $f(t_1, \dots, t_n) := t_0$. Then we must take $\Delta = \{(f(a_1, \dots, a_n), a_0)\}$ with $a_i = val_{S_1}(t_i)$ for $i = 0, \dots, n$. Then we have

$$val_{S_1'}(\ell) = \begin{cases} a_0 & \text{if } \ell = f(a_1, \dots, a_n) \\ val_{S_1}(\ell) & \text{else} \end{cases}$$

for any location ℓ. With $\sigma(\Delta) = \{(f(\sigma(a_1), \dots, \sigma(a_n)), \sigma(a_0))\}$ we obtain $val_{S_2'}(\sigma(\ell)) = \begin{cases} \sigma(a_0) & \text{if } \ell = f(a_1, \dots, a_n) \\ val_{S_2}(\sigma(\ell)) & \text{else} \end{cases} = \sigma(val_{S_1'}(\ell))$, which gives $S_2' = \sigma(S_1')$ as desired. The same argument applies to update multisets.

For a conditional rule $r = \textbf{if } \varphi \textbf{ then } r' \textbf{ endif}$ let $\zeta_2 = \sigma(\zeta_1)$. Then $val_{S_1,\zeta_1}(\varphi) = true$ iff $val_{S_2,\zeta_2}(\varphi) = true$. This implies

$$\begin{aligned} & S_2 + \Delta(r, S_2, \zeta_2) \\ = & \begin{cases} S_2 + \sigma(\Delta(r', S_1, \zeta_1)) & \text{if } val_{S_2,\zeta_2}(\varphi) = true \\ S_2 & \text{else} \end{cases} \\ = & \begin{cases} \sigma(S_1 + \Delta(r', S_1, \zeta_1)) & \text{if } val_{S_1,\zeta_1}(\varphi) = true \\ \sigma(S_1) & \text{else} \end{cases} \\ & = \sigma(S_1 + \Delta(r, S_1, \zeta_1)). \end{aligned}$$

The other cases are proven analogously.

□

4.2 Database Abstract State Machines

We are now prepared to define DB-ASMs and show that they satisfy the five postulates for database transformations from the previous section.

Definition 15. A *Database Abstract State Machine* (DB-ASM) $\mathcal{M}$ over signature Σ as in Postulate 2 and with a background as in Postulate 3 consists of

- a set $\mathcal{S}_\mathcal{M}$ of states over Σ, non-empty subsets $\mathcal{I}_\mathcal{M} \subseteq \mathcal{S}_\mathcal{M}$ of initial states and $\mathcal{F}_\mathcal{M} \subseteq \mathcal{S}_\mathcal{M}$ of final states, satisfying the requirements in Postulate 2,
- a closed DB-ASM rule $r_\mathcal{M}$ over Σ, and
- a binary relation $\tau_\mathcal{M}$ over $\mathcal{S}_\mathcal{M}$ determined by $r_\mathcal{M}$ such that

  $$\{S_{i+1} \mid (S_i, S_{i+1}) \in \tau_\mathcal{M}\} = \{S_i + \Delta \mid \Delta \in \Delta(r_\mathcal{M}, S_i)\}$$

  holds.

Theorem 6. *Each DB-ASM $\mathcal{M}$ defines a database transformation t with the same signature and background as $\mathcal{M}$.*

Proof. We have to show that the five postulates for database transformations are satisfied. As for the sequential time and background postulates 1 and 3, these are already built into the definition of a DB-ASM. The same holds for the abstract state postulate 2 as far as the definition of states is concerned, and the preservation of isomorphisms follows from Lemma 3. Thus, we have to concentrate on the bounded exploration and bounded non-determinism postulates 4 and 5.

Regarding bounded exploration we noted above that assignment rules within a DB-ASM rule r that defines $\tau_{\mathcal{M}}$ are decisive for a set $\Delta(r, S)$ of update sets over any state S. Hence, if $f(t_1, \dots, t_n) := t_0$ is an assignment rule occurring within r, and $val_{S,\zeta}(t_i) = val_{S',\zeta}(t_i)$ holds for all $i = 0, \dots, n$ and all variable assignments ζ that have to be considered, then we obtain $\Delta(r, S) = \Delta(r, S')$.

We use this to define a bounded exploration witness T. If t_i is ground, we add the access term $\alpha = t_i$ to T. If t_i is not ground, then the corresponding assignment rule must appear within the scope of forall and choice rules introducing the database variables in t_i, as r is closed. Thus, variables in t_i are bound by a Boolean term φ, i.e. for $fr(t_i) = \{x_1, \dots, x_k\}$ the relevant variable assignments are $\zeta = \{x_1 \mapsto b_1, \dots, x_k \mapsto b_k\}$ with $val_{S,\zeta}(\varphi) = true$. Bringing φ into a form that only uses conjunction, negation and existential quantification with atoms $\beta_i = \alpha_i$ $(i = 1, \dots, \ell)$, we can extract a set of access terms $\{(\beta_1, \alpha_1), \dots, (\beta_\ell, \alpha_\ell)\}$ such that if S and S' coincide on these access terms, they will also coincide on the formula φ. This is possible, as we evaluate access terms by sets, so conjunction corresponds to union, existential quantification to projection, and negation to building the (finite) complement. We add all the access terms $(\beta_1, \alpha_1), \dots, (\beta_\ell, \alpha_\ell)$ to T.

More precisely, if φ is a conjunction $\varphi_1 \wedge \varphi_2$, then $\Delta(r, S_1) = \Delta(r, S_2)$ will hold, if $\{(b_1, \dots, b_k) \mid val_{S_1,\zeta}(\varphi) = true\} = \{(b_1, \dots, b_k) \mid val_{S_2,\zeta}(\varphi) = true\}$ holds (with $\zeta = \{x_1 \mapsto b_1, \dots, x_k \mapsto b_k\}$). If T_i is a set of access terms such that whenever S_1 and S_2 coincide on T_i, then $\{(b_1, \dots, b_k) \mid val_{S_1,\zeta}(\varphi_i) = true\} = \{(b_1, \dots, b_k) \mid val_{S_2,\zeta}(\varphi_i) = true\}$ will hold $(i = 1, 2)$, then $T_1 \cup T_2$ is a set of access terms such that whenever S_1 and S_2 coincide on $T_1 \cup T_2$, then $\{(b_1, \dots, b_k) \mid val_{S_1,\zeta}(\varphi) = true\} = \{(b_1, \dots, b_k) \mid val_{S_2,\zeta}(\varphi) = true\}$ will hold.

Similarly, a set of access terms for ψ with the desired property will also be a witness for $\varphi = \neg\psi$, and $\bigcup_{b_{k+1} \in B_{db}} T_{b_{k+1}}$ with sets of access terms $T_{b_{k+1}}$ for $\psi[x_{k+1}/t_{k+1}]$ with $val_S(t_{k+1}) = b_{k+1}$ defines a finite set of access terms for $\varphi = \exists x_{k+1}\psi$. In this way, we can restrict ourselves to atomic formulae, which are equations and thus give rise to canonical access terms.

Then by construction, if S and S' coincide on T, we obtain $\Delta(r, S) = \Delta(r, S')$. As there are only finitely many assignment rules within r and only finitely many choice and forall rules defining the variables in such assignment rules, the set T of access terms must be finite, i.e. r satisfies the bounded exploration postulate.

Regarding bounded non-determinism, assuming that $\mathcal{M}$ does not satisfy the bounded non-determinism postulate. It means that there does not exist any non-ground access term (β, α) in T even when $\Delta(r, S)$ contains more than one update sets. However, according to our remark above r must contain a choice rule **choose** $x_1, \dots, x_k$ **with** φ **do** r' **enddo**. Hence, it implies that there exist at least one non-ground access term in T contradicting our assumption. □

5 A Characterisation Theorem

In this section we want to show that DB-ASMs capture all database transformations. This constitutes the converse of Theorem 6, i.e. that every database transformation can be behaviouraly simulated by a DB-ASM. We start with some preliminaries that are only slight adaptations of corresponding definitions and results for the sequential ASM-thesis, except that the term *critical value* has to be defined differently due to the use of variables in access terms. We then show first that a one-step transition from a state to successor states can be expressed by a DB-ASM rule. Here we rely heavily on the abstract state postulate and the bounded non-determinism postulate, which allows us to deal with the restricted non-determinism appropriately.

In a second step we generalise the proof to the complete database transformation using a construction that is similar to the one used in the sequential ASM-thesis. Again, the fact that our bounded exploration witness contains non-ground terms makes up most of the difficulty.

5.1 Critical Terms and Critical Elements

Throughout this section we only deal with consistent update sets, which define the progression of states in a run. We now start providing the key link from updates as implied by the state transitions to DB-ASM rules. Same as the previous subsection this is only a slight extension to the work done for the sequential ASM thesis.

Definition 16. Let T be a bounded exploration witness for the database transformation t. A term that is constructed out of the subterms of $\alpha \in T$ and variables $x_1, \dots, x_k$, for which there are access terms $(\beta_1, \alpha_1), \dots, (\beta_\ell, \alpha_\ell) \in T$ such that $\bigcup_{i=1}^{\ell} fr(\beta_i) \cup fr(\alpha_i) = \{x_1, \dots, x_k\}$ holds is called a *critical term.*

This definition differs from the one given in [10] in that we consider also non-ground terms. For access terms in T we cannot simply require closure under subterms, as coincidence of structures on T does not carry over to subterms of "associative" access terms (β, α). Therefore, we need a different approach to define critical values.

If γ is a critical term, let $(\beta_1, \alpha_1), \dots, (\beta_\ell, \alpha_\ell)$ be the access terms used in its definition. For a state S choose $b_1, \dots, b_k \in B_{db}$ with $val_{S,\zeta}(\beta_i) = val_{S,\zeta}(\alpha_i)$ with $\zeta = \{x_1 \mapsto b_1, \dots, x_k \mapsto b_k\}$ for $i = 1, \dots, \ell$, and let $a = val_{S,\{x_1 \mapsto b_1, \dots, x_k \mapsto b_k\}}(\gamma)$.

Definition 17. For each state S of a database transformation t, let $C_S = \{val_S(\alpha) \mid \alpha \in T\} \cup \{true, false, \bot\}$ and $B_S = \{a_i \mid f(a_1, \dots, a_n) \in val_S(\beta, \alpha)$ for some access term $(\beta, \alpha) \in T\}$. Then $\bar{C}_S$ is the *background closure* of $C_S \cup B_S$ containing all complex values that can be constructed out of $C_S \cup B_S$ using the constructors and function symbols (interpreted in S) in V_K. The elements of $\bar{C}_S$ are called the *critical elements* of S.

The following lemma and its proof are analogous to the result in [10, Lemma 6.2].

Lemma 4. *For all updates $(f(a_1, \dots, a_n), a_0) \in \Delta(t, S, S')$ for $(S, S') \in \tau_t$ the values $a_0, \dots, a_n$ are critical elements of S.*

Proof. Assume one of the a_i is not critical. Then choose a structure S_1 by replacing a_i with a fresh value b without changing anything else. Thus, S_1 is a state isomorphic to S by the abstract state postulate.

Let (β, α) be an access term in T. Then we must have $val_S(\beta, \alpha) = val_{S_1}(\beta, \alpha)$, so S and S_1 coincide on T. From the bounded exploration postulate we obtain $\Delta(t, S) = \Delta(t, S_1)$ and thus $(f(a_1, \dots, a_n), a_0) \in \Delta(t, S_1, S_1')$ for some $(S_1, S_1') \in \tau_t$.

However, a_i does not appear in the structure S_1, and hence cannot appear in S_1' either, nor in $\Delta(t, S_1, S_1')$, which gives a contradiction. □

5.2 Rules for One-Step Updates

In [10] it is a straighforward consequence of Lemma 6.2 that individual updates can be represented by assignments rules, and consistent update sets by par-blocks of assignment rules. In our case showing that $\Delta(t, S)$ can be represented by a DB-ASM rule requires a bit more work, which relies heavily on the abstract state postulate and the bounded non-determinism postulate. We address this in the next lemma.

Lemma 5. *Let t be a database transformation. For every state $S \in \mathcal{S}_t$ there exists a rule r_S such that $\Delta(t, S) = \Delta(r_S, S)$, and r_S only uses critical terms.*

Proof. $\Delta(t, S)$ is a set of update sets. Let $\{S_1, \dots, S_m\} = \{S' \mid (S, S') \in \tau_t\}$. Then $\Delta(t, S) = \{\Delta(t, S, S_i) \mid 1 \leq i \leq m\}$.

Now consider any update $u = (f(a_1, \dots, a_n), a_0) \in \Delta(t, S, S_i)$ for some $i \in \{1, \dots, m\}$. According to Lemma 4 the values $a_0, \dots, a_n$ are critical and hence representable by terms involving variables from access terms in T, i.e. $a_i = val_{S,\zeta}(t_i)$ with either $fr(t_i) \subseteq \{x_1, \dots, x_k\}, \zeta = \{x_1 \mapsto b_1, \dots, x_k \mapsto b_k\}$ and

$$(b_1, \dots, b_k) \in \mathcal{B}_u = \{(b_1, \dots, b_k) \in B_{db}^k \mid \bigwedge_{1 \leq i \leq \ell} val_{S,\zeta}(\beta_i) = val_{S,\zeta}(\alpha_i)\}$$

with access terms $(\beta_i, \alpha_i) \in T$ $(i = 1, \dots, \ell)$ and $fr(\beta_i) \subseteq \{x_1, \dots, x_k\}$, or t_i is a ground critical term.

Therefore, we distinguish two cases:

I. At least one of the terms $t_0, \dots, t_n$ is not a ground term.

II. All terms $t_0, \dots, t_n$ are ground terms.

Case I. We first assume that none of terms $t_0, \dots, t_n$ contain location operators. The access terms (β_i, α_i) define a finite set of locations

$$\begin{aligned} L \quad = \quad & \{f(a_1, \dots, a_n) \mid a_i = val_{S,\zeta}(t_i) \text{ for } i = 1, \dots, n, \text{ and} \\ & \zeta = \{x_1 \mapsto b_1, \dots, x_k \mapsto b_k\} \text{ for } (b_1, \dots, b_k) \in \mathcal{B}_u\}. \end{aligned}$$

However, instead of looking at updates at these locations we switch to a relational perspective, i.e. we replace $f \in \Sigma$ with arity n by a relation symbol Rf of arity $n+1$, so $f_S(a_1, \dots, a_n) = a_0$ holds iff $Rf_S(a_1, \dots, a_n, a_0) = true$. A non-trivial update $u = (f(a_1, \dots, a_n), a_0)$ is accordingly represented by two relational updates

$$u_d = (Rf(a_1, \dots, a_n, f_S(a_1, \dots, a_n)), false) \text{ and } u_i = (Rf(a_1, \dots, a_n, a_0), true).$$

So, instead of locations in L we consider locations in $L_{pre} \cup L_{post}$ with

$$\begin{aligned} L_{pre} =& \{Rf(a_1, \dots, a_n, a_0) \mid a_i = val_{S,\zeta}(t_i) \text{ for } 1 \leq i \leq n, \\ & a_0 = val_{S,\zeta}(f(t_1, \dots, t_n)) \text{ for } \zeta = \{x_1 \mapsto b_1, \dots, x_k \mapsto b_k\} \\ & \text{and } (b_1, \dots, b_k) \in \mathcal{B}_u\} \end{aligned}$$

and

$$\begin{aligned} L_{post} =& \{Rf(a_1, \dots, a_n, a_0) \mid a_i = val_{S,\zeta}(t_i) \text{ for } 0 \leq i \leq n \\ & \text{for } \zeta = \{x_1 \mapsto b_1, \dots, x_k \mapsto b_k\} \text{ and } (b_1, \dots, b_k) \in \mathcal{B}_u\}. \end{aligned}$$

Furthermore, we may assume that the set $\mathcal{B}_u$ is minimal in the sense that we may not find additional access terms that would define a subset $\mathcal{B}'_u \subsetneq \mathcal{B}_u$ still containing the value tuple $(b_1, \dots, b_k)$ that is needed to define the update u.

Then each tuple $(a_1, \dots, a_n, a_0) \in L_{pre} \cup L_{post}$ defines a substructure of S with base set $B' = \{a_0, \dots, a_n, true, false\}$ and all functions (in fact: relations) restricted to this base set. In doing so all substructures defined by L_{pre} (and analogously by L_{post}) are pairwise equivalent, and the induced isomorphisms are defined by permutations of tuples in $\mathcal{B}_u$. If they were not equivalent, we could find a distinguishing access structure $(\beta_{\ell+1}, \alpha_{\ell+1}) \in T$ that would define a subset $\mathcal{B}'_u \subsetneq \mathcal{B}_u$ thereby violating the minimality assumption for $\mathcal{B}_u$. Let E_{pre} and E_{post} denote these equivalence classes of substructures, respectively.

If $\ell' \in L$ is not updated in S_i – hence, corresponding locations in L_{pre} and L_{post} are neither updated – then the substructures $S_{\ell'}$ in E_{pre} (and E_{post}, respectively) that are defined by ℓ' are preserved in S_i, i.e. $S_{\ell'} \preceq S_i$. From Lemma 2 we can get that $\sigma(\Delta(t, S)) = \Delta(t, \sigma(S)) = \Delta(t, S)$ for the case that σ is an automorphism of S. It means that for an update in $\Delta(t, S, S_i)$ there is a translated update (by means of σ) in $\Delta(t, S, \sigma(S_i))$. Then we conclude that for every $\ell'' \in L$ there is some successor state $S_j = \sigma'(S_i)$ of S with $S_{\ell''} \preceq S_j$ for an automorphism σ' of S that maps $S_{\ell'}$ to $S_{\ell''}$, and each $S_{\ell''} \in E_{pre}$ (and $S_{\ell''} \in E_{post}$, respectively) is preserved in some S_j. Thus, there is some successor state S_j of S with $S_\ell \preceq S_j$ for $u = (\ell, a_0)$.

Then, we obtain two subcases:

1) If ℓ is updated in all $S_1, \dots, S_m$, i.e. there exist values $a_0^1, \dots, a_0^m$ with $(\ell, a_0^i) \in \Delta(t, S, S_i)$ for all $i = 1, \dots, m$, then all $\ell' \in L$ are also updated in all S_i. If (ℓ, a_0^i) is represented by the assignment rule $f(t_1, \dots, t_n) := t_0^i$ with $x_1, \dots, x_k$ interpreted by $(b_1, \dots, b_k) \in \mathcal{B}_u$, then the fact that almost all substructures defined by these interpretations of $t_1, \dots, t_n$ and any value other than a_0^i are preserved – and hence by virtue of Lemma 2 as we explained before equivalent substructures are preserved in the other S_j – implies that each instantiation of the rule $f(t_1, \dots, t_n) := t_0^i$ with values from $\mathcal{B}_u$ defines an update in one of the update sets $\Delta(t, S, S_j)$. Hence these updates can be collectively represented by the rule

   $$\textbf{choose } x_1, \dots, x_k \textbf{ with } \beta_1(\vec{x}_1) = \alpha_1(\vec{x}_1) \wedge \cdots \wedge \beta_\ell(\vec{x}_\ell) = \alpha_\ell(\vec{x}_\ell)$$
   $$\textbf{do } f(t_1, \dots, t_n) := t_0^i \textbf{ enddo}$$

   Here the $\vec{x}_1, \dots, \vec{x}_\ell$ denote vectors of variables among $x_1, \dots, x_k$ appearing in $\beta_1, \dots, \beta_\ell$, respectively. In case all the terms t_0^i for $i = 1, \dots, m$ are identical (to say t_0), we obtain in fact the rule $r_S^{(u)}$ as

   $$\textbf{choose } x_1, \dots, x_k \textbf{ with } \beta_1(\vec{x}_1) = \alpha_1(\vec{x}_1) \wedge \cdots \wedge \beta_\ell(\vec{x}_\ell) = \alpha_\ell(\vec{x}_\ell)$$
   $$\textbf{do } f(t_1, \dots, t_n) := t_0 \textbf{ enddo}$$

   In general, however, it is possible that different terms t_0^i must be chosen, so all updates to locations in L are represented by the rule $r_S^{(u)}$, which becomes

   ```
   choose $x_1^{(1)}, \dots, x_k^{(1)}, \dots, x_1^{(m)}, \dots, x_k^{(m)}$
   with $\bigwedge_{1 \le j_1 < j_2 \le m} (x_1^{(j_1)}, \dots, x_k^{(j_1)}) \neq (x_1^{(j_2)}, \dots, x_k^{(j_2)})$
           $\wedge \bigwedge_{1 \le j \le m} \beta_1(\vec{x}_1^{(j)}) = \alpha_1(\vec{x}_1^{(j)}) \wedge \cdots \wedge \beta_\ell(\vec{x}_\ell^{(j)}) = \alpha_\ell(\vec{x}_\ell^{(j)})$
   do par
     $f(t_1, \dots, t_n)[x_1^{(1)}/x_1, \dots, x_k^{(1)}/x_k] := t_0^1[x_1^{(1)}/x_1, \dots, x_k^{(1)}/x_k]$
       $\vdots$                $\vdots$
     $f(t_1, \dots, t_n)[x_1^{(m)}/x_1, \dots, x_k^{(m)}/x_k] := t_0^m[x_1^{(m)}/x_1, \dots, x_k^{(m)}/x_k]$
   endpar enddo
   ```

2) If only ℓ is updated in S_i, but no other $\ell'' \in L$ is, then, for each $\ell' \in L$, only ℓ' is updated in some S_j for $j \in [1, m]$, which is isomorphic to S_i. Analogously, if only i locations in L are updated in S_i, then for any $\{\ell_1, ..., \ell_i\} \subseteq L$ there is some state S_j for $j \in [1, m]$, in which only locations $\ell_1 \dots \ell_i$ are updated. Using exactly the same arguments as in case 1), we now can represent these updates collectively by the rule $r_S^{(u)}$, which now becomes

```
choose $x_1^{(1)}, \dots, x_k^{(1)}, \dots, x_1^{(i)}, \dots, x_k^{(i)}$
with $\bigwedge_{1 \le j_1 < j_2 \le i} (x_1^{(j_1)}, \dots, x_k^{(j_1)}) \neq (x_1^{(j_2)}, \dots, x_k^{(j_2)})$
        $\wedge \bigwedge_{1 \le j \le i} \beta_1(\vec{x}_1^{(j)}) = \alpha_1(\vec{x}_1^{(j)}) \wedge \dots \wedge \beta_\ell(\vec{x}_\ell^{(j)}) = \alpha_\ell(\vec{x}_\ell^{(j)})$
do par
  $f(t_1, \dots, t_n)[x_1^{(1)}/x_1, \dots, x_k^{(1)}/x_k] := t_0^1[x_1^{(1)}/x_1, \dots, x_k^{(1)}/x_k]$
          $\vdots$                    $\vdots$
  $f(t_1, \dots, t_n)[x_1^{(i)}/x_1, \dots, x_k^{(i)}/x_k] := t_0^i[x_1^{(i)}/x_1, \dots, x_k^{(i)}/x_k]$
endpar enddo
```

By exploiting Lemma 2, we showed how to create a proper choice rule with respect to $\mathcal{B}_u$ that is minimal. However, the created choice rule does not capture all update sets in $\Delta(t, S)$. If there exists another update in an update set that is not in the orbit of $\Delta(t, S, S_i)$ ($1 \le i \le m$) under σ, we can use the same argument to obtain another choice rule. As the orbits are disjoint, we end up with a choice of choice rules, which can be combined into a single choice rule. The underlying condition for constructing such a single choice rule is the finiteness of $\Delta(t, S)$, i.e., there are only finitely many update sets created by t over state S. This can be assured by Lemma 4 and the bounded non-determinism postulate. Consequently there can only be finitely many successor states for each state S, depending on the database part of S that is a finite structure.

Now we revise the previous assumption that none of terms $t_0, \dots, t_n$ contain location operators to a general case, i.e., location operators may appear in the terms $t_0, ..., t_n$ of an assignment rule $f(t_1, ..., t_n) := t_0$. Let f_ℓ be a unary function symbol such that $x_{t_i} = f_\ell(i)$, then, without loss of generality, we can replace the terms $t_1, ..., t_n$ of an assignment rule $f(t_1, ..., t_n) := t_0$ with the variables $x_{t_1}, ..., x_{t_n}$, such that

```
seq
  par
    $x_{t_1} := t_1$
        $\vdots$
    $x_{t_n} := t_n$
  endpar
  $f(x_{t_1}, ..., x_{t_n}) := t_0$
endseq
```

It means that we can simplify the construction of rules for updates which may correspond to terms with location operators by only considering the case that location operators appear at the right hand side of an assignment rule. If a term t_i ($i \in [1, n]$) at the left hand side contains a location operator, by the above translation, we may treat it as being a term at the right hand side of another assignment rule again.

Suppose that the outermost function symbol of term t_0 is a location operator ρ, e.g.,$t_0 = \rho(m)$ where $m = \langle t_0^{'} |$ for all values $\overline{a} = (a_1, ..., a_p)$ in $\overline{y} = (y_1, ..., y_p)$ such that $val_{S,\zeta[x_1 \mapsto b_1, \ldots, x_k \mapsto b_k]}(\varphi(\overline{x}, \overline{y})) = true \rangle$, and $\overline{x}$ denotes a tuple of variables among $x_1, ..., x_k$. Then for each assignment rule $f(x_{t_1}, ..., x_{t_n}) := t_0$ in which t_0 contains a location operator as described before, we can construct the following rule to remove the location operator ρ by a let rule and a forall rule:

```
let $\theta(f(x_{t_1}, ..., x_{t_n})) = \rho$ in
    forall $y_1,...,y_p$ with $\varphi(\overline{x}, \overline{y})$
    do
        $f(x_{t_1}, ..., x_{t_n}) := t_0^{'}$
    enddo;
endlet
```

This construction can be conducted iteratively. If the outermost function symbol of the above term $t_0^{'}$ is a location operator, then we need to construct a rule in a similar way to replace the assignment rule $f(x_{t_1}, ..., x_{t_n}) := t_0^{'}$. This procedure continues until the right hand side of an assignment rule is a term without any location operator.

Case II. In case of a simple update $f(t_1, \ldots, t_n) := t_0$ without free variables we consider the substructure defined by $\{a_1, \ldots, a_n, val_S(f(t_1, \ldots, t_n)), true, false\}$ as before. However, in this case it is the only substructure in its equivalence class. Furthermore, the substructure can be represented by ground access terms. According to the bounded non-determinism postulate, we know that, ground access terms can access only the algorithmic part of a state and there is no non-determinism. Consequently, $(f(a_1, \ldots, a_n), a_0) \in \Delta(t, S, S_i)$ for all $i = 1, \ldots, m$, and these updates can be collectively represented by the simple assignment rule $r_S^{(u)}$, which now becomes

$$f(t_1, \ldots, t_n) := t_0.$$

Finally, we construct r_S by using the **par**-construct:

$$r_S = \textbf{par } r_S^{(u_1)} \ldots r_S^{(u_p)} \textbf{ endpar}$$

for $\{u_1, \ldots, u_p\} = \bigcup_{i=1}^{m} \Delta(t, S, S_i)$. □

5.3 Rules for Multiple-Steps Updates

Let us now extend Lemma 5 to the construction of a DB-ASM rule that captures the complete behaviour of a database transformation t. According to Definition 16, a set of critical terms can be obtained from a bounded exploration witness. For this fix a bounded exploration witness T and the set CT of critical terms derived from it. Furthermore, for a state S of t fix the rule r_S as in Lemma 5.

For $\gamma \in CT$ let $(\beta_1, \alpha_1), \dots, (\beta_\ell, \alpha_\ell)$ be the access terms in T defining $fr(\gamma) = \{x_1, \dots, x_k\}$. For a state $S \in \mathcal{S}_t$ define

$$val_S(\gamma) = \{val_{S,\zeta}(\gamma) \mid \zeta = (x_1 \mapsto b_1, \dots, x_k \mapsto b_k) \text{ and } \bigwedge_{1 \le i \le \ell} val_{S,\zeta}(\beta_i) = val_{S,\zeta}(\alpha_i)\}.$$

The following two lemmata extend Lemma 5 first to state that coincide with S on critical terms, then to isomorphic states.

Lemma 6. *Let $S, S' \in \mathcal{S}_t$ be states that coincide on the set CT of critical terms. Then $\Delta(r_S, S') = \Delta(t, S')$ holds.*

Proof. As S and S' coincide on CT, they also coincide on T, which gives $\Delta(t, S) = \Delta(t, S')$ by the bounded exploration postulate. Furthermore, we have $\Delta(r_S, S) = \Delta(t, S)$ by Lemma 5. As r_S uses only critical terms, the updates produced in state S must be the same as those produced in state S', i.e. $\Delta(r_S, S) = \Delta(r_S, S')$, which proves the lemma. □

Lemma 7. *Let S, S_1, S_2 be states with S_1 isomorphic to S_2 and $\Delta(r_S, S_2) = \Delta(t, S_2)$. Then also $\Delta(r_S, S_1) = \Delta(t, S_1)$ holds.*

Proof. Let σ denote an isomorphism from S_1 to S_2. Then $\Delta(r_S, S_2) = \sigma(\Delta(r_S, S_1))$ holds by Lemma 2, and the same applies to $\Delta(t, S_2) = \sigma(\Delta(t, S_1))$. As we presume $\Delta(r_S, S_2) = \Delta(t, S_2)$, we obtain $\sigma(\Delta(r_S, S_1)) = \sigma(\Delta(t, S_1))$ and hence $\Delta(r_S, S_1) = \Delta(t, S_1)$, as σ is an isomorphism. □

Next, in the spirit of [10] we want to extend the equality of sets of update sets for t and r_S to a larger class of states by exploiting the finiteness of the bounded exploration witness T. For this we define the notion of T-equivalence similar to the corresponding notion for the sequential ASM thesis, with the difference that in our case we cannot take T, but must base our definition and the following lemma on CT.

Definition 18. States $S, S' \in \mathcal{S}_t$ are called *T-similar* iff $E_S = E_{S'}$ holds, where E_S is an equivalence relation on CT defined by

$$E_S(\gamma_1, \gamma_2) \Leftrightarrow val_S(\gamma_1) = val_S(\gamma_2).$$

Lemma 8. *We have $\Delta(r_S, S') = \Delta(t, S')$ for every state S' that is T-similar to S.*

Proof. Replace every element in S' that also belongs to S by a fresh element. This defines a structure S_1 isomorphic to S' and disjoint from S. By the abstract state postulate S_1 is a state of t. Furthermore, by construction S_1 is also T-similar to S' and hence also to S.

Now define a structure S_2 isomorphic to S_1 such that $val_{S_2}(\gamma) = val_S(\gamma)$ holds for all critical terms $\gamma \in CT$. This is possible, as S and S_1 are T-similar, i.e. we have $val_S(\gamma_1) = val_S(\gamma_2)$ iff $val_{S_1}(\gamma_1) = val_{S_1}(\gamma_2)$ for all critical terms γ_1, γ_2. By the abstract state postulate S_2 is also a state of t.

Using Lemma 6 we conclude $\Delta(r_S, S_2) = \Delta(t, S_2)$, and by Lemma 7 we obtain $\Delta(r_S, S') = \Delta(t, S')$ as claimed. □

We are now able to prove our main result, first generalising Lemma 5 to multiple-steps updates in the next lemma, from which the proof of the main characterisation theorem is straightforward.

Lemma 9. *Let t be a database transformation with signature Σ. Then there exists a DB-ASM rule r over Σ, with same background as t such that $\Delta(r, S) = \Delta(t, S)$ holds for all states $S \in \mathcal{S}_t$.*

Proof. In order to decide whether equivalence relations E_S and $E_{S'}$ coincide for states $S, S' \in \mathcal{S}_t$ it is sufficient to consider the subset $CT' \subseteq CT$ defined by the bounded exploration witness T as in Definition 16. Hence, as T is finite, CT' is also finite, and consequently there can only be finitely many such equivalence relations. Let these be $E_{S_1}, \dots, E_{S_n}$ for states $S_1, \dots, S_n \in \mathcal{S}_t$.

For $i = 1, \dots, n$ construct Boolean terms φ_i such that $val_S(\varphi_i) = true$ holds iff S is T-similar to S_i. For this let $CT' = \{\gamma_1, \dots, \gamma_m\}$, and define terms

$$\bar{\gamma}_j = \begin{cases} \gamma_j & \text{if } \gamma_j \text{ is closed} \\ \langle (x_1, \dots, x_k) \mid \bigwedge\limits_{1 \le i \le \ell} \beta_i = \alpha_i \rangle & \text{if } \gamma_j = (x_1, \dots, x_k) \text{ with variables taken} \\ & \text{from } (\beta_1, \alpha_1), \dots, (\beta_\ell, \alpha_\ell) \end{cases}$$

exploiting the fact that the background structures provide constructors for multisets and pairs (and thus also tuples). Then

$$\varphi_i = \bigwedge_{\substack{1 \le j_1, j_2 \le m \\ E_{S_i}(\gamma_{j_1}, \gamma_{j_2})}} \bar{\gamma}_{j_1} = \bar{\gamma}_{j_2} \quad \wedge \bigwedge_{\substack{1 \le j_1, j_2 \le m \\ \neg E_{S_i}(\gamma_{j_1}, \gamma_{j_2})}} \bar{\gamma}_{j_1} \neq \bar{\gamma}_{j_2}$$

asserts that $E_S = E_{S_i}$ holds. Now define the rule r by

```
par if $\varphi_1$ then $r_{S_1}$ endif
    if $\varphi_2$ then $r_{S_2}$ endif
        ⋮
    if $\varphi_n$ then $r_{S_n}$ endif endpar
```

If $S \in \mathcal{S}_t$ is any state of t, then S is T-equivalent to exactly one S_i $(1 \le i \le n)$, which implies $val_S(\varphi_j) = true$ iff $j = i$, and hence $\Delta(r, S) = \Delta(r_{S_i}, S) = \Delta(t, S)$ by Lemma 8. □

Theorem 7. *For every database transformation t there exists an equivalent DB-ASM $\mathcal{M}$.*

Proof. By Lemma 9 there is a DB-ASM rule r with $\Delta(r, S) = \Delta(t, S)$ for all $S \in \mathcal{S}_t$. Define $\mathcal{M}$ with the same signature and background as t (and hence $\mathcal{S}_{\mathcal{M}} = \mathcal{S}_t$), $\mathcal{I}_{\mathcal{M}} = \mathcal{I}_t$, $\mathcal{F}_{\mathcal{M}} = \mathcal{F}_t$, and program $\pi_{\mathcal{M}} = r$. □

Note that for the proof of Theorem 7 we constructed a DB-ASM rule that does not use sequence rules, so by Theorem 6 this construction can be considered to be merely "syntactic sugar". As discussed before the let rules capture aggregate updates that exploit parallelism. Whether this can be extended to capture various aspects of parallelism, thus looking deeper inside database transformations, is an open problem.

6 Discussion and Conclusions

In this article we presented a variant of Gurevich's sequential ASM-thesis [10] dealing with database transformations in general. In analogy to Gurevich's seminal work we formulated five intuitive postulates for database transformations, and discussed why database transformations should satisfy these postulates. We then defined a variant of Abstract State Machines, which we called Database Abstract State Machines (DB-ASMs), and showed that DB-ASMs capture exactly all database transformations.

Despite many little technical differences of minor importance – such as final states, finite runs, and undefined successor state in case of an inconsistent update set – we stayed rather close to Gurevich's seminal work, but added as much as we felt is necessary to capture the essentials of database transformations as opposed to sequential algorithms. The important differences to Gurevich's sequential time postulate are the permission of non-determinism in a limited form with limitations enforced by the bounded non-determinism postulate, the exploitation of meta-finite states to capture the finiteness of databases, and an extended bounded exploration postulates, in which non-ground terms can appear in the bounded exploration witness.

Let us first summarise and discuss these important differences again. Regarding states we stayed with Gurevich's fundamental idea that states are first-order structures. We only adopted the notion of meta-finiteness [9]. As for the sequential ASM-thesis closure under isomorphisms is requested. So, apart from the incorporation of meta-finite states that are composed of a finite database part and a usually infinite algorithmic part with bridge functions linking them we more or less kept Gurevich's abstract state postulate. The idea of using meta-finite states was already expressed in our previous work in [22], but we were not yet able to handle the bridge functions in a satisfactory way. This gap is now closed.

Though for database transformations the projection of a run to the database part of states is most decisive, we did not build such a restriction into our model. This in turn implies that all sequential algorithms are also captured by DB-ASMs. Parallel algorithms, however, are not captured, as we only permit bounded parallelism as in the sequential ASM thesis. Combining our insights on database transformations with the parallel ASM thesis, i.e. permitting unrestricted parallelism on the algorithmic part of states, is a problem left for continued research.

Due to the permission of non-determinism that is restricted to choice among query results we capture more than sequential algorithms. Even without the non-determinism this is still the case because of the more general bounded exploration witnesses and the exploitation of location operators, which permit a limited form of parallelism by aggregating multisets of values.

In order to capture different data models, we originally thought of manipulating the notion of state, e.g. in our first attempt in [21] we tried to employ higher-order structures to capture tree-structured databases such as object-oriented and XML-based databases. In this article, we formulated that different data models should be captured by means of different background structures, which led us to formulate a background postulate. We are currently investigating this approach in more detail showing which particular constructors have to be present to capture data models such as the relational, nested-relational, complex values, object-oriented and XML models. The results achieved so far show that shifting specific data model requirements to background structures does indeed do the trick; we hope to be able to publish results shortly.

Nevertheless, we believe it is a promising idea for future research to consider "playing" with the notion of states, maybe even not only within the context of databases. For instance, for tree-based databases we may think of structures that can be recognised by certain tree-automata, or we could investigate automatic structures that are recognised by finite automata, etc. While these define restrictions to the general computational model, they may on one side provide interesting links to various logics, and on the other side define the challenge to integrate these automata into the formalism of DB-ASMs in order to capture exactly a particular class of database transformations.

The logical links would be of particular interest for queries, for which declarative approaches are preferred. In this paper we only touched the surface of queries in our extension of Gurevich's bounded exploration postulate. Instead of requesting the existence of a finite set of ground terms that determines update sets (or sets of update sets due to the assumed non-determinism) we widen this to a set of access terms, which in a sense capture associative access that is considered to be essential for databases. Nevertheless, there is still a big gap between access terms and similarly the access conditions in choice and forall rules in DB-ASMs on one side and highly declarative query languages. Bringing these different aspects together is a challenge for future research.

Finally, the notion of database transformation developed in this paper is limited to sequential database transformations over centralised databases, i.e. aspects of parallel and distributed databases have been neglected. With respect to parallelism we only considered aggregate update by means of location operators, i.e. we accumulate a multiset of updates on a location and then let them collapse to a single update. This form of parallelism is limited to the same computation on different data. Capturing parallelism and distribution as used in the architecture in [13] would require to investigate a more elaborate DB-ASM thesis picking up ideas from the parallel ASM thesis [5].

References

[1] Abiteboul, Serge and Kanellakis, Paris C. Object identity as a query language primitive. *Journal of the ACM*, 45(5):798–842, 1998.

[2] Abiteboul, Serge and Vianu, Victor. Datalog extensions for database queries and updates. *Journal of Computer and Systems Science*, 43(1):62–124, 1991.

[3] Beeri, Catriel, Milo, Tova, and Ta-Shma, Paula. On genericity and parametricity (extended abstract). In *PODS '96: Proceedings of the Fifteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 104–116, New York, NY, USA, 1996. ACM.

[4] Beeri, Catriel and Thalheim, Bernhard. Identification as a primitive of data models. In Polle, Torsten, Ripke, Torsten, and Schewe, Klaus-Dieter, editors, *Fundamentals of Information Systems*, pages 19–36. Kluwer Academic Publishers, Boston Dordrecht London, 1999.

[5] Blass, Andreas and Gurevich, Jury. Abstract state machines capture parallel algorithms. *ACM Transactions on Computational Logic*, 4(4):578–651, 2003.

[6] Börger, Egon and Stärk, Robert. *Abstract State Machines: A Method for High-Level System Design and Analysis.* Springer-Verlag, Berlin Heidelberg New York, 2003.

[7] Cohen, Sara. User-defined aggregate functions: bridging theory and practice. In *SIGMOD '06: Proceedings of the 2006 ACM SIGMOD International Conference on the Management of Data*, pages 49–60, New York, NY, USA, 2006. ACM Press.

[8] Ebbinghaus, H.-D. and Flum, J. *Finite Model Theory.* Springer-Verlag, 2 edition, 1999.

[9] Grädel, Erich and Gurevich, Yuri. Metafinite model theory. *Information and Computation*, 140(1), 1998.

[10] Gurevich, Jury. Sequential abstract state machines capture sequential algorithms. *ACM Transactions on Computational Logic*, 1(1):77–111, 2000.

[11] Gurevich, Yuri and Tillmann, Nikolai. Partial updates. *Theoretical Computer Science*, 336(2-3):311–342, 2005.

[12] Gurevich, Yuri and Yavorskaya, Tanya. On bounded exploration and bounded nondeterminism. Technical Report MSR-TR-2006-07, Microsoft Research, January 2006.

[13] Kirchberg, M., Schewe, K.-D., Tretiakov, A., and Wang, R. A multi-level architecture for distributed object bases. *Data and Knowledge Engineering*, 60(1):150–184, 2007.

[14] Ma, Hui, Schewe, Klaus-Dieter, Thalheim, Bernhard, and Wang, Qing. A theory of data-intensive software services. *Service Oriented Computing and Applications*, 3(4):263–283, 2009.

[15] Schewe, Klaus-Dieter and Thalheim, Bernhard. Fundamental concepts of object oriented databases. *Acta Cybernetica*, 11(4):49–84, 1993.

[16] Schewe, Klaus-Dieter and Wang, Qing. XML database transformations. Journal of Universal Computer Science (to appear).

[17] Van den Bussche, J. *Formal Aspects of Object Identity in Database Manipulation.* PhD thesis, University of Antwerp, 1993.

[18] Van den Bussche, Jan and Van Gucht, Dirk. Semi-determinism (extended abstract). In *PODS '92: Proceedings of the Eleventh ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 191–201, New York, NY, USA, 1992. ACM Press.

[19] Van den Bussche, Jan and Van Gucht, Dirk. Non-deterministic aspects of object-creating database transformations. In *Selected Papers from the Fourth International Workshop on Foundations of Models and Languages for Data and Objects*, pages 3–16, London, UK, 1993. Springer-Verlag.

[20] Van Den Bussche, Jan, Van Gucht, Dirk, Andries, Marc, and Gyssens, Marc. On the completeness of object-creating database transformation languages. *J. ACM*, 44(2):272–319, 1997.

[21] Wang, Qing and Schewe, Klaus-Dieter. Axiomatization of database transformations. In *Proceedings of the 14th International ASM Workshop (ASM 2007)*, University of Agder, Norway, 2007.

[22] Wang, Qing and Schewe, Klaus-Dieter. Towards a logic for abstract metafinite state machines. In Hartmann, Sven and Kern-Isberner, Gabriele, editors, *Foundations of Information and Knowledge Systems – 5th International Symposium (FoIKS 2008)*, volume 4932 of *Lecture Notes in Computer Science*, pages 365–380. Springer-Verlag, 2008.

Received 11th September 2009

CONTENTS

Hungarian Computational Linguistics **673**

Regular Papers **733**

ISSN 0324—721 X

Felelős szerkesztő és kiadó: Csirik János