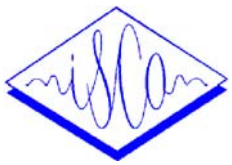

6th European Conference on
Speech Communication and Technology
(EUROSPEECH'99)
Budapest, Hungary, September 5-9, 1999

COMPILING MULTI−TIERED SPEECH DATABASES INTO THE RELATIONAL MODEL: EXPERIMENTS WITH THE EMU SYSTEM

Steve Cassidy

Speech Hearing and Language Research Centre
Macquarie University
Sydney, Australia

Steve.Cassidy@mq.edu.au

ABSTRACT

The Emu speech database system enables the annotation of speech signals at many levels of detail and provides a mechanism for making links between these levels to produce a hierarchical annotation. Emu provides facilities for searching collections of these annotations according to both sequential and hierarchical criteria. The results of a search can be used to retrieve acoustic and other data stored along with the annotations. One perceived problem with the Emu system is its ability to scale to large databases containing many thousands of utterances. To address this problem we propose a method of translating an Emu database into the relational model, as used by most commercial database systems. Using a Tcl script, the Emu database is converted into a set of tables for the relational database. Queries in the Emu query syntax are translated into SQL and comparisons are made between the query processing time for Emu and the relational database. The results show a marked increase in speed for the relational system on most queries.

1. INTRODUCTION

The increasing use of large collections of annotated speech materials in speech science and technology research raises many issues for data management and utilisation. The Emu speech database system [1] is a set of software tools that supports multi−leveled annotation of speech signals with links between segments at different levels. These annotations can be searched for examples matching constraints on segment labels and their sequential and hierarchical neighbourhood. The results of a search can be used to extract speech data corresponding to these segments for analysis.

The database query subsystem of Emu provides a rich query language which can be used to specify arbitrary constraints on the segments required. Searches are performed via a linear search of each utterance in the database for segments matching all of the specified criteria. This naïve approach has two consequences: there is a file input overhead as each label file is opened, read and searched, and, since there is no index maintained, that every segment at some level must be examined. Even with this naïve approach, most Emu searches on moderately sized (1000 utterances, 136000 segments) databases run in under a minute. It is clear though that this search method will not scale well to larger databases and so some thought must be given to alternative methods.

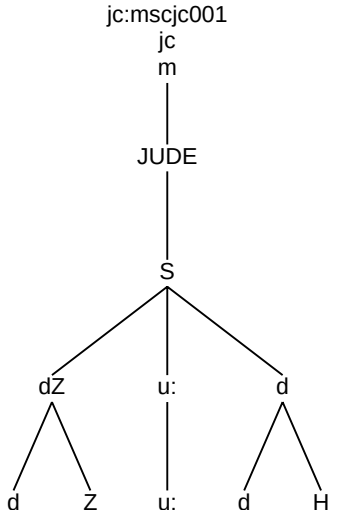

Figure 1. An Emu hierarchical with segments at Phonetic, Phoneme, Syllable, Word and Utterance levels and hierarchical links between segments.

The dominant general purpose database model is the relational model [2] which stores data as interrelated sets of tables. Relational database systems are capable of storing millions of records and handling large numbers of queries quickly and efficiently. Importantly, the relational model has well understood mathematical properties, if Emu annotation structures are readily mapped onto relational tables, then the formal results of relational database theory might also be applied to reasoning about Emu databases and the implementation of database search tools.

Relational database systems have been used to store speech database annotations [3] but the structures used are generally specific to one project or method of annotation. The Emu system provides a natural annotation structure which can be adapted to most speech annotation tasks. Since an Emu annotation is a set of segments and optional relations between these segments, it is possible to recast an Emu database as a set of tables in the relational model. The aim is to retain the flexibility of the Emu system while taking advantage of the efficiency and scalability of the

ID	Level	Start	End	Seq	Utterance	Label
jc:mscjc001/10	1	5500.7	6081.7	1	jc:mscjc001	jc:mscjc001
jc:mscjc001/6	2	5500.7	6081.7	1	jc:mscjc001	JUDE
jc:mscjc001/9	3	5500.7	6081.7	1	jc:mscjc001	S
jc:mscjc001/5	4	5500.7	5713.7	1	jc:mscjc001	dZ
jc:mscjc001/7	4	5713.7	5973.7	2	jc:mscjc001	u:
jc:mscjc001/8	4	5973.7	6081.7	3	jc:mscjc001	d
jc:mscjc001/0	5	5500.7	5640.7	1	jc:mscjc001	d
jc:mscjc001/1	5	5640.7	5713.7	2	jc:mscjc001	Z
jc:mscjc001/2	5	5713.7	5973.7	3	jc:mscjc001	u:
jc:mscjc001/3	5	5973.7	6011.7	4	jc:mscjc001	d
jc:mscjc001/4	5	6011.7	6081.7	5	jc:mscjc001	H

Figure 2. The **tokens** table corresponding to the utterance in Figure 1 as it would appear in the relational database. Each row corresponds to a single token. The **Level** column refers to a table of level names, the **Seq** column contains the sequence number of each token within its level.

relational database system. This paper investigates the practicality and advantages of this conversion.

2. THE STRUCTURE OF EMU ANNOTATIONS

An Emu database consists of a number of *utterances* corresponding to some convenient unit of analysis such as a word or a sentence. This follows the common practice in speech database design of segmenting a recording into individual files containing one word or sentence. Each utterance consists of one or more *levels* of annotation: each level contains zero or more *tokens* which may or may not be associated with time information. The tokens within each level form a sequence based either on the times of the tokens or on an order defined by the user. Each token may have one or more *labels*.

Hierarchical relations may exist between tokens in different levels. These imply that one token (the parent) dominates one or more tokens at another level and that the start and end times of the parent can be derived from those of its leftmost and rightmost children. Tokens at the child level may have more than one parent which means that tokens at the parent level will overlap.

Figure 1 shows an example hierarchy for an isolated word annotated at various levels of detail.

3. RELATIONAL STRUCTURE

A relational database consists of a set of tables with predefined column headings. Tables are related by shared columns. An Emu database can be expressed as a set of three tables recording the tokens, levels, and hierarchical relations between tokens (links). An example of the *tokens* table is shown in Figure 2 for the same utterance as shown in Figure 1. In addition to these tables indexes are created on the token label for the tokens table and on the parent and child tokens for the links table. These indexes facilitate fast access to the rows of these tables.

Tables were generated from an Emu database using a simple tcl script which iterates over the utterances in the database and outputs table entries for each token and each relation in the utterance. The script is general purpose and will work for any Emu database with arbitrary complex structure (the script and further notes on these experiments have been made available via the Emu web site [4]). In the experiments reported in this paper, these tables were read into the MySQL database system [5,6] running on a Linux system. MySQL is a freely available (for non−profit use) relational database system which supports most of the SQL query language. Any commercial relational database system could be used in it's place with minor modifications to the translation scripts.

4. DATABASE QUERIES

The Emu Query Language

The Emu system supports a special purpose query language designed to make the most common queries easy to express while allowing the experienced user to take full advantage of complex hierarchical structure.

The simplest query consists of a level name and a comparison; for example, to find phonetic segments labelled 'A' use the query:

```
Phonetic = A
```

alternative labels can also be specified:

```
Phonetic = A|E|I|O|U|V
```

More complex queries can refer to sequential and hierarchical relations within the annotation. A sequential query consists of two simple queries and the sequence operator:

```
[Phonetic=A –> Phonetic=p]
```

this query matches a sequence of 'A' followed by 'p' at the phonetic level. Hierarchical queries use the domination operator to relate tokens at different levels, the query:

```
[Phonetic=A ^ Syllable=S]
```

matches phonetic ‘A’ tokens which are dominated by a syllable labelled ‘S’. These queries can be nested arbitrarily to specify more complex constraints; the query:

```
[[Phonetic=A -> Phonetic=p] ^
  Syllable=S]
```

matches sequences of phonetic ‘A’ ollowed by ‘p’ both dominated by an ‘S’ syllable level tokens.

The Emu query syntax has a few other constructs but this subset is sufficient for the purposes of this paper.

The result of an Emu query is a table with one row per matching token and columns for the token label, start time, end time and utterance name. This table can be used to extract the corresponding speech data for each token.

Relational Database Queries

Queries are made to the MySQL system via the Standard Query Language (SQL) SELECT statement. A query specifies constraints on rows of the tables to be returned and the names of the columns to be included in the returned table. Columns may be taken from more than one table to construct the result. Using the table structure outlined above, the SQL query corresponding to the simplest Emu query in the first example above is:

```
SELECT DISTINCT
     tokens.label,
     tokens.stime, tokens.etime,
     tokens.utt
FROM
     tokens, levels
WHERE
     tokens.label='A' and
     tokens.level=levels.id and
     levels.name='Phonetic';
```

This query uses two tables (tokens and levels) and matches rows where the label column of the tokens table has the value ‘A’ and the level column has the value corresponding to the ’Phonetic’ level in the levels table. Alternatives for the label can be specified by substituting the following query fragment:

```
tokens.label in
('A','E','I','O','U','V')
```

Sequence queries can be constructed using the seq column in the tokens table which records the sequential position of each token in the level. The following SQL query mimics the first sequence query given above:

```
SELECT DISTINCT
     n1.label, n2.label,
     n1.stime, n2.etime,
     n1.utt
FROM
     tokens n1, tokens n2,levels
WHERE
     levels.name='Phonetic' and
     n1.label = 'A' and
     n2.level=levels.id and
     n1.level=n2.level and
     n2.label = 'p' and
     n1.seq = n2.seq - 1 and
     n2.utt=n1.utt;
```

Here, two tokens are individuated by the names n1 and n2 and they are constrained to be on the same level and have consecutive sequence numbers. The query returns the labels of both tokens and the start and end time of the sequence, this is consistent with the behaviour of the Emu query system.

Translating a domination query to SQL involves the use of the third table of links between tokens. To mimic the first domination example given earlier the query must describe two tokens at the appropriate levels and then specify that an entry in the links table must contain both tokens as parent and child. The query is as follows:

```
SELECT DISTINCT
     n1.label, n1.stime,
     n1.etime, n1.utt
FROM
     tokens n1, tokens n2,
     levels l1, levels l2, links
WHERE
     n1.label = 'A' and
     n1.level = l1.id and
     l1.name = 'Phonetic' and
     n2.label = 'S' and
     n2.level = l2.id and
     l2.name = 'Syllable' and
     links.parent = n2.id and
     links.child = n1.id;
```

It should be clear that more complex Emu queries involving both sequence and domination will translate into even more complex SQL. Fortunately the translation process is relatively easy to automate and in a real system the user could still enter queries in the more concise Emu syntax.

5. EXPERIMENTS

In order to evaluate the compilation to the relational model, a test database consisting of 1000 read sentences labelled at phonetic, phoneme, syllable, word and intermediate and intonational phrase levels. The database contains 136749 tokens and 438559 links. Both the Emu label files and the MySQL database files were stored on the same local hard disk. For comparison purposes the three queries described earlier and two more complex queries were submitted to Emu and MySQL. The additional queries were a query for short vowels (A,E,I,O,U or V) dominated by strong syllables and a query for a sequence of A followed by p both

dominated by a strong syllable. The elapsed time for three executions of the query were averaged. Tests were done on a Pentium 266 based Toshiba laptop computer with 96MB of memory running the Linux operating system.

	Emu	***MySQL***
Simple Query	34 sec	5 sec
Sequence Query	34 sec	22 sec
Domination Query	31 sec	24 sec
Label Disjunction	45 sec	45 sec
Complex Query	40 sec	22 sec

The results from the Emu system show a remarkable stability over the different classes of query. The query time is only extended when multiple labels are supplied or when the structure of the query is made more complex. The reason for this is that the query processing time is dominated by the time taken to open and read labels from the individual files for each utterance.

The query processing times from the MySQL system show a marked increase in performance over the Emu system in most cases. There is a very obvious advantage to using indexes on the segment label as illustrated by the very fast response to the simple query.

6. CONCLUSION

The experiments described in this paper were aimed at addressing the scalability of the Emu speech database system to very large speech databases. The issue of concern was that databases consisting of many thousands of small label files would be inefficiently searched using Emu's method of opening and examining every file in sequence. It was suggested that using the relational database model which has proven scalability would provide a solution to this perceived problem.

We have outlined one possible relational database structure to which an Emu hierarchical annotation could be mapped. Queries in the Emu query syntax can also be mapped into the relational SQL syntax. Some simple timing experiments with a 1000 utterance database showed that the relational database was faster than the Emu system on most queries used in testing.

A number of observations can be made about these results. It is clear that the relational model offers a sufficiently rich data structure to allow the whole of the Emu annotation structure to be represented with no loss of information. This may prove a valuable observation as it implies that Emu implements a special case of the relational database model and as such may be able to take advantage of the research and implementation experience of these database systems. Some elements of the Emu structure are missing from the current experiments namely: multiple labels for each token, the segment/event distinction and some aspects of the query language such as the positional predicates Start(), End() etc. None of these present major problems for an automatic translation into the relational model. Alternative representations might use one table per level (e.g. one for phonetic labels, another for syllables) or encode further structural information within each token (such as the dominating word or the number of siblings at some lower level) which might improve the performance on specific kinds of query.

The speed of the relational system is impressive with this medium sized database. We have every reason to believe that the performance will scale well, as relational databases are regularly used for applications containing millions of records in many tables.

It is encouraging to see the speed with which the current Emu system is able to process queries on this database relative to the speed of the relational system. The time taken for searches in Emu scales linearly with the number of utterances in the database since the search looks at every label file and uses no indexing. Clearly some simple improvements to the query processing algorithm could show dramatic benefits. For example, if all label files were concatenated into one large master label file, the number of separate open/close operations would be reduced. These require Emu to wait on system resources and can result in significant delays to query processing. Secondly, some form of indexing could be used to advantage to reduce the number of utterances which need to be examined; an index on segment labels for example could be used to restrict a search to only those utterances containing all labels mentioned in the query.

In conclusion, the Emu speech database system has been shown to provide relatively fast search of medium sized speech databases. A mapping to the relational database model has shown some promise in improving the scalability of the Emu system while retaining the flexibility of the Emu annotation structure.

REFERENCES

[1] Cassidy, S. and Harrington, J. (1996) *Emu: an Enhanced Hierarchical Speech Data Management System.* In Proceedings of the Australian Speech Science and Technology Conference, Adelaide.

[2] Laws, M. and Kilgour, M. *MOOSE: Management Of Otago Speech Environment* (1998) In Proceedings of the 5th International Conference on Spoken Language Processing, Sydney.

[3] Codd, E. F. (1990) *The relational model for database management* Addison Wesley.

[4] The Emu web site, http://www.shlrc.mq.edu.au

[5] MySQL web site: http://www.mysql.org

[6] Yarger R. J., Reese G. and King T. (1999) *MySQL and mSQL* O'Reilly & Associates.