Themes and Challenges in Temporal Databases

Christian S. Jensen

Department of Computer Science
Aalborg University
csj@cs.auc.dk

1 Introduction

Since the early eighties, an active temporal database research community has sought new insight into the management of time-referenced, or temporal, data and has developed concepts, tools and techniques that better support the management of such data. Much of this activity has been motivated by the observations that most databases contain substantial amounts of temporal data and that conventional database technology offers precious little support for temporal data management [5, 6].

The temporal database research community's activities are comparable to the assembly of a large jigsaw puzzle. In this research puzzle, the pieces collectively map out a variety of contributions that aim to improve the support for temporal database management. When assembling a regular jigsaw puzzle, all the pieces are expected to be available at the outset, and anybody who has tried to assemble a jigsaw puzzle knows the anguish that missing pieces can cause. A research puzzle is different. In this puzzle, it is the task of the members of the community to invent the pieces to be used in the puzzle, to shape existing pieces to make them fit with each other, and to discard pieces that turn out not to fit after all. This makes for a dynamic, ever-changing and expanding puzzle.

During the past decade, pieces have been provided for the temporal-database puzzle at quite a rapidly accelerating rate, and the puzzle has now been completed to a degree where it clearly and constructively demonstrates the ability to substantially enhance the support for temporal-data management as offered by conventional technology. In fact, initial studies show that many of the temporal query languages that are now available reduce the amount of database code needed for temporal data queries by as much as a factor of three in comparison to the standard SQL query language, and reduce the conceptual complexity by far more, thus promising a qualitative improvement in the efficiency of application development and maintenance.

2 Presentation Overview

In an attempt to offer both broad coverage and some technical depth, the presentation focuses on a single topic within temporal database research, but explores four themes in relation to this topic. The topic chosen is that of providing support for the concept of *now*, which is one of the most unique temporal concepts. The following themes of temporal database research are covered.

- **Temporal Database Semantics** (i) Semantics of *now*-relative data.
- **Query Languages** (ii) Query language support for *now*, including querying and modification.
- **Query Processing Techniques** (iii) Transaction support for *now* and (iv) indexing of *now*-relative data.

By restricting the coverage to *now*, it is hoped that the presentation will give the audience a flavor of these four general and quite diverse research themes as they related to temporal databases.

The author has conducted research on these themes for almost a decade, and the presentation will mainly offer a personal view of his contributions.

2.1 Themes

The coverage for each theme is described next.

Semantics of *now*-Relative Data

The ever-increasing current time, *now*, is a quite intriguing concept, which is omni-present temporal database management. There is frequently a need to record that facts are valid or current in the database until the current time. However, SQL only permits the storage of fixed, or ground, time values in the database. The presentation illustrates a framework for defining the semantics of the variable databases that result when allowing the use of the variable *now* as a timestamp value in databases [4]. The framework additionally covers so-called *now-relative* and *now-relative indeterminate* timestamp values.

Query Language Support for *now*

The semantic framework also provides a foundation for the querying of variable databases via existing query languages. This is achieved with a so-called *bind* operation that may be applied in a pre-processing step during query processing to eliminate all occurrences of *now*.

Conducting modifications involving variable *now* and of tuples timestamped with intervals containing *now* offers new challenges [8]. This part of the presentation briefly illustrates the semantics of such modifications. The main challenges stem from modifications that extend into the future and from tuples with end times in the future. The semantics of modifications on *now*-extended databases necessitate the introduction of two new timestamp values.

Transaction Support for *now*

The presentation proceeds to briefly consider the challenges that occur when performing timestamping in the context of user transactions, as opposed to single modification statements. The challenge is to maintain the zero-duration illusion for transactions that refer to *now* and timestamp the results of their modifications with *now*. Timestamping the results of a transaction with the commit time of the transaction is a promising approach, and a spectrum of techniques for how to do this are explored [7]. A central complication is that transactions may need their commit time before this time is actually known.

Indexing of *now*-Relative Data

Bitemporal data, where both the valid time and transaction time of the data are recorded [6], offer new challenges in the context of indexing. Like spatial data, bitemporal data has associated two-dimensional regions. Such data is in part naturally now-relative: some data is currently true in the mini-world or is part of the current database state. So, unlike for spatial data, the regions of now-relative bitemporal data grow continuously. Existing indices, including commercially available indices such as B^+- and R-trees, do not contend well with even small amounts of now-relative bitemporal data.

An extended R^*-tree is presented that efficiently accommodates the possibly growing two-dimensional regions associated with bitemporal data, by also letting the internal bounding regions grow [1, 3]. In this index, internal bounding regions may be triangular as well as rectangular. New heuristics for the algorithms that govern the index structure are provided. As a result, dead space and overlap, now also functions of time, are reduced. Performance studies indicate that this index is typically significantly faster than the existing R-tree based indices. Use of this index requires changes to the database kernel.

A second indexing technique is presented that eliminates the different kinds of growing data regions by means of transformations and then indexes the resulting stationary data regions with four regular R^*-trees [2]; and queries on the original data are correspondingly transformed to queries on the transformed data. Performance studies they indicate that the new technique yields a performance that is competitive with first index. And unlike this index, the new technique is applicable without access to the database kernel.

2.2 Contribution Overview and Challenges

The presentation proceeds with an overview of the author's contributions in temporal databases [5], covering *temporal semantics and database design*, *data models and query languages*, and *query processing techniques*.

The presentation is concluded with a description of what the author perceives as important challenges to temporal database research in the future.

Acknowledgments

The author would like to thank the organizer of the Zobis working group, Gerhard Knolmayer, and Thomas Myrach for their help in bringing this presentation to life.

The author would also like to thank the twenty colleagues with whom he worked on the themes covered in this presentation.

References

[1] Bliujūtė, R., C. S. Jensen, S.Šaltenis, and G. Slivinskas. R-tree-based Indexing of Now-Relative Bitemporal Data. *Proceedings of the 24th International Conference on Very Large Databases*, New York City, NY, August 1998, pp. 345–356.

[2] Bliujūtė, R., C. S. Jensen, S.Šaltenis, and G. Slivinskas. Light-Weight Indexing of General Bitemporal Data. *Proceedings of the Twelfth International Conference on Scientific and Statistical Database Management*, Berlin, Germany, July 2000, pp. 125–138.

[3] Bliujūtė, R., S. Šaltenis, G. Slivinskas, and C. S. Jensen. Developing a DataBlade for a New Index. *Proceedings of the Fifteenth IEEE International Conference on Data Engineering*, Sydney, Australia, March 1999, pp. 314–323.

[4] Clifford, J., C. Dyreson, T. Isakowitz, Jensen, C. S. and R. T. Snodgrass. On the Semantics of "Now" in Databases. *ACM Transactions on Database Systems*, 22(2): 171–214, June 1997.

[5] Jensen, C. S. *Temporal Database Management*. Faculty of Engineering and Science, Aalborg University, August 1999, 1328+xxxviii pages. Available via www.cs.auc.dk/~csj.

[6] Jensen, C. S. and R. T. Snodgrass. Temporal Data Management. *IEEE Transactions on Knowledge and Data Engineering*, 11(1): 36–44, January/February 1999.

[7] Torp, K., R. T. Snodgrass, and C. S. Jensen. Effective Timestamping in Databases. *The VLDB Journal*, 8(4): 267–288, February 2000.

[8] Torp, K., C. S. Jensen, and R. T. Snodgrass. Modification Semantics in Now-Relative Databases. Manuscript submitted for publication. Available via www.cs.auc.dk/TimeCenter/TimeCenterPublications/TR-43.ps.gz.