MEDINFO 2004
M. Fieschi et al. (Eds)
Amsterdam: IOS Press

A Knowledge-Based Time-Oriented Active Database Approach for Intelligent Abstraction, Querying and Continuous Monitoring of Clinical Data

Alex Spokoiny and Yuval Shahar

Medical Informatics Research Center, Dept. of Information Systems Engineering, Ben Gurion University, Beer Sheva, Israel

Abstract

Query and interpretation of time-oriented medical data involves two subtasks: Temporal-reasoning--intelligent analysis of time-oriented data, and temporal-maintenance--effective storage, query, and retrieval of these data. Integration of these tasks into one system, known as temporal-mediator, has been proven to be beneficial to biomedical applications such as monitoring, therapy, quality assessment, visualization and exploration of time-oriented data. One potential problem in existing temporal-mediation approaches is lack of sufficient responsiveness when querying or continuously monitoring the database for complex abstract concepts that are derived from the raw data, especially regarding a large patient group. We propose a new approach: the ***knowledge-based time-oriented active database****, a temporal extension of the active-database concept, and a merger of temporal reasoning and temporal maintenance within a persistent database framework. The approach preserves the efficiency of databases in handling data storage and retrieval, while enabling specification and performance of complex temporal reasoning using an incremental-computation approach. We implemented our approach within the Momentum system. Initial experiments are encouraging; an evaluation is underway.*

Keywords:

Temporal reasoning, temporal abstraction, temporal maintenance, temporal mediation, active temporal-abstraction mediation, knowledge-based systems, temporal databases, active databases, time-oriented active databases.

Introduction: Temporal Reasoning and Temporal Maintenance in Medicine

Representation, continuous monitoring, querying, and analysis of time-oriented clinical data are crucial for both care providers and automated decision-support systems. In both cases, certain information regarding one or more patient records is needed to support diagnosis, therapy, monitoring, quality assessment, or clinical research. Thus, during treatment by an oncology protocol, a clinical guideline processing system may apply a following eligibility criteria, "locate all patients who have had, within the past 9 months, more than two episodes of *grade II or higher bone marrow toxicity* (defined by protocol), each lasting at least 2 weeks", which in fact is a temporal query for a complex abstract concept. **(Figure 1.)**

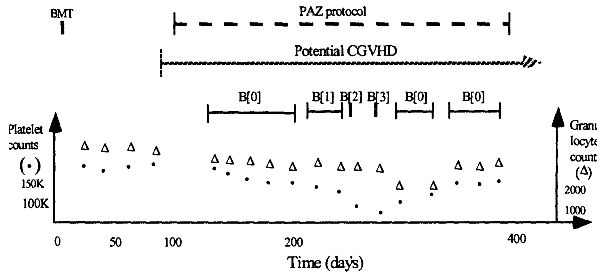

Figure 1 - Abstraction of time-oriented data in the bone-marrow transplantation domain. Raw data are plotted over time at the bottom. External events and the abstract concepts computed from the data are plotted as intervals above the data. = an external event (medical intervention); • = platelet counts; D = granulocyte counts; ⊢→ = a context interval; ⊢⊣ = a derived abstract concept interval; BMT = bone-marrow transplantation (an external event); PAZ = a protocol for treating chronic graft-versus-host disease (CGVHD), a complication of BMT; B[n] = bone-marrow–toxicity grade n, an abstract concept computed (through an intermediate abstraction layer) from platelet and granulocyte counts.

Handling the time dimension typically requires performance of two distinct tasks: **temporal reasoning (TR)** and **temporal data maintenance (TM)**. TR supports inference regarding time-oriented data, such as when monitoring patients, diagnosing disorders, planning and applying therapy. TM deals with storage and retrieval of data that have heterogeneous temporal dimensions. Decision-support applications that involve time-oriented medical data require performance of both tasks. Thus, it is highly desirable to have a single module that performs both tasks, with a well-defined interface.

Effective querying and continuous monitoring of abstract, temporally extended concepts, in particular within large sets of patients, is a major potential problem of current TR and TM integration efforts, often referred to as **temporal mediation**. To overcome these deficiencies, we propose a new approach, an **knowledge-based time-oriented active database**. This approach is a temporal extension of the *active database* concept (see below), and might be considered as merging the TR and TM tasks within a persistent database framework. The approach preserves the efficiency of databases in handling data storage and retrieval, while supporting specification and performance of complex temporal reasoning (in particular, temporal abstraction) using an incremental computation approach. The approach we suggest provides persistence and truth-maintenance of the resultant temporal abstractions. We demonstrate the new approach in the **Momentum** system[1], which we have implemented as part of an architecture for *active temporal-abstraction mediation*.

Background: Temporal Abstraction, Temporal Mediation, and Active Databases

To clarify the ideas leading to the Momentum system, it is best to start by a brief presentation of the *temporal-abstraction*, *temporal-mediation*, and *active database* concepts.

Temporal Abstraction

A crucial part of temporal reasoning is creating high-level temporally extended concepts from raw time-stamped data. This task is often called **temporal abstraction (TA)**.

The **knowledge-based temporal-abstraction (KBTA)** problem-solving method [7] was originally proposed specifically to solve the TA task. The KBTA method was designed with clear semantics for both the problem-solving method and its domain-specific knowledge requirements. The KBTA input includes a set of time-stamped parameters (e.g., blood-glucose values), external events (e.g., insulin injections). All input types can induce temporally extended contexts that can change the interpretation of one or more parameters [8]. The KBTA output includes a set of interval-based, context-specific abstract concepts of several types: state (e.g., High), gradient (e.g., Decreasing), rate (e.g., Slow), and pattern (e.g., Multi-organ-toxicity). **Figure. 1** shows an example of input for the TA task, and the resulting output, in the case of a patient who is being treated by a clinical protocol for treatment of chronic graft-versus-host disease (GVHD), a complication of bone-marrow transplantation.

The KBTA method was implemented within the **RÉSUMÉ** system [6]. A part of the KBTA method was implemented in the **RASTA** system [4].

Neither the RÉSUMÉ system nor the RASTA system allowed external querying or dynamic exploration of the high-level time-oriented abstract concepts generated by the TA process, as an inherent part of the system. Thus, *temporal-database mediators* were introduced to provide a more complete solution.

Temporal Mediation

A general approach to the TR and TM integration, called a *temporal-database mediator*, was suggested in an early architecture, the **Tzolkin** system [3] and in the more recent work describing the **Chronus-II** temporal database mediator [5]. This approach encapsulates the TR and the TM capabilities in a reusable software component that can answer raw or abstract, time-oriented queries. Such a system is called a *mediator* because it serves as an intermediate layer of processing between client applications and databases [10]. As a result, the mediator is tied to neither a particular application, nor to a particular database [11]. The Tzolkin system (which, technically, is a *temporal-abstraction mediator*, since it uses knowledge to answer queries about abstract concepts) consisted of the RÉSUMÉ TR module and the **Chronus** TM module; one was executed within working memory, and the other embedded within a database. RÉSUMÉ generated all abstractions mentioned in the query and wrote them into a temporary database; then, the Chronus module applied the temporal constraints of the query to the temporary database (which now included also the desired abstractions), to generate the complete answer. A similar relationship exists, respectively, between the RASTA and Chronus-II systems in the Chronus-II temporal database mediator (Technically, Chronus-II is a *temporal mediator*, since it does not use knowledge).

Execution of complex temporal queries in these mediator systems raises the problems of the *defeasibility* (*nonmonotonicity*) of the computed temporal abstract concepts, and maintenance of the validity of the conclusions over time. The RÉSUMÉ system used a **truth-maintenance system (TMS)** that allows specific temporal abstractions to be withdrawn from working memory when new, contradictory data arrive (e.g., the result of a laboratory test that was taken 2 weeks ago). However, the TMS did not extend to the external database. To avoid this problem, Tzolkin computes (de-novo) all abstractions based on the content of the database at the moment that the query is evaluated, a strategy that can obviously lead to significant response-time problems in the case of queries referring to complex temporal patterns, large longitudinal patient records, or a large set of patients. The Chronus-II mediator faces the same kind of problem.

In the Momentum system, we provide a solution to all the presented problems, while preserving the strengths inherent in the TA mediation approach.

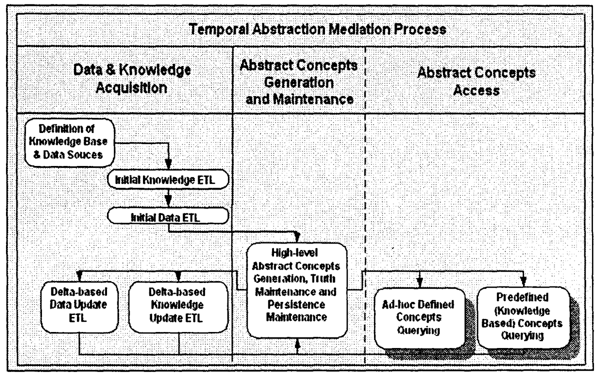

Figure 2 - The temporal abstraction mediation process.

Active Databases

Active databases (ADBs) extend "passive" databases by enabling the specification of reactive behavior as part of the database [1]. **Event-condition-action rules (ECA-rules)** in ADBs consist of events, conditions and resulting actions. The meaning of an ECA rule is: "when an event occurs, check the condition and if it holds, execute the action". Once a set of rules has been

1. Momentum: A force that increases the rate of development of a process

defined, the active database system monitors the relevant events. An ADB provides a **rule definition language (RDL)** as a means of specifying ECA-rules. Thus, an ADB supports rule specification, event detection, and rule execution. We based the Momentum system on a knowledge-based *time-oriented active database* approach, which is a temporal extension of the *active database* concept.

The Temporal-Abstraction Mediation Process

Before describing the Momentum system in detail, we present the context in which the system operates.

We define a *temporal-abstraction mediation (TAM)* process as a sequence of high-level tasks required to eventually generate temporal abstractions and to answer temporal queries. The process involves all phases, from data and knowledge input, through processing input raw data and delivering it in a form of knowledge-based abstract concepts to the TA mediation client applications (e.g., patient monitoring, application of clinical guidelines, assessing of the quality of guideline application, visualization and exploration of clinical data, etc.). The TAM process **(Figure 2)** includes three phases: temporal knowledge and data acquisition, abstract-concept generation (including truth and persistence maintenance), and the abstract-concept access. During the first phase, an extract, transform, and load **(ETL)** process is performed on the knowledge base and operational data sources: the sources are defined, the knowledge and data are extracted, transformed (for example, in accordance with a controlled medical dictionary) and loaded for the subsequent temporal abstract concepts generation. We distinguish initial knowledge/data ETL from incremental knowledge/data updates (delta updates). During the second phase, the new knowledge-defined high-level abstract concepts are generated, the truth is maintained for the previously generated abstract concepts for each ETL (initial or delta-based), and the persistence is maintained. This phase also allows continuous monitoring for abstract concepts. During the third phase, the generated abstract concepts can be accessed. Abstract concepts can be predefined in the underling knowledge base, or defined ad-hoc during the abstract-concepts access phase. The proposed process deliberately separates the data and knowledge acquisition and abstract concept generation from the abstract concept access. Because all the available abstract concepts are generated before they can be accessed, the queries do not need to generate abstract concepts on the fly, and thus they can be quickly answered even for complex queries and large sets of patients. To allow some level of flexibility for those client applications which cannot or do not desire to define in advance all abstract concepts in the knowledge base, the process allows ad-hoc abstract concepts definition during the data access phase. In this case, the ad-hoc defined abstract concepts are generated during the query processing.

To support the TAM process, we have developed the *temporal-abstraction mediation architecture.*

The Temporal-Abstraction Mediation Architecture

The *temporal-abstraction mediation architecture* **(Figure 3)** has two major aspects that are sufficiently different to warrant independent consideration. We labeled these two parts as the *back room* and the *front room*[1]. The *back room* supports the first two phases of the TAM – from the data and knowledge acquisition to the high-level abstract-concepts generation, truth and persistence maintenance, while the *front room* supports the third phase–the abstract-concepts access.

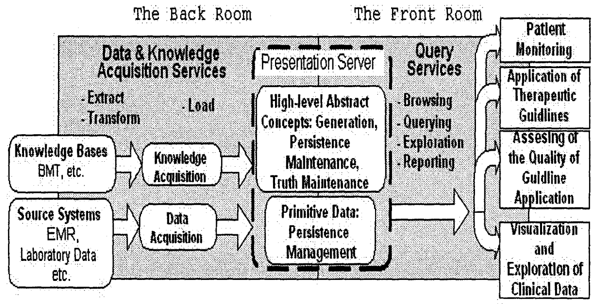

Figure 3 - The temporal abstraction mediator architecture.

In the back room data and knowledge moves from the source systems and the knowledge bases to the acquisition areas using the applications provided as part of the data and knowledge acquisition services layer. This flow is driven by the definitions of the sources and targets and the transformation details (for example medical dictionaries mapping). The front room delivers the generated abstract concepts to the user community: clinicians and computer based decision support systems, by using the query-services layer. Both rooms share the **presentation server**, which loads the primitive data and knowledge in the case of the back room, generates abstract concepts, provides truth, persistence maintenance, and continuous monitoring for the abstract concepts, and, in the case of the front room, presents the generated abstract concepts to the query services in a fashion that facilitates answering the user-issued temporal queries. The presentation server is the "glue" which brings the two rooms together, hiding the complexities of the high-level abstract-concepts generation and enabling the decision support data to flow virtually seamlessly to the users community in a form of the high-level abstract concepts generated from the data sources, according to the knowledge defined in the knowledge bases. The presentation server in fact integrates the TR tasks (abstract concept generation, truth and persistence maintenance) with the TM tasks (data and knowledge load, and the query services). The Momentum system implements the presentation server of the architecture presented in **Figure 3**.

The Momentum System

Systems integrating both the TR and the TM tasks have requirements that span from efficient data-manipulation, typical of databases, to inference capabilities, typical of TA systems. Active databases may be considered as a bridge between these two kinds of systems: they have both the efficiency of databases in

1. The back and front room notations originated in the data warehouse engineering community [2].

handling data storage and retrieval and the power of expressing complex inference mechanisms.

The Momentum system extends the approach of active databases to knowledge-based time-oriented active databases, by treating the temporal dimension in a unique fashion. In analogy to ECA-rules and rule execution of active databases, we provide Momentum with the domain knowledge required by the KBTA method and with the TA mechanisms. We also provide a **knowledge definition language (KDL)**, which, in analogy to RDL, allows us to specify domain knowledge according to the KBTA ontology, and a **temporal abstraction query and instruction language (TAQILA)**. TAQILA supports the tasks of raw-data specification and querying, abstract concepts querying, data-monitoring alerts specification, explanation and dynamic sensitivity analysis of the derived concepts.

The Momentum Architecture

The Momentum architecture is modular, and is independent of any domain or application. The Momentum system includes, for each knowledge-base/data-sources configuration, an instance of a *dynamic-processing environment*–the run-time environment, and an instance of a *dynamic repository*–the storage. The dynamic-processing environment is where the abstract-concept generation and querying take place. The dynamic repository provides persistence to the loaded knowledge, the loaded raw data, and the generated abstract concepts, which are stored along with their generation details (i.e., logical justifications such as *abstracted-from* and *abstracted-into* links). The dynamic-processing environment uses the repository to manage its persistence. Each knowledge base and data source, or a group of conformed data sources (sources containing different data aspects of the same patients), define a unique dynamic repository, which allows Momentum clients to work with different data and knowledge sources according to the temporal-abstraction mediation architecture.

The Momentum system implements the presentation server's back room functionality by providing KDL rules for knowledge definition (KBTA ontology specification) and TAQILA raw-data specification instructions for the data loaded from data sources. The front room functionality is implemented by providing TAQILA querying.

The Temporal Abstraction Query and Instruction Language (TAQILA)

TAQILA is simple yet expressive general-purpose query language. TAQILA provides: raw data instructions–for inserting (ADD), updating (UPDATE), and removing (REMOVE) primitive data elements; data-monitoring alerts specification (CREATE ALERT ON); querying (FIND) – for population querying of raw and derived concepts (ad-hoc or knowledge defined); exploration querying–for justification of the abstraction process (EXPLAIN), which provide the data instances from which an abstract concept instance was derived and dynamic sensitivity analysis (WHAT-IF), which simulates the effect of modifying the data or the knowledge, using the TMS. Retrieval and exploration querying return sets of temporal intervals satisfying the query constraints. The TAQILA queries and instructions form an application interface (API) that is typically manipulated by various client applications. TAQILA examples can be seen in Table 1.

Processing of a set of TAQILA instructions and queries

TAQILA	Momentum Processing
ADD (...), UPDATE (...), REMOVE (...)	Momentum adds primitive instances to the repository, generates new abstract concepts, and propagates updates via the TMS.
FIND SECOND **concept** {AS ...} WHEN start-time < 'Jan 12,2002' AND stop-time BEFORE NOW WHERE CASE.id = 555 RETURN value; EXPLAIN tuple_id	Momentum looks for a second instance of a primitive or abstract **concept** for patient 555 constrained by start and stop times and returns the found value. If **concept** is temporal pattern, it can also be defined on the fly using the AS clause The returned result set can be farther explored by the EXPLAIN instruction.
CREATE ALERT alert_name ON <FIND> SET ALERT ON/OFF	Data monitoring alerts can be defined by ALERT definition on the regular FIND query. ALERT will be fired on each generated instance added or updated in the <FIND> result set.
SET WHAT-IF ON ADD (...), UPDATE (...), REMOVE (...), FIND (..) ... SET WHATIF OFF	The WHAT-IF instruction allows dynamic sensitivity analysis of hypothetical modifications of either data or knowledge. SET WHAT-IF ON starts the what-if session. TAQILA instructions can then be performed. After the SET WHAT-IF OFF instruction, the repository returns to the initial pre-session state.

The Momentum Dynamic-Processing Data Structures and Algorithms

Due to lack of space, we will not present here the internal workings of Momentum. However, several points are worth mentioning. Momentum uses several specialized time-oriented structures to represent the data, both for the runtime environment and the repository, and specialized knowledge structures to represent KDL rules – the knowledge hub. Those data structures are designed to optimize concepts generation, using computational mechanisms similar to those of the KBTA method, but implemented quite differently. For example, interpolation tree structure, which provides interpolation between similar concepts, using a context-sensitive interpolation table [9] is highly efficient; its worst case complexity is $O(N^2)$, but typically (apart from a one-time sorting operation) it is a linear process. The data structures facilitate **incremental modification** of the repository through a built-in TMS. An inference network maintains links among data, knowledge, and derived concept instances.

Implementation Details

We implemented the Momentum system in Java Programming Language. KDL and TAQILA as well as results sets are described as XMLs. For the data repository we use a native XML database which is compliant to the XML:DB specification and allows us to store unstructured data.

Discussion and Future Work

In this prototyping phase of the system, although we are using real clinical data and knowledge (mostly from chronic-disease domains, such as oncology), no complete clinical validation has been carried out yet. Nevertheless, the results of the described work are significant mainly because they prove that a knowledge-based time-oriented active database approach may be effectively used for temporal-abstraction mediation. To tackle the responsiveness issue Momentum provides a new focus on the persistence and reuse of previously derived abstract concepts. Momentum dynamically generates and manages the persistence of all relevant abstractions for a given set of data previous to their use, and then enables applications to perform continuous monitoring, querying, and exploration, which are very effective because all the possible abstractions have already been generated and stored (somewhat similar to Tzolkin's *batch computation* mode, but performed dynamically and incrementally). TAQILA syntax also allows ad-hoc temporal patterns specification during the concept access phase (like Tzolkin TSQL and Chronus-II TSQL2). Unlike the early mediators, Momentum generates new abstract concepts only when relevant primitive (raw) data is added to the system, without re-computing previously generated abstractions (i.e., an incremental computational approach) while maintaining truth in the already generated abstract concepts, using dynamically formed logical links. To solve the nonmonotonicity and validity maintenance problems encountered by the early temporal mediators, the Momentum abstraction and querying modules share the same run-time data structure and truth maintenance systems.

The clean APIs allow Momentum to be used as part of a larger architecture supporting TAM process.

One of the potentially highly interesting experiments would be substituting Momentum with its persistent storage system instead of the pure transient-memory temporal-abstraction component of the IDAN temporal-mediation architecture [12].

In the short term, we plan to perform clinical validation of the Momentum system on one or more large clinical databases. We also plan to further improve the performance by making the abstraction algorithms parallel and distributed.

Acknowledgments

This research was supported in part by NIH award No. LM-06806. We thank Martin O'Connor for useful discussions regarding the Chronus-II and RASTA.

References

[1] ACT-NET (1996). The Active Database Management System Manifesto: A Rulebase of ADBMS features. *ACM SIGMOD Record*, 25(3):20--49, September.

[2] Kimball R, and Ross M. (2002). *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling* (2nd Edition). John Wiley & Sons.

[3] Nguyen JH, Shahar Y, Tu SW, Das A, and Musen MA. (1999). Integration of Temporal Reasoning and Temporal-Data Maintenance into a Reusable Database Mediator to Answer Abstract, Time-Oriented Queries: The Tzolkin System. *Journal of Intelligent Information Systems* **13** (1/2), 121-145

[4] O'Connor M.J., Grosso W.E., Tu S.W., and Musen M.A.. (2001). RASTA: A Distributed Temporal Abstraction System to Facilitate Knowledge-Driven Monitoring of Clinical Databases. *Proceedings of MEDINFO-2001, the Tenth World Congress on Medical Informatics*, pp. 508-512, London, UK

[5] O'Connor M. J., Tu S. W., Musen M. A.. (2002). The Chronus II Temporal Database Mediator. *Proceedings of the American Medical Informatics Association (AMIA) 2002 Annual Fall Symposium*, San Antonio, TX.

[6] Shahar Y, Musen MA. (1996) Knowledge-based temporal abstraction in clinical domains. *Artificial Intelligence in Medicine* **8**, 267-298

[7] Shahar,Y. (1997). A framework for knowledge-based temporal abstraction. *Artificial Intelligence* 90 (1–2), 79–133

[8] Shahar, Y. (1998). Dynamic temporal interpretation contexts for temporal abstraction. *Annals of Mathematics and Artificial Intelligence* **22** (1-2) 159-192

[9] Shahar, Y. (1999). Knowledge-based temporal interpolation. *Journal of Experimental and Theoretical Artificial Intelligence* **11**, 123-144

[10]Wiederhold, G. (1992). Mediators in the architecture of future information systems. *IEEE Computer*, 25:38–50

[11]Wiederhold, G. and Genesereth, M. (1997).The Conceptual Basis of Mediation Services. *IEEE Expert*,12(5),38–47

[12]Boaz, D., and Shahar, Y. (2003). Idan: A distributed temporal-abstraction mediator for medical databases. Proceedings of the 9th Conference on Artificial Intelligence in Medicine—Europe (AIME) '03, Protaras, Cyprus.

Address for Correspondence

Alex Spokoiny
Medical Informatics Research Center,
Department of Information Systems Engineering,
Ben-Gurion University, Beer Sheva 84105, Israel
spokoiny@bgumail.bgu.ac.il