Report Number: WUCS-89-48

1989-11-02

The Display and Manipulation of Temporal Information

Steve B. Cousins, Michael G. Kahn Washington University in St. Louis, and Mark E. Frisse Washington University in St. Louis

Because medical data have complex temporal features, special techniques are required for storing, retrieving, and displaying clinical data from electronic databases. One significant problem caused by the temporal nature of medical data has been called the temporal granularity problem. The temporal granularity problem is said to occur when the set of facts relevant to a specific problem changes as the time scale changes. We argue that what is needed to deal with changes in the relevant time scale are temporal granularity heuristics. One heuristic that we have explored is that, for any level of problem abstraction, and for each... Read complete abstract on page 2.

The Display and Manipulation of Temporal Information

Steve B. Cousins, Michael G. Kahn Washington University in St. Louis, and Mark E. Frisse Washington University in St. Louis

Complete Abstract:

Because medical data have complex temporal features, special techniques are required for storing, retrieving, and displaying clinical data from electronic databases. One significant problem caused by the temporal nature of medical data has been called the temporal granularity problem. The temporal granularity problem is said to occur when the set of facts relevant to a specific problem changes as the time scale changes. We argue that what is needed to deal with changes in the relevant time scale are temporal granularity heuristics. One heuristic that we have explored is that, for any level of problem abstraction, and for each type of data item in the record, there exists an optimal level of temporal abstraction. We describe an implemented database kernel and a graphical user interface that have features designed specially to support this temporal granularity heuristic. The basis for our solution is the use of temporal abstraction and temporal granularity. This heuristic encodes the relevant behavior of each type of event at different levels of temporal granularity. In doing so, we can define a specific behavior for each type of data as the level of abstraction changes.

THE DISPLAY AND MANIPULATION OF TEMPORAL INFORMATION

Steve B. Cousins, Michael G. Kahn
and Mark E. Frisse

WUCS-89-48

November 1989

Department of Computer Science
Washington University
Campus Box 1045
One Brookings Drive
Saint Louis, MO 63130-4899

Presented at the Thirteenth Annual Symposium on Computer Applications in Medical Care, November, 1989, Washington D.C.

THE DISPLAY AND MANIPULATION OF TEMPORAL INFORMATION

Steve B. Cousins, Michael G. Kahn
and Mark E. Frisse

WUCS-89-48

November 1989

Department of Computer Science
Washington University
St. Louis, Missouri 63130

Presented at the Thirteenth Annual Symposium on Computer Applications in Medical Care, November, 1989, Washington, D.C.

The Display and Manipulation of Temporal Information

Steve B. Cousins

Michael G. Kahn

Mark E. Frisse

Medical Informatics Group

Departments of Internal Medicine and Computer Science

Washington University, St. Louis, Missouri 63130

November 2, 1989

Abstract

Because medical data have complex temporal features, special techniques are required for storing, retrieving, and displaying clinical data from electronic databases. One significant problem caused by the temporal nature of medical data has been called the *temporal granularity problem*. The temporal granularity problem is said to occur when the set of facts relevant to a specific problem changes as the time scale changes. We argue that what is needed to deal with changes in the relevant time scale are *temporal granularity heuristics*. One heuristic that we have explored is that, for any level of problem abstraction, and for each type of data item in the record, there exists an optimal level of temporal abstraction.

We describe an implemented database kernel and a graphical user interface that have features designed specifically to support this temporal granularity heuristic. The basis for our solution is the use of temporal abstraction and temporal decomposition to support changes in temporal granularity. This heuristic encodes the relevant behavior of each type of event at different levels of temporal granularity. In doing so, we can define a specific behavior for each type of data as the level of abstraction changes.

Introduction

Most medical databases store data using the time-oriented data (TOD) model [10]. In TOD databases, information is stored as $< attribute, time, value >$ tuples. In these databases, the time point is the atomic unit of concern. Laboratory data, physical findings, and therapeutic interventions are all represented as temporal points. The simplicity and flexibility of the TOD model make it an extremely powerful representation method for medical data.

Unfortunately, not all medical information is easily amenable to point-based representation. Effective diagnosis and therapy planning requires an understanding of temporal trends and the clinical contexts in which these patterns occur. These issues are understood best through a view of data as intervals rather than as discrete points. From this perspective, it is critical to know if data were measured during a period of illness or while an administered drug was present in therapeutic levels. We believe that a medical database is more useful if it can represent data as both points and multiple, overlapping intervals. The latter concepts cannot be represented clearly using the TOD model.

The adoption of a more sophisticated temporal data model presents new theoretical and practical problems. The issue of temporal granularity is of interest to our research. Temporal granularity is the unit of a time scale appropriate for a given problem-solving context. The problem with temporal granularity is that the relevant facts change whenever a shift in temporal granularity occurs. For example, recording and retrieving information in units of minutes and hours is appropriate in an ICU setting, while weeks or months usually are more appropriate temporal units for the analysis of chronic disease data. Even in an acute setting like the ICU, previously recorded information is manipulated at a different level of temporal granularity than are current data. After discharge from the hospital, the entire ICU stay may be combined into a single abstract fact.

Temporal abstraction is an effective mechanism for combining smaller temporal entities into larger, but less detailed, concepts. Abstraction simplifies retrieving and reasoning by combining multiple features into a single entity. Temporal decomposition is the inverse operation of temporal abstraction. In temporal decomposition, the entities contained within a larger temporal abstraction are used for more refined reasoning. Details suppressed by the abstraction are made available by temporal decomposition. Conversely, the number of entities to manipulate is increased by decomposing an abstraction.

We believe that temporal abstraction and decomposition are the expression of a powerful heuristic used by humans to focus only on features that are relevant to solve a specific problem [5]. Additional computer methods must be developed that abstract and decompose temporal entities and display temporal features in a format which exploits the temporal pattern-matching superiority of humans. We present a computer program that can effectively store, retrieve, and display time-oriented medical data at a level of temporal granularity appropriate to the user's needs.

A Clinical Scenario

A patient with long-standing systemic lupus erythematosus (SLE) is seen by her physician because of vague constitutional symptoms. Over the past four years, she has had frequent flares and has received numerous therapies. He notes that she has been seen eight times in the past six months. In reviewing her previous visit, her physician notes specific laboratory results. By examining other visits, he sees that these results have not changed significantly over the past year. When he focuses on a series of flares over the past two years, he notices that most flares were preceded by a clinic visit in which the patient had non-focal complaints and the clinical exam was unremarkable. Based on this perspective, the physician realizes that, although the complaints preceeding a flare are non-specific, they represent the start of a significant clinical event for this patient. With this new insight, the physician can modify his treatment plan when the patient has these symptoms in an attempt to pre-empt the development of a signficiant clinical flare.

In solving this clinical problem, the physician moved between three levels of temporal granularity. He began by examining details of the recent patient history (eg, medical events over the past six months). At this point, he did not consider individual lab values because his use of abstraction as

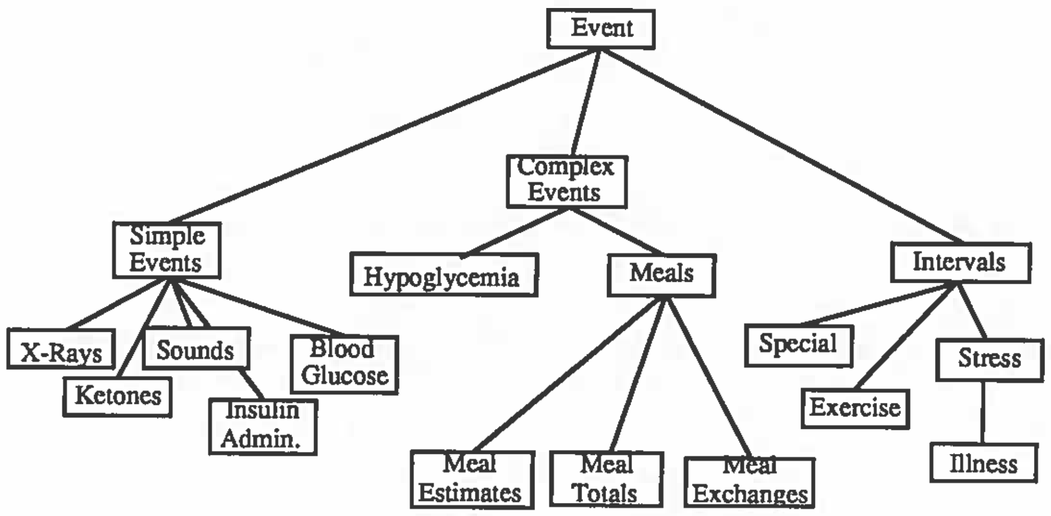

Figure 1: A taxonomy of temporal classes in diabetes data.

a temporal granularity heuristic supressed items at that level of detail. Seeing the large number of visits in a short period of time, he began a more detailed search for findings that may explain or aid in diminishing these events. Here the physician is using temporal decomposition to focus on specific details. When this examination was unrevealing, he examined the patient record at a higher level of abstraction which revealed a useful pattern not visible at the previous levels of temporal granularity.

We have developed a system that supports changes in temporal granularity such as those in this scenario. We will describe the technical details of our implementation and its interface. We then will return to this problem to show how our system could support the type of temporal exploration used by the physician in this scenario.

The Database Kernel

To support changes in temporal granularity, our databse groups events into classes that have similar temporal characteristics. This grouping allows common properties and behaviors to be associated with each class of temporal concept defined in the database. We use properties and behaviors specific to each class to perform temporal abstraction and decomposition.

A taxonomy of temporal classes is used to describe temporal and atemporal properties and behaviors associated with all database entities. Table 1 presents the features of the three temporal classes defined in our prototype system. *Simple events* are analogous to the simple $< attribute, time, value >$ tuples in TOD databases. They represent events with no significant temporal duration, called atomic events. *Complex events* represent point events whose time of occurrence cannot be identified with certainty. *Intervals* represent events with temporal duration.

Table 1: Event classes in the database.

Class Name	Temporal Fields	Temporal Representation
Simple Event	Date	Point event without temporal uncertainty
Complex Event	Date, Interval	Point event with an interval of uncertainty
Interval	Event, Event	Interval event with or without uncertainty

Our taxonomy distinguishes between point and interval classes because the properties and behaviors associated with each class differ significantly. Point events are atomic; they contain no additional events. Points occur before, during, or after other points. Intervals may contain both points and subintervals. Intervals also may be overlapping, concurrent, contained, or contiguous to other intervals [1]. We may need to retrieve all events which occur within the duration of a specified interval, but it makes no sense to make the same query of a point event. We also note that the display behavior of point and interval events are different. Interval events need to display temporal duration, whereas point events do not.

Our prototype database is designed for diabetes data management. Data common to this problem domain are blood glucose measurements, meals, and insulin injections (point events), and exercise and unusual events such as hypoglycemia or illness (interval events). Although our initial prototype addresses issues in diabetes data storage and retrieval, our system is applicable to a much wider range of applications, both inside and outside of the domain of medicine.

We have defined application-specific subclasses of the three basic temporal classes to encode temporal entities typically found in diabetes data (Figure 1). Application-specific classes which are interval-based, such as illness-intervals, are subclasses of interval, while classes which are point-based are subclasses of either simple-event or complex-event.

We use object-oriented programming methods [3,8] to implement the temporal classes and to store data instances. A class object stores properties that are common to all instances of that class. Programming procedures associated with each class object implement behaviors that are common to all class instances. Operations among database entities are performed by sending messages to data objects in the database. Object-oriented databases have been the subject of much recent interest [9,7] and their use in biomedical applications is growing [4,6,2].

In Figure 1, the root of the hierarchy *Events,* contains a default action for most defined behaviors. In the object-oriented paradigm, this amounts to defining a function to perform the default behavior. Subclasses specialize the default behaviors by defining functions with the same name as the parent class. These specialized behaviors then become the default behavior for that subclass and any child subclass (Table 2).

Table 2: Sample event-object default and specialized behaviors.

Object Class	Representative Action		
	Draw	Y-location	Double-click
Blood Glucose	Small circle	BG value	Show dialog box
X-Ray	X-Ray icon	0.75	Show XR picture
Event (default)	Small circle	0.50	No action

Our temporal granularity heuristic is encoded within class objects. For each class, we encode knowledge of the temporal granularities at which the events of this class are relevant. In this way, objects can change behaviors depending on the current temporal granularity. We use this property to modify an object's response to queries and for the visual display of information.

Graphical User Interface

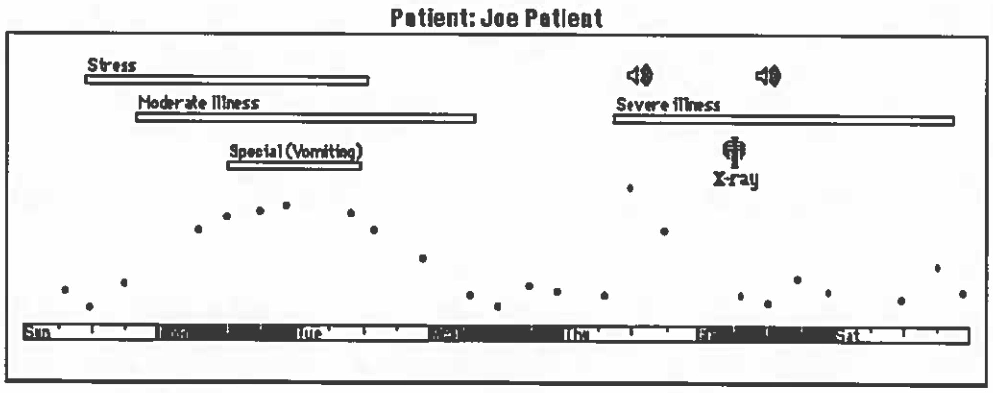

Figure 2: A week of diabetes data in time-line format.

Because people manipulate temporal concepts in a wide variety of ways, a temporal graphical interface must accommodate display forms ranging from a simple interactive plot to an active calendar of complex events. The most basic temporal construct is the simple time-line graph (Figure 2).

In our diabetes display application, we use the vertical position to indicate the value of the blood glucose concentration readings. Other information, such as insulin doses, meals, and clinical studies are positioned uniformly across the display. We minimize display clutter through the use of filters that are accessed using an interactive menu of object classes. These filters are used to determine which object classes will appear on the current display screen. Changing the temporal granularity of data is a second method used to control screen clutter. A change in temporal granularity conceptually is equivalent to "zooming in" on a smaller period of time, effectively adjusting the scale in the horizontal direction. To get a broader view of the data, the user "zooms out" to a week, a month, a year, or some other temporal interval appropriate to the user's needs.

All objects on the display screen are active. This allows us to suppress some of the less important (but still relevant) information about data points while still having that information easily accessible. In general, events respond with a one line summary when the graphical icon representing that event is clicked on with the mouse. Events also respond with a more complete description or appropriate action when double-clicked. For example, when an x-ray icon is single-clicked, it describes itself as "A chest x-ray, taken at 3:00 p.m. on March 3, 1988." Double-clicking on an x-ray icon immediately displays the x-ray in a separate window on the screen (Figure 3).[1] Similarly, we use speaker icons to represent digital sound data, such as voice mail or heartbeats.

The Scenario Revisited

We now examine how the capabilities we have described would have been used by the physician in the SLE patient. Initially, the physician viewed the medical record for the past six months and chose to examine previous visits. In our model, this corresponds to an initial display of six months of the time line and selecting visits as the current level of temporal abstraction. Once at the granularity of a specific visit, the various laboratory tests are visible as icons, and their values can be retrieved by clicking on them directly. In order to perceive the larger trend, the physician needed to change the temporal granularity to view two years. He then could examine summaries of the visits that preceded significant SLE flares by selecting the proper visit icons.

How would changes in temporal granularity be handled using traditional data-handling methods? Most database systems can easily retrieve primary data. In example, the primary data would consist of clinical findings and laboratory values at specific calendar dates. Traditional methods would be able to provide the physician with a series of laboratory values collected over numerous previous visits. The unique feature in our system is a mechanism for dynamically abstracting and decomposing temporal events to respond to changing levels of temporal granularity. Coupled with the interative visualization of time, our system allows the physician to browse patient data in a manner that traditional systems do not provide. We believe that, in many cases, a physician would have difficulties arriving at the conclusions in our example because the level of data abstraction required is typically unavailable.

[1] Although our current screens do not support the display of x-rays at a resolution sufficient to allow for radiologic diagnosis, we believe that for many attending physicians, some decrease in quality may be an acceptable alternative to the time and frustration often associated with manual retrieval of radiographs. The MRI scan shown in this figure is printed on a black-and-white-only device, which is entirely unacceptable for clinical use.

Discussion

Our system design is derived from the object-oriented database used in the ONCOCIN temporal database [4]. ONCOCIN represents temporal points and intervals as objects arranged in complex intertwining hierarchies. A temporal query language specifies access to context-sensitive information residing in the ONCOCIN database. Our system extends this design by incorporating behaviors directly into the data objects. By arranging temporal data into specific classes, we can define class-specific methods for each event. These behaviors allow our system to support changing temporal granularity and temporal abstraction.

Support of Changing Temporal Granularity

Temporal granularity can be defined operationally in terms of the end points of the interval under consideration. Efficiently changing the temporal granularity requires collecting relevant points in the current interval. Locating all point and interval events that are wholly contained within an interval event could be done in a brute force way by examining every point in the database and comparing it to the endpoints of the interval. However, this is very inefficient for large databases. Our solution is to keep all events, including the start and end of interval events, in a temporally ordered list. We locate all point and interval events contained within the temporal extent of an interval simply by beginning at the start of the interval and collecting all events until the end of the interval. Search time is limited to the length of the interval and not by the size of the total data set.

Because of the efficiency of this technique, we have defined a special interval, called the ***display-interval***, which is used extensively by the graphical user interface. The display-interval is used to collect candidate points and intervals that may be displayed in the current screen. Additional criteria supplied by the user, such as a temporal granularity heuristic or specialized data filters, are used to further restrict the set of objects that ultimately appear on the screen.

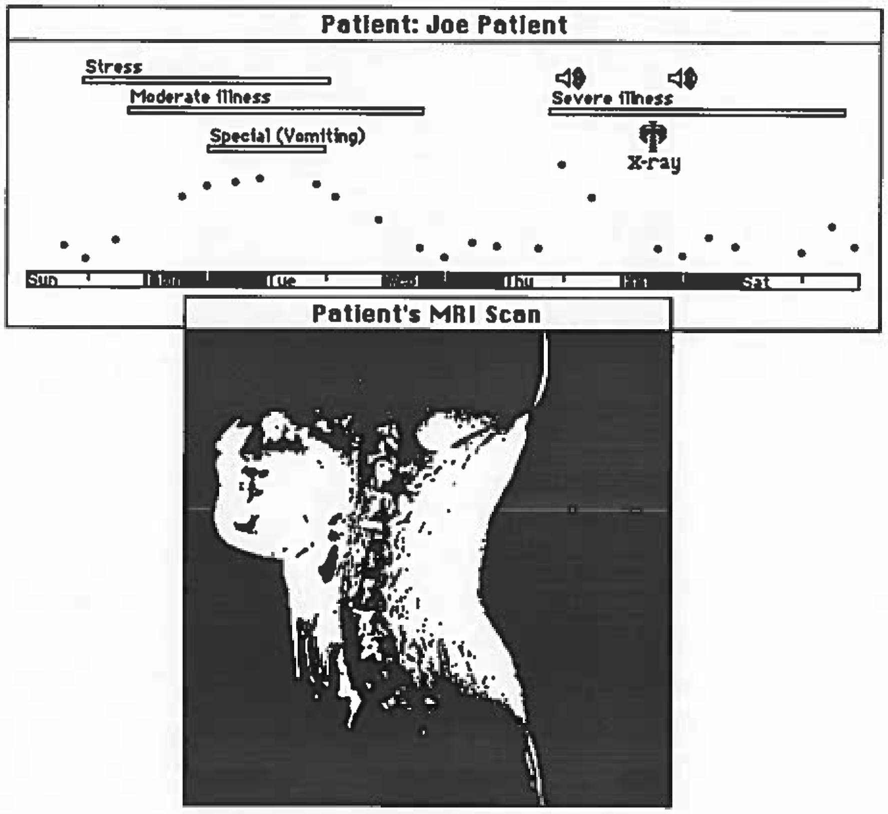

Figure 3: The response of an x-ray icon to double clicking is to display the image.

Support of Temporal Granularity Heuristics

The purpose of a temporal granularity heuristic is to minimize the extraneous detail that always exists at a particular temporal granularity. We seek to find an automated method that suppresses specific information in some problem-solving contexts, but makes these details available when appropriate. Because of the vague nature of "when appropriate", the best we can hope for is a heuristic. By encoding the information about relevant temporal granularities in each class of events, our architecture provides a simple way to customize various temporal granularity heuristics.

Handling Database Growth

Accommodating large quantities of data is a serious issue in any real database. We minimize this problem through the use of data abstraction techniques which aggregate irrelevant data. For example, when reasoning about a previous hospitalization, it is often reasonable (and desirable) to suppress minor details of the hospital course. Abstraction aggregates detailed information into a less detailed composite concept. This technique represents a trade-off between the legally necessary retention of all medical data and the computationally expensive retrieval of data in a medical problem-solving environment. We propose to remove the primary data from the active portion of the medical record, leaving only a hospital interval abstraction, which has a pointer to a long-term storage device (such as a tape label) that contains the detailed data. If the hospital abstraction interval ever is queried for more details, the object's response might be to request that the appropriate magnetic tape be mounted so that the requested details could be re-incorporated into the patient's active history.

Future Directions

The use of LISP to implement our prototype imposes some significant limitations. The LISP object-oriented system we are using to prototype assumes all data objects are in memory. Object LISP is not efficient enough in terms of memory usage. Although we are not concerned with the efficiency of initial prototypes, large data sets will readily overload the capacity of our prototype. Our object system needs to be extended to disk-based objects. Two alternatives that we are considering that could minimize these limitations are to recode the database kernel in a compiled language such as C++ and to implement disk-based object-caching techniques within Object LISP.

We have described an object-oriented database which allows computer programs to have efficient access to temporal data. To complement the database, we have implemented an interactive graphical browser which lets humans visualize the contents of a diabetes database at different levels of temporal granularity. The ability to easily visualize clinically-meaningful temporal abstractions opens new possibilities for interactive data browsing. We are working on defining an algebra of operations that can be used to graphically examine temporal data in search of patterns and trends.

Acknowledgments

This work was supported by grants from Miles Laboratory, Inc., the New Medical Foundation, and Apple Computer Corporation. Dr. Frisse is a Teaching and Research Scholar of the American College of Physicians. We thank Drs. Gil Jost and G. James Blaine, Electronic Radiology Laboratory, Mallinckrodt Institute of Radiology, Washington University School of Medicine, for providing us with a stimulating environment for this work. Finally, we thank the reviewers for making comments which guided us to significant improvements in this manuscript.

References

[1] JF Allen. Maintaining knowledge about temporal intervals. *Communcations of the ACM*, 26:832–843, 1983.

[2] T Barsalou. An object-based architecture for biomedical expert database systems. In RA Greenes, Editor, *Proceedings, Symposium on Computer Applications in Medical Care*, pages 572–578, Washington DC, 1988. IEEE Computer Society.

[3] BJ Cox. *Object-Oriented Programming: An Evolutionary Approach*. Addison-Wesley, Reading, MA, 1986.

[4] MG Kahn, JC Ferguson, EH Shortliffe, and LM Fagan. Representation and use of temporal information in ONCOCIN. In MJ Ackerman, Editor, *Proceedings Symposium on Computer Applications in Medical Care*, pages 172–176, Baltimore, MD, 1985. IEEE Computer Society.

[5] B Kuipers. Abstraction by time-scale in qualitative simulation. In *Proceedings AAAI-87*, Philadelphia, PA, 1987. Morgan Kaufmann.

[6] CD Lane, ME Frisse, LM Fagan, and EH Shortliffe. Object-oriented graphics in medical interface design. In A Levey and B Williams, Editors, *Proceedings AAMSI Congress 86*, pages 293–297, Anaheim, CA, 1986. American Association for Medical Systems and Informatics.

[7] RW Peterson. Object-oriented database design. *AI Expert*, pages 26–31, 1987.

[8] M Stefik and DG Bobrow. Object-oriented programming: Themes and variations. *AI Magazine*, pages 40–62, 1986.

[9] G Wiederhold. Views, objects, and databases. *IEEE Computer*, 19:37–44, 1986.

[10] G Wiederhold, JF Fries, and S Weyl. Structured organization of clinical data bases. In DA Meier and SW Miller, Editors, *AFIPS Conference Proceedings*, Volume 44, pages 479–485, Montvale, NJ, 1975. AFIPS Press.