Temporal and Spatio-Temporal Databases

(continuation of Spatial …)

Robert M. Keller

Harvey Mudd College

April 2004

1

Possible Applications

- Personnel or medical records
- Reservation systems
- Archaeological records
- Calendars, planning, scheduling, configuration management
- Real-time process monitoring, data streaming

2

Breadth of Interest

- According to Zaniolo, et al., 1995:
  - It is hard to find applications that *don't* involve some form of temporal information.
  - Over 40 distinct temporal data models have been defined.

3

Temporal Concepts

- Represent time of **events**
- Represent time **intervals**
- Represent **histories** (= mapping from time interval to events or value)
- Represent **periods** of repeating events
- Sometimes only need ***sequences*** of events rather than times themselves:
  - Loosely think of this as "time".

4

"Valid-Temporal" Concepts

- Conventional data base:
  - Tuple represents a fact
- Temporal database:
  - Tuple represents a fact that:
    - holds at a specified time (called the ***valid* time**), ***or***
    - holds over an **interval** of time (called an **invariant**) over the interval (which can include the future as well as the past)
    - holds at **some** time during an interval

5

Sample Valid-Temporal Queries

- Was Mary the manager of the IT Department during the year 2002?
- Find all managers of the IT Department from 1/1/2000 to 12/31/2003.
- Find out who was promoted during the year 2003.
- What was the configuration of the system on March 12, 2001?

6

"Transaction-Temporal"

- Tuple can also show that time at which it was **stored** in the database.
- Note possible connection with multi-version DB.
- "Bi-temporal": both temporal and transaction temporal.
- Ancillary information: Who authorized the change?
- Might also want to show when info **deleted,** when a request to delete was made, etc.

7

Transaction-Temporal Queries = Introspective

- Queries that ask questions about changes to the database.
- When was the information that John was promoted to manager inserted?
- When was the fact that John is manager of the sales department deleted?
- When was the configuration of the system changed to include the TeraTuple 2000 disk farm?

8

Transaction Time

- Interval representation - **DBMS maintained**

		Transaction Time	
Movie	*Actor*	*start*	*end*
Toy Story	Tom	Jan/4/1999	*now*
Lion King	Mustafa	Jan/4/1999	*now*
True Lies	Arnuld	Jan/4/1999	Jul/1/2000
True Lies	Arnold	Jul/2/2000	*now*

- On Jul/2/2000

```
DELETE FROM Movie WHERE Actor='Arnuld';
INSERT INTO Movie ('True Lies', 'Arnold');
```

- **Schema evolution** (e.g., Roddick and Snodgrass, *TSQL2*, 1995) Schema itself can change

9

Example: Stock Market DB

- Stock price changes throughout session.
- Price has 3 components: **ask** (by seller), **bid** (by buyer), **sold**
- Bid transaction includes amount buying, price, expiration, buyer, broker
- Ask transaction includes amount offered, price, expiration, seller, broker
- Sell transaction includes amount sold, price, buyer, seller, broker
- Other transactions:
  - Cancel transaction
  - Limit order transaction
  - Stop-loss order transaction
- Which are valid-time and which are transaction-time aspects?

10

Formal Time Models

- Continuous, Rational, Discrete, or Refinable
- Finite, singly-infinite, doubly-infinite
- Linear or Branching (Non-Determinism)
- Centralized or Distributed
- Qualitative or Quantitative

11

Example: TEMPOS

- **T**emporal **E**xtension **M**odels for **P**ersistent **O**bject **S**ervers
- Dumas, et al., University of Grenoble
- Extends the Object Database Management Group's (ODMG) object model by enabling classes to be declared as temporal.
- Values are classified as either:
  - Temporal, or
  - Fleeting (only the most recent value kept)
- Recall that this is a precursor to JDO.
- Extends languages:
  - ODL →Temporal ODL
  - OQL →Temporal OQL
- Was implemented atop O_2 (an OODBMS).

12

Formal Modeling

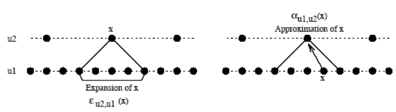
Definition 1 (*data model*). A data model M is as a quadruplet (D, Q, U, $[\![\,]\!]_M$) composed of a set of database instances D, a set of legal query statements Q, a set of legal update statements U, and an evaluation function $[\![\,]\!]_M$. Given an update statement u $\in$ U, a query statement q $\in$ Q and a database instance db $\in$ D, $[\![u(db)]\!]_M$ yields a database instance, and $[\![q(db)]\!]_M$ yields an instance of some data structure. $\Box$

TEMPOS Time Granularity Modeling

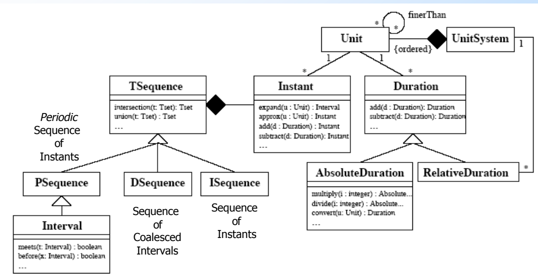
We adopt a discrete, linear and bounded time model in which time is structured in a *multi-granular* way [CR87, WJS95] by means of *time units*. A time unit is a partition of the time line into a set of convex sets: each of the elements of this partition is then seen as an atomic *granule* and every point of the time-line is approximated by the granule which contains it. Thus a time unit defines the precision at which time is observed. The granules of a time unit are numbered by natural integers: the order among these integers defines the notion of succession in time and the distance between them defines the notion of duration.

TEMPOS Time Granularity Modeling

For each pair of time units ⟨u1, u2⟩ such that u1 $\prec$ u2, two conversion functions are defined: one for *expanding* a granule of the coarser unit (u2) into an interval of granules of the finer one (u1) (noted $\epsilon_{u2,u1}$), and the other for *approximating* a granule of u1 by a granule of u2 (noted $\alpha_{u1,u2}$), as shown in figure 1. Units u1 and u2 (u1 $\prec$ u2) are said to be *regular* if the intervals of granules generated by $\epsilon_{u2,u1}$ have all the same cardinality.



Temporal Types Class Diagram

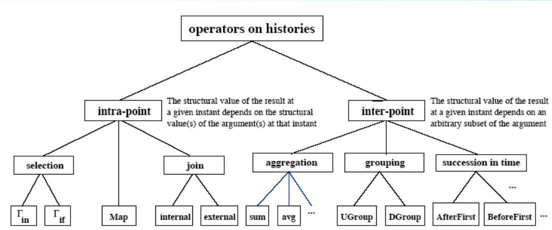


Histories

Histories may be represented in several ways, mainly by means of collections of timestamped values, termed *chronicles*. Among these representations some are useful for query expression, so that specific operators are defined on the History ADT allowing to represent a history in any of these forms. Concretely, a history may be represented by at least three kinds of chronicles:

- Instant-based representation: chronologically ordered sequence of instant-timestamped values, e.g. [⟨1, v1⟩, ⟨2, v1⟩, ⟨4, v1⟩, ⟨5, v2⟩, ⟨6, v2⟩, ⟨7, v2⟩, ⟨8, v3⟩, ⟨9, v1⟩, ⟨10, v1⟩]. Such sequences are termed IChronicles.
- Interval-based representation: chronologically ordered, coalesced sequence of interval-timestamped values, e.g. [⟨[1..2], v1⟩, ⟨[4..4], v1⟩, ⟨[5..7], v2⟩, ⟨[8..8], v3⟩, ⟨[9..10], v1⟩]. This kind of sequence is called an XChronicle.
- Temporal sequence-based representation: set of distinct values timestamped by disjoint temporal sequences, e.g. { ⟨{1, 2, 4, 9, 10}, v1⟩, ⟨{5, 6, 7}, v2⟩, ⟨{ 8 }, v3⟩ }, which are termed DChronicles.

Operators on Histories



Example: Temporal Join

The inner temporal join of two histories ($h1 *_\cap h2$) is a history whose structural values are pairs obtained by combining "synchronous" values of h1 and h2 (i.e. values attached to the same instant). The outer temporal join ($h1 *_\cup h2$), is similar to the corresponding inner temporal join, except that it attaches structural values of the form $\langle v, Nil \rangle$ or $\langle Nil, v \rangle$, to those instants where one of the argument histories is defined while the other is not. Here, Nil denotes the neutral element of the structural domain type (e.g. 0 for integers, nil for objects, etc.). More precisely:

$\_ *_\cap \_ :$ History<T1>, History<T2> $\rightarrow$ History<$\langle$T1,T2$\rangle$>

/* $h1 *_\cap h2 = \{\langle I, \langle v1, v2\rangle\rangle \mid \langle I, v1\rangle \in h1 \wedge \langle I, v2\rangle \in h2\}$ */

/* *precondition: Unit(h1) = Unit(h2)* */

Temporal Join Example

(from Zaniolo, et al. 1995)

- Employee1:

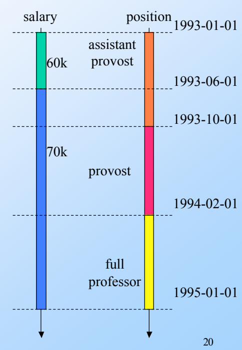
Name	Salary	Start	Stop
Bob	60000	1993-01-01	1993-06-01
Bob	70000	1993-06-01	1995-01-01

- Employee2:

Name	Title	Start	Stop
Bob	Assistant Provost	1993-01-01	1993-10-01
Bob	Provost	1993-10-01	1994-02-01
Bob	Full Professor	1994-02-01	1995-01-01

- Result of temporal join:

Name	Salary	Title	Start	Stop
Bob	60000	Assistant Provost	1993-01-01	1993-06-01
Bob	70000	Assistant Provost	1993-06-01	1993-10-01
Bob	70000	Provost	1993-10-01	1994-02-01
Bob	70000	Full Professor	1994-02-01	1995-01-01



Temporal Join in Plain SQL

(from Zaniolo, et al. 1995)

```
SELECT Employee1.Name, Salary, Title,
                 Employee1.Start, Employee1.Stop
FROM Employee1, Employee2
WHERE Employee1.Name = Employee2.Name
     AND Employee2.Start <= Employee1.Start
     AND Employee1.Stop <= Employee2.Stop
UNION ALL
SELECT Employee1.Name, Salary, Title,
                 Employee1.Start, Employee2.Stop
FROM Employee1, Employee2
WHERE Employee1.Name = Employee2.Name
     AND Employee1.Start > Employee2.Start
     AND Employee2.Stop < Employee1.Stop
     AND Employee1.Start < Employee2.Stop
UNION ALL
SELECT Employee1.Name, Salary, Title
                 Employee2.Start, Employee1.Stop
FROM Employee1, Employee2
WHERE Employee1.Name = Employee2.Name
      AND Employee2.Start > Employee1.Start
      AND Employee1.Stop <= Employee2.Stop
      AND Employee2.Start < Employee1.Stop
UNION ALL
SELECT Employee1.Name, Salary, Title
                 Employee2.Start, Employee2.Stop
FROM Employee1, Employee2
WHERE Employee1.Name = Employee2 Name
      AND Employee2.Start => Employee1.Start
      AND Employee2.Stop <= Employee1.Stop
      AND NOT (Employee1.Start = Employee2.Start
               AND Employee1.Stop = Employee2.Stop)
```


Temporal Join in TSQL2

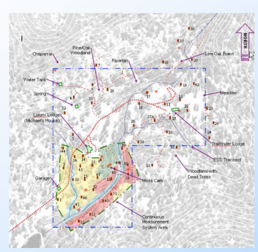
(from Zaniolo, et al. 1995)

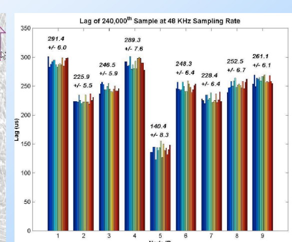
```
SELECT Employee1.Name, Salary, Title
FROM Employee1, Employee2
WHERE Employee1.Name = Employee2.Name
```


Example Spatio-Temporal Applications

- Record the historical evolution of a landform (such as a volcanic region, or global earthquakes)
- Understand weather data, or flow of groundwater and contaminants
- Chart the propagation of spoken or written languages
- Trace the migration of a population of animals
- Anticipate the spread of an epidemic
- MRI or PET imaging
- Track terrorists
- Choreography, any motion picture or animation, video games

A well-sampling application that uses wireless sensor networks (Ganeson, et al., UCLA)



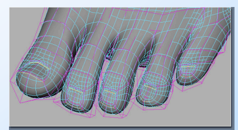


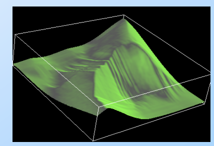
General Desiderata for Spatio-Temporal Models

- Should be upward compatible from classical methods (i.e. should subsume them)
- Deal with interactions of time and space, not just the two independently (e.g. space-time constraints)
- Expressive, yet easy to use
- Compatible with existing frameworks

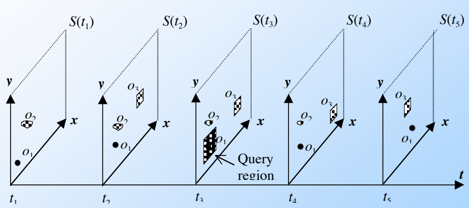
Possible Test-beds

- Any of a variety of 3D modeling tools, such as Maya, AutoCAD, GeoToolKit



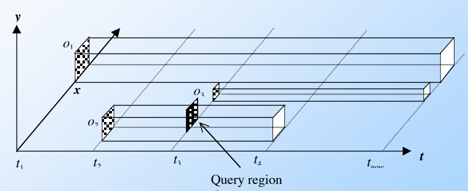


Spatio-temporal Evolution



Indexing using R-trees

- Assume that time is another dimension, use a 3D R-tree



Additional Directions

- Constraint Logic Programming
- Machine learning
- Data mining

Assertion: The WWW is a Database

- What aspects are:
  - relational?
  - object-oriented?
  - temporal?

Query Language

- URL is, in effect, a primitive query
  - identifies single, whole document as the answer
- Some information exists, but cannot be queried
  - Requires natural language text understanding

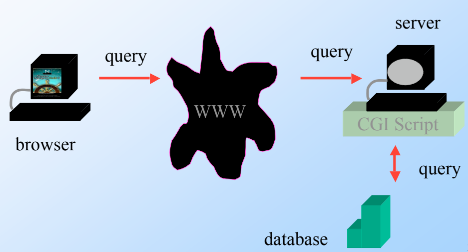
31

Other Ways to Access Data over WWW

- CGI/HTTP
- Applets
- Servlets (= server-side applets)
- .NET

32

HTTP (CGI) Gateway



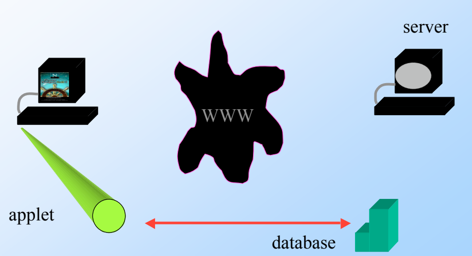
33

HTTP (CGI) Gateway

- HTML forms, limited GUI
- result formatting
- HTTP/database protocol mismatch
  - "stateless" protocol
  - single query transactions

34

Java Applets



35

Java Applet

- "stateful"
- fancy GUI
- robust, secure, platform-independent
- database API - JDBC

36

WWW Data Structure

- WWW is data repository
- unstructured (semi-structured)
  - HTML tags
  - XML tags
  - WWW graph (graph of hyperlinks)
- primitive query
  - URL identifies single, whole document

37

Consistency

- redundant information
  - plenty of it: hardly normalized
- incorrect information
  - 5% of links are broken
  - little certification, if any
- triggers/assertions/constraints
  - guarantee consistency

38

An Issue for Investigation

- Does the WWW support transaction-time temporality? If not, can it be extended to do so?

39