MINING TOPOLOGICAL RELATIONSHIP PATTERNS FROM SPATIOTEMPORAL DATABASES

K.Venkateswara Rao[1], Dr.A.Govardhan[2] and Dr.K.V.Chalapati Rao[1]

[1]Department of Computer Science and Engineering, CVR College of Engineering, Ibrahimpatnam RR District, Andhra Pradesh, India
kvenkat.cse@gmail.com, chalapatiraokv@gmail.com
[2]JNTUH College of Engineering, Jagityala, Karimnagar Dist, Andhra Pradesh, India
govardhan_cse@yahoo.co.in

ABSTRACT

Mining topological relationship patterns involve three aspects. First one is the discovery of geometric relationships like disjoint, cover, intersection and overlap between every pair of spatiotemporal objects. Second one is tracking the change of such relationships with time from spatiotemporal databases. Third one is mining the topological relationship patterns. Spatiotemporal databases deal with changes to spatial objects with time. The applications in this domain process spatial, temporal and attribute data elements to find the evolution of spatial objects and changes in their topological relationships with time. These advanced database applications require storing, management and processing of complex spatiotemporal data. In this paper we discuss a model-view-controller based architecture of the system, the design of spatiotemporal database and methodology for mining spatiotemporal topological relationship patterns. Prototype implementation of the system is carried out on top of open source object relational spatial database management system called postgresql and postgis. The algorithms are experimented on historical cadastral datasets that are created using OpenJump. The resulting topological relationship patterns are presented.

KEYWORDS

Spatiotemporal database, topological relationships, topological relationship pattern, model-view-controller

1. INTRODUCTION

Spatio-Temporal applications like temporal geographic information systems [1] and environmental systems [2] process spatial, temporal and attribute data elements of spatiotemporal objects for knowledge discovery. The spatial objects are characterized by their position, shape and spatial attributes. Spatial attributes are properties of space and spatial objects located in specific positions that inherit these attributes. Spatial attributes refer to the whole space and can be represented as layers and each layer represents one theme. The temporal objects are characterized by two models of time that are used to record facts and information about spatial objects. The two models of time are time points and time intervals. A time point is considered as one chronon, while a time interval has duration and is defined as set of chronons. Time points and time intervals can represent valid or transaction time. Valid time shows when a fact is true. There are two basic facts, events and states, for which time is recorded. An event occurs at an exact time point, i.e., an event has no duration. Example events are "car crash," "sunrise," etc. A state is defined for each chronon in a time interval, hence it has duration. For example, a "meeting" takes place from 9am until 11am.
Spatiotemporal object captures simultaneously spatial and temporal aspects of data and deal with geometry changing over time. It can be represented by a four tuple - object id, geometry, time and attributes [3]. Recording a spatial object at a time point results in a snapshot of it. For

example, capturing snapshots of a "landparcel" that changes its shape (e.g., split, expanded etc) during certain period of time. Recording a spatial object in a time interval is translated into capturing its evolution over time, i.e., capturing the possible change*s* of its shape over time. Consider the example of recording a "landparcel" in [2006, 2009], which changes shape.

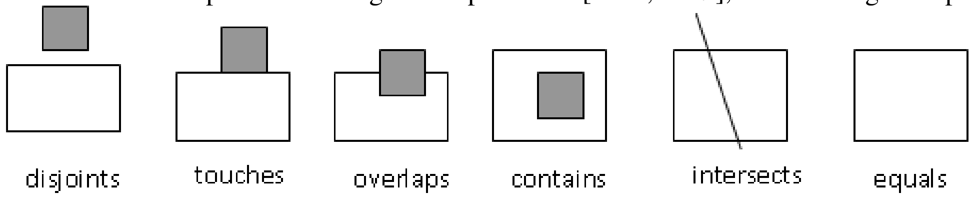

Fig 1: Spatial topological relationships

Topology describes spatial relationships like intersects, meets, overlaps, equals etc, between spatial objects. The spatial objects may be point, line or polygon. Different types of spatial topological relationships between two objects are shown in fig., 1.
The topological relationship between two spatial objects may change if geometry of any one of the spatial objects changes. The geometry changes of spatial objects with time are generally captured and stored in spatiotemporal databases. The changing topological relationship among spatial objects with time is represented using topological relationship pattern. For example, the topological relationship change between two spatial objects O1 and O2 from time t1 to t4 is shown in Fig., 2. The topological relationship pattern for this example can be represented as D-O-C-T where D, O, C, T corresponds to disjoints, overlaps, contains and touches respectively.

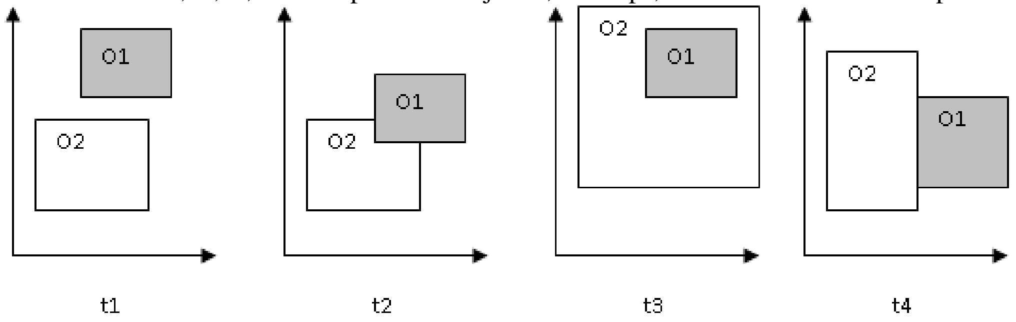

Fig 2: Time-varying Topology in an interval [t1, t4]

The purpose of this research is design of system architecture and an extendable spatiotemporal database, and also developing a methodology to mine topological relationship patterns from spatiotemporal data bases. The architecture of the system is described in section 2. Section 3 discusses the database design that can capture the changing geometry of spatiotemporal objects. Section 4 elaborates methodology and algorithms for mining topological relationship patterns. Section 5 briefs about the implementation. Results are provided in section 6. Conclusion and some directions for future work are given in section 7.

2. MODEL-VIEW-CONTROLLER ARCHITECTURE OF THE SYSTEM

A model-view-controller (MVC) architectural pattern [4, 5] shown in Fig., 3 can be used to specify the architecture of the system for mining topological relationship patterns from spatiotemporal databases. The model encapsulates the spatiotemporal data and contains functions and application logic. The View of MVC pattern presents topological relationship patterns to the user. It gets this information from the model. Multiple views can be there for each model. Each view will have an associated controller.

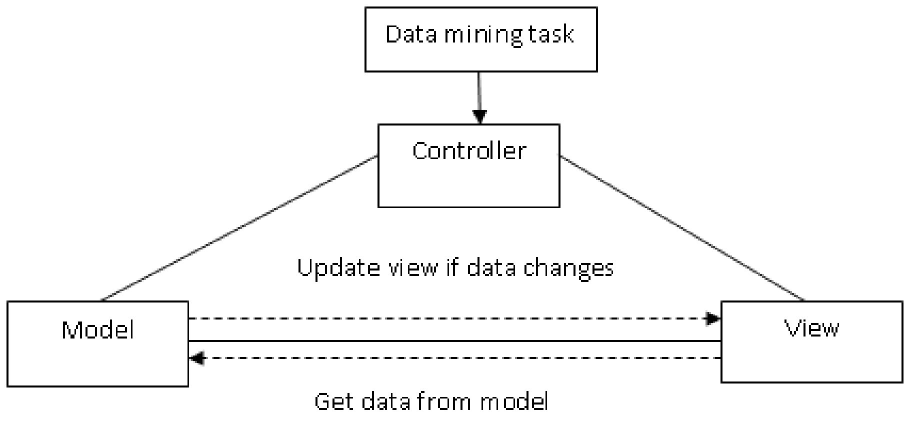

Fig 3: MVC Architecture

The controller receives user input and translates it into a service requests for the model or the view. For example, user can give duration of the time in terms of from-time and to-time, and type of spatial objects for which topological relationship patterns are to be discovered over the given time period. The model activates appropriate functions to fetch task relevant data, based on user input that comes through the controller. It then, processes the data to discover spatiotemporal topological relationships and organize the results in multidimensional model. It applies an appropriate logic or functions to mine the topological relationship patterns from the multidimensional data model. The views can display spatiotemporal topological relationships [6] and topological relationship patterns to the user. The model, view and controller are described using Class – Responsibility – Collaboration (CRC) cards as follows.

Class Model	**Collaborators** View Controller
Responsibility Fetches task relevant data from underlying spatiotemporal database. Pre-Processes the data. Discovers spatiotemporal topological relationships. Handles intermediate results in multidimensional model Mines topological relationship patterns. Registers dependent views and controllers. Notifies dependent components (Views) about knowledge discovery or change.	

Class View	**Collaborators** Controller Model
Responsibility Creates and initializes its associated controller. Displays spatiotemporal topological relationships and topological relationship patterns. Implements the update procedure to reflect changes in the model. Retrieves knowledge from the model. Allows controller to select view.	

Class Controller	**Collaborators** View Model
Responsibility Accepts spatiotemporal topological relationship pattern mining tasks from user as events. Processes user events into service request to the model. Select view to provide response to the user task.	

3. SPATIOTEMPORAL DATABASE DESIGN

An object relational spatiotemporal database which consists of a set of tables and relationships among them is designed to meet the spatiotemporal database application requirements [7]. Different sorts of spatiotemporal data to be handled are states, events and episodes. A state represents a version of an entity in a given moment. States can consist of different versions of an individual entity. An event is the moment in time when an occurrence takes place. Event causes one state to change to another. An episode is the length of time during which change occurs, a state exists or an event lasts. Two main strategies to represent multiple versions of an object are tracking the versions either at the level of objects or attributes. First one involves a different identifier (oid) to each new version and chaining the older versions to new oid. Second one involves a single object identity (oid) with versions actually associated with attributes. The attributes of spatiotemporal objects can be categorized as version significant, non-version significant and invariant. The version significant attribute values are to be updated in non-destructive manner, the non-version significant attribute values are to be updated in a destructive manner and invariant attributes values are not allowed to be changed. Following entities are designed to address these requirements.

1. Temporal_tab : This table stores timestamps which correspond to the time at which change to any spatial object has taken place.
2. Spatial_obj_tab : This table contains spatiotemporal objects with unique identifier, geometry and existence time which has 'from time(st) and to time(et)' as attributes. It also has other attributes to indicate category and type of spatial object and type of change. The events and episodes or processes are also considered as spatiotemporal objects and stored in this table.
3. split_tab: This composite entity is used to record splitting of any spatial object into multiple objects. It has object identifier that got split and new object identifiers for objects derived due to split and timestamp attribute that records time of split.
4. Merge_tab: This composite entity keeps the data related to merging of two or more objects into a single object. It maintains object identifiers which are merged, new object identifier for object derived due to merging and timestamp attribute which records the time of merge. The new objects created due to split or merge are stored in spatial_obj_tab with their new object identifiers.
5. Geom_version: This table records geometry changes by creating new object identifier for each change to the geometry of the object. It has oid of the object changed, new object identifier and timestamp attribute that records time of the change.The new object is stored in spatial_obj_tab table.
6. VsAttr_tab: This table manages version significant attributes of all objects in spatial_obj_tab. It has oid, attribute name, timestamp and attribute value as its fields.
7. VinAttr_tab: This table manages version insignificant and invariant attributes of all objects in spatial_obj_tab. It has oid, attribute name and attribute value as its fields.
8. Result_tab : This entity is used by analysis algorithms to store results back into database. This table can be accessed using OpenJump (An Open source GIS software) to visualize the results.

4. METHODOLOGY

Mining topological relationship patterns from spatiotemporal databases for the given two types of spatial objects over a specified period of time involves following steps.

1. Fetching the task relevant data by the model based on input received from the controller and discovering spatiotemporal topological relationships[8] for every pair of spatial objects (each is from one type specified in the input) using the functions in the model.
2. Generation of multidimensional model, say M, containing topological relationships in the cells of the model with following dimensions.
   Dimension 1: List of spatial objects of type1.
   Dimension 2: List of spatial objects of type2.
   Dimension 3: Time points in the given time period.
   The topological relationships can be represented in the cells as T for touches, D for disjoints, O for Overlaps, C for contains, I for intersects and E for equals.
3. Finding topological relationship changes for every pair of objects (each from one dimension) along the time dimension and generating the following intermediate table. This is done by functions in the model of MVC.

Type1 Object	Type2 Object	Topological relationship pattern
O1	O2	D-T-O-C-D
…	…	…

4. Finding the support count for each topological pattern by the model. Based on user specified threshold value for minimum support, display of frequent topological relationship patterns is done by the View.

The algorithms designed for tracking spatial object, finding topological relationships and mining topological relationship patterns to implement the methodology are described below.

4.1 Algorithms

```
Algorithm 1: Tracking Spatial object.
Input: Spatial Object Identifier (oid), LIST – List of spatial object identifiers, empty LinkedList
Output: Linked List containing Object identifiers.
Track_Obj(Obj,LIST,LinkedList)
Begin
Next = Obj.changeType
If( Next == 'C')
Begin
        Look into Geom_version table and find new object identifier (n_oid).
        Check for the presence of n-oid in the LIST.
        If n_oid is in LIST,
        Begin
          add it to the LinkedList.
         Track(n_oid,LIST)
        End
End
End
```

```
Algorithm 2: Finding topological relationships between two objects.
Input: : Spatiotemporal dataset (D), Object Identifier (oid1, oid2) and time (t).
Output: Topological relationships between the oid1 and oid2.
Method : ComputeTrelationship(oid,oid2,t)
Begin
```

```
    1. Access the data set and get geometry details of the given objects oid1 and oid2 at given
       time t. Create an object of type GeometryRelation class for oid1 and oid2.
    2. Using the methods of the class, compute topological relationships between the pair of
       objects.
    3. Store the results in an appropriate cell of the multidimensional model M.
End
```

```
Public class GeometryRelation
{
        PGgeometry obj1, obj2;
        Methods:
        PGgeometry Union();
        PGgeometry Intersection();
        Float Distance();
        Boolean isintersects()
        Boolean istouches()
        Boolean isequals()
        Boolean isdisjoint()
        Boolean iscrosses()
        Boolean isoverlaps()
        Boolean iscovers()
        Boolean iscoveredby()
}
```

All the methods of GeometryRelation class are implemented using Postgis application programming interface.

Algorithm 3: Mining spatiotemporal topological relationship patterns.
Input: : Spatiotemporal dataset (D), Object Types (type11, type2), from time(f_t) and to time(t_t)
Output: Topological relationship patterns between sets of two types objects over a specified period of time.
Method :

```
Begin
    1. Find two lists of object identifiers which are existing over the given period of time in
       the spatiotemporal dataset, D. Say LIST1 and LIST2. Each list is for one type of
       objects. Arrange oids in LIST1 and LIST2 in ascending order of their creation time.
    2. For each remaining object  identifier oid in the LIST1,
    3. Generate linked list, L, for the object, oid, by calling Track_Obj(oid,LIST1,L)
       algorithm. Eliminate object identifiers which are in L from the LIST.
    4.  Steps 2 and 3generate array of linked lists, say ArrayOfLnksForLIST1, for LIST1.
    5. Repeat step2 to step4 for LIST2 and generate ArrayOfLnksForLIST2 for LIST2.
    6. Represent first object identifier in each linked list in ArrayOfLnksForLIST1 as a co-
       ordinate point in Dimension1. This yields points like d11,d12,d13, …,d1i on
       Dimension1. Note that each point has its linked list.
    7. Represent first object identifier in each linked list in ArrayOfLnksForLIST2 as a co-
       ordinate point in Dimension2. This yields points like d21,d22,d23, …,d2j on
       Dimension2. Note that each point has its linked list.
    8. Collect all timestamps from ArrayOfLnksForLIST1 and ArrayOfLnksForLIST2.
       Arrange them in ascending order on Dimension3.  Steps 6, 7, 8 create a three
       dimensional model, say M.
    9. For each cell c(m,n,t) of three dimensional model, call
       ComputeTrelationship(oid1,oid2,t) where oid1 is an appropriate object identifier from
```

the linked list corresponding to mth ordinate of the cell c, oid2 is an appropriate object identifier from the linked list corresponding to nth ordinate of the cell c. Both oid1 and oid2 are appropriate means they have an existence at time point t. This can be found by looking into the respective linked lists.

10. For each pair of coordinates from Dimension1 and Dimension2 of the Multidimensional model, search for topological relationships along the time dimension and generate a pattern. Arrange these patterns in a table format specified in step3 of the methodology.
11. Sort the table generated in step10 based on the topological relationship pattern field and compute support count for each pattern. Those Patterns whose support count exceeds user specified threshold for minimum support are considered for display by the view of MVC .

End

5. IMPLEMENTATION

The system is implemented using postgresql [9], postgis [10], Java, JDBC and OpenJump [11] technologies. The object relational database is created and used in prototype implementation of the system for mining spatiotemporal topological relationship patterns. The versions created due to split and merge processes or events are managed by considering each version as a new object. The linkage between parent and children objects is maintained using primary key foreign key relationship among the appropriate tables.

The sample database objects are generated using OpenJump and loaded into the database. The modules for accessing the database and the designed algorithms are implemented in JAVA. The application programming interface provided by the postgis is used for all geometry related computations.

6. RESULTS

The system is tested with cadastral datasets to discover how topological relationships between "land parcel" objects and flood objects change over a given period of time. Then topological relationship patterns are mined. In the test data "land parcels" are considered as one type of spatial objects and flood geometries are considered as another data type. The duration of time considered is from 2000-01-01 12:12:12 to 2000-12-31 12:12:12. The topological relationships between every pair of the objects are generated [8] and stored in an intermediate table for further processing to discover topological relationship patterns. The Discovered topological relationship patterns with their support count are shown in the following table as a sample.

Topological relationship Pattern	Support count
D-T-O-C-D	305
D-T-D	290
D-O-D-O-T-D	508

7. CONCLUSION AND FUTURE WORK

In this paper, the concepts of topological relationships and topological relationship patterns are described. A Model-View-Controller architectural pattern is used to describe the system architecture. The responsibilities of the model, the view the controller with respect to mining topological relationship patterns are elaborated using Class-Responsibility-Collaboration (CRC) cards.. A generic design of spatiotemporal database whose schema handles spatial and non-spatial attributes of spatial objects is discussed. The database also facilitates to capture change of such attributes over time. The methodology and algorithms to discover spatiotemporal topological relationship patterns are described. The system is implemented using open source software postgresql, postgis. Spatiotemporal data sets are created using

OpenJump. The algorithms are tested for their accuracy and the results of mining topological relationship patterns from cadastral database are provided. The algorithms can be extended to indicate length of time for each topological relationship within the topological relationship pattern. The system can be extended for further analysis of spatiotemporal attributes to reflect their trend and patterns of change.

REFERENCE

[1] Yang ping , Tang Xinming , Wang Shengxiao, “ Dynamic cartographic representation of Spatio-Temporal data “ The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences. Vol. XXXVII. Part B2. Beijing 2008

[2] Z. Obradovic , D. Das, V.Radosavljevic, K.Ristovski, S.Vucetic, “ Spatio-Temporal characterization of aerosols through active use of data from multiple sensors “ ISPRS TC VII Symposium – 100 Years ISPRS, Vienna, Austria, July 5–7, 2010

[3] Nikos Pelekis et al, “ Literature Review of Spatio-Temporal Database Models “, The Knowledge Engineering Review, 2004, 235-274 Cambridge university Press.

[4] Frank Buschmann etal, Pattern-Oriented Software Architecture: A System of Patterns,Wiley India, 125-144.

[5] M. Castellano, N. Pastore, F. Arcieri, V. Summo & G. Bellone de Grecis, A Model-View-Controller architecture for Knowledge Discovery, WIT eLibrary.

[6] Peiquan Jin and Lihua Yue,”A Framework for the Description of Spatiotemporal Relationships”, Proceeings of the Joint conference on information Sciences, October, 2006.

[7] K.Venkateswara Rao, .A.Govardhan and K.V.Chalapati Rao (2011) “An Object-Oriented Modeling and Implementation of Spatio-Temporal Knowledge Discovery System”**,** *International Journal of Computer Science & Information Technology (IJCSIT),* Vol 3, No 2, April 2011.

[8] K. Venkateswara Rao, A. Govardhan and K.V.Chalapati Rao (2011) “Discovering Spatiotemporal Topological Relationships”, Proceeding of First International Conference on Advances in Computing and Information Technology, Chennai, India, July 2011.

[9] http://www.postgresql.org/docs/manuals/

[10] http://postgis.refractions.net/documentation/

[11] http://www.openjump.org/