

Spatio-temporal change representation and map updates in a dynamic Voronoi data structure

Darka Mioc^{*†}, François Anton^{*†}, Christopher M. Gold^{*†}, and Bernard Moulin^{◇†}

^{*}Industrial Chair in Geomatics

Université Laval, 0722 Casault

Sainte-Foy, Quebec, Canada, G1K 7P4

Phone: (+1) 418-656-3308; Fax: (+1) 418-656-7411

Email: {francois, dmioc}@gmt.ulaval.ca; Christopher.Gold@scg.ulaval.ca.

[†] Centre for Research in Geomatics, Laval University

[◇] Département d'Informatique

Faculté des sciences et de génie, Université Laval

Sainte-Foy, Quebec, Canada, G1K 7P4

Phone: (+1) 418-656-5580; Fax: (+1) 418-656-2324

Email: Bernard.Moulin@ift.ulaval.ca

Abstract

In this paper, we present a formalism for representing spatio-temporal changes and a method for map updates, based on the topological properties of the line Voronoi diagram. This new approach can overcome some of the problems with existing spatio-temporal models. We preserve the spatio-temporal changes in a data structure that combines events and corresponding state changes in topology. The spatio-temporal semantics of the map are also captured in this spatio-temporal data structure (the hierarchical Quad-Edge data structure). We can relate our hierarchical Quad-Edge data structure to event structures, which are essential for answering spatio-temporal queries.

1 Introduction

Nowadays, there is a growing demand from the user community, for a new type of GIS, that will be able to support temporal data, spatio-temporal queries (Proulx 1995) and interactive spatial updates (Van Oosterom 1993).

Several authors (Pequet and Wentz **1994**), (Bedard and al. 1996), (Langran 1992) proposed spatio-temporal data models based on extending existing GIS models, in order to include temporal information. The problem they are facing is that most of commercial GISs are closed systems, which cannot be extended or modified (Van Oosterom 1993). Therefore, the solutions they propose are based on a "dual architecture" (Van Oosterom 1993), which is composed of two subsystems: a commercial GIS and a relational DBMS. (Van Oosterom 1997) stated the problem of the complexity of maintenance of the proposed spatio-temporal data models.

There are several spatio-temporal functionalities that can not be handled with such hybrid models, such as retroactive map updates and the incorporation of temporal data without exact temporal information. (Frank 1994). Another problem with existing GISs, is that the semantics of map topology construction are lost. When a map is processed in batch mode,

its topology is built, and it is not possible to go back to previous states, nor to reuse the past map states, because the information about spatiotemporal objects and the operations that have been executed upon them is lost. This problem is better known under the term of “long transactions in GIS” (Newell and Batty 1994). The problem of long transaction is so acute (time consumption, database inconsistency problems, etc.), that the underlying problem is not well defined. The problem lies in the fact that “real topology” (Gold 1994) is not used: geometric algorithms can not be executed interactively upon spatial objects while maintaining topological relationships.

Van Oosterom proposed a data model in which both time and topology are consistently maintained in the database updates. His topological structure is based on the “CHAIN” method (Van Oosterom 1997), similar to the winged edge data structure. He also addresses the problems of long transactions in GIS, the loss of the transactional semantics and possible database inconsistency problems (Van Oosterom 1993, 1997). In his approach, the history of complex map objects can be obtained only by using spatial overlap queries, with respect to the given objects over time. Indeed, the model he proposed can be queried for historic versions only for simple object changes: it does not work for splits, joins nor more complicated spatial editing, unless spatial overlay is performed. Within his data structure, retroactive map updates are not implemented, due to possible consistency problems.

(Frank 1994) provides a theoretical framework for spatiotemporal reasoning based on the relative order of events. He proposed the integration of ordered event structures in a GIS. (Gold 1996) proposed a spatio-temporal model based on event ordering also, which responds instantly to map construction commands given by the user, changing the state of map objects and their spatial topology, and storing all the changes in topological states. In his model, the map topology states can be reconstructed at any time because all the operations on his structure are reversible.

In this paper, we present a formalism for representation of spatio-temporal changes in a dynamic Voronoi data structure and a method for map updates, based on the topological properties of the line Voronoi diagram (see (Gold 1996) and Figure 1). This new approach can overcome some of the problems that exist with the spatiotemporal models proposed by other authors (Bedard and al. 1996), (Langran 1992), (Van Oosterom 1993), (Pequet and Wentz 1994), such as retroactive map updates and spatio-temporal queries on complex spatial objects.

2 The dynamic spatio-temporal Voronoi data structure

The line Voronoi diagram is a tessellation of the space where each object (point or line segment) is assigned an influence zone (or Voronoi region), that is the set of points closer to that object than to any other object (see Figure 1). The boundaries between the regions of this tessellation form a net (the Voronoi diagram), whose dual graph (the Delaunay triangulation) stores the spatial adjacency (topology) relationships among objects. The underlying data structure used is the Quad-Edge data structure (Guibas and Stolfi 1985).

The Quad-Edge data structure was introduced by (Gui ?s and Stolfi 1985) as a primitive topological structure for the representation of any subdivision on a two-dimensional manifold. We use the Quad-Edge data structure for computing the line Voronoi diagram (Gold 1997), which is the base of the dynamic Voronoi data structure for points and line segments. The Quad-Edge data structure is the implementation of an edge algebra (Guibas and Stolfi 1985),

which is the mathematical structure used for traversing any subdivision. The Quad-Edge data structure represents a graph and its geometric dual (see Figure 2) and has the following specification:

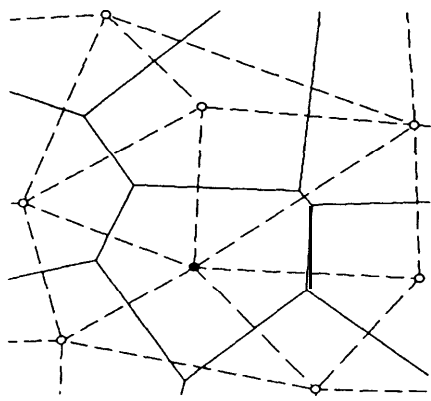


Figure 1: A Voronoi diagram

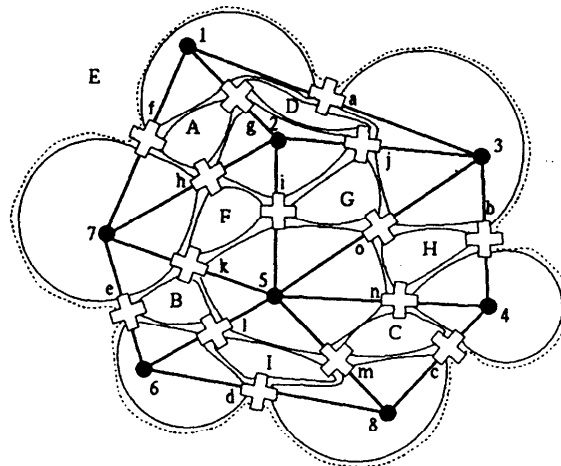


Figure 2: The corresponding Quad-Edge data structure

- For each quad-edge, there are:
 - two pointers to the next Quad-Edge in an anticlockwise order on the closed circuit around a vertex object;
 - two pointers to the next Quad-Edge in an anticlockwise order on the closed circuit around a face;
 - two pointers to vertex objects;
 - two pointers to faces.

The PAN graph (Gold 1988) of the Quad-Edge data structure gives a representation more suitable within GIS context (see Figure 3). In the context of the application of the Quad-Edge data structure to the computation of Voronoi diagrams, both a primal planar graph (the Voronoi diagram) and its dual graph (the Delaunay triangulation) are stored in the Quad-Edge data structure.

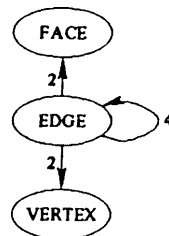


Figure 3: The PAN graph of the Quad-Edge data structure

3 Formalizing operations and corresponding changes in topology

The dynamic Voronoi spatial data model is based on an event-condition-action paradigm (Gold 1996)) which seems to provide many advantages over traditional GIS data models.

The main advantage of the dynamic Voronoi data structure is its dynamic, incremental and explicit topology, which allows one to automatically keep track of each event and change of map state.

3.1 The atomic actions on the dynamic Voronoi data structure

These map state changes are produced by map commands (Gold 1992), that are composed of atomic actions. Each atomic action in the map command executes the geometric algorithm for addition, deletion or change of map objects and corresponding Voronoi cells.

The **atomic actions** are:

- the **Split** action inserts a new point into the structure by splitting the nearest point from the pointed location into two points (see Figure 4);

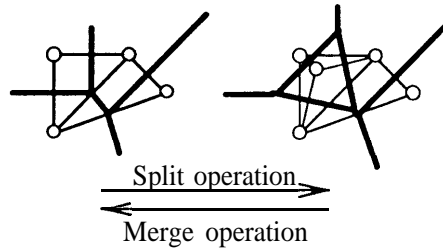


Figure 4: The topological changes due to the Split and Merge operations

- the Merge action deletes the selected point by merging it with its nearest neighbour (see Figure 4);
- the **Switch** action is performed when a point moves and a topological event occurs (i.e. the moving point enters or exits a circle circumscribed to a Delaunay triangle, see Figure 7), switching the common boundary of two adjacent triangles (see Figure 5);

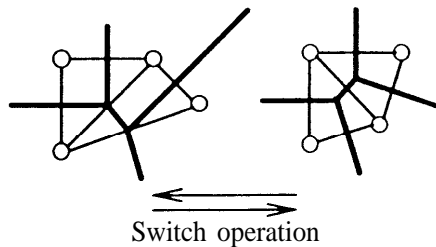


Figure 5: The topological changes due to the Switch operation

- the **Link** action adds a line segment' between the points obtained after a Split action (see Figure 6);

'A line segment is composed of two half-line segments, whose Voronoi regions are on each side of the line segment, having the line segment as common boundary (see Figure 6)

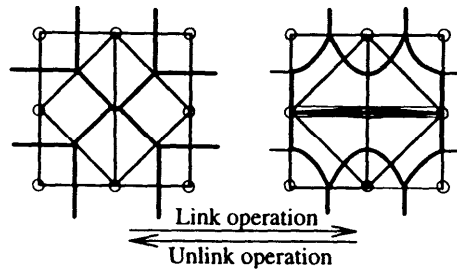


Figure 6: The topological changes due to the Link and Unlink operations

- the *Unlink* action removes the selected line segment (see Figure 6).

The map changes produced by the atomic actions on this data structure are the changes in the spatial adjacency relationships among spatial objects. Each *atomic action* produces different changes in spatial topology. The possible changes are:

- the triangle switches (topological events) changing the corresponding Voronoi edges (see Figure 7),

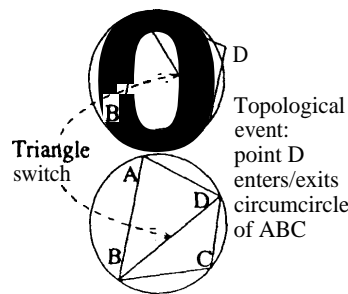


Figure 7: A topological event

- the creation of a new map object (point or line) and the corresponding appearance of its Voronoi region,
- the inactivation of a map object and the corresponding disappearance of its Voronoi region. Objects and spatial adjacency relationships are not removed, but inactivated, in order to be able to record all the history information.

Atomic Operation	Symbol	New Edges	Inactivated Edges
Split	S	6	3
Merge	M	3	6
Switch	N	1	1
Link	L	11	5
Unlink	U	5	11

Table1: The Atomic actions and their associated changes in topology

The topological changes in the map are represented by the numbers of newly created and inactivated spatial adjacency links (i.e. Voronoi edges or Quad-Edges), and are presented for each atomic action in Table 1. Each atomic action is uniquely characterized by the numbers of new Quad-Edge (or Voronoi) edges and inactivated Quad-Edge (or Voronoi)

edges (see Table 1). This means that from changes in topology we can determine which atomic action was applied, and vice versa. In other words, the actions on the data structure have a deterministic behaviour. More precisely, we can say that the set of atomic actions is naturally isomorphic to the set of the number of new edges as well as to the set of the number of inactivated edges.

3.2 The map construction commands

All the map construction commands (Gold 1992) of this dynamic Voronoi data structure are complex operations composed of atomic actions (illustrated in Figures 4, 5 and 6). The composition of atomic actions into map commands is provided by syntactic rules. The meaning of the word “syntax” is based on the theory of formal language and grammar (Hopcroft and Ullman 1979). In the theory of formal languages, the semantics of the basic operations that can be applied on the set of objects is described by a grammar. A grammar provides a set of rules, known as production rules, showing how the sequence of atomic actions will be applied to the elemental map object (currently a point or a line segment). Thus, the update of the Voronoi data structure given by map commands can be interpreted as the execution of production rules or rewrite rules (Jacob 1996). Rewriting is a useful technique for defining complex objects by successively replacing parts of elemental map objects using a set of rewrite rules or production rules (Jacob 1996).

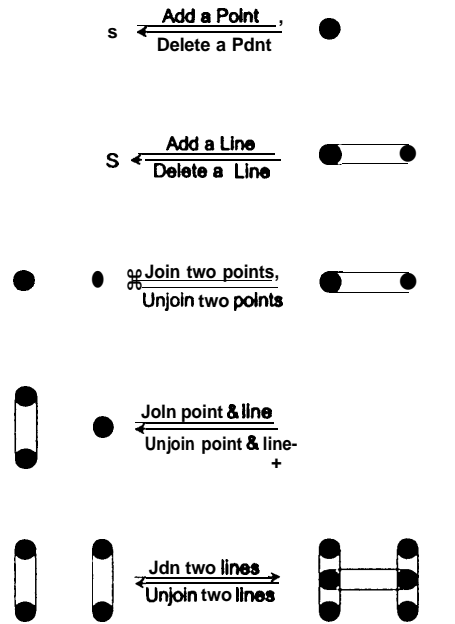


Figure 8: The map commands. (S is the starting symbol in the production rule).

The *map construction commands* are illustrated in Figure 8. On the left side of Figure 8, we can see the map objects on which the map command will be applied, and on the right side, the map objects that have been rewritten. In the graphical illustration of map commands, the topological part of the model is left out for better understanding of the general principle, and also the line-line collisions and their effects are not shown. The decomposition of map commands into sequences of atomic actions is given in Table 2.

Map construction command	Decomposition (the terms in parentheses appear at each line-line collision, $i =$ collision index, $i \in \{1, \dots, c\}$, $c =$ number of collisions; t, t_x denote numbers of topological events)
Move a Point	N^t
Add a Point	SN^t
Delete a Point	$N^t M$
Add a line	$SN^{t_1} SLN^{t_2} (SLN^{t_{2i+1}} MSLN^{t_{2i+2}})$
Delete a line	$N^{t_{2i+2}} UMSN^{t_{2i+1}} UM N^{t_2} UMN^{t_1} M$
Join 2 points	$SLN^{t_1} SLN^{t_2} MSLN^{t_{2i+1}} M$
Unjoin 2 points	$(N^{t_{2i+1}} UMSN^{t_{2i}} UM N^{t_1} UM$
Join pt & line	$SLN^{t_1} (SLN^{t_{2i+1}} MSLN^{t_{2i+2}}) SLN^{t_2} M$
Unjoin pt & line	$SN^{t_2} UM N^{t_{2i+2}} UMSN^{t_{2i+1}} UM N^{t_1} UM$
Join 2 lines	$SLN^{t_1} SLN^{t_2} SLN^{t_{2i+2}} MSLN^{t_{2i+3}} SLN^{t_3} M$
Unjoin 2 lines	$SN^{t_3} UM N^{t_{2i+3}} UMSN^{t_{2i+2}} UM N^{t_2} UMN^{t_1} UM$

Table2: The map commands and their decomposition into atomic actions

The corresponding changes in topology are described in Tables 3 and 4.

Map construction command	Inactivated Voronoi region	Newly created Voronoi region
	$c =$ number of line intersections	
Move a Point	0	0
Add a Point	0	1
Delete a Point	1	0
Add a line	0	$4 + 5c$
Delete a line	$4 + 5c$	0
Join 2 points	0	$2 + 5c$
Unjoin 2 points	$2 + 5c$	0
Join point & line	0	$5 + 5c$
Unjoin point & line	$5 + 5c$	0
Join 2 lines	0	$8 + 5c$
Unjoin 2 lines	$8 + 5c$	0

Table3: The changes induced by map commands in Voronoi regions

In Table 3, the corresponding “state changes” for each map command are represented as the number of newly created and inactivated Voronoi cells. We can observe that except for the “Move a Point” map command, the formulas for the numbers of inactivated Voronoi regions and newly created Voronoi regions generate couples of numbers, which pertain to different couples of residual classes modulo the prime number 5. This corresponding couple of numbers of inactivated Voronoi regions and newly created ones for the “Move a point” map command is $(0, 0)$, and it does not pertain to any other corresponding couple of numbers (of the other map commands). Therefore, we have an isomorphism between the set of map commands and the set of couples of numbers of inactivated Voronoi regions and of newly created ones. Moreover, this isomorphism gives rise to a discrimination of map commands, that allows one to determine the number of line-line collisions that occurred in a given map

update. The discrimination just described allows us to determine which map commands were applied just by knowing the predecessor and successor map topology states expressed in the number of newly created Voronoi regions and of inactivated ones (see Table 3).

In Table 4, the corresponding “state changes” for each map command are represented as the difference between the predecessor state and the successor state, expressed as the difference between the numbers of inactivated and newly created Voronoi edges. These state changes take into account the number c of intersections (between the newly created line segment and any existing objects) that occurred in the execution of the map command. When a moving point of a newly added line segment enters the last circumcircle of the line segment, the mutual splitting of line segments occurs. This operation is visible as a replicating sequence in Table 2.

The map commands can be recognized by the changes between the predecessor and successor map topology states, expressed by the difference between the numbers of newly created Voronoi edges and of inactivated Voronoi edges (see Table 4), and vice versa. Indeed, the numbers generated by the formulas of differences (in the third column) pertain to sets (in the fourth column) which are mutually exclusive. Moreover, this discrimination allows us to determine the number of line-line collisions, and then the number of topological events that occurred in a map update (see example treated below). Mathematically speaking, there exists an isomorphism between the set of map commands and the set of sets (in the fourth column) of possible corresponding changes between the predecessor and successor map topology states.

Map construction command	New Voronoi Edges	Inactivated Voronoi Edges	Difference (New - Inactivated)	Discrimination (set of the numbers corresponding to the difference New - Inactivated Voronoi Edges)
	t = total number of topological events			
Move a Point	t	t	0	$\{0\}$
Add a Point	$t + 6$	$t + 3$	3	$\{3\}$
Delete a Point	$t + 3$	$t + 6$	- 3	$\{-3\}$
Add a line	$t + 23 + 37c$	$t + 11 + 22c$	$12 + 15c$	$\{z \in \mathbb{Z}, z \equiv 12 \pmod{15} \wedge z \geq 12\}$
Delete a line	$t + 11 + 22c$	$t + 23 + 37c$	$-12 - 15c$	$\{z \in \mathbb{Z}, z \equiv 3 \pmod{15} \wedge z \leq -12\}$
Join 2 points	$t + 20 + 37c$	$t + 14 + 22c$	$6 + 15c$	$\{z \in \mathbb{Z}, z \equiv 6 \pmod{15} \wedge z \geq 6\}$
Unjoin 2 points	$t + 14 + 22c$	$t + 20 + 37c$	$-6 - 15c$	$\{z \in \mathbb{Z}, z \equiv 9 \pmod{15} \wedge z \leq -6\}$
Join pt & line	$t + 37 + 37c$	$t + 22 + 22c$	$15 + 15c$	$\{z \in \mathbb{Z}, z \equiv 0 \pmod{15} \wedge z \geq 15\}$
Unjoin pt & line	$t + 22 + 22c$	$t + 37 + 37c$	$-15 - 15c$	$\{z \in \mathbb{Z}, z \equiv 0 \pmod{15} \wedge z \leq -15\}$
Join 2 lines	$t + 54 + 37c$	$t + 30 + 22c$	$24 + 15c$	$\{z \in \mathbb{Z}, z \equiv 9 \pmod{15} \wedge z \geq 24\}$
Unjoin 2 lines	$t + 30 + 22c$	$t + 54 + 37c$	$-24 - 15c$	$\{z \in \mathbb{Z}, z \equiv 6 \pmod{15} \wedge z \leq -24\}$

Table 4: The discrimination of map commands by means of their changes in topology.

In the following example, we will see the changes in a topology induced by map update. If we have nine new Voronoi regions appeared in the map update level, from that result, we can determine (from Table 3) the number of line-line collisions that occurred in that map update: $9 \equiv 4 \pmod{5}$. This implies that the update was “Add a line” and therefore we get the final result: $4 + 5c = 9 \Rightarrow c = 1$. We can see that the difference in the numbers of newly created and of inactivated Voronoi edges is 27. From that result ($27 \equiv 12 \pmod{15}$), $12 + 15c = 27 \Rightarrow c = 1$), we arrive (see Table 4) at the same conclusion: the map update corresponds to the addition

of a new line segment intersecting one existing line segment. Knowing the numbers of new Voronoi edges and of inactivated Voronoi edges, we can determine the exact number of topological events ($t + 23 + 37$ equals the number of new Voronoi or Quad-Edge edges, see Table 4). Therefore we know that the decomposition of the map command in atomic actions is of the following form: $SN^{t_1}SLN^{t_2}(SLN^{t_3}MSLN^{t_4})$ where $t_1 + t_2 + t_3 + t_4$ is the known total number of topological events.

4 Map updates and map history

The map construction commands have a direct translation into the spatio-temporal topology, as the hierarchy of map objects and their corresponding Voronoi cells that have been added, moved, or inactivated through time. Each application of map commands is timestamped and represented as an update level in our spatio-temporal structure. The execution traces (for more detailed explanations see (Mazurkiewicz 1989)) give us a formalism to represent map history at a more abstract level.

The history of the map is maintained as a hierarchy of map objects and their ancestral dependency relationships (Luck and Lück 1976), and the corresponding topological changes in the Quad-Edge data structure. A dependency relationship occurs when one atomic action uses the results of another: e.g. a Link action always follows a Split action, and it uses the point that has been split and the point from which it has been split (see section 3). Ancestral dependency relationships are defined as timed division hierarchies of elemental map objects and their corresponding Voronoi regions, which are ordered by the dependency of one atomic action upon another. In the event-driven systems, such dependency relationships are isomorphic to event structures (for more details see (Winskel 1989)). In the following example, we can see two levels corresponding to the map update shown in Figure 9, and their dependency relationships (see Figures 10 and 11).

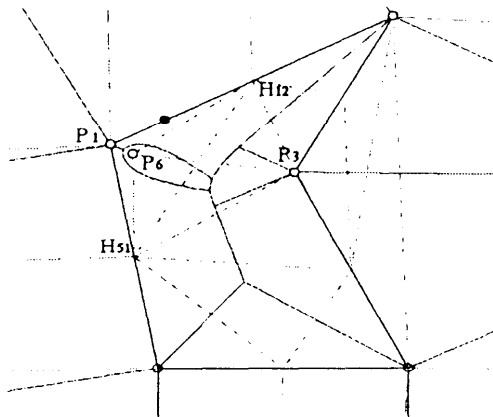


Figure 9: A map update: the addition of point P_6

Thus the spatiotemporal model combines events and corresponding state changes in topology. We can say that the hierarchical Quad-Edge data structure is equivalent to an event structure. Event structures together with “semantics of change” are essential for answering spatiotemporal queries. The temporal ordering in the hierarchical Quad-edge data structure is maintained through history links (see Figure 10 and 11): parent-child links (corresponding

to the origin/destination of a Split/Merge or Link/Unlink action), and modification link (representing a change in the Voronoi region of an object).

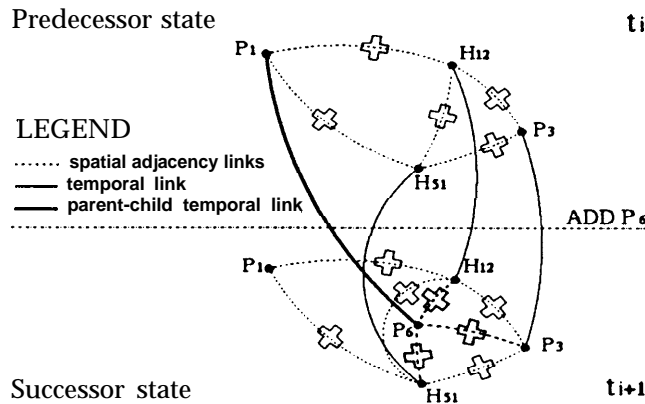


Figure 10: Two update levels of the Quad-Edge data structure and their dependency links

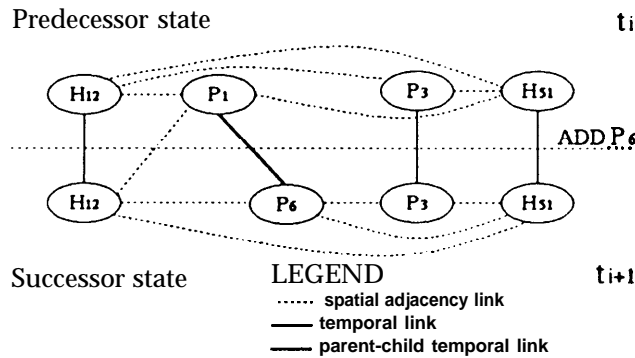


Figure 11: Two update levels of the spatio-temporal structure and their dependency links

The temporal ordering of past map events as well as states provides a suitable model for integration of Allen's temporal algebra, that is needed for reasoning about spatial objects and their temporal relationships. Furthermore, the query processing can be optimized by running the query only at the update levels of interest. The topological state changes given as the difference in the number of newly created and inactivated Voronoi edges between two update levels contain the information about the atomic actions applied at this level, and about the number of line-line intersections and of topological events that occurred during the execution of that **map** command. By exploiting this information we can decide whether to run a query at that level of update or not.

Further benefits of this method include retroactive map updates. Even though there are some other methods, such as transactions logging, to keep track of the operations that have been applied, these methods have several drawbacks in the case of retroactive map updates. For example, for the map updates in the past, the "log file" needs to be updated with the exact sequence of atomic actions, while in our approach the "semantics of change" is simply maintained as a difference in the number of Voronoi edges between two update levels.

5 Conclusions

In this paper, we have presented a new spatio-temporal model based on a dynamic Voronoi data structure for points and line segments. The approach is based on local changes in topology induced by spatio-temporal map updates. These map updates are performed through map construction commands that are composed of atomic actions on the dynamic Voronoi data structure. Formalizing the operations needed for constructing a Voronoi diagram for points and line segments and corresponding topological changes provides us with a tool for modelling spatial and temporal semantics.

We have shown in this paper that the behaviour of the basic map operations is deterministic, and well defined in terms of topology changes. The recognition of map commands from the corresponding changes in spatial topology allowed us to extend our data structure towards the hierarchical Quad-Edge data structure that can be used for spatiotemporal data representation.

We can say that the hierarchical Quad-Edge data structure is equivalent to an event structure. The temporal ordering of past map events as well as states in a hierarchical Quad-Edge data structure provide a suitable model for integration of Allen's temporal algebra, that is needed for reasoning spatial objects and their temporal relationships (Frank 1994). Furthermore, the spatio-temporal structure that combines events and past map states is essential to answer the queries about the map changes over space and time.

There are significant implications of the temporal ordering of map construction events in the Voronoi data structure. The model has an implicit time ordering of events, visible through changes in topology. The dynamic Voronoi spatial data structure can support temporal data without precise temporal information. The changes in the spatiotemporal data structure capture the temporal and spatial semantics.

This formal model of spatiotemporal change representation is currently applied to retroactive spat&temporal map updates and visualization of map evolution. It offers new possibilities in the domains of temporal GIS, visualization, simulation, real-time, and robotics.

Acknowledgments

The first and third authors have received the financial support from a grant of the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Association des Industries Forestières du Quebec (AIFQ).

References

- Bedard, Y., Caron, C., Maamar, Z., Moulin, B., and Valliere, **D.**, 1996, **Adapting data models for the design of spatio-temporal databases**, *Comput., Environ. and Urban Systems*, **Vol. 20, No. 1**, pp. 19-41.
- Frank, A.U., 1994, Qualitative temporal reasoning in GIS-ordered time scales, *Sixth International Symposium on Spatial Data Handling SDH'94, Edinburgh, Scotland, UK, in Advances in GIS Research, Proceedings, Vol. 1*, pp. 410-430.
- Gold, C.M., 1997, The Global GIS, *Proceedings of the International Workshop on Dynamic and Multi-Dimensional GIS, Hong-Kong, China, 12* pp.
- Gold, C.M., 1996, An Event-Driven Approach to SpatioTemporal Mapping, *Spatio- Temporal special issue, Geomatica, Vol. 50*, pp. 415-424.

- Gold, C. M., 1994, Three approaches to automated topology, and how computational geometry helps, ***Proceedings, Sixth International Seminar on Spatial Data Handling, Edinburgh***, pp. 145-158.
- Gold, C.M., 1992, An object-based dynamic spatial data model, and its applications in the development of a user-friendly digitizing system, ***Proceedings, Fifth International Symposium on Spatial Data Handling, Charleston***, pp. 495-504.
- Gold, C.M., 1988, PAN graphs - An aid to GIS analysis, ***International Journal of Geographical Information Systems, 1988, Vol. 2, No. 1***, pp. 29-41.
- Guibas, L., and Stolfi, J., 1985, Primitives for the Manipulation of General Subdivisions and the Computation of Voronoi Diagrams, ***ACM Transactions on Graphics, Vol. 4, No. 2***, pp. 74-123.
- Hopcroft, J., and Ullman, J., 1979, Introduction to Automata Theory, ***Addison- Wesley, Reading, Mass.***
- Jacob, C., 1996, Evolving Evolution Programs: Genetic Programming and L-Systems, ***Genetic Programming, Proceedings of the First Annual Conference 1996, The MIT Press, Cambridge, Mass.***, pp. 107-115.
- Langran, G., 1992, Time in Geographic Information Systems, ***Technical Issues in Geographic Information Systems, Taylor & Francis, London, UK.***
- Luck, H.B., and Luck, J., 1976, Cell number and cell size in filamentous organisms in relation to ancestrally and positionally dependent generation times, in ***Automata, Languages, Development, North-Holland, Amsterdam, Netherlands***, pp. 109-124.
- Mazurkiewicz, A., 1989, Basic Notions of Trace Theory, ***Proceedings of REX Workshop "Linear Time, Branching Time, and Partial Order in Logics and Models for Concurrency", Springer LNCS 354***, pp. 285-363.
- Newell, R., and Batty, M., 1994, GIS databases are different, ***AM/FM'94***, pp. 279-288.
- Pequet, D., and Wentz, E., 1994, An Approach for Time-Based Spatial Analysis of Spatio-Temporal Data, ***Advances in GIS Research, Proceedings 1***, pp. 489-504.
- Proulx, M-J., 1995, Développement d'un nouveau langage d'interrogation de bases de données spatio-temporelles, ***These de Maitrise, Département des Sciences Géomatiques, Université' Laval, Quebec.***
- Van Oosterom, P., 1997, Maintaining Consistent Topology including Historical Data in a Large Spatial Database, in ***1997 ACSM/ASPRS Annual Convention & Exposition, Seattle, Washington, 1997, Auto-Car-to 13, Vol. 5***, pp. 327-336.
- Van Oosterom, P.J.M., 1993, Reactive Data Structures for Geographic Information Systems, ***Oxford University, Bookcraft Ltd.***
- Winkel, G., 1989, An introduction to event structures, ***Proceedings of REX Workshop "Linear Time, Branching Time, and Partial Order in Logics and Models for Concurrency", Springer LNCS 354***, pp. 365-397.