1998 International Computer Symposium
Workshop on Artificial Intelligence
December 17-19, 1998, N.C.K.U., Tainan, Taiwan, R.O.C.

Interval Algebra for Spatio-Temporal Relation Reasoning

Timothy K. Shih, *Anthony Y. Chang and
Ken C. Wang
Department of Computer Science and Information Engineering
Tamkang University
Tamsui,Taiwan
R.O.C.
*e-mail: g5190315@tkgis.tku.edu.tw

Abstract

Constraint satisfaction techniques play an important role in current computer science. Many difficult problems involving search from areas such as machine vision, scheduling, graph algorithms, machine design, and manufacturing can be considered to be the cases of the constraint satisfaction problem. In this paper, we construct a Infinite Temporal Interval Group for temporal constraint propagation. This interval algebra is also extended for spatial constraint reasoning. A temporal transitive closure table is derived, and we propagate the time constraints of arbitrary two objects across long distances n by linear time. A set of algorithms is proposed to manage spatio-temporal knowledge. The algorithms are extended for time-based media in an n-dimensional space. Possible conflicts in the user specification are detected and eliminated. We derive and prove many properties of spatio-temporal relations.

1 Introduction

Relations among temporal intervals can be used to model time dependent objects. The extension of temporal relation results in many researches related to the spatio-temporal modeling of symbolic objects.

We use the qualitative representation to composite the spatio-temporal relations, since humans are not very good at determining exact object lengths, volumes, etc., whereas they can easily perform context-dependent comparisons. For example, given the relations between objects X and Y, and between objects Y and Z, we want to know the relation between X and Z. If specify "X is left_below to Y" and "Y is left_close to Z", we can derived the relations implies "X is left_below to Z".

The importance of knowledge underlying temporal interval relations was found in many disciplines. As pointed out in [1], researchers of artificial intelligence, linguistics, and information science use temporal intervals as a time model for knowledge analysis. For instance, in a robot planning program, the outside world is constantly changed according to a robot's actions. The notion of "number three box is on the left of number two box" is true only within a spatial interval relation (assuming that the robot is given a command to move a number of boxes to a destination). As other examples, linguisticians use temporal intervals to process queries such as " who is the president of company X within the period of January 31, 1995 to November 15, 1997?" The work discussed in [1] analyzes the relations among temporal intervals. This research contribution has been used in many temporal modeling of multimedia systems including ours [2, 10, 11]. However, the work [1] only states temporal interval relations. No spatial relation was discussed. We found that these relations can be generalized for spatial modeling.

Many researchers propose temporal/spatial modeling of object representations. Seven generalized n-ary relations were used to describe the temporal relations among n objects. The authors also defined spatial events in terms of these n-ary relations. Temporal events were then specified in terms of these spatial events. However, there was no discussion of the conflict situation among relations. Based on Allen's temporal interval relations, a set of directional and topological spatial relations was addressed in [7]. The authors also provided a set of spatial inference rules

for automatic deduction. A methodology for spatial and temporal object composition was proposed in [8]. A set of n-ary temporal relations with their temporal constraints were discussed in [9], which is an early result of the work addressed in [3]. The temporal model of reverse play and partial interval evaluation for midpoint suspension and resumption were also discussed. Algorithms for accessing objects in a database were presented. However, no discussion of the conflict situation among relations was found. Efficient indexing schemes based on the R-tree spatial data structure were proposed in [12]. The mechanism handles 1-D and 2-D objects, as well as 3-D objects, which treat a time line as the third axis. The discussion in [4] identified various temporal interaction forms and discussed their temporal dependencies. The work also integrated some interaction forms into a temporal model.

We have surveyed many researches related to the spatio-temporal semantics of constraint reasoning and object modeling. We found that, the use of spatio-temporal relations can serve as a reasonable semantic tool for the underlying representation of objects in many applications. Composite objects can have arbitrary timing relationships. These might be specified to achieve some particular visual effect of sequence. In this paper, we extend the interval relations by means of a complete analysis of all temporal relation domains and constraint reasoning. These domains are also extended for spatial computation.

2 Spatio-Temporal Symbolic Constraint Propagation

According to the interval temporal relations introduced in [1], there are 13 relations ({e, <, >, m, mi, d, di, o, oi, s, si, f, fi}) between two temporal intervals. Abbreviations of these relations between are also due to [1]. We describe the symbolic constraint propagation. The general idea is to use the existing information about the relations among time intervals or instants to derive the composition relations.

The composition may result in a *multiple derivation*. For example, if "X before Y " and "Y during Z " , the composed relation for X and Z could be "before", "overlaps", "meets", "during", or "starts". If the composed relation could be any one of some relations, these derived relations are called reasonable relations in our discussion.

In some cases, relation compositions may result in a *conflict specification* due to the user specification or involved events synchronously. For example, if specifications " X before Y " ," Y before Z " , and " X after Z " are declared by the user, there exists a conflict between X and Z. When the specific relations are not found in derived reasonable set, the specification may cause conflicts.

We analyze the domain of interval temporal relations and use an directed graph to compute the relations of all possibilities.

Definition : An *user edge* denotes a relation between a pair of objects defined by the user. The relation may be reasonable or non-reasonable. ■

Definition : A *derived edge* holds a non-empty set of reasonable relations derived by our algorithm. The relation of the two objects connected by the derived edge can be any reasonable relation in the set. ■

For an arbitrary number of objects, some of the relations are specified by the user while others are derived. If there exists a cycle in the directed relation graph, a conflict derivation may occur.

We also extent the temporal relations to qualitative spatial knowledge. The temporal operators can split the images orthogonally to the coordinate axis. Such splitting, generally called *cuttings*, play a fundamental role in some approaches to *symbolic projecting* for image information retrieval.

Using the extended spatial relations can deduct and reduct spatial relationships. If every picture that satisfies all the spatial relations, the set of temporal knowledge can be derived and maintained. For example, the spatial relations (A left_below B) and (B left_overlap C), we can derived the relations implies (A left_below C).

3 The Finite Temporal Relations Group

Based on Allen's work, compositions of three or more relations are computed using algorithms based on set operations, such as set union and intersection. These set operations are expensive. We argue that, an extension of *Table13* (Allen's Table [1]), named *Table29*, can be calculated. The compositions of three or more relations can be obtained directly from our table. Based on the *table29*, we found many properties of spatio-temporal relations and proved the temporal relation composition is a algebraic *group*.

Firstly, we define some terminology. An interval has a *name*, which is an ASCII string. The term $P(X)$ represents a power set of objects of type X. The *13Rel* is the domain of the 13 interval relations. Inverse relations are also defined. The *29Relset* is a domain of relation sets. Each element in *29Relset* contains one or more interval relations which represent the possible

composition results between two intervals.

Name==*P* (*string*)
13Rel == { <, >, *d, di, o, oi, m, mi, s, si, f, fi, e* }
29Relset ⊂ *P* (*13Rel*)
$\forall rs \in 29RelSet \bullet rs^{-1} = \{ r^{-1} \in 13Rel \mid r \in rs \}$
TemporalTuple==*Name* × *29RelSet* × *Name*
$\forall tt : TemporalTuple \bullet$
$tt = (A, rs, B) \Leftrightarrow tt^{-1} = (B , rs^{-1} , A)$
$\bullet : TemporalTuple \times TemporalTuple \rightarrow$
TemporalTuple

The following table gives a summary of the 29 relation sets which contain all possible composition results:

Table 1: The 29 Interval Relation Sets

ID	Relation Sets
1	{ < }
2	{ > }
3	{ d }
4	{ di }
5	{ o }
6	{ oi }
7	{ m }
8	{ mi }
9	{ s }
10	{ si }
11	{ f }
12	{ fi }
13	{ e }*
14	{ o, di, fi }
15	{ oi, d, f }
16	{ o, d ,s }
17	{ oi, di, si }
18	{ <, o, m }
19	{ >, oi, mi }
20	{ f, fi, e }*
21	{ s, si, e }*
22	{ <, o, m, d, s }
23	{ >, oi, mi, di, si }
24	{ <, o, m, di, fi }
25	{ > , oi, mi, d, f }
26	{ o, oi, d, di, s, si, f, fi, e }*
27	{ <, m, d, di, o, oi, f, fi, s, si, e }
28	{ > , mi, di , d, oi, o, fi, f , si, s, e }
29	{ < , > ,m, mi, di , d, oi, o, fi, f , si, s, e }*

Table29 is generated by our program implemented based on the following algorithms.

Algorithm : *Relcomp*
Input : $rs_1 \in 29RelSet, rs_2 \in 29RelSet$
Output : $rs \in 29RelSet$
Preconditions : true
Postconditions : true
Steps :
1. $rs = \cup \forall r_1 \in rs_1 , \forall r_2 \in rs_2 \bullet (r_1 , r_2) \in rs_1 \times rs_2$
Table13 (r_1 , r_2)

Algorithm : *ComputeTable29*
Input : *Table13*
Output : *Table29*
Preconditions : true
Postconditions : relation composition is closed under *I*
Steps :
1. Construct a set of 13 atomic sets from the 13 relations, assuming that this set is called *I*, which is an index set for table look up.
2. Let $Table29(i ,j) = Table13(i, j), i \in I, j \in I$
3. $\forall$ $Table29(i, j), i \in I , j \in I$, do
   3.1: if $k = Table29(i,j) \notin I$ then
      3.1.1 : $I = I \cup Table29(i, j)$
      3.1.2 : $\forall m \in I$,do
         3.1.2.1 $Table29(k, m) = Relcomp(k,m)$
         3.1.2.2 $Table29(m, k) = Relcomp(m,k)$

Algorithm *ComputeTable29* adds new relation sets computed by *RelcCmp* to the index set *I*, and computes the new elements of *Talble29*. There are
$C(13, 0) + C(13, 1) + C(13, 2) + ... + C(13, 13) = 2^{13}$
possible elements of *I*. However, from the computation of algorithm *ComputeTable29* , the cardinality of *I* results in 29. Based on this result, we argue that, for an arbitrary pair of temporal intervals, the possible relations between them must be an element of set *I*.

Using *Table29*, when composing temporal relations, the set union operation is replaced by a table look up operation. Therefore, the time complexity of relation composition is reduced. The cost of memory used in *Table29* is tolerable.

Assuming that $a = (A, rs, B)$, where *A*, and *B* are interval names, and rs is a temporal relation set. We want to find a rs^{-1} for each *rs*. The following table shows the inverse relation sets rs^{-1} for each rs :

Table 2: Inverse Relation Sets

rs	rs^{-1}	*rs*	rs^{-1}
1	2	18	19
3	4	20	20
5	6	21	21
7	8	22	23
9	10	24	25
11	12	26	26
13	13	27	28
14	15	29	29
16	17		

There are five relation sets which are the inverse of themselves (i.e., the one marked with a subscript "*" in the 29 relation sets). Since each relation set has its inverse, for an arbitrary $a = (A, rs , B)$, we can always find $b = (B, rs^{-1}, A) \in S$, such that $a \circ b = b \circ a =$ (A,{e},A) ■

Table 3: The Temporal Transitive Closure Table

o	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
01	01	29	22	01	01	02	01	22	01	01	22	01	01	01	22	22	22	01	29	22	01	22	29	01	29	22	22	29	29
02	29	02	25	02	25	02	25	02	25	02	02	02	02	25	25	25	02	29	02	02	25	29	02	29	25	25	29	25	29
03	01	02	03	29	22	25	01	02	03	25	03	22	03	29	25	22	29	22	25	22	25	22	29	29	25	29	29	29	29
04	24	23	26	04	14	17	14	17	14	04	17	04	04	14	26	26	17	24	23	17	14	27	23	24	28	26	27	28	29
05	01	23	16	24	18	26	01	17	05	14	16	18	05	24	26	22	27	18	28	22	14	22	29	24	28	27	27	29	29
06	24	02	15	23	26	19	14	02	15	19	06	17	06	28	25	26	23	27	19	17	25	27	23	29	25	28	29	28	29
07	01	23	16	01	01	16	01	20	07	07	16	01	07	01	16	22	22	01	28	22	07	22	29	01	28	22	22	29	29
08	24	02	15	02	15	02	21	02	15	02	08	08	08	25	25	15	02	27	02	08	25	27	02	29	25	25	29	25	29
09	01	02	03	24	18	15	01	08	09	21	03	18	09	24	15	22	27	18	25	22	21	22	29	24	25	27	27	29	29
10	24	02	15	04	14	06	14	08	21	10	06	04	10	14	15	26	17	24	19	17	21	27	23	24	25	26	27	28	29
11	01	02	03	23	16	19	07	02	03	19	11	20	11	28	25	16	23	22	19	20	25	22	23	29	25	28	29	28	29
12	01	23	16	04	05	17	07	17	05	04	20	12	12	14	26	16	17	18	23	20	14	22	23	24	28	26	27	28	29
13	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
14	24	23	26	24	24	26	24	17	14	14	26	24	14	24	26	27	27	24	28	27	14	27	29	24	28	27	27	29	29
15	24	02	15	29	27	25	24	02	15	25	15	27	15	29	25	27	29	27	25	27	25	27	29	29	25	29	29	29	29
16	01	23	16	29	22	28	01	23	16	28	16	22	16	29	28	22	29	22	28	22	28	22	29	29	28	29	29	29	29
17	24	23	26	23	26	23	14	23	26	23	17	17	17	28	28	26	23	27	23	17	28	27	23	29	28	28	29	28	29
18	01	29	22	24	18	27	01	27	18	24	22	18	18	24	27	22	27	18	29	22	24	22	29	24	29	27	27	29	29
19	29	02	25	23	28	19	28	02	25	19	19	23	19	28	25	28	23	29	19	23	25	29	23	29	25	28	29	28	29
20	01	23	16	23	16	23	07	23	16	23	20	20	20	28	28	16	23	22	23	20	28	22	23	29	28	28	29	28	29
21	24	02	15	24	24	15	24	08	21	21	15	24	21	24	15	27	27	24	25	27	21	27	29	24	25	27	27	29	29
22	01	29	22	29	22	29	01	29	22	29	22	22	22	29	29	22	29	22	29	22	29	22	29	29	29	29	29	29	29
23	29	23	28	23	28	23	28	23	28	23	23	23	23	28	28	28	23	29	23	23	28	29	23	29	28	28	29	28	29
24	24	29	27	24	24	27	24	27	24	24	27	24	24	24	27	27	27	24	29	27	24	27	29	24	29	27	27	29	29
25	29	02	25	29	29	25	29	02	25	25	25	29	25	29	25	29	29	29	25	29	25	29	29	29	25	29	29	29	29
26	24	23	26	29	27	28	24	23	26	28	26	27	26	29	28	27	29	27	28	27	28	27	29	29	28	29	29	29	29
27	24	29	27	29	27	29	24	29	27	29	27	27	27	29	29	29	27	29	27	29	27	29	29	29	29	29	29	29	29
28	29	23	28	29	29	28	29	23	28	28	28	29	28	29	28	29	29	29	28	29	28	29	29	29	28	29	29	29	29
29	29	29	29	29	29	29	29	29	29	29	29	29	29	29	29	29	29	29	29	29	29	29	29	29	29	29	29	29	29

4 Maintaining Time Constraints

Based on *Table29*, we propose a set of algorithms, using a directed graph, for fast temporal relation compositions. These algorithms can be used to compute the binary relation between an arbitrary pair of intervals. User edge conflicts are eliminated and derived edges and cycles without conflict are added. This conflict elimination is achieved by invoking the *EliminateConflict* algorithm. Suppose G is a graph of the reduce relation domain, and GV and GE are the vertex set and edge set of G, respectively. The algorithm computes derived edges based on user edges.

Algorithm : *ComputeRD1*
Input : $G = (GV, GE)$
Output: $K_n = (K_nV, K_nE)$
Preconditions : true
Postconditions : $GV = K_nV \wedge GE \setminus UE \cup UE' \subseteq K_nE$
Steps :
1 : G = *EliminateConflicts* (G)
2 : $K_n = G \wedge pl = 2$
3 : repeat until $|K_nE| = |K_nV| * (|K_nV| - 1) / 2$
3.1 : for each $e = (a, b) \wedge e \notin K_nE \wedge a \in K_nV \wedge b \in K_nV$ •
there is a path of user edges from a to b, with path length = pl
3.2 : suppose $((n_1, n_2), (n_2, n_3), \ldots, (n_{k-1}, n_k))$ is a path with $a = n \wedge b = n \wedge k = pl + 1$
3.3 : set $e.rs$ = *Table29* $((a, n_{k-1}).rs, (n_{k-1}, b).rs)$
3.4 : $K_nE = K_nE \cup \{ e \}$
3.5 : $pl = pl + 1$

The first algorithm, *ComputeRD1*, starts from taking each path of user edges of length 2, and computes a derived edge from that path. The insertion of edge $e = (a, b)$ results a cycle, but no conflict. The reasonable set of edge e (i.e., $e.rs$) is computed from two edges, (a, n_{k-1}) and (n_{k-1}, b), which are user edges or derived edges. Since we increase the path length, pl, of the path of user edges one by one. The derived edge (a, n_{k-1}) must have been computed in a previous interaction. The algorithm repeats until all edges are added to the complete graph K_n.

Algorithm : *EliminateConflicts*

Input : $G = (GV, GE)$
Output : $G' = (G'V, G'E)$
Preconditions : G contains only user edges $\wedge$ $G' = G$
Postconditions : $G' = G$, but the reasonable sets of edges in G' may be changed.
Steps :
1. for each $P = ((n_1, n_2), (n_2, n_3),..., (n_{k-1}, n_k))$ in G' with $n_1 = n_k \quad \wedge \quad k>3$
   1.1 : for each i, $1 \le i \le k$-2
      1.1.1 : set $(n_i, n_{i+2}).rs = Table29 ((n_i, n_{i+1}).rs, (n_{i+1}, n_{i+2}).rs)$
   1.2 : $rs = Table29 ((n_k, n_{k-2}).rs, (n_{k-2}, n_{k-1}).rs)$
   1.3 : if $(n_k, n_{k-1}).r \notin rs$ then
      1.3.1 : ask user to choose a $r' \in rs$
      1.3.2 : set $(n_k, n_{k-1}).r = r'$

Considering the five user edges, the algorithm computes derived edges until the last edge is added to K_n :

User edges :
(A, B) = { < } = [1]
(B, C) = { m } = [7]
(C, D) = { d } = [3]
(C, E) = { s } = [9]
(F, D) = { < } = [1]

Derivation based on user edges:

1. Path Length = 2
   $(A, C) = (A, B) \circ (B, C) = [1] \circ [7] = [1] = \{ < \}$
   $(B, D) = (B, C) \circ (C, D) = [7] \circ [3] = [16] = \{o, d, s \}$
   $(C, F) = (C, D) \circ (D, F) = [3] \circ [1]^{-1} = [3] \circ [2] = \{ > \}$
   $(D, E) = (D, C) \circ (C, E) = [4] \circ [9] = [14] = \{o, di, fi \}$
   $(B, E) = (B, C) \circ (C, E) = [7] \circ [9] = [7] = \{ m \}$

2. Path Length = 3
   $(A, E) = (A, B) \circ (B, C) \circ (C, E) = (A, C) \circ (C, E)$
   $= [1] \circ [9] = [1] = \{ < \}$
   $(A, D) = (A, B) \circ (B, C) \circ (C, D) = (A, C) \circ (C, D)$
   $= [1] \circ [3] = [22] = \{ <, o, m, d, s \}$
   $(B, F) = (B, C) \circ (C, D) \circ (D, F) = (B, D) \circ (D, F)$
   $= [16] \circ [1]^{-1} = [23] = \{ >, oi, mi, di, si \}$
   $(E, F) = (E, C) \circ (C, D) \circ (D, F) = (E, D) \circ (D, F)$
   $=[14]^{-1} \circ [2] = [15] \circ [2] = [2] = \{ > \}$

3. Path Length = 4
   $(A, F) = (A, B) \circ (B, C) \circ (C, D) \circ (D, F)$
   $= ((A, B) \circ (B, C)) \circ ((C, D) \circ (D, F))$
   $= (A, C) \circ (C, F) = [1] \circ [2] = [29]$
   $= \{ <, >, d, di, o, oi, m, mi, f, fi, s, si, e \}$

The restricted relation domain is a tree. The reason for using a tree is to avoid cycles which may introduce conflicts. A distributed system and a multimedia presentation contains a number of objects. When a new object is added to the presentation, an user edge and a node representing the new object is added. A number of derived edges are also inserted. Adding a new node to the complete graph K_n requires adding n new edges, where n is the number of nodes, to complete K_{n+1}. Since a complete graph is strongly connected, there exists an edge between each pair of nodes. When a new user edge is added, we can compute other derived edges from checking the composed relations between an existing edge and this new user edge. Algorithm *AddUERD* adds an user edge $l = (a, b)$ to a complete graph K_n :

Algorithm : *AddUERD*
Input : $l = (a, b), K_n = (K_nV, K_nE)$
Output : $K_{n+1} = (K_{n+1}V, K_{n+1}E)$
Preconditions : $l \notin K_nE \quad \wedge \quad a \in K_nV \quad \wedge \quad b \notin K_nV$
Postconditions : $| K_{n+1}V | = | K_nV | + 1 \quad \wedge$
$| K_{n+1}E | = | K_nE | + n$
Steps :
1: $K_{n+1}E = K_nE \cup \{ l \}$
2: for each $e = (c, b) \quad \wedge \quad c \neq a \quad \wedge \quad c \in K_nV$
   2.1: $e.rs = \cap \forall d \in K_nV, (c, d) \in K_nE, (d, b) \in K_nE$
   ($Table29((c, d).rs, (d, b).rs)$)
   2.2: $K_{n+1}E = K_{n+1}E \cup \{e\}$
3: $K_{n+1}V = K_nV \cup \{ b \}$

Considering the example above, if we add an user edges toaccording to the complete graph K_6 , the derived edges are computed :

Adding an user edge to K_6 :
Add (G, E) = { f } = [11]

Derived edges :

Derive (E, G) = (G, E)$^{-1}$ = [11]$^{-1}$ = [12] = { fi }
Derive (A, G) = (A, E) $\circ$ (E, G) = [1] $\circ$ [12]
= [1] = { < }
Derive (B, G) = (B, E) $\circ$ (E, G) = [7] $\circ$ [12]
= [1] = { < }
Derive (C, G) = (C, E) $\circ$ (E, G) = [9] $\circ$ [12] = [18]
= {<, o, m }
Derive (D, G) = (D, E) $\circ$ (E, G) = [14] $\circ$ [12]
= [24] = { <, o, m, di, fi }
Derive (F, G) = (F, E) $\circ$ (E, G) = (E, F)$^{-1}$ $\circ$ (E, G)
= [1] $\circ$ [12] = [1] = { < }

5 Qualitative Representation of Spatial Knowledge

5.1 Modeling Spatial Relations

To analyze a generalized model of spatio-temporal relations, we consider the following situations. Conclusively, the generalized model of temporal/spatial relations include four cases:

- two points on a line
- two points on a plan
- two line segments on a line

- two line segments on a plan
- two 2-D objects on a plan
- two 3-D object project onto x-y, z-x and y-z plans

For an arbitrary pair of points, *A* and *B*, located on a 1-dimensional line, there are three *point relations*: $A < B$, $A = B$, or $A > B$, for *A* is before *B*, *A* is at the same position as *B*, and *A* is after *B*, respectively. If these two points are located on a 2-dimensional plan, there exists nine (i.e., 3 * 3) cases. The X and the Y coordinates of these two points on the plan are independent. The possible relations between these two points on a plan can be denoted as *A* (<, <) *B*, *A* (<, =) *B*, *A* (<, >) *B*, *A* (=, <) *B*, *A* (=, =) *B*, *A* (=, >) *B*, *A* (>, <) *B*, *A* (>, =) *B*, and *A* (>, >) *B*, where the first element in the pair representing a point relation denotes the order on the X coordinate while the second is for the Y coordinate.

Considering two line segments on a 2-dimensional plan, according to the above table and since the position of these two line segments are independent at the X and the Y coordinates, there exists $13^2 = 169$ possible relations between these two line segments on a plan. These relations, similar to those of two points on a plan, are denotes by pairs as: (< , <), (< , >), (< , *d*), (< , *di*), ..., (00 , 01e), and (00 , 00). We use these 169 binary relations to model spatial point-interval relations of two lines on a plan. In Figure 1, X_A and X_B are the projection of two segments A and B. Y_A and Y_B are the projection on Y. X_A is {*start*} to X_B and Y_A is {*before*} to Y_B. We represent the spatial relation between A and B as (A, (s, <),B)

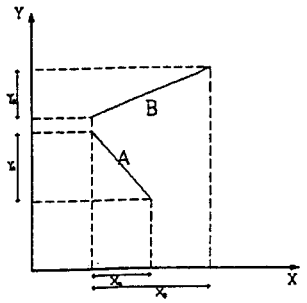

Figure 1: Projection of Two Line Segments on a 2-D Plan.

Relations of n-D objects can be used in object representation and recognition. A object in 3-D space can be projected onto *y-x*, *z-x*, and *y-z* planes. The projections correspond to surfaces generated from 3D objects. Similarly, a 2-D object is projected to *x* and *y* axes (see Figure 2). If we look at two objects in the n-dimensional space, we can project the positional relation between these two objects from *n* directions to *n* 1-D space. The projections of 2-D object are x-interval and y-interval, but not the point. Thus, an n-dimensional relation can be formularized by a conjunction of *n* 1-D interval relations. A conjunction of two 1-D relations, which denotes a 2-D relation, has 13^2 variations, i.e. {(<, <), (<, >), (<, d),..., (=, fi), (=, =)} where the first element in the pair representing a interval relation denotes the order on the X coordinate while the second is for the Y coordinate. Similarly, there are 13^3 3-D relations.

The first two cases are trivial and they do not efficiently express intervals. The third case can represent the relations of two temporal intervals. The fourth case is the semantic tool that we rely on to develops shape matching. The fifth and sixth cases can be used for symbolic subsequence matching.

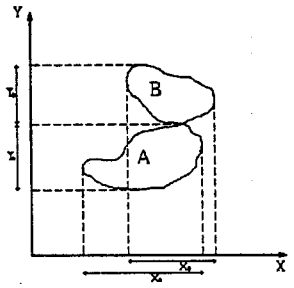

Figure 2: Relations Between Two 2-D Objects Project onto x-y Plan.

5.2 Spatial Reasoning

In this section, we introduce a mechanism to extend the spatial reasoning of objects to an n-dimensional space.

Let *rs* denote a set of 1-D temporal interval relations (i.e., *rs* ∈ *29Relset*). The relation composition table discussed in Table 2 can be refined (e.q., make each relation as an atomic set of that relation) to a function maps from the Cartesian product of two *rs* to a *rs*. Assuming that f^1 is the mapping function interpreting Allen's table, we can compute f^2, the relation composition function of 2-D objects, and f^3, the one for 3-D objects, from f^1. There are 13 relations for 1-D objects. A conjunction of two 1-D relations, which denotes a 2-D relation, has 13^2 variations. Similarly, there are 13^3 3-D relations. Fortunately, 4-D relations are not quite applicable and the memory space required for 2-D and 3-D relation tables is manageable by nowadays computers.

Since a 2-D relation is conjunction of two 1-D relations, we use the notation, $rs_1 \times rs_2$, to denote a 2-D relation set where rs_1 and rs_2 are two 1-D relation sets. Thus f^2 is a mapping from Cartesian product of two (*rs* o *rs*) to an (*rs* o *rs*). Similarly f^3 is obtained. The following are signatures of these functions:

$f^1 = 29RelSet \times 29RelSet \rightarrow 29RelSet$
$f^2 = 29RelSet \times 29RelSet \times 29RelSet \times 29RelSet \rightarrow 29RelSet \times 29RelSet$
$f^3 = 29RelSet \times 29RelSet \times 29RelSet \times 29RelSet \times 29RelSet \times 29RelSet \rightarrow 29RelSet \times 29RelSet \times 29RelSet$
where $29RelSet \times 29RelSet \in \{ \{<\} \times \{<\}, \{<\} \times \{>\}, \ldots, \{=\} \times \{=\} \}$
$29RelSet \times 29RelSet \times 29RelSet \in \{ \{<\}\times\{<\}\times\{<\}, \{<\}\times\{<\}\times\{>\}, \ldots, \{=\}\times\{=\}\times\{=\} \}$

Functions f^2 and f^3 are computed according to the following formulas :

$\forall\, i_1 \times j_1, i_2 \times j_2 \in P(29RelSet \times 29RelSet)$
$f^2(i_1 \times j_1, i_2 \times j_2) = \prod f^1(i_1, i_2) \times f^1(j_1, j_2)$
$\forall\, i_1 \times j_1 \times k_1, i_2 \times j_2 \times k_2 \in P(29RelSet \times 29RelSet \times 29RelSet)$
$f^3(i_1 \times j_1 \times k_1, i_2 \times j_2 \times k_2) = \prod f^1(i_1, i_2) \times f^1(j_1, j_2) \times f^1(k_1, k_2)$
where $\prod A \times B = \{ a \times b \mid \forall\, a \in A, b \in B \}$
$\prod A \times B \times C = \{ a \times b \times c \mid \forall\, a \in A, b \in B, c \in C \}$

The functions are implemented as table mappings. Table generated by the above formula are stored in memory to reduce run-time computation load. These tables are used in the algorithm discussed in Section 3.2, depending on which dimension of objects the algorithm is computing.

Example 5.1: Considering the five user spatial relations exist between objects. The reasoning algorithm discussed in Section 3 computes derived spatial relation edges until the last edge is added to K_n :

User edges:

(A, B) = { (<, m) } = ([1], [7])
(B, C) = { (oi, s) } = ([6], [9])
(C, D) = { (f, m) } = ([11], [7])
(B, E) = { (fi, di) } = ([12], [4])

Derivation based on user edges:

1. Path Length = 2
(*A*, *C*) = (*A*, *B*) o (*B*, *C*) = ([1], [7]) o ([6], [9])
= ([1] o [6], [7] o [9]) = ([2], [7]) = { (>, m) }
(*B*, *D*) = (*B*, *C*) o (*C*, *D*) = ([6], [9]) o ([11], [7])
= ([6] o [11], [9] o [7]) = ([6], [1]) = { (oi, <) }
(*A*, *E*) = (*A*, *B*) o (*B*, *E*) = ([1], [7]) o ([12], [4])
= ([1] o [12], [7] o [4]) = ([1], [1]) = { (<, <) }
(C, E) = (*C*, *B*) o (*B*, *E*) = (*B*, *C*)$^{-1}$ o (*B*, *E*)
= ([5], [10]) o ([12], [4]) = ([5] o [12], [10] o [4])
= ([18], [4]) = { (<, di), (o, di), (m, di) }

2. Path Length = 3
(A, *D*) = (*A*, *B*) o (*B*, *C*) o (*C*, *D*) = (*A*, *C*) o (*C*, *D*)
= ([2], [7]) o ([11], [7]) = ([2] o [11], [7] o [7])
= ([2], [1]) = { (>, <) }

6 Relational-Distance Computing

In this section we discuss some of the properties that need to be satisfied by spatial similarity functions. Relations are similar to each other in certain degree. For example, "during" and "starts" are similar since the only difference is the starting points of the two intervals are different. However, "before" and the inverse of "meets" are not quite the same.

A *relational-distance* of two relations belong to two different temporal relations occurs if those two temporal relations hold different relations.

Definition 6.1: A *point relation distance* (PRD) defined with respect to a point relation *r* of index *n* have *n* incompatible differences from *r*. The following table gives a definition of point relation distance:

Table 4: Point Relation Distance (PRD)

PRD	>	=	<
>	0	1	2
=	1	0	1
<	2	1	0

Definition 6.2: An *interval relation distance* (IRD) defined with respect to a interval relation *r* of index *n* have *n* incompatible differences from *r*. Let R and R' are two interval relations. The encoding point relation of R is $R_{As_{\diamond}Bs}$, $R_{As_{\diamond}Be}$, $R_{Ae_{\diamond}Bs}$, $R_{Ae_{\diamond}Be}$, and the encoding point relation of R' is $R'_{As_{\diamond}Bs}$, $R'_{As_{\diamond}Be}$, $R'_{Ae_{\diamond}Bs}$, $R'_{Ae_{\diamond}Be}$. We have a *IRD* formula:

$$IRD(R, R') = PRD(R_{As_{\diamond}Bs}, R'_{As_{\diamond}Bs}) + PRD(R_{As_{\diamond}Be}, R'_{As_{\diamond}Be}) + PRD(R_{Ae_{\diamond}Bs}, R'_{Ae_{\diamond}Bs}) + PRD(R_{Ae_{\diamond}Be}, R'_{Ae_{\diamond}Be})$$

Table 5: Interval Relation Distance (IRD)

IRD	<	>	d	di	o	oi	m	mi	s	si	f	fi	e
<	0	8	4	4	2	6	1	7	3	5	5	3	4
>	8	0	4	4	6	2	7	1	5	3	3	5	4
d	4	4	0	4	2	2	3	3	1	3	1	3	2
di	4	4	4	0	2	2	3	3	3	1	3	1	2
o	2	6	2	2	0	4	1	5	1	3	3	1	2
oi	6	2	2	2	4	0	5	1	3	1	1	3	2
m	1	7	3	3	1	5	0	6	2	4	4	2	3
mi	7	1	3	3	5	1	6	0	4	2	2	4	3
s	3	5	1	3	1	3	2	4	0	2	2	2	1

si	5	3	3	1	3	1	4	2	2	0	2	2	1
f	5	3	1	3	3	1	4	2	2	2	0	2	1
fi	3	5	3	1	1	3	2	4	2	2	2	0	1
e	4	4	2	2	2	2	3	3	1	1	1	1	0

7 Conclusions

The main contributions of this paper are to prove the properties of spatio-temporal transitive composition and to give a complete discussion of different possible temporal domains and spatial models. Most importantly, we propose a fast computation mechanism to maintain spatio-temporal constraints. We extended interval algebra to represent and recognize n-D object by projection relations.

We also argue that, many interesting researches can benefit from using the Infinite/Finite Temporal Interval Group for spatio-temporal constraint reasoning. The proposed algorithms are used in the implementation of a spatio-temporal relation computation program. This program is used in a system to detect inconsistent relations between objects. Temporal and spatial knowledge are managed by Finite Temporal Interval Group. The theorems are able to help the system to keep away from conflicts of synchronization specification.

In Section 5.1, the 2-D objects are represented, higher dimensional spatial objects project to planes are a subject for future research. The spatio-temporal computation model which we discussed can potentially be extended to moving objects representation. We can derived the relationships between arbitrary moving objects by temporal and spatial reasoning using Infinite Temporal Interval Group. This extension of integrated spatial/temporal computation models will significantly reduce the computation and storage requirement for representing moving objects in a scene.

The theocratic analysis and the algorithms proposed in this paper can be used in other computer applications for maintaining spatial/temporal knowledge. We hope that, the knowledge underlying interval algebra can be used in many computer applications, especially in AI applications.

References

[1] James F. Allen , " Maintaining Knowledge about Temporal Intervals " ,Communications of the ACM , Vol. 26 , No. 11 , 1983.

[2] Chi-Ming Chung, Timothy K. Shih, Jiung-Yao Huang, Ying-Hong Wang, and Tsu-Feng Kuo, " An Object-Oriented Approach and System for Intelligent Multimedia Presentation Designs , " in proceeding of the International Conference on Multimedia Computing and Systems (ICMCS '95) , Washington DC, U.S.A. May 15-18,1995, pp 278-281.

[3] Young Francis Day, et. al. ," Spatio-Temporal Modeling of Video Data for On-Line Object-Oriented Query Processing ," in proceedings of the International Conference on Multimedia Computing and Systems, Washington DC,U.S.A., May 15-18 ,1995 , pp 98-105 .

[4] Thomas Wahl, et. al., "TIEMPO: Temporal Modeling and Authoring of Interactive Multimedia " in proceedings of the international conference on multimedia computing and systems, Washington DC, U.S.A., May 15-18, 1995, pp 274-277.

[5] Venkat N. Gudivada, and Vijay V. Raghavan, "Content-Based Image Retrieval Systems," IEEE Computer, September 1995, pp 18 -- 22.

[6] Cherif Keramane and Andrzej Duda, " Interval Expressions -a Function Model for Interactive Dynamic Multimedia Presentations," in proceedings of the 1996 International Conference on Multimedia Computing and Systems , Hiroshima, Japan, June 17-23, 1996 , pp 119-133.

[7] John Z. Li, M. TamerOzsu, and Duane Szafron, " Spatial Reasoning Rules in Multimedia Management Systems,"in proceedings of the 1996 Multimedia Modeling International Conference (MMM'96), Toulouse, France, November 12-5,1996, pp 119-133.

[8] Thomas D. C. Little and Arif Ghafoor, " Spatio-Temporal Composition of Distributed Multimedia Objects for Value-Added Networks," IEEE Computer, October 1991, pp 42-50.

[9] Thomas D. C. Little and Arif Ghafoor " Interval Based Conceptual Models for Time-Dependent Multimedia Data," IEEE transactions on knowledge and data engineering , Vol. 5, No. 4,1993, pp 551-563.

[10] Timothy K. Shih, Steven K. C. Lo, Szu-Jan Fu, and Julian B. Chang, " Using Interval Temporal Logic and Inference Rules for the Automatic Generation of Multimedia Presentations," in Proceedings of the IEEE International Conference on Multimedia Computing and Systems, Hiroshima, Japan, June 17-23, 1996, pp. 425-428.

[11] Timothy K. Shih, Chin-Hwa Kuo, Huan-Chao Keh, Chao T. Fang-Tsou, and Kuan-Shen An, " An Object-Oriented Database for Intelligent Multimedia Presentations," in proceedings of the 1996 IEEE International Conference on Systems, Man and Cybernetics, Beijing, China, October 14-17, 1996.

[12] Yannis Theodoridis, Michael Vazirgiannis, and Timos Sellis, " Spatial Temporal Indexing for Large Multimedia Applications," in proceedings of the 1996 International Conference on Multimedia Computing and Systems, Hiroshima, Japan, June 17-23, 1996, pp 441-448.

[13] Michael Vazirgiannis, Yannis Theodoridis, and Timos Sellis " Spatio-Temporal Composition in Multimedia Applications," in proceedings of the International Workshop on Multimedia Software Development, March 25-26, Berlin, Germany, 1996, pp 120-127.