使用區段代數作爲時空定性推理
Using Interval Algebra for Qualitative Spatio-Temporal Reasoning

施國琛 (Timothy K. Shih), 張祐城 (Anthony Y. Chang),
林尤群 (Yule-Chyun Lin) and 高盟欣 (Meng-Hsin Kao)
Department of Computer Science and Information Engineering
Tamkang University
Tamsui,Taiwan
R.O.C.
e-mail: g5190315@tkgis.tku.edu.tw
fax: Intl. (02) 620 9749

摘要

我們建構一個時序遞移運算之代數結構，並證明其爲一群。依此代數結構所推算之演算法可偵測並刪除時空關係之衝突。

Abstract

The relations among temporal intervals can be used to model all time dependent objects. We propose a fast mechanism for temporal relation compositions. A temporal transitive closure table is derived, and an interval-based temporal relation algebraic system is constructed. Thus, we propagate the time constraints of arbitrary two objects across long distances n by linear time. We also give a complete discussion of different possible domains of interval relations. A set of algorithms is proposed to detect time conflicts and to derive reasonable interval relations. The algorithms are extended for time-based media in an arbitrary n-dimensional space.

1 Introduction

Communication networks and multimedia applications usually contains a number of resources to be presented sequentially or concurrently. Temporal in-terval relations represent the timing among resources. These resources need to be analyzed to ensure that there is no time conflict among resources. Moreover, many of these resources, occupy period of time and screen space. These data can be heavily time-dependent, such as audio and video in a motion picture, and can require time-ordered presentation. The spatio-temporal relations among resources need to be computed and represented.

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 temporal interval. The work discussed in [1] analyzes the relations among temporal intervals. However, the work [1] only states temporal interval relations. No spatial relation were discussed. We found that these relations can be generalized for spatial modeling.

We have surveyed many researches related to the spatio-temporal semantics of multimedia and distributed objects. However, no discussion of the conflict situation among relations were found. We found that, the use of spatio-temporal relations serves as a reasonable semantic tool for the underlying representation of objects in many multimedia applications. Composite objects can have arbitrary timing relationships. These might be specified to achieve some particular visual effect of sequence.

2 The Spatio-Temporal Relation Domains

According to the interval temporal relations intro-duced 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 propa-gation. The general idea is to use the existing infor-mation 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. A reasonable set is a set of reasonable relations according to our definition.

In some cases, relation compositions may result in a *conflict specification* due to the user specification or involved events synchronously. For example, if speci-fications " 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 suggest that the computation domain reveals four types, as discussed below.

- The **complete relation domain** (a complete graph) : contains user edges and derived edges, with possible cycles and possible conflicts.
- The **reasonable relation domain** (a graph) : contains user edges and derived edges, with possible cycles but no conflict.
- The **reduced relation domain** (a graph) : contains only user edges, with possible cycles and possible conflicts.
- The **restricted relation domain** (a tree) : contains only user edges, without cycle.

The four domains are used in the analysis and computation of object relations. In section 4, we propose two algorithms computing the reasonable relation domain.

3 Finite Temporal Relations Group

Based on Allen's work, transitivity table for the twelve temporal relations (omitting "=") showing the composition of interval temporal relations. Compo-sitions 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), named *Table29*, can be calculated. The compositions of three or more relations can be ob-tained directly from our table. Algorithm *Compute-Table29* calculates *Table29*, which consists of the compositions of 29 temporal relation sets. 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 ob-jects 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.

$$Name == P\,(string)$$
$$13Rel == \{ <, >, d, di, o, oi, m, mi, s, si, f, fi, e \}$$
$$<^{-1} = > \;\wedge\; d^{-1} = di \;\wedge\; o^{-1} = oi \;\wedge\; m^{-1} = mi$$
$$\wedge\; s^{-1} = si \;\wedge\; f^{-1} = fi \;\wedge\; e^{-1} = e$$
$$29Relset \subset P\,(13Rel)$$
$$\forall\, rs \in 29RelSet \bullet rs^{-1} = \{ r^{-1} \in 13Rel \mid r \in rs \}$$
$$TemporalTuple == Name \times 29RelSet \times Name$$
$$\forall\, tt : TemporalTuple \bullet$$
$$tt = (A, rs, B) \Leftrightarrow tt^{-1} = (B, rs^{-1}, A)$$
$$\circ : TemporalTuple \times TemporalTuple \rightarrow TemporalTuple$$
$$\forall\, tt_1, tt_2, tt_3 : TemporalTuple \bullet tt_1 = (A, rs_1, B) \;\wedge$$
$$tt_2 = (B, rs_2, C) \;\wedge\; tt_3 = (A, rs_3, C) \bullet$$
$$tt_1 \circ tt_2 = tt_3 \Leftrightarrow$$
$$(A = C \wedge rs_2 = rs_1^{-1} \Rightarrow rs_3 = \{ e \} \vee$$
$$A = C \wedge rs_2 \neq rs_1^{-1} \Rightarrow rs_3 = \bot \vee$$
$$A \neq C \Rightarrow rs_3 = Table29\,(rs_1, rs_2))$$

A temporal tuple contains two interval names and a relation set. The temporal tuple composition operator (i.e., o) checks whether the interval names A and C are equal. If so, and if the relation set of tt_1 and tt_2 are the inverse of each other, the composition results in an equality (i.e., { e }). On the other hand, if A and C are not equal, *Table29* is used for looking up the compo-sition result.

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

The 29 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. $rs = \cup\ \forall r_1 \in rs_1,\ \forall r_2 \in rs_2\ \bullet (r_1, r_2) \in rs_1 \times rs_2$ $Table13\ (r_1, r_2)$

In Function *RelComp*, the reasonable set computed must be the union of all possible combinations of the pair of relations obtained from the two input relation sets, name rs_1 and rs_2. The function uses a table lookup function to obtain a set of relations .

Table 1 : 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	27	29	27	29	27	29	27	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

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 $C(13, 0) + C(13, 1) + C(13, 2) + \ldots + C(13, 13) = 2^{13}$ 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.

Definition 3.1 : Given a nonempty set S = *TemporalTuple* , o is a binary operation on S, o is the temporal relation transitive function , the domain is $S \times S$, the codomain is S, i.e. $\circ : S \times S \rightarrow S$. This mapping is sometimes called a law of composition.

Definition 3.2 : To combine S and binary operation o is an algebraic system and was denoted by $< S , \circ >$.

Theorem 3.1 : Let $< S , \circ >$ be a temporal algebraic system, and S be a set with a law of composition, then $< S, \circ >$ is closed.

Proof : Since function $\circ : S \times S \rightarrow S$, and S is equal to *29RelSet*, the function is closed to *29RelSet*. ■

Theorem 3.2 : Let $< S, \circ >$ be a temporal algebraic system, and S be a set with a law of composition, then all $a \in S$, exists $b \in S$, such that $a \circ b = b \circ a =$ (A,{e}A) , b is called inverse of a.

Proof : 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 :

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) ■

Theorem 3.3 : Let $< S, \circ >$ be a temporal algebraic system, and S be a set with a law of composition, then $< S, \circ >$ has an unique identity (A,{e},A). i.e. all $a \in S$, $a \circ (A, \{e\}, A) = a = (A, \{e\}, A) \circ a$.

Proof : To prove the identity of function o , we need to show that

$\forall\ tt \in TemporalTuple \bullet$
$a \circ a^{-1} = (A, \{e\}, A) \ \wedge \ a^{-1} \circ a = (A, \{e\}, A) \ \wedge$
$a \circ (A, \{e\}, A) = a \ \wedge \ (A, \{e\}, A) \circ a = a$

From the table lookup of *Table29* , we can easily verify that $\forall\ rs \in 29RelSet \bullet rs \circ \{e\} = rs \wedge \{e\} \circ rs = rs$. It is clear that $\forall\ a \in TemporalTuple \bullet a \circ (A, \{e\}, A) = a \wedge (A,\{e\}, A) \circ a = a$. Due to *Theorem 3.2*, and the inverse relation sets table given above, we can look at *Table29* for the composition of each pair of rs and rs^{-1}, as well as for each pair of rs^{-1} and rs. ■

Theorem 3.4 : Let $< S, \circ >$ be a temporal algebraic system, and S be a set with a law of composition, then $< S, \circ >$ is associative. i.e. all $a , b , c \in S$, $(a \circ b) \circ c = a \circ (b \circ c)$.

Proof : Let L be an ordered list of relation sets obtained from I according to the order given in the 29 relation set table (i.e., $L = (1, 2, 3,\ldots, 29)$). We further define L^2 to be an ordered list of elements obtained from *Table29* according to the row major order. L^2 has 841 (i.e., 29^2) elements. We can easily compute a table $T_{29 \times 841}$ from L and L^2 by :

$\forall\ X, Y, Z : TemporalTuple \bullet$
$X = (A, rs_1, B) \wedge$
$rs_1 = L (i) \ \wedge \ 1 \leq i \leq 29 \wedge$
$Y = (B, rs_2, C) \wedge$
$rs_2 = L^2 (j) \ \wedge \ 1 \leq j \leq 841 \wedge$
$Z = (A, rs_3, c) \wedge$
$rs_3 = T (i, j) \Leftrightarrow X \circ Y = Z$

There are 24389 (i.e., 29 * 841) elements in table T. Similarly, we can compute another table from L^2 and L, named $T'_{841\times29}$. Assuming that L^3 is an ordered list of elements obtained from T according to the row major order. And L'^3 is a similar list obtained from T'. For an arbitrary i, $1 \le i \le 24389$, if $L^3(i)$ is the relation set of $X \circ (Y \circ Z)$, then $L'^3(i)$ is the relation set of $(X \circ Y) \circ Z$. We can verify that $L^3 = L'^3$. An implemented program shows the result holds.

4 Temporal Relation Inference

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.

If there is a conflict cycle in the original reduced relation domain, the algorithm eliminates that conflict first by altering the user to select a reasonable relation to replace the original one. This is why the resulting graph may contain some new user edges (i.e. UE'). 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. The reason of using the user edges is that these edges contain the minimal and sufficient information of what the user wants.

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 $e = (a, b) \wedge e \notin K_nE \wedge a \in K_nV \wedge b \in K_nV \bullet$
    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}) (or user edge, if $pl = 2$) must have been computed in a previous interaction. The algorithm repeats until all edges are added to the complete graph K_n, which contains $n*(n-1)/2$ edges.

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 $P = ((n_1, n_2), (n_2, n_3), \ldots, (n_{k-1}, n_k))$ in G' with $n_1 = n_k \wedge k > 3$
  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 com-putes 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
  $(A, F) = (A, B) \circ (B, C) \circ (C, D) \circ (D, F)$
  $= ((A, B) \circ (B, C)) \circ ((C, D) \circ (D, F))$
  $= (A, C) \circ (C, F) = [1] \circ [2] = [29]$
  $= \{ <, >, d, di, o, oi, m, mi, f, fi, s, si, e \}$

5 N-Dimensional Spatial Relations

Let *rs* denote a set of 1-D temporal interval relations (i.e., $rs \in 29Relset$). The relation composition table discussed in [1] 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.

$$f^1 = 29RelSet \times 29RelSet \rightarrow 29RelSet$$

$$f^2 = 29RelSet \times 29RelSet \times 29RelSet \times 29RelSet \rightarrow 29RelSet \times 29RelSet$$

$$f^3 = 29RelSet \times 29RelSet \times 29RelSet \times 29RelSet \times 29RelSet \times 29RelSet \rightarrow 29RelSet \times 29RelSet \times 29RelSet$$

where $29RelSet \times 29RelSet \in \{ \{<\} \times \{<\}, \{<\} \times \{>\}, \dots, \{=\} \times \{=\} \}$

$$29RelSet \times 29RelSet \times 29RelSet \in \{ \{<\}\times\{<\}\times\{<\}, \{<\}\times\{<\}\times\{>\},\dots, \{=\}\times\{=\}\times\{=\}\}$$

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

$$\forall\, i_1 \times j_1,\, i_2 \times j_2 \in \mathrm{P}\,(29RelSet \times 29RelSet)$$

$$f^2\,(i_1 \times j_1,\, i_2 \times j_2) = \prod f^1\,(i_1, i_2) \times f^1\,(j_1, j_2)$$

$$\forall\, i_1 \times j_1 \times k_1,\, i_2 \times j_2 \times k_2 \in \mathrm{P}\,(29RelSet \times 29RelSet \times 29RelSet)$$

$$f^3\,(i_1 \times j_1 \times k_1,\, i_2 \times j_2 \times k_2) = \prod f^1\,(i_1, i_2) \times f^1\,(j_1, j_2) \times f^1\,(k_1, k_2)$$

$$\text{where} \prod A \times B = \{ a \times b \mid \forall\, a \in A, b \in B \}$$

$$\prod A \times B \times C = \{ a \times b \times c \mid \forall\, a \in A, b \in B, c \in C \}$$

The functions are implemented as table mappings. Table generated by the above formula are stored in memory to reduce run-time computation load.

6 Conclusions

In an appli-cation, the presentation system can use our algorithms to compute the schedule of a presentation. Also, in natural language processing, temporal intervals are used to model the timing of events. Our algorithms thus can be used in con-structing semantics of sentences. We believe that, spatio-temporal relations can be used in many re-lated applications for maintaining time constraints.

The main contributions of this paper is in building the algebra system of spatio-temporal interval relations and the set of enhanced mechanism for spatio-temporal relation composition. These algo-rithms deal with an arbitrary number of objects in an arbitrary n-dimensional space. We propose many properties of temporal interval relations and prove the correctness of these proper-ties. We also argue that, many interesting researches in multimedia applications can benefit from using these spatio-temporal relations and our algorithms.

The algorithm proposed in this paper can be used in other computer applications. We hope that, with our analysis and algorithms, the knowledge underlying temporal interval relations can be used in many computer applications, especially in distributed multi-media computing and networking.

References

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

[2] 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 .

[3] 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 12-15,1996,pp 119-133.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.