

Graph-based Consistency Checking in Spatial Information Systems

Jun KONG

Kang ZHANG

*Department of Computer Science, The University of Texas at Dallas,
Richardson, TX 75080-0688, USA
{jxk019200, kzhang}@utdallas.edu*

Abstract

Consistency checking of cardinal directions is one of the important problems in qualitative spatial reasoning. This paper presents a graph model to visually represent direction specifications. In the model, nodes represent regions occupied by objects, and directed edges indicate direction relationships between objects. This graph model can be applied not only to consistency checking, but also to general spatial reasoning. Based on this model, we present an efficient algorithm that performs consistency checking on a set of definitive direction specifications by analyzing the connectivity of the participating nodes. The consistency checking algorithm is performed in $O(n^4)$ time.

1. Introduction

The real world is made up of numerous objects with spatial relationships, and it is necessary to recognize their locations relative to each other. Therefore, spatial knowledge consisting of information about topology [1] [2], orientation [3], distance [4] and shape [5] etc is an important class of commonsense knowledge, and attracts much attention.

Qualitative approach to the representation of spatial knowledge, which is characterized by making only as many distinctions in the domain of discourse as necessary, gains popularity recently [6], since it is considered to be analogue to how human represents and reasons about commonsense knowledge. The issues in qualitative spatial reasoning (QSR) include spatial inference, consistency checking, path finding etc, which can be applied to many applications, such as spatial databases [7].

This paper focuses on the problem of consistency checking, i.e. whether we can find a solution to assign a location to every object satisfying a set of direction specifications. In the previous researches, many spatial models approximated an object by a point or by a minimal bounding box [8]. Such an approximation, however, is often inaccurate. Goyal and Egenhofer [9] presented a model that only approximated the reference object while

using the exact shape of the primary object. Skiadopoulos and Koubarakis [10] formally defined and studied the composition operation for cardinal direction relationships expressed in the model of Goyal and Egenhofer. Furthermore, the first consistency checking algorithm based on Goyal and Egenhofer's spatial model was developed through constraint solving [11].

Goyal and Egenhofer [9] defined nine atomic relationships between two objects, southwest (SW), west (W), northwest (NW), north (N), northeast (NE), east (E), southeast (SE), south (S) and same (O). We present a graph model to visually represent direction specifications, and perform a consistency checking on the graph model rather than through a constraint solver used by Skiadopoulos and Koubarakis [11]. In the graph model, nodes represent the regions occupied by objects, and directed edges indicate the direction relationships between objects. Based on the model, consistency checking is performed by analyzing the connectivity of participating nodes. Our main contributions include a graph-based model for effective spatial reasoning and a consistency checking algorithm that runs in $O(n^4)$ time instead of $O(n^5)$ as required by the constraint-based approach of Skiadopoulos and Koubarakis.

The rest of the paper is organized as follows. Section 2 briefly introduces the spatial model of Goyal and Egenhofer [9]. Section 3 presents our graph model in representing direction relationships. Section 4 illustrates how to refine a generated directed graph. Section 5 analyzes the connectivity constraints among nodes. Section 6 presents a consistency checking algorithm and analyzes its time complexity. Section 7 reviews related work, and finally Section 8 gives the conclusion and proposes future research.

2. The spatial model for direction definition

We define directions in the Euclidean space R^2 , where the region occupied by an object is specified as a non-empty set of points. In the following description, the term "object" denotes the region occupied by an object.

Definition 1: An object i occupies a *region* $Reg_i = \{(x,y) \mid f(x,y) = \text{true}\}$. Reg_i is a closed area with boundary lines.

The projections of the greatest lower bound of an object i on the y-axis and x-axis are denoted as $Inf_y(i)$ and $Inf_x(i)$, and the projections of the least upper bound on the y-axis and x-axis as $Sup_y(i)$ and $Sup_x(i)$ respectively.

Definition 2: An object i is *valid in the vertical direction* iff $Inf_y(i) < Sup_y(i)$.

Definition 3: An object i is *valid in the horizontal direction* iff $Inf_x(i) < Sup_x(i)$.

Definition 4: An object i is *valid* iff it is *valid* in both vertical and horizontal directions.

Concerning an object i , the straight lines at $Inf_x(i)$, $Sup_x(i)$, $Inf_y(i)$ and $Sup_y(i)$ construct a minimal bounding box denoted as mbb_i , and divide the first quadrant into eight areas denoted as NW_i , N_i , NE_i , W_i , E_i , SW_i , S_i and SE_i as shown in Figure 1.

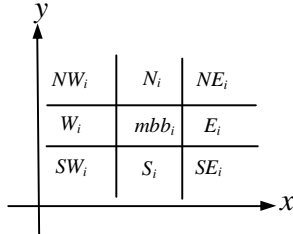


Figure 1. Eight direction areas

Definition 5: An object p , called *primary object*, locates at P of another object r , called *reference object*, iff $Reg_p \cap P_r \neq \text{NULL}$, where $P \in \{NW, N, NE, W, E, SW, S, SE\}$. The *same* relationship defined by Goyal and Egenhofer [9], i.e. the relationship between the overlapping parts of two overlapped objects, is not considered in this paper.

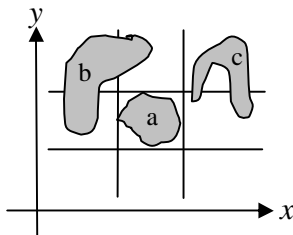


Figure 2. An example

A primary object can lie in more than one direction area determined by a reference object. We, therefore, use a set to represent a direction relationship between objects. For example, Figure 2 demonstrates that the primary object b occupies three direction areas relative to the reference object a , and the direction relationship is denoted as $b\{N,NW,W\}a$. Moreover, we impose a counterclockwise order on the direction set, which implies that $b\{W,NW,N\}a$ is not a valid direction specification.

In order to specify the condition of several possible direction relationships between objects, the notation “+” is introduced to represent disjunctive directions. For example, $a\{NW,W\}+\{W,SW\}b$ specifies that a may locate northwest and west, or west and southwest to b . The consistency checking on a set of disjunctive direction specifications is NP-Complete [11].

Definition 6: A definitive direction relationship between two objects is represented as bRa , where b is the primary object, and a is the reference object. In general, $R \in 2^P$, where $P = \{N, NW, W, SW, S, SE, E, NE\}$.

Given bRa , the following properties hold:

- $R \subseteq \{NW, N, NE\}$ iff $Sup_y(a) \leq Inf_y(b)$;
- $R \subseteq \{SW, S, SE\}$ iff $Sup_y(b) \leq Inf_y(a)$;
- $R \subseteq \{NW, W, SW\}$ iff $Sup_x(b) \leq Inf_x(a)$;
- $R \subseteq \{NE, E, SE\}$ iff $Sup_x(a) \leq Inf_x(b)$.

We assume that every sub-region of a primary object divided by a reference object is also closed without a hole. In Figure 2, for example since object c contains two separate regions in E_a , c 's relationship with a is beyond our consideration.

3. A graph model for spatial reasoning

Based on the afore-mentioned spatial model, this section presents a graph model to represent the direction relationships between objects for effective consistency checking and spatial reasoning.

3.1. Preliminaries

We use D_x to denote a set of direction specifications among a set of objects X .

Definition 7: D_x is *consistent in vertical direction* iff $\forall x \in X, x$ is *valid in the vertical direction*.

Definition 8: D_x is *consistent in horizontal direction* iff $\forall x \in X, x$ is *valid in the horizontal direction*.

Definition 9: D_x is *consistent* iff $\forall x \in X, x$ is *valid*.

According to the above definition, the set of direction specifications $\{a\{N\}b, b\{N\}c\}$ is consistent. On the contrary, $\{a\{N\}b, b\{N\}c, c\{N\}a\}$ is inconsistent, which can be easily verified. If the above set is consistent, it should satisfy the condition $Inf_y(r) < Sup_y(r)$, where $r \in \{a,b,c\}$. According to direction specifications, the following properties can be easily derived:

$$\begin{aligned} Inf_y(a) &\geq Sup_y(b) \\ Inf_y(b) &\geq Sup_y(c) \\ Inf_y(c) &\geq Sup_y(a) \end{aligned}$$

Therefore, we derive $Inf_y(a) \geq Sup_y(a)$, which contradicts with the assumed condition.

Consistency checking can be performed independently on both vertical and horizontal directions, since they are orthogonal. In the following sections, we only illustrate

how to detect inconsistency in the vertical direction. The same principle applies to that in the horizontal direction.

3.2. The graph model

If nodes are used to represent the regions occupied by objects, and directed edges to indicate the direction relationships between objects, we can convert a set of direction specifications into a directed graph. Consequently, consistency checking becomes the problem of connectivity checking on the directed graph.

A reference object r divides a primary object p into one or more pieces while keeping itself as one entity. Therefore, r is always represented by a single node in our graph model. On the other hand, p is represented by node(s), the number of which is equal to that of sub-regions obtained from the division by r . Briefly, the process of generating a directed graph proceeds as follows:

1. Generate a node to represent the region occupied by r , and node(s) to represent sub-region(s) of p .
2. Connect two nodes representing adjacent sub-regions of p directed from south to north.
3. Connect the node representing a sub-region of p and the node representing r directed from south to north.

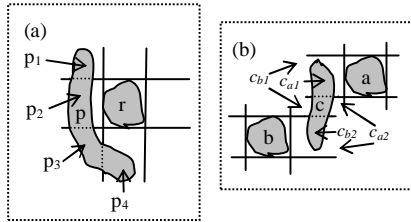


Figure 3. Visual illustration about divisions

Given a specification $p\{NW,W,SW,S\}r$, object p is divided into four regions, namely p_1, p_2, p_3 and p_4 , by r as illustrated in Figure 3(a). When performing consistency checking in the vertical direction, we only care horizontal divisions (respectively, vertical divisions are considered when we perform consistency checking in the horizontal direction), which divide object p into three regions, i.e. p_1, p_2 and $(p_3 \cup p_4)$. Due to orthogonality, we only illustrate consistency checking in the vertical direction. The same principle applies to the horizontal direction. In the following description, we will omit the mention of directions when the context is clear. Regarding the primary object, each sub-region obtained from the division by the reference object is represented by a node, and a directed edge connects two nodes representing adjacent sub-regions of a single object from south to north.

As mentioned above, a directed graph is generated to designate the division to a primary object according to the direction specification. Figure 4 lists all possible directed graphs, called *partitions*, which illustrate the direction relationships between sub-regions of a primary object

obtained from the division by a reference object. Figure 4 only presents one example of direction specifications corresponding to every type of *partitions*, while different direction specifications can generate the same type of *partition*. For example, $p\{NW,W\}r$ and $p\{W,SW\}r$ lead to the same *partition* (*Partition 3*) concerning the primary object p .

We define a *Level* function for each node and a *Depth* function for every afore-mentioned directed graph $D=(V, E)$, and $a_i, a_j \in V$:

$$Level(a_i) = \begin{cases} 1, & \text{if } a_i \text{ has no outgoing edge;} \\ MAX\{Level(a_j)\} + 1, & (a_i, a_j) \in E, \text{ otherwise.} \end{cases}$$

$$Depth(D) = MAX\{Level(a_i)\}.$$

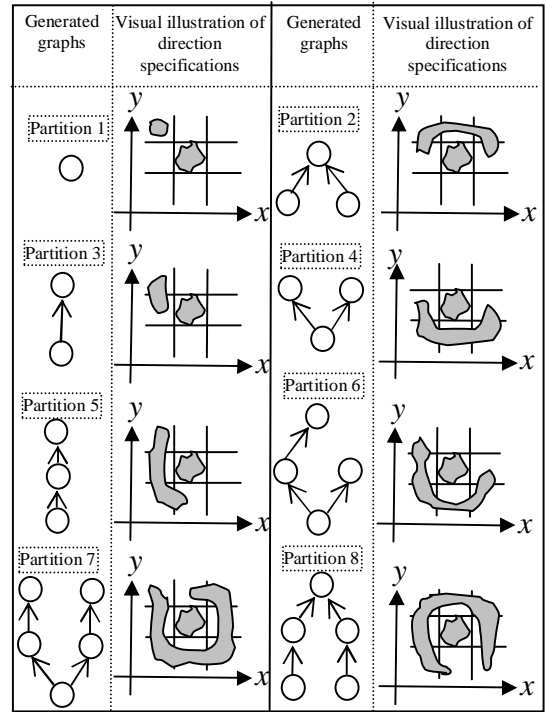


Figure 4. Directed graphs representing all possible horizontal divisions

A primary object can be specified in relation to different reference objects in D_x . Consequently, it will be divided several times by different reference objects. For every object a in X , we introduce a set of regions denoted as $A=\{a_{xj} \mid a_{xj} \text{ is a sub-region belonging to } a\}$. Every element in A is distinguished by two indices. The first index x indicates the object in the direction specification of aRx or xRa , and the second index j indicates the j_{th} sub-region obtained from a division according to the above specification. Specifically, when object a serves as a reference object in relation to object x , a_{x1} is added to A . For example, $D_x = \{a\{NW,W\}b, c\{SE,E,NE\}b,$

$c\{N,NW,W\}a\}$ involves three objects. Correspondingly, three sets of regions, namely A , B and C , are created. The first occurrence of a is in $a\{NW,W\}b$, and a is divided into two sub-regions by b . Therefore, a_{b1} and a_{b2} are inserted into A . In its second occurrence, a acts as a reference object in $c\{N\}a$. Correspondingly, a_{c1} is added to A . Since there is no more occurrence of a in D_x , $A=\{a_{b1}, a_{b2}, a_{c1}\}$. The other two sets of regions are presented as follows:

$$B=\{b_{a1}, b_{c1}\}$$

$$C=\{c_{b1}, c_{b2}, c_{b3}, c_{a1}, c_{a2}\}$$

In the graph model, nodes are used to represent regions satisfying the condition that regions with common boundaries must be mapped to the same node. When an object x remains one entity, its boundaries, i.e. $Inf_y(x)$ and $Sup_y(x)$, are unchanged. Therefore, in a set A , elements representing the entity of a should be mapped to the same node. For example, given $D_x=\{a\{N\}b, c\{N\}a\}$, $A=\{a_{b1}, a_{c1}\}$. Since a_{b1} and a_{c1} represent the entity with common boundaries of $Inf_y(a)$ and $Sup_y(a)$, they are mapped to a single node. However, if an object is divided into several sub-regions, which share no common boundaries, then each sub-region is represented by a unique node. For example, in Figure 3(b), c is divided into two sub-regions by a , namely c_{a1} and c_{a2} . Also, c is divided into c_{b1} and c_{b2} by b . Obviously, the four sub-regions share no common boundaries. Each of them maps to a unique node. After combining elements representing an identical region and connecting every pair of nodes representing adjacent sub-regions from south to north, a *component graph*, which demonstrates all divisions to an object in a set of direction specifications, is constructed.

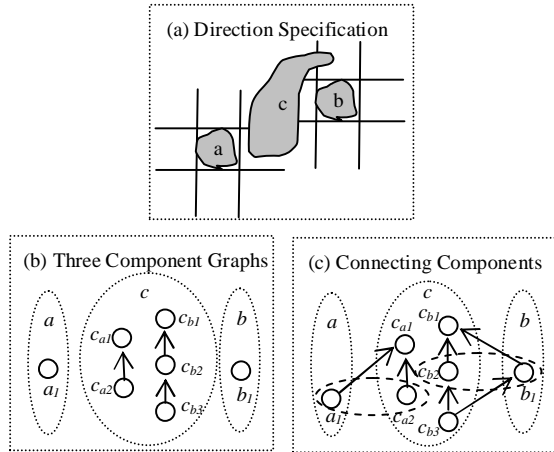


Figure 5. Spatial graph in vertical direction

After constructing component graphs for all objects, we can build relationships between nodes in different component graphs, i.e. between primary and reference objects. If a sub-region of a primary object p locates south to a reference object r , starting from the node representing the sub-region of p and ending at the node representing r ,

a directed edge connects the two nodes. On the other hand, if a sub-region of p locates north to r , we reverse the connecting direction.

For example, a set of direction specifications $D_x=\{c\{E,NE\}a, c\{N,NW,W,SW\}b\}$ is visualized in Figure 5(a). Object c is divided into two sub-regions by a , namely c_{a1} and c_{a2} , and into three sub-regions by b , namely c_{b1} , c_{b2} and c_{b3} . The *component graph* of c is illustrated in Figure 5(b). Since c_{a1} is north to a , an edge connects from a to c_{a1} as illustrated in Figure 5(c). We call such a graph *Spatial Graph in Vertical Direction* or *SGV* (the two dashed circles containing a_1 and c_{a2} , and b_1 and c_{b2} will be explained in Section 4).

Theorem 1: In a *SGV*, the following properties hold for every node in Partitions 2 to 8:

- a_i has no outgoing edge if $Level(a_i) = 1$, and
- a_i has no incoming edge if $Level(a_i) = Depth(D)$, where $D = (V, E)$ is a partition, and $a_i \in V$.

Proof:

(1) First, we prove that a_i has no outgoing edge if $Level(a_i) = 1$.

Assume an edge $e=(a_i, b_j)$, which indicates $Inf_y(b_j) \geq Sup_y(a_i)$. According to the definition of *Level*, it follows that a_i and b_j belong to different component graphs. If a_i is the reference object, a_i falls in Partition 1, which contradicts with the assumption. Therefore, b_j must be the reference object.

If the highest point of a is contained in the region represented by a_i , every point in a is lower than any point in b_j , i.e. $Inf_y(b_j) \geq Sup_y(a)$. Therefore, a falls in Partition 1, which contradicts with the assumption. Otherwise, assume the region represented by a_m contains the highest point. If $Sup_y(a_m) \leq Inf_y(b_j)$, a falls in Partition 1; if $Sup_y(a_m) > Inf_y(b_j)$, the line of $Inf_y(b_j)$ divides a into two regions. a_m represents the upper region, and a_i the lower one. Therefore, there is an edge from a_i to a_m , which contradicts with the assumption $Level(a_i) = 1$.

(2) The second statement can be proven in a similar way. ■

4. Refining the SGV

A *SGV* only builds relationships between nodes in the vertical direction. However, relationships in the horizontal direction function as a bridge between different objects, and play an important role in consistency checking. For example, given $\{a\{W\}b, c\{N\}a, b\{N\}c\}$, we construct a *SGV* as illustrated in Figure 6(a).

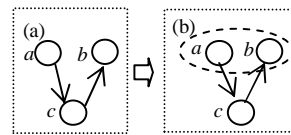


Figure 6. Two nodes merged into a super node

The above set of direction specifications is inconsistent, which cannot be detected from the *SGV*. Since object a locates west to object b , we can treat them as a

single object when analyzing consistency in the vertical direction. Through a super-node obtained by merging a and b as illustrated in Figure 6(b), a cycle indicating inconsistency appears in the graph. In general, when performing consistency checking in the vertical direction, we need to merge each pair of nodes representing two objects with west/east relationships into a super-node denoted by a dashed circle (similarly, we merge the nodes representing objects with south/north relationships when performing consistency checking in the horizontal direction). For example, in Figure 5(c), a_1 and c_{a2} are merged into a super-node, and b_1 and c_{b2} into another one.

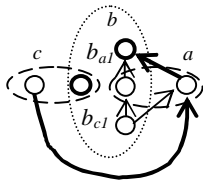


Figure 7. An incorrect merge

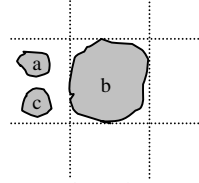


Figure 8. A valid case for the specifications in Figure 7

Such a simple merge, however, is not always correct. Given $\{b\{SE,E,NE\}a, c\{W\}b, a\{N\}c\}$, the corresponding SGV is illustrated in Figure 7, and a path from b_{c1} to b_{a1} is formed with the help of super-nodes. The path indicates inconsistency (the explanation will be given in Section 5), but we can find a consistent case as shown in Figure 8. The reason resulting in a wrong deduction is that when $a\{W\}b$ or $a\{E\}b$ is defined in D_x , the two dividing lines of $Inf_y(b)$ and $Sup_y(b)$ may not cross over object a . Since $Inf_y(a)$ and $Sup_y(a)$ must cross over object b in the condition of $a\{W\}b$, we replace $a\{W\}b$ with a new direction specification by swapping the reference object and the primary object ($a\{E\}b$ is processed in a similar way as $a\{W\}b$). The new direction specification is determined by the size of a . Specifically, there are four possible cases regarding the position and height of a as illustrated in Figure 9(a):

- a is smaller than b , and there is no common upper or lower boundary;
- a is smaller than b , and there is a common upper boundary;
- a is smaller than b , and there is a common lower boundary;
- a and b are of the same height.

Corresponding to each case, a SGV is presented in Figure 9(b) after swapping the reference and primary objects. A subscript “ h ” is used to indicate that two objects are of the same height. For example, $b\{E_h\}a$ indicates that b is located east to a , and they are of the same height.

Investigating all possible cases is inefficient, and will cause exponential increase in run-time when the number of objects is increased. Fortunately, we only need to check the first case due to the following Theorem 2.

Theorem 2 uses the notation of D_x^1 to represent the D_x obtained by replacing $a\{W\}b$ with $b\{SE,E,NE\}a$. Similarly, D_x^2 , D_x^3 and D_x^4 represent the D_x obtained by replacing $a\{W\}b$ with $b\{SE,E\}a$, $b\{E,NE\}a$ and $b\{E_h\}a$ respectively.

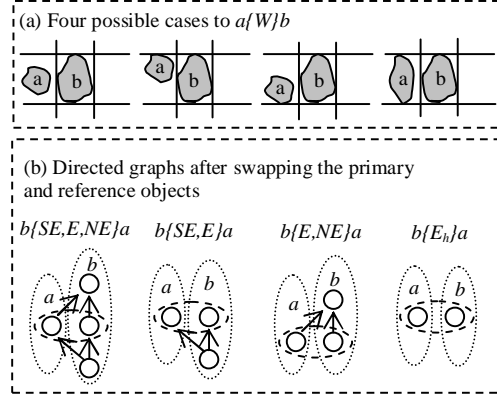


Figure 9. Four cases for $a\{W\}b$, and their corresponding directed graphs after swapping the primary and reference objects

Theorem 2: Given $a\{W\}b$ in D_x , if D_x^1 is inconsistent, then D_x^2 , D_x^3 and D_x^4 are all inconsistent.

Proof: We only need to prove that if one of D_x^2 , D_x^3 , and D_x^4 is consistent, D_x^1 is consistent too.

(1) Assuming that D_x^2 is consistent, we need to prove that D_x^1 is consistent too.

We extend the length of object b to the north while keeping object a unchanged. The direction relationship between a and b is changed to $b\{SE,E,NE\}a$. Then we extend other relevant objects according to the algorithm *Extension* in Figure 10. In the algorithm, Q represents a queue storing objects. p and q denote objects.

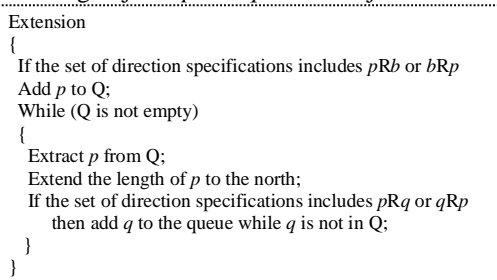


Figure 10. The algorithm *Extension*

Since we extend the length of every relevant object in the same proportion, the extension will not affect the consistency. Therefore, if D_x^2 is consistent, D_x^1 is consistent too.

(2) The other two sub-cases can be proved in a similar way. ■

In summary, whenever $a\{W\}b$ or $a\{E\}b$ is defined, we merge a and b through the following approach:

- Represent the primary object by a single node as illustrated in Figure 11(a).
- Represent the reference object by a directed graph as illustrated in Figure 11(b),
- Merge nodes a and b_{a2} .

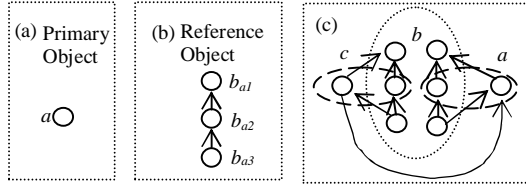


Figure 11. A correct merge

According to the above approach, the SGV illustrated in Figure 7 is revised to that in Figure 11(c), which denotes a consistent status on the set of direction specifications.

5. Connectivity constraints

In the previous sections, we introduced a graph model to visually represent definitive direction relationships. Based on the graph model, consistency checking is performed by analyzing the connectivity information. This section will analyze connectivity constraints among participating nodes. If an object is derived to locate totally or partially north/south to the object itself, the set of direction specifications is inconsistent. We will transform the above statement to connectivity constraints through the following theorems.

Theorem 3: (1) If a SGV contains a cycle, the set of direction specifications is inconsistent. (2) If (1) is true, only super-nodes or nodes in Partition 1 are in the cycle.

Proof:

(1) Assume that the set of direction specifications is consistent, and there is a cycle denoted as $a_1 a_2 \dots a_n$, where $a_1 = a_n$. We can deduce that $Inf_y(a_{i+1}) \geq Sup_y(a_i)$, where $1 \leq i < n$. Therefore, $Inf_y(a_n) \geq Sup_y(a_1) = Sup_y(a_n)$, which contradicts with the assumption.

(2) Assume that a node a , which is neither in Partition 1 nor inside a super-node.

- If $Level(a)=1$, a has no outgoing edge according to Theorem 1, which contradicts with the assumption.
- If $Level(a)=2$, there is only one outgoing edge connecting to node b , where $Level(b) = 1$. If b is in a super-node, a has no incoming edge; if b is not in a super-node, b has no outgoing edge. Either case contradicts with the assumption.
- If $Level(a)=3$, a has no incoming edge according to Theorem 1, which contradicts with the assumption. ■

The connectivity constraint described in Theorem 3 reflects the inconsistency that an object is derived to

locate totally north/south to the object itself. The connectivity constraint illustrated in Theorem 4 reveals the inconsistency that one object is specified to locate partially north/south to the object itself.

Theorem 4: There is a path denoted as a_i, \dots, a_j , where $a_i \in V_i$, $a_j \in V_j$, partition $D_i = (V_i, E_i)$ and partition $D_j = (V_j, E_j)$ belong to the same component graph representing object a ,

- (1) If $Level(a_i) = 1$, D_x is inconsistent; or
- (2) If $Level(a_j) = Depth(D_j)$, D_x is inconsistent.

Proof:

(1) We prove the statement with the first condition by considering the condition in two cases.

CASE 1: Assume that there is no node a_m except a_i in D_i that $Level(a_m) = 1$.

According to the assumption, the sub-region represented by a_i must contain the highest point of object a , i.e. $Sup_y(a_i) = Sup_y(a)$.

If a_i falls in Partition 1, it is obvious that:

$$Inf_y(a_j) > Sup_y(a_i) = Sup_y(a) \text{ ---(1)}$$

If a_i does not fall in Partition 1, a_i must be inside a super-node, and connect to other nodes through the super-node obtained by merging a_i with a reference object b . We can derive relationship (1) from the following relationships:

$$Inf_y(a_j) > Sup_y(b) \text{ ---(2)}$$

$$Sup_y(b) > Sup_y(a_i) \text{ ---(3)}$$

Relationship (2) is obvious, and relationship (3) is true; otherwise, b will divide a_i further into two nodes. Since a_j represents a sub-region of object a , $Sup_y(a) \geq Sup_y(a_j) > Inf_y(a_j)$, which contradicts with the assumption.

CASE 2: Assume that there is another node a_m in D_j in addition to a_i such that $Level(a_m) = 1$. If the region represented by a_j contains the highest point, it has already been proved based on the above proof. Otherwise, assume that a_m contains the highest point, i.e. $Sup_y(a_m) = Sup_y(a)$. Since D_i contains at least two nodes, a_m and a_i , D_i cannot fall in Partition 1. It follows that a_i must be inside a super-node, which also contains node a_m and the reference object b as illustrated in Figure 12. Therefore, the following relationships are satisfied:

$$Inf_y(a_j) \geq Sup_y(b)$$

$$Sup_y(b) \geq Sup_y(a_m)$$

$Inf_y(a_j) \geq Sup_y(a_m) = Sup_y(a)$ is derived, which contradicts with the assumption.

(2) The proof for the second condition is similar to that for the first one. ■

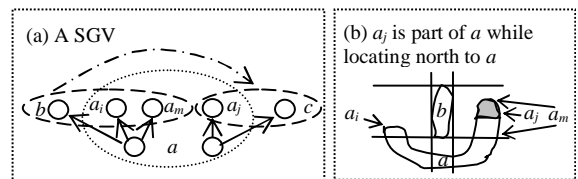


Figure 12. Illustration about Theorem 4

6. A consistency checking algorithm

As described previously, we construct a directed graph from a set of definitive direction specifications. Having proven Theorems 3 and 4, we can detect inconsistency on a set of direction specifications by analyzing the connectivity of the participating nodes. In order to retrieve the connectivity information between nodes, we perform a breadth-first search starting from every node, and store the connectivity information in a matrix. This algorithm proceeds as follows:

1. Convert a set of definitive direction specifications into a directed graph as described in Section 3.
2. Refine the directed graph as described in Section 4.
3. Perform a breadth-first search for connectivity information starting from every node.
4. Analyze the connectivity of the participating nodes as described in Section 5, and deduce whether the set of direction specifications is “consistent”.

Given n objects, the number of the direction specifications is $O(n^2)$. Since at most 5 nodes for a primary object and one node for a reference object can be generated, there are $O(n^2)$ nodes in a SGV. It will take $O(n^2)$ time to complete a breadth-first search. Since there are $O(n^2)$ iterations, the third step will take totally $O(n^4)$ time. The last step will take $O(n^4)$ since there are $O(n^4)$ pairs of nodes. Consequently, our algorithm will perform in $O(n^4)$ time.

Theorem 5: The checking algorithm deduced from Theorems 3 and 4 is complete.

Proof: Different partitions belonging to a component graph reflect different divisions of a single object. If a set of definitive direction specifications is inconsistent, at least one object a cannot be defined in a valid minimal bounding box, i.e. $Sup_y(a) \leq Inf_y(a)$. If we translate the above case to the connectivity in a SGV, there exists a path between two nodes, one of which represents the sub-region covering either the highest point or the lowest point, and the other represents a sub-region belonging to the same object. Therefore, it is sufficient to check the connectivity constraints described in Theorems 3 and 4. ■

Theorem 6: If a set of direction specifications is consistent in both vertical and horizontal directions, it is consistent.

Proof: It is obvious since the vertical and horizontal directions are orthogonal. ■

7. Related work

In general, there are three models of spatial knowledge: quantitative, qualitative and hybrid [12]. Quantitative models are relatively well known in robotics and vision

etc. However, the abstraction capability of such a model is usually weak. A qualitative model makes only as many distinctions as necessary [6] so that the granularity is dependent on applications. A hybrid model is of a mixture of quantitative and qualitative approaches or a further division of qualitative regions into more detailed levels. Since the qualitative approach is considered to be analogue to how human represents and reasons about commonsense knowledge, it gains popularity recently.

In qualitative spatial reasoning, Frank discussed two approaches about directions [13]. One is a cone-shaped approach, which divides the reasoning space into eight regions around the reference point; the other is a projection-based approach. According to Ligozat [3], assuming representing directions among objects through a binary constraint network, the consistency problem can be viewed as a spatial version of a general problem of temporal reasoning, which has been extensively studied.

However, objects have different shapes, and it is often inaccurate to approximate an object by a point. Goyal and Egenhofer [9] presented a spatial model, which only approximates the reference object while using the exact shape of the primary object. Skiadopoulos and Koubarakis [11] gave an algorithm for consistency checking through constraint-solving. The algorithm is implemented in three steps: initially, it introduces the atomic cardinal direction constraints and additional constraints into a set; then the algorithm detects the consistency of the set of constraints; if the set is consistent, it finally considers the set-union constraints. This algorithm is performed in $O(n^5)$ time.

8. Conclusion

Spatial knowledge is an important aspect of commonsense, and consistency checking is a significant research field in qualitative spatial reasoning. This paper has presented a graph model for representing direction relationships and effective spatial reasoning. In the graph model, nodes represent the regions of objects, and directed edges indicate the direction relationships between objects. Based on the model, we present an efficient algorithm that performs consistency checking in $O(n^4)$ time by analyzing the connectivity of participating nodes.

Consistency checking has potential applications. For example, in spatial databases, it can be used to detect inconsistent spatial queries and prune the search space. In particular, we are interested in applying the approach to visual programming and design of visual languages. Spatial and structural (abstract) information are two essential aspects to visual languages. A graph grammar formalism [14] has been proposed to intuitively integrate spatial specifications with structural specifications using visual notations. Maintain spatial consistency inside a grammar rule is trivial. It, however, is challenging to

detect inconsistency in a spatial configuration defined by a set of grammar rules. Since this issue can be addressed as detecting inconsistency in a given set of direction relationships, it is natural to apply our approach to verify spatial configurations defined through spatial graph grammars.

Our future research will focus on the following two directions:

- This paper concentrated on connected regions without a hole. However, disconnected regions or regions with holes are useful to model various geographical situations [11]. We will extend our approach to accommodate such regions by decomposing them into connected regions.
- The ability of using graph grammars to explicitly specify the physical layout apart from the logical structure of a graph is extremely useful in many applications, such as graph layout [15], spatial databases, multimedia design, and pattern recognition. We extended a context-sensitive graph grammar formalism, i.e. *Reserved Graph Grammar (RGG)* [16] [17], with the capability of spatial specifications [14]. It is desirable to provide a mechanism to automatically analyze the consistency on a set of user-provided spatial specifications. Integrating our consistency checking algorithm with the extended RGG is planned in the next stage of our research framework.

References:

- [1] E. Clementini and P.D. Felice, "A Comparison of Methods for Representing Topological Relations", *Information Sciences*, Vol.3, 1995, pp. 149-178.
- [2] A. Islı, L. M. Cabedo, T. Barkowsky, and R. Moratz, "A Topological Calculus for Cartographic Entities", *Spatial Cognition II - Integrating Abstract Theories, Empirical Studies, Formal Models, and Practical Applications*, Berlin: Springer, 2000, pp. 225-238.
- [3] G. Ligozat, "Reasoning about Cardinal Directions", *Journal of Visual Languages and Computing*, Vol.9, 1998, pp. 23-44.
- [4] E. Clementini, P. D. Felice, and D. Hernández, "Qualitative Representation of Positional Information", *Artificial Intelligence*, Vol.95, Sep. 1997, pp. 215-444.
- [5] A. G. Cohn, "A Hierarchical Representation of Qualitative Shape Based on Connection and Convexity", *Proc. the International Conference on Spatial Information Theory, COSIT'95, LNCS 988*, 1995, pp. 311-326.
- [6] D. Hernández and E. Jungert, "Special Section on Qualitative Spatial Reasoning", Guest Editors' Introduction, *Journal of Visual Languages and Computing*, Vol.9, 1998, pp. 1-3.
- [7] D. Papadias, *Relation-based Representation of Spatial Knowledge*, Ph.D. Thesis, National Technical University of Athens, 1994.
- [8] C. Freksa, "Using Orientation Information for Qualitative Spatial Reasoning", *Proc. International Conference GIS - From Space to Territory: Theories and Methods of Spatio-Temporal Reasoning*, 1992, pp. 162-178.
- [9] R. Goyal and M. J. Egenhofer, "Cardinal Directions between Extended Spatial Objects", *IEEE Transactions on Knowledge and Data Engineering*, (in press), Available at <http://www.spatial.maine.edu/~max/RJ36.html>.
- [10] S. Skiadopoulos and M. Koubarakis, "Composing Cardinal Direction Relations", *Proc. 7th International Symposium on Advances in Spatial and Temporal Databases*, 2000, pp. 299-317.
- [11] S. Skiadopoulos and M. Koubarakis, "Consistency Checking for Qualitative Spatial Reasoning with Cardinal Directions", *Proc. 8th International Conference on Principles and Practice of Constraint Programming*, 2002, pp. 341-356.
- [12] D. Hernández and A. Mukerjee, "Tutorial on Representation of Spatial Knowledge", *Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, Montreal, Canada, 1995.
- [13] A. U. Frank, "Qualitative Spatial Reasoning with Cardinal Directions", *Journal of Visual Languages and Computing*, Vol.3, 1992, pp. 343-371.
- [14] M. K. Qiu, G. L. Song, J. Kong, and K. Zhang, "Spatial Graph Grammars for Web Information Transformation", *Proc. VL2003 2003 IEEE Symposium on Visual/Multimedia Languages*, Auckland, New Zealand, 28-31 October 2003.
- [15] F. J. Brandenburg, "Designing Graph Drawings by Layout Graph Grammars", *Proc. the DIMACS International Workshop on Graph Drawing, LNCS 894*, 1995, pp. 416-428.
- [16] D. Q. Zhang, *Generation of Visual Programming Languages*, Ph.D. Thesis, Macquarie University, 1998.
- [17] D. Q. Zhang, K. Zhang, and J. Cao, "A Context-sensitive Graph Grammar Formalism for the Specification of Visual Languages", *The Computer Journal*, Vol.44, Issue.3, 2001, pp. 187-200.