# SEMANTIC INTEGRITY CONSTRAINT VIOLATIONS CHECK FOR SPATIAL DATABASE UPDATING

K. P. Udagepola [a,] *, L. Xiang [b], A.W. Wijeratne [c], Y. Xiaozong [a]

[a] School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, P.R.China-(kalum,yangxz)@ftcl.hit.edu.cn
[b] School of Computer Science and Technology, Harbin Engineering University, Harbin, 150001, P.R.China-leexiang@hrbeu.edu.cn
[c] School of Science, Department of Mathematics, Harbin Institute of Technology, Harbin 150001, P.R.China-awijeratne@sab.ac.lk

**Commission II**

**KEY WORDS:** SSR$^O$-Tree, integrity constraints, spatial join query, spatial database, repository, spatial relations

## ABSTRACT:

An efficient spatial data structure in a GIS system for database updating is required in order to minimising of spatial constraint violations and timesaving. An automated constraint checking procedure has been introduced to perform constraint violations check at compiling time before updating the database. Formal definitions of spatial data types were used in attempt to formulate novel equations and architectures to detect constraint violations and framework for spatial repository. A data structure called Semantic Spatial Outlier R-Tree (SSR$^O$-Tree) was proposed for the Semantic Integrity Constraints Checking System to improve the functionality of the proposed method. The R-Tree or its variants have been widely-used data structures for this purpose and which are based on a heuristic optimization but unable to perform semantic spatial join queries at database updating. An experiment was conducted using actual spatial data and results revealed that the performance of SSR$^O$-Tree is notably superior to the R*-Tree and R$^O$-Tree for conducting semantic spatial join queries.

## 1. INTRODUCTION

Due to the growing demand for Geographic data and Geographic Information Systems (GIS) in engineering and industrial purposes, the necessity of an efficient and error free technique is vital to update current spatial data within a reasonable time frame. Operations such as spatial data inserting, modifying and deleting should guarantee that maintain the integrity constraints rules while database updating. Therefore, it is required to have a technique that performs constraint violations checking before updating the database. Integrity constraints such as domain constraints, key and relationship structure constraints, and general semantic integrity constraints frequently occur at Spatial Database (SD) updating (Duboisset, 2005; Schneider, 2004; Udagepola, 2006a; Stell, 2004). The procedure during the compilation and before the updating takes place, the R-tree is a widely-used data structures in GIS systems that allows efficient accesses to the spatial data in a GIS system, which is based on a heuristic optimization (Guttman, 1984; Kame, 1993; Leutenegger, 2000; Roussopoulos, 1985). The functionality of R*-Tree is further improved to support semantic spatial join queries (SSJQ) and we denote the proposed technique as Semantic Spatial Outlier R-Tree *(*SSR$^O$-Tree*)* to handle SSJQ in SD updating.

The paper is organized as follows. Section 2 discusses updating the spatial objects. Section 3 defines topological relationships using set topology theory. Section 4 introduces the semantic constraints rules, together with the algebra and also discuss role of sub classes of feature layers. Section 5 describes SSR$^O$-Tree for SSJQ. Section 6 describes implementation of semantic integrity constraints in SDs and repository for Semantic Constraint Rules. In section 7, the SSJQ performance results are discussed using SSR$^O$-Tree, R*-Tree and R$^O$-Tree and the result of an experimental performance comparison reported and finally concludes this paper.

## 2. UPDATING THE SPATIAL OBJECTS

### 2.1 Creation of new objects

When a new object is created in the semantic class, the method of object creation will be called to notify all related semantic spatial objects class. The ideal setting will be taken place when new objects are created in the “considered-class” and their corresponding class objects are automatically computed through suitable methods and stored in the related class. However, before updating all related classes, user needs to provide approval to avoid possible problems in the operation. Therefore, this derivation can not be fully automated, and given the fact that this research is concerned with operations under the properties of real world semantic spatial data objects with regard to semantic spatial rules.

### 2.2 Modification to existing objects

An object can be modified in several ways:
*Attribute updates*: All attribute updates must be made in the consider class, which is the primary representation. These updates will then be automatically propagated to the other dependent class by update propagation mechanisms.
*Object resized*: If the geometric shape of the area objects changes, the system will notify the user to update the

---

* Corresponding author. Tel.:+86-451-86400863; fax: +86-451-86414093.

corresponding object related to the modified area object, if it detects no valid spatial relationship between the two.

*Object moved*: If an object under consideration is moved to a new position, the related object will be moved together with the area object to maintain the consistency.

*Objects merged or split*: This will result in the creation of new object(s), and will therefore be treated as outlined above.

### 2.3 Deletion of objects

If the objects under consideration to be deleted are not independent of other objects, all dependent objects must be properly maintained after the deletion is completed. Thus deleting an object will cause the system to automatically delete the related objects if and only if they are fully dependant classes and the part which has to be kept in the map after the deletion should represent the real world situation (Twumasi, 2002).

Here, the consideration is given for the above three classes of updates and the objective of this research is to generate rules that make the process more efficient. Further, semantic rules, efficient access method for spatial data under semantic spatial objects operations are considered which are more powerful in achieving the task than existing systems.

## 3. TOPOLOGICAL RELATIONSHIP

Spatial relationships (Stell, 2004) can specially be used to formulate consistency constraints for enforcing consistency of SD. Within the geographic context, topologic relations and other spatial relationships are fundamentally important to the definition of semantic integrity constraint. The topological relations between arbitrary objects based on concepts of algebraic and set theory (Egnhofer, 1990; Egnhofer, 1991). These relations are preserved under a group of transformations, such as scaling, rotation, and translation. The model for binary topological relations have considered the possible intersection of boundary, interior and exterior of objects which is called 9-intersection model (9IM). With these intersections it will be possible to formulate consistency rules as different groups of relations.

| Basic Relations | |
|---|---|
| disjoint (A,B) | $(\exists A, B \in \{line, point, region\})$<br>$(\neg\exists x_i(x_i \in A^{\Theta} \wedge x_i \in B^{\Theta}) \wedge$<br>$\neg\exists x_j(x_j \in A^{\Theta} \wedge x_j \in B^{-}) \wedge$<br>$\neg\exists x_l(x_l \in A^{O} \wedge x_l \in B^{\Theta}) \wedge$<br>$\neg\exists x_m(x_m \in A^{O} \wedge x_m \in B^{O})) =$<br>$\begin{bmatrix} 0 & 0 & - \\ 0 & 0 & - \\ - & - & - \end{bmatrix}$ |
| meet (A,B) | $(\exists A, B \in \{line, point, region\})$<br>$(\exists x_i(x_i \in A^{\Theta} \wedge x_i \in B^{\Theta}) \wedge$<br>$(\exists x_j(x_j \in A^{\Theta} \wedge x_j \in B^{O}) \vee$<br>$\exists x_l(x_l \in A^{O} \wedge x_l \in B^{\Theta}) \vee$<br>$\exists x_m(x_m \in A^{O} \wedge x_m \in B^{O}))) =$<br>$\begin{bmatrix} 0 & 1 & - \\ - & - & - \\ - & - & - \end{bmatrix} \vee \begin{bmatrix} 0 & - & - \\ 1 & - & - \\ - & - & - \end{bmatrix} \vee \begin{bmatrix} 0 & - & - \\ - & 1 & - \\ - & - & - \end{bmatrix}$ |
| inside(A,B) | $(\exists A, B \in \{line, point, region\})$<br>$(\exists x_i(x_i \in A^{\Theta} \wedge x_i \in B^{\Theta}) \wedge$<br>$\neg\exists x_k(x_k \in A^{\Theta} \wedge x_k \in B^{-}) \wedge$<br>$\neg\exists x_m(x_m \in A^{O} \wedge x_m \in B^{O}) \wedge$<br>$\exists x_r(x_r \in A^{-} \wedge x_r \in B^{\Theta})) =$<br>$\begin{bmatrix} 1 & - & 0 \\ - & 0 & - \\ 1 & - & - \end{bmatrix}$ |
| contains (A,B) | $(\exists A, B \in \{line, point, region\})$<br>$(\exists x_i(x_i \in A^{\Theta} \wedge x_i \in B^{\Theta}) \wedge$<br>$\exists x_k(x_k \in A^{\Theta} \wedge x_k \in B^{-}) \wedge$<br>$\neg\exists x_m(x_m \in A^{O} \wedge x_m \in B^{O}) \wedge$<br>$\exists x_r(x_r \in A^{-} \wedge x_r \in B^{\Theta})) =$<br>$\begin{bmatrix} 1 & - & 1 \\ - & 0 & - \\ 1 & - & - \end{bmatrix}$ |
| overlap (A,B) | $(\exists A, B \in \{line, point, region\})$<br>$(\exists x_i(x_i \in A^{\Theta} \wedge x_i \in B^{\Theta}) \wedge$<br>$\exists x_k(x_k \in A^{\Theta} \wedge x_k \in B^{-}) \wedge$<br>$\exists x_r(x_r \in A^{-} \wedge x_r \in B^{\Theta})) =$<br>$\begin{bmatrix} 1 & - & 1 \\ - & - & - \\ 1 & - & - \end{bmatrix}$ |
| coverdby (A,B) | $(\exists A, B \in \{line, point, region\})$<br>$(\exists x_i(x_i \in A^{\Theta} \wedge x_i \in B^{\Theta}) \wedge$<br>$\neg\exists x_k(x_k \in A^{\Theta} \wedge x_k \in B^{-}) \wedge$<br>$\exists x_m(x_m \in A^{O} \wedge x_m \in B^{O}) \wedge$<br>$\exists x_r(x_r \in A^{-} \wedge x_r \in B^{\Theta})) =$<br>$\begin{bmatrix} 1 & - & 0 \\ - & 1 & - \\ 1 & - & - \end{bmatrix}$ |
| covers (A,B) | $(\exists A, B \in \{line, point, region\})$<br>$(\exists x_i(x_i \in A^{\Theta} \wedge x_i \in B^{\Theta}) \wedge$<br>$\exists x_k(x_k \in A^{\Theta} \wedge x_k \in B^{-}) \wedge$<br>$\exists x_m(x_m \in A^{O} \wedge x_m \in B^{O}) \wedge$<br>$\neg\exists x_r(x_r \in A^{-} \wedge x_r \in B^{\Theta})) =$<br>$\begin{bmatrix} 1 & - & 1 \\ - & 1 & - \\ 0 & - & - \end{bmatrix}$ |
| equal(A,B) | $(\exists A, B \in \{line, point, region\})$<br>$\neg\exists x_j(x_j \in A^{\Theta} \wedge x_j \in B^{O}) \wedge$<br>$\neg\exists x_k(x_k \in A^{\Theta} \wedge x_k \in B^{-}) \wedge$<br>$\neg\exists x_l(x_l \in A^{O} \wedge x_l \in B^{\Theta}) \wedge$<br>$\neg\exists x_n(x_n \in A^{O} \wedge x_n \in B^{-}) \wedge$<br>$\neg\exists x_r(x_r \in A^{-} \wedge x_r \in B^{\Theta}) \wedge$<br>$\neg\exists x_s(x_s \in A^{-} \wedge x_s \in B^{O}) =$<br>$\begin{bmatrix} - & 0 & 0 \\ 0 & - & 0 \\ 0 & 0 & - \end{bmatrix}$ |

Table 1. Topological relationships

The 4- Intersections model (4IM) can not distinguish all the detailed relationships between any two arbitrary objects with different dimensions, such as line-region and line-line, etc. in a 2-dimensional space and 9IM can represents all spatial relationships better than 4IM. The 9IM has 512 ($2^9$ and 3 x 3 matrix) distinguished knowing the fact that they are topologically invariant. However, all the 512 relationships are not topologically valid in a 2-dimensional space when real world situations are considered (Egnhofer, 1990).

For an example binary topological relations between two objects A and B are defined in terms of the nine intersections of A's boundary $(A^{O})$, A's interior $(A^{\Theta})$, A's exterior $(A^{-})$ with the B's boundary $(B^{O})$, B's interior $(B^{\Theta})$ and B' exterior $(B^{-})$. Each object A and B can be a point, a line or a region in simple or complex spatial data.

$$\mathrm{9IM}_{(A,B)} = \begin{array}{cc} & \begin{array}{ccc} B^{\Theta} & B^{O} & B^{-} \end{array} \\ \begin{array}{c} A^{\Theta} \\ A^{O} \\ A^{-} \end{array} & \begin{bmatrix} i & j & k \\ l & m & n \\ r & s & t \end{bmatrix} \end{array} \quad (1)$$

A generalization of the aforementioned eight topological predicates to complex regions can be found in (Schneider, 2001); their implementation is described in (Schneider, 2003) and (Schneider, 2006). In our research, we are interested in building the common model of purely topological predicates on simple and complex objects because it is not possible to treat objects only as simple where complex object types appear in real world situation. Table 1 presents the spatial operators, and their characteristic schema is presented in 2D space. Those rules are formulated by using a set of matrices in 9IM. The base object and reference object is either complex or simple spatial object.

Above defined predicates can be used to formulate functions, which can detect the property of two semantic spatial objects. At any time, the system can activate the function and check the validity under topological relationship. It gives flexibility at a certain level to the research. These functions can be stored in a repository that is to be activated at once when the update process begins.

## 4. SEMANTIC CONSTRAINT RULES

All layers in a topographic database are classified into sub classes of features and each feature is identified by a feature code. Semantic integrity constraints are defined between two geographical objects and the topological relation is the main part of the constraint rules. In the spatial domain, they are mostly a group of forbidden relationships between pairs of objects. It is easier to define a case that should not happen than to define a case that must exist. The formula for the constraint rule will be as follows:

$$CR_{(BO,O)} = \{ \overset{K}{\underset{i=1}{P}} (BO, R_i, O, S) \} \quad (2)$$

Where P = Intersection ( $\cap$ );
R$_i$ = one of relation specification between two objects (either spatial relations (Table 1) or alphanumeric with join spatial relation or real world semantic spatial object relationships)
K = the number of relations between *BO* and *O*
BO= a base object (New)
O = a set of reference object (Exiting) to be found
S = a validity specification. The specification can be true or false
CR$_{(BO,O)}$ = total TSIC rules for given spatial two object type(*BO* and *O*) in SD

Finally, we can identify above concept be taken as association of two geographical objects, a topological relation (spatial operator) between them and a specification.

According equation (2), we can give some of the major possible constraint rules under real world situation as follows.

a) *Semantic constraint between polygon/polygon objects*

- High raised buildings cannot *intersect* the roads (topological relation=*disjoint*).
- Buildings cannot be *Inside* the water bodies unless it is a special building (topological relation=*In*).
- Any two land use parcels can not *overlap* (topological relation=*overlap*)
- Parking place must have an *access to* road (topological relation=*meet*)
- All built-up areas must have road *access* (topological relation=*meet*).
- Roads should be *adjoined with* other road types (within the theme) (topological relation=*meet*).
- Road can *intersect* the built-up area, if the intersection is part of the underneath object code topological relation=*overlap*).
- Dockyard must be *adjoined with* the water body (topological relation=*meet*).
- Landing stage must be *adjoined with* the water body (topological relation=*meet*).

b) *Semantic constraint between line/polygon objects*

- Ditches can not *cross* the buildings or built-up area (topological relation=*disjoint*)
- Ditches can not *cross* roads unless the intersect portion has the underneath object code (topological relation=*disjoint*).
- Walls can not *cross* buildings or built-up area (topological relation=*disjoint*).
- Railroad can not *cross* buildings or built-up area unless the intersection is underneath/above object code (topological relation=*disjoint*).
- Footpath and street can not *cross* the highway unless it is below/above the other object (topological relation=*overlap*).
- Bridge must be *part of* road or water-bodies (topological relation=*meet*).
- Shoreline or riverbank cannot *cross* the road (topological relation=*disjoint*).
- Shoreline or riverbank cannot *cross* the buildings/built-up area (topological relation=*disjoint*).
- Lock door for ships is *part of* water body (topological relation=*In*).

c) *Semantic constraint between point/polygon objects*

- Police office, post office, municipality office, religious building, railway station (buildings symbol) should be *within* the buildings or built-up area (topological relation=*In*).
- Milestone pole should be *within* 6 meters of the roads.
- Signpost should be *within* 6 meters of the roads (topological relation=*disjoint*).
- Culvert must be *part of* road/railway/river (topological relation=*In*).
- Dam must be *adjoining* or a *part of* the water bodies (topological relation=*meet*).
- A monument should not be *inside* the road (topological relation=*In*).
- Buildings symbol can not be *inside* the road (topological relation=*In*).

d) *Semantic constraint between line/line objects*

- Railway line should not *cross* the ditch unless the intersection is an underneath object code (topological relation=*disjoint*).
- Contour line cannot *cross* another contour line (topological relation=*disjoint*).

*e) Semantic constraint between point/line objects*

- Railway milepost should be *within* 6 meters of the railway line (topological relation=*disjoint*).
- High tension posts are *part of* high tension line should intersect on common point (topological relation=*In*).

*f) Semantic constraint between point / point objects*

- Tree can not *equal* the symbol *for* building (topological relation=*equal*).
- Symbols for building cannot *intersect* with itself (topological relation=*disjoint*).

*"Intersect"*, "*Access* to", "*Overlap*", and "*Cross*" are the constraint operators which should be unique for all the groups of relations. It should be relatively easy to define the relations using spatial relations called spatial operators (Udagepola, 2006b) rather than selecting different names for constraint operators. The objects associated with the constraint could be further combined with the sub classes of the feature because a subclass in the definition of constraints will make the model more practicable for the user.

$$CR_{(BO,O)} = \underset{i=1}{\overset{K}{P}}(BO\ ,M_i, R_i, O, N_i, S) \qquad (3)$$

Where P= Intersection ( $\cap$ )
$R_i$ = one of relation specification between two objects (either spatial relations (Table 1) or alphanumeric with join spatial relation)
K = the number of relations between subclass of *BO* and subclass of *O*
BO= a base object (New)
O = a set of reference object (Exiting) to be found
$M_i$= $i^{th}$ subclass of a subclass ( New object)
$N_i$ = $i^{th}$ subclass of a subclass (Exiting object)
$S_i$ = a validity specification. The specification can be one of the following
(a) Forbidden (b) Unless: condition (c) Allowed (d) At least n times (e) At most n times (e) Exactly *n* times
$CR_{(BO,O)}$ = total Semantic s\Integrity Constraint (TSIC) rules for given spatial two objects(BO and O) in SD

The specification *forbidden* is the most interesting and usable one. The specification *unless* will be followed by a *condition*, where the feature code of the object will differ from the features. For example if a build-up area intersects with the road, the intersected feature will have the feature code with the last digit of 2 or 9 (last digit 2 = object is below the other object and 9 = object is above the other object). This type of hypothesis is too strong (costly). TSIC rule functions (see eq. (2) and (3)) are defined using this specification for the end-users to describe topological situations where they do not want to see it in their database.

The model built here is significantly superior to the currently available model (Servige, 2000) because it provides more usable interface while grouping common topological relations sharing attributes into subsets. Such subsets have been built in each group of relations. For example, buildings can not be *inside* the roads. In this case three constraints are needed to be tested, such as an *overlap*, *covered_by* and *equal.*

## 5. SSR$^{O}$-TREE

A spatial data structure with built-in semantic spatial information can answer semantic spatial join queries more effectively. In addition, it is required to identify outlier semantic spatial objects. For this purpose, a spatial data structure with built-in semantic information with facility for detection of outlier objects, called a SSR$^{O}$-Tree, is proposed. Figure 1 briefs the operation (SSR$^{O}$-Tree: Semantic R-Tree (Chen, 2003) + Outlier R*-Tree (R$^{O}$-Tree) (Xia, 2005) → Semantic spatial-object outlier R-Tree (SSR$^{O}$-Tree) (Udagepola, 2007).

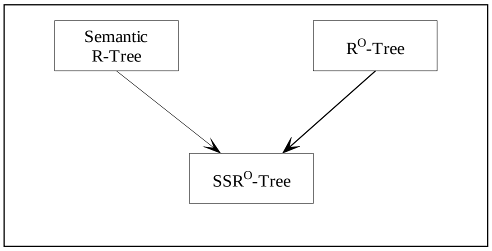

Figure 1. The taxonomy of SSR$^{O}$-Tree

However, for a SD, more specific research on data structures is needed to categorize data into subclasses that guarantees the proper functioning and good performance. Both can be achieved by semantic spatial categorizing technique and the outlier object detection technique. Thereafter, the R-Tree structure can be changed according to those techniques.

Without such built-in semantic information, a spatial data structure has difficulty in answering a semantic query such as "*Find all Buildings on the Roads within relationship Overlaps*" efficiently. Searching the SSR$^{O}$-Tree will get all the objects on the Roads within relationship Overlaps and further processing is then required to get the desired objects (i.e., the Buildings) that answers to this query. With built-in semantic information, some sub-trees containing unrelated information can be pruned to make semantic searching quite efficient. In our design, the semantic spatial object class (Example: semantic spatial object road with a subclass highway, street cycle path…etc.) is used to build a part of SSR$^{O}$-Tree and another part considers detection of outlier objects within the class. The algorithm used is based on R*-Tree because it has well re-inserting capabilities and minimum overlapping. For each node, its semantic information is assorted and organizes the semantic spatial objects that detect outlier objects (outlier identification) and divides small minimum bounding rectangle (MBR*)*. After the outlier is divided, "search" function will visit the few remaining spatial objects before the result is generated. The decision on which nodes to visit is made based on the evaluation of spatial predicates, in addition the MBRs are sorted on the *x* or *y* coordinates either one of the corners of the rectangle. Sorting MBRs is similar to the method proposed by Roussopoulos and other (Roussopoulos, 1985). In each class, it is generally depicted on level one and the rest of the levels are shown as same category node, which satisfies *m* and *M*. Finally, leaf node has categorized objects that make the scan very simple. But there might be some underflow nodes (less than *M/2* children). Since only a fixed number of elements exist in one semantic subset (usually this number is small), there might be only a few underflow nodes. The outlier identification is integrated throughout the construction/ update of the SSR$^{O}$-Tree such as reinsertion process, overflow/ underflow handing and splitting etc.

### 5.1 Quality and Gain/Loss:

**Definition 1**(Zhang, 2004): Given a rectangle *r* with width *w* and height *h*, the quality of the rectangle is defined as

$$Q(r) = \frac{1}{w \times h} \times \left( \frac{min\{w,h\}}{max\{w,h\}} \right)^{\alpha} \tag{4}$$

Where $\alpha \in$ [0, 1] is a constant.

But $\left( \frac{min\{w,h\}}{max\{w,h\}} \right)^{\alpha} \leq 1$, therefore Q(r) is dependent only $\left( \frac{1}{w \times h} \right)$. But $(w \times h)$ is area of the rectangle. Then the small rectangle is obtained good quality.

Assume w $\geq$h;

where w and h call respectively width and height.

Let $ratio = \frac{w}{h}$ , Therefore $area = \frac{ratio^{-\alpha}}{Q}$

**Definition 2**: If a rectangle $r_1$ is shrunk to $r_2$ ($r_1$ spatially contains $r_2$), the gain is defined as

$$G(r_1, r_2) = 1 - \frac{Q(r_1)}{Q(r_2)} \tag{5}$$

Therefore if $r_1$ is expanded to $r_2$, then the loss is created at the $\frac{Q(r_1)}{Q(r_2)} > 1$.

Now, the threshold ( $\delta$ ) can be defined because the new rectangle dose not need to be very close to previous one. Therefore Gain can be limited to very small value (eg. 0.001).

**Theorem 1**: A successful new rectangle needs satisfy $G(r_1, r_2) > \delta$ .

This research also uses four lines method (Zhang, 2004) to build new MBR. The four lines and their properties can be defined according figure 7.

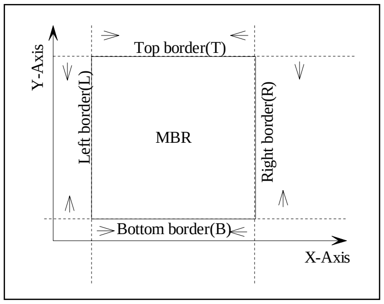

Figure 2. Four lines on a MBR

The algorithm to handle the outlier object detection is shown in algorithm 1. It is adapted from the $R^O$-Tree's greedy-pick-p algorithm.

*Algorithm 1:* pseudo-code of *Greedy-pick-p* Algorithm

**Greedy-pick-p** *($S \in SpatialObjects$, p,m: integer): MBR*

```
1. For each category
2. Build the border structure of initial MBR.
3. MBR(S)=M;
4. P =0 ;.
5. while {
6. $g_{max}$ = 0;
7. $B_{max}$ = LEFT
8. $k_{max}$ = 0;
9. For each border /* LEFT,BOTTOM,RIGHT,TOP*/
10. For each k /* k $\in$ {1,..m} such that there exist k more un
       removed levels for B and the number of objects in these k
       levels plus |P| is no larger than p*/
11. Compute the g (gain) per removed object; /* The gain
       defined in equation (5) */
12. If g > gmax then
13. gmax = g;
14. $B_{max}$ = B;
15. $k_{max}$ = k;
16. end if/* end step 12*/
17. end for/* end step 10*/
18. end for/* end step 9*/
19. If $g_{max}$ = 0 then stop;
20. Adjust new MBR according border $B_{max}$ and add the
       removed objects to P (outlier object list)
21. }/* end step 1*/
22. Endfor /* end step 5*/
23. End /* end Greedy-pick-p */
```

## 6. IMPLEMENTING REPOSITORY FOR CONSTRAINT RULES

A repository can be implement either internally by building a set of custom data structures and files for storing the repository data by using programming language or externally by using existing DBMS software to store the repository data (Cockcroft, 2004). The first option is used by this research. Although the second option is the best way to reduce the amount of work required to implement the repository, it is impossible to fulfill all requirements of users. This is the reason of why the commercial database developers are keeping away from these types of works. The repository can also be attached with user defined rules to the database. Figure 3 shows a context diagram of repository model.

The repository is active at data entry, ensuring that constraints on data are not violated, unless for legitimate reasons in which case a log of such violations are stored in the repository. If the spatial relationships between the objects are explicitly stored, the spatial queries can be easily processed. But to store the information it requires large amount of disk space and the maintenance is also very much costly.

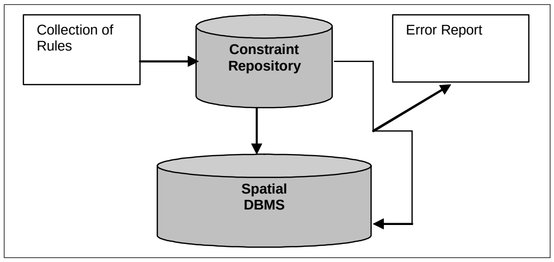

Figure 3. Constraint Repository

**Definition 3.** $\forall$ *Feature layer, Repository ={Name of the Layers, Topological relationships, Specification}*

$\exists$ *Specification* $\in$ *{forbidden, allowed, unless, Allowed, At least once, At most once}*

Instead of storing the relationship among objects, it is more convenient to compute them. These integrity constraints can be maintained by a Constraint Repository (CR), which stores data. Through the use of CR, constraint rules concerning between a pair of spatial objects can be stored and imposed at the updating process. In repository, user defined rules are also attached to the database. The repository was organized in relational database and implemented in Microsoft Access. In the implemented stage the topographic data is in the Oracle database and the CR is used to store the semantic and user defined rules. A user interface was created in "MapObjects" with Visual Basic for checking the constraint rule process. Data manipulation process involves getting the data from database and updating the database after checking the inconsistencies of the new data.

## 7. EXPERIMENTAL RESULTS

We performed an experiment for SSJQ performances of the SSR$^{O}$-Tree, R*-Tree and R$^{O}$-Tree. Especially, on the spatial joint query, we also compared the improved R*-Tree with them. The R*-Tree wass implemented in C++. Our experiment was performed using real datasets, which was acquired from the R*-Tree. All the data structures and algorithms were performed on Acer Laptop with a 1.73-GHz processor Intel Centrino processor (LaptopC), Acer Laptop with a 1.73-GHz processor Intel Mobile processor(LaptopM) and HP Desktop PC with a 2.40-GHz (Desktop), running WinXP Professional.

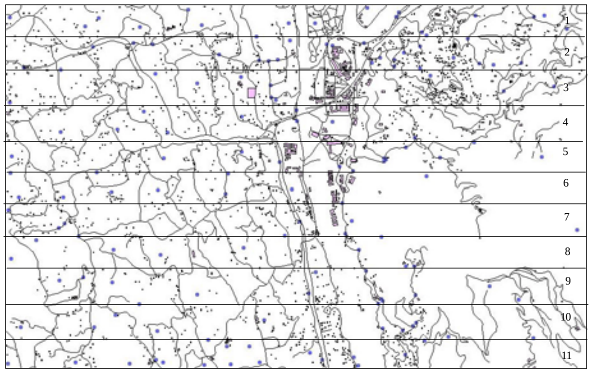

Figure 4. The digital map partitions 11 areas

We used various real spatial datasets in our experiment, which are points, lines, polylines and regions. The tested data represents a map of the Kandy city in Sri Lanka consisting of 15,868 spatial objects (Figure 4). Each group of spatial object was experimented at 11 different spatial areas (spatial area is proportional to the number of object) and results are taken by repeating the each test five times.

The SSR$^{O}$-Tree was implemented and an experiment was conducted to demonstrate the capability and usefulness of the proposed semantic approach, applied to GIS semantic queries. We compared a SSR$^{O}$-Tree with an R*-Tree, R$^{O}$-Tree that uses quadratic split algorithm with the three computers. The quadratic split algorithm is chosen because there is no essential performance gain resulting from the linear split algorithm (Cockcroft, 1997).

The response time of SSJQ to detect violations is used as performance criteria. For the SSJQs, an experiment was conducted to show how well the system is able to answer spatial-join using the SSR$^{O}$-Tree (See Figure 5: $Y$-axis is response time and the $X$-axis is the number of spatial objects). The linear regression analysis was performed to compare the performance of the system where the fitted equations were significant at the probability level of 0.0001 ($Y = 0.000203X$, $Y = 0.00140X$ and $Y = 0.000842X$ for SSR$^{O}$-Tree, R*-Tree and R$^{O}$-Tree respectively). Then, 95% confidence intervals were calculated for the fitted regression equations when the number of objects is equal to 15868. The confidence intervals show that the minimum and maximum time that would take at 95% probability level for violation detection by the three approaches. It observed that the confidence level for SSR$^{O}$-Tree (3.0627, 3.3716) has no overlapping with the other two ((21.055, 23.458) and (12.634, 14.076) for R*-Tree and R$^{O}$-Tree respectively).

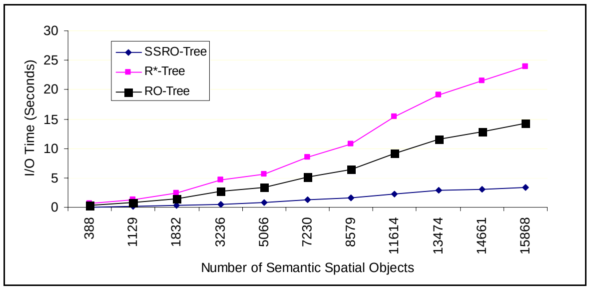

Figure 5. The response time of semantic spatial join queries

Thus, for queries with large outlier objects, the SSR$^{O}$-Tree outperforms the R*-tree and the R$^{O}$-Tree. The experimental results show that the SSR$^{O}$-Tree is an efficient method for answering SSJQs. As expected, the performance gain of the SSR$^{O}$-Tree over an R*-tree in answering spatial join is statistically significant too. In many SD updates systems, the support for spatial join is more desirable since this kind of query can accelerate update information to the SD. Hence, a SSR$^{O}$-Tree is more suitable in those situations.

## 8. CONCLUSION

In this paper, we define precisely and unambiguously characteristics relevant to semantic constraints rules and provide a formal definition of spatial relationships based on these characteristics. In addition, we have presented a framework for the specification of SSR$^{O}$-Tree for sematic spatial join query that can be used to violation detection of integrity constraints on spatial database. An experiment was conducted to compare the performances between novel SSR$^{O}$-Tree with R*-Tree and R$^{O}$-Tree structures about semantic constraint violations checking. The experimental results show that the SSR$^{O}$-Tree performs better than the R*-trees and R$^{O}$-Tree for sematic spatial join query which is statistically significant too. To achieve this, we plan to introduce software for the Integrity Constraint Checking for spatial database updating.

## ACKNOWLEDGMENTS

Real data of Kandy city was provided by the Sri Lanka survey department of Sri Lanka. This research was Supported by the High Technology Research and Development Program of China (No. 2006AA01A103) and the National Natural Science Foundation of China (No. 60503015). The authors extend sincere gratitude to Prof. Zuo Decheng, Prof. Wu Zhibo and Dr. Heyshan Mendis for their valuable comments.

## REFERENCES

Cockcroft, S.(1997): A taxonomy of spatial data integrity constraints. *Geoinformatca*, 1(4), pp. 327-343.

Cockcroft, S.(2004): The design and implementation of a repository for the management of spatial data integrity constraints, *GeoInformatica* , 8(1), pp. 49-69.

Chen, S., Wang, X., Rishe, N., and Weiss, M.A.W. (2003): A Web Based spatial data access system using semantic R-Trees, *Elsevier Science*.

Duboisset, M., Pinet, F., Kang, M. and Schneider, M. (2005): Precise Modeling and Verification of Topological Integrity Constraints in Spatial Databases: From an Expressive Power Study to Code Generation Principles. *ER 2005: 24th International Conference on Conceptual Modeling*, Klagenfurt, Austria, pp. 465-482.

Egenhofer, M. J., and Herring, J. (1990): Categorizing binary topological relations between regions, lines, and points in geographic databases, Tech. rep. 90-12, National Center for Geographic Information and Analysis, University of California, Santa Barbara.

Egenhofer, M. J., and Franzosa, R.(1991): Point-set topological spatial relations, *International Journal of Geographical Information Systems*, 5(2), pp. 161-174.

Guttman, A.(1984): R-tree: A Dynamic Index Structure for Spatial Search, *Proc. ACM SIGMOD International Conference*, pp. 47-57, ACM Press.

Kame, I., and Faloutsos, C. (1993): On Packing Rtrees. *CIKM*, pp. 490-499.

Leutenegger, S. T., and Lopez, M. A. (2000): The Effect of Buffering on the Performance of R-Trees, *International IEEE Transactions on Knowledge and Data Engineering*, 12(1), pp. 33-44.

Roussopoulos, N., and Leifker, D. (1985): Direct Spatial Search on Pictorial Database Using Packed R-trees, *Proceeding ACM SIGMOD*.

Servige, S., Ubeda, T., Puricelli, A. and Laurini R. (2000): A methodology for spatial consistency improvement of geographic databases, *GeoInformatica*, 14(1), pp.7-34.

Schneider, M., and Behr, T. (2004): Topological Relationships between complex spatial objects. Technical Report011, University of Florida, Department of computer & Information Science & Engineering.

Schneider, M. (2001): A design of topological predicates for complex crisp and fuzzy regions, *In Proceedings of the International Conference on Conceptual Modeling*, pp. 103-116.

Schneider, M. (2002): Implementing topological predicates for complex regions, *In Proceedings of the International Symposium on Spatial Data Handling*, pp. 313-328.

Schneider, M., and Behr, T. (2006): Topological Relationships between Complex, Spatial Objects, *ACM Transactions on Database Systems*, 31(1), pp. 39-81.

Stell, J.G. (2004): Part and Complement: Fundamental Concepts in Spatial Relations, *Annals of Mathematical and Artificial Intelligent*, 41(1), pp. 1-17.

Twumasi, B.O. (2002): Modelling spatial object behaviors in object-relational geodatabase MSc Thesis, ITC, The Netherlands.

Udagepola, K. P., Xiang, L., Wei, L. H., and Zong, Y. X. (2006a): Efficient Management of Spatial Databases by Data Consistency and Integrity Constraints**,** *WSEAS Transactions on Computers*, 5(2), pp. 447-454.

Udagepola, K. P., Xiang, L., Wei, L. H., and Zong, Y. X. (2006b): Methodology to Improve the Spatial Relations and Integrity Constraints in Topographic Databases, *WSEAS Transactions on Computers*, 5 (2), pp. 455-462.

Udagepola, K.P., Decheng, Z., Zhibo, W., and Xiaozong, Y. (2007): Semantic Spatial Objects Data Structure for Spatial Access Method, *XX. International Conference on Computer, Information, and Systems Science, and Engineering,* Spain.

Xia, T., and Zhang, D. (2005): Improving the R*-tree with Outlier Handing Techniques, *GIS'05*.

Zhang, D., and Xia, T. (2004): A Novel Improvement to the R*tree Spatial Index using Gain/Loss Metrics, *GIS'04*.