Computers, Environment and Urban Systems 45 (2014) 34–49

Contents lists available at ScienceDirect

Computers, Environment and Urban Systems

journal homepage: www.elsevier.com/locate/compenvurbsys

STModelViz: A 3D spatiotemporal GIS using a constraint-based approach

Jing Li [a,*], David W.S. Wong [b,c]

[a] Department of Geography and the Environment, University of Denver, Boettcher West, Room 120, 2050 E. Iliff Ave., Denver, CO 80208, USA
[b] Department of Geography and Geoinformation Science, George Mason University, 4400 University Drive, MS 6C3, Fairfax, VA 22030-4444, USA
[c] Department of Geography, University of Hong Kong, Pokfulam, Hong Kong

ARTICLE INFO

Article history:
Received 24 September 2012
Received in revised form 7 February 2014
Accepted 8 February 2014
Available online 12 March 2014

Keywords:
3D spatiotemporal system
Constraints
Data integrity
Query
Object Constraint Language

ABSTRACT

Advances in data acquisition techniques and model simulations produce increasing volume of 3D spatiotemporal data. However, existing systems provide limited capabilities to manage such data. This paper reports an effort to design and implement a prototype system for 3D spatiotemporal data. Due to the complexity of such data, the ability to verify their integrity is central to the data management system. We adopted a constraint-based approach which addresses data integrity explicitly. In the article, we define constraint conditions, formulate constraints using a formal language we have extended and evaluate constraints using enhanced computational algorithms. We focus on a set of relational integrity constraints pertaining to the spatial, temporal and spatiotemporal properties of 3D spatiotemporal data. We extended the Object Constraint Language (OCL) to handle spatiotemporal (ST) objects. ST–OCL is used to describe and record constraints. Constraints expressed by ST–OCL statements are evaluated by the enhanced algorithms to identify different topological relations between 3D spatiotemporal objects. The prototype system demonstrates how a constraint-based approach can be used to develop DBMS capabilities in managing 3D spatiotemporal objects. Using the dynamic repartitioning of airspace sectors as an application example, we show that the capabilities of the prototype to manage 3D spatiotemporal objects can be customized for specific domain applications.

1. Introduction

During the past decade or so, spatiotemporal data have attracted some attentions in the development of Database Management Systems (DBMS) and Geographic Information System (GIS) (e.g., Kim, Ryu, & Park, 2002; Sakr & Güting, 2011; Yu & Shaw, 2008; Zhao, Jin, Zhang, Wang, & Lin, 2011). As 3D spatiotemporal data become more available partly due to the advances in data acquisition techniques and model simulations, several systems have been developed to manage and analyze 3D spatiotemporal objects (e.g. Balovnev et al., 2004; Breunig, Butwilowski, Kuper, Golovko, & Thomsen, 2013). In developing these systems, theories and concepts of 2D and 3D spatiotemporal GIS were also formulated.

However, only a limited number of theories for higher dimensional analysis are available (Yuan et al., 2010). Data models used for describing properties, relations and behaviors of 3D dynamic objects and the associated geometric and topological operations have not been well formulated yet. Thus, existing systems have significant functional deficiencies. For example, these systems support limited geometric and topological operations on 3D spatiotemporal objects (Breunig & Zlatanova, 2011). In addition, a systematic mechanism for maintaining the integrity of 3D spatiotemporal data is missing (Salehi, Bédard, Mostafavi, & Brodeur, 2011). Moreover, queries, which are critical to support advanced analyses, are limited to simple geometric types (Breunig et al., 2013). Breunig and Zlatanova (2011) suggested several key research areas, including temporal geometrical and topological operations in spatial database systems, spatiotemporal integrity checking, and pattern search.

The objective of this study is to explore the potentials of using a constraint-based approach to maintain data integrity in 3D spatiotemporal data. A related objective is to investigate how constraints can support the queries of 3D spatiotemporal data, serving as the foundation of spatiotemporal data analysis. Constraints, also known as filters or principles, are conditions that need to be satisfied during a modeling process (Kuper, Libkin, & Paredaens, 2000; Revesz, 2002). Constraints have been incorporated into data management systems to maintain data integrity and facilitate queries

* Corresponding author. Tel.: +1 303 871 4687; fax: +1 303 871 2201.
E-mail address: Jing.Li145@du.edu (J. Li).

http://dx.doi.org/10.1016/j.compenvurbsys.2014.02.002
0198-9715/

(e.g., Grumbach, Rigaux, & Segoufin, 2001; Lin, Zhang, Liu, & Gao, 2005; Louwsma, Zlatanova, van Lammeren, & van Oosterom, 2006). Constraints for 3D spatiotemporal objects have been discussed briefly, but have not been formulated and implemented yet (Salehi et al., 2011). Successfully formulating and implementing such constraints should advance the development of spatiotemporal GIS, which is an ongoing research topic (Goodchild, 2013). Therefore, our overall objective is to demonstrate the feasibility of using constraints in 3D spatiotemporal data management. The proposed constraint approach is implemented and integrated with a system called "SpatioTemporal Modeling and Visualization system (STModelViz)". Our contributions to the constraint-based approach in spatial data management include the followings:

- Implement a formal constraint language to model relational constraints for spatiotemporal objects.
- Extend and integrate a set of algorithms to identify spatial, temporal and spatiotemporal relations between 3D spatiotemporal objects and to evaluate relational constraints imposed onto these objects.
- Develop a 3D spatiotemporal prototype system with a constraint module that provides interfaces to manage constraints, control constraint enforcements and represent constraint evaluation results.

The rest of the paper proceeds as follows. In the next section (2), we briefly review recent work on using constraints in facilitating the development of spatiotemporal DBMS, highlighting the potentials, statuses and challenges. In Section 3, we discuss the design of the constraint-based framework in maintaining data integrity and supporting queries. We provide an overview of the prototype system in Section 4. In Section 5, we present an application of the prototype system in facilitating the repartitioning of airspace sectors. Conclusions and ideas for future work are discussed in Section 6.

2. Constraints in spatiotemporal DBMS

Using constraints to maintain data integrity and facilitate queries in spatial domain is not new. Practices from Computer Aided Design (CAD) have demonstrated that constraints are essential in the creation of 3D objects. Ma, Zhong, Tso, and Zhou (2004) proposed a constraint-based modeling approach which enforces geometric and topological requirements when designing 3D objects. Louwsma et al. (2006) developed a Virtual Reality (VR) system where multiple constraints were created and enforced to assist users placing new objects on the landscape precisely. Grumbach et al. (2001) describes an approach to query moving objects in 2D space when the trajectories of objects are stored using linear constraints.

The importance of constraints has been widely recognized in DBMS and geospatial applications in which constraints serve at least three functions: (1) describing objects, (2) maintaining data integrity and (3) facilitating queries (Kuper et al., 2000). Constraints can be used to describe the spatial information of objects. For example, the coordinates of every point on the boundary of a circle can be described by an equality constraint of $x^2 + y^2 = R^2$ (where (x, y) is a pair of coordinates and R is the radius of the circle) (Kuper et al., 2000). Thus, instead of storing the infinite coordinates that describe an object, using constraints to describe objects can improve the efficiency of storing and retrieving data from databases. Data integrity in spatial domain can be expressed as a set of constraints that defines the rules of constructing objects, nature of interactions among objects and operations that can be performed on objects (Ma et al., 2004). Enforcing these constraints ensures data consistency (e.g., Oracle). Query criteria can be expressed as constraints using formal constraint languages (e.g., Object Control Language – OCL, Richters & Gogolla, 1998). Solving the query is a constraint satisfaction problem that identifies objects meeting the conditions defined by the constraints. Constraint computing techniques, such as identifying constraints conflicting with each other, can be used to improve the efficiency of processing queries (Clementini, Sharma, & Egenhofer, 1994). In this paper, we explore the use of constraints in maintaining data integrity and facilitating the query process in 3D spatiotemporal systems, the latter two functions of constraints as mentioned above by Kuper et al. (2000).

Previous developments have demonstrated the effectiveness of constraints in maintaining data integrity and supporting queries in 2D spatiotemporal or 3D static cases (e.g., Arens, Stoter, & van Oosterom, 2005; Grumbach et al., 2001; Louwsma et al., 2006). Works on maintaining data integrity for 3D spatiotemporal objects have been very limited. For example, the GeoToolkit (Balovnev et al., 2004) provides some functions to verify if geological objects meet the constraint conditions. However, the temporal properties of objects are limited to discrete time points only. Literature about the toolkit is not clear about what types of constraints the toolkit can handle and how constraints are evaluated. Louwsma et al. (2006) described an approach to incorporate constraints in a geo-Virtual Reality system. The proposed approach was divided into three tasks: "(1) a classification and clarification of constraints, (2) a formal description using the Unified Modeling Language/Object Constraint Language (UML/OCL) and (3) implementation characteristics" (Louwsma et al., 2006, pg1). As classification of constraints has been widely discussed in the existing literature (e.g., Currim & Ram, 2012; Mas & Reinhardt, 2009; Salehi et al., 2011), we focus on the formalization of constraints using a formal modeling language and the implementation challenges in the context of 3D spatiotemporal data.

2.1. Specifying constraints with a formal modeling language

Warmer and Kleppe (2003) argued that in order to develop constraints to be implemented by a system, taxonomy classifying constraints should be developed, and a formal language is needed to record constraints efficiently and precisely. Louwsma et al. (2006) and Richters and Gogolla (1998) also supported similar arguments. Several taxonomies of constraints have been developed for geospatial applications (e.g., Belussi, Bertino, & Catania, 1997; Currim & Ram, 2012; Mas et al., 2009; Louwsma et al., 2006). The more comprehensive one was proposed by Salehi et al. (2011) in which integrity constraints are classified into spatial, temporal, and thematic and the combinations of these three categories. However, they did not describe the conditions under which how different types of constraints should be enforced when these constraints are imposed onto spatiotemporal objects. In addition, existing constraint taxonomies do not explicitly consider 3D dynamic objects. Our objectives here are to define the constraint conditions and to develop formal expressions of constraints for 3D spatiotemporal objects such that they can be implemented.

One of the most frequently adopted approaches to describe constraints formally is to use an OCL (Demuth & Hussmann, 1999). Using an OCL to specify constraints has the following advantages (Bejaoui, Pinet, Schneider, & Bédard, 2010). First, OCL provides declarative expressions of constraints. Second, OCL has been integrated with UML through which users can specify constraints at the modeling stage. Third, OCL statements can be interpreted by code compilers to be evaluated automatically. Spatial and temporal extensions of OCL have been developed to capture spatial *or* temporal constraints (Weder, 2009; Ziemann & Gogolla, 2003), but these OCL extensions are not capable of modeling constraints for spatiotemporal objects effectively. For example, Expressions 1, 2 and 3 below use spatial OCL, temporal OCL, and combined spatial