Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 1996 Proceedings

Americas Conference on Information Systems (AMCIS)

8-16-1996

A Model for Concurrent Heterogeneous Databases

Danilo Montesi
School of InformationSystems, University of East Anglia, dm@sys.uea.ac.uk

Jose Felix Costa
Departamento de Informatica, Universidade de Ciencias de Lisboa, fgc@di.fc.ul.pt

A Model for Concurrent Heterogeneous Databases

Danilo Montesi
School of Information Systems
University of East Anglia
Norwich NR4 7TJ, UK
dm@sys.uea.ac.uk
Jose Felix Costa
Departamento de Informatica
Universidade de Ciencias de Lisboa
Campo Grande C5, 1700 Lisboa, Portugal
fgc@di.fc.ul.pt
MIIORA94

Extended Abstract

During the seventies centralized databases were dominant. Technology and cultural corporate advance changed the focus to decentralized or distributed databases. Distributed databases have a single logical database that is physically distributed and supports just one data model and transaction language [4]. A federated database is a collection of concurrently interacting heterogeneous databases without an attempt to integrate them using a unified schema [7]. This is due to the proliferation of different database systems during the past three decades. It has created difficult problems arising from the need to access concurrent heterogeneous databases through a single data definition and data manipulation language designed under a single data model. There are two important points in the above vision: heterogeneity and concurrency.

Heterogeneity means different data models, transaction languages and schemas [11,13]. Hereafter, we focus on the logical level and does not consider physical heterogeneity about different lengths of strings, computers, operating systems and networks nor implementation issues. However, heterogeneity also means that two different databases, with the same data model can have different schemas (i.e. names to denotes different concepts such as attributes, relations and relations with different ariety) and operations [9 ,12]. For instance, the attributes telephone of relation ADDRESSBOOK in a database DB1 and the attribute Phone of relation AGENDA in a database DB2 are syntactically different but denote the same information. We refer to this problem as syntactic heterogeneity.

Consider instead another example. The attribute Mealcost in a relation RESTAURANT in a database DB1 that describes the cost of a meal without service charge and tax. The database DB2 has relation BOARDING with attribute Mealcost that describes the cost of a meal including service charge and tax. We refer to this problem as semantic heterogeneity. In this paper we consider syntactic heterogeneity of schemas and operations. Indeed, detecting semantic heterogeneity is a difficult problem even in the case of the same data model. This is due to the fact that current database schemas do not provide enough semantics to interpret data consistently. As we have said, we consider

heterogeneous databases with the same data model but with different schemas and operations. We choose as reference model the well known relational one that has a formal and solid foundations. Thus each database of the federation is a relational database, or according to [8] each database schema is first converted into an equivalent relational one, and the federated schema is constructed as a view of these relational schemas.

As we have said, another important point is concurrency. Cooperation among databases leads to several important problems. For instance, cooperation through attributes and/or operations sharing turn into concurrent interactions among databases. Concurrency in database context is (historically) related to a community of users that concurrently provide transactions to the database.

Concurrency among transactions allow to support several users that can provide transactions to the databases and many important results have been achieved in this direction [3]. Interacting databases, instead, lead to an orthogonal concurrency notion, that among the databases. Obviously, the two notions are addressing different problems. The former is addressing concurrent interactions of users with the database and the latter is addressing concurrent interactions of databases. Heterogeneity and concurrency issues raise some important questions. How can we homogenize these heterogeneous databases? How can we describe the concurrent behavior of the federated databases? These questions are hard to answer, mainly for the lack of a formal unifying model to answer both the questions. Heterogeneous databases can be handled through category theory [2]. Informally, a category is a pair: a collection of objects and a morphism over these objects. The objects are databases and the morphisms (function that preserve the structure of the objects) are used to handle syntactic heterogeneity through renaming. In addition category theory is also well known as a good conceptual tool to model concurrency [5]. Thus the important advantage of category theory is that it brings together heterogeneity and concurrency concepts in a unifying language. This is the main motivation to use category theory in our approach. Finally, categories reflect a computational model that can be used to build a software tool for aiding multidatabase designers to handle the above mentioned heterogeneity and concurrency problems. We note that concurrent interacting databases mirror interacting processes or objects, where cooperation is provided by means of operation sharing [6]. This view drives us to use as much as possible the concepts and models already developed in the concurrency area and extend them whenever the original is not sufficient for our context.

At this point we should clarify the difference between parallel and concurrent systems. Any sequential language, for instance Datalog, can be parallelized, and research is devoted to effective way of doing so [10]. Nevertheless, Datalog, whether executed sequentially or in parallel, should not be termed a concurrent language. According to Harel and Pnueli [9], we call concurrent system one whose components maintain continuous interaction at least with each other and possibly also with the environment.

The contribution of this paper is to provide a formal model for concurrent heterogeneous databases. We consider a simple, yet practically useful, notion of heterogeneity based on different database schema and operation names. The proposed model allows us to

homogenize different attributes, relations and operations and consider the concurrent interaction among databases. For our purpose a database is made of a static part, that is its schema and database instance and a dynamic part expressing the set of operations and the sequences of operations forming the legal transactions of the database. The generality of our approach is to consider an abstract database model, yet practically useful, based on the relational data model. Moreover, our approach to handle heterogeneity through categories turn into a practical one and this can be used to build tools to aid multidatabase designers. In addition, considering databases as extended processes we can take advantage of a large body of results and techniques already developed in the concurrency area [1].

References

[1] G.R. Andrews. Concurrent Programming - Principles and Practice. Benjamin Cummings, 1991.

[2] M. Barr and C. Wells. Category Theory for Computer Science. Prentice-Hall, 1989.

[3] P. Bernstein, V. Hadzilacos, and N. Goodman. Concurrency control and Recovery in Database system. Addison-Wesley, 1987.

[4] S. Ceri and G. Pelagatti. Distributed Database - Principles and Systems. MacGraw-Hill, 1984.

[5] F. Costa, A. Sernadas, and C. Sernadas. Object Inheritance Beyond Subtyping. Acta Informatica, 31:5--26, Springer-Verlag, Berlin, 1994.

[6] H-D. Ehrich, J. Goguen, and A. Sernadas. A categorial theory of objects as observed processes. In J. W. de Bakker, W.P. de Roever, and G. Rozenberg, editors, Proc. REX90/Workshop on Foundations of Object-Oriented languages, volume 489 of Lecture Notes in Computer Science, pages 203--228. Springer-Verlag, Berlin, 1991.

[7] G. Thomas et al. Heterogeneous distributed database systems for production use. ACM Computing Surveys, 22(3):237--266, September 1990.

[8] W. Litwin et al. Msql: A Multidatabase Language. Information Sciences, 49, June 1987.

[9] D. Harel and A. Pnueli. On the Development of Reactive Systems. In K.~R. Apt, editor, Logics and Models of Concurrent Systems, pages 477--498. Springer-Verlag, Berlin, 1985.

[10] W. Kim and J. Seo. Classifying Schematic and Data Heterogeneity in Multidatabase Systems. Computer, pages 12--17, December 1991.

[11] G. Lausen and J. Seib. Parallelizing Datalog Programs by Generalized Pivoting. In Proc. of the ACM Symposium on Principles of Database Systems, pages 241--251. ACM, New York, USA, 1991.

[12] W. Litwin, L. Mark, and N. Rossopulos. Interoperability of multiple autonomous databases. ACM Computing Surveys, 22(3):267--2293, September 1990.

[13] R.J. Miller, Y.E. Ioannidis, and R. Ramakrishnan. Schema Equivalence in Heterogenous Systems: Bridging Theory and Practice. In M. Jarke, J. Bubenko, and K. Jeffery, editors, Proc. Fourth Int'l Conf. on Extending Database Technology, pages 73--80, 1994.

[14] A. P. Sheth and J. A. Larson. Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases. ACM Computing Surveys, 22(3):183--236, September 1990.