Navigation Spaces for the Conceptual Analysis of Software Structure

Richard Cole and Peter Becker

School of Information Technology and Electrical Engineering (ITEE),
The University of Queensland,
QLD 4072, Australia
{pbecker, rcole}@itee.uq.edu.au

Abstract. Information technology of today is often concerned with information that is not only large in quantity but also complex in structure. Understanding this structure is important in many domains – many quantitative approaches such as data mining have been proposed to address this issue. This paper presents a conceptual approach based on Formal Concept Analysis. Using software source code as an example of a complex structure we present a framework for conceptually analysing relational structures. In our framework, a browsable space of sub-contexts is automatically derived from a database of relations augmented by a rule engine and schema information. Operations are provided for the user to navigate between sub-contexts. We demonstrate how the use of these operations can lead to quick identification of an area of software source code that establishes an unecessary dependency between software parts.

1 Introduction

Many modern information systems contain not only large amounts of information, but also complex structures. From operating on large but simple tables, information technology moved on to complex models represented by object-relational structures. This can be seen in the database world in the form of *Entity-Relationship Modeling* (ER) or *Object-Role Modeling* (ORM) [Hal96], it can be found in *Object-Oriented Programming* (OOP) and it is the basis of disciplines such as *Knowledge Engineering* (KE) [Smi96].

While modeling tools have been advanced over time to accommodate the complexity found, approaches to retrieve information from such structured sources are not much more sophisticated than they were decades ago. The main methods of querying are still based on relational joins and projections, combined with statistical methods to reduce the resulting information into sizes a human can handle. But while the result of the former usually produces tabular data too large to be understood in its entirety, the latter suffers from reducing information in an opaque manner.

Formal Concept Analysis (FCA) [GW99] is a successful technique of data analysis for data that fits well into the structure of a *many-valued context*. But while the structure of a many-valued context (attribute-value data) is common, it is still quite restricting, requiring the identification of a single object set and assuming functional dependencies.

B. Ganter and R. Godin (Eds.): ICFCA 2005, LNAI 3403, pp. 113–128, 2005.

In many examples significant steps were required in order to convert relational data into the form of a many-valued context.

To address this issue, this paper proposes other approaches to use FCA in combination with relationally structured data. This does not only apply to relational databases, but also knowledge bases in the sense of Conceptual Graphs or RDF. Other data can often be mapped into a suitable structure easily; in this paper we will demonstrate the techniques with an example analysing software source code.

Part of the problem posed by complex data is that the user has to be able to select views on the whole data, which can be displayed and understood easily. Toscana systems solve this problem by predefining a number of scales to be used. Since the scales are predefined it can be the case during analysis that desired scales have not yet been constructed, or that constructed scales are never used. If the number of scales is large it can also become difficult for the user to identify which scale meeting their analysis requirements. Furthermore structural changes on the input data can invalidate scales.

To overcome these problems we propose a different approach: defining and using a *navigation space*. The navigation space consists of points and functions. A point in the navigation space corresponds to a view on the data. The user moves from point to point within the space by selecting a function to apply to their current point. This mode of goal directed interaction is akin to browsing within hypertext documents. Similar to web browsers the user should be allowed to return to points visited earlier to follow different paths of investigation.

We demonstrate that FCA can be used as a powerful visualisation technique for querying complex relational structures by applying this notion of a navigation space to a concrete example in the domain of software engineering.

Before we start the discussion of our approaches, Section 2 introduces the general notion of relational structures and our problem domain: relational structures in software. We then outline our approach in Section 3, before introducing the notion of a navigation space in Section 4. Section 5 discusses our prototypical implementation, Section 6 gives an overview of related work and Section 7 concludes the paper with an outlook on further research.

2 Relational Structures

One of the most important building blocks of modern large-scale software systems is the notion of a *relational database*. A relational database can be defined as:

> **Relational Database:** [A] collection of data items organised as a set of formally-described tables from which data can be accessed or reassembled in many different ways without having to reorganise the database tables.[1]

Quite often the notion of a relational database is connected to *relational model* as defined by E.F.Codd in [Cod70], which is historically correct, although technically modern databases do not use pure relational algebra due to the introduction of the `NULL` value in SQL.

[1] `http://his.osu.edu/help/database/terms.cfm`