Simplifying the Formulation of a Wide Range of Object-Oriented Complex Queries

REDA ALHAJJ, American University of Sharjah

We present a model that simplifies the formulation of a wide range of complex, mainly selection-based, object-oriented queries, including linear recursive queries. They are complex because it is almost impossible for naive users to predict the formulation of their predicate expressions. Naive users are mainly decision makers who are most probably not computer professionals. Therefore, it is necessary to provide them with a straightforward and easy-to-handle approach to retrieve the information required for the decision making process. To achieve this, the definition of the selection operation and the predicate definition are adjusted to make it possible to have in the output only a subset of the objects from the actual result of a linear recursive query. Otherwise, it is infeasible to achieve the same output without an additional selection with a complicated predicate. We also define an operation that facilitates applying aggregate functions on objects. The two operations proved to be very useful and necessary to study the characteristics of trees and directed graphs. The presented model has been implemented as a part of our object-oriented database management system prototype.

One of the important reasons to have a query model is to answer queries with minimum effort from the user, i.e., the user does not need to write programs in order to retrieve required information from a given database. However, it is not acceptable to explicitly store in a database all the information necessary to answer queries. In other words, a query model should not be dedicated to retrieve information explicitly stored in a database; more important than that is to have a query model capable of retrieving implicit information. This is a capability of deductive database systems that conventional database systems do not support well if at all. An advanced query model is the one that provides its users with all the facilities and constructs required to retrieve the information explicitly stored in the database and to dig out the information implicitly present in the database. The power of a query model depends on how much implicit information it is capable to deduce from the stored database contents. Another important objective behind introducing query models is to simplify the coding of complex queries. It is not necessary for a query to be based on multiple classes in order to be identified complex; also single class based queries can be complex. Recursive queries and queries that return a subset of the transitive closure are two important categories of single-class based complex queries. As a result, a query model should be powerful enough to provide its naive users with the capability of expressing complex queries in a simple way to deduce from the database contents the information necessary and sufficient for their decision-making process.

A query model inherits power from the underlying data model. In case the underlying data model does not provide much power, as is the case with the conventional relational model, different operations are defined and existing operations are revised to improve the power of a query model. As object-oriented databases are concerned, the class-composition hierarchy provides implicit joins and hence many queries can be coded using the selection operation. Object-oriented databases are powerful and well accepted to model and handle advanced applications that are modeled in terms of
trees and directed graphs like geographical information systems applications. The selection operation must be powerful enough to benefit from the features of an object-oriented data model and serve end-users with the required information regardless of whether the information is explicitly or implicitly present in the database. It should support partial linear recursion as well as full linear recursion to provide traversing a whole tree or a part of a tree, a whole graph or a part of a graph. The location of the required subset within a tree or a graph should not affect the query formulation process. A user should feel comfortable to code a query to get a branch of a tree and the query model should hide from the user the complexity of reaching that branch. This could be illustrated by (recursively) executing a method for each subpart connected to a given part object. There may be cases where it is necessary to execute a method only on a subset of the subparts connected to a given part. Even if recursive queries are not special to object-oriented query languages only, query languages supporting advanced applications must include some form of recursion (Abiteboul and Bonner, 1995; Agrawal, 1988; Alhajj and Polat, 1996; Cluet and Delobel, 1994; Colby, 1989; Gardarin and Valduriez, 1992; Kim, Kim and Dale, 1989; Lee and Henschen, 1995; Hagen, 1994). Recursive queries are important for object-oriented databases because the schema of an object-oriented data model may contain cycles (by having the domain of an attribute being objects in a class). Also, recursive queries are of great interest to the application areas of object-oriented databases, e.g., engineering database applications including CAD/CAM and software engineering applications, which are modeled in terms of recursive definitions.

In this paper, we present a novel approach to extend the definition of the selection operation in an object-oriented query model to have it capable of handling a wide range of selection-based queries including linear recursive queries and queries that are not recursive but identified as complex queries. They are complex because it is almost impossible for naive users to predict the formulation of their predicate expressions. Naive users are mainly decision makers who are in general not computer professionals; hence it is necessary to provide them with a straightforward and easy-to-handle approach to retrieve the information required for the decision-making process. We also define an operation that facilitates the application of aggregate functions on objects in a given class. The two operations, selection and aggregation, are required to study the characteristics of links in information sources that are modeled in terms of trees and graphs. Our approach facilitates the formulation of linear recursive queries by merely allowing object variables to be bound to objects in the result. We also extend the format of the predicate expression allowed in the selection operation to achieve more expressiveness. This way, it becomes possible to express queries that return only a subset of the objects in the actual transitive closure. Such queries are not feasible otherwise without performing a complicated selection on the transitive closure. Different illustrative examples are presented.

The rest of the paper is organized as follows. A general overview of the related work is presented. Then a section is presented that includes a description of the model where the basic definitions and notation are presented. Our approach to simplify the coding of queries is discussed and some illustrating examples are presented. This is followed by the summary and conclusions.

Related Work

Recursion improves the power of a query language and many applications require the underlying query model to support recursion. Abiteboul and Bonner (1995) pointed out that it is important for a query model to support recursion; they only considered general recursive queries. As reported in the literature, conventional query models as well as advanced query models, including object-oriented query models involve recursion in one way or the other (Abiteboul and Bonner, 1995; Agrawal, 1988; Alhajj and Polat, 1996; Cluet and Delobel, 1994; Colby, 1989; Gardarin and Valduriez, 1992; Kim, Kim and Dale, 1989; Lee and Henschen, 1995; Hagen, 1994). Recursion provides the possibility to deduce some information, which is not explicitly present in the database.

Computing the transitive closure of a given database relation attracted the attention of many researchers because many of the recursive real-life problems are linear in nature. Agrawal discussed transitive closure queries within the realm of conventional relational databases (Agrawal and Kiernan, 1993; Agrawal, 1988). His approach is restricted to general transitive closure queries, where he presented some algorithms to compute the transitive closure. Lu, et al. (1987) also developed some algorithms to compute the transitive closure of database relations. Bancillon and Ramkrishnan introduced some approaches to deal with and process recursive queries (Bancillon and Ramkrishinan, 1986). The evaluation of recursive queries was also studied by Lee and Henschen (1995) within a rule-based deductive database query language. Colby (1989) considered the impact of a recursive algebra on query optimization within the nested relational model. Kim, et al. (1989), handled simple recursive queries in object-oriented databases. Their approach was implemented as a part of the query model of the ORION database management system. However, their query language is restricted to only selection-based queries. The AQUA data model (Subramanian, 1995) has an object algebra that has the capability of manipulating trees and graphs. However, the provided operators return either a complete tree or a branch of a tree (from the root to the leaves) and are not capable of returning a subset of a branch from a tree.

In this paper, we present a novel approach to extend the definition of the selection operation in an object-oriented query model to have it capable of handling a wide range of selection-based queries including linear recursive queries and queries that are not recursive but identified as complex queries. They are complex because it is almost impossible for naive users to predict the formulation of their predicate expressions. Naive users are mainly decision makers who are in general not computer professionals; hence it is necessary to provide them with a straightforward and easy-to-handle approach to retrieve the information required for the decision-making process. We also define an operation that facilitates the application of aggregate functions on objects in a given class. The two operations, selection and aggregation, are required to study the characteristics of links in information sources that are modeled in terms of trees and graphs. Our approach facilitates the formulation of linear recursive queries by merely allowing object variables to be bound to objects in the result. We also extend the format of the predicate expression allowed in the selection operation to achieve more expressiveness. This way, it becomes possible to express queries that return only a subset of the objects in the actual transitive closure. Such queries are not feasible otherwise without performing a complicated selection on the transitive closure. Different illustrative examples are presented.

The rest of the paper is organized as follows. A general overview of the related work is presented. Then a section is presented that includes a description of the model where the basic definitions and notation are presented. Our approach to simplify the coding of queries is discussed and some illustrating examples are presented. This is followed by the summary and conclusions.

Related Work

Recursion improves the power of a query language and many applications require the underlying query model to support recursion. Abiteboul and Bonner (1995) pointed out that it is important for a query model to support recursion; they only considered general recursive queries. As reported in the literature, conventional query models as well as advanced query models, including object-oriented query models involve recursion in one way or the other (Abiteboul and Bonner, 1995; Agrawal, 1988; Alhajj and Polat, 1996; Cluet and Delobel, 1994; Colby, 1989; Gardarin and Valduriez, 1992; Kim, Kim and Dale, 1989; Lee and Henschen, 1995; Hagen, 1994). Recursion provides the possibility to deduce some information, which is not explicitly present in the database.

Computing the transitive closure of a given database relation attracted the attention of many researchers because many of the recursive real-life problems are linear in nature. Agrawal discussed transitive closure queries within the realm of conventional relational databases (Agrawal and Kiernan, 1993; Agrawal, 1988). His approach is restricted to general transitive closure queries, where he presented some algorithms to compute the transitive closure. Lu, et al. (1987) also developed some algorithms to compute the transitive closure of database relations. Bancillon and Ramkrishnan introduced some approaches to deal with and process recursive queries (Bancillon and Ramkrishnan, 1986). The evaluation of recursive queries was also studied by Lee and Henschen (1995) within a rule-based deductive database query language. Colby (1989) considered the impact of a recursive algebra on query optimization within the nested relational model. Kim, et al. (1989), handled simple recursive queries in object-oriented databases. Their approach was implemented as a part of the query model of the ORION database management system. However, their query language is restricted to only selection-based queries. The AQUA data model (Subramanian, 1995) has an object algebra that has the capability of manipulating trees and graphs. However, the provided operators return either a complete tree or a branch of a tree (from the root to the leaves) and are not capable of returning a subset of a branch from a tree.