Query Languages for Graph Databases

Giorgio Ghelli and Luca Pardini

No Institute Given

1 Graph Databases

Graphs are mathematical objects used to represent collections equipped with a binary relation. Graphs arise in many application fields, such as social networks, transport networks, natural science. Graphs have trees and forests as important, and ubiquitous, special cases. Many typical graphs operations have been described and studied, such as connectivity analysis, analysis of distance, construction of spanning trees.

Graph databases are databases that represent such collections, possibly combined with data that has a non-graphical nature. In traditional database jargon, a graph is just a collection with a binary association; hence, any DBMS is able to manage a graph database. However, typical graph operations escape the expressive power of database languages, and in particular the expressive power of relational algebra or of first order logic.

Graph databases have been studied in the nineties [3, 6, 7, 5, 1, 2], in the early period of research on semi-structured databases, but the interest faded with the rise of XML and the move from graph-based to tree-based data models for semi-structured data. New interest in graph databases is now rising, partly because the old problems have not been solved yet, and partly because of new interest in the RDF data model [8, 4].

RDF (Resource Description Framework) is a formalism defined by the W3C to describe resources, and entities of any nature, through sets of triples ‘subject-predicate-object’. RDF allows entities to be directly described at the semantic level, has happens with the Object-Oriented data model, with no need to translate into a more rigid structure, as happens with the relational data model. With respect to the Object-Oriented data model, RDF is more flexible, since data can be described without a schema. Moreover, entities are referenced through explicit URIs, while the object-oriented data model uses opaque ‘object references’, which only make sense inside a closed scope. Hence, the former reference schema is much better suited to the World Wide Web. W3C defined a query language for RDF, SPARQL [9], but this language is deeply relational, hence its expressive power is quite limited.

2 The Open Problems

Query languages for graph databases present a wide set of open problems. One should first understand whether graph data can be queried using exact languages
or whether they rather need languages with a ranking semantics. Exact languages can be used to solve questions like ‘is X connected to Y?’, but ranking query languages seem better suited to questions like: ‘is there a shop of this kind which is near-enough to my current location?’ Another basic choice is between languages that are designed for data with a schema and languages for semi-structured data.

Even assuming a conservative choice, based on traditional exact database algebras enriched with recursion, the problem of query processing is wide open. Graph algorithms have been deeply studied, and the optimization of relational algebra has been studied as well, but no satisfactory approach is known for the efficient execution of queries that combine the two aspects.

View update is another interesting problem in this context. View update is defined as the problem of transferring an update that is defined on the result of a query back to the data source. This classical problem has been recently faced through a new approach, called ‘bidirectional programming’, where the query is defined through a rich language that, for any query operation, specifies enough information to define the inverse direction. In this field, we are currently studying the extension of this approach to graph query languages.

References