Path Querying on Graph Databases

Jelle Hellings
Hasselt University and transnational University of Limburg

Central in graph structured data are data-elements (the nodes) and the relations between data-elements (the edges). An example of information easily captured by a graph structure are road networks. Due to its simplicity the graph structured data model has a broad range of applications including applications in the social sciences, bioinformatics, the semantic web, geographical data, and in formal verification. Also frequently used data models such as the XML data model and the RDF data model are variants of the graph structured data model.

The broad applicability of the graph structured data model gives rise to database systems with full support of the graph data model. Key in such a database system is efficiency. Graph structured datasets such as social networks, biological information and formalized models can contain billions of nodes and even more edges. This order of scale in itself already provides many new challenges in handling with graph structured data.

The practical challenges faced with the graph structured data model asks for a broad understanding on the theoretical side of graph structured databases. A key subject therein is the study of query languages for graph structured data.

Graph structured data can be represented by traditional relational structures. For querying we thus can fall back to traditional first-order relational query languages. For first order logic we already have deep knowledge of the expressive power (what kind of queries the language can and cannot express) and also efficient query evaluation algorithms are known. Even with these good properties first-order logic is not always the best choice for querying graph structured data. The main reason is that first-order logic is restricted to 'local' reasoning: first-order logic cannot reason about graph-concepts such as paths and reachability.

Traditional extensions of the first-order query languages such as logics with recursion and the (monadic) second-order logics only partially lift the restrictions of first-order logic. These extensions do allow some reasoning on paths and reachability; but they do so in an non-intuitive way.

In practice we see a large collection of application-specific query languages. Examples are XPath and XQuery for querying XML data sets; modal logics such as CTL* used in formal verification; and several regular expression based languages used for querying RDF data sets. None of these languages seems to be applicable as a general query language for graph structured data due to the specific nature of these languages.

It remains interesting, however, to understand what is obtained when first order logic is extended with path querying primitives but not much else. Concretely we propose to study what we call *Walk Logic*: the extension of first-order logic with explicit quantifiers over the walks in a graph and over the positions on these walks. Thereby a walk is a non-empty sequence of nodes connected by edges.

Walk Logic is not only intended as a query language; it is also intended as a yardstick to understand the expressiveness of path queries on graphs. Indeed, we will compare the expressiveness of Walk Logic with a number of established graph query languages, and will see that doing this exercise raises various natural questions about the expressive power of such languages.