Chapter 11
An Overview of Graph Indexing and Querying Techniques

Sherif Sakr
University of New South Wales, Australia

Ghazi Al-Naymat
University of Tabuk, Saudi Arabia

ABSTRACT

Recently, there has been a lot of interest in the application of graphs in different domains. Graphs have been widely used for data modeling in different application domains such as: chemical compounds, protein networks, social networks and Semantic Web. Given a query graph, the task of retrieving related graphs as a result of the query from a large graph database is a key issue in any graph-based application. This has raised a crucial need for efficient graph indexing and querying techniques. In this chapter, we provide an overview of different techniques for indexing and querying graph databases. An overview of several proposals of graph query language is also given. Finally, we provide a set of guidelines for future research directions.

INTRODUCTION

The field of graph databases and graph query processing has received a lot of attention due to the constantly increasing usage of graph data structure for representing data in different domains such as: chemical compounds (Klinger & Austin, 2005), multimedia databases (Lee et al., 2005), social networks (Cai et al., 2005), protein networks (Huan et al., 2004) and semantic web (Manola & Miller, 2004). To effectively understand and utilize any collection of graphs, a graph database that efficiently supports elementary querying mechanisms is crucially required. Hence, determining graph database members which constitute the answer set of a graph query q from a large graph database is a key performance issue in all graph-based applications. A primary challenge in computing the answers of graph queries is that pair-wise comparisons of graphs are usually really hard problems. For example, subgraph isomorphism is known to be NP-complete (Garey & Johnson,

DOI: 10.4018/978-1-4666-3604-0.ch011

1979). A naive approach to compute the answer set of a graph query q is to perform a sequential scan on the graph database and to check whether each graph database member satisfies the conditions of q or not. However, the graph database can be very large which makes the sequential scan over the database impracticable. Thus, finding an efficient search technique is immensely important due to the combined costs of pair-wise comparisons and the increasing size of modern graph databases. It is apparent that the success of any graph database application is directly dependent on the efficiency of the graph indexing and query processing mechanisms. Recently, there are many techniques that have been proposed to tackle these problems. This chapter gives an overview of different techniques of indexing and querying graph databases and classifies them according to their target graph query types and their indexing strategy.

The rest of the chapter is organized as follows. The Preliminary section introduces preliminaries of graph databases and graph query processing. In Section (Subgraph Query Processing), a classification of the approaches of subgraph query processing problem and their index structures is given while the section (Supergraph Query Processing) focuses on the approaches for resolving the supergraph query processing problem. Section (Graph Similarity Queries) discusses the approach of approximate graph matching queries. Section (Graph Query Languages) gives an overview of several proposals of graph query languages. Finally, Section (Discussion and Conclusions) concludes the chapter and provides some suggestions for possible future research directions on the subject.

PRELIMINARIES

In this section, we introduce the basic terminologies used in this chapter and give the formal definition of graph querying problems.

Graph Data Structure

Graphs are very powerful modeling tool. They are used to model complicated structures and schemeless data. In graph data structures, *vertices* and edges represent the *entities* and the relationships between them respectively. The attributes associated with these entities and relationships are called labels. A graph database D is defined as a collection of member graphs $D = \{g_1, g_2, \ldots, g_n\}$ where each member graph database member g_i is denoted as $(V, E, L_v, L_e, F_v, F_e)$ where V is the set of vertices; $E \subseteq V * V$ is the set of edges joining two distinct vertices; L_v is the set of vertex labels; L_e is the set of edge labels; F_v is a function $V \rightarrow L_v$ that assigns labels to vertices and F_e is a function $E \rightarrow L_e$ that assigns labels to edges. In general, graph data structures can be classified according to the direction of their edges into two main classes:

- **Directed-labeled graphs:** Such as XML, RDF and traffic networks.
- **Undirected-labeled graphs:** Such as social networks and chemical compounds.

In principle, there are two main types of graph databases. The first type consists of few numbers of very large graphs such as the Web graph and social networks (*non-transactional graph databases*). The second type consists of a large set of small graphs such as chemical compounds and biological pathways (*transactional graph databases*). The main focus of this chapter is on giving an overview of the efficient indexing and querying mechanisms on the second type of graph databases.