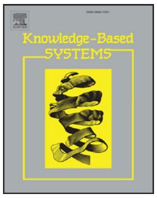
Knowledge-Based Systems 000 (2017) 1–13

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys



Personalized graph pattern matching via limited simulation

Ruihuan Du, Jiannan Yang, Yongzhi Cao*, Hanpin Wang

Key Laboratory of High Confidence Software Technologies (MOE), School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China

ARTICLE INFO

Article history:
Received 19 January 2017
Revised 5 November 2017
Accepted 8 November 2017
Available online xxx

Keywords:
Graph pattern matching
Graph simulation
Limited simulation
Limited bisimulation

ABSTRACT

It is well known that graph simulation and bisimulation can capture the semantics of graphs, described by the type and attribute of the nodes and edges. Graph pattern matching via simulation has been widely used in a variety of applications such as social network. Recently, some works have investigated the personality of graph simulation. Intuitively, the personality allows each node or edge to have different strengths of matching conditions, which are typically restricted by the similarity of nodes or edges in graphs. Motivated by the notion of *k*-limited bisimilarity, which was proposed by Milner to measure the similarity of nodes in the neighbouring subgraphs, and some examples in practice, we employ the notion of *k*-limited similarity, a weaker version of *k*-limited bisimilarity, to revisit graph pattern matching in this paper. After establishing the framework for the graph pattern matching via limited simulation, we give an efficient algorithm to compute the maximum match for limited simulation and analyze its computational complexity. To evaluate the algorithm, a group of experiments are conducted for limited simulation and graph simulation. As an extension of limited simulation, we also consider the notion of limited bisimulation for graph pattern matching, and unfortunately, we find that the graph pattern matching via limited bisimulation is NP-hard.

1. Introduction

Graph pattern matching plays an increasingly important role in a variety of applications, e.g., artificial intelligence, electronics, information sciences, graph theory, machine learning, and data mining [1–3]. Given a graph as the pattern and another graph as the data graph, the problem of graph pattern matching is to find the matches for the pattern in the data graph. Subgraph isomorphism is typically viewed as the structure based graph pattern matching [1,4]. However, it is NP-complete [5]. Moreover, such a method fails to capture the semantics of graphs, which is represented by the type and attribute of the nodes and edges. The notion of bisimulation, proposed by Milner [6], has been widely applied in concurrency systems, and it is able to capture the semantics of graphs. However, the graph pattern matching via bisimulation is also NP-complete [7]. Fortunately, graph simulation [6], which is close to but weaker than bisimulation, can capture the semantics of graphs to a certain extent and can be solved in polynomial time [8].

Graph simulation has been utilized to find meaningful matches in data graphs or optimize the structures of graph queries for database [9–12]. There are works that extend graph simulation to face specific requirements. For example, bounded simulation maps edges in a pattern to paths in a data graph [13], rather than edge-to-edge mapping. Fan et al. [14] added regular expressions into a pattern graph in order to restrict the label sequence of a path in a data graph. They also studied the incremental algorithms for graph simulation, bounded simulation, and subgraph isomorphism, and gave some important results about incremental graph pattern matching. Different from that graph simulation preserves the same label and child relationship for a pattern graph in its matches, Ma et al. [15] proposed the dual simulation to preserve the parent relationship. The graph simulation has also been adopted in distributed graph pattern matching [16,17]. For example, Ma et el. [16] investigated the distributed algorithms and optimization techniques for graph simulation. They gave an analysis of several distributed algorithms for graph pattern matching. The possibility and impossibility of distributed graph simulation have also been studied in [18]. Some adapted versions of graph simulation have been used to find matches for patterns in social network [10].

Graph pattern matching is often based on the similarity of nodes, which is an important notion in a great number of applications about information network [19]. In particular, Milner [20] proposed the notion of *k*-limited bisimilarity (also known as *k*-bisimilarity). The notion has been widely adopted in concurrent systems and logics [21–25]. To decide the *k*-limited bisimilarity of two nodes, only nodes and edges that are close to the nodes will

* Corresponding author.
E-mail addresses: duruihuan@pku.edu.cn (R. Du), yjn19920627@pku.edu.cn (J. Yang), caoyz@pku.edu.cn (Y. Cao), whpxhy@pku.edu.cn (H. Wang).

https://doi.org/10.1016/j.knosys.2017.11.008
0950-7051/© 2017 Elsevier B.V. All rights reserved.

be considered, rather than the complete graph. There are also a number of works that apply the k-limited bisimilarity to node set partitioning and summary for graph structured documents [26–29]. For example, Luo et al. [26] used the notion of k-limited bisimilarity to partition the node set, by restricting that two nodes are in the same block iff they are k-limited bisimilar to each other. Several distributed algorithms are presented in their work to compute the blocks.

It seems that the notion of k-limited bisimilarity can be naturally used to measure the similarity of influence range of individuals in a social network and the influenced area of news in an information network. But there is no such work that uses the notion in graph pattern matching. In this paper, we revisit the graph pattern matching with limited bisimulation, which is based on k-limited bisimilarity. Unfortunately, we find that graph pattern matching via limited bisimulation is NP-hard. Thus, the paper mainly focuses on the notion of limited simulation, which is based on k-limited similarity, a weaker version of k-limited bisimilarity. Each node in a pattern graph that we consider has a weight, which might be a natural number k or infinity. Such a weight maps each node in the pattern to a neighbouring subgraph centered at the node consisting of all nodes and edges that are close to the node. Intuitively, it can be used to measure the sphere of authority of a manager and the infected area of an epidemic disease. To decide whether a node v is a match for a node u in a pattern via limited simulation, the neighbouring subgraph centered at v in the data graph is considered, instead of the complete data graph. Since the weight of each node might be different, the matching approach to k-limited similarity or k-limited bisimilarity is personalized.

For limited simulation, not only the k-limited similarity is adopted, but the edge label is considered. The type of relationship between nodes is often a very important property of nodes. We illustrate the motivation of graph pattern matching via limited simulation by examples in practice, with which we show that limited simulation is able to find more meaningful matches than graph simulation and seems applicable to graphs in many problem domains such as social network and information network. With the fast development of the technology of Big Data and Artificial Intelligence, the personalized features of nodes in pattern graphs become very important in graph-based applications [30], which however, are difficult to be captured by the traditional approaches of graph pattern matching. For example, consider a start-up company which wants to recruit a team to develop new products, where both the professional skills and the cooperation among the members are expected in the team. In such cases, limited simulation can capture the influence of each member, which is crucial for teamwork. We also prove that graph simulation is a special case of limited simulation. In order to compute the matches of a pattern graph in a data graph, we study the properties of k-limited similarity and limited simulation. We provide an efficient algorithm to compute the maximum match and analyze its computational complexity, where the algorithm shows that the computation can be completed in polynomial time. Since limited simulation is not much harder than graph simulation, the method is feasible in the applications of graph simulation with little additional cost. To evaluate the algorithm, we conduct an experimental study with real-life data. The experiments show that the algorithm is efficient, and that the weight of node significantly influences the size of match. On the other hand, due to the NP-hardness of limited bisimulation, we do not develop an algorithm for it in the paper.

In the literature, there are works that add restrictions on matching conditions or area to be considered in a data graph for graph simulation. For instance, Ma et al. [15] proposed a strong simulation to restrict the locality of the match in the data graph. Cao et al. [31] introduced the boundedness for pattern graphs. They required matches to be contained in a subgraph, which restricts the number of nodes and edges to be considered. But still, whether a certain node in a data graph is a match for a node in the pattern graph is influenced by all the nodes and edges in the subgraph. Compared to their works, our work differs at that, the range of a data graph to be considered is determined by the node weight when deciding the similarity between nodes. There are also works that investigate the personalized matching conditions for nodes. Sokolsky et al. [32] proposed mq-simulation to give each node different conditions. Their work is for rooted graphs, and the similarity is decided by the most similar paths started with the nodes. However, the limited similarity in our work is decided by all the nodes and edges in the neighbouring subgraphs. The changing semantics of the graphs has been studied by recent works. For example, Xuan et al. [33] identified the different levels of semantic uncertainty in keywords network of webpages based on Keyword Association Linked Network [34]. They constructed a semantic pyramid to express the uncertainty hierarchy of a web event, with which the effect of webpage recommendation might be improved.

The paper is organized as follows. In Section 2, we review the basic notions of graph theory, subgraph bisimulation, and graph simulation. The notion of k-limited similarity and limited simulation is defined in Section 3. An algorithm is presented and experimentally studied in Sections 4 and 5, respectively. In Section 6, we define the notion of limited bisimulation and analyze the complexity of the graph pattern matching via limited bisimulation. The paper is concluded in Section 7.

2. Preliminaries

In this section, we review the basic notions of graph and pattern in Section 2.1. We then introduce the problem of subgraph bisimulation and graph simulation for the traditional graph pattern matching in Section 2.2.

2.1. Data graphs and pattern graphs

We denote a graph by $G = (V, E)$, where V is the node set and E is the edge set [35]. Let Σ_V be a finite set of node labels, and Σ_E be a finite set of edge labels. Label sets Σ_V and Σ_E denote all possible attributes of nodes and edges, respectively. A data graph is a directed node-labeled and edge-labeled graph specified as follows.

Definition 2.1 [26]**.** A *data graph* is a directed graph $G = (V, E, f_V, f_E)$, where

(1) V is a finite set of nodes;
(2) $E \subseteq V \times V$ is a finite set of edges, in which $(v, v') \in E$ denotes an edge from nodes v to v';
(3) f_V is a function that maps each node $v \in V$ to a node label $f_V(v)$ in Σ_V;
(4) f_E is a function that maps each edge $e \in E$ to an edge label $f_E(e)$ in Σ_E.

Intuitively, the node label function f_V specifies the attributes of nodes, such as labels, keywords, social positions, and ratings [16]; the edge label function f_E specifies the attributes of edges, such as relation types and information flow.

A subgraph of a data graph G is another data graph formed by some nodes and edges of G. The formal definition of the subgraph of a data graph is as follows.

Definition 2.2 [35]**.** Given two data graphs $G = (V, E, f_V, f_E)$ and $G' = (V', E', f_{V'}, f_{E'})$, we say that G' is a *subgraph* of G if

(1) $V' \subseteq V$ and $E' \subseteq E$;
(2) for each node $v \in V'$, $f_{V'}(v) = f_V(v)$;
(3) for each edge $e \in E'$, $f_{E'}(e) = f_E(e)$.