Information Processing Letters 134 (2018) 57–61

Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

Are unique subgraphs not easier to find?

Mirosław Kowaluk [a], Andrzej Lingas [b,*]

[a] Institute of Informatics, University of Warsaw, Warsaw, Poland
[b] Department of Computer Science, Lund University, Lund, Sweden

ARTICLE INFO

Article history:
Received 11 January 2017
Received in revised form 15 January 2018
Accepted 6 February 2018
Available online 8 February 2018
Communicated by P. Wong

Keywords:
Algorithms
Subgraph isomorphism
Induced subgraph isomorphism
Unique subgraph occurrence
Time complexity

ABSTRACT

Consider a pattern graph H with l edges, and a host graph G which may contain several occurrences of H. In [15], we claimed that the time complexity of the problems of finding an occurrence of H (if any) in G as well as that of the decision version of the problem are within a multiplicative factor $\widetilde{O}(l^3)$ of the time complexity for the corresponding problem, where the host graph is guaranteed to contain at most one occurrence of a subgraph isomorphic to H, and the notation $\widetilde{O}(\,)$ suppresses polylogarithmic in n factors. We show a counterexample to this too strong claim and correct it by providing an $\widetilde{O}((l(d-1)+2)^l)$ bound on the multiplicative factor instead, where d is the maximum number of occurrences of H that can share the same edge in the input host graph. We provide also an analogous correction in the induced case when occurrences of induced subgraphs isomorphic to H are sought.

1. Introduction

The problems of detecting, finding, counting and listing subgraphs or induced subgraphs of a host graph that are isomorphic to a given pattern graph have been widely studied. They are important both in their own rights as well as subproblems for other problems in algorithmics. Their recent applications include among other things bioinformatics [1,5], social networks [19], and automatic design [24].

The aforementioned problems are generally termed *subgraph isomorphism* and *induced subgraph isomorphism* problems, respectively. Their decision, finding, counting and even enumeration versions have been extensively investigated in the literature. In particular, the decision versions include as special cases such well-known NP-hard problems as the independent set, clique, Hamiltonian cycle and path problems [10]. For arbitrary graphs, they are known to admit polynomial-time solutions only in the case when the pattern graph is of fixed size.

For a given pattern graph H on k vertices and an arbitrary host graph on n vertices, the detection, finding and counting versions of subgraph isomorphism and induced subgraph isomorphism admit algorithms running in time $O(n^{\omega(\lfloor k/3\rfloor,\lceil (k-1)/3\rceil,\lceil k/3\rceil)})$, where $\omega(p,q,r)$ is the exponent of fast arithmetic matrix multiplication of an $n^p \times n^q$ matrix by an $n^q \times n^r$ matrix (cf. [7,11,13,17]).

The subgraph isomorphism and induced subgraph isomorphism for pattern graphs of fixed size are known to have more efficient algorithmic solutions when restricted to special graph classes, e.g., sparse graphs [6,7,23] or in particular planar graphs [8].

Already a restriction of the pattern graph of fixed size to a special graph class, e.g., graphs of bounded treewidth, cycles, graphs having a relatively large independent set leads to faster algorithms (cf. [3,4,12,13,16,18,22,23]).

In this paper, following [15], we address the question of whether or not the guarantee that the host graph contains at most one occurrence of the pattern graph (up to automorphisms) can yield more efficient solutions to the sub-

* Corresponding author.
E-mail addresses: kowaluk@mimuw.edu.pl (M. Kowaluk), Andrzej.Lingas@cs.lth.se (A. Lingas).

https://doi.org/10.1016/j.ipl.2018.02.010

graph isomorphism problem with pattern graph of fixed size, than those in the general case where the number of occurrences of the pattern graph is unrestricted.

There are several known examples of combinatorial problems admitting more efficient algorithms under the assumption of solution uniqueness.

For instance, Gabow et al. [9] show that detecting if a given graph has a unique perfect matching, and finding one if it exists, can be done in time $O(m\log^4 n)$. In a weighted setting, a variation of this problem is to decide whether a given perfect maximum-weight matching in a graph is unique. The latter problem has applications in computational biology, namely, in RNA structure prediction.

Next, unique lowest common ancestors in directed acyclic graphs can be found more efficiently than those non-necessarily unique [14]. On the other hand, it is well known that the SAT problem restricted to instances having at most one satisfying assignment is as hard as SAT [21].

In [15], we believed that we provided a negative answer to the addressed question. Namely, we claimed that the subgraph (or, induced subgraph, respectively) isomorphism problem with pattern graph of fixed size efficiently reduces to its restricted case where the host graph is guaranteed to have at most one occurrence of the pattern graph. More precisely, we stated that if the pattern graph has l edges then the time complexity of the subgraph isomorphism is within a multiplicative factor $\widetilde{O}(l^3)$ of that for the aforementioned restricted case. The reductions were randomized and they could be regarded as a generalization of the reduction of the problem of finding witnesses of Boolean matrix product to the problem of finding unique witnesses of Boolean matrix product [2,20].

In this paper, we show a counterexample to the aforementioned reductions from [15]. Instead, we prove a weaker claim under the additional assumption that the number of occurrences of the pattern graph with l edges that can share the same edge in the input host graph is bounded by d. We show then that the time complexity of the subgraph isomorphism is within a multiplicative factor $\widetilde{O}((l(d-1)+2)^l)$ of that for the restricted case, where the host graph is guaranteed to have at most one occurrence of the pattern graph. We provide also an analogous correction of the too strong claim in [15] in the induced case, when occurrences of induced subgraphs isomorphic to the pattern graph are sought.

Our paper is structured as follows. In the next section, we present the counterexample to the claim from [15] and prove the aforementioned weaker claim for standard subgraph isomorphism whereas in Section 3, we prove an analogous weaker claim for induced subgraph isomorphism. We conclude with Final remarks.

2. Additional assumptions are needed

Let H be a pattern graph with l edges and q vertices. In Theorems 2.2, 3.2 in [15], we expressed the time complexities of the problems of finding or detecting a copy, or an induced copy, of H in a host graph G in terms of those for the restricted variants, where the host graph is guaranteed to contain at most one copy of H. It turns out that an additional assumption is needed for Theorems 2.2, 3.2 to hold. Namely, the multi-occurrences of the pattern graph H in G have to be mutually edge-disjoint in the standard case and vertex-disjoint in the induced case in order to achieve the efficient reduction of finding or detecting a copy of H in G to the corresponding problem for subgraphs of the input graph containing at most one copy of H. The reduction is given by Algorithm 1 in the standard case and by Algorithm 2 in the induced case in [15]. (Below, we present a slight generalization of Algorithm 1, termed Algorithm 1′, where the probability of edge deletion is just set to $1-p$ instead of $1-\frac{1}{2^{\frac{1}{l}}}$ as in [15].)

Algorithm 1′

Input: A host graph G with n vertices and a pattern graph H with l edges.

Output: An occurrence of H in G or the answer "no" (the answer "yes" or "no" in the decision version).

Set F to G;

while F has at least l edges **do**:

1. Run the hypothetical procedure for unique occurrence of H on F for $U(n)$ time-steps. If an occurrence of H in F is found then output it and stop (alternatively in the decision version, if the existence of an occurrence of H in F is reported then report this and stop);
2. Delete each edge in F independently with probability $1-p$

Output "no H"

If the aforementioned additional assumption is not fulfilled, then for example, the input graph G on n vertices in Algorithm 1 could contain $n-2$ copies of a triangle with a common base. Note that the removal of the base edge annihilates all the triangle copies. In order to prune G to a subgraph F containing exactly one triangle copy, in a single run of Algorithm 1, the expected number of edge deletion iterations would need to be $\Omega(\log n)$ and during each of the iterations, the base edge would have to survive. The probability of such a run of Algorithm 1 would be not greater than $\frac{1}{n^\epsilon}$ for some constant ϵ. Hence, a polylogarithmic number of runs of Algorithm 1 very likely would not be sufficient to find or detect a triangle in such a G. This counterexample works also in the induced case.

The too strong claims in [15] are caused by an erroneous probabilistic analysis in Lemma 2.1 for the standard case and Lemma 3.1 for the induced case. First, we provide a correct equivalent of Lemma 2.1 relying on the additional assumption on the maximum number of occurrences of the pattern graph in the host graph that can share the same edge. Recall that the probability of edge deletion in Algorithm 1′ is set to $1-p$.