Information and Computation 243 (2015) 75–85

Contents lists available at ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

Faster exponential-time algorithms in graphs of bounded average degree ☆

Marek Cygan, Marcin Pilipczuk *

Institute of Informatics, University of Warsaw, Poland

ARTICLE INFO

Article history:
Received 15 August 2013
Available online 11 December 2014

Keywords:
Moderately-exponential algorithms
Bounded average degree
Counting perfect matchings
Minimum weight Hamiltonian cycle

ABSTRACT

We present a number of exponential-time algorithms for problems in sparse matrices and graphs of bounded average degree. First, we obtain a simple algorithm that computes a permanent of an $n \times n$ matrix over an arbitrary commutative ring with at most dn non-zero entries using $\mathcal{O}^\star(2^{(1-1/(3.55d))n})$ time and ring operations,[1] improving and simplifying the recent result of Izumi and Wadayama [FOCS 2012].
Second, we present a simple algorithm for counting perfect matchings in an n-vertex graph in $\mathcal{O}^\star(2^{n/2})$ time and polynomial space; our algorithm matches the complexity bounds of the algorithm of Björklund [SODA 2012], but relies on inclusion–exclusion principle instead of algebraic transformations.
Third, we show a combinatorial lemma that bounds the number of "Hamiltonian-like" structures in a graph of bounded average degree. Using this result, we show that

1. a minimum weight Hamiltonian cycle in an n-vertex graph with average degree bounded by d can be found in $\mathcal{O}^\star(2^{(1-\varepsilon_d)n})$ time and exponential space for a constant ε_d depending only on d;
2. the number of perfect matchings in an n-vertex graph with average degree bounded by d can be computed in $\mathcal{O}^\star(2^{(1-\varepsilon'_d)n/2})$ time and exponential space, for a constant ε'_d depending only on d.

The algorithm for minimum weight Hamiltonian cycle generalizes the recent results of Björklund et al. [TALG 2012] on graphs of bounded degree.

1. Introduction

Improving upon the 50-year-old $\mathcal{O}^\star(2^n)$-time dynamic programming algorithms for the Traveling Salesman Problem by Bellman [1] and Held and Karp [2] is a major open problem in the field of exponential-time algorithms [3]. A similar situation appears when we want to count perfect matchings in a graph: a half-century old $\mathcal{O}^\star(2^{n/2})$-time algorithm of Ryser for bipartite graphs [4] has only recently been transferred to arbitrary graphs [5], and breaking these time complexity barriers seems like a very challenging task.

☆ A preliminary version of this work has been presented at ICALP 2013.
* Corresponding author.
E-mail addresses: cygan@mimuw.edu.pl (M. Cygan), malcin@mimuw.edu.pl (M. Pilipczuk).
[1] The $\mathcal{O}^\star$-notation suppresses factors polynomial in the input size.

http://dx.doi.org/10.1016/j.ic.2014.12.007

From a broader perspective, improving upon a trivial brute-force or a simple dynamic programming algorithm is one of the main goals of the field of exponential-time algorithms. The last few years brought a number of positive results in that direction, most notably the $\mathcal{O}^\star(1.66^n)$ randomized algorithm for finding a Hamiltonian cycle in an undirected graph [6]. However, it is conjectured (the so-called Strong Exponential Time Hypothesis [7]) that the central problem of satisfying a general CNF-SAT formulae does not admit any exponentially better algorithm than the trivial brute-force one. A number of lower bounds were proven using this assumption [8–10].

Although the aforementioned 2^n or $2^{n/2}$-barriers may be difficult or even outright impossible to break, it seems reasonable to suspect that additional assumptions on the graph structure, such as bounded degree or bounded average degree, simplify the computational tasks significantly. In the case of the problem of counting perfect matchings in bipartite graphs, the classic algorithm of Ryser [4] the best known improvement in general graphs is an algorithm running in expected time $\mathcal{O}^\star(2^{(1-\Omega(n^{2/3}\log n))\cdot n/2})$ due to Bax and Franklin [11]. If one assumes bounded average degree, faster algorithms have been given by Servedio and Wan [12] and, very recently, by Izumi and Wadayama [13]. In Section 2 we continue this line of research and show the following.

Theorem 1. *For any commutative semiring R, given an $m \times n$ matrix M, $m \le n$ with elements from R with at most dm non-zero entries for some $d \ge 2$, one can compute the permanent of M using $\mathcal{O}^\star(2^{(1-1/(3.55d))m})$ time and performing $\mathcal{O}^\star(2^{(1-1/(3.55d))m})$ operations over the semiring R. The algorithm may require to use exponential space and store an exponential number of elements from R.*

Note that the number of perfect matchings in a bipartite graph is equal to the permanent of the adjacency matrix of this graph (computed over $\mathbb{Z}$). Hence, we improve the running time of [13,12] in terms of the dependency on d. We would like to emphasize that our proof of Theorem 1 is elementary and does not need the advanced techniques of coding theory used in [13].

Since the algorithm of Theorem 1 is able to handle the computation of the permanent of any matrix over commutative semiring, not only the special case of computing the number of perfect matchings, our result shows also that the running time of the algorithm of Björklund et al. [14] can be improved for sparse matrices.

In Section 3, we move to the problem of counting perfect matchings in general graphs. An algorithm solving this problem in $\mathcal{O}^\star(2^{n/2})$ time, that is, in time matching the bound for bipartite graphs, has been discovered very recently, in 2012, by Björklund [5]. Björklund's result improved upon previous algorithms with running time $\mathcal{O}^\star(1.732^n)$ due to Björklund and Husfeldt [15] and with running time $\mathcal{O}^\star(1.619^n)$ due to Koivisto [16]. We remark that, in contrast, the corresponding algorithm of Ryser [4] for bipartite graphs is already 50-years old. In Section 3, we observe that the problem of counting perfect matchings in general graphs can be reduced to a problem of counting some special types of cycle covers, which, in turn, can be done in $\mathcal{O}^\star(2^{n/2})$-time and polynomial space for an n-vertex graph, using the inclusion–exclusion principle. Thus, we obtain a new proof of the main result of [5], using the inclusion–exclusion principle instead of advanced algebraic transformations.

The problem of counting some special types of cycle covers, introduced in Section 3, moves us to the area of Hamiltonian-like problems. In 2008 Björklund et al. [17] observed that the classic dynamic programming algorithm for finding minimum weight Hamiltonian cycle can be trimmed to running time $\mathcal{O}^\star(2^{(1-\varepsilon_\Delta)n})$ in graphs of maximum degree Δ. The cost of this improvement is the use of exponential space, as we can no longer easily translate the dynamic programming algorithm into an inclusion–exclusion formula. The ideas from [17] were also applied to the Fast Subset Convolution algorithm [18], yielding a similar improvements for the problem of computing the chromatic number in graphs of bounded degree [19].

In Section 4 we show a combinatorial lemma that bounds the number of "Hamiltonian-like" structures in a graph of bounded average degree. Using this result, in Section 5 we show the following.

Theorem 2. *For every $d \ge 1$ there exists a constant $\varepsilon_d > 0$ such that, given an n-vertex graph G of average degree bounded by d, in $\mathcal{O}^\star(2^{(1-\varepsilon_d)n})$ time and exponential space one can find in G a minimum weight Hamiltonian cycle.*

Theorem 3. *For every $d \ge 1$ there exists a constant $\varepsilon'_d > 0$ such that, given an n-vertex graph G of average degree bounded by d, in $\mathcal{O}^\star(2^{(1-\varepsilon'_d)n/2})$ time and exponential space one can count the number of perfect matchings in G.*

Theorem 2 generalizes the results of [17] to the graphs of bounded average degree. To the best of our knowledge, Theorem 3 is the first result that breaks the $2^{n/2}$-barrier for counting perfect matchings in not necessarily bipartite graphs of bounded (average) degree. We note that in Theorems 2 and 3 the constants ε_d and ε'_d depend on d in a doubly-exponential manner, which is worse than the single-exponential behavior of [17] in graphs of bounded degree.

Let us now shortly elaborate on the techniques used to prove the above theorems. Following the same general approach as the results of [17] for graphs of bounded degree, we want to limit the number of states of the classic dynamic programming algorithm for minimum weight Hamiltonian cycle or of the algorithm developed in Section 3 for counting perfect matchings (written as a dynamic programming algorithm, instead of one based on the inclusion–exclusion principle). However, in order to deal with graphs of bounded average degree, we need to introduce new concepts and tools. Recall that, by a standard averaging argument, if the average degree of an n-vertex graph G is bounded by d, for any $D \ge d$ there are at most dn/D vertices of degree at least D. However, it turns out that this bound cannot be tight for a large number of