Finding the smallest H-subgraph in real weighted graphs and related problems

Virginia Vassilevska[1], Ryan Williams[1], and Raphael Yuster[2]

[1] Computer Science Department, Carnegie Mellon University, Pittsburgh, PA
{virgi,ryanw}@cs.cmu.edu
[2] Department of Mathematics, University of Haifa, Haifa, Israel
raphy@math.haifa.ac.il

Abstract. Let G be a graph with real weights assigned to the vertices (edges). The weight of a subgraph of G is the sum of the weights of its vertices (edges). The MIN H-SUBGRAPH problem is to find a minimum weight subgraph isomorphic to H, if one exists. Our main results are new algorithms for the MIN H-SUBGRAPH problem. The only operations we allow on real numbers are additions and comparisons. Our algorithms are based, in part, on fast matrix multiplication.

For vertex-weighted graphs with n vertices we obtain the following results. We present an $O(n^{t(\omega,h)})$ time algorithm for MIN H-SUBGRAPH in case H is a fixed graph with h vertices and $\omega < 2.376$ is the exponent of matrix multiplication. The value of $t(\omega, h)$ is determined by solving a small integer program. In particular, the smallest triangle can be found in $O(n^{2+1/(4-\omega)}) \leq o(n^{2.616})$ time, the smallest K_4 in $O(n^{\omega+1})$ time, the smallest K_7 in $O(n^{4+3/(4-\omega)})$ time. As h grows, $t(\omega, h)$ converges to $3h/(6-\omega) < 0.828h$. Interestingly, only for $h = 4, 5, 8$ the running time of our algorithm essentially matches that of the (unweighted) H-subgraph detection problem. Already for triangles, our results improve upon the main result of [VW06]. Using rectangular matrix multiplication, the value of $t(\omega, h)$ can be improved; for example, the runtime for triangles becomes $O(n^{2.575})$. We also present an algorithm whose running time is a function of m, the number of edges. In particular, the smallest triangle can be found in $O(m^{(18-4\omega)/(13-3\omega)}) \leq o(m^{1.45})$ time.

For edge-weighted graphs we present an $O(m^{2-1/k} \log n)$ time algorithm that finds the smallest cycle of length $2k$ or $2k-1$. This running time is identical, up to a logarithmic factor, to the running time of the algorithm of Alon et al. for the unweighted case. Using the color coding method and a recent algorithm of Chan for distance products, we obtain an $O(n^3/\log n)$ time randomized algorithm for finding the smallest cycle of any fixed length.

1 Introduction

Finding cliques or other types of subgraphs in a larger graph are classical problems in complexity theory and algorithmic combinatorics. Finding a maximum clique is NP-Hard, and also hard to approximate [Ha98]. This problem is also conjectured to be *not* fixed parameter tractable [DF95]. The problem of finding (induced) subgraphs on k vertices in an n-vertex graph has been studied extensively (see, e.g., [AYZ95,AYZ97,CN85,EG04,KKM00,NP85,YZ04]). All known algorithms for finding an induced subgraph on k vertices have running time $n^{\Theta(k)}$. Many of these algorithms use fast matrix multiplication to obtain improved exponents.

The main contribution of this paper is a set of improved algorithms for finding an (induced) k-vertex subgraph in a real vertex-weighted or edge-weighted graph. More formally, let G be a graph with real weights assigned to the vertices (edges). The weight of a subgraph of G is the sum of the weights of its vertices (edges). The MIN H-SUBGRAPH problem is to find an H-subgraph of minimum weight, if one exists. Some of our algorithms are based, in part, on *fast* matrix multiplication. In several cases, our algorithms use fast *rectangular* matrix multiplication algorithms. However, for simplicity reasons, we express most of our time bounds in terms of ω, the exponent of fast *square* matrix multiplications. The best bound currently available on ω is $\omega < 2.376$, obtained by Coppersmith and Winograd [CW90]. This is done by reducing each rectangular matrix product into a collection of smaller square matrix products. Slightly improved bounds can be obtained by using the best available rectangular matrix multiplication algorithms of Coppersmith [Cop97] and Huang and Pan [HP98]. In all of our algorithms we assume that the graphs are *undirected*, for simplicity. All of our results are applicable to directed graphs as well. Likewise, all of our results on the MIN-H-SUBGRAPH problem hold for the analogous MAX-H-SUBGRAPH problem. As usual, we use the *addition-comparison* model for handling real numbers. That is, real numbers are only allowed to be compared or added.

Our first algorithm applies to *vertex-weighted* graphs. In order to describe its complexity we need to define a small integer optimization problem. Let $h \geq 3$ be a positive integer. The function $t(\omega, h)$ is defined by the following optimization program.

Definition 1.

$$b_1 = \max\{b \in N \ : \ \frac{b}{4-\omega} \ \leq \ \lfloor \frac{h-b}{2} \rfloor\}. \tag{1}$$

$$s_1 = h - b_1 + \frac{b_1}{4-\omega}. \tag{2}$$

$$s_2(b) = \max\{h - b + \lfloor \frac{h-b}{2} \rfloor \ , \ h - (3-\omega)\lfloor \frac{h-b}{2} \rfloor\}. \tag{3}$$

$$s_2 = \min\{s_2(b) \ : \ \lfloor \frac{h-b}{2} \rfloor \leq b \leq h-2\}. \tag{4}$$

$$t(\omega, h) = \min\{s_1, s_2\}. \tag{5}$$

By using fast rectangular matrix multiplication, an alternative definition for $t(\omega, h)$, resulting in slightly smaller values, can be obtained (note that if $\omega = 2$, as conjectured by many researchers, fast rectangular matrix multiplication has no advantage over fast square matrix multiplication).

Theorem 1. *Let H be a fixed graph with h vertices. If $G = (V, E)$ is a graph with n vertices, and $w : V \to \Re$ is a weight function, then an induced H-subgraph of G (if exists) of minimum weight can be found in $O(n^{t(\omega,h)})$ time.*

It is easy to establish some small values of $t(\omega, h)$ directly. For $h = 3$ we have $t(\omega, 3) = 2 + 1/(4 - \omega) < 2.616$ by taking $b_1 = 1$ in (1). Using fast rectangular matrix multiplication this can be improved to 2.575. In particular, a triangle of minimum weight can be found in $o(n^{2.575})$ time. This should be compared to the $O(n^\omega) \le o(n^{2.376})$ algorithm for detecting a triangle in an *unweighted* graph. For $h = 4$ we have $t(\omega, 4) = \omega + 1 < 3.376$ by taking $b = 2$ in (4). Interestingly, the fastest algorithm for detecting a K_4, that uses square matrix multiplication, also runs in $O(n^{\omega+1})$ time [NP85]. The same phenomena also happens for $h = 5$ where $t(\omega, 5) = \omega + 2 < 4.376$ and for $h = 8$ where $t(\omega, 8) = 2\omega + 2 < 6.752$, but in no other cases! We also note that $t(\omega, 6) = 4+2/(4-\omega)$, $t(\omega, 7) = 4+3/(4-\omega)$, $t(\omega, 9) = 2\omega + 3$ and $t(\omega, 10) = 6 + 4/(4 - \omega)$. However, a closed formula for $t(\omega, h)$ cannot be given. Already for $h = 11$, and for infinitely many values thereafter, $t(\omega, h)$ is only piecewise linear in ω. For example, if $7/3 \le \omega < 2.376$ then $t(\omega, 11) = 3\omega + 2$, and if $2 \le \omega \le 7/3$ then $t(\omega, 11) = 6 + 5/(4 - \omega)$. Finally, it is easy to verify that both s_1 in (2) and s_2 in (4) converge to $3h/(6 - \omega)$ as h increases. Thus, $t(\omega, h)$ converges to $3h/(6 - \omega) < 0.828h$ as h increases.

Prior to a few months ago, the only known algorithm for MIN H-SUBGRAPH in the vertex-weighted case was the naïve $O(n^h)$ algorithm. Very recently, [VW06] gave an $O(n^{h \cdot \frac{\omega+3}{6}}) \le o(n^{0.896h})$ randomized algorithm, for h divisible by 3. Our algorithms are deterministic, and uniformly improve upon theirs, for all values of h.[3]

A slight modification in the algorithm of Theorem 1, without increasing its running time by more than a logarithmic factor, can also answer the decision problem: "is there an H-subgraph whose weight is in the interval $[w_1, w_2]$ where $w_1 \le w_2$ are two given reals?" Another feature of Theorem 1 is that it makes a relatively small number of comparisons. For example, the smallest triangle can be found by the algorithm using only $O(m + n \log n)$ comparisons, where m is the number of edges of G.

Since Theorem 1 is stated for induced H-subgraphs, it obviously also applies to not-necessarily induced H-subgraphs. However, the latter problem can, in some cases, be solved faster. For example, we show that the $o(n^{2.616})$ time bound for finding the smallest triangle also holds if one searches for the smallest H-subgraph in case H is the complete bipartite graph $K_{2,k}$.

Several H-subgraph detection algorithms take advantage of the fact that G may be sparse. Improving a result of Itai and Rodeh [IR78], Alon, Yuster and Zwick obtained an algorithm for detecting a triangle, expressed in terms of m [AYZ97]. The running time of their algorithm is $O(m^{2\omega/(\omega+1)}) \le o(m^{1.41})$. This is faster than the $O(n^\omega)$ algorithm when $m = o(n^{(\omega+1)/2})$. The best known running times in terms of m for $H = K_k$ when $k \ge 4$ are given in [EG04].

[3] [VW06] also give a deterministic $O(B \cdot n^{(\omega+3)/2}) \le o(B \cdot n^{2.688})$ algorithm, where B is the number of bits needed to represent the (absolute) maximum weight. Note this algorithm is *not* strongly polynomial.

Sparseness can also be used to obtain faster algorithms for the vertex-weighted MIN H-SUBGRAPH problem. The triangle algorithm of [VW06] extends to a randomized $O(m^{1.46})$ algorithm. We prove:

Theorem 2. *If $G = (V, E)$ is a graph with m edges and no isolated vertices, and $w : V \to \Re$ is a weight function, then a triangle of G with minimum weight (if exists) can be found in $O(m^{(18-4\omega)/(13-3\omega)}) \le o(m^{1.45})$ time.*

We now turn to edge-weighted graphs. An $O(m^{2-1/\lceil k/2 \rceil})$ time algorithm for detecting the existence of a cycle of length k is given in [AYZ97]. A small improvement was obtained later in [YZ04]. However, the algorithms in both papers fail when applied to edge-weighted graphs. Using the *color coding* method, together with several additional ideas, we obtain a randomized $O(m^{2-1/\lceil k/2 \rceil})$ time algorithm in the edge-weighted case, and an $O(m^{2-1/\lceil k/2 \rceil} \log n)$ deterministic algorithm.

Theorem 3. *Let $k \ge 3$ be a fixed integer. If $G = (V, E)$ is a graph with m edges and no isolated vertices, and $w : E \to \Re$ is a weight function, then a minimum weight cycle of length k, if exists, can be found with high probability in $O(m^{2-1/\lceil k/2 \rceil})$ time, and deterministically in $O(m^{2-1/\lceil k/2 \rceil} \log n)$ time.*

In a recent result of Chan [Ch05] it is shown that the distance product of two $n \times n$ matrices with real entries can be computed in $O(n^3/\log n)$ time (again, reals are only allowed to be compared or added). [VW06] showed how to reduce the MIN H-SUBGRAPH problem in edge-weighted graphs to the problem of computing a distance product. (The third author independently proved this as well.)

Theorem 4 ([VW06]). *Let H be a fixed graph with h vertices. If $G = (V, E)$ is a graph with n vertices, and $w : E \to \Re$ is a weight function, then an induced H-subgraph of G (if exists) of minimum weight can be found in $O(n^h/\log n)$ time.*

We can strengthen the above result considerably, in the case where H is a cycle. For (not-necessarily induced) cycles of fixed length we can combine distance products with the color coding method and obtain:

Theorem 5. *Let k be a fixed positive integer. If $G = (V, E)$ is a graph with n vertices, and $w : E \to \Re$ is a weight function, a minimum weight cycle with k vertices (if exist) can be found, with high probability, in $O(n^3/\log n)$ time.*

In fact, the proof of Theorem 5 shows that a minimum weight cycle with $k = o(\log \log n)$ vertices can be found in (randomized) sub-cubic time.

Finally, we consider the related problem of finding a certain chromatic H-subgraph in an edge-colored graph. We consider the two extremal chromatic cases. An H-subgraph of an edge-colored graph is called *rainbow* if all the edges have distinct colors. It is called *monochromatic* if all the edges have the same color. Many combinatorial problems are concerned with the existence of rainbow and/or monochromatic subgraphs.

We obtain a new algorithm that finds a rainbow H-subgraph, if it exists.

Theorem 6. *Let H be a fixed graph with $3k+j$ vertices, $j \in \{0,1,2\}$. If $G=(V,E)$ is a graph with n vertices, and $c: E \to C$ is an edge-coloring, then a rainbow H-subgraph of G (if exists) can be found in $O(n^{\omega k+j}\log n)$ time.*

The running time in Theorem 6 matches, up to a logarithmic factor, the running time of the induced H-subgraph detection problem in (uncolored) graphs.

We obtain a new algorithm that finds a monochromatic H-subgraph, if it exists. For fixed H, the running time of our algorithm matches the running time of the (uncolored) H-subgraph detection problem, except for the case $H = K_3$.

Theorem 7. *Let H be a fixed connected graph with $3k+j$ vertices, $j \in \{0,1,2\}$. If $G=(V,E)$ is a graph with n vertices, and $c: E \to C$ is an edge-coloring, then a monochromatic H-subgraph of G (if exists) can be found in $O(n^{\omega k+j})$ time, unless $H = K_3$. A monochromatic triangle can be found in $O(n^{(3+\omega)/2}) \le o(n^{2.688})$ time.*

Due to space limitation, the proofs of Theorems 6 and 7 will appear in the journal version of this paper.

The rest of this paper is organized as follows. In Section 2 we focus on vertex-weighted graphs, describe the algorithms proving Theorems 1 and 2, and some of their consequences. Section 3 considers edge-weighted graphs and contains the algorithms proving Theorems 3, 4 and 5. The final section contains some concluding remarks and open problems.

2 Minimal H-subgraphs of real vertex-weighted graphs

In the proof of Theorem 1 it would be convenient to assume that $H = K_h$ is a clique on h vertices. The proof for all other induced subgraphs with h vertices is only slightly more cumbersome, but essentially the same.

Let $G=(V,E)$ be a graph with real vertex weights, and assume $V = \{1,\ldots,n\}$. For two positive integers a,b, the *adjacency system* $A(G,a,b)$ is the 0-1 matrix defined as follows. Let S_x be the set of all $\binom{n}{x}$ x-subsets of vertices. The *weight* $w(U)$ of $U \in S_x$ is the sum of the weights of its elements. We *sort* the elements of S_x according to their weights. This requires $O(n^x \log n)$ time, assuming x is a constant. Thus, $S_x = \{U_{x,1},\ldots,U_{x,\binom{n}{x}}\}$ where $w(U_{x,i}) \le w(U_{x,i+1})$. The matrix $A(G,a,b)$ has its rows indexed by S_a. More precisely, the j'th row is indexed by $U_{a,j}$. The columns are indexed by S_b where the j'th column is indexed by $U_{b,j}$. We put $A(G,a,b)[U,U'] = 1$ if and only if $U \cup U'$ induces a K_{a+b} in G (this implies that $U \cap U' = \emptyset$). Otherwise, $A(G,a,b)[U,U'] = 0$. Notice that the construction of $A(G,a,b)$ requires $O(n^{a+b})$ time.

For positive integers a,b,c, so that $a+b+c = h$, consider the Boolean product $A(G,a,b,c) = A(G,a,b) \times A(G,b,c)$. For $U \in S_a$ and $U' \in S_c$ for which $A(G,a,b,c)[U,U'] = 1$, define their *smallest witness* $\delta(U,U')$ to be the smallest element $U'' \in S_b$ for which $A(G,a,b)[U,U''] = 1$ and also $A(G,b,c)[U'',U'] = 1$. For each $U \in S_a$ and $U' \in S_c$ with $A(G,a,b,c)[U,U'] = 1$ and with $U \cup U'$ inducing a K_{a+c}, if $U'' = \delta(U,U')$ then $U \cup U' \cup U''$ induces a K_h in G whose weight is the smallest of all the K_h copies of G that contain $U \cup U'$. This follows from the fact that S_b is sorted. Thus, by computing the smallest witnesses of all plausible pairs $U \in S_a$ and $U' \in S_c$ we can find a K_h in G with minimum weight, if it exists, or else determine that G does not have K_h as a subgraph.

Let $A = A_{n_1 \times n_2}$ and $B = B_{n_2 \times n_3}$ be two 0-1 matrices. The *smallest witness matrix* of AB is the matrix $W = W_{n_1 \times n_3}$ defined as follows. $W[i,j] = 0$ if $(AB)[i,j] = 0$. Otherwise, $W[i,j]$ is the smallest index k so that $A[i,k] = B[k,j] = 1$. Let $f(n_1, n_2, n_3)$ be the time required to compute the smallest witness matrix of the product of an $n_1 \times n_2$ matrix by an $n_2 \times n_3$ matrix. Let $h \geq 3$ be a fixed positive integer. For all possible choices of positive integers a, b, c with $a + b + c = h$ denote

$$f(h,n) = \min_{a+b+c=h} f(n^a, n^b, n^c).$$

Clearly, the time to sort S_b and to construct $A(G,a,b)$ and $A(G,b,c)$ is overwhelmed by $f(n^a, n^b, n^c)$. It follows from the above discussion that:

Lemma 1. *Let $h \geq 3$ be a fixed positive integer and let G be a graph with n vertices, each having a real weight. A K_h-subgraph of G with minimum weight, if exists, can be found in $O(f(h,n))$ time. Furthermore, if $f(n^a, n^b, n^c) = f(h,n)$ then the number of comparisons needed to find a minimum weight K_h is $O(n^b \log n + z(G, a+c))$ where $z(G, a+c)$ is the number of K_{a+c} in G.*

In fact, if $b \geq 2$, the number of comparisons in Lemma 1 can be reduced to only $O(n^b + z(G, a+c))$. Sorting S_b reduces to sorting the sums $X + X + \ldots + X$ (X repeated b times) of an n-element set of reals X. Fredman showed in [Fr76a] that this can be achieved with only $O(n^b)$ comparisons.

A simple randomized algorithm for computing (not necessarily first) witnesses for Boolean matrix multiplication, in essentially the same time required to perform the product, is given by Seidel [Sei95]. His algorithm was derandomized by Alon and Naor [AN96]. However, computing the matrix of first witnesses seems to be a more difficult problem. Improving an earlier algorithm of Bender et al. [BFPSS05], Kowaluk and Lingas [KL05] show that $f(3,n) = O(n^{2+1/(4-\omega)}) \leq o(n^{2.616})$. This already yields the case $h = 3$ in Theorem 1. We will need to extend and generalize the method from [KL05] in order to obtain upper bounds for $f(h,n)$. Our extension will enable us to answer more general queries such as "is there a K_h whose weight is within a given weight interval?"

Proof of Theorem 1: Let $h \geq 3$ be a fixed integer. Suppose a, b, c are three positive integers with $a+b+c = h$ and suppose that $0 < \mu \leq b$ is a real parameter. For two 0-1 matrices $A = A_{n^a \times n^b}$ and $B = B_{n^b \times n^c}$ the *μ-split* of A and B is obtained by splitting the columns of A and the rows of B into consecutive parts of size $\lceil n^\mu \rceil$ or $\lfloor n^\mu \rfloor$ each. In the sequel we ignore floors and ceilings whenever it does not affect the asymptotic nature of our results. This defines a partition of A into $p = n^{b-\mu}$ rectangular matrices $A_1, \ldots, A_p$, each with n^a rows and n^μ columns, and a partition of B into p rectangular matrices $B_1, \ldots, B_p$, each with n^μ rows and n^c columns. Let $C_i = A_i B_i$ for $i = 1, \ldots, p$. Notice that each element of C_i is a nonnegative integer of value at most n^μ and that $AB = \sum_{i=1}^{p} C_i$. Given the C_i, the smallest witness matrix W of the product AB can be computed as follows. To determine $W[i,j]$ we look for the smallest index r for which $C_r[i,j] \neq 0$. If no such r exists, then $W[i,j] = 0$. Otherwise, having found r, we now look for the smallest index k so that $A_r[i,k] = A_r[k,j] = 1$. Having found k we clearly have $W[i,j] = (r-1)n^{\mu} + k$.

We now determine a choice of parameters a, b, c, μ so that the time to compute $C_1, \ldots, C_p$ and the time to compute the first witnesses matrix W, is $O(n^{t(\omega,h)})$. By Lemma 1, this suffices in order to prove the theorem. We will only consider $\mu \leq \min\{a, b, c\}$. Taking larger values of μ results in worse running times. The rectangular product C_i can be computed by performing $O(n^{a-\mu}n^{c-\mu})$ products of square matrices of order n^{μ}. Thus, the time required to compute C_i is

$$O(n^{a-\mu}n^{c-\mu}n^{\omega\mu}) = O(n^{a+c+(\omega-2)\mu}).$$

Since there are p such products, and since each of the n^{a+c} witnesses can be computed in $O(p + n^{\mu})$ time, the overall running time is

$$\begin{aligned} O(pn^{a+c+(\omega-2)\mu} + n^{a+c}(p + n^{\mu})) &= O(n^{h-(3-\omega)\mu} + n^{h-\mu} + n^{h-b+\mu}) \\ &= O(n^{h-(3-\omega)\mu} + n^{h-b+\mu}). \end{aligned} \tag{6}$$

Optimizing on μ we get $\mu = b/(4-\omega)$. Thus, if, indeed, $b/(4-\omega) \leq \min\{a, c\}$ then the time needed to find W is $O(n^{h-b+b/(4-\omega)})$. Of course, we would like to take b as large as possible under these constraints. Let, therefore, b_1 be the largest integer b so that $b/(4-\omega) \leq \lfloor (h-b)/2 \rfloor$. For such a b_1 we can take $a = \lfloor (h-b_1)/2 \rfloor$ and $c = \lceil (h-b_1)/2 \rceil$ and, indeed, $\mu \leq \min\{a, c\}$. Thus, (6) gives that the running time to compute W is

$$O(n^{h-b_1+b_1/(4-\omega)}).$$

This justifies s_1 appearing in (2) in the definition of $t(\omega, h)$. There may be cases where we can do better, whenever $b/(4-\omega) > \min\{a, c\}$. We shall only consider the cases where $a = \mu = \lfloor (h-b)/2 \rfloor \leq b$ (other cases result in worse running times). In this case $c = \lceil (h-b)/2 \rceil$ and, using (6), the running time is

$$O(n^{h-(3-\omega)\lfloor \frac{h-b}{2} \rfloor} + n^{h-b+\lfloor \frac{h-b}{2} \rfloor}).$$

This justifies s_2 appearing in (4) in the definition of $t(\omega, h)$. Since $t(\omega, h) = \min\{s_1, s_2\}$ we have proved that W can be computed in $O(n^{t(\omega,h)})$ time. ■

As can be seen from Lemma 1 and the remark following it, the number of comparisons that the algorithm performs is relatively small. For example, in the case $h = 3$ we have $a = b = c = 1$ and hence the number of comparisons is $O(n \log n + m)$. In all the three cases $h = 4, 5, 6$ the value $b = 2$ yields $t(\omega, h)$. Hence, the number of comparisons is $O(n^2)$ for $h = 4$, $O(n^2 + mn)$ for $h = 5$ and $O(n^2 + m^2)$ for $h = 6$.

Suppose $w : \{1, \ldots, n^b\} \to \Re$ so that $w(k) \leq w(k+1)$. The use of the μ-split in the proof of Theorem 1 enables us to determine, for each i, j and for a real interval

$I(i,j)$, whether or not there exists an index k so that $A[i,k] = B[k,j] = 1$ and $w(k) \in I(i,j)$. This is done by performing a binary search within the $p = n^{b-\mu}$ matrices $C_i, \ldots, C_p$. The running time in (6) only increases by a $\log n$ factor. We therefore obtain the following corollary.

Corollary 1. *Let H be a fixed graph with h vertices, and let $I \subset \Re$. If $G = (V,E)$ is a graph with n vertices, and $w : V \to \Re$ is a weight function, then, deciding whether G contains an induced H-subgraph with total weight in I can be done $O(n^{t(\omega,h)} \log n)$ time.*

Proof of Theorem 2: We partition the vertex set V into two parts $V = X \cup Y$ according to a parameter Δ. The vertices in X have degree at most Δ. The vertices in Y have degree larger than Δ. Notice that $|Y| < 2m/\Delta$. In $O(m\Delta)$ time we can scan all triangles that contain a vertex from X. In particular, we can find a smallest triangle containing a vertex from X. By Theorem 1, a smallest triangle induced by Y can be found in $O((m/\Delta)^{t(\omega,3)}) = O((m/\Delta)^{2+1/(4-\omega)})$ time. Therefore, a smallest triangle in G can be found in

$$O\left(m\Delta + \left(\frac{m}{\Delta}\right)^{2+1/(4-\omega)}\right)$$

time. By choosing $\Delta = m^{(5-\omega)/(13-3\omega)}$ the result follows. ■

The results in Theorems 1 and 2 are useful not only for real vertex weights, but also when the weights are large integers. Consider, for example, the graph parameter $\beta(G,H)$, the H *edge-covering number* of G. We define $\beta(G,H) = 0$ if G has no H-subgraph. Otherwise, $\beta(G,H)$ is the maximum number of edges incident with an H-subgraph of G. To determine $\beta(G,K_k)$ we assign to each vertex a weight equal to its degree. We now use the algorithm of Theorem 1 to find the *maximum* weighted K_k. If the weight of the maximum weighted K_k is w, then $\beta(G,K_k) = w - \binom{k}{2}$. In particular, $\beta(G,K_k)$ can be computed in $O(n^{t(\omega,k)})$ time.

Finally, we note that Theorems 1 and 2 apply also when the weight of an H-subgraph is not necessarily defined as the sum of the weights of its vertices. Suppose that the weight of a triangle (x,y,z) is defined by a function $f(x,y,z)$ that is monotone in each variable separately. For example, we may consider $f(x,y,z) = xyz$, $f(x,y,z) = xy + xz + yz$ etc. Assuming that $f(x,y,z)$ can be computed in constant time given x,y,z, it is easy to modify Theorems 1 and 2 to find a triangle whose weight is minimal with respect to f in $O(n^{2+1/(4-\omega)})$ time and $O(m^{(18-4\omega)/(13-3\omega)})$ time, respectively.

We conclude this section with the following proposition.

Proposition 1. *If $G = (V,E)$ is a graph with n vertices, and $w : V \to \Re$ is a weight function, then a (not necessarily induced) minimum weight $K_{2,k}$-subgraph can be found in $O(n^{2+1/(4-\omega)})$.*

Proof. To find the smallest $K_{2,k}$ we simply need to find, for any two vertices i,j, the first k smallest weighted vertices $v_1, \ldots, v_k$ so that each v_i is a common neighbor of i and j. As in Lemma 1, this reduces to finding the first k smallest witnesses of a 0-1 matrix product. A simple modification of the algorithm in Theorem 1 achieves this goal in the same running time (recall that k is fixed).

3 Minimal H-subgraphs of real edge-weighted graphs

Given a vertex-colored graph G with n vertices, an H-subgraph of G is called *colorful* if each vertex of H has a distinct color. The *color coding* method presented in [AYZ95] is based upon two important facts. The first one is that, in many cases, finding a colorful H-subgraph is easier than finding an H-subgraph in an uncolored graph. The second one is that in a random vertex coloring with k colors, an H-subgraph with k vertices becomes colorful with probability $k!/k^k > e^{-k}$ and, furthermore, there is a derandomization technique that constructs a family of not too many colorings, so that each H-subgraph is colorful in at least one of the colorings. The derandomization technique, described in [AYZ95], constructs a family of colorings of size $O(\log n)$ whenever k is fixed.

By the color coding method, in order to prove Theorem 3, it suffices to prove that, *given* a coloring of the vertices of the graph with k colors, a colorful cycle of length k of minimum weight (if exists) can be found in $O(m^{2-1/\lceil k/2 \rceil})$ time.

Proof of Theorem 3: Assume that the vertices of G are colored with the colors $1,\ldots,k$. We first show that for each vertex u, a minimum weight colorful cycle of length k that passes through u can be found in $O(m)$ time. For a permutation π of $1,\ldots,k$, we show that a minimum weight cycle of the form $u = v_1, v_2, \ldots, v_k$ in which the color of v_i is $\pi(i)$ can be found in $O(m)$ time. Without loss of generality, assume π is the identity. For $j = 2,\ldots,k$ let V_j be the set of vertices whose color is j so that there is a path from u to $v \in V_j$ colored consecutively by the colors $1,\ldots,j$. Let $S(v)$ be the set of vertices of such a path with minimum possible weight. Denote this weight by $w(v)$. Clearly, V_j can be created from V_{j-1} in $O(m)$ time by examining the neighbors of each $v \in V_{j-1}$ colored with j. Now, let $w_u = \min_{v \in v_k} w(v) + w(v,u)$. Thus, w_u is the minimum weight of a cycle passing through u, of the desired form, and a cycle with this weight can be retrieved as well.

We prove the theorem when k is even. The odd case is similar. Let $\Delta = m^{2/k}$. There are at most $2m/\Delta = O(m^{1-2/k})$ vertices with degree at least Δ. For each vertex u with degree at least Δ we find a minimum weight colorful cycle of length k that passes through u. This can be done in $O(m^{2-2/k})$ time. It now suffices to find a minimum weight colorful cycle of length k in the subgraph G' of G induced by the vertices with maximum degree less than Δ. Consider a permutation π of $1,\ldots,k$. For a pair of vertices x, y, let S_1 be the set of all paths of length $k/2$ colored consecutively by $\pi(1),\ldots,\pi(k/2),\pi(k/2+1)$. There are at most $m\Delta^{k/2-1} = m^{2-2/k}$ such paths and they can be found using the greedy algorithm in $O(m^{2-2/k})$ time. Similarly, let S_2 be the set of all paths of length $k/2$ colored consecutively by $\pi(k/2+1),\ldots,\pi(k),\pi(1)$. If u, v are endpoints of at least one path in S_1 then let $f_1(\{u,v\})$ be the minimum weight of such a path. Similarly define $f_2(\{u,v\})$. We can therefore find, in $O(m^{2-2/k})$ a pair u, v (if exists) so that $f_1(\{u,v\})+f_2(\{u,v\})$ is minimized. By performing this procedure for each permutation, we find a minimum weight colorful cycle of length k in G'. ■

Let $A = A_{n_1\times n_2}$ and $B = B_{n_2\times n_3}$ be two matrices with entries in $\Re \cup \infty$. The *distance product* $C = A \star B$ is an $n_1 \times n_3$ matrix with $C[i,j] = \min_{k=1\ldots,n_2} A[i,k] + B[k,j]$. Clearly, C can be computed in $O(n_1 n_2 n_3)$ time in the addition-comparison model. However, Fredman showed in [Fr76] that the distance product of two square matrices of order n can be performed in $O(n^3(\log\log n/\log n)^{1/3})$ time. Following a sequence of improvements over Fredman's result, Chan gave an $O(n^3/\log n)$ time algorithm for distance products. By partitioning the matrices into blocks it is obvious that Chan's algorithm computes the distance product of an $n_1 \times n_2$ matrix and an $n_2 \times n_3$ matrix in $O(n_1 n_2 n_3/\log\min\{n_1,n_2,n_3\})$ time. Distance products can be used to solve the MIN H-SUBGRAPH problem in edge weighted graphs.

Proof of Theorem 4: We prove the theorem for $H = K_h$. The proof for other induced H-subgraphs is essentially the same. Partition h into a sum of three positive integers $a+b+c = h$. Let S_a be the set of all K_a-subgraphs of G. Notice that $|S_a| < n^a$ and that each $U \in S_a$ is an a-set. Similarly define S_b and S_c. We define A to be the matrix whose rows are indexed by S_a and whose columns are indexed by S_b. The entry $A[U,U']$ is defined to by ∞ if $U \cup U'$ does not induce a K_{a+b}. Otherwise, it is defined to be the sum of the weights of the edges induced by $U \cup U'$. We define B to be the matrix whose rows are indexed by S_b and whose columns are indexed by S_c. The entry $A[U,U']$ is defined to by ∞ if $U \cup U'$ does not induce a K_{b+c}. Otherwise, it is defined to be the sum of the weights of the edges induced by $U \cup U'$ with *at least* one endpoint in U'. Notice the difference in the definitions of A and B. Let $C = A \star B$. The time to compute C using Chan's algorithm is $O(n^h/\log n)$. Now, for each $U \in S_a$ and $U' \in S_c$ so that $U \cup U'$ induces a K_{a+c}, let $w(U,U')$ be the sum of the weights of the edges with one endpoint in U and the other in U' plus the value of $C[U,U']$. If $w(U,U')$ is finite then it is the weight of the smallest K_h that contains $U \cup U'$. Otherwise, No K_h contains $U \cup U'$. ■

The weighted DENSE k-SUBGRAPH problem (see, e.g., [FKP01]) is to find a k-vertex subgraph with maximum total edge weight. A simple modification of the algorithm of Theorem 4 solves this problem in $O(n^k/\log n)$ time. To our knowledge, this is the first non-trivial algorithm for this problem. Note that the maximum total weight of a k-subgraph can potentially be much larger than a k-clique's total weight.

Proof of Theorem 5: We use the color coding method, and an idea similar to Lemma 3.2 in [AYZ95]. Given a coloring of the vertices with k colors, it suffices to show how to find the smallest colorful path of length $k-1$ connecting any pair of vertices in $2^{O(k)}n^3/\log n$ time. It will be convenient to assume that k is a power of two, and use recursion. Let C_1 be a set of $k/2$ distinct colors, and let C_2 be the complementary set of colors. Let V_i be the set of vertices colored by colors from C_i for $i = 1,2$. Let G_i be the subgraph induced by V_i.

Recursively find, for each pair of vertices in G_i, the minimum weight colorful path of length $k/2-1$. We record this information in matrices A_1, A_2, where the rows and columns of A_i are indexed by V_i. Let B be the matrix whose rows are indexed by V_1 and whose columns are indexed by V_2 where $B[u,v] = w(u,v)$. The distance product $D_{C_1,C_2} = (A_1 \star B) \star A_2$ gives, for each pair of vertices of G, all shortest paths of length $k-1$ where the first $k/2$ vertices are colored by colors from C_1 and the last $k/2$ vertices are colored by colors from C_2. By considering all $\binom{k}{k/2} < 2^k$ possible choices for (C_1, C_2), and computing D_{C_1,C_2} for each choice, we can obtain an $n \times n$ matrix D where $D[u,v]$ is the shortest colorful path of length $k-1$ between u and v. The number of distance products computed using this approach satisfies the recurrence $t(k) \le 2^k t(k/2)$. Thus, the overall running time is $2^{O(k)} n^3 / \log n$. ∎

The proof of Theorem 5 shows that, as long as $k = o(\log\log n)$, a cycle with k vertices and minimum weight can be found, with high probability, in $o(n^3)$ time. The previous best known algorithm (to our knowledge) for finding a minimum weight cycle of length k, in real weighted graphs, has running time $O(k! n^3 2^k)$ [PV91].

4 Concluding remarks and open problems

We presented several algorithms for the MIN finding H-SUBGRAPH in both real vertex weighted or real edge weighted graphs, and algorithms for the related problem of finding monochromatic or rainbow H-subgraphs in edge-colored graphs. It may be possible to improve upon the running times of some of our algorithms. More specifically, we raise the following open problems.
(i) Can the exponent $t(\omega, 3)$ in Theorem 1 be improved? If so, this would immediately imply an improved algorithm for first witnesses.
(ii) Can the logarithmic factor in Theorem 3 be eliminated? We know from [AYZ97] that this is the case in the unweighted version of the problem. Can the logarithmic factor in Theorem 6 be eliminated?
(iii) Can monochromatic triangles be detected faster than the $O(n^{(3+\omega)/2})$ algorithm of Theorem 7? In particular, can they be detected in $O(n^\omega)$ time?

Acknowledgment

The authors thank Uri Zwick for for some useful comments.

References

[AN96] N. Alon and M. Naor. Derandomization, witnesses for Boolean matrix multiplication and construction of perfect hash functions. *Algorithmica*, 16:434–449, 1996.

[AYZ95] N. Alon, R. Yuster, and U. Zwick. Color-coding. *Journal of the ACM*, 42:844–856, 1995.

[AYZ97] N. Alon, R. Yuster, and U. Zwick. Finding and counting given length cycles. *Algorithmica*, 17:209–223, 1997.

[BFPSS05] M. Bender, M. Farach-Colton, G. Pemmasani, S. Skiena, and P. Sumazin. Lowest common ancestors in trees and directed acyclic graphs. *J. Algorithms*, 57(2):75–94, 2005.

[Ch05] T.M. Chan. All-Pairs Shortest Paths with Real Weights in $O(n^3/\log n)$ Time. In *Proc. of the 9th WADS*, Lecture Notes in Computer Science 3608, Springer (2005), 318–324.

[CN85] N. Chiba and L. Nishizeki. Arboricity and subgraph listing algorithms. *SIAM Journal on Computing*, 14:210–223, 1985.

[Cop97] D. Coppersmith. Rectangular matrix multiplication revisited. *Journal of Complexity*, 13:42–49, 1997.

[CW90] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. *J. Symbol. Comput.*, 9:251–280, 1990.

[DF95] R.G. Downey and M.R. Fellows. Fixed-parameter tractability and completeness II. On completeness for W[1]. *Theoret. Comput. Sci.*, 141(1-2):109-131, 1995.

[EG04] F. Eisenbrand and F. Grandoni. On the complexity of fixed parameter clique and dominating set. *Theoret. Comput. Sci.*, 326(1-3):57–67, 2004.

[FKP01] U. Feige, G. Kortsarz and D. Peleg. The Dense k-Subgraph Problem. *Algorithmica*, 29(3):410–421, 2001.

[Fr76] M.L. Fredman. New bounds on the complexity of the shortest path problem. *SIAM Journal on Computing*, 5:49–60, 1976.

[Fr76a] M.L. Fredman. How good is the information theory bound in sorting? *Theoret. Comput. Sci.*, 1:355–361, 1976.

[Ha98] J. Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. *Acta Math.*, 182(1):105-142, 1998.

[HP98] X. Huang and V.Y. Pan. Fast rectangular matrix multiplications and applications. *Journal of Complexity*, 14:257–299, 1998.

[IR78] A. Itai and M. Rodeh. Finding a minimum circuit in a graph. *SIAM Journal on Computing*, 7:413–423, 1978.

[KKM00] T. Kloks, D. Kratsch, and H. Müller. Finding and counting small induced subgraphs efficiently. *Inf. Process. Lett.*, 74(3-4):115–121, 2000.

[KL05] M. Kowaluk and A. Lingas. LCA Queries in Directed Acyclic Graphs. In *Proc. of the 32nd ICALP*, Lecture Notes in Computer Science 3580, Springer (2005), 241–248.

[NP85] J. Nešetřil and S. Poljak. On the complexity of the subgraph problem. *Comment. Math. Univ. Carolin.*, 26(2):415–419, 1985.

[PV91] J. Plehn and B. Voigt. Finding Minimally Weighted Subgraphs. In *Proceedings of the 16th International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, Springer-Verlag, 1991.

[Sei95] R. Seidel. On the All-Pairs-Shortest-Path Problem in Unweighted Undirected Graphs. *J. Comput. Syst. Sci.*, 51(3):400–403, 1995.

[VW06] V. Vassilevska and R. Williams. Finding a maximum weight triangle in $n^{3-\delta}$ time, with applications. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC)*, to appear.

[YZ04] R. Yuster and U. Zwick. Detecting short directed cycles using rectangular matrix multiplication and dynamic programming. In *Proc. of the 15th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, ACM/SIAM (2004), 247–253.