Approximation schemes, cliques, colors and densest subgraphs

Maria Liazi ⋆

National and Kapodistrian University of Athens
Department of Informatics and Telecommunications
mliazi@di.uoa.gr

Abstract. In this thesis we study the problem of finding the densest k-subgraph of a given graph $G = (V, E)$. We present algorithms of polynomial time as well as approximation results on special graph classes.
Analytically, we study polynomial time algorithms for the densest k-subgraph problem on weighted graphs of maximal degree two, on weighted trees even if the solution is disconnected, and on interval graphs with intersection only between two consecutive cliques.
Moreover, we present a polynomial time approximation scheme for the densest k-subgraph problem on a star of cliques and a polynomial time algorithm on a tree of cliques of bounded degree. Finally, in the last part of the thesis we analyze a constant-factor approximation algorithm for the densest k-subgraph problem on chordal graphs.

Key Words: densest k-subgraph, polynomial time algorithms, approximation algorithms, polynomial time approximation scheme.

1 Introduction

In the *Densest k-subgraph* (DkS) problem we are given a graph $G = (V, E)$, $|V| = n$, and an integer $k \leq n$, and we ask for a subgraph of G induced by exactly k of its vertices such that the number of edges of this subgraph is maximized. The problem is directly NP-hard as a generalization of the well known *Maximum Clique* problem. The weighted version of the DkS problem is called *Heaviest k-subgraph* (HkS). In the HkS problem we are also given non negative weights on the edges of G and the goal is to find a k-vertex induced subgraph of maximum total edge weight.

During last years a large body of work has been concentrated on the design of approximation algorithms for both the DkS problem and its weighted version, based on a variety of techniques including greedy algorithms, LP relaxations and semidefinite programming. For a brief presentation of this body of work the reader is referred to [3] and the references therein. The best known approximation ratio for the DkS problem, which performs well for all values of k, is $O(n^{\delta})$, for some $\delta < \frac{1}{3}$ [6], while a simple greedy algorithm in [2] achieves an approximation ratio of $O(\frac{n}{k})$ even for the weighted version of the problem. On the other hand, it has been shown that the DkS problem does not admit a polynomial time approximation scheme (PTAS) if $NP \nsubseteq \cap_{\epsilon>0} BPTIME(2^{n^{\epsilon}})$ [13]. However, there is not a negative result that achieving an approximation ratio of $O(n^{\epsilon})$, for some $\epsilon > 0$, is NP-hard. Concerning approximation algorithms for special cases of the problem it is known that the DkS problem admits a PTAS for graphs of minimum degree $\Omega(n)$ as well as for dense graphs (of $\Omega(n^2)$ edges) when k is $\Omega(n)$ [1]. Moreover, algorithms achieving approximation factors of 4 [24] and 2 [11] have been proposed for the weighted DkS problem on complete graphs where the weights satisfy the triangle inequality.

⋆ Dissertation Advisor: Vassilis Zissimopoulos, Professor

The DkS problem is trivial on trees (any subtree of k vertices contains exactly $k-1$ edges). It is also known that DkS can be solved in polynomial time on graphs of maximal degree two [7] as well as on cographs, split graphs and k-trees [4]. On the other hand the DkS problem remains NP-hard on bipartite graphs [4], even of maximum degree three [7], as well as on comparability graphs, chordal graphs [4] and, if we are looking for a connected solution, on planar graphs [12] . The weighted version of the DkS problem if we ask for a connected solution is polynomial on trees [10, 21, 22].

In Section 2 we study the HkS problem on graphs of maximal degree 2 and on trees either if we ask for a connected or a disconnected solution. We also propose a dynamic programming algorithm for the DkS problem on interval graphs that have a simple path as clique graph. In Section 3 we study the DkS problem on clique graphs. In particular, we present a polynomial time approximation scheme (PTAS) on star of cliques and a polynomial time algorithm on tree of cliques of bounded degree. Finally, in Section 4 we present a constant approximation algorithm for the DkS problem on chordal graphs. We conclude on Section 5.

2 Polynomial variants of the HkS/DkS problem

We generalize the algorithms proposed in [22], [21] and [10], which yield a connected optimal solution for the HkS problem on trees. In fact, we present a polynomial time dynamic programming algorithm yielding an optimum solution that can be either connected or not. Despite this generalization the algorithm keeps the same time complexity as the known algorithms for the connected version of the problem. To obtain the optimal solution the algorithm solves recursively, starting from the leaves of the tree, the HjS problems for each vertex of the tree and for each value of $j = 1, 2, ..., k$ and therefore in [18] we prove that there is an $O(nk^2)$ time algorithm for the HkS problem on a tree.

The above mentioned dynamic programming algorithm solves the HkS problem on chains in $O(nk)$ time. In fact, the result for the latter case -the HkS problem on chains- is implied by a result for the solution of the quadratic 0-1 knapsack problem on edge series-parallel graphs in [23].

We also prove that there is a polynomial time algorithm yielding an optimal solution for the HkS problem on graphs of maximum degree two. We denote by g_i, $1 \leq i \leq m$, a connected component (either a cycle or a chain) of a graph G of maximal degree two. If by n_i we denote the number of vertices of each component g_i then the algorithm in its first phase solves optimally, for each connected component g_i of G, the j-vertex heaviest subgraph problems for each value of $j = 1, 2, \ldots, \min\{n_i, k\}$. If g_i is a chain this step can be carried out in $O(nk)$ by the dynamic programming algorithm for the HkS on trees. If g_i is a cycle we prove that there is an $O(nk)$ time algorithm for the HkS problem on a cycle of n vertices [18].

What we want in the second phase of the algorithm is to select at most one optimal solution (for some value of j) from each connected component such that the total number of their vertices is k and their total cost is maximized. This problem is, essentially, a generalization of the KNAPSACK problem, where we are given a partition of the items into groups and we have the restriction to select at most one item from each group. It is known that this problem can be solved in $O(mk^2)$ time by a dynamic programming algorithm proposed in [14]. Since m is $O(n)$ the second phase of the algorithm takes $O(nk^2)$ time. Combining the two phases of the algorithm we obtain an $O(nk^2)$ time algorithm for the HkS problem on graphs of maximal degree two [18].

A graph $G = (V, E)$ is called *interval* if there is a mapping I of the vertices of G into sets of consecutive integers such that for each pair of vertices $u, v \in V$, $(u, v) \in E \Longleftrightarrow I(u) \cap I(v) \neq \emptyset$. A *maximal clique* is a clique, which is not contained in any larger clique. A largest maximal clique is called *maximum clique*. The *intersection graph* of a family, F, of subsets of a set is defined as a graph, $\mathcal{G}$, whose vertices correspond to the subsets in F, and there is an edge between two vertices of $\mathcal{G}$ if the corresponding pair of subsets intersect. Given these definitions, the *clique graph* of a graph G is defined as the intersection graph of the maximal cliques of G.

Here, we focus on the subclass of interval graphs that have a simple path as clique graph and we give a dynamic programming algorithm which yields an optimum solution for the DkS problem on this subclass of interval graphs. Hence, in [17, 18] we prove that there is an $O(nk^3)$ algorithm for the DkS problem on interval graphs that have a simple path as clique graph. Recall that the DkS problem on chordal graphs is NP-hard [4], while its complexity on interval graphs still remains an open question.

3 The D*k*S problem on clique graphs

In this section we study graphs having as clique graph a star of cliques. Let $C_0, C_1, \ldots, C_{m-1}$ be the maximal cliques of such a star such that C_0 intersects with each other clique and no other intersection exists (by convention we denote by C_i both the clique C_i and the set of its vertices). Since such a star is the clique graph of a graph G, there is no edge of G between vertices belonging to different cliques.

We shall call C_0 central clique and all other cliques, $C_i,\ 1 \leq i \leq m-1$, exterior cliques. For each exterior clique C_i we denote by a_i the number of vertices in its intersection with C_0 i.e., $a_i = |C_i \cap C_0|$ and by b_i the number of its vertices outside C_0 i.e., $b_i = |C_i| - a_i > 0$. By C_0' we denote the clique consisting of the vertices of C_0 not belonging to any other clique i.e., $C_0' = C_0 \setminus \bigcup_{i=1}^{m-1} C_i$. By S we denote a solution to the DkS problem i.e., a subset of $|S| = k$ vertices, and by $E(S)$ we denote the number of edges in the subgraph induced by S. By S^* we denote an optimal solution to the DkS problem. By $n > k$ is denoted the total number of vertices in all cliques.

We say that a clique $C_i,\ 0 \leq i \leq m-1$, is completely in a solution S if all its vertices are in S. On the other hand, we say that the cliques C_0 and C_0' are partially in a solution S if a non-empty subset of their vertices, but not all, are in S. However, we say that an exterior clique $C_i,\ 1 \leq i \leq m-1$, is partially in S if a non-empty subset of its $C_i \setminus C_0$ vertices, but not all, are in S. We distinguish the definition of the partial inclusion in a solution S for an exterior clique C_i because if only some of its $C_i \cap C_0$ vertices are in S, they can be considered as vertices of C_0. In general we say that a clique is participating in a solution S if it is either completely or partially in S.

Concerning an optimal solution, S^*, we observe that if an exterior clique C_i is partially in S^*, then all its $|C_i \cap C_0| = a_i$ vertices are in S^*. Otherwise replacing a vertex $y \in C_i \setminus C_0,\ y \in S^*$ by a vertex $x \in C_i \cap C_0,\ x \notin S^*$ yields a better solution, a contradiction.
In the following we assume that:
(i) $k > |C_i|,\ i = 0, 1, \ldots, m-1$. Otherwise S^* consists of any subset of k vertices of some clique for which $|C_i| \geq k$.
(ii) $m > 2$. For $m = 1$ the point (i) holds. For $m = 2$, if $k > |C_0| \geq |C_1|$, then S^* consists

of the vertices of C_0 plus any subset of $k - |C_0|$ vertices of $C_1 \setminus C_0$.

Using these definitions and assumptions we prove some structural properties of an optimal solution S^*.

- At most one of the cliques $C_0', C_1, \ldots, C_{m-1}$ is partially in an optimal solution.
- If C_0 is the largest clique i.e., $|C_0| > |C_i|,\ 1 \leq i \leq m-1$, then C_0 belongs completely to every optimal solution.
- If C_0 is partially in an optimal solution S^*, then $|C_0| \leq |C_i|$ for every clique C_i participating in S^*.

Despite the nice structural properties of an optimal solution many natural greedy criteria based on the sizes of the cliques or/and the sizes of intersections fail to give such an optimal solution. In the following we are able to give a polynomial time dynamic programming algorithm for the case where the central clique is completely in the optimal solution and a polynomial time approximation scheme for the general case.

We prove that if clique C_0 is completely in the optimal solution, then there is an $O(nk^2)$ dynamic programming algorithm for the DkS problem on a star of cliques [15, 16]. Notice that if C_0 is the largest clique then, by the structural properties of the star graphs, C_0 belongs completely to every optimal solution and the above dynamic programming algorithm applies.

In the general case, C_0 is partially in the optimal solution and, by the structural properties of the star graphs, there are exterior cliques larger than C_0. Let c be the number of those cliques of size at least $|C_0|$. Moreover, we know that the cliques participating in the optimal solution are some of these c cliques. First we give a weak upper bound for the number c. If C_0 is partially in an optimal solution, then the number of exterior cliques of size at least $|C_0|$ is at most $\sqrt{n}$.

To proceed towards a polynomial time approximation scheme we argue further on the number of the exterior cliques of size at least $|C_0|$. We define $r = \lfloor \frac{k}{|C_0|} \rfloor$. Then the number of exterior cliques of size at least $|C_0|$ that can be involved in an optimal solution is at most r. Let also δ be a fixed number which will be defined later. Comparing r with δ we distinguish between two cases.

Case 1: $r < \delta$
If r is "small", then we proceed in an exhaustive manner. We examine all the possible sets of r cliques out of c cliques of size at least $|C_0|$ i.e., $\binom{c}{r}$ sets of cliques. A technical detail here is that clique C_0' should be also considered as one of the c cliques. It can be easily done by considering clique C_0' as an external clique with zero vertices outside clique C_0.

By the fact that the number of exterior cliques of size at least $|C_0|$ is at most $\sqrt{n}$, it follows that the number of all the $\binom{c}{r}$ sets of cliques is $O(n^{\frac{r}{2}})$. For each one of these sets of r cliques we compute the k vertices that maximize the number of edges as follows:

Let R be a set of r cliques. We have already proved that at most one of the cliques in R is partially in S^*. We consider all the $2^r - 1$ subsets of R. Let R_i be one of these subsets and let C_i^j be the $j^{\text{th}},\ 1 \leq j \leq |R_i|$, clique of the set R_i. Clearly if $\sum_{j=1}^{|R_i|} |C_i^j| < k$, we discard the set R_i. Otherwise, let $k(j) = \sum_{t=1, t \neq j}^{|R_i|} |C_i^t|$, for each $j = 1, 2, \ldots, |R_i|$. If $k(j) > k$ then we discard this j. Otherwise (if $k(j) \leq k$) we obtain a k-vertex solution by taking $k - k(j)$ vertices from clique C_i^j, starting from vertices which belong to its intersection with C_0.

Consider now all the solutions obtained for each $j = 1, 2, \ldots, |R_i|$, and for each $R_i \subseteq R$. By their construction, these solutions are all the possible k-vertex solutions for the set R of cliques, under the restriction that at most one of them is partially taken. Therefore, to find the optimal solution we simply have to choose the one with the maximum number of edges.

For a set R of r cliques, there are $2^r - 1$ subsets R_i, and for each subset there are at most r possible solutions. Therefore, the number of solutions to compare is $O(r\, 2^r)$. Recalling that we have to examine $O(n^{\frac{r}{2}})$ sets of r cliques, it follows that for the case $r < \delta$, δ be a fixed number, an optimal solution to the DkS problem in a star of cliques can be found in $O(r\, 2^r\, n^{\frac{r}{2}})$ time.

Case 2: $r \geq \delta$

If r is "large", then we proceed in a greedy manner. We consider the solution, S, obtained by the following simple algorithm:

Let $C_1 \geq C_2 \geq \ldots \geq C_{m-1}$ and t be the largest integer number such that $k \geq \sum_{i=1}^{t} |C_i| = k'$. Return all the vertices of the cliques $C_1 \geq C_2 \geq \ldots \geq C_t$ and $k - k'$ vertices of clique C_{t+1}.

We prove that if R_1 and R_2 are two sets of independent cliques with all cliques in R_1 of size at least L and all cliques in R_2 of size exactly L then for any pair of sets of k vertices S_1 and S_2 in R_1 and R_2, respectively, such that in both sets at most one clique is taken partially, it holds that $E(S_1) \geq E(S_2)$. This will be useful for bounding the deviation of our solution from the optimal one.

Let us now consider the solution S obtained by the algorithm. We know that the optimal solution S^* involves exterior cliques of size at least $|C_0|$. Since our algorithm finds a solution S by choosing k vertices from the larger exterior cliques, it follows that all cliques in S are of size at least $|C_0|$. Moreover, since $r = \lfloor \frac{k}{|C_0|} \rfloor$, we need at least r cliques of size $|C_0|$ in order to fill k. Hence, choosing k vertices from a set of independent cliques of size $|C_0|$, yields at least $rE(C_0)$ edges. Therefore, it follows that $E(S) \geq rE(C_0)$.

Clearly, an optimal solution, S^*, could contain cliques of smaller size than those chosen by our algorithm. These small cliques are selected by S^* due to the edges between their overlaps with C_0. Since these edges belong to C_0, the optimal solution cannot be greater than $E(S)$ plus the edges of C_0 i.e., $E(S^*) \leq E(S) + E(C_0) \leq E(S) + \frac{E(S)}{r} \leq E(S) + \frac{E(S)}{\delta} = E(S)\frac{\delta+1}{\delta}$. Thus, for the case $r \geq \delta$, where $\delta = \frac{1-\epsilon}{\epsilon}$, $0 < \epsilon < 1$, there is an $(1-\epsilon)$-approximation algorithm for the DkS problem in a star of cliques.

The complexity of the greedy approximation algorithm is $O(n\, logn)$. The complexity of the exhaustive optimal algorithm is exponential to $r \leq \delta = \frac{1-\epsilon}{\epsilon}$, that is exponential to $\frac{1}{\epsilon}$. Hence, we obtain a polynomial time approximation scheme for the DkS problem on a star of cliques [15, 16].

We also give a dynamic programming algorithm which yields an optimal solution for the DkS problem for graphs having as clique graph a tree. Let $C_1, C_2, ..., C_t$ be the cliques of such a tree and m its maximum degree. We consider $|C_i| < k$, $i = 1, \ldots, t$, otherwise the problem is trivial. The algorithm traverses the tree of cliques starting from the leaves cliques. In each step it computes an optimal solution for all the j-vertex densest subgraph problems, for $j = 1, ..., k$, on the subtree rooted at clique C_i. Thus, we can prove that there is an $O(nk^{m+1})$ algorithm for the DkS problem on a tree of cliques of maximum degree m [15, 16]. It directly follows that there is an $O(nk^3)$ optimal algorithm for the DkS problem on a path of cliques.

4 The DkS problem on chordal graphs

A graph is called *chordal* if every cycle of length strictly greater than three possesses a chord, that is, an edge joining two nonconsecutive vertices of the cycle. In the rest of this section by $G = (V, E)$ we denote a chordal graph. It is well known that for a chordal graph, $G = (V, E)$, the following hold:

(i) G has at most $m \leq |V|$ maximal cliques, $\mathcal{C} = \{C_1, C_2, \ldots, C_m\}$, which can be found in polynomial time [9].

(ii) G has a simplicial vertex, that is a vertex that all its adjacent vertices induce a complete subgraph in G. Actually, if G is not a clique, then it has two nonadjacent simplicial vertices [5].

(iii) G has a perfect elimination order, that is an order $\langle u_1, u_2, \ldots, u_n \rangle$ of the vertices of G in which each u_i is a simplicial vertex of the subgraph of G induced by the vertices $\{u_i, u_{i+1}, \ldots, u_n\}$. Moreover, any simplicial vertex can start such an order [8].

By G_A we denote the subgraph of G induced by a subset $A \subseteq V$ of its vertices and by G^F we denote the subgraph of G induced by a subset $F \subseteq E$ of its edges. A direct consequence of the definition of the class of chordal graphs is that being chordal is a hereditary property inherited by every vertex-induced subgraph G_A of G, but not by every edge-induced subgraph G^F of G. It is also obvious that for every maximal clique C_i of a vertex-induced or an edge-induced subgraph of G, there is at least one maximal clique C_j of G such that $C_i \subseteq C_j$.

By $E(A)$ we denote the set of edges in a subgraph G_A of G, while by $E(A, B)$ we denote the set of edges between two disjoint subsets $A, B \subseteq V$ of vertices of G i.e., the set of edges with one of their endpoints in A and the other in B. By S we denote a solution to the DkS problem, that is a subset $S \subseteq V$ such that $|S| = k$, while by S^* we denote an optimal solution, that is a solution S for which $|E(S)|$ is maximized. Finally, we assume that $k > |C_i|,\ 1 \leq i \leq m$, for otherwise S^* consists of any subset of k vertices of some clique for which $|C_i| \geq k$.

Since all the maximal cliques of a chordal graph $G = (V, E)$ can be found in polynomial time it is natural to study the DkS problem on those maximal cliques instead on G itself. In this section we analyze the following simple greedy algorithm for finding an approximate solution to the DkS problem on a chordal graph G.

Greedy Algorithm:

1. Let $C_1, C_2, \ldots C_m$ be the maximal cliques of G, sorted in non-increasing order of their sizes.
2. Find the largest integer t such that $k > |\bigcup_{i=1}^{t-1} C_i| = k'$.
3. Return the solution S consisting of all the vertices of the cliques $C_1, C_2, \ldots, C_{t-1}$ plus $k - k' > 0$ vertices of clique C_t.

The size of the maximal clique C_t plays a crucial role in our analysis and it will be denoted by $L = |C_t|$.

We first obtain a lower bound on the number of edges $|E(S)|$ in the solution S derived by the *Greedy Algorithm* for a chordal graph G. This bound is obtained by relating the solution S to the solution that the *Greedy Algorithm* returns for a graph consisting of independent cliques of size L. Formally, for a chordal graph G and the parameter L we consider the chordal graph $\widetilde{G}$ consisting of at least $\lceil k/L \rceil$ independent cliques all of size L. We can prove that if S and $\widetilde{S}$ are the solutions that the *Greedy Algorithm* returns for the DkS problem on graphs G and $\widetilde{G}$, respectively, then it holds that $|E(S)| \geq |E(\widetilde{S})| = \frac{k(L-1)-b(L-b)}{2}$, where $b = k \bmod L$.

Next we give un upper bound, which is of independent interest, on the number of edges of a chordal graph as a function of the size of its maximum clique.

Let $c \geq 2$ be the size of a maximum clique of a chordal graph $G = (V, E)$. The graph G, as a chordal one, has a perfect elimination order. We remove from G vertices (and their incident edges) in a perfect elimination order until the remaining number of its vertices is c. Since the size of a maximum clique of G is c, each removed vertex has degree at most $c - 1$. Thus, the number of the edges removed during this process, let $|E_1|$, is at most $(|V| - c)(c - 1)$. Moreover, the remaining number of edges, let $|E_2|$, is at most $\frac{c(c-1)}{2}$ (the remaining c vertices form a clique). Thus, $|E| = |E_1| + |E_2| \leq (|V| - c)(c - 1) + \frac{c(c-1)}{2} = (c - 1)(|V| - \frac{c}{2})$ and this bound is the best possible [19, 20].

To prove that this bound is the best possible consider the chordal graph $G = (V, E)$ consisting of a clique, C, of size $c - 1$ and $|V| - c + 1$ independent vertices each one of them adjacent to all vertices of C. Observe that a maximum clique of G consists of the clique C plus one of the independent vertices, and it is of size c. For this graph G it holds that $|E| = \frac{(c-1)(c-2)}{2} + (|V| - c + 1)(c - 1) = (c - 1)(|V| - \frac{c}{2})$. Note that if $c = |V|$, then G becomes a complete graph.

Let us now relate the solution S of the *Greedy Algorithm* to an optimal solution S^* to the DkS problem on a chordal graph G. Let $S^* = A \cup B$, where $A = S^* \cap S$ is the subset of vertices of S^* that belong to S and $B = S^* \setminus A$ is the subset of vertices of S^* that do not belong to S. Let also $\Gamma \subseteq A$ be the subset of vertices in A that have adjacent vertices in B and $F = E(B) \cup E(\Gamma, B)$. Obviously, $\Gamma \cap B = \emptyset$ and $|E(S^*)| = |E(A)| + |E(B)| + |E(\Gamma, B)| = |E(A)| + |F|$.

In order to bound the number of edges in an optimal solution S^* we shall consider the edge-induced subgraph $G^F = (\Gamma \cup B, F)$ as well as the vertex-induced subgraph $G_{B \cup \Gamma}$ of G. Note that $G_{B \cup \Gamma}$, as a vertex-induced subgraph of G, is a chordal graph, while G^F, as an edge-induced subgraph of G, is in general a non chordal one. A useful structural property of the subgraph G^F is that all the maximal cliques of the graph $G^F = (\Gamma \cup B, F)$ are of size at most L. Using this property we can now prove that for the edge-induced graph $G^F = (\Gamma \cup B, F)$ of G it holds that $|F| \leq (L-1)(|\Gamma \cup B| - \frac{L}{2})$. Applying this bound on G^F, (with $|\Gamma \cup B| \leq |S^*| = k$) we obtain $|F| = |E(B)| + |E(\Gamma, B)| \leq (L - 1)(k - \frac{L}{2})$.

For the edges $E(A)$ it holds that $|E(A)| \leq |E(S)|$, since $A \subseteq S$. We also know that $|E(S)| \geq \frac{k(L-1)-b(L-b)}{2}$. Therefore, $\frac{|E(S^*)|}{|E(S)|} = \frac{|E(A)|+|F|}{|E(S)|} \leq 1 + \frac{|F|}{|E(S)|} \leq 1 + \frac{(L-1)(2k-L)}{k(L-1)-b(L-b)}$. By recalling that $b = k \bmod L \leq L - 1$ and by distinguishing between two cases for b ($b \leq L/2$ and $b > L/2$) it is easy to prove that $\frac{(L-1)(2k-L)}{k(L-1)-b(L-b)} \leq 2$. Thus, it follows that there is a 3-approximation algorithm for the DkS problem on chordal graphs [19, 20].

5 Concluding remarks

At first we gave polynomial time algorithms for the HkS problem on graphs of maximal degree 2 and on trees either if we ask for a connected or a disconnected solution. We also proposed a dynamic programming algorithm for the DkS problem on interval graphs that have a simple path as clique graph.

Then we presented a polynomial time approximation scheme for the DkS problem on a star of cliques and an $O(nk^{m+1})$ time algorithm for the same problem on a tree of cliques, where n is the total number of vertices in all the cliques and m is the maximum degree of the tree. This last algorithm gives an $O(nk^3)$ optimal algorithm for paths of cliques. Since interval and chordal graphs can be seen as clique graphs our result could be exploited in the direction of exploring the complexity and the approximability of the DkS problem in these classes of graphs.

Finally, we gave a 3-approximation algorithm for the DkS problem on chordal graphs, which, up to our knowledge, is the first constant approximation algorithm for an NP-hard variant of the problem on non-dense graphs. Concerning the tightness of our analysis we succeeded to construct counterexamples for which our algorithm gives a solution of at least half of the edges of an optimal one.

On the other hand many questions concerning the frontier between hard and polynomial solvable or approximable, within a constant ratio, variants of the DkS problem, remain open. Such an outstanding open question concerns the complexity of the DkS problem on interval graphs or even on proper interval graphs. The existence of a constant approximation algorithm for the NP-hard variant of the DkS problem on planar graphs is another interesting open question.

References

1. S. Arora, D. Karger, and M. Karpinski. Polynomial time approximation schemes for dense instances of NP-hard problems. In *Proceedings of the 27th Annual ACM Symposium on Theory of Computing*, pages 284–293, 1995.
2. Y. Asahiro, K. Iwama, H. Tamaki, and T. Tokuyama. Greedily finding a dense subgraph. *Journal of Algorithms*, 34(2):203–221, 2000.
3. A. Billionnet and F. Roupin. A deterministic algorithm for the densest k-subgraph problem using linear programming. Technical Report No486, CEDRIC, CNAM-IIE, Paris, 2004.
4. D. G. Corneil and Y. Perl. Clustering and domination in perfect graphs. *Discrete Applied Mathematics*, 9:27–39, 1984.
5. G. A. Dirac. On rigid circuit graphs. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 25:71–76, 1961.
6. U. Feige, G. Kortsarz, and D. Peleg. The dense k-subgraph problem. *Algorithmica*, 29(3):410–421, 2001.
7. U. Feige and M. Seltser. On the densest k-subgraph problem. Technical Report CS97-16, Weizmann Institute, 1997.
8. D. R. Fulkerson and O. A. Gross. Incidence matrices and interval graphs. *Pacific Journal of Mathematics*, 15:835–855, 1965.
9. F. Gavril. Algorithms for minimum coloring, maximum clique, minimum covering by cliques and maximum independent set of chordal graph. *SIAM Journal on Computing*, 1(2):180–187, 1972.
10. O. Goldschmidt and D. Hochbaum. k-edge subgraph problems. *Discrete Applied Mathematics and Combinatorial Operations Research and Computer Science*, 74(2):159–169, 1997.
11. R. Hassin, S. Rubinstein, and A. Tamir. Approximation algorithms for maximum dispersion. *Operations Research Letters*, 21(3):133–137, 1997.
12. J. M. Keil and T. B. Brecht. The complexity of clustering in planar graphs. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 9:155–159, 1991.

13. S. Khot. Ruling out PTAS for graph min-bisection, densest subgraph and bipartite clique. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 136–145, 2004.
14. N. Laoutaris, V. Zissimopoulos, and I. Stavrakakis. Joint object placement and node dimensioning for internet content distribution. *Information Processing Letters*, 89(6):273–279, 2004.
15. M. Liazi, I. Milis, F. Pascual, and V. Zissimopoulos. The densest k-subgraph problem on clique graphs. *Journal of Combinatorial Optimization*, 14(4):465–474, 2007.
16. M. Liazi, I. Milis, F. Pascual, and V. Zissimopoulos. The densest k-subgraph problem on clique graphs. In *International Combinatorics, Geometry and Computer Science Conference*, Marseille, 2007.
17. M. Liazi, I. Milis, and V. Zissimopoulos. Polynomial variants of the densest/heaviest k-subgraph problem. In *APPOL II, Final Workshop on Approximation and on line Algorithms*, Athens, 2004.
18. M. Liazi, I. Milis, and V. Zissimopoulos. Polynomial variants of the densest/heaviest k-subgraph problem. In *20th British Combinatorial Conference*, Durham, 2005.
19. M. Liazi, I. Milis, and V. Zissimopoulos. The densest k-subgraph problem on chordal graphs. In *2nd Athens Colloquium on Algorithms and Complexity*, Athens, 2007.
20. M. Liazi, I. Milis, and V. Zissimopoulos. A constant approximation algorithm for the densest k-subgraph problem on chordal graphs. *Information Processing Letters*, 108(1):29–32, 2008.
21. F. Maffioli. Finding a best subtree of a tree. Technical Report 91.041, Politechnico di Milano, Dipartimento di Elektronica, 1991.
22. Y. Perl and Y. Shiloach. Efficient optimization of monotonic functions on trees. *SIAM Journal on Algebraic and Discrete Methods*, 4(4):512–516, 1983.
23. D. J. Jr. Rader and G. J. Woeginger. The quadratic 0-1 knapsack problem with series-parallel support. *Operation Research Letters*, 30(3):159–166, 2002.
24. S. S. Ravi, D. J. Rosenkrantz, and G. K. Tayi. Heuristic and special case algorithms for dispersion problems. *Operations Research*, 42(2):299–310, 1994.