The densest k-subgraph problem on clique graphs

M. Liazi[1]*, I. Milis[2], F. Pascual[3] and V. Zissimopoulos[1]**

[1] Department of Informatics and Telecommunications,
University of Athens, 157 84 Athens, Greece
{mliazi,vassilis}@di.uoa.gr

[2] Department of Informatics,
Athens University Economics and Business,
104 34 Athens, Greece
milis@aueb.gr

[3] IBISC, University of Évry,
523, Place des Terrasses de l'agora,
91000 Évry, France
fpascual@lami.univ-evry.fr

Abstract. The Densest k-Subgraph (DkS) problem asks for a k-vertex subgraph of a given graph with the maximum number of edges. The problem is strongly NP-hard, as a generalization of the well known *Clique* problem and we also know that it does not admit a Polynomial Time Approximation Scheme (PTAS). In this paper we focus on special cases of the problem, with respect to the class of the input graph. Especially, towards the elucidation of the open questions concerning the complexity of the problem for interval graphs as well as its approximability for chordal graphs, we consider graphs having special clique graphs. We present a PTAS for stars of cliques and a dynamic programming algorithm for trees of cliques.

Key Words: Densest k-subgraph, Clique graph, Polynomial Time Approximation Scheme, Dynamic programming

* The project is co-financed within Op. Education by the ESF (European Social Fund) and National Resources.

** Partially supported by the Special Research Grants Account of the University of Athens under Grant 70/4/5821.

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 generalization of the well known *Maximum Clique* problem. In the weighted version of the DkS we 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 [2, 3, 5–7, 10, 14, 19, 20] 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 the most recent of these articles [3]. However, the best known approximation ratio for the DkS problem, which performs well for all values of k, is $O(n^{\delta})$, for some $\delta < \frac{1}{3}$ [5]. On the other hand, it has been shown that the DkS problems does not admit a Polynomial Time Approximation Scheme (PTAS) [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 [18] and 2 [11] have been proposed for the weighted DkS problem on complete graphs where the weights satisfy the triangle inequality.

The DkS problem is trivial on trees (any subtree of k vertices contains exactly $k - 1$ edges). It is also known that DkS is polynomial for graphs of maximal degree two [7] as well as for cographs, split graphs and k-trees [4]. On the other hand the DkS problem remains NP-hard for bipartite graphs [4], even of maximum degree three [7], as well as for comparability graphs, chordal graphs [4] and planar graphs [12]. The weighted version of the DkS problem is polynomial on trees either if we ask for a connected solution [9, 15, 16] or not [17]. In fact, the result for the later case is implied by a result for the solution of the quadratic 0-1 knapsack problem on edge series-parallel graphs in [17].

An outstanding open question concerns the complexity of the DkS problem on interval graphs as well as its approximability for chordal graphs. Towards this direction we focus, in this paper, on chordal or interval graphs having special clique graphs. A *clique* of an undirected graph, $G = (V, E)$, is a subset of its vertices inducing a complete subgraph in G. 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. It is well known that all maximal cliques, and hence the clique graph, of a chordal graph can be found in polynomial time [8]. It is, clearly, convenient to study the DkS problem on the clique graph of a chordal graph G instead on the G itself.

In this paper we consider chordal or interval graphs having special clique graphs, in order to further identify the frontier between hard and polynomial solvable/approximable variants of the the DkS problem. In the next section we present a PTAS for graphs that having as clique graph a star (star of cliques) and in Section 3 we present an $O(nk^{m+1})$ time dynamic programming algorithm for graphs having as clique graph a tree (tree of cliques) of maximum degree m. This algorithm gives an $O(nk^3)$ time algorithm for graphs having as clique graph a path (path of cliques). Note that, in general, stars of cliques as well as trees of cliques are neither graphs of minimum degree $\Omega(n)$ nor dense graphs (of $\Omega(n^2)$ edges) for which a PTAS is already known [1].

2 The DkS problem on stars of cliques

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 the 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 give in the next propositions some structural properties of an optimal solution S^*.

Proposition 1. *At most one of the cliques $C_0', C_1, \ldots, C_{m-1}$ is partially in an optimal solution.*

proof: We prove first that at most one of the exterior cliques is partially in S^*. Suppose that two exterior cliques $C_i, C_j,\ 1 \leq i \neq j \leq m-1$ are partially in S^* and assume w.l.o.g. that $|S^* \cap C_i| \geq |S^* \cap C_j|$.

Let $x \notin S^*$ be a vertex in $C_i \setminus C_0$ and $y \in S^*$ be a vertex in $C_j \setminus C_0$. Then consider the solution S in which we replace y by x. Then, $E(S) = E(S^*) - (|S^* \cap C_j| - 1) + |S^* \cap C_i| \geq E(S^*) + 1$, a contradiction to the optimality of S^*.

To complete the proof it suffices to prove that is not possible both clique C_0' and an exterior clique C_j to be partially in S^*. This fact follows by using the same arguments as before, but now we consider C_0 instead of C_i and $x \notin S^*$ to be a vertex in C_0'. □

Proposition 2.

(i) 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.

(ii) If C_0 is partially in an optimal solution S^*, then $|C_0| \leq |C_i|$ for every clique C_i participating in S^*.

proof:

(i) Suppose that S^* does not contain some $q > 0$ vertices of C_0 and consider a solution S obtained from S^* by replacing q vertices of exterior cliques not in C_0 by the q vertices of C_0 not in S^*. Let us denote by E^- and E^+ the number of edges which are removed and inserted, respectively, to $E(S^*)$ by this replacement. Then, $E(S) = E(S^*) - E^- + E^+$. E^- equals to the number of edges that q vertices of exterior cliques contribute to $E(S^*)$. This number, even if all the q vertices belong to the same exterior clique, is strictly less than $\binom{q}{2} + (|C_0| - q)q$. On the other hand, E^+ equals to the number of edges that the q vertices of C_0 not in S^* will contribute to $E(S)$. This number is equal to $\binom{q}{2} + (|C_0| - q)q$. Therefore, $E(S) > E(S^*)$, a contradiction to the optimality of S^*.

(ii) Suppose that there is an exterior clique C_i in S^* such that $|C_0| > |C_i|$ and that S^* does not contain some $q > 0$ vertices of C_0. Notice that $|C_0| - q > a_i$, since if $|C_0| - q \leq a_i$, then no other exterior clique participates in S^*, that is S^* is part of a single clique (either C_i or C_0).

Consider a solution S obtained from S^* by replacing vertices of $S^* \cap C_i$ not in C_0 by vertices of C_0 not in S^*. Let $b'_i = |S^* \cap (C_i \setminus C_0)|$, $0 < b'_i \leq b_i$. Using again E^- and E^+ as in part (i) we have $E(S) = E(S^*) - E^- + E^+$. Now we distinguish between two cases w.r.t. the values of q and b'_i.

If $q \geq b'_i$ then E^- equals to the number of edges that b'_i vertices of the exterior clique C_i contributes to $E(S^*)$ while E^+ equals to the number of edges that the b'_i vertices of C_0 not in S^* will contribute to $E(S)$. Then $E(S) = E(S^*) - E^- + E^+ = E(S^*) - (\binom{b'_i}{2} + b'_i a_i) + (\binom{b'_i}{2} + b'_i(|C_0| - q)) = E(S^*) + b'_i((|C_0| - q) - a_i) > E(S^*)$, a contradiction to the optimality of S^*.

If $q < b'_i$ then E^- equals to the number of edges that q vertices of the exterior clique C_i contributes to $E(S^*)$ while E^+ equals to the number of edges that the q vertices of C_0 not in S^* will contribute to $E(S)$. Then $E(S) = E(S^*) - E^- + E^+ = E(S^*) - (\binom{q}{2} + q(a_i + b'_i -$

$q)) + (\binom{q}{2} + q(|C_0| - q)) = E(S^*) + q(|C_0| - (a_i + b'_i)) \geq E(S^*) + q(|C_0| - |C_i|) > E(S^*)$, a contradiction to the optimality of S^*. □

Despite the nice structural properties of an optimal solution in Propositions 1 and 2, 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 PTAS for the general case.

2.1 Clique C_0 is completely in the optimal solution.

Lemma 1. *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.*

proof: Since clique C_0 is completely in the optimal solution we have to choose $k' = k - |C_0|$ vertices from exterior cliques. If we choose q vertices from an exterior clique C_j, then they contribute $q \cdot a_j + \binom{q}{2}$ edges to the solution.

Let $f(i, j)$ be the maximum number of edges in a solution choosing i vertices from the first j exterior cliques (recall that there are $m-1$ exterior cliques). Thus for $i = 0, 1, 2, \ldots, k'$ and $j = 2, 3, \ldots, m - 1$

$$f(i, j) = \max_{0 \leq q \leq \min\{i, b_j\}} \{f(i - q, j - 1) + q \cdot a_j + \binom{q}{2}\}$$

For $j = 1$ the following boundary conditions hold for $0 \leq i \leq \min\{k', b_1\}$

$$f(i, 1) = \begin{cases} \binom{i}{2} + i \cdot a_1, & \text{if} \quad i \leq \min\{k', b_1\} \\ -\infty, & \text{otherwise} \end{cases}$$

The complexity of the dynamic programming algorithm is $O(nk^2)$. The computation of a single $f(i,j)$ value takes $O(k)$ time due to the possible values of q ($0 \leq q \leq \min\{i, b_j\} \leq k' < k$) and $f(i,j)$ values are computed for every $i \leq k' < k$ and $j \leq m-1 < n$. The optimal solution, for the DkS problem is $f(k', m-1) + \binom{|C_0|}{2}$. □

Notice that if C_0 is the largest clique, then, by Proposition 2(i), C_0 belongs completely to every optimal solution and the above dynamic programming algorithm applies.

2.2 A PTAS for the general case.

In the general case, C_0 is partially in the optimal solution and, by Proposition 2(i), there are exterior cliques larger than C_0. Let c be the number of those cliques of size at least $|C_0|$. Moreover, by Proposition 2(ii), the cliques participating in the optimal solution are some of these c cliques. Next proposition gives a weak upper bound for the number c.

Proposition 3. *If C_0 is partially in an optimal solution, then the number of exterior cliques of size at least $|C_0|$ is at most $\sqrt{n}$.*

proof: The number, c, of exterior cliques is smaller than or equal to $|C_0|$, since $C_i \cap C_0 \neq \emptyset$ and $C_j \cap C_i = \emptyset$, $1 \leq i \neq j \leq m-1$. Thus, if $|C_0| \leq \sqrt{n}$, then $c \leq \sqrt{n}$. Otherwise $|C_0| > \sqrt{n}$. Then, the total number of vertices in these c cliques is at least $c \times \sqrt{n}$ and at most n. Hence, $c \leq \sqrt{n}$. □

To proceed towards a PTAS 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 Proposition 3 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. By Proposition 1, 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^{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, the next lemma follows.

Lemma 2. *For the case $r < \delta$, δ be a fixed number, an optimal solution for 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}.

Next proposition for the case of independent cliques will be useful for bounding the deviation of our solution from the optimal one.

Proposition 4. *Let R_1 and R_2 be 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. 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 $E(S_1) \geq E(S_2)$.*

proof: Transform S_1 to an equivalent to S_2 set as following. First, remove from each clique in S_1 some vertices such that each clique in S_1 has now size exactly L. Let k' the number of the removed vertices. Then, replace the k' vertices with $\lceil \frac{k'}{L} \rceil$ cliques, all, but one, of size exactly L. All the removed vertices now have degree at most $L-1$ while in S_1 they had degree at least $L-1$. Thus, $E(S_1) \geq E(S_2)$. □

Let us now consider the solution S obtained by our algorithm. By Proposition 2(ii), 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, by Proposition 4, 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, next lemma follows.

Lemma 3. *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 of Lemma 3 is $O(n\, logn)$. The complexity of the exhaustive optimal algorithm of Lemma 2 is exponential in $r \leq \delta = \frac{1-\epsilon}{\epsilon}$, that is exponential in $\frac{1}{\epsilon}$. Hence, we obtain

Theorem 1. *There is a polynomial time approximation scheme for the DkS problem in stars of cliques.*

3 The D*k*S problem on trees of cliques.

In this section we present 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.

We consider the tree rooted at a leaf clique, say clique C_t. This way the root clique C_t has at least one vertex outside its intersections with its children cliques. Let C_i be a non-leaf clique with $m_i \geq 1$ children, $C_{i_1}, \ldots, C_{i_{m_i}}$. We denote by Q_h the intersection of C_i with its h^{th} child clique, C_h, for $h = i_1, \ldots, i_{m_i}$ i.e., $Q_h = C_i \cap C_h$. We denote by F_i the intersection of the clique C_i with its father clique, C_f, in the tree i.e., $F_i = C_i \cap C_f$ and by D_i the vertices of a clique C_i not belonging to any intersection i.e., $D_i = C_i - F_i - \bigcup_{h=i_1}^{i_{m_i}} Q_h$. By convention we consider F_t, for the root clique C_t, to be consisted of a single vertex i.e., $|F_t| = 1$.

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 (DjS) problems, for $j = 1, ..., k$, on the subtree rooted at clique C_i.

We denote by $f_i(j)$ the value of the optimal solution of the DjS problem on the subtree rooted at clique C_i. By $f_i(j, a)$ we denote the value of an optimal solution to the DjS problem on the subtree rooted at clique C_i including *exactly* a vertices from the clique F_i. It is clear that

$$f_i(j) = \max_{0 \leq a \leq |F_i|} \{f_i(j, a)\}.$$

To compute an $f_i(j, a)$ value for a non leave clique C_i we consider its children cliques $C_{i_1}, \ldots, C_{i_{m_i}}, m_i \geq 1$. Let $f_h(j_h, a_h)$ be the value of an optimal solution of the j_h-vertex densest subgraph, for $j_h = 1, 2, ..., k$, on the subtree rooted at clique C_h, $h = i_1, \ldots, i_{m_i}$, using a_h vertices of F_h. Note that for the intersection of the clique C_i with its child C_h it holds that $Q_h = C_i \cap C_h = F_h$. We compute the value of $f_i(j, a)$, as follows:

If $a = 0$, then no vertex of $F_i \cup D_i$ belongs to the optimal solution of the corresponding DjS problem on the subtree rooted at C_i. Therefore, the value of this solution is the same as the value of the optimal solution to the DjS problem on the subgraph which is the union of the subtrees rooted at the children of C_i plus the edges between the vertices in their

Q_h's, that is

$$f_i(j,a) = \max_{\sum_{h=i_1}^{i_{m_i}} j_h = j} \{\sum_{h=i_1}^{i_{m_i}} f_h(j_h, a_h) + \sum_{\substack{i,j=i_1 \\ i \neq j}}^{i_{m_i}} \frac{a_i \cdot a_j}{2}\}.$$

If $a > 0$, then an optimal solution to the DjS problem including a vertices of F_i can also include $b \geq 0$ vertices of D_i, and $a_h \geq 1$ vertices [4] of each Q_h, $h = i_1, \ldots, i_{m_i}$. Then,

$$f_i(j,a) = \begin{cases} \binom{j}{2}, & \text{if } j \leq \sum_{h=i_1}^{i_{m_i}} |Q_h| + |D_i| + a \\ \max_{a+b+\sum_{h=i_1}^{i_{m_i}} j_h = j} \{\sum_{h=i_1}^{i_{m_i}} f_h(j_h, a_h) + \binom{a+b}{2} + (a+b) \sum_{h=i_1}^{i_{m_i}} a_h + \\ \qquad \sum_{\substack{i,j=i_1 \\ i \neq j}}^{i_{m_i}} \frac{a_i \cdot a_j}{2}\}, & \text{otherwise.} \end{cases}$$

For all cliques C_i that are leaves in the tree the following boundary conditions hold for $1 \leq j \leq k$:

If $a = 0$, then $f_i(j,a) = 0$.

If $1 \leq a \leq |F_i|$, then

$$f_i(j,a) = \begin{cases} \binom{j}{2}, & \text{if } j \leq |D_i| + a \\ -\infty, & \text{otherwise.} \end{cases}$$

The algorithm terminates by computing the value $f_t(k)$ for the root clique C_t. Recall that we consider $|F_t| = 1$ and thus the optimal solution for the k-vertex densest subgraph problem is $f_t(k) = \max_{a=0,1}\{f_t(k,a)\}$.

The computation of a single $f_i(j)$ value for a clique C_i with m_i children takes $O(k^{m_i+1})$ time due to the combinations of a, b and $\sum_{h=i_1}^{i_{m_i}} j_h$, such that $a + b + \sum_{h=i_1}^{i_{m_i}} j_h = j$. The

[4] or $a_h \geq 0$ for a disconnected solution.

algorithm computes $f_i(j)$, for every $j = 1, 2, ..., k$ and for every $i = 1, 2, ..., t$. Since in the worst case t is $O(n)$ and $\max_i\{m_i\} = m - 1$ the next theorem follows:

Theorem 2. *There is an $O(nk^{m+1})$ algorithm for the DkS problem on a tree of cliques of maximum degree m.*

Next corollary follows directly from Theorem 2.

Corollary 1. *There is an $O(nk^3)$ optimal algorithm for the DkS problem on a path of cliques.*

4 Conclusions

We have presented a PTAS for the densest k-subgraph problem on a star of cliques and an $O(nk^{m+1})$ time optimal algorithm for the same problem on trees of cliques, where n is the total number of vertices in all the cliques and m 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.

Acknowledgment: The authors would like to thank the two anonymous referees for their valuable comments that significantly improved the presentation.

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. U. Feige, G. Kortsarz, and D. Peleg. The dense k-subgraph problem. *Algorithmica*, 29(3):410–421, 2001.

6. U. Feige and M. Langberg. Approximation algorithms for maximization problems arising in graph partitioning. *Journal of Algorithms*, 41(2):174–211, 2001.

7. U. Feige and M. Seltser. On the densest k-subgraph problem. Technical Report CS97-16, Weizmann Institute, 1997.

8. 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.

9. O. Goldschmidt and D. Hochbaum. k-edge subgraph problems. *Discrete Applied Mathematics and Combinatorial Operations Research and Computer Science*, 74(2):159–169, 1997.

10. Q. Han, Y. Ye, and J. Zhang. An improved rounding method and semidefinite programming relaxation for graph partition. *Mathematical Programming*, 92(3):509–535, 2002.

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. G. Kortsarz and D. Peleg. On choosing a dense subgraph. In *Proceedings of the 34th Annual IEEE Symposium on Foundations of Computer Science*, pages 692–701, 1993.

15. F. Maffioli. Finding a best subtree of a tree. Technical Report 91.041, Politechnico di Milano, Dipartimento di Elektronica, 1991.

16. Y. Perl and Y. Shiloach. Efficient optimization of monotonic functions on trees. *SIAM Journal of Algebric and Discrete Methods*, 4(4):512–516, 1983.

17. 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.

18. 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.

19. A. Srivastav and K. Wolf. Finding dense subgraphs with semidefinite programming. In *Proceedings of the International Workshop on Approximation Algorithms for Combinatorial Optimization*, pages 181–191, 1998.

20. Y. Ye and J. Zhang. .519 approximation of dense-n/2-subgraph. Working Paper, Department of Management Sciences, Henry B. Tippie College of Business, The University of Iowa, 1999.