ELSEVIER
Theoretical Computer Science 161 (1996) 307–314
Theoretical Computer Science

Note

Constructing the highest degree subgraph for dense graphs is in $\mathcal{NCAS}$

Alexander E. Andreev [a,1], Andrea E.F. Clementi [b,1], José D.P. Rolim [c,*]

[a] *Department of Mathematics and Mechanics, University of Moscow, Russia*
[b] *Dip. di Scienze dell'Informazione, Universitá "La Sapienza" di Roma, Italy*
[c] *Centre Universitaire d'Informatique, Université de Genève, Switzerland*

Received 1 July 1995

Abstract

In this paper, we first show that the *Highest Degree Subgraph* problem remains $\mathcal{P}$-complete for *dense* graphs (i.e. when $m = \Omega(n^2/\text{poly}\log n)$). This hardness result gives a clear motivation in studying the approximability of the *Highest Degree Problem* even for this restricted case. We then provide an $\mathcal{NC}$-approximation scheme computing approximate solutions for *dense* graphs, thus proving that, in this case, the problem belongs to the $\mathcal{NCAS}$ class.

1. Introduction

In this paper, we study the *Highest Degree Subgraph* problem, denoted as the HDS problem, that consists of searching an induced subgraph of a given graph whose minimum vertex degree is maximum. Besides being a natural problem in graph theory, this problem was previously studied in [9] to develop parallel techniques to approximate other $\mathcal{P}$-complete problems and in [6] to investigate strictly related problems having different complexities. The HDS problem has also consequences on the study of network reliability since, informally speaking, any constructive algorithm for this problem provides the part of a given network which maximizes the minimum number of connections that must fail in order to completely isolate a network agent. Furthermore, in some cases, the study of the HDS problem gives useful information to investigate important fault-tollerance parameters such as vertex and edge connectivity of a network.

In [1, 9], Anderson and Mayr revealed a 'threshold' behavior of the parallel complexity of the HDS problem. On one hand, besides being $\mathcal{P}$-complete, the HDS problem

* Corresponding author.

[1] Supported by the Swiss National Science Foundation, grant 21-43309.95, and by the project INTAS 94-3936.

PII S0304-3975(96)00014-X

cannot be approximated by any $\mathcal{NC}$-algorithm with approximation ratio less or equal than two. On the other hand, the same authors introduced an elegant $\mathcal{NC}$-approximation scheme that achieves any constant approximation ratio greater than two. However, this algorithm does not provide approximate *constructive solutions* but only their measure. Indeed, given a graph G, it performs a parallel 'pruning' procedure (see [1, 8]) that provides an integer value d such that: (i) an induced subgraph H of G exists having minimum degree not less than d and, (ii) no induced subgraph of G exists having minimum degree more than $(2+\delta)d$, where δ is an arbitrary positive constant. The algorithm is strongly based on a combinatorial lemma, due to Erdös [5], that states that in any graph with n vertices and m edges an induced subgraph exists having minimum degree at least $\lceil m/n \rceil$. More precisely, the algorithm would turn into a constructive one if the proof of the Erdös' lemma could be efficiently parallelized. However, in [2], the $\mathcal{P}$-completeness of this proof is shown thus suggesting that, for general graphs, a more sophisticated (or completely different) approach should be found (notice that two different proofs have been proposed in literature for Erdös' lemma and both of them have been shown to be $\mathcal{P}$-complete in [2]). To our knowledge, no $\mathcal{NC}$-approximation algorithm for the HDS problem has been found even for particular classes of graphs (clearly, we are interested in non-'trivial' classes or, even better, classes of graphs for which the problem is still $\mathcal{P}$-complete). We thus focus on the class of dense graphs (in the sequel, for a *dense* graph we will always intend a graph $G(V,E)$ for which $m = \Omega(n^2/\text{poly}\log n)$, where $|V| = n$, $|E| = m$). Recent works on approximation algorithms for 'hard' problems on dense graphs, or more generally, on 'dense' instances can be found in [3, 4, 10] (density for non-graph problem can be easily defined similarly). In Section 3, we show that the HDS problem is still $\mathcal{P}$-complete even in the case of dense graphs. The proof consists in a simple reduction from the HDS problem for general graphs. The interest in this 'hardness' result lies in the fact that, if exact (i.e. optimum) solutions are sought, then the *dense* property is not helpful from a parallel complexity point of view. We thus have a clear motivation in studying the approximability of the HDS problem even for this restricted case.

In Section 4, we present an $\mathcal{NC}$-approximation scheme for this problem in the case of dense graphs. More precisely, our algorithm computes a vertex subset H which induces a subgraph having minimum degree $d(H)$ such that

$$\frac{d^*(G)}{d(H)} \leqslant \frac{1}{1-\varepsilon},$$

where $d^*(G)$ denotes the measure of an optimum solution for the input graph G and ε is any constant such that $0 < \varepsilon < 1$. Furthermore, on input $G(V,E)$ where $m = \Omega(n^2/\log^q n)$ (for some constant $q \geqslant 0$), the algorithm runs in $O((1/\varepsilon)\log^{q+2} n)$ parallel time, on a *SIMD CREW PRAM*, using $O(nm)$ processors. Notice that, for $m = \Theta(n^2)$ our algorithm has a parallel complexity equivalent to that of the Anderson and Mayr's algorithm.

2. Preliminaries

Given a graph $G(V,E)$ (with $V = \{1,\ldots,n\}$ and $|E| = m$), we denote the degree of a vertex v as $d(v)$ and, moreover, the notation $d_H(v)$ denotes the degree of v in the subgraph induced by the subset $H \subseteq V$. We then define the following functions:

$$d(H) = \min\{d_H(v):\ v \in H\} \quad \text{and} \quad d^*(G) = \max\{d(H):\ H \subseteq V\}\ .$$

The HDS problem can be defined as follows. *Instance*: a graph $G(V,E)$. *Question*: Find a subset $H \subseteq V$ such that $d(H) = d^*(G)$.

Basic definitions on $\mathcal{NC}$, $\mathcal{P}$-completeness and parallel approximation algorithms for $\mathcal{P}$-complete problems and the importance of these notions can be found in [7, 11]. We adapt here the standard notion of efficient parallel approximability to the above maximization problem. Let r be any fixed constant ($r > 1$), an *r-$\mathcal{NC}$-approximation algorithm* for the HDS problem is an $\mathcal{NC}$-algorithm that, given a graph G, generates a vertex subset H such that $d^*(G)/d(H) \leqslant r$. The HDS problem admits an *$\mathcal{NC}$-approximation scheme* if there is an $\mathcal{NC}$-algorithm that, given a graph G and an input *parameter* $r > 1$, generates a vertex subset H such that $d^*(G)/d(H) \leqslant r$. In general, an optimization problem is in the class $\mathcal{NCAS}$ if it can be solved by an $\mathcal{NC}$-approximation scheme. Notice that, according to the previous definition, no bound on the complexity of the algorithm is required with respect to the parameter r.

3. A simple hardness result

In this section, we show that the HDS problem is still $\mathcal{P}$-complete even in the case of dense graphs. The proof of this fact is a simple reduction from the HDS problem for general graphs. This hardness result provides a clear motivation in seeking approximate solutions for the HDS problem also in this restricted case.

As mentioned in the Introduction, for general graphs, the following results hold.

Theorem 1 (Anderson and Mayr [1]). *The* HDS *problem is $\mathcal{P}$-complete. Moreover, the* HDS *problem does not admit r-$\mathcal{NC}$-approximation algorithms for any constant $r \leqslant 2$, unless $\mathcal{P} = \mathcal{NC}$. On the other hand, there is an $\mathcal{NC}$-scheme which r-approximates the value* $d^*(G)$, *for any $r > 2$.*

The $\mathcal{P}$-completeness of the HDS problem holds even in the restricted case of dense graphs.

Corollary 2. *The* HDS *problem is $\mathcal{P}$-complete for dense graphs.*

Proof. In order to prove the thesis it will be sufficient to show an $\mathcal{NC}$-reduction from the HDS problem to the same problem restricted to dense graphs.

Given any graph $G(V,E)$ with $n = |V|$ and $m = |E|$, we consider the graph $G^d(V^d,E^d)$ obtained from G by the following simple construction. We add a clique $K_n(V^n,E^n)$ of n vertices to G and we then connect each vertex of V to every vertex in K_n. This construction can be easily performed in $\mathcal{NC}$ and observe also that the resulting graph $G^d(V^d,E^d)$ is dense since $|V^d| = 2n$ and $|E^d| \geqslant m + (n(n-1))/2 = \Omega(|V^d|^2)$. It is then not hard to prove that V contains a subset H with $d(H) \geqslant h$ if and only if V^d contains a subset H^d with $d(H^d) \geqslant h + n - 1$. Indeed, to the G-subgraph induced by any subset $H \subseteq V$ we associate the G^d-subgraph induced by the subset $H^d = H \cup V^n \subseteq V^d$. □

We also observe that the above simple reduction does not preserve constant approximation ratios and, consequently, it cannot be used to extend any eventual approximation algorithm for dense graphs to general graphs.

4. The $\mathcal{NC}$-approximation scheme

Corollary 2 states that, concerning efficient parallel constructions of exact (i.e. optimal) solutions for the HDS problem, the 'dense' property of the input graph is not helpful. In this section, however, we prove that the dense property permits us to derive an $\mathcal{NC}$-approximation scheme for the HDS problem.

The algorithm consists of $n - \lceil m/n \rceil$ parallel executions of a particular 'pruning' procedure (see [1, 8]), each of them starting with a different value h ($h = 0,\ldots,n - \lceil m/n \rceil - 1$) as a parameter. Every *phase* of the hth execution of the pruning procedure removes in parallel all vertices having degree less than the parameter function $k(h) = \lceil m/n \rceil + h$ until either an empty graph is generated or a graph having a sufficiently 'large' subset of vertices with degree not less than $k(h)$ is generated. Observe first that if $d^*(G) \geqslant k(h)$ then the hth execution of the pruning procedure terminates with a non-empty graph. Furthermore, as we will formally prove later, the above choice of the function $k(h)$ and the exact definition of *sufficiently large subset* represent the key ingredients to obtain both a good quality of approximation and a polylogarithmic parallel time. A subset W of a vertex set V' is thus *sufficiently large* if contains at least $(1-\lambda)|V'|$ elements, where $\lambda = \varepsilon(m/n^2)$. Observe that, when the input graph is dense, the ratio n^2/m is bounded by a polylogarithmic function; this fact will permit us to prove that the number of phases is polylogarithmic. Then, if the hth execution of the pruning procedure terminates with a non-empty graph G^h, the algorithm computes the induced subgraph obtained from G^h by removing (again) the set S^h of those vertices having degree less than $k(h)$. The fact that $S^h \leqslant \lambda|V'|$ implies a useful lower bound on the minimum degree of the computed subgraph. Finally, the algorithm outputs the induced subgraph computed by the Mth execution of the pruning procedure, where

$$M = \max\{h = 0,\ldots,n - \lceil m/n \rceil - 1 : G^h \text{ is not empty}\}.$$

Algorithm ALG
input: a graph $G(V,E)$; a parameter ε $(0 < \varepsilon < 1)$.
begin
$\lambda := \varepsilon \frac{m}{n^2}$;
for $h := 0$ **to** $n - \lceil m/n \rceil - 1$ **do in parallel**
begin
$Z := 0$; $G_Z^h(V_Z^h, E_Z^h) := G(V,E)$;
$k(h) := \lceil \frac{m}{n} \rceil + h$; $S^h(Z) = \{v \in V_Z^h : d(v) < k(h)\}$;
while $|S^h(Z)| > \lambda |V_Z^h|$ **and** $V_Z^h - S^h(Z) \neq \emptyset$ **do**
begin
$Z := Z + 1$;
$V_Z^h := V_{Z-1}^h - S^h(Z-1)$;
Consider the subgraph G_Z^h induced by V_Z^h;
$S^h(Z) = \{v \in V_Z^h : d(v) < k(h)\}$;
end
$T := Z + 1$;
if $V_{T-1}^h - S^h(T-1) \neq \emptyset$ **then**
consider G_T^h induced by $V_T^h := V_{T-1}^h - S^h(T-1)$;
end
output: The vertex subset V_T^M (which induces the subgraph G_T^M) where
$M := \max \{h = 0, \ldots, n - \lceil m/n \rceil - 1 : G_T^h \text{ is not empty}\}$.
end.

In order to prove the correctness of ALG, we need the following combinatorial result due to Erdös.

Lemma 3 (Erdös [5]). *Given any graph $G = (V,E)$ having n vertices and m edges, a subset H of V exists such that* $\mathrm{d}(H) \geqslant \lceil m/n \rceil$.

Lemma 4. *For any graph $G(V,E)$,* ALG *always generates an induced subgraph G_T^M which is not empty.*

Proof. It is sufficient to observe that the *while* loop terminates with a non-empty induced subgraph (i.e. G_T^h) at least for $h = 0$. Indeed, in this case we have $k = \lceil m/n \rceil$ and, from Lemma 3, an induced graph exists having minimum degree at least $\lceil m/n \rceil$. □

The above lemma is not helpful in order to provide the approximation ratio achieved by ALG since, for $h = 0$, the subgraph G_T^h could be a 'bad' approximation of the optimum solutions. We thus need to prove the following further result.

Lemma 5. *For any constant ε $(0 < \varepsilon < 1)$,* ALG *generates a vertex subset V_T^M such that:*

$$\frac{d^*(G)}{(V_T^M)} \leqslant \frac{1}{1-\varepsilon}.$$

Proof. We first observe that, from the definition of the positive integer M in ALG, the input graph G cannot contain induced subgraphs having minimum degree greater than $\lceil m/n \rceil + M$. Finally, since for $h = M$ the set $S^h(T-1)$ has size at most $\lambda |V^h_{T-1}| \leqslant \lambda n$, the minimum degree of G^M_T satisfies the inequality $d(V^M_T) \geqslant \lceil m/n \rceil + M - \lambda n$. Notice that in the last inequality we have used the fact that, in the worst case, a vertex in the induced subgraph G^M_{T-1} can be adjacent to all vertices in the subset $S^h(T-1)$. From the above results, we obtain the following inequalities:

$$\frac{d^*(G)}{d(V^M_T)} \leqslant \frac{\lceil m/n \rceil + M}{\lceil m/n \rceil + M - \lambda n}$$

$$= \frac{\lceil m/n \rceil + M}{\lceil m/n \rceil + M - \varepsilon \lceil m/n \rceil} .$$

We thus have that

$$\frac{d^*(G)}{d(V^M_T)} \leqslant \frac{\lceil m/n \rceil + M}{\lceil m/n \rceil + M - \varepsilon \lceil m/n \rceil}$$

$$= \frac{\lceil m/n \rceil + M}{(1-\varepsilon)\lceil m/n \rceil + M}$$

$$= \frac{1}{1-\varepsilon} . \qquad \square$$

As the final result, we determine the computational complexity of ALG.

Lemma 6. *For any graph $G(V,E)$ such that $m = \Omega(n^2/poly \log n)$ and for any constant ε $(0 < \varepsilon < 1)$,* ALG *can be executed on a* SIMD CREW PRAM *in polylogarithmic parallel time using* $O(nm)$ *processors. In particular, for $m = \Omega(n^2/\log^q n)$ (for some fixed $q \geqslant 0$), the parallel time is $O((1/\varepsilon)\log^{q+2} n)$.*

Proof. We first observe that each *phase* generated by the *while*-loop can be performed in $O(\log n)$ time (on a CREW PRAM) using m processors. Furthermore, if the *while* condition is satisfied for some $Z > 0$ then the following inequality holds: $|V^h_Z| \leqslant |V^h_{Z-1}| - \lambda |V^h_{Z-1}|$. Consequently, we have that after t phases the current number of vertices satisfies: $|V^h_t| \leqslant (1-\lambda)^t n$.

For $m = cn^2$ (for some positive constant $c < 1$), we thus have that the global parallel time of the algorithm is

$$O\left(\frac{1}{\log(\frac{1}{1-\varepsilon})} \log^2 n\right) = O\left(\frac{1}{\varepsilon} \log^2 n\right)$$

using $O(nm)$ processors (i.e. m processors for each value assumed by the index h in the range $0, \ldots, n - \lceil m/n \rceil - 1$).

Consider now the case $m = c(n^2/\log n)$ where c is any positive constant in the interval $(0,1)$. The proof for the remaining cases $m = \Theta(n^2/\log^q n)$ $(q \geqslant 2)$ can be derived using the same arguments. We thus have that

$$|V_t^h| \leqslant (1-\lambda)^t n = \left(1 - \frac{\varepsilon c}{\log n}\right)^t n.$$

The number of phases is thus bounded by any positive integer t satisfying the following inequality:

$$\left(1 - \frac{\varepsilon c}{\log n}\right)^t \leqslant \frac{1}{n}. \tag{1}$$

Since,

$$\left(1 - \frac{\varepsilon c}{\log n}\right)^t \leqslant \exp\left(-\frac{\varepsilon c}{\log n} t\right),$$

we can state that there is a positive integer t for which Inequality (1) holds and such that $t = O((1/\varepsilon)\log^2 n)$. Finally, the global parallel time of the algorithm is thus $O((1/\varepsilon)\log^{1+2} n)$, using $O(nm)$ processors. □

The above lemmas prove the main result of this paper.

Theorem 7. *The* HDS *problem for dense graphs belongs to the class* $\mathcal{NCAS}$.

5. Concluding remarks

A first consequence of our results is that the 'threshold' behavior of the parallel approximability of the HDS problem, revealed by Anderson and Mayr, strongly depends on the density of the input graph. Our results can also be seen as another proof of the fact that the presence of *density* in the instances of a 'hard' problem often helps in deriving efficient approximation schemes as widely shown in [3] for $\mathcal{NP}$-hard problems. Concerning parallel complexity, Kirousis et al. [8] have shown that both the *Maximum Edge Connectivity Subgraph* and the *Maximum Vertex Connectivity Subgraph* problems are approximable by parallel algorithms which have performances equivalent to that of Anderson and Mayr. However, for equivalent reasons to those observed for Anderson and Mayr's algorithm, the corresponding algorithms do not give *constructive* solutions. An interesting future work thus consists in deriving efficient parallel algorithms which provide constructive solutions for such problems. We believe that the algorithm presented in this paper can give potential ideas for this future research at least in the case of dense graphs.

Finally, we observe that our $\mathcal{NC}$-approximation scheme implies also a first rough evaluation of the parallel complexity of the HDS problem in the *average-case*. Indeed, if, for any n, we introduce a uniform distribution on the space of graphs with n vertices

and we randomly select one element from this space, we then have that, with 'high' probability, the selected graph is dense thus representing a 'good' instance for our algorithm. This implies that, according to this input distribution, the problem is easy to approximate in average. A more sophisticated analysis of the average-case complexity of the HDS problem (even for *sparse random* graphs) is provided in [2].

Acknowledgements

We would like to thank Maria Serna for helpful ideas.

References

[1] R. Anderson and E.W. Mayr, A P-complete problem and approximation to it, Tech. Report STAN-CS-84-1014 (Department of Computer Science, Stanford University, 1984).

[2] A.E. Alexander, A.E.F. Clementi, P. Crescenzi, S. De Agostino, E. Dahlhaus and J.D.P. Rolim, The Parallel Complexity of Approximating the High Degree Subgraph Problem, *Proc. VI Ann. Internat. Sympo. on Algorithms and Computation (ISAAC)*, Lecture Notes in Computer Science (Springer, Berlin, 1995), to appear.

[3] S. Arora, D. Karger and M. Karpinski, Polynomial time approximation schemes for dense instances of NP-hard problems, *Proc. 27th ACM-STOC* (1995) 284–293.

[4] K. Edwards, The complexity of coloring problems on dense graphs, *Theoret. Comput. Sci.* **43** (1986) 337–343.

[5] P. Erdös, On the structure of linear graphs, *Israel J. Math.* **1** (1963) 156–160.

[6] R. Greenlaw, Ordered vertex removal and subgraph problems, *J. Comput. System Sci.* **39** (1989) 323–342.

[7] R. Greenlaw, H.J. Hoover and W.L. Ruzzo, *Limits to Parallel Computation: P-completeness Theory* (Oxford Univ. Press, Oxford, 1995).

[8] L.M. Kirousis, M.J. Serna and P. Spirakis, The parallel complexity of the connected subgraph problem, *Proc. 30th IEEE-FOCS* (1989) 446–456.

[9] E.W. Mayr, Parallel approximation algorithms, *Proc. 5th Generation Comput. Systems* (1988), 542–551.

[10] W.F. de la Vega, MAXCUT has a randomized approximation scheme for dense graphs, Unpublished Manuscript (1994).

[11] P.G. Spirakis, PRAM models and fundamental parallel algorithmic techniques, in: A. Gibbons and P.G. Spirakis, eds., *Lectures on Parallel Computation*, Ch. III (Cambridge Univ. Press, Cambridge, 1993).