On the Hardness of Optimization in Power Law Graphs*

Alessandro Ferrante[1] Gopal Pandurangan[2] Kihong Park[2]

[1] Dipartimento di Informatica ed Applicazioni "R.M. Capocelli", University of Salerno, Via Ponte don Melillo - 84084 Fisciano (SA), Italy.
E-mail: ferrante@dia.unisa.it
[2] Department of Computer Science, Purdue University, West Lafayette, IN 47907, USA. E-mail: {gopal, park}@cs.purdue.edu

Abstract. Our motivation for this work is the remarkable discovery that many large-scale real-world graphs ranging from Internet and World Wide Web to social and biological networks exhibit a power-law distribution: the number of nodes y_i of a given degree i is proportional to $i^{-\beta}$ where $\beta > 0$ is a constant that depends on the application domain. There is practical evidence that combinatorial optimization in power-law graphs is easier than in general graphs, prompting the basic theoretical question: Is combinatorial optimization in power-law graphs easy? Does the answer depend on the power-law exponent β? Our main result is the proof that many classical NP-hard graph-theoretic optimization problems remain NP-hard on power law graphs for certain values of β. In particular, we show that some classical problems, such as CLIQUE and COLORING, remains NP-hard for all $\beta \geq 1$. Moreover, we show that all the problems that satisfy the so-called "optimal substructure property" remains NP-hard for all $\beta > 0$. This includes classical problems such as MIN VERTEX-COVER, MAX INDEPENDENT-SET, and MIN DOMINATING-SET. Our proofs involve designing efficient algorithms for constructing graphs with prescribed degree sequences that are tractable with respect to various optimization problems.

1 Overview and Results

The elegant theory of NP-hardness serves as an important cornerstone in understanding the difficulty of solving various combinatorial optimization problems in graphs. A natural and relevant question is whether such hardness results on combinatorial problems are applicable to "real-world" graphs since such graphs possess certain well-defined special properties which may very well render them tractable. Our motivation for this work is the remarkable discovery that many large-scale real-world graphs ranging from Internet and World Wide Web to social and biological networks exhibit a power-law distribution. In such networks, the number of nodes y_i of a given degree i is proportional to $i^{-\beta}$ where $\beta > 0$

* Work partially supported by funds for research from MIUR ex 60% 2005.

is a constant that depends on the application domain. Power-law degree distribution has been observed in the Internet ($\beta = 2.1$), World Wide Web ($\beta = 2.1$), social networks (movie actors graph with $\beta = 2.3$, citation graph with $\beta = 3$), and biological networks (protein domains with $\beta = 1.6$, protein-protein interaction graphs with $\beta = 2.5$). In most real-world graphs, β ranges between 1 and 4 (see [3] for a comprehensive list). Thus, power-law graphs have emerged as a partial answer to the perennial search for representative real-world graphs in combinatorial optimization.

There is practical evidence that combinatorial optimization in real-world power law graphs is easier than in general graphs. For example, experiments in Internet measurement graphs (power law with $\beta = 2.1$) show that a simple greedy algorithm that exploits the power law property yields a very good approximation to the MINIMUM VERTEX COVER problem (much better than random graphs with no power law) [10, 11]. Gkantsidis, Mihail, and Saberi [7] argue that the performance of the Internet suggests that multi-commodity flow can be routed more efficiently (i.e., with near-optimal congestion) in Internet graphs than in general graphs. Eubank et al. [6] show that in power-law social networks, a simple and natural greedy algorithm that again exploits the power-law property (choose enough high-degree vertices) gives a $1+o(1)$ approximation to the DOMINATING SET problem. There is also similar practical evidence that optimization in power-law biological networks is easier [8]. All these results on disparate problems on various real-world graphs motivate a coherent and systematic algorithmic theory of optimization in power law graphs (and in general, graphs with prescribed degree sequences).

In this work, we study the following theoretical questions: What are the implications of power-law degree distribution to the algorithmic complexity of NP-hard optimization problems? Can the power-law degree distribution property alone be sufficient to design polynomial-time algorithms for NP-hard problems on power-law graphs? And does the answer depend on the exponent β?

A number of power law graph models have been proposed in the last few years to capture and/or explain the empirically observed power-law degree distribution in real-world graphs. They can be classified into two types. The first takes a power-law degree sequence and generates graph instances with this distribution. The second type arises from attempts to explain the power-law starting from basic assumptions about a growth evolution. Both approaches are well motivated and there is a large literature on both (e.g., [4, 1, 2]). Following Aiello, Chung, and Lu [1, 2], we adopt the first approach, and use the following model for (undirected) power-law graphs (henceforth called ACL model): the number of vertices y_i with degree i is roughly given[3] by $y_i = e^\alpha / i^\beta$, where e^α is a normalization constant (so that the total number of vertices sum to the size of the graph, thus α determines the size). While the above model is potentially less accurate than the detailed modeling approach of the second type, it has the advantage of being robust and general [1]: the structural properties that are true in this model will be true for all graphs with the given degree sequence.

[3] Our model is defined precisely in Section 2.

Investigating the complexity of problems in power law graphs (in particular, the ACL model) involves an important subtlety. The ACL model allows graphs with self-loops and multi-edges. However, many real-world networks, such as Internet domain graphs, are simple undirected power-law graphs. Thus, we restrict ourselves to simple undirected power-law graphs (no multi-edges or self-loops). In this paper we study the complexity of many classical graph problems in the ACL model. In particular, we first show that problems such as COLORING and CLIQUE remains NP-hard in simple power-law graphs of the the ACL model for all $\beta \geq 1$, and then we show that all the graph problems that satisfy an "optimal substructure" property (such as MINIMUM VERTEX COVER, MAXIMUM INDEPENDENT SET and MINIMUM DOMINATING SET) remain NP-hard on simple power law graphs of the ACL model for all $\beta > 0$. This property essentially states that the optimal solution for a problem on given graph is the union of the optimal (sub-)solutions on its maximal connected components. A main ingredient in our proof is a technical lemma that guarantees that any arbitrary graph can be "embedded" in a suitably large (but polynomial in the size of the given graph) graph that conforms to the prescribed power-law degree sequence. This lemma may be of independent interest and can have other applications as well e.g., in showing hardness of *approximation* of problems in power-law graphs. Another contribution is constructions of graphs with prescribed degree sequences that admit polynomial time algorithms. These constructions are useful in showing the NP-hardness of certain optimization problems that do not satisfy the optimal substructure property. In particular, we will use them to show the NP-hardness of CLIQUE and COLORING for all $\beta \geq 1$.

Our results show that the worst-case complexity of power law graphs is hard with respect to many important graph optimization problems. However, experimental evidence shows that optimization is considerably easier in real-world power-law graphs. This suggests that real-world graphs are not "worst-case" instances of power-law graphs, but rather typical instances which may be well modeled by power law *random graph* models (e.g., [1, 6, 3, 4, 7, 9]). Combinatorial optimization is generally easier in random graphs and hence from an optimization perspective this somewhat justifies using power law random graphs to model real-world power law graphs. We believe that further investigation, both in the modeling of real-world graphs and in the optimization complexity of real-world graphs and their models, is needed to gain a better understanding of this important issue.

2 Notations and Definitions

In this section we introduce some notations and definitions that we will use throughout the paper. For all $x, y \in \mathbb{N}$ with $x \leq y$, we will use $[x, y]$ to denote $\{x, x+1, \cdots, y\}$ and $[x]$ to denote $[1, x]$.

Given a graph, we will refer to two types of sequence of integers: y-degree sequence and d-degree sequence. The first type lists the number of vertices with a certain degree (i.e., the degree distribution) and the latter lists the degrees of

the vertices in non-increasing order (i.e., the degree sequence of the graph in non-increasing order). More formally, we can define the y-degree sequence as follows. Given a graph $G = (V, E)$ with maximum degree m, the y-degree sequence is the sequence $Y^G =< y_1^G, \cdots, y_m^G >$ where $y_i = |\{v \in V : degree(v) = i\}|$, $i \in [m]$. Given a graph of n vertices, the d-degree sequence will be denoted by $D^G =< d_1^G, \cdots, d_n^G >$, where d_i^G's are the vertex degrees in non-increasing order. When the referred graph is clear from the context, we will use only Y and D to denote the y- and d-degree sequence respectively. (We note that we don't allow vertices with zero degree (i.e., singletons) in G. This is not really a issue, because we will deal with problems in which singletons can be treated separately from the rest of the graph to obtain a (optimum) solution to the problem with "minor effects" on the running time.)

Given a sequence of integers $S =< s_1, \cdots, s_m >$, we define the following operator that expands S in a new non increasing sequence of integers.

Definition 1 ((Expansion)). *Let $S =< s_1, \cdots, s_n >$ be a sequence of integers and $j \in [n]$. Then we define*

$$\mathbb{EXP}(S) =< \overbrace{n, \cdots, n}^{s_n}, \cdots, \overbrace{1, \cdots, 1}^{s_1} > .$$

Note that the expansion operation converts a y-degree sequence into a $d-$sequence. In the rest of the paper, given two degree sequences $S =< s_1, \cdots, s_n >$ and $T =< t_1, \cdots, t_m >$ with $n \geq m$, we will denote $S - T =< x_1, \cdots, x_n >$ with $x_i = s_i - t_i$ if $i \in [m]$ and $x_i = s_i$ otherwise.

The ACL model of power-law graphs introduced in [1] have a particular kind of y-degree sequence which we henceforth call (β, α)-degree sequence and is defined as follows.

Definition 2 (((β, α)-degree sequence)). *Given $\alpha, \beta \in \mathbb{R}^+$, the y-degree sequence of a graph $G = (V, E)$ is a (β, α)-degree sequence (denoted by $Y^{(\beta,\alpha)} =< y_1^{(\beta,\alpha)}, \cdots, y_m^{(\beta,\alpha)} >$) if $m = \lfloor e^{\alpha/\beta} \rfloor$ and, for $i \in [m]$*

$$y_i = \begin{cases} \left\lfloor \frac{e^\alpha}{i^\beta} \right\rfloor & \text{if } i > 1 \text{ or } \sum_{k=1}^{m} \left\lfloor \frac{e^\alpha}{k^\beta} \right\rfloor \text{ is even} \\ \lfloor e^\alpha \rfloor + 1 & \text{otherwise.} \end{cases}$$

In the rest of the paper, given a sequence of integers $S =< s_1, \cdots, s_k >$, we will define $tot(S) = \sum_{i=1}^{k} s_i$ and $w(S) = \sum_{i=1}^{k} i s_i$. Note that if S is the y-degree sequence of a graph, then $w(S)$ is the total degree of the graph, whereas if S is the d-degree sequence of a graph, then $tot(S)$ is the total degree of the graph.

Our aim is to study the NP-hardness of graph-theoretic optimization problems when they are restricted to ACL power-law graphs with a fixed β, in particular, simple graphs belonging to this class. (Of course, showing hardness results for this class implies hardness for arbitrary power law graphs as well.) Formally, we define such graphs as:

Definition 3 (β-graph)). *Given $\beta \in \mathbb{R}^+$, a graph $G = (V, E)$ is a β-graph if it is simple and there exists $\alpha \in \mathbb{R}^+$ such that the y-degree sequence of G is a (β, α)-degree sequence.*

3 NP-Hardness of CLIQUE AND COLORING

In this section we introduce a general technique to prove the NP-hardness of some optimization problems. The main idea of the proof is the following. Given an arbitrary graph G, it is possible to construct a simple graph G_1 which contains G as a set of maximal connected components. Let $G_2 = G_1 \backslash G$ be the remaining graph. Obviously, G_2 is simple and if we can show that we can efficiently (i.e., in polynomial time) compute the optimal solution in G_2 then this essentially gives us the result. However, it is a priori not obvious how to design an efficient algorithm given a particular problem. The key idea we will use here is that we have the choice of constructing G_1 (and hence G_2) and thus we can construct the graph in such a way that it admits an efficient algorithm. If we construct the graph in a careful way, it will be possible to design a polynomial time algorithm that finds the optimal.

Below we illustrate this idea by showing the NP-completeness of certain problems, including CLIQUE AND COLORING, in β-graphs for $\beta \geq 1$. Our idea here is to make G_2 to be a *simple bipartite* graph. Since bipartite graphs are 2-colorable and have a maximum clique of size 2, this immediately gives the reduction. Obviously, the main difficulty is in constructing the bipartite graph. We first need the following definitions.

Definition 4 ((Contiguous Sequence)). *A sequence* $D =< d_1, \cdots, d_n >$ *with maximum value* m *is* contiguous *if* $y_i^D > 0$ *for all* $i \in [m]$*, where* $y_i^D = |\{j \in [n] \text{ s.t. } d_j = i\}|$.

Definition 5 ((Bipartite-Eligible Sequence)). *A sequence* $D =< d_1, \cdots, d_n >$ *with maximum value* m *is* bipartite-eligible *if it is contiguous and* $m \leq \lfloor n/2 \rfloor$.

Given a simple graph $G = (V, E)$, for every vertex $u \in V$ we will denote $NEIG(u) = \{v \in V \backslash \{u\} \text{ s.t. } (u, v) \in E\}$.

Lemma 1. *Let* $D =< d_1, \cdots, d_n >$ *be a sequence. If* D *is non increasing and bipartite-eligible and* $tot(D)$ *is even, then it is possible to construct in time* $O(n^2)$ *a simple bipartite graph* $G = (V, E)$ *such that* $D^G = D$.

Proof. First note that since D is non increasing and bipartite-eligible, $d_1 \leq \lfloor n/2 \rfloor$. We build the graph iteratively by adding some edges to certain vertices. Define the *residual* degree of a vertex as its final degree minus its "current" degree. Initially all the vertices have degree 0. To build the graph we use the following algorithm:

1. let $d(s_i)$ and $d(t_i)$ be the *residual degree* of the i-th vertex of S and T;
2. $E \leftarrow \emptyset$; $S \leftarrow \emptyset$; $T \leftarrow \emptyset$; $tot(S) \leftarrow 0$; $tot(S) \leftarrow 0$; $k \leftarrow |S|$; $l \leftarrow |T|$;
3. **while** $i \leq n$ **do**
   (a) **while** $i \leq n$ **and** $tot(S) \leq tot(T)$ **do**
      i. $S \leftarrow S \cup \{u \mid u \notin S\}$; $k \leftarrow k + 1$; $d(s_k) \leftarrow d_i$; $tot(S) \leftarrow tot(S) + d_i$;
   (b) **while** $i \leq n$ **and** $tot(T) \leq tot(S)$ **do**
      i. $T \leftarrow T \cup \{v \mid v \notin T\}$; $l \leftarrow l + 1$; $d(t_l) \leftarrow d_i$; $tot(T) \leftarrow tot(T) + d_i$;

4. **while** $tot(S) > 0$ **do**
   (a) **SORT** S and T separately in non increasing order of the residual degree;
   (b) **for** $i \leftarrow 1$ **to** $d(s_1)$ **do**
      i. $E \leftarrow E \cup \{(s_1, t_i)\}$; $d(s_1) \leftarrow d(s_1) - 1$; $d(t_i) \leftarrow d(t_i) - 1$; $tot(S) \leftarrow tot(S) - 1$; $tot(T) \leftarrow tot(T) - 1$;
   (c) **for** $i \leftarrow 2$ **to** $d(t_1) + 1$ **do**
      i. $E \leftarrow E \cup \{(t_1, s_i)\}$; $d(t_1) \leftarrow d(t_1) - 1$; $d(s_i) \leftarrow d(s_i) - 1$; $tot(T) \leftarrow tot(T) - 1$; $tot(S) \leftarrow tot(S) - 1$;
5. **return** $G = (S \cup T, E)$;

Note that the entire loop 3) requires $O(n^2)$ time to be completed. Moreover, in every iteration of the loop 4), at least one vertex is completed and will be no longer considered in the algorithm. Therefore, the loop 4) is completed in $O(n^2)$ time and the algorithm has complexity $O(n^2)$.

Now we prove that the algorithm correctly works. We first introduce some notations. The residual degree of the set S (T respectively) after the *SORT* instruction of the round i is denoted by $R_i(S)$ ($R_i(T)$ respectively). The number of vertices with positive residual degree (*non full* vertices) in S (T) is denoted by $N_i(S)$ ($N_i(T)$). The set S is $s_1^i, \cdots, s_h^i$ and the set T is $t_1^i, \cdots, t_k^i$.

The proof is by induction on the round i. More exactly, we prove the following invariant: *After the SORT instruction we have: (i)* $R_i(S) = R_i(T)$ *and (ii)* $N_i(T) \geq d(s_1^i)$ *and* $N_i(S) \geq d(t_1^i)$.

It is easy to see that if this invariant holds, then the algorithm correctly builds a bipartite graph. We start proving the base ($i = 1$) by showing that the above two conditions hold.

1. Let $tot_j(S)$ and $tot_j(T)$ be the total degree of the sets S and T after the insertion of the j-th vertex. We first show that $|tot_j(S) - tot_j(T)| \leq d_{j+1}$ for all $j \in [2, n-1]$. This is obvious for $j = 2$ since the sequence is non increasing and contiguous. Let us suppose that this is true until $j - 1$ and let us show it for j.
   Without loss of generality, let us suppose that the j-th vertex is assigned to T. Then this implies that $tot_{j-1}(S) \geq tot_{j-1}(T)$ and by induction $tot_{j-1}(S) - tot_{j-1}(T) \leq d_j$ and, therefore, $tot_j(S) - tot_j(T) \leq 0$.
   Now we can complete the proof of the bases. w.l.o.g. let us suppose that the last one vertex is assigned to T. Then we have $R_1(S) \geq R_1(T) - 1$. But from the preceding proof we also know that $R_1(S) \leq R_1(T)$ and, from the fact that the last one vertex has degree 1 and that the total degree of D is even, we have the claim.
2. Since the degree sequence is contiguous and after the insertion we have $tot(S) = tot(T)$, it is easy to see that after the insertion we have $-1 \leq |S| - |T| \leq 1$. From this and from the hypothesis $d_1 \leq \lfloor n/2 \rfloor$ the claim follows.

Let us suppose that the invariant is true until $i - 1$ and let us prove it for i.

1. We have $R_i(S) = R_{i-1}(S) - d(s_1^{i-1}) - (d(t_1^{i-1}) - 1) = R_{i-1}(T) - d(t_1^{i-1}) - (d(s_1^{i-1}) - 1) = R_i(T)$ as claimed.

2. The case $d(s_1^i) = 0$ is trivial, therefore let us suppose that $d(s_1^i) \geq 1$. If $d(s_2^{i-1}) = 1$, then $d(s_1^i) = 1$ since the degrees in S are non increasing. Moreover, from item (1) we have $R_i(T) = R_i(S) \geq 1$ and this completes this case.
   If $d(s_2^{i-1}) > 1$, then we have two cases. If $d(t_2^{i-1}) = 1$, from item (1) and the fact that $d(t_j^i) \leq 1$ for all j we simply have the claim. On the other hand, if $d(s_2^{i-1}) > 1$, we have $N_i(T) = N_{i-1}(T) \geq d(s_1^{i-1}) \geq d(s_1^i)$. □

The following lemma shows that for $\beta \geq 1$ it is possible to embed a simple graph G in a polynomial-size β-graph G_1 such that G is a set of maximal connected components of G_1 and $G_2 = G_1 \backslash G$ is bipartite-eligible.

Lemma 2. *Let $G = (V, E)$ be a simple graph with n_1 vertices and $\beta \geq 1$. Let $\alpha_0 = \max\{4\beta, \beta \ln n_1 + \ln(n_1 + 1)\}$. Then, for all $\alpha \geq \alpha_0$ the sequence $D = \mathbb{EXP}(Y^{(\beta,\alpha)} - Y^G)$ is contiguous and bipartite-eligible.*

Proof. Let n_2 be the number of elements in D and $\alpha \geq \alpha_0$. We have

$$n_2 \geq \sum_{i=1}^{\lfloor e^{\alpha/\beta} \rfloor} \left\lfloor \frac{e^\alpha}{i^\beta} \right\rfloor - n_1 > e^\alpha \sum_{i=1}^{\lfloor e^{\alpha/\beta} \rfloor} \frac{1}{i^\beta} - \lfloor e^{\alpha/\beta} \rfloor - n_1 \geq e^\alpha \int_{i=1}^{\lfloor e^{\alpha/\beta} \rfloor + 1} \frac{1}{i^\beta} - \lfloor e^{\alpha/\beta} \rfloor - n_1.$$

If $\beta = 1$, then we have $n_2 \geq \alpha e^\alpha - e^\alpha - n_1 \geq 4e^\alpha - 2e^\alpha + 1 \geq 2m + 1$. If $\beta > 1$ we have $n_2 \geq \frac{e^\alpha}{\beta - 1} - e^{\alpha/\beta} - n_1 \geq 4e^{\alpha/\beta} - 2e^{\alpha/\beta} + 1 \geq 2m + 1$. Moreover, $y_{n_1}^{(\beta,\alpha)} \geq \left\lfloor \frac{e^\alpha}{n_1^\beta} \right\rfloor > \frac{e^\alpha}{n_1^\beta} - 1 \geq \frac{n_1^{\beta+1} + n_1^\beta}{n_1^\beta} - 1 = n_1$, that is $\mathbb{EXP}(Y)$ is contiguous. Therefore, $\mathbb{EXP}(Y)$ is bipartite-eligible and this completes the proof of this lemma. □

We finally show the NP-completeness of certain problems in β-graphs with $\beta \geq 1$. The following definition is useful to introduce the class of problems we analyze in what follows.

Definition 6 (c-Oracle). *Let P be an optimization problem and $c > 0$ a constant. A c-oracle for the problem P is a polynomial-time algorithm $A_c^P(I)$ which takes in input an instance I of P and correctly returns an optimum solution for P given that on the instance I the problem has an optimum solution with size at most c.*

The following theorem shows the NP-completeness of a particular class of decision problems defined using the c-oracle in β-graphs with $\beta \geq 1$.

Theorem 1. *Let $\beta \geq 1$. Let P be a graph decision problem such that its optimization version obeys the following properties:*

1. *$OPT(G) = \max_{1 \leq i \leq k} OPT(C_i)$ (where C_i are the maximal connected components of G),*
2. *exists a constant $c > 0$ such that for all bipartite simple graphs H it holds $|OPT(H)| \leq c$ and*

3. it admits a c-oracle.

If P is NP-complete in general graphs, then it is NP-complete in β-graphs too.

Proof. From Lemmas 2 and 1, it is possible to construct, in time $poly(|G|)$, a β-graph G_1 embedding G such that $|G_1| = poly(|G|)$, G is a set of maximal connected components of G_1 and $G_2 = G_1 \backslash G$ is a simple bipartite graph. Since $OPT(G_1) = \max_k\{OPT(C_k)\}$, $|OPT(G_2)| \leq c$ and the optimization version of P admits a c-oracle, it is easy to see that P can be reduced in polynomial time to β-P (where β-P is P restricted to β-graphs). □

Since CLIQUE and COLORING satisfy all conditions of Theorem 1 with $c = 2$, we easily obtain the following corollary.

Corollary 1. *CLIQUE, and COLORING are NP-Complete in β-graphs for all $\beta \geq 1$.*

4 Hardness of Optimization Problems with Optimal Substructure

We show that if an optimization problem is NP-hard on (simple) general graphs (i.e., computing a solution in polynomial time is hard) and it satisfies the following "optimal substructure" property, then it is NP-hard on β-graphs also. We state this property as follows. Let P be an optimization problem which takes a graph as input. For *every* input G, the following should be true: every optimum solution of P on G should contain an optimum solution of P on each of G's maximal connected components. To illustrate with an example, it is easy to see that MINIMUM VERTEX COVER problem satisfies this property: an optimal vertex cover on any graph G should contain within it an optimal vertex cover of its maximal connected components. On the other hand, MINIMUM COLORING does not satisfy the above property, since the optimal coloring of a graph need not contain an optimal coloring of all its maximal connected components. We first need some definitions. We say that a sequence D is *graphic* if there exists a simple graph G such that $D^G = D$.

Definition 7 ((Eligible Sequence)). *A sequence of integers $S =< s_1, \cdots, s_n >$ is eligible if $s_1 \geq \cdots \geq s_n$ and, for all $k \in [n]$, $f_S(k) \geq 0$, where*

$$f_S(k) = k(k-1) + \sum_{i=k+1}^{n} \min\{k, s_i\} - \sum_{i=1}^{k} s_i.$$

The following result due to Havel and Hakimi ([5]) gives a straightforward algorithm to construct a simple graph from a graphic degree sequence.

Lemma 3 ([5]). *A sequence of integers $D =< d_1, \cdots, d_n >$ is graphic if and only if it is non-increasing, and the sequence of values $D' =< d_2 - 1, d_3 - 1, \cdots, d_{d_1+1} - 1, d_{d_1+2}, \cdots, d_n >$ when sorted in non-increasing order is graphic.*

In the next technical lemma, which proof is missing because of lack of space, we introduce a new sufficient condition for a sequence of integers to be eligible.

Lemma 4. *Let $Y^{(1)}$ and $Y^{(2)}$ be two degree sequences with m_1 and m_2 elements respectively such that (i) $y_j^{(1)} \leq y_j^{(2)}$ for all $j \in [m_1]$, and (ii) $D^{(1)} = \mathbb{EXP}(Y^{(1)})$ and $D^{(2)} = \mathbb{EXP}(Y^{(2)})$ are contiguous. If $D^{(1)}$ is eligible then $D^{(2)}$ is eligible.*

The previous lemma is useful to show the following key lemma (*Embedding Lemma*) that shows that it is possible to quickly construct a β-graph with a certain property.

Lemma 5 ((Embedding Lemma)). *Let $G = (V, E)$ be a simple undirected graph and $\beta \in \mathbb{R}^+$. Then there exists a simple undirected graph $G_1 = (V_1, E_1)$ such that G is a set of maximal connected components of G_1, $|V_1| = poly(|V|)$ and G_1 is a β-graph. Furthermore, given G, we can construct G_1 in time polynomial in the size of G.*

Proof. Let $n_1 = |V|$. From Lemma 3, we have only to show that there exist $\alpha_0 = O(\ln n_1)$ such that for all $\alpha \geq \alpha_0$, the degree sequence $D = \mathbb{EXP}(Y) = Y^{(\beta,\alpha)} - Y^G$ is graphic, that is, from Lemma 3 such that D is eligible. For $\beta \geq 1$ the proof directly comes from Lemmas 2 and 1. Let us complete the proof for $0 < \beta < 1$.

Note that, $y_i^{(1,\alpha)} \leq y_i^{(\beta,\alpha)}$ and $\lfloor e^{\alpha/\beta} \rfloor \geq \lfloor e^{\alpha} \rfloor$ for $0 < \beta < 1$ and $i \in \lfloor e^{\alpha} \rfloor$ and, from Lemma 2, $\mathbb{EXP}(Y^{(1,\alpha)} - Y^G)$ is contiguous for $\alpha \geq \max\{4, \ln n_1 + \ln(n_1+1)\}$.

Therefore, from Lemma 4, the sequence $\mathbb{EXP}(Y^{(\beta,\alpha)} - Y^G)$ is eligible for $0 < \beta < 1$ and $\alpha \geq \max\{4, \ln n_1 + \ln(n_1 + 1)\}$ and this completes the proof of this lemma. □

Now we are ready to show the main theorem of this section.

Theorem 2. *Let P be an optimization problem on graphs with the optimal substructure property. If P is NP-hard on (simple) general graphs, then it is also NP-hard on β-graphs for all $\beta > 0$.*

Proof. We show that we can reduce the problem of computing an optimal solution on general graphs to computing an optimal solution on β-graphs and this reduction takes polynomial time. Let $G = (V, E)$ be a simple undirected graph. Lemma 5 says that we can construct (in time polynomial in the size of G) a simple undirected graph $G_1 = (V_1, E_1)$ such that G is a set of maximal connected components of G_1, and G_1 is a β-graph with $|V_1| = poly(|V|)$. Since P has the optimal substructure property and G is a set of maximal connected components of G_1, this implies that an optimum solution for the graph G can be computed easily from an optimal solution for G_1. □

5 Concluding Remarks and Open Problems

We showed a general technique for establishing NP-hardness and NP-completeness of a large class of problems in power-law graphs. Our technique of "embedding"

any arbitrary (given) graph into a polynomial-sized power-law graph is quite general and can have other applications, e.g., in showing hardness of *approximation* in power-law graphs (which is the next important question, now that we have established hardness). On the positive side, one may investigate approximation algorithms that exploit the power law property to get better approximation ratios compared to general graphs. Another interesting and relevant direction is to investigate the hardness or easiness of non-trivial restrictions of the ACL model.

We conclude by mentioning some open problems that follow directly from our work. We showed NP-hardness of CLIQUE and COLORING only for power law graphs with $\beta \geq 1$. We believe that a different construction might show that these problems are NP-Complete for all $\beta > 0$. It will also be interesting to investigate the complexity of node- and edge-deletion problems, that is a general and important class of problems defined in [12]. We finally note that our technique does not directly imply hardness in *connected* power-law graphs. We conjecture that our techniques can be extended to show these results.

References

1. W. Aiello, F.R.K. Chung, L. Lu, "A Random Graph Model for Massive Graphs", in *Proceedings of STOC 2000*, 171-180, ACM 2000.
2. W. Aiello, F.R.K. Chung, L. Lu, "A random graph model for power law graphs", in *Experimental Mathematics*, 10, 53-66, 2000.
3. A. Barabasi, "Emergence of Scaling in Complex Networks", in *Handbook of Graphs and Networks*, S. Bornholdt and H. Schuster (Ed.), Wiley 2003.
4. B. Bollobas, O. Riordan, "Mathematical Results on Scale-free Random Graphs", in *Handbook of Graphs and Networks*, S. Bornholdt and H. Schuster (Ed.), 2003.
5. J.A. Bondy, U.S.R. Murty, "Graph Theory with Applications", North Holland 1976.
6. S. Eubank, V.S.A. Kumar, M.V. Marathe, A. Srinivasan, N. Wang, "Structural and Algorithmic Aspects of Massive Social Networks", in *Proceedings of 15th ACM-SIAM Symposium on Discrete Algorithms (SODA 2004)*, 711-720, SIAM 2004.
7. C. Gkantsidis, M. Mihail, A. Saberi, "Throughput and Congestion in Power-Law Graphs", in *Proceedings of SIGMETRICS 2003*, 148-159, ACM 2003.
8. M. Koyuturk, A. Grama, W. Szpankowski, "Assessing significance of connectivity and conservation in protein interaction networks", in *Proceedings of RECOMB 2006*, LLNCS 3909, 45-49, Springer 2006.
9. M. Mihail, C. Papadimitriou, A. Saberi, "On Certain Connectivity Properties of the Internet Topology", in *Proc. of FOCS 2003*, 28-35, IEEE Comp. Soc. 2003.
10. K. Park, H. Lee, "On the effectiveness of route-based packet filtering for distributed DoS attack prevention in power-law internets", in *Proceedings of SIGCOMM 2001*, 15-26, ACM 2001.
11. K. Park, "The Internet as a complex system", in *The Internet as a Large-Scale Complex System*, K. Park and W. Willinger (Editors), Santa Fe Institute Studies on the Sciences of Complexity, Oxford University Press 2005.
12. M. Yannakakis, "Node- and Edge-Deletion NP-Complete Problems", in *Proceedings of STOC 1978*, San Diego, California (USA), 253-264, SIAM 1978.