

On the hardness of optimization in power-law graphs

Alessandro Ferrante^a, Gopal Pandurangan^{b,*}, Kihong Park^b

^a *Dipartimento di Informatica ed Applicazioni “R.M. Capocelli”, Università degli Studi di Salerno, Via Ponte don Melillo, 84084, Fisciano (SA), Italy*

^b *Department of Computer Science, Purdue University, West Lafayette, IN 47907, USA*

Received 24 January 2007; received in revised form 14 September 2007; accepted 8 December 2007

Communicated by O.H. Ibarra

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 appear to 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, remain NP-hard for all $\beta \geq 1$. Moreover, we show that all the problems that satisfy the so-called “optimal substructure property” remain NP-hard for all $\beta > 0$. This includes classical problems such as MINIMUM VERTEX COVER, MAXIMUM INDEPENDENT SET, and MINIMUM DOMINATING SET. Our proofs involve designing efficient algorithms for constructing graphs with prescribed degree sequences that are tractable with respect to various optimization problems.

© 2007 Elsevier B.V. All rights reserved.

Keywords: Power-law graphs; Graph optimization problems; NP-hardness; Graph construction algorithms

1. Overview and results

In this paper, we focus on studying the hardness of combinatorial optimization in power-law graphs. 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 appear to exhibit a power-law distribution [3]. In power-law networks, 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. Power-law degree distribution has been observed in the Internet ($\beta = 2.1$), World Wide Web

* Corresponding author. Tel.: +1 765 494 0916; fax: +1 765 494 0739.

E-mail addresses: ferrante@dia.unisa.it (A. Ferrante), gopal@cs.purdue.edu (G. Pandurangan), park@cs.purdue.edu (K. Park).

URLs: <http://www.dia.unisa.it/dottorandi/ferrante> (A. Ferrante), <http://www.cs.purdue.edu/people/gopal> (G. Pandurangan), <http://www.cs.purdue.edu/people/park> (K. Park).

($\beta = 2.1$), 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).

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) [13,14]. In particular, for Internet graphs (AS topologies), they show that the sizes of the vertex covers obtained by the greedy algorithm are very close to the optimal, with an accuracy level greater than 90%. On the other hand, for random graphs the corresponding accuracy level is less than 75%. Gkantsidis, Mihail, and Saberi [9] 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. More formally they show that power-law random graphs can support routing of $O(d_u d_v)$ units of flow between each pair of vertices u and v with degrees d_u and d_v respectively, with congestion $O(n \log^2 n)$. This congestion is optimal to within a logarithmic factor (achieved by linear sized regular graphs with constant expansion). Eubank et al. [7] 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. 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 power-law 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 the (β, α) model): the number of vertices y_i with degree i is roughly given¹ 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). We note that unlike the model of Aiello, Chung, and Lu [1,2], the (β, α) model is *not* a random graph model.

Investigating the complexity of problems in power-law graphs (in particular, the (β, α) model) involves an important subtlety. The (β, α) 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 (β, α) model. We first show that problems such as COLORING and CLIQUE remain NP-hard in simple power-law graphs of the (β, α) model for all $\beta \geq 1$. We then 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 (β, α) model for all $\beta > 0$. This property essentially states that any optimal solution for a problem on a 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 (cf. Section 5). 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 $\beta \geq 1$. We should point out that our results do not directly imply hardness of *connected* power-law graphs. This is because the (simple) power-law graphs that we construct in our proofs are

¹ Our model is defined precisely in Section 2.

disconnected. To have more relevance to real-world power-law graphs (see e.g., [10]), it will be interesting to extend our techniques to show hardness of simple and connected power-law graphs (cf. Section 5).

Our results show that, many important graph optimization problems remain NP-hard even for power-law graphs. 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., [1,7,3,4,9,11,12]). 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.

Organization: In Section 2 we will introduce some notations and definitions that will be used throughout the paper. In Section 3 we first introduce a general technique to prove the NP-completeness of decision problems in power-law graphs and then use this technique to show the NP-completeness of CLIQUE and COLORING. In Section 4 we will show that optimization problems obeying the optimal substructure property that are NP-hard in general graphs are NP-hard in power-law graphs too. We conclude with some open questions in Section 5.

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 $\{x, x + 1, \dots, y\}$ and $[x]$ to denote $[1, x]$.

Given a graph, we will refer to two types of sequences of integers: y -degree sequences and d -degree sequences. 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 $Y^G = \langle y_1^G, \dots, y_m^G \rangle$ where $y_i = |\{v \in V : \text{degree}(v) = i\}|$, $i \in [m]$. Given a graph of n vertices, the d -degree sequence will be denoted by $D^G = \langle d_1^G, \dots, d_n^G \rangle$, 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 sequences respectively. (We note that we do not allow vertices with zero degree (i.e., isolated nodes) in G . This is not really a issue, because we will deal with problems in which isolated nodes 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 $S = \langle s_1, \dots, s_m \rangle$, we define the following operator that expands S into a new non-increasing sequence of integers.

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

$$\text{EX}(S) = \langle \overbrace{n, \dots, n}^{s_n}, \dots, \overbrace{1, \dots, 1}^{s_1} \rangle.$$

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

The (β, α) model of power-law graphs uses the following y -degree sequences which we henceforth call (β, α) -degree sequences 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 (and is denoted by $Y^{(\beta, \alpha)} = \langle y_1^{(\beta, \alpha)}, \dots, y_m^{(\beta, \alpha)} \rangle$) 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 $S = \langle s_1, \dots, s_k \rangle$, we will define $\text{tot}(S) = \sum_{i=1}^k s_i$ 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 $\text{tot}(S)$ 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 (β, α) model 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 $G_2 = G_1 \setminus G$ be the remaining graph. Obviously, G_2 is simple and if we can show that we can efficiently (i.e., in polynomial time) compute an 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 an optimal solution.

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 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 Sequences*). A sequence $D = \langle d_1, \dots, d_n \rangle$ with maximum value m is *contiguous* if $y_i^D > 0$ for all $i \in [m]$, where $y_i^D = |\{j \in [n] \text{ s.t. } d_j = i\}|$. Intuitively, D is contiguous if each value $i \in [m]$ appears at least once in D .

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

Lemma 1. *Let $D = \langle d_1, \dots, d_n \rangle$ be a sequence. If D is non-increasing and bipartite-eligible and $\text{tot}(D)$ 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 (let S and T be two initially empty sets of vertices):

- (1) let $rd(s_i)$ and $rd(t_i)$ be the *residual degree* of the i th vertex of S and T ;
- (2) $E \leftarrow \emptyset$; $S \leftarrow \emptyset$; $T \leftarrow \emptyset$; $\text{tot}(S) \leftarrow 0$; $\text{tot}(T) \leftarrow 0$; $k \leftarrow |S|$; $l \leftarrow |T|$;
- (3) **while** $i \leq n$ **do**
 - (a) **while** $i \leq n$ **and** $\text{tot}(S) \leq \text{tot}(T)$ **do**
 - (i) let u be a new vertex;
 - (ii) $S \leftarrow S \cup \{u\}$; $k \leftarrow k + 1$; $rd(s_k) \leftarrow d_i$; $\text{tot}(S) \leftarrow \text{tot}(S) + d_i$;
 - (b) **while** $i \leq n$ **and** $\text{tot}(T) \leq \text{tot}(S)$ **do**
 - (i) let v be a new vertex;
 - (ii) $T \leftarrow T \cup \{v\}$; $l \leftarrow l + 1$; $rd(t_l) \leftarrow d_i$; $\text{tot}(T) \leftarrow \text{tot}(T) + d_i$;
- (4) **while** $\text{tot}(S) > 0$ **do**
 - (a) **SORT** S and T separately in non-increasing order of the residual degree;
 - (b) **for** $i \leftarrow 1$ **to** $rd(s_1)$ **do**
 - (i) $E \leftarrow E \cup \{(s_1, t_i)\}$; $rd(s_1) \leftarrow rd(s_1) - 1$; $rd(t_i) \leftarrow rd(t_i) - 1$; $\text{tot}(S) \leftarrow \text{tot}(S) - 1$; $\text{tot}(T) \leftarrow \text{tot}(T) - 1$;
 - (c) **for** $i \leftarrow 2$ **to** $rd(t_1) + 1$ **do**
 - (i) $E \leftarrow E \cup \{(t_1, s_i)\}$;
 - (ii) $rd(t_1) \leftarrow rd(t_1) - 1$; $rd(s_i) \leftarrow rd(s_i) - 1$; $\text{tot}(T) \leftarrow \text{tot}(T) - 1$; $\text{tot}(S) \leftarrow \text{tot}(S) - 1$;
- (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 works correctly. 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, \dots, s_h^i and the set T is t_1^i, \dots, 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:*

- (1) $R_i(S) = R_i(T)$ and
- (2) $N_i(T) \geq rd(s_1^i)$ and $N_i(S) \geq rd(t_1^i)$.

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

- (1) Let $tot_j(S)$ and $tot_j(T)$ be the total degrees 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 base case. Without loss of generality let us suppose that the last 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 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 $R_i(S) = R_{i-1}(S) - rd(s_1^{i-1}) - (rd(t_1^{i-1}) - 1) = R_{i-1}(T) - rd(t_1^{i-1}) - (rd(s_1^{i-1}) - 1) = R_i(T)$ as claimed.

- (2) The case $rd(s_1^i) = 0$ is trivial, therefore let us suppose that $rd(s_1^i) \geq 1$. If $rd(s_2^{i-1}) = 1$, then $rd(s_1^i) = 1$ 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 $rd(s_2^{i-1}) > 1$, then we have two cases. If $rd(t_2^{i-1}) = 1$, from item (1) and the fact that $rd(t_j^i) \leq 1$ for all j the claim follows. On the other hand, if $rd(s_2^{i-1}) > 1$, we have $N_i(T) = N_{i-1}(T) \geq rd(s_1^{i-1}) \geq rd(s_1^i)$. \square

The following lemma shows that for $\beta \geq 1$ it is possible to embed a simple graph G in a polynomial-sized β -graph G_1 such that G is a set of maximal connected components of G_1 and $G_2 = G_1 \setminus G$ 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 $D = \mathbb{E}\mathbb{X}(Y^{(\beta, \alpha)} - Y^G)$ is contiguous and bipartite-eligible.*

Proof. Let n_2 be the number of elements in D , $\alpha \geq \alpha_0$ and $m = \lfloor e^{\alpha/\beta} \rfloor$ be the maximum value in D . 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

$$n_2 \geq \alpha e^\alpha - e^\alpha - n_1 \geq 4e^\alpha - 2e^\alpha + 1 \geq 2m + 1$$

and if $\beta > 1$ we have

$$n_2 \geq \frac{e^\alpha}{\beta - 1} - e^{\alpha/\beta} - n_1 \geq 4e^{\alpha/\beta} - 2e^{\alpha/\beta} + 1 \geq 2m + 1,$$

that is $m \leq n_2/2 - 1 \leq \lfloor n_2/2 \rfloor$. Moreover, as

$$y_{n_1}^{(\beta, \alpha)} \geq \left\lfloor \frac{e^\alpha}{n_1^\beta} \right\rfloor > \frac{e^\alpha}{n_1^\beta} - 1 \geq \frac{n_1^{\beta+1} + n_1^\beta}{n_1^\beta} - 1 = n_1,$$

so $\mathbb{E}\mathbb{X}(Y)$ is contiguous. Therefore, $\mathbb{E}\mathbb{X}(Y)$ is bipartite-eligible and this completes the proof of this lemma. \square

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 as 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) there 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 for general graphs, then it is NP-complete for β -graphs too.

Proof. From Lemmas 1 and 2, 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 $G_2 = G_1 \setminus G$ 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). \square

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. Other important problems which satisfy this property include MINIMUM DOMINATING SET, MAXIMUM CUBIC SUBGRAPH, MAXIMUM INDEPENDENT SET [8]. 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. Similarly the MAXIMUM CLIQUE problem does not satisfy the property. 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 Sequences*). A sequence of integers $S = \langle s_1, \dots, s_n \rangle$ is eligible if $s_1 \geq \dots \geq s_n$ and, for all $k \in [n]$, $f_S(k) \geq 0$, where

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

Erdős and Gallai showed a key result [6,5] which gives a necessary and sufficient condition for a sequence of integers to be graphic.

Lemma 3 ([6,5]). *A sequence of integers D is graphic if and only if it is non-increasing, $\text{tot}(D)$ is even and D is eligible.*

Note that this condition is different from the somewhat simpler condition that $\max_i d_i^2 < \sum_i d_i$ and $\sum_i d_i$ is even which is also due to Erdős and Gallai [6]. This condition only ensures that the d_i 's can be realized as a multi-graph (without self-loops), but not as a simple graph. The following result due to Havel and Hakimi [5] gives a straightforward algorithm to construct a simple graph from a graphic degree sequence.

Lemma 4 ([5]). *A sequence of integers $D = \langle d_1, \dots, d_n \rangle$ is graphic if and only if it is non-increasing, and the sequence of values $D' = \langle d_2 - 1, d_3 - 1, \dots, d_{d_1+1} - 1, d_{d_1+2}, \dots, d_n \rangle$ when sorted in non-increasing order is graphic.*

In the next technical lemma, we introduce a new sufficient condition for a sequence of integers to be eligible.

Lemma 5. *Let $Y^{(1)}$ and $Y^{(2)}$ be two y -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) $D^{(1)} = \mathbb{E}\mathbb{X}(Y^{(1)})$ and $D^{(2)} = \mathbb{E}\mathbb{X}(Y^{(2)})$ are contiguous. If $D^{(1)}$ is eligible then $D^{(2)}$ is eligible.*

Proof. Let us note that the transformation from the degree sequence $Y^{(1)}$ to the degree sequence $Y^{(2)}$ (and hence from $D^{(1)}$ to $D^{(2)}$) can be seen as a sequence of rounds of the following type: in every step a vertex with degree d is transformed into a vertex with degree $(d + 1)$ and the global sequence is rearranged with respect to the relation $y_j^{(1)} \leq y_j^{(2)}$ for all $j \in [m_1]$. In other words, to transform $Y^{(1)}$ to $Y^{(2)}$ (and hence $D^{(1)}$ to $D^{(2)}$) we can execute the following simple algorithm (for a better readability, in the rest of the paper, given a sequence $S = \langle s_1, \dots, s_n \rangle$ and an integer x we will use the notation $S + x = \langle s_1, \dots, s_n, x \rangle$ to denote the concatenation of S with the integer x):

- (1) $S^{(0)} \leftarrow D^{(1)}$;
- (2) $i \leftarrow 0$;
- (3) **while** $S^{(i)} \neq D^{(2)}$ **do**
 - (a) **for** $j \leftarrow m_2$ **downto** 2 **do**
 - (i) **if** $|\{x \in S^{(i+1)} \text{ s.t. } x = j\}| < y_j^{(2)}$ **and** $|\{x \in S^{(i+1)} \text{ s.t. } x = j - 1\}| > 0$ **then**
 - (A) $k \leftarrow \min\{x \in S^{(i+1)} \text{ s.t. } s_x^{(i+1)} = j - 1\}$;
 - (B) $s_k^{(i+1)} \leftarrow s_k^{(i+1)} + 1$;
 - (b) $S^{(i+1)} \leftarrow S^{(i+1)} + x$;
 - (c) $i \leftarrow i + 1$;

Let $n_2 = |D^{(2)}|$. From the definition of eligibility, $D^{(2)}$ is eligible if $f_{D^{(2)}}(k) > 0$ for all $k \in [n_2]$. Since $f_{S^{(0)}}(k) \geq 0$ for $k \in n^{(0)}$, to show this we only have to prove that at the end of each iteration of the *while* loop of the previous algorithms, if $f_{S^{(i-1)}}(k) \geq 0$ for all $i \in [n^{(i-1)}]$ then $f_{S^{(i)}}(k) \geq 0$ for all $i \in [n^{(i)}]$, where $n^{(i)} = |S^{(i)}|$.

Let $k \in n^{(i)}$ and $r \in [2, m_2]$ be the maximum integer such that the *if* of the algorithm is executed. Notice that in each iteration of the *while* loop of the algorithm, the *if* clause is executed for all $j \in [2, r]$ and is not executed for $j \in [r + 1, m_2]$. Let $m^{(i)}$, $i \geq 0$, be the maximum value in the sequence $S^{(i)}$. Now, we consider the following cases.

- **Case A** ($k \leq s_k^{(i-1)}$ **and** $r < k$): Notice that $r \leq s_k^{(i-1)}$. Therefore, since the *if* clause will be executed only for $j \in [2, r]$ and $S^{(i-1)}$ is non-increasing, then it is obvious that $s_t^{(i)} = s_t^{(i-1)}$ for $t \in [k]$. Therefore the following formula holds.

$$\sum_{j=1}^k s_j^{(i)} = \sum_{j=1}^k s_j^{(i-1)}.$$

Moreover, from the algorithm it is obvious that $n^{(i)} > n^{(i-1)}$ and $s_j^{(i)} \geq s_j^{(i-1)}$ for $j \in n^{(i-1)}$. Therefore we have the following result.

$$\sum_{j=k+1}^{n^{(i)}} \min\{k, s_j^{(i)}\} > \sum_{j=k+1}^{n^{(i-1)}} \min\{k, s_j^{(i-1)}\}.$$

From the previous two formulas we easily have that

$$\begin{aligned} f_{S^{(i)}}(k) &= k(k-1) + \sum_{j=k+1}^{n^{(i)}} \min\{k, s_j^{(i)}\} - \sum_{j=1}^k s_j^{(i)} \\ &> k(k-1) + \sum_{j=k+1}^{n^{(i-1)}} \min\{k, s_j^{(i-1)}\} - \sum_{j=1}^k s_j^{(i-1)} = f_{S^{(i-1)}}(k) \geq 0 \end{aligned}$$

which completes the proof of this case.

- **Case B** ($k \leq s_k^{(i-1)}$ and $r \geq k$): From the algorithm we can see that $s_j^{(i)} \leq s_j^{(i-1)} + 1$ for all $j \in [k]$. Therefore we have that

$$\sum_{j=1}^k s_j^{(i)} \leq \sum_{j=1}^k s_j^{(i-1)} + k.$$

From the algorithm, it is easy to show that $S^{(i-1)}$ and $S^{(i)}$ are contiguous. To see this, first note that $S^{(0)} = D^{(1)}$ is contiguous. Moreover, it can be easily showed that if the **for** loop is executed for a given $l \in [m_2]$, then it is executed for all $l' \in [l]$. By considering that the **for** loop, when executed for a given l increases by one the number of elements equal to l and, if $l > 1$ decreases by one the number of elements equal to $l - 1$, we can easily obtain that $S^{(i-1)}$ and $S^{(i)}$ are contiguous.

Therefore, there is at least one element $t(j) \in n^{(i-1)}$ such that $s_{t(j)}^{(i)} = s_{t(j)}^{(i-1)} + 1$ for all $j \in [k - 1]$. Moreover, from the algorithm it is obvious that $n^{(i)} = n^{(i-1)} + 1$. Therefore, we have that the following holds.

$$\sum_{j=k+1}^{n^{(i)}} \min\{k, s_j^{(i)}\} \geq \sum_{j=k+1}^{n^{(i-1)}} \min\{k, s_j^{(i-1)}\} + k.$$

From the previous two relations we easily have that:

$$\begin{aligned} f_{S^{(i)}}(k) &= k(k-1) + \sum_{j=k+1}^{n^{(i)}} \min\{k, s_j^{(i)}\} - \sum_{j=1}^k s_j^{(i)} \\ &\geq k(k-1) + \sum_{j=k+1}^{n^{(i-1)}} \min\{k, s_j^{(i-1)}\} + k - \sum_{j=1}^k s_j^{(i-1)} - k \\ &= f_{S^{(i-1)}}(k) \geq 0 \end{aligned}$$

which completes the proof for this case.

- **Case C** ($s_k^{(i-1)} < k = s_k^{(i)}$): Note that, since $k = s_k^{(i)}$ then $k \leq m^{(i)}$. Let $k = s_k^{(i)} = m^{(i)} - t$ where $t \geq 0$. Since $S^{(i)}$ is contiguous and non-increasing, we have that

$$\begin{aligned} \sum_{j=1}^k s_j^{(i)} &= s_k^{(i)} + s_{k-1}^{(i)} + \dots + s_{k-t+1}^{(i)} + s_{k-t}^{(i)} + \dots + s_1^{(i)} \\ &= (m^{(i)} - t) + (m^{(i)} - t + 1) + \dots + (m^{(i)} - 1) + m^{(i)} + \dots + m^{(i)} \\ &= \sum_{j=m^{(i)}-t}^{m^{(i)}-1} j + m^{(i)}(k - t) \end{aligned}$$

$$\begin{aligned}
 &= \frac{(m^{(i)} - 1)m^{(i)}}{2} - \frac{(m^{(i)} - t)(m^{(i)} - t + 1)}{2} + m^{(i)}(m^{(i)} - 2t) \\
 &= (m^{(i)})^2 - tm^{(i)} - m^{(i)} - \frac{t^2}{2} + \frac{t}{2}.
 \end{aligned}$$

Since $S^{(i)}$ is non-increasing, it obviously holds that $k = s_k^{(i)} \geq s_j^{(i)}$ for all $j \in [k + 1, n^{(i)}]$; moreover, since $S^{(i)}$ is contiguous then in the set $[k + 1, n^{(i)}]$ there is at least one element l_j with value $s_{l_j}^{(i)} = j$ for all $j \in [s_{k+1}^{(i)}]$. Thus we have (recall that $s_{k+1}^{(i)} \geq s_k^{(i)} - 1$)

$$\begin{aligned}
 \sum_{j=k+1}^{n^{(i)}} \min\{k, s_j^{(i)}\} &= \sum_{j=k+1}^{n^{(i)}} s_j^{(i)} \geq \sum_{j=1}^{s_k^{(i)}-1} j = \frac{(s_k^{(i)} - 1)s_k^{(i)}}{2} \\
 &= \frac{(m^{(i)} - t - 1)(m^{(i)} - t)}{2} \\
 &= \frac{(m^{(i)})^2 - 2tm^{(i)} - m^{(i)} + t^2 + t}{2} \\
 &= \frac{(m^{(i)})^2}{2} - tm^{(i)} - \frac{m^{(i)}}{2} + \frac{t^2}{2} + \frac{t}{2}.
 \end{aligned}$$

Moreover we have:

$$k(k - 1) = (m^{(i)} - t)(m^{(i)} - t - 1) = (m^{(i)})^2 - 2tm^{(i)} + t^2 - m^{(i)} + t.$$

From the previous formulas we have

$$\begin{aligned}
 f_{S^{(i)}}(k) &= k(k - 1) + \sum_{j=k+1}^{n^{(i)}} \min\{k, s_j^{(i)}\} - \sum_{j=1}^k s_j^{(i)} \\
 &\geq (m^{(i)})^2 - 2tm^{(i)} + t^2 - m^{(i)} + t + \frac{(m^{(i)})^2}{2} - tm^{(i)} - \frac{m^{(i)}}{2} \\
 &\quad + \frac{t^2}{2} + \frac{t}{2} - (m^{(i)})^2 + tm^{(i)} + m^{(i)} + \frac{t^2}{2} - \frac{t}{2} \\
 &= \frac{(m^{(i)})^2}{2} - \frac{m^{(i)}}{2} + 2t^2 - t(2m^{(i)} - 1).
 \end{aligned}$$

It is easy to see that the function

$$g(t) = 2t^2 - t(2m^{(i)} - 1)$$

has minimum value in $t^* = (2m^{(i)} - 1)/4$ and its value is $g(t^*) = 0$ therefore

$$f_{S^{(i)}}(k) \geq \frac{(m^{(i)})^2}{2} - \frac{m^{(i)}}{2} \geq 0$$

which completes the proof of this case.

- **Case D** ($k > s_k^{(i)}$): Since, from case C, $f_{S^{(i)}}(s_k^{(i)}) \geq 0$, to prove this case it is enough to show that for all $k > s_k^{(i)}$, $f_{S^{(i)}}(k) \geq f_{S^{(i)}}(k - 1)$ which is true since

$$\begin{aligned}
 f_{S^{(i)}}(k) &= k(k - 1) + \sum_{j=k+1}^{n^{(i)}} \min\{k, s_j^{(i)}\} - \sum_{j=1}^k s_j^{(i)} \\
 &= (k - 1)(k - 2) + 2(k - 1) + \sum_{j=k}^{n^{(i)}} \min\{k, s_j^{(i)}\} - \min\{k, s_k^{(i)}\} - \sum_{j=1}^{k-1} s_j^{(i)} - s_k^{(i)} \\
 &= f_{S^{(i)}}(k - 1) + 2(k - s_k^{(i)} - 1) \geq f_{S^{(i)}}(k - 1)
 \end{aligned}$$

since $k > s_k^{(i)}$, and this completes the proof of the lemma. \square

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

Lemma 6 (*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 , $|V_1| = \text{poly}(|V|)$ and G_1 is a β -graph. Furthermore, given G , we can construct G_1 in polynomial time in the size of G .*

Proof. Let $n_1 = |V|$. From Lemma 4, we only have to show that there exists $\alpha_0 = O(\ln n_1)$ such that for all $\alpha \geq \alpha_0$, the degree sequence $D = \mathbb{E}\mathbb{X}(Y) = Y^{(\beta, \alpha)} - Y^G$ is graphic, that is, from Lemma 4 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, $\mathbb{E}\mathbb{X}(Y^{(1, \alpha)} - Y^G)$ is contiguous for $\alpha \geq \max\{4, \ln n_1 + \ln(n_1 + 1)\}$.

Therefore, from Lemma 5, the sequence $\mathbb{E}\mathbb{X}(Y^{(\beta, \alpha)} - Y^G)$ 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. \square

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 6 says that we can construct (in polynomial time 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 $|V_1| = \text{poly}(|V|)$. 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 . \square

5. Concluding remarks and open problems

We have shown 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* of NP-hard problems in power-law graphs (which is the next important question, now that we have established hardness). For example, a concrete question is: Can MAXIMUM INDEPENDENT SET be approximated better in β -graphs than in general graphs? Using the Embedding Lemma and our general methodology (cf. Section 3), one can derive bounds on hardness of approximation on β -graphs based on known hardness bounds on general graphs. It is not difficult to verify that to get non-trivial hardness bounds one needs good bounds on the optimum solution of G_2 . As it stands now, our Embedding Lemma construction does not directly yield a good bound on the optimum solution of G_2 .

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 (β, α) model. In particular, we note that our technique does not directly imply hardness in *connected* power-law graphs. Establishing hardness in connected and simple graphs in our model (i.e., connected β -graphs) is an important and relevant question. We conjecture that our techniques can be extended to show these results.

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. This is a general and important class of problems defined in [15].

Acknowledgments

We are grateful to the referees for their careful reading of the paper and detailed comments which helped in improving the presentation of the paper.

References

- [1] W. Aiello, F.R.K. Chung, L. Lu, A random graph model for massive graphs, in: Proceedings of 32nd Annual Symposium on Theory of Computing, STOC 2000, ACM, 2000, pp. 171–180.
- [2] W. Aiello, F.R.K. Chung, L. Lu, A random graph model for power-law graphs, *Experimental Mathematics* 10 (2000) 53–66.
- [3] A. Barabasi, Emergence of scaling in complex networks, in: S. Bornholdt, H. Schuster (Eds.), *Handbook of Graphs and Networks*, Wiley, 2003.
- [4] B. Bollobás, O. Riordan, Mathematical results on scale-free random graphs, in: S. Bornholdt, H. Schuster (Eds.), *Handbook of Graphs and Networks*, Wiley, 2003.
- [5] J.A. Bondy, U.S.R. Murty, *Graph Theory with Applications*, North Holland, 1976.
- [6] P. Erdős, T. Gallai, Graphs with prescribed degree of vertices, *Matematikai Lapok* 11 (1960) 264–274 (in Hungarian).
- [7] 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, SIAM, 2004, pp. 711–720.
- [8] M. Garey, D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman, 1975.
- [9] C. Gkantsidis, M. Mihail, A. Saberi, Throughput and congestion in power-law graphs, in: Proceedings of the International Conference on Measurements and Modeling of Computer Systems, SIGMETRICS 2003, ACM, 2003, pp. 148–159.
- [10] L. Li, D. Alderson, J. Doyle, W. Willinger, Towards a theory of scale-free graphs: Definition, properties, and implications, *Internet Mathematics* 2 (4) (2005) 431–523.
- [11] M. Mihail, C. Papadimitriou, A. Saberi, On certain connectivity properties of the internet topology, in: Proceedings of the 44th Symposium on Foundations of Computer Science, FOCS 2003, IEEE Computer Society, 2003, pp. 28–35.
- [12] M. Newman, Random graphs as models of networks, in: S. Bornholdt, H. Schuster (Eds.), *Handbook of Graphs and Networks*, Wiley, 2003.
- [13] K. Park, H. Lee, On the effectiveness of route-based packet filtering for distributed DoS attack prevention in power-law internets, in: Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, SIGCOMM 2001, ACM, 2001, pp. 15–26.
- [14] K. Park, The Internet as a complex system, in: K. Park, W. Willinger (Eds.), *The Internet as a Large-Scale Complex System*, Santa Fe Institute Studies on the Sciences of Complexity, Oxford University Press, 2005.
- [15] M. Yannakakis, Node- and edge-deletion NP-complete problems, in: Proceedings of Tenth Annual SIAM Symposium on Theory of Computing, STOC 1978, SIAM, San Diego, CA, USA, 1978, pp. 253–264.