

Enumerating and Generating Labeled k -degenerate Graphs

Reinhard Bauer

Marcus Krug

Dorothea Wagner

Faculty of Informatics,
Karlsruhe Institute of Technology (KIT), Germany
{reinhard.bauer, marcus.krug, dorothea.wagner}@kit.edu

Abstract

A k -degenerate graph is a graph in which every induced subgraph has a vertex with degree at most k . The class of k -degenerate graphs is interesting from a theoretical point of view and it plays an interesting role in the theory of fixed parameter tractability since some otherwise $W[2]$ -hard domination problems become fixed-parameter tractable for k -degenerate graphs.

It is a well-known fact that the k -degenerate graphs are exactly the graphs whose vertex-set can be *well-ordered* such that each vertex is incident to at most k larger vertices with respect to this ordering. A *well-ordered k -degenerate graph* is a *labeled graph* with vertex-labels $1, \dots, n$ such that the ordering of the vertices by their labels is a well-ordering of the graph.

We consider the problem of enumerating and generating *well-ordered k -degenerate graphs* with a given number of vertices and with a given number of vertices and edges, respectively, uniformly at random. By generating well-ordered k -degenerate graphs we generate at least one labeled copy of each *unlabeled k -degenerate graph* and we filter some but not all isomorphies compared to the classical labeled approach.

We also introduce the class of strongly k -degenerate graphs, which are k -degenerate graphs with minimum degree k . These graphs are a natural generalization of k -regular graphs which can be used in order to generate graphs with predefined core-decomposition.

We present efficient algorithms for generating well-ordered k -degenerate graphs with given number of vertices (and edges). After a precomputation which must only be performed once when generating more than one well-ordered k -degenerate graph these algorithms are almost optimal. Additionally, we present complete non-uniform generators for these classes with optimal running time. We also present an efficient and complete generator for well-ordered strongly k -degenerate graphs with given number of vertices (and edges). Finally, we present efficient algorithms for enumerating well-ordered k -degenerate and strongly k -degenerate graphs.

1 Introduction

A k -degenerate graph is a graph in which every induced subgraph has a vertex with degree at most k . The most prominent class of degenerate graphs is the class of planar graphs: Since every induced subgraph of a planar graph is itself planar and since every planar graph contains a vertex of degree at most 5 every planar graph is 5-degenerate. Many other well-known classes of graphs are degenerate, including graphs with bounded genus, bounded maximum degree, bounded tree-width as well as H -minor-free graphs.

The class of k -degenerate graphs plays an interesting role in the theory of fixed parameter tractability since some otherwise $W[2]$ -hard domination problems become fixed-parameter tractable for k -degenerate graphs [1, 10]. Cai et. al use the method of random separation to obtain fixed-parameter tractable algorithms for the problem of finding induced cycles and trees in degenerate graphs [5]. Degenerate graphs have also been studied from a theoretical point of view [6, 14, 4].

Often, it is desired that isomorphic graphs are merged into one class and graph classes are considered instead of single graphs. This is called *unlabeled graph generation*. *Labeled graph generation* on the other hand, assumes that the vertices of the graph are integers $1, \dots, n$ and different graphs count as separate entities even though they might be isomorphic.

It is a well-known fact that the k -degenerate graphs are exactly the graphs whose vertex-set v_1, \dots, v_n can be *well-ordered* such that the degree of v_i in the graph induced by the vertices v_i, \dots, v_n is at most k for each i . These vertex-sequences are also called *Erdős-Hajnal sequences*. A *labeled k -degenerate graph* with vertex labels $1, \dots, n$ is called *well-ordered*, if each vertex with label i is incident to at most k vertices with label greater than i . Given a vertex-set and -order, we consider the problem of enumerating and generating such well-ordered k -degenerate graphs uniformly at random. By generating well-ordered k -degenerate graphs we generate at least one labeled copy of each *unlabeled k -degenerate graph* and we filter some but not all isomorphies compared to the classical labeled approach. Therefore this

Table 1: Contribution

		well-ordered k -degenerate	well-ordered strongly k -degenerate
uniform generation	n	$O(n \log k + m) + O(nk)$	open
	(n, m)	$O(n \log k + m) + O(nmk)$	open
non-uniform generation	n	$O(n + m)$	$O(n^2 k)$
	(n, m)	$O(n + m)$	$O(nmk)$
enumeration	n	$O(nm + m^2)$	$O(nm + m^2)$
	(n, m)	$O(nm + m^2)$	$O(n^{3/2} m^2)$

method of generation might be preferable to the classical labeled approach.

Sampling labeled planar graphs—a subset of the 5-degenerate graphs—uniformly at random has been studied extensively, e.g. [8, 3]. To the best of our knowledge, however, there have been no previous results for the general problem of enumerating and generating labeled degenerate graphs uniformly at random.

We investigate the problem of sampling from the set of well-ordered k -degenerate graphs and we will propose fast algorithms which are almost optimal after a precomputation under the real-RAM machine model. Since the precomputation must only be performed once for the uniform samplers the precomputation can be amortized when generating a large set of graphs. In addition to the uniform samplers we also propose practical complete non-uniform samplers with optimal running time. These generators are preferable to the uniform generators when generating only few very large graphs.

We then introduce the notion of strongly k -degenerate graphs which are a natural generalization of k -regular graphs. These graphs can be used to design generators for graphs with given core decomposition [2]. We present fast and complete algorithms for this class of graphs and show how to efficiently enumerate well-ordered strongly k -degenerate graphs with given number of vertices (and edges). The problem of generating regular graphs has been studied, e.g., in [12, 15, 11]. A survey of the results obtained for random regular graphs can be found in [16].

Content Section 2 deals with ordinary well-ordered k -degenerate graphs. We present uniform as well as fast complete non-uniform generators for well-ordered k -degenerate graphs with a given number of vertices (and edges). Section 3 deals with well-ordered strongly k -degenerate graphs. In this section we present fast and complete non-uniform generator for the class of well-ordered strongly k -degenerate graphs with n vertices (and m edges). In Section 4 we present efficient algorithms for enumerating all well-ordered k -degenerate and strongly k -degenerate graphs with n vertices (and m edges) in amortized polynomial time per generated graph. An overview of our contribution can be found in Table 1.

Preliminaries We will only consider loopless graphs without multiple edges. Throughout this paper let $G = (V, E)$ be an undirected labeled graph with edge-set E and totally ordered vertex-set $V := \{v_1, \dots, v_n\}$ such that $v_i < v_j$ if and only if $i < j$ and vertex v_i is assigned label i . For a vertex v_i we define the *successor-degree* $d^+(v_i) := |\{v_j \in V : \{v_i, v_j\} \in E, i < j\}|$ as well as the *predecessor-degree* $d^-(v_i) := |\{v_j \in V : \{v_i, v_j\} \in E, j < i\}|$. In the forthcoming, we will crucially exploit the simple observation that a graph with totally ordered vertex-set is a well-ordered k -degenerate graph if and only if the maximum successor-degree $d_{\max}^+(v_i)$ of vertex v_i is at most $\min\{n - i, k\}$. The degree of a vertex v_i in G is denoted by $d_G(v_i)$. If the context is clear we will omit the subscript G . The *neighborhood* $N(v_i)$ of a vertex v_i is the set of vertices v_j such that there is an edge $\{v_i, v_j\}$.

With some abuse of notation we will denote the set of all s -element subsets of a set X by $\binom{X}{s}$. The set of edges of G will be denoted by $E(G)$. Let X be a finite set and let $f : X \rightarrow \mathbb{R}$. Choosing $x \in X$ *proportional to f* means choosing x with probability $f(x) / \sum_{z \in X} f(z)$. The term $G[u_1, \dots, u_k]$ refers to the subgraph of G induced by the vertices u_1, \dots, u_k .

We denote the class of well-ordered k -degenerate graphs with n vertices by $\mathcal{D}_n^{(k)}$ and its cardinality by $D_n^{(k)}$. Accordingly, we denote the class of well-ordered k -degenerate graphs with n vertices and m edges, by $\mathcal{D}_{n,m}^{(k)}$ and its cardinality by $D_{n,m}^{(k)}$. We will use the distributions listed in Table 2.

2 Generating Well-Ordered k -Degenerate Graphs Uniformly

In this section we investigate the problem of generating well-ordered k -degenerate graphs with n vertices or n vertices and m edges uniformly at random. We also present fast complete uniform and non-uniform generators, respectively.

Every k -degenerate graph with n vertices can be decomposed into a k -degenerate graph with $n - 1$ vertices and an isolated vertex with at most k edges. The following recursive formulae for the cardinalities $D_{n,m}^{(k)}$ of $\mathcal{D}_{n,m}^{(k)}$ and $D_n^{(k)}$ of $\mathcal{D}_n^{(k)}$ derive from independently considering v_1 and $G[v_2, \dots, v_n]$:

$$(2.1) \quad D_n^{(k)} = \sum_{i=0}^{\min\{n-1, k\}} \binom{n-1}{i} D_{n-1}^{(k)}.$$

Table 2: Distributions

Distribution	Notation	Domain	Density
k -restricted Binomial	$\text{Binom}_{\leq k}(n)$	$x \in \{0, \dots, k\}$	$\binom{n}{x} \cdot \left(\sum_{i=0}^k \binom{n}{i}\right)^{-1}$
Uniform	$\text{Uniform}(X)$	$x \in X$	$1/ X $
Normal	$\text{Normal}(\mu, \sigma^2)$	$x \in \mathbb{R}$	$\frac{1}{\sigma\sqrt{2\pi}} \cdot \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$

$$(2.2) \quad D_{n,m}^{(k)} = \sum_{i=0}^{\min\{n-1, m, k\}} \binom{n-1}{i} D_{n-1, m-i}^{(k)}$$

The uniform samplers presented in the following sections crucially rely on the fact that these recursive formulae can be computed efficiently.

2.1 A Uniform Generator for $\mathcal{D}_n^{(k)}$ Our approach is a recursive algorithm that expects an index i as well as the number of vertices n of the graph as parameters. It works as follows: If $i = n$ it returns a graph consisting of an isolated vertex v_n . Otherwise, it randomly chooses the successor-degree d_i of vertex v_i according to a $\min\{n-i, k\}$ -restricted binomial distribution (see Table 2) and recursively calls itself generating a well-ordered k -degenerate graph G' with $n-i$ vertices labeled v_{i+1}, \dots, v_n . Then it chooses d_i distinct vertices uniformly at random, creates a new vertex labeled i and inserts new edges from v_i to all d_i selected vertices. The resulting graph G is returned (see Algorithm 1).

```

1 if  $i = n$  then
2   return  $(\{v_n\}, \emptyset)$ 
3  $d_i \leftarrow \text{Binom}_{\leq \min\{n-i, k\}}(n-i);$   $X_i$ 
4  $(V', E') \leftarrow \text{DEGENERATE}(i+1, n, k);$   $Y_{i+1}$ 
5  $S \leftarrow \text{Uniform}\left(\binom{V'}{d_i}\right);$   $Z_i$ 
6  $E \leftarrow E' \cup \{\{v_i, s\} \mid s \in S\}$ 
7  $G \leftarrow (V \cup \{v_i\}, E)$ 
8 return  $G;$   $Y_i$ 

```

Algorithm 1: DEGENERATE(i, n, k)

Next, we will show that Algorithm 1 samples well-ordered k -degenerate graphs with n vertices uniformly at random if invoked with DEGENERATE($1, n, k$). Let Y_1 denote the random variable representing the outcome of the call DEGENERATE($1, n, k$).

THEOREM 2.1. *Algorithm 1 generates all graphs in $\mathcal{D}_n^{(k)}$ with equal probability, i.e. $Y_1 \sim \text{Uniform}(\mathcal{D}_n^{(k)})$, if invoked with DEGENERATE($1, n, k$).*

Proof. Let X_i, Y_{i+1}, Z_i denote the random variables representing the outcome of the randomized operations in lines 3, 4 and 5 of the algorithm, respectively. We prove that $Y_i \sim \text{Uniform}(\mathcal{D}_{n-i+1}^{(k)})$ by induction on the parameter i of the algorithm. Since there is only one well-ordered k -degenerate

graph with one vertex we have $D_1^{(k)} = 1$ and, thus, the claim holds for $i = n$. Assume that $i < n$. Let $G = (\{v_i, \dots, v_n\}, E)$ be a well-ordered k -degenerate graph and let d_i denote the degree of v_i . Further, let $N(v_i) \subseteq \{v_{i+1}, \dots, v_n\}$ denote the neighbors of v_i . According to the definition of the algorithm the probability for G with input i and n is given by

$$(2.3) \quad \mathbb{P}(Y_i = G) = \mathbb{P}(X_i = d_i) \cdot \mathbb{P}(Y_{i+1} = G - v_i) \cdot \mathbb{P}(Z_i = N(v_i) \mid X_i = d_i).$$

Using Equation (2.1) and the induction hypothesis $Y_{i+1} \sim \text{Uniform}(\mathcal{D}_{n-i}^{(k)})$ we obtain

$$(2.4) \quad \mathbb{P}(Y_i = G) = \frac{\binom{n-i}{d_i}}{\sum_{j=0}^{\min\{k, n-i\}} \binom{n-i}{j}} \cdot \frac{1}{D_{n-i}^{(k)}} \cdot \frac{1}{\binom{n-i}{d_i}} \\ = \frac{1}{\sum_{j=0}^{\min\{k, n-i\}} \binom{n-i}{j} D_{n-i}^{(k)}} = \frac{1}{D_{n-i+1}^{(k)}}.$$

The algorithm described above can be implemented in two phases, a precomputation phase in which we compute the cumulative distribution function of the k -restricted binomial distribution and the actual generating phase. The precomputation can be done in $O(nk)$ time and space by precomputing the values $\sum_{j=0}^x \binom{n-i}{j} / \sum_{j=0}^k \binom{n-i}{j}$ for $i = 1, \dots, n$ and $x = 0, \dots, k$. Using binary search on the values for a given i step 3 of the algorithm can be performed in time $O(\log k)$. Since we insert at most m edges the generation can be implemented to run in time $O(n \log k + m)$ time.

We gain a faster proceeding by approximating the probabilities in Phase 2. We assume that we can obtain random values in $[0, 1]$ in constant time. To generate a random variate that is approximately k -restricted binomial distributed, we use the *inversion method* [7]. To that end, we approximate the cumulative distribution function by

$$(2.5) \quad F(x) = \frac{\sum_{i=0}^x \varphi_{\mu, \sigma^2}(i)}{\Phi_{\mu, \sigma^2}(k)} \approx \frac{\Phi_{\mu, \sigma^2}(x)}{\Phi_{\mu, \sigma^2}(k)}$$

using $\mu := n/2$ and $\sigma^2 = n/4$ as well as the cumulative distribution function Φ_{μ, σ^2} of the normal distribution. The inverse of this function can be expressed as $F^{-1}(p) = \Phi_{\mu, \sigma^2}^{-1}(p \cdot \Phi_{\mu, \sigma^2}(k))$. Instead of evaluating F^{-1} exactly, we

evaluate only a constant number of terms of the series expansions of both Φ_{μ,σ^2} and Φ_{μ,σ^2}^{-1} in constant time. The resulting generator has a running time in $O(nk + n)$ which is in $O(n + m)$.

2.2 A Uniform Generator for $\mathcal{D}_{n,m}^{(k)}$ In this section we present a uniform generator for well-ordered k -degenerate graphs with n vertices and m edges. Throughout this section we assume $m \leq nk - \binom{k+1}{2}$ since this is the maximum number of edges of any k -degenerate graph with n vertices. The algorithm is similar to the algorithm presented in the previous section. At first we choose $0 \leq d_i \leq \min\{n-1, m, k\}$ proportional to $\binom{n-1}{d_i} D_{n-1, m-d_i}^{(k)}$. Then we recursively choose G' uniformly at random from $\mathcal{D}_{n-1, m-d_i}^{(k)}$. Next we choose d_i vertices uniformly at random from G' and finally we create a new vertex which is connected to all selected vertices in G' . The pseudo-code is listed in Algorithm 2. Again we let X_i, Y_{i+1}, Z_i denote the random variables for the outcome of the randomized operations in lines 3, 4 and 5 of the algorithm, respectively.

```

1 if  $m = 0$  then
2   return  $(\{v_1, \dots, v_n\}, \emptyset)$ 
3  $d_i \leftarrow$  choose  $0 \leq d_i \leq \min\{n-i, m, k\}$  proportional to
    $\binom{n-i}{d_i} D_{n-i, m-d_i}^{(k)}$ ;  $X_i$ 
4  $(V, E') \leftarrow$  DEGENERATE( $i+1, n-i, m-d_i, k$ );  $Y_{i+1}$ 
5  $S \leftarrow$  Uniform( $\binom{V}{d_i}$ );  $Z_i$ 
6  $E \leftarrow E' \cup \{(v_i, s) \mid s \in S\}$ 
7  $G \leftarrow (V \cup \{v_i\}, E)$ 
8 return  $G$ ;  $Y_i$ 

```

Algorithm 2: DEGENERATE(i, n, m, k)

THEOREM 2.2. *Algorithm 2 generates all graphs in $\mathcal{D}_{n,m}^{(k)}$ with equal probability, i.e. $Y_1 \sim \text{Uniform}(\mathcal{D}_{n,m}^{(k)})$, if invoked with DEGENERATE($1, n, m, k$).*

Proof. The proof is analogous to the proof of Theorem 2.1. By induction we show that $Y_i \sim \text{Uniform}(\mathcal{D}_{n-i+1, m}^{(k)})$. Note that there is only one well-ordered k -degenerate graph with t vertices and no edges in our model of generation. Hence, the induction hypothesis holds analogous to the proof of Theorem 2.1. Using Equation 2.2 the probability for the graph G obtained by the algorithm with input i, n, m, k is given by

(2.6)

$$\begin{aligned} \mathbb{P}(Y_i = G) &= \frac{1}{\binom{n-1}{d_i}} \cdot \frac{1}{D_{n-i, m-d_i}^{(k)}} \cdot \frac{\binom{n-i}{d_i} D_{n-i, m-d_i}^{(k)}}{\sum_{j=0}^{\min\{n-i, m, k\}} \binom{n-i}{j} D_{n-i, m-j}^{(k)}} \\ &= \frac{1}{D_{n-i+1, m}^{(k)}}. \end{aligned}$$

Again the algorithm can be implemented in two phases as follows: In the precomputation phase, we compute the values

$$(2.7) \quad \frac{\sum_{d_i=0}^j \binom{n-i}{d_i} D_{n-i, m-d_i}^{(k)}}{\sum_{d_i=0}^{\min\{n-i, m', k\}} \binom{n-i}{d_i} D_{n-i, m'-d_i}^{(k)}}$$

of the cumulative distribution function used in step 3 of the algorithm for all $1 \leq i \leq n$ and $0 \leq j \leq \min\{n-i, m, k\}$ and $0 \leq m' \leq m$ using $O(nmk)$ time and space. In line 3 of the algorithm we choose at most k edges. Using binary search on the precomputed values, we can evaluate the inverse cumulative distribution function in time $O(\log k)$. The total number of vertex insertions is n and the total number of vertex selections and edge insertions is $O(m)$. Thus, we obtain a running time of $O(n \log k + m)$ if we have precomputed the cumulative distribution as described above and can access these numbers in constant time.

Therefore, the algorithm can be implemented to run in time $O(n \log k + m)$ after a precomputation of $O(nmk)$. Note that the precomputation can be used to generate any graph in $\mathcal{D}_{n', m'}^{(k)}$ such that $n' \leq n, m' \leq m$ and $0 \leq k' \leq k-1$. That is, if the generator is used to compute a benchmark with graphs of specified sizes, the precomputation must only be done once with a dominant combination of the parameters n, m and k .

2.3 A Fast and Complete Generator for $\mathcal{D}_{n,m}^{(k)}$ In this section we propose an efficient complete non-uniform generator for the class of well-ordered k -degenerate graphs with n vertices and m edges. The algorithm first chooses a successor-degree-sequence by iteratively choosing vertices uniformly at random among the vertices v_i which have been picked at most $\min\{n-i, k\} - 1$ times. This can be done in $O(m)$ time. Then it iterates over the vertices starting at the vertex with index $n-1$ and chooses d_i target vertices with higher index where d_i denotes the successor-degree of vertex v_i . The pseudo-code can be found in Algorithm 3.

```

1  $V \leftarrow \{v_1, \dots, v_n\}$ 
2  $E \leftarrow \emptyset$ 
3  $d_i \leftarrow 0$  for all  $1 \leq i \leq n$ 
4  $C \leftarrow \{v_1, \dots, v_{n-1}\}$ 
5 for  $i = 1$  to  $m$  do
6    $v_i \leftarrow$  Uniform( $C$ )
7    $d_i \leftarrow d_i + 1$ 
8   if  $d_i = \min\{n-i, k\}$  then
9      $C \leftarrow C \setminus \{v_i\}$ 
10 for  $i = n-1$  to  $1$  do
11    $X \leftarrow$  Uniform( $\binom{\{v_{i+1}, \dots, v_n\}}{d_i}$ )
12    $E \leftarrow E \cup \{(v_i, x) \mid x \in X\}$ 
13 return  $G := (V, E)$ 

```

Algorithm 3: DEGENERATE(n, m, k)

The loop in line 5 can be implemented to run in time $O(m)$ and the loop in line 10 can be implemented to run in time $O(n + m)$ using a suitable data-structure. Hence, the algorithm can be implemented to run in this time.

It is clear, that the algorithm is complete. However, the algorithm does not generate well-ordered k -degenerate graphs according to a uniform distribution: In order to see this, we consider the sequence of successor-degrees induced by the well-ordering of the vertices of a k -degenerate graph. Any 2-degenerate graph with successor-degree sequence 1, 1, 0 is generated with probability $3/8$ and the only 2-degenerate graph with successor-degree sequence 2, 0, 0 is generated with probability $1/4$.

3 Generating Well-Ordered Strongly k -Degenerate Graphs

In this section we consider the problem of generating well-ordered strongly k -degenerate graphs, i.e., well-ordered k -degenerate graphs with minimum degree k . We denote the class of well-ordered strongly k -degenerate graphs with n vertices by $\mathcal{S}_n^{(k)}$ and the class of well-ordered strongly k -degenerate graphs with n vertices and m edges by $\mathcal{S}_{n,m}^{(k)}$, respectively. The generators presented in the previous sections heavily relied on the fact that for any well-ordered k -degenerate graph the subgraphs $G_i := G[v_i, \dots, v_n]$ are well-ordered k -degenerate. Unfortunately, this is not true for well-ordered strongly k -degenerate graphs.

We propose the following approach: First we generate a random well-ordered k -degenerate graph with n vertices (and m edges). Then we transform this graph into a well-ordered strongly k -degenerate graph. We have seen how ordinary well-ordered k -degenerate graphs with n vertices (and m edges) can be generated in the previous sections. Next, we describe how an ordinary well-ordered k -degenerate graph can be transformed into a well-ordered strongly k -degenerate graph in $O(kn^2)$ time. We define the *deficiency* of a well-ordered k -degenerate graph $G = (V, E)$ as

$$(3.8) \quad \Delta(G) := \sum_{v \in V} \max\{k - d(v), 0\}$$

Clearly, a well-ordered k -degenerate graph G is a well-ordered strongly k -degenerate graph if and only if $\Delta(G) = 0$. In order to transform an ordinary well-ordered k -degenerate graph into a well-ordered strongly k -degenerate graph we use the following lemma.

LEMMA 3.1. *Let $G = (V, E)$ be a well-ordered k -degenerate graph with n vertices, $m \geq \frac{nk}{2}$ edges and deficiency $\Delta(G) > 0$. Then there is an edge $\{v, x\} \in E$ and a vertex u such that $G' := (G - \{v, x\}) + \{u, x\}$ is well-ordered k -degenerate and $\Delta(G') < \Delta(G)$.*

Proof. Since $\Delta(G) > 0$ there must be at least one vertex u such that $d(u) < k$. On the other hand, there must be a vertex

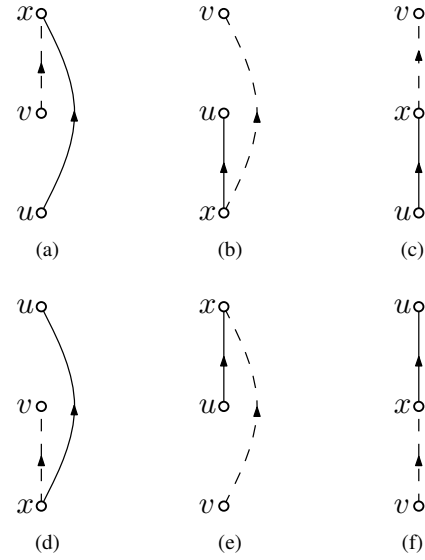


Figure 1: The possible orderings of u, v, x in Lemma 3.1: dashed edges are replaced by solid edges.

v such that $d(v) > k$. Otherwise

$$(3.9) \quad m = \frac{1}{2} \sum_{v \in V} d(v) \leq \frac{1}{2} \left(\sum_{v \in V \setminus \{u\}} d(v) + d(u) \right) < \frac{(n-1)k + k}{2} = \frac{nk}{2}$$

in a contradiction to the assumption that $m \geq \frac{nk}{2}$.

Let u be an arbitrary vertex such that $d(u) < k$ and let v be the largest vertex such that $d(v) > k$. Since $d(v) > k$ and $d(u) < k$ there is a vertex $x \in N(v) \setminus N(u)$. Hence $G' := (G - \{v, x\}) + \{u, x\}$ is simple and

$$(3.10) \quad \begin{aligned} \Delta(G') &= \sum_{v \in V' \setminus \{u, v\}} \max\{k - d_G(v), 0\} \\ &\quad + \max\{k - (d_G(u) + 1), 0\} \\ &= \Delta(G) - 1 \end{aligned}$$

Thus we need to show that G' is k -degenerate. Recall that a well-ordered graph is k -degenerate if and only if all vertices v satisfy $d^+(v) \leq k$. Note that this property can only be violated by edge-insertions (but not by edge-deletions), i.e., if the successor-degree sequence is increased for some vertex. Further $d(u) < k$ implies $d^+(u) < k$. There are six possible orderings of the vertices u, v, x . We will consider three cases:

- (i) $x < u, v$: The successor-degree sequence is not changed by the operation (Fig. 1b, 1d).
- (ii) $u < x$: The successor-degree is increased only for u . Since $d_G^+(u) < k$ we have $d_{G'}^+(u) \leq k$ (Fig. 1a, 1c, 1e).

- (iii) $v < x < u$: If $d_G^+(x) < k$ then $d_{G'}^+(x) \leq k$ (Fig. 1f). Otherwise $d^+(x) = k$. Since we have assumed that v is the largest vertex with degree greater than k we have $d(x) \leq k$, i.e., $d(x) = k$. However, this implies $d^-(x) = 0$ in contradiction to the assumption that v is a neighbor of x and $v < x$.

■

In order to obtain a unique transformation we may assume (without loss of generality) that x and u are the smallest (respectively largest) vertices with the desired properties.

We can use the previous Lemma to obtain efficient and complete generators for well-ordered strongly k -degenerate graphs in $\mathcal{S}_n^{(k)}$ and $\mathcal{S}_{n,m}^{(k)}$, respectively. The pseudo-code is listed in Algorithm 4. The Algorithm generates a random well-ordered k -degenerate graph and repeatedly applies Lemma 3.1 in order to transform the well-ordered k -degenerate graph into a well-ordered strongly k -degenerate graph.

```

1  $G \leftarrow \text{GENERATE-DEGENERATE}(1, n[, m])$ 
2 while  $\Delta(G) > 0$  do
3    $u \leftarrow$  smallest vertex with  $d(u) < k$ 
4    $v \leftarrow$  largest vertex with  $d(v) > k$ 
5    $x \leftarrow$  smallest vertex in  $N(v) \setminus N(u)$ 
6    $G \leftarrow (G - \{v, x\}) + \{u, x\}$ 
7 return  $G$ 

```

Algorithm 4: STRONGLY-DEGENERATE($n[, m]$)

THEOREM 3.1. *Algorithm 4 generates each well-ordered strongly k -degenerate graph on n vertices with probability at least $1/D_n$ in time $O(kn^2)$ and each well-ordered strongly k -degenerate graph with n vertices and $m \geq \lceil \frac{nk}{2} \rceil$ edges with probability at least $1/D_{n,m}$ in time $O(nmk)$.*

Proof. The probabilities result from sampling uniformly from \mathcal{D}_n and $\mathcal{D}_{n,m}$ in line 1, respectively, and from $\mathcal{S}_n^{(k)} \subset \mathcal{D}_n^{(k)}$ and $\mathcal{S}_{n,m}^{(k)} \subset \mathcal{D}_{n,m}^{(k)}$.

The first step of the algorithm can be performed in time $O(n \log k + m)$ after a precomputation of $O(nk)$ and $O(nmk)$, respectively. The algorithm then terminates after exactly $\Delta(G)$ applications of Lemma 3.1, where G denotes the graph generated in line 1. For a well-ordered k -degenerate graph G with n vertices $\Delta(G) \leq nk$. An application of Lemma 3.1 can be implemented to run in $O(n)$ time. Therefore, the algorithm can be implemented to run in $O(kn^2)$ and $O(nmk)$ time, respectively. ■

Algorithm 4 can also be used in combination with the acceptance-rejection method. By rejecting all graphs with deficiency greater than zero, we obtain a slow but uniform sampler for well-ordered strongly k -degenerate graphs. If we accept all graphs, on the other hand we obtain a fast

generator with non-uniform distribution. By rejecting graphs with high deficiency and accepting those with low deficiency, we obtain generators in between the two extremes. Note that the algorithm can be used to generate k -regular graphs in $O(k^2n^2)$ time.

4 Enumerating Well-Ordered (Strongly) k -Degenerate Graphs

In this section we show how to enumerate well-ordered strongly k -degenerate graphs with n vertices (and m edges). In order to obtain a unique representation of the well-ordered k -degenerate graphs we order edges by their smaller vertex and break ties by their bigger vertex. Further, we order edge-sets using the edge-order and order graphs *lexicographically* by lexicographically ordering their edge-sets.

4.1 Enumerating Well-Ordered (Strongly) k -Degenerate Graphs with n Vertices

The enumeration algorithm outlines as follows: At first we generate all (edge-)maximal well-ordered k -degenerate graphs. For each maximal well-ordered k -degenerate graph G we identify and enumerate those well-ordered strongly k -degenerate subgraphs whose lexicographically smallest well-ordered k -degenerate supergraph is exactly G . Thus, we generate each graph exactly once.

At first we describe how to generate all maximal well-ordered k -degenerate graphs. We recall that, for a well-ordered k -degenerate graph G , the maximum outdegree $d_{\max}^+(v_i)$ of vertex v_i is $\min\{n - i, k\}$. Then, the maximum number of edges of a well-ordered k -degenerate graph with n vertices equals

$$(4.11) \quad M(n, k) := \sum_{i=1}^n d_{\max}^+(v_i) = nk - \binom{k+1}{2}.$$

A k -degenerate graph is *maximal* if it has $M(n, k)$ edges. This leads to the following observation.

OBSERVATION 1. *Let G be a maximal well-ordered k -degenerate graph on n vertices and let G' be the graph induced by v_{n-k}, \dots, v_n . Then G' is a clique on $k+1$ vertices. Thus, any maximal well-ordered k -degenerate graph is strongly k -degenerate.*

For an integer i , let C_i denote the set $\{v_{i+1}, \dots, v_n\}$, i.e., C_i is the set of possible neighbors of v_i which are larger than v_i . Using the previous observation we obtain:

LEMMA 4.1. *Let \mathcal{M} be the set of all well-ordered maximal k -degenerate graphs. Then there is a bijection*

$$(4.12) \quad \Phi_{n,k} : \mathcal{M} \rightarrow \bigtimes_{i=1}^{n-k-1} \binom{C_i}{k}.$$

Proof. Let G be a well-ordered k -degenerate graph and let $N^+(v_i)$ be defined as the set of neighbors of v_i with index greater than i . We define $\Phi_{n,k}(G) := (N^+(v_i))_{i=1}^{n-k-1}$. Clearly, $\Phi_{n,k}$ is injective by Observation 1. To show that $\Phi_{n,k}$ is surjective let $(X_i)_{i=1}^{n-k-1} \in \times_{i=1}^{n-k-1} \binom{C_i}{k}$. Then $\Phi_{n,k}^{-1}(x)$ is a graph $G = (V, E)$ such that $V = \{v_1, \dots, v_n\}$ and $E = \{\{v_i, x\} \mid 1 \leq i \leq n - k - 1, x \in X_i\} \cup \{\{v_i, v_j\} \mid n - k \leq i < j \leq n\}$. Since every vertex v_i satisfies $d^+(v_i) = d_{\max}^+(v_i)$ the graph G is a maximal well-ordered k -degenerate graph. ■

The representation given in Equation (4.12) can also be encoded by a sequence of strings as follows. Each set $A \in \binom{C_i}{k}$ can be represented by a string $s(A)$ consisting of $|C_i| - k$ zeros and k ones. Then each graph can be represented by the concatenation of these strings. Since it is possible to enumerate all possible binary strings consisting of $|C_i| - k$ zeros and k ones in constant amortized time [13], we can enumerate each graph in constant amortized time and, thus, print every maximal well-ordered k -degenerate graph in $O(n + m)$ amortized time per printed graph using a suitable data-structure.

For a well-ordered k -degenerate graph G we denote the lexicographically smallest maximal well-ordered k -degenerate supergraph by $S(G)$. A well-ordered k -degenerate graph G with n vertices is called a *proper subgraph* of a maximal well-ordered k -degenerate graph S with n vertices if and only if $S = S(G)$. We observe that the lexicographically smallest maximal well-ordered k -degenerate supergraph of G can be obtained by including the $d_{\max}^+(v_i) - d^+(v_i)$ lexicographically smallest edges for every vertex v_i which are not yet incident to v . Let $X(G)$ denote this set of edges. Given a graph G , the *foundation* $F(v_i)$ of a vertex v_i is the largest set of edges $\{v_i, v_{i+1}\}, \dots, \{v_i, v_{i+\ell}\}$ such that $F(v_i) \in E$. The foundation of a graph G is defined to be $F(G) = \bigcup_{v \in V} F(v)$. We state the following:

OBSERVATION 2. *Let S be a maximal well-ordered k -degenerate graph with n vertices and let $G \neq S$ be a subgraph of S with n vertices. Then G is a proper subgraph of S if and only if $E(S) \setminus E(G) \subseteq F(S)$.*

Using this observation we can characterize the proper subgraphs of a maximal well-ordered k -degenerate graph S by

$$(4.13) \quad P(S) = \{S - E' \mid E' \subseteq F(S), S - E' \text{ is well-ordered (strongly) } k\text{-degenerate}\}.$$

This characterization can be used to design an efficient enumeration algorithm for well-ordered (strongly) k -degenerate graphs. The following Algorithm 5 enumerates all well-ordered k -degenerate graphs using this characterization by branching on the subsets of the foundation. The input of the algorithm consists of a graph G , a set of edges X and a maximal well-ordered k -degenerate graph S . At each step the

algorithm branches on some edge $e \in (F(S) \cap E(G)) \setminus X$. For each such edge the algorithm checks if $G - e$ is well-ordered (strongly) k -degenerate. In this case, the algorithm outputs $G - e$ and recursively calls itself on the smaller graph. If the graph is not well-ordered (strongly) k -degenerate, then no subgraph of G with n vertices can be well-ordered (strongly) k -degenerate: Hence, we need not branch further.

Input: well-ordered (strongly) k -degenerate graph G , forbidden-edges X , supergraph S
Output: all well-ordered (strongly) k -degenerate proper subgraphs of S containing no edges in X

```

1 if  $(F(S) \cap E(G)) \setminus X \neq \emptyset$  then
2    $e \leftarrow$  choose an edge in  $(F(S) \cap E(G)) \setminus X$ 
3   if  $G - e$  is well-ordered (strongly)  $k$ -degenerate then
4     print  $G - e$ 
5     ENUMERATE-FOR-GRAPH( $G - e, X, S$ )
6   ENUMERATE-FOR-GRAPH( $G, X \cup e, S$ )

```

Algorithm 5: ENUMERATE-FOR-GRAPH(G, X, S)

LEMMA 4.2. *Let S be a maximal well-ordered k -degenerate graph. Then Algorithm 5 enumerates and prints all proper subgraphs of S in time $O(nm + m^2)$ per printed graph on input (S, \emptyset, S) .*

Proof. The algorithm recursively branches on the edges of $F(S)$ enumerating the subsets of $F(S)$ with decreasing size. Clearly, the algorithm only prints proper subgraphs of S by Equation (4.13).

On the other hand, let G be a proper subgraph of S , i.e., G is well-ordered k -degenerate and $E(S) \setminus E(G) \subseteq F(S)$. Let $Y \subseteq E(S) \setminus E(G)$. Since G and S are well-ordered k -degenerate, so is $G + Y$. Let e_1, \dots, e_ℓ be any sequence of edges such that $\{e_1, \dots, e_k\} = E(S) \setminus E(G)$ then $G_i := G + \{e_1, \dots, e_i\}$ is well-ordered k -degenerate. Therefore the algorithm reaches and prints G when branching on the edges in $F(S)$.

In order to see why the amortized time per generated graph equals $O(nm + m^2)$ we consider the search tree of the algorithm. Let G be a (strongly) k -degenerate graph and let N be the node in the search tree which is responsible for printing G . The maximum number of nodes which can be reached from N in the subtree rooted at N without printing another (strongly) k -degenerate graph is at most m . Since the running time per node is $O(n + m)$ the algorithm has an amortized running time of $O(nm + m^2)$ per graph. ■

Finally, Algorithm 6 contains the pseudo-code for the topmost level. Note that the algorithm can be used to enumerate both regular and well-ordered strongly k -degenerate graphs with n vertices.

Input: number of vertices n
Output: all well-ordered (strongly) k -degenerate graphs
1 **for** all maximal well-ordered k -degenerate graphs S
 do
2 | print S
3 | ENUMERATE-FOR-GRAPH($S, \emptyset, S[, m]$)

Algorithm 6: ENUMERATE-DEGENERATE($n[, m]$)

THEOREM 4.1. *Algorithm 6 enumerates all well-ordered (strongly) k -degenerate graphs in amortized $O(nm + m^2)$ time per graph.*

Proof. Since the algorithm enumerates the proper subgraphs of all maximal well-ordered k -degenerate graphs it enumerates every graph exactly once. By Equation (4.13) and correctness of Algorithm 5 (Lemma 4.2) the algorithm enumerates all proper subgraphs for a given maximal well-ordered k -degenerate graph. Since we can generate and print the maximal well-ordered k -degenerate graphs in amortized linear time the overall running time of the algorithm is $O(nm + m^2)$ per enumerated and printed graph. ■

4.2 Enumerating Well-Ordered (Strongly) k -Degenerate Graphs with n Vertices and m Edges In this section, we describe how to enumerate well-ordered (strongly) k -degenerate graphs with n vertices and m edges. In order to obtain an efficient algorithm we first establish a simple criterion to decide if a given graph G contains a subgraph H such that $G - E(H)$ is strongly k -degenerate.

Let G be a given well-ordered strongly k -degenerate graph and let $X \subseteq E(G)$ be a set of edges. We define G_X^F as the graph induced by the edges in $(E(G) \setminus X) \cap F(S(G))$. Consider the following problem: Given G_X^F as well $u_i = d_G(v_i) - k$ for $i = 1, \dots, n$ find the subgraph H of G_X^F with the maximum number of possible edges such that $d_H(v_i) \leq u_i$. We call the maximum number of possible edges of H the *excess* of G with respect to X , denoted by $xs(G, X)$. The excess can be computed in time $O(\sum_{i=1}^n \sqrt{u_i} \cdot m) \subseteq O(n^{3/2}m)$ [9].

LEMMA 4.3. *Let $G = (V, E(G))$ be a well-ordered strongly k -degenerate graph with n vertices and $m > m'$ edges and let $X \subseteq E(G)$ be a set of edges. Then G contains a well-ordered strongly k -degenerate subgraph G' with n vertices and m' edges which is a proper subgraph of $S(G)$ and which contains all edges in X if and only if $xs(G, X) \geq m - m'$.*

Proof. “if”: Assume that $xs(G, X) \geq m - m'$. Then G contains a subgraph H with edge-set $E(H)$ such that $|E(H)| \geq m - m'$ and $E(H) \subseteq (E(G) \setminus X) \cap F(S(G))$. Let $E' \subseteq E(H)$ with $|E'| = m - m'$. Then $G - E'$ is a well-ordered strongly

k -degenerate proper subgraph of $S(G)$. Clearly, $G - E'$ is strongly k -degenerate since all vertices have degree $\geq k$ by definition of u_i . Additionally, $G - E'$ is a proper subgraph of $S(G)$ since $E' \subseteq F(S(G))$ by Observation 2.

“only if”: Assume that G contains a well-ordered strongly k -degenerate subgraph G' with m' edges which is a proper subgraph of $S(G)$ and which contains all edges in X . Let $E' := E(G) \setminus E(G')$. Then clearly, $E' \subseteq E(G) \setminus X$ and $E' \subseteq F(S(G))$ by Observation 2, i.e., $E' \subseteq (E(G) \setminus X) \cap F(S(G))$. Hence, E' is a subset of $E(G_X^F)$. Let $H := G(V, E')$, then H is a subgraph of G_X^F and for all vertices v_i we have $d_H(v_i) \leq u_i = d_G(v_i) - k$ and, therefore, $xs(G, X) \geq |E'| = m - m'$. ■

Using Lemma 4.3 in combination with Algorithms 6 and 5 we can enumerate all well-ordered strongly k -degenerate graphs with n vertices and m edges by slightly modifying Algorithm 5: We extend the check in line 3 by testing if $xs(G - e, X) \geq |E(G)| - m$. We further add a similar check in line 6. The pseudo-code is listed in Algorithm 7.

Input: (well-ordered strongly) k -degenerate graph G , forbidden-edges X , supergraph S
Output: all well-ordered (strongly) k -degenerate proper subgraphs of S containing no edges in X

```

1 if  $(F(S) \cap E(G)) \setminus X \neq \emptyset$  then
2   |  $e \leftarrow$  choose an edge in  $(F(S) \cap E(G)) \setminus X$ 
3   | if  $G - e$  is (strongly)  $k$ -degenerate and
   |  $xs(G - e, X) \geq |E(G - e)| - m$  then
4     | if  $|E(G)| = m$  then
5       | | print  $G - e$ 
6       | | ENUMERATE-FOR-GRAPH( $G - e, X, S$ )
7   | if  $xs(G, X \cup \{e\}) \geq |E(G)| - m$  then
8     | | ENUMERATE-FOR-GRAPH( $G, X \cup \{e\}, S$ )

```

Algorithm 7: ENUMERATE-FOR-GRAPH(G, X, S, m)

THEOREM 4.2. *Algorithm 7 enumerates and prints all well-ordered (strongly) k -degenerate graphs with n vertices and m edges in amortized $O(n^{3/2}m^2)$ time per printed graph.*

Proof. The proof is similar to the proof of Theorem 4.1. The running time results from the fact that we can check $xs(G, X) \geq i$ in time $O(n^{3/2}m)$. Let G be a well-ordered (strongly) k -degenerate graph which is printed by the algorithm. In the search tree we then consider the vertex associated with G as well as the path from the source to G and all vertices which can be reached from this path at distance one. Clearly, this path contains $O(m)$ vertices. In total these vertices account for at most $O(n^{3/2}m^2)$ time of the algorithm. Since we only branch on graphs which have at most one well-ordered (strongly) k -degenerate subgraph, we

cover the total running time by assigning $O(n^{3/2}m^2)$ time to each printed graph. ■

In order to enumerate ordinary well-ordered k -degenerate graphs with n vertices and m edges we need only check if $|E(G - e) \cap F(S(G))| \geq |E(G - e)| - m$. This can be done in constant time if we allow to have a marker for edges in the foundation of $S(G)$. Testing whether the graph is well-ordered k -degenerate and printing the graph can be done in time $O(n + m)$. Therefore, we can enumerate ordinary well-ordered k -degenerate graphs with n vertices and m edges in time $O(nm + m^2)$ per enumerated graph.

5 Open Problems and Future Work

In this paper we have presented efficient algorithms for enumerating and generating well-ordered k -degenerate graphs. The problem of generating well-ordered strongly k -degenerate graphs uniformly at random remains open. This problem seems particularly interesting since the strongly k -degenerate graphs with n vertices and $nk/2$ edges are exactly the k -regular graphs. Solving this problem may yield new insights into the problem of generating labeled k -regular graphs uniformly at random which has been studied for some time.

Acknowledgments. We would like to thank Marco Gaertler for the discussions about the strongly k -degenerate graphs.

References

- [1] Noga Alon and Shai Gutner. *Computing and Combinatorics*, chapter Linear Time Algorithms for Finding a Dominating Set of Fixed Size in Degenerated Graphs. Springer Berlin / Heidelberg, 2007.
- [2] Michael Baur, Marco Gaertler, Robert Görke, Marcus Krug, and Dorothea Wagner. Generating Graphs with Predefined k -Core Structure. In *Proceedings of the European Conference of Complex Systems (ECCS'07)*, October 2007. Online proceedings <http://cssociety.org/ECCS07-Programme>.
- [3] Manuel Bodirsky, Clemens Gröpl, and Mihyun Kang. Generating labeled planar graphs uniformly at random. *Theor. Comput. Sci.*, 379(3):377–386, 2007.
- [4] Béla Bollobás, Alexandr Kostochka, and Kittikorn Nakprasit. Packing d -degenerate graphs. *Journal of Combinatorial Theory, Series B*, 98(1):85 – 94, 2008.
- [5] Leizhen Cai, Siu Man Chan, and Siu On Chan. Random separation: A new method for solving fixed-cardinality optimization problems. In *IWPEC*, pages 239–250, 2006.
- [6] Leizhen Cai and Xuding Zhu. Game chromatic index of k -degenerate graphs. *J. Graph Theory*, 36(3):144–155, 2001.
- [7] Luc Devroye. *Non-Uniform Random Variate Generation*. Springer Berlin / Heidelberg, 1986.
- [8] Éric Fusy. Quadratic exact-size and linear approximate-size random generation of planar graphs. In Conrado Martínez, editor, *2005 International Conference on Analysis of Algorithms*, volume AD of *DMTCS Proceedings*, pages 125–138. Discrete Mathematics and Theoretical Computer Science, 2005.
- [9] Harold N. Gabow. An efficient reduction technique for degree-constrained subgraph and bidirected network flow problems. In *STOC '83: Proceedings of the fifteenth annual ACM symposium on Theory of computing*, pages 448–456, New York, NY, USA, 1983. ACM.
- [10] Petr A. Golovach and Yngve Villanger. *Graph-Theoretic Concepts in Computer Science*, chapter Parameterized Complexity for Domination Problems on Degenerate Graphs. Springer Berlin / Heidelberg, 2008.
- [11] Jeong Han Kim and Van H. Vu. Generating random regular graphs. In *STOC '03: Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 213–222, New York, NY, USA, 2003. ACM.
- [12] Brendan D. McKay and Nicholas C. Wormald. Uniform generation of random regular graphs of moderate degree. *J. Algorithms*, 11(1):52–67, 1990.
- [13] Frank Ruskey and Aaron Williams. The coolest way to generate combinations. *Discrete Mathematics*, 309(17):5305 – 5320, 2009. Generalisations of de Bruijn Cycles and Gray Codes; Graph Asymmetries; Hamiltonicity Problem for Vertex-Transitive (Cayley) Graphs.
- [14] Xiao Zhou Shuji Isobe and Takao Nishizeki. Total colorings of degenerate graphs. *Combinatorica*, 2007.
- [15] A. Steger and N. C. Wormald. Generating random regular graphs quickly. *Comb. Probab. Comput.*, 8(4):377–396, 1999.
- [16] Nick Wormald. Models of random regular graphs. In *Surveys in Combinatorics*, pages 239–298. Cambridge University Press, 1999.