2015 IEEE 56th Annual Symposium on Foundations of Computer Science

Uniform generation of random regular graphs

Pu Gao
Department of Combinatorics and Optimization
University of Waterloo
Waterloo, Canada
p3gao@uwaterloo.ca

Nicholas Wormald
School of Mathematical Sciences
Monash University
Melbourne, Australia
nick.wormald@monash.edu

Abstract

We develop a new approach for uniform generation of combinatorial objects, and apply it to derive a uniform sampler *REG* for d-regular graphs. *REG* can be implemented such that each graph is generated in expected time $O(nd^3)$, provided that $d = o(\sqrt{n})$. Our result significantly improves the previously best uniform sampler, which works efficiently only when $d = O(n^{1/3})$, with essentially the same running time for the same d. We also give a linear-time approximate sampler *REG**, which generates a random d-regular graph whose distribution differs from the uniform by $o(1)$ in total variation distance, when $d = o(\sqrt{n})$.

Keywords

Uniform generation; regular graphs; switching; Markov chain.

I. Introduction

Research on uniform generation of random graphs is almost as old as modern computing. Tinhofer [16] gave a generation algorithm in which the probabilities of the graphs are computed *a posteriori*. In theory, this could be used to produce any desired distribution by rejection sampling, but no explicit bounds on the time complexity of this method are known. The earliest method useful in practice for achieving exactly uniform generation arose from the enumeration methods of Békéssy, Békéssy and Komlós [1], Bender and Canfield [3] and Bollobás [4]. This works in linear expected time for graphs with bounded maximum degree d, though the multiplicative constant behaves like $e^{(d-1)^2/4}$ as a function of d (making it rather impractical for moderately large d, even $d = 12$). It generalises easily to a simple algorithm for uniform generation of graphs with given degrees. (See, for example, [17], which in addition gives an algorithm for 3-regular graphs that has linear deterministic time.) The algorithm starts by generating a pairing, to be defined below, uniformly at random. If the corresponding graph is simple then it is outputted. Otherwise, the algorithm is restarted.

An advance was made by McKay and Wormald [12], whose algorithm uniformly generates random graphs with given degrees, in polynomial time, as long as the maximum degree is $O(M^{1/4})$ where M is the total degree. In the case of d-regular graphs the expected running time is $O(nd^3)$ provided that $d = O(n^{1/3})$. This is currently the best result in this direction.

When uniform generation of some class of objects seems difficult, a fallback position is to investigate approximate solutions. One approach is to use the Markov Chain Monte Carlo (MCMC) method. In this, an ergodic Markov chain on the set of graphs with given degrees is designed so that the stationary distribution is uniform. Then, the random graph obtained after taking a sufficiently large number of steps (i.e. the so-called mixing time of the chain) has distribution that is close to uniform. Jerrum and Sinclair [9] gave a fully polynomial time approximation scheme (FPTAS) for approximate uniform generation of graphs with given degrees. Their algorithm works for a large class of degree sequences, in particular, all regular graphs. However, the degree of the polynomial is too high for any practical use (and is not optimised in their paper). A similar result for bipartite graphs, but with a different chain that switches edges around, was obtained by Kannan, Tetali and Vempala [10]. This Markov chain was further extended by Cooper, Dyer and Greenhill [5] for generation of random d-regular graphs. They showed that the mixing time is then bounded by roughly $d^{24}n^9 \log n$. Very recently, Greenhill [6] extended that work to the non-regular case, requiring a running time of $\Delta^{14}M^{10} \log M$ where Δ and M denote the maximum and total

Research supported by NSERC
Research supported by Australian Laureate Fellowships grant FL120100125.

0272-5428/15 $31.00 © 2015 IEEE
DOI 10.1109/FOCS.2015.78

degrees respectively. This result applies only for $3 \leq \Delta \leq \frac{1}{4}\sqrt{M}$. These MCMC-based algorithms generate every graph with a probability within a factor $1 \pm \epsilon$ of the probability in the uniform distribution, and $\epsilon > 0$ can be made arbitrarily small by running the chain sufficiently long. So the output of such algorithms is almost as good as that from an exactly uniform sampler, for any practical use. However, the high time complexity such as $d^{24}n^9$ renders them impractical.

There is a variation of MCMC called *coupling from the past*, which is capable of producing a target distribution precisely. However, it is often hard to prove useful bounds on the running time, and the method has not been successfully applied to generating random graphs given degrees.

There are algorithms faster than the MCMC-based ones, that generate graphs with a weaker approximation of the distribution to the uniform. Steger and Wormald [15] gave an $O(d^2n)$-time algorithm that generates random d-regular graphs for d up to $n^{1/28}$, where all graphs are generated with asymptotically the same probability. Kim and Vu [11] proved that the same algorithm works to the same extent for all $d \leq n^{1/3-\epsilon}$. Bayati et al. [2] subsequently modified and extended it for the non-regular case, under certain conditions, and generated random d-regular graphs for all $d \leq n^{1/2-\epsilon}$, but with a weaker approximation to uniform (bounding the total variation distance by o(1)). Using a different approach, Zhao [18] reports a similar approximation obtained by a $O(dn)$-time algorithm provided $d = o(n^{1/3})$. These algorithms are much faster than using MCMC, but the approximation (to the uniform) is achieved only asymptotically as $n \to \infty$. Unlike MCMC, the error cannot be improved, for a fixed size, by more computation.

The main purpose of the present paper is to introduce a new general framework of uniform generation of combinatorial structures. We will apply it in this paper to the special, nevertheless particularly interesting, case of random d-regular graphs on n vertices. The result is effective for $d = o(\sqrt{n})$. This significantly improves the bounds on d in [12]. Our algorithm is an exactly uniform sampler and thus there is no uncontrollable distortion as in [2], [11], [15], [18]. Moreover, the expected running time of our algorithm per graph generated is $O(nd^3)$ (as in [12]), which remains quite practical, comparing favourably with the MCMC approach.

We outline the general framework in Section III. The framework includes several operations and parameters that will be defined in accordance with the types of combinatorial structures to be generated. For the application in the present paper, this framework is used in each of the three phases of the main algorithm, called *REG*, that is a uniform sampler for d-regular graphs on n vertices. The operations and parameters for the various phases of *REG* are defined in Section VI.

Theorem 1. *Algorithm* REG *generates d-regular graphs uniformly at random.*

Theorem 2. REG *can be implemented so that for $1 \leq d = o(\sqrt{n})$, the expected time complexity for generating a graph is $O(nd^3)$.*

In some applications, one might care more for a low time complexity than a perfect uniform sampling. As a byproduct of Theorem 1, by omitting several features of *REG*, we will obtain a simpler, linear-time algorithm (that is, linear in the output size, which is the number of edges) called *REG**, which approximately generates a random d-regular graph. *REG** is defined in Section VII. For this problem, it improves the running time $O(nd^2)$ of [2], and the range $d = o(n^{1/3})$ of [18].

Theorem 3. *Algorithm* REG* *randomly generates a d-regular graph whose total variation distance from the uniform distribution is $o(1)$, for any $d = o(\sqrt{n})$. Moreover, the expected number of steps required for generating a graph is $O(dn)$.*

We note that several papers have adapted the approach of [12] for generation of other structures (e.g. McKay and Wormald [14]), and also for enumeration (e.g. Greenhill and McKay [7]). Such works have not led to improvements in the result [12] achieves for d-regular graphs. We expect the ideas introduced here will filter out to improved results for several kinds of structures, including graphs with non-regular degree sequences. Such issues will be examined elsewhere.

II. The old and the new

In this section we first summarise the procedure *DEG* used in [12], which provides some of the foundations required for applying our method to regular graphs. We then highlight the new ideas used in *REG*, and give a

skeleton description of that algorithm. Finally, we describe the layout of the paper in relation to exposing the new framework and defining and analysing *REG* and *REG**.

For generating random graphs with given degrees, we use the *pairing model*, first introduced in [4]. Let $\mathbf{d} = (d_1, \ldots, d_n)$ be a degree sequence (so $\sum_{i=1}^{n} d_i$ is even). Represent each vertex $i \in [n]$ as a bin v_i. Place d_i distinct points in bin v_i for every $1 \leq i \leq n$. Take a uniformly random perfect matching of the $\sum_i d_i$ points. This perfect matching is called a *pairing*; each pair of points joined in the matching is called a *pair* of the pairing. Note that each pairing P corresponds to a multigraph with degree sequence $\mathbf{d}$, denoted by $G(P)$, obtained by regarding each pair in the pairing as an edge. Moreover, by a simple counting argument, we see that every simple graph of degree sequence $\mathbf{d}$ corresponds to the same number of pairings. Thus, letting Φ denote the whole set of pairings and $\mathcal{B} \subseteq \Phi$ the set of pairings corresponding to simple graphs, if a pairing $P \in \mathcal{B}$ is generated uniformly at random, then $G(P)$ has the uniform distribution over all graphs with degree sequence $\mathbf{d}$.

Given a pairing P, and two vertices i and j, the set of pairs between i and j in P, if non-empty, is called an *edge*, i.e. edge ij, and the size of the set is the *multiplicity* of ij. If the multiplicity of ij is 1, then ij is a *single* edge; otherwise it is a *multi-edge*. We say ij is a *double*, *triple* or *quadruple* edge if its multiplicity is 2, 3 or 4 respectively. An edge ij with $i = j$ called a *loop* at i.

Outline of *DEG*

An appropriate set $\mathcal{A} \subseteq \Phi$ is pre-defined, such that pairings in $\mathcal{A}$ have no multi-edges other than double non-loop edges, and have limited numbers of double edges and loops. *DEG* begins by repeatedly generating a uniformly random pairing $P_0 \in \Phi$ until $P_0 \in \mathcal{A}$. When $d = O(n^{1/3})$, only a constant number of iterations is needed. Afterwards, two phases are applied in turn, to reduce the loops and then the double edges. Each phase consists of a sequence of operations called *switchings* applied to the initial pairing, causing the number of undesired structure (loops or double edges) to decrease by one in each step. If a switching converts a pairing P to P' in a certain step, a rejection without an output can occur with a small probability. This probability is a function of P and P', and is designed in a way that guarantees the uniformity of P' in $\mathcal{S}_i$, the set of pairings containing exactly i double edges (or loops, depending on the phase), conditional on P being uniformly random in $\mathcal{S}_{i+1}$. Inductively, the output of a phase is uniformly distributed in $\mathcal{S}_0$.

The range of applicable degree sequences for *DEG* is determined by the probability of rejection at some time during the algorithm. In [12], the bound $d = O(n^{1/3})$ for d-regular graphs could not be weakened because the probability of rejection would get too close to 1. The double-edge reduction phase of *DEG*, critical in determining the applicable range of d, is the one that we improve here.

New features in *REG*

Our new approach extends that used in *DEG*, introducing some major new features in both the algorithm specification and its analysis, employed specifically in the double-edge reduction phase. For one thing, we will permit certain switchings, called *class B*, that do not have the desired effect on reducing the number of multi-edges. This achieves a smaller variation in the number of permitted switchings of a pairing in $\mathcal{S}_i$, and thus the probability of a rejection. We also introduce a feature called *boosting*, whereby a different type of switching occasionally creates some otherwise under-represented elements of $\mathcal{S}_i$. This also reduces the probability of a rejection. As a result, the algorithm no longer proceeds through the sets $\mathcal{S}_i$ step-by-step, decreasing i by 1 at each step, though this is still the most likely route. Additionally, unlike in [12], the probability that a pairing is reached in the algorithm does not depend only on the set $\mathcal{S}_i$ to which it belongs, and not all switchings to pairings in such a set will be performed with the same probability. As a result of these changes, the analysis becomes intricate. In particular, we are forced to relinquish keeping the current pairing uniformly distributed in its set $\mathcal{S}_i$ during the algorithm. Instead, we focus on the expected numbers of visits to the states in the associated Markov chain.

The rest of the story

As in [12], a set $\mathcal{A} \subseteq \Phi$ is specified such that $\mathbb{P}(\mathcal{A})$ is bounded away from zero. A uniformly random pairing $P \in \Phi$ is generated, and initial rejection is performed if $P \notin \mathcal{A}$. Pairings in $\mathcal{A}$ will in general contain loops, and double and triple non-loop edges, but no other multi-edges. Three phases are performed sequentially, for reduction of loops, then triple edges, and finally double edges.

Since the new features of *REG* will be useful in other contexts, we set up a general framework for the description of a phase in Section III. This new framework results in a different analysis from [12], which will be given in

Section IV. The proof that each phase ends with a uniformly random object of the required type is rather involved, so an example appropriate to the double-edge reduction phase is given in Section V. This includes an illustration of how to set some of the parameters of a phase appropriately. Since the only complicated phase in this paper is for double-edge reduction, the definitions of the switchings and other parameters in this phase, and the analysis required for bounding the time complexity, is done in Section VI.

Finally in Section VII, we prove the main theorems by bounding the expected running time. This is basically determined by the task of computing the probability of rejection. By ignoring rejections (carrying on regardless) and simplifying the steps for reducing double edges, we obtain the approximate sampler *REG** in Section VII. We prove that *REG** runs in linear time in expectation for generating one random regular graph, and bound the total variation distance between the distribution of the output of *REG** and the uniform distribution.

III. General description of a phase

We present here the definition of a phase that will be common to any application of our approach to reducing the number of occurrences of some undesired configuration (for instance, a double edge), by using repetitions of operations, called switchings here, that can be defined to suit the application.

A phase begins with a set Φ_1 partitioned into subsets $\mathcal{S}_i$, $0 \le i \le i_1$, called *strata*, for some integer $i_1 \ge 0$. For $P \in \Phi_1$, set $\mathcal{S}(P) = i$ if $\mathcal{S}_i$ contains P. A set of possible operations, called switchings, is specified. Each switching converts some $P \in \Phi_1$ to another element of Φ_1, and has both a *type* and a *class*. The set of types is $\mathcal{T}$, and the set of classes is $\mathcal{C}$. The phase begins with a random element $P \in \Phi_1$ with the uniform distribution π_0, and either outputs a uniformly random element of $\mathcal{S}_0$, or terminates with no output (rejection). Each phase requires specified parameters $\rho_\tau(i)$, $\overline{m}_\tau(i)$ and $\underline{m}_\alpha(i)$, for each switching type τ and class α, and each $i \le i_1$.

The parameters $\rho_\tau(i)$ satisfy $\sum_\tau \rho_\tau(i) \le 1$ for each i. If an element $P \in \mathcal{S}_i$ ($i > 0$) arises during the phase, a switching type τ is chosen with probability $\rho_\tau(i)$. If $\sum_\tau \rho_\tau(i) < 1$, a rejection will occur with the remaining probability; we call this a *t-rejection* (where 't' stands for 'type').

The number of switchings of type τ that can be applied to $P \in \Phi_1$ is denoted by $f_\tau(P)$. The parameters $\overline{m}_\tau(i)$ satisfy

$$\overline{m}_\tau(i) \ge \max_{P \in \mathcal{S}_i} f_\tau(P).$$

Similarly, the number of switchings of class α that can be applied (to other elements of Φ_1) to produce P is denoted by $b_\alpha(P)$. The parameters $\underline{m}_\alpha(i)$ satisfy

$$\underline{m}_\alpha(i) \le \min_{P \in \mathcal{S}_i} b_\alpha(P).$$

The phase consists of repetitions of a *switching step*, specified as follows.

Given $P \in \mathcal{S}_i$,

(i) If $i = 0$, output P.

(ii) Choose a type: choose τ with probability $\rho_\tau(i)$. Otherwise, i.e. with probability $1 - \sum_\tau \rho_\tau(i)$, t-rejection occurs. Then select u.a.r. a switching of type τ that can be performed on P.

(iii) Let P' be the element that the selected switching would produce if applied to P, let α be the class of the selected switching and let $i' = \mathcal{S}(P')$. Perform a rejection (called an f-rejection, where 'f' stands for 'forward') with probability $1 - f_\tau(P)/\overline{m}_\tau(i)$ and then perform another rejection (called a b-rejection, where 'b' stands for 'backward') with probability $1 - \underline{m}_\alpha(i')/b_\alpha(P')$;

(iv) if no rejection occurred, replace P with P'.

The switching step is repeated until the phase terminates, which happens whenever an element $P \in \mathcal{S}_0$ is reached or a rejection occurs.

Note that in each switching step only a switching type is selected, not a switching class. Only after a switching is chosen in (ii) is the class of the switching determined.

To complete the definition of a phase, it is sufficient to specify Φ_1, i_1, the sets $\mathcal{S}_i$, the set of switchings and their types and classes, and choose the parameters $\rho_\tau(i)$, $\overline{m}_\tau(i)$ and $\underline{m}_\alpha(i)$. These numbers will be chosen to ensure that the expected number of times that a given element in $\mathcal{S}_i$ is visited during the phase depends only on i. Given this, and the fact that termination of the phase occurs as soon as $\mathcal{S}_0$ is reached, it follows that the element outputted is distributed u.a.r. from $\mathcal{S}_0$. Subject to this, in choosing the parameters we also aim to keep the probability of rejection small.

IV. General analysis of a phase

In this section we lay the groundwork for specification of the predefined parameters described in Section III, in such a way that the algorithm performs the desired uniform sampling. Consider a phase with specified switchings and parameters $\mathcal{S}_i$, $\rho_\tau(i)$, $\overline{m}_\tau(i)$ and $\underline{m}_\alpha(i)$ for all appropriate i and τ, as well as a specified value of i_1. Assume that

$$\rho_\tau(i) \geq 0 \quad \text{for all } \tau \in \mathcal{T}, \qquad \sum_{\tau \in \mathcal{T}} \rho_\tau(i) \leq 1 \quad \text{for all } 0 \leq i \leq i_1; \tag{1}$$

$$f_\tau(P) \leq \overline{m}_\tau(i) \text{ and } b_\alpha(P) \geq \underline{m}_\alpha(i) \quad \text{for all } 0 \leq i \leq i_1,\ P \in \mathcal{S}_i,\ \tau \in \mathcal{T},\ \alpha \in \mathcal{C}. \tag{2}$$

Recall that $\Phi_1 = \bigcup_{i=0}^{i_1} \mathcal{S}_i$. The parameters determine a Markov chain, denoted by $\mathscr{M}$, on states $\Phi_1 \cup \{R, F\}$, where R and F are two artificially introduced absorbing states. The chain moves from an element P directly to R (rather than to P') if rejection occurs in a switching step from P to P', and it moves to F with probability 1 in the next step after reaching any element in $\mathcal{S}_0$. We refer to the state at step t as P_t, permitting $P_t = R$ or F. We assume

(A1) all states in Φ_1 are transient in $\mathscr{M}$;

(A2) the initial probability distribution is $\pi_0(P) = 1/|\Phi_1|$ at each $P \in \Phi_1$, and zero elsewhere.

Let Q be the matrix of transition probabilities between all states of $\mathscr{M}$ in Φ_1. Thus $Q = (q_{P,P'})$ where $q_{P,P'}$ is the transition probability from P to P'.

The transition probability $q_{P,P'}$ can be computed as follows. Assume $P \in \mathcal{S}_i$ and that S is a type τ, class α switching that converts P into $P' \in \mathcal{S}_j$. Condition on state P being reached in $\mathscr{M}$. Then, from part (ii) of the switching step, the probability that S is chosen equals $\rho_\tau(i)/f_\tau(P)$. Hence, from part (iii), the probability S is performed and neither t- nor f-rejection occurs is $\rho_\tau(i)/\overline{m}_\tau(i)$. On the other hand, the probability that b-rejection does not occur is $\underline{m}_\alpha(j)/b_\alpha(P')$. Thus

$$q_{P,P'} = \sum_{(\tau,\alpha)} N_{\tau,\alpha}(P,P') \frac{\rho_\tau(i)\,\underline{m}_\alpha(j)}{\overline{m}_\tau(i)\,b_\alpha(P')}, \tag{3}$$

where $N_{\tau,\alpha}(P,P')$ is the number of switchings of type τ and class α that convert P to P'.

By assumption (A1), Q is the submatrix of the transition matrix that refers to the transient states. Hence, the matrix $(I-Q)^{-1}$ exists. Indeed, this matrix is known as the fundamental matrix of $\mathscr{M}$, and it is clearly equal to $I + Q + Q^2 + \cdots$. (An easy argument shows that this series is convergent because these states are transient.) Moreover, the (P,P') entry of $(I-Q)^{-1}$ is clearly the expected number of visits (counting the initial position) to state P' given that the chain starts in state P. Let $\sigma(P)$ be the expected number of times that P is visited in $\mathscr{M}$. Then the vectors $\vec{\sigma}$ and $\vec{\pi}_0$, composed of the values $\sigma(P)$ and $\pi_0(P)$ respectively, are related by

$$\vec{\pi}_0 (I-Q)^{-1} = \vec{\sigma}. \tag{4}$$

A key feature of our approach is to specify $\rho_\tau(i)$ in such a way that $\sigma(P)$ depends only on $\mathcal{S}(P)$, i.e., there are fixed numbers $\sigma(i)$ $(0 \leq i \leq i_1)$ such that

$$\text{for all } i, \text{ and all } P \in \mathcal{S}_i, \quad \sigma(P) = \sigma(i). \tag{5}$$

To aid in finding such ρ_τ easily, we require that, for a given switching S, the expected number of switching steps during the phase in which S is chosen in (ii) for which f-rejection does not occur in (iii) is some number $q_\alpha(j)$ depending only on the class α of S and the set $\mathcal{S}_j$ containing the element it creates. Considering the derivation of (3), this is equivalent to requiring that, for all j and α,

$$\frac{\sigma(i)\rho_\tau(i)}{\overline{m}_\tau(i)} = q_\alpha(j) \text{ for all } (i,j,\tau,\alpha) \in \mathscr{S} \tag{6}$$

where $\mathscr{S}$ is the set of all (i,j,τ,α) for which there exists a switching of type τ and class α taking an element in $\mathcal{S}_i$ to an element in $\mathcal{S}_j$.

Rewrite (4) as $\vec{\sigma} = \vec{\pi}_0 + \vec{\sigma}Q$, and note from (3) that the component of $\vec{\sigma}Q$ referring to $P' \in \mathcal{S}_j$ is

$$\sum_P \sigma(P) q_{P,P'} = \sum_{(\tau,\alpha,P)} \sigma(P) N_{\tau,\alpha}(P,P') \frac{\rho_\tau(\mathcal{S}(P))\, \underline{m}_\alpha(j)}{\overline{m}_\tau(\mathcal{S}(P))\, b_\alpha(P')},$$

where the sum is over all τ, α and P for which there is at least one type τ, class α switching that converts P into P'. By (5) and (6), this summation is

$$\sum_{(\tau,\alpha,P)} N_{\tau,\alpha}(P,P') \frac{q_\alpha(j)\underline{m}_\alpha(j)}{b_\alpha(P')} = \sum_\alpha q_\alpha(j)\underline{m}_\alpha(j)$$

since for each $\alpha \in \mathcal{C}$, the number of class α switchings that converts some element P to P' is $\sum_{\tau,P} N_{\tau,\alpha}(P,P')$, which is $b_\alpha(P')$ by definition. Thus, provided that (5) and (6) hold, (4) is equivalent to

$$\sigma(j) = \frac{1}{|\Phi_1|} + \sum_{\alpha\in\mathcal{C}} q_\alpha(j)\underline{m}_\alpha(j) \text{ for all } j, \tag{7}$$

by (A2). Noting that (4) determines $\vec{\sigma}$, we have proved the following lemma.

Lemma 4. *Suppose that for given numbers $\rho_\tau(i)$, $\underline{m}_\alpha(j)$ and $\overline{m}_\tau(j)$ ($i,j \in [0,i_1]$, $\tau \in \mathcal{T}$, $\alpha \in \mathcal{C}$) satisfying the conditions (1)–(2), there is a simultaneous solution $(\sigma(i), q_\alpha(i))_{0\le i\le i_1}$ to equations (6) and (7). Then, for each $i \in [0,i_1]$, the expected number of visits to any given element in $\mathcal{S}_i$ is $\sigma(i)$.*

Remark. Consider the case that there is exactly one type τ and one class α of switchings involved in a phase, and each switching converts an element in $\mathcal{S}_{i+1}$ to another element in $\mathcal{S}_i$ for some $i \ge 0$. Then we may simply set $\rho_\tau(i) = 1$ for every $0 \le i \le i_1$. Clearly (A1) is satisfied, as there is no cycling in the Markov chain, and (6) and (7) combine to give

$$\sigma(j) = \frac{1}{|\Phi_1|} + \frac{\sigma(j+1)}{\overline{m}_\tau(j+1)} \cdot \underline{m}_\alpha(j).$$

Uniformity is guaranteed as $\sigma(j)$ can be inductively computed from $\sigma(j+1)$. This inductive approach is the essence of *DEG* [12] described in Section II, which is consequently a special case of the present method. The possibility of using different types and classes of switchings, the flexibility of setting non-trivial values to $\rho_\tau(i)$, and the much more flexible choice of Markov chains permitting cycling, all provide power to the approach in the present paper.

V. An example: calculating ρ and proving uniformity

The parameters $\overline{m}_\tau(i)$ and $\underline{m}_\alpha(i)$ will be specified, depending on the particular application, such that (2) is satisfied. The tightness of these bounds on $\max f_\tau(P)$ and $\min b_\alpha(P)$ effectively influence the efficiency of the phase, as tighter bounds yield smaller rejection probabilities in substep (iii) of a switching step. However, we need to set $\rho_\tau(i)$ properly to ensure (6), as well as to minimise the probability of t-rejection. We achieve this by deducing a system of equations and inequalities that the parameters $\rho_\tau(i)$ and the variables $\sigma(j)$ must satisfy. Then we find a desirable solution to the system, bearing in mind the rejection probabilities, and set the value of $\rho_\tau(i)$ accordingly. We give an example that will later be applied to phase 3 of *REG*, where double edges are eliminated.

We assume that all strata have been specified, as well as parameters $\overline{m}_\tau(i)$ and $\underline{m}_\alpha(i)$ satisfying (2). We assume that $\mathcal{T} = \{I, II\}$ and $\mathcal{C} = \{A, B\}$ and there will be three kinds of switchings in the phase: IA (type I, class A), converting an element in $\mathcal{S}_{j+1}$ to an element in $\mathcal{S}_j$; IB (type I, class B), maintaining the same stratum; and IIB (type II, class B), converting from $\mathcal{S}_{j-1}$ to $\mathcal{S}_j$. See Figure 1 for an illustration of these transitions. If j is such that $\mathcal{S}_{j-1}$ or $\mathcal{S}_{j+1}$ does not exist in Φ_1, the corresponding stratum is omitted from the diagram. Additionally, there is no IIB switching from $\mathcal{S}_0$ to $\mathcal{S}_1$, and the Markov chain always transits from an element in $\mathcal{S}_0$ to F in the next step.

From Figure 1 and (6) with $\alpha =$ B in the two cases $i = j$ and $i = j-1$, the expected number of times that a given Class B switching produces $P \in \mathcal{S}_j$ (including the time it is b-rejected, in part (iii) of the switching step, if that occurs) is

$$q_B(j) = \frac{\sigma(j)\rho_I(j)}{\overline{m}_I(j)} = \frac{\sigma(j-1)\rho_{II}(j-1)}{\overline{m}_{II}(j-1)} \qquad (1 \le j \le i_1), \tag{8}$$

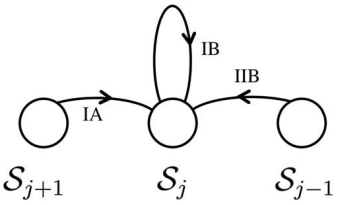

Figure 1. *Transitions into $\mathcal{S}_j$*

and at times we will use either of these quantities in place of $q_B(j)$. For similar reasons,

$$q_A(j) = \sigma(j+1)\rho_I(j+1)/\overline{m}_I(j+1) \quad (0 \le j \le i_1 - 1). \tag{9}$$

It is convenient to set

$$x_j = \sigma(j)|\Phi_1|. \tag{10}$$

Substitution of the two equations above into (7) gives

$$x_j = 1 + \frac{x_{j+1}\rho_I(j+1)}{\overline{m}_I(j+1)} \cdot \underline{m}_A(j) + \frac{x_j\rho_I(j)}{\underline{m}_I(j)} \cdot \underline{m}_B(j) \quad (1 \le j \le i_1 - 1). \tag{11}$$

We need separate equations for $j \in \{0, i_1\}$ as (8) does not hold for $j = 0$ and (9) does not hold for $j = i_1$. No $P \in \mathcal{S}_0$ can be reached by any class B switching, because whenever any element in $\mathcal{S}_0$ is reached, the Markov chain proceeds to state F in the very next step. Therefore, $q_B(0) = 0$. Similarly, no element $P \in \mathcal{S}_{i_1}$ can be reached via a type A switching because Φ_1 does not contain $\mathcal{S}_{i_1+1}$. Therefore, $q_A(i_1) = 0$. So for $j = i_1$ or 0 we have in place of (11)

$$x_{i_1} = 1 + \frac{x_{i_1}\rho_I(i_1)}{\overline{m}_I(i_1)} \cdot \underline{m}_B(i_1), \qquad x_0 = 1 + \frac{x_1\rho_I(1)}{\overline{m}_I(1)} \cdot \underline{m}_A(0). \tag{12}$$

Moreover, the second equality in (8) implies

$$\rho_{II}(j) = \rho_I(j+1)\frac{x_{j+1}}{x_j} \cdot \frac{\overline{m}_{II}(j)}{\overline{m}_I(j+1)} \quad \text{for all } 0 \le j \le i_1 - 1. \tag{13}$$

As boundary cases, we require

$$\rho_I(0) = \rho_{II}(0) = 0, \quad \rho_{II}(i_1) = 0, \tag{14}$$

the first two equalities because every element in $\mathcal{S}_0$ is forced to transit to F once it is reached, and the last because Φ_1 does not contain $\mathcal{S}_{i_1+1}$. The four equations above determine a system that will be required to have a solution, which we also demand to satisfy

$$\rho_I(i) + \rho_{II}(i) \le 1, \quad \rho_I(i),\ \rho_{II}(i) \ge 0 \quad \text{for all } 0 \le i \le i_1. \tag{15}$$

This condition ensures that $\rho_\tau(i)$ satisfy (1), as required if they are to be used as probabilities. Naturally, it needs to be checked in any particular case that a solution of the desired type exists.

For any solution $(\rho^*_\tau(j), x^*_j)$ of the system (11)–(15), we may set $\rho_\tau(j)$ equal to $\rho^*_\tau(j)$ for every $0 \le j \le i_1$ and each $\tau \in \{I, II\}$. Then $(\sigma(j), q_\alpha(j))_{0 \le j \le i_1}$ for $\alpha \in \{A, B\}$ can be computed using x^*_j and (10), (8) and (9). Thus, $(\sigma(j), q_\alpha(j))_{0 \le j \le i_1}$ is a solution to equations (6) and (7), and so by Lemma 4, the expected number of visits to any given element in any stratum $\mathcal{S}_i$ is $\sigma(i)$. Since every element in $\mathcal{S}_0$ is reached at most once, $\sigma(0)$ is equal to the probability that a given $P \in \mathcal{S}_0$ is reached, the same for all such P. That is, we may conclude the following.

Lemma 5. *Assume that $(\rho^*_\tau(i), x^*_i)$ is a solution to system (11)–(15). Set $\rho_\tau(i)$ to be $\rho^*_\tau(i)$ for every $1 \le i \le i_1$ as the type probabilities in the phase. Assume that no rejection occurs during the algorithm. Then the last element visited in the phase is distributed uniformly at random in $\mathcal{S}_0$.*

VI. PROCEDURE *REG*

We now turn to giving the explicit construction of the algorithm *REG*, and its analysis. We begin with Phase 3, which is the most interesting phase and has been partially described in Section V. The first two phases are defined at the end of this section. The types and classes of switchings, and the transitions between $(\mathcal{S}_i)_{0\le i\le i_1}$ caused by performing them, are just as described in Section V, which led to system (11)–(15). We will specify Φ_1, $\mathcal{S}_i$ and i_1, and the set of switchings for this phase. We will analyse these switchings to obtain appropriate parameters $\overline{m}_\tau(i)$ and $\underline{m}_\alpha(i)$ as required for the system (11)–(15). After that, we will specify values for the variables $\rho_I(i)$ and show that, given these, the whole system has a solution. Then, we bound the probability that phase 3, once begun, terminates in a rejection.

Specifying Φ_1, $\mathcal{S}_i$ and i_1 for phase 3

As noted before, each phase has uniform output distribution. Define Φ_1 by

$$\Phi_1 = \{P \in \Phi :\ P \text{ has at most } i_1 \text{ double edges and no other types of multi-edges or loops}\} \tag{16}$$

where $i_1 = d^2$, and let $\mathcal{S}_i$ be the set of pairings in Φ_1 having exactly i double edges. (The long version of this paper sets i_1 more flexibly, to reduce time complexity for specific values on n and d.) Here, we may assume that the algorithm enters phase 3 with a pairing P_0 uniformly distributed in Φ_1. We will verify this assumption in Lemma 9 below.

Defining the switchings for phase 3

Both types, I and II, of switching are versions of the same basic switching operation, which we call a *d-switching*, defined as follows, to act upon a pairing P that contains at least one double edge. As in Figure 2, pick a double edge with parallel pairs $\{1,2\}$ and $\{3,4\}$, with points 1 and 3 in the same vertex u_1 and points 2 and 4 in the same vertex v_2. Also pick another two pairs $\{5,6\}$ and $\{7,8\}$, that are single edges, and let u_2, v_2, u_3 and v_3 denote the vertices containing the points 5, 6, 7 and 8 respectively. If all six vertices u_i, v_i, $1 \le i \le 3$, are distinct, then a d-switching replaces the pairs {1,2}, {3,4}, {5,6} and {7,8} by {1,5}, {3,7}, {2,6} and {4,8}, producing the situation in the right side of Figure 2.

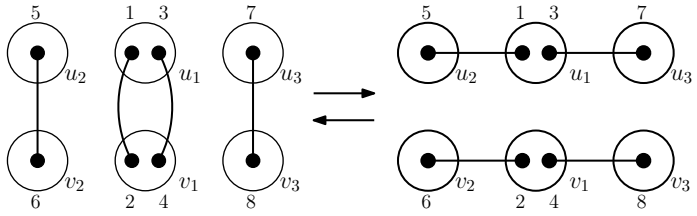

Figure 2. *A type I switching*

The specification of the labels in the such switching definitions is significant when counting the ways that a switching can be applied to a pairing: different labellings give different switchings.

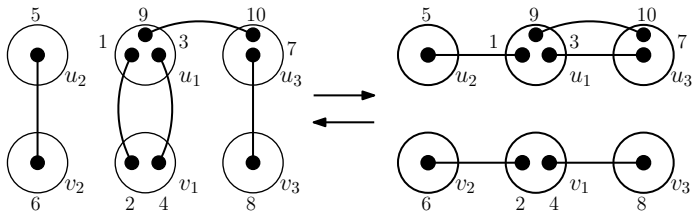

Figure 3. *A type I switching of class B*

Type I switching. If the d-switching operation does not create more than one new double edge, it is a valid type I switching. The *class* of a type I switching depends on how many new double edges it creates. If none, it is in class A, whilst if there is one, then it is in class B (as in Figure 3). Note that the class of a switching is not chosen; when

a random type I switching is chosen in part (ii) of a switching step, the class is determined after the switching is chosen. For a type I class B switching, there was an existing pair between u_1 and u_i or between v_1 and v_i ($i = 2, 3$) before the d-switching was applied. For purposes of counting type I class B switchings, the existing pair is then labelled $\{9, 10\}$, where 9 is in u_1 (or v_1 as the case may be). Of course, this pair remains after the switching is applied. See Figure 3 for an example.

Type II switching. If exactly two new double edges, both incident with u_1 or both incident with v_1, are created by performing a d-switching, then it is a valid type II switching. For counting purposes, the existing pairs that become part of double edges are labelled $\{9, 10\}$ and $\{11, 12\}$, with 9 and 12 both in u_1 or both in v_1. (See Figure 4.) A type II switching is always in class B. Note that the choice of placing point 10 in u_2 or in u_3, say, leads to different type II switchings arising from the same d-switching. See Figures 4 and 5 for an example.

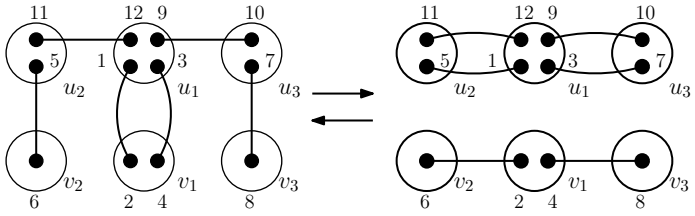

Figure 4. *A type II switching*

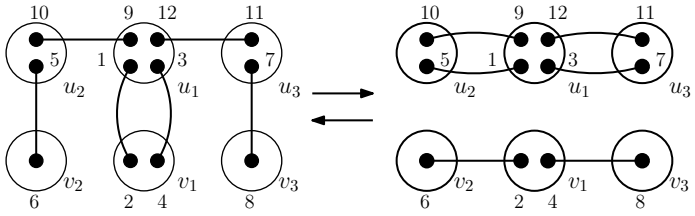

Figure 5. *A different type II switching*

Specifying $\overline{m}_\tau(i)$ and $\underline{m}_\alpha(i)$

For any integer $k \geq 1$, define $M_k = d(d-1)\cdots(d-k+1)n$. Define

$$\begin{array}{lcllcl} \overline{m}_I(i) & = & 4i(M_1 - 4i)^2 & \underline{m}_A(i) & = & M_2^2 - M_2(d-1)(16i + 3d^2 + d + 6) \\ \overline{m}_{II}(i) & = & 16i(d-1)^2(d-2)(d-3) & \underline{m}_B(i) & = & 16i(d-2)M_2 - 16id(8id + 9d^2 + 3d^3). \end{array} \tag{17}$$

By simple counting arguments (details omitted), we see that these parameters satisfy (2).

Fixing $\rho_\tau(i)$

With the definition of the three kinds of switchings (IA, IB, IIB) above, the transitions among strata $\mathcal{S}_j$ in the Markov chain is exactly as in the example in Section V.

Lemma 6. *Let $0 < \epsilon = O(d^2/n^2)$. Assume that $d = o(\sqrt{n})$. Given the values of $\overline{m}_\tau(i)$ and $\underline{m}_\alpha(i)$ in (17), the system (11)–(15) has a unique solution $(\rho^*_\tau(i), x^*_i)$ satisfying $\rho^*_I(i) = 1 - \epsilon$ for every $1 \leq i \leq i_1$, for n sufficiently large.*

Proof. Since $\rho^*_I(i)$ is given for each i, (12) determines $x^*_{i_1}$. Recursively, the other x^*_i are determined by (11) and (12). From here, each $\rho^*_{II}(i)$ is determined using (13). It is straightforward to verify that $(\rho^*_\tau(i), x^*_i)$ satisfies (15) for large n, and thus is a solution to (11)–(15). ∎ ∎

Note that the proof of Lemma 6 also provides a computation scheme for a solution to system (11)–(15), which runs in $O(i_1) = O(d^2)$ time. Let $(\rho^*_\tau(i), x^*_i)$ be the solution specified in Lemma 6. (If no solution, we artificially reset $i_1 = 0$ so that the algorithm is still well defined, and this phase will be empty. By the lemma, this can only

happen for small n, so our complexity estimates are unaffected.) As described in Section V, we set $\rho_\tau(i)$ to be $\rho^*_\tau(i)$ for phase 3.

Probability of rejection

By Lemma 6, $\rho_I(i) = 1-O(d^2/n^2)$. It is easy to see that conditional on type I being chosen for a given switching step, the probability that the random switching chosen is in class A is $1 - O(d/n)$. Thus, for every switching step, the number of double edges decreases by one with probability $1 - O(d/n)$. This intuition immediately leads to the following, and then we consider t-rejection.

Lemma 7. *The number of switching steps in phase 3 is $O(i_1) = O(d^2)$ in expectation.*

Lemma 8. *During phase 3, the probability of a t-rejection is $O(d^4/n^2)$. Moreover, assuming $d = o(\sqrt{n})$, the probability of an f- or b-rejection is $O(d^2/n)$.*

Proof. (Much of this proof is only sketched.) In each switching step of phase 3, t-rejection occurs with probability at most $\max_{1\le i\le i_1}(1 - \rho_I(i) - \rho_{II}(i)) = O(\epsilon) = O(d^2/n^2)$, so the first claim follows from Lemma 7.

Consider f-rejection when $d = o(\sqrt{n})$. Conditional upon a given pairing P being obtained after the t-th switching step, the probability that f-rejection occurs in the next switching step is

$$\rho_I(i)\big(1 - f_I(P)/\overline{m}_I(i)\big) + O(d^2/n^2),$$

where the first term is for type I and the second for type II. The expected number of times that P is reached is $\sigma(i)$ for every $P \in \mathcal{S}_i$, and also $\rho_I(i) \le 1$. Hence the total probability of f-rejection is at most

$$\sum_{1\le i\le i_1} \sum_{P\in\mathcal{S}_i} \sigma(i) \left(1 - \frac{f_I(P)}{\overline{m}_I(i)}\right) + \sum_{1\le i\le i_1} |\mathcal{S}_i|\sigma(i) \cdot O(d^2/n^2). \tag{18}$$

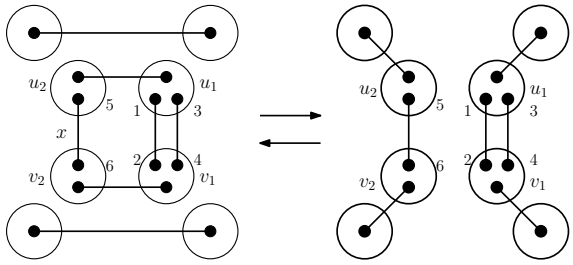

Figure 6. *A subsidiary switching for u_1u_2 and v_1v_2 being single edges*

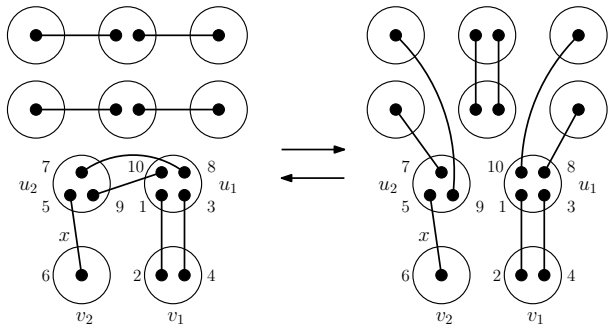

Figure 7. *Another subsidiary switching for u_1u_2 being a double edge*

The second summation is $O(d^4/n^2) = o(d^2)$ since by Lemma 7, $\sum_{1\le i\le i_1} |\mathcal{S}_i|\sigma(i) = O(d^2)$. Hence, using $i_1 = O(d^2)$, it suffices to show that $\mathbb{E}\,(1 - f_I(P)/\overline{m}_I(i)) = O(1/n)$ for every $1 \le i \le i_1$, where $\mathbb{E}$ denotes a *uniformly* random $P \in \mathcal{S}_i$. To evaluate such an expectation, we argue using subsidiary switchings (see Figures 6

and 7) to estimate the probability of events such as pairs being present in certain locations after two random 2-paths in P are chosen. This averaging argument resembles the enumeration arguments in [13, Lemma 4.4], and we omit the details, noting only that it produces the bound $O(d^2/n^2) = O(1/n)$, as required. ∎

Initial rejection and the first two phases

The set of pairings $\mathcal{A}$ is defined as in [12], except that now, triple edges are permitted, but only up to d^3/n of them. The first phase eliminates loops, and the second eliminates triple edges. In each of the first two phases, exactly one type and one class of switchings are used and they are the same switchings as used in [13] (see Figures 8 and 9). To deal with the first two phases and verify the assumptions following (16), we have the following lemma. The proof is by analysis similar to that in [12], so is omitted.

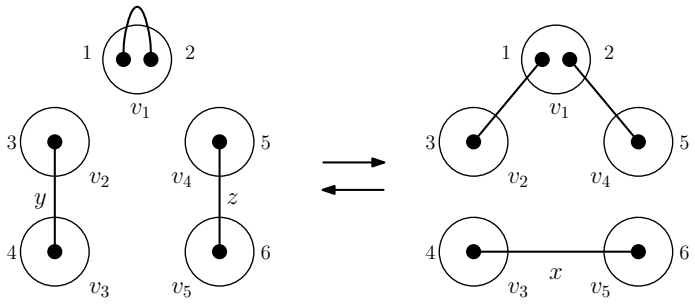

Figure 8. *switching for loops*

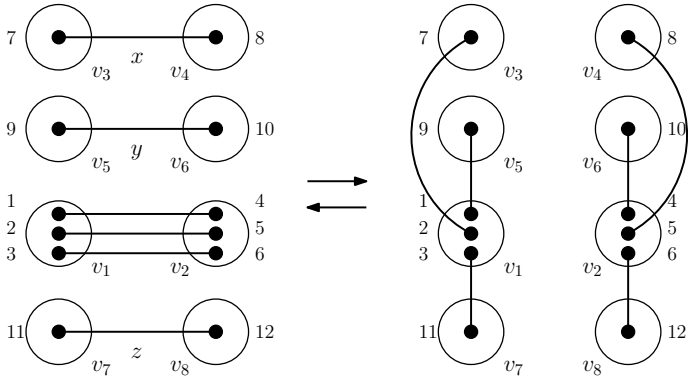

Figure 9. *switching for triple edges*

Lemma 9. *With probability at least* $1/8+o(1)$*, no rejection occurs initially or during the first two phases. Moreover, the output of the second phase, if no rejection has occured, is uniformly distributed in* Φ_1 *defined in (16).*

VII. *REG** AND PROOFS OF THE MAIN THEOREMS

Proof of Theorem 1. The uniformity of the output of *REG* follows by Lemmas 9 and 5, since the parameters $\rho_\tau(i)$ are set for *REG*, just after Lemma 6, using a solution $(\rho^*_\tau(i), x^*_i)$ of the system (11)–(15). ∎

Next, we define *REG**. This starts by repeatedly generating a random pairing $P \in \Phi$ until $P \in \mathcal{A}$. Then there are three phases for reduction of loops, triple edges and double edges. In each phase, there will be only one type of switching: types L and T for phases 1 and 2 respectively and type I for phase 3. Each phase consists of a sequence of switching steps in which a random switching is chosen and performed. In other words, *REG** is the obvious simplification of *REG* got by omitting f or b-rejections in each phase, and by discarding type II switchings in phase 3.

Proof of Theorem 3. We can couple *REG* and *REG** so that they follow the same track unless a rejection or type II switching occurs in *REG*. This event has probability $o(1)$ by Lemmas 8 and 6 respectively in phase 3, and similarly for the other phases. Hence, with probability $1 - o(1)$, *REG** and *REG* have the same output, having uniform distribution. The first part of the theorem follows.

Uniformly generating a pairing P takes $O(dn)$ steps, and by Lemma 9 this is repeated $O(1)$ times to get $P \in \mathcal{A}$. The total number of switching steps in all phases is bounded by $O(B_L + B_D + B_T) = O(d^2)$ in expectation, and a switching step can easily be done in $O(1)$ expected time. Thus, the total time complexity of *REG** is $O(dn + d^2) = O(dn)$ in expectation. ∎

Proof of Theorem 2. (Sketch.) As described in Sections III and V, *REG* requires solving (11)–(15) and also computing $f_\tau(P)$ and $b_\alpha(P)$. The former takes $O(d^2)$ steps, as remarked below Lemma 6. There is an easy way to avoid computing $f_\tau(P)$, as it is only needed to compute the f-rejection probability at substep (iii) of a switching step. For instance, take type I switching in phase 3. Randomly pick a double edge, and then two pairs with replacement, from the remaining $M_1/2 - 2i$ pairs. The number of choices, including point labelling, is exactly $\overline{m}_I(i)$. Perform the switching if it is valid, and perform an f-rejection otherwise, which therefore occurs with probability $1 - f_I(P)/\overline{m}_I(i)$ as required. The other cases are similar, so we omit the details.

Computing $b_\alpha(P)$ in phase 3 involves counting the pairs of 2-paths, as on the right of Figure 2, satisfying certain constraints. This is done in $O(nd^3)$ time in [12] by inclusion-exclusion, which requires keeping track of the number of 3-paths between any pair of points, or any pair of vertices, and similar information. This can be done in time $O(nd^3)$ initially, and updated in $O(d^2)$ per switching using appropriate data structures. (We expect $O(d^2)$ switchings.) The same approach easily extends to count pairs of 3-stars as on the right of Figure 9. ∎

REFERENCES

[1] A. Békéssy, P. Békéssy and J. Komlós, Asymptotic enumeration of regular matrices. *Studia Sci. Math. Hungar.* **7** (1972), 343–353.

[2] M. Bayati, J.H. Kim, and A. Saberi. A sequential algorithm for generating random graphs. *Algorithmica* **58** (2010), 860–910.

[3] E. A. Bender and E. R. Canfield. The asymptotic number of labeled graphs with given degree sequences. *J. Combinatorial Theory Ser. A* **24** (1978), 296–307.

[4] B. Bollobás. A probabilistic proof of an asymptotic formula for the number of labelled regular graphs. *European J. Combin.* **1** (1980), 311–316.

[5] C. Cooper, M. Dyer, and C. Greenhill. Sampling regular graphs and a peer-to-peer network. *Combin. Probab. Comput.* **16** (2007), 557–593; Corrigendum: arXiv:1203.6111.

[6] C. Greenhill. The switch markov chain for sampling irregular graphs. *Proc. SODA'15 (accepted).*

[7] C. Greenhill and B. McKay, Asymptotic enumeration of sparse multigraphs with given degrees, *SIAM J. Discrete Math.* **27** (2013), 2064–2089.

[8] S. Janson. The probability that a random multigraph is simple. *Combin. Probab. Comput.* **18** (2009), 205–225.

[9] M. Jerrum and A. Sinclair. Fast uniform generation of regular graphs. *Theoret. Comput. Sci.* **73** (1990), 91–100.

[10] R. Kannan, P. Tetali and S. Vempala. Simple Markov-chain algorithms for generating bipartite graphs and tournaments. *Random Structures Algorithms* **14** (1999), 293–308.

[11] J.H. Kim and V.H. Vu. Generating random regular graphs. *Combinatorica* **26** (2006), 683–708.

[12] B.D. McKay and N.C. Wormald. Uniform generation of random regular graphs of moderate degree. *J. Algorithms* **11** (1990), 52–67.

[13] B.D. McKay and N.C. Wormald, Asymptotic enumeration by degree sequence of graphs with degrees $o(\sqrt{n})$, *Combinatorica* **11** (1991), 369–382.

[14] B.D. McKay and N.C. Wormald, Uniform generation of random Latin rectangles, *J. Combin. Math. Combin. Comput.* **9** (1991), 179–186.

[15] A. Steger and N.C. Wormald. Generating random regular graphs quickly. *Combin. Probab. Comput.* **8** (1999), 377–396.

[16] G. Tinhofer. On the generation of random graphs with given properties and known distribution. *Appl. Comput. Sci., Ber. Prakt. Inf* **13** (1979), 265–297.

[17] N.C. Wormald. Generating random regular graphs. *J. Algorithms* **5** (1984), 247–280.

[18] J.Y. Zhao, Expand and Contract: Sampling graphs with given degrees and other combinatorial families, arXiv:1308.6627.