Research Report 306

Randomly Sampling Molecules

Leslie Goldberg
Mark Jerrum

RR306

We give the first polynomial-time algorithm for the following problem: Given a degree sequence in which each degree is bounded from above by a constant, select, uniformly at random, an unlabelled connected multigraph with the given degree sequence. We also give the first polynomial-time algorithm for the following related problem: Given a molecular formula, select, uniformly at random, a structural isomer having the given formula.

Department of Computer Science
University of Warwick
Coventry CV4 7AL
United Kingdom

June 1996

Randomly Sampling Molecules[*]

Leslie Ann Goldberg[†] Mark Jerrum[‡]

June 17, 1996

Abstract

We give the first polynomial-time algorithm for the following problem: Given a degree sequence in which each degree is bounded from above by a constant, select, uniformly at random, an unlabelled connected multigraph with the given degree sequence. We also give the first polynomial-time algorithm for the following related problem: Given a molecular formula, select, uniformly at random, a structural isomer having the given formula.

1 Introduction

In this paper, we give a polynomial-time algorithm for the following problem: Given a degree sequence in which each degree is bounded from above by a constant, select, uniformly at random, an unlabelled connected multigraph with the given degree sequence. We also give a polynomial-time algorithm for the following related problem: Given an (empirical) molecular formula, select, uniformly at random, a *structural isomer* having the given formula. An empirical molecular formula [15] simply gives the number of atoms of each kind that occur in a molecule. A *structural formula* [14] is a method of representing the way in which the atoms in a molecule are linked together. A *structural isomer* is a structural formula, viewed as an unlabelled multigraph in which the vertices are of several different kinds. Some of the structural isomers corresponding to a given empirical formula are chemically irrelevant due to geometric (and other) constraints. Nevertheless, counting all of the structural isomers corresponding to a given empirical formula is a long-standing open problem for which no practical general solution has been

[*] Research Report RR 306, Department of Computer Science, University of Warwick, Coventry CV4 7AL, United Kingdom. This work was partially supported by ESPRIT Project no. 21726 — RAND-II

[†] `leslie@dcs.warwick.ac.uk`. Department of Computer Science, University of Warwick, Coventry CV4 7AL, United Kingdom. This work was partially supported by ESPRIT LTR Project no. 20244 — ALCOM-IT

[‡] `mrj@dcs.ed.ac.uk`. Department of Computer Science, University of Edinburgh, Edinburgh EH9 3JZ, United Kingdom.

found [15]. Solutions do exist for certain restricted clases of chemical compounds [13, 14, 15]. Faulon [7] has developed an algorithm for *listing* all of the structural isomers corresponding to a given empirical formula. This algorithm is useful if the number of structural isomers corresponding to the relevant formula is sufficiently small. Faulon has argued [9] that randomly sampling structural isomers is useful for structural elucidation and molecular design in cases in which the number of isomers is too large to list them all. He [8] has developed a program for randomly sampling structural isomers and has used it for chemical applications such as a statistical study of the potential energy distribution of the isomers of C_8H_{10} and the structural elucidation of several compounds. Faulon's program applies to a realistic chemical problem including 3-D simulation of molecules and chemical analysis. However, his methods are heuristic. By contrast, we study an idealisation of the problem (randomly sampling structural isomers without regard to geometric and other chemical constraints) but we achieve rigorous performance guarantees — polynomial-time computation and exactly uniform generation. Thus, we describe the first polynomial-time algorithm that uniformly samples structural isomers given an empirical formula. Our isomer-sampling algorithm is based on our algorithm for uniformly sampling unlabelled connected multigraphs with a given degree sequence.

Previous work: Nijenhuis and Wilf [12] showed how to uniformly sample unlabelled rooted trees with a specified number of vertices. This approach was extended by Wilf [18], who showed how to uniformly sample free (unrooted) trees. Their algorithms are based on an inductive definition (i.e., a generating function) for the trees. This approach has been systematised by Flajolet, Zimmerman and Van Cutsem in an forthcoming paper [11]. More complicated techniques are required when the graphs to be sampled are not trees. Dixon and Wilf [6] were the first to give an algorithm for uniformly sampling unlabelled graphs with a specified number, n, of vertices. Their algorithm is based on Burnside's Lemma. First, a permutation of the n vertices is chosen with the appropriate probability and then a graph is chosen uniformly at random from those graphs which are fixed by the chosen permutation. The choice of the permutation requires a calculation of the number of unlabelled graphs with n vertices. Wormald's algorithm [21] avoids doing this expensive calculation. Instead, it achieves a uniform distribution by restarting itself when appropriate. Wormald's method can also be used to sample r-regular graphs uniformly at random for any fixed degree $r \geq 3$. The method relies on the fact that most unlabelled r-regular graphs are rigid (without non-trivial symmetries) when $r \geq 3$. This is not true for $r = 1$ or $r = 2$.

Outline of our algorithm Our algorithm for sampling unlabelled connected multigraphs with a given degree sequence combines the above ideas with other ideas from the field of random graphs. A natural approach to the problem is the approach of Wormald — first generate a permutation of the vertices, then generate a random connected multigraph fixed by the permutation, and finally use rejection/restarting to obtain the correct distribution. However, this approach relies heavily on the fact that many of the desired structures are rigid (so the algorithm will be likely to choose the identity permutation, which leads to a quick result without restarting). This is not the case for the set of unlabelled connected multigraphs with a given degree sequence, because the degree sequence may have many vertices of degree 1 and 2. Thus, we first reduce our problem to that of sampling unlabelled connected multigraphs with degree sequences that do not have any vertices of degree 1 or 2. Every multigraph G is associated with a unique "core" which has no vertices of degree 1 or 2. To generate G, we will generate the core of G and we will then extend the core by adding trees and chains of trees to obtain G.

For the generation of the core, we work in the configuration model of Bender and Canfield [1] and Bollobás [3]; a similar model was proposed independently by Wormald [19]. The correctness of our algorithm follows from a careful analysis of unlabelled configurations in which all block sizes are at least 3. This analysis extends Bollobás's analysis of unlabelled regular graphs [2]. Our algorithm rejects the generated core if it is not connected. The fact that this does not happen too often follows from a result of Wormald [20]. After generating the core of our random multigraph, we extend the core by adding trees and chains of trees. This part of our algorithm is based on the generating function approach mentioned earlier. An alternative approach, also based on generating functions, is to use Pólya's theorem. This approach was used to enumerate molecules with certain specified "frames" (a frame is somewhat similar to a core) by Pólya, Read and others [13].

Outline of this paper Section 2 sets up the machinery that we will use to reduce the general multigraph problem to the problem in which the degree sequence has no vertices of degree 1 or 2. Section 3 solves the problem when there are no vertices of degree 1 or 2. Section 4 describes the tools that we will use to lift the solution from Section 3 to a solution for general degree sequences. Section 5 gives our sampling algorithm and proves that it is correct. Section 6 extends our result to the chemical problem — given a molecular formula, select, uniformly at random, a structural isomer having the given formula.

2 Cores and coloured configurations

A d-rooted n-vertex *multigraph* is a tuple $G = (V_n, E, r_0, \dots, r_{d-1})$, in which $V_n = \{v_1, \dots, v_n\}$ is the *vertex set* of G, E is the *edge multiset* of G, and $r_0, \dots, r_{d-1}$ are distinct roots in V_n. Each element of E is an unordered pair of vertices. The expression $E(v, w)$ denotes the multiplicity of (v, w) in E. A *cycle* of G is a (closed, simple) path from a vertex v to itself that uses each edge (x, y) at most $E(x, y)$ times. (If any edge is used twice then the path is in fact of length two.) We use the term *rooted multigraph* to refer to any d-rooted multigraph (for any d, including $d = 0$) and we use the term *multigraph* to refer to any 0-rooted multigraph. A rooted *tree* is a connected rooted multigraph (in fact a graph) with no cycles. The definitions imply that a connected unicyclic multigraph is either a connected unicyclic graph, or a multigraph obtained from a tree by doubling one of its edges.

The *degree* of vertex v in a rooted multigraph $G = (V_n, E)$ is

$$d(v) = 2\,E(v,v) + \sum_{w \in V, w \neq v} E(v,w).$$

Let Δ be any fixed constant. In this paper we will be concerned with rooted multigraphs whose vertices have degree at most Δ. The *degree sequence* of such a rooted multigraph G is the sequence $\mathbf{n} = n_0, \ldots, n_\Delta$, where n_i denotes the number of vertices of G with degree i. The integers $n_0, \ldots, n_\Delta$ are represented in *unary,* so the input size of the degree sequence $\mathbf{n} = n_0, \ldots, n_\Delta$ is $n = n_0 + \cdots + n_\Delta$. If $\mathbf{n}'$ and $\mathbf{n}$ are degree sequences, then we write $\mathbf{n}' \leq \mathbf{n}$ to indicate that $n'_i \leq n_i$ for every $i \in [0, \ldots, \Delta]$ and, we write $\mathbf{n}' < \mathbf{n}$ to indicate that $\mathbf{n}' \leq \mathbf{n}$ and $\mathbf{n}' \neq \mathbf{n}$.

Let $\mathcal{G}_n$ be the set containing

- Every connected multigraph with degree sequence $\mathbf{n}$ that has at least two cycles, and
- every 1-rooted connected tree with degree sequence $\mathbf{n}$, and
- every 1-rooted connected unicyclic multigraph with degree sequence $\mathbf{n}$, in which the root is part of the cycle.

d-rooted multigraphs $G = (V, E, r_0, \ldots, r_{d-1})$ and $G' = (V, E', r'_0, \ldots, r'_{d-1})$ are said to be *isomorphic* (written $G \cong G'$) if there is a permutation π (an *isomorphism*) of the vertices in V such that, for all unordered pairs (v, w) of vertices in V, $E(v,w) = E'(\pi(v), \pi(w))$, and for all roots r_j of G, $\pi(r_j) = r'_j$. Isomorphism is an equivalence relation and the equivalence classes are called isomorphism classes. We use the notation $\Psi(G)$ to denote the isomorphism class of rooted multigraph G and the notation $\Psi^{-1}(U)$ to denote the lexicographically least rooted multigraph in isomorphism class U. If $G_1 \cong G_2$ then G_1 and G_2 have the same number of roots and the same degree sequence, and either both of them are connected or neither of them is connected. We use the notation $\widetilde{\mathcal{G}}_n$ to denote the set of isomorphism classes of $\mathcal{G}_n$.

We consider two non-deterministic transformations which may be applied to a rooted multigraph $G \in \mathcal{G}_n$ with vertex set V and edge multiset E. Similar transformations were used by Zhan in [22].

T_1: Choose a degree-1 vertex v other than the root of G. Remove v from V and the edge containing v from E.

T_2: If T_1 cannot be applied to G, choose a degree-2 vertex v other than the root of G such that for vertices $w \neq v$ and $x \neq v$, (v, w) and (v, x) are in E. (We allow $w = x$, but naturally insist that (v, w) and (v, x) are taken to be distinct elements from the edge multiset.) Remove v from V. Remove (v, w) and (v, x) from E and add (w, x) to E.

A rooted multigraph $G \in \mathcal{G}_n$ is *irreducible* if neither transformation T_1 nor T_2 can be applied to it.

Observation 2.1 *If* $G \in \mathcal{G}_n$ *and* G *can be transformed into* G' *by* T_1 *or* T_2 *then, for some* $n' < n$, $G' \in \mathcal{G}_{n'}$.

Informally, the transformations T_1 and T_2 preserve the properties of being connected, and of having at least two cycles.

Observation 2.2 *If* $G \in \mathcal{G}_n$ *and some sequence of* T_1 *and* T_2 *transforms* G *into* G' *then the sequence is of length less than* n.

We say that a degree sequence $\mathbf{n}$ is *irreducible* if any of the following applies, and that it is *degenerate* if one of the first two possibilities applies.

1. $\mathbf{n}$ describes the single-vertex multigraph. That is, $n_0 = 1$ and $n_i = 0$ for $i \neq 0$.
2. $\mathbf{n}$ describes the single-self-loop multigraph. That is, $n_2 = 1$ and $n_i = 0$ for $i \neq 2$.
3. $\mathbf{n}$ describes multigraphs without low-degree vertices. That is, $n_0 = n_1 = n_2 = 0$ and $n_i > 0$ for some $i \in [3, \ldots, \Delta]$.

We say that a rooted multigraph is *degenerate* if its degree sequence is degenerate.

Observation 2.3 $G \in \mathcal{G}_n$ *is irreducible iff its degree sequence is irreducible.*

Lemma 2.4 *If* $G \in \mathcal{G}_n$ *and* G *can be transformed into irreducible rooted multigraphs* G_1 *and* G_2 *using a sequence of transformations* T_1 *and* T_2 *then* $G_1 = G_2$.

Proof: Suppose G has vertex set V and edge multiset E. We will show that if G can be transformed into distinct rooted multigraphs G_1 and G_2 by a single transformation then there is a rooted multigraph G_3 such that a (possibly empty) sequence of transformations transforms G_1 into G_3 and another (possibly empty) sequence of transformations transforms G_2 into G_3. Thus, the transformation process is *locally confluent* [17]. As the process terminates in finite time (see Observation 2.2), it is *confluent,* which implies the result [17].

Suppose that T_1 with choice v transforms G into G_1 and T_1 with choice $w \neq v$ transforms G into G_2. Note that $(v, w) \notin E$ (otherwise, one of v and w would be the root of G). Let G_3 be the result of applying T_1 to G_1 with choice w. Then T_1 with choice v transforms G_2 into G_3.

Suppose that T_2 with choice v transforms G into G_1 and T_2 with choice $w \neq v$ transforms G into G_2. Note that (v, w) does not appear twice in E (otherwise, one of v and w would be the root of G). Let G_3 be the result of applying T_2 to G_1 with choice w. Then T_2 with choice v transforms G_2 into G_3. □

Note that Lemma 2.4 fails if trees and unicyclic graphs are unrooted, and it is for this reason that the definition of $\mathcal{G}_n$ is slightly more complicated that might be expected. As we observed in Observation 2.2, a rooted multigraph G can only be transformed a finite number of times before an irreducible rooted multigraph G' is reached. G' is called the *core* of G and is denoted by $A(G)$. $A(G)$ is uniquely defined, by Lemma 2.4.

Lemma 2.5 *If* G_1 *and* G_2 *are in* $\mathcal{G}_n$ *and* $\pi(G_1) = G_2$ *then* $\pi(A(G_1)) = A(G_2)$.

Proof: Consider a sequence of transformations that transforms G_1 into $A(G_1)$. Now apply this sequence of transformations to G_2, but choose $\pi(v)$ instead of v for each vertex v that is chosen. Clearly, the result is $A(G_2)$. Thus, $v \notin A(G_1)$ exactly when $\pi(v) \notin A(G_2)$. □

Lemma 2.5 implies that if $G_1 \cong G_2$ then $A(G_1) \cong A(G_2)$. (This property was also used by Zhan [22].) We use the notation $\widetilde{\mathcal{G}}_{n,n'}$ to denote the set $\{U \in \widetilde{\mathcal{G}}_n \mid \Psi(A(\Psi^{-1}(U))) \in \widetilde{\mathcal{G}}_{n'}\}$; loosely, $\widetilde{\mathcal{G}}_{n,n'}$ is the set of unlabelled connected multigraphs with degree sequence n whose cores have degree sequence n'. We will need the following definitions. Let $\mathcal{B}$ be an (infinite) set containing one representative from each isomorphism class of the set of 1-rooted trees. A *tree-chain* with two roots is constructed from any sequence $T_1, \ldots, T_k$ of 1-rooted trees as follows: If the sequence is empty, then the tree-chain consists of r_0 and r_1 and an edge between them. Otherwise, the tree-chain graph is constructed as follows: Choose distinct labels for the vertices of $T_1, \ldots, T_k$. Let $r'_1, \ldots, r'_k$ be the roots of $T_1, \ldots, T_k$. For $i \in [1, \ldots, k-1]$, add edge (r'_i, r'_{i+1}). Add the new roots r_0 and r_1 and edges (r_0, r'_1) and (r'_k, r_1). For every tree-chain G, we use the notation $R(G)$ to denote the tree-chain constructed from G by swapping r_0 and r_1. Let $\mathcal{P}$ be a set containing one representative from each isomorphism class of tree-chains. (Note that the two roots of a tree-chain are distinguishible, and any isomorphism of tree-chains must respect this distinction.)

A *colouring* of a rooted multigraph G is a function λ that maps each vertex in the vertex set of G to an element of $\mathcal{B}$ and each edge in the edge multiset of G to an element of $\mathcal{P}$.

We will describe a function Γ that maps each coloured rooted multigraph (G, λ) to an isomorphism class. $\Gamma(G, \lambda)$ is constructed as follows, where V denotes the vertex set of G and E denotes the edge multiset of G: Start with the collection of rooted trees $\{\lambda(v) \mid v \in V\} \cup \{\lambda(e) \mid e \in E\}$. Let the roots of the resulting forest be the roots of those trees that correspond to the roots of G. For each edge $(u, w) \in E$ with $u \leq w$, identify root r_0 of $\lambda(u, w)$ with the root of the tree $\lambda(u)$ and root r_1 of $\lambda(u, w)$ with the root of the tree

$\lambda(w)$. Relabel to avoid name clashes. Let $\Gamma(G,\lambda)$ be the isomorphism class of the resulting rooted multigraph.

Given a degree sequence $\mathbf{n}$, let $m = \frac{1}{2}\sum_i i n_i$ and let $B_{\mathbf{n}}$ be the lexicographically least partition of the point set $R_{\mathbf{n}} = \{1, \ldots, 2m\}$ into *blocks* (subsets) such that, for each i, there are n_i blocks of size i. A d-rooted *configuration* C with degree sequence $\mathbf{n}$ [1, 3] is a tuple $(R_{\mathbf{n}}, B_{\mathbf{n}}, P, r_0, \ldots, r_{d-1})$ where P is a partition of the points in $R_{\mathbf{n}}$ into *pairings,* which are unordered pairs of points and $r_0, \ldots, r_{d-1}$ are distinct blocks (roots). We use the phrase *configuration* to mean a 0-rooted configuration and the phrase *rooted configuration* to mean a d-rooted configuration for any d (including $d = 0$). We let $f(C)$ denote the rooted multigraph obtained from C by identifying the points in each block. We say that C is *connected* if $f(C)$ is connected. If $\mathbf{n}$ is degenerate, let $\mathcal{C}_{\mathbf{n}}$ be the set containing the 1-rooted configuration with degree sequence $\mathbf{n}$. For all other irreducible degree sequences $\mathbf{n}$, let $\mathcal{C}_{\mathbf{n}}$ be the set containing all connected unrooted configurations with degree sequence $\mathbf{n}$.

A *colouring* of a rooted configuration $C = (R, B, P)$ is a function λ that maps each block $b \in B$ to an element of $\mathcal{B}$ and each pairing $p \in P$ to an element of $\mathcal{P}$. The function Γ is defined in terms of the corresponding function for rooted multigraphs. In particular, $\Gamma(C,\lambda)$ is defined to be equal to $\Gamma(f(C),\lambda)$. We use the notation $\mathcal{C}_{\mathbf{n},\mathbf{n}'}$ to denote the set $\{(C,\lambda) \mid C \in \mathcal{C}_{\mathbf{n}'} \text{ and } \Gamma(C,\lambda) \in \widetilde{\mathcal{G}}_{\mathbf{n},\mathbf{n}'}\}$.

For degree sequence $\mathbf{n}'$ let $K_{\mathbf{n}'}$ denote the *Kranz* group [5] operating on the points in $R_{\mathbf{n}'}$. Each permutation π in $K_{\mathbf{n}'}$ is associated with a tuple $(\pi_0, \ldots, \pi_{|B_{\mathbf{n}'}|})$ where π_0 is a permutation of blocks and π_i for $i > 0$ is a permutation of the points within block i. To apply π to $R_{\mathbf{n}'}$, one first permutes the blocks using π_0, and then permutes the points within block i (for each i) using π_i. A rooted configuration $C_1 = (R_{\mathbf{n}'}, B_{\mathbf{n}'}, P_1, r_{0,1}, \ldots, r_{d-1,1})$ is said to be *isomorphic* to a configuration $C_2 = (R_{\mathbf{n}'}, B_{\mathbf{n}'}, P_2, r_{0,2}, \ldots, r_{d-1,2})$ if there is a permutation $\pi = (\pi_0, \ldots, \pi_{|B_{\mathbf{n}'}|}) \in K_{\mathbf{n}'}$ such that for all pairings $(u, v) \in P_1$ we have $(\pi(u), \pi(v))$ in P_2 and for all $j \in [0, d-1]$ we have $\pi_0(r_{1,j}) = r_{2,j}$. The coloured rooted configuration $C_1' = (C_1, \lambda_1)$ is said to be *isomorphic* to the coloured rooted configuration $C_2' = (C_2, \lambda_2)$ if there is an isomorphism $\pi = (\pi_0, \ldots, \pi_{|B_{\mathbf{n}'}|})$ between C_1 and C_2 such that for all blocks $b \in B_{\mathbf{n}'}$, $\lambda_1(b) = \lambda_2(\pi_0(b))$ and for all pairings $(u, v) \in P_1$, $\lambda_1(u, v) = \lambda_2(\pi(u), \pi(v))$. Note that if $(C_1, \lambda_1) \cong (C_2, \lambda_2)$ and $(C_1, \lambda_1) \in \mathcal{C}_{\mathbf{n},\mathbf{n}'}$ then $(C_2, \lambda_2) \in \mathcal{C}_{\mathbf{n},\mathbf{n}'}$. We use the notation $\widetilde{\mathcal{C}}_{\mathbf{n},\mathbf{n}'}$ to denote the set of isomorphism classes in $\mathcal{C}_{\mathbf{n},\mathbf{n}'}$. The *automorphism group* of rooted configuration C (denoted Aut(C)) is the group of isomorphisms between C and itself. The *coloured automorphism group* of rooted coloured configuration (C, λ) (denoted Aut(C, λ)) is the group of isomorphisms between (C, λ) and itself.

Lemma 2.6 *If $G \in \mathcal{G}_{\mathbf{n}}$ and C is a rooted configuration such that $f(C) \cong A(G)$ then there is a colouring λ such that $\Psi(G) = \Gamma(C, \lambda)$.*

Proof: The process of forming core $A(G)$ with vertex set V and edge multiset E can be viewed as deleting a tree $h(v)$ for each node $v \in V$ and a tree-chain $h(u, v)$ for each edge $(u,v) \in E$. Suppose that $\pi(f(C)) = A(G)$. Let λ be a colouring of $f(C)$ defined by $\lambda(v) = h(\pi(v))$ and

$$\lambda(u,v) = \begin{cases} h(\pi(u),\pi(v)) & \text{if } \pi(u) < \pi(v), \\ R(h(\pi(u),\pi(v))) & \text{otherwise,} \end{cases}$$

where we assume the endpoints of the edge (u,v) are normalised so that $u < v$. Then $G \in \Gamma(f(C),\lambda)$ so $G \in \Gamma(C,\lambda)$. □

Lemma 2.7 *Suppose that C_1 and C_2 are rooted configurations with irreducible degree sequence $\mathbf{n}'$. If $\Gamma(C_1,\lambda_1) = \Gamma(C_2,\lambda_2)$ then $(C_1,\lambda_1) \cong (C_2,\lambda_2)$.*

Proof: Let G_1 be the multigraph obtained in the construction of $\Gamma(C_1,\lambda_1)$. Make sure that the relabelling that occurs in the construction of G_1 does not change the labels of the vertices of $f(C_1)$. Similarly, let G_2 be the multigraph obtained in the construction of $\Gamma(C_2,\lambda_2)$ in which the labels of the vertices of $f(C_2)$ are unchanged. Now, by the definition of Γ, $A(G_1) = f(C_1)$ and $A(G_2) = f(C_2)$. Suppose that $\pi(G_1) = G_2$. By Lemma 2.5, $\pi(f(C_1)) = f(C_2)$. Thus, for any vertex v in the vertex set of $f(C_1)$, $\lambda_1(v) = \lambda_2(\pi(v))$. Furthermore, for any unordered pair (u,v) of vertices, and any colour ℓ, the number of copies of (u,v) in the edge multiset of $f(C_1)$ that are coloured ℓ by λ_1 is equal to the number of copies of $(\pi(u),\pi(v))$ in the edge multiset of $f(C_2)$ that are coloured ℓ by λ_2. Hence, π can be extended to an isomorphism mapping (C_1,λ_1) to (C_2,λ_2). □

Corollary 2.8 *There is a bijection between $\widetilde{\mathcal{G}}_{\mathbf{n},\mathbf{n}'}$ and $\widetilde{\mathcal{C}}_{\mathbf{n},\mathbf{n}'}$.*

Proof: The corollary follows from Lemma 2.6 and Lemma 2.7. □

Lemma 2.9 *Each isomorphism class in $\widetilde{\mathcal{C}}_{\mathbf{n},\mathbf{n}'}$ comes up $|K_{\mathbf{n}'}|$ times in*

$$\{(C,\lambda,\pi) \mid (C,\lambda) \in \mathcal{C}_{\mathbf{n},\mathbf{n}'} \quad \textit{and} \quad \pi \in \mathrm{Aut}(C,\lambda)\}.$$

Proof: This is a straightforward application of Burnside's Lemma [5]. □

3 Sampling irreducible multigraphs

The goal of this section is a polynomial-time algorithm that takes as input an irreducible degree sequence, $\mathbf{n}$, and samples, uniformly at random (u.a.r.), a pair (C,π), where $C \in \mathcal{C}_{\mathbf{n}}$ is a rooted connected configuration with degree sequence $\mathbf{n}$, and $\pi \in K_{\mathbf{n}}$ is an automorphism of C. This is straightforward if $\mathbf{n}$ is degenerate, so we focus on the non-degenerate case in which $n_0 = n_1 = n_2 = 0$ and $n_i > 0$ for some $i \in [3,\ldots,\Delta]$. In this case, $\mathcal{C}_{\mathbf{n}}$

is the set of connected unrooted configurations with degree sequence $\mathfrak{n}$. (The configurations are unrooted because every connected multigraph in which each vertex has degree at least 3 has more than one cycle.) Thus, our goal is equivalent to generating, u.a.r., an unlabelled connected multigraph (possibly with self-loops) with degree sequence $\mathfrak{n}$. So we obtain a solution to our basic problem in the special case in which all vertex degrees are at least 3. The techniques described in Section 2 provide a reduction from the case of general degrees sequences to the restricted ones considered here, as we shall see in Section 5.

Our approach borrows freely from Bollobás's treatment of unlabelled regular graphs [2], though we find it more convenient to work throughout with configurations in place of (multi)graphs. Recall that $2m = \sum_i i n_i$. We say that a triple (s, s_2, s_3) of non-negative integers is *legal* if $2s_2 + 3s_3 \leq s \leq 2m$. For every legal triple (s, s_2, s_3), let $K_n(s, s_2, s_3)$ denote the set of permutations in K_n that contain exactly s_2 transpositions, s_3 3-cycles, and move exactly s points in all. For convenience, we introduce $s_4 = (s - 2s_2 - 3s_3)/4$; note that s_4 is not necessarily an integer. Of course, only three of the four parameters need to be specified in any situation, but the freedom to move between different triples according to context is convenient.

To generate the pair (C, π) we first select a legal triple (s, s_2, s_3), then a permutation $\pi = K_n(s, s_2, s_3)$, and finally a configuration $C \in \operatorname{Fix}\pi$, where $\operatorname{Fix}\pi$ denotes the set of configurations with degree sequence $\mathfrak{n}$ that are fixed by π. In the unlikely event that C is not connected, we return $\perp$ (see Figure 1 and Theorem 3.3). For every legal triple (s, s_2, s_3), define

$$F_n(s, s_2, s_3) = \left\lceil 4 \times \frac{(2m)!}{m!\,2^m} \times \left(\frac{6s_2}{m^2}\right)^{s_2/2} \left(\frac{3s_3}{m^3}\right)^{s_3/2} \left(\frac{21s_4}{m^4}\right)^{s_4/2} \right\rceil . \tag{1}$$

The significance of $F_n(s, s_2, s_3)$, as we shall see presently, is that it is a uniform upper bound on $|\operatorname{Fix}\pi|$ over all $\pi \in K_n(s, s_2, s_3)$. Note that in equation (1), and throughout the proof of Lemma 3.1 (below), we shall encounter expressions such as

$$\left(\frac{6s_2}{m^2}\right)^{s_2/2}$$

that are formally undefined when s_2 (or s_3 or s_4 or s) is equal to 0. The intended meaning is the limit as the variable in question (here s_2) tends to 0 from above. In all cases the upshot is that the factor concerned is 1 when the variable is 0. Note that F is the square-root of a rational number, rounded up, and hence can be computed exactly in polynomial time. Define

$$W_n(s, s_2, s_3) = |K_n(s, s_2, s_3)| \times F_n(s, s_2, s_3), \tag{2}$$

and let

$$W_n = \sum_{s,s_2,s_3} W_n(s, s_2, s_3), \tag{3}$$

where the sum is over all legal triples (s, s_2, s_3). Observe that W_n is a bound on the size of the set of pairs (C, π) we wish to sample from.

The proposed sampling procedure is conceptually very simple, and is presented in Figure 1 towards the end of the section. Its analysis rests on the following technical lemma.

Lemma 3.1 *With* F_n *and* W_n *defined as above:*

1. $|\mathrm{Fix}()| = \frac{1}{4}F_n(0,0,0)$, *where* $()$ *denotes the identity permutation in* K_n*;*

2. $|\mathrm{Fix}\,\pi| \leq F_n(s, s_2, s_3)$, *for all* $\pi \in K_n(s, s_2, s_3)$*;*

3. $W_n \leq A\, W_n(0,0,0)$, *where* A *depends only on* Δ.

Proof: The total number of configurations with degree sequence $\mathbf{n}$ is equal to the number of ways of choosing m pairings in a set of size $2m$. All configurations are fixed by the identity permutation, so we have

$$|\mathrm{Fix}()| = (2m-1)(2m-3)\cdots 3 \cdot 1 = \frac{(2m)!}{m!\,2^m}.$$

Comparing the above expression with the definition of F already gives us part (a) of the lemma.

An asymptotic expression for the number of configurations can be obtained using the usual Stirling's approximation. For our purposes, it is convenient to have absolute upper and lower bounds, which can be obtained using a more refined version of Stirling's approximation due to Robbins [16] (or see [4, p. 4]):

$$\left(\frac{2m}{e}\right)^m \leq \frac{(2m)!}{m!\,2^m} \leq \sqrt{2}\left(\frac{2m}{e}\right)^m. \tag{4}$$

While we are on the subject of Stirling's formula, let us note for future reference the following slight strengthening of a familiar bound on binomial coefficients:

$$\sum_{i=0}^{t} \binom{n}{i} \leq \left(\frac{en}{t}\right)^t. \tag{5}$$

To verify this inequality, first observe that the right-hand side is monotonically increasing (viewed as a real function) for $t \in (0, 1)$, and is greater than 2^n for $t \geq n/3$. In the case $t < n/3$, the ratio between successive terms on the left-hand side exceeds 2, so the sum is bounded by the sum of a geometric series with common ratio $\frac{1}{2}$. Thus

$$\sum_{i=0}^{t} \binom{n}{i} \leq 2\binom{n}{t} \leq \frac{2n^t}{t!} \leq n^t \left(\frac{e}{t}\right)^t,$$

again using a sufficiently strong form of Stirling's approximation.

Consider $C \in \operatorname{Fix}\pi$, with $\pi \in K_n(s, s_2, s_3)$. Each point in a 3-cycle of π must be paired with a point in a *different* three cycle, and the other two pairings of C incident at the first cycle are then forced. Thus $|\operatorname{Fix}\pi| = 0$ unless s_3 is even, in which case C induces a set of "higher level pairings" on the 3-cycles of π. Given these higher level pairings, there are $3^{s_3/2}$ ways to choose the pairings themselves. In all there are

$$\frac{s_3!\, 3^{s_3/2}}{(s_3/2)!\, 2^{s_3/2}} \leq \sqrt{2}\left(\frac{3s_3}{e}\right)^{s_3/2}$$

ways to choose the restriction of C to the 3-cycles of π. For transpositions, the calculation is similar, except we must now allow for the pairing to join the two points in single transposition. But this new freedom can only blow up the number of choices by (crudely) a factor 2^{s_2}, so that there are at most

$$\sqrt{2}\left(\frac{8s_2}{e}\right)^{s_2/2}$$

ways to choose the restriction of C to the transpositions of π. An optimisation over the distribution of cycle lengths greater than 3 confirms that the number of ways of choosing the restriction of C to those cycles is at most

$$\sqrt{2}\left(\frac{16s_4}{e}\right)^{s_4/2},$$

the bound we would obtain by assuming all the remaining cycles have length exactly 4. The number of ways of extending C to the fixed points of π is clearly bounded by

$$\sqrt{2}\left(\frac{2m}{e}\right)^{(2m-s)/2} \leq \sqrt{2} \times \frac{(2m)!}{m!\, 2^m} \times \left(\frac{2m}{e}\right)^{-s/2},$$

where we have used the other part of inequality (4). Multiplying these four bounds together, recalling $s = 2s_2 + 3s_3 + 4s_4$, yields the following upper bound on $|\operatorname{Fix}\pi|$:

$$|\operatorname{Fix}\pi| \leq 4 \times \frac{(2m)!}{m!\, 2^m} \times \left(\frac{2es_2}{m^2}\right)^{s_2/2} \left(\frac{3e^2 s_3}{8m^3}\right)^{s_3/2} \left(\frac{e^3 s_4}{m^4}\right)^{s_4/2};$$

comparing this expression with equation (1) defining F gives us the second part of the lemma.

For the third part, we introduce a more refined partitioning of the group K_n according to cycle structure. For each cycle of a permutation $\pi \in K_n$, we distinguish whether the cycle touches more than one block of R_n (type 1), or whether its action is entirely confined to a single block (type 2). We write, for example, $s_2 = s_2' + s_2''$, where s_2' is the number of type 1 transpositions, and s_2'' the number of type 2 transpositions. The prime and double prime convention is applied consistently, so that we write $s = s' + s''$, where s' is the total number of points contained in all type 1 cycles, and s'' the number in all type 2 cycles. Naturally, s_4' and s_4'' are defined by $s' = 2s_2' + 3s_3' + 4s_4'$ and $s'' = 2s_2'' + 3s_3'' + 4s_4''$. Denote by

$$K_n(s, s_2', s_3'; s'', s_2'', s_3'') \subseteq K_n(s' + s'', s_2' + s_2'', s_3' + s_3'')$$

the set of permutations with s_2' type 1 transpositions, s_2'' type 2 transpositions, and so on.

The strategy for establishing the final part of the lemma is: (i) compute an upper bound on $|K_n(s', s_2', s_3'; s'', s_2'', s_3'')|$, (ii) optimise over the feasible region to obtain an upper bound on $|K_n(s, s_2, s_3)|$ and hence on $W_n(s, s_2, s_3)$, and (iii) sum over feasible s, s_2, s_3 to obtain an upper bound on W_n. Our upper bound for (i) will be of the form $\kappa'(s_2', s_3', s_4') \times \kappa''(s_2'', s_3'', s_4'')$, where κ' and κ'' are bounds on the number of ways of choosing the type 1 cycles and type 2 cycles, respectively. The latter is more tractable, so we deal with it first.

Let $\pi \in K_n(s', s_2', s_3'; s'', s_2'', s_3'')$. The number of ways of choosing the $i \le s_2''$ blocks containing the s_2'' type 2 transpositions in π is at most

$$\sum_{i=0}^{s_2''} \binom{n}{i} \le \left(\frac{en}{s_2''}\right)^{s_2''},$$

using inequality (5), and so the total number of ways of choosing the transpositions themselves is at most

$$\left(\frac{ecn}{s_2''}\right)^{s_2''},$$

where $c = \Delta!$. Similar bounds hold for the longer cycles, yielding an overall bound of

$$\kappa''(s_2'', s_3'', s_4'') = \left(\frac{ecn}{s_2''}\right)^{s_2''} \left(\frac{ecn}{s_3''}\right)^{s_3''} \left(\frac{ecn}{s_4''}\right)^{s_4''} \tag{6}$$

on the number of ways of choosing all the type 2 cycles.

We now consider the type 1 cycles of π. Denote by $B = B(s', s_2', s_3')$ the set of integer triples (b, b_2, b_3) satisfying

$$b_2, b_3 \ge 0, \quad 2b_2 + 3b_3 \le b, \quad 3b + [2s_2' - 6b_2] + [3s_3' - 9b_3] \le s', \tag{7}$$

where $[x] = \max\{x, 0\}$. The intended interpretation of (b, b_2, b_3) is as follows: b is the total number of blocks moved by π, b_2 is the number of transpositions of blocks induced by π, and b_3 is the number of 3-cycles on blocks induced by π. The significance of B is that it contains, as we shall demonstrate, all feasible choices for (b, b_2, b_3) consistent with (s', s'_2, s'_3).

Only the final inequality of (7) requires explanation. The weaker inequality $3b \le s'$ is easy enough to justify, as each block contains at least 3 points, so we just have to account for the other two terms. If a block contains $p \ge 4$ points, regard $p - 3$ of the points as constituting an "excess." All s'_2 type 1 transpositions in π must be contained within the $2b_2$ blocks that are transposed by π. If $2s'_2 > 6b_2$, then $2s'_2 - 6b_2$ points in type 1 cycles must be in the excess. Similarly, if $3s'_3 > 9b_3$, then $3s'_3 - 9b_3$ further points in type 1 cycles must be in the excess. This justifies the final inequality in (7).

Applying a crude bound on the number of ways of choosing the type 1 cycles, given (b, b_2, b_3), we have

$$\sum_{(b,b_2,b_3)\in B} \frac{(cn)^b}{b_2!\, b_3!} \le (s'+1)^3 \max\left\{ \frac{(cn)^b}{b_2!\, b_3!} : (b, b_2, b_3) \in B \right\} \tag{8}$$

as a bound on the number of ways of choosing the type 1 cycles. The right hand side of (8) presents a small optimisation problem. We claim that at the maximum, $b_2 \ge \lfloor s'_2/3 \rfloor$ (otherwise $b_2 \leftarrow b_2 + 1$ and $b \leftarrow b + 2$ leads to an improvement), and $b_3 \ge \lfloor s'_3/3 \rfloor$ (otherwise $b_3 \leftarrow b_3 + 1$ and $b \leftarrow b + 3$ does), and in any case $b \le s'/3$. So, from (8), and using the lower bound $b! \ge (b/e)^b$, the number of ways of choosing the type 1 cycles is at most $\kappa'(s'_2, s'_3, s'_4)$, where

$$\begin{aligned}
\kappa'(s'_2, s'_3, s'_4) &= (s'+1)^5 (cn)^{s'/3} \left(\frac{3e}{s'_2}\right)^{s'_2/3} \left(\frac{3e}{s'_3}\right)^{s'_3/3} && (9)\\
&= (s'+1)^5 \left(\frac{3ec^2n^2}{s'_2}\right)^{s'_2/3} \left(\frac{3ec^3n^3}{s'_3}\right)^{s'_3/3} (cn)^{4s'_4/3}. && (10)
\end{aligned}$$

The extra factor $(s'+1)^2$ in (9) takes account of the floor functions. Our upper bound for $|K_n(s', s'_2, s'_3; s'', s''_2, s''_3)|$ is thus

$$|K_n(s', s'_2, s'_3; s'', s''_2, s''_3)| \le \kappa'(s'_2, s'_3, s'_4) \times \kappa''(s''_2, s''_3, s''_4),$$

where $\kappa'(s'_2, s'_3, s'_4)$ and $\kappa''(s''_2, s''_3, s''_4)$ are as defined in (6) and (10).

The next stage is to bound $|K_n(s, s_2, s_3)|$. Clearly we have

$$\begin{aligned}
|K_n(s, s_2, s_3)| &\le \sum_{S} \kappa'(s'_2, s'_3, s'_4) \times \kappa''(s''_2, s''_3, s''_4)\\
&\le (s+1)^3 \max_{S} \left\{ \kappa'(s'_2, s'_3, s'_4) \times \kappa''(s''_2, s''_3, s''_4) \right\}, && (11)
\end{aligned}$$

where S is the region

$$\begin{aligned} S \;=\; \Big\{ & (s_2', s_3', s_4', s_2'', s_3'', s_4'') \in (\mathbb{R}^+)^6 : \\ & s_2' + s_2'' = s_2,\ s_3' + s_3'' = s_3,\ \text{and } s_4' + s_4'' = s_4 \Big\} \end{aligned}$$

If we bound the $(s'+1)^5$ factor in κ' simply by $(s+1)^5$, then the factors in s_2', s_2'', in s_3', s_3'', and in s_4', s_4'' appearing in the objective function of (11) separate out, and we can optimise over each pair separately.

- The s_4 factor is

$$(cn)^{4s_4'/3}\left(\frac{cen}{s_4''}\right)^{s_4''} \le (cn)^{4s_4/3}, \tag{12}$$

  since the maximum is achieved at $s_4' = s_4$ and $s_4'' = 0$.

- The s_3 factor is

$$\begin{aligned} \left(\frac{3ec^3n^3}{s_3'}\right)^{s_3'/3}\left(\frac{e^3c^3n^3}{(s_3'')^3}\right)^{s_3''/3} &\le 2\left(\frac{e^3c^3n^3}{s_3'}\right)^{s_3'/3}\left(\frac{e^3c^3n^3}{s_3''}\right)^{s_3''/3} && (13) \\ &\le 2\left(\frac{2e^3c^3n^3}{s_3}\right)^{s_3/3}. && (14) \end{aligned}$$

  Inequality (13) uses the fact that $x^{-x} \le 2x^{-x/3}$ for all positive x, and inequality (14) follows from symmetry and unimodality.

- The s_2 factor is

$$\begin{aligned} \left(\frac{3ec^2n^2}{s_2'}\right)^{s_2'/3}\left(\frac{e^3c^3n^3}{(s_2'')^3}\right)^{s_2''/3} &\le 2\left(\frac{e^3c^3n^3}{(s_2')^2}\right)^{s_2'/3}\left(\frac{e^3c^3n^3}{(s_2'')^2}\right)^{s_3''/3} \\ &\le 2\left(\frac{4e^3c^3n^3}{s_2^2}\right)^{s_2/3}, && (15) \end{aligned}$$

  by similar considerations to the previous case.

Plugging (12), (14) and (15) into (11) gives

$$|K_n(s, s_2, s_3)| \le 4(s+1)^8\left(\frac{4e^3c^3n^3}{s_2^2}\right)^{s_2/3}\left(\frac{2e^3c^3n^3}{s_3}\right)^{s_3/3}(cn)^{4s_4/3},$$

which, on recalling the definitions (1) and (2) of F_n and W_n, leads to:

$$W_n(s, s_2, s_3) \le 32(s+1)^8 \times \frac{(2m)!}{m!\,2^m} \times \left(\frac{c_2}{s_2}\right)^{s_2/6}\left(\frac{c_3 s_3}{n^3}\right)^{s_3/6}\left(\frac{c_4 s_4^3}{n^4}\right)^{s_4/6}, \tag{16}$$

Step 1 If n is degenerate, let C be the sole member of $\mathcal{C}_n$, choose $\pi \in \mathrm{Aut}(C)$ u.a.r., and output (C, π). Otherwise, perform Steps 2–6.

Step 2 Choose the triple (s, s_2, s_3) with probability $W_n(s, s_2, s_3)/W_n$.

Step 3 Choose $\pi \in K_n(s, s_2, s_3)$, u.a.r.

Step 4 Choose $C \in \mathrm{Fix}\,\pi$, u.a.r.

Step 5 If C is not connected, output $\perp$ and halt.

Step 6 With probability $|\mathrm{Fix}\,\pi|/F_n(s, s_2, s_3)$ output (C, π); otherwise output $\perp$.

Figure 1: Procedure CONFIGSAMPLE for sampling a pair (C, π)

where c_2, c_3 and c_4 are constants depending only on c and hence only on Δ. (The multiplicative factor has been boosted from 16 to 32 to allow for the ceiling function in the definition of F.)

To finish off the proof of the final part of the lemma, we just need to sum (16) over all legal triples (s, s_2, s_3):

$$\begin{aligned} W_n &= \sum_{s,s_2,s_3} W_n(s, s_2, s_3) \\ &\leq 32 \times \frac{(2m)!}{m!\,2^m} \times \sum_{s_2} \left(\frac{c_2}{s_2}\right)^{s_2/6} \sum_{s_3,s_4} (2s_2 + 3s_3 + 4s_4 + 1)^8 \left(\frac{c_3\Delta}{3n^2}\right)^{s_3/6} \left(\frac{c_4\Delta^3}{64n}\right)^{s_4/6} \\ &\approx 32 \times \frac{(2m)!}{m!\,2^m} \times \sum_{s_2} \left(\frac{c_2}{s_2}\right)^{s_2/6} (2s_2 + 1)^8 \\ &\approx 4c' \times \frac{(2m)!}{m!\,2^m} = c' W_n(0, 0, 0), \end{aligned}$$

where c' depends only on c_2, and hence only on Δ. □

Lemma 3.2 *There is a polynomial-time algorithm for computing* $|K_n(s, s_2, s_3)|$, *and hence for computing* $W_n(s, s_2, s_3)$ *and* W_n. *There is also a polynomial-time algorithm for sampling, uniformly at random, a permutation from* $K_n(s, s_2, s_3)$.

Proof: By partitioning $|K_n(s, s_2, s_3)|$, first according to the length of the first induced cycle on blocks, and then on the exact pattern of cycles within those blocks (at most $\Delta!$ possibilities), we obtain an inductive formula for $|K_n(s, s_2, s_3)|$. Only polynomially many distinct assignments to the parameters n, s, s_2, and s_3 arise during the induction, so $|K_n(s, s_2, s_3)|$ can be computed in time polynomial in n by dynamic programming. □

Theorem 3.3 *The procedure* CONFIGSAMPLE *presented in Figure 1 is correct: (a) the probability that the algorithm returns a value other than $\perp$ is bounded away from 0; (b) for any configuration $C \in \mathcal{C}_n$ with degree sequence $\mathbf{n}$, and any automorphism $\pi \in \mathrm{Aut}(C)$ of C, the probability that the pair (C, π) is returned by* CONFIGSAMPLE *is a constant, namely W_n^{-1}, independent of C and π; and (c) the procedure* CONFIGSAMPLE *runs in time polynomial in n, provided the maximum degree Δ is bounded. Indeed, we have the following strengthening of the first part: (d) the probability that a pair with $\pi = ()$ is returned is bounded away from 0.*

Proof: By the second part of Lemma 3.1, the acceptance probability in Step 6 is well defined. By the third part of Lemma 3.1, the particular triple $(0, 0, 0)$ is selected in Step 2 with probability at least A^{-1}, which is bounded away from 0. This forces the identity permutation to be selected in Step 3. In this case, the probability of rejection in Step 5 is bounded away from 0. (By Bollobás and Bender and Canfield (See [4], page 48), the probability that a random configuration with degree sequence $\mathbf{n}$ corresponds to a simple graph is bounded away from 0. Each simple graph corresponds to an equal number of configurations, and by Wormald [20], the probability that a simple graph with degree sequence $\mathbf{n}$ is connected is bounded away from 0.) By the first part of Lemma 3.1, we know that the pair $(C, ())$ survives Step 6 with probability $\frac{1}{4}$. This deals with (a) and its strengthening (d).

Now consider an arbitrary pair (C, π) satisfying $C \in \mathrm{Fix}\,\pi$, and suppose $\pi \in K_n(s, s_2, s_3)$. For (C, π) to be generated, a certain well defined event must occur at each step of the algorithm. The probability that (C, π) is generated is simply the product of these four probabilities:

$$\frac{W_n(s, s_2, s_3)}{W_n} \times \frac{1}{|K_n(s, s_2, s_3)|} \times \frac{1}{\mathrm{Fix}\,\pi} \times \frac{\mathrm{Fix}\,\pi}{F_n(s, s_2, s_3)} = \frac{1}{W_n},$$

which is clearly independent of C and π, as asserted in (b).

According to Lemma 3.2 the procedure can be implemented to run in polynomial time.
□

4 Sampling unlabelled trees

The previous section showed how to sample, u.a.r., an unlabelled connected multigraph with a specified irreducible non-degenerate degree sequence. In Section 5 we will show how to sample, u.a.r., an unlabelled connected multigraph with *any* specified degree sequence $\mathbf{n}$. Our basic strategy will be to select an irreducible degree sequence $\mathbf{n}' \leq \mathbf{n}$ with the "appropriate" probability, sample u.a.r. an unlabelled connected multigraph G' with degree sequence $\mathbf{n}'$, and finally colour G' to obtain a multigraph G with degree sequence $\mathbf{n}$. (G' will be the core of G as defined in Section 2.) Instead of constructing G' directly, we will select a pair (C', π) as described in section 3 such that $C' \in \mathcal{C}_{n'}$ and $\pi \in \mathrm{Aut}(C')$. Then we will choose a colouring λ such that $\Gamma(C', \lambda) \in \widetilde{\mathcal{G}}_{n}$. The process of choosing $\mathbf{n}'$ and λ involves counting and sampling unlabelled rooted trees. We provide the relevant tree results in this section.

The basic framework in which we will work is as follows: We will consider "structures" (trees with one or more root), each of which has a "weight" (a $(\Delta+2)$-tuple of integers) The weight of a d-rooted n-vertex tree G (which is denoted $\mu(G)$) is the tuple $(n-d, i_0, \dots, i_\Delta)$, where i_r is the number of vertices of degree r, *excluding* the roots. We let G_1 denote the 1-rooted tree consisting of a single vertex and G_2 denote the 2-rooted tree consisting of a single edge. We define the following operations on trees. If G is a 1-rooted tree, we let $[d]G$ denote the 1-rooted tree obtained by taking d copies of G, identifying the roots of the d copies, and then relabelling the remaining vertices of trees $2, \dots, d$ to avoid name clashes. If G and G' are 1-rooted trees, we let $G \times G'$ denote the tree obtained by identifying their roots and relabelling the remaining vertices of G' to avoid name clashes. If G is a tree-chain and G' is a 1-rooted tree then we let $G * G'$ denote the tree-chain constructed from G and G' as follows. Root r_1 of G is is disconnected from its neighbour, v, in G, and is connected to the root of G'. The root of G' is connected to v. The vertices in G' are then relabelled go avoid name clashes. If G is a d-rooted tree and G' is a d'-rooted tree, we let $G + G'$ be the $(d + d')$-rooted tree obtained by relabelling the vertices and roots of G' to avoid name clashes.

Following Flajolet, Zimmerman and Van Cutsem [10], we form sets of structures from $\{G_1\}$ and $\{G_2\}$ using the following constructors.

- $S + S'$: The disjoint union of S and S'
- For $d > 1$, $[d]S$: $\{[d]G \mid G \in S\}$
- $S \times S'$: $\{G \times G' \mid G \in S, G' \in S'\}$
- If all structures in S' have a single root of a given degree, $S * S'$: $\{G * G' \mid G \in S, G' \in S'\}$
- $S \cdot S'$: $\{G + G' \mid G \in S, G' \in S'\}$

The constructors $+$ and $\cdot$ are from [10], which also considers other constructors. We use the notation $m \cdot S$ as an abbreviation for the disjoint union of m copies of S, $S + \cdots + S$, and we use the notation S^m as an abbreviation for the Cartesian product of m copies of S, $S \cdots S$.

A *specification* of sets $S_0, \dots, S_r$ (which is sometimes referred to as a specification of S_r) is defined to be a sequence of equations

$$m_0 \cdot S_0 = \Psi_0(), m_1 \cdot S_1 = \Psi_1(S_0), \dots, m_r \cdot S_r = \Psi_r(S_0, \dots, S_{r-1}),$$

where $m_0, \ldots, m_r$ are positive integers and, for $i \in [0, \ldots, r]$, Ψ_i is a term built from $\{G_1\}$, $\{G_2\}$, and $S_0, \ldots, S_{i-1}$ using the constructors. An ℓ-specification is a specification using ℓ constructors.[1] For every set S of structures, we use the notation $S(i_0, \ldots, i_j)$ to denote the set

$$\{s \in S \mid \text{for some } i_{j+1}, \ldots, i_{\Delta+1},\ \mu(s) = (i_0, \ldots, i_{\Delta+1})\}.$$

The generating function for the set S is a function $s(x_0, \ldots, x_{\Delta+1})$ such that the coefficient of $x_0^{i_0} \cdots x_{\Delta+1}^{i_{\Delta+1}}$ in $s(x_0, \ldots, x_{\Delta+1})$, which is denoted

$$[x_0^{i_0} \cdots x_{\Delta+1}^{i_{\Delta+1}}]\, s(x_0, \ldots, x_{\Delta+1}),$$

is equal to $|S(i_0, \ldots, i_{\Delta+1})|$. The following is a straightforward extension of a theorem of Flajolet et al. [10]

Theorem 4.1 *Given an ℓ-specification for sets $S_0, \ldots, S_r$, a set of equations for the corresponding generating functions is obtained automatically by the following translation rules:*

$$\begin{array}{ll}
m \cdot S = S' + S'' & \Rightarrow s(x_0, \ldots, x_{\Delta+1}) = (1/m)(s'(x_0, \ldots, x_{\Delta+1}) + s''(x_0, \ldots, x_{\Delta+1})), \\
m \cdot S = [d]S' & \Rightarrow s(x_0, \ldots, x_{\Delta+1}) = (1/m)s'(x_0^d, \ldots, x_{\Delta+1}^d), \\
m \cdot S = S' \times S'' & \Rightarrow s(x_0, \ldots, x_{\Delta+1}) = (1/m)(s'(x_0, \ldots, x_{\Delta+1}) \cdot s''(x_0, \ldots, x_{\Delta+1})), \\
m \cdot S = S' * S'' & \Rightarrow s(x_0, \ldots, x_{\Delta+1}) = (x_{r+2}/m)(s'(x_0, \ldots, x_{\Delta+1}) \times s''(x_0, \ldots, x_{\Delta+1})), \\
 & \quad \textit{where } r \textit{ is the degree of the roots of the structures in } S'', \\
m \cdot S = S' \cdot S'' & \Rightarrow s(x_0, \ldots, x_{\Delta+1}) = (1/m)(s'(x_0, \ldots, x_{\Delta+1}) \times s''(x_0, \ldots, x_{\Delta+1})).
\end{array}$$

Furthermore, there is a polynomial p such that all coefficients

$$[x_0^{i'_0} \ldots x_{\Delta+1}^{i'_{\Delta+1}}] S_j(x_0, \ldots, x_{\Delta+1})$$

such that $j \in [0, \ldots, r]$ and $(i'_0, \ldots, i'_{\Delta+1}) \le (i_0, \ldots, i_{\Delta+1})$ can be computed in at most $p(i_0, \ldots, i_{\Delta+1}, r, \ell)$ steps and a member of $S_j(i_0, \ldots, i_{\Delta+1})$ can be sampled u.a.r. in $p(i_0, \ldots, i_{\Delta+1}, r, \ell)$ steps.

Proof: The coefficients of $S_0, \ldots, S_r$ can be computed in order and stored in a table. Sampling u.a.r. from $S_j(i_0, \ldots, i_{\Delta+1})$ is accomplished as follows. If $m \cdot S_j = \{G_1\}$ or $m \cdot S_j = \{G_2\}$, this is straightforward. If $m \cdot S_j = S_a + S_b$, sample u.a.r. from $S_a(i_0, \ldots, i_{\Delta+1})$ with probability

$$|S_a(i_0, \ldots, i_{\Delta+1})| / |S_j(i_0, \ldots, i_{\Delta+1})|,$$

and from $S_b(i_0, \ldots, i_{\Delta+1})$ with the remaining probability. If $m \cdot S_j = [d]S_a$ then recursively sample from $S_a(i_0/d, \ldots, i_{\Delta+1}/d)$ and make d copies of the resulting structure. If $m \cdot S_j =$

[1] For this, we count the constructor $[d]$ in $[d]S$ as d constructors.

$S_c \times S_d$, evaluate

$$N_{i'} = |S_c(i'_0,\ldots,i'_{\Delta+1})| \times |S_d(i_0 - i'_0,\ldots,i_{\Delta+1} - i'_{\Delta+1})|$$

for all tuples $\mathbf{i}' = (i'_0,\ldots,i'_{\Delta+1})$ such that $(0,\ldots,0) < \mathbf{i}' < \mathbf{i} = (i_0,\ldots,i_{\Delta+1})$. Choose $\mathbf{i}'$ with probability $N_{i'}/\sum_{j'} N_{j'}$. Recursively sample structures from $S_c(i'_0,\ldots,i'_{\Delta+1})$ and $S_d(i_0 - i'_0,\ldots,i_{\Delta+1} - i'_{\Delta+1})$ and combine the structures. The cases in which $m\cdot S_j = S_c * S_d$ and $m\cdot S_j = S_c \cdot S_d$ are handles similarly, except that in the case $m\cdot S_j = S_c * S_d$ we replace i_{r+2} with $i_{r+2} - 1$. □

We will now use the above framework to show how to count and sample unlabelled rooted trees. Let S_r be the set containing one representative from each isomorphism class in the set of 1-rooted trees with degree-r roots. We will first give a sequence of equations to define the sets $S_r(n)$ in terms of the constructors and we will then argue that the definition is a specification (that is, the equations can be ordered in such a way that each equation depends only on sets previously defined). The set of equations is adapted from Nijenhuis and Wilf [12]. First, note that $S_0(0) = \{G_1\}$ and $S_0(n) = \emptyset$ for $n \neq 0$. Furthermore, $S_r(0) = \emptyset$, for $r \neq 0$. For $n > 0$, we have

$$\begin{aligned} S_1(n) &= \sum_{\substack{0\le r\le\Delta-1\\ i_0+\cdots i_\Delta = n-1}} S_r(n-1, i_0,\ldots,i_r, i_{r+1}-1, i_{r+2},\ldots,i_\Delta), \text{ and} \\ n\cdot S_r(n) &= \sum_{\substack{1\le s\le r\\ 1\le sd\le n}} d\cdot([s]S_1(d+1) \times S_{r-s}(n-sd)), \text{ for } r > 1. \end{aligned}$$

The first equation expresses the correspondence between $n+1$-vertex trees rooted at a vertex of degree 1, and n-vertex trees with unrestricted root degree. The second equation expresses a construction for trees which has the property that each unlabelled $n+1$-vertex tree is represented n times: choose numbers s, d satisfying $1 \le s \le r$ and $1 \le sd \le n$; choose a tree τ' with $n+1-sd$ vertices, rooted at a vertex of degree $r-s$, and a tree τ'' with $d+1$ vertices rooted at a vertex of degree 1; take s copies of τ' and one copy of τ'' and identify all the roots (the identified vertices constitute the new root). Make d copies of the resulting rooted tree. Nijenhuis and Wilf [12, p. 274] give a combinatorial proof of the equation by establishing an explicit bijection between the figures enumerated by the left and right sides.

To see that the sequence of equations given is a specification, consider a 2-dimensional table. The first $r+1$ entries of column n correspond to sets $S_0(n),\ldots,S_r(n)$ (in the given order). The remaining entries correspond to the sets $[s]S_1(n/s+1)$, where $s > 1$ divides n. Note that the equation corresponding to each table entry only uses sets corresponding to smaller rows or columns of the table. Thus, we have a specification for the sets $S_r(n)$.

As we described in the beginning of this section, our algorithm in Section 5 will sample u.a.r. a colouring λ of a configuration $C = (R_{n'}, B_{n'}, P)$ such that $\Gamma(C,\lambda) \in \widetilde{\mathcal{G}}_n$. Recall that a colouring λ of C is a mapping from $B_{n'}$ to $\mathcal{B}$ (the set of block-colours), and from P to $\mathcal{P}$ (the set of pairing-colours). The blocks in $B_{n'}$ are ordered and the pairings in P can be ordered according to the ordering of the blocks, so a colouring may be specified as a sequence of n' block-colours followed by a sequence of $m' = \frac{1}{2}\sum_i in'_i$ pairing colours. Thus, the set of available colourings depends only on $\mathbf{n}$ and $\mathbf{n}'$. Let $\Lambda_{n,n'}$ denote the set of available colourings. Given the specification for the set $S_r(n)$, we can derive specifications for $\mathcal{B}$, $\mathcal{P}$, and therefore, $\Lambda_{n,n'}$. We start by observing that $\mathcal{B} = S_0 + \cdots + S_\Delta$. Let $\mathcal{P}_\ell$ denote the set containing one representative from each isomorphism class of length-ℓ tree-chains (thus, $\mathcal{P} = \mathcal{P}_0 + \mathcal{P}_1 + \mathcal{P}_2 + \cdots$). A specification for $\mathcal{P}$ is as follows:

$$\begin{aligned} \mathcal{P}_0 &= \{G_2\}. \\ \mathcal{P}_\ell &= (\mathcal{P}_{\ell-1} * S_0) + \cdots + (\mathcal{P}_{\ell-1} * S_{\Delta-2}). \end{aligned}$$

Let $L_{n',r_0,\ldots,r_{n'}}$ denote the set of colourings of a configuration with degree sequence $\mathbf{n}'$ in which the block-colouring for the ith block is a tree whose root has degree r_i. Then the set $L_{n',r_0,\ldots,r_{n'}}$ can be specified using the equation

$$L_{n',r_0,\ldots,r_{n'}} = \mathcal{P}^{m'} \cdot S_{r_0} \cdots S_{r_{n'}}.$$

Finally, note that $\Lambda_{n,n'}$ is the disjoint union, over all (polynomially many) choices of $r_0, \ldots, r_{n'}$ of $L_{n',r_0,\ldots,r_{n'}}(n - n'', n_0 - n''_0, \ldots, n_\Delta - n''_\Delta)$, where

$$n''_i = \left|\{j : 1 \leq j \leq n' \text{ and } v_j + r_j = i\}\right|,$$

where v_i denotes the size of the ith block in $B_{n'}$.

Thus, we have a specification for $\Lambda_{n,n'}$. While some of the sets used in the specification such as $\mathcal{P}$ and $S_0,\ldots,S_\Delta$ are infinite, these sets are made up by taking the disjoint union of finite subsets. Accordingly, there is a polynomial-sized specification for $\Lambda_{n,n'}$ and the following is a corollary of Theorem 4.1.

Corollary 4.2 *There is a polynomial* p *such that computing* $|\Lambda_{n,n'}|$ *and sampling u.a.r. from* $\Lambda_{n,n'}$ *take at most* $p(n)$ *steps.*

5 Sampling unlabelled multigraphs

Let $\mathcal{H}_n$ be the set of connected multigraphs with degree sequence $\mathbf{n}$ and let $\widetilde{\mathcal{H}}_n$ be the set of isomorphism classes of $\mathcal{H}_n$. In this section, we will describe a procedure MULTI-SAMPLE that samples u.a.r. from $\widetilde{\mathcal{H}}_n$. The procedure will first (see Steps 1–4 of Figure 2)

Step 1 Select a degree sequence $\mathbf{n}' \leq \mathbf{n}$ according to the probability distribution $p_{\mathbf{n}}$.

Step 2 Select a pair (C, π) using the procedure CONFIGSAMPLE developed in Section 3 (see Figure 1), with parameter $\mathbf{n}'$. If that procedure returns $\perp$, then output $\perp$ and halt; otherwise the result is a pair selected u.a.r. from the set of pairs (C, π), with $C \in \mathcal{C}_{\mathbf{n}'}$ and $\pi \in \mathrm{Aut}(C)$.

Step 3 Select a colouring λ u.a.r. from $\Lambda_{\mathbf{n},\mathbf{n}'}$.

Step 4 If $\pi \in \mathrm{Aut}(C, \lambda)$ then let G be any rooted multigraph in $\Gamma(C, \lambda)$; otherwise output $\perp$ and halt.

Step 5 If G has at least two cycles then output $\Psi(G)$. Otherwise, let k be the number of non-isomorphic 1-rooted multigraphs with the same vertex and edge set as G. (The choice of root is arbitrary in the case of trees, but must be on the cycle in the case of unicyclic multigraphs.) With probability k^{-1} output $\Psi(G)$; otherwise output $\perp$.

Figure 2: Procedure MULTISAMPLE for sampling u.a.r. from $\widetilde{\mathcal{H}}_{\mathbf{n}}$.

sample u.a.r. from $\widetilde{\mathcal{G}}_{\mathbf{n}}$ and will then use rejection to obtain a uniform distribution on $\widetilde{\mathcal{H}}_{\mathbf{n}}$.

All the components of MULTISAMPLE are now ready: Section 2 introduced the machinery that we will use to reduce the general problem to the special case in which $\mathbf{n}$ is irreducible, Section 3 solved the irreducible case, and Section 4 described the tools that we will use to lift the solution for the irreducible case up to a solution for general degree sequences. It only remains to assemble the pieces.

Given a degree sequence $\mathbf{n}$, let the probability distribution $p_{\mathbf{n}}$ assign probability

$$p_{\mathbf{n}}(\mathbf{n}') = \frac{W_{\mathbf{n}'}|\Lambda_{\mathbf{n},\mathbf{n}'}|}{M\,|K_{\mathbf{n}'}|}. \tag{17}$$

to irreducible degree sequences satisfying $\mathbf{n}' \leq \mathbf{n}$, and zero probability to the others. Here,

$$M = \sum_{\mathbf{n}' \leq \mathbf{n}} \frac{W_{\mathbf{n}'}|\Lambda_{\mathbf{n},\mathbf{n}'}|}{|K_{\mathbf{n}'}|}$$

is the normalising factor required to form a probability distribution. (The sum is over irreducible degree sequences $\mathbf{n}' \leq \mathbf{n}$. The fact that this is the right summation follows from Observation 2.3.) The significance of $p_{\mathbf{n}}$ is that it is the "correct" distribution from which to sample the degree sequence of the core. This is the final ingredient in the sampling procedure MULTISAMPLE, which is presented in Figure 2.

Lemma 5.1 *The procedure* MULTISAMPLE *presented in Figure 2 is correct: (a) the probability that the algorithm produces an output other than $\perp$ is $\Omega(n^{-1})$; (b) for each isomorphism class $U \in \widetilde{\mathcal{H}}_{\mathbf{n}}$, the probability that U is returned by* MULTISAMPLE *is a constant,* *namely M^{-1}, independent of U; and (c) the procedure* MULTISAMPLE *runs in time polynomial in n, assuming the maximum degree Δ is bounded.*

Proof: The procedure successfully completes Step 2 precisely if some value other than $\perp$ is returned by procedure CONFIGSAMPLE; the probability of this event is bounded away from 0, by part (a) of Theorem 3.3. Indeed, part (d) of that theorem tells us more: namely that the automorphism $\pi \in \mathrm{Aut}(C)$ returned by CONFIGSAMPLE is the identity with probability bounded away from 0. But if $\pi = ()$, Step 4 is guaranteed to be successful. The probability that Step 5 is successful it at least $1/n$. This completes the proof of (a).

We now proceed to compute the probability that a certain isomorphism class $U \in \widetilde{\mathcal{H}}_n$ appears as output. We start by showing that, after Step 4, the probability that G is in any given class in $\widetilde{\mathcal{G}}_n$ is M^{-1}. Let U be a class in $\widetilde{\mathcal{G}}_n$. By Lemma 2.4, U has a uniquely defined core with degree sequence $\mathbf{n}'$, say. By Lemma 2.7, A condition for U to be returned in Step 4 is that the degree sequence $\mathbf{n}'$ is selected in Step 1, an event which occurs with the probability $p_n(\mathbf{n}')$, given in equation (17). Now fix attention on a particular triple (C, π, λ), satisfying $C \in \mathcal{C}_{n'}$ and $\pi \in \mathrm{Aut}(C, \lambda)$. By Theorem 3.3, the probability that (C, π, λ) is selected in Steps 2 and 3, conditioned on the particular choice of degree sequence $\mathbf{n}'$, is $(W_{n'}|\Lambda_{n,n'}|)^{-1}$. By Corollaries 2.8 and 2.9, exactly $|K_{n'}|$ of these triples correspond to the desired output U. Thus, again conditioned on the choice of $\mathbf{n}'$, the probability that U is returned is

$$\frac{|K_{n'}|}{W_{n'}|\Lambda_{n,n'}|}.$$

Multiplying this expression by the probability (17) that degree sequence $\mathbf{n}'$ is selected in Step 1, we see that the overall probability that U is returned at the end of Step 4 is a constant, in fact M^{-1}. If $U \in \widetilde{\mathcal{H}}_n$ has at least 2 cycles, it comes up once in $\widetilde{\mathcal{G}}_n$. Otherwise, it appears k times in $\widetilde{\mathcal{G}}_n$, where k is as in Figure 2. By accepting U only with probability k^{-1}, the output distribution after Step 5 is uniform on $\widetilde{\mathcal{H}}_n$.

Step 1 is polynomial time by Lemma 3.2 and Corollary 4.2; Step 2 is polynomial time by Theorem 3.3; and Step 3 by Corollary 4.2. Step 4 is clearly polynomial time. Step 5 is reducible to isomorphism of 1-rooted trees, which can conveniently be decided by a recursive canonical labelling scheme: if the root is the only vertex assign it label $()$; otherwise let $l_1, l_2, \ldots, l_t$ be the labels of the t subtrees of the root, ordered lexicographically, and assign label $(l_1 l_2 \ldots l_t)$ to the root. By induction, two 1-rooted trees are isomorphic iff their root labels are equal. Thus, we have established (c). □

6 Sampling molecules

In this section we extend our results to the chemical problem — given a molecular formula, select, uniformly at random, a structural isomer having the given formula. We start by extending the algorithm in section 5 so that it can be used to uniformly sample unlabelled connected *self-loop-less* multigraphs with a given degree sequence. For this we use procedure MULTISAMPLE, except that if the degree sequence of the core is not degenerate and the core has a self-loop, it is rejected. Furthermore, if the degree sequence of the core is that of the single-self-loop multigraph, we reject the trivial colouring of the single pairing in the core. The rejection probability is not too high in either case. The reason for this (in the non-degenerate case) is that the core is a simple graph with probability bounded away from 0 (see section 5)

The modified version of procedure MULTISAMPLE, which uniformly samples unlabelled connected self-loop-less multigraphs with a given degree sequence, solves the following problem: Given a molecular formula in which each atom has a distinct valence, select, uniformly at random, a structural isomer having the given formula. We can further modify procedure MULTISAMPLE so that it can be used to uniformly sample structural isomers even when the molecular formula has different atoms with the same valence. Formally, we fix t *types* of vertices and we interpret a typed degree sequence

$$n_{0,1}, \ldots, n_{0,t}, \ldots, n_{\Delta,1}, \ldots, n_{\Delta,t}$$

as a requirement that a multigraph have $n_{i,j}$ degree-i vertices of type j. An isomorphism between typed multigraphs must map each vertex to a vertex of the same type. Procedure MULTISAMPLE can be extended in a straightforward way to give a polynomial-time algorithm that takes as input a typed degree sequence and selects, uniformly at random, an unlabelled connected multigraph with the given degree sequence. The generation of the core is as before, except that the definition of the group K_n changes since blocks can only be mapped to other blocks of the same type. The inductive specifications in Section 4 must be modified slightly to account for the types, so the choice of n' is modified accordingly. The choice of the colouring λ is also modified slightly. The colouring of each block must have a root that has the same type as the block and a colouring of a pairing between blocks of types i and j must have roots of types i and j, respectively. Everything else is as before.

Acknowledgements: We thank Jean-Loup Faulon for proposing the problem and explaining the chemical applications. We also thank Alan Frieze for suggesting the use of the core.

References

[1] E.A. Bender and E.R. Canfield, The asymptotic number of labelled graphs with given degree sequences, *Journal of Combinatorial Theory, Series A* **24** (1978) 296–307.

[2] B. Bollobás, The asymptotic number of unlabelled regular graphs, *Journal of the London Mathematical Society* **26** (1982), 201–206.

[3] B. Bollobás, Almost all regular graphs are Hamiltonian, *European Journal on Combinatorics* **4** (1983) 97–106.

[4] B. Bollobás, *Random Graphs*, Academic Press, 1985.

[5] N. G. De Bruijn, Pólya's theory of counting, in *Applied Combinatorial Mathematics,* Beckenbach, E.F., Ed., John Wiley and Sons, Inc., 1964, see especially Section 5.13.

[6] J.D. Dixon and H.S. Wilf, The random selection of unlabeled graphs, *Journal of Algorithms* **4** (1983) 205–213.

[7] J.L. Faulon, On using graph-equivalent classes for the structure elucidation of large molecules, *J. Chem. Inf. Comput. Sci.* **32(4)** (1992) 337–348.

[8] J.L. Faulon, Stochastic generator of chemical structure: 1. Application to the structure elucidation of large molecules, *J. Chem. Inf. Comput. Sci.* **34(5)** (1994) 1204–1218.

[9] J.L. Faulon, personal communication.

[10] P. Flajolet, P. Zimmerman and B. Van Cutsem, A calculus for the random generation of labelled combinatorial structures, *Theoretical Computer Science* **132** (1994) 1–35.

[11] P. Flajolet, P. Zimmerman and B. Van Cutsem, A calculus for the random generation of unlabelled combinatorial structures, in preparation.

[12] A. Nijenhuis and H. S. Wilf, *Combinatorial Algorithms* (2nd edition), Academic Press, 1978.

[13] G.Pólya and R.C. Read, *Combinatorial Enumeration of Groups, Graphs, and Chemical Compounds,* Springer-Verlag, 1987.

[14] R.C. Read, Some recent results in chemical enumeration, in *Graph Theory and its Applications,* Springer-Verlag, 1972.

[15] R.C. Read, The enumeration of acyclic chemical compounds, in *Chemical Applications of Graph Theory,* Balaban, A.T., Ed., Academic Press, 1976.

[16] H. Robbins, A remark on Stirling's formula, *American Mathematical Monthly* **62** (1955), 26–29.

[17] V. Sperschneider and G. Antoniou, *Logic: A Foundation for Computer Science*, Addison-Wesley, 1991, chapter 13.

[18] H.S. Wilf, The uniform selection of free trees, *Journal of Algorithms* **5** (1984) 247–280.

[19] N.C. Wormald, Some problems in the enumeration of labelled graphs, *Ph.D. Thesis, Department of Mathematics, University of Newcastle, New South Wales*, 1978.

[20] N.C. Wormald, The asymptotic connectivity of labelled regular graphs, *Journal of Combinatorial Theory, Series B* **31** (1981) 156–167.

[21] N.C. Wormald, Generating random unlabelled graphs, *SIAM Journal of Computing* **16(4)** (1987) 717–727.

[22] S. Zhan, On Hamiltonian line graphs and connectivity, *Discrete Mathematics* **89** (1991) 89–95.