Randomly Sampling Unlabelled Structures

Leslie Ann Goldberg*

April 20, 1999

Abstract

Informally, an "unlabelled combinatorial structure" is an object such as an unlabelled graph (in which the vertices are indistinguishable) or a structural isomer in chemistry (in which different atoms of the same type are indistinguishable). Computational experiments such as those described in this volume often rely on random sampling to generate inputs for the experiments. This paper surveys work on the problem of efficiently sampling unlabelled combinatorial structures from a uniform distribution.

1 Introduction

Most of the experimental work described in this volume involves first randomly sampling combinatorial structures and second using the randomly-chosen structures as inputs to computational experiments. In order for the experiments to be valid, the distribution from which the combinatorial structures are drawn must be precisely specified. In order for the experiments to be computationally feasible, the random-sampling algorithms must be efficient.

This survey is devoted to the problem of efficiently sampling unlabelled combinatorial structures from a uniform distribution. For information on the related problem of efficiently *listing* unlabelled combinatorial structures without duplicates, see [11] and [12].

We start by giving an example of an unlabelled combinatorial structure — in particular, an unlabelled graph [1]. Two graphs with vertex set $V_n = \{v_1, \ldots, v_n\}$ are said to be *isomorphic* if there is a permutation of V_n which transforms one graph into the other. For example, the graphs G_1 and G_2 from Figure 1 are isomorphic because relabelling the vertices of G_1 according to the permutation $(\, v_1\, v_2\, v_3\,)\,(\, v_4\, v_5\,)\,(\, v_6\,)$ transforms G_1 into G_2. The permutations of V_n partition the n-vertex graphs into *equivalence classes.* Two graphs

*`leslie@dcs.warwick.ac.uk`, `http://www.dcs.warwick.ac.uk/~leslie/`, Department of Computer Science, University of Warwick, Coventry, CV4 7AL, United Kingdom. This work was partially supported by the ESPRIT Projects RAND-II (Project 21726) and ALCOM-IT (Project 20244) and by EPSRC grant GR/L60982.

[1]For graph-theoretic definitions, see [10] and [17].

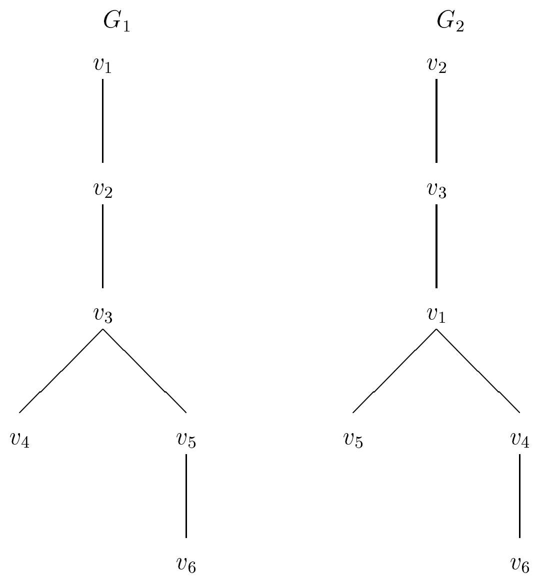

Figure 1: G_1 and G_2 are isomorphic. The permutation $(v_1 v_2 v_3)(v_4 v_5)(v_6)$ transforms G_1 into G_2.

are in the same equivalence class if and only if there is some permutation which transforms one into the other. (That is, they are in the same equivalence class if and only if they are isomorphic.) Formally, an "unlabelled graph" is just an isomorphism class of labelled graphs. Informally, an "unlabelled graph" is what you get when you take a labelled graph and erase the labels on the vertices. For example, the unlabelled graph corresponding to the graphs in Figure 1 is shown in Figure 2.

We will focus on computational problems such as the following:

> **Input:** A positive integer n.
>
> **Output:** An unlabelled graph chosen uniformly at random from the set of all unlabelled graphs with n vertices.

This computational problem illustrates the difference between unlabelled sampling and labelled sampling: To choose a labelled graph uniformly at random (u.a.r.) from the set of all n-vertex graphs, one would just need to consider each unordered pair of the n vertices and choose, independently with probability $1/2$, whether or not to make it an edge. However, this approach would not work for sampling unlabelled graphs. For example, the probability of producing a simple path containing n vertices (using this approach) is $(n!/2)$ times as high as the probability of producing the complete graph on n vertices. In fact, it wasn't until 1987 that a polynomial expected-time algorithm for uniformly sampling unlabelled graphs was discovered. This algorithm was discovered by Wormald [31], building on work of Dixon and Wilf [5]. These algorithms are discussed in Section 3.1.

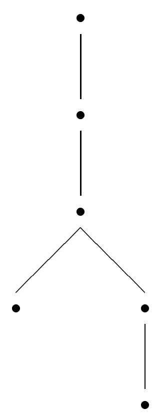

Figure 2: The unlabelled graph corresponding to the graphs in Figure 1.

The general techniques that have been used for sampling unlabelled combinatorial structures are

1. inductive definitions/generating functions, and
2. Burnside's Lemma (Pólya Theory).

These techniques are described in Section 2 and Section 3 respectively.

2 Inductive Definitions

We will use the following definitions to illustrate the "Inductive Definition" approach for sampling unlabelled structures. A tree is a connected graph with no cycles and a rooted tree is a tree in which a particular vertex is distinguished as the root. An unlabelled rooted tree is an isomorphism class of rooted trees, where an isomorphism between two rooted trees is required to map the root of one tree to the root of the other. Informally, an unlabelled rooted tree is what you get when you take a rooted tree and erase the labels on the vertices, remembering which vertex is the root. For example, see Figure 3. Let $\mathcal{U}_n$ denote the set of unlabelled rooted trees with n vertices and let t_n denote the size of $\mathcal{U}_n$.

Nijenhuis and Wilf [25] showed that $\mathcal{U}_n$ and t_n can be defined inductively and that this inductive definition can be used to sample u.a.r. from $\mathcal{U}_n$. Their main observation was that every member of $\mathcal{U}_n$ is constructed exactly $n-1$ times by the following procedure: For every $d \in \{1, \ldots, n-1\}$ and for every positive integer j such that $jd < n$, let T' be a member of $\mathcal{U}_{n-jd}$ and let T'' be a member of $\mathcal{U}_d$. Let T''' be the unlabelled rooted n-vertex graph formed by making j copies of T'' and using j new edges to join the root of each copy

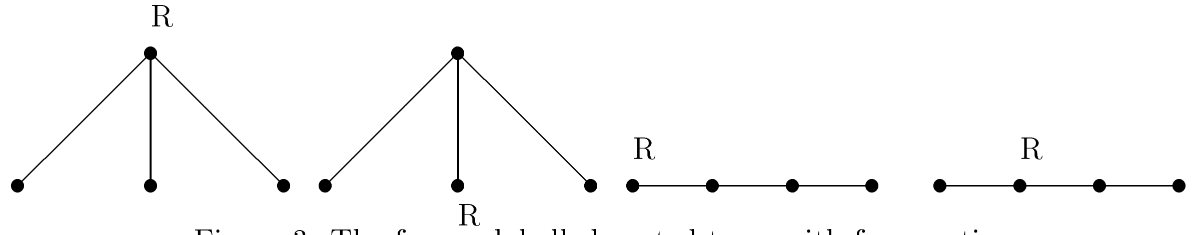

Figure 3: The four unlabelled rooted trees with four vertices.

to the root of T'. (The root of T''' is taken to be the root of T'.) Output d copies of T'''. Thus,

$$t_n(n-1) = \sum_{d=1}^{n-1} \sum_{j\,:\,jd<n} d\, t_{n-jd}\, t_d,$$

and so $t_1, t_2, \ldots$ can be computed by dynamic programming. Finally, the outline of the sampling algorithm (for $n > 2$) is as follows (see also [27]):

1. Choose the pair (j, d) with probability $\frac{dt_{n-jd}t_d}{(n-1)t_n}$.

2. Recursively choose T' u.a.r. from $\mathcal{U}_{n-jd}$.

3. Recursively choose T'' u.a.r. from $\mathcal{U}_d$.

4. Make j copies of T'' and attach the root of each copy to the root of T'.

5. Let the root of T' be the root of the new n-vertex tree and output the new tree.

Nijenhuis and Wilf's approach was extended by Wilf [28], who showed how to uniformly sample free (unrooted) trees. Wilf's algorithm is also based on an inductive definition (i.e., a generating function) for the trees. This approach will be systematised by Flajolet, Zimmerman and Van Cutsem in a forthcoming paper [9]. The companion paper by the same authors on sampling *labelled* combinatorial structures [8] gives a good idea of the systematic approach. The idea is to specify the unlabelled structures using a formal grammar involving set, sequence and cycle constructions. Uniform sampling can then be done automatically using dynamic programming. This systematic approach has also been used by Goldberg and Jerrum to sample some other tree-like unlabelled structures in Section 4 of [15].

3 Burnside's Lemma

Burnside's Lemma is an important ingredient in the generating-function-based sampling approach of [9]. However, it turns out that this ingredient is useful for sampling even when it is not possible directly to write a generating function for the unlabelled structures.

We will start by setting up the framework for Burnside's lemma. Let $\Sigma = \{0, \ldots, k-1\}$ be a finite alphabet of cardinality k, and let G be a permutation group on $[m] = \{0, \ldots, m-1\}$. For example, G could be the group in Example 1.

Example 1: An example of a group with degree[2] $m = 4$ is the group consisting of the following permutations.

- identity,
- (0 2),
- (1 3), and
- (0 2)(1 3).

For $g \in G$ and $i \in [m]$, let i^g denote the image of i under g. For example, if $g = (02)$ we have $0^g = 2$, $1^g = 1$, $2^g = 0$ and $3^g = 3$. Consider the set Σ^m of all words of length m over the alphabet Σ. The group G has a natural action on Σ^m which is induced by permutations of the m "bit positions". Under this induced action, the permutation $g \in G$ maps the word $\alpha = a_0 a_1 \ldots a_{m-1}$ to the word $\alpha^g = b_0 b_1 \ldots b_{m-1}$ defined by $b_j = a_i$ for all $i, j \in [m]$ satisfying $i^g = j$. For example, if $g = (0\ 2)$ and $\alpha = 0010$ then $\alpha^g = 1000$ because $a_0 = 0$, $a_1 = 0$, $a_2 = 1$, $a_3 = 0$ and $b_2 = a_0$, $b_1 = a_1$, $b_0 = a_2$ and $b_3 = a_3$. The action of G partitions Σ^m into a number of *orbits*, which are the equivalence classes of Σ^m under the equivalence relation that identifies α and β whenever there exists $g \in G$ mapping α to β. For example, if $k = 2$ (so $\Sigma = \{0, 1\}$), $m = 4$, and G is the group in Example 1, then the nine orbits of Σ^4 are (0000), $(0001, 0100)$, $(0010, 1000)$, (0101), (1010), $(0011, 0110, 1001, 1100)$, $(1110, 1011)$, $(1101, 0111)$, and (1111). The "Orbit Sampling Problem" (for fixed Σ) is as follows.

> **Input:** A permutation group G of degree m.
>
> **Output:** A word in Σ^m so that each orbit is equally likely to be output.

Most of the unlabelled sampling problems that we consider in this survey can be coded up in terms of the orbit sampling problem. For example, we can represent an N-vertex graph as its adjacency matrix, which is a word in $\Sigma^{\binom{N}{2}}$ where $\Sigma = \{0, 1\}$. The *unlabelled* N-vertex graphs are just the orbits of words in $\Sigma^{\binom{N}{2}}$ with respect to the "pair group" $S_N^{(2)}$. (Each of the permutations in $S_N^{(2)}$ corresponds to a different permutation of the N vertices. However the vertices are permuted, the corresponding permutation in $S_N^{(2)}$ performs the corresponding action on the adjacency matrix.)

Now Burnside's Lemma[3] says that each orbit comes up $|G|$ times (as the first component) in the set of pairs

$$\Upsilon(\Sigma, G) = \{(\alpha, g) \mid \alpha \in \Sigma^m, g \in G \text{ and } \alpha^g = \alpha\}. \tag{1}$$

---

[2] The *degree* of a permutation group is the number of objects on which the permutations act. Not all degree-4 permutation groups have 4 permutations.

[3] Although this lemma is commonly referred to as "Burnside's Lemma", it is really due to Cauchy and Frobenius [24].

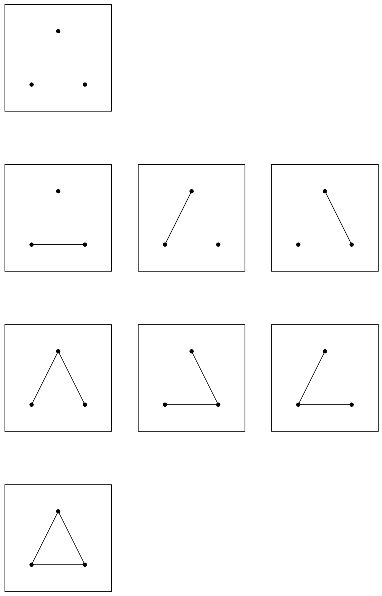

Figure 4: An illustration of Equation 2 for $N = 3$. The eight labelled 3-vertex graphs are divided into four orbits (one on each row) which represent the four unlabelled graphs with 3 vertices. Each row is represented 6 times in Υ_3 because any of the 6 permutations of the vertices maps the graph on the top row to itself. Similarly, the graph on the bottom row has 6 automorphisms (permutations which map it to itself). However, any graph on the middle two rows has only two automorphisms: the identity, and the permutation which transposes the two degree-1 vertices.

3.1 Unlabelled Graphs

Translating Burnside's Lemma to the special case of unlabelled graphs, we find that each unlabelled N-vertex graph comes up $N!$ times (as the first component) in the set of pairs

$$\Upsilon_N = \{(\Gamma, g) \mid \Gamma \text{ is a (labelled) } N\text{-vertex graph and } g \text{ is a permutation of the } N \text{ vertices which maps } \Gamma \text{ to itself.}\}. \quad (2)$$

The special case $N = 3$ is illustrated in Figure 4.

The significance of Burnside's Lemma (Equation 2) is that in order to uniformly sample N-vertex unlabelled graphs, we need only sample u.a.r. from Υ_N. This observation was first made by Dixon and Wilf [5]. In order to simplify our presentation of their idea (and Wormald's refinement [31]), for every permutation g of N vertices, we will let Υ_g denote

$$\{(\Gamma, g) \mid \Gamma \text{ is a (labelled) } N\text{-vertex graph and } g \text{ maps } \Gamma \text{ to itself.}\}.$$

Thus, $\Upsilon_N = \bigcup_g \Upsilon_g$, where the union is over all permutations g of N vertices.

We can now state the outline of Dixon and Wilf's algorithm for sampling u.a.r. from Υ_N:

1. Input N

2. Choose g with probability $\frac{|\Upsilon_g|}{|\Upsilon_N|}$

3. Choose (Γ, g) u.a.r from Υ_g.

Step 3 of the algorithm can be implemented in polynomial time: For each cycle of (unordered) vertex pairs induced by g, decide independently (with probability $1/2$) whether to make the pairs edges or non-edges. It is not known how to implement Step 2 in polynomial time. In particular, it is not known how to compute $|\Upsilon_N|$ in polynomial time[4].

Wormald's algorithm [31] uses the *rejection sampling* method to get around computing $|\Upsilon_N|$. The basic idea of rejection sampling is as follows. It may be too difficult to sample from the desired distribution. So what the user does instead is to sample from some other (more tractable) distribution. Imagine the desired distribution as being "scaled down" so that it fits underneath the more tractable distribution. To draw a sample from the desired distribution, the user first draws a sample from the more tractable distribution. The user then uses the sample to determine the probability with which the more tractable distribution over-represents this sample (relative to the "scaled down" desired distribution). With this probability, the sample is rejected and the sampling process is re-started. Otherwise, the sample is output. The method is useful when it is easy to determine the probability with which a given sample should be rejected and, furthermore, the overall rejection probability is low (so the expected running time until a sample is output is small).

The outline of Wormald's algorithm is as follows, where i_N represents the identity permutation on N vertices and p_g represents the probability with which permutation g is chosen (so $\sum_g p_g = 1$). Appropriate choices for p_g will be discussed below.

1. Input N

2. Choose g with probability p_g.

3. Choose (Γ, g) u.a.r. from Υ_g.

4. Output (Γ, g) with probability $\frac{p_{i_N}}{|\Upsilon_{i_N}|} \frac{|\Upsilon_g|}{p_g}$. Otherwise reject.

It is straightforward to check that the probability that any given pair (Γ, g) from Υ_N is output is $\frac{p_{i_N}}{|\Upsilon_{i_N}|}$, so the algorithm does sample u.a.r. from Υ_N as long as Criterion 1 (below) is met (so Step 4 can be implemented).

[4]Dixon and Wilf show how to implement Step 2 in polynomial time *on average* provided the value of $|\Upsilon_N|$ is known. (Their method involves choosing a conjugacy class with the appropriate probability and then choosing a representative permutation g from within the conjugacy class.)

Criterion 1: The probabilities p_g must be chosen so that

$$\frac{p_{i_N}}{|\Upsilon_{i_N}|}\frac{|\Upsilon_g|}{p_g} \leq 1.$$

Furthermore, the rejection probability is 0 whenever $g = i_N$. Thus, the expected running time of the sampling algorithm is polynomial as long as the other criteria below are met.

Criterion 2: The probabilities p_g must be chosen so that Step 2 can be implemented in polynomial time[5].

Criterion 3: The probabilities p_g must be chosen so that Step 4 can be implemented in polynomial time.

Criterion 4: The probabilities p_g must be chosen so that, for some positive constant c and every N, we have $p_{i_N} \geq N^{-c}$. (This ensures that the expected number of trials before an output is produced (with no rejection) is at most N^c.)

Wormald [31] showed how to choose the probabilities p_g so that these criteria are met. Thus, he gave an expected polynomial-time algorithm for sampling unlabelled graphs.

3.2 Extending Wormald's Method

Let us now consider how to extend Wormald's method to the general "Burnside's Lemma" framework. Let Σ be a fixed alphabet. For any permutation g of $[m]$, let

$$\Upsilon(\Sigma, g) = \{(\alpha, g) \mid \alpha \in \Sigma^m \text{ and } \alpha^g = \alpha\}.$$

Then $\Upsilon(\Sigma, G)$ from Equation 1 equals $\bigcup_{g\in G} \Upsilon(\Sigma, g)$. We can write the outline of Wormald's algorithm using this notation, where i_m denotes the identity permutation of degree m:

1. Input a permutation group G
2. Choose $g \in G$ with probability p_g.
3. Choose (α, g) u.a.r. from $\Upsilon(\Sigma, g)$.
4. Output (α, g) with probability $\frac{p_{i_m}}{|\Upsilon(\Sigma,i_m)|}\frac{|\Upsilon(\Sigma,g)|}{p_g}$. Otherwise reject.

An appropriate measure of "input size" for the input G is the degree of G, because every degree-m permutation group can be specified by a set of $\mathrm{O}(m)$ permutations [18]. Thus, the analogue of Criterion 4 states that there is a positive constant c such that for every

[5] Note that choosing $p_g = \frac{|\Upsilon_g|}{|\Upsilon_N|}$ would make Wormald's algorithm equivalent to Dixon and Wilf's. Thus, it would satisfy all criteria except Criterion 2.

possible input group G of degree m, we must have $p_{i_m} \geq m^{-c}$. On the other hand, the analogue of Criterion 1 implies

$$\frac{p_g}{p_{i_m}} \geq \frac{|\Upsilon(\Sigma, g)|}{|\Upsilon(\Sigma, i_m)|}, \tag{3}$$

which implies

$$p_{i_m} \leq \frac{|\Upsilon(\Sigma, i_m)|}{|\Upsilon(\Sigma, G)|} = \frac{|\Sigma|^m}{|\Upsilon(\Sigma, G)|}.$$

Thus, we cannot simultaneously satisfy the two criteria unless there is a positive constant c such that for every possible input G of degree m we have

$$|\Upsilon(\Sigma, G)| \leq m^c |\Sigma|^m. \tag{4}$$

That is, the size of $\Upsilon(\Sigma, G)$ must be at most a polynomial times the size of the part of $\Upsilon(\Sigma, G)$ which corresponds to the identity permutation. We will say that a family of permutation groups has "too many non-trivial symmetries" (with respect to the fixed alphabet Σ) if there is not a positive constant c such that every group G in the family satisfies Equation 4. Since some families of unlabelled combinatorial structures do correspond to families of groups with too many non-trivial symmetries, Wormald's method cannot be used *in general* for unlabelled sampling. Wormald has shown [31] how to use the method to efficiently sample unlabelled r-regular graphs for $r \geq 3$. (Note that for fixed $r \geq 3$, the unlabelled r-regular graphs do not have too many non-trivial symmetries. In fact, most unlabelled r-regular graphs have only the identity as a symmetry.)

3.3 Structures with Too Many Non-Trivial Symmetries

A (labelled) *multigraph* is given by a set of vertices and a *multiset*[6] of unordered pairs of vertices, which are called edges. An unlabelled multigraph is an isomorphism class of labelled multigraphs. The *degree sequence* of multigraph G is a sequence telling how many vertices of each degree G has. Consider the following unlabelled sampling 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. This sampling problem arises in the context of sampling *structural isomers* in chemistry [7, 15]. It is an example of a sampling problem to which Wormald's method does not apply: If the degree sequence has many vertices of degree 1 or 2 then the unlabelled multigraphs with the degree sequence have too many non-trivial symmetries.

Goldberg and Jerrum [15] have given a polynomial expected-time algorithm for this sampling problem based on the following idea. Every unlabelled multigraph G is associated with a unique "core"[7] which has no vertices of degree 1 or 2. To randomly generate a multigraph G, the algorithm first generates the core of G and then extends the core by adding trees and chains of trees to obtain G.

[6] A *multiset* is like a set except that it can contain more than one copy of an object, so a multigraph can contain several (indistinguishable) edges between any given pair of vertices.

[7] For other uses of the "core" idea, see Zhan [32].

The algorithm for generating the core is described using the configuration model of Bender and Canfield [1], Bollobás [3] and Wormald [29]. A configuration (for a given degree sequence) is a labelled combinatorial structure which can be viewed as a refinement of a multigraph with the degree sequence. For any given degree sequence, the orbits of all configurations (with respect to the appropriate permutation group) correspond to the unlabelled multigraphs with the degree sequence. Since the degree sequence of the core has no vertices of degree 1 or 2, the cores do not have too many non-trivial symmetries. (This follows from an extension of Bollobás's analysis of unlabelled regular graphs [2].) Thus, the algorithm uses Wormald's method to generate the core. (If the core is not connected, it is rejected. The fact that this does not happen too often follows from a result of Wormald [30].)

After generating the core of the random multigraph, the algorithm extends the core by adding trees and chains of trees. This part of the algorithm is based on the generating-function approach mentioned in Section 2.

It is an open problem to sample unlabelled multigraphs given a *general* degree sequence (in which degrees need not be bounded from above by a constant). Goldberg and Jerrum's method is not applicable when the degrees are unbounded. In fact, the problem with unbounded degrees seems to be difficult even in the labelled case (see [21, 23, 6]).

3.4 A General Approach Based on Burnside's Lemma

Jerrum [19] proposed a general way to use Burnside's Lemma for unlabelled sampling. The idea is to consider the following bipartite graph: The vertices on the left-hand side are all words in Σ^m. The vertices on the right-hand side are all permutations in G. There is an edge from word α to permutation g if and only if $\alpha^g = \alpha$. For example, when $\Sigma = \{0, 1\}$ and $m = 4$ and G is the group described in Example 1, we get the graph in Figure 5. This graph essentially implements Burnside's Lemma: The edges in the graph are the pairs in $\Upsilon(\Sigma, G)$. Thus, Burnside's Lemma (Equation 1) says that the set of left-endpoints of edges contains $|G|$ representatives of each orbit.

Now consider a random walk on this graph: When the walk is at a given node, it chooses a neighbour of the node u.a.r. and moves to the neighbour. If we view the random walk from the perspective of the right-hand vertices we can see that it has a stationary distribution, and that, in the stationary distribution, the probability of being at any node is proportional to its degree. Thus, the stationary distribution allows us to sample edges (and thus, left end-points) u.a.r. By Burnside's Lemma, the stationary distribution therefore allows us to sample orbits u.a.r. Three issues arise when we consider how to use this method to sample orbits:

1. How long does it take to simulate a right-to-left step of the random walk?

2. How long does it take to simulate a left-to-right step of the random walk?

3. How many steps of the random walk do we need to simulate in order to get close to the stationary distribution?

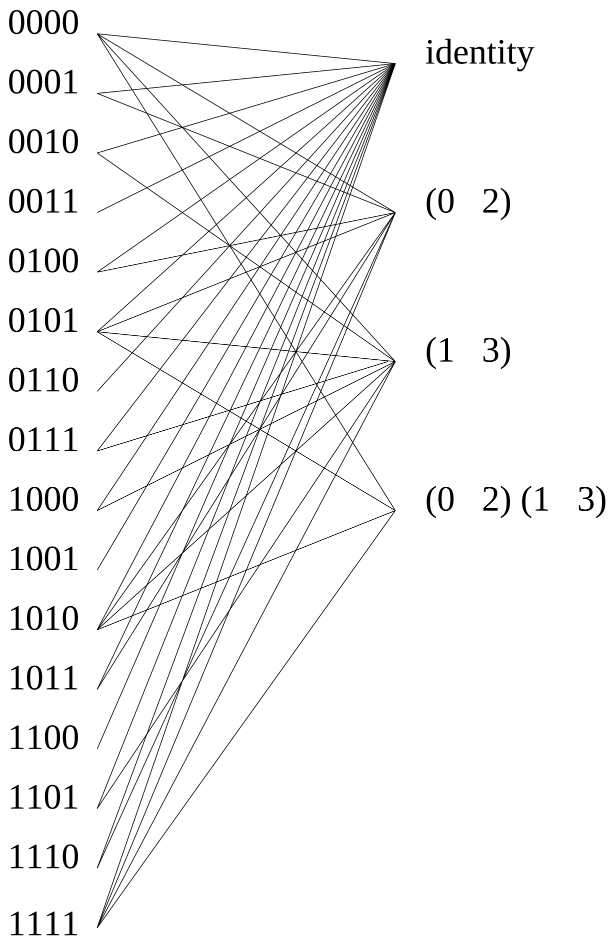

Figure 5: The bipartite graph depicting Burnside's Lemma for $\Sigma = \{0, 1\}$, $m = 4$, and the group G from Example 1.

The first two issues arise because the bipartite graph is, in general, too large to construct explicitly. We are looking for sampling algorithms with run-times that are polynomial in m, the degree of G, but the number of left-hand vertices is $|\Sigma|^m$ and the number of right-hand vertices could be as large as $m!$. Fortunately, we can simulate the random walk without explicitly constructing the graph. Taking a step from right to left can be done in polynomial time: For each cycle of the permutation corresponding to the current right-hand vertex, a letter in Σ is chosen u.a.r. (This is just a generalisation of Step 3 of Dixon and Wilf's algorithm.) Taking a step from left to right is not so easy. In fact, it is equivalent under randomised polynomial-time reductions to the *Setwise Stabiliser* problem, which includes *Graph Isomorphism* as a special case. Nevertheless, there are significant classes of groups G for which this step can be implemented in polynomial time. Luks has shown that the class of p-groups—groups in which every element has order a power of p for some prime p—is an example of such a class [22].

Before moving on to the third issue, we mention another situation in which the simulation of the left-to-right step seems to work sufficiently well. Peter Cameron has observed that the random walk described in this section could be defined for any group action, not just for the special case of a permutation group G acting on Σ^m by permuting positions. The general setting is as follows: Given a point α, select u.a.r. a group element g that fixes α, and then select a point that is fixed by g. This random walk has been implemented in certain algorithms for determining the conjugacy classes of a finite group [26].

Clearly the effectiveness of the random walk as a basis for general-purpose sampling procedures for unlabelled structures depends upon the third issue: "How many steps of the random walk do we need to simulate in order to get close to the stationary distribution?" In order to answer this question, we need some definitions. For any two probability distributions π and π' on a finite set Ψ, we define the *variation distance* between π and π' to be

$$D(\pi, \pi') = \max_{A \subseteq \Psi} |\pi(A) - \pi'(A)| = \frac{1}{2} \sum_{x \in \Psi} |\pi(x) - \pi'(x)|.$$

We say that the "tolerance-δ mixing time" of the random walk is the minimum t such that for all start vertices v_0 and for all $t' \geq t$, $D(\pi_{t'}(v_0), \pi) \leq \delta$, where π denotes the stationary distribution of the walk, and $\pi_{t'}(v_0)$ denotes the distribution after t' steps, given that the walk starts at vertex v_0. We say that the random walk is "rapidly mixing" if its "tolerance-δ mixing time" is at most a polynomial in m (the input size — the degree of the group G) and $\log(1/\delta)$.

Jerrum [19] showed that the random walk is rapidly mixing if the group G is cyclic or symmetric. If G is cyclic (i.e., it is generated by a single permutation) then the orbits do not have too many non-trivial symmetries, so a sampling algorithm based on Wormald's method might also work. If G is the symmetric group then the orbits have too many non-trivial symmetries to satisfy Equation 4 so Wormald's method will not work. However it is interesting to note that an analogous situation occurs: A polynomially-large fraction of the pairs in $\Upsilon(\Sigma, G)$ have word $\mathtt{00}\cdots\mathtt{0}$ in the first component.

Goldberg and Jerrum [14] showed that the random walk is not always rapidly mixing: In particular, for any fixed alphabet there is an infinite family of groups G for which the tolerance-$\frac{1}{3}$ mixing time is exponential in the degree of G. Thus, there is an infinite family of groups for which the variation distance is at least $1/3$ after exponentially-many steps. The groups in this family are uncomplicated from a group-theoretic point of view: The permutations in the groups commute and have order three. Thus, the groups are p-groups, and each step of the random walk can be simulated in polynomial time. However, the groups are complicated from a combinatorial point of view[8]. It would be interesting to know for *which* families of groups the random walk is rapidly mixing. As far as I know, [19] and [14] are the only known results about this.

Goldberg and Jerrum's negative result leaves open the possibility that for a fixed alphabet there might be some *other* efficient sampling algorithm (one which does not simulate the random walk) which takes input G and outputs a right-hand vertex (or a left-hand vertex) drawn from the stationary distribution of the random walk (or from a distribution close to the stationary distribution). As we have seen before, sampling from the stationary distribution of the left-hand vertices is easy provided we can sample from the stationary distribution of the right-hand vertices. A more general result of this form is given in [19].

Thus, an interesting open question is whether there exists an efficient sampling algorithm which takes input G and outputs a right-hand vertex (i.e., a group element) with (approximately) the correct distribution. We conclude this section by describing weak evidence indicating that such an algorithm may not exist.

First, we will be more specific about the notion of an "approximately correct" distribution. A "good" sampling algorithm will

1. run in $\mathrm{poly}(m, \epsilon^{-1})$ time, where m is the degree of the input group, and ϵ is an accuracy parameter, and
2. produce a random right-hand vertex v such that for any vertex v, the probability with which v is returned is in the range $[(1-\epsilon)p, (1+\epsilon)p]$, where p is the probability of being at vertex v in the stationary distribution of the random walk.

Second, we will define the "cycle index polynomial" of a permutation group G. When this polynomial is evaluated at k, the value is

$$\frac{1}{|G|}\sum_{g\in G} k^{c(g)},$$

where $c(g)$ denotes the number of cycles in permutation g.

Finally, we will state a curious consequence which would occur if a good sampling algorithm did exist.

[8]The groups are constructed to mimic the "Swendsen-Wang Process," which is a dynamics for the q-state Potts model. The slow mixing is connected to a first-order phase transition in the number of "ordered" Potts configurations. See [14], [16] and [4] and for details.

If a good sampling algorithm exists, then the following are all true:

1. For any fixed $k > 1$, evaluating the cycle index polynomial of a permutation group G at k is #P-hard.

2. For any fixed positive *non-integer* k, *approximately* evaluating the cycle index polynomial of a permutation group G at k is hard. (Specifically, there is no *fully polynomial randomised approximation scheme (fpras)* unless RP=NP).

3. For any fixed positive *integer* k, *approximately* evaluating the cycle index polynomial of a permutation group G at k is easy. (Specifically, there is an fpras.)

Items 1 and 2 are true (whether or not there is a good sampling algorithm). Item 1 was shown to be true by Goldberg [13] and item 2 was shown to be true by Goldberg and Jerrum [13, 20]. Jerrum [19] showed that Item 3 would be true if there were a good sampling algorithm for every fixed alphabet size k. It may be that items 1–3 are all true (though this would be somewhat surprising). Alternatively, it may be that there is no efficient algorithm for sampling permutations (right-hand vertices) in the "Burnside's Lemma" framework. Resolving this open question would be very interesting.

Acknowledgements

I thank Mark Jerrum for useful comments on an earlier draft of this survey. Most of my work in this area is joint with Mark and I have used material from our joint papers in the survey.

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, G. Grimmett and S. Janson, The random-cluster model on the complete graph, *Probability Theory and Related Fields* **104** (1996), 283–317.

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

[6] M. Dyer and C. Greenhill, A genuinely polynomial-time algorithm for sampling two-rowed contingency tables, *Proceedings of the International Colloquium on Automata, Languages and Programming* **25** Springer Lecture Notes in Computer Science **1443** (1998) 339–350.

[7] 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.

[8] 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.

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

[10] A. Gibbons, *Algorithmic Graph Theory*, (Cambridge University Press, 1995).

[11] L. A. Goldberg, Efficient algorithms for listing unlabeled graphs, *Journal of Algorithms,* **13** (1992) 128–143.

[12] L. A. Goldberg, *Efficient algorithms for listing combinatorial structures,* (Cambridge University Press, 1993).

[13] L. A. Goldberg, Automating Pólya theory: the computational complexity of the cycle index polynomial, *Information and Computation,* **105(2)** (1993) 268–288.

[14] L. A. Goldberg and M. Jerrum, The "Burnside Process" Converges Slowly, In "Randomization and Approximation Techniques in Computer Science (Proceedings of RANDOM 1998)" (M. Luby, J. Rolim and M. Serna, ed.), *Springer Lecture Notes in Computer Science* **1518**, (1998) 331–345.

[15] L.A. Goldberg and M. Jerrum, Randomly Sampling Molecules, To appear in *SIAM Journal on Computing*, (1999).

[16] V. Gore and M. Jerrum, The Swendsen-Wang process does not always mix rapidly, *Proceedings of the ACM Symposium on Theory of Computation* **29** (1997), 674–681.

[17] F. Harary and E. M. Palmer, *Graphical Enumeration*, (Academics Press, 1973).

[18] M. Jerrum, A compact representation for permutation groups, *Journal of Algorithms,* **7** (1986) 60–78.

[19] M. Jerrum, Uniform sampling modulo a group of symmetries using Markov chain simulation. In "Expanding Graphs" (Joel Friedman, ed.), *DIMACS Series in Discrete Mathematics and Theoretical Computer Science* **10**, (American Mathematical Society, 1993) 37–47.

[20] M. Jerrum, Computational Pólya Theory, In "Surveys in Combinatorics 1995," *London Mathematical Society Lecture Note Series* **218** (Cambridge University Press, 1995) 103–118.

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

[22] E. M. Luks, Isomorphism of graphs of bounded valence can be tested in polynomial time, *Journal of Computer and System Sciences* **25** (1982) 42–65.

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

[24] P. M. Neumann, A lemma that is not Burnside's, *Mathematical Scientist* **4** (1979) 133–141.

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

[26] L. Soicher, personal communication.

[27] G. Tinhofer, Generating graphs uniformly at random, *Computing, Supp. 7*, (1990) 235–255.

[28] H. S. Wilf, The uniform selection of free trees, *Journal of Algorithms* **2** (1981) 204–207.

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

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

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

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