Uniform generation of spanning regular subgraphs of a dense graph

Pu Gao[*]
Department of Combinatorics and Optimization
University of Waterloo
ON, N2L 3G1, Canada
pu.gao@uwaterloo.ca

Catherine Greenhill[†]
School of Mathematics and Statistics
UNSW Sydney
NSW 2052, Australia
c.greenhill@unsw.edu.au

Submitted: Oct 18, 2018; Accepted: Oct 1, 2019; Published: Nov 8, 2019

Abstract

Let H_n be a graph on n vertices and let $\overline{H_n}$ denote the complement of H_n. Suppose that $\Delta = \Delta(n)$ is the maximum degree of $\overline{H_n}$. We analyse three algorithms for sampling d-regular subgraphs (d-factors) of H_n. This is equivalent to uniformly sampling d-regular graphs which avoid a set $E(\overline{H_n})$ of forbidden edges. Here $d = d(n)$ is a positive integer which may depend on n.

Two of these algorithms produce a uniformly random d-factor of H_n in expected runtime which is linear in n and low-degree polynomial in d and Δ. The first algorithm applies when $(d+\Delta)d\Delta = o(n)$. This improves on an earlier algorithm by the first author, which required constant d and at most a linear number of edges in $\overline{H_n}$. The second algorithm applies when H_n is regular and $d^2 + \Delta^2 = o(n)$, adapting an approach developed by the first author together with Wormald. The third algorithm is a simplification of the second, and produces an approximately uniform d-factor of H_n in time $O(dn)$. Here the output distribution differs from uniform by $o(1)$ in total variation distance, provided that $d^2 + \Delta^2 = o(n)$.

Mathematics Subject Classifications: 05C80, 68W20, 68Q25

1 Introduction

Enumeration and uniform generation of graphs of given degrees has been an active research area for four decades, with many applications. Research on graph enumeration started in 1978 when Bender and Canfield [3] obtained the asymptotic number of graphs with

[*]Supported by the Australian Research Council Discovery Project DP160100835 and NSERC.
[†]Supported by the Australian Research Council Discovery Project DP140101519.

bounded degrees. The constraint of bounded degrees was slightly relaxed by Bollobás [4] to the order of $\sqrt{\log n}$. A further relaxation on the degree constraints was obtained by McKay [15] to $d = o(n^{1/3})$, and by McKay and Wormald [18] to $d = o(\sqrt{n})$, for the d-regular graph case. On the other hand, asymptotic enumeration for dense d-regular graphs was performed in 1990 by McKay and Wormald [19] for $d, n-d \geqslant n/\log n$, leaving a gap in which no result was known: namely, when $\min\{d, n-d\}$ lies between $n^{1/2}$ and $n/\log n$. This gap was filled recently by Liebenau and Wormald [14]. It is not known whether the problem of counting all graphs with a given degree sequence is #P-complete. Erdős et al. [7] proved that the problem is self-reducible, in the bipartite setting, and hence approximately counting such graphs can be reduced to sampling.

Uniform generation of graphs with given degrees has an equally long history, and is closely related to asymptotic enumeration. Tinhofer [21] is among the first who investigated algorithmic heuristics, and realised that efficient uniform generation, or approximately uniform generation, is difficult. Graph enumeration arguments can sometimes be adapted to provide algorithms for uniform generation of graphs with given degrees. The proofs in [3, 4] immediately yield a simple rejection algorithm, which uniformly generates graphs of degrees at most $O(\sqrt{\log n})$ in expected polynomial time (polynomial as a function of n). The switching arguments in [15, 18] were adapted to give a polynomial-time (in expectation) algorithm [17] for uniformly generating random d-regular graphs when $d = O(n^{1/3})$. However, overcoming the $n^{1/3}$ barrier was extremely challenging, with no progress for more than two decades. A major breakthrough was obtained by Wormald and the first author [9], by modifying the McKay-Wormald algorithm [17] to a more flexible form, which allows the use and classification of different types and classes of switchings. This new technique greatly extended the range of degree sequences for which uniform generation is possible. The first application [9] of the new technique gave an algorithm for uniform generation of d-regular graphs with expected runtime $O(nd^3)$ when $d = o(\sqrt{n})$. The second application [10] gave an algorithm for uniform generation of graphs with power-law degree sequences with exponent slightly below 3, with expected runtime $O(n^{2.107})$ with high probability.

Although uniform generation of random regular graphs has been challenging and remains open for d of order at least $\sqrt{n}$, various approximate samplers have been developed and proven to run in polynomial time. Jerrum and Sinclair gave a MCMC-based scheme [12] which gives polynomial-time sampling of graphs with a given degree sequence, so long as the degree sequence satisfies a condition called P-*stability*. No explicit bound on the runtime was given. Since all regular sequences are P-stable, the Jerrrum–Sinclair algorithm generates d-regular graphs approximately uniformly in polynomial time. Kannan, Tetali and Vempala used another Markov chain to sample random regular bipartite graphs. They proved that the mixing time is polynomial without giving an explicit bound. Later this chain was extended by Cooper, Dyer and Greenhill [5, 6] to generate random regular graphs, with mixing time bounded by $d^{24}n^9 \log n$. There are asymptotically approximate samplers [2, 9, 13, 20, 22] which generate d-regular graphs very fast, typically with linear or up to quadratic runtime, and with an output of total variation distance $o(1)$ from the uniform.

Jerrum and Sinclair's algorithm [12] built on their earlier work on a Markov chain for sampling perfect matchings (1-factors) in a given graph [11]. A natural generalisation of this problem is that of sampling random d-factors in a given graph, which we call the *host graph* and denote by H_n, where n is the number of vertices.

We call $\overline{H_n}$, the complement of H_n, the *forbidden graph.* Let Δ denote the maximum degree of $\overline{H_n}$.

The computational complexity of counting d-factors in a given host graph is not known in general, though the special case $d = 1$ (counting perfect matchings in a given graph) is #P-complete. There are asymptotic enumeration results for graphs with given degrees and a specified set of forbidden edges, see for example [15, 16]. However, there has not been much research in the direction of uniform generation of d-factors. The first author gave a rejection algorithm in [8] which has an expected linear runtime when $d = O(1)$ and $\overline{H_n}$ contains at most a linear number of edges. Here H_n is not necessarily regular, and the maximum degree of $\overline{H_n}$ can be linear. We are not aware of any other algorithms which have explicit polynomial bounds on the runtime and vanishing bounds on the approximation error. For approximate sampling, the Jerrum–Sinclair algorithm [12, Section 4] generates an approximately uniform d-factor of H_n in polynomial time, as long as $d+\Delta \leqslant n/2+1$. It seems unlikely that the approach of Cooper et al [5] can be adapted to the setting of sampling d-factors, due to the complexity of the analysis. Erdős et al. [7] analysed a Markov chain algorithm which uniformly generates bipartite graphs with a given half-regular degree sequence, avoiding a set of edges which is the union of a 1-factor and a star. Here "half-regular" means that the degrees on one side of the bipartition are all the same, with the possible exception of the centre of the star.

The aim of this paper is to develop efficient algorithms that sample d-factors of H_n uniformly or approximately uniformly. We will describe and analyse three different algorithms, which we call `FactorEasy`, `FactorUniform` and `FactorApprox`. Our main focus is `FactorUniform`, which is an algorithm for uniformly generating d-factors of an $(n-1-\Delta)$-regular host graph H_n, when d and Δ are not too large. For smaller d and Δ, the simpler algorithm `FactorEasy` is more efficient, and does not require H_n to be regular (here Δ is the maximum degree of $\overline{H_n}$). Finally, `FactorApprox` is a linear-time algorithm for generating d-factors of H_n asymptotically approximately uniformly, under the same conditions as `FactorUniform`.

Our results are stated formally below. All asymptotics are as the number of vertices n tends to infinity, along even integers if d is odd. Throughout, $d = d(n)$ and $\Delta = \Delta(n)$ are positive integers which may depend on n.

Theorem 1.1. *Let H_n be a graph on n vertices such that the maximum degree of $\overline{H_n}$ is Δ. The algorithm* `FactorEasy` *uniformly generates a d-factor of H_n. If $(d+\Delta)d\Delta = o(n)$ then* `FactorEasy` *runs in time $O((d+\Delta)^3 n)$ in expectation.*

Theorem 1.2. *Let H_n be an $(n-\Delta-1)$-regular graph on n vertices. The algorithm* `FactorUniform` *uniformly generates a d-factor of H_n. If $d^2+\Delta^2 = o(n)$ then the time* *complexity of* `FactorUniform` *is* $O((d+\Delta)^4 n + d^3 n\log n)$ *a.a.s., and is* $O(M)$ *in expectation, where*

$$M = O\Big((d+\Delta)^4(n+\Delta^3) + (d+\Delta)^8 d^2\Delta^2/n + (d+\Delta)^{10}d^2\Delta^3/n^2\Big).$$

Note added in proof: A new technique called *incremental relaxation* has recently been developed by Arman, Wormald and the first author [1]. This technique significantly improves the run time of switching-based algorithms by incrementally performing rejections. It is very likely that the run time of `FactorEasy` and `FactorUniform` can be significantly improved by adapting the incremental relaxation scheme from [1].

Theorem 1.3. *Let* H_n *be an* $(n-\Delta-1)$*-regular graph on* n *vertices. Assume that* $d^2+\Delta^2 = o(n)$*. The algorithm* `FactorApprox` *approximately generates a uniformly random* d*-factor of* H_n *in time* $O(dn)$ *in expectation. The distribution of the output of* `FactorApprox` *differs from uniform by* $o(1)$ *in total variation distance.*

Remark: For simplicity we considered regular spanning subgraphs in this paper, and the host graph is assumed regular for `FactorUniform` and `FactorApprox`. However, all of these algorithms are flexible and can be modified to cope with more general degree sequences for the spanning subgraph and for the host graph. For instance, the McKay-Wormald algorithm [17] can uniformly generate a subgraph of K_n with a given degree sequence where the maximum degree is not too large. We believe that `FactorEasy` can be easily modified for general degree sequences, by calling [17], and then slighly modifying the analysis, with essentially the same switching. To cope with denser irregular subgraphs or sparser irregular host graphs, `FactorUniform` and `FactorApprox` can be modified accordingly, by possibly introducing new types of switchings. We do not pursue this here.

In Section 2, we describe the common framework of `FactorEasy` and `FactorUniform`, and define some key parameters that will appear in these two algorithms and in the approximate sampling algorithm `FactorApprox`. More detail on the structure of the paper can be found at the end of Section 3.1.

2 The framework for uniform generation

The new approach of Gao and Wormald [9] gives a common framework for Algorithms `FactorEasy` and `FactorUniform`, which we will now describe. The Gao–Wormald scheme reduces to the McKay–Wormald algorithm [17] by setting certain parameters to some trivial values. Our algorithm `FactorEasy` is indeed an adaptation of the simpler McKay-Wormald algorithm, whereas `FactorUniform` uses the full power of [9]which allows it to cope with a larger range of d and Δ.

It is convenient to think of the host graph H_n as being defined by a 2-colouring of the complete graph K_n with the colours red and black, where edges of $\overline{H}$ are coloured red, while edges of H_n are coloured black. Then our aim is to uniformly sample d-factors of K_n which contain no red (forbidden) edges.

Both algorithms `FactorEasy` and `FactorUniform` begin by generating a uniformly random d-regular graph G on $\{1, 2, \dots, n\}$. For the range of d which we consider, this can be done using the Gao–Wormald algorithm REG [9]. Typically this initial graph G will contain some red edges. Let $\mathcal{S}_i$ denote the set of all d-regular graphs containing precisely i red edges. The sets $\mathcal{S}_0, \mathcal{S}_1, \dots$ are called *strata*. For some positive integer parameter i_1, which we must define for each algorithm, let

$$\mathcal{A}_0 = \bigcup_{i=0}^{i_1} \mathcal{S}_i.$$

If the initial graph G does not belong to $\mathcal{A}_0$ then the algorithm will reject G and restart. Otherwise, the initial graph G belongs to $\mathcal{A}_0$ and so it does not contain too many red edges. Then the algorithm will perform a sequence of switching operations (which we must define), starting from G, until it reaches a d-regular graph with no red edges. At each switching step there is a chance of a *rejection*: if a rejection occurs then the algorithm will restart. This rejection scheme must also be defined, for each algorithm.

In `FactorEasy`, only one type of switching (Type I) is used. Each such switching reduces the number of red edges by exactly one. As soon as `FactorEasy` reaches a d-regular graph with no red edges, it outputs that graph, provided that no rejection has occurred. When $(d+\Delta)d\Delta$ is of order n or greater, the probability of a rejection occurring in `FactorEasy` is very close to 1 and `FactorEasy` becomes inefficient.

`FactorUniform` reduces the probability of rejection by permitting some switchings that are invalid in `FactorEasy`, as well as introducing other types of switchings. Switchings which typically reduce the number of red edges by exactly one will still be the most frequently applied switchings, although in `FactorUniform` we relax these switchings slightly so that certain operations which were forbidden in `FactorEasy` will be permitted in `FactorUniform`. The other types of switchings do not necessarily reduce the number of red edges. Rather counterintuitively, some switchings will create more red edges. The new types of switchings are introduced for the same reason as the use of the rejection scheme: to remedy the distortion of the distribution which arises by merely applying Type I switchings.

We will specify parameters $\rho_\tau(i)$, for $0 \leqslant i \leqslant i_1$ and $\tau \in \Gamma$, where Γ is the set of the types of switchings to be applied in `FactorUniform`. In each step of `FactorUniform`, ignoring rejections that may occur with a small probability, a switching type $\tau \in \Gamma$ is chosen with probability $\rho_\tau(i)$ if the current graph G is in $\mathcal{S}_i$. Then, given τ, a random switching of type τ is performed. As mentioned before, the Type I switchings are the most common: in fact, we set $\rho_{\mathrm{I}}(i)$ close to 1 for each $0 \leqslant i \leqslant i_1$. If the current graph lies in $\mathcal{S}_0$ then a Type I switching applied to that graph will simply output the graph, but any other type of switching will not. Hence `FactorUniform` does not always immediately produce output as soon as it reaches a graph in $\mathcal{S}_0$, unlike `FactorEasy`.

As a preparation, we compute the expected number of red edges in a random d-regular subgraph of K_n.

Lemma 2.4. *Let G be a uniformly random d-regular graph on $\{1, 2, \dots, n\}$. The expected number of red edges in G is $|E(\overline{H_n})|d/(n-1)$.*

Proof. Let uv be a red edge in K_n (that is, an edge in $\overline{H_n}$). We know that u is incident with d edges in G, and by symmetry each of the $n-1$ edges incident with u in K_n is equally likely to be in G. Thus, the probability that $uv \in G$ is $d/(n-1)$. There are exactly $|E(\overline{H_n})|$ red edges in K_n. By linearity of expectation, the expected number of red edges in G is $|E(\overline{H_n})|d/(n-1)$. □

3 The algorithm `FactorEasy`

Define

$$\begin{aligned} i_1 &= 2|E(\overline{H_n})|d/n; && (3.1)\\ \mathcal{A}_0 &= \cup_{i=0}^{i_1} \mathcal{S}_i. && (3.2) \end{aligned}$$

The following is a direct corollary of Lemma 2.4, using Markov's inequality.

Corollary 3.5. *With probability at least $1/2 + o(1)$, a uniformly random d-regular graph on $\{1, 2, \dots, n\}$ contains at most i_1 red edges.*

We now define the switching operation which we use in `FactorEasy`, called a *3-edge-switching.* To define a 3-edge-switching from the current graph G, choose a sequence of vertices $(v_0, \dots, v_5)$ such that v_0v_1 is a red edge in G, v_2v_3 and v_4v_5 are edges in G (with repetitions allowed), and the choice satisfies the following conditions:

- v_2v_3 and v_4v_5 are black edges in G,
- v_0v_5, v_1v_2 and v_3v_4 are all absent in G, and are all black in K_n;
- The vertices $v_0, \dots, v_5$ are distinct, except that $v_2 = v_5$ is permitted.

We say that the 6-tuple $\boldsymbol{v} = (v_0, v_1, \dots, v_5)$ is *valid* if it satisfies these conditions. Given a valid 6-tuple $\boldsymbol{v}$, the 3-edge-switching determined by $\boldsymbol{v}$ deletes the three edges v_0v_1, v_2v_3, v_4v5 and replaces them with the edges v_1v_2, v_3v_4, v_0v_5, producing a new graph G'. This switching operation is denoted by $(G, \boldsymbol{v}) \mapsto G'$, and is illustrated in Figure 1. Red edges and red non-edges are also labelled 'r', to ensure visibility.

Each *3-edge switching* reduces the number of red edges by exactly one. The inverse operation, obtained by reversing the arrow in Figure 1, is called an *inverse 3-edge switching.* The 6-tuple $(v_0, v_1, \dots, v_5)$ is valid for the inverse 3-edge switching if exactly one new red edge v_0v_1 is introduced, no multiple edges are introduced, and all vertices are distinct except possibly $v_2 = v_5$.

Define

$$\overline{m}(i) = 2i(dn)^2,$$
$$\underline{m}(i) = (2|E(\overline{H_n})| - 2i)d^2(dn - 2i - 8d) - 4|E(\overline{H_n})|d^3(d + \Delta) - 4i\Delta d^2 n.$$

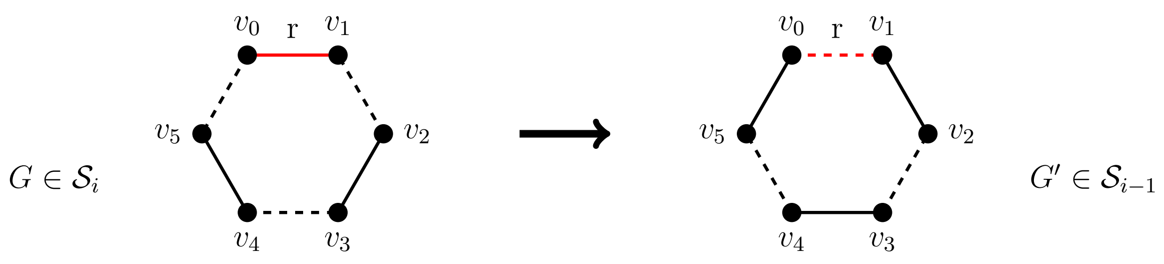


Figure 1: A 3-edge switching

Let $f(G)$ denote the number of valid 6-tuples $(v_0, v_1, \dots, v_5)$ which determine a 3-edge switchings that can be applied to G, and let $b(G)$ denote the number of valid 6-tuples $(v_0, v_1, \dots, v_5)$ which determine a inverse 3-edge switchings that can be applied to G. In the following lemma, we show that $f(G)$ is approximately $\overline{m}(i)$ and $b(G)$ is approximately $\underline{m}(i)$ for $G \in \mathcal{S}_i$.

Lemma 3.6. *Suppose that $(d + \Delta)d\Delta = o(n)$. Let $i \in \{1, \dots, i_1\}$. For all $G \in \mathcal{S}_i$ we have*

$$2i(dn - 2i - 4d)(dn - 2i - 7d) - 6id^2(d + \Delta)n \leqslant f(G) \leqslant \overline{m}(i),$$
$$\underline{m}(i) \leqslant b(G) \leqslant 2|E(\overline{H_n})|d^3n.$$

Hence

$$f(G) = \overline{m}(i)\left(1 + O\left(\frac{d+\Delta}{n}\right)\right), \quad b(G) = 2|E(\overline{H_n})|d^3n\left(1 + O\left(\frac{d+\Delta}{n}\right)\right).$$

Proof. For the upper bound of $f(G)$, note that there are exactly $2i$ ways to choose (v_0, v_1), and then at most dn ways to choose (v_2, v_3) and at most dn ways to choose (v_4, v_5).

For the lower bound of $f(G)$, there are $2i$ ways to choose (v_0, v_1). Given that, there are at least $(dn - 2i - 4d)$ ways to choose (v_2, v_3) such that v_2v_3 is a black edge in G, and v_0, v_1, v_2 and v_3 are all distinct. Then there are at least $(dn - 2i - 7d)$ ways to choose (v_4, v_5) such that v_4v_5 is a black edge in G and

$$v_4 \notin \{v_0, v_1, v_2, v_3\}, \qquad v_5 \notin \{v_0, v_1, v_3\}$$

(as v_5 is allowed to coincide with v_2). This gives at least $2i(dn-2i-4d)(dn-2i-7d)$ choices of $(v_0, \dots, v_5)$, but some of these choices are not valid: specifically, we must subtract the number of choices such that one of v_0v_5, v_1v_2 or v_3v_4 is either an edge in G (either black or red), or is a red non-edge (that is, an edge in $\overline{H_n} \cap \overline{G}$).

- There are at most $3 \cdot 2id^2dn$ choices such that v_0v_5 or v_1v_2 or v_3v_4 is an edge in G;
- There are at most $3 \cdot 2idn\Delta d$ choices such that v_0v_5 or v_1v_2 or v_3v_4 is a red non-edge.

Subtracting these yields the desired lower bound for $f(G)$.

Next we consider $b(G)$. For the upper bound, there are at most $2|E(\overline{H})|$ ways to choose (v_0, v_1), since $|E(\overline{H})|$ is an upper bound on the number of red non-edges in G. Then there are at most d ways to choose v_2, at most d ways to choose v_5 and at most dn ways to choose (v_3, v_4). This yields the required upper bound.

For the lower bound, there are exactly $2(|E(\overline{H})| - i)$ ways to choose (v_0, v_1), as the number of red non-edges in G is exactly $|E(\overline{H})| - i$. Then there are exactly d^2 ways to choose v_2 and v_5 such that v_0v_5 and v_1v_2 are edges in G (not necessarily black). Note here that $v_2 = v_5$ is permitted. The number of ways to choose (v_3, v_4) such that v_3v_4 is a black edge in G, and $\{v_3, v_4\} \cap \{v_0, v_1, v_2, v_5\} = \varnothing$ is at least $dn - 2i - 8d$. This gives at least $2(|E(\overline{H})| - i)d^2(dn - 2i - 8d)$ choices for $(v_0, \ldots, v_5)$, but some of these choices are not valid: specifically, we must subtract the number of choices where one of v_2v_3, v_4v_5 is either an edge in G or a red non-edge, or one of v_0v_5, v_1v_2 is a red edge.

- There are at most $2 \cdot 2|E(\overline{H})|d^3(d + \Delta)$ choices such that v_2v_3 or v_4v_5 is either an edge in G or a red non-edge;
- There are at most $2 \cdot 2i\Delta d \cdot dn$ choices such that v_0v_5 or v_1v_2 is a red edge.

Subtracting these gives the stated lower bound for $b(G)$.

Finally, the last statement follows immediately from the above bounds by (3.1) and noting that $|E(\overline{H_n})| \leqslant \Delta n/2$. □

3.1 Algorithm `FactorEasy`: definition and analysis

First, `FactorEasy` calls the Gao–Wormald algorithm REG [9] to generate a uniformly random d-regular graph G on $\{1, 2, \ldots, n\}$. If G contains more than i_1 red edges then `FactorEasy` restarts. Otherwise, `FactorEasy` iteratively performs a sequence of switching steps. At each step, there is a chance that the current graph, G, might be rejected (this is called *f-rejection*) and there is a chance that the graph G' selected as the "next" graph might be rejected (this is called *b-rejection*). Here "f" is short for "forward" and "b" is short for "backward". The probability of f-rejection and b-rejection is carefully chosen to maintain uniformity.

Let G_t be the graph obtained after t switching steps, and assume $G_t = G \in \mathcal{S}_i$. The $(t+1)$-th switching step is composed of the following substeps:

(i) If $i = 0$ then output G.

(ii) If $i > 0$ then uniformly at random choose a red edge in G and choose two further edges in G (of any colour), with repetition allowed. Randomly label the two endvertices of the red edge as v_0 and v_1, and the endvertices of the other two edges as v_2 and v_3, and v_4 and v_5 respectively. If $\boldsymbol{v} = (v_0, \ldots, v_5)$ is a valid 6-tuple then let $(G, \boldsymbol{v}) \mapsto G'$ be the 3-edge switching induced by this 6-tuple, as in Figure 1. Otherwise (when the 6-tuple is not valid), perform an f-rejection.

(iii) If no f-rejection is performed then perform a b-rejection with probability

$$1 - \frac{b(G')}{\underline{m}(i-1)}.$$

(iv) If no b-rejection is performed then set $G_{t+1} = G'$.

If any rejection occurs then `FactorEasy` restarts.

There is no deterministic upper bound on the running time of the algorithm, due to the chance of rejections. That is, `FactorEasy` is a *Las Vegas* algorithm. But to prove Theorem 1.1 we will show that the expected number of restarts is $O(1)$.

The proof of the following lemma is deferred to Section 6.

Lemma 3.7. `FactorEasy` *can be implemented so that if there are $O(1)$ restarts during its run, its time complexity is $O((d+\Delta)^3 n)$.*

We now prove Theorem 1.1, restated below for convenience.

Theorem 1.1. *Let H_n be a graph on n vertices such that the maximum degree of $\overline{H_n}$ is Δ. The algorithm* `FactorEasy` *uniformly generates a d-factor of H_n. If $(d+\Delta)d\Delta = o(n)$ then* `FactorEasy` *runs in time $O((d+\Delta)^3 n)$ in expectation.*

Proof. First we prove uniformity by induction. Recall that G_0 is a uniformly random d-regular graph in $\mathcal{A}_0$ if no initial rejection occurs. If $G_0 \in \mathcal{S}_i$ then clearly G_0 is uniformly distributed over $\mathcal{S}_i$.

Next we prove that if G_t is uniformly distributed over $\mathcal{S}_i$ then G_{t+1} is uniformly distributed over $\mathcal{S}_{i-1}$, assuming that no rejection occurs. For every $G \in \mathcal{S}_i$ and $G' \in \mathcal{S}_{i-1}$, let $\Psi(G, G')$ denote the set of valid 6-tuples $\boldsymbol{v} = (v_0, \ldots, v_5)$ such that $(G, \boldsymbol{v}) \mapsto G'$ is a 3-edge-switching, and let

$$\Psi(G') = \bigcup_{G \in \mathcal{S}_i} \Psi(G, G'). \tag{3.3}$$

Given $G_t = G$, note that $\overline{m}(i)$ is exactly the number of choices of a 6-tuple of vertices $(v_0, \ldots, v_5)$, with repetition allowed, such that v_0v_1 is a red edge in G, and v_2v_3 and v_4v_5 are edges of G. Thus, the probability that G is converted to G' without f-rejection in step $t+1$ is equal to

$$\frac{|\Psi(G, G')|}{\overline{m}(i)}.$$

The probability that no b-rejection occurs is equal to $\underline{m}(i-1)/b(G') = \underline{m}(i-1)/|\Psi(G')|$. Write $\rho_t = \mathbb{P}(G_t = G \mid G_t \in \mathcal{S}_i)$, which is invariant over all graphs $G \in \mathcal{S}_i$ by the inductive hypothesis. Then

$$\begin{aligned}\mathbb{P}(G_{t+1} = G' \mid G_t \in \mathcal{S}_i) &= \sum_{G \in \mathcal{S}_i} \mathbb{P}(G_t = G \mid G_t \in \mathcal{S}_i) \cdot \frac{|\Psi(G, G')|}{\overline{m}(i)} \cdot \frac{\underline{m}(i-1)}{|\Psi(G')|} \\ &= \rho_t \frac{\underline{m}(i-1)}{\overline{m}(i)},\end{aligned}$$

using the fact that the union in (3.3) is disjoint. Hence $\mathbb{P}(G_{t+1} = G' \mid G_t \in \mathcal{S}_i)$ does not depend on G', which implies that G_{t+1} is uniformly distributed over $\mathcal{S}_{j-1}$ if no rejection occurs.

Next, we prove that if $(d+\Delta)d\Delta = o(n)$ then `FactorEasy` runs in $O((d+\Delta)^3 n)$ time in expectation. The runtime for generating a random d-regular graph on $\{1, 2, \ldots, n\}$ using the Gao–Wormald algorithm [9] is $O(d^3 n)$ in expectation. By Lemma 3.6, the probability of an f-rejection or a b-rejection in each step is $O((d + \Delta)/n)$. By definition of i_1 and $\mathcal{A}_0$, the algorithm `FactorEasy` performs at most $i_1 = O(|E(\overline{H_n})|d/n) = O(d\Delta)$ switching steps. Thus, the overall probability of any f-rejection or b-rejection is at most

$$O\left(\frac{d+\Delta}{n}\right) i_1 = O\left(\frac{(d+\Delta)d\Delta}{n}\right)$$

which is $o(1)$ because $(d + \Delta)d\Delta = o(n)$. It follows from this and from Corollary 3.5, that in expectation `FactorEasy` restarts $O(1)$ times. Therefore, by Lemma 3.7, the time complexity of `FactorEasy` is $O((d + \Delta)^3 n)$ in expectation. □

We close this section by proving the following lemma, which follows easily from Lemma 3.6. This result, which will be useful later, only requires a rather weak condition on d and Δ.

Lemma 3.8. *Assume that $d + \Delta = o(n)$. Then for any $i = O(d\Delta)$,*

$$\frac{|\mathcal{S}_{i-1}|}{|\mathcal{S}_i|} = \frac{i\,n}{|E(\overline{H_n})|d}\left(1 + O\left(\frac{d+\Delta}{n}\right)\right).$$

Proof. Let $\mathbb{E}\, f(G)$ be the expected value of $f(G)$ when $G \in \mathcal{S}_i$ is chosen uniformly at random, and let $\mathbb{E}\, b(G')$ be the expected value of $b(G')$ when $G' \in \mathcal{S}_{i-1}$ is chosen uniformly at random. Then

$$\frac{|\mathcal{S}_{i-1}|}{|\mathcal{S}_i|} = \frac{\mathbb{E}\, f(G)}{\mathbb{E}\, b(G')}$$

and the result follows by Lemma 3.6. □

When $(d + \Delta)d\Delta$ is no longer negligible compared to n, the probability that an f-rejection or b-rejection occurs in `FactorEasy` before reaching $\mathcal{S}_0$ is very close to 1. In this case, `FactorEasy` becomes very inefficient as it has to restart many times. In Section 5 we define `FactorUniform`, which will use 4-edge switchings instead of 3-edge switchings, giving more room for performing valid operations. In addition, various new ideas will be incorporated into the design of `FactorUniform` to achieve uniformity in the output and efficiency when $d^2+\Delta^2 = o(n)$. Since the uniform sampler `FactorUniform` can be treated as an extension of the approximate sampler `FactorApprox`, we will introduce `FactorApprox` first, in Section 4 below. The runtime analysis of `FactorEasy` and `FactorUniform`, and the proof that the output of `FactorApprox` is sufficiently close to uniform, are deferred to Section 6.

For the algorithms `FactorUniform` and `FactorApprox`, we restrict the host graph H_n to be regular (specifically, $(n-\Delta-1)$-regular), to simplify the analysis. However, we believe that `FactorUniform` (and `FactorApprox`) can be modified to work for irregular host graphs H_n.

4 The approximate algorithm: `FactorApprox`

We now assume that H_n is $(n-\Delta-1)$-regular, which implies that $|E(\overline{H_n})| = \Delta n/2$. Define

$$i_1 = \frac{2}{3} d\Delta; \tag{4.1}$$

$$\mathcal{A}_0 = \cup_{i=0}^{i_1} \mathcal{S}_i. \tag{4.2}$$

Corollary 4.9. *With probability at least* $1/4 + o(1)$*, a uniformly random* d*-regular graph on* $\{1, 2, \ldots, n\}$ *contains at most* i_1 *red edges.*

Proof. By Lemma 2.4, the expected number of red edges in a random d-factor of G is asymptotic to $\Delta d/2$. The result follows by Markov's inequality. □

Gao and Wormald [9] gave an algorithm called REG* for generating regular graphs asymptotically approximately uniformly in runtime $O(dn)$. `FactorApprox` uses REG* to generate a random d-regular graph on K_n.

Corollary 4.10. *With probability at least* $1/4 + o(1)$*, the output of REG* contains at most* i_1 *red edges.*

Proof. When $d^2 = o(n)$, the distribution of the output of REG* differs from the uniform by $o(1)$ in total variation distance, see [9, Section 10]. Combining this with Corollary 4.9 completes the proof. □

Both `FactorUniform` and `FactorApprox` use *4-edge-switchings*, which we now define. To define a 4-edge-switching from the current graph G, choose a sequence of 8 vertices $(v_0, v_1, \ldots, v_7)$ such that v_0v_1 is a red edge in G and v_2v_3, v_4v_5, v_6v_7 are other edges in G, and such that

- v_0v_7, v_1v_2, v_3v_4, v_5v_6 are *not present* in G;
- none of v_3v_4, v_4v_5, v_5v_6 is red in K_n;
- no two of the eight vertices are equal except for possibly $v_2 = v_7$;
- Either none of v_1v_2, v_2v_3, v_0v_7 and v_6v_7 is red;
  or exactly one of v_1v_2, v_2v_3, v_0v_7 and v_6v_7 is red;
  or both v_1v_2 and v_2v_3 are red, and both v_0v_7 and v_6v_7 are black;
  or both v_1v_2 and v_2v_3 are black, and both v_0v_7 and v_6v_7 are red.

The sequence of vertices $(v_0, \dots, v_7)$ is said to be *valid* if it satisfies the above properties. Given a valid 8-tuple of vertices, the switching operation deletes the four edges v_0v_1, v_2v_3, v_4v_5, v_6v_7, and replaces them with the edges v_0v_7, v_1v_2, v_3v_4 and v_5v_6, as in Figure 2. The resulting graph is denoted by G'. This operation is called a 4-edge-switching. We say that the 4-edge-switching is *valid* if it arises from a valid 8-tuple.

Under the switching, the colour of each edge or non-edge stays the same, but edges become non-edges and vice-versa. In Figure 2, a solid line indicates an edge of G and a dashed line represents an edge of $\overline{G}$; that is, a non-edge in G_t. We label each edge (or non-edge) by the allowed colour, where 'b', 'r' and 'b/r' denote 'black', 'red', and 'black or red', respectively. Edges or non-edges which are definitely red are also shown coloured red in the figure.

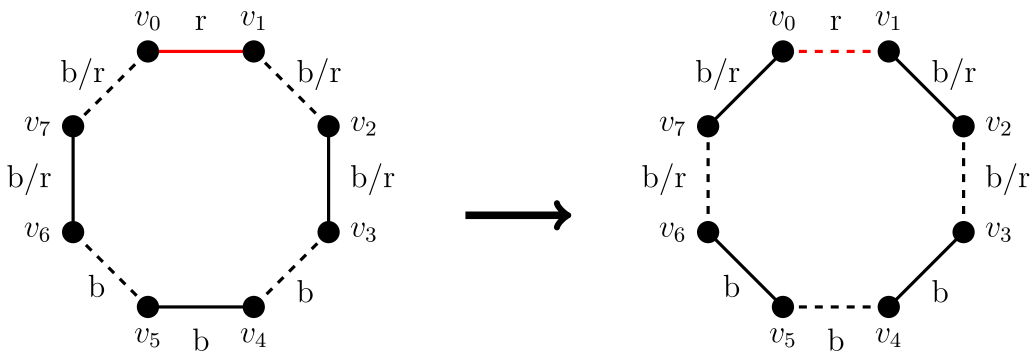


Figure 2: A 4-edge-switching

The structure of `FactorApprox` is similar to that of `FactorEasy`, except that there is no rejection after the first step.

First, `FactorApprox` uses REG* to repeatedly generate a random d-regular graph on the vertex set $\{1, 2, \dots, n\}$ until the resulting graph belongs to $\mathcal{A}_0$. Next, `FactorApprox` repeatedly applies random 4-edge switchings from the current graph, until a graph without red edges is produced. Finally, `FactorApprox` outputs this graph, which is a d-factor of H_n.

To choose a uniformly random 4-edge switching from a current graph $G = G_t$, we can uniformly at random choose a red edge and label its end vertices by v_0 and v_1. Then uniformly choose three edges of G (repetition allowed) and label their endvertices. If the resulting 8-tuple is valid then perform the corresponding 4-edge-switching to produce the new graph $G' = G_{t+1}$. Otherwise, repeat until a valid 4-edge-switching is obtained.

The proof of the following lemma is given in Section 6.

Lemma 4.11. *Under the conditions of Theorem 1.3, the output of* `FactorApprox` *is a random d-factor of H_n whose distribution differs from the uniform distribution by $o(1)$ in total variation distance.*

Using this lemma, we can prove Theorem 1.3, restated here for convenience.

Theorem 1.3. *Let H_n be an $(n-\Delta-1)$-regular graph on n vertices. Assume that $d^2+\Delta^2 = o(n)$. The algorithm* `FactorApprox` *approximately generates a uniformly random d-factor of H_n in time $O(dn)$ in expectation. The distribution of the output of* `FactorApprox` *differs from uniform by $o(1)$ in total variation distance.*

Proof. The runtime of REG* is $O(dn)$ and by Corollary 4.10, a constant number of attempts will be sufficient to generate a random d-regular graph with at most i_1 red edges in expectation. Now a given graph G, consider choosing v_0v_1 to be a uniformly random red edge of G and choosing each of v_2v_3, v_4v_5, v_6v_7 to be a uniformly random edge of G (with repetition allowed). It is easy to see that with high probability, this random choice of edges v_0v_1, v_2v_3, v_4v_5, v_6v_7 defines a valid 4-edge switching which reduces the number of red edges by exactly one. (The argument is very similar to the proof of Lemma 5.22.) Hence, the cost of time in performing one 4-edge switching is $O(1)$, and `FactorApprox` consists of performing $O(i_1)$ switching steps in expectation and with high probability. Since $i_1 = O(d\Delta)$, it follows that the runtime of `FactorApprox` is $O(dn + i_1) = O(dn)$. This completes the proof, by Lemma 4.11. □

5 The exactly uniform sampler: `FactorUniform`

In this section, our aim is to define and analyse an algorithm for uniform generation of d-factors of H_n, which is efficient for larger values of d and Δ than `FactorEasy`. Specifically, we assume that $d^2 + \Delta^2 = o(n)$. As in Section 4, we assume that H_n is $(n-\Delta-1)$-regular and define i_1 and $\mathcal{A}_0$ as in (4.1), (4.2).

The analysis of `FactorEasy` given in Section 3.1 shows that the probability of an f-rejection or a b-rejection depends on the gap between the upper and lower bounds of $f(G)$ and $b(G)$. To obtain an algorithm which is still efficient for larger values of d and Δ, we must reduce the variation of $f(G)$ and $b(G)$ among $G \in \mathcal{S}_i$, for some family of switchings. We use several techniques to reduce this variation:

- We use 4-edge-switchings instead of 3-edge switchings.
- We will perform more careful counting than the analysis of Lemma 3.6.
- We will occasionally allow switchings which create new red edges.
- We will introduce other types of switchings to "boost" the probability of graphs which are otherwise not created sufficiently often. These types of switchings are called *boosters*.

The last two of these techniques follow the approach of [9]. In particular, every switching introduced will have a *type* and a *class*. The number of switchings of type τ that can be performed on a given graph G is denoted by $f_\tau(G)$, and the number of switchings of class

α that can be applied to other graphs to produce G is denoted by $b_\alpha(G)$. We will also need parameters $\overline{m}_\tau(i)$ and $\underline{m}_\alpha(i)$ which satisfy

$$\overline{m}_\tau(i) \geqslant \max_{G\in\mathcal{S}_i} f_\tau(G), \qquad \underline{m}_\alpha(i) \leqslant \min_{G\in\mathcal{S}_i} b_\alpha(G).$$

Further details of the structure of algorithm `FactorUniform` will be explained in Section 5.3.

5.1 Type I switchings

`FactorUniform` will mainly use the 4-edge-switching shown in Figure 2, which replaces four edges by four new edges. We will call this the *Type I* switching, as we will define other types of switchings for use in `FactorUniform` later.

Suppose that a Type I switching transforms a graph G into a graph $G' \in \mathcal{S}_i$. Then the initial graph G can be in different strata, depending on the colour of v_1v_2, v_2v_3, v_6v_7 and v_0v_7. If these edges are all black then we say that the Type I switching is in *Class A*. In this case, G has exactly one more red edge than G'. Other Type I switchings are categorised into different classes, as shown in Table 1.

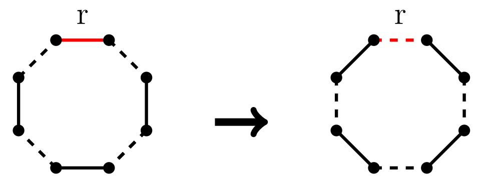
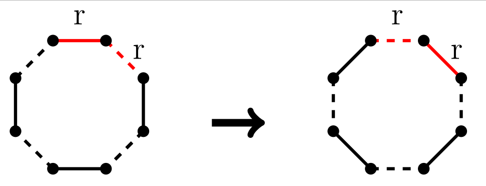
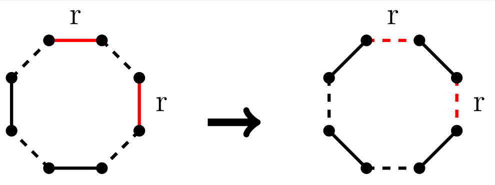
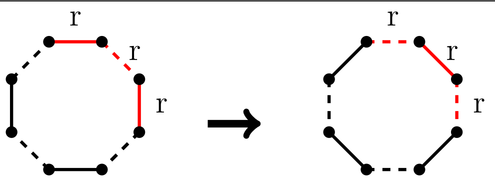
class	action	the switching
A	$\mathcal{S}_{i+1} \to \mathcal{S}_i$	
B1±	$\mathcal{S}_i \to \mathcal{S}_i$	
B2±	$\mathcal{S}_{i+2} \to \mathcal{S}_i$	
C±	$\mathcal{S}_{i+1} \to \mathcal{S}_i$	

Table 1: The different classes of Type I switchings

Each row of Table 1, other than the first row, defines two new classes of switching, depending on how the vertices are labelled. For example, the second row defines Classes B1+ and B1−, which we refer to collectively as B1±. Every class with a name ending "+" arises from using the vertex labelling shown on the left of Figure 3, while those classes ending in "−" arise from using the vertex labelling shown on the right of Figure 3. For example, if v_1v_2 (respectively, v_0v_7) is red but not present in G, and all the other edges and non-edges involved in the switching, except for v_0v_1, are black, then this switching is in Class B1+ (respectively, B1−) and both G and G' belong to $\mathcal{S}_i$.

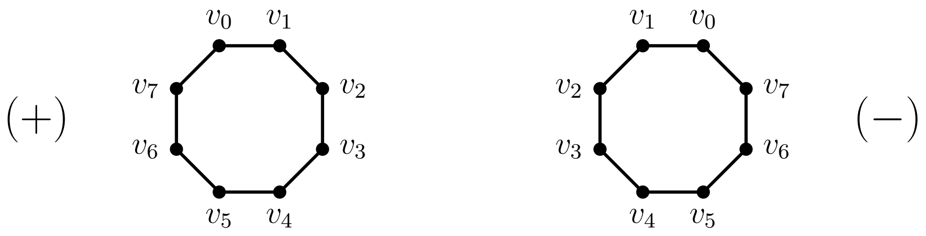


Figure 3: The vertex labellings for "+" and "−" classes, respectively

Next we bound the number of Type I switchings which can be performed in an arbitrary $G \in \mathcal{S}_i$. Define

$$\begin{aligned}&\overline{m}_{\mathrm{I}}(i)\\ &= 2i(dn)^3\left(1 + 28\left(\frac{(\Delta+d)^2}{n^2} + \frac{1}{n}\right)\right) - 8i(d-1)^2d^2n^2 - 4i\Delta d^3n^2 - 4i^2(dn)^2. \qquad (5.1)\end{aligned}$$

The proof of the following lemma is similar to that of Lemma 3.6.

Lemma 5.12. *Suppose that $i \in \{1, \dots, i_1\}$. Then for any $G \in \mathcal{S}_i$,*

$$\begin{aligned} f_{\mathrm{I}}(G) &\leqslant \overline{m}_I(i), &(5.2)\\ f_{\mathrm{I}}(G) &= \overline{m}_I(i)(1 + O((d^2+\Delta^2)/n^2 + 1/n)). &(5.3)\end{aligned}$$

Proof. Recall that $f_{\mathrm{I}}(G)$ is the number of ways that a Type I switching can be applied to a given graph $G \in \mathcal{S}_i$. There are

$$2i(dn - O(1))^3 \leqslant 2i(dn)^3$$

ways to choose the 8-tuple of vertices $(v_0, \dots, v_7)$ so that v_0v_1 is a red edge in G, and v_2v_3, v_4v_5, v_6v_7 are all edges in G, and these four edges are distinct. From this we will subtract the following terms:

- Those with an unwanted vertex coincidence, of which there are $O(id^3n^2)$. We will ignore these cases for the upper bound (5.2), but include them in the error term in (5.3).

- Those in which one of the dashed edges v_1v_2, v_3v_4, v_5v_6, v_0v_7 is present, either black or red, in G. Using inclusion-exclusion, there are at least

$$4 \times 2i(d-1)^2(dn-6)(dn-8) - X$$

of these, where X accounts for choices where at least two of them are present. It is easy to see that $X \leqslant \binom{4}{2} \cdot 2id^5n = 12id^5n$. So this expression is bounded below by

$$8i(d-1)^2d^2n^2 - 2i(dn)^3\left(56/n^2 + 6d^2/n^2\right).$$

(Here, and below, we treat some negligible terms as errors relative to the main term.)

- Those for which at least one of v_3v_4, v_5v_6 or v_4v_5 is red. Suppose first that v_3v_4 is a red non-edge. There are $2i$ ways to choose v_0v_1, then $\Delta n - 2i$ ways to choose v_3v_4 to be red and d^2 ways to choose v_2 and v_5, then at least $dn-6$ ways to choose v_6v_7 to be distinct from all chosen edges. The number of choices where v_5v_6 is a red non-edge is identical, and the number of choices such that both v_3v_4 and v_5v_6 are red non-edges is at most $2id^3\Delta^2n$. There are at most $16id^4\Delta n$ choices of 8-tuple such that one of v_3v_4, v_5v_6 is a red non-edge and one of the dashed edges is present in G. This gives the expression

$$\begin{aligned}&4i(\Delta n - 2i)d^2(dn-6) - 2id^3\Delta^2 n - 16id^4\Delta n\\ &\geqslant 4i\Delta d^3n^2 - 8i^2d^3n - 24i\Delta d^2n - 2id^3\Delta^2n - 16id^4\Delta n\\ &= 4i\Delta d^3n^2 - 2i(dn)^3\left(4i/n^2 + 12\Delta/(dn^2) + \Delta^2/n^2 + 8d\Delta/n^2\right)\\ &\geqslant 4i\Delta d^3n^2 - 2i(dn)^3\left(11d\Delta/n^2 + \Delta^2/n^2 + 12/n\right)\end{aligned}$$

since $i \leqslant i_1 = \frac{2}{3}d\Delta$ and $\Delta < n$.

Continuing, the number of choices of 8-tuple such that v_4v_5 is a red edge is $2i(2i-2)(dn-4)(dn-6)$. From this we remove those where also one of the four dashed edges is present (at most $16i^2d^3n$ choices) or where also one of v_3v_4 or v_5v_6 is a red non-edge (at most $8i^2\Delta d^2n$ choices). This gives

$$\begin{aligned}&4i^2(dn)^2 - 40i^2dn - 4i(dn)^2 - 16i^2d^3n - 8i^2\Delta d^2n\\ &= 4i^2(dn)^2 - 2i(dn)^3\left(20i/(d^2n^2) + 2/(dn) + 8i/n^2 + 4i\Delta/(dn^2)\right)\\ &\geqslant 4i^2(dn)^2 - 2i(dn)^3\left(13\Delta^2/n^2 + 2/n + 6d\Delta/n^2\right).\end{aligned}$$

- As a final adjustment, which will appear with positive sign in the inclusion-exclusion, we must consider choices where one of v_1v_2, v_2v_3 is red and one of v_6v_7, v_7v_0 is red: there are at most $2idn(2i+\Delta d)^2 \leqslant 2i(dn)^3 \cdot 9\Delta^2/n^2$ of these, since $i \leqslant i_1$.

Putting this together, we have proved that (5.3) holds, and that

$$\begin{aligned}
&f_{\mathrm{I}}(G)\\
&\leqslant 2i(dn)^3\left(1+\frac{56}{n^2}+\frac{14}{n}+\frac{14\Delta^2}{n^2}+\frac{8d\Delta}{n}+\frac{6d^2}{n^2}\right)-8i(d-1)^2d^2n^2-4i\Delta d^3n^2-4i^2(dn)^2\\
&\leqslant 2i(dn)^3\left(1+28\left(\frac{(\Delta+d)^2}{n^2}+\frac{1}{n}\right)\right)-8i(d-1)^2d^2n^2-4i\Delta d^3n^2-4i^2(dn)^2\\
&=\overline{m}_{\mathrm{I}}(i).
\end{aligned}$$

This completes the proof. □

Remark 5.13. *If we did not allow the creation of structures in classes B1±, B2± and C± then the f-rejection probability would be too big. For instance, suppose that we did not allow Class B1+ (and that the other Class B and C switchings are permitted). Then for most graphs in $\mathcal{S}_i$, the typical number of forward switchings would be approximately*

$$2i(dn)^3-8i(d-1)^2(dn)^2-4i\Delta d^3n^2-4i^2(dn)^2-2i\Delta d(dn)^2.$$

(This is approximately $\overline{m}_I(i)$ with the term $2i\Delta d(dn)^2$ subtracted, which is the typical number of 8-tuples corresponding to a Class B1+ switching.) However, consider an extreme "worst case", when $d=\Delta$ and d divides $2i$, and the d-factor G is composed of two d-regular graphs, one with only red edges and the other with only black edges. Then there are no red edges in G that are incident with a red dashed edge, so no Class B1+ switchings need to be ruled out (as none are possible). In this case, the number of forward switchings in G is

$$2i(dn)^3-8i(d-1)^2(dn)^2-4i\Delta d^3n^2-4i^2(dn)^2.$$

Hence, $f_I(G)$ differs from $\max_{G\in\mathcal{S}_i} f_I(G)$ by $\Omega(i\Delta d(dn)^2)$ for most graphs $G\in\mathcal{S}_i$, causing an f-rejection probability of $\Omega(i\Delta d(dn)^2/i(dn)^3)=\Omega(\Delta/n)$ in a single step. But then the overall rejection probability will be too big, since there will be up to $i_1=\Theta(d\Delta)$ steps, and $d\Delta^2/n$ may not be $o(1)$ under the conditions of Theorem 1.2. To reduce the probability of an f-rejection we must allow these Class B1+ switchings to proceed (and Class B1− switchings too, by symmetry). Similar arguments explain the introduction of Classes B2± and C±.

5.2 New switching types and counting inverse switchings

For each $\alpha\in\{\mathrm{A},\mathrm{B1}\pm,\mathrm{B2}\pm,\mathrm{C}\pm\}$ we count the inverse switchings of Class α.

5.2.1 Class A

Class A switchings are all of Type I. In this section we will obtain a lower bound for $b_A(G)$ and an upper bound for the average of $b_A(G)$ over all $G\in\mathcal{S}_i$. The following lemma will be useful.

Lemma 5.14. *Assume that $d + \Delta = o(n)$ and $i \leqslant i_1$. Let G be a d-factor chosen uniformly at random from $\mathcal{S}_i$. Then the expected number of red 2-paths in G is $O(i^2/n)$ and the expected number of pairs of red edges $\{\{u_1, v_1\}, \{u_2, v_2\}\}$ in G such that u_1u_2 is either a red edge in G or a red dashed edge is $O(i^2\Delta/n)$.*

Proof. Let uvw be a red 2-path in K_n. We now bound the probability that uvw is contained in a random $G \in \mathcal{S}_i$. Let $\mathcal{W}$ denote the set of graphs in $\mathcal{S}_i$ which contain uvw and let $\mathcal{W}'$ denote the set of graphs in $\mathcal{S}_{i-2}$ which do not contain neither of uv, vw. Consider the switching as shown in Figure 4, where the 7 vertices must be distinct and all edges shown in the figure other than uv, vw must be black.

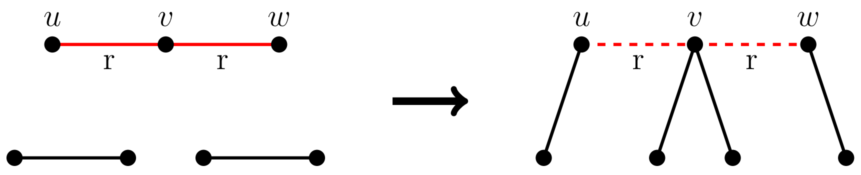


Figure 4: A switching for red 2-paths

Such a switching switches a graph in $\mathcal{W}$ to $\mathcal{W}'$. It is easy to see that the number of forward switchings is at least $(dn)^2(1 - O((d + \Delta)/n)) = \Omega((dn)^2)$, whereas the number of inverse switchings is at most d^4. Hence,

$$\frac{|\mathcal{W}|}{|\mathcal{W}'|} = O(d^4/(dn)^2) = O(d^2/n^2).$$

Moreover, $|\mathcal{W}'| \leqslant |\mathcal{S}_{i-2}|$ as $\mathcal{W}' \subseteq \mathcal{S}_{i-2}$ and $|\mathcal{S}_{i-2}|/|\mathcal{S}_i| = O(i^2/(d\Delta)^2)$ by Lemma 3.8. Thus,

$$\text{Prob}(uvw \in G) = \frac{|\mathcal{W}|}{|\mathcal{S}_i|} = \frac{|\mathcal{W}|}{|\mathcal{W}'|} \cdot \frac{|\mathcal{W}'|}{|\mathcal{S}_{i-2}|} \cdot \frac{|\mathcal{S}_{i-2}|}{|\mathcal{S}_i|} = O\left(\frac{d^2}{n^2}\frac{i^2}{(d\Delta)^2}\right) = O(i^2/\Delta^2n^2).$$

The total number of red 2-paths in K_n is $O(\Delta^2 n)$. Thus, by linearity of expectation, the expected number of red 2-paths contained in a random $G \in \mathcal{S}_i$ is

$$O(\Delta^2 n) \cdot O(i^2/\Delta^2 n^2) = O(i^2/n).$$

The proof for the second claim is similar. Fix a red 3-path $v_1u_1u_2v_2$ in K_n. Let G be a uniformly random graph in $\mathcal{S}_i$. Using another switching (see Figure 5) and a similar argument as above, we can bound the probability that u_1v_1 and u_2v_2 are edges in G by $O(i^2/\Delta^2n^2)$. The red dotted edge marked "?" denotes a red edge which may be either present or absent in G.

The total number of choices for u_1, u_2, v_1, v_2 in K_n is $O(\Delta^3 n)$. Thus, by linearity of expectation, the expected number of pairs of red edges $\{u_1, u_2\}$ and $\{v_1, v_2\}$ as in the lemma is $O(\Delta^3 n) \cdot O(i^2/\Delta^2 n^2) = O(i^2\Delta/n)$. $\square$

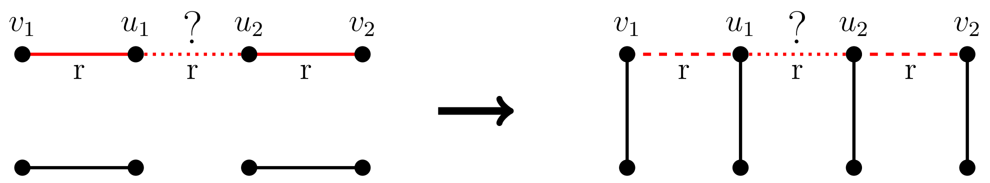


Figure 5: A switching for red edges joined by a red edge or red non-edge

Define

$$\underline{m}_A(i) = (\Delta n - 2i)d^2(dn)^2\left(1 - \frac{30}{n}\right) - 3d^5\Delta n^2 - 8i\Delta d^3 n^2 - 3\Delta^2 d^4 n^2. \tag{5.4}$$

Lemma 5.15. *For any $G \in \mathcal{S}_i$ we have $b_A(G) \geqslant \underline{m}_A(i)$. Furthermore, if G is chosen uniformly at random from $\mathcal{S}_i$ then*

$$\mathbb{E}b_A(G) = \underline{m}_A(i)(1 + O((d+\Delta)^2/n^2 + 1/n)).$$

Proof. Firstly, note that all Class A switchings are of Type I. (See Table 4.) Thus we only need to count inverse Type I Class A switchings.

Consider the right hand side of Figure 2. First we find a lower bound for the number of ways to select an 8-tuple $(v_0, \ldots, v_7)$ such that v_0v_1 is a red non-edge, v_1v_2, v_3v_4, v_5v_6 are all edges and v_2v_3, v_4v_5, v_6v_7 are non-edges, with all vertices distinct except possibly v_2 and v_7. There are $(\Delta n - 2i)$ choices for (v_0, v_1), then d^2 choices for (v_2, v_7), then $dn - 4$ choices for v_2v_3 avoiding the two chosen edges, and then $dn - 6$ choices for v_4v_5 avoiding the three chosen edges. This gives the expression $(\Delta n - 2i)d^2(dn - 4)(dn - 6)$.

For the lower bound, we must subtract from this expression the number of choices of 8-tuple with at least one defect. The possible defects are: vertex coincidence; a dashed edge (other than v_0v_1) is present in G; a dashed edge (other than v_0v_1) is a red non-edge; or a chosen edge is a red edge in G. We now give upper bounds on the number of choices with particular defects.

- Vertex coincidences: Out of $\binom{8}{2} = 28$ possible vertex coincidences, one is allowed and 7 are impossible, leaving 20 vertex coincidences that must be explicitly ruled out: at most $O(d^4\Delta n^2)$ choices.
- One of the dashed edges (other than v_0v_1) is present in G: at most

  $$3d^2(d-1)^2(\Delta n - 2i)(dn - 8) \leqslant 3d^5\Delta n^2$$

  choices.
- v_1v_2 is red, or v_0v_7 is red: at most $2 \cdot 2i\Delta d(dn)^2$ choices. For later use, we remark that this upper bound includes cases where the edge v_0v_1 is red and present in G.
- v_3v_4 is red, or v_5v_6 is red: at most $2 \cdot 2i(\Delta n - 2i)d^2(dn) = 2 \cdot 2i\Delta d^3 n^2$ choices;

- v_2v_3 is a red non-edge, or v_6v_7 is a red non-edge: at most $2 \cdot \Delta^2 d^4 n^2$ choices;
- v_4v_5 is a red non-edge: at most $(\Delta n - 2i)^2 d^4 \leqslant \Delta^2 d^4 n^2$ choices.

Subtracting these choices leads to the inequality

$$b_A(G) \geqslant (\Delta n - 2i)d^4 n^2 \left(1 - \frac{30}{n}\right) - 3d^5 \Delta n^2 - 8i\Delta d^3 n^2 - 3\Delta^2 d^4 n^2,$$

proving the first statement of the lemma.

To prove the second statement, we must investigate the average value of $b_A(G)$ over all $G \in \mathcal{S}_i$. We continue inclusion-exclusion, calculating the number (or, in two cases, the expected number) of 8-tuples containing two defects.

- Two or more dashed edges (other than v_0v_1) are present: at most $O(d^6\,\Delta n)$ choices, giving a relative error of $O(d^2/n^2)$.
- One of the dashed edges (other than v_0v_1) is a red non-edge and one of the chosen edges is red: a t most $O(i\Delta^2 d^3 n)$ such choices, giving a relative error of $O(\Delta^2/n^2)$, since $i \leqslant i_1$.
- One of the dashed edges (other than v_0v_1) is present in G, and one of the chosen edges is red: at most $O(i\Delta d^4 n)$ such choices, giving a relative error of $O(d\Delta/n^2)$.
- Two of the chosen edges are red: at most $O(i^2 d^2 \Delta n)$ for all $G \in \mathcal{S}_i$, giving a relative error of $O(i^2/(d^2n^2)) = O(\Delta^2/n^2)$, unless the two chosen edges are v_1v_2 and v_0v_7. The number of choices such that v_1v_2 and v_0v_7 are red, with v_0v_1 a red non-edge, can vary a lot across $\mathcal{S}_i$, and here we will need to calculate the average. (We come back to this, below.)
- One dashed edge is a red non-edge and another dashed edge is present in G (neither edge is v_0v_1): at most $O(\Delta^2 d^5 n)$ such choices, giving a relative error of $O(d\Delta/n^2)$.
- Two dashed edges (other than v_0v_1) are both red non-edges: at most $O(d^4\Delta^3 n)$ choices, giving a relative error of $O(\Delta^2/n^2)$.

There are two cases that must be considered further.

$*$ Recall that in the lower bound, we subtracted some "illegal" cases where v_0v_1 is red and present. For the average-case expression we must add these cases back in. The term $4i\Delta d(dn)^2$, which we subtracted to obtain the lower bound, was an upper bound for the number of 8-tuples in which v_1v_2 or v_7v_0 is red. To obtain this bound, we first choose a red edge v_0v_7, say, in $2i$ ways, and then are at most Δ choices for v_1. This upper bound of Δ includes the possibility that the edge v_0v_1 is present in G. But such choices are not valid inverse Type I switchings, and so we must undo this subtraction by adding them back in now. The number of choices of (v_7, v_0, v_1, v_2) such that v_0v_7 and v_0v_1 are red edges in G, and v_1v_2 is an edge in G (of any colour), varies quite widely among different $G \in \mathcal{S}_i$. By Lemma 5.14, the expected number of choices for this 4-tuple for a uniformly random $G \in \mathcal{S}_i$ is $O(i^2\, d/n)$.

* The second thing we must consider is the choices for the 8-tuple switching in which v_0v_7 and v_1v_2 are both red edges in G. By Lemma 5.14, the expected number of 4-tuples (v_7, v_0, v_1, v_2) with this property (and with v_0v_1 a red non-edge in G) is $O(i^2\Delta/n)$.

Adding these counts together and multiplying by $(dn)^2$, the expected number of choices of $(v_0, v_1, \dots, v_7)$ which must be added to the lower bound

$$(dn)^2\, O((d+\Delta)i^2/n) = O(i^2d^2(d+\Delta)n),$$

leading to a relative error of $O((d+\Delta)\Delta/n^2)$.

This completes the proof of the second statement of the lemma. □

5.2.2 Classes B2±

Classes B2± are easy to handle, so we discuss them before Classes B1±. Define

$$\underline{m}_\alpha(i) = (\Delta n - 2i)\Delta d^4 n - 8i\Delta^2 d^3 n - 2i\Delta d^4 n - 2\Delta^2 d^5 n - 12\Delta^2 d^4 n \quad \text{for } \alpha \in \{\text{B2}\pm\}. \tag{5.5}$$

Lemma 5.16. *For any $G \in \mathcal{S}_i$ and for $\alpha \in \{\text{B2}\pm\}$,*

$$\underline{m}_\alpha(i) \leqslant b_\alpha(G) \leqslant (\Delta n - 2i)\Delta d^4 n,$$

and thus

$$b_\alpha(G) = \Delta^2 d^4 n^2 \left(1 + O\left(\frac{d+\Delta}{n}\right)\right).$$

Proof. Observe that all Class B2± switchings are of Type I. (See Table 4.) Thus we only need to count inverse Type I Class B2+ switchings, say, and the same bounds will hold for Class B2−, by symmetry.

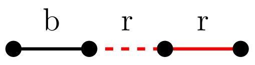
There are $\Delta n - 2i$ ways to choose v_0 and v_1. Then d^2 ways to fix v_7 and v_2. Then there are at most Δd ways to choose v_3 and v_4 and finally at most dn ways to choose v_5 and v_6. So the total number of inverse switchings is at most $(\Delta n - 2i)\Delta d^4 n$, giving the upper bound as desired. To deduce a lower bound, we subtract the number of the following structures, for which we only need an upper bound:

- v_0v_7 or v_1v_2 is red: at most $2 \cdot 2i\Delta^2 d^2 \cdot dn = 4i\Delta^2 d^3 n$ choices.
- v_2v_3 is red and present: at most $2i\Delta d^3 \cdot dn = 2i\Delta d^4 n$ choices.
- v_3v_4 is red: at most $2i\Delta^2 d^2(dn) = 2i\Delta^2 d^3 n$ choices.
- v_5v_6 is red: at most $2i(\Delta n - 2i)\Delta d^3 \leqslant 2i\Delta^2 d^3 n$ choices.
- v_6v_7 or v_4v_5 is present: at most $2 \cdot (\Delta n - 2i)\Delta d^5 \leqslant 2\Delta^2 d^5 n$ choices.
- vertex coincidence, other than $v_2 = v_7$: at most $12 \cdot (\Delta n - 2i)d^4\Delta \leqslant 12\Delta^2 d^4 n$.

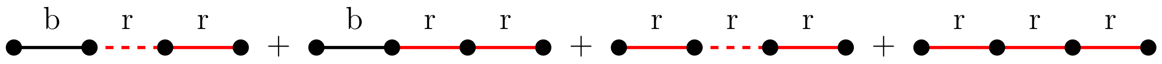
This immediately gives the required lower bound on the number of available inverse Class B2+ switchings, completing the proof. □

5.2.3 Classes B1±

The inverse switching of Type I Class B1± is indeed the same as the forward switching, up to a permutation of the labelling of the vertices involved in the switching. Recall the example discussed in Remark 5.13, where the d-factor is composed of a union of a d-regular graph with only red edges and a d-regular graph with only black edges. It is easy to see that in such a graph, the number of inverse Type I Class B1± switchings is zero. In general, the number of the following structure in G can vary a lot among $G \in \mathcal{S}_i$:



However, we do know that the sum of the number of the following structures in any d-factor $G \in \mathcal{S}_i$ is between $2i(\Delta - 1)(d - 1)$ and $2i(\Delta - 1)d$:



This motivates the introduction of switchings of other types than Type I. We display these new switchings in Table 2. Here, the colours of the edges and non-edges must be black unless specified as red. These new types of switchings are categorised into Class B1+ or B1−, under the rule that if the type ends with "+" then the class also ends with "+", and similarly for "−". Note that due to symmetry, some switchings of different types have the same definition. For instance, type IIa+ and type IIa− switchings are defined in the same way. However, they are introduced as booster switchings for different classes, and thus are categorised into different types. Again, if the class name ends with a "+" then the vertices are labelled as shown on the left of Figure 3, while if the class name ends with a "−" then the vertices are labelled as shown on the right of Figure 3.

We will show that for any $G \in \mathcal{S}_i$, the number of inverse Class B1+ (or B1−) switchings does not vary much, even though the number can be zero if restricted to inverse Type I Class B1+ (respectively, Class B1−) switchings only.

As shown in Table 2, Type IIa± switchings are described by an 8-tuple $\boldsymbol{v} = (v_0, \dots, v_7)$, while Type IIb± switchings are described by 8-tuple $(v_0, \dots, v_7)$ together with 8 additional vertices, and Type IIc± switchings are described by an 8-tuple $(v_0, \dots, v_7)$ together with 12 additional vertices, providing the additional edges used to perform the switching. We denote the sequence of these additional vertices by $\boldsymbol{y}$, where the vertices are arranged in some prescribed order: see Figure 6 for Type IIc+. An inverse Type IIb± switching is described by choosing the 8-tuple $\boldsymbol{v}$ and an 8-tuple $\boldsymbol{y}$ of additional vertices, while an inverse Type IIc± switching is described by choosing the 8-tuple $\boldsymbol{v}$ and a 12-tuple $\boldsymbol{y}$ of additional vertices.

Suppose that a Type IIb± or Type IIc± switching based on the 8-tuple $\boldsymbol{v}$ creates a graph G'. We refer to the subgraph of G' formed by vertices in $\boldsymbol{v}$ as an *octagon*. If an 8-tuple $\boldsymbol{v} = (v_0, \dots, v_7)$ in G can be combined with an 8-tuple (respectively, 12-tuple) of additional vertices $\boldsymbol{y}$ on which an inverse Type IIb±, (respectively, inverse Type IIc±)

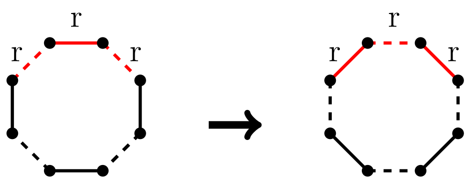
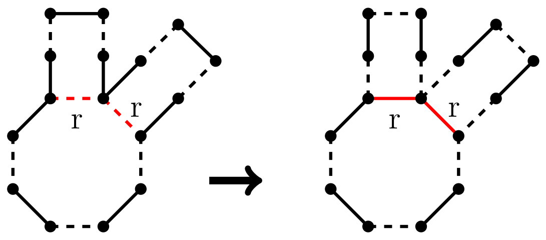
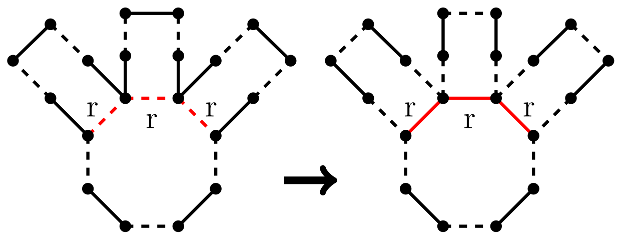
type, class	action	the switching
IIa±, B1±	$\mathcal{S}_{i-1} \to \mathcal{S}_i$	
IIb±, B1±	$\mathcal{S}_{i-2} \to \mathcal{S}_i$	
IIc±, B1±	$\mathcal{S}_{i-3} \to \mathcal{S}_i$	

Table 2: The booster switchings for classes B1±

switching can be performed, then we call $\boldsymbol{v}$ an octagon of Type IIb± (respectively, Type IIc±). The switching operation is denoted by $(G, \boldsymbol{v}, \boldsymbol{y}) \mapsto G'$. Note that octagons of different types induce different subgraph structures and (non-)edge colour restrictions. Each octagon which can result from a Type IIb± (respectively, Type IIc±) switching is not created equally often, due to the varying number of ways to select the additional vertices needed to perform the inverse switching. Thus we introduce another sort of rejection, called *pre-b-rejection*, to equalise the frequency of the creation of each octagon, given a switching type $\tau \in \{\text{IIb}\pm, \text{IIc}\pm\}$.

Given $G \in \mathcal{S}_i$, $\tau \in \{\text{IIb}\pm, \text{IIc}\pm\}$ and an octagon induced by the 8-tuple $\boldsymbol{v}$, let $\widehat{b}_\tau(G, \boldsymbol{v})$ be the number of ways to choose the sequence of additional vertices $\boldsymbol{y}$ (the length of $\boldsymbol{y}$ depends on τ) so that an inverse Type τ switching can be performed using $\boldsymbol{v}$ and $\boldsymbol{y}$. Define

$$\underline{\widehat{m}}_{\text{IIb}\pm}(i) = (dn - 2i - 12)^4 - 6(dn)^3 d^2 - 6(dn)^3 \Delta d; \tag{5.6}$$

$$\underline{\widehat{m}}_{\text{IIc}\pm}(i) = (dn - 2i - 14)^6 - 9(dn)^5 d^2 - 9(dn)^5 \Delta d. \tag{5.7}$$

Lemma 5.17. *Let $\tau \in \{\text{IIb}+, \text{IIc}+\}$ and $G \in \mathcal{S}_i$. For any octagon $\boldsymbol{v} = (v_0, \ldots, v_7)$ in G*

that can be created by a type τ switching,

$$\underline{\widehat{m}}_\tau(i) \leqslant \widehat{b}_\tau(G, \boldsymbol{v}) = \underline{\widehat{m}}_\tau(i)(1 + O((d + \Delta)/n)).$$

Proof. We only prove the result for τ = IIb+, as the argument for τ = IIc+ is similar and by symmetry, the same bounds will hold for τ = IIb− and τ = IIc−, respectively.

Let $\boldsymbol{v} = (v_0, \ldots, v_7)$ be a fixed 8-tuple which gives rise to the octagon shown in Figure 6. We bound the number of ways to choose an 8-tuple $\boldsymbol{y} = (y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8)$ of additional vertices so that dashed lines in Figure 6 correspond to black non-edges in G.

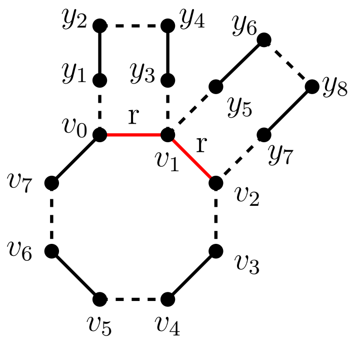


Figure 6: Choosing the additional vertices for an inverse Type IIc+ switching

The upper bound $(dn)^4$ is obvious. For the lower bound, first notice that this number is at least $(dn - 2i - 12)^4$, as the 4 extra edges involved in the inverse switching are black, and are distinct from each other and from the 3 black edges in the octagon Further, we need to subtract the number of choices where at least one defect appears. There are at most $6 \cdot (dn)^3 d^2$ choices where one designated non-edge (such as v_0y_1 or y_2y_4) is actually present, and at most $6 \cdot (dn)^3 d\Delta$ choices where one designated non-edge is red. Subtracting these counts gives the required lower bound. □

We will specify $\overline{m}_\tau(i)$ for $\tau \in \{\text{IIa}\pm, \text{IIb}\pm, \text{IIc}\pm\}$ in (5.11)–(5.13). It is trivial to see that $f_\tau(G) \leqslant \overline{m}_\tau(i)$ for each such τ and for $G \in \mathcal{S}_i$. These types of switchings are performed so rarely that the trivial lower bound $f_\tau(G) \geqslant 0$ is sufficient for our analysis: see the proof of Lemma 5.23.

Pre-b-rejection

When we count the number of inverse Class B1± switchings applicable to $G \in \mathcal{S}_i$, we count the number of choices of $(v_0, \ldots, v_7)$ that are allowed to be created by a Class B1± switching. However, some types of switchings create structures with more vertices than those in an octagon. For instance, let $\boldsymbol{v}$ be an octagon of type IIb+ and let $\boldsymbol{x} = (x_1, \ldots, x_4)$ be the four extra edges that are created by a Type IIb+ switching. We can consider $(G, \boldsymbol{v}, \boldsymbol{x})$ as a *pre-state* of $(G, \boldsymbol{v})$. Each octagon $\boldsymbol{v}$ in G corresponds to exactly $\widehat{b}_\tau(G, \boldsymbol{v})$

pre-states, and $\widehat{b}_\tau(G,\boldsymbol{v}) \approx \underline{\widehat{m}}_\tau(i)$ by Lemma 5.17. By carefully designing the pre-b-rejection scheme, we can ensure that each octagon $\boldsymbol{v}$ in G is created equally often if each of its pre-states are created equally often.

When a Type τ switching converting G to G' is chosen, corresponding to a valid 8-tuple $\boldsymbol{v}$, we will reject the algorithm and restart with probability

$$1-\frac{\underline{\widehat{m}}_\tau(i)}{\widehat{b}_\tau(G,\boldsymbol{v})}.$$

This restart will be called a *pre-b-rejection.*

The pre-b-rejection is incorporated in the formal definition of the algorithm in Section 5.3. We close this section by bounding $b_\alpha(G)$ for $\alpha \in \{\mathrm{B1}\pm\}$.

Define

$$\underline{m}_\alpha(i) = 2i(\Delta-1)(d-1)(dn-2i-10d)^2 - 6i(\Delta-1)d^3n(d+\Delta) \quad \text{for } \alpha \in \{\mathrm{B1}\pm\}. \tag{5.8}$$

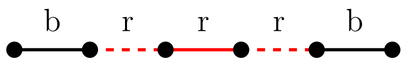
Lemma 5.18. *For any $G \in \mathcal{S}_i$, and $\alpha \in \{\mathrm{B1}\pm\}$,*

$$\underline{m}_\alpha(i) \leqslant b_\alpha(G) = \underline{m}_\alpha(i)\left(1+O\left(\frac{1}{d}+\frac{d+\Delta}{n}\right)\right).$$

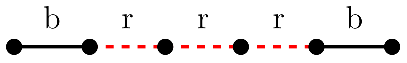
Proof. The number of ways to choose v_7, v_0, v_1 and v_2 is between $2i(\Delta-1)(d-1)$ and $2i(\Delta-1)d$. The number of ways to choose the other four vertices is at least $(dn-2i-10d)^2$ so that there are no vertex coincidence and both v_3v_4 and v_5v_6 are black. We subtract the choices where v_2v_3 or v_4v_5 or v_6v_7 is present in G, or is a red non-edge. There are at most $3\cdot 2i(\Delta-1)d(d^2(dn)+\Delta d(dn))$ such choices. This verifies the desired lower bound. The upper bound $2i(\Delta-1)d(dn)^2$ is trivial, which yields the required relative error. □

5.2.4 Classes C±

Now consider Class C±. As we will show later, the probability that a Type I switching is in Class C± is very small. Thus, we only need a rather rough lower bound on the number of inverse Class C± switchings, so that the probability of a b-rejection is not too close to 1. However, there are very rare graphs in $\mathcal{S}_i$ that cannot be created by a Type I Class C± switching. For instance, this may occur if $d=\Delta$ and the set of all red edges in G form a red d-regular subgraph. Then G does not contain the following structure,



and thus cannot be created by a Type I switching. In this case, that the probability of a b-rejection would equal 1 due to the existence of such graphs. In order to reduce the probability of a b-rejection, we introduce a new type of switching, namely Type III+ for Class C+ and Type III− for Class C−, that boost the probability of graphs which contain the following structure:



It turns out that for any $G \in \mathcal{S}_i$, the number of choices of 6-tuples of vertices $(x_1, \dots, x_6)$ such that x_1x_2 and x_5x_6 are black edges in G, x_2x_3 and x_4x_5 are red non-edges, and x_3x_4 is either a red edge or a red non-edge, is always sufficiently concentrated. See Lemma 5.19 for a precise bound. This is why we boost the second structure, to transform a highly varying count into a well-concentrated count.

The Type III$\pm$, Class C$\pm$ switchings are shown in Table 3.

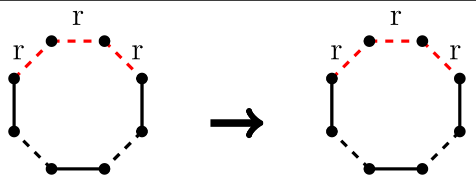
type, class	action	the switching
III$\pm$, C$\pm$	$\mathcal{S}_i \to \mathcal{S}_i$	

Table 3: The booster switchings for Classes C$\pm$

Unusually, the Type III$\pm$ switchings do not perform any switch of edges, except for designating an 8-tuple of vertices satisfying certain constraints, as shown in Table 3. They can be viewed as adding a small "do nothing" probability to the algorithm. As we will see later, the probability of ever performing a Class C$\pm$ switching is extremely small.

As before, although type III+ and III− switchings have the same definition, they are booster switchings for classes C+ and C− respectively, and thus have to be categorised into different types.

Define

$$\underline{m}_\alpha(i) = d^3\Delta^3n^2(1 - 8(d+\Delta)/n) \qquad \text{for } \alpha \in \{\text{C}\pm\}, \tag{5.9}$$
$$\overline{m}_\tau(i) = \Delta^3d^3n^2 \qquad \text{for } \tau \in \{\text{III}\pm\}. \tag{5.10}$$

Lemma 5.19. *For each $G \in \mathcal{S}_i$ and for $\tau \in \{\text{III}\pm\}$,*

$$\overline{m}_\tau(i)(1 - 8(d+\Delta)/n) \leqslant f_\tau(G) \leqslant \overline{m}_\tau(i).$$

For each $G \in \mathcal{S}_i$ and for $\alpha \in \{\text{C}\pm\}$,

$$\underline{m}_\alpha(i) \leqslant b_\alpha(G) \leqslant d^3\Delta^3n^2.$$

Proof. We only discuss the case $\tau = \text{III+}$, as the case $\tau = \text{III−}$ is symmetric. There are at most $\Delta n - 2i \leqslant \Delta n$ ways to choose (v_0, v_1), and then at most Δ^2 ways to choose (v_2, v_7). Then there are at most d^2 ways to choose (v_3, v_6). Finally, there are at most $dn - 2i \leqslant dn$ ways to choose (v_4, v_5). This gives the required upper bound for $f_{\text{III+}}(G)$. To obtain the lower bound, we need to subtract from these $d^2\Delta^2(\Delta n - 2i)(dn - 2i)$ choices of $(v_0, \dots, v_7)$ the following cases:

(a) v_1v_2 or v_0v_7 is a red edge in G;

(b) v_2v_3 or v_6v_7 is a red edge in G;

(c) v_3v_4 or v_5v_6 is a edge in G.

The number of choices for (a) is at most $2 \cdot 2id^2\Delta^2 dn = 4id^3\Delta^2 n$. To see this, there are at most $2i$ ways to fix v_1 and v_2 if v_1v_2 is a red edge in G; then at most d ways to fix v_3, at most Δ^2 ways to fix v_0 and v_7, at most d ways to fix v_6 and finally at most dn ways to fix v_4 and v_5. The factor of 2 covers the case that v_2v_7 is a red edge present in G.

The number of choices for (b) is at most $2 \cdot 2i\Delta^3 ddn = 4id^2\Delta^3 n$.

The number of choices for (c) is at most $2 \cdot \Delta n \Delta^2 d^4 = 2\Delta^3 d^4 n$.

Hence, we have

$$f_{\text{III}+}(G) \geqslant d^2\Delta^2(\Delta n - 2i)(dn-2i) - 4id^3\Delta^2 n - 4id^2\Delta^3 n - 2\Delta^3 d^4 n \geqslant d^3\Delta^3 n^2(1 - 8(d+\Delta)/n),$$

as required, since $i < d\Delta$ by (4.1).

Next we bound $b_\alpha(G)$ for $\alpha = \text{C}+$, as the case $\alpha = \text{C}-$ is symmetric. To perform an inverse Class C+ switching, we need to designate an 8-tuple $\boldsymbol{v} = (v_0, \dots, v_7)$ such that either an inverse Type I Class C+ switching can be performed on $\boldsymbol{v}$, or an inverse Type Va switching can be performed on $\boldsymbol{v}$. Note that an inverse type C+ switching is just the same as a type C+ switching. The lower bound on $f_{\text{III}+}(G)$ naturally is a lower bound for $b_{\text{C}+}(G)$. So immediately we have $b_{\text{C}+}(G) \geqslant \underline{m}_{\text{C}+}(i)$ as specified in (5.9). It is not hard to see that $d^3\Delta^3 n^2$ is an upper bound for $b_{\text{C}+}(G)$, because there are at most Δn ways to fix v_0 and v_1 (either v_0v_1 is present or not present in G), and at most $\Delta^2 d^2$ ways to fix v_2, v_3, v_7, v_6 and then at most dn ways to fix v_4 and v_5. $\square$

5.3 The algorithm: `FactorUniform`

Now we have defined all types and classes of switchings involved in `FactorUniform`. Figure 7 depicts all switchings which produce an element of $\mathcal{S}_i$, labelled by their type and class.

We now describe the algorithm `FactorUniform` formally. First, `FactorUniform` calls REG to generate a uniformly random d-regular graph G_0 on $\{1, 2, \dots, n\}$. If G_0 contains more than i_1 red edges then `FactorUniform` restarts. Otherwise, `FactorUniform` iteratively performs a sequence of switching steps. In each switching step, if the current graph G is in $\mathcal{S}_i$ then `FactorUniform` chooses a switching type τ from a set of types Γ, with probability $\rho_\tau(i)$. Here

$$\Gamma = \{\text{I, IIa}\pm\text{, IIb}\pm\text{, IIc}\pm\text{, III}\pm\}$$

and we insist only that $\sum_{\tau\in\Gamma} \rho_\tau(i) \leqslant 1$ for all $i \leqslant i_1$. With probability $1 - \sum_{\tau\in\Gamma} \rho_\tau(i)$ we perform a rejection called "t-rejection" instead of choosing a switching type. If no t-rejection is performed then `FactorUniform` chooses a random Type τ switching, and either performs this chosen switching, or restarts with a small probability (the sum of the

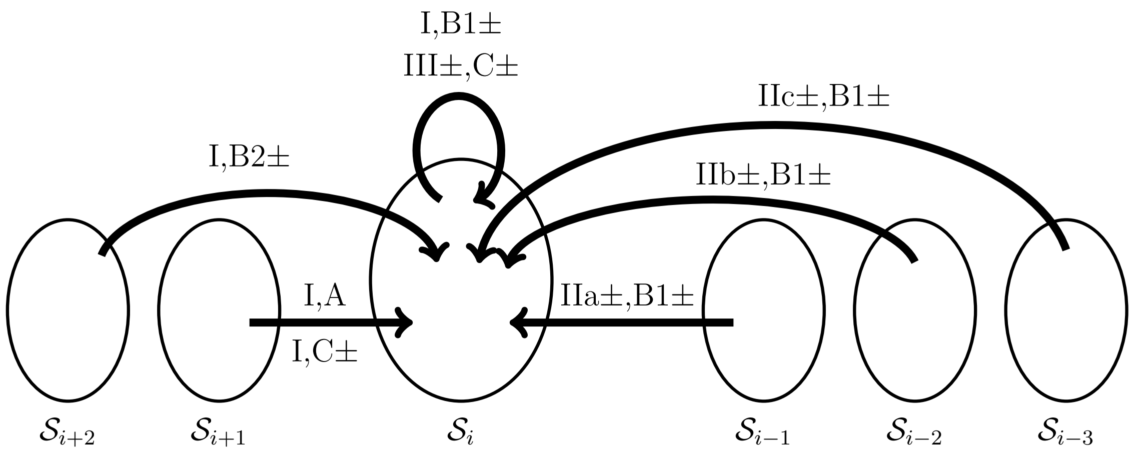


Figure 7: All switchings into $\mathcal{S}_i$, labelled by type and class

f-rejection, pre-b-rejection and b-rejection probabilities). The parameters $\rho_\tau(i)$ will be specified in the next section. If $i = 0$ then a Type I switching is interpreted as outputting the current graph.

To be more specific, let G_t be the graph obtained after t switching steps, and suppose that $G_t = G \in \mathcal{S}_i$. The $(t+1)$'th switching step is composed of the following substeps:

(i) Choose switching type $\tau \in \Gamma$ with probability $\rho_\tau(i)$. If no type is chosen, perform a t-rejection.

(ii) Assume that no t-rejection was performed. If $i = 0$ and $\tau = I$ then output the current graph G. Otherwise, choose a random Type τ switching S for G and let G' be the graph obtained from G by performing S. Perform an f-rejection with probability $1 - f_\tau(G)/\overline{m}_\tau(i)$.

(iii) If no f-rejection is performed then perform a pre-b-rejection, if applicable. (See the description given above Lemma 5.18.)

(iv) If no f-rejection or pre-b-rejection is performed then let α be the class of S and suppose that $G' \in \mathcal{S}_{i'}$. Perform a b-rejection with probability $1 - \underline{m}_\alpha(i')/b_\alpha(G')$.

(v) If no b-rejection is performed then set $G_{t+1} = G'$.

If any rejection occurs then `FactorUniform` restarts.

Again, this is a Las Vegas algorithm with no deterministic upper bound on the running time of the algorithm, due to the chance of rejections. But we will show that the probability of a rejection occurring is small, under the assumptions of Theorem 1.2.

5.4 Uniformity: fixing $\rho_\tau(i)$

We complete the definition of `FactorUniform` by specifying the parameters $\rho_\tau(i)$, for $\tau \in \Gamma$. Let $\sigma(G)$ denote the expected number of times that G is reached by `FactorUniform`. We will design $\rho_\tau(i)$ such that $\sigma(G) = \sigma_i$ for some σ_i, for every $G \in \mathcal{S}_i$ and every $0 \leqslant i \leqslant i_1$. A method of designing these parameters is discussed in [9, Section 5] in a general setting. In the rest of this section, we carry out this method and apply it to our specific problem.

For convenience, we summarise which switching types occur for each class in Table 4. As usual, a type ending in "+" goes with a class ending in "+", and similarly for those ending in "−".

Class	Types associated with the given class
A	I
B1±	I, IIa±, IIb±, IIc±
B2±	I
C±	I, III±

Table 4: Reference table for types and classes.

Below is a list of parameters $\overline{m}_\tau(i)$, which are upper bounds on the number of ways to perform a switching of each type on a given $G \in \mathcal{S}_i$:

$$\overline{m}_{\mathrm{I}}(i) = 2i(dn)^3\left(1 + 28\left(\frac{(\Delta+d)^2}{n^2} + \frac{1}{n}\right)\right) - 8i(d-1)^2d^2n^2 - 4i\Delta d^3n^2 - 4i^2(dn)^2,$$

$$\overline{m}_\tau(i) = 2i\Delta^2d^3n, \qquad \tau \in \{\mathrm{IIa}\pm\}, \tag{5.11}$$

$$\overline{m}_\tau(i) = \Delta^2d^9n^5, \qquad \tau \in \{\mathrm{IIb}\pm\}, \tag{5.12}$$

$$\overline{m}_\tau(i) = \Delta^3d^{11}n^6, \qquad \tau \in \{\mathrm{IIc}\pm\}, \tag{5.13}$$

$$\overline{m}_\tau(i) = \Delta^3d^3n^2, \qquad \tau \in \{\mathrm{III}\pm\}.$$

Next we list parameters $\underline{m}_\alpha(i)$, which are lower bounds on the number of ways to perform switchings of each class to produce a given $G \in \mathcal{S}_i$:

$$\underline{m}_A(i) = (\Delta n - 2i)d^2(dn)^2\left(1 - \frac{30}{n}\right) - 3d^5\Delta n^2 - 8i\Delta d^3n^2 - 3\Delta^2d^4n^2,$$

$$\underline{m}_\alpha(i) = 2i(\Delta-1)(d-1)(dn-2i-10d)^2 - 6i(\Delta-1)d^3n(d+\Delta), \qquad \alpha \in \{\mathrm{B1}\pm\},$$

$$\underline{m}_\alpha(i) = (\Delta n - 2i)\Delta d^4n - 8i\Delta^2d^3n - 2i\Delta d^4n - 2\Delta^2d^5n - 12\Delta^2d^4n, \qquad \alpha \in \{\mathrm{B2}\pm\},$$

$$\underline{m}_\alpha(i) = d^3\Delta^3n^2(1 - 8(d+\Delta)/n), \qquad \alpha \in \{\mathrm{C}\pm\}.$$

Fix a class α and let τ be a type such that class α and type τ appear together in some row of Table 4. For each relevant $i \leqslant i_1$, let $q_\alpha^\tau(i)$ denote the expected number of times that an element of $\mathcal{S}_i$ is reached by a Type τ, Class α switching. We will choose our parameters to ensure that the value of $q_\alpha^\tau(j)$ *does not depend on* τ, for any type τ associated with class α. This common value is denoted by $q_\alpha(i)$; that is, $q_\alpha(i) = q_\alpha^\tau(i)$ for any type τ associated with class α.

It follows then that

$$\sigma_i = \frac{1}{|\mathcal{A}_0|} + \sum_\alpha q_\alpha(i)\, \underline{m}_\alpha(i), \quad \text{for every } 0 \leqslant i \leqslant i_1. \tag{5.14}$$

This equation holds because every $G \in \mathcal{S}_i$ can be chosen as the initial graph, if not initially rejected; or is reached via some switching. The probability that G is the graph obtained at Step 0 is $1/|\mathcal{A}_0|$, since G_0 is chosen uniformly, and by our design of the algorithm, $q_\alpha(i)\, \underline{m}_\alpha(i)$ is exactly the expected number of times that G is reached via some class α switching and is not t-rejected, f-rejected, pre-b-rejected or b-rejected.

Immediately we have

$$q_A(i) = \frac{\sigma_{i+1}\, \rho_{\mathrm{I}}(i+1)}{\overline{m}_{\mathrm{I}}(i+1)}. \tag{5.15}$$

This is because a graph $G \in \mathcal{S}_i$ can be created via a Class A switching only via a Type I switching on a graph $G' \in \mathcal{S}_{i+1}$. (See the first line of Table 4.) Every graph in $\mathcal{S}_{i+1}$ is visited σ_{i+1} times in expectation, and given any $G' \in \mathcal{S}_{i+1}$ such that $S = (G', G)$ is a valid Type I Class A switching, the probability that `FactorUniform` chooses Type I is $\rho_{\mathrm{I}}(i+1)$, and the probability that `FactorUniform` chooses the particular switching S is $1/\overline{m}_{\mathrm{I}}(i+1)$.

Next, consider $\alpha \in \{\mathrm{B1}\pm\}$. A Class B1$\pm$ switching can be of Type I, IIa$\pm$, IIb$\pm$or IIc$\pm$ (see the second line of Table 4). Now $G \in \mathcal{S}_i$ might be created from some $G' \in \mathcal{S}_i$ via a Type I Class B1$\pm$ switching. Thus, arguing as above, we have

$$q_{\mathrm{B1}+}(i) = q_{\mathrm{B1}-}(i) = \frac{\sigma_i\, \rho_{\mathrm{I}}(i)}{\overline{m}_{\mathrm{I}}(i)}. \tag{5.16}$$

To ensure that the expected number of times G is visited via Type τ Class B1$\pm$ switchings does not depend on τ, for $\tau \in \{I, \mathrm{IIa}\pm, \mathrm{IIb}\pm, \mathrm{IIc}\pm\}$, we must choose $\rho_\tau(\cdot)$ for $\tau \in \{\mathrm{IIa}\pm, \mathrm{IIb}\pm, \mathrm{IIc}\pm\}$ such that

$$\frac{\sigma_i\, \rho_{\mathrm{I}}(i)}{\overline{m}_{\mathrm{I}}(i)} = q_{\mathrm{B1}+}(i) = \frac{\sigma_{i-2}\, \rho_{\mathrm{IIb}+}(i-2)}{\overline{m}_{\mathrm{IIb}+}(i-2)} \cdot \widehat{\underline{m}}_{\mathrm{IIb}+}(i) \tag{5.17}$$

$$= \frac{\sigma_{i-1}\, \rho_{\mathrm{IIa}+}(i-1)}{\overline{m}_{\mathrm{IIa}+}(i-1)} \tag{5.18}$$

$$= \frac{\sigma_{i-3}\, \rho_{\mathrm{IIc}+}(i-3)}{\overline{m}_{\mathrm{IIc}+}(i-3)} \cdot \widehat{\underline{m}}_{\mathrm{IIc}+}(i), \tag{5.19}$$

and

$$\frac{\sigma_i\,\rho_{\mathrm{I}}(i)}{\overline{m}_{\mathrm{I}}(i)} = q_{\mathrm{B1}-}(i) = \frac{\sigma_{i-2}\,\rho_{\mathrm{IIb}-}(i-2)}{\overline{m}_{\mathrm{IIb}-}(i-2)} \cdot \underline{\widehat{m}}_{\mathrm{IIb}-}(i) \tag{5.20}$$

$$= \frac{\sigma_{i-1}\,\rho_{\mathrm{IIa}-}(i-1)}{\overline{m}_{\mathrm{IIa}-}(i-1)} \tag{5.21}$$

$$= \frac{\sigma_{i-3}\,\rho_{\mathrm{IIc}-}(i-3)}{\overline{m}_{\mathrm{IIc}-}(i-3)} \cdot \underline{\widehat{m}}_{\mathrm{IIc}-}(i). \tag{5.22}$$

Next we consider $\alpha \in \{\mathrm{B2}\pm\}$. A Class B2$\pm$ switching can only be of Type I (see the third line of Table 4). Thus we immediately have

$$q_{\mathrm{B2}+}(i) = q_{\mathrm{B2}-}(i) = \frac{\sigma_{i+2}\,\rho_{\mathrm{I}}(i+2)}{\overline{m}_{\mathrm{I}}(i+2)}. \tag{5.23}$$

Finally, consider $\alpha \in \{\mathrm{C}\pm\}$. A Class C$\pm$ switching can be of Type I or Type III$\pm$(see the fourth row of Table 4). Considering switchings of Type I and Class C$\pm$, we have

$$q_{\mathrm{C}+}(i) = q_{\mathrm{C}-}(i) = \frac{\sigma_{i+1}\,\rho_{\mathrm{I}}(i+1)}{\underline{m}_{\mathrm{I}}(i+1)}. \tag{5.24}$$

To ensure that the expected number of times G is visited by Type III$\pm$ Class C$\pm$ switchings, we must choose $\rho_\tau(\cdot)$ for $\tau \in \{\mathrm{III}\pm\}$ such that

$$\frac{\sigma_{i+1}\,\rho_{\mathrm{I}}(i+1)}{\underline{m}_{\mathrm{I}}(i+1)} = q_{\mathrm{C}+}(i) = \frac{\sigma_i\,\rho_{\mathrm{III}+}(i)}{\overline{m}_{\mathrm{III}+}(i)}, \tag{5.25}$$

and

$$\frac{\sigma_{i+1}\,\rho_{\mathrm{I}}(i+1)}{\underline{m}_{\mathrm{I}}(i+1)} = q_{\mathrm{C}-}(i) = \frac{\sigma_i\,\rho_{\mathrm{III}-}(i)}{\overline{m}_{\mathrm{III}-}(i)}. \tag{5.26}$$

Combining (5.15)–(5.26), and using (5.14), we deduce that

$$\sigma_i = \frac{1}{|\mathcal{A}_0|} + \sum_\alpha q_\alpha(i)\,\underline{m}_\alpha(i) \tag{5.27}$$

for $i = 0, \ldots, i_1$, where α ranges over all possible classes $\{A, \mathrm{B1}\pm, \mathrm{B2}\pm, \mathrm{C}\pm\}$. Using the change of variables $x_i = \sigma_i\,|\mathcal{A}_0|$, for $i = 0, \ldots, i_1$, as in [9], we rewrite this as

$$x_i = 1 + \sum_\alpha q_\alpha(i)\,\underline{m}_\alpha(i)\,|\mathcal{A}_0|. \tag{5.28}$$

5.4.1 Boundary conditions and solving the system

Recall that `FactorUniform` rejects the initial d-regular graph if it contains more than i_1 red edges. Thus, we set $\rho_\tau(i) = 0$ for all $i > i_1$. The parameter $\rho_{\mathrm{I}}(i)$ is already defined for all $0 \leqslant i \leqslant i_1$, recalling that $\rho_{\mathrm{I}}(0)$ is interpreted as the probability of outputting the current graph. The Type I switchings may be of Class B1±, B2± or C±, as shown in Figure 7. A Type I Class B1± switching converts a graph from $\mathcal{S}_i$ to $\mathcal{S}_i$, and the booster switchings for Class B1± are of Type IIa±, IIb±, IIc±. By Table 2, we must define $\rho_\tau(i)$ for all

$$\begin{cases} 0 \leqslant i \leqslant i_1 - 1 & \text{if } \tau \in \{\text{IIa}\pm\}, \\ 0 \leqslant i \leqslant i_1 - 2 & \text{if } \tau \in \{\text{IIb}\pm\}, \\ 0 \leqslant i \leqslant i_1 - 3 & \text{if } \tau \in \{\text{IIc}\pm\}. \end{cases}$$

Similarly, consider the booster switchings for classes C±. When $\tau \in \{\text{III}\pm\}$ we must define $\rho_\tau(i)$ for all $0 \leqslant i \leqslant i_1$.

Note that in general there are switchings converting graphs in $\mathcal{S}_j$ to graphs in $\mathcal{S}_i$ for $i-3 \leqslant j \leqslant i+2$, as shown in Figure 7. For i close to i_1 or 0 there are fewer switching types involved. For instance, graphs in $\mathcal{S}_{i_1}$ cannot be reached by a Type I Class A switching, because the boundary conditions will be set so that no graphs in strata $j > i_1$ will ever be reached. Similarly, no graphs in $\mathcal{S}_0$ can be reached by a Type IIa± switching, because any such switching increases the number of red edges in the graph, and this number can never be negative.

Let ϵ be a function of n, d and Δ to be specified later. We first give a computation scheme that determines $\rho_{\mathrm{I}}(i)$, $\rho_{\mathrm{III}+}(i)$, $\rho_{\mathrm{III}-}(i)$ and x_i for all $0 \leqslant i \leqslant i_1$ such that

$$\rho_{\mathrm{I}}(i_1) = 1 - \epsilon; \qquad \rho_{\mathrm{III}}(i_1) = 0; \tag{5.29}$$

$$\rho_{\mathrm{I}}(i) + \rho_{\mathrm{III}+}(i) + \rho_{\mathrm{III}-}(i) = 1 - \epsilon \quad \text{for all } 0 \leqslant i \leqslant i_1 - 1. \tag{5.30}$$

By symmetry, let $\rho_{\mathrm{III}}(i) = \rho_{\mathrm{III}+}(i) = \rho_{\mathrm{III}-}(i)$. We determine $\rho_{\mathrm{I}}(i)$, $\rho_{\mathrm{III}}(i)$ and x_i recursively for i in descending order.

Substituting

$$q_A(i) = \frac{\sigma_{i+1}\,\rho_{\mathrm{I}}(i+1)}{\overline{m}_{\mathrm{I}}(i+1)}; \tag{5.31}$$

$$q_{\mathrm{B1}+}(i) = q_{\mathrm{B1}-}(i) = \frac{\sigma_i\,\rho_{\mathrm{I}}(i)}{\overline{m}_{\mathrm{I}}(i)}; \tag{5.32}$$

$$q_{\mathrm{B2}+}(i) = q_{\mathrm{B2}-}(i) = \frac{\sigma_{i+2}\,\rho_{\mathrm{I}}(i+2)}{\overline{m}_{\mathrm{I}}(i+2)}; \tag{5.33}$$

$$q_{\mathrm{C}+}(i) = q_{\mathrm{C}-}(i) = \frac{\sigma_{i+1}\,\rho_{\mathrm{I}}(i+1)}{\underline{m}_{\mathrm{I}}(i+1)} \tag{5.34}$$

into (5.28), we have

$$x_i = 1 + \frac{x_{i+1}\,\rho_{\mathrm{I}}(i+1)}{\overline{m}_{\mathrm{I}}(i+1)}\underline{m}_A(i) + 2\cdot\frac{x_i\,\rho_{\mathrm{I}}(i)}{\overline{m}_{\mathrm{I}}(i)}\underline{m}_{\mathrm{B1}+}(i) + 2\cdot\frac{x_{i+2}\,\rho_{\mathrm{I}}(i+2)}{\overline{m}_{\mathrm{I}}(i+2)}\underline{m}_{\mathrm{B2}+}(i) + 2\cdot\frac{x_{i+1}\,\rho_{\mathrm{I}}(i+1)}{\overline{m}_{\mathrm{I}}(i+1)}\underline{m}_{\mathrm{C}+}(i). \tag{5.35}$$

Here we used the fact that $\underline{m}_{\mathrm{B1}+}(i) = \underline{m}_{\mathrm{B1}-}(i)$, $\underline{m}_{\mathrm{B2}+}(i) = \underline{m}_{\mathrm{B1}-}(i)$ and $\underline{m}_{\mathrm{C}+}(i) = \underline{m}_{\mathrm{C}-}(i)$. Similarly, by (5.25) and (5.26), and using the fact that $\overline{m}_{\mathrm{III}+}(i) = \overline{m}_{\mathrm{III}-}(i)$,

$$\rho_{\mathrm{III}}(i) = \frac{x_{i+1}}{x_i}\,\frac{\overline{m}_{\mathrm{III}+}(i)}{\overline{m}_{\mathrm{I}}(i+1)}\,\rho_{\mathrm{I}}(i+1). \tag{5.36}$$

Base case (a): $i = i_1$. We have set $\rho_{\mathrm{I}}(i_1) = 1-\epsilon$, $\rho_{\mathrm{III}}(i_1) = 0$ and $\rho_{\mathrm{I}}(i) = 0$ for all $i > i_1$. Hence, by (5.35),

$$\left(1 - 2\cdot\frac{\rho_{\mathrm{I}}(i_1)}{\overline{m}_{\mathrm{I}}(i_1)}\underline{m}_{\mathrm{B1}+}(i_1)\right) x_{i_1} = 1$$

which determines x_{i_1}.

Base case (b): $i = i_1 - 1$. We have

$$\rho_{\mathrm{I}}(i_1-1) + 2\rho_{\mathrm{III}}(i_1-1) = 1-\epsilon, \tag{5.37}$$

and

$$\left(1 - \kappa_1\,\rho_{\mathrm{I}}(i_1-1)\right) x_{i_1-1} = \kappa_2, \tag{5.38}$$

where

$$\kappa_1 = \frac{2\,\underline{m}_{\mathrm{B1}+}(i_1-1)}{\overline{m}_{\mathrm{I}}(i_1-1)}, \qquad \kappa_2 = 1 + \frac{\rho_{\mathrm{I}}(i_1)}{\overline{m}_{\mathrm{I}}(i_1)}\big(\underline{m}_A(i_1-1) + 2\,\underline{m}_{\mathrm{C}+}(i_1-1)\big)\,x_{i_1}.$$

Hence, by (5.36),

$$\rho_{\mathrm{III}}(i_1-1)\,x_{i_1-1} = \kappa_3 \tag{5.39}$$

where

$$\kappa_3 = x_{i_1}\,\frac{\overline{m}_{\mathrm{III}+}(i_1-1)\,\rho_{\mathrm{I}}(i_1)}{\overline{m}_{\mathrm{I}}(i_1)}.$$

Solving (5.37), (5.38) and (5.39) gives

$$x_{i_1-1} = \frac{\kappa_2 - 2\,\kappa_1\kappa_3}{1-\kappa_1(1-\epsilon)},$$

and substituting this into (5.39) and then into (5.37) yields $\rho_{\mathrm{III}}(i_1-1)$ and $\rho_{\mathrm{I}}(i_1-1)$.

Inductive step. Now assume that $i \leqslant i_1 - 2$ and that $\rho_{\mathrm{I}}(j)$, $\rho_{\mathrm{III}}(j)$ and x_j have been determined for all $j > i$. By (5.35) we have

$$\left(1 - \kappa_1\rho_{\mathrm{I}}(i)\right) x_i = \kappa_2, \tag{5.40}$$

where

$$\begin{aligned}
\kappa_1 &= \frac{2\,\underline{m}_{\mathrm{B1}+}(i)}{\overline{m}_{\mathrm{I}}(i)}, \\
\kappa_2 &= 1 + \frac{\rho_{\mathrm{I}}(i+1)}{\overline{m}_{\mathrm{I}}(i+1)}\big(\underline{m}_A(i) + 2\,\underline{m}_{\mathrm{C}+}(i)\big)\,x_{i+1} + 2\cdot\frac{\rho_{\mathrm{I}}(i+2)\,\underline{m}_{\mathrm{B2}+}(i)}{\overline{m}_{\mathrm{I}}(i+2)}\,x_{i+2}.
\end{aligned}$$

We also have

$$\rho_{\mathrm{I}}(i) + 2\rho_{\mathrm{III}}(i) = 1 - \epsilon, \tag{5.41}$$

$$\rho_{\mathrm{III}}(i)x_i = \kappa_3, \quad \text{where } \kappa_3 = x_{i+1}\,\frac{\overline{m}_{\mathrm{III}+}(i)\,\rho_{\mathrm{I}}(i+1)}{\overline{m}_{\mathrm{I}}(i+1)}. \tag{5.42}$$

As in base case (b), we obtain

$$x_i = \frac{\kappa_2 - 2\kappa_1\kappa_3}{1-\kappa_1(1-\epsilon)}, \tag{5.43}$$

and immediately this gives $\rho_{\mathrm{III}}(i)$ and $\rho_{\mathrm{I}}(i)$. Thus we have uniquely determined $\rho_{\mathrm{I}}(i)$, $\rho_{\mathrm{III}}(i)$ and x_i that satisfy (5.29) and (5.30).

Next we specify ϵ. Define

$$\epsilon = 5\left(\frac{\Delta + d}{n}\right)^2. \tag{5.44}$$

Lemma 5.20. *For ϵ as defined in* (5.44)*, equations* (5.15)–(5.28) *have a unique solution* $(x_i^*, \rho_\tau^*(i) : \tau \in \Gamma, i \geqslant 0)$ *which satisfies* (5.29) *and* (5.30)*. Moreover, for all* $0 \leqslant i \leqslant i_1$*,*

$$x_i^* > 0; \qquad \frac{x_i^*}{x_{i-1}^*} = O(i/d\Delta) \quad \textit{if} \quad i \geqslant 1; \tag{5.45}$$

$$0 \leqslant \rho_\tau^*(i) \leqslant 1 \quad \textit{for all} \quad \tau \in \Gamma; \qquad \sum_{\tau\in\Gamma} \rho_\tau^*(i) \leqslant 1. \tag{5.46}$$

Proof. We have already shown that there are unique x_i^*, $\rho_{\mathrm{I}}^*(i)$ and $\rho_{\mathrm{III}+}^*(i)$, $\rho_{\mathrm{III}-}^*(i)$ satisfying (5.29) and (5.30). Substituting these values into (5.17)–(5.22) uniquely determines $\rho_\tau^*(i)$ for all $\tau \in \{$IIa$\pm$, IIb$\pm$, IIc$\pm\}$ and all relevant values of i.

It is easy to verify that $x_{i_1}^* > 0$ and $x_{i_1-1}^* > 0$ and that (5.46) is satisfied for $i = \{i_1, i_1 - 1\}$. We will prove by induction on i, for all $0 \leqslant i \leqslant i_1 - 1$, that (5.46) is satisfied as well as the following strengthening condition of (5.45).

$$x_{i+1}^* > 0; \quad x_i^* > 0, \quad \frac{x_{i+1}^*}{x_i^*} \leqslant 2.3(i+1)/d\Delta. \tag{5.47}$$

By (5.35), for every $i \leqslant i_1 - 1$, provided that $x_{i+2} \geqslant 0$ and $\rho_{\mathrm{I}}(i+2) \geqslant 0$, we have

$$\frac{x_{i+1}}{x_i} \leqslant \frac{1.1\,\overline{m}_{\mathrm{I}}(i+1)}{\underline{m}_A(i)} \leqslant 2.3(i+1)/d\Delta, \tag{5.48}$$

since

$$x_i\left(1 - 2\cdot\frac{\rho_{\mathrm{I}}(i)}{\overline{m}_{\mathrm{I}}(i)}\underline{m}_{\mathrm{B1}+}(i)\right) > \frac{x_{i+1}\,\rho_{\mathrm{I}}(i+1)}{\overline{m}_{\mathrm{I}}(i+1)}\Big(\underline{m}_A(i) + 2\underline{m}_{\mathrm{C}+}(i)\Big),$$

and $\underline{m}_{\mathrm{B1}+}(i)/\overline{m}_{\mathrm{I}}(i) = o(1)$ and $\underline{m}_{\mathrm{C}+}(i) = o(\underline{m}_A(i))$. Hence (5.47) and (5.46) are satisfied for $i = i_1 - 1$.

Now assume $i \leqslant i_1 - 1$. By induction, $x_j^* > 0$ and $\rho_{\rm I}^*(j) \geqslant 0$ for all $j \geqslant i+1$. Thus, $\kappa_2 > 0$ where κ_2 is as below (5.40). It is easy to see that $\kappa_1 < 1$ by the definition of $\underline{m}_{\rm B1+}(i)$ and $\overline{m}_{\rm I}(i)$. Thus, $x_i^* > 0$ by (5.43). Since $\rho_{\rm I}(i+2) \geqslant 0$ and $x_{i+2}^* \geqslant 0$ by induction, (5.48) follows and thus (5.47) holds for i.

Substituting (5.43) to (5.42) we have

$$\rho_{\rm III}(i) = \frac{x_{i+1}}{x_i}\frac{\overline{m}_{\rm III+}(i)}{\overline{m}_{\rm I}(i+1)}\rho_{\rm I}(i+1) \leqslant \frac{2.3(i+1)}{d\Delta}\,\frac{\overline{m}_{\rm III+}(i)}{\overline{m}_{\rm I}(i+1)} \leqslant \frac{1.2\Delta^2}{dn}. \tag{5.49}$$

as we have already established (5.47) for i, and $\rho_{\rm I}(i+1) \leqslant 1$ by induction. Substituting the bound to (5.41), we have $\rho_{\rm I}(i) = 1 - \epsilon - 2\rho_{\rm III}(i) = 1 - o(1) \geqslant 0$.

Finally, by (5.17)–(5.22) and using (4.1), we have

$$\begin{aligned}
\rho_{\rm IIb+}(i) &= \rho_{\rm IIb-}(i) = \frac{x_{i+2}}{x_i}\frac{\rho_{\rm I}(i+2)\,\overline{m}_{\rm IIb+}(i)}{\overline{m}_{\rm I}(i+2)\,\widehat{\underline{m}}_{\rm IIb+}(i+2)} < 4d\Delta/n^2, & (5.50)\\
\rho_{\rm IIa+}(i) &= \rho_{\rm IIa-}(i) = \frac{x_{i+1}}{x_i}\frac{\rho_{\rm I}(i+1)\,\overline{m}_{\rm IIa+}(i)}{\overline{m}_{\rm I}(i+1)} \leqslant 2\Delta^2/n^2, & (5.51)\\
\rho_{\rm IIc+}(i) &= \rho_{\rm IIc-}(i) = \frac{x_{i+3}}{x_i}\frac{\rho_{\rm I}(i+3)\,\overline{m}_{\rm IIc+}(i)}{\overline{m}_{\rm I}(i+3)\,\widehat{\underline{m}}_{\rm IIc+}(i+3)} = O(\Delta^2 d/n^3) < \tfrac{1}{2}\,\Delta^2/n^2. & (5.52)
\end{aligned}$$

With these bounds on $\rho_\tau(i)$ for $\tau \in \{\text{IIa}\pm, \text{IIb}\pm, \text{IIc}\pm\}$, we can easily verify that

$$\sum_{\tau\in\{\text{IIa}\pm,\text{IIb}\pm,\text{IIc}\pm\}} \rho_\tau^*(i) < \epsilon.$$

This verifies (5.46), completing the proof. □

Now we define parameters $\rho_\tau(i)$ in `FactorUniform` by $\rho_\tau(i) = \rho_\tau^*(i)$, where $\big(\rho_\tau^*(i)\big)$ is the unique solution guaranteed by Lemma 5.20. This completes the definition of `FactorUniform`. Next we show that the output of `FactorUniform` is correct.

Lemma 5.21. *The output of* `FactorUniform` *is a uniformly random d-factor of H_n.*

Proof. By the definition of `FactorUniform`, the output graph contains no red edges. Thus it is a d-factor of H_n. Recall that the parameters $\rho_\tau(i)$ are set according to the solution of (5.15)–(5.28). Hence, the expected number of times that a graph in $\mathcal{S}_0$ is visited equals σ_0, by (5.27), which is the same for every graph in $\mathcal{S}_0$. It follows that for every d-factor G of H_n, the probability that G is the output of `FactorUniform` is $\sigma_0\rho_{\rm I}(0)$, which, again, is independent of G. Hence, `FactorUniform` is a uniform sampler for d-factors of H_n. □

5.5 Rejection probability and number of switching steps

We bound the number of switching steps performed by `FactorUniform` and the probability of any rejection in `FactorUniform`. The proofs of Lemmas 5.22 and 5.23 are standard, and very similar to those in [9].

Lemma 5.22. `FactorUniform` *performs at most $O(i_1)$ switching steps in expectation and with high probability.*

Proof. The proof is almost identical to [9, Lemma 8]. We omit the details here and only sketch the main idea. It is easy to verify that with probability $1-o(1)$, a Type I Class A switching is performed in each step. Thus, we can easily bound the probability by $o(1)$ that more than 5% of steps are implemented with a switching that is not of Type I Class A. With a Type I Class A switching, the number of red edges reduces by exactly one. On the other hand, the number of red edges can increase by at most 3 in each step. Since initially there are at most i_1 red edges, it follows immediately that with high probability the number of switching steps performed by `FactorUniform` is $O(i_1)$. □

Lemma 5.23. *Assume that $\epsilon\, i_1 = o(1)$. Then the probability of a t-rejection, or an f-rejection, or a b-rejection, or pre-b-rejection occurring in* `FactorUniform` *is $o(1)$.*

Proof. By Lemma 5.20, the probability of a t-rejection in each step is at most ϵ. Thus, the probability of a t-rejection in `FactorUniform` is $O(\epsilon i_1) = o(1)$ by Lemma 5.22.

Next, consider f-rejections. Let G_t be the graph obtained after t switching steps. Given $G_t = G \in \mathcal{S}_i$, the probability that an f-rejection occurs at step $t+1$ is

$$\sum_{\tau\in\Gamma} \rho_\tau(i)\left(1-\frac{f_\tau(G)}{\overline{m}_\tau(i)}\right)$$

Summing over all $G\in\mathcal{S}_i$ and summing over all steps t, the probability of an f-rejection in `FactorUniform` is at most

$$\sum_{t\geqslant 0}\sum_{0\leqslant i\leqslant i_1}\sum_{G\in\mathcal{S}_i}\mathbb{P}(G_t=G)\sum_{\tau\in\Gamma}\rho_\tau(i)\left(1-\frac{f_\tau(G)}{\overline{m}_\tau(i)}\right).$$

Next, we verify that for every $\tau\in\Gamma$ and for every $i\leqslant i_1$,

$$\rho_\tau(i)\left(1-\frac{f_\tau(G)}{\overline{m}_\tau(i)}\right) = O\left(\frac{d^2+\Delta^2}{n^2}+\frac{1}{n}+\frac{\Delta^2(d+\Delta)}{dn^2}\right).$$

For $\tau\notin\Gamma\backslash\{\text{I,III}\pm\}$, we use the bounds in (5.50)–(5.52) for $\rho_\tau(i)$ and the trivial bound 1 for $1-f_\tau(G)/\overline{m}_\tau(i)$. For $\tau=\text{III}\pm$, we use (5.49) for $\rho_\tau(i)$ and Lemma 5.19 for $1-f_\tau(G)/\overline{m}_\tau(i)$. Lastly, for $\tau = I$, we use the trivial bound 1 for $\rho_{\text{I}}(i)$ and Lemma 5.12 for $1-f_{\text{I}}(G)/\overline{m}_{\text{I}}(i)$. These yield the desired bound above.

We also have

$$\sum_{t\geqslant 0}\sum_{0\leqslant i\leqslant i_1}\sum_{G\in\mathcal{S}_i}\mathbb{P}(G_t=G) = \sum_{0\leqslant i\leqslant i_1}\sum_{G\in\mathcal{S}_i}\sum_{t\geqslant 0}\mathbb{P}(G_t=G) = \sum_{0\leqslant i\leqslant i_1}\sigma_i|\mathcal{S}_i|,$$

which is the number of switching steps in `FactorUniform`. By Lemma 5.22, this is $O(i_1)$ in expectation and with high probability. Thus, the probability of an f-rejection is at most

$$O\left(\frac{d^2+\Delta^2}{n^2}+\frac{1}{n}+\frac{\Delta^2(d+\Delta)}{dn^2}\right)O(i_1) = O\left(\frac{(d^2+\Delta^2)d\Delta}{n^2}+\frac{d\Delta}{n}+\frac{\Delta^3(d+\Delta)}{n^2}\right),$$

and this is $o(1)$ when $d^2 + \Delta^2 = o(n)$.

Next, consider pre-b-rejections. Pre-b-rejections can happen when a switching of type $\tau \in \{\text{IIb}\pm,\, \text{IIc}\pm\}$ is performed. Given $G_t = G \in \mathcal{S}_i$, the probability that a pre-b-rejection occurs at step $t+1$ is

$$\sum_{\tau \in \{\text{IIb}\pm,\, \text{IIc}\pm\}} \rho_\tau(i) \left(1 - \frac{\widehat{m}_\tau(i)}{\widehat{b}_\tau(G, \boldsymbol{v})}\right).$$

By Lemma 5.17,

$$\left(1 - \frac{\widehat{m}_\tau(i)}{\widehat{b}_\tau(G, \boldsymbol{v})}\right) = O((d+\Delta)/n).$$

Thus, by Lemma 5.20, the probability of a pre-b-rejection occurring at step $t+1$, given $G_t = G$, is

$$\sum_{\tau \in \{\text{IIb}\pm,\, \text{IIc}\pm\}} O(\epsilon\,(d+\Delta)/n) = O\left(\frac{d^3 + \Delta^3}{n^3}\right).$$

Arguing as above, the probability of any pre-b-rejection during `FactorUniform` is at most

$$\begin{aligned}\sum_{t \geqslant 0} \sum_{i \leqslant i_1} \sum_{G \in \mathcal{S}_i} \mathbb{P}(G_t = G) \cdot O\left(\frac{d^3+\Delta^3}{n^3}\right) &= O\left(\frac{d^3+\Delta^3}{n^3}\right) i_1 \\ &= O\left(\frac{(d^3+\Delta^3)d\Delta}{n^3}\right),\end{aligned}$$

and this is $o(1)$ when $d^2 + \Delta^2 = o(n)$.

Finally, we consider b-rejections. Let $\Psi_{\tau,\alpha}(G, G')$ denote the set of switchings of Type τ and Class α which convert G to G'. An element of $\Psi_{\tau,\alpha}(G, G')$ is either an 8-tuple $\boldsymbol{v}$ such that $(G, \boldsymbol{v}) \mapsto G'$ is a switching of Type τ and Class α, if $(\tau, \alpha) \notin \{\text{IIb}\pm, \text{B1}\pm), (\text{IIc}\pm, \text{B1}\pm)\}$ or a pair $(\boldsymbol{v}, \boldsymbol{y})$ which determines a switching $(G, \boldsymbol{v}, \boldsymbol{y}) \mapsto G'$ of Type τ and Class B1$\pm$, where $\tau \in \{\text{IIb}\pm,\, \text{IIc}\pm\}$. Let

$$\Psi_{\tau,\alpha}(G') = \bigcup_G \Psi_{\tau,\alpha}(G, G')$$

be the set of switchings into G' which are of Type τ and Class α. For any $S \in \Psi_{\tau,\alpha}(G, G')$ where $G \in \mathcal{S}_i$ and $G' \in \mathcal{S}_{i'}$ and $\Psi_{\tau,\alpha}(G, G') \neq \varnothing$, the probability that S is performed and b-rejected, given $G_t = G$, is

$$\frac{\rho_\tau(i)}{\overline{m}_\tau(i)} \left(1 - \frac{m_\alpha(i')}{b_\alpha(G')}\right).$$

Then the probability that a b-rejection ever occurs in `FactorUniform` is

$$\sum_\alpha \sum_\tau \sum_{t \geqslant 1} \sum_{i' \leqslant i_1} \sum_{G' \in \mathcal{S}_{i'}} \sum_G \mathbb{P}(G_{t-1} = G)\, |\Psi_{\tau,\alpha}(G, G')|\, \frac{\rho_\tau(i)}{\overline{m}_\tau(i)} \left(1 - \frac{m_\alpha(i')}{b_\alpha(G')}\right).$$

Here i is the index of the strata which contains G, which is determined by τ, α and i'. Hence, the above sum is

$$\begin{aligned}&\sum_{\alpha}\sum_{i'\leqslant i_1}\sum_{\tau}\frac{\rho_\tau(i)}{\overline{m}_\tau(i)}\sum_{G'\in\mathcal{S}_{i'}}\frac{b_\alpha(G')-\underline{m}_\alpha(i')}{b_\alpha(G')}\sum_{G}|\Psi_{\tau,\alpha}(G,G')|\sum_{t\geqslant 1}\mathbb{P}(G_{t-1}=G)\\ =\ &\sum_{\alpha}\sum_{i'\leqslant i_1}\sum_{\tau}\frac{\rho_\tau(i)\sigma_i}{\overline{m}_\tau(i)}\sum_{G'\in\mathcal{S}_{i'}}\frac{b_\alpha(G')-\underline{m}_\alpha(i')}{b_\alpha(G')}|\Psi_{\tau,\alpha}(G')|,\end{aligned}$$

since $\sum_{t\geqslant 1}\mathbb{P}(G_{t-1}=G)=\sigma_i$. By the design of the algorithm, the expected number of times that G' is reached via a Type τ, Class α switching equals $q_\alpha(i')$ for all relevant τ, as displayed in (5.15)–(5.26). Therefore, for every τ,

$$\frac{\rho_\tau(i)\sigma_i}{\overline{m}_\tau(i)}=q_\alpha(i')=\frac{\rho_{\mathrm{I}}(i'+1)\sigma_{i'+1}}{\overline{m}_\tau(i'+1)}.$$

Thus the above summation is

$$\sum_{\alpha}\sum_{i'\leqslant i_1}\frac{\rho_{\mathrm{I}}(i'+1)\,\sigma_{i'+1}}{\overline{m}_{\mathrm{I}}(i'+1)}\sum_{G'\in\mathcal{S}_{i'}}\frac{b_\alpha(G')-\underline{m}_\alpha(i')}{b_\alpha(G')}\sum_{\tau}|\Psi_{\tau,\alpha}(G')|.$$

Since $\sum_\tau|\Psi_{\tau,\alpha}(G')|=b_\alpha(G')$, by definition of $b_\alpha(G')$, the probability of a b-rejection in `FactorUniform` is

$$\begin{aligned}&\sum_{\alpha}\sum_{i'\leqslant i_1}\frac{\rho_{\mathrm{I}}(i'+1)\sigma_{i'+1}}{\overline{m}_{\mathrm{I}}(i'+1)}\sum_{G'\in\mathcal{S}_{i'}}\Big(b_\alpha(G')-\underline{m}_\alpha(i')\Big)\\ &=\sum_{\alpha}\sum_{i'\leqslant i_1}\frac{\rho_{\mathrm{I}}(i'+1)\sigma_{i'+1}}{\overline{m}_{\mathrm{I}}(i'+1)}|\mathcal{S}_{i'}|\Big(\mathbb{E}b_\alpha(G')-\underline{m}_\alpha(i')\Big)\\ &\leqslant\sum_{\alpha}\sum_{i'\leqslant i_1}\frac{(i'+1)\sigma_{i'}}{d\Delta\cdot\overline{m}_{\mathrm{I}}(i'+1)}|\mathcal{S}_{i'}|\Big(\mathbb{E}b_\alpha(G')-\underline{m}_\alpha(i')\Big)\\ &\qquad\qquad(\text{since }\rho_{\mathrm{I}}(i'+1)\leqslant 1\text{ and }\sigma_{i'+1}=O((i+1)\sigma_{i'}/d\Delta)\text{ by Lemma 5.20})\\ &\leqslant\frac{1}{d\Delta(dn)^3}\sum_{\alpha}\sum_{i'\leqslant i_1}\sigma_{i'}\,|\mathcal{S}_{i'}|\Big(\mathbb{E}b_\alpha(G')-\underline{m}_\alpha(i')\Big),\end{aligned}\tag{5.53}$$

where $\mathbb{E}b_\alpha(G')$ is the expectation of $b_\alpha(G')$ on a uniformly random $G'\in\mathcal{S}_{i'}$.

For $\alpha=\mathrm{A}$, by Lemma 5.15,

$$\mathbb{E}b_{\mathrm{A}}(G')-\underline{m}_{\mathrm{A}}(i')=O((d^2+\Delta^2)/n^2+1/n)\,\underline{m}_{\mathrm{A}}(i')=O((d^2+\Delta^2)/n^2+1/n)\Delta nd^2(dn)^2.$$

Thus, the contribution to (5.53) from $\alpha=\mathrm{A}$ is

$$O\left(\Big(\frac{d^2+\Delta^2}{n^2}+\frac{1}{n}\Big)\frac{\Delta nd^2(dn)^2}{d\Delta(dn)^3}\right)\sum_{i'\leqslant i_1}\sigma_{i'}|\mathcal{S}_{i'}|=o(1),$$

as $\sum_{i'\leqslant i_1} \sigma_{i'}|\mathcal{S}_{i'}| = O(i_1) = O(d\Delta)$.

For $\alpha \in \{\mathrm{B1}\pm\}$, by Lemma 5.18,

$$\begin{aligned}\mathbb{E}b_\alpha(G') - \underline{m}_\alpha(i') &= O(1/d + (d+\Delta)/n)\,\underline{m}_\alpha(i') = O(1/d + (d+\Delta)/n)i_1\Delta d(dn)^2\\ &= O(1/d + (d+\Delta)/n)d^4\Delta^2 n^2.\end{aligned}$$

Thus, the contribution to (5.53) from $\alpha \in \{\mathrm{B1}\pm\}$ is

$$\frac{d^4\Delta^2 n^2}{d\Delta(dn)^3}\cdot O(1/d + (d+\Delta)/n)\,i_1 = O\left(\frac{d^5\Delta^3 n^2}{d\Delta(dn)^3}\cdot\left(\frac{1}{d}+\frac{d+\Delta}{n}\right)\right) = o(1).$$

For $\alpha \in \{\mathrm{B2}\pm\}$, by Lemma 5.16,

$$\mathbb{E}b_\alpha(G') - \underline{m}_\alpha(i') = O((d+\Delta)/n)\,\underline{m}_\alpha(i') = O((d+\Delta)/n)\Delta^2 d^4 n^2.$$

Thus, the contribution to (5.53) from $\alpha \in \{\mathrm{B2}\pm\}$ is

$$\frac{\Delta^2 d^4 n^2}{d\Delta(dn)^3}\cdot O((d+\Delta)/n)i_1 = O\left(\frac{d^5\Delta^3 n^2}{d\Delta(dn)^3}\cdot\left(\frac{d+\Delta}{n}\right)\right) = o(1).$$

For $\alpha = \mathrm{C}\pm$, by Lemma 5.19,

$$\mathbb{E}b_\alpha(G') - \underline{m}_\alpha(i') = O((d+\Delta)/n)\,\underline{m}_\alpha(i') = O((d+\Delta)/n)\Delta^3 d^3 n^2.$$

Thus, the contribution to (5.53) from $\alpha \in \{\mathrm{C}\pm\}$ is

$$\frac{\Delta^3 d^3 n^2}{d\Delta(dn)^3}\cdot O((d+\Delta)/n)\,i_1 = O\left(\frac{d^4\Delta^4 n^2}{d\Delta(dn)^3}\cdot\left(\frac{d+\Delta}{n}\right)\right) = o(1).$$

This completes the proof that the probability of any b-rejection in `FactorUniform` is $o(1)$. □

6 Deferred analysis of time complexity and distance from uniform

In this section we present the deferred analysis of the time complexity of algorithms `FactorEasy` and `FactorUniform`, and the proof that the output of `FactorApprox` is within $o(1)$ of uniform. As usual, asymptotics are as $n \to \infty$ where $d = d(n)$ and $\Delta = \Delta(n)$ satisfy the assumptions of the relevant theorem.

Lemma 6.24. `FactorEasy` *can be implemented so that if there are $O(1)$ restarts during its run, its time complexity is $O((d+\Delta)^3 n)$.*

Proof. In each iteration, the switching of a bounded number of edges can be done in $O(1)$ time. The time-consuming part is to compute $b(G)$ to determine the probability of a b-rejection.

First consider the initial graph G produced by REG. To compute $b(G)$ for the initial graph, we use brute force to search for all possible choices of v_5, v_0, v_1, v_2. This can be done in time $O(d^2\Delta n)$. Denote the number of choices by X. Multiplying X by $dn - 2i$ gives the first estimate for $b(G)$. Next we accurately compute $b(G)$ using inclusion-exclusion. The choices of v_3 and v_4 must satisfy a set U of constraints. We can count the number of choices which satisfy all constraints using inclusion-exclusion. The inclusion-exclusion argument involves a bounded number of terms counting choices where a subset $W \subseteq U$ of constraints are violated. We will show that each such term can be computed in time $O((d+\Delta)^3 n)$ with the aid of a proper data structure.

Let $\widetilde{G}$ be the supergraph of G consisting of G together with all red edges in $K_n \setminus G$. We use $\widetilde{\text{red}}$ for the colour of edges in $\widetilde{G}$ which are not in G. When computing X using brute force search, we can record the number of 3-paths in $\widetilde{G}$ between any two vertices of any type (for example, red-black-red, or black-red-black, or black-$\widetilde{\text{red}}$-black); we can also record the number of 3-paths and 2-paths in $\widetilde{G}$ of any given type starting from any given vertex. The time complexity for computing all these numbers is $O((d+\Delta)^3 n)$ since the maximum degree in $\widetilde{G}$ is bounded above by $d + \Delta$. The number of other local structures of at most 4 vertices can be computed and recorded within this time complexity bound, that is, triangles, 4-cycles, etc. We can also record lists of pairs of vertices which are joined by a 3-path, or 2-path, or an edge, within the same time complexity.

Given W, let b_W be the choices of the sequence of the six vertices that violate constraints in W (here constraints in $U \setminus W$ may or may not be violated). Since the structure counted by b_W uses up to six vertices, it is easy to see that b_W can be computed using the numbers we have recorded. For instance, if $W = \varnothing$ then $b_W = X(dn - 2i)$. If $W = \{v_4v_5 \text{ is a black non-edge}\}$, then $b_W = b_{W,1} + b_{W,2} + b_{W,3}$, where $b_{W,1}$ counts those choices where W is violated by taking v_4v_5 as a red edge; $b_{W,2}$ counts those choices where v_4v_5 is a black edge, and $b_{W,3}$ counts those choices where v_4v_5 is a red non-edge. In each case, we can run through n choices for v_5, and compute $b_{W,i}$ using the number of 3-paths and 2-paths starting from v_5 that have been recorded. The time complexity is then $O(n)$. For every other W it is easy to check that a similar scheme works. Thus, it takes $O((d+\Delta)^3 n)$ time to compute $b(G)$ for the initial graph G.

Next suppose that G is produced by a switching step, during the run of `FactorEasy`. We do not need to recompute $b(G)$ from scratch: instead, we can update the data recorded in our data structure very efficiently, because only 3 new edges are added, and 3 edges are deleted. Since the data we store are counts of structures involving only up to 4 vertices, changing each edge will alter at most $O((d+\Delta)^2)$ entries. For each entry change, we can update $b(G)$ by updating the corresponding b_W terms in the inclusion-exclusion formula. Thus the time complexity for computing $b(G)$ is $O((d+\Delta)^2)$ after each subsequent switching step and there are $O(d\Delta)$ switching steps in expectation, since `FactorEasy` restarts

$O(1)$ times in expectation. Thus the total time complexity for `FactorEasy` is

$$O((d+\Delta)^3 n + d\Delta(d+\Delta)^2) = O((d+\Delta)^3 n)$$

in expectation, completing the proof. □

For convenience, we restate Theorem 1.2 below.

Theorem 1.2. *Let H_n be an $(n-\Delta-1)$-regular graph on n vertices. The algorithm* `FactorUniform` *uniformly generates a d-factor of H_n. If $d^2+\Delta^2 = o(n)$ then the time complexity of* `FactorUniform` *is $O((d+\Delta)^4 n + d^3 n \log n)$ a.a.s., and is $O(M)$ in expectation, where*

$$M = O\Big((d+\Delta)^4(n+\Delta^3) + (d+\Delta)^8 d^2\Delta^2/n + (d+\Delta)^{10} d^2\Delta^3/n^2\Big).$$

Proof. In Lemma 5.21 we have shown that `FactorUniform` is a uniform sampler. It only remains to prove the efficiency. By Corollary 4.10, `FactorUniform` restarts only $O(1)$ times in expectation and $O(\log n)$ times a.a.s. before finding a d-regular graph containing at most i_1 red edges. The total time complexity for finding such a graph is $O(d^3 n)$ in expectation, and $O(d^3 n \log n)$ a.a.s.. By Lemma 5.23, the probability that `FactorUniform` restarts afterwards is $o(1)$. Thus, we only need to bound the remaining runtime of `FactorUniform` assuming no rejections. By Lemma 5.22, after finding a d-regular graph with at most i_1 red edges, `FactorUniform` will perform $O(i_1)$ switching steps in expectation and with high probability. In each switching step, the most time-consuming part is to compute $f_\tau(G)$, $\widehat{b}_\tau(G,\boldsymbol{v})$, and $b_\alpha(G)$ for $\tau \in \{\mathrm{I}, \mathrm{IIa}\pm, \mathrm{IIb}\pm, \mathrm{IIc}\pm, \mathrm{III}\pm\}$ and $\alpha \in \{\mathrm{A}, \mathrm{B1}\pm, \mathrm{B2}\pm, \mathrm{C}\pm\}$.

We first bound the a.a.s. time complexity. Note that $f_\tau(G)$ and $\widehat{b}_\tau(G,\boldsymbol{v})$ will only need to be evaluated once a type τ switching is performed. By (5.30) and (5.44), the probability that a type τ switching is ever performed in `FactorUniform` for any $\tau \notin \{\mathrm{I}, \mathrm{III}\pm\}$ is $O(\epsilon i_1) = o(1)$. It follows immediately that a.a.s., only $f_\tau(G)$ for $\tau \in \{\mathrm{I}, \mathrm{III}\pm\}$ and $b_\alpha(G)$, $\alpha \in \{\mathrm{A}, \mathrm{B1}\pm, \mathrm{B2}\pm, \mathrm{C}\pm\}$, will ever be computed during the implementation of `FactorUniform`.

First consider $f_\tau(G)$ for $\tau = \mathrm{I}$. We want to count choices of $(v_0, \ldots, v_7)$ such that v_0v_1 is a red edge in G, v_2v_3 and v_6v_7 are edges (red or black) in G, and v_4v_5 is a black edge in G, satisfying a set U of constraints (that is, no vertex collision except for $v_2 = v_7$ and certain edges are forbidden in G and must be with certain colour in K_n). As before, using inclusion-exclusion, we can express this number by b_W, $W \subseteq U$, where b_W is the number of choices where the conditions in W are violated. We use similar data structures as in Theorem 1.1, but we record counts of structures containing up to 5 vertices. Thus the time complexity for constructing the data structures is $O((d+\Delta)^4 n)$ in the first iteration. It is easy to see, as in the proof of Theorem 1.1, that all terms b_W in the inclusion-exclusion can be computed using the recorded data. Moreover, it takes $O((d+\Delta)^3)$ time to update the data structure after each subsequent switching step. Thus, the total time complexity for computing $f_{\mathrm{I}}(G)$ throughout `FactorUniform` is $O((d+\Delta)^4n+d\Delta(d+\Delta)^3) = O((d+\Delta)^4n)$. The same time complexity bound holds for computing $f_{\mathrm{III}+}(G)$ and $f_{\mathrm{III}-}(G)$ throughout `FactorUniform`, as the same number of vertices are involved in a Type III± switching as in a Type I switching.

Next we consider $b_\alpha(G)$. Similar arguments as for $f_{\mathrm{I}}(G)$ show that the time complexity of computing $b_\alpha(G)$ for every α throughout `FactorUniform` is at most $O((d+\Delta)^4n)$. The Gao–Wormald algorithm [9], used to produce the initial d-regular graph, has time complexity $O(d^3n)$ in expectation. Hence a.a.s. the time complexity to produce the initial d-regular graph is $O(d^3n\log n)$. Therefore, the a.a.s. time complexity bound for `FactorUniform` is $O(d^3n\log n + (d+\Delta)^4n)$.

Now we consider the time complexity in expectation. To do this, we obtain an upper bound for the time complexity of computing $f_\tau(G)$ and $\widehat{b}_\tau(G,\boldsymbol{v})$, $\tau \notin \{\mathrm{I}, \mathrm{III}\pm\}$, and then multiply by the probability that the switching type τ is chosen in a single step, and finally multiply by $O(i_1) = O(d\Delta)$, which is an upper bound for the expected number of switching steps performed by `FactorUniform`. These switchings are performed rarely, so we do not attempt to update the data after every switching step. Instead we simply reconstruct the data structure whenever it is needed. Obviously for d and Δ in different ranges, different counting schemes can be used to optimise the runtime: we have not attempted this. Here we simply use the scheme which naturally extends that given in Theorem 1.1.

For each τ, we use data structures to record counts of connected small structures up to j vertices, where

$$j = \begin{cases} 5 & \text{for } \tau \in \{\mathrm{IIa}\pm\}, \\ 9 & \text{for } \tau \in \{\mathrm{IIb}\pm\}, \\ 11 & \text{for } \tau \in \{\mathrm{IIc}\pm\}. \end{cases}$$

This leads to the following bounds on the complexity of computing $f_\tau(G)$ for a particular G, in these cases:

$$\mathrm{IIa}\pm: \ O((d+\Delta)^4n); \qquad \mathrm{IIb}\pm: \ O((d+\Delta)^8n); \qquad \mathrm{IIc}\pm: \ O((d+\Delta)^{10}n).$$

Now for $\tau \neq \mathrm{I}$, the Type τ switchings are only implemented occasionally. Let $\rho^*_\tau = \max_{0\leqslant i\leqslant i_1} \rho_\tau(i)$. Multiplying the above bounds by $O(i_1)\rho^*_\tau$, using (5.49)–(5.52), yields the following overall bounds on the expected time complexity for computing $f_\tau(G)$ during `FactorUniform`:

$$\begin{array}{llll} \mathrm{IIa}\pm: & O((d+\Delta)^4d\Delta^3/n); & \mathrm{IIb}\pm: & O((d+\Delta)^8d^2\Delta^2/n); \\ \mathrm{IIc}\pm: & O((d+\Delta)^{10}d^2\Delta^3/n^2); & \mathrm{III}\pm: & O((d+\Delta)^4\Delta^3). \end{array}$$

Combining the contribution from every type τ, the expected time complexity for computing $f_\tau(G)$ throughout `FactorUniform` is bounded above by

$$O\Big((d+\Delta)^4(n+\Delta^3) + (d+\Delta)^8d^2\Delta^2/n + (d+\Delta)^{10}d^2\Delta^3/n^2\Big). \tag{6.1}$$

Finally, we consider computation of $\widehat{b}_\tau(G, \boldsymbol{v})$ for $\tau \in \{\text{IIb}\pm, \text{IIc}\pm\}$, for a given 8-tuple $\boldsymbol{v}$. (Recall that these are the only switching types which have pre-b-rejections.) Due to (6.1) we only need rough bounds for the time complexity of computing $\widehat{b}_\tau(G, \boldsymbol{v})$. For Type IIb±, it is sufficient to use a data structure to record counts of connected structures involving up to 5 vertices, or up to 9 vertices, one of which belongs to $\boldsymbol{v}$. The runtime to construct data structures of the first type is $O((d + \Delta)^4 n)$; and $O((d + \Delta)^8)$ for the second type. For Type IIc±, similar arguments give an upper bound of

$$O\big((d + \Delta)^6 n + (d + \Delta)^{12}\big)$$

on the complexity of constructing the data structure. Multiplying these bounds by $O(i_1)\rho^*_\tau$, using (5.50) and (5.52), yields the following upper bound on the complexity of computing $\widehat{b}_\tau(G, \boldsymbol{v})$ during the implementation of `FactorUniform`:

$$\begin{aligned} \text{IIb}\pm: \quad & O\left(\frac{d^2\Delta^2(d+\Delta)^4}{n} + \frac{d^2\Delta^2(d+\Delta)^8}{n^2}\right); \\ \text{IIc}\pm: \quad & O\left(\frac{d^2\Delta^3(d+\Delta)^6}{n^2} + \frac{d^2\Delta^3(d+\Delta)^{12}}{n^3}\right). \end{aligned}$$

Since $d^2 + \Delta^2 = o(n)$, the above terms are dominated by (6.1). Thus, combining everything together, the expected time complexity of `FactorUniform` is bounded by (6.1). □

Finally, we restate and prove Lemma 4.11.

Lemma 6.25. *Under the conditions of Theorem 1.3, the output of* `FactorApprox` *is a random d-factor of H_n whose distribution differs from the uniform distribution by $o(1)$ in total variation distance.*

Proof. Recall that `FactorUniform` calls REG to generate a uniformly random d-regular graph on $[n]$, whereas `FactorApprox` calls REG* which generates an approximately uniformly random d-regular graph. It was proved in [9, Section 10] that REG and REG* can be coupled so that with probability $1 - o(1)$ they have the same output. Assume REG and REG* both output G which contains at most i_1 red edges. Consider continuing the run of `FactorUniform` but restricting the choice of type to $\tau \in \{\text{I}, \text{III}\pm\}$ and ignoring all t-rejections, f-rejections, b-rejections and pre-b-rejections. Recall that Type III± switchings do not switch any edges. If we ignore the f-rejections in the implementation of Type III± switchings then the switching has no effect and we may simply skip that step. Hence this modification of `FactorUniform` behaves identically to `FactorApprox`, that is, by repeatedly performing valid Type I switchings.

Therefore, we can couple the implementation of `FactorUniform` and `FactorApprox` such that `FactorUniform` and `FactorApprox` output the same graph G as long as no rejections occur in `FactorUniform` and no types of switchings other than $\text{I}, \text{III}\pm$ are chosen in `FactorUniform`. By Lemma 5.23, the probability of performing any rejection in `FactorUniform` is $o(1)$. By (5.30) and (5.44), the probability of performing any switchings in `FactorUniform` of type other than I, III$\pm$ is $O(\epsilon) = o(1)$. Hence, the total variation distance between the output of `FactorApprox`, and that of `FactorUniform`, which is uniform, is $o(1)$. Here $o(1)$ also accounts for the probability that the coupled REG and REG* have distinct outputs. $\square$

Acknowledgements

We thank Brendan McKay for his idea of permitting $v_2 = v_5$ in a 3-edge-switching and the similar treatment in a 4-edge-switching. This crucial idea significantly simplified our algorithms and analysis. We also thank the referee for their helpful comments.

References

[1] A. Arman, P. Gao and N. Wormald, Fast uniform generation of random graphs with given degree sequences. Preprint, 2019. arXiv:1905.03446

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

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

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

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

[6] C. Cooper, M.E. Dyer and C. Greenhill, Corrigendum: Sampling regular graphs and a peer-to-peer network. arXiv:1203.6111

[7] P.L. Erdős, S.Z. Kiss, I. Miklós and L. Soukup, Approximate counting of graphical realizations, *PLoS ONE* **10** (2015), e0131300.

[8] P. Gao, Uniform generation of d-factors in dense host graphs, *Graphs Combin.* **30(3)** (2014), 581–589.

[9] P. Gao and N. Wormald, Uniform generation of random regular graphs, *SIAM J. Comput.* **46(4)** (2017), 1395–1427.

[10] P. Gao and N. Wormald, Uniform generation of random graphs with power-law degree sequences, Proceedings of SODA 2018, 1741–1758.

[11] M. Jerrum and A. Sinclair, Approximating the permanent, *SIAM J. Comput.* **18** (1989), 1149–1178.

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

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

[14] A. Liebenau and N. Wormald, Asymptotic enumeration of graphs by degree sequence, and the degree sequence of a random graph. arXiv:1702.08373

[15] B.D. McKay, Asymptotics for symmetric 0-1 matrices with prescribed row sums, *Ars Combin.* **19A** (1985), 15–25.

[16] B.D. McKay, Subgrpahs of dense random graphs with specified degrees, *Combin. Probab. Comput.* **20** (2011), 413–433.

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

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

[19] B.D. McKay and N.C. Wormald, Asymptotic enumeration by degree sequence of graphs of high degree, *European J. Combin.* **11**, 1990, 565–580.

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

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

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