PLOS ONE

RESEARCH ARTICLE

Approximate Counting of Graphical Realizations

Péter L. Erdős[1☯¤*], **Sándor Z. Kiss**[2,3☯], **István Miklós**[1,2☯], **Lajos Soukup**[1☯]

1 Alfréd Rényi Institute of Mathematics, Hungarian Academy of Sciences, Budapest, Hungary, **2** Institute for Computer Science and Control, Hungarian Academy of Sciences, Budapest, Hungary, **3** Department of Algebra, University of Technology and Economics, Budapest, Hungary

☯ These authors contributed equally to this work.
¤ Current address: Reáltanoda u 13-5, Budapest, H-1053, Hungary
* erdos.peter@renyi.mta.hu

Citation: Erdős PL, Kiss SZ, Miklós I, Soukup L (2015) Approximate Counting of Graphical Realizations. PLoS ONE 10(7): e0131300. doi:10.1371/journal.pone.0131300

Editor: Arndt von Haeseler, Max F. Perutz Laboratories, AUSTRIA

Received: March 1, 2015

Accepted: May 12, 2015

Published: July 10, 2015

Data Availability Statement: All relevant data are within the paper.

Funding: PLE and IM acknowledge financial support from grant #FA9550-12-1-0405 from the U.S. Air Force Office of Scientific Research (AFOSR) and the Defense Advanced Research Projects Agency (DARPA). PLE was partly supported by the Alexander von Humboldt-Foundation, when this author visited Universität Hamburg in Fall of 2014. SZK was partly supported by Hungarian NSF, under contract K77476 and NK105645. IM was partly supported by Hungarian NSF, under contract PD84297. LS was partly supported by Hungarian NSF, under contract NK 83726. The funders had no

Abstract

In 1999 Kannan, Tetali and Vempala proposed a MCMC method to uniformly sample all possible realizations of a given graphical degree sequence and conjectured its rapidly mixing nature. Recently their conjecture was proved affirmative for regular graphs (by Cooper, Dyer and Greenhill, 2007), for regular directed graphs (by Greenhill, 2011) and for half-regular bipartite graphs (by Miklós, Erdős and Soukup, 2013).

Several heuristics on counting the number of possible realizations exist (via sampling processes), and while they work well in practice, so far no approximation guarantees exist for such an approach. This paper is the first to develop a method for counting realizations with provable approximation guarantee. In fact, we solve a slightly more general problem; besides the graphical degree sequence a small set of forbidden edges is also given. We show that for the general problem (which contains the Greenhill problem and the Miklós, Erdős and Soukup problem as special cases) the derived MCMC process is rapidly mixing. Further, we show that this new problem is *self-reducible* therefore it provides a *fully polynomial randomized approximation scheme* (a.k.a. FPRAS) for counting of all realizations.

Introduction

In the Age of the Internet, network theory has been undergoing exponential growth. One of its important problems is to algorithmically construct networks (or *graphs*) with predefined parameters, or to uniformly sample networks with these parameters. For general background, the interested reader can turn to the now-classic book of Newman, Barabási and Watts ([1]) or to the more recent book of Newman ([2]).

One of the earliest and still most important problems in graph theory is uniformly sampling all possible graph realizations of given degree sequence. (For the definitions see Section "Degree sequences".) One possible method for this is a simple MCMC approach (proposed by Kannan, Tetali and Vempala [3]); take an arbitrary realization of the degree sequence, then perform a series of randomly chosen local transformations (called *swap* or *switch*). They

role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

conjectured that the process is *rapidly mixing*, i.e., a random realization is achieved after polynomial many steps.

The first result with a flawless proof in connection with this conjecture is due to Cooper, Dyer and Greenhill (2007, [4]) for the special case when the degree sequence is regular. Greenhill proved in 2011 the analogous result for (in- and out-)regular directed graphs ([5]). In 2013 Miklós, Erdős and Soukup proved the conjecture for *half-regular* bipartite graphs ([6]). Here the degree sequence on one class is regular, while there is no constraint on the other class. (A comprehensive survey on the topic is [5] or is [6].)

In modern network applications sampling the solutions is just one requirement. Sometimes the actual number of all solutions (or at least a good approximation of it) is also important. It is well known (Jerrum, Valiant and Vazirani 1986, [7]) that for *self-reducible counting problems* a rapidly mixing sampling method also provides a quick estimation of that number (with small relative error and with very high probability). Unfortunately none of the sampling problems listed above belongs to this class.

The main purpose of this paper is to remedy this imperfection. For that end we introduce a slightly more general degree sequence problem, which has all the good characteristics of the sampling procedures above (including their rapidly mixing nature), furthermore, which belongs to the class of self-reducible counting problems. This new problem is a common generalization of the regular directed graph and of the half-regular bipartite graph cases. Therefore, showing the rapidly mixing nature of the corresponding MCMC procedure provides new proofs for both problems. We prove only the existence of a polynomial upper bound on the mixing time, but do not prove the tight upper bound in Greenhill's theorem in [5].

In Section "Degree sequences" we recall the known definitions and facts on degree sequences problems in simple graphs. Then we introduce and study in full generality our proposed new *restricted degree sequence* (or **ReDeSe** for short) problem, where we deal with forbidden edges. Next, we study a specific instance of the general ReDeSe problem: bipartite degree sequences with a forbidden (but not necessarily perfect) 1-factor and a forbidden (but maybe empty) star.

In Section "Sampling" we first discuss some known results to sample degree sequence realizations. Then, we formulate our main result (Theorem 10): the proposed MCMC process on half-regular bipartite degree sequences with a well-defined small forbidden edge set is rapidly mixing. Our proof is based on Sinclair's *multicommodity flow method* ([8]), and follows closely the proof in [6]. We discuss the similarity between the two proofs in this section, while in Sections "Milestones" and "The analysis" we study the details of our new Markov chain approach which require different treatment.

In Section "Counting" we show that the studied sampling problem leads to a self-reducible counting problem. Therefore, our almost uniform sampling method provides a good approximation on the size of the set of all realizations, strengthening also Greenhill's result on regular directed graphs ([5]), and Miklós, Erdős and Soukup's result on half-regular bipartite graphs ([6]).

Degree Sequences

In this paper, every graph is assumed to be *simple*; there are no loops or multiple edges.

Degree sequences and realizations

Let V be a labeled set of n elements. The *degree sequence* $\mathbf{d}(G)$ of a graph $G = (V, E)$ is the sequence $\mathbf{d}(G)_i = d(v_i)$ of its vertex degrees. A non-negative integer sequence $\mathbf{d} = (d_1, \ldots, d_n)$ is *graphical* iff $\mathbf{d}(G) = \mathbf{d}$ for some simple graph G, and then G is a *graphical realization* of $\mathbf{d}$.

The first successful approach to decide whether a degree sequence is graphical is due to Havel ([9]), his simple but surprising observation provides immediately a greedy algorithm to build such a realization. His work was rediscovered independently later by Hakimi ([10]). Their method is based on the so-called *swap* operation. (The expressions *switch* or *rewiring* are also widely used. In this paper the word *switch* will be used for a similar, but slightly more general, operation.) The *swap* operation is defined as follows:

Let G be a simple graph and assume that a, b, c and d are different vertices. Furthermore, assume that $(a, c), (b, d) \in E(G)$ while $(b, c), (a, d) \notin E(G)$. Then

$$E(G') = E(G) \setminus \{(a, c), (b, d)\} \cup \{(b, c), (a, d)\} \quad (1)$$

is another realization of the same degree sequence. We denote this operation by $ac, bd \Rightarrow bc, ad$. Havel's nice observation is the following:

Lemma 1 (Havel, [9]). *Assume that in graph G vertex v is adjacent to vertex x but not to vertex y. Assume furthermore that $d(x) \leq d(y)$. Then one can find a swap operation which produces a new graphical realization G' of the degree sequence* $\mathbf{d}(G)$ *with the property:* $\Gamma'(v) = \Gamma(v) \setminus \{x\} \cup \{y\}$. *(These are the corresponding neighborhoods of vertex v.)*

The analogous notions for bipartite graphs are the following: if B is a simple bipartite graph then its vertex classes will be denoted by $U(B) = \{u_1, \ldots, u_k\}$ and $W(B) = \{w_1, \ldots, w_\ell\}$, and we keep the notation $V(B) = U(B) \cup W(B)$. The *bipartite degree sequence* of B, $\mathbf{D}(B)$ is defined as follows:

$$\mathbf{D}(B) = ((d(u_1), \ldots, d(u_k)), (d(w_1), \ldots, d(w_\ell))).$$

We can define the swap operation for bipartite realizations similarly to Eq (1) but we must take some care: it is not enough to assume that $(b, c), (a, d) \notin E(G)$ but we have to know that a and b are in one vertex class, and c and d are in the other.

To make clear whether a vertex pair is not forbidden to be an edge we will call a vertex pair a **chord** if it can hold an actual edge in a realization. Those pairs that cannot accommodate an edge are **non-chords**. (For example, pairs from the same vertex class of a bipartite graph are non-chords.) It can also be found in [11, Theorem 6].

Denote $\vec{G}$ a directed graph (no parallel edges, no loops, but oppositely directed edges between two vertices are allowed) with vertex set $X(\vec{G}) = \{x_1, x_2, \ldots, x_n\}$ and edge set $E(\vec{G})$. For every vertex v we associate two numbers: the *in-degree* and the *out-degree* of v.

Instead of introducing the matching definitions, we will apply the following *representation* of the directed graph $\vec{G}$: let $B(\vec{G}) = (U, W; E)$ be a bipartite graph where each class consists of one copy of every vertex of $\vec{G}$. The edges adjacent to a vertex u_x in class U represent the out-edges from x, while the edges adjacent to a vertex w_x in class W represent the in-edges to x (so a directed edge xy corresponds the edge $u_x w_y$). If a vertex has zero in- (respectively out-) degree in the directed version, then we delete the corresponding vertex from $B(\vec{G})$. (Actually, this representation is an old trick used already by Gale [12].) There is no loop in our directed graph, therefore there is no (u_x, v_x) type edge in its bipartite realization—these vertex pairs are non-chords.

Consider two different realizations, G and H, of the same degree sequence (either simple or bipartite one). It is a well-known fact that the first can be transformed to the second one (and vice versa) with consecutive swap operations. Formally, there exists a series of realizations $G = G_0, \ldots, G_{i-1}, G_i = H$, such that for each $j = 0, \ldots, i-1$ there exists a swap operation which transforms G_j into G_{j+1}.

For simple graphs this was proved already in 1891 by Petersen [13]. It can be shown that lemma 1 also provides a solution via the so-called *canonical realizations*. The analogous result

for bipartite graphs (with possible multiple edges but no loops) is due to Ryser ([14]). For simple bipartite graphs this is common-knowledge.

For directed graphs an analogous result is known. It was discovered by Kleitman and Wang (see [15]) and later rediscovered in [16]. It is important however to recognize that in case of directed graphs the "classical" Havel-type swap operation is not always adequate. To see this, it is enough to consider an example with three vertices: each vertex incident to one in-edge and one out-edge. There are exactly two possible realizations of this directed degree sequence, therefore a swap operation with six chords is necessary: we exchange three edges with three non-edges in one step. In papers [15, 16] it was shown that this extra operation is always sufficient.

Restricted degree sequences

In this paper we study the following common generalization of all previously mentioned degree sequence problems:

The **restricted degree sequence** (i.e. ReDeSe) problem $\mathbf{d}^{\mathcal{F}}$ consists of a degree sequence $\mathbf{d}$ and a set $\mathcal{F} \subset \binom{V}{2}$ of **forbidden** edges. The problem is to decide whether there is a simple graph G on V with the given degree sequence and with $E(G) \cap \mathcal{F} = \emptyset$.

It is clear that this problem is essentially identical with Tutte's f-factor problem [17]: the f-factor to be found is our degree sequence while the graph what the f-factor is searched for is the complement of the forbidden edges. Therefore Tutte's theorem and the famous blossom algorithm of Edmonds apply nicely for the ReDeSe problem. However the focus of our approach is quite different from the f-factor problem: at first we are interested several (or all) solutions of the ReDeSe problem instead of finding one solution, and often enough we want to find "typical" solutions. At second: this sampling problem seems to be hopeless in general. In our studies we restrict ourself for carefully chosen small instances.

The **bipartite restricted degree sequence** problem $\mathbf{D}^{\mathcal{F}}$ consists of a bipartite degree sequence $\mathbf{D}$ on (U, W), and a set $\mathcal{F} \subset [U, W]$ of **forbidden** edges. The problem is to decide whether there is a simple bipartite graph G on V with the given degree sequence and with $E(G) \cap \mathcal{F} = \emptyset$.

Clearly, a bipartite restricted degree sequence problem $\mathbf{D}^{\mathcal{F}}$ on (U, W) is the restricted degree sequence problem $\mathbf{d}^{\mathcal{F}'}$ on $U \cup W$, where $\mathcal{F}' = \mathcal{F} \cup [U]^2 \cup [W]^2$.

Furthermore, we already studied one instance of the bipartite restricted degree sequence problem, namely the bipartite representation of directed degree sequences: here $\mathcal{F}$ is one 1-factor, which corresponds to the forbidden loops.

It is important to add that while the fundamental result of Jerrum, Sinclair and Vigoda on sampling perfect matchings in graphs ([18]) provides a uniform sampling approach for the possible realizations, their method is not useful in practice. That is the reason that so much effort has been made on this topic. We return to this issue at the end of Section "Counting".

In the remaining of this subsection we study the general ReDeSe problem. The next subsection will be devoted to a particular bipartite restricted degree sequence problem which will play a central role later in the paper.

Definition 2. Let $\mathbf{d}^{\mathcal{F}}$ be a restricted degree sequence problem and let G be a realization of it. The sequence of vertices $\mathcal{C} = (x_1, x_2, \ldots, x_{2i})$ is a **chord-circuit** if:

(D1) all pairs $x_1x_2, x_2x_3, \ldots, x_{2i-1}x_{2i}, x_{2i}x_1$ are chords;

(D2) each of these chords is different.

A chord-circuit is **elementary** if

(D3) no vertex occurs more than twice;

(D4) when two copies of the same vertex exist, then their distance along the circuit is odd.

Definition 3. The chord-circuit $\mathcal{C}$ is said to **alternate** in G, if the chords along $\mathcal{C}$ are in turn edges and non-edges in G. (For example $x_{2j-1}x_{2j}$ are edges for $1 \leq j \leq i$, while the other chords are non-edges in G.)

Deleting the actual edges along $\mathcal{C}$ from G and adding the other chords as edges constructs a new graph G' which is again a realization of $\mathbf{d}^{\mathcal{F}}$. This is a $\mathcal{C}$-**swap** and this operation is known in general as a **circular C_{2i}-swap**.

Finally, two *different* vertices x, y of the alternating chord-circuit $\mathcal{C}$ form a **PV-pair** if the distance of the vertices along the circuit (the number of chords between them) is odd and greater than 1. If all PV-pairs are non-chords (so they belong to $\mathcal{F}$), then this circular $\mathcal{C}$-swap is called a $\mathcal{F}$-**compatible swap** or $\mathcal{F}$-**swap** for short.

The $\mathcal{F}$-swap is one of the central notions of this paper. When $i = 2$ then the circular C_4-swap coincides with the classical Havel type swap. When $i = 3$ then we get back the notion of the **triangular C_6-swap**, which occurs in connection with directed degree sequences (see [19]).

We define the **weight** of the $\mathcal{F}$-compatible circular C_{2i}-swap as $w(C_{2i}) = i-1$. This definition sets the weight of the classical Havel type swaps to 1 and the weight of a C_6-swap to 2, which agree with the definitions used in paper [19]. Furthermore it is well known (see for example again [19]) that $(i-1)$ Havel type swaps are needed to alternate the edges along C_{2i} in case of simple graphs with no forbidden edges. As we will see next the same applies for any elementary circular C_{2i}-swap:

Lemma 4. *Let G be a realization of $\mathbf{d}^{\mathcal{F}}$ and let the elementary chord-circuit $\mathcal{C}$ of length $2i$ be alternating. Then the circular $\mathcal{C}$-swap operation can be carried out by a sequence of $\mathcal{F}$-swaps of total weight $i-1$.*

In other words there exists a sequence $G = G_0, G_1, \ldots, G_\ell$ of realizations such that for each $j = 0, \ldots, \ell-1$ there exists an $\mathcal{F}$-compatible swap operation from G_j to G_{j+1}. The difference between G and G_ℓ is exactly the alternating circuit $\mathcal{C}$. Finally, the total weights of those $\mathcal{F}$-swap operations is $i-1$. We will say that this swap sequence does **process** the prescribed circular swap operation.

Proof. We apply mathematical induction for the length of the chord-circuit: when $i = 2$ then the statement is trivial. Assume now that this is true for all circuits of length at most $2i-2$. Then take an alternating elementary chord-circuit $\mathcal{C}$ of length $2i$ in a realization of $\mathbf{d}^{\mathcal{F}}$.

If each PV-pair in $\mathcal{C}$ is a non-chord, then the circular C_{2i}-swap itself is a $\mathcal{F}$-swap of weight $i-1$. So we may assume that there is a PV-pair uv in $\mathcal{C}$ which is a chord. This chord together with the two "half-circuits" of $\mathcal{C}$ form chord-circuits $\mathcal{C}_1$ and $\mathcal{C}_2$ using the chords of the original circuit $\mathcal{C}$ and twice the chord uv. One of them, say $\mathcal{C}_1$, is alternating. The length of $\mathcal{C}_1$ is $2j < 2i$ therefore there exists a $\mathcal{F}$-compatible swap sequence of total weight $j-1$ to process it. After the procedure the *status* of uv (the property of the chord whether it is an edge or a non-edge) will alter into the other status. With this new status of the chord the circuit $\mathcal{C}_2$ becomes an alternating one with length $2i+2-2j$, so it can be processed with $\frac{2i+2-2j}{2} - 1$ total weight—and after this procedure the chord uv is switched back to its original status. We found a swap sequence of total weight $i-1$ which finishes the proof.

The space of all realizations of $\mathbf{d}^{\mathcal{F}}$: Consider now the set of all possible realizations of a restricted graphical degree sequence $\mathbf{d}^{\mathcal{F}}$. Let G and H be two different realizations. The natural question, similar to the case of classical degree sequence problems, is whether G can be transformed into H using $\mathcal{F}$-swaps? The answer is affirmative:

Theorem 5. *The space* $\mathbb{G} = (\mathbb{V}, \mathbb{E})$ *of all realizations of the restricted degree sequences problem* $\mathbf{d}^{\mathcal{F}}$ *is connected.*

Proof. What we have to prove is the following: let G and H be two realizations of $\mathbf{d}^{\mathcal{F}}$. Then we have to find a series of realizations $G = G_0, \ldots, G_{i-1}, G_i = H$, such that for each $j = 0, \ldots, i-1$ there exists an $\mathcal{F}$-swap from G_j to G_{j+1}.

Consider the symmetric difference of the edge sets of the two realizations: $\Delta = E(G) \triangle E(H)$. This set is two-colored by the original hosts of the edges: there are G-edges and H-edges. It is clear that for each vertex v in the graph $\mathcal{G} = (V, \Delta)$ the numbers of G-edges and H-edges incident to v are the same: $d_G(v) = d_H(v)$. It is well known that this can be decomposed into alternating circuits $\mathcal{C}_1, \ldots, \mathcal{C}_\ell$.

We will use the notions of **circuit** and **cycle** in a simple graph G as usual: therefore a circuit is a chord-circuit where all chords are edges. A cycle is a circuit without repeated vertices. A circuit is **alternating** in Δ if the edges come in turns from $E(G)$ and $E(H)$. When this is the case then the corresponding chord-circuit in realization G (as well as in H) is also alternating.

We can find a decomposition, such that no circuit contains a vertex v twice and their distance δ (the number of edges between the copies is even. Indeed, if δ is even, then δ is at least four, consequently the vertex v splits the original circuit into two smaller, but still alternating circuits. Furthermore, if a circuit contains a vertex v at least three times, then there are at least two of them with even distance.

It is clear that any alternating circuit decomposition can be transformed into a decomposition where each (chord)-circuit is elementary with successive transformations. It is also clear that if, by chance, we start with a circuit decomposition of maximal number of circuits, then all circuits in this decomposition are automatically elementary. (Of course, finding such a decomposition may be very hard.)

The application of Lemma 4 proves that each circuit $\mathcal{C}$ can be processed with $|\mathcal{C}|/2-1$ total weight. This finishes the proof.

It seems to be interesting that using a result from paper [19] one can determine the minimum weight of an $\mathcal{F}$-compatible swap sequence which transforms G into H, however we do not discuss this question here.

Bipartite 1-Factor + 1 Star Restricted Degree Sequences

In the previous subsection we studied the restricted degree sequence problem in its full generality. However, our real interest lays in a quite simple case: $\mathbf{d}^{\mathfrak{F}}$ is called a **1-Factor + 1 Star Restricted Degree Sequence** problem (or **1F1S** problem for short), if

(Ψ) the set $\mathfrak{F}$ of forbidden edges is a bipartite graph where the edges are the union of an 1-factor and a star with center s.

Similarly, if $\mathbf{D}$ is a bipartite degree sequence, and (Ψ) holds for $\mathfrak{F}$, then $\mathbf{D}^{\mathfrak{F}}$ is called a **Bipartite 1F1S** problem.

Everything discussed in this subsection applies to all 1F1S degree sequence problems in simple graphs. However, we are particularly interested in the bipartite case, therefore we will discuss these observations for the bipartite case only. We fix the underlying vertex set $V = (U, W)$. Then $\mathbf{D}^{\mathfrak{F}}$ is a bipartite 1F1S problem where the center s of the forbidden star belongs to U.

Lemma 6.

(i) *The space of all realizations of* $\mathbf{D}^{\mathfrak{F}}$ *is closed under* $\mathfrak{F}$*-compatible swap operations.*

(ii) *The* $\mathfrak{F}$*-compatible swap operations are circular* C_4- and C_6*-swaps.*

Proof. (i) As we saw already that any bipartite 1F1S can be understood as an 1F1S on simple graphs, therefore considering $\mathcal{F} = \mathfrak{F} \cup [U]^2 \cup [W]^2$ and applying Theorem 5 for the problem $\mathbf{d}^{\mathcal{F}}$ proves (i).

(ii) Let us consider any alternating elementary circuit C in the symmetric difference $\triangle$ of two different realizations. There is a vertex $u \in C \cap U$ which is $\neq s$. There is at most one forbidden chord in $\mathfrak{F}$ which is adjacent to u. If C has more than 6 vertices, then C has at least 4 vertices in W therefore there exists a vertex $w \in C \cap W$, such that uw is a chord and uw is not in C. Therefore the corresponding C-swap is not compatible with $\mathfrak{F}$.

As we already mentioned, Tutte's f-factor theorem can always be utilized to find actual graphical realizations of the bipartite 1F1S problem. However, in this special case we can prove a Havel type result (similar to Lemma 1) and can construct a greedy algorithm to produce such realizations.

Consider the bipartite 1F1S degree sequence problem $\mathbf{D}^{\mathfrak{F}}$. If the forbidden star is not empty, then let $u := s$. Otherwise let $u \in U$ be any given vertex and denote $N(u) \subset W$ the set of those vertices which form chords together with u. (It is clear that if $u \neq s$ then $|W|-1 \leq |N(u)|$.)

Observation 7. *For any $y \in N(u)$ there is at most one vertex, denoted by $y^{\mathfrak{F}}$, such that $yy^{\mathfrak{F}}$ is a non-chord, so it belongs to $\mathfrak{F}$. Furthermore if $y, z \in N(u)$ and $y^{\mathfrak{F}} = z^{\mathfrak{F}}$ then $y = z$.*

Now a linear order $\prec_u$ on $N(u)$ is called **good** if it satisfies the following properties: for $y, z \in N(u)$ and $y \prec_u z$ we have

$d(y) \geq d(z)$ and in case of $d(y) = d(z)$ we also have $d(y^{\mathfrak{F}}) \geq d(z^{\mathfrak{F}})$.

It is obvious that there always exists a good order on $N(u)$. Furthermore whenever $d(y) = d(z)$ and $d(y^{\mathfrak{F}}) = d(z^{\mathfrak{F}})$, then there are more than one good order.

Lemma 8. *Let G be a graphical realization of the 1F1S sequence $\mathbf{D}^{\mathfrak{F}}$, let $u := s$ if the forbidden star is not empty and take any $u \in U$ otherwise. Let $y, z \in N(u)$ where $y \prec_u z$ with $uz \in E$ while $uy \notin E$. Then there exists an alternating chord-cycle C of length at most 6 in G with $y, u, z \in C$. Processing C with $\mathfrak{F}$-compatible swap operations, we have $\Gamma_{G'}(u) = \Gamma_G(u) \setminus \{z\} \cup \{y\}$ in the acquired new realization.*

Proof. We have $uz \in E$ but $uy \notin E$. At first assume that there exists a vertex $\mu \in U \setminus \{u\}$, such that $\mu y \in E$, and $\mu z \notin E$ but $\mu \neq z^{\mathfrak{F}}$. When such vertex exists then $\mathcal{C} = (u, z, \mu, y)$ is a suitable alternating chord-cycle.

When $d(y) > d(z)$ then there are two vertices μ and $\mu' \in U$ such that $y\mu \in E$ and $z\mu \notin E$, and $y\mu' \in E$ and $z\mu' \notin E$. Now either $z\mu$ or $z\mu'$ is a chord.

However, if $d(y) = d(z)$ then it can happen that $z^{\mathfrak{F}}\, y \in E$ and

$$\text{for all } x \in U \setminus \{u, y^{\mathfrak{F}}, z^{\mathfrak{F}}\} \quad \text{we have} \quad xy \in E \Leftrightarrow xz \in E. \tag{2}$$

It is important to observe that in this case $y^{\mathfrak{F}}\, z \notin E$, otherwise some x would not satisfy Eq (2) (in order to keep $d(y) = d(z)$).

So the only case when we do not find automatically an appropriate circular C_4-swap with u, y and z is when $d(y) = d(z)$, $yz^{\mathfrak{F}}$ is an edge and $zy^{\mathfrak{F}}$ is a chord but not an edge. In this case, we can find a $\mu \in W \setminus \{y, z\}$ such that $y^{\mathfrak{F}}\mu \in E$ but $z^{\mathfrak{F}}\mu \notin E$ since $d(y^{\mathfrak{F}}) \geq d(z^{\mathfrak{F}})$. Observe that $z^{\mathfrak{F}}\mu$ is a chord because $\mu^{\mathfrak{F}} \neq z^{\mathfrak{F}}$.

Now $\mathcal{C} = (y, u, z, y^{\mathfrak{F}}, \mu, z^{\mathfrak{F}}, y)$ is the required alternating chord circle. When $u\mu$ is a chord, then the circular C_6-swap is not $\mathfrak{F}$-compatible, but we can process the cycle properly (as it was shown in the proof of Lemma 4). When it is a non-chord, then the circular C_6 is an $\mathfrak{F}$-compatible operation.

Lemma 8 provides the following easy Havel type greedy algorithm to decide whether our bipartite 1F1S restricted degree sequence is graphical. (The algorithm is essentially the same as the original Havel procedure.)

A greedy algorithm to decide whether a bipartite 1F1S degree sequence is graphical: the degree sequence is **D** while the set of forbidden edges is $\mathfrak{F}$.

(H_1) Let $u := s$ if the forbidden star is not empty, otherwise take an arbitrary $u \in U$. Consider a good order $\prec_u$ on $N(u)$. Connect u the first $d(u)$ vertices from (with respect to $\prec_u$) of $N(u)$. If $d(u) > |N(u)|$ then the algorithm FAILS, our sequence $\mathbf{D}^{\mathfrak{F}}$ is not graphical. Delete u from U, and update the degree sequence and the set W accordingly. Finally delete the edges adjacent to u from $\mathfrak{F}$.

(H_2) Repeat the previous step while U is not empty.

Theorem 9 (Generalized Havel theorem for the bipartite 1F1S ReDeSe problem). *The 1F1S restricted degree sequence* $\mathbf{D}^{\mathfrak{F}}$ *is graphical if and only if the previous greedy algorithm provides a realization*.

Proof. The proof is exactly the same as in the original case, described by Havel; if the sequence is graphical, then consider a realization. Fix vertex $u \in U$ as described in Lemma 8. Recursive applications of the lemma provide a realization where u is connected to the first $d(u)$ vertices in $N(u)$ with respect to $\prec_u$. The repeated application of the previous reasoning finishes the proof.

Sampling Degree Sequence Realizations with Sinclair's Multicommodity Flow Method

There are several available methods to sample uniformly the space of all realizations of a given degree sequence. One of these approaches is a Markov Chain Monte Carlo method, proposed by Kannan, Tetali and Vempala (1999, [3]). They consider a local transformation (the swap operation) on the realizations, which in turn defines an irreducible, reversible and aperiodic finite Markov chain on these realizations; at any given realization they choose two independent edges from some probability distribution and perform the corresponding swap operation if it is feasible. They conjecture that the resulted MCMC is *rapidly mixing*. They were studying the particular case when the degree sequence is regular bipartite using Sinclair's multicommodity flow method.

Their conjecture was proved for regular graphs by Cooper, Dyer and Greenhill (2007, [4]). (Their result does not apply to bipartite graphs, since their version does not allow forbidden edges.) An analogous theorem was proved by Greenhill on regular directed graphs ([5]). Here she proved at first that for regular directed degree sequences circular C_4-swaps alone make the space of the realizations connected, then she gave a strong upper bound on the mixing time. (However, as we saw it earlier, the space of directed realizations are not always connected when using only C_4-swaps.) Finally, in 2013 Miklós, Erdős and Soukup proved ([6]) that the corresponding Markov process is rapidly mixing on each bipartite **half-regular** degree sequence, superseding the original study of Kannan, Tetali and Vempala ([3]).

In this paper we study the realizations of half-regular bipartite 1F1S restricted degree sequences $\mathbf{D}^{\mathfrak{F}}$. The vertex set is (U, W) where the center of the forbidden star s is $\in U$ and where all vertex in U (except possible s) have the same degree. The degrees in W are not constrained.

The state space of our **Markov chain** is the graph $\mathbb{G} = (V(\mathbb{G}), E(\mathbb{G}))$ where $V(\mathbb{G})$ consists of all possible realizations of our problem, while the edges represent the possible swap operations: two realizations (which will be indicated by upper case letters like X or Y) are connected if

there is a valid $\mathfrak{F}$-swap operation which transforms one realization into the other one (and the inverse swap transforms the second one into the first one as well). Recall that there are two kinds of $\mathfrak{F}$-compatible swap operations: the circular C_4-swaps and certain C_6-swaps (in the latter case opposite vertex pair in the C_6 must be non-chord), Furthermore, these two kinds of operations make the state space connected (see Theorem 5).

The *transition (probability) matrix P* of the Markov chain is defined as follows: let the current realization be G. Then

(a) with probability 1/2 we stay in the current state (that is, our Markov chain is **lazy**);

(b) with probability 1/4 we choose uniformly two-two vertices $u_1, u_2; v_1, v_2$ from classes U and W respectively and perform the swap if it is possible;

(c) finally with probability 1/4 choose three—three vertices from U and W and check whether they form three pairs of forbidden chords. If this is the case, then we perform a circular C_6-swap if it is possible.

The swaps moving from G to its image G' is unique, therefore the probability of this transformation (the *jumping probability* from G to $G' \neq G$) is:

$$\mathrm{Prob}(G \rightarrow_b G') := P(G'|G) = \frac{1}{4} \cdot \frac{1}{\binom{|U|}{2}\binom{|W|}{2}}, \tag{3}$$

and

$$\mathrm{Prob}(G \rightarrow_c G') := P(G'|G) = \frac{1}{4} \cdot \frac{1}{\binom{|U|}{3}\binom{|W|}{3}}. \tag{4}$$

(These probabilities reflect the fact, that G' should be derived from G by a regular swap or by a C_6-swap.) The probability of transforming G to G' (or vice versa) is time-independent and *symmetric*. Therefore P is a symmetric matrix, where the entries in the main diagonal are non-zero, but (probably) distinct values. Our Markov chain is *irreducible* (the state space is connected), and it is clearly aperiodic, since it is lazy. Therefore, as it is well known, the Markov process is reversible with the uniform distribution as the globally stable stationary distribution.

Our main result is the following:

Theorem 10. *The Markov process defined above is rapidly mixing on each bipartite half-regular 1F1S restricted degree sequence.*

Remark 11. When we apply this setup for directed graphs then the out-degrees are regular (except, perhaps, the out-degree of the vertex s), while we have no constrains on the in-degrees. However, it is important to see, that while this result provides a rapidly mixing sampling procedure on regular directed graphs as well, the applied Markov chain is not the same as the one in Greenhill's model. Hence, this result does not supersede Greenhill's result.

The proof of Theorem 10 follows closely the proof developed in paper [6]. (More precisely we need to slightly generalize it. The required minor technical issue will be discussed in the Section "Some further technical details of the Sinclair's method".) Consider two realizations $X \in \mathbb{G}$ and $Y \in \mathbb{G}$ of the problem $\mathbf{D}^{\mathfrak{F}}$, and take the symmetric difference $\Delta = E(X)\Delta E(Y)$. As we saw already in the proof of Theorem 5 for each vertex v in the bipartite graph $(U, W; \Delta)$ the number of adjacent X-edges $(= E(X) \setminus E(Y))$ and the number of the adjacent Y-edges are equal. Therefore Δ can be decomposed into alternating circuits and later into alternating cycles. The way the decomposition is executed is described in details in Section 5 of the paper [6]. Here we just summarize the high points:

At first we decompose the symmetric difference Δ into alternating circuits on all possible ways. In each cases we get an ordered sequence $W_1, W_2, \ldots, W_\kappa$ of circuits. (Usually there are a huge number of different circuit decompositions.) Each circuit is endorsed with a fixed cyclic order.

Now we fix one circuit decomposition. Each circuit W_i from the ordered decomposition determines one unique alternating cycles decomposition: $W_i = C^i_1, C^i_2, \ldots, C^i_{k_i}$. (This unique decomposition is one of the most delicate points of the entire proof in [6]. The main problem is that a circuit can be "long"—linear in the number of vertices—therefore, it can happen that it is decomposed into a linear number of cycles. Keeping track of all possible changes along the circuit is necessary, and without clever data handling it may require an unacceptable big data set. Section 5.2 in paper [6] found a way around this problem.)

The ordered circuit decomposition of Δ together with the ordered cycle decompositions of all circuits provide a well defined ordered cycle decomposition $C_1, \ldots, C_\ell$ of Δ. This decomposition does not depend on any $\mathfrak{F}$-compatible swap operations (actually no swap operation was performed yet), only on the symmetric difference of realization X and Y. So this part of the original proof can be used freely in our current reasoning without any modification.

This ordered cycle decomposition singles out $\ell-1$ different realizations $H_1, \ldots, H_{\ell-1}$ of $\mathbf{D}^{\mathfrak{F}}$ with the following property: for each $j = 0, \ldots, \ell-1$ we have $E(H_j)\Delta E(H_{j+1}) = C_{j+1}$ if we apply the notations $H_0 = X$ and $H_\ell = Y$. This mean that

$$E(H_i) = E(X) \triangle \left(\bigcup_{i' \le i} E(C_{i'}) \right).$$

What remains is to design a unique canonical path from X to Y determined by the circuit decompositions which use the realizations H_j as *milestones* along the path. With other words, for each pair H_j, H_{j+1} we have to design the actual swap sequences which turn one milestone into the next one.

So, the canonical path under construction is a sequence $X = G_0, \ldots, G_i, \ldots, G_m = Y$ of realizations, where each G_i can be derived from G_{i-1} with one feasible circular C_4- or C_6-swap operation, and there exists an increasing subscript subsequence $0 = n_0 < n_1 < n_2 < \cdots < n_\ell = m$ such that we have $G_{n_i} = H_i$.

In paper [6] the following result was proved:

Theorem 12 (Section 4 in [6]). *If the designed canonical path system satisfies the three (rather complicated) conditions below, then the MCMC process is rapidly mixing. The conditions are:*

(Θ) *For each $i < \ell$ the constructed path $H_i = G'_0, G'_1, \ldots, G'_{m'} = H_{i+1}$ satisfies that $m' \le c \cdot |C_{i+1}|$ for a suitable constant c.*

(Ω) *$\forall j$ there exists a $K_j \in V(\mathbb{G})$ s.t. $\mathfrak{d}\left(M_X + M_Y - M_{G'_j}, M_{K_j}\right) \le \Omega_2$, where M_G is the* **bipartite adjacency matrix** *of G, and $\mathfrak{d}$ stands for the Hamming distance of two matrices, finally Ω_2 is a small constant.*

(Ξ) *For each vertex G'_j in the path under construction the following three objects together uniquely determine the realizations X, Y and the path itself.*

- *The value of the auxiliary matrix $M_X + M_Y - M_{G'_j}$;*
- *the symmetric difference $\Delta = E(X) \triangle E(Y)$;*
- *finally a polynomial size parameter set $\mathbb{B}$.*

The meaning of condition (Ξ) is that these structures can be used to control certain features of the canonical path system: namely their numbers gives a bound on the number of canonical paths between any realization pairs X, Y which go through any given realization G'_j. Then condition (Ω) ensures, that the overall number of the used auxiliary matrices is small.

So while we determine our canonical paths among any pair X, Y we have take care for these three conditions. We will describe the construction itself in the following two Sections.

The construction of swap sequences between consecutive "milestones"

Now we are going to implement our plan described above. At first we introduce some shorthand. Instead of H_{i-1} and H_i we will use the names G and G'. These two graphs have almost the same edge set. More precisely

$$(E(G) \setminus (C_i \cap E(X))) \cup (C_i \cap E(Y)) = E(G')$$
$$(E(G') \setminus (C_i \cap E(Y))) \cup (C_i \cap E(X)) = E(G).$$

Of course $E(G)\Delta E(G') = C_i$ also holds. We refer to the elements of $C_i \cap E(X)$ as X-edges, while the others are Y-edges. We denote the cycle itself by $\mathcal{C}$, it has 2ℓ edges and its vertices are u_1, w_1, u_2, w_2, . . ., u_ℓ, w_ℓ. Since $\mathcal{C}$ has at least four vertices, therefore we may assume that $u_1 \neq s$ (thus u_1 is not the center of the forbidden star). Finally, w.l.o.g. we may assume that the chord u_1w_1 is a Y-edge (and, of course, $w_\ell\, u_1$ is an X-edge).

We are going to construct the realizations G'_j one by one. We build our canonical path from G toward G'. At any particular step the last constructed realization is denoted by Z. (At the beginning of the process we have $Z = G$.) We are looking for the next realization, denoted by Z'.

Before we continue the discussion of the canonical path system, we introduce our control mechanism, mentioned in condition (Ω). This *auxiliary* structure originally was introduced by Kannan, Tetali and Vempala in [3]:

For any particular realization G from $V(\mathbb{G})$ the matrix M_G denotes the *adjacency matrix* of the bipartite realization G where the columns and rows are indexed by the vertices of U and W respectively (Therefore the column sums are the same in each realization, except perhaps at column s.) Our indexing method is a bit unusual: the columns are numbered from left to right while the rows are numbered from bottom to the top. (Like in the Cartesian coordinate system.) This matrix is not necessarily symmetric, and elements $M_{i,i}$ can be different from 0.

For example, if we consider the submatrix in M_G spanned by u_1, . . ., u_ℓ and w_1, . . ., w_ℓ then we have $M_G(i, i) = 0$ for $i = 1, \ldots, \ell$, while $M_G(i, i-1) = 1$ (for $i = 2, \ldots, \ell$) and $M_G(1, \ell) = 1$. (So the first value gives the column, the second one gives the row.) The non-chords between vertices in the same vertex class are not considered at all, while non-chords which are forbidden are denoted by ✠. As it is clear from the previous sentence, we will identify each chord or non-chord with the corresponding position in the matrix.

Our auxiliary structure is the matrix

$$\widehat{M}(X + Y - Z) = M_X + M_Y - M_Z.$$

By definition, each entry of a bipartite adjacency matrix is 0 or 1 (or ✠). Therefore only −1, 0, 1, 2 can be the "meaningful" entries of $\widehat{M}$. An entry is −1 if the edge is missing from both X and Y but it exists in Z. It is 2 if the edge is missing from Z but exists in both X and Y. It is 1 if the edge exists in all three graphs (X, Y, Z) or it is there only in one of X and Y but not in Z. Finally it is 0 if the edge is missing from all three graphs, or the edge exists in exactly one of X and Y

and in Z. (Therefore if an edge exists in exactly one of X and Y then the corresponding chord in $\widehat{M}$ is always 0 or 1.) One more important, but easy fact is the following:

Observation 13. *The row and column sums of $\widehat{M}(X + Y - Z)$ are the same as row and column sums in M_X (or M_Y or M_Z).*

Next we will determine the swap sequence between G and G' through an iterative algorithm. At the first iteration we check, step by step, the positions (u_1, w_2), (u_1, w_3), . . ., (u_1, w_ℓ) and take the smallest j for which (u_1, w_i) is an actual edge in G. Since (u_1, w_ℓ) is an edge, therefore such i always exists. So we may face to the following configuration:

We call the chord u_1w_i the **start-chord** of the current sub-process and u_1w_1 is the **end-chord**. We will *sweep* the alternating chords along the cycle from the start-edge w_iu_i (non-edge), u_iw_{i-1} (an edge) toward the end-edge w_1u_1 (non-edge)—switching their status in twos and fours. We check positions u_1w_{i-1}, u_1w_{i-2} (all are non-edges) and choose the first chord among them, we call this the **current-chord**. (Since $u_1 \neq s$ therefore we never have to check more than two edges to find the first chord, and we need to check two edges only once, since there is at most one non-chord adjacent to u_1.)

Case 1: As we just explained, the typical situation is that the current-chord is the "next" one, so when we start this is typically u_1w_{i-1}. Assume that this is a chord. Then we can proceed with the swap operation $w_{i-1}u_i$, $w_iu_1 \Rightarrow u_1w_{i-1}$, u_iw_i. We just produced the first "new" realization in our sequence, this is G'_1. For the next swap operation this will be our new current realization. This operation will be called a **single-step**.

In a realization Z we call a chord **bad**, if its current status (being edge or non-edge) is different from its status in G (or, what is the same, in G', since they differ only on the chords along the cycle $\mathcal{C}$). After the previous swap, we have two bad chords in G'_1, namely u_1w_{i-1} and w_iu_1.

Consider now the auxiliary matrix $\widehat{M}(X + Y - Z)$ (here $Z = G'_1$). As we saw earlier, for each position outside the chords in $\mathcal{C}$ the status of that particular position in Z is the same as in X or Y or in both. Accordingly, the corresponding matrix value is 0 or 1. We call a position **bad** in $\widehat{M}$ if this value is −1 or 2. (A bad position in $\widehat{M}$ always corresponds to a bad chord.) Since in Case 1 we switch the start-chord into a non-edge, it may become 2 in $\widehat{M}$. (In case if in both X and Y it is an edge. Otherwise it is 0 or 1, so in that case it is not a bad position.) The current-chord turned into an edge. If it is a non-edge in both X and Y then the value becomes −1, otherwise it does not become a bad position. After this single-step, we have at most two bad positions in the matrix, at most one position with 2-value and at most one with −1-value.

Case 2: If the previous case does not apply then the pair u_1w_{i-1} is a non-chord, therefore we cannot produce the previous swap. Then the non-edge u_1w_{i-2} is the current-chord. For sake of simplicity we assume that i–2 = 2, this case is represented in Fig 1. Consider now the alternating C_6 cycle: u_1, w_2, u_3, w_3, u_4, w_4. It has a total of three vertex pairs which may be chords. We know already that u_1w_3 is a non-chord. If none of the three positions is a chord, then this is an $\mathcal{F}$-compatible circular C_6-swap—and accordingly to the definitions we can swap it in one step. Again, we found the valid swap w_2u_3, w_3u_4, $w_4u_1 \Rightarrow u_1w_2$, u_3w_3, u_4w_4. After that we again have 2 bad chords, namely u_1w_2 and w_4u_1, and together we have at most two bad positions in the new $\widehat{M}(X + Y - Z)$ with at most one 2-value and at most one −1-value.

Finally, if one position, say w_2u_4, is a chord then we can process this C_6 with two swap operations. If this chord is, say, an actual edge, then we swap w_2u_4, $w_4u_1 \Rightarrow u_1w_2$, u_4w_4. After this we can take care of the w_2, u_3, w_3, u_4 cycle. Along this sequence we never create more, than 3 bad chords: the first swap makes chords w_2u_4, w_4u_1 and u_1w_2 bad ones, and the second one "cures" w_2u_4 but does not touch u_1w_2 and w_4u_1. So along this swap sequence we have 3 bad chords, at the end we have only 2. On the other hand, if the chord w_2u_4 is not an edge, then we can swap w_2u_3, $w_3u_4 \Rightarrow u_3w_3$, u_4w_2, creating one bad edge, then taking care the four-cycle u_1,

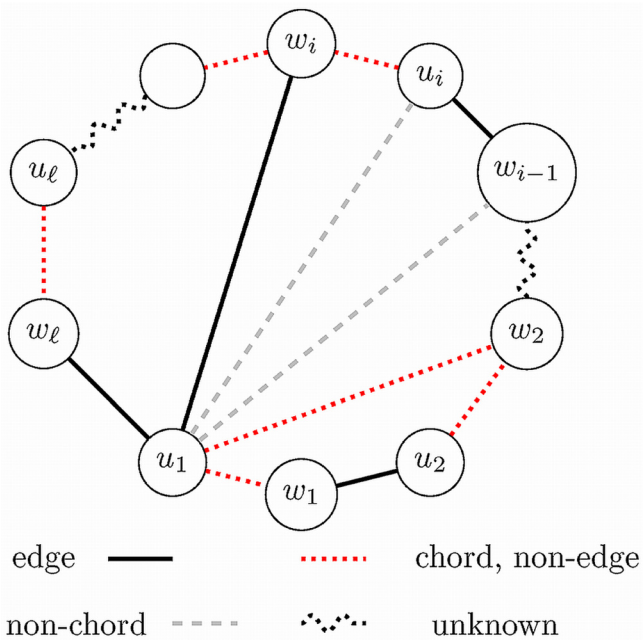

Fig 1. Sweeping a cycle.

doi:10.1371/journal.pone.0131300.g001

w_2, u_4, w_4 we "cure" w_2u_4 but we switch u_1w_2 and w_4u_1 into bad chords. We finished our **double-step** along the cycle.

In a double-step in any moment we have at most three bad chords. When the first swap uses three chords along the cycle then we may have at most one bad chord (with $\widehat{M}$-value 0 or −1) and then the next swap switches back the chord into its original status, and makes two new bad chords (with at most one 2-value and one −1-value). When the first swap uses only one chord from the cycle, then it makes three bad chords (changing two chords into non-edge and one into edge), therefore it may make at most two 2-values and one −1-value. After the second swap there will be only two bad chords, with at most one 2-value, and at most one −1-value.

When only the third position corresponds to a chord in our C_6 then after the first swap we may have two −1-values and one 2-value. However, again after the next swap we will have at most one of both types.

Remark 14. When two realizations are one swap apart (so they are adjacent in $\mathbb{G}$) then we say that their auxiliary matrices are at swap-distance one. Since one swap changes four positions of the matrix, therefore the Hamming distance of these matrices is 4.

Finishing our single- or double-step, the previous current-chord becomes the new start-chord. Then we repeat our procedure. There is only one important point to be mentioned: along the step, the start-chord switches back into its original status, therefore it stops being a bad chord. Thus, even if we face a double-step the number of bad chords never will be bigger than three (together with the chord $w_i\,u_1$ which is still in the wrong status, so it is bad), and we have always at most two 2-values and at most one −1-value in $\widehat{M}(X+Y-Z)$.

When w_1u_2 becomes the current-chord the last step will switch the last start-chord back into its correct status, hence the last current-chord cannot be in bad status. Finally, when the sweep from $w_i\,u_1$ to w_1u_1 is finished we only have one bad chord (with a possible 2-value in $\widehat{M}$). This concludes the first iteration of our algorithm.

For the next iteration we seeks a new start-chord between $w_i u_1$ and $w_\ell u_1$. Chord $w_i u_1$ becomes the new end-chord. We will repeat our sweeping process for this setup, until all chords are processed. If there was a double-step in the first sweep, then it will not occur again, thus there are never more than three bad chords; at most two 2-values and at most one −1-value.

However, if the double-step occurs sometime later, for example in the second sweep, then we face one of the following two cases: the circular C_6-swap under consideration is either $\mathfrak{F}$-compatible or not. If it is $\mathfrak{F}$-compatible, we perform the circular C_6-swap. This does not change the number of bad chords, except if this swap finishes a current sweep. If, however, the circular C_6-swap is not compatible, then there exists a chord in the chord-cycle which is suitable for a swap. If this chord is a non-edge, then the swap corresponding to it produces one bad chord, and at most one bad position in $\widehat{M}$. If this chord is an edge in the current realization, then after the first swap there are four bad chords, and there may be at most three 2-values and at most one −1 value. After the next swap (which finishes the double step) we annihilate one of the 2-values, and after that swap there are at most two 2-values and at most −1-value along the entire swap sequence. When we finish our second sweep, then chord $w_i u_1$ will be switched back into its original status, hence it will not be bad anymore.

We apply the same algorithm iteratively. After at most ℓ sweep sequences the entire cycle $\mathcal{C}$ will be processed. This finishes the construction of the required swap sequence (and the required realization sequence).

Meanwhile we also proved the following important observation:

Lemma 15. *Along our procedure each occurring auxiliary matrix $\widehat{M}(X + Y - Z)$ is at most swap-distance one from a matrix with at most three bad positions: with at most two 2-values and with at most one −1-value in the same column, which does not coincide with the center of the forbidden star.*

The Analysis of the Swap Sequences Between "Milestones"

What remains is to show that the defined swap sequences between H_i and H_{i+1} satisfy conditions (Θ), (Ω) and (Ξ) of Theorem 12. The first one is easy to see, since we can process a cycle of length 2ℓ in $\ell-1$ swaps. Therefore the derived constant c in (Θ) is actually 1.

Now we introduce the new **switch** operation on 0/1 matrices with forbidden positions: we fix the four corners of a submatrix (none of them is forbidden), and we add 1 to two corners in a diagonal, and add −1 to the corners on the other diagonal. This operation clearly does not change the column and row sums of the matrix. For example if we consider the matrix M_G of a realization of $\mathbf{d}^{\mathfrak{F}}$ and make a valid swap operation, then this is equivalent to a switch in this matrix. The next statement is trivial but very useful:

Lemma 16. *If two matrices have* **switch-distance** *1, then their Hamming distance is* 4. *Consequently if the switch-distance is c then the Hamming distance is bounded by* 4*c*.

We prove that property (Ω) holds for auxiliary matrices:

Theorem 17. *For any realizations X and Y and for any realization Z on a swap sequence from X to Y there exists a realization K such that*

$$\mathfrak{d}(\widehat{M}(X + Y - Z), M_K) \leq 16.$$

Due to Lemmas 15 and 16 it is enough to show that:

Lemma 18. *Any matrix $\widehat{M}(X + Y - Z)$ with constant column sums (this does not necessarily hold for the center of the forbidden star) and with at most three bad positions (where there are at most two 2-values and at most one −1-value) can be transformed into a valid M_K adjacency matrix with at most three switch operations.*

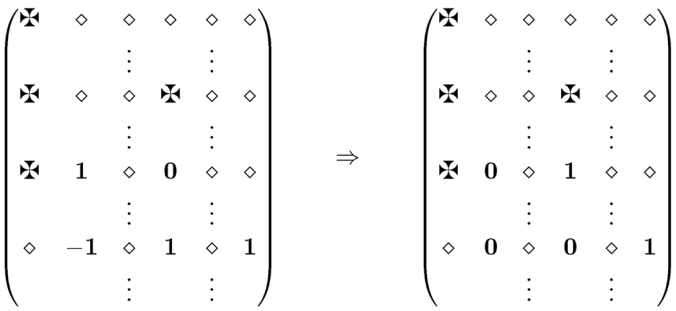

Fig 2. Case 1 ✠ = forbidden ◇ = 0/1/✠.

doi:10.1371/journal.pone.0131300.g002

Proof. Consider now a given $\widehat{M}$ which is not necessarily a valid adjacency matrix of a realization. We show in figures the submatrix in this matrix that describes the current alternating cycle $\mathcal{C}$. If it happens that $s \in \mathcal{C}$ then we choose a submatrix representation such that the center s of the forbidden star is in the first column. (We choose this submatrix as an illustration tool, but we still consider the entire matrix to work with.) We know that this matrix contains at most two 2-values and at most one 1-value. All three positions are adjacent to the center u_1 of the sweeping sequence (see Fig 1), hence they are in the same column.

For simplicity we denote the center of the sweep as well the column with u. The forbidden positions are denoted with ✠. Any column (except column 1) may contain at most one of them, and any row may contain at most two of them. Finally, in the figures the character ◇ stands for a character which we are not interested in. That is, it can be 0 or 1 or ✠.

We distinguish multiple cases, depending on the occurring of values 2 and −1.

Case 1. Column u has one bad position, which can be −1 or 2, or it has two 2-values. Consider at first the case when $\widehat{M}[uw] = -1$. By definition this means that chord uw is an edge in Z but non-edge in both X and Y. So vertex $w \in W$ has at least one adjacent edge, therefore the row-sum in its row is at least 1. Therefore there are at least two positions in row w with entries 1. They are in column u_1 and u_2. At least one of them, say u_1, differs from s. Since the column sums are constant, therefore there exists at least two rows w_1 such that $\widehat{M}[uw_1] = 1$ while $\widehat{M}[u_1w_1] = 0$ or ✠. However, there can be at most one forbidden position in u_1, so at least in one of the rows the entry is 0. Using these positions for the corresponding switch it eliminates the bad position without creating a new one. (See Fig 2.)

Before we continue, we prove an important observation:

Observation If w belongs to the alternating cycle $\mathcal{C}$ and $\widehat{M}[uw] = 2$ then row w contains at least two 0-values.

Indeed, there are α forbidden chords in row w. Since w is in an alternating cycle, therefore $d(w) \leq |U| - \alpha - 1$. Therefore the sum of row w in $\widehat{M}(X + Y - Z) \leq |U| - \alpha - 1$. But it contains a 2 and it does not contain -1 therefore there are at least two 0's in it.

When the single bad value in $\widehat{M}$ is 2 then, due to our previous Observation, in its row there are two 0's. And with them one can repeat the reasoning which we used about the unique −1-value.

Finally, when there are two 2-values which raises a very similar situation. Here we can do the same procedure independently on both rows. In this case, however, we need two switch operations.

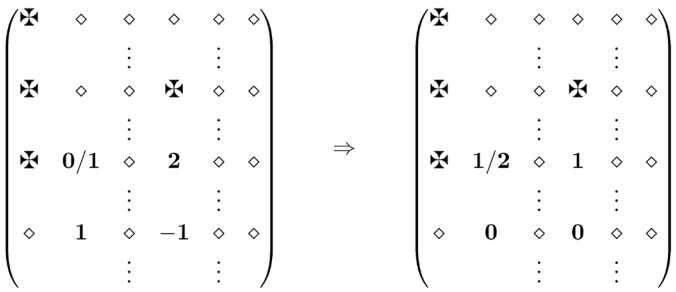

Fig 3. Case 2a ✠ = forbidden ◇ = 0/1/✠.

doi:10.1371/journal.pone.0131300.g003

Case 2. Here we assume that there is one 2-value and one −1-value in column u. For example $\widehat{M}[uw_1] = 2$ and $\widehat{M}[uw_2] = -1$. Again, in row w_2 there are at least two 1-values.

Case 2a Assume at first that we have $u_1 \in U$ s.t. $\widehat{M}[u_1w_2] = 1$ and $\widehat{M}[uw_1] \neq$ ✠. Then the corresponding switch will produce $\widehat{M}[u_1w_1] = 1/2$ while the other three positions are 0 or 1. (See Fig 3.) If now $\widehat{M}[u_1w_1] = 2$ then we are back to Case 1, and one more switch eliminates the last bad position as well. So we needed at most two switches.

Case 2b It can happen, that there are only two 1-values in row w_2 and both are facing with forbidden positions in row w_1. Then at least one 0 in row w_2 faces a chord in row w_1. (See Fig 4) The appropriate switch kills 2 bad chords and can make at most one −1-value. At this point we are either finished or back to Case 1.

Case 3. Finally suppose that there are three bad positions, two 2-values at positions uw_1 and uw_2 and one −1-value at position uw_3. Now both rows w_1 and w_2 contain at least two 0's. If any of them face a 1 in row w_3 then an appropriate switch annihilates one 2 and one −1 and does not create new bad position. We are back to Case 1. Altogether we need two switches.

If this is not the case then we consider the following: assume that $\widehat{M}[u_1w_1] = 0$. Since the column sums are the same, and we assumed that $\widehat{M}[u_1w_3] = 0$ therefore there exists a row w_4 s. t. $\widehat{M}[u_1w_4] = 1$ while $\widehat{M}[uw_4] = 0$. Then we can switch this 2-value without making a new bad position. After that we are back to Case 2. Altogether this requires at most three switches. The proof of Lemma 18 is finished.

If this is not the case then we consider the following: assume that $\widehat{M}[u_1w_1] = 0$. The column sums are the same, and we assumed that $\widehat{M}[u_1w_3] = 0$ or ✠. Therefore the difference between column sums in u and u_1 is 1 due to rows w_1 and w_3, and the difference increase at least 1 for row w_2, where against a 2-value in column u there is either 1 or 0 in column u_1. Therefore there exists at least two further rows, where there is a 1 in column u_1 against a 0 or ✠ in column u. Since column u can contain at most one ✠, one of the rows must contain a 0. Let it be denoted by w_4. Hence $\widehat{M}[u_1w_4] = 1$ while $\widehat{M}[uw_4] = 0$. Then we can switch this 2-value without making a new bad position. After that we are back to Case 2. Altogether this requires at most three switches. We finished the proof of Lemma 18.

There was no word yet about condition (Ξ) in Theorem 12. We discuss this in the next Section, because the magnitude of parameter $\mathbb{B}$ heavily depends on. To finish the Theorem 12, let us assume for now that we have the proper upper bound on $\mathbb{B}$. Then (Ω) in Theorem 12 and therefore the theorem itself is proved as well. Thus, our Markov chain is rapidly mixing as Theorem 10 stated.

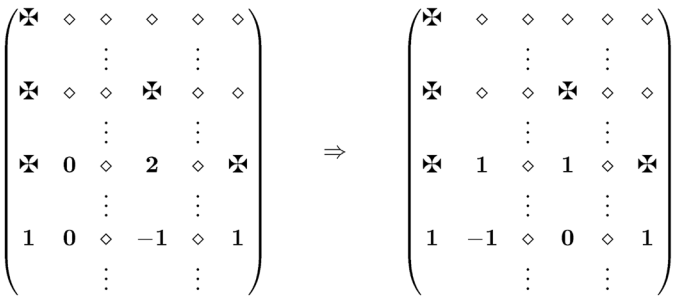

Fig 4. Case 2b ✠ = forbidden ◇ = 0/1/✠.

doi:10.1371/journal.pone.0131300.g004

Some further technical details of the Sinclair's method

In the last two sections we proved the rapidly mixing nature of our proposed MCMC method on the 1F1S restricted degree sequence problems through a special instance of Sinclair's method, developed in [6]. However, we need slightly generalize this method in order to finish the proof.

Let's recall that the method takes two realizations, X and Y, of the same degree sequence. It considers all possible ordered circuit decompositions of the symmetric difference of the edge sets, then it uniquely decomposes each such decomposition into an ordered sequence $\mathcal{C} = C_1, \ldots, C_m$ of oriented cycles. Based on this latter decomposition the method determines a well defined unique path between X and Y in the Markov chain $\mathbb{G}$.

To find this unique path the method first defines a sequence of "milestones". These are different realizations $X = H_0, H_1, \ldots, H_{m-1}, H_m = Y$ of the degree sequence where the edge set of any two consecutive realizations H_{i-1}, H_i differ exactly in the edges along the cycle C_i. (Until this point no swap operation happened.)

In the next phase, for any particular $i = 0, \ldots, m-1$ the method determines a sequence of valid swap operations transforming H_{i-1} into H_i—describing a unique path $Z_0, Z_1, \ldots, Z_\ell$ between H_{i-1} and H_i in the Markov chain $\mathbb{G}$. This sequence of course depends on the available swap operations. In [6] these are the usual (bipartite) circular C_4-swap operations. In this work these correspond to the restricted swap operations. These operations, while exchanging chords in the realizations along the alternating cycle C_i, also use some further chords. Therefore the edge set of any Z_j is not completely contained by $E(X) \cup E(Y)$; there exist a small number of edges in Z_j which are non-edges in X and in Y, or non-edges in Z_j but are edges in X and Y. If Z_j is between the milestones H_{i-1} and H_i, then C_k for $k \leq i-1$ alternates in Z_j, and C_i alternates with a "small error": there is a very small number of vertices where the alternation does not hold.

Sinclair's method requires this number to be small. In the original application this number is actually one (See [6], Section 5, (F)(c).) In the original application this number is actually one. Here, as we saw in Section "Milestones", this number is three: that many bad chords may occur after any particular ReDeSe. As we saw all these chords are adjacent to the same vertex u_1.

These numbers are used by our method to determine the size of a parameter set $\mathbb{B}$. This parameter set must have a polynomial size. When we have one bad chord, then it is determined by its end points—there are at most n^2 possibilities for them. This contributes with an n^2 multiplicative factor to the size of $\mathbb{B}$. When we have at most three bad chords, then they can be

chosen in at most n^4 ways: vertex u_1 is fixed (n different choices), while the other three end points can be chosen at most n^3 independent ways. Altogether it contributes with an at most n^4 multiplicative factor to the size of $\mathbb{B}$. This remark finishes the proof for the case of 1F1S restricted swap operations.

1F1S Restricted Degree Sequence Problem is a Self-Reduced Counting Problem

In Computer Science there are two special complexity classes, FPRAS and FPAUS, which are concerned with the approximability of counting problems. One can find detailed definitions for these complexity classes, for example, in [7]. Here we only give a sketchy description of the points that are important to our case.

Roughly speaking a counting problem is in FPRAS (Fully Polynomial Randomized Approximation Scheme) if the number of solutions can be estimated fast with a randomized algorithm, such that the estimation has a small relative error with high probability.

A counting problem is in FPAUS (Fully Polynomial Almost Uniform Sampler) if the solution can be sampled fast with a randomized algorithm that generates samples following a distribution being very close to the uniform one.

It is easy to see that a counting problem is in FPAUS if there is a rapidly mixing Markov chain for which

- a starting state can be generated in polynomial running time;
- one step in the Markov chain can be conducted in polynomial running time; and
- the relaxation time of the Markov chain grows only polynomially with the size of the problem.

The Markov chain we defined in the 1F1S problem satisfies all these requirements.

Jerrum, Valiant and Vazirani proved that any *self-reducible counting problem* is in FPRAS iff it is in FPAUS [7]. A counting problem is self-reducible if the solutions for any problem instance can be generated recursively such that after each step in the recursion, the remaining task is another problem instance from the same problem, and the number of possible branches at each recursion step is polynomially bounded by the size of the problem instance.

Clearly, a graph with prescribed degree sequence can be built recursively by telling the neighbors of a node at each step, then removing the node in question and reducing the degrees of the selected neighbors. However, this type of recursion does not satisfy all the requirement for being self-reducible since there might be exponentially many possibilities how to select the neighbors of a given vertex.

On the other hand, the degree sequence problem with a forbidden one factor and one star is a self-reducible counting problem. Indeed, consider the center of the (possibly empty) star, $s \in U$, and the vertex $v \in V$ with the smallest index for which (s, v) is a chord. Any solution for the current problem instance belongs to one of the following two cases:

- The chord (s, v) is not present in the solution. In this case, extend the size of the star by adding chord (s, v) to the forbidden set, and do not change the degrees. This is another problem instance from the 1F1S problem, whose solutions are the continuations of the original problem belonging to this case.
- The chord (s, v) is present in the solution. In this case, extend the size of the star by adding chord (s, v) to the forbidden set, and decrease both d_s and d_v by one. The new degree sequence is still a bipartite 1F1S restricted degree sequence which is half-regular in class U

(except, possibly, at vertex s), and the solutions of this new problem extended with the previously decided step provide solutions to the original problem.

Since the 1F1S counting problem is a self reducible counting problem, and we proved that it is in FPAUS therefore it is also in FPRAS: via our sampling process one can solve the approximate counting problem with high probability.

We finish this paper with a short analysis of the connections between our approach and the paper [18] of Jerrum, Sinclair and Vigoda. Their seminal result from 2004 solved the uniform sampling problem of perfect 1-factors of a given graph. As their Corollary 8.1 pointed out this method can be applied for uniform sampling of the set of all possible realizations of a given f-factor of a complete graph. It also proves that the problem is in FPAUS therefore in FPRAS as well.

Since the restricted degree sequence problem in general is equivalent to the f-factor problem, therefore our 1F1S ReDeSe problem is only a special case of the f-factor problem, so the JSV result applies to it. This describes the similarity.

The important differences lay in the swap operations applied in the JSV method and in the Kannan-Tetali-Vempala Markov chain. In the JSV method a special graph $\mathfrak{G}$ is introduced for the sampling via Tutte's gadgets. Then the swap operations are working on the graph $\mathfrak{G}$ with the unintended result that for a (sometimes very long) sequence of swaps does not change at all the generated f-factor. Combining this issue with the known relative slow mixing time of the Jerrum-Sinclair-Vigoda's Markov chain, the resulted approach in not suitable for any practical application.

Our Markov chain operates in the original graph and each jump provides a new realization of the original degree sequence problem. Therefore our Markov chain is presumably much faster than the JSV chain, furthermore the JSV theorem does not proves the rapidly mixing nature of our Markov chain. Similarly it does not prove that this Markov chain is a self reducible procedure.

Acknowledgments

PLE and IM acknowledge financial support from grant #FA9550-12-1-0405 from the U.S. Air Force Office of Scientific Research (AFOSR) and the Defense Advanced Research Projects Agency (DARPA). PLE was partly supported by the Alexander von Humboldt-Foundation, when this author visited Universität Hamburg in Fall of 2014. SZK was partly supported by Hungarian NSF, under contract K77476 and NK105645. IM was partly supported by Hungarian NSF, under contract PD84297. LS was partly supported by Hungarian NSF, under contract NK 83726

A preliminary version of this paper can be found as *arXiv* **1301.7523**

Author Contributions

Wrote the paper: PLE SZK IM LS.

References

1. Newman, M.E.J., Barabási, A.L., Watts, D.J.: *The Structure and Dynamics of Networks* (Princeton Studies in Complexity, Princeton UP) (2006), pp 624.
2. Newman M.E.J.: *Networks: An Introduction* Oxford University Press, March 2010, pp. 784.
3. Kannan R., Tetali P., Vempala S.: Simple Markov-chain algorithms for generating bipartite graphs and tournaments, *Rand. Struct. Alg.* 14 (4) (1999), 293–308. (It's extended abstract was published in the proceeding of FOCS 1997.) doi: 10.1002/(SICI)1098-2418(199907)14:4%3C293::AID-RSA1%3E3.0.CO;2-G

4. Cooper C., Dyer M., Greenhill C.: Sampling regular graphs and a peer-to-peer network, *Comb. Prob. Comp.* 16 (4) (2007), 557–593. doi: 10.1017/S0963548306007978
5. Greenhill C.: A polynomial bound on the mixing time of a Markov chain for sampling regular directed graphs, *Elec. J. Combinatorics* 18 (2011), #P234.
6. Miklós I., Erdős P.L., Soukup L.: Towards random uniform sampling of bipartite graphs with given degree sequence, *Electronic J. Combinatorics* 20 (1) (2013), #P16, 1–49.
7. Jerrum M. R., Valiant L. G., Vazirani V. V.: Random generation of combinatorial structures from a uniform distribution, *Theoret. Comput. Sci.*, 43 (2–3) (1986), 169–188. doi: 10.1016/0304-3975(86)90174-X
8. Sinclair A.: Improved bounds for mixing rates of Markov chains and multicommodity flow, *Combin. Probab. Comput.* 1 (1992), 351–370. doi: 10.1017/S0963548300000390
9. Havel V.: A remark on the existence of finite graphs. (in Czech), *Časopis Pěst. Mat.* 80 (1955), 477–480.
10. Hakimi S.L.: On the realizability of a set of integers as degrees of the vertices of a simple graph. *J. SIAM Appl. Math*. 10 (1962), 496–506. doi: 10.1137/0110037
11. Kim Hyunju, Toroczkai Z., Erdős P.L., Miklós I., Székely L.A.: Degree-based graph construction, *J. Phys. A: Math. Theor.* 42 (2009) 392001 (10pp) doi: 10.1088/1751-8113/42/39/392001
12. Gale D.: A theorem on flows in networks, *Pacific J. Math*. 7 (2) (1957), 1073–1082. doi: 10.2140/pjm.1957.7.1073
13. Petersen J.: Die Theorie der regularen Graphen, *Acta Math*. 15 (1891), 193–220. doi: 10.1007/BF02392606
14. Ryser H. J.: Combinatorial properties of matrices of zeros and ones, *Canad. J. Math*. 9 (1957), 371–377. doi: 10.4153/CJM-1957-044-3
15. Kleitman D.J., Wang D.L.: Algorithms for constructing graphs and digraphs with given valences and factors, *Discrete Math*. 6 (1973), 79–88. doi: 10.1016/0012-365X(73)90037-X
16. Erdős P.L., Miklós I., Toroczkai Z.: A simple Havel-Hakimi type algorithm to realize graphical degree sequences of directed graphs, *Elec. J. Combinatorics* 17 (1) (2010), R66 (10pp)
17. Tutte W.T.: The factors of graphs, *Canad. J. Math*. 4 (1952), 314–328. doi: 10.4153/CJM-1952-028-2
18. Jerrum M.R., Sinclair A., Vigoda E.: A Polynomial-Time Approximation Algorithm for the Permanent of a Matrix with Nonnegative Entries, *Journal of the ACM* 51(4) (2004), 671–697. doi: 10.1145/1008731.1008738
19. Erdős P.L., Király Z., Miklós I.: On graphical degree sequences and realizations, *Combinatorics, Probability and Computing* 22 (3) (2013), 366–383. doi: 10.1017/S0963548313000096