Matching Algorithms in the Sparse Stochastic Block Model

Anna Brandenberger
Department of Mathematics, Massachusetts Institute of Technology, U.S.A.

Byron Chin
Department of Mathematics, Massachusetts Institute of Technology, U.S.A.

Nathan S. Sheffield
Massachusetts Institute of Technology, U.S.A.

Divya Shyamal
Massachusetts Institute of Technology, U.S.A.

Abstract

In sparse Erdős–Rényi graphs, it is known that a linear-time algorithm of Karp and Sipser achieves near-optimal matching sizes asymptotically almost surely, giving a law-of-large numbers for the matching numbers of such graphs in terms of solutions to an ODE [9]. We provide an extension of this analysis, identifying broad ranges of stochastic block model parameters for which the Karp–Sipser algorithm achieves near-optimal matching sizes, but demonstrating that it cannot perform optimally on general stochastic block model instances. We also consider the problem of constructing a matching *online*, in which the vertices of one half of a bipartite stochastic block model arrive one-at-a-time, and must be matched as they arrive. We show that, when the expected degrees in all communities are equal, the competitive ratio lower bound of 0.837 found by Mastin and Jaillet for the Erdős–Rényi case [14] is achieved by a simple greedy algorithm, and this competitive ratio is optimal. We then propose and analyze a linear-time online matching algorithm with better performance in general stochastic block models.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis

Keywords and phrases Matching Algorithms, Online Matching, Stochastic Block Model

Digital Object Identifier 10.4230/LIPIcs.AofA.2024.24

Related Version *Extended Version*: https://doi.org/10.48550/arXiv.2403.02140

Acknowledgements We thank Elchanan Mossel for suggesting the problem, and David Jerison for helpful comments.

1 Introduction

Real-life allocation problems, particularly those related to the display of ads in search engine results, have motivated a substantial line of research investigating the problem of online matching in random graphs. Most prior work on this problem assumes the vertices on one side of a bipartite graph are drawn i.i.d. from an adversarially-chosen distribution. In that setting, upper bounds on the competitive ratio are known [13]. However, in Erdős–Rényi graphs, it is possible to exceed these bounds [14]. One might therefore hope that graphs arising in nature tend to permit better online matching algorithms than adversarial distributions. In this work, we consider matching problems in the ***stochastic block model***, in which vertices belong to one of a constant number of classes, and the probability of an edge between two vertices depends only on their classes. This is a broad class of structured distributions on graphs, which includes the Erdős–Rényi model as a special case. In stochastic block models, neither the optimal online matching algorithm nor the (offline) matching number are known.

We make progress on both of these questions, finding expressions for the matching number in a number of regimes, and proposing and analyzing several heuristics for online matching.

1.1 Preliminaries

The idea of a population divided into a fixed number of distinct but internally-homogeneous groups is captured by the **stochastic block model**, first proposed by Holland et. al. to model social networks [6].

▶ **Definition 1** (Stochastic block model). *For fixed q, consider q disjoint sets of vertices (classes) $S_1, \ldots, S_q$, and a symmetric probability matrix associating a value $p_{ij} \in [0, 1]$ to each pair i, j of classes. Given these parameters, the associated **stochastic block model** is the distribution over graphs obtained by including each edge (u, v) independently with probability $p_{\sigma_u \sigma_v}$, where σ_u and σ_v are the classes to which u and v, respectively, belong.*

If $q = 1$, we recover the Erdős–Rényi graph $G(n, p)$ on n vertices with edge probability p. If $q = 2$, where each class has n vertices and $p_{12} = p$, $p_{11} = p_{22} = 0$, we recover the bipartite Erdős–Rényi graph $G(n, n, p)$.

We are interested in the number of vertices in a maximum-cardinality matching on a graph drawn from this distribution, which we call the ***matching number***. We are also interested in the problem of constructing a matching online.

▶ **Definition 2** (Online bipartite matching problem). *Given a bipartite stochastic block model, we can define an associated online matching problem. An algorithm is given the classes of n right vertices, and knows a distribution over the left classes. For each of n time-steps, a new left-node is revealed (a left label is drawn from the distribution, and coins are flipped according to the block model probabilities to determine which edges it has to the right vertices) – the algorithm then decides which, if any, of these edges to add to M. Once it has made this decision, it is never allowed to revisit that vertex.*

The most interesting range of stochastic block model parameters turns out to be the ***sparse regime***, when all probabilities p_{ij} are $\Theta(1/n)$. When p_{ij} grows faster than $1/n$, as n grows to infinity the graph becomes dense enough such that with high probability, there is a near perfect matching between S_i and S_j. On the other hand, when p_{ij} grows slower than $1/n$, the graph becomes so sparse that we can include almost every edge in M. Therefore, we consider the regime in which $p_{ij} = c_{ij}/n$ for some constants c_{ij}.

1.2 Background

In 1981, Karp and Sipser demonstrated that a simple linear-time heuristic achieves matchings within $o(n)$ of the true matching number of Erdős–Rényi graphs with high probability [9]. By associating the performance of that algorithm with a Markov chain and examining the limiting differential equation, they were able to prove a law of large numbers on the matching number of such graphs. In 1998, Frieze, Pittel, and Aronson improved the error estimate of the Karp–Sipser algorithm in Erdős–Rényi graphs from $o(n)$ to $n^{1/5+o(1)}$ [1]. In 2011, Bohman and Frieze extended analysis of the Karp–Sipser algorithm to the model of graphs drawn uniformly over a fixed degree sequence, showing that a log concavity condition is sufficient for the algorithm to find near-perfect matchings in such graphs [2]. Because of its simplicity, the Karp–Sipser algorithm has also received attention as a practical method for data reduction; some recent work investigates efficient implementations [11, 12].

The online bipartite matching problem was first introduced by Karp, Vazirani and Vazirani in 1990; they achieved a tight $1 - 1/e$ competitive ratio on worst-case inputs [10]. In 2009, Feldman et. al. showed when the left vertices are instead drawn from an arbitrary known distribution, with integral expected arrival rates, it is possible to get a competitive ratio strictly better than $1-1/e$ [5]. For arbitrary distributions, the best known algorithm achieves competitive ratio 0.716, and there is an upper bound of 0.823 [8, 13]. There has also been work considering algorithms for specific left vertex distributions. Mastin and Jaillet found that in $G(n, n, p)$, the random bipartite graph where all edges are independent and equally likely to exist, all greedy algorithms achieve competitive ratio at least 0.837 – they conjectured that this lower bound is optimal, but were unable to compute the matching number [14]. Sentenac et. al. studied the problem in the 1-dimensional geometric model, where they found expressions for both the matching number and the performance of a particular online heuristic [16]. To the best of our knowledge, the only previous work that considers the stochastic block model is by Soprano-Loto et. al., who consider the regime where the graph is dense (i.e. all probabilities are constants not depending on n), and characterize when it is possible to achieve an asymptotically near-perfect matching [17].

1.3 Main Contributions

We show that the Karp–Sipser algorithm achieves near-optimal matchings for probability matrices p_{ij} satisfying any of the following conditions. However, the algorithm does not achieve near-optimal matchings in general stochastic block models.

- **Equitable**: We call a stochastic block model "equitable" if each vertex has the same expected degree, i.e., for every i, $\sum_j p_{ij}|S_j| = \sum_j c_{ij}|S_j|/n = c$ for some constant c. We show that the asymptotic matching number for any such graph is $\alpha n + o(n)$ where α is an explicit constant. See Theorem 5 for the full statement.
- **Sub-Critical**: When $\sum_j c_{ij}|S_j|/n < e$ for all i, we show that the model is in a sub-critical regime similar to the one found for Erdős–Rényi graphs. We show that the asymptotic matching number converges to the solution of an explicit ODE, see Theorem 9.
- **Bipartite Erdős–Rényi**: We also determine, in terms of the solution of an explicit ODE, the asymptotic matching number of $G(kn, n, c/n)$, the bipartite graph with part sizes kn and n and independent edge probability c/n; see Lemma 13. This case is of particular interest as a simple example for which usual arguments about the Karp–Sipser algorithm fail.

With regards to the online matching problem, we will first show that a simple strategy, GREEDY, of matching to uniform random neighbours achieves the optimal 0.837 competitive ratio in any equitable model. We then propose the more sophisticated heuristic, SHORTSIGHTED, of preferring the available class that maximizes the probability of finding a match on the next step. While SHORTSIGHTED performs extremely close to the online optimal in experiments, we are able to show analytically that it is not an asymptotically optimal algorithm in general.

2 Analysis of the Karp–Sipser Algorithm for Offline Matching

2.1 Karp–Sipser Algorithm and Outline of Analysis

Karp and Sipser proposed the following greedy algorithm: whenever there exists a vertex of degree 1 in the graph, choose one uniformly at random, add its edge to the matching, and remove the vertex and its neighbor from the graph. When there are no degree 1 vertices,

instead choose a uniform random edge to add to the matching, removing both of its incident vertices from the graph.

When there exist degree 1 vertices it is always "safe" to add their edges in the sense that there exists an optimal matching which includes these edges. Therefore, any "mistakes" the algorithm makes can only happen after the first time the graph no longer contains any degree 1 vertices (we call the steps before this ***Phase 1***, and the steps after this ***Phase 2***). The analysis of this algorithm in the Erdős–Rényi graph setting proceeds as follows [9, 1]:

1. Conditioned on the state of a Markov process on small tuples of integers, the graph maintains a simple distribution law even after several steps of the algorithm.
2. Estimating transition probabilities from the degree distribution, we can appeal to known approximation theorems to conclude that the Markov process stays close to the solution of a corresponding ODE as $n \to \infty$ with high probability.
3. In Phase 2 of the algorithm, it is very likely that the algorithm finds a near-perfect matching on the remaining graph. Since the algorithm makes only optimal decisions in Phase 1, this means that overall it finds a matching within $o(n)$ vertices of the true optimal value, and so the matching number of the graph is described by solutions to the ODE from step 2.

We apply steps analogous to 1 and 2 in the general stochastic block model. On the other hand, we find that step 3 does not always hold – that is, there exist stochastic block models in which Phase 2 does not find a near-perfect matching. We list some interesting cases in which we can show Phase 2 does find a near-perfect matching, and give an example where a precise analysis is possible even when it does not.

2.2 Convergence of Phase 1 Transitions to Continuous Approximation

In Appendix A, we show that to an observer tracking only the following values for all class labels i, j: the number E_{ij} of j-type half-edges (i.e. edges whose other endpoint lies in class j) incident to label-i vertices, the number T_i of label-i vertices of degree exactly 1 (the "thin" vertices), and the number F_i of label-i vertices of degree at least 2 (the "fat" vertices), the system follows a Markov chain, whose transition probabilities we determine from the limiting distribution of degrees in the block model. Letting $\bar{E}_{ij} = \frac{E_{ij}}{n}$, $\bar{F}_i = \frac{F_i}{n}$, and $\bar{T}_i = \frac{T_i}{n}$, where n is the total number of vertices in the original graph, this analysis gives

$$\mathbb{E}[\Delta \bar{E}_{ij}] = \frac{-\omega_{ij} - \omega_{ji} - \sum_l \omega_{li}\delta_{ij} - \sum_l \omega_{lj}\delta_{ji}}{n}$$

$$\mathbb{E}[\Delta \bar{F}_i] = -\frac{(\sum_l \omega_{li})\left(\frac{\bar{h}_i - \bar{T}_i}{\bar{h}_i}\right) - \sum_j \sum_l \omega_{jl}\delta_{li}\theta_i}{n}$$

$$\mathbb{E}[\Delta \bar{T}_i] = -\frac{(\sum_l \omega_{il}) - (\sum_l \omega_{li})\left(\frac{\bar{T}_i}{\bar{h}_i}\right) - \sum_j \sum_l \omega_{jl}\delta_{li}\left(\frac{\bar{T}_i}{\bar{h}_i} - \theta_i\right)}{n},$$

where $h_i = \sum_l E_{il}$, $\omega_{ij} = \frac{T_i}{\sum_l T_l} \cdot \frac{E_{ij}}{h_i}$, $\delta_{ij} = \frac{h_i - T_i}{h_i} \cdot \frac{\lambda_i}{1 - e^{-\lambda_i}} \cdot \frac{E_{ij}}{h_i}$, and $\theta_i = \frac{h_i - T_i}{h_i} \cdot \frac{\lambda_i}{e^{\lambda_i} - 1}$, for λ_i a solution to $\frac{\lambda_i(e^{\lambda_i} - 1)}{e^{\lambda_i} - 1 - \lambda_i} = \frac{h_i - T_i}{F_i}$. These values come from our degree distribution estimates in Appendix A, where h_i is the total number of half-edges attached to class i, ω_{ij} is the probability that a random degree-1 vertex is in class i and has neighbour in class j, δ_{ij} is the expected number of (additional) j-type half edges shared by the endpoint of a random class-i half-edge, and θ_i is the probability that a random class-i half-edge is incident to a degree-2 vertex.

This is a process that takes order n time steps, and where the expected change at each time step scales like $\frac{1}{n}$. Informally, we can observe that in the limit of n, the many small steps should average out and produce a process evolving according to their expectations; this suggests looking at the following system of equations:

$$\frac{d}{dt}\bar{E}_{ij}(t) = -\omega_{ij}(t) - \omega_{ji}(t) - \sum_l \omega_{li}(t)\delta_{ij}(t) - \sum_l \omega_{lj}(t)\delta_{ji}(t)$$

$$\frac{d}{dt}\bar{F}_i(t) = -\left(\sum_l \omega_{li}(t)\right)\left(\frac{\bar{h}_i(t) - \bar{T}_i(t)}{\bar{h}_i(t)}\right) - \sum_j \sum_l \omega_{jl}(t)\delta_{li}(t)\theta_i(t)$$

$$\frac{d}{dt}\bar{T}_i(t) = -\left(\sum_l \omega_{il}(t)\right) - \left(\sum_l \omega_{li}(t)\right)\left(\frac{\bar{T}_i(t)}{\bar{h}_i(t)}\right) - \sum_j \sum_l \omega_{jl}(t)\delta_{li}(t)\left(\frac{\bar{T}_i(t)}{\bar{h}_i(t)} - \theta_i(t)\right)$$

with initial conditions $\bar{E}_{ij}(0) = c_{ij}\bar{S}_i\bar{S}_j$, $\bar{F}_i(0) = \bar{S}_i\left(1 - \left(1 + \sum_j c_{ij}\bar{S}_j\right)e^{-\sum_j c_{ij}\bar{S}_j}\right)$, and $\bar{T}_i(0) = \bar{S}_i\left(\sum_j c_{ij}\bar{S}_j\right)e^{-\sum_j c_{ij}\bar{S}_j}$, where c_{ij} and $\bar{S}_i = |S_i|/n$ are the connection probabilities and label class sizes, respectively, of the stochastic block model instance. We justify this passage to differential equations by Wormald's theorem [19], which guarantees that, in the limit of n, the evolution of Phase 1 stays close to the unique solution of this ODE with probability approaching 1 (this is formally justified in Appendix B). This implies the following:

▶ **Lemma 3.** *If $\mathcal{Y}(t) = \{\bar{\mathcal{E}}_{ij}, \bar{\mathcal{F}}_i, \bar{\mathcal{T}}_i\}$ is a solution to the above ODE, with high probability the total number of unmatched isolated vertices created in Phase 1 of the Karp–Sipser algorithm is $n\left(1 - 2\tau - \sum_i \bar{\mathcal{F}}_i(\tau)\right) + o(n)$, where τ is the first time such that $\bar{\mathcal{T}}_i(\tau) = 0$ for all i.*

In the Erdős–Rényi case, studying Phase 2 of the Karp–Sipser algorithm reveals that with high probability at most $o(n)$ unmatched isolated vertices are created, meaning that the algorithm is asymptotically optimal, and that the matching number is $n\left(2\tau + \bar{\mathcal{F}}_i(\tau)\right) + o(n)$ with high probability. However, this analysis turns out not to work for general stochastic block model instances. We first illustrate a few examples (namely the equitable and sub-critical cases) where, with a little bit of work, we can prove similar results; then, we examine where the algorithm fails.

2.3 Equitable Case

▶ **Definition 4.** *We call stochastic block model parameters* ***equitable*** *if there is some constant c such that for all classes i, $\sum_j \frac{c_{ij}|S_j|}{n} = c$.*

In other words, although the edge density in some parts of the graph may be higher than other parts, the expected degree of every vertex is c regardless of what label class it belongs to. In these cases, we show that not only does the Karp–Sipser algorithm construct an asymptotically-optimal matching, but that the matching size it constructs is asymptotically the same as the matching number of the Erdős–Rényi graph $G(n, c/n)$. The intuition behind this claim is that, despite the nontrivial correlation between the edges of the graph, we expect the degree distributions to look the same everywhere. The crucial point we need to justify to make this intuition precise is that the degree distributions necessarily *remain* close to equal across classes, given that they start that way.

▶ **Theorem 5.** *With high probability, the matching number of an equitable stochastic block model is $\left(1 - \frac{x + ce^{-x} + xce^{-x}}{2c}\right)n + o(n)$, where x is the smallest solution to $x = ce^{-ce^{-x}}$, and the Karp–Sipser algorithm achieves within $o(n)$ of this value.*

The theorem is proven with the help of the following three lemmas.

▶ **Lemma 6.** *Given an equitable stochastic block model, after Phase 1 of the Karp–Sipser algorithm, with high probability for every i, F_i (number of vertices of degree at least 2; note that this is the same as total number of vertices because at the end of Phase 1 there are no vertices of degree 1) and h_i (total number of incident edges) differ by at most $o(n)$ from the corresponding values in $G(n, \frac{c}{n})$.*

Proof. This follows directly from our application of Wormald's theorem. Observe that $\left(\frac{h_i(t)}{S_i}, \frac{F_i(t)}{S_i}, \frac{T_i(t)}{S_i}\right)$ satisfy a symmetric system of equations, and have a symmetric initial condition, so the symmetry will be preserved by a solution. Since Wormald's theorem guarantees that with high probability the result of Phase 1 differs by at most $o(n)$ from a solution to this ODE, we have the desired statement. ◀

▶ **Lemma 7.** *Fix a constant $\epsilon > 0$, and suppose, at the start of Phase 2 of the Karp–Sipser algorithm, the values of $F_i/\bar{S}_i$ and $h_i/\bar{S}_i$ each differ by at most $o(n)$ between classes. If the Karp–Sipser algorithm is run until some class has average degree less than $2 + \epsilon$, with high probability at most $o(n)$ isolated vertices will have been produced since the start of Phase 2, and all classes will have average degree $2 + \epsilon + o(1)$*

Proof. In the regime where the average degree $h_i/F_i > 2 + \epsilon$ for all classes i, we want to argue that the number of steps between the times when the graph is free of thin vertices is small, so that by bunching those runs of degree-1 stripping together we can again control the evolution by an ODE. The following is a high-level treatment of the argument; formal justification from Wormald's theorem is given in Appendix B:

When we first remove a random edge, this may create some vertices of degree 1. In removing those, we may create more vertices of degree 1. In general, the expected number of new degree-1 vertices created when a degree-1 vertex of class i is removed is $\sum_j \frac{E_{ij}}{h_i}\delta_i\theta_j \leq (\max_i \delta_i)(\max_j \theta_j)$, where δ_i is the expected number of half-edges sharing a vertex with a random class-i half-edge, and θ_i is the probability that a random class-i half-edge is attached to a degree-2 vertex. Now, when we have that the average degree in each class is at least $2 + \epsilon$, and that the difference in average degree between any pair of classes i and j is very small (say, less than γ), then we know by our degree estimates (see Appendix A) that $\delta_i\theta_j = \frac{\lambda_i}{1-e^{-\lambda_i}} \cdot \frac{\lambda_j}{e^{\lambda_j}-1} < 1 - \eta$ for any pair i, j of classes, where η depends on γ and ϵ (the existence of some $\gamma > 0$, $\eta > 0$ in terms of ϵ with this property is guaranteed by a continuity argument, see Appendix B). So, the expected number of degree-1 vertices created for each degree-1 vertex removed is at most a constant, $1 - \eta$, that is bounded away from 1. The size of a subcritical Galton–Watson tree with $1 - \eta$ expected offspring is very unlikely to exceed $\mathcal{O}(1/\eta)$ – by a Chernoff bound, we can argue that it is exponentially unlikely in n to achieve size $n^{\Omega(1)}$. Thus, while we are within the region where average degrees are γ-close and greater than $2 + \epsilon$, the duration of a "run" of degree-1 stripping is w.h.p. much smaller than n, and we can appeal to the law of large numbers to claim the process evolves like its expectation. Whenever the average degree is the same in all classes, the expected change in the number of edges of a given type is proportional to the number of edges currently of that type (all edges in the graph are equally likely to be chosen as the first edge removed, equally likely to be the edge chosen one step into the run, etc). So, since average degrees start out $o(1)$ away from each other, we expect them to remain that way up until one of them drops below $2 + \epsilon$.

Now, note that the rate of creation of isolated vertices on a given step is always proportional to the fraction of thin vertices in the graph: when we remove a vertex (the neighbour of a thin vertex), it has δ_i other neighbours in expectation, and by the Markov property shown in Appendix A, we know that those neighbour edges are equally likely to be any of the edges leaving the class. So, the number of vertices isolated at each step is in expectation at most $\frac{\delta_i}{v}$ times the total number of degree-1 vertices, where v is the total number of vertices remaining (i.e. not matched or isolated) in the graph. For a run of length $n^{o(1)}$, there can only ever be $n^{o(1)}$ thin vertices present at any given moment, so we expect to create $o(1)$ isolated vertices in any given run. Thus, the total number created while we're in this regime is $o(n)$ with high probability. ◀

▶ **Lemma 8.** *If the Karp–Sipser algorithm is run to completion, starting from average degree* $2 + \epsilon + o(1)$ *in every class, with high probability at most* $2\epsilon n\left(1 - 2\log(2\epsilon)\right)$ *isolated vertices are produced.*

Proof. We're no longer guaranteed that individual runs are short, so can't argue that each run isolates only $o(1)$ vertices in expectation. Instead, we will bound the number of isolated vertices by controlling the number of degree-1 vertices that are ever produced in this regime. The only way to remove a thin vertex and create more than one in its place is to have its neighbour have degree greater than 2. Since no class has average degree more than $2 + 2\epsilon$, we know that the entire graph has at most $2n\epsilon$ edges associated with vertices of degree greater than 2. Those edges are the only places we can branch out and create more degree-1 things than we consume, so, throughout the course of the rest of the algorithm, there can never be more than $2n\epsilon$ thin vertices in the graph at once.

As in Lemma 7, we observe that the expected number of vertices isolated in a single step is at most $\frac{\delta_i}{v} \leq \frac{2}{v}$ times the total number of degree-1 vertices in the graph, where v is the total number of remaining remaining. Using our bound on the number of degree-1 vertices, and a law-of-large numbers argument, we can upper bound the number of isolated vertices created in all remaining steps by $2\epsilon n + \int_{2\epsilon n}^{n} \frac{2}{v} 2\epsilon n dv = 2\epsilon n\left(1 - 2\log(2\epsilon)\right)$. ◀

Proof of Theorem 5. In Phase 1, the Karp–Sipser algorithm is guaranteed to perform optimally, and Lemmas 6, 7, and 8 gurantee that, for any constant choice of ϵ, with high probability Phase 2 isolates only $o(n) + 2\epsilon n\left(1 - 2\log(2\epsilon)\right)$ vertices. We can make this arbitrarily small by choice of ϵ; so, with high probability the total number of vertices isolated in Phase 2 is $o(n)$, meaning that overall the algorithm is within $o(n)$ of optimal. Since the ODE determining the evolution of Phase 1 evolves the same as in an Erdős–Rényi graph with parameter c/n, the total number of lost vertices must be within $o(n)$ of the number that would be lost in $G(n, c/n)$. The matching number of $G(n, c/n)$ is known to be $\left(1 - \frac{x + ce^{-x} + xce^{-x}}{2c}\right) n + o(n)$, where x is the smallest solution to $x = ce^{-ce^{-x}}$ [9, 1], so that must also be the matching number in this case. ◀

2.4 Sub-Critical case

In the Erdős–Rényi case, Karp and Sipser proved that the number of unmatched non-isolated vertices remaining in the graph after Phase 1 (which we follow recent literature in calling the "Karp–Sipser core" [3]) is $o(n)$ with high probability if $c < e$, and $\Theta(n)$ with high probability if $c > e$ [9]. Our above analysis implies that any equitable stochastic block model also follows this critical transition at $c = e$. In this section, we examine criticality in the non-equitable case. We prove the following theorem:

▶ **Theorem 9.** *If $\sum_j c_{ij}\bar{S}_j < e$ for all i, then the Karp–Sipser core has size $o(n)$ with high probability.*

Interestingly, however, we find that the converse is not true – in fact, it's possible for the model to be subcritical even when $\sum_j c_{ij}\bar{S}_j > e$ for all i. We note the following facts:

▶ **Fact 10** (Karp and Sipser [9])**.** *The set of vertices removed by Phase 1 is fixed, regardless of the order in which degree-1 vertices are stripped. So, if there exists some valid sequence of degree-1 strippings that removes a given vertex v from the graph (either matching or isolating it), that vertex is not in the Karp–Sipser core.*

▶ **Fact 11** (Implied by a result of Mossel, Neeman and Sly [15] for 2 classes; Sly and Chin [4] for greater than 2 classes)**.** *For any constant d, in the limit of n the d-neighbourhood of any vertex of G (i.e., the subgraph obtained by a BFS of depth d from the vertex) converges in distribution to the first d levels of a multitype branching process, where nodes of type i have independently Pois($c_{ij}\bar{S}_j$) children of type j, and the root class corresponds to the class of the vertex in G.*

If we can show that, under certain conditions, with probability tending to 1 with d, there exists a sequence of valid Karp–Sipser vertex removals in this tree, all of which are at depth at most d, and which result in the root being removed, this then implies that the Karp–Sipser algorithm is subcritical. This is because we know that, in the limit of n, any structure that appears in the first d levels of the tree is equally likely to appear in the d-neighbourhood of a given vertex in G; so, if with probability at least 1 - ϵ there is a way to remove the root of such a tree for any root class, the expected number of vertices remaining in G after Phase 1 is at most ϵn. The following characterizes these conditions:

▶ **Lemma 12.** *The probability of removing the root of this branching process tends to 1 in the limit of d whenever the system $x_i = e^{-\left(\sum_j c_{ij}\bar{S}_j e^{-\left(\sum_k c_{jk}\bar{S}_k x_k\right)}\right)}$ has no more than one solution $(x_1, \ldots, x_q)$ in $[0,1]^q$.*

The proof of Lemma 12 is given in the extended version of the paper, where it is derived as a corollary of results about winning probabilities of games on multitype branching processes (those results are an extension of work of Holroyd and Martin on Galton–Watson trees [7], and may be of independent interest). We now prove the forward (true) direction of our claim.

Proof of Theorem 9. By the above analysis, it suffices to show that whenever $\sum_j c_{ij}\bar{S}_j < e$ for all classes i, the function $\begin{bmatrix} x_1 \\ \ldots \\ x_q \end{bmatrix} \mapsto \begin{bmatrix} e^{-\left(\sum_j c_{1j}\bar{S}_j e^{-\left(\sum_k c_{jk}\bar{S}_k x_k\right)}\right)} \\ \ldots \\ e^{-\left(\sum_j c_{qj}\bar{S}_j e^{-\left(\sum_k c_{jk}\bar{S}_k x_k\right)}\right)} \end{bmatrix} - \begin{bmatrix} x_1 \\ \ldots \\ x_q \end{bmatrix}$ has only one root on $[0,1]^q$. Denoting $e^{-\left(\sum_j c_{ij}\bar{S}_j e^{-\left(\sum_k c_{jk}\bar{S}_k x_k\right)}\right)}$ as f_i, the Jacobian of this function looks like

$$\begin{bmatrix} f_1\left(\sum_j \left(c_{1j}\bar{S}_j\right)\left(c_{j1}\bar{S}_1\right) e^{-\left(\sum_k c_{jk}\bar{S}_k x_k\right)}\right) - 1 & \ldots & f_1\left(\sum_j \left(c_{1j}\bar{S}_j\right)\left(c_{jq}\bar{S}_q\right) e^{-\left(\sum_k c_{jk}\bar{S}_k x_k\right)}\right) \\ \ldots & \ldots & \ldots \\ f_q\left(\sum_j \left(c_{qj}\bar{S}_j\right)\left(c_{j1}\bar{S}_1\right) e^{-\left(\sum_k c_{jk}\bar{S}_k x_k\right)}\right) & \ldots & f_q\left(\sum_j \left(c_{qj}\bar{S}_j\right)\left(c_{jq}\bar{S}_q\right) e^{-\left(\sum_k c_{jk}\bar{S}_k x_k\right)}\right) - 1 \end{bmatrix}.$$

The sum of the entries in the ith row of this matrix is $f_i\left(\sum_j \left(c_{ij}\bar{S}_j e^{-\left(\sum_k c_{jk}\bar{S}_k x_k\right)} \cdot \sum_l \left(c_{jl}\bar{S}_l\right)\right)\right) - 1$. By assumption, we know $\sum_l \left(c_{jl}\bar{S}_l\right) < e$. So, the above expression is strictly less than

$ef_i \log f_i - 1$. For any value of f_i, $ef_i \log f_i$ is at most 1 (taking the derivative, we find a unique maximum at $f_i = 1/e$). So, we have shown that the sum of every row of the Jacobian is negative everywhere. Now, suppose that this function has two distinct roots, $x = (x_1, \ldots, x_q)$ and $y = (y_1, \ldots, y_q)$. Let i be the index where $y_i - x_i$ is maximal. We have increased x_i by $(y_i - x_i)$, and increased all the other coordinates of x by at most $(y_i - x_i)$. We know that the directional derivative of the ith coordinate in the $[1, \ldots, 1]^\top$ direction is negative, and that the partial derivative with respect to every $j \neq i$ is positive; this implies that the ith coordinate of the function at y must be smaller than the ith coordinate of the function at x, so they cannot both be roots. ◀

As a consequence of this, since it is guaranteed to be optimal in Phase 1, we know the Karp–Sipser algorithm gives a near-optimal matching whenever $\sum_j c_{ij}\bar{S}_j < e$ for all i. By examining the system given in Lemma 12, however, we find that this is not a necessary condition for subcriticality – the phase transition boundary is much more complicated.

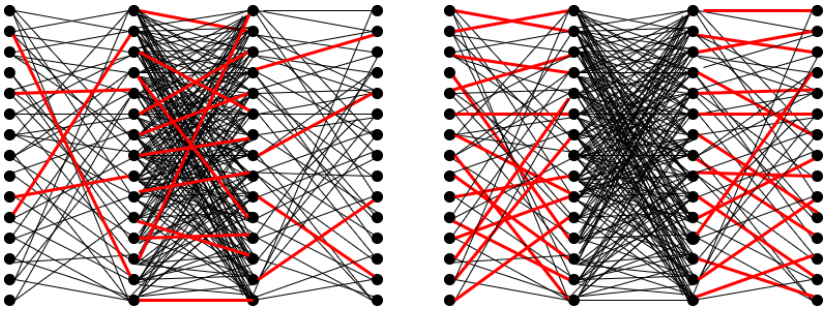
2.5 Failure of the Karp–Sipser algorithm

In the previous sections we gave two instances where we can guarantee that the Karp–Sipser algorithm achieves a near-optimal matching, both which use essentially the same framework as the Erdős–Rényi case (i.e., showing that it can achieve within $o(n)$ of a perfect matching during Phase 2, either because all degrees are close to 2 in the equitable case, or because the entire remaining graph has $o(n)$ vertices in the subcritical case). However, the algorithm does **not** return a near-optimal matching in general, and even when it does, this analytical framework does not always work. For example, consider a stochastic block model with 4 classes, all of size $n/4$, and the following probability matrix:

$$\begin{bmatrix} 0 & \frac{100}{n} & 0 & 0 \\ \frac{100}{n} & 0 & \frac{10000}{n} & 0 \\ 0 & \frac{10000}{n} & 0 & \frac{100}{n} \\ 0 & 0 & \frac{100}{n} & 0 \end{bmatrix}.$$

The matching number is very close to perfect. Even ignoring the edges between classes 2 and 3 entirely, we are left with two copies of $G(n/4, n/4, \frac{100}{n})$, on which our equitable analysis guarantees a matching containing more than a .99 fraction of vertices. However, directly analyzing the Karp-Sipser algorithm, we find that Phase 1 finishes very quickly, because the graph is dense enough that very few degree 1 vertices are created. Then, in Phase 2, for a long time we are in the regime of short runs as described in our analysis of the equitable case; in this regime, the algorithm chooses many of its edges uniformly at random, and so likely choose many of them from between classes 2 and 3. Every edge choosen between classes 2 and 3, however, effectively decreases the matching number by 2. Formalizing this argument reveals that the Karp–Sipser algorithm on this graph finds a matching containing only slightly more than 1/2 of the vertices; an illustration is given in Figure 1.

This suggests the need for a modified version of the Karp–Sipser algorithm that is given the stochastic block model parameters, and takes into account the label classes of the vertices. We propose the following: whenever the algorithm must choose a random edge, instead of choosing uniformly over all edges it instead chooses uniformly from the edge type that we estimate will create the fewest degree-1 vertices on this step. This algorithm performed well in our experimental simulations for a range of block model parameters; we conjecture that it performs near-optimally asymptotically. We have been unable to prove this, however. Part of the reason for the difficulty of analysis lies in the fact that we no longer expect a perfect



(a) Karp–Sipser **(b)** Optimal

Figure 1 A stochastic block model on which the Karp–Sipser algorithm performs suboptimally.

matching to be possible in Phase 2 – the following section illustrates this with a simple example.

2.6 Bipartite Erdős–Rényi case

The bipartite graph $G(n, n, c/n)$ with equal part sizes and i.i.d. edges corresponds to an equitable stochastic block model: each vertex on the left and each vertex on the right has c expected neighbours. So, with high probability, the Karp–Sipser algorithm performs optimally and returns a matching the same size as that of $G(2n, c/2n)$. However, the asymmetric case $G(kn, n, c/n)$ where $k \neq 1$ is not equitable, because left vertices have c expected neighbours while right vertices have kc of them. It is this case that we analyze in this section. Note that, as before, the algorithm is guaranteed to be optimal for Phase 1; we need to show that it's also near optimal for Phase 2. However, now even the optimal algorithm on Phase 2 is not guaranteed to find a near-perfect matching. Consider a graph with very unequal part sizes, and very high c; for example, $G(n, 10n, 1000/n)$. This graph is dense enough that we expect at least $5n$ vertices to remain on the right after Phase 1, however we know that the matching number of this graph is at most n – so, we can't actually expect Phase 2 to find a near-perfect matching. Thus, this is an example where the typical analytical framework does not apply [9, 1]. However, we can still use the Karp-Sipser algorithm to determine the matching numbers of such graphs. We sketch the argument.

▶ **Lemma 13.** *Let F_l, F_r and E denote the number of non-isolated unmatched vertices on the left, non-isolated unmatched vertices on the right, and edges in the graph after Phase 1 of the Karp–Sipser algorithm. With high probability, there exists a matching of size $\min(F_l, F_r) - o(n)$ on this graph. As a consequence, the asymptotic behavior of the matching numbers of bipartite Erdős–Rényi graphs can be determined from solutions to the ODEs described in Section 2.2.*

Proof. Assume without loss of generality that $F_l \leq F_r$. We would like to show that there exists a matching of size $F_l - o(n)$. To do so, we recall by the Markov property (see Appendix A) of the Karp-Sipser algorithm that the distribution of the graph G at this point in the process is a uniform random configuration model with E edges on $F_l + F_r$ vertices, conditioned on being bipartite with part sizes F_l and F_r, and having min degree 2. We'll couple G with a graph G' that's drawn from a bipartite configuration model with E edges, min degree 2, and parts both of size F_r. The way the coupling will work is to first generate G', then delete all but the first F_l vertices on the left (where "first" refers to some arbitrary ordering), and redistribute the half-edges connected to the deleted vertices among the remaining vertices on the left. Note there must exist some such randomized redistribution process that causes the

resulting graph to be distributed as G, since degree sequences of the first F_l left vertices in a graph distributed as G' are dominated by degree sequences of the left vertices in a graph distributed as G.

To show that G with high probability contains a matching of size $F_l - o(n)$, we draw (G', G) according to this coupling. By our analysis of Phase 2 of the Karp–Sipser algorithm in equitable block models, we know that with high probability G' contains a near-perfect matching. So, in particular, there exists a matching in G' that contains $F_l - o(n)$ of the first F_l vertices on the left. By construction of the coupling, all the edges involved in that matching are also present in G, so this implies the existence of a matching of size $F_l - o(n)$ in G.

This shows a lower bound on the matching number of $G(kn, n, c/n)$. Since Phase 1 of the Karp–Sipser algorithm is optimal, and once Phase 1 ends the matching number is clearly bounded by $\min(F_l, F_r)$, this lower bound is tight (up to $o(n)$). ◀

This analysis works because, even though the Karp–Sipser core does not admit a perfect matching, we have an upper bound on its true matching number (namely, the size of the smaller of the two parts) that we can show is nearly achieved. This is a natural approach to analyzing the matching number of a random graph: try to show an upper bound on the matching number of the Karp–Sipser core, and then show a variation of the Karp–Sipser algorithm that achieves close to that upper bound on the core. It would be interesting to find other block models where this approach is successful. A simple case we note as an open direction is the bipartite setting where there is only one class on the left (i.e., the class graph is "star shaped"). On such graphs, Label-Aware Karp–Sipser simplifies to "prefer edges to the available right class with minimum average degree" – it seems conceivable that some form of analogous argument could show that this is optimal.

3 Analysis of Online Matching Heuristics

In this section, we consider the problem of ***online matching*** in bipartite stochastic block models with q left and q right classes. We assume that vertices are assigned classes uniformly and independently at random, with classes of the right vertices known to the algorithm ahead of time, and vertices on the left (along with all the edges incident to them) arriving one-at-a-time. Uniformity of the class distribution is roughly without loss of generality; variations could be approximated by further subdividing the classes.

Before discussing algorithms for online matching in stochastic block models, we recall the simpler setting of $G(n, n, c/n)$. First, note that any online algorithm that chooses not to match a left vertex when it is adjacent to at least one available right vertex is sub-optimal. Since, in this setting, all right vertices are indistinguishable, any online algorithm that always matches when possible is optimal. This gives the following:

▶ **Fact 14** (Mastin and Jaillet [14]). *In the online setting, the expected size of a matching in $G(n, n, c/n)$ produced by an optimal online algorithm is given by $\left(1 - \frac{\ln(2-e^{-c})}{c}\right) n$.*

The stochastic block model setting allows for more nuance; designing optimal algorithms is nontrivial in general. While it is possible to brute-force compute an optimal online strategy in time $\Theta(n^{q+1})$ using dynamic programming, we leave open the question of whether such an optimum can be achieved asymptotically in constant time per decision. Instead, we first show that in the restricted class of *equitable* stochastic block models, a simple greedy heuristic still achieves asymptotically optimal matchings. We then propose a more sophisticated

heuristic with good experimental performance, but demonstrate a case where it can be shown analytically to be asymptotically sub-optimal.

3.1 Analysis of GREEDY

First, consider the simplest possible matching heuristic. For each left vertex that arrives, GREEDY chooses uniformly at random one of the available edges adjacent to that vertex to add to the matching. While GREEDY is sub-optimal in general, we can show that it returns an asymptotically optimal matching in any equitable block model.

▶ **Lemma 15.** *In the equitable case, i.e. when all of the vertices have the same expected degree* $c = \frac{1}{q}\sum_i c_{ij}$*, GREEDY returns a matching of expected size* $\left(1 - \frac{\ln(2-e^{-c})}{c}\right)n$ *(as in the bipartite Erdős-Rényi case; see Fact 14).*

Proof. Let R_j denote the set of vertices in right class j, and $R_j^{(t)} \subseteq R_j$ denote the subset that remain unmatched after t steps. If the next left vertex to arrive has class i, the probability that it has no available matches is $\left(\prod_{j=1}^q (1-p_{ij})^{|R_j^{(t)}|}\right)$. Conditional on at least one available match, the probability that GREEDY chooses to match to a vertex of R_j is $\frac{p_{ij}|R_j^{(t)}|}{\sum_{l=1}^q p_{il}|R_l^{(t)}|} + \mathcal{O}\left(\frac{1}{\sum_l |R_l^{(t)}|}\right)$ – this can be seen by replacing the selection process with "repeatedly choose a random right vertex and match with probability p_{ij}", which has the same distribution up to $\mathcal{O}\left(\frac{1}{\sum_l |R_l^{(t)}|}\right)$ replacement error. So, we have

$$|R_j^{(t+1)}| \approx \begin{cases} |R_j^{(t)}| - 1 \text{ with probability} & \sum_{i=1}^q \frac{1}{q}\left(1 - e^{-\sum_{l=1}^q p_{il}|R_l^{(t)}|}\right)\frac{p_{ij}|R_j^{(t)}|}{\sum_{l=1}^q p_{il}|R_l^{(t)}|} \\ |R_j^{(t)}| \quad \text{with probability } 1 - \sum_{i=1}^q \frac{1}{q}\left(1 - e^{-\sum_{l=1}^q p_{il}|R_l^{(t)}|}\right)\frac{p_{ij}|R_j^{(t)}|}{\sum_{l=1}^q p_{il}|R_l^{(t)}|} \end{cases}.$$

Letting $x_j(\tau) = \frac{|R_j^{(\tau n)}|}{n}$, we can appeal to Wormald's theorem to argue that this process is controlled in the limit $n \to \infty$ by the following ODE:

$$x_j'(\tau) = -\sum_{i=1}^q \frac{1}{q}\left(1 - e^{-\sum_{l=1}^q c_{il}x_l(\tau)}\right)\frac{c_{ij}x_j(\tau)}{\sum_{l=1}^q c_{il}x_l(\tau)} \qquad j \in \{1, 2, \dots q\},$$

with initial conditions $x_j(0) = \frac{1}{q}$ for all $j \in \{1, 2, \dots q\}$. If we guess that the solution is symmetric – that is, that $x_j(\tau) = \frac{x(\tau)}{q}$ for some x and all j, τ – then these equations simplify to

$$x'(\tau) = -\sum_{i=1}^q \left(1 - e^{-\sum_{l=1}^q c_{il}x(\tau)/q}\right)\frac{c_{ij}x(\tau)/q}{\sum_{l=1}^q c_{il}x(\tau)/q} = -\sum_{i=1}^q \left(1 - e^{-cx(\tau)}\right)\frac{c_{ij}}{c} = 1 - e^{-cx(\tau)},$$

since when all the $x_j(\tau)$ are equal, we also have all the $x_j'(\tau)$ equal. Solving this ODE yields $x(1) = \left(\frac{\ln(2-e^{-c})}{c}\right)$, meaning by Wormald's that the expected matching size converges to $\left(1 - \frac{\ln(2-e^{-c})}{c}\right)n$. ◀

Note that, unlike in $G(n, n, c/n)$, it is **not** the case that any strategy for matching available vertices achieves this value – for example, the strategy of always preferring to match class R_1 will perform worse in general. It was important that we chose a strategy that evenly distributed matches across right classes. We'll now show that this lower bound is tight; i.e., that no algorithm can do asymptotically better than GREEDY on an equitable block model.

▶ **Lemma 16.** *In an equitable stochastic block model, if there are a total of $|R^{(t)}|$ unmatched vertices on the right, the probability of matching on the next step is maximized when those $|R^{(t)}|$ vertices are equally distributed among all classes (that is, each right class r has $|R_r^{(t)}| = \frac{|R^{(t)}|}{q}$ unmatched vertices).*

Proof. Let $\rho_r = \frac{|R_r^{(t)}|}{|R^{(t)}|}$ denote the fraction of unmatched right vertices belonging to class r at time t. The probability of there being no available edge in the next step of the algorithm is, as $n \to \infty$, $\frac{1}{q}\sum_l e^{-\sum_r c_{lr}\rho_r}$. By AM-GM inequality, this is at least $\left(\prod_l e^{-\sum_r c_{lr}\rho_r}\right)^{1/q} = e^{-\sum_l \frac{1}{q}\sum_r c_{lr}\rho_r} = e^{-\sum_r \frac{1}{q}(\sum_l c_{lr})\rho_r} = e^{-c}$, which is precisely the value obtained by setting all $\rho_r = \frac{1}{q}$. ◀

From this fact, we see that no algorithm can do asymptotically better than GREEDY.

▶ **Theorem 17.** *The optimal competitive ratio for any online algorithm in an equitable stochastic block model is matchingsize$\frac{c-\ln(2-e^{-c})}{(2c-x+ce^{-x}+xce^{-x})}$, where x is the smallest solution of $x = ce^{-ce^{-x}}$.*

Proof. First, note that this value is precisely what we've shown for the competitive ratio of GREEDY; this can be found by dividing the asymptotic matching size we proved in that case by the offline matching number we proved in Theorem 5. Therefore, it is left to establish that no algorithm can do better than GREEDY. We proceed by contradiction - assume there is a better algorithm, and consider the expected values of ρ_i as in Lemma 16 in this algorithm. If the ρ_i's stay within $o(1)$ of equal over all time, then the algorithm looks asymptotically identical to GREEDY. If at some point they become unequal, then at that time the algorithm must fall behind GREEDY and never catch up, by Lemma 16. ◀

Note that this value is the competitive ratio lower bound conjectured by Mastin and Jaillet to be tight for Erdős–Rényi graphs; as a special case, we have shown that conjecture [14].

3.2 A Less Greedy Algorithm

Although GREEDY is optimal in equitable models, it's not difficult to describe non-equitable models where it's very far from optimal. For example, if $p_{ij} < p_{il}$ for all l, it is always strictly better to match to vertices of class j than class l if possible (i.e. if one class is very rare to match to, it's good to take those matches when the chance arises), but GREEDY does not do this. To correct this, one might consider always matching to the right class with the lowest expected degree – however it's easy to see that this is not optimal, since when all classes have close to the same expected degree, it's better to distribute matches more evenly across classes than to focus on the one with the smallest expected degree. Determining the appropriate extent to prefer rare classes versus trying to keep class sizes balanced is a non-trivial task.

A reasonable heuristic – which we'll term SHORTSIGHTED– is to, at each step, minimize the probability of being unable to match the next vertex (i.e., the probability that the next left vertex to arrive has no available edges). That is, SHORTSIGHTED always chooses to match to the available class l that minimizes $\sum_{i=1}^{q}\frac{1}{q}\prod_{j=1}^{q}(1-p_{ij})^{|R_j^{(t)}|-\mathbb{1}_{j=l}}$.

One might hope that looking a single step into the future is sufficient to determine the optimal class to match to, and so SHORTSIGHTED is an asymptotically optimal approach. Indeed, in a range of computer experiments we found that SHORTSIGHTED found a matching

of almost the same size as a brute-force optimal online algorithm. However, it turns out that SHORTSIGHTED is not asymptotically optimal.

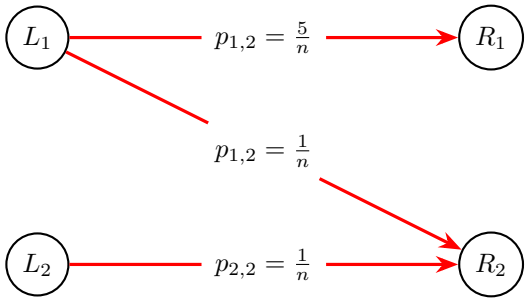


Figure 2 An example of stochastic block model parameters in which BRUTE-FORCE beats SHORTSIGHTED. Here all classes are of size $\frac{n}{2}$, and $p_{2,1} = 0$.

In the model shown in Figure 2, SHORTSIGHTED prefers class 1 whenever the number of unmatched vertices of class R_1 is at least $\frac{2 \ln 2}{5} n$. By associating SHORTSIGHTED's behaviour in the limit $n \to \infty$ with an ODE, we obtain that the expected size of the matching is $\approx 0.574946n$ (this analysis is outlined in Appendix B). To show this is suboptimal, consider the following algorithm: prefer class R_1 until $.88n$ vertices have arrived, then thereafter prefer R_2. With essentially the same differential equation analysis as above, we find that the expected size of the matching is $\approx 0.575597n$. This algorithm out-performs SHORTSIGHTED by a linear factor in n (albeit small, $\approx .0006n$), implying that SHORTSIGHTED is asymptotically sub-optimal.

References

1 Jonathan Aronson, Alan M. Frieze, and Boris G. Pittel. Maximum matchings in sparse random graphs: Karp-sipser revisited. *Random Struct. Algorithms*, 12:111–177, 1998.

2 Tom Bohman and Alan M. Frieze. Karp–sipser on random graphs with a fixed degree sequence. *Combinatorics, Probability and Computing*, 20:721 – 741, 2011.

3 Thomas Budzinski, Alice Contat, and Nicolas Curien. The critical Karp–Sipser core of random graphs, 2022. `arXiv:2212.02463`.

4 Byron Chin and Allan Sly. Optimal reconstruction of general sparse stochastic block models, 2021. `arXiv:2111.00697`.

5 Jon Feldman, Aranyak Mehta, Vahab S. Mirrokni, and S. Muthukrishnan. Online stochastic matching: Beating 1-1/e. *CoRR*, abs/0905.4100, 2009. URL: `http://arxiv.org/abs/0905.4100`, `arXiv:0905.4100`.

6 Paul W. Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109–137, 1983. URL: `https://www.sciencedirect.com/science/article/pii/0378873383900217`, `doi:10.1016/0378-8733(83)90021-7`.

7 Alexander E. Holroyd and James B. Martin. Galton–watson games. *Random Structures & Algorithms*, 59(4):495–521, 2021. URL: `https://onlinelibrary.wiley.com/doi/abs/10.1002/rsa.21008`, `arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/rsa.21008`, `doi:10.1002/rsa.21008`.

8 Zhiyi Huang, Xinkai Shu, and Shuyi Yan. The power of multiple choices in online stochastic matching, 2022. `arXiv:2203.02883`.

9 R. M. Karp and M. Sipser. Maximum matching in sparse random graphs. In *22nd Annual Symposium on Foundations of Computer Science (SFCS 1981)*, pages 364–375, 1981. doi:10.1109/SFCS.1981.21.

10 Richard M. Karp, Umesh V. Vazirani, and Vijay V. Vazirani. An optimal algorithm for on-line bipartite matching. In *Symposium on the Theory of Computing*, 1990.

11 Kamer Kaya, Johannes Langguth, Ioannis Panagiotas, and Bora Uçar. *Karp-Sipser based kernels for bipartite graph matching*, pages 134–145. Society for Industrial and Applied Mathematics, 2020. URL: https://epubs.siam.org/doi/abs/10.1137/1.9781611976007.11, arXiv:https://epubs.siam.org/doi/pdf/10.1137/1.9781611976007.11, doi:10.1137/1.9781611976007.11.

12 Johannes Langguth, Ioannis Panagiotas, and Bora Uçar. Shared-memory implementation of the karp-sipser kernelization process. In *2021 IEEE 28th International Conference on High Performance Computing, Data, and Analytics (HiPC)*, pages 71–80, 2021. doi:10.1109/HiPC53243.2021.00021.

13 Vahideh H. Manshadi, Shayan Oveis Gharan, and Amin Saberi. Online stochastic matching: Online actions based on offline statistics, 2011. arXiv:1007.1673.

14 Andrew Mastin and Patrick Jaillet. Greedy online bipartite matching on random graphs. *CoRR*, abs/1307.2536, 2013. URL: http://arxiv.org/abs/1307.2536, arXiv:1307.2536.

15 Elchanan Mossel, Joe Neeman, and Allan Sly. A proof of the block model threshold conjecture. *Combinatorica*, 38(3):665–708, 2018.

16 Flore Sentenac, Nathan Noiry, Matthieu Lerasle, Laurent Ménard, and Vianney Perchet. Online matching in geometric random graphs, 2023. arXiv:2306.07891.

17 Nahuel Soprano-Loto, Matthieu Jonckheere, and Pascal Moyal. Online matching for the multiclass stochastic block model, 2023. arXiv:2303.15374.

18 Lutz Warnke. On wormald's differential equation method, 2019. arXiv:1905.08928.

19 Nicholas C. Wormald. Differential Equations for Random Processes and Random Graphs. *The Annals of Applied Probability*, 5(4):1217 – 1235, 1995. doi:10.1214/aoap/1177004612.

A Karp–Sipser Transition Probabilities

A.1 Passing to a Configuration Model

For analytical purposes, we consider the Karp–Sipser algorithm on multigraphs drawn from a configuration model, and then justify why the results translate to stochastic block models.

▶ **Definition 18** (Blocked configuration model). *Let G be a graph drawn from a stochastic block model. For each pair of labels i, j, let E_{ij} denote the total number of edges between vertices of label i and vertices of label j in G (when $i = j$, we let E_{ij} denote* twice *the number of edges lying within class $i = j$, so that E_{ij} always refers to the number of j-type half-edges attached to label-i vertices). Construct a random multigraph as follows: if $i \neq j$, distribute E_{ij} half-edges among the i vertices and E_{ij} half-edges among the j vertices uniformly at random (à la balls-in-bins), then define edges by a uniform random pairing between the half edges on the i and j sides. This process defines a distribution over multigraphs, because in the process of "reshuffling" the half-edges of G we introduce the possibility of multiple edges and self-loops.*

Note that any result which holds with high probability in the blocked configuration model also holds with high probability for the stochastic block model, so we are justified in proving our results for this model instead (this is formally justified in the extended version of the paper).

A.2 Markov Property of Karp–Sipser Algorithm

As we run the Karp–Sipser algorithm on a random multigraph, the distribution changes, since the earlier steps of the algorithm have produced some effects conditional on the previous states of the graph. Fortunately, it turns out that these effects have a simple description.

▶ **Lemma 19.** *Suppose we generate a graph from the blocked configuration model and run the Karp–Sipser algorithm for an arbitrary number of steps. Then conditioned on the following, the resulting graph is still distributed according to the blocked configuration model.*

- *For each pair of (i, k) of block model labels, the number E_{ij} of edges between label class i and j.*
- *For each label class i, the number T_i of label-i vertices of degree exactly 1 (the "thin" vertices).*
- *For each label class i, the number F_i of label-i vertices of degree at least 2 (the "fat" vertices).*

Thus, if we collect all of these values into a tuple $Y = (E_{ij}, T_i, F_i)$, the algorithm's progress can be described by a Markov chain on Y.

The proof of this lemma is a relatively straightforward counting argument; it is described in detail in the extended version of the paper.

A.3 Estimates on Degree Distributions

In order to determine transition probabilities of this Markov process, we first note the following straightforward lemma about the degree distribution (whose proof we will again defer to the extended version):

▶ **Lemma 20.** *Fix a description tuple Y. Let nY be the tuple obtained by scaling every element of Y by n, and let G_n be a multigraph drawn from a blocked configuration model conditional on tuple nY. Choose a uniform half-edge incident to class i, and let v be its incident vertex. In the limit as $n \to \infty$, the degree of v converges in distribution to*

$$\mathbb{P}[d(v) = k] = \begin{cases} \frac{T_i}{\sum_l E_{il}} & \text{for } k = 1 \\ \frac{\left(\sum_l E_{il}\right) - T_i}{\sum_l E_{il}} \cdot \frac{\lambda_i^{k-1}}{(k-1)! e^{\lambda_i}\left(1 - e^{-\lambda_i}\right)} & \text{for } k > 1, \end{cases}$$

where λ_i is a solution to $\frac{\lambda_i e^{\lambda_i} - \lambda_i}{e^{\lambda_i} - 1 - \lambda_i} = \frac{\left(\sum_l E_{il}\right) - T_i}{F_i}$.

This allows us to determine the probability that a random edge is adjacent to a degree 2 vertex, which is important for understanding the evolution of the algorithm.

▶ **Corollary 21.** *In particular, as $n \to \infty$, the probability that a random edge into class i connects to a vertex of degree exactly 2 tends to $\frac{\left(\sum_l E_{il}\right) - T_i}{\left(\sum_l E_{il}\right)} \cdot \frac{\lambda_i}{e^{\lambda_i} - 1}$.*

The above lemma also allows us to determine the label distribution in the neighborhood of a randomly chosen edge.

▶ **Corollary 22.** *Let uv be a randomly chosen edge where u has label i and v has label j. As $n \to \infty$, the number of neighbors of u with label k tends to $\frac{\left(\sum_l E_{il}\right) - T_i}{\sum_l E_{il}} \cdot \frac{\lambda_i}{1 - e^{-\lambda_i}} \cdot \frac{E_{ik}}{\left(\sum_l E_{il}\right)}$.*

A.4 Computing Transition Probabilities

From these degree distribution estimates, we produce estimates on the transition probabilities of the Markov process. On each step of the algorithm, if there exist degree 1 vertices, the algorithm chooses one of them and removes it and its neighbour. So, the graph loses one edge from between class i and j whenever that degree 1 vertex has its edge between i and j. When the neighbour vertex is in class i, the graph also loses edges equal to however many neighbours it had in class j. Similar accounting can be made for the number of fat or thin vertices in a class: the graph loses a fat vertex either by having it as the neighbour of the degree 1 vertex we removed, or by having it initially have degree 2 and appear as the neighbour of a removed neighbour, so that it's then reduced to degree 1. A thin vertex is lost whenever its the removed degree 1 vertex, whenever its the neighbour of the removed vertex, or whenever its a neighbour of the neighbour of a removed vertex, but gain one whenever a degree 2 vertex is a neighbour-of-the-neighbour. To make these expressions explicit, we define the notation:

- Let h_i denote the total number of half-edges in class i, $h_i = \sum_l E_{il}$.
- Let ω_{ij} denote the probability that a randomly selected degree-1 vertex is in class i and has its neighbour in class j, $\omega_{ij} = \frac{T_i}{\sum_l T_l} \cdot \frac{E_{ij}}{h_i}$.
- Let δ_{ij} denote expected number of other j-type half-edges attached to the vertex a random half-edge in class i is attached to. If λ_i is a solution to $\frac{\lambda_i(e^{\lambda_i}-1)}{e^{\lambda_i}-1-\lambda_i} = \frac{h_i - T_i}{F_i}$, then, by our degree estimates, $\delta_{ij} = \frac{h_i - T_i}{h_i} \cdot \frac{\lambda_i}{1-e^{-\lambda_i}} \cdot \frac{E_{ij}}{h_i}$.
- Let θ_i denote the probability that a random half-edge attached to class i is attached to a degree 2 vertex. Again, letting $\frac{\lambda_i(e^{\lambda_i}-1)}{e^{\lambda_i}-1-\lambda_i} = \frac{h_i - T_i}{F_i}$, we have $\theta_i = \frac{h_i - T_i}{h_i} \cdot \frac{\lambda_i}{e^{\lambda_i}-1}$.

Now, while there are degree-1 vertices remaining in the graph, we can write the expected change in the description tuple after one step of the algorithm as

$$\begin{aligned}
\mathbb{E}[\Delta E_{ij}] &= -\omega_{ij} - \omega_{ji} - \sum_l \omega_{li}\delta_{ij} - \sum_l \omega_{lj}\delta_{ji} \\
\mathbb{E}[\Delta F_i] &= -\left(\sum_l \omega_{li}\right)\left(\frac{h_i - T_i}{h_i}\right) - \sum_j \sum_l \omega_{jl}\delta_{li}\theta_i \\
\mathbb{E}[\Delta T_i] &= -\left(\sum_l \omega_{il}\right) - \left(\sum_l \omega_{li}\right)\left(\frac{T_i}{h_i}\right) - \sum_j \sum_l \omega_{jl}\delta_{li}\left(\frac{T_i}{h_i} - \theta_i\right).
\end{aligned}$$

On the other hand, when there are no degree 1 vertices remaining, we choose an edge uniformly at random, so, by similar reasoning,

$$\begin{aligned}
\mathbb{E}[\Delta E_{ij}] &= -\left(\frac{2E_{ij}}{\sum_k h_k}\right) - \left(\frac{2h_i}{\sum_k h_k}\right)\delta_{ij} - \left(\frac{2h_j}{\sum_k h_k}\right)\delta_{ji} \\
\mathbb{E}[\Delta F_i] &= -\left(\frac{2h_i}{\sum_k h_k}\right) - \sum_l \left(\frac{2h_l}{\sum_k h_k}\right)\delta_{li}\theta_i \\
\mathbb{E}[\Delta T_i] &= \sum_l \left(\frac{2h_l}{\sum_k h_k}\right)\delta_{li}\theta_i.
\end{aligned}$$

An important point to note is that the expected transitions above are "scale-invariant", meaning that they remain the same upon re-scaling all entries in Y by the same amount. So,

letting $\bar{E}_{ij} = E_{ij}/n$, $\bar{T}_i = T_i/n$, $\bar{F}_i = F_i/n$, we can write (for Phase 1 – Phase 2 is similar):

$$\mathbb{E}[\Delta \bar{E}_{ij}] = \frac{-\omega_{ij} - \omega_{ji} - \sum_l \omega_{li}\delta_{ij} - \sum_l \omega_{lj}\delta_{ji}}{n}$$

$$\mathbb{E}[\Delta \bar{F}_i] = -\frac{\left(\sum_l \omega_{li}\right)\left(\frac{\bar{h}_i - \bar{T}_i}{\bar{h}_i}\right) - \sum_j \sum_l \omega_{jl}\delta_{li}\theta_i}{n}$$

$$\mathbb{E}[\Delta \bar{T}_i] = -\frac{\left(\sum_l \omega_{il}\right) - \left(\sum_l \omega_{li}\right)\left(\frac{\bar{T}_i}{\bar{h}_i}\right) - \sum_j \sum_l \omega_{jl}\delta_{li}\left(\frac{\bar{T}_i}{\bar{h}_i} - \theta_i\right)}{n}.$$

B Conditions of Wormald's Theorem

We on several occasions in this paper claim that particular Markov processes remain close to a limiting system of differential equations. In this section, we step through for each of those instances the justification of those claims. The key tool is a theorem of Wormald, restated in its general form below. Here, n indexes a family of discrete time random processes, each of which has "history" sequence $H_n \in S_n^+$. The notation Y_t is shorthand for $y(H_t)$.

▶ **Theorem 23** (Wormald [19]). *Let a be fixed. For $1 \le l \le a$, let $y^{(l)} : \bigcup_n S_n^+ \to \mathbb{R}$ and $f_l : \mathbb{R}^{a+1} \to \mathbb{R}$, such that for some constant C and all l, $|y^{(l)}(h_t)| < Cn$ for all $h_t \in S_n^+$ for all n. Suppose also that for some function $m = m(n)$:*

1. *for some functions $w = w(n)$ and $\lambda = \lambda(n)$ with $\lambda^4 \log n < w < n^{2/3}/\lambda$ and $\lambda \to \infty$ as $n \to \infty$, for all l and uniformly for all $t < m$,*

$$\mathbb{P}\left[|Y_{t+1}^{(l)} - Y_t^{(l)}| > \frac{\sqrt{w}}{\lambda^2\sqrt{\log n}}\middle| H_t\right] = o(n^{-3});$$

2. *for all l and uniformly over all $t < m$, we always have*

$$\mathbb{E}(Y_{t+1}^{(l)} - Y_t^{(l)}|H_t) = f_l(t/n, Y_t^{(1)}/n, \ldots, Y_t^{(a)}/n) + o(1);$$

3. *for each l, the function f_l is continuous and satisfies a Lipschitz condition on D, where D is some bounded connected open set containing the intersection of $\{(t, z^{(1)}, \ldots, z^{(a)}) : t \ge 0\}$ with some neighbourhood of $\{(0, z^{(1)}, \ldots, z^{(a)}) : \mathbb{P}(Y_0^{(l)} = z^{(l)}n, 1 \le l \le a) \ne 0$ for some $n\}$.*

Then,

1. *For $(0, \hat{z}^{(1)}, \ldots, \hat{z}^{(a)}) \in D$, the system of differential equations*

$$\frac{dz_l}{ds} = f_l(s, z_1, \ldots, z_a), \qquad l = 1, \ldots, a,$$

has a unique solution in D for $z_l : \mathbb{R} \to \mathbb{R}$ passing through

$$z_l(0) = \hat{z}^{(l)}, \qquad 1 \le l \le a$$

and which extends to points arbitrarily close to the boundary of D.

2. *Almost surely,*

$$Y_t^l = nz_l(t/n) + o(n)$$

uniformly for $0 \le t \le \min\{\sigma n, m\}$ and for each l, where $z_l(t)$ is the solution in (i) with $\hat{z}^{(l)} = Y_0^{(l)}/n$, and $\sigma = \sigma(n)$ is the supremum of those s to which a solution can be extended.

For a simple proof of Wormald's theorem, we refer the reader to Warnke's recent exposition [18].

B.1 Phase 1 of the Karp–Sipser algorithm

The first time we make use of this differential equations method is in the analysis of the first phase of the Karp–Sipser algorithm. We will outline how to apply Wormald's theorem in this case. Here, we take Y as we defined it, f as the derivative we wrote down for the corresponding ODE, and note that with high probability, $y^{(l)}(h_t) < 100(\max_{ij} c_{ij})n$ for all sufficiently large n (the F_i and T_i components of $y^{(l)}(h_t)$ are clearly bounded by n; w.h.p. the edge counts are all initially within a factor of 100 of their expectations, and once they start that way they never increase). We take the stopping time m to be the first time all of the T_i entries of Y drop below $n^{0.01}$. Now,

1. Take for instance $w = n^{0.5}$ and $\lambda = \log n$. The probability that any vertex in the initial graph has degree polynomial in n decays exponentially in n, and it can be observed that the magnitude of a transition is bounded by twice the maximum degree, so we certainly have the desired condition.
2. This convergence of expected transition size of the Markov process to f is precisely what is guaranteed by the convergence of our degree estimates. Note that this convergence holds as the total number of thin vertices, fat vertices, and edges is going to infinity, so taking our stopping time to be m prevents border cases once these values drop down to constant sizes.
3. To show that f is Lipschitz in the neighbourhood of solutions, it suffices to show that solutions to $z'(t) = f(t, z)$ always maintain constant average degree in each label class. To do so, note that with high probability the initial average degrees are at most $100(\max_{ij} c_{ij})$ in each class, and then observe from the equations that whenever the average degree in a given class i is more than 100 at some time t, $-\sum_j \bar{\mathcal{E}}'_{ij}(t) > \bar{\mathcal{F}}'_i(t) + \bar{\mathcal{T}}'_i(t)$.

Note that we have only shown that the conditions of the theorem hold with high probability over initial graph configurations; clearly, this is sufficient for our desired result.

B.2 Phase 2 of the Karp–Sipser algorithm (equitable case)

We also use Wormald's theorem to justify our analysis of the second phase of the algorithm in the equitable case. Here, Y_i is the state of the tuple after the ith run of the algorithm – i.e., the ith time where there are no thin vertices. We define f_l to be the expected change in the tuple that one of these runs would incur if the transition probability estimates from Appendix A held exactly and did not change throughout the run.

In order for this process to have the desired Lipschitz properties, we will need to define a more restricted domain than the entire possible space of tuples. In particular, we will fix some constants γ and ϵ, and consider the domain of Y to consist only of tuples where the average degree into each class differs by at most γ, and is at least $2 + \epsilon$. The value of ϵ is chosen in the analysis; to determine γ, we observe the following:

- λ_i is defined such that $\frac{\lambda_i(e^{\lambda_i}-1)}{e^{\lambda}-1-\lambda_i}$ is equal to the average degree of class i. This is monotonically increasing in the average degree of class i; so there is some constant $\lambda > 0$ such that if the average degree in class i is at least $2 + \epsilon$, then $\lambda_i > \lambda$.
- The function $\delta_i\theta_i = \frac{\lambda_i}{1-e^{-\lambda_i}} \cdot \frac{\lambda_i}{e^{\lambda_i}-1}$ is bounded above by 1 and monotonically decreasing for $\lambda_i > 0$. So, there exists some constant $\eta > 0$ such that $\frac{\lambda}{1-e^{-\lambda}} \cdot \frac{\lambda}{e^{\lambda}-1} < 1 - 2\eta$.
- $\lambda_i, \lambda_j \mapsto \frac{\lambda_i}{1-e^{-\lambda_i}} \cdot \frac{\lambda_j}{e^{\lambda_j}-1}$ is uniformly continuous, so there exists some constant κ such that $|\lambda_i - \lambda_j| < \kappa$ implies that $|\left(\frac{\lambda_i}{1-e^{-\lambda_i}} \cdot \frac{\lambda_j}{e^{\lambda_j}-1}\right) - \left(\frac{\lambda_i}{1-e^{-\lambda_i}} \cdot \frac{\lambda_i}{e^{\lambda_i}-1}\right)| < \eta$ – which, if $\lambda_i > 2 + \epsilon$, implies $\left(\frac{\lambda_i}{1-e^{-\lambda_i}} \cdot \frac{\lambda_j}{e^{\lambda_j}-1}\right) < 1 - \eta$.

- When defined on the domain $[2+\epsilon, \infty)$, λ_i is uniformly continuous as a function of the average degree of i. So, we can define γ such that |average degree in class i − average degree in class $j| < \gamma$ implies $|\lambda_i - \lambda_j| < \kappa$ whenever average degrees are greater than $2+\epsilon$.

We will define the stopping time m of the process to be the first time it leaves this domain. Note that, as in Phase 1, the value of $|Y|$ is bounded by $100c_{\max}n$ with probability $1-o(1)$, since it is initially and can only decrease. We're now ready to verify the criteria of Wormald's theorem.

1. Take again $w = n^{0.5}$ and $\lambda = \log n$. We can describe a single run of the algorithm as a branching process: each degree-1 vertex created by the algorithm corresponds to a node that has children according to the number of degree-2 vertices adjacent to its neighbour; this corresponds to expected offspring number of $\delta_i\theta_j < 1-\eta$. After each removal we are uniform over graphs with the remaining statistics; so, as long as the branching process size is $o(n)$, we can treat the treat these offspring distributions roughly independently for each node (in particular, as long as the branching process is $o(n)$ with high probability, all offspring distributions have expectation at most $1-\frac{\eta}{2}$ regardless of the values for the other nodes). To prove that $\mathbb{P}\left[|Y_{t+1}^{(l)} - Y_t^{(l)}| > \frac{\sqrt{w}}{\lambda^2\sqrt{\log n}}\middle| H_t\right] = o(n^{-3})$, we therefore just need to show that the probability that a Galton–Watson tree with $\mu < 1-\frac{\eta}{2}$ reaches size $\frac{n^{0.25}}{(\log n)^{5/2}}$ is $o\left(\frac{1}{n^3}\right)$; this follows from a standard Chernoff bound.
2. The reason that $\mathbb{E}(Y_{t+1}^{(l)} - Y_t^{(l)}|H_t) = f_l(t/n, Y_t^{(1)}/n, \ldots, Y_t^{(a)}/n) + o(1)$ holds is because we expect a constant run duration, and we know our initial degree estimates will hold with small error terms when we remove a constant portion of the graph.
3. Continuity of f_l is clear from continuity of our degree estimates and the fact that small changes in the edge densities of the graph can't bias the branching process too heavily. Similar justification that it's Lipschitz on the given domain can be found by examining the degree estimate functions.

Since Y_0/n is with high probability $o(1)$ from having equal average degrees, and this ODE keeps equal average degrees equal, with high probability the degrees stay within $o(1)$ of equal until the stopping time.

B.3 Analysis of SHORTSIGHTED

Define the 2-D Markov Chain Z_t, where the first coordinate represented the number of unmatched R_0 vertices while the second coordinate represents the number of unmatched R_1 vertices at time t during a run of SHORTSIGHTED. We have that $Z_0 = (|R_0|, |R_1|)$. If we are in the regime where we prefer class 0, we have transition probabilities as follows:

$$\begin{aligned}\mathbb{P}(Z_{t+1} = (x-1, y)|Z_t = (x,y)) &= \frac{1}{2}\left(1-(1-p_{0,0})^x\right) + \frac{1}{2}\left(1-(1-p_{1,0})^x\right)\\ &\to \frac{1}{2}\left(1-e^{-c_{0,0}x/n}\right) + \frac{1}{2}\left(1-e^{-c_{1,0}x/n}\right)\end{aligned}$$

$$\begin{aligned}\mathbb{P}(Z_{t+1} = (x, y-1)|Z_t = (x,y)) &= \frac{1}{2}(1-p_{0,0})^x\left(1-(1-p_{0,1})^y\right) + \frac{1}{2}(1-p_{1,0})^x\left(1-(1-p_{1,1})^y\right)\\ &\to \frac{1}{2}e^{-c_{0,0}x/n}\left(1-e^{-c_{0,1}y/n}\right) + \frac{1}{2}e^{-c_{1,0}x/n}\left(1-e^{-c_{1,1}y/n}\right)\end{aligned}$$

$$\mathbb{P}(Z_{t+1} = (x, y) | Z_t = (x, y)) = \frac{1}{2}(1 - p_{0,0})^x (1 - p_{0,1})^y + \frac{1}{2}(1 - p_{1,0})^x (1 - p_{1,1})^y$$
$$\to \frac{1}{2} e^{-c_{0,0}x/n} e^{-c_{0,1}y/n} + \frac{1}{2} e^{-c_{1,0}x/n} e^{-c_{1,1}y/n}$$

We now verify the three conditions of Wormald's:

1. This simply comes from the fact that at each time step, each coordinate of Z_t can change by at most 1.
2. We use the following fact from [14]: for $n > 0$, $c \leq n/2$, and $x \in [0, 1]$, we have

$$\left| e^{-cx} - \left(1 - \frac{c}{n}\right)^{nx} \right| \leq \frac{c}{ne}$$

   We can apply this term-wise to each of our probabilities, giving us the desired condition.
3. e^x is Lipschitz continuous on $[0, 1]$, therefore we have the third condition as well.

An essentially identical proof follows for the Markov chain where we prefer class 1. Therefore we may apply Wormald's theorem to treat these expect transitions as exact in the continuous limit, and solve to determine the matching size. Verification of these conditions is similar for GREEDY.