Combinatorics, Probability and Computing (2021), **30**, pp. 762–780
doi:10.1017/S0963548320000620

CAMBRIDGE UNIVERSITY PRESS

ARTICLE

Full rainbow matchings in graphs and hypergraphs

Pu Gao[1,†], Reshma Ramadurai[2], Ian M. Wanless[3,‡] and Nick Wormald[3,*,§]

[1]Department of Combinatorics and Optimization, University of Waterloo, Ontario, N2L 3G1, Canada; [2]School of Mathematics & Statistics, Victoria University of Wellington, Wellington 6140, New Zealand and [3]School of Mathematics, Monash University, Victoria 3800, Australia

*Corresponding author. Email: nick.wormald@monash.edu

(Received 14 October 2019; revised 30 October 2020; accepted 15 May 2020; first published online 20 January 2021)

Abstract

Let G be a simple graph that is properly edge-coloured with m colours and let $\mathcal{M} = \{M_1, \ldots, M_m\}$ be the set of m matchings induced by the colours in G. Suppose that $m \leqslant n - n^c$, where $c > 9/10$, and every matching in $\mathcal{M}$ has size n. Then G contains a full rainbow matching, *i.e.* a matching that contains exactly one edge from M_i for each $1 \leqslant i \leqslant m$. This answers an open problem of Pokrovskiy and gives an affirmative answer to a generalization of a special case of a conjecture of Aharoni and Berger. Related results are also found for multigraphs with edges of bounded multiplicity, and for hypergraphs.

Finally, we provide counterexamples to several conjectures on full rainbow matchings made by Aharoni and Berger.

2020 MSC Codes: Primary 05D15; Secondary 05C70

1. Introduction

Throughout this paper the setting is a multigraph G whose edges are properly coloured with m colours, so that each colour i induces a matching M_i. We say that G contains a *full rainbow matching* if there is a matching M that contains exactly one edge from M_i for each $1 \leqslant i \leqslant m$. This paper is motivated by the following conjecture of Aharoni and Berger [1, Conjecture 2.4].

Conjecture 1.1. *If G is bipartite and each matching M_i has size $m + 1$, then G has a full rainbow matching.*

Consider a $k \times n$ array A. A *partial transversal of length* ℓ in A is a selection of ℓ cells of A from different rows and columns, and containing different symbols. A *transversal* of A is a partial transversal of length $\min(k, n)$. If A has no repeated symbol within a row it is called *row-Latin*. We say that A is *Latin* if it and its transpose are both row-Latin. If $k = n$ and A is Latin and contains exactly n symbols, then A is a *Latin square*. Conjecture 1.1 was motivated by a longstanding

†Research supported by the ARC grant DE170100716. The research was conducted when the author was affiliated with Monash University.

‡Research supported by the ARC grant DP150100506.

§Research supported by the Australian Laureate Fellowships grant FL120100125.

© The Author(s), 2021. Published by Cambridge University Press.

https://doi.org/10.1017/S0963548320000620 Published online by Cambridge University Press

conjecture of Stein [28], that every $(n-1)\times n$ row-Latin array has a transversal. This is equivalent to the restriction of Conjecture 1.1 to the case where each matching covers the same set of vertices on one side of the bipartite graph. Stein's conjecture was in turn motivated by the question of what length of partial transversal can be guaranteed to exist in a Latin square. His conjecture implies that every Latin square of order n has a partial transversal of length $n-1$ (this statement was independently conjectured by Brualdi slightly earlier; see [29] for a full survey of these conjectures and related results). The best result to date is by Hatami and Shor [17], who showed that every Latin square of order n has a partial transversal of length $n-O(\log^2 n)$. It is known that for even orders n there are at least $n^{n^{3/2}(1/2-o(1))}$ (equivalence classes of) Latin squares that do not have transversals [12]. However, a famous conjecture of Ryser [27] states that all Latin squares of odd order have transversals. In terminology similar to Conjecture 1.1, Ryser's conjecture is that if G is $K_{m,m}$ and m is odd, then G should have a full rainbow matching. This conjecture is known to fail if a single edge is removed from $K_{m,m}$. Also there are Latin arrays of odd order n containing more than n symbols but having no transversal (again, see [29] for details).

Barát and Wanless [10] considered an intermediate step between Conjecture 1.1 and its variant (that we know fails) with $m+1$ replaced by m. They showed that $\lfloor m/2\rfloor-1$ matchings of size $m+1$ together with $m-\lfloor m/2\rfloor+1$ matchings of size m need not have a full rainbow matching. They also constructed m matchings of size m inducing a bipartite multigraph with m vertices in the first part of the bipartition and $m^2/2-O(m)$ in the second part, and with no rainbow matching. This raises the question of how large one part can be before a rainbow matching is unavoidable. In [11] it is shown that if one part has m vertices and the other has at least $\lceil\frac{1}{4}(5-\sqrt{5})m^2\rceil$ vertices then there will be a full rainbow matching. Clearly, the threshold is quadratic in m for this problem. However, things change significantly if the induced bipartite graph must be simple. Montgomery, Pokrovskiy and Sudakov [24] showed in that case that if one part has m vertices and the other has at least εm^2 vertices then there will be many full rainbow matchings. Also Keevash and Yepremyan [21] showed that if one part has m vertices and the other has at least $m^{399/200}$ vertices then there will be a full rainbow matching. In particular, the threshold for this variant of the problem is subquadratic.

The first progress towards Conjecture 1.1 was by Aharoni, Charbit and Howard [4], who showed that n matchings of size $\lfloor 7n/4\rfloor$ must have a full rainbow matching. The $\lfloor 7n/4\rfloor$ term was successively improved to $\lfloor 5n/3\rfloor$ by Kotlar and Ziv [22], then $(3+\varepsilon)n/2$ by Clemens and Ehrenmüller [13], and then $\lceil 3n/2\rceil+1$ by Aharoni, Kotlar and Ziv [5]. Finally, for any fixed $\varepsilon>0$, Pokrovskiy [25] showed that if the matchings are edge-disjoint (so that G is simple) then n matchings of size $(1+\varepsilon)n$ have a full rainbow matching when n is sufficiently large. He also posed two challenges regarding improving the error term in his result (we make some progress in this direction) and generalizing to bipartite multigraphs (a feat he himself achieved in [26]).

A related result is due to Häggkvist and Johansson [16], who showed that if the matchings are all perfect and edge-disjoint then n matchings of size $(1+\varepsilon)n$ can be decomposed into full rainbow matchings, provided n is sufficiently large.

All the results discussed so far pertain to bipartite graphs. So far this case has attracted more scrutiny than the unrestricted case. However, Aharoni *et al.* [3] and Barát, Gyárfás and Sárközy [9] both consider the question of how large a rainbow matching can be found across any set of matchings. A preprint version of the former paper makes a conjecture [2, Conjecture 5.4] which includes this variant of Conjecture 1.1 as a special case.

Conjecture 1.2. *If each matching has size $m+2$ then G has a full rainbow matching.*

It is not viable to replace $m+2$ with $m+1$. For example, a 1-factorization of two copies of K_4 provides three matchings of size 4 that do not possess a full rainbow matching.

Our aim is to investigate approximate versions of Conjecture 1.2. Our main result is an analogue of Pokrovskiy's theorem from [25], but without the requirement that G is bipartite. Our

https://doi.org/10.1017/S0963548320000620 Published online by Cambridge University Press

results in this direction are stated in the next section, and include some information on the role of the maximum degree of the graph G. Related to this, in Section 7, we discuss and refute several further conjectures on full rainbow matchings made by Aharoni and Berger [1] and Aharoni *et al.* [2]. In particular, both papers include the following conjecture, which turns out not to be true.

Conjecture 1.3. *Let G be a bipartite multigraph, with maximum degree $\Delta(G)$, whose edges are (not necessarily properly) coloured. If every colour appears on at least $\Delta(G)+1$ edges, then G has a full rainbow matching.*

Finally, we note that the main result of Keevash and Yepremyan [20] implies that in any multigraph with edge multiplicities $o(n)$ that is properly edge-coloured by n colours with at least $n(1+\varepsilon)$ edges of each colour, there must be a rainbow matching that is close to full.

2. Main results

Recall our setting: G is a multigraph that is properly edge-coloured with m colours, and matching M_i is induced by colour i. Let $\mathcal{M}=\{M_1,\dots,M_m\}$ denote the family of m matchings. We say $\mathcal{M}$ is *non-intersecting* if G is a simple graph, *i.e.* if $M_i\cap M_j=\varnothing$ for every $1\leqslant i<j\leqslant m$. Otherwise it is called intersecting. Let $\Delta(\mathcal{M})=\Delta(G)$ denote the maximum degree of G.

Theorem 2.1. *Suppose that $0\leqslant\delta<1/4$ and $0<c<(1-4\delta)/10$ and n is sufficiently large. If $\mathcal{M}$ is a non-intersecting family of $m\leqslant(1-n^{-c})n^{1+\delta}$ matchings, each of size n, and $\Delta(\mathcal{M})\leqslant(1-n^{-c})n$, then $\mathcal{M}$ contains a full rainbow matching.*

Clearly, for any $\mathcal{M}$ we always have $\Delta(\mathcal{M})\leqslant m$. Thus, by taking $\delta=0$, we immediately have the following corollary.

Theorem 2.2. *Suppose that $0<c<1/10$ and n is sufficiently large. If $\mathcal{M}$ is a non-intersecting family of $m\leqslant(1-n^{-c})n$ matchings, each of size n, then $\mathcal{M}$ contains a full rainbow matching.*

Remark. Theorem 2.2 proves an approximate version of Conjecture 1.2, and thus of Conjecture 1.1. Compared with [26] we get better approximation by improving m from $(1-o(1))n$ to $n-n^{9/10+\varepsilon}$, and of course Theorem 2.2 also approximates the non-bipartite case of Conjecture 1.2. However, Theorem 2.2 does not cover the multigraph case, whereas [26] does.

Remark. One can ask about possible strengthenings of this theorem. For instance, Aharoni and Berger [1, Conjecture 2.5] conjectured essentially that one can drop the upper bound on m, and even drop the condition that each element of $\mathcal{M}$ is a matching, as long as $\Delta(\mathcal{M})\leqslant n-1$. It turns out that their conjecture is false, as shown by a graph whose components are double stars, which we give in Section 7. But the question for matchings remains open.

Our proof of Theorem 2.1 easily extends to intersecting $\mathcal{M}$, where the underlying graph G is a multigraph with relatively low multiplicity.

Theorem 2.3. *For every $\varepsilon_0>0$, if $\mathcal{M}$ is a family of $m\leqslant(1-\varepsilon_0)n$ matchings, each of size n, and every edge is contained in at most $\sqrt{n}/\log^2 n$ matchings, then $\mathcal{M}$ contains a full rainbow matching.*

The proof of Theorem 2.1 also immediately extends to rainbow matchings in uniform hypergraphs. A hypergraph $G=(V,E)$ is defined on a set of vertices V where the set E of hyperedges is a set of subsets of V. We say G is a k-uniform hypergraph if every hyperedge has size k. A matching M in G is a set of hyperedges such that no vertex in G is contained in more than one hyperedge in M.

https://doi.org/10.1017/S0963548320000620 Published online by Cambridge University Press

Theorem 2.4. *For every $\varepsilon_0 > 0$ and every integer $k \geqslant 2$, if $\mathcal{M}$ is a family of $m \leqslant (1-\varepsilon_0)n$ edge disjoint matchings in a k-uniform hypergraph H, each of size n, and every pair of vertices is contained in at most $\sqrt{n}/\log^2 n$ hyperedges, then $\mathcal{M}$ contains a full rainbow matching.*

Remark. If we restrict further by saying that every pair of vertices is contained in at most a constant number of hyperedges, then Theorem 2.4 holds if we replace $m \leqslant (1-\varepsilon_0)n$ with $m \leqslant (1-n^{-1/10+\varepsilon_0})n$. That is, we can fully recover Theorem 2.2 for arbitrary $k \geqslant 2$. This indeed covers some interesting families of hypergraphs, such as linear hypergraphs, where no two vertices are contained in more than one hyperedge. By further restricting the maximum degree of G, improved bounds on m can be achieved with minor modification of the proof. Similarly the bound on m in Theorem 2.3 can easily be improved by restricting to smaller multiplicity, or placing additional constraints on the maximum degree. We will not develop this idea further in this paper.

3. A heuristic approach

In this section we give a simplified description of the algorithm we use for full rainbow matchings, and also a heuristic argument as to why we expect it to successfully find the full rainbow matching required for Theorem 2.1. The actual proof, showing that all aspects work as intended, will be given in Section 4.

Given a non-intersecting family $\mathcal{M}$ of m matchings of size n, we do the following.

First, randomly partition $\mathcal{M}$ into subfamilies containing about εm matchings each, which we call 'chunks'. Here ε is a function of n.

Next, 'process' the chunks iteratively using the following three steps in iteration i.

(i) Pick one edge uniformly at random from each of the matchings in chunk i. Any picked edge x that is not incident to any other is added to the rainbow matching M_0 that will be output, and the end vertices of x are deleted from the graph. Edges that 'collide' with others are not added (but see step (iii)).

(ii) For each vertex surviving the first step, calculate the probability that it was deleted, and then artificially delete with such a probability to ensure that all vertices have the same probability of surviving these first two steps of the iteration. (A suitable probability will be specified in the precise analysis of the algorithm.)

(iii) For any matching M containing a 'colliding' edge in step (i), greedily choose a replacement edge x in M to add to M_0 and delete the end vertices of x from the graph.

These steps are performed for all chunks except the last, which is treated instead by greedily choosing edges from the remaining matchings. (We will show that this is highly likely to succeed.)

We now give a rough overview of the analysis of the algorithm. Let τ denote the number of iterations of the algorithm, *i.e.* the number of chunks. (For definiteness, we call the treatment of the last chunk an 'iteration', even though it is treated differently to the other chunks.) Also, let $d_v^j(i)$ denote the number of edges of chunk j that are (still) incident with vertex v after iteration i. We will specify functions g and r, and the correct probabilities in step (ii), for the following to hold iteratively for each $i = 1, \ldots, \tau - 1$ and all $j > i$:

(a) after iteration i, all surviving matchings have size approximately $r(i\varepsilon)n$,

(b) $d_v^j(i) \approx \varepsilon g(i\varepsilon) d_v$ for all surviving vertices v, where d_v is the degree of v initially (in G).

Here the sign $\approx$ is used to denote some version of concentration around the stated value.

https://doi.org/10.1017/S0963548320000620 Published online by Cambridge University Press

Specific versions of (a) and (b) are proved using the initial concentration, together with an inductive argument that computes the expected changes in the variables concerned during each iteration and shows concentration close to the expected changes. The precise inductive statements are chosen with a margin of error that rigorously contains the effect of collisions, so we will ignore these in the outline here. Our estimates are least accurate near the end of the algorithm, which is why we use a greedy algorithm at that stage. This final iteration works because there are few matchings left compared to their remaining sizes.

Here is an outline of why we expect the algorithm to succeed. Initially we have $|M| = n$ for all matchings M. Since the matchings are randomly allocated into chunks, we expect the initial set-up to satisfy

$$d_v^j(0) \approx \varepsilon d_v \quad \text{for all } j \geqslant 1 \text{ and } v \in G.$$

Assume that the first i iterations of the algorithm are complete. For iteration $i+1$, we specify the probability of v being artificially deleted in step (ii) so that the probability of surviving the first two steps is $f(i\varepsilon)$ for every remaining vertex. Then, for every vertex v that survives iteration $i+1$, the expected change in $d_v^j(i)$ $(j > i)$ is

$$\mathbb{E}(d_v^j(i+1) - d_v^j(i)) \approx -f(i\varepsilon) d_v^j(i), \tag{3.1}$$

since each of v's neighbours is deleted with probability $f(i\varepsilon)$. (Here we ignore collisions and the effect of step (iii), as mentioned above.) Hence the degrees of two different vertices remain roughly in the same proportion as long as they both survive.

For each matching in chunk j, the expected change in its size while chunk i is processed is roughly

$$-2f(i\varepsilon)|M(i)|, \tag{3.2}$$

neglecting what will turn out to be an $O(\varepsilon^2 n)$ error from the case that the two ends of an edge in M are both deleted.

Consider a vertex v that has survived i iterations. During iteration $i+1$, each matching in chunk $i+1$ has one of its edges chosen. So, using (a) above, any given edge in a matching in chunk $i+1$ that is incident with v is chosen with probability $p \approx 1/(r(i\varepsilon)n)$. This means that the probability that v is not deleted in step (i) of iteration $i+1$ is roughly $(1-p)^{d_v^{i+1}(i)} \approx 1 - p d_v^{i+1}(i)$, and hence the probability that it is deleted here is roughly

$$\frac{d_v^{i+1}(i)}{r(i\varepsilon)n}.$$

With this in mind, we can define $f(i\varepsilon)$ so that it is approximately equal to the maximum value of this probability over all v, which is determined by the maximum vertex degree. (We also add a little elbow-room to account for the collisions.) Then the probability can be appropriately specified in step (ii). We can find the maximum value by tracking the maximum degree via (3.1). Let γn denote the maximum degree of the initial graph G. By our assumption in Theorem 2.1, $\gamma \leqslant 1 - n^{-c}$. Using (b), we find that the approximate size of $\max_v d_v^j(i)$ is $\varepsilon g(x)\gamma n$, where $x = i\varepsilon$. We have

$$f(x) \approx \frac{\varepsilon g(x)\gamma n}{r(x)n} = \frac{\varepsilon \gamma g(x)}{r(x)}.$$

Letting

$$\hat{f}(x) = \gamma \frac{g(x)}{r(x)},$$

https://doi.org/10.1017/S0963548320000620 Published online by Cambridge University Press

we have $f(x) \approx \varepsilon \hat{f}(x)$. Then, if the size of each matching is approximated, as mentioned in (a), by $r(i\varepsilon)n$, equation (3.1) suggests (as $\varepsilon \to 0$, and applying it to a vertex of maximum degree) the differential equation

$$g'(x) = -\hat{f}(x)g(x) = -\gamma \frac{g(x)^2}{r(x)}.$$

Similarly, (3.2) suggests

$$r'(x) = -2\gamma g(x).$$

But then $dg/dr = g/2r$, which gives $r = Cg^2$. Initially $r(0) = 1$ and $g(0) = 1$, which yields $C = 1$. So the solution to these differential equations is

$$r(x) = (1 - \gamma x)^2, \quad g(x) = 1 - \gamma x. \tag{3.3}$$

Thus we have $d_v^j(i) \approx \varepsilon(1 - \gamma i\varepsilon)d_v$ for every surviving vertex v. For those vertices whose degrees are initially lower than the maximum degree, the derivative is proportionally lower, and hence the degrees stay in proportion. The process cannot 'get stuck' until the error in the approximation in assumption (a) becomes significantly large compared to $r(x)$. The function $r(x)$ is positive for all $0 \leqslant x \leqslant 1$, because our hypothesis $\Delta(\mathcal{M}) \leqslant (1 - n^{-c})n$ guarantees that $1 - \gamma x \geqslant n^{-c} > 0$. Thus, if the error of approximations is small enough (as we shall show), the process proceeds until the last iteration. The final (greedy) iteration is shown to work using the previous analysis to estimate the size of the remaining matchings. Note that $g(x)$ tends to zero along with $r(x)$.

The astute reader may have noticed that, in the above sketch of proof, all we needed from assumption (b) was an upper bound on all vertex degrees, and this is the approach we will take in the formal proof in the following section. Thus we will replace assumption (b) with

(b′) $d_v^j(i) \leqslant \varepsilon g(i\varepsilon)\gamma n$ for all $j > i$ and all surviving vertices v.

4. Algorithm and proof

In this section we define the algorithm precisely and then analyse it to prove Theorem 2.1.

4.1 The algorithm

Let $\mathcal{M} = \{M_1, M_2, \ldots, M_m\}$ be a non-intersecting family of matchings, each of size n. The algorithm has an initial stage, then some repeated iterations, then one final iteration. The initial stage consists of the following. First order the matchings in $\mathcal{M}$ uniformly at random. Then, for some $\varepsilon > 0$ of our choosing, partition $\mathcal{M}$ into 'chunks' $\mathcal{M}^1, \mathcal{M}^2, \ldots$, where $\mathcal{M}^1$ contains the first $\lceil \varepsilon m \rceil$ matchings, $\mathcal{M}^2$ contains the next $\lceil \varepsilon m \rceil$ matchings, and so on, up to the last chunk, which contains at most $\lceil \varepsilon m \rceil$ matchings. For ease of calculations, we will choose ε so that εm is an integer.

We next define some notation useful in defining the iterations of the algorithm. Let G be the graph induced by $\cup_{M \in \mathcal{M}} M$, and let V be its vertex set. For any $u \in V$, let E_u denote the set of edges in G that are incident with u. During the algorithm, vertices are removed from consideration for several distinct reasons, which we discuss shortly. The set $U(i)$ is the set of vertices that were removed at some point during the first i iterations. After i iterations, vertices in $V \setminus U(i)$ are said to be *surviving* and matchings are said to be *surviving* if they do not belong to the first i chunks. Edges are said to be *surviving* if both their endpoints are surviving vertices and they are part of a surviving matching. At any point in the algorithm, M_0 denotes the set of edges added so far to the rainbow matching (initially $M_0 = \varnothing$). The graph $G(i)$ denotes the graph with vertex set $V \setminus U(i)$ and edge set $\bigcup_{j>i} \mathcal{M}^j$ restricted to $V \setminus U(i)$. For all matchings $M \in \mathcal{M}$, we let $M(i)$ denote $M \cap E(G(i))$ and $\mathcal{M}^j(i) = \{M(i) \colon M \in \mathcal{M}^j\}$. Let $E(\mathcal{M}^j(i)) = \cup_{M \in \mathcal{M}^j} M(i)$, so that $E(\mathcal{M}^j(i))$

https://doi.org/10.1017/S0963548320000620 Published online by Cambridge University Press

is the set of edges in matchings in chunk j that still survive after iteration i. The above definitions are all intended to apply to the $i = 0$ case in the obvious way, with $U(0) = \varnothing$, $G(0) = G$, and so on.

After the initial stage, the algorithm performs iterations consisting of the three steps below. We consider the situation after $i \geqslant 0$ iterations have been completed, and describe how to perform the $(i+1)$th iteration. For simplicity, we describe certain edges and vertices being deleted from $G(i)$ as the algorithm progresses. More accurately, the algorithm takes a copy of $G(i)$ at the start of the $(i+1)$th iteration and edits this copy, which will end up becoming $G(i+1)$.

We assume that f is a given function (and will specify a particular one below).

(i) For each $M \in \mathcal{M}^{i+1}(i)$, choose one edge in M uniformly at random. Let $\Psi(i+1)$ denote the set of edges that are chosen. Vertices incident with edges in $\Psi(i+1)$ are called *marked*. For $x \in \Psi(i+1)$, if $x \cap y \neq \varnothing$ for some $y \in \Psi(i+1)$, we say there is a *vertex collision* involving x. For each $x \in \Psi(i+1)$ not involved in such a collision, add x into M_0 and delete the end vertices of x from $G(i)$; vertices deleted this way are called 'killed'.

(ii) Independently delete each existing vertex v in $G(i)$ with probability $P_{i+1}(v)$, where

$$Q_{i+1}(v) + P_{i+1}(v)(1 - Q_{i+1}(v)) = f(i\varepsilon),$$

and $Q_{i+1}(v)$ denotes the probability that v is marked in step (i). Vertices deleted this way are called 'zapped'. If $P_{i+1}(v) < 0$ or $P_{i+1}(v) > 1$ for some v then restart the algorithm.

(iii) Deal with vertex collisions greedily. Let $\Phi(i+1)$ denote the set of matchings in chunk $i+1$ that are not yet processed due to a vertex collision in step (i). Sequentially for each $M \in \Phi(i+1)$, choose a valid edge $x \in M$ using a greedy algorithm; for example, choose x incident with a vertex with the lowest index. Add x into the rainbow matching M_0 and delete the end vertices of x from the remaining graph. Unmark any vertices that were marked but not deleted.

The final iteration of the algorithm consists of treating the last chunk of matchings. Here edges are chosen greedily one by one from those matchings. A simple observation is as follows. If we choose an edge x that can validly be added to M_0, then the removal of the end vertices of x will decrease the size of each remaining matching by at most 2. Hence, when the algorithm comes to process the last chunk, if the sizes of the remaining matchings are all at least twice the number of matchings remaining, then a full rainbow matching will be successfully completed by the greedy method.

4.2 Proof of Theorem 2.1

Our proof involves many parameters. For ease of reference we summarize their definitions in Table 1, although each will be defined again at the appropriate time. Note that we are changing the definition of γ slightly from Section 3; from now on, $\gamma = 1 - n^{-c}$. Recall also that $d_v^j(i)$ denotes the number of edges of chunk j that are (still) incident with vertex v after iteration i.

By the hypotheses of Theorem 2.1, we may assume that $\mathcal{M}$ is non-intersecting and contains $m = \lfloor \gamma n^{1+\delta} \rfloor$ matchings each of size n, and $\Delta(\mathcal{M}) \leqslant \gamma n$. (This exact value of m is achieved by adding, if necessary, new matchings that are vertex-disjoint from all previous ones. This does not affect $\Delta(\mathcal{M})$ or the existence of a full rainbow matching.) By the hypotheses of the theorem we assume that $0 \leqslant \delta < 1/4$ and $0 < c < (1 - 4\delta)/10$. It follows that there exists a constant α in the range $2c + \delta < \alpha < (1 - \delta - 4c)/3$. We define $\varepsilon = \lceil n^{-\alpha} m \rceil / m \sim n^{-\alpha}$ and note that

$$n^{-1/3} < \varepsilon = o(1), \tag{4.1}$$

and that εm is an integer. Recall that $\tau = \lceil 1/\varepsilon \rceil$ is the number of iterations of the algorithm. It must be noted that our randomized algorithm only applies to the first $\tau - 1$ iterations.

https://doi.org/10.1017/S0963548320000620 Published online by Cambridge University Press

Table 1. Summary of parameters defined in terms of a sufficiently large constant C_0 and $\xi = \xi(n)$ which goes to zero sufficiently slowly.

$0 \leqslant \delta < \frac{1}{4}$
$0 < c < \frac{1-4\delta}{10}$
$2c + \delta < \alpha < \frac{1-\delta-4c}{3}$
$\gamma = 1 - n^{-c}$
$m = \lfloor \gamma n^{1+\delta} \rfloor \sim n^{1+\delta}$
$\varepsilon = \frac{\lceil n^{-\alpha} m \rceil}{m} \sim n^{-\alpha}$
$\tau = \lceil m/(\varepsilon m) \rceil = \lceil 1/\varepsilon \rceil$
$r(x) = (1-\gamma x)^2$ and $r_i = r(i\varepsilon)$
$g(x) = 1 - \gamma x$ and $g_i = g(i\varepsilon)$
$a_{i+1} = C_0\left(\frac{\varepsilon^2 g_i m}{r_i} + \sqrt{\varepsilon m}\log n\right) + 2c_i r_i n + a_i\left(1 - \frac{2\varepsilon\gamma g_i}{r_i}\right)$ for $i \geqslant 0$, with $a_0 = 0$
$b_{i+1} = (1 - f(i\varepsilon))b_i + C_0((\varepsilon g_i/r_i)^2 + \sqrt{\varepsilon m}\log n)$ for $i \geqslant 0$, with $b_0 = (\varepsilon\gamma n)^{1/2}\log n$
$c_i = \frac{\varepsilon\gamma a_i g_i n(1+2\xi)}{r_i^2 n^2} + \frac{b_i(1+2\xi)}{r_i n}$
$f(i\varepsilon) = \varepsilon\gamma\frac{g_i}{r_i} + c_i$
$A_i = C_0(\varepsilon^2 g_i m/r_i + \sqrt{\varepsilon m}\log n) + 4b_i,$
$B_i = 1 + \frac{4\xi\varepsilon\gamma g_i}{r_i}$

For simplicity, we let r_i and g_i denote $r(i\varepsilon)$ and $g(i\varepsilon)$, respectively, where $r(\cdot)$ and $g(\cdot)$ are given in (3.3). For $0 \leqslant i \leqslant \tau - 1$, we will specify non-negative real numbers a_i and b_i such that at the start of the $(i+1)$th iteration of the algorithm, the following hold with probability at least $1 - o((i+1)\varepsilon)$, where the function hidden in $o()$ is independent of i:

(A1) every surviving matching has size between $r_i n - a_i$ and $r_i n + a_i$, and
(A2) every surviving vertex v satisfies

$$d_v^j(i) \leqslant \varepsilon\gamma g_i n + b_i \quad \text{for all } j > i.$$

Values of the function f required in step (ii) of the $(i+1)$th iteration will be defined by

$$f(i\varepsilon) = \varepsilon\gamma\frac{g_i}{r_i} + c_i, \tag{4.2}$$

where $c_i \geqslant 0$ will also be specified.

The proof of this property of a_i and b_i is by induction on i. For the base case, $i = 0$, we regard the state at the start of the first iteration. The initial graph is $G = G(0)$ and we define $a_0 = 0$ and $b_0 = (\varepsilon\gamma n)^{1/2}\log n$. Then (A1) is trivially true. To verify (A2) holds for $i = 0$, we need to consider the variation in degrees caused by the initial random permutation of the matchings.

Lemma 4.1. *With probability* $1 - o(\varepsilon)$, *property* (A2) holds for $i = 0$.

https://doi.org/10.1017/S0963548320000620 Published online by Cambridge University Press

Proof. Here $d_v^j(0)$ is determined by the random permutation π of matchings in $\mathcal{M}$. Obviously $\mathbb{E}d_v^j(0) = \varepsilon d_v \leqslant \varepsilon\gamma n$. We will apply McDiarmid's inequality [23, Theorem 1.1] to prove concentration. Let λ denote the median of $d_v^j(0)$. Observe the following.

- Interchanging two elements in π can change $d_v^j(0)$ by at most 1, because all edges incident with v in $G(0)$ belong to different matchings.
- For every $s > 0$, if $d_v^j(0) \geqslant s$ then there is a set of s elements $\{i_1, \dots, i_s\} \subseteq [m]$ such that $\pi(i_1), \dots, \pi(i_s)$ certifies $d_v^j(0) \geqslant s$.

By McDiarmid's inequality, for any $t \geqslant 0$,

$$\mathbb{P}(|d_v^j(0) - \lambda| \geqslant t) \leqslant 4\exp\left(-\frac{t^2}{16(\lambda + t)}\right). \tag{4.3}$$

It follows immediately that

$$\begin{aligned}|\mathbb{E}d_v^j(0) - \lambda| &\leqslant \mathbb{E}|d_v^j(0) - \lambda| \\ &\leqslant \int_{t=0}^{\infty} 4\exp\left(-\frac{t^2}{16(\lambda + t)}\right)dt \\ &\leqslant \int_{t=0}^{\lambda} 4e^{-t^2/32\lambda}dt + \int_{t=\lambda}^{\infty} 4e^{-t/32}dt \\ &= O(\sqrt{\lambda} + 1).\end{aligned}$$

This implies that

$$\lambda = \mathbb{E}d_v^j(0) + O\left(1 + \sqrt{\mathbb{E}d_v^j(0)}\right).$$

Since $\mathbb{E}d_v^j(0) \leqslant \varepsilon\gamma n \to \infty$ as $n \to \infty$, we have $\lambda \leqslant \varepsilon\gamma n + O(\sqrt{\varepsilon\gamma n})$. Hence (4.3) with

$$t = \frac{1}{2}\sqrt{\varepsilon\gamma n}\log n = b_0/2$$

yields

$$\mathbb{P}(|d_v^j(0) - \lambda| \geqslant b_0/2) = \exp(-\Omega(\log^2 n)),$$

since $16(\lambda + t) = O(\varepsilon\gamma n) = O(t^2/\log^2 n)$. As $\lambda \leqslant \varepsilon\gamma n + o(b_0)$, this means

$$\mathbb{P}(d_v^j(0) \geqslant \varepsilon\gamma n + b_0) = o(n^{-6}).$$

Taking the union bound over the $O(mn) = O(n^{2+\delta})$ choices for v and $O(m) = O(n^{1+\delta})$ choices for j, we can conclude that with probability $1 - o(\varepsilon)$ we have $d_v^j(0) \leqslant \varepsilon\gamma n + b_0$ for every v and j, which is (A2) for $i = 0$. □

Next assume the claim holds for some $i \geqslant 0$. We can then condition on the event that (A1) and (A2) hold after the first i iterations of the algorithm, and just need to show that the probability they both hold in the next iteration is then $O(\varepsilon)$. Note that most edges in $\Psi(i+1)$ will have their endpoints killed in step (i), whereas some will survive due to vertex collision. We say a vertex is *condemned* if it is either zapped or marked. We desire each vertex to be condemned with probability $f(i\varepsilon)$ as specified in (4.2). This is made use of in step (ii). Of course, we require that $0 \leqslant P_{i+1}(v) \leqslant 1$, which is true if $Q_{i+1}(v) \leqslant f(i\varepsilon) \leqslant 1$. By (A1), after the ith iteration every surviving

https://doi.org/10.1017/S0963548320000620 Published online by Cambridge University Press

matching has size at least $r_i n - a_i$, which implies that the probability of a given edge being chosen is at most $1/(r_i n - a_i)$. From (A2), the degree of a vertex is at most $\varepsilon\gamma g_i n + b_i$, so we have

$$Q_{i+1}(v) \leqslant \frac{\varepsilon\gamma g_i n + b_i}{r_i n - a_i}.$$

Hence $Q_{i+1}(v) \leqslant f(i\varepsilon)$ would be guaranteed by

$$\frac{\varepsilon\gamma g_i n + b_i}{r_i n - a_i} \leqslant \varepsilon\gamma \frac{g_i}{r_i} + c_i. \tag{4.4}$$

We will appropriately define non-negative a_i, b_i and c_i with the following constraints:

$$a_i < r_i n/2, \quad b_i \leqslant \varepsilon\gamma g_i n, \quad c_i \leqslant \varepsilon\gamma g_i/r_i \leqslant 1/2, \tag{4.5}$$

Note that requiring $c_i \leqslant \varepsilon\gamma g_i/r_i \leqslant 1/2$ ensures that we satisfy $f(i\varepsilon) \leqslant 1$. So our definitions of these numbers just need to satisfy (4.4) and (4.5) to allow (A1) and (A2) to hold with i replaced by $i+1$.

In order to show that condition (A1) is satisfied after the $(i+1)$th iteration with sufficiently high probability, we need to estimate $|M(i+1)|$ for any $M \in \mathcal{M}^j(i)$, where $j > i+1$. First we bound the number of edges in M that have at least one end vertex condemned in steps (i) or (ii). We also call such edges *condemned*. Given $uv \in M(i)$, we know that the probability that u (or v) is condemned after step (ii) is $f(i\varepsilon)$. However, while the probability that a vertex is condemned in iteration $(i+1)$ is the same for all surviving vertices, vertices are not condemned independently. The following lemma shows that the probability that both u and v will be condemned is $O(f(i\varepsilon)^2)$. Note that the constants implicit in our $O(\cdot)$ notation are absolute. In the interest of continuity, we state the lemmas we need to prove Theorem 2.1 below and discuss some aspects of their proofs, but defer their proofs to Section 4.3.

Lemma 4.2. *If u and v are distinct vertices in $G(i)$, then the probability that both u and v are condemned in iteration $i+1$ is $O(f(i\varepsilon)^2)$.*

From this lemma, the probability that the edge uv is condemned is $2f(i\varepsilon) + O(f(i\varepsilon)^2)$. By linearity, the expected number of condemned edges in any given surviving matching $M(i) \in \mathcal{M}^j(i)$ in the $(i+1)$th iteration is $(2f(i\varepsilon) + O(f(i\varepsilon)^2))|M(i)|$. Next we address the effect of vertex collisions on the size of the surviving matchings. The following two lemmas bound the expected number of vertex collisions, and size of $\Phi(i+1)$, respectively.

Lemma 4.3. *Let X_u be the number of edges incident with u that are chosen in step (i), and let $Y_u = X_u I_{X_u \geqslant 2}$. With probability $1 - o(\varepsilon)$,*

$$Y_u = O\left(\max\left\{\frac{\varepsilon g_i}{r_i^2 n} d_u^{i+1}(i), \log^2 n\right\}\right).$$

Lemma 4.4. *With probability $1 - o(\varepsilon)$, we have $|\Phi(i)| = O(\varepsilon f(i\varepsilon)m + \sqrt{\varepsilon m}\log n)$.*

Thus the treatment of vertex collisions does not change the size of each matching obtained from step (ii) significantly. The number of edges that are condemned but do survive, or are not condemned in steps (i) and (ii) but are deleted in step (iii), is bounded by $O(|\Phi(i)|)$. It also follows from Lemma 4.4 that step (iii) will not usually fail, as the number of matchings to be treated in that step is usually of much smaller order than $r_i n$, the approximate size of each matching.

Using such considerations, we are able to show that with high probability, the size of each surviving matching is concentrated around its expectation.

https://doi.org/10.1017/S0963548320000620 Published online by Cambridge University Press

Lemma 4.5. *With probability* $1-o(\varepsilon)$, *for every* $M \in \mathcal{M}^j(i)$ *and* $j > i+1$, *we have*

$$|M(i+1)| = (1 - 2f(i\varepsilon) + O(f(i\varepsilon)^2))|M(i)| + O(\varepsilon f(i\varepsilon)m + \sqrt{\varepsilon m}\log n).$$

This provides us with enough information to specify a_{i+1} as required for (A1) after iteration $i+1$.

Next we consider (A2). This requires us to bound $d_v^j(i+1)$ for all $j > i+1$. Recall from Section 3 that E_v denotes the set of edges in G that are incident with v. Let $\mathcal{E} = E_v \cap E(\mathcal{M}^j(i))$. Since every vertex in $G(i)$ is condemned with probability $f(i\varepsilon)$, by linearity, the expected number of edges in $\mathcal{E}$ that are condemned is $f(i\varepsilon)d_v^j(i)$, if v survives after the ith iteration. Again, Lemma 4.4 ensures that the effect from vertex collision is small. This yields the following lemma.

Lemma 4.6. *With probability* $1-o(\varepsilon)$, *for every* $v \in G(i+1)$ *and* $j > i+1$,

$$d_v^j(i+1) \leqslant (1-f(i\varepsilon))d_v^j(i) + O\left(\frac{\varepsilon g_i}{r_i^2 n}\right)d_v^{i+1}(i) + O(\sqrt{\varepsilon m}\log n).$$

This lemma is strong enough for us to choose b_{i+1} appropriately.

We are now ready to complete the proof of Theorem 2.1. We first write the requirements for (the $i+1$ versions of) (A1), (A2) and (4.4) to be satisfied, using the inductive hypothesis, and then determine a_{i+1}, b_{i+1} and c_i so as to satisfy these requirements as well as (4.5). We have

$$|M(i+1)| - r_{i+1}n = T_1 + T_2 + T_3, \tag{4.6}$$

where

$$T_1 = |M(i+1)| - |M(i)| = -2f(i\varepsilon)|M(i)| + O(f(i\varepsilon)^2|M(i)| + \varepsilon f(i\varepsilon)m + \sqrt{\varepsilon m}\log n)$$

(by Lemma 4.5),

$$T_2 = |M(i)| - r_i n,$$

$$T_3 = (r_i - r_{i+1})n = 2\varepsilon\gamma n g_i + O(\varepsilon^2\gamma^2 n),$$

where the last equation holds since $r'(x) = -2\gamma g(x)$ and $r''(x) = 2\gamma^2$. Now

$$-2f(i\varepsilon)|M(i)| + 2\varepsilon\gamma n g_i = -2f(i\varepsilon)T_2 - 2c_i r_i n$$

and hence (4.6) gives

$$|M(i+1)| - r_{i+1}n = (1-2f(i\varepsilon))T_2 - 2c_i r_i n + O(f(i\varepsilon)^2|M(i)| + \varepsilon f(i\varepsilon)m + \sqrt{\varepsilon m}\log n + \varepsilon^2 n).$$

It is an easy observation that (A1) implies $|T_2| \leqslant a_i$, and this along with (4.5) yields

$$f(i\varepsilon)^2|M(i)| = O(\varepsilon^2 n g_i^2/r_i) = O(\varepsilon^2 n).$$

Indeed, $f(i\varepsilon) = O(\varepsilon g_i/r_i)$ and thus (A1) is satisfied after iteration $i+1$ provided that we define

$$a_{i+1} = C_0(\varepsilon^2 g_i m/r_i + \sqrt{\varepsilon m}\log n) + 2c_i r_i n + a_i\left(1 - \frac{2\varepsilon\gamma g_i}{r_i}\right), \tag{4.7}$$

where C_0 is a sufficiently large constant (subsequently to have a further condition imposed on it). Note that $\varepsilon^2 n$ is absorbed by $\varepsilon^2 g_i m/r_i$.

For (A2) we first rewrite

$$d_v^j(i+1) - \varepsilon\gamma g_{i+1}n = d_v^j(i+1) - \varepsilon\gamma g_i n(1-f(i\varepsilon)) + \varepsilon\gamma n(g_i - g_{i+1} - g_i f(i\varepsilon)).$$

By the definition of g, we have $g_i - g_{i+1} = \varepsilon\gamma$. Also, $g_i f(i\varepsilon) \geqslant \varepsilon\gamma g_i^2/r_i = \varepsilon\gamma$ as $c_i \geqslant 0$. Hence, using Lemma 4.6 to bound the value of $d_v^j(i+1)$ occurring in the right-hand side, we have (using $d_v^{i+1}(i) = O(\varepsilon g_i n)$ by (A2) and (4.5))

$$d_v^j(i+1) - \varepsilon\gamma g_{i+1}n \leqslant (1-f(i\varepsilon))(d_v^j(i) - \varepsilon\gamma g_i n) + O(\varepsilon^2 g_i^2/r_i^2 + \sqrt{\varepsilon m}\log n).$$

https://doi.org/10.1017/S0963548320000620 Published online by Cambridge University Press

Thus (A2) is satisfied after iteration $i+1$ provided that, for a sufficiently large constant C_0,

$$b_{i+1} = (1 - f(i\varepsilon))b_i + C_0(\varepsilon^2 g_i^2/r_i^2 + \sqrt{\varepsilon m}\log n). \tag{4.8}$$

We choose C_0 sufficiently large to satisfy the bounds on it implied in deriving both (4.7) and (4.8). As part of the induction we are going to ensure the following strengthening of the constraints on a_i and b_i in (4.5) (justified below):

$$a_i \leqslant \xi r_i n, \quad b_i \leqslant \xi \varepsilon g_i n \quad \text{for some fixed function } \xi = \xi(n) \to 0. \tag{4.9}$$

Then it follows that (4.4) is satisfied for n sufficiently large, provided we choose

$$c_i = \frac{\varepsilon\gamma a_i g_i n(1+2\xi)}{r_i^2 n^2} + \frac{b_i(1+2\xi)}{r_i n} \leqslant \frac{\varepsilon\gamma a_i g_i(1+2\xi)}{r_i^2 n} + \frac{2\xi\varepsilon\gamma g_i}{r_i}. \tag{4.10}$$

To complete the induction to the end of step $\tau - 1$, it only remains to check the growth rates of a_i, b_i and c_i and see that they satisfy (4.5) and (4.9) (for an appropriate ξ), which can be assumed for smaller values of i by induction.

Plugging (4.10) into (4.7) and using (4.9), we get

$$a_{i+1} \leqslant A_i + B_i a_i, \tag{4.11}$$

where

$$A_i = C_0(\varepsilon^2 g_i m/r_i + \sqrt{\varepsilon m}\log n) + 4b_i, \tag{4.12}$$

$$B_i = \frac{2\varepsilon\gamma g_i(1+2\xi)}{r_i} + 1 - \frac{2\varepsilon\gamma g_i}{r_i} = 1 + \frac{4\xi\varepsilon\gamma g_i}{r_i}.$$

If we turn inequality (4.11) into an equality, we obtain a recurrence whose solution, from initial condition $a_0 = 0$, is easily solved, and thus (since all coefficients are positive) implies

$$a_i \leqslant \sum_{j=0}^{i-1} A_j \prod_{k=j+1}^{i-1} B_k. \tag{4.13}$$

Recall that the number of iterations the algorithm takes is $\tau = \lceil 1/\varepsilon \rceil$. For any $i \leqslant \lceil 1/\varepsilon \rceil - 1$,

$$\begin{aligned}\prod_{k=j+1}^{i-1} B_k &\leqslant \exp\left((4\varepsilon\gamma\xi)\sum_{k=j+1}^{i-1}\frac{1}{1-\varepsilon\gamma k}\right)\\ &= \exp\left((4\xi + o(1))\int_{(\varepsilon\gamma)j}^{(\varepsilon\gamma)i}(1-x)^{-1}dx\right)\\ &\leqslant (1-\gamma)^{o(1)}\\ &= n^{o(1)}\end{aligned} \tag{4.14}$$

since $\gamma = 1 - n^{-c}$.

By iterating (4.8) (ignoring the negative term, which turns out to give no significant help), we find that

$$\begin{aligned}b_i &\leqslant b_0 + iC_0(\varepsilon^2 g_i^2/r_i^2 + \sqrt{\varepsilon m}\log n)\\ &\leqslant (1 + iC_0)\sqrt{\varepsilon m}\log n + iC_0\varepsilon^2/r_i,\end{aligned} \tag{4.15}$$

recalling that $b_0 = \sqrt{\varepsilon\gamma n}\log n$ and $m = \lfloor \gamma n^{1+\delta} \rfloor$. This easily establishes the bound on b_i in (4.9) given that $c < 1/10 < 1/3$ and $1/2 > (3\alpha + \delta)/2 + c$ because $(1 - \delta - 2c)/3 > (1 - \delta - 4c)/3 > \alpha$. Now we turn to a_i. Substituting (4.15) into (4.12) gives

$$A_j = O(\varepsilon^2 g_j m/r_j + j\sqrt{\varepsilon m}\log n + j\varepsilon^2/r_j). \tag{4.16}$$

https://doi.org/10.1017/S0963548320000620 Published online by Cambridge University Press

Using this and (4.14) in (4.13), and the bound $i \leqslant \lceil 1/\varepsilon \rceil - 1$ gives

$$a_i \leqslant n^{o(1)} \cdot O\Bigg(\sum_{j=0}^{i-1} \varepsilon^2 g_j m/r_j + \sum_{j=0}^{i-1} j\sqrt{\varepsilon m} \log n + \sum_{j=0}^{i-1} j\varepsilon^2/r_j \Bigg).$$

We can approximate $\sum_{j=0}^{i-1} \varepsilon^2 g_j m/r_j$ and $\sum_{j=0}^{i-1} j\varepsilon^2/r_j$ as follows:

$$\sum_{j=0}^{i-1} \varepsilon^2 g_j m/r_j = O\Bigg(\varepsilon m \int_0^{(\varepsilon\gamma)i} \frac{1}{1-x} dx \Bigg) = O(\varepsilon m \log(1/(1-\gamma))) = O(\varepsilon m \log n),$$

and

$$\sum_{j=0}^{i-1} j\varepsilon^2/r_j = O\Bigg(\int_0^{(\varepsilon\gamma)i} \frac{x}{(1-x)^2} dx \Bigg) = O(1/(1-\gamma)) = O(n^c).$$

It then follows that

$$a_i = n^{o(1)} \cdot O(\varepsilon m + \sqrt{m}\varepsilon^{-3/2} + n^c)$$

as the logarithmic factors are absorbed by $n^{o(1)}$. Since $\tau = \lceil 1/\varepsilon \rceil$, we have $\tau - 1 \leqslant 1/\varepsilon$, and thus

$$r_{\tau-1} \geqslant (1-\gamma)^2 = n^{-2c}. \tag{4.17}$$

Recalling that r is monotonically decreasing, $m \sim n^{1+\delta}$ and $\varepsilon \sim n^{-\alpha}$, the above estimate for a_i implies the bound for a_i required in (4.9), provided that

$$1 - 2c > \max\{-\alpha + 1 + \delta, (1 + \delta + 3\alpha)/2, c\}. \tag{4.18}$$

Our definitions for c and α ensure that (4.18) holds. As mentioned before, the first two bounds in (4.5) follow from (4.9). The upper bound on c_i in (4.5) follows immediately from its definition in (4.10), in view of (4.9). Also (4.18) implies that $2c < \alpha$ which, together with $\gamma g_i \leqslant 1$, means that $c_i \leqslant 1/2$ for large n, as required in (4.5).

We conclude that (A1) and (A2) are satisfied by induction for any given value of i, with probability $1 - o(\varepsilon)$, conditional upon their validity in the previous step. The inductive statement, that the failure probability is $1 - o((i+1)\varepsilon)$, follows immediately. Using $\tau = O(1/\varepsilon)$, with probability $1 - o(1)$) the algorithm successfully reaches the second-to-last iteration. Moreover, at the beginning of the last iteration, each surviving matching has size at least $r_{\tau-1}n - a_{\tau-1} \geqslant r_{\tau-1}n/2$ by (4.9).

Now we argue that with probability $1 - o(1)$ the algorithm finds a full rainbow matching in the last iteration. From (4.18) we know that $1 + \delta - \alpha < 1 - 2c$, so

$$2\varepsilon m \leqslant \frac{n^{1-2c}}{2} \leqslant \frac{r_{\tau-1}n}{2} \tag{4.19}$$

for large n, by (4.17). There are at most εm matchings remaining in the last iteration. So we can greedily choose one edge from each matching sequentially, since $2\varepsilon m \leqslant r_{\tau-1}n/2$, by (4.19). □

4.3 Proofs of lemmas

Proof of Lemma 4.2. Vertices u and v are both marked in step (i) if either

$uv \in E(\mathcal{M}^{i+1}(i))$ and uv is chosen; or

one edge in $E(\mathcal{M}^{i+1}(i)) \cap E_u$ is chosen and another edge in $E(\mathcal{M}^{i+1}(i)) \cap E_v$ is chosen.

https://doi.org/10.1017/S0963548320000620 Published online by Cambridge University Press

This probability is at most

$$\frac{1}{r_i n - a_i} + \frac{d_u^{i+1}(i)}{r_i n - a_i} \cdot \frac{d_v^{i+1}(i)}{r_i n - a_i} = O\left(\varepsilon \frac{g_i}{r_i}\right)^2 = O(f(i\varepsilon)^2) \tag{4.20}$$

because of (4.5) and (4.1), which imply that $1/(r_i n - a_i) = O(\varepsilon^2 g_i^2 / r_i^2)$. Vertices u and v are both condemned (marked or zapped) after step (ii) if and only if

they are both marked in step (i); or

one is condemned, and the other is zapped in step (ii).

We have shown that the probability of the first case is $O(f(i\varepsilon)^2)$. The probability of the second case is at most

$$f(i\varepsilon)P_{i+1}(v) + f(i\varepsilon)P_{i+1}(u) = O(f(i\varepsilon)^2).$$

This is because the probability of condemning u is at most $f(i\varepsilon)$, and conditional on u being condemned and v not being killed (with probability at most 1), the probability that v is zapped is at most $P_{i+1}(v)$, as vertices are zapped independently in step (ii). The lemma follows. □

Proof of Lemma 4.3. Recall that X_u denotes the number of edges incident with u that are chosen in step (i), and $Y_u = X_u I_{X_u \geqslant 2}$. Immediately we have $X_u - 1 \leqslant Y_u \leqslant X_u$. Note that

$$Y_u \leqslant X_u(X_u - 1) = \sum_{x,y \in E_u \cap E(\mathcal{M}^{i+1}(i))} I_x I_y,$$

where I_x is the indicator variable that x is chosen, and the summation is over all ordered pairs (x, y). For each $u \in G(i)$, u is incident with $d_u^{i+1}(i) \leqslant \varepsilon\gamma g_i n + b_i$ edges in $E(\mathcal{M}^{i+1}(i))$. Note that all edges in $E_u \cap E(\mathcal{M}^{i+1}(i))$ must belong to different matchings and therefore $\{I_x \colon x \in E_u \cap E(\mathcal{M}^{i+1}(i))\}$ are independent variables. Thus the probability that any given x and y are both chosen is at most

$$(r_i n - a_i)^{-2}.$$

Hence

$$\mathbb{E}Y_u \leqslant (d_u^{i+1}(i))^2 (r_i n - a_i)^{-2} = O\left(\frac{\varepsilon g_i}{r_i^2 n}\right) d_u^{i+1}(i).$$

It follows immediately that

$$\mathbb{E}X_u \leqslant 1 + O\left(\frac{\varepsilon g_i}{r_i^2 n}\right) d_u^{i+1}(i),$$

as $X_u \leqslant 1 + Y_u$. Note that $X_u = \sum_{x \in E_u \cap E(\mathcal{M}^{i+1}(i))} I_x$, which is the sum of independent indicator variables. Applying Chernoff's bound, we obtain that with probability $1 - o(\varepsilon)$,

$$Y_u \leqslant X_u \leqslant \max\{2\mathbb{E}X_u, \log^2 n\} = O\left(\max\left\{\frac{\varepsilon g_i}{r_i^2 n} d_u^{i+1}(i), \log^2 n\right\}\right) \quad \text{for all } u \in G(i).$$

The lemma follows. □

Proof of Lemma 4.4. Let $Y = \sum_{u \in G(i)} Y_u$, where Y_u is defined as in Lemma 4.3. Then $|\Phi(i+1)| \leqslant Y$. Thus it immediately follows that

$$\mathbb{E}|\Phi(i+1)| \leqslant \mathbb{E}Y = \sum_u \mathbb{E}Y_u = O\left(\frac{\varepsilon g_i}{r_i^2 n}\right) \sum_u d_u^{i+1}(i).$$

https://doi.org/10.1017/S0963548320000620 Published online by Cambridge University Press

By (A1), $\sum_u d_u^{i+1}(i) \leqslant 2(r_i n + a_i) \cdot \varepsilon m$. Thus

$$\mathbb{E}|\Phi(i+1)| = O\left(\frac{\varepsilon^2 m g_i}{r_i}\right) = O(\varepsilon f(i\varepsilon)m).$$

Apply Azuma's inequality to $|\Phi(i+1)|$. Changing the choice x to another edge y in a matching M would affect $|\Phi(i+1)|$ by at most 3. To see this, let $x = uv$ and $y = u'v'$. A matching M' that was in $\Phi(i+1)$ could be removed after changing x to y, if $z \in M'$ was chosen, and z is the only chosen edge, besides x, that is incident with u (or v). There can only be at most two such matchings. So changing x to y would decrease $|\Phi(i+1)|$ by at most 3, counting M itself. Similarly, changing x to y would increase $|\Phi(i+1)|$ by at most 3. Thus, by Azuma's inequality with Lipschitz constant 3, we have that with probability $1 - o(\varepsilon)$,

$$|\Phi(i+1)| = \mathbb{E}|\Phi(i+1)| + O(\sqrt{\varepsilon m}\log n) = O(\varepsilon f(i\varepsilon)m + \sqrt{\varepsilon m}\log n). \qquad \square$$

We will use the following Azuma–Hoeffding inequality to prove concentration of various variables.

Theorem 4.1 ([8, 18]). *Let $X_0, X_1, \ldots$ be a martingale satisfying $|X_i - X_{i-1}| \leqslant \delta_i$ for every $i \geqslant 1$. Then, for every $t \geqslant 0$,*

$$\mathbb{P}(|X_n - X_0| \geqslant t) \leqslant 2\exp\left(-t^2 \Big/ 2\sum_{i=1}^{n} \delta_i^2\right).$$

Proof of Lemma 4.5. We have argued that

$$\mathbb{E}(|M(i+1)| \mid G(i)) = (1 - 2f(i\varepsilon) + O(f(i\varepsilon)^2))|M(i)| + O(\varepsilon f(i\varepsilon)m + \sqrt{\varepsilon m}\log n), \tag{4.21}$$

where the main term comes from considering the edges that are condemned, and the error term accounts for a correction term due to vertex collision, by Lemma 4.4.

For concentration, first consider X, the number of edges chosen in $\Psi(i+1)$ in $M \in \mathcal{M}^j(i)$ in step (i) of the algorithm as described in Section 4.1. Let $Y_0, \ldots, Y_{\varepsilon m}$ be the Doob martingale constructed from X under the edge exposure process where edges in $\Psi(i+1)$ are revealed sequentially; that is, Y_i is the expected value of X conditional upon the first i edges that are exposed, and in particular $Y_0 = \mathbb{E}X$ and $Y_m = X$. (See Alon and Spencer [7, Chapter 7] for an explanation of this method.) Apply Theorem 4.1 to the martingale (Y_i). It is easy to see that changing a single edge $x \in \Psi(i+1)$ to another edge y would change X by at most two. Thus the probability that X deviates from $\mathbb{E}X$ by more than $t = \sqrt{\varepsilon m}\log n$ is at most $2\exp(-t^2/8\varepsilon m) = o(n^{-2})$. Taking the union bound over all $M \in \mathcal{M}^j(i)$, we obtain the desired deviation in the lemma with probability at least $1 - o(\varepsilon)$. Next we consider the number of edges zapped in M in step (ii). Condition on the set of edges that survive step (i). Each surviving vertex u is zapped independently with probability $P_{i+1}(v)$. For each $M \in \mathcal{M}^j(i)$ consisting of edges surviving after step (i), the $2|M|$ vertices incident with M are independently zapped with probabilities all bounded by $f(i\varepsilon)$. Let Y denote the number of vertices zapped. Then Y is the sum of at most $2n$ independent Bernoulli variables. By the Chernoff–Hoeffding bound [14, Theorem 1.1], the probability that Y deviates from its expectation by more than $\sqrt{f(i\varepsilon)n}\log n$ is at most n^{-2}. Taking the union bound over all M, again with probability at least $1 - o(\varepsilon)$ we have the desired deviation as in the lemma. Finally, the change to $|M(i)|$ due to step (iii) is absorbed by the error term in (4.21) by Lemma (4.4). $\square$

Proof of Lemma 4.6. The argument is so similar to the proof of Lemma 4.5 that we omit details. We can apply Theorem 4.1 to prove concentration for the number of neighbours of v (in chunk j) condemned in step (i) and then for the number of neighbours zapped in step (ii) using the Chernoff–Hoeffding bound. By Lemma 4.3, vertex collision will affect $d_v^j(i+1)$ by $O(\varepsilon\gamma g_i/r_i^2 n)d_v^{i+1}(i)$. The treatment of vertex collisions in step (iii) can only decrease (the bound on) $d_v^j(i+1)$. The lemma follows. $\square$

https://doi.org/10.1017/S0963548320000620 Published online by Cambridge University Press

5. Multigraphs

The proof of Theorem 2.3 follows almost exactly that of Theorem 2.1 with $\delta = 0$. We run the same randomized algorithm with the same parameters g_i and r_i, but with different a_i, b_i and c_i. The reason is that due to the multiplicities of the multiple edges, variables $|M(i)|$ and $d_v^j(i)$ are not as concentrated as in the simple graph case and thus we expect larger a_i, b_i and c_i here. We briefly sketch the proof. Now we assume that $m = \lfloor \gamma n \rfloor$, where $\gamma = 1 - \varepsilon_0$ and $\varepsilon_0 > 0$ is an arbitrarily small constant, and $\varepsilon > 0$ is going to be a constant that depends on ε_0. Let μ denote the maximum multiplicity of the multiple edges in G. Note also that here $m = \Theta(n)$.

Lemma 4.1 holds in the multigraph case with the same b_0. For Lemma 4.2, the probability that an edge between u and v is chosen is bounded by $\mu/(r_i n - a_i)$. Thus, for (4.20) to hold, we require

$$\frac{\mu}{r_i n - a_i} = O\left(\frac{\varepsilon^2 (g_i m)^2}{(r_i n)^2}\right),$$

which is guaranteed if we assume

$$\varepsilon^2 \geqslant \mu/n. \tag{5.1}$$

Thus Lemma 4.2 holds after replacing the condition $\varepsilon^2 \geqslant 1/n$ by (5.1).

Lemmas 4.3, 4.4 and 4.5 hold as they are. For Lemma 4.6, note that deleting a single vertex (in both steps (i) and (ii)) can alter $d_v^j(i+1)$ by μ. Therefore the Lipschitz constant becomes μ. Applying Azuma's inequality to both steps (i) and (ii), we deduce Lemma 4.6, where $O(\sqrt{\varepsilon m}\log n)$ is replaced by $O(n/\log n)$, if $\mu = O(\sqrt{n}/\log^2 n)$.

These lead to recursions for a_i as in (4.11), and for b_i as

$$b_{i+1} = (1 - f(i\varepsilon))b_i + O(n/\log n).$$

Immediately we have $b_i = O(in/\log n)$. Substituting into (4.11), we have

$$a_{i+1} \leqslant \left(1 + \frac{4\xi\varepsilon\gamma g_i}{r_i}\right)a_i + O(\varepsilon^2 n g_i/r_i + in/\log n).$$

Solving the recursion as before, we get

$$a_i = \varepsilon_0^{o(1)} \cdot O\left(\sum_{j=0}^{i-1} \varepsilon^2 n g_j/r_j + \sum_{j=0}^{i-1} jn/\log n\right)$$

$$= \varepsilon_0^{o(1)} \cdot O(\varepsilon n + n/\varepsilon \log n).$$

Hence there exists a constant $C > 0$ such that $a_i \leqslant C(\varepsilon n + n/\varepsilon \log n)$ for all $1 \leqslant i \leqslant 1/\varepsilon$. Let $i_1 = \lceil 1/\varepsilon \rceil - 1$. Then $r_{i_1} \geqslant (1-\gamma)^2 = \varepsilon_0^2$. Choose $\varepsilon > 0$ sufficiently small such that

$$C_0(\varepsilon n + n/\varepsilon \log n) \leqslant \frac{\varepsilon_0^2}{4} n.$$

Then, with the same argument as before, a greedy search in the last iteration of the randomized algorithm succeeds in finding a full rainbow matching in $\mathcal{M}$ with high probability.

6. Hypergraphs

The proof of Theorem 2.4 is again similar. Let G be a k-uniform hypergraph. We give a quick sketch here and just point out the differences. The randomized algorithm extends to hypergraphs in a natural way. Thus every vertex is deleted in the $(i+1)$th iteration with probability

$$f(i\varepsilon) \approx \frac{\varepsilon\gamma g(i\varepsilon)}{r(i\varepsilon)},$$

https://doi.org/10.1017/S0963548320000620 Published online by Cambridge University Press

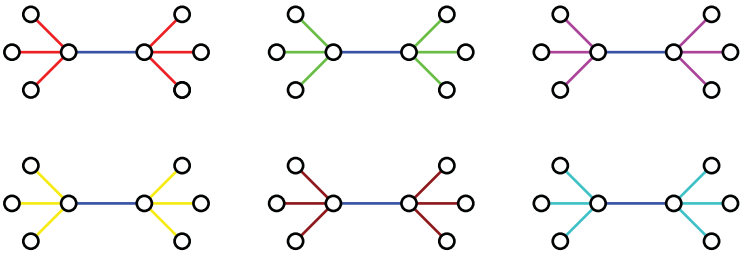


Figure 1. The graph $\mathcal{G}_6$.

where $\gamma = 1 - \varepsilon_0$. Now every hyperedge in a matching is deleted with probability approximately $kf(i\varepsilon)$, as there are k vertices in a hyperedge, For each surviving vertex v, each incident hyperedge is deleted with probability approximately $(k-1)f(i\varepsilon)$, as this hyperedge is deleted if one of the other $k-1$ vertices contained in it is deleted. Hence we find that $r(x)$ and $g(x)$ obey the following differential equations:

$$r' = -k\gamma g(x), \quad g'(x) = -(k-1)\gamma \frac{g(x)^2}{r(x)},$$

with initial conditions $r(0) = 1$ and $g(0) = 1$. The solution to these differential equations is

$$r(x) = (1-\gamma x)^k, \quad g(x) = (1-\gamma x)^{k-1}.$$

The proof that $|M(i)|$ and $d_v^j(i)$ are concentrated around $r_i n$ and $\varepsilon\gamma g_i n$ follows in the same manner as in Theorem 2.1. Lemmas 4.2 and 4.6 need to be modified as in Theorem 2.3. Here, the effect of codegrees, *i.e.* the maximum number of hyperedges containing a pair of vertices, plays the same role of affecting the Lipschitz constants as the maximum multiplicity in Theorem 2.3. This yields Theorem 2.4. □

7. Counterexamples to some conjectures on rainbow matchings

In this section we describe counterexamples to Conjectures 2.5 and 2.9 in [1], as well as Conjectures 5.3, 5.4, 6.1 and 6.2 in [2] (which were removed from later versions of that paper, after the appearance of the counterexamples in our preprint [15]). Before doing so, we need to describe how rainbow matchings in graphs can be viewed as matchings in 3-uniform hypergraphs. Suppose that G is a (not necessarily properly) edge-coloured graph in which we are interested in finding a rainbow matching. We make a 3-uniform hypergraph H from G as follows. The vertices of H are $V(G) \cup V_1$, where V_1 is the set of colours used on edges of G. For each edge $\{u, v\}$ of G with colour $c \in V_1$, there is a hyperedge $\{u, v, c\}$ in H. Now a full rainbow matching in G corresponds to a matching of H that covers all of the vertices in V_1. If G happens to be bipartite with bipartition $V_2 \cup V_3$, then H will be *tripartite*, because its vertices can be partitioned as $V_1 \cup V_2 \cup V_3$ such that every hyperedge includes one vertex from each of these three sets.

Let m be a positive even integer. We now construct a bipartite graph $\mathcal{G}_m$ whose edges are (not properly) coloured using m colours in such a way that there is no full rainbow matching. There are m components in $\mathcal{G}_m$, each isomorphic to a double star which has two adjacent central vertices, each of which has $m/2$ leaves attached to it. The edge between the central vertices in each double star is coloured blue. In each component, the edges connected to leaves all have one colour (not blue), which is specific to that component. Hence there are $m+1$ colours overall, and each colour appears on m edges. Figure 1 shows $\mathcal{G}_6$. There is no full rainbow matching in $\mathcal{G}_m$ because such a matching must include a blue edge from some double star S. However, the colour of the other edges in S then cannot be represented in the matching.

https://doi.org/10.1017/S0963548320000620 Published online by Cambridge University Press

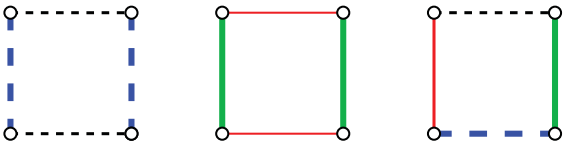


Figure 2. A 2-regular graph with no rainbow matching.

Let $\mathcal{H}_m$ be the tripartite hypergraph corresponding to $\mathcal{G}_m$. Let V_1 be the vertices of $\mathcal{H}$ corresponding to the colours, and let V_2, V_3 be the sets of vertices corresponding to a bipartition of $\mathcal{G}$. Then every vertex in V_1 has degree m. The vertices in $V_2 \cup V_3$ all have degree either 1 or $m/2+1$. Thus the (minimum) degree $\delta(V_1)$ of a vertex in V_1 is nearly double the maximum degree $\Delta(V_2 \cup V_3)$ of the vertices outside V_1. Interestingly, Aharoni and Berger [1, Theorem 2.6] showed that in any tripartite hypergraph, if $\delta(V_1) \geqslant 2\Delta(V_2 \cup V_3)$ then there must be a $|V_1|$-matching. Our hypergraph $\mathcal{H}_m$ shows that their theorem is close to tight. However, they made the following conjecture [1, Conjecture 2.5] (repeated as [3, Conjecture 5.3], and rephrased as Conjecture 1.3 in our introduction).

Conjecture 7.1. *Let H be a hypergraph with a vertex tripartition $V(H) = V_1 \cup V_2 \cup V_3$ such that every hyperedge includes exactly one vertex from V_i for $i = 1, 2, 3$. If $\delta(V_1) > \Delta(V_2 \cup V_3)$, then H has a $|V_1|$-matching.*

Note that $\mathcal{H}_m$ disproves Conjecture 7.1 whenever $m \geqslant 4$. Another counterexample to Conjecture 7.1 is based on the graph in Figure 2, which has no rainbow matching. The corresponding tripartite hypergraph has $\delta(V_1) = 3 > 2 = \Delta(V_2 \cup V_3)$. The line graph of the graph in Figure 2 was published in [6] and its complement was published in [19]. In both cases the focus of the investigation was slightly different from ours, so the generalizations that were offered are not relevant for us.

Conjecture 2.9 of [1] generalizes Conjecture 7.1, so it too is false. Similarly, [3, Conjecture 6.1] asserts that if $\delta(V_1) \geqslant 2 + \Delta(V_2 \cup V_3)$ then there must be a $|V_1|$-matching, so $\mathcal{H}_m$ is a counterexample whenever $m \geqslant 6$.

Finally, we consider Conjectures 5.4 and 6.2 from [3]. These deal with the case when the initial graph is not necessarily bipartite, so the resulting hypergraph is not necessarily tripartite. Nevertheless they consider full rainbow matchings in an edge-coloured graph; or equivalently, $|V_1|$-matchings in a 3-uniform hypergraph H in which every hyperedge includes exactly one vertex in the set V_1. The conjectures assert that such a matching will exist provided that $\delta(V_1) \geqslant 2 + \Delta(V(H) \setminus V_1)$. Again $\mathcal{H}_m$ provides a counterexample. Indeed, it shows that the 2 cannot be replaced by any constant.

References

[1] Aharoni, R. and Berger, E. (2009) Rainbow matchings in r-partite r-graphs. *Electron. J. Combin.* **16** #R119.

[2] Aharoni, R., Berger, E., Chudnovsky, M., Howard, D. and Seymour, P. (2016) Large rainbow matchings in general graphs. arXiv:1611.03648v3

[3] Aharoni, R., Berger, E., Chudnovsky, M., Howard, D. and Seymour, P. (2019) Large rainbow matchings in general graphs. *European J. Combin.* **79** 222–227.

[4] Aharoni, R., Charbit, P. and Howard, D. (2015) On a generalization of the Ryser–Brualdi–Stein conjecture. *J. Graph Theory* **78** 143–156.

[5] Aharoni, R., Kotlar, D. and Ziv, R. (2017) Representation of large matchings in bipartite graphs. *SIAM J. Discrete Math.* **31** 1726–1731.

[6] Alon, N. (1992) The strong chromatic number of a graph. *Random Struct. Algorithms* **3** 1–7.

[7] Alon, N. and Spencer, J. (2000) *The Probabilistic Method*, Wiley.

[8] Azuma, K. (1967) Weighted sums of certain dependent random variables. *Tohoku Math. J.* (Second Series) **19** 357–367.

https://doi.org/10.1017/S0963548320000620 Published online by Cambridge University Press

[9] Barát, J., Gyárfás, A. and Sárközy, G. N. (2017) Rainbow matchings in bipartite multigraphs. *Period. Math. Hungar.* **74** 108–111.
[10] Barát, J. and Wanless, I. M. (2014) Rainbow matchings and transversals. *Australas. J. Combin.* **59** 211–217.
[11] Best, D., Hendrey, K., Wanless, I. M., Wilson, T. E. and Wood, D. R. (2018) Transversals in Latin arrays with many distinct symbols. *J. Combin. Des.* **26** 84–96.
[12] Cavenagh, N. J. and Wanless, I. M. (2017) Latin squares with no transversals. *Electron. J. Combin.* **24** #P2.45.
[13] Clemens, D. and Ehrenmüller, J. (2016) An improved bound on the sizes of matchings guaranteeing a rainbow matching. *Electron. J. Combin.* **23** #P2.11.
[14] Dubhashi, D. P. and Panconesi, A. (2009) *Concentration of Measure for the Analysis of Randomized Algorithms*, Cambridge University Press.
[15] Gao, P., Ramadurai, R., Wanless, I. M. and Wormald, N. (2017) Counterexamples on matchings in hypergraphs and full rainbow matchings in graphs. arXiv:1710.04807
[16] Häggkvist, R. and Johansson, A. (2008) Orthogonal Latin rectangles. *Combin. Probab. Comput.* **17** 519–536.
[17] Hatami, P. and Shor, P. W. (2008) A lower bound for the length of a partial transversal in a Latin square. *J. Combin. Theory Ser. A* **115** 1103–1113.
[18] Hoeffding, W. (1963) Probability inequalities for sums of bounded random variables. *J. Amer. Statist. Assoc.* **58** (301) 13–30.
[19] Jin, G. (1992) Complete subgraphs of r-partite graphs. *Combin. Probab. Comput.* **1** 241–250.
[20] Keevash, P. and Yepremyan, L. (2018) Rainbow matchings in properly colored multigraphs. *SIAM J. Discrete Math.* **32** 1577–1584.
[21] Keevash, P. and Yepremyan, L. (2020) On the number of symbols that forces a transversal. *Combin. Probab. Comput.* **29** 234–240.
[22] Kotlar, D. and Ziv, R. (2014) Large matchings in bipartite graphs have a rainbow matching. *European J. Combin.* **38** 97–101.
[23] McDiarmid, C. (2002) Concentration for independent permutations. *Combin. Probab. Comput.* **11** 163–178.
[24] Montgomery, R., Pokrovskiy, A. and Sudakov, B. (2019) Decompositions into spanning rainbow structures. *Proc. Lond. Math. Soc.* (3) **119** 899–959.
[25] Pokrovskiy, A. (2017) Rainbow matchings and rainbow connectedness. *Electron. J. Combin.* **24** #P1.13.
[26] Pokrovskiy, A. (2018) An approximate version of a conjecture of Aharoni and Berger. *Adv. Math.* **333** 1197–1241.
[27] Ryser, H. (1967) Neuere Probleme der Kombinatorik. In *Vorträge über Kombinatorik, Oberwolfach*, pp. 69–91.
[28] Stein, S. K. (1975) Transversals of Latin squares and their generalizations. *Pacific J. Math.* **59** 567–575.
[29] Wanless, I. M. (2011) Transversals in Latin squares: a survey. In *Surveys in Combinatorics 2011* (R. Chapman, ed.), Vol. 392 of London Mathematical Society Lecture Note Series, pp. 403–437. Cambridge University Press.

Cite this article: Gao P, Ramadurai R, Wanless IM and Wormald N (2021). Full rainbow matchings in graphs and hypergraphs. *Combinatorics, Probability and Computing* **30**, 762–780. https://doi.org/10.1017/S0963548320000620

https://doi.org/10.1017/S0963548320000620 Published online by Cambridge University Press