Combinatorics, Probability and Computing (2023), **32**, pp. 225–268
doi:10.1017/S0963548322000219

CAMBRIDGE UNIVERSITY PRESS

ARTICLE

Towards the 0-statement of the Kohayakawa-Kreuter conjecture

Joseph Hyde*

University of Victoria, Victoria, BC V8P 5C2, Canada
Email: josephhyde@uvic.ca

(Received 2 August 2021; revised 19 July 2022; accepted 22 August 2022; first published online 27 September 2022)

Abstract

In this paper, we study asymmetric Ramsey properties of the random graph $G_{n,p}$. Let $r \in \mathbb{N}$ and $H_1, \dots, H_r$ be graphs. We write $G_{n,p} \to (H_1, \dots, H_r)$ to denote the property that whenever we colour the edges of $G_{n,p}$ with colours from the set $[r] := \{1, \dots, r\}$ there exists $i \in [r]$ and a copy of H_i in $G_{n,p}$ monochromatic in colour i. There has been much interest in determining the asymptotic threshold function for this property. In several papers, Rödl and Ruciński determined a threshold function for the general symmetric case; that is, when $H_1 = \cdots = H_r$. A conjecture of Kohayakawa and Kreuter from 1997, if true, would fully resolve the asymmetric problem. Recently, the 1-statement of this conjecture was confirmed by Mousset, Nenadov and Samotij.

Building on work of Marciniszyn, Skokan, Spöhel and Steger from 2009, we reduce the 0-statement of Kohayakawa and Kreuter's conjecture to a certain deterministic subproblem. To demonstrate the potential of this approach, we show this subproblem can be resolved for almost all pairs of regular graphs. This therefore resolves the 0-statement for all such pairs of graphs.

Keywords: Ramsey theory; random graphs; asymmetric Ramsey properties; Kohayakawa-Kreuter conjecture

2020 MSC Codes: Primary: 05C55, 05C80; Secondary: 05C85

1. Introduction

Ramsey theory is one of the most studied areas in modern combinatorics. Let $r \in \mathbb{N}$ and let $G, H_1, \dots, H_r$ be graphs. We write $G \to (H_1, \dots, H_r)$ to denote the property that whenever we colour the edges of G with colours from the set $[r] := \{1, \dots, r\}$ there exists $i \in [r]$ and a copy of H_i in G monochromatic in colour i. Thus, in this notation, the classical result of Ramsey [18] asserts that for n sufficiently large $K_n \to (H_1, \dots, H_r)$. One may posit that this is only true because K_n is very dense, but Folkman [2], and in a more general setting Nešetřil and Rödl [17], proved that for any graph H there are locally sparse graphs $G = G(H)$ such that $G \to (H_1, \dots, H_r)$ when $H_1 = \cdots = H_r = H$.

1.1 Symmetric Ramsey properties: Rödl and Ruciński's theorem

If we transfer our study of the Ramsey property to the random setting, we discover that such graphs G are in fact very common. Let $G_{n,p}$ be the binomial random graph with n vertices and edge probability p. Improving on earlier work of Frankl and Rödl [3], Łuczak, Ruciński and Voigt [13] proved that $p = n^{-1/2}$ is a threshold for the property $G_{n,p} \to (K_3, K_3)$. Following this, Rödl

*J.H. was supported by UKRI Future Leaders Fellowship grant MR/S016325/1 and ERC Advanced Grant 101020255.

and Ruciński [19–21] determined a threshold for the general symmetric case. For a graph H, we define

$$d_2(H) := \begin{cases} (e_H - 1)/(v_H - 2) & \text{if } H \text{ is non-empty with } v(H) \geq 3, \\ 1/2 & \text{if } H \cong K_2, \\ 0 & \text{otherwise} \end{cases}$$

and the 2-*density of* H to be

$$m_2(H) := \max\{d_2(J) : J \subseteq H\}.$$

We say that a graph H is 2-*balanced* if $d_2(H) = m_2(H)$, and *strictly* 2-*balanced* if for all proper subgraphs $J \subsetneq H$, we have $d_2(J) < m_2(H)$.

Theorem 1.1. (Rödl and Ruciński [21]) *Let $r \geq 2$ and let H be a non-empty graph such that at least one component of H is not a star. If $r = 2$, then in addition restrict H to having no component which is a path on 3 edges. Then there exist positive constants $b, B > 0$ such that*

$$\lim_{n\to\infty} \mathbb{P}[G_{n,p} \to (\underbrace{H, \ldots, H}_{r \text{ times}})] = \begin{cases} 0 & \text{if } p \leq bn^{-1/m_2(H)}, \\ 1 & \text{if } p \geq Bn^{-1/m_2(H)}. \end{cases}$$

The statement for $p \leq bn^{-1/m_2(H)}$ in Theorem 1.1 is known as a 0-*statement* and the statement for $p \geq Bn^{-1/m_2(H)}$ is known as a 1-*statement*.

The assumption on the structure of H in Theorem 1.1 is necessary. If every component of H is a star then $G_{n,p} \to (H, \ldots, H)$ as soon as sufficiently many vertices of degree $r(\Delta(H) - 1) + 1$ appear in $G_{n,p}$. A threshold for this property in $G_{n,p}$ is $p = n^{-1-1/(r(\Delta(H)-1)+1)}$, but $m_2(H) = 1$. For the case when $r = 2$ and at least one component of H is a path on 3 edges while the others are stars, the 0-statement of Theorem 1.1 becomes false. Indeed, one can show that, if $p = cn^{-1/m_2(P_3)} = cn^{-1}$ for some $c > 0$, then the probability that $G_{n,p}$ contains a cycle of length 5 with an edge pending at every vertex is bounded from below by a positive constant $d = d(c)$. One can check that every colouring of the edges of this augmented 5-cycle with 2 colours yields a monochromatic path of length 3. This special case was missed in [21], and was eventually observed by Friedgut and Krivelevich [4], who corrected the 0-statement to have the assumption $p = o(n^{-1/m_2(H)})$ instead. Note that Nenadov and Steger [16] produced a short proof of Theorem 1.1 using the hypergraph container method.

The intuition behind the threshold in Theorem 1.1 is as follows: Firstly, assume H is 2-balanced. The expected number of copies of a graph H in $G_{n,p}$ is $\Theta(n^{v(H)}p^{e(H)})$ and the expected number of edges is $\Theta(n^2 p)$. For $p = n^{-1/m_2(H)}$ (the threshold in Theorem 1.1), these two expectations are of the same order since H is 2-balanced. That is to say, if the expected number of copies of H at a fixed edge is smaller than some small constant c, then we can hope to colour without creating a monochromatic copy of H: very roughly speaking, each copy will likely contain an edge not belonging to any other copy of H, so by colouring these edges with one colour and all other edges with a different colour we avoid creating monochromatic copies of H. If the expected number of copies of H at a fixed edge is larger than some large constant C then a monochromatic copy of H may appear in any r-colouring since the copies of H most likely overlap heavily.

Sharp thresholds for $G_{n,p} \to (H, H)$ have also been obtained when H is a tree [4], a triangle [5] or, more generally, a strictly 2-balanced graph that can be made bipartite by the removal of some edge [22].

Theorem 1.2. ([4, 5, 22]) *Suppose that H is either (i) a tree that is not a star or the path of length three or (ii) a strictly 2-balanced graph with $e(H) \geq 2$ that can be made bipartite*

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

by the removal of some edge. Then there exist constants c_0, c_1, *and a function* $c:\mathbb{N}\to[c_0,c_1]$, *such that*

$$\lim_{n\to\infty}\mathbb{P}[G_{n,p}\to(H,H)]=\begin{cases}0 & \text{if } p\le(1-\varepsilon)c(n)n^{-1/m_2(H)},\\ 1 & \text{if } p\ge(1+\varepsilon)c(n)n^{-1/m_2(H)}\end{cases}$$

for every positive constant ε.

1.2 Asymmetric Ramsey properties: The Kohayakawa-Kreuter Conjecture

In this paper, we are interested in asymmetric Ramsey properties of $G_{n,p}$, that is, finding a threshold for when $G_{n,p}\to(H_1\ldots,H_r)$ and $H_1,\ldots,H_r$ are not all the same graph. In classical Ramsey theory, the study of asymmetric Ramsey properties sparked off many interesting routes of research (see, e.g. [1]), including the seminal work of Kim [9] on establishing an asymptotically sharp lower bound on the Ramsey number $R(3,t)$. In $G_{n,p}$, asymmetric Ramsey properties were first considered by Kohayakawa and Kreuter [10]. For graphs H_1 and H_2 with $m_2(H_1)\ge m_2(H_2)$, we define

$$d_2(H_1,H_2):=\begin{cases}\dfrac{e(H_1)}{v(H_1)-2+\frac{1}{m_2(H_2)}} & \text{if } H_2 \text{ is non-empty and } v(H_1)\ge 2,\\ 0 & \text{otherwise}\end{cases}$$

and the *asymmetric* 2*-density of the pair* (H_1,H_2) to be

$$m_2(H_1,H_2):=\max\left\{d_2(J,H_2):J\subseteq H_1\right\}.$$

We say that H_1 is *balanced with respect to* $d_2(\cdot,H_2)$ if we have $d_2(H_1,H_2)=m_2(H_1,H_2)$ and *strictly balanced with respect to* $d_2(\cdot,H_2)$ if for all proper subgraphs $J\subsetneq H_1$ we have $d_2(J,H_2)<m_2(H_1,H_2)$. Note that $m_2(H_1)\ge m_2(H_1,H_2)\ge m_2(H_2)$ (see Proposition 1.7).

Kohayakawa and Kreuter [10] conjectured the following generalisation of Theorem 1.1. (We give here a slight rephrasing of the conjecture: we consider r colours (instead of 2) and add the assumption of Kohayakawa, Schacht and Spöhel [11] that H_1 and H_2 are not forests.[1])

Conjecture 1.3. (Kohayakawa and Kreuter [10]) *Let* $r\ge 2$ *and suppose that* $H_1,\ldots,H_r$ *are non-empty graphs such that* $m_2(H_1)\ge\cdots\ge m_2(H_r)$ *and* $m_2(H_2)>1$. *Then there exist constants* $b,B>0$ *such that*

$$\lim_{n\to\infty}\mathbb{P}[G_{n,p}\to(H_1,\ldots,H_r)]=\begin{cases}0 & \text{if } p\le bn^{-1/m_2(H_1,H_2)},\\ 1 & \text{if } p\ge Bn^{-1/m_2(H_1,H_2)}.\end{cases}$$

Observe that we would always need $m_2(H_2)\ge 1$ as an assumption, otherwise $m_2(H_2)=1/2$ (that is, H_2 is the union of a matching and some isolated vertices) and we would have that $m_2(H_1,H_2)=e_J/v_J$ for some non-empty subgraph $J\subseteq H_1$. For any constant $B>0$, the probability that $G_{n,p}$ with $p=Bn^{-1/m_2(H_1,H_2)}$ contains no copy of H_1 exceeds a positive constant $C=C(B)$; see, for example [8]. We include the assumption of Kohayakawa, Schacht and Spöhel [11], that $m_2(H_2)>1$, to avoid possible complications arising from H_2 (and/or H_1) being certain forests, such as those excluded in the statement of Theorem 1.1.

The intuition behind the threshold in Conjecture 1.3 is most readily explained in the case of $r=3$, $H_2=H_3$ and when $m_2(H_1)>m_2(H_1,H_2)$. (The following explanation is adapted from [6].) Firstly, observe that we can assign colour 1 to every edge that does not lie in a copy of H_1. Since $m_2(H_1)>m_2(H_1,H_2)$, we expect that the copies of H_1 in $G_{n,p}$ with $p=\Theta(n^{-1/m_2(H_1,H_2)})$ do

[1] This version of the conjecture is the same as that given in [15].

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

not overlap much (by similar reasoning as in the intuition for the threshold in Theorem 1.1). Hence the number of edges left to be coloured is of the same order as the number of copies of H_1, which is $\Theta(n^{v(H_1)}p^{e(H_1)})$. If we further assume that these edges are randomly distributed (which is not correct, but gives good intuition) then we get a random graph G^* with edge probability $p^* = \Theta(n^{v(H_1)-2}p^{e(H_1)})$. Now we colour G^* with colours 2 and 3, and apply the intuition from the symmetric case (as $H_2 = H_3$): if the copies of H_2 are heavily overlapping then we cannot hope to colour without getting a monochromatic copy of H_2, but if not then we should be able to colour. As observed before, a threshold for this property is $p^* = n^{-1/m_2(H_2)}$. Solving $n^{v(H_1)-2}p^{e(H_1)} = n^{-1/m_2(H_2)}$ for p then yields $p = n^{-1/m_2(H_1,H_2)}$, the conjectured threshold.

After earlier work (see e.g. [6, 7, 10, 11, 14]), the 1-statement of Conjecture 1.3 was proven by Mousset, Nenadov and Samotij [15].

1.3 The 0-statement of the Kohayakawa-Kreuter Conjecture

In this paper, we are interested in the 0-statement of Conjecture 1.3, which has so far only been proven when H_1 and H_2 are both cycles [10], both cliques [14] and, recently, when H_1 is a clique and H_2 is a cycle [12]. We also note that the authors of [6] prove, under certain balancedness conditions, the 0-statement of a generalised version of Conjecture 1.3 which allows $H_1, \dots, H_r$ to be uniform hypergraphs.

Definition 1.4. Let H_1 and H_2 be non-empty graphs. We say that a graph G has a valid edge-colouring for H_1 and H_2 if there exists a red/blue colouring of the edges of G that does not produce a red copy of H_1 or a blue copy of H_2.

To prove the 0-statement of Conjecture 1.3, one only needs to show that $G = G_{n,p}$ has a valid edge-colouring for H_1 and H_2 asymptotically almost surely (a.a.s.) (that is, we ignore $H_3, \dots, H_r$ and colours $3, \dots, r$). Further, when $m_2(H_1) > m_2(H_2)$ we can assume when trying to prove the 0-statement of Conjecture 1.3 that H_2 is strictly 2-balanced and H_1 is strictly balanced with respect to $d_2(\cdot, H_2)$. Indeed, if either of these assumptions do not hold then one can replace H_1 and H_2 with subgraphs $H_1' \subseteq H_1$ and $H_2' \subseteq H_2$ such that H_2' is strictly 2-balanced and H_1' is strictly balanced with respect to $d_2(\cdot, H_2')$. Then we would aim to show that G has a valid edge-colouring for H_1' and H_2' a.a.s.[2] Similarly, when $m_2(H_1) = m_2(H_2)$, we can assume when proving the 0-statement of Conjecture 1.3 that both H_1 and H_2 are strictly 2-balanced.

In past work on attacking 0-statements of Ramsey problems (e.g. Conjecture 1.3 and Theorem 1.1), researchers have applied variants of a standard and natural approach (see e.g. [10, 12, 14, 19]). The main contribution of this paper is to prove that every step of this approach, except one, holds with respect to Conjecture 1.3. That is, we reduce Conjecture 1.3 to a single subproblem. To state this subproblem we require a number of definitions adapted from [14].

Definition 1.5. For any graph G we define the families

$$\mathcal{R}_G := \{R \subseteq G : R \cong H_1\} \text{ and } \mathcal{L}_G := \{L \subseteq G : L \cong H_2\}$$

of all copies of H_1 and H_2 in G, respectively. Furthermore, we define

$$\mathcal{L}_G^* := \{L \in \mathcal{L}_G : \forall e \in E(L)\ \exists R \in \mathcal{R}_G \text{ s.t. } E(L) \cap E(R) = \{e\}\} \subseteq \mathcal{L}_G,$$

the family of copies L of H_2 in G with the property that for every edge e in L there exists a copy R of H_1 such that the edge sets of L and R intersect uniquely at e;

$$\mathcal{C} = \mathcal{C}(H_1, H_2) := \{G = (V, E) : \forall e \in E\ \exists (L, R) \in \mathcal{L}_G \times \mathcal{R}_G \text{ s.t. } E(L) \cap E(R) = \{e\}\},$$

[2] Which would immediately imply that G has a valid edge-colouring for H_1 and H_2.

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

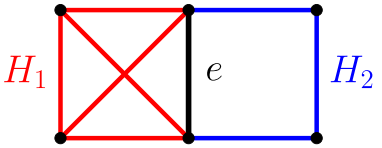

Figure 1. A copy of H_1 intersecting a copy of H_2 at an edge e where $H_1 = K_4$, $H_2 = C_4$ and e is yet to be coloured.

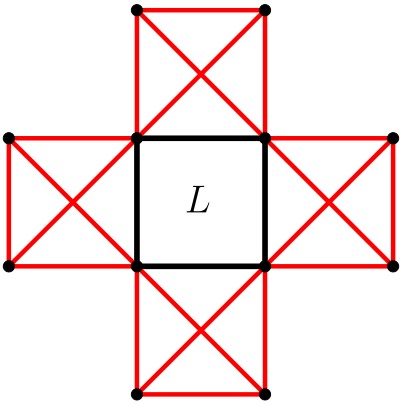

Figure 2. A copy L of H_2 such that $L \in \mathcal{L}^*_{G'}$ where $H_1 = K_4$, $H_2 = C_4$ and the edges of L are yet to be coloured.

the family of graphs G where every edge is the unique edge-intersection of some copy L of H_2 and some copy R of H_1; and

$$\mathcal{C}^* = \mathcal{C}^*(H_1, H_2) := \{G = (V, E) : \forall e \in E\ \exists L \in \mathcal{L}^*_G \text{ s.t. } e \in E(L)\},$$

the family of graphs G where every edge is contained in a copy L of H_2 which has at each edge e some copy R of H_1 attached such that $E(L) \cap E(R) = \{e\}$.

Note that $\mathcal{C}^*(H_1, H_2) \subseteq \mathcal{C}(H_1, H_2)$. Also, it is important to note that for $L \in \mathcal{L}^*_G$, the vertex sets of copies of H_1 appended at edges of L may overlap with each other and/or intersect with more than 2 vertices of L (that is, more than the 2 vertices of the edge of L they are appended at).

Let us now discuss the relevance of these sets of graphs to proving the 0-statement of Conjecture 1.3. Recall that we aim to show that $G = G_{n,p}$ has a valid edge-colouring for H_1 and H_2. Now, certain obstacles relating to $\mathcal{C}$, $\mathcal{L}^*_G$ and $\mathcal{C}^*$ may appear while we are constructing such a valid edge-colouring. For instance, say there exists a subgraph $G' \subseteq G$ such that $G' \in \mathcal{C}$. Then each edge $e \in E(G')$ has a copy R of H_1 and a copy L of H_2 in G that uniquely edge-intersect in that edge. Say during the construction of our colouring we come to a point where every edge of $E(R) \setminus \{e\}$ is coloured red, every edge of $E(L) \setminus \{e\}$ is coloured blue and e is yet to be coloured (see Figure 1). Then however we colour the edge e, we have a red copy of H_1 or a blue copy of H_2. Hence one must be careful when constructing a valid edge-colouring in graphs from $\mathcal{C}$.

Now say there exists a subgraph $G' \subseteq G$ such that $G' \in \mathcal{C}^*$. Then each edge $e \in E(G')$ is contained in some copy L of H_2 such that $L \in \mathcal{L}^*_{G'}$, that is, L has a different copy $R_{e'}$ of H_1 appended at each edge $e' \in E(L)$ such that $E(L) \cap E(R_{e'}) = \{e\}$. Similarly as before, say during the construction of our colouring we come to a point where every edge of $E(R_{e'}) \setminus \{e'\}$ is red (see Figure 2).

However we colour the edges of L, we will produce a red copy of H_1 or a blue copy of H_2. Observe that the structure in Figure 2 is essentially a generalisation of the structure in Figure 1. Indeed, $\mathcal{C}^* \subseteq \mathcal{C}$. Hence graphs in $\mathcal{C}^*$ are also possibly difficult to construct a valid edge-colouring in.

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

For the next section we need also the following definition. For a graph H, we define $d(H) := e_H/v_H$ if $v(H) \geq 1$ and $d(H) := 0$ otherwise. Also, define $m(H) := \max\{d(J) : J \subseteq H\}$.

1.4 The family of graphs $\hat{\mathcal{A}}(H_1, H_2, \varepsilon)$

Throughout the rest of this section, assume that n is sufficiently large and for some constant $b > 0$ that $p = bn^{-1/m_2(H_1,H_2)}$. We now define a very important family of graphs which we will need to state our reduction of Conjecture 1.3. These graphs present potentially significant obstacles to constructing a valid edge-colouring in $G_{n,p}$ a.a.s. After defining this family of graphs we will explain the thinking behind each part of the definition.

Definition 1.6. Let H_1 and H_2 be non-empty graphs such that $m_2(H_1) \geq m_2(H_2) > 1$. Let $\varepsilon := \varepsilon(H_1, H_2) > 0$ be a constant. Define $\hat{\mathcal{A}} = \hat{\mathcal{A}}(H_1, H_2, \varepsilon)$ to be

$$\hat{\mathcal{A}} := \begin{cases} \{A \in \mathcal{C}^*(H_1, H_2) : m(A) \leq m_2(H_1, H_2) + \varepsilon \wedge A \text{ is 2-connected}\} \text{ if } m_2(H_1) > m_2(H_2), \\ \{A \in \mathcal{C}(H_1, H_2) : m(A) \leq m_2(H_1, H_2) + \varepsilon \wedge A \text{ is 2-connected}\} \text{ if } m_2(H_1) = m_2(H_2). \end{cases}$$

1.4.1 Why do we have $m(A) \leq m_2(H_1, H_2) + \varepsilon$?

We require the following definition: For any graph F, let

$$\lambda(F) := v(F) - \frac{e(F)}{m_2(H_1, H_2)}.$$

The definition of $\lambda(F)$ is motivated by the fact that the expected number of copies of F in $G_{n,p}$ has order of magnitude

$$n^{v(F)}p^{e(F)} = b^{e(F)}n^{\lambda(F)}.$$

Let $A \in \hat{\mathcal{A}}(H_1, H_2, \varepsilon)$ for some pair of graphs H_1 and H_2 with $m_2(H_1) \geq m_2(H_2) > 1$ and a constant $\varepsilon := \varepsilon(H_1, H_2) > 0$. Now, assuming $m(A) = d(A)$ $(= |E(A)|/|V(A)|)$, observe that the following inequalities are equivalent.

$$\begin{aligned} m(A) &< m_2(H_1, H_2) \\ \frac{|E(A)|}{m_2(H_1, H_2)} &< |V(A)| \\ 0 &< \lambda(A). \end{aligned}$$

Thus, if $m(A) < m_2(H_1, H_2)$, then the expected number of copies of A in $G_{n,p}$ has order of magnitude $\omega(1)$. Thus such graphs A can be expected to be found in $G_{n,p}$ a.a.s.

Now consider if instead we had $m(A) > m_2(H_1, H_2) + \varepsilon$. Then the following inequalities are equivalent.

$$\begin{aligned} m(A) &> m_2(H_1, H_2) + \varepsilon \\ \lambda(A) &< -\frac{|V(A)|\varepsilon}{m_2(H_1, H_2)}. \end{aligned}$$

Thus the expected number of copies of A in $G_{n,p}$ has order of magnitude $o(1)$, that is, there are no copies of A in $G_{n,p}$ a.a.s.

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

Hence graphs $A \in \hat{\mathcal{A}}$ have sufficiently low density to plausibly exist in $G_{n,p}$. Note that we stipulate $m(A) \leq m_2(H_1, H_2) + \varepsilon$ rather than $m(A) < m_2(H_1, H_2)$ so that later we can argue[3] that 2-connected graphs $A \in C^*(H_1, H_2)$ that do not belong to $\hat{\mathcal{A}}$ have $m(A) > m_2(H_1, H_2) + \varepsilon$, and so, as just noted, do not appear in $G_{n,p}$ a.a.s.

1.4.2 Why is $A \in \mathcal{C}$ or $\mathcal{C}^*$?

As discussed at the end of Section 1.3, graphs in $\mathcal{C}$ and $\mathcal{C}^*$ contain certain structures which could be obstacles to constructing a valid edge-colouring.

1.4.3 Why is whether $A \in C^*$ or $A \in C$ dependent on $m_2(H_1)$ and $m_2(H_2)$?

Now let us consider why in the definition of $\hat{\mathcal{A}}$ we have that $A \in \mathcal{C}^*(H_1, H_2)$ when $m_2(H_1) > m_2(H_2)$ but $A \in \mathcal{C}(H_1, H_2)$ when $m_2(H_1) = m_2(H_2)$. In short, the defining structure of graphs in $\mathcal{C}^*(H_1, H_2)$ – that of a copy of H_2 with appended copies of H_1 at its edges (see Figure 2 e.g.) – is 'meaningful' when $m_2(H_1) > m_2(H_2)$ but not when $m_2(H_1) = m_2(H_2)$. The following result will aid us in elaborating on this remark. It illuminates the relationship between the one and two argument m_2 measures and can be readily proven using elementary arguments.

Proposition 1.7. *Suppose that H_1 and H_2 are non-empty graphs with $m_2(H_1) \geq m_2(H_2)$. Then we have*

$$m_2(H_1) \geq m_2(H_1, H_2) \geq m_2(H_2).$$

Moreover,

$$m_2(H_1) > m_2(H_1, H_2) > m_2(H_2) \text{ whenever } m_2(H_1) > m_2(H_2).$$

Recall from earlier that we can assume when proving the 0-statement of Conjecture 1.3 that $m_2(H_1) \geq m_2(H_2) \geq 1$, H_2 is strictly 2-balanced, and H_1 is strictly balanced with respect to $d_2(\cdot, H_2)$ if $m_2(H_1) > m_2(H_2)$ and strictly 2-balanced if $m_2(H_1) = m_2(H_2)$. Assuming these density conditions, one can view the values of $m_2(H_1)$, $m_2(H_1, H_2)$ and $m_2(H_2)$ in a particular way which will be relevant across this paper. Take a copy of H_1 and attach a copy of H_2 to get the same structure as in Figure 1. Then, if we take the number of edges we added to the copy of H_1 and divide it by the number of vertices we added, this is precisely

$$\frac{|E(H_2)| - 1}{|V(H_2)| - 2} = m_2(H_2).$$

Similarly, if one takes a copy of H_1 and attaches a copy of H_2 with $|E(H_2)| - 1$ appended copies of H_1 at each of its edges precisely as in Figure 2, then the number of edges added to the copy of H_1 over the number of vertices added is

$$\frac{|E(H_1)|(|E(H_2)| - 1)}{(|V(H_1)| - 2)(|E(H_2)| - 1) + (|V(H_2)| - 2)} = \frac{|E(H_1)|}{|V(H_1)| - 2 + \frac{|V(H_2)|-2}{|E(H_2)|-1}}$$
$$= \frac{|E(H_1)|}{|V(H_1)| - 2 + \frac{1}{m_2(H_2)}} = m_2(H_1, H_2),$$

where in the $m_2(H_1) = m_2(H_2)$ case the final equality follows from H_1 being strictly 2-balanced and Proposition 1.7.[4] Also, if we attach a copy of H_1 to a copy of H_1 or H_2 in a similar manner to the structure in Figure 1 (at a single edge with no additional vertices or edges overlapping)

[3] For instance, in the proof of Claim 6.1.

[4] That is, H_1 is balanced with respect to $d_2(\cdot, H_2)$ when $m_2(H_1) = m_2(H_2)$ and H_1 is strictly 2-balanced.

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

and H_1 is strictly 2-balanced, then the number of edges added over the number of vertices added is $m_2(H_1)$. For brevity, let F_1 be the graph in Figure 1 and F_2 be the graph in Figure 2, ignoring the colouring of the edges. If $m_2(H_1) = m_2(H_2)$, then $m_2(H_1) = m_2(H_1, H_2) = m_2(H_2)$ by Proposition 1.7 and one can calculate that $\lambda(H_1) = \lambda(H_2) = \lambda(F_1) = \lambda(F_2)$, that is, the expected numbers of these graphs in $G_{n,p}$ are approximately the same; they have the same orders of magnitude. Moreover, if we appended less than $|E(H_2)| - 1$ copies of H_1 to the copy of H_2[5] – call such a graph F_2' – then we would still have $\lambda(F_2') = \lambda(H_1)$. However, when $m_2(H_1) > m_2(H_2)$ we have $m_2(H_1) > m_2(H_1, H_2) > m_2(H_2)$ by Proposition 1.7, and one can calculate that $\lambda(H_1) = \lambda(F_2)$, but $\lambda(H_1) < \lambda(F_1)$. Thus, speaking broadly, F_1 is more likely to appear in $G_{n,p}$ than H_1. In fact, $\lambda(H_1) < \lambda(F_2')$, irrelevant of the position and number of the appended copies of H_1 in F_2'.

Thus when $m_2(H_1) = m_2(H_2)$, A_2 is not a particularly meaningful construction, but when $m_2(H_1) > m_2(H_2)$ we see that A_2 makes more sense to consider. One can observe that this accords with Proposition 1.7, in that either $m_2(H_1) = m_2(H_1, H_2) = m_2(H_2)$, and so $m_2(H_1, H_2)$ does not have a distinct value, or $m_2(H_1) > m_2(H_1, H_2) > m_2(H_2)$ and $m_2(H_1, H_2)$ does have a distinct value. See Section 9 for additional discussion on why for $A \in \hat{\mathcal{A}}$ we take $A \in \mathcal{C}$ when $m_2(H_1) = m_2(H_2)$.

1.4.4 Why is A a 2-connected graph?

Let $G = G_{n,p}$ and consider the collection of $\boldsymbol{F}_2(G)$ of maximally 2-connected subgraphs A of G.[6] Thus $E(G)$ can be partitioned into $\boldsymbol{F}_2(G)$ and a forest $\boldsymbol{F}_1(G)$. Later, we will prove that the density conditions we can assume for H_1 and H_2 when proving the 0-statement of Conjecture 1.3 imply that H_1 and H_2 are 2-connected (see Lemma 4.2). Thus, assuming these density conditions, no copy of H_1 or H_2 has edges that lie in two different graphs in $\boldsymbol{F}_2(G)$. Importantly, this means that if each graph in $\boldsymbol{F}_2(G)$ has a valid edge-colouring for H_1 and H_2, then there exists a valid edge-colouring covering every graph in $\boldsymbol{F}_2(G)$. Further, the edges of the forest $\boldsymbol{F}_1(G)$ belong to no copies of H_1 and H_2 in G, thus we can colour them any way we want. So with regard to finding a valid edge-colouring of G, we can reduce to looking at 2-connected graphs. Hence we reduce to looking at 2-connected graphs for $\hat{\mathcal{A}}$.

1.5 Reduction of Conjecture 1.3

We now state the subproblem we reduce Conjecture 1.3 to as the following conjecture.

Conjecture 1.8. *Let H_1 and H_2 be non-empty graphs such that $H_1 \neq H_2$ and $m_2(H_1) \geq m_2(H_2)$. Assume H_2 is strictly 2-balanced. Moreover, assume H_1 is strictly balanced with respect to $d_2(\cdot, H_2)$ if $m_2(H_1) > m_2(H_2)$ and strictly 2-balanced if $m_2(H_1) = m_2(H_2)$. Then there exists a constant $\varepsilon := \varepsilon(H_1, H_2) > 0$ such that the set $\hat{\mathcal{A}}$ is finite and every graph in $\hat{\mathcal{A}}$ has a valid edge-colouring for H_1 and H_2.*

Notice that we can assume $H_1 \neq H_2$ as the $H_1 = H_2$ case of Conjecture 1.3 is handled by Theorem 1.1.

To be clear, the main purpose of this paper is to show that if Conjecture 1.8 holds then the rest of a variant of a standard approach for attacking the 0-statement of Conjecture 1.3 falls into place. That is, Conjecture 1.8 is a natural subproblem of Conjecture 1.3. Thus we prove the following theorem.

Theorem 1.9. *If Conjecture 1.8 is true then the 0-statement of Conjecture 1.3 is true.*

[5] In the same manner as F_2 - attaching the copies at single edges and not overlapping additionally with any other vertices or edges of the copy of H_2 or any of the other copies of H_1.

[6] That is, if $e \in A$ for some graph A in $\boldsymbol{F}_2(G)$, then there does not exist some $A' \subseteq G$ such that $A \subset A'$ and A' is 2-connected.

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

We prove Conjecture 1.8 for almost every pair of regular graphs, which, by Theorem 1.9, significantly extends the class of graphs for which the 0-statement of Conjecture 1.3 is resolved.

Theorem 1.10. *Let H_1 and H_2 meet the criteria in Conjecture 1.8. In addition, let H_1 and H_2 be regular graphs, excluding the cases when (i) H_1 and H_2 are a clique and a cycle, (ii) H_2 is a cycle and $|V(H_1)| \geq |V(H_2)|$ or (iii) $(H_1, H_2) = (K_3, K_{3,3})$. Then Conjecture 1.8 is true for H_1 and H_2.*

As a natural subproblem of the 0-statement of Conjecture 1.3, we believe that Conjecture 1.8 is a considerably more approachable problem than the 0-statement of Conjecture 1.3. Indeed, the techniques used in the proof of Theorem 1.10 are elementary and uncomplicated. Thus, we hope that a full resolution of Conjecture 1.3 can be achieved via Theorem 1.9.

2. Overview of the proof of Theorem 1.9

As mentioned earlier, to prove Theorem 1.9 we will employ a variant of a standard approach for attacking 0-statements of Ramsey problems. For attacking the 0-statement of Conjecture 1.8, this standard approach is as follows:

- For $G = G_{n,p}$, assume $G \to (H_1, H_2)$;
- Use structural properties of G (resulting from this assumption) to show that G contains at least one of a sufficiently small collection of non-isomorphic graphs $\mathcal{F}$;
- Show that there exists a constant $b > 0$ such that for $p \leq bn^{-1/m_2(H_1,H_2)}$ we have that G contains no graph in $\mathcal{F}$ a.a.s.;
- Conclude, by contradiction, that $G \not\to (H_1, H_2)$ a.a.s.

The variant of this approach we will use is due to Marciniszyn, Skokan, Spöhel and Steger [14], who proved Conjecture 1.3 for cliques. In [14], for $r > \ell \geq 3$, they employ an algorithm called Asym-Edge-Col which either produces a valid edge-colouring for K_r and K_ℓ of G (showing that $G \not\to (K_r, K_\ell)$) or encounters an error. Instead of assuming $G \to (K_r, K_\ell)$, they assume algorithm Asym-Edge-Col encounters an error, and proceed with the standard approach from there. One of the advantages of this approach is that it provides an algorithm for constructing a valid edge-colouring for K_r and K_ℓ, rather than just proving the existence of such a colouring.

2.1 On Conjecture 1.8

As mentioned earlier, we provide all but one step, Conjecture 1.8, of this approach. Let us consider how Conjecture 1.8 relates to previous work on the 0-statement of Conjecture 1.3. Firstly, Conjecture 1.8 was implicitly proven for pairs of cliques in [14] and pairs of a clique and a cycle in [12]. More specifically, when H_1 and H_2 are both cliques (except when $H_1 = H_2 = K_3$)[7], the authors of [14] prove a slightly more general version of Conjecture 1.8 (Lemma 8 in [14]) where $\hat{\mathcal{A}}(H_1, H_2, \varepsilon)$ is replaced with the set

$$\mathcal{A}(H_1, H_2) := \{A \in \mathcal{C}(H_1, H_2) : m(A) \leq m_2(H_1, H_2) + 0.01 \wedge A \text{ is 2-connected}\}.$$

Note that the proof of Lemma 8 in [14] shows that $\mathcal{A}(H_1, H_2) \neq \emptyset$ for certain pairs of cliques H_1 and H_2. When H_1 is a clique, H_2 is a cycle and $H_1 \neq H_2$ (that is, excluding again the case when $H_1 = H_2 = K_3$), the proof of Lemma 3.3 in [12] implies that there exists a constant $\varepsilon > 0$ such that $\hat{\mathcal{A}}(H_1, H_2, \varepsilon) = \emptyset$.

For reference, we note here the places in our proof of Theorem 1.9 where we specifically need Conjecture 1.8 to hold:

- the proof of Lemma 5.3;

[7] The case $H_1 = H_2$ of Conjecture 1.3 is, of course, covered by Theorem 1.1.

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

- the proofs of Claims 6.6 and 7.6;
- the definition of $\gamma = \gamma(H_1, H_2)$ in Section 6.

2.2 Proof sketch of Theorem 1.9

Let us now proceed with describing the proof of Theorem 1.9 in detail. In what follows, we write *(Result A; Result B)* to mean that 'Result B in [14] fulfils the same role (in [14]) as Result A does in our proof of Theorem 1.9'. This is to illustrate how we indeed provide every step bar one (Conjecture 1.8) of a proof of the 0-statement of Conjecture 1.3.

Firstly, as in [14], we give an algorithm Asym-Edge-Col that, assuming Conjecture 1.8 holds, produces a valid edge-colouring for H_1 and H_2 of $G = G_{n,p}$ provided it does not encounter an error (Lemma 5.4; Lemma 11). Our aim then is to prove that Asym-Edge-Col does not encounter an error a.a.s. (Lemma 5.5; Lemma 12), that is, $G \not\to (H_1, H_2)$ a.a.s. We split our proof of Lemma 5.5 into two cases: when $m_2(H_1) > m_2(H_2)$ and when $m_2(H_1) = m_2(H_2)$.

Suppose for a contradiction that Asym-Edge-Col encounters an error. Let $G' \subseteq G$ be the graph that Asym-Edge-Col got stuck on when it encountered this error. In the $m_2(H_1) > m_2(H_2)$ case, we input G' into an auxiliary algorithm Grow which always outputs a subgraph $F \subseteq G'$ (Claim 6.1; Claim 13) belonging to a sufficiently small collection of non-isomorphic graphs $\mathcal{F}$. The definition of $\mathcal{F}$ will be such that with high probability no copy of any $F \in \mathcal{F}$ will be present in $G_{n,p}$, provided that $|\mathcal{F}|$ is sufficiently small.

In order to show $|\mathcal{F}|$ is sufficiently small, we carefully analyse the possible outputs of Grow. Assuming Conjecture 1.8 holds, we show that only a constant number of graphs can be produced by Grow if one of two special cases occurs. If neither of these special cases occur, then, starting from a copy of H_1, in each step of Grow our subgraph F is constructed iteratively by either (i) appending a copy of H_1 to F or (ii) appending a 'flower-like' structure to F, consisting of a central copy of H_2 with 'petals' that are appended copies of H_1. We say an iteration is *degenerate* if it is of type (i) or, loosely speaking, of type (ii) where 'the flower is folded in on itself or into F'. Otherwise an iteration is called *non-degenerate*. Denote by $\lambda(F)$ the order of magnitude of the expected number of copies of F in $G_{n,p}$ with $p = bn^{-1/m_2(H_1,H_2)}$. Key to showing $|\mathcal{F}|$ is sufficiently small is proving that $\lambda(F)$ stays the same after a non-degenerate iteration (Claim 6.2; Claim 14) and decreases by a *constant* amount after a degenerate iteration (Claim 6.3; Claim 15). Indeed, one of the termination conditions for Grow is that $\lambda(F) < -\gamma$ (where $\gamma = \gamma(H_1, H_2, \varepsilon) > 0$ is defined later in Section 6, given $\varepsilon = \varepsilon(H_1, H_2) > 0$, the constant acquired from assuming Conjecture 1.8 holds), that is, only a constant number of such *degenerate* steps occur before Grow terminates (Claim 6.4; Claim 16). Proving Claim 6.3 is the main work of this paper. An important step in proving it is showing that if an iteration of type (ii) occurs where, loosely speaking, 'the flower is folded in on itself', we get a helpful inequality comparing this iteration with a non-degenerate iteration (Lemma 6.8; Lemma 21). Indeed, the most novel work of this paper is the proof of Lemma 6.8.

The proof of Lemma 5.5 in the $m_2(H_1) = m_2(H_2)$ case is both similar and significantly simpler. Notably, we use a different algorithm, Grow-Alt, to grow our subgraph $F \subseteq G'$. Our analysis of Grow-Alt is much quicker than that of Grow, allowing us to easily prove a result analogous to Claim 6.3.

2.3 Differences between our work and [14]

As mentioned earlier, our approach to proving Theorem 1.9 builds on the work of Marciniszyn, Skokan, Spöhel and Steger in [14]. For readers familiar with [14], we include the following list of differences between this paper and [14] (some of which we have already noted):

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

- We prove and employ a new result (Lemma 4.2) concerning types of balancedness and 2-connectivity;
- When $m_2(H_1) > m_2(H_2)$, we refine the proof in [14] to consider $\hat{\mathcal{A}}(H_1, H_2, \varepsilon)$ instead of $\mathcal{A}(H_1, H_2)$;
- We generalise from considering triangle-sparse graphs to considering (H_1, H_2)-sparse graphs (see Section 5);
- Lemma 6.8 and its setup (see Sections 6.2 and 6.3) are quite different to Lemma 21 and its setup in [14];
- Although Claim 6.9 is analogous to Claim 19 in [14], its proof is quite different, utilising the balancedness properties of H_1 and H_2;
- Although Claim 6.10 is analogous to Claim 22 in [14], our proof is slightly different, swapping the latter two steps of the proof of Claim 22 in order to apply our Lemma 6.8 in place of Lemma 21;
- To account for H_1 and H_2 possibly having less structure than cycles or cliques[8], the statement of Claim 6.5 and the proofs of Claims 6.5 and 6.6 differ somewhat from their counterparts (Claims 17 and 18) in [14];
- When $m_2(H_1) = m_2(H_2)$, we use a slightly different algorithm Grow-Alt (see Section 7) to algorithm Grow.

3. Organisation of paper

The paper is organised as follows. In Section 4, we collect together notation, density measures and several useful results we will need. In Section 5, we give our algorithm Asym-Edge-Col for producing a valid edge-colouring for H_1 and H_2 of $G = G_{n,p}$ provided it does not encounter an error (and Conjecture 1.8 holds for H_1 and H_2). In Sections 6-6.3, we prove that Asym-Edge-Col does not encounter an error a.a.s. (Lemma 5.5) in the case when $m_2(H_1) > m_2(H_2)$. In Section 7, we prove Lemma 5.5 in the case when $m_2(H_1) = m_2(H_2)$. In Section 8, we prove Theorem 1.10, before providing some concluding remarks in Section 9.

4. Notation, density measures and useful results

As far as possible we keep to the notation used in [14]. Also, we repeat several definitions used earlier for ease of reference.

Let $G = (V, E)$ be a graph. We denote the number of vertices in G by $v(G) = v_G := |V(G)|$ and the number of edges in G by $e(G) = e_G := |E(G)|$. Moreover, for graphs H_1 and H_2 we let $v_1 := |V(H_1)|$, $e_1 := |E(H_1)|$, $v_2 := |V(H_2)|$ and $e_2 := |E(H_2)|$.

Let H be a graph. The most well-known density measure is

$$d(H) := \begin{cases} e_H/v_H & \text{if } v(H) \geq 1, \\ 0 & \text{otherwise.} \end{cases}$$

Taking the maximum value of d over all subgraphs $J \subseteq H$, we have the following measure

$$m(H) := \max\{d(J) : J \subseteq H\}.$$

(We say that a graph H is *balanced with respect to d*, or just *balanced*, if we have $d(H) = m(H)$. Moreover, we say H is *strictly balanced* if for every proper subgraph $J \subsetneq H$, we have $d(J) < m(H)$.)

[8] In particular, for $i \in \{1, 2\}$, H_i may not have the property that when one removes any single edge (and no vertices) from H_i one gets the same isomorphic graph, irrespective of which edge is removed.

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

In [19], Rödl and Ruciński introduced the following so-called 2-*density measure*.

$$d_2(H) := \begin{cases} (e_H - 1)/(v_H - 2) & \text{if } H \text{ is non-empty with } v(H) \geq 3, \\ 1/2 & \text{if } H \cong K_2, \\ 0 & \text{otherwise.} \end{cases}$$

As with d, we have an associated measure based on maximising d_2 over subgraphs of H:

$$m_2(H) := \max\left\{d_2(J) : J \subseteq H\right\}.$$

Analogously to the notion of balancedness, we say that a graph H is 2-*balanced* if $d_2(H) = m_2(H)$, and *strictly* 2-*balanced* if for all proper subgraphs $J \subsetneq H$, we have $d_2(J) < m_2(H)$.

Regarding asymmetric Ramsey properties, in [10], Kohayakawa and Kreuter introduced the following generalisation of d_2. Let H_1 and H_2 be any graphs, and define

$$d_2(H_1, H_2) := \begin{cases} \frac{e_1}{v_1 - 2 + \frac{1}{m_2(H_2)}} & \text{if } H_2 \text{ is non-empty and } v_1 \geq 2, \\ 0 & \text{otherwise.} \end{cases}$$

Similarly to before, we have the following measure based on maximising d_2 over all subgraphs $J \subseteq H_1$.

$$m_2(H_1, H_2) := \max\left\{d_2(J, H_2) : J \subseteq H_1\right\}.$$

We say that H_1 is *balanced with respect to* $d_2(\cdot, H_2)$ if we have $d_2(H_1, H_2) = m_2(H_1, H_2)$ and *strictly balanced with respect to* $d_2(\cdot, H_2)$ if for all proper subgraphs $J \subsetneq H_1$ we have $d_2(J, H_2) < m_2(H_1, H_2)$.

Observe that $m_2(\cdot, \cdot)$ is not symmetric in both arguments. Recall Proposition 1.7.

Proposition 1.7. *Suppose that H_1 and H_2 are non-empty graphs with $m_2(H_1) \geq m_2(H_2)$. Then we have*

$$m_2(H_1) \geq m_2(H_1, H_2) \geq m_2(H_2).$$

Moreover,

$$m_2(H_1) > m_2(H_1, H_2) > m_2(H_2) \text{ whenever } m_2(H_1) > m_2(H_2).$$

Note that if $m_2(H_1) = m_2(H_2)$ and H_1 and H_2 are non-empty graphs, then H_1 cannot be strictly balanced with respect to $d_2(\cdot, H_2)$ unless $H_1 \cong K_2$. Indeed, otherwise, by Proposition 1.7 we would then have that

$$m_2(H_2) = m_2(H_1, H_2) > d_2(K_2, H_2) = m_2(H_2).$$

The following fact will be useful in the proofs of Lemmas 4.2 and 6.8.

Fact 4.1. *For $a, c, C \in \mathbb{R}$ and $b, d > 0$, we have*

$$(i)\ \frac{a}{b} \leq C \wedge \frac{c}{d} \leq C \implies \frac{a+c}{b+d} \leq C \text{ and } (ii)\ \frac{a}{b} \geq C \wedge \frac{c}{d} \geq C \implies \frac{a+c}{b+d} \geq C$$

and similarly, if also $b > d$,

$$(iii)\ \frac{a}{b} \leq C \wedge \frac{c}{d} \geq C \implies \frac{a-c}{b-d} \leq C \text{ and } (iv)\ \frac{a}{b} \geq C \wedge \frac{c}{d} \leq C \implies \frac{a-c}{b-d} \geq C.$$

The following result will be very useful for us, creating an important connection between types of balancedness and 2-connectivity.

Lemma 4.2. *Let H_1 and H_2 be graphs such that either (i) $m_2(H_1) > m_2(H_2) > 1$, H_2 is strictly 2-balanced and H_1 is strictly balanced with respect to $d_2(\cdot, H_2)$; or (ii) $m_2(H_1) = m_2(H_2) > 1$ and H_1 and H_2 are both strictly 2-balanced. Then H_1 and H_2 are both 2-connected.*

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

Proof. By [16, Lemma 3.3], strictly 2-balanced graphs are 2-connected, hence (*ii*) holds and H_2 is 2-connected in case (*i*). We now use a very similar method to that of the proof of Lemma 3.3 to show that H_1 is 2-connected in case (*i*). Since H_1 is strictly balanced with respect to $d_2(\cdot, H_2)$, we have that H_1 is connected. Indeed, assume not. Let H_1 have $k \geq 2$ components and denote the number of vertices and edges in each component by $u_1, \ldots, u_k$ and $d_1, \ldots, d_k$, respectively. Then since H_1 is strictly balanced with respect to $d_2(\cdot, H_2)$, we must have that

$$\frac{\sum_{i=1}^{k} d_i}{\sum_{i=1}^{k} u_i - 2 + \frac{1}{m_2(H_2)}} > \frac{d_1}{u_1 - 2 + \frac{1}{m_2(H_2)}}$$

and

$$\frac{\sum_{i=1}^{k} d_i}{\sum_{i=1}^{k} u_i - 2 + \frac{1}{m_2(H_2)}} > \frac{\sum_{i=2}^{k} d_i}{\sum_{i=2}^{k} u_i - 2 + \frac{1}{m_2(H_2)}}.$$

Since $m_2(H_2) > 1$, by Fact 4.1(i) we get that

$$\frac{\sum_{i=1}^{k} d_i}{\sum_{i=1}^{k} u_i - 2 + \frac{1}{m_2(H_2)}} \geq \frac{\sum_{i=1}^{k} d_i}{\sum_{i=1}^{k} u_i - 4 + \frac{2}{m_2(H_2)}} > \frac{\sum_{i=1}^{k} d_i}{\sum_{i=1}^{k} u_i - 2 + \frac{1}{m_2(H_2)}},$$

a contradiction.

Assume H_1 is not 2-connected. Then there exists a cut vertex[9] $v \in V(H_1)$. Further, using Proposition 1.7 alongside that H_1 is strictly balanced with respect to $d_2(\cdot, H_2)$ and $m_2(H_1) > m_2(H_2) > 1$, we can show that H_1 does not contain any vertex of degree 1. Indeed, otherwise $\frac{e_1 - 1}{v_1 - 3 + \frac{1}{m_2(H_2)}} > \frac{e_1}{v_1 - 2 + \frac{1}{m_2(H_2)}} = m_2(H_1, H_2)$, contradicting that H_1 is strictly balanced with respect to $d_2(\cdot, H_2)$. Thus there exist subgraphs J_1 and J_2 of H_1 such that $|E(J_1)|, |E(J_2)| \geq 1$, $J_1 \cup J_2 = H_1$ and $V(J_1) \cap V(J_2) = \{v\}$. Using Fact 4.1(i) and that H_1 is strictly balanced with respect to $d_2(\cdot, H_2)$, we have that

$$\begin{aligned} e_1 = e_{J_1} + e_{J_2} &< m_2(H_1, H_2)\left(v_{J_1} - 2 + \frac{1}{m_2(H_2)} + v_{J_2} - 2 + \frac{1}{m_2(H_2)}\right) \\ &= m_2(H_1, H_2)\left(v_1 - 3 + \frac{2}{m_2(H_2)}\right). \end{aligned}$$

However, since $m_2(H_2) > 1$ we also have that

$$\frac{e_1}{v_1 - 3 + \frac{2}{m_2(H_2)}} > \frac{e_1}{v_1 - 2 + \frac{1}{m_2(H_2)}} = m_2(H_1, H_2),$$

contradicting the inequality above. Hence H_1 is 2-connected. □

5. Algorithm for computing valid edge-colourings: Asym-Edge-Col

To prove Theorem 1.9, we can clearly assume H_1 and H_2 are non-empty graphs satisfying the criteria of Conjecture 1.8 and that Conjecture 1.8 itself holds. Suppose $G = G_{n,p}$ and $p \leq bn^{-1/m_2(H_1,H_2)}$ where b will be a small constant defined later. As noted earlier, to prove Conjecture 1.3 we can show that a.a.s. G has a valid edge-colouring for H_1 and H_2. We construct our valid edge-colouring using an algorithm Asym-Edge-Col (see Figure 3). In order to state the algorithm succinctly, we need to define a considerable amount of notation, almost all of which we keep very similar to that in [14].

[9] That is, removing v and its incident edges from H_1 produces a disconnected graph.

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

```
1: procedure ASYM-EDGE-COL(G = (V, E))
2:     s ← EMPTY-STACK()
3:     E' ← E
4:     L ← L_G
5:     while G' = (V, E') is not (H_1, H_2)-sparse or not an Â-graph do
6:         if ∃e ∈ E' s.t. ∄(L, R) ∈ L × R_{G'} : E(L) ∩ E(R) = {e} then
7:             for all L ∈ L : e ∈ E(L) do
8:                 s.PUSH(L)
9:                 L.REMOVE(L)
10:            end for
11:            s.PUSH(e)
12:            E'.REMOVE(e)
13:        else
14:            if ∃L ∈ L : ∃e ∈ E(L) s.t. ∄R ∈ R_{G'} with E(L) ∩ E(R) = {e} then
15:                s.PUSH(L)
16:                L.REMOVE(L)
17:            else
18:                error "stuck"
19:            end if
20:        end if
21:    end while
22:    A-COLOUR(G' = (V, E'))
23:    while s ≠ ∅ do
24:        if s.TOP() is an edge then
25:            e ← s.POP()
26:            E'.ADD(e)
27:            e.SET-COLOUR(blue)
28:        else
29:            L ← s.POP()
30:            if L is entirely blue then
31:                f ← any e ∈ E(L) s.t. ∄R ∈ R_{G'} : E(L) ∩ E(R) = {e}
32:                f.SET-COLOUR(red)
33:            end if
34:        end if
35:    end while
36: end procedure
```

Figure 3. The implementation of algorithm Asym-Edge-Col.

Recall Definition 1.5. In particular, recall that for any graph G we define the families

$$\mathcal{R}_G := \{R \subseteq G : R \cong H_1\} \text{ and } \mathcal{L}_G := \{L \subseteq G : L \cong H_2\}$$

of all copies of H_1 and H_2 in G, respectively. Also, recall the family

$$\mathcal{L}_G^* := \{L \in \mathcal{L}_G : \forall e \in E(L)\ \exists R \in \mathcal{R}_G \text{ s.t. } E(L) \cap E(R) = \{e\}\}.$$

We highlight here that if $E(L) \cap E(R) = \{e\}$ for some $L \in \mathcal{L}_G$ and $R \in \mathcal{R}_G$ then it is still possible that $|V(L) \cap V(R)| > 2$.

Recall Definition 1.6. Intuitively, the graphs in $\hat{\mathcal{A}}$ are the building blocks of the graphs $\hat{G}$ which may remain after the edge deletion process in Asym-Edge-Col (described later).

For any graph G, define

$$\mathcal{S}_G := \{S \subseteq G : S \cong A \in \hat{\mathcal{A}} \wedge \nexists S' \supset S \text{ with } S' \subseteq G,\ S' \cong A' \in \hat{\mathcal{A}}\},$$

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

that is, the family $\mathcal{S}_G$ contains all maximal subgraphs of G isomorphic to a member of $\hat{\mathcal{A}}$. Hence, there are no two members $S_1, S_2 \in \mathcal{S}_G$ such that $S_1 \subsetneq S_2$. For any edge $e \in E(G)$, let

$$\mathcal{S}_G(e) := \{S \in \mathcal{S}_G : e \in E(S)\}.$$

Definition 5.1. We call G an $\hat{\mathcal{A}}$-graph if, for all $e \in E(G)$, we have

$$|\mathcal{S}_G(e)| = 1.$$

In particular, an $\hat{\mathcal{A}}$-graph is an edge-disjoint union of graphs from $\hat{\mathcal{A}}$. In an $\hat{\mathcal{A}}$-graph G, a copy of H_1 or H_2 can be a subgraph of G in two particular ways: either it is a subgraph of an $S \in \mathcal{S}_G$ or it is a subgraph with edges in at least two different graphs from $\mathcal{S}_G$. The former we call *trivial* copies of H_1 and H_2, and we define

$$\mathcal{T}_G := \left\{ T \subseteq G : (T \cong H_1 \vee T \cong H_2) \wedge \left| \bigcup_{e \in E(T)} \mathcal{S}_G(e) \right| \geq 2 \right\}$$

to be the family of all *non-trivial* copies of H_1 and H_2 in G.

Definition 5.2. We say that a graph G is (H_1, H_2)-sparse if $\mathcal{T}_G = \emptyset$.

Our next lemma asserts that (H_1, H_2)-sparse $\hat{\mathcal{A}}$-graphs are easily colourable, provided Conjecture 1.8 holds.

Lemma 5.3. *There exists a procedure A-Colour that returns for any (H_1, H_2)-sparse $\hat{\mathcal{A}}$-graph G a valid edge-colouring for H_1 and H_2.*

Proof. By Conjecture 1.8, there exists a valid edge-colouring for H_1 and H_2 of every $A \in \hat{\mathcal{A}}$. Using this we define a procedure A-Colour(G) as follows: Assign a valid edge-colouring for H_1 and H_2 to every subgraph $S \in \mathcal{S}_G$ locally, that is, regardless of the structure of G. Since G is an (H_1, H_2)-sparse $\hat{\mathcal{A}}$-graph, we assign a colour to each edge of G without producing a red copy of H_1 or a blue copy of H_2, and the resulting colouring is a valid edge-colouring for H_1 and H_2 of G. □

Note that we did not use that $\hat{\mathcal{A}}$ is finite, as given by Conjecture 1.8, in our proof of Lemma 5.3, only that 'every graph in $\hat{\mathcal{A}}$ has a valid edge-colouring for H_1 and H_2'. The finiteness of $\hat{\mathcal{A}}$ will be essential later for the proofs of Claims 6.6 and 7.6.

Now let us describe the algorithm Asym-Edge-Col which if successful outputs a valid edge-colouring of G. In Asym-Edge-Col, edges are removed from and then inserted back into a working copy $G' = (V, E')$ of G. Each edge is removed in the first while-loop only when it is not the unique intersection of the edge sets of some copy of H_1 and some copy of H_2 in G' (line 6). It is then 'pushed'[10] onto a stack s such that when we reinsert edges (in reverse order) in the second while-loop we can colour them to construct a valid edge-colouring for H_1 and H_2 of G; if at any point G' is an (H_1, H_2)-sparse $\hat{\mathcal{A}}$-graph, then we combine the colouring of these edges with a valid edge-colouring for H_1 and H_2 of G' provided by A-Colour. We also keep track of the copies of H_2 in G and push abstract representations of some of them (or all of them if G' is never an (H_1, H_2)-sparse $\hat{\mathcal{A}}$-graph during Asym-Edge-Col) onto s (lines 8 and 15) to be used later in the colour swapping stage of the second while-loop (lines 30-32).

Let us consider algorithm Asym-Edge-Col in detail. In line 5, we check whether G' is an (H_1, H_2)-sparse $\hat{\mathcal{A}}$-graph or not. If not, then we enter the first while-loop. In line 6, we choose an edge e which is not the unique intersection of the edge sets of some copy of H_1 and some copy

[10] For clarity, by 'push' we mean that the object is placed on the top of the stack s.

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

of H_2 in G' (if such an edge e exists). Then in lines 7-12 we push each copy of H_2 in G' that contains e onto s before pushing e onto s as well. Now, if every edge $e \in E'$ is the unique intersection of the edge sets of some copy of H_1 and some copy of H_2 in G', then we push onto s a copy L of H_2 in G' which contains an edge that is not the unique intersection of the edge set of L and the edge set of some copy of H_1 in G'. If no such copies L of H_2 exist, then the algorithm has an error in line 18. If Asym-Edge-Col does not run into an error, then we enter the second while-loop with input G'. Observe that G' is either the empty graph on vertex set V or some (H_1, H_2)-sparse $\hat{\mathcal{A}}$-graph. By Lemma 5.3, G' has a valid edge-colouring for H_1 and H_2. The second while-loop successively removes edges (line 25) and copies of L (line 29) from s in the reverse order in which they were added onto s, with the edges added back into E'. Each time an edge is added back it is coloured blue, and if a monochromatic blue copy L of H_2 is constructed, we make one of the edges of L red (lines 30-32). This colouring process is then repeated until we have a valid edge-colouring for H_1 and H_2 of G.

The following lemma confirms that our colouring process in the second while-loop produces a valid edge-colouring for H_1 and H_2 of G.

Lemma 5.4. *Algorithm Asym-Edge-Col either terminates with an error in line 18 or finds a valid edge-colouring for H_1 and H_2 of G.*

Proof. Our proof is almost identical to the proof of Lemma 11 in [14]. We include it here for completeness.

Let G^* denote the argument in the call to A-Colour in line 22. By Lemma 5.3, there is a valid edge-colouring for H_1 and H_2 of G^*. It remains to show that no forbidden monochromatic copies of H_1 or H_2 are created when this colouring is extended to a colouring of G in lines 23-35.

Firstly, we argue that the algorithm never creates a blue copy of H_2. Observe that *every* copy of H_2 that does not lie entirely in G^* is pushed on the stack in the first while-loop (lines 5-21). Therefore, in the execution of the second loop, the algorithm checks the colouring of every such copy. By the order of the elements on the stack, each such test is performed only after all edges of the corresponding copy of H_2 were inserted and coloured. For every blue copy of H_2, one particular edge f (see line 31) is recoloured to red. Since red edges are never flipped back to blue, no blue copy of H_2 can occur.

We need to show that the edge f in line 31 always exists. Since the second loop inserts edges into G' in the reverse order in which they were deleted during the first loop, when we select f in line 31, G' has the same structure as at the time when L was pushed on the stack. This happened either in line 8 when there exists no copy of H_1 in G' whose edge set intersects with L on some particular edge $e \in E(L)$, or in line 15 when L is not in $\mathcal{L}^*_{G'}$ due to the if-clause in line 14. In both cases we have $L \notin \mathcal{L}^*_{G'}$, and hence there exists an edge $e \in E(L)$ such that the edge sets of all copies of H_1 in G' do not intersect with L exactly in e.

It remains to prove that changing the colour of some edges from blue to red by the algorithm never creates an entirely red copy of H_1. By the condition on f in line 31 of the algorithm, at the moment f is recoloured there exists no copy of H_1 in G' whose edge set intersects L exactly in f. So there is either no copy of H_1 containing f at all, or every such copy contains also another edge from L. In the latter case, those copies cannot become entirely red since L is entirely blue. □

To prove Theorem 1.9, it now suffices to prove the following lemma.

Lemma 5.5. *There exists a constant $b = b(H_1, H_2) > 0$ such that for $p \leq bn^{-1/m_2(H_1,H_2)}$ algorithm Asym-Edge-Col terminates on $G_{n,p}$ without error a.a.s.*

We split our proof of Lemma 5.5 into two cases: (1) when $m_2(H_1) > m_2(H_2)$ and (2) when $m_2(H_1) = m_2(H_2)$. Notice that this accords with our definition of $\hat{\mathcal{A}}$.

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

6. Case 1: $m_2(H_1) > m_2(H_2)$.

We will prove Case 1 of Lemma 5.5 using an auxiliary algorithm Grow (see Figure 4). If Asym-Edge-Col has an error, then Grow computes a subgraph $F \subseteq G$ which is either too large in size or too dense to appear in $G_{n,p}$ a.a.s. (with p as in Lemma 5.5). Indeed, letting $\mathcal{F}$ be the class of all graphs that can possibly be returned by Grow, we will show that the expected number of copies of graphs from $\mathcal{F}$ contained in $G_{n,p}$ is $o(1)$, which with Markov's inequality implies that $G_{n,p}$ a.a.s. contains no graph from $\mathcal{F}$. This in turn implies Lemma 5.5 by contradiction. Note that algorithm Grow is only used for proving Lemma 5.5 and hence does not add anything on to the run-time of Asym-Edge-Col.

```
1: procedure GROW(G' = (V, E))
2:     if ∀e ∈ E : |S_{G'}(e)| = 1 then
3:         T ← any member of T_{G'}
4:         return ⋃_{e∈E(T)} S_{G'}(e)
5:     end if
6:     if ∃e ∈ E : |S_{G'}(e)| ≥ 2 then
7:         S_1, S_2 ← any two distinct members of S_{G'}(e)
8:         return S_1 ∪ S_2
9:     end if
10:    e ← any e ∈ E : |S_{G'}(e)| = 0
11:    F_0 ← any R ∈ R_{G'} : e ∈ E(R)
12:    i ← 0
13:    while (i < ln(n)) ∧ (∀F̃ ⊆ F_i : λ(F̃) > −γ) do
14:        if ∃R ∈ R_{G'} \ R_{F_i} : |V(R) ∩ V(F_i)| ≥ 2 then
15:            F_{i+1} ← F_i ∪ R
16:        else
17:            e ← ELIGIBLE-EDGE(F_i)
18:            F_{i+1} ← EXTEND-L(F_i, e, G')
19:        end if
20:        i ← i + 1
21:    end while
22:    if i ≥ ln(n) then
23:        return F_i
24:    else
25:        return MINIMISING-SUBGRAPH(F_i)
26:    end if
27: end procedure
```

```
1: procedure EXTEND-L(F, e, G')
2:     L ← any L ∈ L*_{G'}: e ∈ E(L)
3:     F' ← F ∪ L
4:     for all e' ∈ E(L) \ E(F) do
5:         R_{e'} ← any R ∈ R_{G'} : E(L) ∩ E(R) = {e'}
6:         F' ← F' ∪ R_{e'}
7:     end for
8:     return F'
9: end procedure
```

Figure 4. The implementation of algorithm Grow.

To state Grow we require the following definitions. Let

$$\gamma = \gamma(H_1, H_2) := \frac{1}{m_2(H_1, H_2)} - \frac{1}{m_2(H_1, H_2) + \varepsilon(H_1, H_2)} > 0, \tag{1}$$

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

where $\varepsilon(H_1, H_2)$ is the constant in Conjecture 1.8. Recall that for any graph F, we have

$$\lambda(F) := v(F) - \frac{e(F)}{m_2(H_1, H_2)}$$

and that this definition is motivated by the fact that the expected number of copies of F in $G_{n,p}$ with $p = bn^{-1/m_2(H_1,H_2)}$ has order of magnitude

$$n^{v(F)}p^{e(F)} = b^{e(F)}n^{\lambda(F)}.$$

Also, recall that

$$\mathcal{T}_G := \left\{ T \subseteq G : (T \cong H_1 \vee T \cong H_2) \wedge \left| \bigcup_{e \in E(T)} \mathcal{S}_G(e) \right| \geq 2 \right\}.$$

For any graph F and edge $e \in E(F)$, we say that e is *eligible for extension in Grow* if it satisfies

$$\nexists L \in \mathcal{L}_F^* \text{ s.t. } e \in E(L),$$

and observe that F is in $\mathcal{C}^*$ (see Definition 1.5) if and only if it contains no edge that is eligible for extension in Grow.

Algorithm Grow has as input the graph $G' \subseteq G$ that Asym-Edge-Col got stuck on. Let us consider the properties of G' when Asym-Edge-Col got stuck. Because the condition in line 6 of Asym-Edge-Col fails, G' is in the family $\mathcal{C}$, where we recall

$$\mathcal{C} = \mathcal{C}(H_1, H_2) := \{G = (V, E) : \forall e \in E \; \exists (L, R) \in \mathcal{L}_G \times \mathcal{R}_G \text{ s.t. } E(L) \cap E(R) = \{e\}\}.$$

In particular, every edge of G' is contained in a copy $L \in \mathcal{L}_{G'}$ of H_2, and, because the condition in line 14 fails, we can assume in addition that L belongs to $\mathcal{L}_{G'}^*$. Hence, G' is actually in the family $\mathcal{C}^* = \mathcal{C}^*(H_1, H_2)$ where we recall

$$\mathcal{C}^* = \mathcal{C}^*(H_1, H_2) := \{G = (V, E) : \forall e \in E \; \exists L \in \mathcal{L}_G^* \text{ s.t. } e \in E(L)\}.$$

Lastly, G' is not (H_1, H_2)-sparse or not an $\hat{\mathcal{A}}$-graph because Asym-Edge-Col ended with an error.

We now outline algorithm Grow. Firstly, Grow checks whether either of two special cases occur (lines 2-9). The first case corresponds to when G' is an $\hat{\mathcal{A}}$-graph, which is not (H_1, H_2)-sparse as it is a graph that Asym-Edge-Col got stuck on. The second case happens if there are 2 graphs in $\mathcal{S}_{G'}(e)$ that overlap in (at least) the edge e. The outputs of these two special cases are graphs which the while-loop of Grow could get stuck on. Indeed, if neither of these cases happen, then in line 10 we can choose an edge e that does not belong to any graph in $\mathcal{S}_G$. Crucially, algorithm Grow chooses a graph $R \in \mathcal{R}_{G'}$ which contains such an edge e and makes it the seed F_0 for a growing procedure (line 11). This choice of F_0 will allow us to conclude later that there always exists an edge eligible for extension in Grow (see the proof of Claim 6.1), that is, the while-loop of Grow operates as desired and doesn't get stuck.

In each iteration i of the while-loop, the growing procedure extends F_i to F_{i+1} in one of two ways. The first (lines 14-15) is by attaching a copy of H_1 in G' that intersects F_i in at least two vertices but is not contained in F_i. The second is more involved and begins with calling a function Eligible-Edge which maps F_i to an edge $e \in E(F_i)$ which is eligible for extension in Grow (we will show that such an edge always exists). Importantly, Eligible-Edge selects this edge e to be *unique up to isomorphism of* F_i, that is, for any two isomorphic graphs F and F', there exists an isomorphism ϕ with $\phi(F) = F'$ such that

$$\phi(\text{ELIGIBLE-EDGE}(F)) = \text{ELIGIBLE-EDGE}(F').$$

In particular, our choice of e depends only on F_i and not on the surrounding graph G' or any previous graph F_j with $j < i$ (indeed, there may be many ways that Grow could construct

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

a graph isomorphic to F_i). One could implement Eligible-Edge by having an enormous table of representatives for all isomorphism classes of graphs with up to n vertices. Since we do not care about complexity here, and only want to show the existence of certain structures in G', the time Eligible-Edge would take to be implemented is unimportant. What is important is that Eligible-Edge does not itself increase the number of graphs F that Grow can output.

Once we have our edge $e \in E(F_i)$ eligible for extension in Grow, we apply a procedure called Extend-L which attaches a graph $L \in \mathcal{L}^*_{G'}$ that contains e to F_i (line 18). We then attach to each new edge $e' \in E(L) \setminus E(F_i)$ a graph $R_{e'} \in \mathcal{R}_{G'}$ such that $E(L) \cap E(R_{e'}) = \{e'\}$ (lines 4-6 of Extend-L). (We will show later that such a graph L and graphs $R_{e'}$ exist and that $E(L) \setminus E(F_i)$ is non-empty.) The algorithm comes to an end when either $i \geq \ln(n)$ or $\lambda(\tilde{F}) \leq -\gamma$ for some subgraph $\tilde{F} \subseteq F_i$. In the former, the algorithm returns F_i (line 23); in the latter, the algorithm returns a subgraph $\tilde{F} \subseteq F_i$ that minimises $\lambda(\tilde{F})$ (line 25). For each graph F, the function Minimising Subgraph(F) returns such a minimising subgraph that is *unique up to isomorphism*. Once again, this is to ensure that Minimising Subgraph(F) does not itself artificially increase the number of graphs that Grow can output. As with function Eligible-Edge, one could implement Minimising Subgraph using an enormous look-up table.

We will now argue that Grow terminates without error, that is, Eligible-Edge always finds an edge eligible for extension in Grow and all 'any'-assignments in Grow and Extend-L are always successful.

Claim 6.1. *Algorithm Grow terminates without error on any input graph $G' \in \mathcal{C}^*$ that is not (H_1, H_2)-sparse or not an $\hat{\mathcal{A}}$-graph.*[11] *Moreover, for every iteration i of the while-loop, we have $e(F_{i+1}) > e(F_i)$.*

Proof. Our proof is very similar to the proof of Claim 13 in [14].

We first show that the special cases in lines 2-9 always function as desired. The first case occurs if and only if G' is an $\hat{\mathcal{A}}$-graph. By assumption, G' is not (H_1, H_2)-sparse, hence the family $\mathcal{T}_{G'}$ is not empty. Hence the assignment in line 3 is successful. Clearly, the assignment in line 7 is always successful due to the if-condition in line 6.

One can also easily see that the assignments in lines 10 and 11 are successful. Indeed, neither of the two special cases occur so we must have an edge $e \in E$ that is not contained in any $S \in \mathcal{S}_{G'}$. Also, there must exist a member of $\mathcal{R}_{G'}$ that contains e because G' is a member of $\mathcal{C}^* \subseteq \mathcal{C}$.

Next, we show that the call to Eligible-Edge in line 17 is always successful. Recall (1) on page 16. Indeed, suppose for a contradiction that no edge in F_i is eligible for extension in Grow for some $i \geq 0$. Then every edge $e \in E(F_i)$ is in some $L \in \mathcal{L}^*_{F_i}$, by definition. Hence $F \in \mathcal{C}^*$. Recall that H_1 and H_2 satisfy the criteria of Conjecture 1.8. Hence H_2 is strictly 2-balanced, H_1 is strictly balanced with respect to $d_2(\cdot, H_2))$ and $m_2(H_1) \geq m_2(H_2) > 1$. Then, by Lemma 4.2, H_1 and H_2 are 2-connected, hence F_i is 2-connected by construction. However, our choice of F_0 in line 11 guarantees that F_i is not in $\hat{\mathcal{A}}$. Indeed, the edge e selected in line 10 satisfying $|\mathcal{S}_{G'}(e)| = 0$ is an edge of F_0 and $F_0 \subseteq F_i \subseteq G'$. Thus, by the definition of $\hat{\mathcal{A}}$ and that $m_2(H_1) > m_2(H_2)$, we have that $m(F_i) > m_2(H_1, H_2) + \varepsilon$. Thus, there exists a non-empty graph $\tilde{F} \subseteq F_i$ with $d(\tilde{F}) = m(F_i)$ such that

$$\begin{aligned}
\lambda(\tilde{F}) &= v(\tilde{F}) - \frac{e(\tilde{F})}{m_2(H_1, H_2)} \\
&= e(\tilde{F}) \left(\frac{1}{m(F_i)} - \frac{1}{m_2(H_1, H_2)} \right) \\
&< e(\tilde{F}) \left(\frac{1}{m_2(H_1, H_2) + \varepsilon} - \frac{1}{m_2(H_1, H_2)} \right) \\
&= -\gamma e(\tilde{F}) \leq -\gamma.
\end{aligned}$$

[11] *See Definitions 5.1 and 5.2.*

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

Thus Grow terminates in line 13 without calling Eligible-Edge, and so every call to Eligible-Edge is successful and returns an edge e. Since $G' \in \mathcal{C}^*$, the call to Extend-L(F_i, e, G') is also successful and thus there exist suitable graphs $L \in \mathcal{L}^*_{G'}$ with $e \in E(L)$ and $R_{e'}$ for each $e' \in E(L) \setminus E(F_i)$.

It remains to show that for every iteration i of the while-loop, we have $e(F_{i+1}) > e(F_i)$. Since a copy R of H_1 found in line 14 is a copy of H_1 in G' but not in F_i (and H_1 is connected), we must have that $F_{i+1} = F_i \cup R$ contains at least one more edge than F_i.

So assume lines 17 and 18 are called in iteration i and let e be the edge chosen in line 17 and L the subgraph selected in line 2 of Extend-L(F_i, e, G'). By the definition of $\mathcal{L}^*_{G'}$, for each $e' \in E(L)$ there exists $R_{e'} \in \mathcal{R}_{G'}$ such that $E(L) \cap E(R_{e'}) = \{e'\}$. If $|E(L) \setminus E(F_i)| > 0$, then $e(F_{i+1}) \geq e(F_i \cup L) > e(F_i)$. Otherwise, $L \subseteq F_i$. But since e is eligible for extension in Grow, we must have $L \notin \mathcal{L}^*_{F_i}$. Thus there exists $e' \in L$ such that $R_{e'} \in \mathcal{R}_{G'} \setminus \mathcal{R}_{F_i}$ and $|V(R_{e'}) \cap V(F_i)| \geq 2$, contradicting that lines 17 and 18 are called in iteration i. □

6.1 Proof of Lemma 5.5

We consider the evolution of F_i now in more detail. We call iteration i of the while-loop in algorithm Grow *non-degenerate* (See Figure 5) if all of the following hold:

- The condition in line 14 evaluates to false (and Extend-L is called);
- In line 3 of Extend-L, we have $V(F) \cap V(L) = e$;
- In every execution of line 6 of Extend-L, we have $V(F') \cap V(R_{e'}) = e'$.

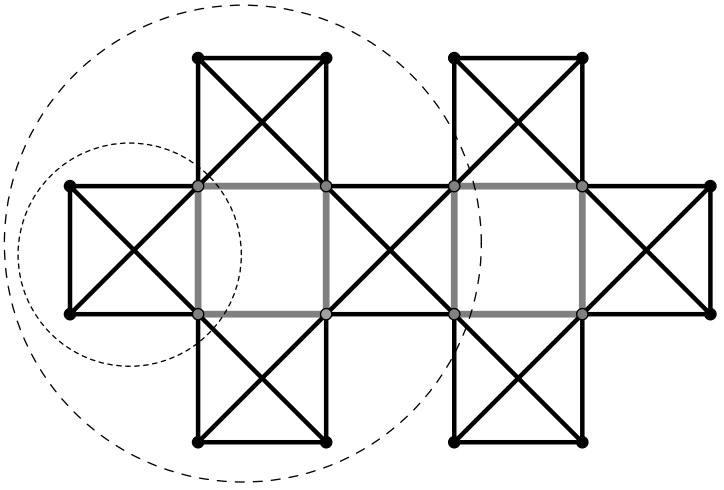

Figure 5. A graph F_2 resulting from two non-degenerate iterations for $H_1 = K_4$ and $H_2 = C_4$. The two central copies of H_2 are shaded.

Otherwise, we call iteration i *degenerate*. Note that, in non-degenerate iterations, there are only a *constant* number of graphs F_{i+1} that can result from any given F_i since Eligible-Edge determines the exact position where to attach the copy L of H_2, $V(F_i) \cap V(L) = e$ and for every execution of line 6 of Extend-L we have $V(F') \cap V(R_{e'}) = e'$ (recall that the edge e found by Eligible-Edge(F_i) is *unique up to isomorphism of* F_i).

Claim 6.2. *If iteration i of the while-loop in procedure Grow is non-degenerate, we have*

$$\lambda(F_{i+1}) = \lambda(F_i).$$

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

Proof. In a non-degenerate iteration we add $v_2 - 2$ vertices and $e_2 - 1$ edges for the copy of H_2 and then $(e_2 - 1)(v_1 - 2)$ new vertices and $(e_2 - 1)(e_1 - 1)$ new edges to complete the copies of H_1. This gives

$$\begin{aligned}\lambda(F_{i+1}) - \lambda(F_i) &= v_2 - 2 + (e_2 - 1)(v_1 - 2) - \frac{(e_2 - 1)e_1}{m_2(H_1, H_2)} \\ &= v_2 - 2 + (e_2 - 1)(v_1 - 2) - (e_2 - 1)\left(v_1 - 2 + \frac{1}{m_2(H_2)}\right) \\ &= 0,\end{aligned}$$

where we have used in the penultimate equality that H_1 is (strictly) balanced with respect to $d_2(\cdot, H_2)$ and in the final inequality that H_2 is (strictly) 2-balanced. □

When we have a degenerate iteration i, the structure of F_{i+1} may vary considerably and also depend on the structure of G'. Indeed, if F_i is extended by a copy R of H_1 in line 15, then R could intersect F_i in a multitude of ways. Moreover, there may be several copies of H_1 that satisfy the condition in line 14. The same is true for graphs added in lines 3 and 6 of Extend-L. Thus, degenerate iterations cause us difficulties since they enlarge the family of graphs algorithm Grow can return. However, we will show that at most a constant number of degenerate iterations can happen before algorithm Grow terminates, allowing us to bound from above sufficiently well the number of non-isomorphic graphs Grow can return. Pivotal in proving this is the following claim.

Claim 6.3. *There exists a constant $\kappa = \kappa(H_1, H_2) > 0$ such that if iteration i of the while-loop in procedure Grow is degenerate then we have*

$$\lambda(F_{i+1}) \leq \lambda(F_i) - \kappa.$$

We prove Claim 6.3 in Section 6.2. Together, Claims 6.2 and 6.3 yield the following claim.

Claim 6.4. *There exists a constant $q_1 = q_1(H_1, H_2)$ such that algorithm Grow performs at most q_1 degenerate iterations before it terminates, regardless of the input instance G'.*

Proof. By Claim 6.2, the value of the function λ remains the same in every non-degenerate iteration of the while-loop of algorithm Grow. However, Claim 6.3 yields a constant κ, which depends solely on H_1 and H_2, such that

$$\lambda(F_{i+1}) \leq \lambda(F_i) - \kappa$$

for every degenerate iteration i.

Hence, after at most

$$q_1 := \frac{\lambda(F_0) + \gamma}{\kappa}$$

degenerate iterations, we have $\lambda(F_i) \leq -\gamma$, and algorithm Grow terminates. □

For $0 \leq d \leq t < \lceil \ln(n) \rceil$, let $\mathcal{F}(t, d)$ denote a family of representatives for the isomorphism classes of all graphs F_t that algorithm Grow can possibly generate after exactly t iterations of the while-loop with exactly d of those t iterations being degenerate. Let $f(t, d) := |\mathcal{F}(t, d)|$.

Claim 6.5. *There exist constants $C_0 = C_0(H_1, H_2)$ and $A = A(H_1, H_2)$ such that*

$$f(t, d) \leq \lceil \ln(n) \rceil^{(C_0+1)d} \cdot A^{t-d}$$

for n sufficiently large.

Proof. By Claim 6.1, in every iteration i of the while-loop of Grow, we add new edges onto F_i. These new edges span a graph on at most

$$K := v_2 + (e_2 - 1)(v_1 - 2)$$

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

vertices. Thus $v(F_t) \leq v_1 + Kt$. Let $\mathcal{G}_K$ denote the set of all graphs on at most K vertices. In iteration i of the while-loop, F_{i+1} is uniquely defined if one specifies the graph $G \in \mathcal{G}_K$ with edges $E(F_{i+1}) \setminus E(F_i)$, the number y of vertices in which G intersects F_i, and two ordered lists of vertices from G and F_i respectively of length y, which specify the mapping of the intersection vertices from G onto F_i. Thus, the number of ways that F_i can be extended to F_{i+1} is bounded from above by

$$\sum_{G \in \mathcal{G}_K} \sum_{y=2}^{v(G)} v(G)^y v(F_i)^y \leq |\mathcal{G}_K| \cdot K \cdot K^K (v_1 + Kt)^K \leq \lceil \ln(n) \rceil^{C_0},$$

where C_0 depends only on v_1, v_2 and e_2, and n is sufficiently large. The last inequality follows from the fact that $t < \ln(n)$ as otherwise the while-loop would have already ended.

Recall that, since Eligible-Edge determines the exact position where to attach the copy of H_2, in non-degenerate iterations i there are at most

$$2e_2(2e_1)^{e_2-1} =: A$$

ways to extend F_i to F_{i+1}, where the coefficients of 2 correspond with the orientations of the edge of the copy of H_2 we attach to F_i and the edges of the copies of H_1 we attach to said copy of H_2. Hence, for $0 \leq d \leq t < \lceil \ln(n) \rceil$,

$$f(t,d) \leq \binom{t}{d} (\lceil \ln(n) \rceil^{C_0})^d \cdot A^{t-d} \leq \lceil \ln(n) \rceil^{(C_0+1)d} \cdot A^{t-d},$$

where the binomial coefficient corresponds to the choice of when in the t iterations the d degenerate iterations happen. □

A reader of [14] may observe that Claim 6.5 is not analogous to Claim 17 in [14]. Indeed, since we have a constant number of non-degenerate iterations, instead of a unique non-degenerate iteration as in [14], we truncated the proof of Claim 17 in order to have the appropriate bound to prove the following claim. Let $\mathcal{F} = \mathcal{F}(H_1, H_2, n)$ be a family of representatives for the isomorphism classes of *all* graphs that can be outputted by Grow (whether Grow enters the while-loop or not). Note that the proof of the following claim requires Conjecture 1.8 to be true; in particular, we need that $\hat{\mathcal{A}}(H_1, H_2, \varepsilon)$ is finite when $m_2(H_1) > m_2(H_2)$.

Claim 6.6. *There exists a constant $b = b(H_1, H_2) > 0$ such that for all $p \leq bn^{-1/m_2(H_1,H_2)}$, $G_{n,p}$ does not contain any graph from $\mathcal{F}(H_1, H_2, n)$ a.a.s.*

Proof. We first consider the two special cases in lines 2-9 of Grow. Let $\mathcal{F}_0 = \mathcal{F}_0(H_1, H_2) \subseteq \mathcal{F}$ denote the class of graphs that can be outputted by Grow if one of these two cases happens. We can see that any $F \in \mathcal{F}_0$ is either of the form

$$F = \bigcup_{e \in E(T)} \mathcal{S}_{G'}(e),$$

for some graph $T \in \mathcal{T}_{G'}$, or of the form

$$F = S_1 \cup S_2,$$

for some edge-intersecting $S_1, S_2 \in \mathcal{S}_{G'}$. Whichever of these forms F has, since every element of $\mathcal{S}_{G'}$ is 2-connected and in $\mathcal{C}^*$, and T is 2-connected[12], we have that F is 2-connected and in $\mathcal{C}^*$. On the other hand, $F \subseteq G'$ is not in $\mathcal{S}_{G'}$ and thus not isomorphic to a graph in $\hat{\mathcal{A}}$. Indeed, otherwise the graphs S forming F would not be in $\mathcal{S}_{G'}$ due to the maximality condition in the definition of $\mathcal{S}_{G'}$. It follows that $m(F) > m_2(H_1, H_2) + \varepsilon(H_1, H_2)$. Since we assumed Conjecture 1.8 holds, the family $\mathcal{F}_0$ is finite. Hence Markov's inequality yields that $G_{n,p}$ contains no graph from $\mathcal{F}_0$ a.a.s.

[12] Since $T \cong H_1$ or $T \cong H_2$ and Lemma 4.2 holds.

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

Let $\tilde{\mathcal{F}} = \tilde{\mathcal{F}}(H_1, H_2, n)$ denote a family of representatives for the isomorphism classes of all graphs that can be the output of Grow with parameters n and $\gamma(H_1, H_2)$ on any input instance G' for which it enters the while-loop. Observe that $\mathcal{F} = \mathcal{F}_0 \cup \tilde{\mathcal{F}}$. Let $\mathcal{F}_1$ and $\mathcal{F}_2$ denote the classes of graphs that algorithm Grow can output in lines 23 and 25, respectively. For each $F \in \mathcal{F}_1$, we have that $e(F) \geq \ln(n)$, as F was generated in $\lceil \ln(n) \rceil$ iterations, each of which introduces at least one new edge by Claim 6.1. Moreover, Claims 6.2 and 6.3 imply that $\lambda(F_i)$ is non-increasing. Thus, we have that $\lambda(F) \leq \lambda(F_0)$ for all $F \in \mathcal{F}_1$. For all $F \in \mathcal{F}_2$, we have that $\lambda(F) \leq -\gamma$ due to the condition in line 13 of Grow. Let $A := A(H_1, H_2)$ be the constant found in the proof of Claim 6.5. Since we have chosen $F_0 \cong H_1$ as the seed of the growing procedure, it follows that for

$$b := (Ae)^{-\lambda(F_0)-\gamma} \leq 1,$$

the expected number of copies of graphs from $\tilde{\mathcal{F}}$ in $G_{n,p}$ with $p \leq bn^{-1/m_2(H_1,H_2)}$ is bounded by

$$\begin{aligned}
\sum_{F \in \tilde{\mathcal{F}}} n^{v(F)} p^{e(F)} &\leq \sum_{F \in \tilde{\mathcal{F}}} b^{e(F)} n^{\lambda(F)} \\
&\leq \sum_{F \in \mathcal{F}_1} (eA)^{(-\lambda(F_0)-\gamma)\ln(n)} n^{\lambda(F_0)} + \sum_{F \in \mathcal{F}_2} b^{e(F)} n^{-\gamma} \\
&= \sum_{F \in \mathcal{F}_1} A^{(-\lambda(F_0)-\gamma)\ln(n)} n^{-\gamma} + \sum_{F \in \mathcal{F}_2} b^{e(F)} n^{-\gamma}. \qquad (2)
\end{aligned}$$

Observe that, since $m_2(F_2) \geq 1$, we have that

$$\lambda(F_0) = v_1 - \frac{e_1}{m_2(F_1, F_2)} = 2 - \frac{1}{m_2(F_2)} \geq 1 \qquad (3)$$

By Claims 6.1, 6.4 and 6.5, and (3), we have that

$$\begin{aligned}
\sum_{F \in \mathcal{F}_1} A^{(-\lambda(F_0)-\gamma)\ln(n)} n^{-\gamma} &\leq \sum_{d=0}^{\min\{t,q_1\}} f(\lceil \ln(n) \rceil, d) A^{(-\lambda(F_0)-\gamma)\ln(n)} n^{-\gamma} \\
&\leq (q_1+1) \lceil \ln(n) \rceil^{(C_0+1)q_1} \cdot A^{\lceil \ln(n) \rceil} A^{(-\lambda(F_0)-\gamma)\ln(n)} n^{-\gamma} \\
&\leq (\ln(n))^{2(C_0+1)q_1} n^{-\gamma}. \qquad (4)
\end{aligned}$$

Observe that, by Claim 6.1, if some graph $F \in \mathcal{F}_2$ is the output of Grow after precisely t iterations of the while-loop then $e(F) \geq t$. Since $b < 1$, this implies

$$b^{e(F)} \leq b^t, \qquad (5)$$

for such a graph F. Using (5) and Claims 6.1, 6.4 and 6.5, we have that

$$\begin{aligned}
\sum_{F \in \mathcal{F}_2} b^{e(F)} n^{-\gamma} &\leq \sum_{t=0}^{\lceil \ln(n) \rceil} \sum_{d=0}^{\min\{t,q_1\}} f(t,d) b^t n^{-\gamma} \\
&\leq \sum_{t=0}^{\lceil \ln(n) \rceil} \sum_{d=0}^{\min\{t,q_1\}} \lceil \ln(n) \rceil^{(C_0+1)d} \cdot A^{t-d} (Ae)^{(-\lambda(F_0)-\gamma)t} n^{-\gamma} \\
&\leq (\lceil \ln(n) \rceil + 1)(q_1+1) \lceil \ln(n) \rceil^{(C_0+1)q_1} n^{-\gamma} \\
&\leq (\ln(n))^{2(C_0+1)q_1} n^{-\gamma}. \qquad (6)
\end{aligned}$$

Thus, by (2), (4) and (6), we have that $\sum_{F \in \tilde{\mathcal{F}}} n^{v(F)} p^{e(F)} = o(1)$. Consequently, Markov's inequality implies that $G_{n,p}$ a.a.s. contains no graph from $\tilde{\mathcal{F}}$.

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

Combined with the earlier observation that $G_{n,p}$ a.a.s. contains no graph from $\mathcal{F}_0$, we have that $G_{n,p}$ a.a.s. contains no graph from $\mathcal{F} = \mathcal{F}_0 \cup \tilde{\mathcal{F}}$. □

Proof of Lemma 5.5: Case 1. Suppose that the call to Asym-Edge-Col(G) gets stuck for some graph G, and consider $G' \subseteq G$ at this moment. Then Grow(G') returns a copy of a graph $F \in \mathcal{F}(H_1, H_2, n)$ that is contained in $G' \subseteq G$. Provided Claim 6.3 holds, by Claim 6.6 this event a.a.s. does not occur in $G = G_{n,p}$ with p as claimed. Thus Asym-Edge-Col does not get stuck a.a.s. and, by Lemma 5.4, finds a valid colouring for H_1 and H_2 of $G_{n,p}$ with $p \leq bn^{-1/m_2(H_1,H_2)}$ a.a.s. □

6.2 *Proof of Claim 6.3*

Our strategy for proving Claim 6.3 revolves around comparing our degenerate iteration i of the while-loop of algorithm Grow with any non-degenerate iteration which could have occurred instead. In accordance with this strategy, we have the following technical lemma which will be crucial in proving Claim 6.3.[13] The lemma will play the same role as Lemma 21 does in [14], but is considerably different. In order to state our technical lemma, we define the following families of graphs.

Definition 6.7. Let F, H_1 and H_2 be graphs and $\hat{e} \in E(F)$. We define $\mathcal{H}(F, \hat{e}, H_1, H_2)$ to be the family of graphs constructed from F in the following way: Attach a copy $H_{\hat{e}}$ of H_2 to F such that $E(H_{\hat{e}}) \cap E(F) = \{\hat{e}\}$ and $V(H_{\hat{e}}) \cap V(F) = \hat{e}$. Then, for each edge $f \in E(H_{\hat{e}}) \setminus \{\hat{e}\}$, attach a copy H_f of H_1 to $F \cup H_{\hat{e}}$ such that $E(F \cup H_{\hat{e}}) \cap E(H_f) = \{f\}$ and $(V(F) \setminus \hat{e}) \cap V(H_f) = \emptyset$.

Notice that, during construction of a graph $J \in \mathcal{H}(F, \hat{e}, H_1, H_2)$, the edge of $H_{\hat{e}}$ intersecting at $\hat{e}$ and the edge of each copy H_f of H_1 intersecting at an edge $f \in E(H_{\hat{e}}) \setminus \{\hat{e}\}$ are not stipulated. That is, we may end up with different graphs after the construction process if we choose different edges of $H_{\hat{e}}$ to intersect F at $\hat{e}$ and different edges of the copies H_f of H_1 to intersect the edges in $E(H_{\hat{e}}) \setminus \{\hat{e}\}$. Observe that although $E(F \cup H_{\hat{e}}) \cap E(H_f) = \{f\}$ and $(V(F) \setminus \hat{e}) \cap V(H_f) = \emptyset$ for each $f \in E(H_{\hat{e}}) - \{\hat{e}\}$, the construction may result in one or more graphs H_f intersecting $H_{\hat{e}}$ in more than two vertices, including possibly in vertices of $\hat{e}$ (e.g. H_{f_3} in Figure 6). Also, the graphs H_f may intersect with each other in vertices and/or edges (e.g. H_{f_1} and H_{f_2} in Figure 6).

Borrowing notation and language from [14], for any $J \in \mathcal{H}(F, \hat{e}, H_1, H_2)$ we call $V_J := V(H_{\hat{e}}) \setminus \hat{e}$ the *inner vertices of* J and $E_J := E(H_{\hat{e}}) \setminus \{\hat{e}\}$ the *inner edges of* J. Let $H^J_{\hat{e}}$ be the *inner graph* on vertex set $V_J \dot{\cup} \hat{e}$ and edge set E_J, and observe that this graph $H^J_{\hat{e}}$ is isomorphic to a copy of H_2 minus some edge. Further, for each copy H_f of H_1, we define $U_J(f) := V(H_f) \setminus f$ and $D_J(f) := E(H_f) \setminus \{f\}$ and call

$$U_J := \bigcup_{f \in E_J} U_J(f),$$

the set of *outer vertices* of J and

$$D_J := \bigcup_{f \in E_J} D_J(f),$$

the set of *outer edges* of J. Observe that the sets $U_J(f)$ may overlap with each other and, as noted earlier, with $V(H^J_{\hat{e}})$. However, the sets $D_J(f)$ may overlap only with each other. Further, define $\mathcal{H}^*(F, \hat{e}, H_1, H_2) \subseteq \mathcal{H}(F, \hat{e}, H_1, H_2)$ such that for any $J^* \in \mathcal{H}^*(F, \hat{e}, H_1, H_2)$ we have $U_{J^*}(f_1) \cap U_{J^*}(f_2) = \emptyset$ and $D_{J^*}(f_1) \cap D_{J^*}(f_2) = \emptyset$ for all $f_1, f_2 \in E_{J^*}$, $f_1 \neq f_2$, and $U_{J^*}(f) \cap V\left(H^{J^*}_{\hat{e}}\right) = \emptyset$ for all $f \in E_{J^*}$; that is, the copies of H_1 are, in some sense, pairwise disjoint. Note that each $J^* \in \mathcal{H}^*(F, \hat{e}, H_1, H_2)$ corresponds with a non-degenerate iteration i of the while-loop of algorithm

[13] More specifically, in proving Claim 6.10, stated later.

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

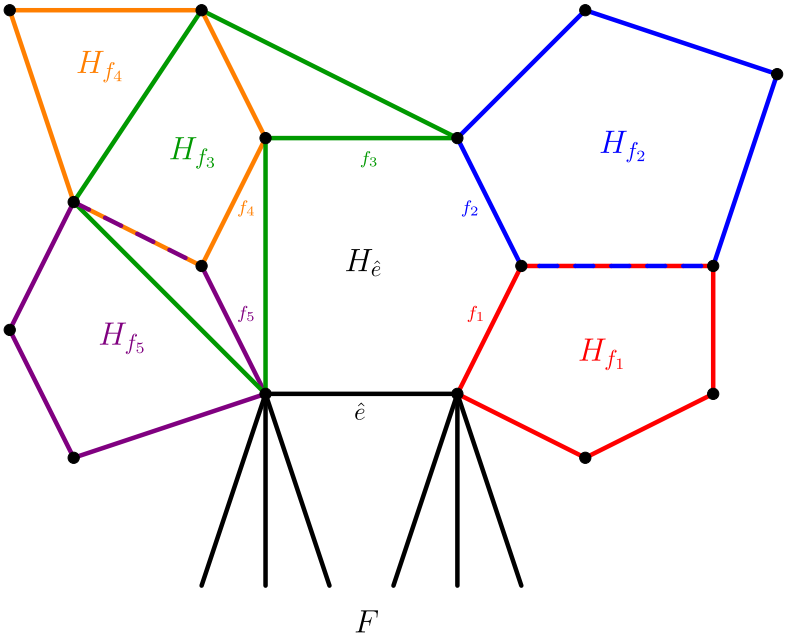

Figure 6. A graph $J \in \mathcal{H}(F, \hat{e}, C_5, C_6) \setminus \mathcal{H}^*(F, \hat{e}, C_5, C_6)$.

Grow when $F = F_i$, $J^* = F_{i+1}$ and $\hat{e}$ is the edge chosen by Eligible-Edge(F_i). This observation will be very helpful several times later. For any $J \in \mathcal{H}(F, \hat{e}, H_1, H_2)$, define

$$v^+(J) := |V(J) \setminus V(F)| = v(J) - v(F),$$

and

$$e^+(J) := |E(J) \setminus E(F)| = e(J) - e(F),$$

and call $\frac{e^+(J)}{v^+(J)}$ the *F-external density of J*. The following lemma relates the F-external density of any $J^* \in \mathcal{H}^*(F, \hat{e}, H_1, H_2)$ to that of any $J \in \mathcal{H}(F, \hat{e}, H_1, H_2) \setminus \mathcal{H}^*(F, \hat{e}, H_1, H_2)$.

Lemma 6.8. *Let F be a graph and $\hat{e} \in E(F)$. Then for any $J \in \mathcal{H}(F, \hat{e}, H_1, H_2) \setminus \mathcal{H}^*(F, \hat{e}, H_1, H_2)$ and any $J^* \in \mathcal{H}^*(F, \hat{e}, H_1, H_2)$, we have*

$$\frac{e^+(J)}{v^+(J)} > \frac{e^+(J^*)}{v^+(J^*)}.$$

We prove Lemma 6.8 in Section 6.3.

Claim 6.3 will follow from the next two claims. We say that algorithm Grow encounters a *degeneracy of type 1* in iteration i of the while-loop if line 14 returns true, that is, $\exists R \in \mathcal{R}_{G'} \setminus \mathcal{R}_{F_i} : |V(R) \cap V(F_i)| \geq 2$. Note that the following claim requires that $m_2(H_1) > m_2(H_2)$.

Claim 6.9. *There exists a constant $\kappa_1 = \kappa_1(H_1, H_2) > 0$ such that if procedure Grow encounters a degeneracy of type 1 in iteration i of the while-loop, we have*

$$\lambda(F_{i+1}) \leq \lambda(F_i) - \kappa_1.$$

Proof. Let $F := F_i$ be the graph before the operation in line 15 is carried out (that is, before $F_{i+1} \leftarrow F_i \cup R$), let R be the copy of H_1 merged with F in line 15 and let $F' := F_{i+1}$ be the output

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

from line 15. We aim to show there exists a constant $\kappa_1 = \kappa_1(H_1, H_2) > 0$ such that

$$\lambda(F) - \lambda(F') = v(F) - v(F') - \frac{e(F) - e(F')}{m_2(H_1, H_2)} \geq \kappa_1.$$

Choose any edge $\hat{e} \in E(F)$ (the edge $\hat{e}$ need not be in the intersection of R and F). Choose any $F^* \in \mathcal{H}^*(F, \hat{e}, H_1, H_2)$. Our strategy is to compare our degenerate outcome F' with F^*. As noted earlier, F^* corresponds to a non-degenerate iteration of the while-loop of algorithm Grow (if $\hat{e}$ was the edge chosen by Eligible-Edge). Then Claim 6.2 gives us that $\lambda(F) = \lambda(F^*)$. Then

$$\lambda(F) - \lambda(F') = \lambda(F^*) - \lambda(F') = v(F^*) - v(F') - \frac{e(F^*) - e(F')}{m_2(H_1, H_2)}.$$

Hence we aim to show that there exists $\kappa_1 = \kappa_1(H_1, H_2) > 0$ such that

$$v(F^*) - v(F') - \frac{e(F^*) - e(F')}{m_2(H_1, H_2)} \geq \kappa_1. \tag{7}$$

Define R' to be the graph with vertex set $V' := V(R) \cap V(F)$ and edge set $E' := E(R) \cap E(F)$, and let $v' := |V'|$ and $e' := |E'|$. Observe that $R' \subsetneq R$. Since F^* corresponds with a non-degenerate iteration of the while-loop of algorithm Grow, H_2 is (strictly) 2-balanced and H_1 is (strictly) balanced with respect to $d_2(\cdot, H_2)$, we have

$$\begin{aligned} v(F^*) - v(F') - \frac{e(F^*) - e(F')}{m_2(H_1, H_2)} &= (e_2 - 1)(v_1 - 2) + (v_2 - 2) - (v_1 - v') \\ &\quad - \frac{(e_2 - 1)e_1 - (e_1 - e')}{m_2(H_1, H_2)} \\ &= (e_2 - 1)(v_1 - 2) + (v_2 - 2) \\ &\quad - (e_2 - 1)\left(v_1 - 2 + \frac{1}{m_2(H_2)}\right) \\ &\quad + \frac{e_1 - e'}{m_2(H_1, H_2)} - (v_1 - v') \\ &= \frac{e_1 - e'}{m_2(H_1, H_2)} - (v_1 - v') \\ &= v' - 2 + \frac{1}{m_2(H_2)} - \frac{e'}{m_2(H_1, H_2)}. \end{aligned} \tag{8}$$

Also, since Grow encountered a degeneracy of type 1, we must have $v' \geq 2$. Hence, if $e' = 0$ and $v' \geq 2$, then

$$v' - 2 + \frac{1}{m_2(H_2)} - \frac{e'}{m_2(H_1, H_2)} \geq \frac{1}{m_2(H_2)} > 0.$$

If $e' \geq 1$, then since R is a copy of H_1, H_1 is *strictly* balanced with respect to $d_2(\cdot, H_2)$ and $R' \subsetneq R$ with $|E(R')| = e' \geq 1$, we have that $0 < d_2(R', H_2) < m_2(H_1, H_2)$, and so

$$-\frac{1}{m_2(H_1, H_2)} > -\frac{1}{d_2(R', H_2)}. \tag{9}$$

Then by (8) and (9), we have that

$$\begin{aligned} v(F^*) - v(F') - \frac{e(F^*) - e(F')}{m_2(H_1, H_2)} &= v' - 2 + \frac{1}{m_2(H_2)} - \frac{e'}{m_2(H_1, H_2)} \\ &> v' - 2 + \frac{1}{m_2(H_2)} - \frac{e'}{d_2(R', H_2)} = 0, \end{aligned}$$

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

using the definition of $d_2(R', H_2)$. Thus (7) holds for

$$\kappa_1 = \min_{R' \subsetneq R} \left\{ \frac{1}{m_2(H_2)},\ v' - 2 + \frac{1}{m_2(H_2)} - \frac{e'}{m_2(H_1, H_2)} \right\}.$$

□

We say that algorithm Grow encounters a *degeneracy of type 2* in iteration i of the while-loop if, when we call Extend-L(F_i, e, G'), the graph L found in line 2 overlaps with F_i in more than 2 vertices, or if there exists some edge $e' \in E(L) \setminus E(F_i)$ such that the graph $R_{e'}$ found in line 5 overlaps in more than 2 vertices with F'. The following result corresponds to Claim 22 in [14]. As in the proof of Claim 22 in [14], we transform F' into the output of a non-degenerate iteration F^* in three steps. However, we swap the order of the latter two steps in our proof. More precisely, we transform F' into a graph $F^2 \in \mathcal{H}(F_i, e, H_1, H_2)$ in the first two steps, then transform F^2 into a graph $F^3 := F^* \in \mathcal{H}^*(F_i, e, H_1, H_2)$. In this last step we require Lemma 6.8.

Claim 6.10. *There exists a constant $\kappa_2 = \kappa_2(H_1, H_2) > 0$ such that if procedure Grow encounters a degeneracy of type 2 in iteration i of the while-loop, we have*

$$\lambda(F_{i+1}) \leq \lambda(F_i) - \kappa_2.$$

Proof. Let $F := F_i$ be the graph passed to Extend-L and let $F' := F_{i+1}$ be its output. We aim to show that there exists a constant $\kappa_2 = \kappa_2(H_1, H_2) > 0$ such that

$$\lambda(F) - \lambda(F') = v(F) - v(F') - \frac{e(F) - e(F')}{m_2(H_1, H_2)} \geq \kappa_2. \tag{10}$$

Recall that F' would be one of a constant number of graphs if iteration i was non-degenerate. Our strategy is to transform F' into the output of such a non-degenerate iteration F^* in three steps

$$F' =: F^0 \overset{(i)}{\to} F^1 \overset{(ii)}{\to} F^2 \overset{(iii)}{\to} F^3 := F^*,$$

with each step carefully resolving a different facet of a degeneracy of type 2. By Claim 6.2, we have $\lambda(F) = \lambda(F^*)$, hence we have that

$$\begin{aligned}\lambda(F) - \lambda(F') = \lambda(F^*) - \lambda(F') &= \sum_{j=1}^{3} \left(\lambda(F^j) - \lambda(F^{j-1})\right) \\ &= \sum_{j=1}^{3} \left(v(F^j) - v(F^{j-1}) - \frac{e(F^j) - e(F^{j-1})}{m_2(H_1, H_2)} \right).\end{aligned}$$

We shall show that there exists $\kappa_2 = \kappa_2(H_1, H_2) > 0$ such that

$$\left(v(F^j) - v(F^{j-1}) - \frac{e(F^j) - e(F^{j-1})}{m_2(H_1, H_2)} \right) \geq \kappa_2, \tag{11}$$

for each $j \in \{1, 2, 3\}$, whenever F^j and F^{j-1} are not isomorphic. In each step we will look at a different structural property of F' that may result from a degeneracy of type 2. We do not know the exact structure of F', and so, for each j, step j may not necessarily modify F^{j-1}. However, since F' is not isomorphic to F^*, as F' resulted from a degeneracy of type 2, we know that for at least one j that F^j is not isomorphic to F^{j-1}. This will allow us to conclude (10) from (11).

We will now analyse the graph that Extend-L attaches to F when a degeneracy of type 2 occurs. First of all, Extend-L attaches a graph $L \cong H_2$ to F such that $L \in \mathcal{L}^*_{G'}$. Let x be the number of new vertices that are added onto F when L is attached, that is, $x = |V(L) \setminus (V(F) \cap V(L))|$. Since L overlaps with the edge e determined by Eligible-Edge in line 17 of Grow, we must have that $x \leq v_2 - 2$. Further, as $L \in \mathcal{L}^*_{G'}$, every edge of L is covered by a copy of H_1. Thus, since the

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

condition in line 14 of Grow came out as false in iteration i, we must have that

$$\text{for all } u, v \in V(F) \cap V(L), \text{ if } uv \in E(L) \text{ then } uv \in E(F). \tag{12}$$

(By Claim 6.1, (12) implies that $x \geq 1$ since F must be extended by at least one edge.)

Let $L' \subseteq L$ denote the subgraph of L obtained by removing every edge in $E(F) \cap E(L)$. Observe that $|V(L')| = |V(L)| = v_2$ and $|E(L')| \geq 1$ (see the remark above). Extend-L attaches to each edge $e' \in E(L')$ a copy $R_{e'}$ of H_1 in line 6 such that $E(L') \cap E(R_{e'}) = \{e'\}$. As the condition in line 14 of Grow came out as false, each graph $R_{e'}$ intersects F in at most one vertex and, hence, zero edges. Let

$$L_R{}' := L' \cup \bigcup_{e' \in E(L')} R_{e'}.$$

Then F' is the same as $F \cup L_R{}'$, and since every graph $R_{e'}$ contains at most one vertex of F, we have that $E(F') = E(F) \dot\cup E(L_R{}')$. Therefore,

$$e(F') - e(F) = e(L'_R).$$

Observe that $|V(F) \cap V(L')| = v_2 - x$ and so

$$\begin{aligned} v(F') - v(F) &= v(L'_R) - |V(F) \cap V(L'_R)| \\ &= v(L'_R) - (v_2 - x) - |V(F) \cap (V(L'_R) \setminus V(L'))|. \end{aligned}$$

Transformation (i): $F^0 \to F^1$. If $|V(F) \cap (V(L'_R) \setminus V(L'))| \geq 1$, then we apply transformation (i), mapping F^0 to F^1: For each vertex $v \in V(F) \cap (V(L'_R) \setminus V(L'))$, transformation (i) introduces a new vertex v'. Every edge incident to v in $E(F)$ remains connected to v and all those edges incident to v in $E(L'_R)$ are redirected to v'. In L'_R we replace the vertices in $V(F) \cap (V(L'_R) \setminus V(L'))$ with the new vertices. So now we have $|V(F) \cap (V(L'_R) \setminus V(L'))| = 0$. Since $E(F) \cap E(L'_R) = \emptyset$, the output of this transformation is uniquely defined. Moreover, the structure of L'_R is completely unchanged. Hence, since $|V(F) \cap (V(L'_R) \setminus V(L'))| \geq 1$, and $|E(F')| = |E(F) \dot\cup E(L'_R)|$ remained the same after transformation (i), we have that

$$v(F^1) - v(F^0) - \frac{e(F^1) - e(F^0)}{m_2(H_1, H_2)} = |V(F) \cap (V(L'_R) \setminus V(L'))| \geq 1.$$

Transformation (ii): $F^1 \to F^2$. Recall the definition of $\mathcal{H}(F, e, H_1, H_2)$. If $x \leq v_2 - 3$, then we apply transformation (ii), mapping F^1 to F^2 by replacing L'_R with a graph L''_R such that $F \cup L''_R \in \mathcal{H}(F, e, H_1, H_2)$.

If $x = v_2 - 2$, observe that already $F \cup L'_R \in \mathcal{H}(F, e, H_1, H_2)$ and we continue to transformation (iii). So assume $x \leq v_2 - 3$. Consider the proper subgraph $L_F := L[V(F) \cap V(L)] \subsetneq L$ obtained by removing all x vertices in $V(L) \setminus V(F)$ and their incident edges from L. Observe that $v(L_F) = v_2 - x \geq 3$ and also that $L_F \subseteq F$ by (12). Assign labels to $V(L_F)$ so that $V(L_F) = \{y, z, w_1, \ldots, w_{v_2-(x+2)}\}$ where $e = \{y, z\}$ and $w_1, \ldots, w_{v_2-(x+2)}$ are arbitrarily assigned. At the start of transformation (ii), we create $v_2 - (x+2)$ new vertices $w'_1, \ldots, w'_{v_2-(x+2)}$ and also new edges such that $\left\{y, z, w'_1, \ldots, w'_{v_2-(x+2)}\right\}$ induces a copy $\hat{L}_F$ of L_F, and for all $i, j \in \{1, \ldots, v_2 - (x+2)\}$, $i \neq j$,

$$\begin{aligned} &\text{if } w_i w_j \in E(L_F) \text{ then } w'_i w'_j \in E(\hat{L}_F); \\ &\text{if } w_i y \in E(L_F) \text{ then } w'_i y \in E(\hat{L}_F); \\ &\text{if } w_i z \in E(L_F) \text{ then } w'_i z \in E(\hat{L}_F); \\ &\text{and } e = yz \in E(\hat{L}_F). \end{aligned}$$

We also transform L'_R. For each edge in $E(L'_R)$ incident to a vertex w_i in L_F, redirect the edge to w'_i, and remove $w_1, \ldots, w_{v_2-(x+2)}$ from $V(L'_R)$. Hence the structure of L'_R remains the same

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

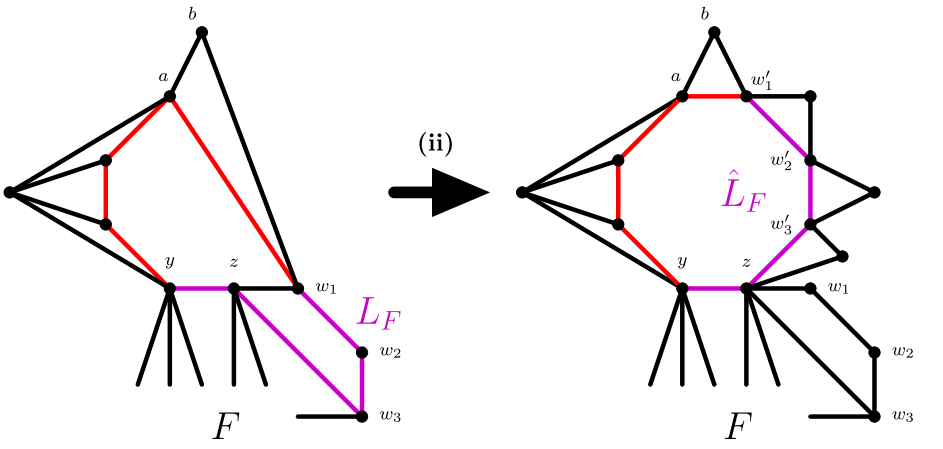

Figure 7. An example of transformation (ii) where $H_1 = K_3$ and $H_2 = C_8$. Observe that edges aw_1 and bw_1 are replaced by edges aw_1' and bw_1'.

except for the vertices $w_1, \ldots, w_{v_2-(x+2)}$ that we removed. Define $L'' := L'_R \cup \hat{L}_F$ and observe that $V(L'') \cap V(F) = e$.

Continuing transformation (ii), for each $e' \in E(\hat{L}_F) \setminus \{e\}$, attach a copy $R_{e'}$ of H_1 to L'' such that $E(R_{e'}) \cap E(L'' \cup F) = \{e'\}$ and $V(R_{e'}) \cap V(L'' \cup F) = e'$. That is, all these new copies $R_{e'}$ of H_1 are, in some sense, pairwise disjoint. Observe that $E(L'') \cap E(F) = \{e\}$ and define

$$L''_R := L'' \cup \bigcup_{e' \in E(\hat{L}_F)\setminus\{e\}} R_{e'}.$$

Then $F \cup L''_R \in \mathcal{H}(F, e, H_1, H_2)$. (See Figure 7 for an example of transformation (ii).) Let $F^2 := F \cup L''_R$. Then,

$$\begin{aligned}
&v(F^2) - v(F^1) - \frac{e(F^2) - e(F^1)}{m_2(H_1, H_2)} \\
&= (e(\hat{L}_F) - 1)(v_1 - 2) + v(\hat{L}_F) - 2 - \frac{(e(\hat{L}_F) - 1)e_1}{m_2(H_1, H_2)} \\
&= v(\hat{L}_F) - 2 - \frac{(e(\hat{L}_F) - 1)}{m_2(H_2)} \\
&= \frac{(v(\hat{L}_F) - 2)\left(m_2(H_2) - \frac{e(\hat{L}_F)-1}{v(\hat{L}_F)-2}\right)}{m_2(H_2)} \\
&\geq \delta_1
\end{aligned}$$

for some $\delta_1 = \delta_1(H_1, H_2) > 0$, where the second equality follows from H_1 being (strictly) balanced with respect to $d_2(\cdot, H_2)$, the third equality follows from $v(\hat{L}_F) = v(L_F) \geq 3$ and the last inequality follows from $\hat{L}_F$ being a copy of $L_F \subsetneq L \cong H_2$ and H_2 being *strictly* 2-balanced.

Transformation (iii): $F^2 \to F^3$. Recall that for any $J \in \mathcal{H}(F, \hat{e}, H_1, H_2)$, we define $v^+(J) := |V(J) \setminus V(F)| = v(J) - v(F)$ and $e^+(J) := |E(J) \setminus E(F)| = e(J) - e(F)$. Remove the edge e from $E(L'')$ (and $E(L''_R)$) to give $E(L'') \cap E(F) = \emptyset$. Then

$$e^+(F \cup L''_R) = e(L''_R),$$

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

and

$$v^+(F \cup L_R'') = v(L_R'') - 2.$$

If $F^2 = F \cup L_R'' \in \mathcal{H}^*(F, e, H_1, H_2)$, then transformation (iii) sets $F^3 := F^2$. Otherwise we have that $F \cup L_R'' \in \mathcal{H}(F, e, H_1, H_2) \setminus \mathcal{H}^*(F, e, H_1, H_2)$. Let $F^3 := J^*$ where J^* is any member of $\mathcal{H}^*(F, e, H_1, H_2)$ and recall that, indeed, J^* is a possible output of a non-degenerate iteration of the while-loop of Grow.

Then, in transformation (iii), we replace $F \cup L_R''$ with the graph J^*. Since H_2 is (strictly) 2-balanced and H_1 is (strictly) balanced with respect to $d_2(\cdot, H_2)$, we have that

$$m_2(H_1, H_2) = \frac{e_1}{v_1 - 2 + \frac{1}{m_2(H_2)}} = \frac{e_1(e_2 - 1)}{(v_1 - 2)(e_2 - 1) + v_2 - 2} = \frac{e^+(J^*)}{v^+(J^*)}. \tag{13}$$

Using (13) and Lemma 6.8, and that H_2 is (strictly) 2-balanced and H_1 is (strictly) balanced with respect to $d_2(\cdot, H_2)$, we have that

$$\begin{aligned}
&v(F^3) - v(F^2) - \frac{e(F^3) - e(F^2)}{m_2(H_1, H_2)} \\
&= v(J^*) - v(F \cup L_R'') - \frac{e(J^*) - e(F \cup L_R'')}{m_2(H_1, H_2)} \\
&= v^+(J^*) - v^+(F \cup L_R'') - \frac{e^+(J^*) - e^+(F \cup L_R'')}{m_2(H_1, H_2)} \\
&\overset{L.6.8}{>} v^+(J^*) - v^+(F \cup L_R'') - \frac{e^+(J^*) - e^+(J^*)\left(\frac{v^+(F \cup L_R'')}{v^+(J^*)}\right)}{m_2(H_1, H_2)} \\
&= \left(v^+(J^*) - v^+(F \cup L_R'')\right)\left(1 - \frac{e^+(J^*)}{v^+(J^*) m_2(H_1, H_2)}\right) \\
&= 0.
\end{aligned}$$

Since $v^+(J^*)$, $v^+(F \cup L_R'')$, $e^+(J^*)$, $e^+(F \cup L_R'')$ and $m_2(H_1, H_2)$ only rely on H_1 and H_2, there exists $\delta_2 = \delta_2(H_1, H_2) > 0$ such that

$$v(F^3) - v(F^2) - \frac{e(F^3) - e(F^2)}{m_2(H_1, H_2)} \geq \delta_2.$$

Taking

$$\kappa_2 := \min\{1, \delta_1, \delta_2\},$$

we see that (11) holds. □

As stated earlier, Claim 6.3 follows from Claims 6.9 and 6.10. All that remains to prove Case 1 of Lemma 5.5 is to prove Lemma 6.8.

6.3 *Proof of Lemma 6.8*

Let $J \in \mathcal{H}(F, \hat{e}, H_1, H_2) \setminus \mathcal{H}^*(F, \hat{e}, H_1, H_2)$. We choose the graph $J^* \in \mathcal{H}^*(F, \hat{e}, H_1, H_2)$ with the following properties.

1) The edge of the copy $H_{\hat{e}}$ of H_2 in J attached at $\hat{e}$ and its orientation when attached are the same as the edge of the copy $H_{\hat{e}}^*$ of H_2 in J^* attached at $\hat{e}$ and its orientation when attached;

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

2) for each $f \in E_J$, the edge of the copy H_f of H_1 in J attached at f and its orientation when attached are the same as the edge of the copy H_f^* of H_1 in J^* attached at f and its orientation when attached.

Then, recalling definitions from the beginning of Section 6.2, we have that $V_J = V_{J^*}$ and $E_J = E_{J^*}$; that is, $H_{\hat{e}}^J = H_{\hat{e}}^{J^*}$. From now on, let $V := V_J$, $E := E_J$ and $H_{\hat{e}}^- := H_{\hat{e}}^J$. Observe for *all* $J' \in \mathcal{H}^*(F, \hat{e}, H_1, H_2)$, that

$$\frac{e^+(J')}{v^+(J')} = \frac{e_1(e_2-1)}{(v_1-2)(e_2-1)+v_2-2}. \tag{14}$$

Hence, to prove Lemma 6.8 it suffices to show

$$\frac{e^+(J)}{v^+(J)} > \frac{e^+(J^*)}{v^+(J^*)}.$$

As in [14], the intuition behind our proof is that J^* can be transformed into J by successively merging the copies H_f^* of H_1 in J^* with each other and vertices in $H_{\hat{e}}^-$. We do this in $e_2 - 1$ steps, fixing carefully a total ordering of the inner edges E. For every edge $f \in E$, we merge the attached outer copy H_f^* of H_1 in J^* with copies of H_1 (attached to edges preceding f in our ordering) and vertices of $H_{\hat{e}}^-$. Throughout, we keep track of the number of edges $\Delta_e(f)$ and the number of vertices $\Delta_v(f)$ vanishing in this process. One could hope that the F-external density of J increases in every step of this process, or, even slightly stronger, that $\Delta_e(f)/\Delta_v(f) < e^+(J^*)/v^+(J^*)$. This does not necessarily hold, but we will show that there exists a collection of edge-disjoint subgraphs A_i of $H_{\hat{e}}^-$ such that, for each i, the edges of $E(A_i)$ are 'collectively good' for this process and every edge not belonging to one of these A_i is also 'good' for this process.

Recalling definitions from the beginning of Section 6.2, let $H_f^- := (U_J(f) \dot\cup f, D_J(f))$ denote the subgraph obtained by removing the edge f from the copy H_f of H_1 in J.

Later, we will carefully define a (total) ordering $\prec$ on the inner edges E.[14] For such an ordering $\prec$ and each $f \in E$, define

$$\Delta_E(f) := D_J(f) \cap \left(\bigcup_{f' \prec f} D_J(f') \right),$$

and

$$\Delta_V(f) := U_J(f) \cap \left(\left(\bigcup_{f' \prec f} U_J(f') \right) \cup V(H_{\hat{e}}^-) \right),$$

and set $\Delta_e(f) := |\Delta_E(f)|$ and $\Delta_v(f) := |\Delta_V(f)|$. We emphasise here that the definition of $\Delta_v(f)$ takes into account how vertices of outer vertex sets can intersect with the inner graph $H_{\hat{e}}^-$. One can see that $\Delta_e(f)$ ($\Delta_v(f)$) is the number of edges (vertices) vanishing from H_f^* when it is merged with preceding attached copies of H_1 and $V(H_{\hat{e}}^-)$.

By our choice of J^*, one can quickly see that

$$e^+(J) = e^+(J^*) - \sum_{f \in E} \Delta_e(f), \tag{15}$$

and

$$v^+(J) = v^+(J^*) - \sum_{f \in E} \Delta_v(f). \tag{16}$$

[14] For clarity, for any $f \in E$, $f \not\prec f$ in this ordering $\prec$.

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

By (15) and (16), we have

$$\frac{e^+(J)}{v^+(J)} = \frac{e^+(J^*) - \sum_{f\in E}\Delta_e(f)}{v^+(J^*) - \sum_{f\in E}\Delta_v(f)}.$$

Then, by Fact 4.1, to show that

$$\frac{e^+(J)}{v^+(J)} > \frac{e^+(J^*)}{v^+(J^*)},$$

it suffices to prove that

$$\frac{\sum_{f\in E}\Delta_e(f)}{\sum_{f\in E}\Delta_v(f)} < \frac{e^+(J^*)}{v^+(J^*)}.^{15} \tag{17}$$

To show (17), we will now carefully order the edges of E using an algorithm Order-Edges (Figure 8). The algorithm takes as input the graph $H^-_{\hat{e}} = (V \dot\cup \hat{e}, E)$ and outputs a stack s containing every edge from E and a collection of edge-disjoint edge sets E_i in E and (not necessarily disjoint) vertex sets V_i in $V \dot\cup \hat{e}$. We take our total ordering $\prec$ of E to be that induced by the order in which edges of E were placed onto the stack s (that is, $f \prec f'$ if and only if f was placed onto the stack s before f'). Also, for each i, we define $A_i := (V_i, E_i)$ to be the graph on vertex set V_i and edge set E_i and observe that $A_i \subsetneq H_2$. We will utilise this ordering and our choice of E_i and V_i for each i, alongside that H_2 is (strictly) 2-balanced and H_1 is *strictly* balanced with respect to $d_2(\cdot, H_2)$, in order to conclude (17).

Let us describe algorithm Order-Edges (Figure 8) in detail. In lines 2-8, we initialise several parameters: a stack s, which we will place edges of E on during our algorithm; sets E_i and V_i for each $i \in [\lfloor e_2/2 \rfloor]$,[16] which we will add edges of E and vertices of $V \dot\cup \hat{e}$ into, respectively; an index j, which will correspond to whichever graph A_j we consider constructing next; a set E', which will keep track of those edges of E we have not yet placed onto the stack s. Line 9 ensures the algorithm continues until $E' = \emptyset$, that is, until all the edges of E have been placed onto s.

In line 10, we begin constructing A_j by finding a pair of distinct edges in E' whose outer edge sets (in J) intersect. In lines 11-14, we place one of these edges, f, onto s, into E_j and remove it from E'. We also set V_j to be the two vertices in f alongside any vertices in the outer vertex set $U_J(f)$ that intersect $V(H^-_{\hat{e}})$.

In lines 15-20, we iteratively add onto s, into E_j and remove from E' any edge $uw \in E'$ which either connects two vertices previously added to V_j or has an outer edge set $D_J(uw)$ that intersects the collection of outer edge sets of edges previously added to E_j. We also update V_j in each step of this process.

In line 21, we increment j in preparation for the next check at line 10 (if we still have $E' \neq \emptyset$). If the condition in line 10 fails then in lines 23-26 we arbitrarily place the remaining edges of E' onto the stack s. In line 29, we output the stack s and in lines 30-32 we output each non-empty E_i and V_i.

We will now argue that each proper subgraph $A_i = (V_i, E_i)$ of H_2 and each edge placed onto s in line 24 are 'good', in some sense, for us to conclude (17).

For each $f \in E$, define the graph

$$T(f) := (\Delta_V(f) \dot\cup f, \Delta_E(f)) \subseteq H^-_f \subsetneq H_1.$$

Observe that one or both vertices of f may be isolated in $T(f)$. This observation will be very useful later.

[15] Note that $\sum_{f\in E}\Delta_v(f) \geq 1$ as otherwise $J = J^*$.

[16] Order-Edges can output at most $\lfloor e_2/2 \rfloor$ pairs of sets E_i and V_i.

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

1: **procedure** ORDER-EDGES($H_{\hat{e}}^- = (V \dot\cup \hat{e}, E)$)
2: $s \leftarrow$ EMPTY-STACK()
3: **for all** $i \in [\lfloor e_2/2 \rfloor]$ **do**
4: $E_i \leftarrow \emptyset$
5: $V_i \leftarrow \emptyset$
6: **end for**
7: $j \leftarrow 1$
8: $E' \leftarrow E$
9: **while** $E' \neq \emptyset$ **do**
10: **if** $\exists f, f' \in E'$ s.t. $(f \neq f') \wedge (D_J(f) \cap D_J(f') \neq \emptyset)$ **then**
11: s.PUSH(f)
12: E_j.PUSH(f)
13: E'.REMOVE(f)
14: $V_j \leftarrow f \cup (U_J(f) \cap V(H_{\hat{e}}^-))$
15: **while** $\exists\, uw \in E'$ s.t. $(u, w \in V_j) \vee \left(D_J(uw) \cap \bigcup_{f \in E_j} D_J(f) \neq \emptyset\right)$ **do**
16: s.PUSH(uw)
17: E_j.PUSH(uw)
18: E'.REMOVE(uw)
19: $V_j \leftarrow \bigcup_{f \in E_j} \left(f \cup (U_J(f) \cap V(H_{\hat{e}}^-))\right)$
20: **end while**
21: $j \leftarrow j + 1$
22: **else**
23: **for all** $f \in E'$ **do**
24: s.PUSH(f)
25: E'.REMOVE(f)
26: **end for**
27: **end if**
28: **end while**
29: **return** s
30: **for all** $i \in [\lfloor e_2/2 \rfloor]$ s.t. $E_i \neq \emptyset$ **do**
31: **return** E_i
32: **return** V_i
33: **end for**
34: **end procedure**

Figure 8. The implementation of algorithm Order-Edges.

For each i and $f \in E_i$, define

$$(V(H_{\hat{e}}^-))_f := (V(H_{\hat{e}}^-) \cap U_J(f)) \setminus \left(\bigcup_{\substack{f' \in E_i: \\ f' \prec f}} f' \cup \left(\bigcup_{\substack{f' \in E_i: \\ f' \prec f}} \left(V(H_{\hat{e}}^-) \cap U_J(f')\right) \right) \right) \subseteq \Delta_V(f),$$

since $(V(H_{\hat{e}}^-))_f$ is a subset of $V(H_{\hat{e}}^-) \cap U_J(f)$. One can see that $(V(H_{\hat{e}}^-))_f$ consists of those vertices of $V(H_{\hat{e}}^-)$ which are new to V_i at the point when f is added to E_i but are not contained in f. Importantly for our purposes, every vertex in $(V(H_{\hat{e}}^-))_f$ is isolated in $T(f)$. Indeed, otherwise there exists $k < i$ and $f'' \in E_k$ such that $D_J(f) \cap D_J(f'') \neq \emptyset$ and f would have been previously added to E_k in line 17.

For each $f \in E$, let $T'(f)$ be the graph obtained from $T(f)$ by removing all isolated vertices from $V(T(f))$. Crucially for our proof, since vertices of f may be isolated in $T(f)$, one or more of them may not belong to $V(T'(f))$. Further, no vertex of $(V(H_{\hat{e}}^-))_f$ is contained in $V(T'(f))$.

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

For all $f \in E$ with $\Delta_e(f) \geq 1$, since $T'(f) \subsetneq H_1$ and H_1 is *strictly* balanced with respect to $d_2(\cdot, H_2)$, we have that

$$m_2(H_1, H_2) > d_2(T'(f), H_2) = \frac{|E(T'(f))|}{|V(T'(f))| - 2 + \frac{1}{m_2(H_2)}}. \tag{18}$$

Recall (13), that is,

$$m_2(H_1, H_2) = \frac{e^+(J^*)}{v^+(J^*)}.$$

We now make the key observation of our proof: Since vertices of f may be isolated in $T(f)$, and so not contained in $V(T'(f))$, and no vertex of $(V(H^-_{\hat{e}}))_f$ is contained in $V(T'(f))$, we have that

$$|V(T'(f))| \leq \Delta_v(f) + |f \cap V(T'(f))| - |(V(H_{\hat{e}}))_f|. \tag{19}$$

Hence, from (18) and (19) we have that

$$\begin{aligned}\Delta_e(f) &= |E(T'(f))| \\ &< m_2(H_1, H_2)\left(\Delta_v(f) - (2 - |f \cap V(T'(f))|) - |(V(H^-_{\hat{e}}))_f| + \frac{1}{m_2(H_2)}\right).\end{aligned} \tag{20}$$

Edges f such that $|f \cap V(T'(f))| = 2$ will be, in some sense, 'bad' for us when trying to conclude (17). Indeed, if $|(V(H^-_{\hat{e}}))_f| = 0$ then we may have that $\frac{\Delta_e(f)}{\Delta_v(f)} \geq m_2(H_1, H_2) = \frac{e^+(J^*)}{v^+(J^*)}$, by (13). However, edges f such that $|f \cap V(T'(f))| \in \{0, 1\}$ will be, in some sense, 'good' for us when trying to conclude (17). Indeed, since $m_2(H_2) \geq 1$, we have for such edges f that $\frac{\Delta_e(f)}{\Delta_v(f)} < m_2(H_1, H_2) = \frac{e^+(J^*)}{v^+(J^*)}$.

We show in the following claim that our choice of ordering $\prec$, our choice of each A_i and the fact that H_2 is (strictly) 2-balanced ensure that for each i there are enough 'good' edges in A_i to compensate for any 'bad' edges that may appear in A_i.

Claim 6.11. *For each i,*

$$\sum_{f \in E_i} \Delta_e(f) < m_2(H_1, H_2) \sum_{f \in E_i} \Delta_v(f).$$

Proof. Fix i. Firstly, as observed before, each A_i is a non-empty subgraph of H_2. Moreover, $|E_i| \geq 2$ (by the condition in line 10). Since H_2 is (strictly) 2-balanced, we have that

$$m_2(H_2) \geq d_2(A_i) = \frac{|E_i| - 1}{|V_i| - 2}. \tag{21}$$

Now let us consider $\Delta_e(f)$ for each $f \in E_i$. For the edge f added in line 12, observe that $\Delta_e(f) = 0$. Indeed, otherwise $\Delta_e(f) \geq 1$, and there exists $k < i$ such that $D_J(f) \cap \left(\bigcup_{f' \in E_k} D_J(f')\right) \neq \emptyset$. That is, f would have been added to E_k in line 17 previously in the algorithm. Since $(V(H^-_{\hat{e}}))_f \subseteq \Delta_V(f)$, we have that

$$\Delta_e(f) = 0 \leq m_2(H_1, H_2)\left(\Delta_v(f) - |(V(H^-_{\hat{e}}))_f|\right). \tag{22}$$

Now, for each edge added in line 17, either $v, w \in V_i$ when uw was added to E_i, or we had $D_J(uw) \cap \left(\bigcup_{\substack{f' \in E_i: \\ f' \prec uw}} D_J(f')\right) \neq \emptyset$, that is, $\Delta_e(uw) \geq 1$, and at least one of u, w did not belong to V_i when uw was added to E_i. In the former case, if $\Delta_e(uw) \geq 1$, then by (20) and that $-(2 - |f \cap$

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

$V(T'(f))|) \leq 0$, we have that

$$\Delta_e(uw) < m_2(H_1, H_2)\left(\Delta_v(uw) - |(V(H_{\hat{e}}^-))_{uw}| + \frac{1}{m_2(H_2)}\right). \tag{23}$$

Observe that (23) also holds when $\Delta_e(uw) = 0$. In the latter case, observe that for all $k < i$, we must have $D_J(uw) \cap \left(\bigcup_{f \in E_k} D_J(f)\right) = \emptyset$. Indeed, otherwise uw would have been added to some E_k in line 17. Combining this with knowing that at least one of u, w did not belong to V_i before uw was added to E_i, we must have that one or both of u, w are isolated in $T(uw)$. That is, one or both do not belong to $T'(uw)$, and so $|uw \cap V(T'(uw))| \in \{0, 1\}$. Thus, since $\Delta_e(uw) \geq 1$, by (20) we have that

$$\Delta_e(uw) < m_2(H_1, H_2)\left(\Delta_v(uw) - 1 - |(V(H_{\hat{e}}^-))_{uw}| + \frac{1}{m_2(H_2)}\right), \tag{24}$$

if $|uw \cap V(T'(uw))| = 1$, and

$$\Delta_e(uw) < m_2(H_1, H_2)\left(\Delta_v(uw) - 2 - |(V(H_{\hat{e}}^-))_{uw}| + \frac{1}{m_2(H_2)}\right), \tag{25}$$

if $|uw \cap V(T'(uw))| = 0$.

In conclusion, except for the two vertices in the edge added in line 12, every time a new vertex x was added to V_i when some edge f was added to E_i, either $x \in (V(H_{\hat{e}}^-))_f$, or $x \in f$ and (24) or (25) held, dependent on whether one or both of the vertices in f were new to V_i. Indeed, x was isolated in $T(f)$. Moreover, after the two vertices in the edge added in line 12 there are $|V_i| - 2$ vertices added to V_i.

Hence, by (22)-(25), we have that

$$\sum_{f \in E_i} \Delta_e(f) < m_2(H_1, H_2)\left(\sum_{f \in E_i} \Delta_v(f) - (|V_i| - 2) + \frac{|E_i| - 1}{m_2(H_2)}\right). \tag{26}$$

By (21),

$$-(|V_i| - 2) + \frac{|E_i| - 1}{m_2(H_2)} \leq 0. \tag{27}$$

Thus, by (26) and (27), we have that

$$\sum_{f \in E_i} \Delta_e(f) < m_2(H_1, H_2) \sum_{f \in E_i} \Delta_v(f),$$

as desired. □

Claim 6.12. *For each edge f placed onto s in line 24, we have $\Delta_e(f) = 0$.*

Proof. Assume not. Then $\Delta_e(f) \geq 1$ for some edge f placed onto s in line 24. Observe that $D_J(f) \cap \left(\bigcup_{f'' \in E_i} D_J(f'')\right) = \emptyset$ for any i, otherwise f would have been added to some E_i in line 17 previously.

Thus we must have that $D_J(f) \cap D_J(f') \neq \emptyset$ for some edge $f' \neq f$ where f' was placed onto s also in line 24. But then f and f' satisfy the condition in line 10 and would both be contained in some E_i, contradicting that f was placed onto s in line 24. □

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

Since $J \in \mathcal{H}(F, \hat{e}, H_1, H_2) \setminus \mathcal{H}^*(F, \hat{e}, H_1, H_2)$, we must have that $\Delta_v(f) \geq 1$ for some $f \in E$. Thus, if $\Delta_e(f) = 0$ for all $f \in E$ then (17) holds trivially. If $\Delta_e(f) \geq 1$ for some $f \in E$, then $E_1 \neq \emptyset$ and A_1 is a non-empty subgraph of $H_{\hat{e}}^-$. Then, by (13) and Claims 6.11 and 6.12, we have that

$$\begin{aligned}\frac{\sum_{f\in E}\Delta_e(f)}{\sum_{f\in E}\Delta_v(f)} &= \frac{\sum_i\sum_{f\in E_i}\Delta_e(f)+\sum_{f\in E\setminus\cup_i E_i}\Delta_e(f)}{\sum_{f\in E}\Delta_v(f)}\\ &< \frac{m_2(H_1,H_2)\sum_i\sum_{f\in E_i}\Delta_v(f)}{\sum_i\sum_{f\in E_i}\Delta_v(f)}\\ &= m_2(H_1,H_2)\\ &= \frac{e^+(J^*)}{v^+(J^*)}.\end{aligned}$$

Thus (17) holds and we are done.

7. Case 2: $m_2(H_1, H_2) = m_2(H_2)$.

In this section we prove Lemma 5.5 when $m_2(H_1) = m_2(H_2)$. Our proof follows that of Case 1 significantly, but uses a different algorithm Grow-Alt. All definitions and notation are the same as previously unless otherwise stated.

For any graph F and edge $e \in E(F)$, we say that e is *eligible for extension in Grow-Alt* if it satisfies

$$\nexists L \in \mathcal{L}_F, R \in \mathcal{R}_F \text{ s.t. } E(L) \cap E(R) = \{e\}.$$

We note here that this is substantially different to Case 1; indeed, the set $\mathcal{C}^*$ will not feature in what follows. Algorithm Grow-Alt is shown in Figure 9. As with Grow, it has input $G' \subseteq G$, the graph that Asym-Edge-Col got stuck on. Grow-Alt operates in a similar way to Grow. In line 14, the function Eligible-Edge-Alt is called which maps F_i to an edge $e \in E(F_i)$ which is eligible for extension in Grow-Alt. As with Eligible-Edge in Case 1, this edge e is selected to be unique up to isomorphism. We then apply a new procedure Extend which attaches either a graph $L \in \mathcal{L}_{G'}$ or a graph $R \in \mathcal{R}_{G'}$ that contains e to F_i. As in Case 1, because the condition in line 6 of Asym-Edge-Col fails, $G' \in \mathcal{C}$.[17]

We now show that the number of edges of F_i increases by at least one and that Grow-Alt operates as desired with a result analogous to Claim 6.1.

Claim 7.1. *Algorithm Grow-Alt terminates without error on any input graph $G' \in \mathcal{C}$ that is not (H_1, H_2)-sparse or not an $\hat{\mathcal{A}}$-graph.*[18] *Moreover, for every iteration i of the while-loop, we have $e(F_{i+1}) > e(F_i)$.*

Proof. The special cases in lines 2-9 and the assignments in lines 10 and 11 operate successfully for the exact same reasons as given in the proof of Claim 6.1.

Next, we show that the call to Eligible-Edge-Alt in line 14 is always successful. Indeed, suppose for a contradiction that no edge in F_i is eligible for extension in Grow-Alt for some $i \geq 0$. Then for every edge $e \in E(F_i)$ there exist $L \in \mathcal{L}_{F_i}$ and $R \in \mathcal{R}_{F_i}$ s.t. $E(L) \cap E(R) = \{e\}$, by definition. Hence $F \in \mathcal{C}$. Recall that H_1 and H_2 satisfy the criteria of Conjecture 1.8. Hence H_1 and H_2 are strictly 2-balanced and $m_2(H_1) = m_2(H_2) > 1$. Then, by Lemma 4.2, we have that H_1 and H_2 are both 2-connected. Hence F_i is 2-connected by construction. However, our choice of F_0 in line 11 guarantees that F_i is not in $\hat{\mathcal{A}}$. Indeed, the edge e selected in line 10 satisfying $|\mathcal{S}_{G'}(e)| = 0$ is an

[17] Note that we could also conclude $G' \in \mathcal{C}^*$, however this will not be necessary as noted earlier. Also, see Section 9 for discussion on why we use Grow-Alt instead of Grow when $m_2(H_1) = m_2(H_2)$.

[18] *See Definitions 5.1 and 5.2.*

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

```
1: procedure GROW-ALT(G' = (V, E))
2:     if ∀e ∈ E : |S_{G'}(e)| = 1 then
3:         T ← any member of T_{G'}
4:         return ⋃_{e∈E(T)} S_{G'}(e)
5:     end if
6:     if ∃e ∈ E : |S_{G'}(e)| ≥ 2 then
7:         S_1, S_2 ← any two distinct members of S_{G'}(e)
8:         return S_1 ∪ S_2
9:     end if
10:    e ← any e ∈ E : |S_{G'}(e)| = 0
11:    F_0 ← any R ∈ R_{G'} : e ∈ E(R)
12:    i ← 0
13:    while (i < ln(n)) ∧ (∀F̃ ⊆ F_i : λ(F̃) > −γ) do
14:        e ← ELIGIBLE-EDGE-ALT(F_i)
15:        F_{i+1} ← EXTEND(F_i, e, G')
16:        i ← i + 1
17:    end while
18:    if i ≥ ln(n) then
19:        return F_i
20:    else
21:        return MINIMISING-SUBGRAPH(F_i)
22:    end if
23: end procedure
```

```
1: procedure EXTEND(F, e, G')
2:     {L, R} ← any pair {L, R} such that L ∈ L_{G'}, R ∈ R_{G'} and E(L) ∩ E(R) = e
3:     if L ⊈ F then
4:         F' ← F ∪ L
5:     else
6:         F' ← F ∪ R
7:     end if
8:     return F'
9: end procedure
```

Figure 9. The implementation of algorithm Grow-Alt.

edge of F_0 and $F_0 \subseteq F_i \subseteq G'$. Thus, by the definition of $\hat{\mathcal{A}}$ and that $m_2(H_1) = m_2(H_2)$, we have that $m(F_i) > m_2(H_1, H_2) + \varepsilon$.

Thus, there exists a non-empty graph $\tilde{F} \subseteq F_i$ with $d(\tilde{F}) = m(F_i)$ such that

$$\begin{aligned}\lambda(\tilde{F}) = \nu(\tilde{F}) - \frac{e(\tilde{F})}{m_2(H_1, H_2)} &= e(\tilde{F})\left(\frac{1}{m(F_i)} - \frac{1}{m_2(H_1, H_2)}\right) \\ &< e(\tilde{F})\left(\frac{1}{m_2(H_1, H_2) + \varepsilon} - \frac{1}{m_2(H_1, H_2)}\right) = -\gamma e(\tilde{F}) \le -\gamma.\end{aligned}$$

Thus Grow terminates in line 13 without calling Eligible-Edge-Alt. Thus every call that is made to Eligible-Edge-Alt is successful and returns an edge e. Since $G' \in \mathcal{C}$, the call to Extend(F_i, e, G') is also successful and thus there exist suitable graphs $L \in \mathcal{L}_{G'}$ and $R \in \mathcal{R}_{G'}$ such that $E(L) \cap E(R) = \{e\}$, that is, line 2 is successful.

It remains to show that for every iteration i of the while-loop, we have $e(F_{i+1}) > e(F_i)$. Since e is eligible for extension in Grow-Alt for F_i and $E(L) \cap E(R) = \{e\}$, we must have that either $L \nsubseteq F_i$ or $R \nsubseteq F_i$. Hence Extend outputs $F' := F \cup L$ such that $e(F_{i+1}) = e(F') > e(F_i)$ or $F' := F \cup R$ such that $e(F_{i+1}) = e(F') > e(F_i)$. □

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

We consider the evolution of F_i now in more detail. We call iteration i of the while-loop in algorithm Grow-Alt *non-degenerate* (See Figure 10) if the following hold:

- If $L \nsubseteq F_i$ (that is, line 3 is true), then in line 4 we have $V(F_i) \cap V(L) = e$;
- If $L \subseteq F_i$ (that is, line 3 is false), then in line 6 we have $V(F_i) \cap V(R) = e$.

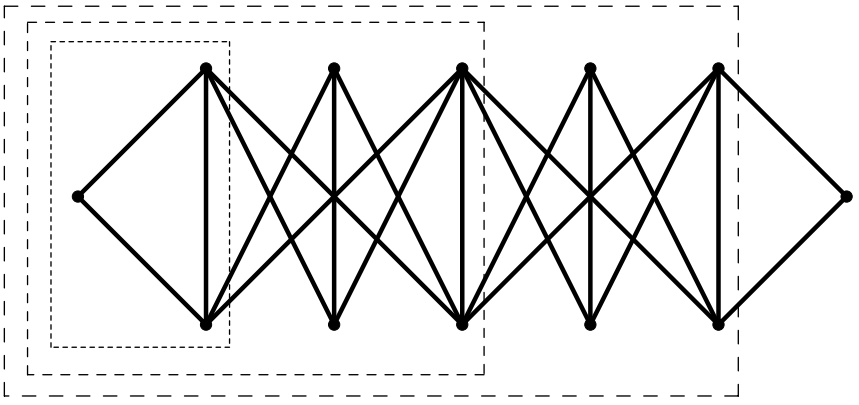

Figure 10. A graph F_3 resulting from three non-degenerate iterations for $H_1 = K_3$ and $H_2 = K_{3,3}$.

Otherwise, we call iteration i *degenerate*. Note that, in non-degenerate iterations i, there are only a constant number of graphs that F_{i+1} can be for any given F_i; indeed, Eligible-Edge-Alt determines the exact position where to attach L or R (recall that the edge e found by Eligible-Edge-Alt(F_i) is *unique up to isomorphism of* F_i).

We now prove a result analogous to Claim 6.2 for Grow-Alt.

Claim 7.2. *If iteration i of the while-loop in procedure Grow-Alt is non-degenerate, we have*

$$\lambda(F_{i+1}) = \lambda(F_i).$$

Proof. In a non-degenerate iteration, we either add $v_2 - 2$ vertices and $e_2 - 1$ edges for the copy L of H_2 to F_i or add $v_1 - 2$ vertices and $e_1 - 1$ edges for the copy R of H_1 to F_i. In the former case,

$$\begin{aligned}\lambda(F_{i+1}) - \lambda(F_i) &= v_2 - 2 - \frac{e_2 - 1}{m_2(H_1, H_2)} \\ &= v_2 - 2 - \frac{e_2 - 1}{m_2(H_2)} \\ &= 0,\end{aligned}$$

where the second equality follows from $m_2(H_1, H_2) = m_2(H_2)$ (see Proposition 1.7) and the last equality follows from H_2 being (strictly) 2-balanced.

In the latter case,

$$\begin{aligned}\lambda(F_{i+1}) - \lambda(F_i) &= v_1 - 2 - \frac{e_1 - 1}{m_2(H_1, H_2)} \\ &= v_1 - 2 - \frac{e_1 - 1}{m_2(H_1)} \\ &= 0.\end{aligned}$$

where the second equality follows from $m_2(H_1, H_2) = m_2(H_1)$ (see Proposition 1.7) and the last equality follows from H_1 being (strictly) 2-balanced. □

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

As in Case 1, when we have a degenerate iteration i, the structure of F_{i+1} depends not just on F_i but also on the structure of G'. Indeed, if F_i is extended by a copy L of H_2 in line 4 of Extend, then L could intersect F_i in a multitude of ways. Moreover, there may be several copies of H_2 that satisfy the condition in line 2 of Extend. One could say the same for graphs added in line 6 of Extend. Thus, as in Case 1, degenerate iterations cause us difficulties since they enlarge the family of graphs algorithm Grow-Alt can return. However, we will show that at most a constant number of degenerate iterations can happen before algorithm Grow-Alt terminates, allowing us to bound from above sufficiently well the number of non-isomorphic graphs Grow-Alt can return. Pivotal in proving this is the following claim, analogous to Claim 6.3.

Claim 7.3. *There exists a constant $\kappa = \kappa(H_1, H_2) > 0$ such that if iteration i of the while-loop in procedure Grow is degenerate then we have*

$$\lambda(F_{i+1}) \leq \lambda(F_i) - \kappa.$$

Compared to the proof of Claim 6.3, the proof of Claim 7.3 is relatively straightforward.

Proof. Let $F := F_i$ be the graph before the operation in line 15 (of Grow-Alt) is carried out and let $F' := F_{i+1}$ be the output from line 15. We aim to show there exists a constant $\kappa = \kappa(H_1, H_2) > 0$ such that

$$\lambda(F) - \lambda(F') = v(F) - v(F') - \frac{e(F) - e(F')}{m_2(H_1, H_2)} \geq \kappa$$

whether Extend attached a graph $L \in \mathcal{L}_{G'}$ or $R \in \mathcal{R}_{G'}$ to F. We need only consider the case when $L \in \mathcal{L}_{G'}$ is the graph added by Extend in this non-degenerate iteration i of the while-loop, as the proof of the $R \in \mathcal{R}_{G'}$ case is identical. Let $V_{L'} := V(F) \cap V(L)$ and $E_{L'} := E(F) \cap E(L)$ and set $v_{L'} := |V_{L'}|$ and $e_{L'} := |E_{L'}|$. Let L' be the graph on vertex set $V_{L'}$ and edge set $E_{L'}$. By Claim 7.1, we have $e(F') > e(F)$, hence L' is a proper subgraph of H_2. Also, observe that since iteration i was degenerate, we must have that $v_{L'} \geq 3$. Let $F_{\hat{L}}$ be the graph produced by a non-degenerate iteration at e with a copy $\hat{L}$ of H_2, that is, $F_{\hat{L}} := F \cup \hat{L}$ and $V(F) \cap V(\hat{L}) = e$. Our strategy is to compare F' with $F_{\hat{L}}$. By Claim 7.2, $\lambda(F) = \lambda(F_{\hat{L}})$. Thus, since $m_2(H_1, H_2) = m_2(H_2)$, we have

$$\begin{aligned}
\lambda(F) - \lambda(F') &= \lambda(F_{\hat{L}}) - \lambda(F') \\
&= v_2 - 2 - (v_2 - v_{L'}) - \frac{(e_2 - 1) - (e_2 - e_{L'})}{m_2(H_1, H_2)} \\
&= v_{L'} - 2 - \frac{e_{L'} - 1}{m_2(H_1, H_2)}. \qquad (28)
\end{aligned}$$

If $e_{L'} = 1$, then since $v_{L'} \geq 3$ we have $\lambda(F) - \lambda(F') \geq 1$. So assume $e_{L'} \geq 2$. Since H_2 is *strictly* 2-balanced and L' is a proper subgraph of H_2 with $e_{L'} \geq 2$ (that is, L' is not an edge), we have that

$$\frac{v_{L'} - 2}{e_{L'} - 1} > \frac{1}{m_2(H_2)}. \qquad (29)$$

Using (28), (29) and that $e_{L'} \geq 2$ and $m_2(H_1, H_2) = m_2(H)$, we have

$$\lambda(F) - \lambda(F') = (e_{L'} - 1)\left(\frac{v_{L'} - 2}{e_{L'} - 1} - \frac{1}{m_2(H_2)}\right) > 0.$$

Letting $\delta := \frac{1}{2} \min\left\{(e_{L'} - 1)\left(\frac{v_{L'}-2}{e_{L'}-1} - \frac{1}{m_2(H_2)}\right) : L' \subset H_2, e_{L'} \geq 2\right\}$, we take

$$\kappa := \min\{1, \delta\}.$$

□

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

Together, Claims 7.2 and 7.3 yield the following claim (analogous to Claim 6.4).

Claim 7.4. *There exists a constant $q_2 = q_2(H_1, H_2)$ such that algorithm Grow-Alt performs at most q_2 degenerate iterations before it terminates, regardless of the input instance G'.*

Proof. Analogous to the proof of Claim 6.4. □

For $0 \leq d \leq t \leq \lceil \ln(n) \rceil$, let $\mathcal{F}_{\mathrm{ALT}}(t, d)$ denote a family of representatives for the isomorphism classes of all graphs F_t that algorithm Grow-Alt can possibly generate after exactly t iterations of the while-loop with exactly d of those t iterations being degenerate. Let $f_{\mathrm{ALT}}(t, d) := |\mathcal{F}_{\mathrm{ALT}}(t, d)|$.

Claim 7.5. *There exist constants $C_0 = C_0(H_1, H_2)$ and $A = A(H_1, H_2)$ such that*

$$f_{\mathrm{ALT}}(t, d) \leq \lceil \ln(n) \rceil^{(C_0+1)d} \cdot A^{t-d}$$

for n sufficiently large.

Proof. Analogous to the proof of Claim 6.5. □

Let $\mathcal{F}_{\mathrm{ALT}} = \mathcal{F}_{\mathrm{ALT}}(H_1, H_2, n)$ be a family of representatives for the isomorphism classes of *all* graphs that can be outputted by Grow-Alt (whether Grow-Alt enters the while-loop or not). Note that the proof of the following claim requires Conjecture 1.8 to be true; in particular, we need that $\hat{\mathcal{A}}$ is finite when $m_2(H_1) = m_2(H_2)$.

Claim 7.6. *There exists a constant $b = b(H_1, H_2) > 0$ such that for all $p \leq bn^{-1/m_2(H_1,H_2)}$, $G_{n,p}$ does not contain any graph from $\mathcal{F}_{\mathrm{ALT}}(H_1, H_2, n)$ a.a.s.*

Proof. Analogous to the proof of Claim 6.6. □

Proof of Lemma 5.5: Case 2. Suppose that the call to Asym-Edge-Col(G) gets stuck for some graph G, and consider $G' \subseteq G$ at this moment. Then Grow-Alt(G') returns a copy of a graph $F \in \mathcal{F}_{\mathrm{ALT}}(H_1, H_2, n)$ that is contained in $G' \subseteq G$. By Claim 7.6, this event a.a.s. does not occur in $G = G_{n,p}$ with p as claimed. Thus Asym-Edge-Col does not get stuck a.a.s. and, by Lemma 5.3, finds a valid edge-colouring for H_1 and H_2 of $G_{n,p}$ with $p \leq bn^{-1/m_2(H_1,H_2)}$ a.a.s.

8. Proof of Theorem 1.10

Since H_1 and H_2 are regular graphs, let ℓ_1 be the degree of every vertex in H_1 and ℓ_2 be the degree of every vertex in H_2. We begin using an approach employed in [10] and [14]. Let A be a 2-connected graph such that $A \in \mathcal{C}^*(H_1, H_2)$ if $m_2(H_1) > m_2(H_2)$ and $A \in \mathcal{C}(H_1, H_2)$ if $m_2(H_1) = m_2(H_2)$. In both cases, $A \in \mathcal{C}(H_1, H_2)$. Then, since every vertex is contained in a copy of H_1 and a copy of H_2 whose edge sets intersect in exactly one edge, we must have that $\delta(A) \geq \ell_1 + \ell_2 - 1$. Hence

$$d(A) = \frac{e_A}{v_A} \geq \frac{\ell_1 + \ell_2 - 1}{2}. \tag{30}$$

We aim to show $d(A) > m_2(H_1, H_2) + \varepsilon$ for some $\varepsilon = \varepsilon(H_1, H_2) > 0$. Indeed, if there exists $\varepsilon = \varepsilon(H_1, H_2) > 0$ such that for all such graphs A we have $d(A) > m_2(H_1, H_2) + \varepsilon$ then $\hat{\mathcal{A}}(H_1, H_2, \varepsilon) = \emptyset$, and so Conjecture 1.8 holds trivially for H_1 and H_2.

Since $m_2(H_1) \geq m_2(H_2) > 1$, we have that H_1 and H_2 cannot be matchings. Hence $\ell_1, \ell_2 \geq 2$. Also, observe that

$$m_2(H_1, H_2) = \frac{e_1}{v_1 - 2 + \frac{1}{m_2(H_2)}} = \frac{\frac{v_1\ell_1}{2}}{v_1 - 2 + \frac{v_2 - 2}{\frac{v_2\ell_2}{2} - 1}} = \frac{v_1\ell_1}{2v_1 - 4 + \frac{4v_2 - 8}{v_2\ell_2 - 2}}. \tag{31}$$

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

Furthermore,

$$\begin{aligned}
&\frac{\ell_1+\ell_2-1}{2} > m_2(H_1,H_2)\\
\iff &\ell_1+\ell_2-1 > \frac{\nu_1\nu_2\ell_1\ell_2-2\nu_1\ell_1}{2\nu_1\nu_2\ell_2-4\nu_2\ell_2-4\nu_1+4\nu_2}\\
\iff &\ell_1+\ell_2-1 > \frac{2\nu_1\nu_2\ell_1\ell_2-4\nu_1\ell_1}{2\nu_1\nu_2\ell_2-4\nu_2\ell_2-4\nu_1+4\nu_2}\\
\iff &0 < \nu_1\nu_2\ell_2(\ell_2-1)-2\nu_1(\ell_2-1)-2\nu_2\ell_1(\ell_2-1)-2\nu_2(\ell_2-1)^2\\
\iff &0 < \nu_1\nu_2\ell_2-2\nu_1-2\nu_2\ell_1-2\nu_2\ell_2+2\nu_2, \qquad (32)
\end{aligned}$$

where in the last implication we used that $\ell_2 \geq 2$.

Let $f(\nu_1,\nu_2,\ell_1,\ell_2) := \nu_1\nu_2\ell_2-2\nu_1-2\nu_2\ell_1-2\nu_2\ell_2+2\nu_2$. Observe that $\ell_1 \leq \nu_1-1$. Hence $-2\nu_2\ell_1 \geq -2\nu_1\nu_2+2\nu_2$ and we have

$$f(\nu_1,\nu_2,\ell_1,\ell_2) \geq \nu_1\nu_2(\ell_2-2)-2\nu_1+4\nu_2-2\nu_2\ell_2.$$

Let $g(\nu_1,\nu_2,\ell_2) := \nu_1\nu_2(\ell_2-2)-2\nu_1+4\nu_2-2\nu_2\ell_2$ so $f(\nu_1,\nu_2,\ell_1,\ell_2) \geq g(\nu_1,\nu_2,\ell_2)$. Observe that, since $\ell_1,\ell_2 \geq 2$, we have that $\nu_1,\nu_2 \geq 3$. If $\ell_2 = 2$, then $g(\nu_1,\nu_2,2) = -2\nu_1 < 0$. However, if $\ell_2 \geq 3$, then since $\nu_1,\nu_2 \geq 3$, we have that

$$\begin{aligned}
\frac{dg}{d\nu_1} &= \nu_2(\ell_2-2)-2 > 0;\\
\frac{dg}{d\nu_2} &= \nu_1(\ell_2-2)+4-2\ell_2 = (\nu_1-2)(\ell_2-2) > 0;\\
\frac{dg}{d\ell_2} &= \nu_1\nu_2-2\nu_2 = \nu_2(\nu_1-2) > 0.
\end{aligned}$$

Further,

$$g(4,5,3) = 20-8+20-30 = 2 > 0.$$

Thus, for all $\nu_1 \geq 4$, $\nu_2 \geq 5$, $\ell_2 \geq 3$, we have that

$$f(\nu_1,\nu_2,\ell_1,\ell_2) \geq g(\nu_1,\nu_2,\ell_2) \geq g(4,5,3) = 2 > 0.$$

Hence, by (30)-(32), we have that there exists a constant $\varepsilon := \varepsilon(H_1,H_2) > 0$ such that

$$d(A) > m_2(H_1,H_2)+\varepsilon.$$

Thus we only have left the cases when $\nu_1 = 3$, $\nu_2 \leq 4$ or $\ell_2 = 2$.

Case 1. $\ell_2 = 2$. □

Then H_2 is a cycle and

$$f(\nu_1,\nu_2,\ell_1,2) = 2\nu_1\nu_2-2\nu_1-2\nu_2\ell_1-2\nu_2.$$

Observe that $\ell_1 \leq \nu_1-2$, as otherwise H_1 is a clique, contradicting that (H_1,H_2) is not a pair of a clique and a cycle. Thus $-\ell_1 \geq -(\nu_1-2)$. Then, since we excluded considering when H_2 is a cycle and H_1 is a graph with $\nu_1 = |V(H_1)| \geq |V(H_2)| = \nu_2$, we have that $\nu_2 > \nu_1$, and so

$$f(\nu_1,\nu_2,\ell_1,2) \geq 2(\nu_2-\nu_1) \geq 2 > 0.$$

Hence by (30)-(32), we have that there exists a constant $\varepsilon := \varepsilon(H_1,H_2) > 0$ such that $d(A) > m_2(H_1,H_2)+\varepsilon$.

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

Case 2. $v_1 = 3$.

Then $H_1 = K_3$ and $\ell_1 = 2$. Thus

$$f(3, v_2, 2, \ell_2) = v_2(\ell_2 - 2) - 6.$$

We can assume $\ell_2 \geq 3$, as otherwise we are in Case 1. Observe that we cannot have that $v_2 = 6$ and $\ell_2 \geq 3$. Indeed, when $\ell_2 = 3$, one can check that the only strictly 2-balanced 3-regular graph on 6 vertices is $K_{3,3}$. But then $(H_1, H_2) = (K_3, K_{3,3})$, which is a pair of graphs we excluded from consideration.

When $\ell_2 \geq 4$, we have that $m_2(H_2) > m_2(H_1)$, contradicting that our choice of H_1 and H_2 meet the criteria in Conjecture 1.8. If $v_2 \leq 5$, then since also $\ell_2 \geq 3$ we must have that H_2 is a copy of K_4 or K_5.[19] But then $m_2(H_2) > m_2(H_1)$.

Hence $v_2 \geq 7$ and $\ell_2 \geq 3$. Thus $f(3, v_2, 2, \ell_2) > 0$ and, as before, we conclude that there exists a constant $\varepsilon := \varepsilon(H_1, H_2) > 0$ such that $d(A) > m_2(H_1, H_2) + \varepsilon$.

Case 3. $v_2 \leq 4$.

Still assuming $\ell_2 \geq 3$, we have that $H_2 = K_4$, $v_2 = 4$ and $\ell_2 = 3$. If $v_1 \geq \ell_1 + 2$, then

$$f(v_1, 4, \ell_1, 3) = 10v_1 - 8(\ell_1 + 2) \geq 2(\ell_1 + 2) > 0.$$

If $v_1 = \ell_1 + 1$, then H_1 is a clique. Since H_1 and H_2 meet the criteria in Conjecture 1.8, we must have that $v_1 \geq 5$. Therefore

$$f(v_1, 4, \ell_1, 3) = 10v_1 - 8(\ell_1 + 2) = 2v_1 - 8 \geq 2 > 0.$$

Then, as before, there exists a constant $\varepsilon := \varepsilon(H_1, H_2) > 0$ such that $d(A) > m_2(H_1, H_2) + \varepsilon$, as desired.

9. Concluding remarks

In [14], the value of ε was set at 0.01 for *every* pair of cliques, that is, the value of ε did not depend explicitly on the graphs H_1 and H_2. It would be interesting to know if there exists a single value of ε satisfying Conjecture 1.8 for *all* suitable pairs of graphs H_1 and H_2.

Let us discuss why we use Grow-Alt when $m_2(H_1) = m_2(H_2)$ instead of Grow. The main reason is that when attaching a copy of H_1 precisely at an edge[20] of F_i to create F_{i+1} - for some iteration i during Grow or Grow-Alt - we have $\lambda(F_i) = \lambda(F_{i+1})$ if $m_2(H_1) = m_2(H_2)$ and $\lambda(F_i) > \lambda(F_{i+1})$ if $m_2(H_1) > m_2(H_2)$. We would say that iteration i was a non-degenerate iteration if $m_2(H_1) = m_2(H_2)$ and a degenerate iteration if $m_2(H_1) > m_2(H_2)$. That is, if we were to use Grow instead of Grow-Alt then the number of possible non-degenerate iterations would depend on the size of F_i and grow too quickly.

Connected to this observation is why for $A \in \hat{\mathcal{A}}$ we take $A \in \mathcal{C}$ when $m_2(H_1) = m_2(H_2)$, rather than $A \in \mathcal{C}^*$.[21] In the proof of Claim 7.1, we need to conclude that F_i is not in $\hat{\mathcal{A}}$ in order to conclude Eligible-Edge-Alt is always successful. As part of concluding this, we need that $F_i \in \mathcal{C}$, which we get by assuming for a contradiction that there is no edge in F_i eligible for extension in Grow-Alt. Hence, if for all $A \in \hat{\mathcal{A}}$ one sets $A \in \mathcal{C}^*$ when $m_2(H_1) = m_2(H_2)$, then one needs to conclude that $F_i \in \mathcal{C}^*$. This means that we must assume for a contradiction that for every edge $e \in E(F_i)$ there exists $L \in L^*_{F_i}$ such that $e \in E(L)$. Such an argument then requires us to change from using Eligible-Edge-Alt to Eligible-Edge in Grow-Alt in order to arrive at the correct contradiction in the proof. But now Extend(F_i, e, G') in Grow-Alt may fail to add an edge as it is possible that every $L \in \mathcal{L}^*_{G'}$ containing e has its copies of H_1 and H_2 intersecting at e both contained fully in F_i,

[19] There exists no graph with both odd regularity and odd order.
[20] That is, $e_1 - 1$ edges and $v_1 - 2$ vertices are added.
[21] As mentioned at the end of Section 1.4.3.

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

that is, we cannot guarantee that in every iteration i of the while-loop of Grow-Alt that $e(F_{i+1}) > e(F_i)$.

In order to overcome this we would need to replace procedure Extend with a procedure similar to procedure Extend-L, which is used in Grow. If we used Extend-L, then the copy L of H_1 could be fully contained in F_i and one of its copies R' of H_2 appended to it could contain precisely one edge of F_i and two vertices of F_i, with every other appended copy of H_2 fully contained in F_i. This would then have to be considered a non-degenerate iteration for Grow-Alt as $\lambda(F_i) = \lambda(F_{i+1})$. But, importantly, Extend-L(F_i, e, G') depends on the structure of G', not on the structure of F_i. Thus R' could be attached at a number of edges of F_i. This would increase the number of possible non-degenerate iterations too much.

What procedure similar to Extend-L do we choose then? One possibility would be to attach a copy L of H_2 with copies of H_1 appended to *every one* of its edges e', but this procedure runs into the same problem as Extend-L. Overall, this discussion shows how difficult - and maybe impossible - it would be to prove the $m_2(H_1) = m_2(H_2)$ case of Lemma 5.5 while stipulating that when $A \in \hat{\mathcal{A}}$ we have $A \in \mathcal{C}^*$.

Also, in [14], Marciniszyn, Skokan, Spöhel and Steger note that they do not know whether $\mathcal{C}^*(H_1, H_2) = \mathcal{C}(H_1, H_2)$ or not for any pair of cliques. The author is unaware if this has been resolved for any pair of graphs. According with the definition of $\hat{\mathcal{A}}$, perhaps it is the case that $\mathcal{C}^*(H_1, H_2) = \mathcal{C}(H_1, H_2)$ whenever $m_2(H_1) = m_2(H_2)$?

Note that our method in this paper does not completely extend to when $m_2(H_2) = 1$. Indeed, in this case H_2 is a forest and so not 2-connected, a property which is specifically used in the proofs of Claims 6.1, 6.6, 7.1 and 7.6. In the proofs of Claims 6.1 and 7.1 we require that F_i is 2-connected for each $i > 0$. However, if the last iteration of the while-loop of either Grow or Grow-Alt was non-degenerate when constructing F_i, then F_i is certainly not 2-connected if H_2 is a tree. A similar problem occurs at the beginning of the proofs of Claims 6.6 and 7.6 if $T \cong H_2$.

Acknowledgements

The author is grateful to Andrew Treglown for suggesting working on the 0-statement of the Kohayakawa-Kreuter Conjecture and to Robert Hancock and Andrew Treglown for providing many helpful comments on a draft of this paper. The author also thanks the referee for their comments and suggestions.

References

[1] Chung, F. and Graham, R. (1998) Erdős on AK Peters, (His legacy of unsolved problems).
[2] Folkman, J. (1970) Graphs with monochromatic complete subgraphs in every edge coloring. *SIAM J. Appl. Math.* **18** 19–24.
[3] Frankl, P. and Rödl, V. (1986) Large triangle-free subgraphs in graphs without K_4. *Graphs Combin.* **2** 135–144.
[4] Friedgut, E. and Krivelevich, M. (2000) Sharp thresholds for certain Ramsey properties of random graphs. *Random Struct. Algor.* **17** 1–19.
[5] Friedgut, E., Rödl, V., Ruciński, A. and Tetali, P. (2006) A sharp threshold for random graphs with a monochromatic triangle in every edge coloring. *Mem. Amer. Math. Soc* **179** vi–66.
[6] Gugelmann, L., Nenadov, R., Person, Y., Škorić, N., Steger, A. and Thomas, H. (2017) Symmetric and asymmetric Ramsey properties in random hypergraphs. *Forum Math. Sigma* **5** 47pages.
[7] Hancock, R., Staden, K. and Treglown, A. (2019) Independent sets in hypergraphs and Ramsey properties of graphs and the integers. *SIAM J. Disc. Math.* **33** 153–188.
[8] Janson, S., Łuczak, T. and Ruciński, A. (2000) Random Graphs. *Wiley-Interscience Series in Discrete Mathematics and Optimization*, Wiley–Interscience.
[9] Kim, J. H. (1995) The Ramsey number R(3,t) has order of magnitude $t^2/\log t$. *Random Struct. Algor.* **7** 173–207.
[10] Kohayakawa, Y. and Kreuter, B. (1997) Threshold functions for asymmetric Ramsey properties involving cycles. *Random Struct. Algor.* **11** 245–276.

https://doi.org/10.1017/S0963548322000219 Published online by Cambridge University Press

[11] Kohayakawa, Y., Schacht, M. and Spöhel, R. (2014) Upper bounds on probability thresholds for asymmetric Ramsey properties. *Random Struct. Algor.* **44** 1–28.
[12] Liebenau, A., Mattos, L., Mendonça, W. and Skokan, J. (2010) Asymmetric Ramsey properties of random graphs for cliques and cycles, arXiv:2010.11933.
[13] Łuczak, T., Ruciński, A. and Voigt, B. (1992) Ramsey properties of random graphs. *J. Combin. Theory Ser. B* **56** 55–68.
[14] Marciniszyn, M., Skokan, J., Spöhel, R. and Steger, A. (2009) Asymmetric Ramsey properties of random graphs involving cliques. *Random Struct. Algor.* **34**(4) 419–453.
[15] Mousset, F., Nenadov, R. and Samotij, W. (2020) Towards the Kohayakawa–Kreuter conjecture on asymmetric Ramsey properties. *Combin Probab. Comput.* **29** 943–955.
[16] Nenadov, R. and Steger, A. (2016) A short proof of the random Ramsey theorem. *Combin. Probab. Comput.* **25**(1) 130–144.
[17] Nešetřil, J. and Rödl, V. (1976) The Ramsey property for graphs with forbidden complete subgraphs. *J. Combin. Theory Ser. B* **20** 243–249.
[18] Ramsey, F. P. (1930) On a problem of formal logic. *Proc. Lon. Math. Soc.* **30** 264–286.
[19] Rödl, V. and Ruciński, A. (1993) Lower bounds on probability thresholds for Ramsey properties. In Combinatorics, Paul Erdos is eighty, Vol. 1, Bolyai Soc. *Math. Stud., János Bolyai Math. Soc.* Budapest, pp. 317–346.
[20] Rödl, V. and Ruciński, A. (1994) Random graphs with monochromatic triangles in every edge coloring. *Random Struct. Algor.* **5** 253–270.
[21] Rödl, V. and Ruciński, A. (1995) Threshold functions for Ramsey properties. *J. Amer. Math. Soc.* **8** 917–942.
[22] Schacht, M. and Schulenburg, F. (2018) Sharp thresholds for Ramsey properties of strictly balanced nearly bipartite graphs. *Random Struct. Algor.* **52**3–40.

Cite this article: Hyde J (2023). Towards the 0-statement of the Kohayakawa-Kreuter conjecture. *Combinatorics, Probability and Computing* **32**, 225–268. https://doi.org/10.1017/S0963548322000219