On an online version of Rota's Basis Conjecture

Bollen, G.P.

Award date:
2014

Technische Universiteit
Eindhoven
University of Technology

Department of Mathematics and Computer Science

On an online version of Rota's Basis Conjecture

Guus P. Bollen

August 29, 2014

Abstract. In this thesis, a conjecture by Gian-Carlo Rota on disjoint independent transversals of bases in a vector space is discussed. The three-dimensional case, which was already proven in literature, is shown in a different manner.

Furthermore, a (new) online version of Rota's Basis Conjecture is given. For even dimensions, this online version turns out to follow from the Alon-Tarsi Conjecture, just like Rota's Basis Conjecture. For odd dimensions, however, this online version turns out to be false. Some variants of this online version are also discussed.

Finally, we zoom in on an online version of Rota's Basis Conjecture for graphic matroids. A concrete heuristic is proposed to solve this problem, which is shown to be correct for trees up to 5 vertices, and is conjectured to work for higher numbers of vertices.

Contents

Preface

This thesis is written as the conclusion of my master's project, which is a requirement for completing the master's program Industrial and Applied Mathematics and obtaining the degree 'Master of Science' at Eindhoven University of Technology.

First of all, I would like to express my gratitude to Jan Draisma for supervising my master's project of which this thesis is the result. The many hours of discussion about many different aspects of my research have greatly aided me in thoroughly understanding the subject.

Next, I want to thank my friends and family, who have been very supportive to me.

Finally, thanks to anyone who reads this thesis for their interest in the subject I have spent so much time investigating.

CHAPTER 1

Introduction

1. Rota's Basis Conjecture

In 1989, Rota conjectured the following [**1**].

CONJECTURE 1.1. *(Rota's Basis Conjecture). Consider bases*

$$(B_1, \ldots, B_n) = ((b_{11}, \ldots, b_{1n}), \ldots, (b_{n1}, \ldots, b_{nn}))$$

of an n-dimensional vector space V over an arbitrary infinite field. Then there exist $\pi_1, \ldots, \pi_n \in S_n$ such that for all $i = 1, \ldots, n$, $(b_{1,\pi_1(i)}, \ldots, b_{n,\pi_n(i)})$ is a basis of V.

A multiset of vectors that contains exactly one vector of each $B_1, \ldots, B_n$ is also called a *transversal* of $(B_1, \ldots, B_n)$.

A way to visualize this conjecture is the following. Place the vectors of the bases in an $n \times n$ table, where the k'th row contains B_k:

$$\begin{matrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{nn} \end{matrix}$$

By construction, the rows are bases. The columns, however, are not necessarily bases. A simple example for this is when the bases $B_1, \ldots, B_k$ are taken to be identical. Then each column contains n copies of the same vector; clearly not a basis. Rota's Basis Conjecture claims that it is always possible to permute the vectors within each row in such a way that the columns, too, are bases.

The simplest 'analogy' that comes to mind is a Sudoku: a popular puzzle where one has to complete a 9×9 square of numbers where in particular each row and each column can only contain one of each number. When $B_1, \ldots, B_9$ consist of the same 9 vectors, and each vector is labeled by the numbers 1 up to 9, each solution of a Sudoku translates directly into a solution of Rota's problem. Of course, in this case there are many solutions; otherwise solving Sudoku's would be rather monotonous. Then again, the assumption that all n bases are the same is a rather big simplification.

Rota's Basis Conjecture should not be confused with Rota's Conjecture, which is a different conjecture from matroid theory by Rota. This conjecture is about excluded minors for representability of matroids over a finite field. In 2013, Geelen, Gerards and Whittle announced that they had proven Rota's Conjecture. In this thesis, there will be no further mention of Rota's Conjecture.

2. Outline of the thesis

The thesis will be structured as follows.

Chapter 1 contains an introduction to Rota's Basis Conjecture, this outline, and a discussion about applications of the results in this thesis.

In Chapter 2, known results about Rota's Basis Conjecture will be briefly discussed.

Chapter 3 will start off with a theorem on three coloured sets in a partial linear space. This theorem is then used to show that Rota's Basis Conjecture is true in dimension three.

In Chapter 4, we discuss an online version of Rota's Basis Conjecture, of which Draisma suspected that it could highlight some of the differences between Rota's Basis Conjecture for even and odd dimensions. As it turns out, this is indeed the case: the online version of Rota's Basis Conjecture appears to be conditioned on the Alon-Tarsi conjecture for even dimensions, while it is simply false for odd dimensions. This is our main result.

Figure 1 shows the relations between various problems. These relations will all be discussed in chapters 2 through 4.

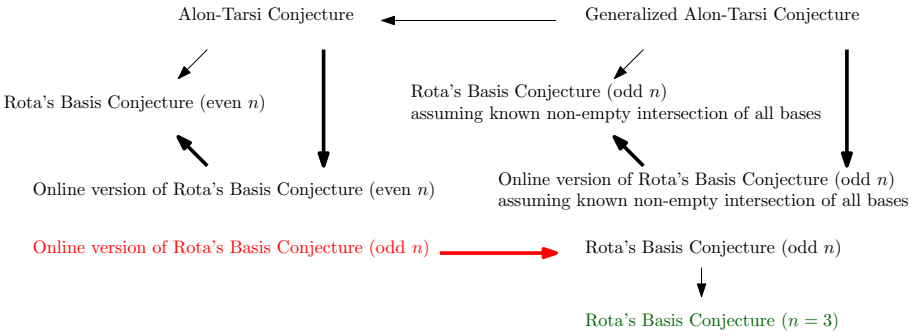


FIGURE 1. An overview of the relations between some of the problems discussed in this thesis. Green means proven; red means false.

Finally, in Chapter 5 we zoom in on one special case of Draisma's online version of Rota's Basis Conjecture, namely the case where we restrict ourselves to graphic matroids rather than general matroids or vector spaces. This allows us to be much more concrete when it comes to describing the algorithms of which, assuming the Alon-Tarsi conjecture holds, only the existence was shown in the previous chapter.

A part of the results covered in this thesis was simultaneously written in the form of an article by my supervisor Jan Draisma and me [**10**].

3. Applications

Many people, most of whom are not mathematicians, have asked me throughout the course of my master's project how the results of my research can be applied in practice. Sadly, I had to disappoint these people by telling them I do not know of any applications in practice. Yet, my research is not completely distanced from reality, although still not known to be very close to reality. The reader who wishes to move straight on to the mathematics can safely skip the remainder of this section.

Suppose there are n entities. They might be people, they might be machines, they could be anything. Suppose each of these entities has n objects. These might be items, these might be jobs, these could be anything. Suppose certain relations

hold that determine which objects can be with the same entity together, and which cannot. Initially, all of these relations are satisfied, since clearly each entity already has n objects that can thus be together.

Now suppose that, for some reason, the n^2 objects need to be redistributed over n (not necessarily the same) entities, in such a way that each of the new entities has precisely one object from each original entity. The objects with one of the new entities must then again be related in such a way that they can be together.

If the relations between the objects follow a certain pattern, then Rota's Basis Conjecture can be applied to achieve this redistribution.

As a very concrete example, highlighting the results from this thesis, consider the following. Let the n entities be students, and let their objects be solutions to exercises. The professor, who is a great believer of the Alon-Tarsi conjecture, does not want to check all of the exercises himself, so he wants to distribute the exercises over n correctors. Since these correctors may all have a different style of correcting the solutions, the professor thinks it is the fairest if each of them gets precisely one exercise to check from each student. He also has some demands about which solutions can be given to the same corrector, and to his relief the relations turn out to be such that Rota's Basis Conjecture tells him it is indeed possible to redistribute the solutions.

However, the professor does not only want to be able to distribute the solutions properly when he receives them all at the same time, like in the case of a written examination, but also for hand-in homework exercises. From experience, he knows that students likely do not all hand their solutions in at the same time. Since the professor does not want to keep the correctors waiting until all students have handed in their solutions, he wants to assign the solutions to the correctors immediately as he receives them. Again, he is happy to know that the solutions, although he does not know precisely what they look like yet, satisfy relations such that the online version of Rota's Basis Conjecture can be applied. So he can assign the solutions to the correctors. At least, if the number of students is even. Shocked, the professor remembers that it may not at all be possible for an odd number of students and solutions. So he decides to make sure that one of the exercises is so easy that he is certain that each student will deliver the exact same solution. Then he is again able to assign the solutions to the correctors due to a special version of the online version of Rota's Basis Conjecture for odd n.

When could all of this be happening, one might wonder. The answer is simple. This could happen at some point in time after the professor has read this thesis. Not just this thesis, though. More results are required, such as for instance the verification of the Alon-Tarsi conjecture for large enough n, and the discovery of correct algorithms to solve (the online version of) Rota's Basis Conjecture, rather than just their existence.

Of course, this example is just hypothetical and very unlikely to be a useful application of Rota's Basis Conjecture in reality. It just shows that applications for seemingly very theoretical problems can be very well hidden, only to be uncovered much later.

CHAPTER 2

Rota's Basis Conjecture

Closely related to Conjecture 1.1 is a conjecture from 1986 by Alon and Tarsi [**2**] on Latin squares. A Latin square of size n is an $n \times n$ table of numbers, where each row is a permutation of $[n]$ and each column is a permutation of $[n]$. The row (column) sign of a Latin square is the product of the signs of all row (column) permutations; the sign of the Latin square is the product of its row and column signs. The Alon-Tarsi conjecture is about the number of Latin squares that have even sign ($\mathrm{els}(n)$) compared to the number of Latin squares that have odd sign ($\mathrm{ols}(n)$).

CONJECTURE 2.1. *(Alon-Tarsi). Let n be even. Then* $\mathrm{els}(n) \neq \mathrm{ols}(n)$.

In 1995, the numbers of even and odd Latin squares have been computed for $n \leq 8$ [**8**], verifying the Alon-Tarsi conjecture for these n. In Table 1, the respective differences between the numbers of even and odd Latin squares are given [**9**].

n	$\mathrm{els}(n) - \mathrm{ols}(n)$
2	2
4	576
6	199065600
8	1262123552342016000

TABLE 1

1. The relation between the Alon-Tarsi Conjecture and Rota's Basis Conjecture

The relation between Conjecture 2.1 and Conjecture 1.1 is captured by the following theorem.

THEOREM 2.2. *Let n be even and suppose Conjecture 2.1 holds for n. Then Conjecture 1.1 holds for n.*

Huang and Rota already proved this relation in their paper in which Conjecture 1.1 first appeared [**1**]. They proved the equivalence between Conjecture 2.1 and a conjecture by Rota on the bracket algebra. Then they proved that this conjecture on the bracket algebra implies Rota's Basis Conjecture using umbral linear operators.

The essence of their proof is the same as that of the proof by Onn, who in 1997 showed the same relation without using the bracket algebra and umbral linear operators [**3**]. Let $\mathfrak{S}^n$ be the collection of n-tuples of permutations in S_n and

suppose ${}^1W, \ldots, {}^nW$ are nonsingular $n \times n$ matrices. When W is a matrix, let W^j be its j'th column vector. Onn showed that the following identity holds.

$$\sum_{\rho \in \mathfrak{S}^n} \prod_{k=1}^{n} \operatorname{sgn}(\rho_k) \prod_{i=1}^{n} \det({}^1W^{\rho_1(i)}, \ldots, {}^nW^{\rho_n(i)}) = (\operatorname{els}(n) - \operatorname{ols}(n)) \cdot \prod_{j=1}^{n} \det({}^jW).$$

Now if Conjecture 2.1 holds for some even n, then the right-hand side of the equation is nonzero. Hence there must be a nonzero summand on the left-hand side. Suppose that, for each j, the basis B_j consists of the columns of ${}^jW$. Then this nonzero summand on the left-hand side indicates the transversals:

$$\prod_{i=1}^{n} \det({}^1W^{\rho_1(i)}, \ldots, {}^nW^{\rho_n(i)}) \neq 0,$$

so $({}^1W^{\rho_1(i)}, \ldots, {}^nW^{\rho_n(i)})$ are disjoint valid transversals for all i.

In this thesis, a stronger statement than Theorem 2.2 is proven. So here it suffices to say that Theorem 2.2 is a direct consequence of Theorem 4.1(a).

Results for special cases of the Alon-Tarsi Conjecture exist in literature. When p is a prime, results in [**12**] show that the conjecture holds for $p + 1$, and results in [**13**] show that the conjecture holds for $p - 1$. Due to these results, the smallest unsolved case for even n of the Alon-Tarsi Conjecture is $n = 26$ [**13**]. Hence, Rota's Basis Conjecture follows for these dimensions.

The Alon-Tarsi conjecture does not say anything about Rota's Basis Conjecture for odd n. There is, however, a generalization of the Alon-Tarsi conjecture that also covers odd n. This generalization relates to a special case of Rota's Basis Conjecture for odd n, which will be described in section 3 of chapter 4.

2. Rota's Basis Conjecture for matroids

An analogous conjecture to Conjecture 1.1 that concerns bases of a matroid rather than bases of a vector space was made simultaneously by Rota [**1**].

Conjecture 2.3. *(Rota's Basis Conjecture for matroids) Consider (disjoint) bases*

$$(B_1, \ldots, B_n) = ((b_{11}, \ldots, b_{1n}), \ldots, (b_{n1}, \ldots, b_{nn}))$$

in a rank n matroid M. Then there exist $\pi_1, \ldots, \pi_n \in S_n$ such that for all $i = 1, \ldots, n$, $(b_{1,\pi_1(i)}, \ldots, b_{n,\pi_n(i)})$ is a basis of M.

This conjecture is more general than Rota's Basis Conjecture for vector spaces. If M is chosen to be a vector matroid, then the two conjectures are equivalent.

There are several results in literature about Rota's Basis Conjecture for matroids.

In 2007, Geelen and Webb proved the following two theorems [**4**].

Theorem 2.4. *For $n \geq 2\binom{k}{2} + 1$, if $(B_1, \ldots, B_n)$ are disjoint bases in a rank n matroid, then there are k disjoint transversals of $(B_1, \ldots, B_n)$ that are bases.*

The global idea of their proof is to divide the elements of $(B_1, \ldots, B_n)$ into bases $(X_1, \ldots, X_{\binom{k}{2}})$ and a remainder S, in such a way that $|X_i \cap B_i| = n - i$ and $|X_i \cap B_{n-i}| = i$ for all i. See Figure 1. Then they show that k disjoint independent

transversals $(T_1, \ldots, T_k)$ can consecutively be found, where

$$T_t \subseteq (S \cup \bigcup_{i=1}^{\binom{t}{2}} X_i) \setminus \bigcup_{r=1}^{t-1} T_r$$

for each $t = 1 \ldots k$.

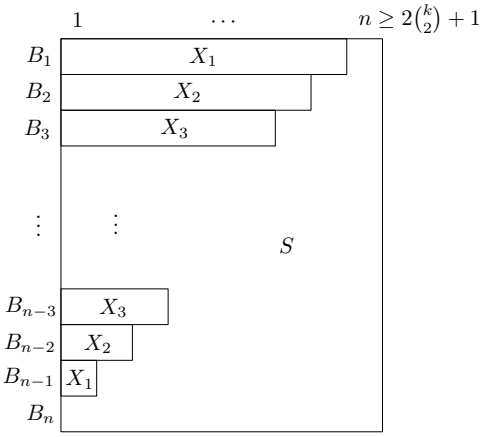


FIGURE 1

The second theorem they prove is the following.

THEOREM 2.5. *If $(B_1, \ldots, B_k)$ are disjoint bases in a rank n matroid where $n \geq \binom{k+1}{2} + 1$, then there are n disjoint independent transversals of $(B_1, \ldots, B_k)$.*

This is proven roughly as follows. See Figure 2. Since k is large enough, disjoint independent sets Z and Z' can be found such that $k + 1 - i = |Z \cap B_i| = k + 1 - |Z' \cap B_i|$. Then they argue that there are $n - k$ disjoint independent transversals of $(B_1, \ldots, B_k)$ that do not contain elements in $Z \cup Z'$. Then, they show that the remaining elements can be divided into transversals of subsets of the bases as shown in Figure 2. Finally, they show that $Z \cup Y_1 \cup \cdots \cup Y_{k-1}$ and $Z' \cup Y_1' \cup \cdots \cup Y_{k-1}'$ can be partitioned into the remaining $2k$ independent transversals, finishing the proof.

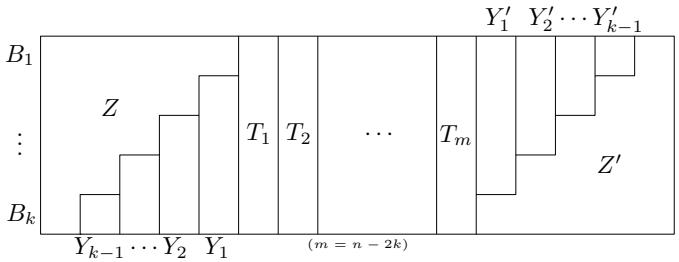


FIGURE 2

In 2006, Geelen and Humphries proved a stronger version of Rota's Basis Conjecture for paving matroids [**5**]. In a paving matroid M of rank n, each circuit

has size n or $n+1$. Their version is stronger in the sense that not all bases are necessarily bases of the same matroid M. Instead, for each i, B_i is a basis of a matroid M_i, where $M_1, \ldots, M_n$ are rank n paving matroids on $\bigcup_{i=1}^{n} B_i$. Then it is shown that there exist transversals $(T_1, \ldots, T_n)$ of $(B_1, \ldots, B_n)$ such that, for all i, T_i is a basis of M_i. If the matroids are chosen such that $M_1 = \ldots = M_n$, then we have Rota's Basis Conjecture for paving matroids.

Geelen and Humphries first show that the Rota's Basis Conjecture for paving matroids is true for $n = 3$. Then by induction on n, they prove that it is true for all $n \geq 3$.

3. Rota's Basis Conjecture: vector spaces versus matroids

Let a set of vectors $\mathcal{V} = \{v_1, \ldots, v_N\}$ be given that span an n-dimensional linear space. Let $\mathcal{I}$ denote the set of subsets of $\mathcal{V}$ in which the vectors are linearly independent. It is well-known that $\{\mathcal{V}, \mathcal{I}\}$ is then a matroid of rank n. Matroids that can be expressed as such a set of vectors with the correct linear independence relations are called linear or representable matroids.

As a consequence, any theorem about Rota's Basis Conjecture for linear spaces implies a similar theorem for linear matroids, and vice versa. This equivalence is particularly useful when theorems about more general matroids are concerned, like in the previous section.

On the other hand, a theorem about Rota's Basis Conjecture for linear spaces does not necessarily imply a similar theorem for non-linear matroids.

CHAPTER 3

Three coloured sets in a partial linear space

In this chapter, we look at a problem involving three coloured sets in a partial linear space in order to prove the vector space version of Rota's Basis Conjecture as well as the matroid version for dimension $n = 3$. The same result was derived in a different manner by Chan [**6**].

A *partial linear space* consists of a set of points and a set of lines satisfying the following axioms.

- Any line is incident with at least two points
- Any pair of distinct points is incident with at most one line.

Let R, W, B (Red, White and Blue) be three sets of $n > 2$ points in a partial linear space such that there is no line that contains more than two points of any of these sets, and let $P := R \sqcup W \sqcup B$ be the disjoint union of these sets. Denote

$$L_P := \{l \subset P \quad | \quad l \cap R \neq \emptyset, l \cap W \neq \emptyset, l \cap B \neq \emptyset, \text{there is a line on which all of the points in } l \text{ and only the points in } l \text{ lie}\}.$$

For convenience, elements of L_P will be called *lines*, rather than the actual lines of the partial linear space. We call a set $T \subset P$ a *valid transversal* of (R, W, B) when $|T \cap R| = |T \cap W| = |T \cap B| = 1$ and $\forall l \in L_P$: $T \not\subseteq l$ (T contains three points, one from each set, that are not aligned). When two (three) points coincide, we call it a *double* (*triple*).

THEOREM 3.1. *Let R, W, B be three sets of $n > 2$ points in a partial linear space such that there is no line that contains more than two points of any of these sets. Then there exist n disjoint valid transversals of (R, W, B).*

A way to visualize this theorem can be seen in Figure 1. In the illustrations hereafter, dots, crosses and circles will be used instead of the coloured disks, so that it is easier to depict doubles and triples. Moreover, while the pictures illustrating the various cases all contain straight lines, this straightness is not demanded by the partial linear space structure.

LEMMA 3.2. *Whenever there are at least three points of each colour, there is always a valid transversal containing any two non-coinciding points of a different colour. In particular, there is always a valid transversal.*

PROOF. Let any two such points be given. Since no three points of the remaining colour are aligned, there must be at least one of them that is not on the line through the two chosen points. This point and the two chosen points form a valid transversal. As not all points can coincide, there is always a pair of non-coinciding points of distinct colours, and hence there is always a valid transversal. □

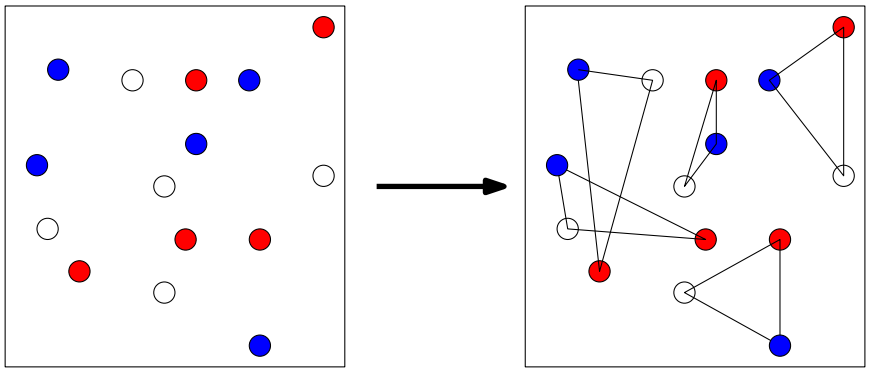
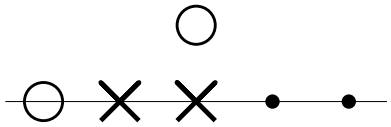


FIGURE 1. An example of five valid disjoint transversals of three sets of points in a two-dimensional affine plane.

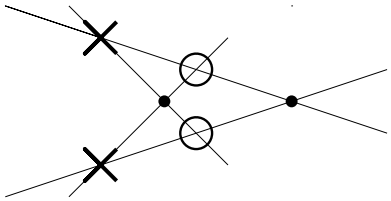
LEMMA 3.3. *Let disjoint valid transversals* $(T_1, \ldots, T_{n-2})$ *of* (R, W, B) *be given such that* $P \setminus T$ *(where* $T = \bigcup_{i=1}^{n-2} T_i$*) satisfies the following properties (also see Figure 2):*

1) $\forall l \in L_{P\setminus T}$, $|l| \leq 4$ *(no more than four points lie on any line);*
2) $(R \setminus T) \cap (W \setminus T) \cap (B \setminus T) = \emptyset$ *(no triples occur in the remainder after removing* T*);*
3) $\max_{l \in L_{P\setminus T}} |l| = 3 \;\Rightarrow$ *Not all* $p \in P \setminus T$ *lie on two lines.*

Then there are transversals T_{n-1}, T_n *of* (R, W, B) *such that* $(T_1, \ldots, T_n)$ *are disjoint valid transversals.*



(A) Property 1.



(B) Property 3.

FIGURE 2. Examples of cases excluded in Lemma 3.3 because they do not allow two valid transversals.

PROOF. Several cases are distinguished.

- *Four points on a line thrice.*
  The only possible configuration contains three non-aligned doubles, and is solved easily. See Figure 3.

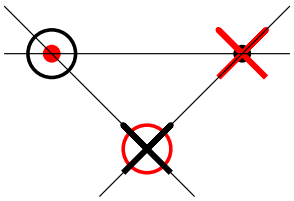


FIGURE 3

- *Four points on a line twice.*
  Since we have only 6 points, this can only happen when we have a double that is on the intersection of both lines. Take one point from that double, and another point from both of the lines. This is a valid transversal, and so is the remainder, because each transversal contains two points that lie on each of the given four-point lines, and one that does not. See Figure 4.

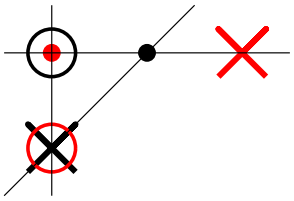


FIGURE 4

- *Four points on a line once.*
  Note that on a line of four points, there is one colour that appears twice, while the other colours appear once. Take two points from this line, of which one from each double if there are any on the line, and of which one from the colour that appears twice. Take the final point outside of the line. Then the remainder is also a valid transversal. See Figure 5.

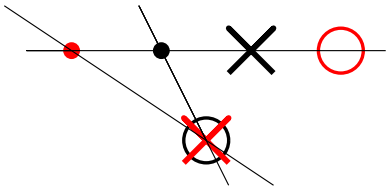


FIGURE 5

- *At most three points on a line.*
  By the third property, there is a point that lies on one or no lines. If it is on a line, pick the transversal containing the two other points on the line. If it is not on a line, pick any valid transversal not containing this point.

Hence in all cases we can find two new disjoint valid transversals. □

LEMMA 3.4. *Let disjoint valid transversals $(T_1, \ldots, T_{n-3})$ of (R, W, B) be given. Then there is a valid transversal T_{n-2} such that $(T_1, \ldots, T_{n-3}, T_{n-2})$ satisfies the conditions in Lemma 3.3.*

PROOF. In this proof, Lemma 3.2 will often be applied without notice.

First look at the general cases when it is not possible to violate the third condition of Lemma 3.3, and distinguish the following cases:

- *At least two triples.*
  Take one point from both triples, and the last point from somewhere else. Then the remainder contains no more triples. The line between the two former triples then contains 4 points. Any line that contained 5 points must have had one of the triples on it, so will now contain at most 4 points. Finally, if any other line contained 6 points, then it must contain all three outside points and a triple, so that only 4 points remain on that line. See Figure 6.

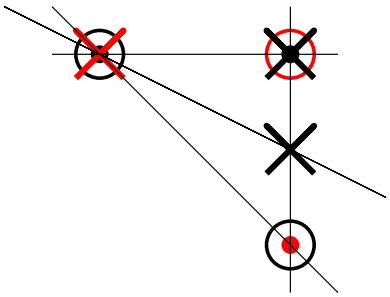


FIGURE 6

- *At most one triple and 6 on a line twice.*
  Note that the intersection of both lines must be a triple, as there are only 9 points. Hence taking one point from the triple, and one other point from each of the lines does the job.
- *At most one triple and 6 on a line once.*
  There can be at most one line of 5 that does not contain a triple, since there are only three points outside of the line of 6. If there is such a line of 5, these three points must hence all be aligned with a double on the line of 6. Take two points on the line of 6, one of which from the triple if there is one on the line, and one outside. If any line of 5 contained a triple, then the number of points on it will be reduced to at most 4. If a line of 5 did not, then it contained all of the points outside of the line of 6, and will thus contain at most 4 points as well. See Figure 7.
- *One triple and no 6 on a line.*
  Note that there is at most one line containing 5 points that does not contain the triple. Hence taking at least one point from the line of 5 not containing the triple, if it exists, and one point from the triple does the job.

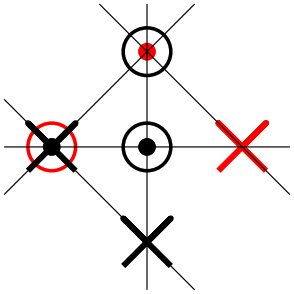


FIGURE 7

- *No triples and no 6 on a line.*
  If there are no lines of 5, pick any transversal. If there is one line of 5, pick a transversal containing a point on the line of 5. If there are two lines of 5, they must intersect in a point as there are only 9 points. Hence pick a transversal containing the point on the intersection. Finally, if there are three lines of 5, they must form a triangle and intersect in doubles. Taking one point from two intersection points, and the last point not on the line that contains both of these points does the job. See Figure 8.

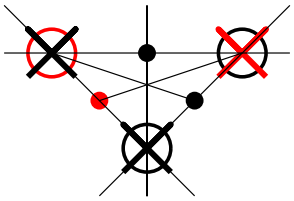


FIGURE 8

This covers all possibilities. Now assume it is possible to violate the third condition of Lemma 3.3. Then the plane must look like Figure 2b (the original points) with a valid transversal of (R, W, B) added (the newly added points). Note that there can never be a line of 6 or a triple in this case.

- *There is a line of 4 or at most one line of 5.*
  One or two points must be added to one of the existing lines, for otherwise the three new points are aligned, which does not happen by assumption. On this line of 4 or 5, there must be a colour that does not appear twice on the line. If we now pick that point along with a newly added point on the line, and a point outside of the line, then that is a valid transversal. See Figure 9. Removing this transversal does the job: After the removal, the former line of 4 or 5 is eliminated, and any remaining original point on this former line can only lie on at most one line.
- *There are several lines of 5.*
  Note that any two lines of 5 must in this case intersect in an original point. The same procedure as previously can be applied to any line of 5, but when picking the newly added point on the line, it should be on

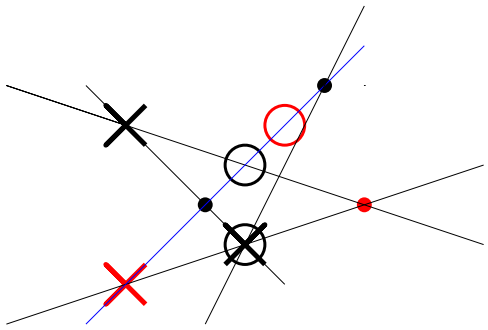


FIGURE 9

another line of 5 if the first point was not already. This ensures that no lines of 5 remain. See Figure 10.

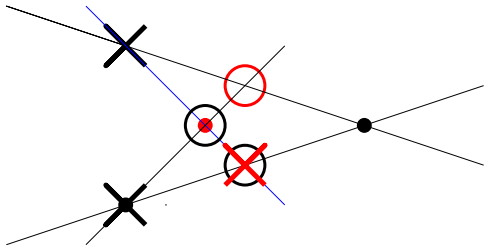


FIGURE 10

- *There are no lines of 4 or 5.*
  Then we can simply take a transversal with two points from the newly added points and one from the original points, when the final newly added point will not be on any line.

Hence in all possible cases we can find a transversal in order to satisfy the conditions in Lemma 3.3. □

Proof of Theorem 3.1. By Lemma 3.2, we can pick disjoint valid transversals $(T_1, \ldots, T_{n-3})$. By Lemma 3.4, we can then pick T_{n-2} such that $(T_1, \ldots, T_{n-2})$ satisfies the conditions of Lemma 3.3, which then yields T_{n-1} and T_n such that $(T_1, \ldots, T_n)$ are disjoint valid transversals. □

1. Rota's Basis Conjecture in dimension three

Let M be a rank 3 matroid on 9 points such that there exist three disjoint bases (B_1, B_2, B_3). If M is a linear matroid, then each of the points can be associated with a vector in $\mathbb{R}^3$. Consider a two-dimensional affine subspace A of $\mathbb{R}^3$ (not containing the origin) that is not parallel to any of these vectors. Then the linear span of each vector intersects A in precisely one point. Let (R, W, B) be the points of intersection of the vectors in bases (B_1, B_2, B_3) respectively. Then, by Theorem 3.1, there are disjoint valid transversals of (R, W, B). Hence, Rota's Basis Conjecture for $n = 3$ follows.

Suppose, on the other hand, that M is a non-linear rank 3 matroid on 9 points such that there exist three disjoint bases. For these matroids, too, it is possible to embed them in a partial linear space:

THEOREM 3.5. *Let M be a rank 3 matroid on 9 points such that there exist three disjoint bases. Then M can be embedded in a partial linear space.*

PROOF. Let the points of the partial linear space be the points of M, where two points coincide if they are dependent, and let the lines be induced by the dependence relations between the points. Since M contains three disjoint bases, there are no loops, and hence by construction each line contains at least two points.

So it remains to show that any two distinct points lie on at most one line. Suppose a, b, c and d are four points in the partial linear space, where a and b do not coincide. Suppose that a, b and c are on a line, and a, b and d are on a line. Then it should follow that a, b, c and d are all on the same line, for otherwise there are two different lines through a and b. See Figure 11.

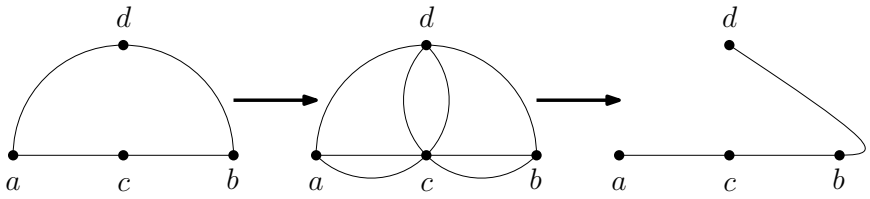


FIGURE 11

Hence we need to show that moreover a, c and d are on a line, and that b, c and d are on a line. But this follows from the matroid exchange property: suppose that a, c and d are not a line, and hence independent. Since a and b do not coincide, they are also independent. Hence there is a point in $\{a, c, d\}$ that can be added to $\{a, b\}$ that maintains independence. However, by the assumptions, none of these points accomplish that. Thus a, c and d are on a line. Analogously, b, c and d are on a line.

So all three-element subsets of $\{a, b, c, d\}$ are aligned. Hence the dependencies between all of these subsets can be integrated in one line through all four elements. Thus, two distinct points lie on at most one line. □

So it turns out that all matroids of rank 3 on 9 points that can be partitioned into three disjoint bases can be embedded in a partial linear space, where the bases can be identified with the sets (R, W, B) in Theorem 3.1. Hence, for $n = 3$, Rota's Basis Conjecture for matroids follows. Consequently, Rota's Basis Conjecture for vector spaces is also true for $n = 3$.

CHAPTER 4

Draisma's online version of Rota's Basis Conjecture

In order to gain a better understanding of Rota's Conjecture, it is useful to look at a stronger version of it. Suppose that not all n bases are given at the same time, but instead one by one. In each step, we then need to fix a permutation for the new basis, based only on the bases given and permutations chosen in the previous steps.

For even n, the online version of Rota's Basis Conjecture turns out to be conditioned on the same conjecture as Rota's Basis Conjecture itself: the Alon-Tarsi conjecture. For odd n, however, the online version is false. The following theorem will elaborate on the details.

THEOREM 4.1. *Let $n > 1$. Consider bases*

$$(B_1, \ldots, B_n) = ((b_{11}, \ldots, b_{1n}), \ldots, (b_{n1}, \ldots, b_{nn}))$$

of an n-dimensional vector space V over $\mathbb{C}$.

(a) *If n is even and the Alon-Tarsi conjecture is true for n, then there exists an n-step online algorithm as follows: in the k'th step, the algorithm is given B_k and is required to fix $\pi_k \in S_n$. After the last step, the algorithm has found $(\pi_1, \ldots, \pi_n)$ such that for all $1 \leq j \leq n$, $(b_{1,\pi_1(j)}, \ldots, b_{n,\pi_n(j)})$ is a basis of V.*

(b) *If n is odd, then there exists no such algorithm.*

The relation between Latin squares and (the online version of) Rota's Basis Conjecture lies in the permutations. At first sight, it might seem that this relation is the correspondence between the permutation chosen for a basis and a permutation of the numbers 1 to n in a row of a Latin square. The relation, however, is not that simple: Latin squares require columns to be permutations as well, while our conjecture does not demand that as long as the columns are bases. For suitable bases, we might even leave each basis in its original order to solve the problem. The relation is more subtle, and will be unveiled in the proof of this theorem.

Before we proceed with the proof, it is good to understand what exactly an algorithm can do in the n'th (last) step. If, after the $(n-1)$'st step, an algorithm can find a correct π_n for each possible basis B_n, then it will succeed. The following theorem pinpoints exactly when this is the case.

THEOREM 4.2. *Let V be an n-dimensional vector space over a field K. Suppose that $V_1, \ldots, V_n \subset V$ are $(n-1)$-dimensional linear subspaces of V. Then:*

For all bases $b_1, \ldots, b_n$ of V, there exists some $\pi \in S_n$ such that for all $i \in \{1, \ldots, n\}$: $b_{\pi(i)} \notin V_i$ (i.e. $\langle V_i, b_{\pi(i)} \rangle = V$) if and only if for all $k \in \{1, \ldots, n\}$ and distinct $i_1, \ldots, i_k \in \{1, \ldots, n\}$: $\dim(V_{i_1} \cap \ldots \cap V_{i_k}) = n - k$.

PROOF. Assume that for some k and $i_1, \ldots, i_k$ we have that $\dim(V_{i_1} \cap \ldots \cap V_{i_k}) \geq n - k + 1$. Pick a basis of V such that $b_1, \ldots, b_{n-k+1} \in V_{i_1} \cap \ldots \cap V_{i_k}$. This leaves only $k-1$ other basis vectors, and as such not all of the V_{i_j} can be paired with a basis vector that it does not contain. Furthermore, the V-codimension of each of the V_i is 1, so $\dim(V_{i_1} \cap \ldots \cap V_{i_k}) \geq n - k$. This proves the 'only if' part.

For the converse, assume that $\dim(V_{i_1} \cap \ldots \cap V_{i_k}) = n - k$ for all $k, i_1, \ldots, i_k$. Let a basis $b_1, \ldots, b_n$ of V be given. The desired result will now be derived by induction.

Certainly, we can match b_1 to one of the V_i, since $\dim(V_1 \cap \ldots \cap V_n) = 0$.

Assume that for some $0 < r < n$, $b_1, \ldots, b_r$ are matched to (without loss of generality) $V_1, \ldots, V_r$ in a way that each matched pair spans V. Now consider b_{r+1}. If it can be matched to some V_i where $i > r$, then we are done. So assume b_{r+1} can only be matched to (again without loss of generality) $V_1, \ldots, V_s$ for some $0 < s \leq r$. Define $N^0 := \{V_1, \ldots, V_s\}$ and

$$N^{p+1} := \{V_i \mid \text{some } b_j \text{ that is matched to a space in } N^p \text{ can be matched to } V_i\}.$$

Note that whenever $V_{r+1}, \ldots, V_n \notin N^p$, we have that $|N^{p+1}| > |N^p|$, since otherwise we would have $|N^p|+1$ independent vectors that all lie in the intersection of $n-|N^p|$ of the V_i, which was assumed to have dimension $|N^p|$. Hence there must be a smallest p for which $\{V_{r+1}, \ldots, V_n\} \cap N^p \neq \emptyset$. So let $W \in \{V_{r+1}, \ldots, V_n\} \cap N^p$. Since $W \in N^p$, there is a sequence $b_{j_0}, \ldots, b_{j_{p-1}}$ such that for each $0 \leq q < p-1$, b_{j_q} can be matched to the space that $b_{j_{q+1}}$ is matched to, where each b_{j_q} is matched to a space in N^q, and $b_{j_{p-1}}$ (matched to a space in N^{p-1}) can be matched to W. Now replace b_{j_0} by b_{r+1}, $b_{j_{q+1}}$ by b_{j_q} for all q and match $b_{j_{p-1}}$ to W. Then this gives a matching between $b_1, \ldots, b_{r+1}$ and $V_1, \ldots, V_r, W$ such that each pair spans V, as desired.

So by induction, the $b_1, \ldots, b_n$ can all be matched to $V_1, \ldots, V_n$ in such a way that each pair spans V. This matching gives rise to a $\pi \in S_n$ such that $b_{\pi(i)} \notin V_i$ for all $1 \leq i \leq n$. □

We are now ready to prove Theorem 4.1.

Proof of Theorem 4.1(a). The structure of the proof is roughly as follows. First, we model the problem by means of exterior algebra. Then we construct hypersurfaces H_k that define the 'bad' solutions in our model. Finally, we show that the Alon-Tarsi conjecture assures us that we can in fact stay outside of the bad solution set, hence giving us a good solution.

The algorithm is required to construct n ordered sets of vectors step by step, where each set should remain independent after each step. We use the exterior algebra to model these sets. An ordered set $(v_1, \ldots, v_k)$ is modeled as $v_1 \wedge \ldots \wedge v_k$. The alternating property of the exterior algebra makes sure that whenever there is a dependency between the vectors given by $v_k = \sum_{i=1}^{k-1} c_i v_i$, we get

$$\begin{aligned} v_1 \wedge \ldots \wedge v_k &= \\ v_1 \wedge \ldots \wedge v_{k-1} \wedge (\sum_{i=1}^{k-1} c_i v_i) &= \\ \sum_{i=1}^{k-1} c_i (v_1 \wedge \ldots \wedge v_{k-1} \wedge v_i) &= 0. \end{aligned}$$

Whenever $v_1, \ldots, v_k$ are independent and $w_1, \ldots, w_k$ span the same space, we get

$$w_1 \wedge \ldots \wedge w_k = (\sum_{i=1}^{k} c_{1,i} v_i) \wedge \ldots \wedge (\sum_{i=1}^{k} c_{k,i} v_i),$$

which is equal to some nonzero scalar multiple of $v_1 \wedge \ldots \wedge v_k$. In this sense, when $v_1, \ldots, v_k$ are independent, their exterior product modulo scalars can be identified with the vector subspace they span.

So in the context of this model, the algorithm needs to make sure that each of the n exterior products of vectors remain nonzero after each step.

In order to model the step of adding a vector to a space, consider the following map:

$$\Psi_{k,\pi} : (\bigwedge^{k-1} V)^n \times V^n \to (\bigwedge^{k} V)^n,\ (\omega, B) \mapsto (\omega_1 \wedge B_{\pi(1)}, \ldots, \omega_n \wedge B_{\pi(n)}).$$

$\Psi_{k,\pi}$ models adding the $\pi(i)$'th vector of the given basis B to the i'th space in the k'th step of the algorithm. Note that this map is also well-defined when B is not a basis.

Now we will construct a hypersurface H_k in $(\bigwedge^k V)^n$ that defines the tuples of spaces after the k'th step that should be avoided by the algorithm. This H_k should clearly contain all n-tuples that have one or more entries equal to zero, since a zero entry corresponds to a linear dependence in the corresponding space. But there are more tuples that need to be avoided, because for general tuples that do not contain a zero, there can be bad choices of remaining bases that force any algorithm to end up with a zero in the last step. For instance, all n-tuples of spaces that do not satisfy the conditions on the dimensions of their intersections in Theorem 4.2 should be in H_{n-1} (and only those).

In order to capture this constraint, we need the following relation to hold between H_k and H_{k-1}: if for some $(\omega, B) \in (\bigwedge^{k-1} V)^n \times V^n$, with B a basis, it holds that for all $\pi \in S_n$, $\Psi_{k,\pi}(\omega, B)$ is in H_k, then ω must already be in H_{k-1}.

Let us now construct these hypersurfaces H_k. In order to do this, we first switch to tensor products rather than working with exterior powers. We can do this, because the dual of the canonical projection $V^{\otimes k} \to \bigwedge^k V$ identifies alternating tensors in $(V^*)^{\otimes k}$ with linear functions on $\bigwedge^k V$.

So consider, analogously to Ψ, the map

$$\Phi_{k,\pi} : (V^{\otimes k-1})^{\otimes n} \times V^n \to (V^{\otimes k})^{\otimes n},\ (t, B) \mapsto (t_1 \otimes B_{\pi(1)}, \ldots, t_n \otimes B_{\pi(n)}).$$

Take $x_i \in V^*$ as the dual of the standard basis vector $e_i \in V$. Now denote the tensor $(x_{a_{11}} \otimes \ldots \otimes x_{a_{k1}}) \otimes \ldots \otimes (x_{a_{1n}} \otimes \ldots \otimes x_{a_{kn}})$ by the tableau

$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{k1} & \cdots & a_{kn} \end{bmatrix}, \tag{4.1}$$

where $a_{ij} \in \{1, \ldots, n\}$ for all $i \in \{1, \ldots, k\}$ and $j \in \{1, \ldots, n\}$. Note that all tableaux of this form together span the space $((V^*)^{\otimes k})^{\otimes n}$. Consider the linear map

$$\Theta_k : ((V^*)^{\otimes k})^{\otimes n} \to ((V^*)^{\otimes k-1})^{\otimes n},$$

$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{k1} & \cdots & a_{kn} \end{bmatrix} \mapsto \sigma \cdot \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{k-1,1} & \cdots & a_{k-1,n} \end{bmatrix}.$$

Here, σ is equal to zero if $(a_{k1}, \ldots, a_{kn})$ is not a permutation of $[n]$, and equal to its sign in S_n if it is a permutation of $[n]$.

Set $F_n := \det^{\otimes n}$, an n-linear function in $((V^*)^{\otimes n})^{\otimes n}$. Set $F_{k-1} := \Theta_k F_k$. Since each Θ_k is linear, the n-linearity property of F_k is preserved by Θ_k. Note that F_n is an n-alternating tensor, and due to linearity of Θ_k, so are all F_k. For all k, define $H'_k \subset (V^{\otimes k})^{\otimes n}$ as the zero set of F_k.

By n-linearity of F_k for each k, the n-tuples of tensors of which some entry is zero are contained in H'_k.

Next, suppose that there is some basis B such that for all $\pi \in S_n$, $F_k \Phi_{k,\pi}(t, B) = 0$. Here it is convenient to assume that $B = (e_1, \ldots, e_n)$. This is indeed a valid assumption: the set $\{t \mid F_k(t) = 0\}$ is $\mathrm{GL}(V)$-invariant. We also know that for given $\pi \in S_n$ and $g \in \mathrm{GL}(V)$, $g\Phi_{k,\pi}(t, B) = \Phi_{k,\pi}(gt, gB)$. So if $F_k \Phi_{k,\pi}(t, B) = 0$, then also $F_k \Phi_{k,\pi}(gt, gB) = F_k g \Phi_{k,\pi}(t, B) = 0$. We can now rewrite F_k as follows:

$$F_k = \sum_{\pi \in S_n} c_\pi \cdot \left[\begin{array}{ccc} | & & | \\ \hat{t_1} & \cdots & \hat{t_n} \\ | & & | \\ \hline \pi(1) & \cdots & \pi(n) \end{array} \right] + R_t.$$

Here, R_t consists of the part of F_k that maps $\Phi_{k,\pi}(t, \cdot)$ to zero. Hence $\Theta_k R_t$ also maps t to zero. The upper part of the tableau, $(\hat{t_1}, \ldots, \hat{t_n})$, represents the $\hat{t} \in ((V^*)^{\otimes k-1})^{\otimes n}$ for which $F_k \Phi_{k,\pi}(\hat{t}, B') \neq 0$ for some B'. By multilinearity of the tensor product, we indeed get a pure tensor for each π in this representation of F_k. Now we obtain:

$$\begin{aligned} F_{k-1} &= \Theta_k F_k \\ &= \sum_{\pi \in S_n} c_\pi \cdot \Theta_k \left[\begin{array}{ccc} | & & | \\ \hat{t_1} & \cdots & \hat{t_n} \\ | & & | \\ \hline \pi(1) & \cdots & \pi(n) \end{array} \right] + \Theta_k R_t \\ &= \sum_{\pi \in S_n} c_\pi \cdot \mathrm{sgn}(\pi) \cdot \begin{bmatrix} | & & | \\ \hat{t_1} & \cdots & \hat{t_n} \\ | & & | \end{bmatrix} + \Theta_k R_t \end{aligned}$$

Now the assumption that for all $\pi \in S_n$ we have that $F_k \Phi_{k,\pi}(t, B) = 0$ implies that all c_π are equal to zero. Thus it follows that $F_{k-1} t = 0$.

Moreover, none of the H'_k is equal to the entire space $(V^{\otimes k})^{\otimes n}$ if the Alon-Tarsi conjecture holds. To see this, consider $F_0 = \Theta_1 \circ \Theta_2 \circ \ldots \circ \Theta_n F_n$. Since $F_n = \det^{\otimes n}$, F_n is a linear combination of tensors of the form (4.1) with $k = n$, where the columns are permutations of $[n]$ and where the coefficient of each such tensor is the product of the signs of the permutations represented by its columns. Then each Θ_k, $k = n, \ldots, 1$, kills a tensor if the k'th row is not a permutation of $[n]$, and removes the row while contributing its sign in S_n when it is a permutation of $[n]$. Then F_0, which is a scalar, has a nonzero contribution only from each tableau in F_n of which both all columns and all rows are permutations of $[n]$. Tableaux with this property are also known as Latin squares. This contribution is exactly

the product of the column signs and the row signs, which is exactly the sign of the Latin square. So F_0 is equal to the number of even Latin squares minus the number of odd Latin squares. Hence, if the Alon-Tarsi conjecture is true, F_0, and thus all F_k, are not equal to zero.

For each k, we can now define H_k as the canonical projection of H'_k onto $(\bigwedge^k V)^n$. Then H_k satisfies the properties we demanded.

Finally, we get to the online algorithm. Start with some $\omega_0 \in (\bigwedge^0 V)^n \setminus H_0$, which exists when the Alon-Tarsi conjecture holds. In the k'th step, we have $\omega_{k-1} \in (\bigwedge^{k-1} V)^n \setminus H_{k-1}$, and a basis B_k of V. Pick π_k in such a way that $\omega_k := \Psi_{k,\pi_k}(\omega_{k-1}, B_k) \not\subseteq H_k$. This is possible since $\omega_{k-1} \notin H_{k-1}$. Then proceed to the next step. In the end, we have $\omega_n \notin H_n$. But for each $i = 1, \ldots, n$, $\omega_{n,i} = b_{1,\pi_1(i)} \wedge \ldots \wedge b_{n,\pi_n(i)}) \neq 0$. Thus $(b_{1,\pi_1(i)}, \ldots, b_{n,\pi_n(i)})$ must be a basis of V, and the algorithm has succeeded. □ □

Proof of Theorem 4.1(b). In this part of the proof, for each algorithm an instance of bases $(B_1, \ldots, B_n)$ will be given that makes the algorithm fail to produce bases. Globally, this will be done as follows. First, regardless of the algorithm, we pick $n-2$ bases for which each algorithm chooses permutations. Then, based on these choices, a $(n-1)$'st basis is constructed in such a way that the algorithm is unable to choose a permutation for this basis in such a way that it can deal with every possible n'th basis.

Choose $B_1 = \ldots = B_{n-2} = (e_1, \ldots, e_n)$, the standard basis. Let $\pi_1, \ldots, \pi_{n-2}$ be given such that $(b_{1,\pi_1(i)}, \ldots, b_{n-2,\pi_{n-2}(i)})$ span codimension 2 spaces for $1 \leq i \leq n$. Each of these spaces misses two standard basis vectors. A graph can now be constructed with as vertex set $\{e_k \mid 1 \leq k \leq n\}$ and with an edge (e_k, e_l) for each of the spaces that misses e_k and e_l. This graph is regular of degree 2, since each e_k is missing in precisely two of the spaces. Hence the graph must be a collection of disjoint cycles of size at least 2. Since n is odd, at least one of the cycles contains an odd number of vertices m. Consider the spaces corresponding to the edges of this odd-length cycle. Without loss of generality, these are the first m spaces, missing $(e_1, e_2), (e_2, e_3), \ldots, (e_{m-1}, e_m)$ and (e_m, e_1) respectively.

Now consider the matrix

$$X = \begin{pmatrix} \mathcal{V} & 0 \\ 0 & I \end{pmatrix}$$

where $\mathcal{V}$ is an $m \times m$ matrix with entries $\mathcal{V}_{k,l} = \zeta^{kl}$. Here $\zeta \in \mathbb{C}$ is a primitive m'th root of unity. The columns of this matrix form the basis B_{n-1}. Now let $\pi_{n-1} \in S_n$ be given such that each column space $V_i := \langle b_{1,\pi_1(i)}, \ldots, b_{n-1,\pi_{n-1}(i)} \rangle$ is $(n-1)$-dimensional. In particular, π_{n-1} maps the first m columns of X in some order to the first m spaces, since these columns are contained in the other $n-m$ spaces. Now we argue that regardless of this choice, there will be a final basis B_n for which there exists no suitable π_n.

For each $i = 1, \ldots, m$, consider the orthogonal complement that is spanned by a normal vector z_i with respect to the inner product of the respective space V_i. Explicitly, such a normal vector must be perpendicular to all e_l with $l \neq i, i+1$ (the indices are regarded modulo m with offset 1, so that $m+1 \equiv 1$), and to the $\pi_{n-1}(i)$'th column of X, which is

$$(\zeta^{\pi_{n-1}(i)}, \zeta^{2\pi_{n-1}(i)}, \ldots, \zeta^{m\pi_{n-1}(i)}, 0, \ldots, 0)^T.$$

Hence $z_i = ae_i + be_{i+1}$ such that $a\zeta^{i\pi_{n-1}(i)} + b\zeta^{(i+1)\pi_{n-1}(i)} = 0$. So we can pick $z_i = \zeta^{\pi_{n-1}(i)}e_i - e_{i+1}$. These z_i turn out to be linearly dependent:

$$\det\begin{pmatrix} \zeta^{\pi_{n-1}(1)} & & & -1 \\ -1 & \zeta^{\pi_{n-1}(2)} & & \\ & \ddots & \ddots & \\ & & -1 & \zeta^{\pi_{n-1}(m)} \end{pmatrix} = \zeta^{\sum_{k=1}^{m}\pi_{n-1}(k)} + (-1)^m.$$

Now since

$$\sum_{k=1}^{m}\pi_{n-1}(k) = \sum_{k=1}^{m} k = \frac{1}{2}m(m+1)$$

is divisible by m due to $m+1$ being even, the determinant of the matrix is

$$\zeta^{\sum_{k=1}^{m}\pi_{n-1}(k)} + (-1)^m = \zeta^{\frac{m+1}{2}m} - 1 = 1 - 1 = 0.$$

So we find that $\dim(V_1^\perp + \ldots + V_m^\perp) < m$. Now we claim that $(V_1^\perp + \ldots + V_m^\perp)^\perp = V_1 \cap \ldots \cap V_m$. Indeed, if v is in the left-hand side, then it must be perpendicular to all $V_1^\perp, \ldots, V_m^\perp$. Hence $v \in (V_1^\perp)^\perp \cap \ldots \cap (V_m^\perp)^\perp$, which equals the right-hand side. Conversely, if w is in the right-hand side, then w is perpendicular to each $V_1^\perp, \ldots, V_m^\perp$, and thus to the sum of these spaces. As such,

$$\dim(V_1 \cap \ldots \cap V_m) = \dim((V_1^\perp + \ldots + V_m^\perp)^\perp) > n - m.$$

By Theorem 4.2, this completes the proof. □

If $n = 3$, then the above counterexample is unique up to symmetries and the choice of ζ as the primitive 3^{rd} root of unity.

Now if n is a multiple of m for some $m < n$ for which the Alon-Tarsi conjecture holds, then it turns out that at least $m+1$ bases can be ordered correctly by means of an online algorithm. First consider the following lemma.

LEMMA 4.3. *Let m be an even divisor of n. Suppose the Alon-Tarsi conjecture holds for m. Then F_{n-m} as in the proof of Theorem 4.1(a) is nonzero.*

PROOF. It suffices to show that there exists a basis vector in $((V^*)^{\otimes n-m})^{\otimes n}$ that has a non-zero coefficient in F_{n-m}. For this, consider a tableau of the following form:

$$T = \begin{bmatrix} A_1 & A_2 & \cdots & A_{\frac{n}{m}} \end{bmatrix}.$$

Each A_i, $i = 1, \ldots, \frac{n}{m}$, is an $(n-m) \times m$ subtableau containing precisely the numbers $[n] \setminus \{(i-1)m+1, \ldots, im\}$ in each column. Note that, while it is not required, it is possible for the rows of T to be a permutation of $[n]$.

The coefficient of this tableau in F_{n-m} is equal to the sum of the signs of the $m \times n$ partial Latin squares with which it can be extended in such a way that no column contains any number more than once. However, due to the properties of its subtableaux, T can only be extended to $n \times n$ squares of the form

$$\begin{bmatrix} A_1 & A_2 & \cdots & A_{\frac{n}{m}} \\ L_1 & L_2 & \cdots & L_{\frac{n}{m}} \end{bmatrix},$$

where L_i, $i = 1, \ldots, \frac{n}{m}$, is an $m \times m$ Latin square on the numbers $\{(i-1)m + 1, \ldots, im\}$. Hence the sum of the signs of all of these extensions is equal to $(\text{els}(m) - \text{ols}(m))^{\frac{n}{m}}$, up to a sign. This was assumed to be nonzero. □

Now we can derive a result that, for some values of n, is stronger than the results in [**4**].

THEOREM 4.4. *Let $n > 1$ be even. Let $m < n$ be an even divisor of n. Consider bases*

$$(B_1, \ldots, B_{m+1}) = ((b_{11}, \ldots, b_{1n}), \ldots, (b_{m+1,1}, \ldots, b_{m+1,n}))$$

of an n-dimensional vector space V.

If the Alon-Tarsi conjecture holds for m, then there exists an $(m+1)$-step online algorithm as follows: in the k'th step, the algorithm is given B_k and is required to fix $\pi_k \in S_n$. After the last step, the algorithm has found $(\pi_1, \ldots, \pi_{m+1})$ such that for all $1 \leq j \leq n$, $(b_{1,\pi_1(j)}, \ldots, b_{m+1,\pi_{m+1}(j)})$ are linearly independent in V.

PROOF. Suppose the Alon-Tarsi conjecture holds for m. It follows from Lemma 4.3 that $F_{n-m} \neq 0$. Let T be a tableau that has nonzero coefficient in F_{n-m} such that its first row is a permutation of $[n]$. Due to symmetry, B_1 can be assumed to be the standard basis. Choose π_1 according to the first row of T: $\pi_1(i)$ is chosen to be the j such that $T_{1,j} = i$. Nonzeroness of the coefficient of T in F_{n-m} implies that, whenever there are multisets of basis vectors $(B'_1, \ldots, B'_{n-m})$ and permutations $(\pi'_1, \ldots, \pi'_{n-m})$ such that $b'_{i,\pi'_i(j)} = e_{T_{ij}}$, there is an online algorithm that can assign any m bases of V by means of m permutations $(\pi'_{n-m+1}, \ldots, \pi'_n)$ in such a way that $(b'_{1,\pi'_1(j)}, \ldots, b'_{n,\pi'_n(j)})$ are linearly independent in V for all j.

Clearly, this online algorithm that finds m such permutations would also work in the simpler case, where there is only the linear independence with $b'_{1,\pi'_1(j)} = b_{1,\pi_1(j)}$ to be taken into account for each j. Hence, there is an online algorithm that can find n disjoint independent transversals of at least $m + 1$ bases, which is what needed to be shown. □

1. Hypersurfaces for odd dimensions

In the proof of Theorem 4.1(a), the assumption that n is even is only used when Conjecture 2.1 is applied. This conjecture assures that whenever n is even, the defining function F_k of the hypersurface H'_k is nonzero for each k. Such assertions do not exist for odd n. In fact, it can be shown that the only non-zero defining functions are F_{n-1} and F_n.

Recall that F_k is an element of $((V^*)^{\otimes k})^{\otimes n}$ that can thus be represented as a linear combination of tableaux of the form (4.1), and that $F_{k-1} = \Theta_k F_k$.

In F_{n-2}, all tableaux must have columns that contain each element of $[n]$ no more than once, since $F_n = \det^{\otimes n}$. A coefficient of such a tableau can only be nonzero if it can be extended to an $n \times n$ tableau of which the columns are permutations of $[n]$ and the last two rows are permutations of $[n]$. Denote by S_T the set of pairs of permutations that can extend a tableau T. The coefficient of T is then given by

$$c_T := \sum_{(\sigma,\tau)\in S_T} \mathrm{sgn}(\sigma)\mathrm{sgn}(\tau)c_{T,\sigma,\tau},$$

where $c_{T,\sigma,\tau}$ is the coefficient of T with last two rows σ and τ added, which is equal to the product of the column signs. However,

$$\mathrm{sgn}(\sigma)\mathrm{sgn}(\tau)c_{T,\sigma,\tau} = -\mathrm{sgn}(\tau)\mathrm{sgn}(\sigma)c_{T,\tau,\sigma},$$

since exchanging two rows in an odd-sized tableau inverts the column sign. Hence $c_T = 0$.

Of course F_n is nonzero. Then F_{n-1} cannot be zero either, because there is a one-to-one correspondence between tableaux that have a nonzero coefficient in F_n

and tableaux that have a nonzero coefficient in F_{n-1}. This is because there can only be one possible permutation that can be added as the last row to a tableau in F_{n-1} in order to match a tableau in F_n with nonzero coefficient, since each column in the tableau in F_{n-1} misses only one number.

The fact that $F_k = 0$ for $k \leq n-2$ is precisely the reason that the proof of Theorem 4.1(a) does not apply to odd n. However, if we make an extra assumption on the given bases, we can adapt the proof to make it work for odd n. This will be discussed in section 3.

2. Decomposed bases

Assume $V = V_1 \oplus \ldots \oplus V_d$. Suppose that it is given that all bases respect this decomposition: the vectors of each basis B_k can be split into bases B_k^l of V_l, for $l = 1, \ldots, d$. We will use the notation $B_k = B_k^1 \oplus \ldots \oplus B_k^d$. This assumption on the bases allows us to slightly simplify the task of finding an online algorithm.

Let $P \subseteq [n]^2$ be a set of positions in an $n \times n$ grid. P is called a *pattern* of size m if $|P \cap (\{i\} \times [n])| = m$ for each i. In other words, P is a pattern of size m if P contains precisely m positions from each row.

THEOREM 4.5. *Let $V = V_1 \oplus \ldots \oplus V_d$ be an n-dimensional vector space. For $k = 1, \ldots, n$, let B_k be a basis of V such that $B_k = B_k^1 \oplus \ldots \oplus B_k^d$. Furthermore, let, for $l = 1, \ldots, d$, a pattern P_l of size $\dim(V_l)$ be given such that $P_1, \ldots, P_d$ are pairwise disjoint.*

Suppose that for each l there is an n-step online algorithm as follows: in the k'th step, the algorithm is given B_k^l and is required to fix an assignment of the vectors in B_k^l to the positions of P_l in row k. After the last step, the algorithm has assigned a vector to all positions of P_l in such a way that the vectors assigned to the same column are independent.

Then simultaneously running these d algorithms results in an assignment of vectors to all positions in $[n]^2$ in such a way that the vectors assigned to the same column are independent.

The d algorithms combined form an algorithm that precisely fits the description of an algorithm in Theorem 4.1(a).

PROOF. Note that the existence of a working algorithm for each l implies that not only each row of $[n]^2$ contains exactly $\dim(V_l)$ positions in P_l, but also each column. Otherwise there would be a column that contains more than $\dim(V_l)$ positions in P_l. The vectors assigned to this column cannot be independent.

The vectors assigned to a column by algorithm l thus precisely span V_l. Hence, combining the algorithms, each column has a subset of positions to which vectors are assigned that span V_l for all l. As a consequence, the vectors assigned to a column span $V_1 \oplus \ldots \oplus V_d = V$. □

Theorem 4.5 allows us to split the task of finding an online algorithm that orders n bases consisting of n vectors, into smaller tasks of finding algorithms that do the same for n bases of a smaller space and some partitioning of $[n]^2$ into patterns, if the decomposition of V which the bases respect is given beforehand.

3. Online Rota for odd dimensions

Zappa found a generalization of Conjecture 2.1 that says something about both even and odd n, rather than just even n [**7**]. Consider Latin squares with ones on the diagonal. Call the number of even and odd Latin squares with that property $\text{dels}(n)$ and $\text{dols}(n)$ respectively. Then the generalization is as follows.

CONJECTURE 4.6. *Let n be a positive integer. Then*

$$\text{dels}(n) \neq \text{dols}(n).$$

If n is even, then switching two rows or columns of a Latin square does not change its sign. By switching rows (or columns) of a Latin square, one can move the ones onto the diagonal. The new diagonalized square hence has the same sign as the original square. For each diagonalized Latin square there are $n!$ general Latin squares that correspond to it: one for each position of the ones. Hence in this case $n! \cdot \text{dels}(n) = \text{els}(n)$ and $n! \cdot \text{dols}(n) = \text{ols}(n)$. So it follows that for even n, Conjectures 2.1 and 4.6 are equivalent.

The choice of putting the ones on the diagonal of the Latin square is, of course, arbitrary. The same equivalences hold when the ones (or other numbers) are fixed to a pattern other than the diagonal.

Note that Theorem 4.5 is independent of the parity of n. When n is odd, Theorem 4.1(b) assures us that there exists no good online algorithm for patterns of size n. It turns out, however, that there does exist such an algorithm for patterns of size $n-1$ if Conjecture 4.6 holds.

Using Zappa's generalization, Aharoni and Kotlar have proven a weaker version of Rota's Basis Conjecture (not the online version) for odd n [**11**]. Namely, rather than n transversals, a set of n bases has at least $n-1$ disjoint independent transversals. Their result bears some similarities with the following theorem for the online version.

THEOREM 4.7. *Let $n > 1$. Consider bases*

$$(B_1, \ldots, B_n) = ((b_{11}, \ldots, b_{1n}), \ldots, (b_{n1}, \ldots, b_{nn}))$$

of an n-dimensional vector space V, such that $b_{kk} = e_n$ for all $k \in [n]$.

If Conjecture 4.6 is true, then there exists an n-step online algorithm as described in Theorem 4.1(a).

Note that any other vector than e_n could have been chosen here as being present in all bases without loss of generality. Moreover, the position of the common vector within the bases does not matter. So for convenience the common vector is assumed to be the k'th basis vector of B_k.

PROOF. If n is even, the theorem follows from Theorem 4.1(a) and the equivalence of Conjectures 2.1 and 4.6.

So suppose n is odd. We will follow the same procedure as in the proof of Theorem 4.1(a), but then for patterns of size $n-1$. The remainder is a pattern of size 1, for which the algorithm is trivial.

So we have $V = W \oplus \langle e_n \rangle$. Define for $k \in [n]$ and $\pi \in \text{Sym}(\{1, 2, \ldots, k-1, k+1, \ldots, n\})$:

$$\Psi_{k,\pi} : ((\bigwedge^{k-2} W)^{k-1} \times (\bigwedge^{k-1} W)^{n-k+1}) \times W^{n-1} \to (\bigwedge^{k-1} W)^k \times (\bigwedge^{k} W)^{n-k},$$

$$(\omega, B) \mapsto (\omega_1 \wedge B_{\pi(1)}, \ldots, \omega_{k-1} \wedge B_{\pi(k-1)}, \omega_k, \omega_{k+1} \wedge B_{\pi(k+1)}, \ldots, \omega_n \wedge B_{\pi(n)}).$$

As before, $\Psi_{k,\pi}$ models adding vectors of a basis B to an n-tuple of 'spaces' ω according to a permutation π. The expression may seem a bit complicated, but the idea behind it is as follows. Since we follow a pattern of size $n-1$, we do not add a vector from W to each of the n columns in each step. This causes an imbalance in the exterior powers in which the ω_j live, illustrated by the following table.

$$\begin{array}{cc} \text{step} & \begin{array}{cccc} \omega_1 & \omega_2 & \ldots & \omega_n \end{array} \\ \begin{array}{c} 1 \\ 2 \\ \vdots \\ n \end{array} & \begin{pmatrix} \cdot & * & * & * \\ * & \cdot & * & * \\ * & * & \ddots & * \\ * & * & * & \cdot \end{pmatrix} \end{array}$$

Wherever there is a dot, no vector is added to the corresponding space in that step. So prior to the k'th step, there are $k-1$ columns that contain $k-2$ vectors, and the remaining columns contain $k-1$ vectors. After the k'th step, there are k columns that contain $k-1$ vectors, and the remaining columns contain k vectors.

Now we will move to the tensor algebra. So instead of $\Psi_{k,\pi}$, we now work with

$$\hat{\Phi}_{k,\pi} : ((W^{\otimes k-2})^{\otimes k-1} \otimes (W^{\otimes k-1})^{\otimes n-k+1}) \times W^{n-1} \to (W^{\otimes k-1})^{\otimes k} \otimes (W^{\otimes k})^{\otimes n-k},$$

$$(t, B) \mapsto (t_1 \otimes B_{\pi(1)}, \ldots, t_{k-1} \otimes B_{\pi(k-1)}, t_k, t_{k+1} \otimes B_{\pi(k+1)}, \ldots, t_n \otimes B_{\pi(n)}).$$

For each k, we can now injectively map $(W^{\otimes k-1})^{\otimes k} \otimes (W^{\otimes k})^{\otimes n-k}$ onto $(V^{\otimes k})^{\otimes n}$, where e_n is added on the diagonal. This injection identifies $\hat{\Phi}_{k,\pi}$ with a map

$$\Phi_{k,\pi} : ((W^{\otimes k-2})^{\otimes k-1} \otimes (W^{\otimes k-1})^{\otimes n-k+1}) \times W^{n-1} \to (V^{\otimes k})^{\otimes n}.$$

Now we will construct the hypersurfaces H_k. In order to do this, we look again at $((V^*)^{\otimes k})^{\otimes n}$, which is spanned by tableaux of the form (4.1). We do not use the same definition of the Θ_k as in the proof of Theorem 4.1(a), however. Instead, define the linear map

$$\Theta_k : ((V^*)^{\otimes k})^{\otimes n} \to ((V^*)^{\otimes k-1})^{\otimes n},$$

$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{k1} & \cdots & a_{kn} \end{bmatrix} \mapsto \tau \cdot \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{k-1,1} & \cdots & a_{k-1,n} \end{bmatrix}.$$

Here, τ is equal to zero if $(a_{k1}, \ldots, a_{kn})$ is not a permutation of $[n]$ with $a_{kk} = n$, and equal to its sign in S_n if it is a permutation of $[n]$ with $a_{kk} = n$.

Now we construct $F_k \in ((V^*)^{\otimes k})^{\otimes n}$ for $k = n, \ldots, 1$ again as follows: $F_n := \det^{\otimes n}$ and $F_{k-1} = \Theta_k F_k$. It remains to show that Conjecture 4.6 implies that all F_k are nonzero. F_n is some linear combination of tableaux of the form (4.1). By definition of F_n, no column in any of the tableaux that have a nonzero coefficient in F_n contains any number from $[n]$ twice. Then for each k, Θ_k annihilates a tableau if its last row is not a permutation σ of $[n]$ such that $\sigma(k) = n$. Tableaux in F_k that do not get annihilated by $\Theta_1 \cdot \Theta_2 \cdots \Theta_k$ have the following structure:

$$\begin{bmatrix} n & a_{12} & \cdots & a_{1k} & \cdots & a_{1n} \\ a_{21} & n & & \vdots & & \vdots \\ \vdots & & \ddots & a_{k-1,k} & & \vdots \\ a_{k1} & \cdots & a_{k,k-1} & n & \cdots & a_{kn} \end{bmatrix},$$

where each row is a permutation of $[n]$. Hence these tableaux correspond to partial Latin squares. Now each $n \times n$ tableau with n's on the diagonal has the product of its column signs as coefficient in F_n. Θ_k adds exactly the sign of the permutation of row k, so that in the end the contribution of such a tableau to F_0 is precisely the sign of the corresponding Latin square. It follows that $F_0 = \mathrm{dels}(n) - \mathrm{dols}(n)$.

If Conjecture 4.6 holds, then $F_0 \neq 0$. Then, by linearity of Θ_k for all k, $F_1, \ldots, F_n \neq 0$ as well.

Now set $H'_k = \{t \in (V^{\otimes k})^{\otimes n} \mid F_k = 0\}$. Project H'_k onto $(\bigwedge^k V)^n$ to obtain H_k. In the same way as in the proof of Theorem 4.1(a), it can be shown that H_k has the following properties. First, it contains all tuples that have a zero entry. Second, whenever there exists a basis B of V (with $B_n = e_n$) such that for all $\pi \in \mathrm{Sym}(1, \ldots, k-1, k+1, \ldots, n)$ it holds that adding B to $\omega \in (\bigwedge^{k-1} V)^n$ gives us a tuple in H_k, then ω must already be in H_{k-1}.

Finally, the algorithm is precisely the algorithm used in the proof of Theorem 4.1(a), using the hypersurfaces $H_0, \ldots, H_n$ constructed in this proof instead.

□

In the theorem we assumed that it was already known which vector all bases had in common. What can we do if we only know that all bases have some vector in common, but are not told which vector that is?

If we would like to prove that an algorithm exists for this case, we could try thinking along the lines of the proof of Theorem 4.7. However, it does not work in this case, since it requires us to fix a decomposition of V into an 1-dimensional subspace and an $(n-1)$-dimensional subspace. The order in which the hypersurfaces $H_1, \ldots, H_n$ are constructed is opposite to the order in which the algorithm uses them. For instance, in the second step, an algorithm chooses π such that $\Psi_{2,\pi}(\omega, B_2) \notin H_2$, where H_2 is based on a decomposition of V into $\langle v \rangle$ and its orthogonal complement in V. But in the next step, it might turn out that it was not v, but some other vector that all bases have in common. When the hypersurfaces are then recalculated with respect to a different decomposition of V and a different pattern, it may turn out that $\Psi_{2,\pi}(\omega, B_2)$, that was just fixed, is not in the new H_2 at all.

In order not to run into this problem, extra conditions should be added. Denote by $H_k^{v,P}$ the hypersurface H_k constructed with respect to some size $n-1$ pattern P and the decomposition of V into $\langle v \rangle$ and its orthogonal complement. Let C_k be the set of vectors that $B_1, \ldots, B_k$ have in common. Then a necessary condition on a successful algorithm is that, in the k'th step, it chooses π such that there exist patterns P_v such that $\Psi_{k,\pi}(\omega, B_k) \notin H_k^{v,P_v}$ for all $v \in C_k$. Surely each H_k^{v,P_v} individually is strictly contained in $(\bigwedge^k V)^n$, as was shown in the proof of Theorem 4.7. But what guarantee is there that there exist patterns P_v such that $\bigcup_{v \in C_k} H_k^{v,P_v} \neq (\bigwedge^k V)^n$?

If, on the other hand, we would like to disprove that an algorithm exists for this case, we need to come up with a different counterexample than the one used in the proof of Theorem 4.1(b): if the algorithm orders the first $n-2$ standard bases in such a way that m is equal to n, we cannot pick the proposed $(n-1)$'st basis, as it would have no vector in common with any of the first $n-2$ bases.

4. Online Rota for matroids

There are several ways to generalize the online version of Rota's Basis Conjecture to matroids.

A natural way is to first fix a matroid M of rank n. Then, an algorithm is given bases of M one by one, and must label each of the n basis elements differently before knowing the remaining bases. In the end, the elements with the same label should form a basis of M.

In applying this generalization to linear matroids M, the results from this chapter still hold. First fix a representation of M. Then, the representations of all basis elements of a basis of M span the same vector space V. Thus these elements can be treated as vectors in V, and the bases of M as bases of V.

A second way to generalize the online version is the following. Rather than fixing a matroid beforehand, the matroid structure is only known to an algorithm for the basis elements it has received. Hence, an algorithm is given, one by one, bases of some priorly unknown matroid M and the independence relations between the elements of the new basis and the elements of the previously labeled bases. Then, the algorithm must label the new basis in such a way that in the end, the elements with the same label form a basis of M.

When discussing vector spaces, it has never been an issue whether the vector space V is known to the algorithm or not. This is due to the fact that, given the first basis, it is already known that the V is precisely the span of the basis vectors of the first basis (and any of the $n-1$ bases to come). For matroids, this property does not hold. Suppose B is a basis of M. Then M can be extended to another matroid M' in such a way that B is still a basis of M'. So, in contrast to vector spaces, bases of a matroid do not uniquely identify a matroid.

Contrary to the first generalization, the results from this chapter do not carry through for this second generalization for linear matroids. A simple example is the following. See Figure 1. Let $n \geq 3$, and let the first $(n-1)$ bases be disjoint bases of the uniform matroid M of rank n on $n(n-1)$ elements. Up to this point, no difference between any labeling can be distinguished by any algorithm. So suppose the algorithm labeled the bases elements in a valid way with the labels $\{1, \ldots, n\}$. Now extend M with an element e in such a way that the only dependent sets of size n are e joined with the set of $n-1$ elements labeled i, for each $i = 1, \ldots, n$. Then any final basis containing e cannot be labeled correctly.

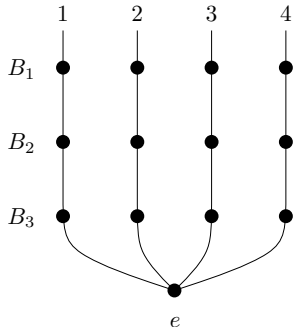


FIGURE 1. An element e that cannot be labeled correctly. The lines correspond to the dependent sets in the matroid.

CHAPTER 5

An online version of Rota's Basis Conjecture for graphic matroids

An interesting question that is closely related to the online version of Rota's Basis Conjecture is the following. Is it possible to find a combinatorial online algorithm for graphic matroids?

Consider a set of $n+1$ vertices. Define a matroid on the edges between the vertices. A set of edges E is defined to be independent if and only if the graph $G = ([n+1], E)$ is a forest. In other words, E is independent if and only if G is cycle-free. This matroid is called the *graphic matroid* of the complete graph K_{n+1}. The bases of these matroids are precisely the spanning trees of the $n+1$ vertices. Hence the rank of the matroid is n.

The graphic matroid is realizable by vectors in $\mathbb{R}^{n+1}$ as follows. An edge between vertices i and j corresponds to the vector $e_i - e_j$, where the choice of the sign is unimportant. Indeed, the vectors corresponding to a set of edges are dependent if and only if the corresponding graph contains a cycle.

Thus, fixing this realization, all previous results for vector spaces apply to this matroid.

THEOREM 5.1. *(Online Rota for graphic matroids) Let an even natural number n be given. Let C be a set of n colours. If Conjecture 2.1 holds for n, then there exists an n-step online algorithm as follows. In each step, a tree on $n+1$ vertices is given. The algorithm then colours each edge of the tree with a different colour from C. After the last step, the edges of each colour form a tree.*

PROOF. Trees correspond to bases, not only of a matroid, but also of an n-dimensional linear space. Moreover, the edges of one colour form a transversal of n trees (bases). Hence this theorem is a weaker version of Theorem 4.1(a), where the weakness lies in the limitation of the bases to sets of vectors corresponding to edges. □

An illustration of how such an algorithm would work is given in Figure 1. This algorithm does not assign the trees in a correct way. In fact, it ignores many of the conditions that are posed in section 1.1 of this chapter. I challenge the reader to find out:

- in which steps the algorithm makes a mistake;
- with which tree(s) each mistake can be punished;
- how the algorithm should have assigned the trees instead.

After discussing the relevant conditions, the mistakes will be analyzed in depth in section 1.1.

In the proof of Theorem 4.1(a), the general algorithm was constructed. However, the hypersurfaces $\{H_k \mid 0 \leq k \leq n\}$ are only shown to have a nonempty

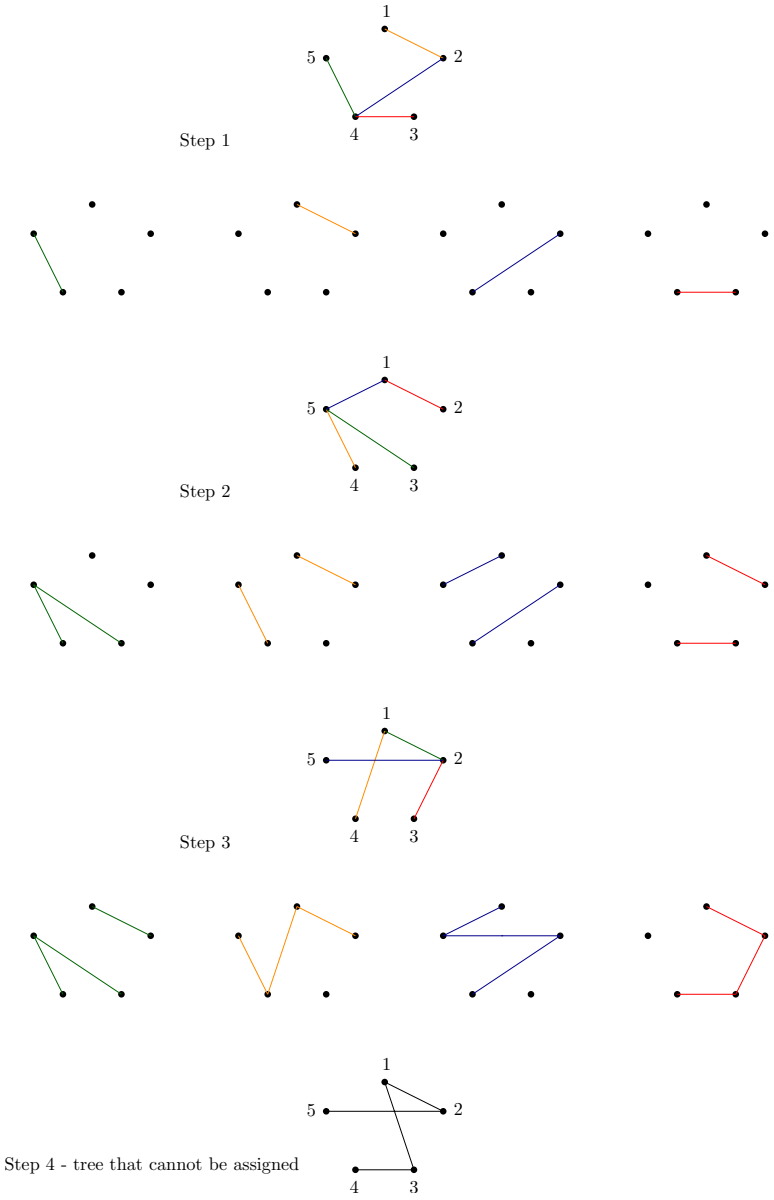


FIGURE 1. An example of the choices of an (incorrect) algorithm, as described in Theorem 5.1
For each step, the tree is first given. The coloured edges in this tree are not part of the input, but illustrate the choice of the algorithm. Then, the forests of each colour after colouring this tree are given.

complement, so that the algorithm can avoid picking tuples inside H_k. In order to construct a combinatorial algorithm, it is necessary describe H_k precisely.

It should be noted that the fact that the online version of Rota's Basis Conjecture is false for odd n does not imply that the version for graphic matroids is

false if n were odd. The counterexample given in the proof of Theorem 4.1(b) relies on the existence of primitive m'th roots of unity for certain $m \leq n$. This is not the case here, as only edges can be used, which correspond to a difference of two standard basis vectors.

Whether the online version of Rota's Basis Conjecture for graphic matroids is true for odd n or not remains to be investigated. For $n = 3$, the counterexample in the proof of Theorem 4.1(b) has been shown to be unique (up to symmetries). So certainly, the theorem is true for $n = 3$. A topic for further research is the question whether there exist other counterexamples for greater odd n.

1. Combinatorial algorithm

Due to Theorem 5.1, there exists an online algorithm, at least for even n provided that the Alon-Tarsi Conjecture holds, that colours trees appropriately. But how can such an algorithm be expressed in a combinatorial way?

For $n = 4$, it is already difficult to find the algorithm. In order to describe the conditions that an algorithm should adhere to in its choices, it is convenient to consider the n constructed forests as partitions of the vertex set $[n+1]$ rather than a graph. The parts of such a partition then correspond to the connected components of the corresponding forest. Modeling the forests in this way discards the irrelevant information of how vertices are connected within each part.

First, let us introduce some terminology.

A *refinement* of a partition $P = \{p_1, \ldots, p_s\}$ of a set X is a partition Q of X such that for each part q of Q, there exists an $i : 1 \leq i \leq s$ such that $q \subseteq p_s$. For instance: $\{\{1\}, \{2\}, \{3, 4\}, \{5\}\}$ is a refinement of $\{\{1, 2\}, \{3, 4, 5\}\}$.

A *coarsening* of a partition Q of X is a partition P of X such that Q is a refinement of P.

The *Coarsest Common Refinement* (CCR) of partitions $P_1, \ldots, P_r$ is a partition Q that is a refinement of all partitions $P_1, \ldots, P_r$, such that no other refinement of $P_1, \ldots, P_r$ is a coarsening of Q. The CCR always exists, since the the partition into singletons is a refinement of any partition. The CCR is also unique: suppose that $P \neq Q$ are CCR's. Since neither is a coarsening of the other, we can assume that there exist a part p of P and a part q of Q such that $q \setminus p \neq \emptyset$ and $p \cap q \neq \emptyset$. Since p and q overlap, $P_1, \ldots, P_r$ all have a part that contains $p \cup q$. Hence if in P all parts containing an element in q are joined with p, we obtain a refinement of $P_1, \ldots, P_r$ that is a strict coarsening of P, contradicting the assumption that P is a CCR.

Similarly, the *Finest Common Coarsening* (FCC) of partitions $Q_1, \ldots, Q_r$ is a partition P that is a coarsening of all partitions $Q_1, \ldots, Q_r$, such that no other coarsening of $Q_1, \ldots, Q_r$ is a refinement of P. Existence is shown by the fact that the partition into one part is a coarsening of any partition. In order to see that the FCC is unique, suppose that $P \neq Q$ are FCC's. Neither is a refinement of the other, so we can assume that there is some part p of P that contains elements from more than one part of Q. Let q be one of these parts of Q. Then splitting p into $p \cap q$ and $p \setminus q$ gives a coarsening of $Q_1, \ldots, Q_r$ that is a strict refinement of P, contradicting the assumption that P is a FCC.

Some more (compact but abusive) notation: CCR_i^k will denote the CCR of i forests in step k. Similarly, FCC_i^k will denote the FCC of i forests in step k. For

instance, the condition $|\mathrm{CCR}_i^k| < j$ means that the CCR of any subset of i forests in step k must have fewer than j parts.

1.1. Necessary conditions. Now we will describe conditions that are necessary for the success of an algorithm. Many conditions can be formulated in terms of the CCR and the FFC.

The first class of conditions ensures that subsets of forests do not have too many edges in common that cannot be added to any of them.

$$|\mathrm{CCR}_i^k| \begin{cases} = n+1, & i > k; \\ > i, & i \leq k. \end{cases} \tag{5.1}$$

A more mathematical way of formulating the same conditions is the following. Let P_l be the partition belonging to forest l. In the k'th step, for all $L \subseteq [n]$ with $|L| = i$, it must hold that

$$|\mathrm{CCR}(P_l \mid l \in L)| \begin{cases} = n+1, & i > k; \\ > i, & i \leq k. \end{cases}$$

The second class of conditions makes sure that subsets of forests do not have too many cuts in common in which edges need to be added.

$$|\mathrm{FCC}_i^{n+1-i}| = 1. \tag{5.2}$$

More mathematically, but less compactly: for all $L \subseteq [n]$ with $|L| = i$, it must hold in step $n+1-i$ that

$$|\mathrm{FCC}(P_l \mid l \in L)| = 1.$$

Moreover, there are conditions that concern subsets of vertices. Let $P_1, \ldots, P_n$ be the partition of the n forests. For any subset S of the vertices, consider the restriction operator r_S that removes vertices from a partition that are not in S. For $l = 1, \ldots, n$, denote $P_l^S = r_S(P_l)$. Now for each subset of vertices S such that $|S| = j$, the following condition should hold in step k:

$$\sum_{l=1}^{n} (|P_l^S| - 1) \geq (j-1)(n-k) \tag{5.3}$$

THEOREM 5.2. *Condition (5.1) is necessary.*

PROOF. First, suppose $i > k$ and $|\mathrm{CCR}_i^k| < n+1$. Then there is a part in the CCR of some i partitions that contains more than one vertex. Suppose the remaining $n-k$ trees contain an edge between two vertices in this part. In each step, this edge will need to be assigned to one of the other $n-i < n-k$ forests. Hence there are too few forests left to garrison this edge. So if $i > k$, then $|\mathrm{CCR}_i^k| = n+1$ is required.

Second, suppose $|\mathrm{CCR}_i^k| \leq i$. Assume the next tree contains $n-i+1$ edges within the parts of the CCR of some i partitions, which is possible due to the assumption that $|\mathrm{CCR}_i^k| \leq i$. Then these $n-i+1$ edges must be assigned to the remaining $n-i$ trees, which cannot be done. So it follows that $|\mathrm{CCR}_i^k| > i$ is required for all k and i, and in particular for $i \leq k$. □

THEOREM 5.3. *Condition (5.2) is necessary.*

PROOF. Suppose that for some i, $|\mathrm{FCC}_i^{n+1-i}| > 1$. Then in some subset of i forests, there is a cut between one part of the FFC and the rest that contains no edges. In order for these forests to be completed to a tree after the final step, an edge between these two sets of vertices must be added to each of them. However, if each of the remaining $i-1$ trees contains only one edge in the cut, then not all i forests can be completed to trees. Hence $|\mathrm{FCC}_i^{n+1-i}| = 1$ is required. □

THEOREM 5.4. *Condition (5.3) is necessary.*

PROOF. $|P_l^S| - 1$ is equal to the number of edges that can be added to P_l^S without creating a cycle, since each addition of an edge unites two parts.

A tree on all $n+1$ vertices can contain up to $j-1$ edges within any subset of j vertices. In step k, there will be $n-k$ trees remaining. Thus in order to be able to assign $n-k$ trees that all have $j-1$ edges within a given subset of j vertices, it is necessary to be able to add at least $(j-1)(n-k)$ edges within this subset across all forests. □

Conditions (5.1), (5.2) and (5.3) together are, however, not sufficient. Consider the example in Figure 2 for $n = 4$ and $k = 2$. With some work one can verify that all three conditions indeed hold. However, if the last two trees both contain the edges $(2, 3)$ and $(4, 5)$, then the forests cannot all be completed to trees.

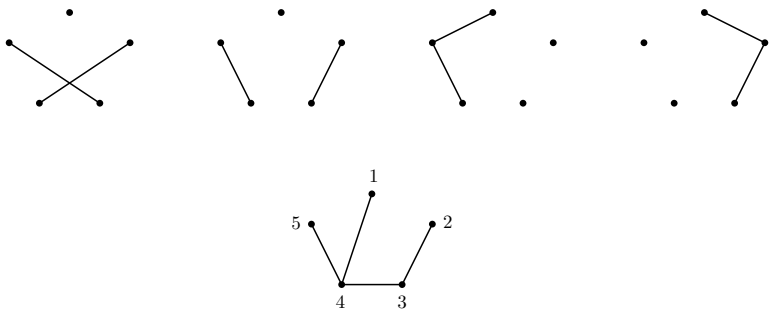


FIGURE 2. A bad example satisfying conditions (5.1), (5.2) and (5.3). $n = 4$; $k = 2$. The upper four graphs are the forests; the lower graph is a tree that cannot be assigned in a way that does not violate the conditions.

Hence condition (5.3) needs to be sharpened. Let $Q = \{q_1, \ldots, q_r\}$ be a partition of the $n+1$ vertices. Suppose that for some $s \le r$, $q_1, \ldots, q_s$ are all of the non-singleton parts of Q. Denote by $\mathcal{P}^s$ the set of all partitions of $[s]$. Then:

$$\sum_{l=1}^{n} \min_{T \in \mathcal{P}^s} \left(\sum_{i=1}^{|T|} (|P_l^{\bigcup_{t \in t_i} q_t}| - 1) \right) \ge (n - r + 1)(n - k) \tag{5.4}$$

THEOREM 5.5. *Condition (5.4) is necessary.*

PROOF. In the first part of the proof, it is shown that

$$C_l := \min_{T \in \mathcal{P}^s} \left(\sum_{i=1}^{|T|} (|P_l^{\bigcup_{t \in t_i} q_t}| - 1) \right)$$

is equal to the number of edges (which we call N_l) that lie within the parts of Q that can be added to forest l without creating a cycle.

Each T partitions the set $\{q_1, \ldots, q_s\}$. As noted before, $|P_l^{\bigcup_{t\in t_i} q_t}|-1$ is precisely the number of edges that can be added within $\bigcup_{t\in t_i} q_t$. Since the partition of the vertices induced by T is a coarsening of $\{q_1, \ldots, q_s\}$,

$$\sum_{i=1}^{|T|}(|P_l^{\bigcup_{t\in t_i} q_t}| - 1)$$

is an upper bound for N_l. Since this is the case for each T, it follows that $N_l \leq C_l$.

Now suppose that $N_l < C_l$. Let T be a minimizing partition in C_l. Then there is a part t_i of T such that the number of edges that can be added within the parts of Q indicated by t_i is strictly less than

$$|P_l^{\bigcup_{t\in t_i} q_t}| - 1.$$

But since this is precisely the number of edges that can be added within $\bigcup_{t\in t_i} q_t$, there is some refinement $\{t'_1, \ldots, t'_m\}$ of $\{t_i\}$ with $m > 1$ such that

$$\sum_{k=1}^{m}(|P_l^{\bigcup_{t\in t'_k} q_t}| - 1) < |P_l^{\bigcup_{t\in t_i} q_t}| - 1.$$

However, this contradicts the minimality of T, as the partition

$$\{t_1, \ldots, t_{i-1}, t'_1, \ldots, t'_m, t_{i+1}, \ldots, t_{|T|}\}$$

would then give a smaller number. Hence it also holds that $N_l \geq C_l$.

Q consists of r parts, so at most $n - r + 1$ independent edges can be chosen within the parts of Q. With $n - k$ trees to go, it is therefore necessary that at least $(n - r + 1)(n - k)$ edges within Q can be added to all l forests together. Hence it is necessary that

$$\sum_{l=1}^{n} N_l \geq (n - r + 1)(n - k).$$

□

One can try to slightly relax condition (5.4) by minimizing over a smaller set than $\mathcal{P}^s$ for each forest. This is, however, to no avail.

$$\sum_{l=1}^{n} \min\left(\sum_{i=1}^{s}(|P_l^{q_i}| - 1), |P_l^{\bigcup_{t=1}^{s} q_t}| - 1\right) \geq (n - r + 1)(n - k) \tag{5.5}$$

for each partition Q could be falsely thought to suffice. One could think that wherever

$$\min\left(\sum_{i=1}^{s}(|P_l^{q_i}| - 1), |P_l^{\bigcup_{t=1}^{s} q_t}| - 1\right)$$

does not accurately represent the number of edges within Q that can be added to forest l, there is another partition Q' for which condition (5.5) is violated. This Q' would be a coarsening of Q such that its only non-singleton parts $\{q'_1, \ldots, q'_{s'}\}$ are precisely those parts for which in forest l,

$$\min\left(\sum_{i=1}^{s'}(|P_l^{q'_i}| - 1), |P_l^{\bigcup_{t=1}^{s'} q'_t}| - 1\right) = \min_{T\in\mathcal{P}^s}\left(\sum_{i=1}^{|T|}(|P_l^{\bigcup_{t\in t_i} q_t}| - 1)\right).$$

The flaw of this reasoning is pointed out by the example in Figure 3. For forest l, the summand in condition (5.5) indeed indicates the correct number of edges that can be added within partition Q' (and Q). However, for other forests, the partition Q' may not be the critical partition.

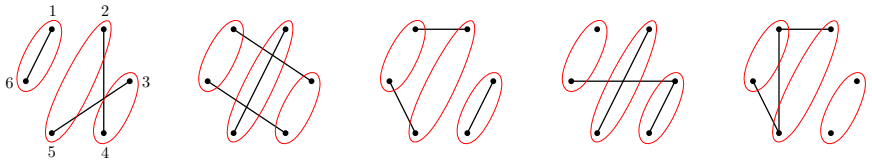


FIGURE 3. An example where condition (5.4) is violated for the partition $\{\{1,6\},\{2,5\},\{3,4\}\}$ indicated by the red ovals, but where conditions (5.1), (5.2) and (5.5) are not violated.

It should be noted that condition (5.5) is equivalent to condition (5.4) when $n \leq 4$. This is because the number of non-singleton parts in a partition of $[n+1]$ is at most $\lfloor \frac{n+1}{2} \rfloor \leq 2$.

Now we return to discuss the mistakes of the faulty algorithm in Figure 1. The reader may want to compare these results to their own findings.

Indeed, the algorithm made a mistake in step 3. As it will turn out in section 1.2, all violated conditions in the $(n-1)$'st step can be written as a violation of condition (5.1).

The first mistake, for which the algorithm is punished by the 4'th tree, is that it allowed the number of parts in the CCR of all four forests to be 4, rather than the required 5. Indeed, $\{1,2\}$ is a part of this CCR, and hence the edge $(1,2)$ cannot be added to any of the forests.

Another violation was made in step 3, which can be easily spotted by the isolated vertex 3 in both the second and the third forest. The CCR of forests 2 and 3 has size 2, rather than the required 3. Or equivalently, the FCC of forests 2 and 3 has size 2. Equivalently as well, condition (5.4) is violated for the partition $\{\{3\},\{1,2,4,5\}\}$. This mistake is punished by any 4'th tree that has only one edge incident to vertex 3 (and therefore three edges within the part $\{1,2,4,5\}$).

Instead, in step 3 the algorithm should have chosen, for instance, the assignment shown in Figure 4. It can be verified that the forests satisfy all conditions after this assignment.

In fact, the algorithm was lucky to get a second chance with the given third tree, for it had already made a few mistakes in step 2, neither of which the tree given in step 3 was suited to punish.

A keen eye may have spotted that the FCC of forests 1, 2 and 4 has size greater than one. In all three of these forests there is a cut between $\{1,2\}$ and $\{3,4,5\}$. Thus if the final two trees both contained only one edge in this cut, this mistake would be punished.

An even keener eye may have spotted that this was not the only mistake in step 2. Condition (5.4) is violated as well for the partition $\{\{3\},\{1,2\},\{4,5\}\}$. The number of edges that can be added within this partition is 3, while the condition

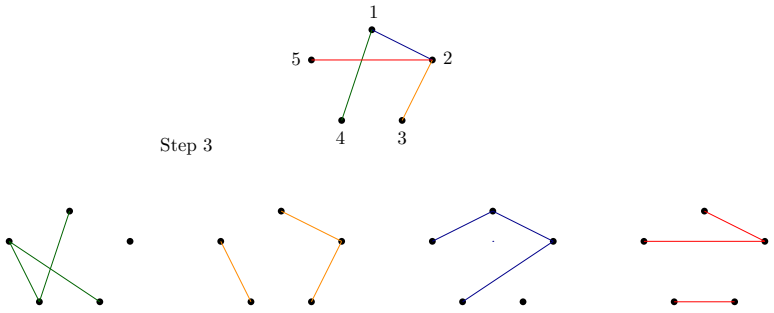


FIGURE 4. A correction in step 3 of the algorithm in Figure 1.

requires 4. Thus if the final two trees both contained the edges $(1, 2)$ and $(4, 5)$, this mistake would be punished.

A correct way to assign the tree is given in Figure 5.

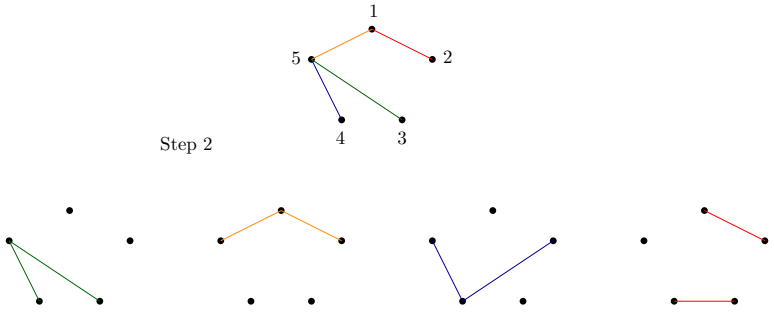


FIGURE 5. A correction in step 2 of the algorithm in Figure 1.

The algorithm did not make a mistake in step 1, but this is hardly an achievement. All assignments are the same up to a permutation of the forests.

1.2. Sufficient conditions. So it turns out that conditions (5.1), (5.2) and (5.4) are necessary conditions, in the sense that if these conditions are not satisfied, there exist trees in the following steps that force any algorithm to make a mistake. But are they also sufficient? In other words, if the conditions are all satisfied in some step k, is it then always possible to assign any tree in such a way that the conditions are also satisfied in step $k + 1$, hence giving rise to a correct algorithm?

Let us first look at the final step, which is the easiest to analyze. It turns out that in this step, conditions (5.2) and (5.4) are special cases of condition (5.1), since the latter condition is not only necessary for a successful algorithm, but also sufficient.

THEOREM 5.6. *Let $k = n - 1$. Then condition (5.1) alone is sufficient.*

PROOF. For $k = n - 1$, condition (5.1) simplifies to

$$|\mathrm{CCR}_i^{n-1}| \geq i + 1.$$

Since $k = n - 1$ and only one tree remains, the partition of each forest consists of two parts. In the CCR of some partitions, every split between two parts is also

present in at least one of the partitions the CCR is taken of, for otherwise there is a coarser common refinement with these two parts joined. Since the partition of each forest only contains one split, the CCR of i such partitions contains at most i splits. Hence we find

$$|\mathrm{CCR}_i^{n-1}| \leq i+1.$$

So in fact, condition (5.1) implies

$$|\mathrm{CCR}_i^{n-1}| = i+1.$$

Recall that the forests live in a graphical matroid, and that each partition corresponding to a forest can be identified with a subspace of $\mathbb{R}^{n+1}$ spanned by $e_i - e_j$ for all edges (i,j) within its parts. Each part of the CCR of i partitions indicates a subset of vertices that is present in some part of all i partitions. So let i forests be given, and let $P_1, \ldots, P_i$ be their corresponding partitions. Let furthermore $V_1, \ldots, V_i$ be the corresponding subspaces of $\mathbb{R}^{n+1}$. Then $\mathrm{CCR}(P_1, \ldots, P_i)$ corresponds to $V_1 \cap \ldots \cap V_i$. Now the dimension of $V_1 \cap \ldots \cap V_i$ is equal to the number of independent edges within the parts of $\mathrm{CCR}(P_1, \ldots, P_i)$. Hence

$$\dim(V_1 \cap \ldots \cap V_i) = n+1-|\mathrm{CCR}(P_1, \ldots, P_i)| = n-i.$$

By Theorem 4.2, it follows that each final tree can be assigned. □

Now we have all the ingredients needed to give and prove correctness of a combinatorial algorithm for $n = 3$ and $n = 4$.

THEOREM 5.7. *Let $n = 3$. Then any algorithm as in Theorem 5.1 that makes sure that in each step condition (5.1) is satisfied is correct. Moreover, such an algorithm exists.*

PROOF. The counterexample for $n = 3$ in the proof of Theorem 4.1(b) has been shown to be unique up to symmetries. Hence, since the bases given there cannot be represented as trees, there exists a valid algorithm.

In the first step, no mistakes can be made due to symmetry. In the second step, existence of a correct algorithm implies that there is always a way to satisfy the necessary condition (5.1). Due to Theorem 5.6, the final step can then also be executed successfully. □

THEOREM 5.8. *Let $n = 4$. Then any algorithm as in Theorem 5.1 that makes sure that in each step conditions (5.1), (5.2) and (5.4) are satisfied is correct. Moreover, such an algorithm exists.*

PROOF. First note that due to Theorem 5.1 and the fact that Conjecture 2.1 is verified for $n = 4$, there exists a correct algorithm. Since all possible assignments of the first tree are equivalent, the first real decision for any algorithm is how to assign the second tree. As conditions (5.1), (5.2) and (5.4) are necessary for success, existence of a correct algorithm implies that there is always a way to assign the second tree in such a way that these conditions are satisfied.

By Theorem 5.6, it now suffices to show that for each 4-tuple of forests on 5 vertices with 2 edges that satisfy conditions (5.1), (5.2) and (5.4), any third tree can be assigned in such a way that condition (5.1) is satisfied.

This can be done by simply checking all cases. First note that, due to symmetry, there are only two different types of forests on 5 vertices with 2 edges, namely those with two connected edges and those with two non-connected edges. So, after

choosing any representative from each of the two types, each 4-tuple of forests can be altered, by permuting the vertices for all forests, into a 4-tuple of forests with either of the two chosen representatives as the first forest. When it turns out that all trees can be assigned in some way, the permutation of vertices can be reversed to obtain the same result for the other 4-tuples with different first forests. This is due to the fact that the set of all trees is invariant under permutations of the vertices. Hence the number of 4-tuples to be checked after applying this symmetry is $2 \cdot 45^3 = 182250$. Then, for each tuple, it must be checked if each of the 125 trees can be assigned.

The case checking has been done by means of computer calculations. For the main part of the Java source code used, see Appendix A. Globally, the program works as follows. For each tuple, it first checks whether it satisfies conditions (5.1), (5.2) and (5.4). If this is the case, it checks for all trees whether there is an assignment of the edges of the tree to the forests such that no cycles are created and the conditions are satisfied. If there is no such assignment for one of the trees, it results in a failure. The program counts the number of failures, which turns out to be zero.

□

For any n, conditions (5.1), (5.2) and (5.4) have been shown to be necessary for success of an algorithm. Whether any algorithm that makes sure these conditions are satisfied is always successful remains, for now, unproven. For $n > 4$, I checked the algorithm `greedyFindPerm(...)` as given in Appendix A by means of random sampling. Specifically, I generated random trees that the algorithm needs to assign. The algorithm failed in none of these cases. However, the total number of possible cases grows more than exponentially with n. Due to Cayley's formula, there are $(n+1)^{n-1}$ trees on $n+1$ vertices. Hence there are $((n+1)^{n-1})^n$ distinct n-tuples of trees, of which only a relatively small fraction can be divided out due to symmetries. The current program is unable to check all of them for any $n > 4$ within a reasonable time frame, which is not surprising, since the number of 5-tuples of trees for $n = 5$ is already approximately 2^{52}.

CONJECTURE 5.9. *Any algorithm as in Theorem 5.1 that makes sure that in each step conditions (5.1), (5.2) and (5.4) are satisfied is correct.*

Of course the 'real' conditions for the success of an algorithm are to make sure that in step k the tuples of forest stay outside H_k as in Theorem 4.1(a). The question is whether conditions (5.1), (5.2) and (5.4) accomplish this. This question can also be posed in another way. For some n and k, is there a necessary condition on the tuples of n forest that each contain k edges, that is not implied by conditions (5.1), (5.2) and (5.4)? In other words, are conditions (5.1), (5.2) and (5.4) a complete set of conditions for success of an algorithm?

The CCR condition (5.1) says something about conditions for any dimension n, step k and subset size i. So this condition appears to be complete in the sense that there seems to be no straightforward way to generalize it.

The FCC condition (5.2) is a bit more limited in its applicability: it is only demanded for the very specific case that $i = n + 1 - k$. Of course, this condition also holds for subset sizes greater than i, but that is implied by the condition for subset size i. Could there be FCC-type conditions on subsets of size smaller than $n + 1 - k$, where the size of the FCC is demanded to be at most $f > 1$? Suppose

the FCC of some subset of i forests in step k has size $f > 1$. Then these forests i all miss $f - 1$ edges within the cut given by the FCC. However, the smallest number of edges that any of the remaining $n - k$ trees can have within this cut is $f - 1$ as well. So it would be necessary that $(n-k)(f-1) \geq i(f-1)$. Hence it can only be violated if $i \geq n + 1 - k$. But for these i, the FCC condition is already as sharp as possible.

The partition condition (5.4) says something about the edges within some partition across all trees. Could it be generalized to say something about the edges within some partition across some subset of size i of trees? This seems unlikely, because the partition condition counts the number of edges that can be added within a partition, and bounds that from below. A similar necessary condition on a subset of forests would thus always be weaker than the condition on all forests, since for the other $n - i$ forests, the most favourable case needs to be assumed. The only loophole for the partition condition might be that it counts the number of edges that can be added to each forest within a partition *separately*. It is not unimaginable that there exists a case where the partition condition is tightly satisfied for some partition Q, but where it is not possible to actually assign all possible $(n-k)$ remaining trees that contain the maximum number of edges within Q. Yet, it is similarly imaginable that such a case does not exist at all, since it could be ruled out by the other conditions.

All in all, Conjecture 5.9 still stands, ready to be proven or disproven. I look forward to new insights on this matter.

Bibliography

[1] Rosa Huang and Gian-Carlo Rota. On the relations of various conjectures on latin squares and straightening coefficients. *Discrete Math.*, 128:225–236, 1994.

[2] Noga Alon and Michael Tarsi. Colorings and orientations of graphs. *Combinatorica*, 12(2):125–134, 1992.

[3] Shmuel Onn. A colorful determinantal identity, a conjecture of Rota, and Latin squares. *Am. Math. Mon.*, 104(2):156–159, 1997.

[4] Jim Geelen and Kerri Webb. On Rota's basis conjecture. *SIAM J. Discrete Math.*, 21(3):802–804, 2007.

[5] Jim Geelen and Peter J. Humphries. Rota's basis conjecture for paving matroids. *SIAM J. Discrete Math.*, 20(4):1042–1045, 2006.

[6] Wendy Chan. An exchange property of matroid. *Discrete Math.*, 146:299–302, 1995.

[7] Paolo Zappa. The Cayley determinant of the determinant tensor and the Alon-Tarsi conjecture. *Adv. in Appl. Math.*, 19(1):31–44, 1997.

[8] Jeannette C.M Janssen. On even and odd latin squares. *J. of Combin. Theory, Ser. A*, 69(1):173–181, 1995.

[9] Sloane, N. J. A. Sequence A114630 *The On-Line Encyclopedia of Integer Sequences*, http://oeis.org/A114630, 2005

[10] Guus P. Bollen and Jan Draisma. An online version of Rota's basis conjecture. `http://arxiv.org/abs/1312.5953`, 2013

[11] Ron Aharoni and Daniel Kotlar. A weak version of rota's basis conjecture for odd dimensions. *SIAM J. Discrete Math.*, 2011. To appear; preprint available at `http://arxiv.org/abs/1110.1830`.

[12] Arthur A. Drisko. On the number of even and odd Latin squares of order $p + 1$. *Adv. Math.*, 128(1):20-35, 1997.

[13] David G. Glynn. The conjectures of alon-tarsi and rota in dimension prime minus one. *SIAM J. Discrete Math.*, 24(2):394-399, 2010.

[14] Timothy Y. Chow. Reduction of Rota's basis conjecture to a problem on three bases. *SIAM J. Discrete Math.*, 23(1): 369–371, 2009.

[15] URL: `http://math.uwaterloo.ca/combinatorics-and-optimization/news/geelen-gerards-and-whittle-announce-proof-rotas-conjecture`, 2013

APPENDIX A

Graphical Matroid Java program

The first important method in the program checks whether all 4-tuples of forests on 5 vertices with 2 edges that satisfy the CCR, FCC and partition conditions can correctly assign each tree.

```
void checkAll2(){
Tree[] all2Forests = all2Forests5();
    //all 45 forests on 2 edges on 5 vertices
Tree[] allTrees = allTrees5();
    //all 125 trees on 5 vertices
Tree[] difFor = {all2Forests[0], all2Forests[30]};

//difFor contains one forest with two incident edges
//and one forest with two non-incident edges.

int nFailures = 0;

//For each tuple of forests on 2 edges that
//satisfies the CCR, FCC and partition conditions,
//check whether each tree can be assigned.

for(int f0=0; f0<2; f0++){
for(int f1=0; f1<45; f1++){
for(int f2=0; f2<45; f2++){
for(int f3=0; f3<45; f3++){
Tree[] conFrsts = new Tree[4];

//Build the tuple of constructed forests.
for(int i=0; i < 4; i++){
    if(i==0){
        conFrsts[i] = difFor[f0];
    }else if(i==1){
        conFrsts[i] = all2Forests[f1];
    }else if(i==2){
        conFrsts[i] = all2Forests[f2];
    }else{
        conFrsts[i] = all2Forests[f3];
    }
```

```
}
//Check whether the tuple satisfies the
//conditions in step 2.
boolean conditionsSatisfied =
            checkConditions(conFrsts,1,5);

//If the tuple satisfies the conditions, check
//whether each tree can be assigned.
//If not, increase the number of failures.

for(int t=0; t<125; t++){
    if(conditionsSatisfied){
        int[] result = greedyFindPerm(
            allTrees[t],conFrsts,2,5);
        if(result[0] == -1){
            //No valid assignment was found.
            nFailures++;
            break;
        }
    }
}

} } } }
System.out.println("Number of failures: " + nFailures);
}
```

In the above method, the following method is used to find an assignment of a tree to the constructed forests.

```
int[] greedyFindPerm(Tree newTree, Tree[] conFrsts,
    int step, int v){
//v = n + 1 and k = step + 1.
for(int i = 0; i < permutations.length; i++){
    boolean validPerm = true;
    //Check whether adding edges according to this
    //permutation creates a cycle in any of the forests:
    for(int k=0; k<v-1; k++){
        validPerm = validPerm &&
            conFrsts[k].addable[newTree.edges[
                permutations[i][k]][0]][
                newTree.edges[permutations[i][k]][1]];
    }

    if(validPerm){
        Tree[] conFrstCopies = new Tree[v-1];
```

```
        //Make copies of the constructed forests with
        //the edges added according to the checked
        //permutation in order to check the conditions.

        for(int k=0; k<v-1; k++){
            conFrstCopies[k] = new Tree(conFrsts[k]);
            conFrstCopies[k].addEdge(
                newTree.edges[permutations[i][k]]);
        }

        if(checkConditions(conFrsts, step, v)){
            //The conditions are met; return the permutation.
            return permutations[i];
        }
    }
}

//There is no valid permutation.
int[] noSolution = new int[v-1];
noSolution[0] = -1;
return noSolution;
}
```

In both previously listed methods, the following method is used to check whether the conditions on a tuple are satisfied.

```
boolean checkConditions(Tree[] conFrsts, int step, int v){

for(int l=2; l<=v-1 &&
    step + 1 < v - 1 && step > 0; l++){

    int[][] comb = combinations[l];
    //comb contains all subsets of size l
    //of [n] (= [v-1]). For each subset,
    //the CCR and FCC conditions are checked.

    for(int c=0; c<comb.length; c++){
        int[][] Fpart = new int[l][v];
        for(int k = 0; k < l; k++){
            Fpart[k] = conFrsts[
                comb[c][k]].partitions;
        }
        //Fpart contains the partitions
        //corresponding to each forest
        //in comb.
```

```
        if(step + 1 == v - l){
            boolean FCCsize1 =
                FCCsize1(Fpart,v);
            //FCCsize1(...) computes the
            //FCC of some partitions and
            //returns true only if it has
            //one part.
            if (!FCCsize1){
                //If the conditions on the FCC
                //are not met, return false.
                return false;
            }
        }

        int CCRsize = CCRsize(Fpart,v);
        //CCRsize(...) computes the CCR
        //of some partitions and returns
        //the number of parts it has.

        if( (l > step + 1 && CCRsize < v) ||
            (l <= step + 1 && CCRsize <= l) ){
                //If the conditions on the CCR
                //are not met, return false.
                return false;
        }
    }
}
//Finally, check the partition conditions.
//allPartitions[l] contains all partitions of [v] = [n+1]
//of size l.
for(int l=3; l<v-1; l++){
    //For l<3 and l>=v-1, this condition is equivalent
    //to either the CCR or the FCC condition.

    for(int p=0; p<allPartitions[l].length; p++){
        int[] checkedPartition = allPartitions[l][p];

        int canBeAdded = addableToPartition(
            checkedPartition, conFrsts, v);
    //addableToPartition(...) counts the number of
    //edges that can be added to each forest within
    //a certain  partition.

        if(canBeAdded < (v-1 - (step+1))*((v-l)) ){
            //If the partition condition is not met,
```

```
            //return false.
            return false;
        }
    }
}
//If none of the conditions is violated, return true.
return true;
}
```

`greedyFindPerm(...)` is also more generally applicable. It can be used for other n. The following method was used to generate random trees, and construct forests from scratch by in each step letting `greedyFindPerm(...)` choose the assignment for the tree.

```
void greedySimulate(int iterations, int vertices){
int nFailures = 0;
for(int iter = 0; iter < iterations; iter++){

    Tree[] conFrsts = new Tree[vertices-1];

    //create n = v-1 empty forests:
    for(int i=0; i < vertices - 1; i++){
        conFrsts[i] = new Tree(vertices,true);
    }

    for(int i=0; i < vertices - 1; i++){
        //Generate a random tree
        Tree randomTree = new Tree(vertices);

        //Find an assignment of the edges of randomTree
        //such that all conditions are met
        int[] result = greedyFindPerm(randomTree,
                                conFrsts,i,vertices);

        //Add the edges and move on to the next step.
        //If no assignment was found, it is a failure.
        if(result[0] != -1){
            for(int k=0; k<vertices-1; k++){
                conFrsts[k].addEdge(
                    randomTree.edges[result[k]]);
            }
        }
        else{
            nFailures++;
            System.out.println("Failure in step " + (i+1));
            break;
```

```
            }
        }
        System.out.println("Simulation " + iter + " complete.");
    }
    System.out.println("Number of failures: " + nFailures);

}
```

The larger n is, however, the more time it takes to check all of the conditions. The number of subsets of the n forests grows quickly; the number of partitions of $[n+1]$ even more so. Yet, in all cases attempted by random sampling using the above method, the algorithm never failed to complete the forests to trees.