Elem. Math. 71 (2016) 21 – 38
0013-6018/16/010021-18
DOI 10.4171/EM/296

Elemente der Mathematik

Counting the number of round-robin tournament schedules

Alyssa Cherry, Drake Olejniczak and Qinghong Zhang

Alyssa D. Cherry obtained her Bachelor of Science in Mathematics from Northern Michigan University in 2013. She completed her master's degree at Northern Illinois University and started a doctoral program at the University of Missouri.

Drake P. Olejniczak obtained his Bachelor of Science in Mathematics from Northern Michigan University in 2014. He is currently in the Ph.D program at Western Michigan University and working on Graph Theory.

Qinghong Zhang obtained his PhD from the University of Iowa in 2002. He is currently a full professor at Northern Michigan University, Marquette, Michigan, USA. His interests are in optimization. Specifically, he is interested in semi-infinite programming, semidefinite programming, and conic linear programming.

1 Introduction

If you are an organizer of a local softball tournament this summer, you probably need to set up a schedule. Suppose there are n teams, each team plays every other team just once, and we don't consider whether the games are at home or away. Then if n is even, we have a total of $n - 1$ rounds with $\frac{n}{2}$ games for each round. If n is odd, we use a dummy team whose opponent does not play and is given a bye that round. So for the case that n is odd, we have a total of n rounds with $\frac{n-1}{2}$ games for each round. In the related literature, such type of tournament is called a round-robin tournament. In this paper, we just simply call it a tournament.

Im Englischen ist "round-robin tournament" die Bezeichnung für einen Wettkampf, bei dem jedes Team genau einmal gegen jedes andere antritt. Ist die Anzahl der Teams gerade, finden in jeder von $n - 1$ Runden $\frac{n}{2}$ Spiele statt. Der "round robin" Algorithmus ist eine effiziente Methode, um einen entsprechenden Spielplan zu erstellen. Aber wieviele Spielpläne sind überhaupt möglich? Die Autoren der vorliegenden Arbeit zeigen, wie dieser Frage mit Hilfe chromatischer Polynome nachgegangen werden kann. Beschränkt man sich auf die Anzahl der Spielpläne, die der "round robin" Algorithmus liefert, so gibt eine elegante Formel die Antwort.

Of course, there are different ways to set up a tournament schedule. A widely used method to generate a tournament schedule, which is called the round-robin tournament scheduling algorithm or simply the round-robin algorithm in this paper, will be reviewed in Section 3 and is described in [3] using modular arithmetic. In real-world problems, often optimal schedules based on some criteria are requested, for example, schedules having a minimum number of breaks [2], schedules in the presence of strength group requirements [1]. In this paper, we do not study specific scheduling strategies, instead we are interested in finding how many different schedules one can set up. We will describe a process to find the number of all tournament schedules using chromatic polynomials in Graph Theory. Since computing chromatic polynomials in general can be hard, to find the number of all tournament schedules could be very challenging. Finding a formula to compute the number of all tournament schedules is even more challenging. However, if we consider a subset of all tournament schedules that are generated by the round-robin algorithm, such a formula exists. We will provide a formula to find the number of schedules that are set up by the round-robin algorithm in this paper.

2 The number of tournament schedules

If there is an odd number of teams, then a dummy team can be added. Therefore, in this paper we assume we have an even number of teams. Suppose there are n teams. A schedule, therefore, consists of $n-1$ rounds of games with $\frac{n}{2}$ games for each round such that each team plays every other team just once. Mathematically, a schedule is a permutation of $n-1$ sets. Each set consists of $\frac{n}{2}$ games. Therefore, two schedules are equal if and only if the set of games in each round are the same, that is, the set of games in round one are the same, the set of games in round two are the same, etc.

In this section, we describe a process to find the number of all schedules using chromatic polynomials in Graph Theory. The process is described using Maple language as follows:

$G := \mathit{CompleteGraph}(n);$

$H := \mathit{LineGraph}(G);$

$P := \mathit{ChromaticPolynomial}(H, \text{‘}x\text{’});$

$P(n-1).$

Of course, we need to know some basic concepts in Graph Theory in order to understand these commands in Maple. We also need to verify that these commands return the number of all tournament schedules. Now we give some basic concepts in Graph Theory, [4].

Definition 1. A *graph* consists of two finite sets, V and E. Each element in V is called a vertex. The elements of E, called edges, are unordered pairs of vertices. A *complete graph* is a graph such that for any two vertices u and v, there is an edge connecting them, in other words, $uv \in E$. The *line graph* $L(G)$ of a graph G is defined in the way: the vertices of $L(G)$ are the edges of G, and two vertices in $L(G)$ are adjacent (there exists an edge connecting them) if and only if the corresponding edges in G share a vertex.

Definition 2. An *edge coloring* of a graph is an assignment of colors to the edges of the graph so that no two adjacent edges have the same color. A *vertex coloring* is a way of coloring the vertices of a graph such that no two adjacent vertices share the same color.

For a given graph G, the number of ways of coloring the vertices with x or fewer colors is denoted by $P(G, x)$ and is called the *chromatic polynomial* of G in terms of x.

We use K_n to denote the complete graph with n vertices. If we do an edge coloring of K_n, then we need at least $n-1$ colors. This is because for each vertex v in K_n, there are $n-1$ edges which have v as one of their end vertices. If we color the edges of K_n with exactly $n-1$ colors, namely, $c_1, c_2, \dots, c_{n-1}$, then for each color c_i and each vertex v, there is one and only one edge with color c_i and with v as an end vertex. Since for each edge there are two end vertices, we therefore obtain that there are $\frac{n}{2}$ edges for each color. Now if we view the n vertices of K_n as the n teams in a tournament and each edge $v_i v_j$ as a game between team v_i and v_j, then an edge coloring of K_n corresponds to a tournament schedule. This can be done by corresponding c_i with the ith round of the tournament, and by viewing $\frac{n}{2}$ edges with color c_i as the $\frac{n}{2}$ games in the ith round. With this in mind, to find the number of all tournament schedules with n teams, we only need to find the number of edge colorings of K_n. Since each edge coloring corresponds to a vertex coloring of its line graph $L(K_n)$ and each vertex coloring of $L(K_n)$ corresponds to an edge coloring of K_n, to find the number of edge colorings of K_n using $n-1$ colors is the same as finding the number of vertex colorings of $L(K_n)$ using $n-1$ colors, which can be obtained by first finding the chromatic polynomial $P(L(K_n), x)$ of $L(K_n)$, and then plugging in $x = n-1$ in $P(L(K_n), x)$.

Now let us go back to the Maple commands and explain their meanings:

$CompleteGraph(n)$ returns a complete graph with n vertices; $LineGraph(G)$ returns the line graph of G; $ChromaticPolynomial\ (H, `x')$ returns the number of its vertex colorings using no more than x colors; and $P(n-1)$ returns the number of its proper vertex colorings using no more than $n-1$ colors.

As an example of this process, we compute the number of schedules for six teams. Using Maple, we have the chromatic polynomial as follows:

$$\begin{aligned} P(x) = x(x-1)(x-2)(x-3)(x-4)\big(&x^{10} - 50x^9 + 1155x^8 \\ &- 16245x^7 + 154083x^6 - 1029213x^5 + 4896820x^4 \\ &- 16356845x^3 + 36630736x^2 - 49547792x + 30666816\big). \end{aligned}$$

Set $x = 5$, we have $P(5) = 720$. In other words, for six teams there are 720 different tournament schedules.

Remark 1. Maple takes a lot of time to find the chromatic polynomial for six teams (close to an hour). For eight teams, Maple returns an error message "Error, (in Matrix) object too large". Computationally, finding the number of tournament schedules could be very challenging.

3 The round-robin algorithm

The round-robin algorithm is to pair the teams off in the first round. For example, if there are eight teams named $p_0, p_1, \ldots, p_7$, we may initiate the first round of games as follows:

Round 1. (p_0 plays p_7 , p_1 plays $p_6, \ldots$) $S_1 = \begin{bmatrix} p_0 & p_1 & p_2 & p_3 \\ p_7 & p_6 & p_5 & p_4 \end{bmatrix}$.

Now, we fix team p_0 and rotate the others clockwise one position. We obtain the games for the second round.

Round 2. (p_0 plays p_6, p_7 plays $p_5, \ldots$) $S_2 = \begin{bmatrix} p_0 & p_7 & p_1 & p_2 \\ p_6 & p_5 & p_4 & p_3 \end{bmatrix}$.

We rotate the teams $p_1, p_2, \ldots, p_7$ clockwise one more position. We obtain the games for the third round.

Round 3. (p_0 plays p_5, p_6 plays $p_4, \ldots$) $S_3 = \begin{bmatrix} p_0 & p_6 & p_7 & p_1 \\ p_5 & p_4 & p_3 & p_2 \end{bmatrix}$.

. . . .

Round 7. (p_0 plays p_1, p_2 plays $p_7, \ldots$) $S_7 = \begin{bmatrix} p_0 & p_2 & p_3 & p_4 \\ p_1 & p_7 & p_6 & p_5 \end{bmatrix}$.

In the above tournament schedule, the games in each round (Round 2 to Round 7) are determined by the round-robin algorithm based on the games for the previous round. Of course, we do not have to follow this order. Actually, any permutation of $S_1, S_2, \ldots, S_7$ will give a specific tournament schedule. For example, $S_2S_5S_7S_1S_3S_6S_4$ indicates that the games in Round 1 are determined by S_2, the games in Round 2 are determined by S_5, etc. Therefore, we have a total of 7! different tournament schedules if one tournament schedule is given. We use the notation $\begin{pmatrix} p_0 & p_1 & \cdots & p_m \\ p_{2m+1} & p_{2m} & \cdots & p_{m+1} \end{pmatrix}$, which is also called a setting of the round-robin algorithm in this paper, to represent the set of all tournament schedules generated by the round-robin algorithm (p_0 is fixed and $p_1,\ p_2, \ldots,\ p_{2m+1}$ are rotated clockwise or equivalently counterclockwise) for $2m+2$ teams, $p_0,\ p_1, \ldots,\ p_{2m+1}$. Using this notation, we can see that $\begin{pmatrix} p_0 & p_1 & p_2 & p_3 \\ p_7 & p_6 & p_5 & p_4 \end{pmatrix}$ is the same as $\begin{pmatrix} p_0 & p_7 & p_1 & p_2 \\ p_6 & p_5 & p_4 & p_3 \end{pmatrix}$, which represents the set of all permutations of $S_1, S_2, \ldots, S_7$ and each permutation represents a schedule of a tournament.

In this study, since we do not consider home or away games, the games determined by $\begin{bmatrix} p_1 & p_2 & p_6 & p_4 \\ p_0 & p_7 & p_3 & p_5 \end{bmatrix}$ are the same as the ones determined by S_7. We should point out the difference between the notation

$$\begin{pmatrix} p_0 & p_1 & \cdots & p_m \\ p_{2m+1} & p_{2m} & \cdots & p_{m+1} \end{pmatrix} \quad \text{and} \quad \begin{bmatrix} p_0 & p_1 & \cdots & p_m \\ p_{2m+1} & p_{2m} & \cdots & p_{m+1} \end{bmatrix}.$$

While $\begin{pmatrix} p_0 & p_1 & \cdots & p_m \\ p_{2m+1} & p_{2m} & \cdots & p_{m+1} \end{pmatrix}$ represents the set of all tournament schedules generated by the round-robin algorithm, $\begin{bmatrix} p_0 & p_1 & \cdots & p_m \\ p_{2m+1} & p_{2m} & \cdots & p_{m+1} \end{bmatrix}$ represents the games, p_0 plays p_{2m+1}, p_1 plays p_{2m}, . . ., p_m plays p_{m+1}, in a round of a tournament.

The round-robin algorithm can also be represented by a graph, see Figure 1 for eight teams.

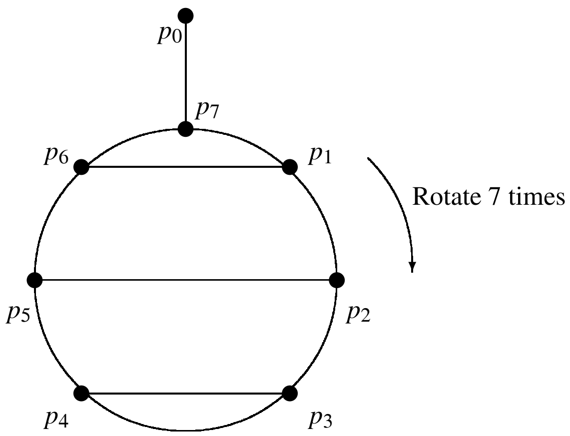

Figure 1 Round robin algorithm diagram

4 The number of tournament schedules by the round-robin algorithm

In this section, we first give several results stated as lemmas. Then we obtain an equality regarding the number of schedules using the round-robin algorithm when a specific team is fixed. Finally, we prove an equality regarding the number of schedules using the round-robin algorithm when the fixed team is arbitrarily selected.

For a setting of the round-robin algorithm, a team is fixed and the others are rotated clockwise or equivalently counterclockwise based on an initial assignment of games. Therefore, the following result is valid due to the fact that one clockwise rotation of M is the same as one counterclockwise rotation of N, where M and N are the ones appearing in Lemma 1.

Lemma 1. *Let*

$$M = \begin{pmatrix} p_0 & p_1 & \cdots & p_m \\ p_{2m+1} & p_{2m} & \cdots & p_{m+1} \end{pmatrix} \quad \textit{and} \quad N = \begin{pmatrix} p_0 & p_{2m} & \cdots & p_{m+1} \\ p_{2m+1} & p_1 & \cdots & p_m \end{pmatrix}.$$

Then $M = N$.

We point out that in the next lemma the notation $\{p_0, p_1\}$ represents a set of two elements p_0 and p_1 and $\{\{p_0, p_{2m+1}\}, \{p_1, p_{2m}\}, \ldots, \{p_m, p_{m-1}\}\}$ represents a set with elements $\{p_0, p_{2m+1}\}$,$\{p_1, p_{2m}\}, \ldots, \{p_m, p_{m-1}\}$. Actually, the proof of the next lemma becomes obvious if we view $\{p_0, p_1\}$ as the game that p_0 plays p_1.

Lemma 2. *Let*

$$M = \begin{pmatrix} p_0 & p_1 & \cdots & p_m \\ p_{2m+1} & p_{2m} & \cdots & p_{m+1} \end{pmatrix} \quad \textit{and} \quad N = \begin{pmatrix} q_0 & q_1 & \cdots & q_m \\ q_{2m+1} & q_{2m} & \cdots & q_{m+1} \end{pmatrix}.$$

If $M = N$ *and* $\{p_i, p_{2m-i+1}\} = \{q_j, q_{2m-j+1}\}$ *for some* $0 \le i, j \le m$*, then*

$$\{\{p_0, p_{2m+1}\}, \{p_1, p_{2m}\}, \ldots, \{p_m, p_{m+1}\}\} = \{\{q_0, q_{2m+1}\}, \{q_1, q_{2m}\}, \ldots, \{q_m, q_{m+1}\}\}.$$

Proof. It is easy to see that p_i and p_{2m-i+1} are in the same column in M and q_i and q_{2m-i+1} are in the same column in N. Consider the round that p_i plays p_{2m-i+1}. Since $M = N$ and $\{p_i, p_{2m-i+1}\} = \{q_j, q_{2m-j+1}\}$, we know that the games for this round generated by M and N are exactly the same. Therefore, the conclusion of the lemma is true. □

If p_0 is fixed, then a circular permutation of $p_1, p_2, \dots, p_{2m+1}$ corresponds to a setting of the round-robin algorithm. Though different circular permutations of $p_1, p_2, \dots, p_{2m+1}$ correspond to different settings of the round-robin algorithm, these different settings of the round-robin algorithm may generate the same set of schedules. For example, as Lemma 1 shows, if we reverse the order of the circular permutation, it will generate the same set of schedules. Even more as the following example shows, two totally different circular permutations can generate the same set of schedules.

Example 1. Let

$$M = \begin{pmatrix} p_0 & p_1 & p_2 & p_3 \\ p_7 & p_6 & p_5 & p_4 \end{pmatrix} \quad \text{and} \quad N = \begin{pmatrix} p_0 & p_4 & p_1 & p_5 \\ p_7 & p_3 & p_6 & p_2 \end{pmatrix}.$$

It can be easily seen that M and N correspond to different circular permutations. Let the first rounds of games based on M and N be the games

$$M_1 = \begin{bmatrix} p_0 & p_1 & p_2 & p_3 \\ p_7 & p_6 & p_5 & p_4 \end{bmatrix} \quad \text{and} \quad N_1 = \begin{bmatrix} p_0 & p_4 & p_1 & p_5 \\ p_7 & p_3 & p_6 & p_2 \end{bmatrix}, \text{ respectively.}$$

Then the second round of games M_2, the third round of games $M_3, \dots$, the seventh round of games M_7 based on M can be obtained by rotating $p_1, p_2, p_3, p_4, p_5, p_6, p_7$ in M_1 clockwise one position, two positions, ..., seven positions, respectively. Similarly, the second round of games N_2, the third round of games $N_3, \dots$, the seventh round of games N_7 based on N can be obtained by rotating $p_4, p_1, p_5, p_2, p_6, p_3, p_7$ in N_1 clockwise one position, two positions, ..., seven positions.

It is obvious that $M_1 = N_1$. It is also easily seen that $M_2 = N_3$, $M_3 = N_5$, $M_4 = N_7$, $M_5 = N_2$, $M_6 = N_4$, and $M_7 = N_6$. Because M and N are the sets of all the permutations of $M_1, M_2, \dots, M_7$ and $N_1, N_2, \dots, N_7$, respectively, we obtain $M = N$.

The next lemma shows that if two different circular permutations generate the same set of schedules, then they cannot be too different.

Lemma 3. *Let M be generated by*

$$\begin{pmatrix} p_0 & p_1 & \cdots & p_m \\ p_{2m+1} & p_{2m} & \cdots & p_{m+1} \end{pmatrix}. \tag{1}$$

If M is also generated by

$$\begin{pmatrix} p_0 & a_1 & \cdots & a_m \\ p_{2m+1} & a_{2m} & \cdots & a_{m+1} \end{pmatrix} \tag{2}$$

and

$$\begin{pmatrix} p_0 & b_1 & \dots & b_m \\ p_{2m+1} & b_{2m} & \dots & b_{m+1} \end{pmatrix} \tag{3}$$

with $a_j = b_j = p_1$ *and* $a_{2m-j+1} = b_{2m-j+1} = p_{2m}$ *for a fixed* $1 \le j \le m$*, then* $a_i = b_i$ *for all* $1 \le i \le m$*.*

Proof. In view of $a_{2m-j+1} = b_{2m-j+1} = p_{2m}$, we consider the round when p_0 plays p_{2m}. The games in this round can be easily seen by an appropriate number of rotations in (1)–(3). Respectively, we obtain

$$\begin{bmatrix} p_0 & p_{2m+1} & p_1 & \dots & p_{m-1} \\ p_{2m} & p_{2m-1} & p_{2m-2} & \dots & p_m \end{bmatrix}, \tag{4}$$

$$\begin{bmatrix} p_0 & a_{2m-j+2} & \dots & a_{2m} & p_{2m+1} & a_1 & \dots & a_{m-j} \\ a_{2m-j+1} & a_{2m-j} & \dots & a_{2m-2j+2} & a_{2m-2j+1} & a_{2m-2j} & \dots & a_{m-j+1} \end{bmatrix}, \tag{5}$$

and

$$\begin{bmatrix} p_0 & b_{2m-j+2} & \dots & b_{2m} & p_{2m+1} & b_1 & \dots & b_{m-j} \\ b_{2m-j+1} & b_{2m-j} & \dots & b_{2m-2j+2} & b_{2m-2j+1} & b_{2m-2j} & \dots & b_{m-j+1} \end{bmatrix}, \tag{6}$$

where (4) is based on (1), (5) is based on (2), and (6) is based on (3). The games determined by (4)–(6) are the same. Therefore, we obtain that $a_{2m-2j+1} = b_{2m-2j+1} = p_{2m-1}$. In other words, p_{2m-1} appears in the same location in (5) and (6), and therefore, appears in the same location in (2) and (3). Now we consider the round that p_0 plays p_{2m-1}. If we make an appropriate number of rotations in (4)–(6) and apply the same argument as above, we obtain that p_{2m-2} appears in the same location in (2) and (3). Keep doing this repeatedly, and we get that $a_i = b_i$ for all $1 \le i \le 2m$. □

Lemma 3 shows that if there is a circular permutation which corresponds to a setting of the round-robin algorithm that generates the same set of schedules as in (1), then up to reversion this circular permutation is uniquely determined by the location of the column $\begin{pmatrix} p_1 \\ p_{2m} \end{pmatrix}$ in its corresponding setting of the round-robin algorithm for the round that p_0 plays p_{2m+1}. However, not every location of the column $\begin{pmatrix} p_1 \\ p_{2m} \end{pmatrix}$ in its corresponding setting of the round-robin algorithm for the round that p_0 plays p_{2m+1} will give the same set of schedules. We have the following example for $m = 4$ indicating that if we shift the second column to the fourth column, these settings do not generate the same set of schedules.

Example 2. Let

$$M = \begin{pmatrix} p_0 & p_1 & p_2 & p_3 & p_4 \\ p_9 & p_8 & p_7 & p_6 & p_5 \end{pmatrix} \quad \text{and} \quad N = \begin{pmatrix} p_0 & x_1 & x_2 & p_1 & x_4 \\ p_9 & x_8 & x_7 & p_8 & x_5 \end{pmatrix}.$$

Then there does not exist $x_1,\ x_2,\ x_4,\ x_5,\ x_7,\ x_8$ such that $M = N$. Suppose there are $x_1, x_2, x_4, x_5, x_7, x_8$ such that $M = N$.

Consider the round that p_0 plays p_8. Based on M, we get the games

$$\begin{bmatrix} p_0 & p_9 & p_1 & p_2 & p_3 \\ p_8 & p_7 & p_6 & p_5 & p_4 \end{bmatrix}. \tag{7}$$

Based on N, we have the games

$$\begin{bmatrix} p_0 & x_7 & x_8 & p_9 & x_1 \\ p_8 & x_5 & x_4 & p_1 & x_2 \end{bmatrix}. \tag{8}$$

Obviously, from (7) we know that p_0 plays p_8 and p_9 plays p_7. From (8) we know that p_0 plays p_8 and p_1 plays p_9. By Lemma 2, we know that $M \neq N$.

It would be interesting to ask what circular permutations will generate the same set of schedules. The next lemma provides a necessary and sufficient condition for circular permutations that lead to the same set of schedules.

Lemma 4. *Let M be*

$$\begin{pmatrix} p_0 & p_1 & \cdots & p_m \\ p_{2m+1} & p_{2m} & \cdots & p_{m+1} \end{pmatrix} \tag{9}$$

and N be

$$\begin{pmatrix} p_0 & a_1 & \cdots & a_m \\ a_{2m+1} & a_{2m} & \cdots & a_{m+1} \end{pmatrix} \tag{10}$$

with $a_{2m+1} = p_{2m+1}$ *and* $a_k = p_1$ *for a fixed* $k \in \{1, 2, \ldots, m\}$*. Then* $M = N$ *if and only if* $gcd(2m+1, k) = 1$ *and* $a_{[ik]} = p_i$ *for* $i \in \{1, 2, \ldots, 2m+1\}$*, where* $[x]$ *denotes the unique integer in* $\{1, 2, \ldots, 2m+1\}$ *congruent to x modulo* $2m+1$ *and* $gcd(k_1, k_2)$ *represents the greatest common divisor of natural numbers* k_1 *and* k_2*.*

Proof. First let us prove that if $M = N$, then $a_{[ik]} = p_i$.
By an appropriate number of rotations, (9) and (10) equivalently become

$$\begin{pmatrix} p_0 & p_2 & p_3 & \cdots & p_{m+1} \\ p_1 & p_{2m+1} & p_{2m} & \cdots & p_{m+2} \end{pmatrix} \tag{11}$$

and

$$\begin{pmatrix} p_0 & a_{k+1} & \cdots & a_{2k-1} & a_{2k} & a_{2k+1} & \cdots & a_{m+k} \\ a_k(= p_1) & a_{k-1} & \cdots & a_1 & p_{2m+1} & a_{2m} & \cdots & a_{m+k+1} \end{pmatrix}, \tag{12}$$

respectively. The games determined by (11) and (12) for the round that p_0 plays p_1 are the same. Therefore, we obtain that $a_{[2k]} = a_{2k} = p_2$. Now let $q_i = p_{[i+1]}$ in (11) and $b_i = a_{[k+i]}$ in (12) for $i = 1, 2, \ldots, 2m+1$. Then (11) and (12) become

$$\begin{pmatrix} p_0 & q_1 & \cdots & q_m \\ q_{2m+1}(= p_1) & q_{2m} & \cdots & q_{m+1} \end{pmatrix} \tag{13}$$

and

$$\begin{pmatrix} p_0 & b_1 & \cdots & b_m \\ b_{2m+1}(= p_1) & b_{2m} & \cdots & b_{m+1} \end{pmatrix} \tag{14}$$

with $b_k = a_{2k} = p_2 = q_1$.

Now we consider the round that p_0 plays p_1. Using (13) and (14) and applying the same argument as the one we use to obtain $a_{[2k]} = a_{2k} = p_2$, we know that $b_{2k} = q_2$, which implies that $a_{[3k]} = p_3$. Similarly, by relabeling the teams and using the same argument, we can prove that $a_{[ik]} = p_i$ for $i = 4, 5, \ldots, 2m + 1$.

Next we prove that if $M = N$, then $gcd(2m + 1, k) = 1$. We prove it by contradiction. Suppose that $gcd(2m+1, k) = l > 1$. Then $\frac{2m+1}{l} + 1$ is an integer in $\{2, \ldots, 2m+1\}$ and $[(\frac{2m+1}{l} + 1)k] = [k]$ because $(\frac{2m+1}{l} + 1)k - k = \frac{k}{l} \times (2m + 1)$. Therefore, $p_1 = a_{[k]} = a_{[(\frac{2m+1}{l}+1)k]} = p_{(\frac{2m+1}{l}+1)}$, which is impossible due to the fact that $\frac{2m+1}{l} + 1 \neq 1$. Hence, $gcd(2m + 1, k) = 1$.

On the other hand, if $gcd(2m + 1, k) = 1$ and $a_{[ik]} = p_i$ for $i \in \{1, 2, \ldots, 2m + 1\}$, then $[ik] \neq [jk]$ for $i \neq j$ and $i, j \in \{1, 2, \ldots, 2m + 1\}$. Therefore, $\{a_i : i = 1, 2, \ldots, 2m + 1\} = \{a_{[ik]} : i = 1, 2, \ldots, 2m + 1\}$. This shows that for each a_i, $i \in \{1, 2, \ldots, 2m + 1\}$, there is a $j \in \{1, 2, \ldots, 2m + 1\}$ such that $a_i = a_{[jk]} = p_j$. We now prove that $M = N$. Consider the round that p_0 plays p_{2m+1}. Let (a_{i_0}, a_{2m+1-i_0}) be any pair representing a column in (10) with $i_0 \in \{1, 2, \ldots, 2m + 1\}$. We need to show that (a_{i_0}, a_{2m+1-i_0}) is also a pair representing a column in (9). For a_{i_0}, there is an $r \in \{1, 2, \ldots, 2m + 1\}$ such that $a_{i_0} = a_{[rk]} = p_r$. For a_{2m+1-i_0}, there is an $s \in \{1, 2, \ldots, 2m + 1\}$ such that $a_{2m+1-i_0} = a_{[sk]} = p_s$. Hence, $i_0 = [rk]$ and $2m + 1 - i_0 = [sk]$, which give that $rk - i_0$ and $sk - (2m + 1 - i_0)$ are multiples of $2m + 1$. Therefore, $(r + s)k$ is a multiple of $2m + 1$. Since $gcd(2m + 1, k) = 1$, $r, s \in \{1, 2, \ldots, 2m + 1\}$ and $r \neq s$, we obtain that $r + s = 2m + 1$, which implies that (p_r, p_s) $((a_{i_0}, a_{2m+1-i_0}))$ is a pair representing a column in (9). Because $i_0 \in \{1, 2, \ldots, 2m + 1\}$ is arbitrarily chosen, we know that M and N lead to the same set of games for the round p_0 plays p_{2m+1}. We now consider the round that p_0 plays p_1. Using (13) and (14) and applying the same argument, we obtain that each game determined by a pair (b_{i_0}, b_{2m+1-i_0}) in (14) for $i_0 \in \{1, 2, \ldots, 2m + 1\}$ is also a game determined by a pair in (13) . Since (13) is the same as (9) and (14) is the same as (10), we obtain that M and N lead to the same set of games for the round p_0 plays p_1. Similarly, we can prove that M and N lead to the same set of games for the rounds that p_0 plays p_2, p_0 plays $p_3, \ldots,$ and p_0 plays p_{2m}. Hence, $M = N$. □

Let us look back at Examples 1 and 2. In Example 1, $m = 3$ and $k = 2$, so $gcd(2m + 1, k) = 1$. By Lemma 4, we know that there is a different setting of the round-robin algorithm leading to the same set of schedules (with p_1 being in the first row and the third column). In Example 2, we know $m = 4$ and $k = 3$, therefore, $gcd(2m + 1, k) = 3$, which implies that no such setting of the round-robin algorithm leading to the same set of schedules exists by Lemma 4.

Now we are ready to give the first result regarding the number of schedules. We use $\phi(2m+1)$ to denote the Euler totient, which is the number of positive integers less than or equal to $2m + 1$ that are relatively prime to $2m + 1$.

Theorem 1. *Suppose there are* $2m+2$ *teams, namely* $p_0, p_1, \ldots, p_{2m+1}$. *Let* $n(m)$ *denote the number of different schedules with* p_0 *being fixed using the round-robin algorithm. Then*

$$n(m) = \frac{(2m)!(2m + 1)!}{\phi(2m + 1)}.$$

Proof. There are $2m+2$ teams, and hence there are $2m+1$ rounds. If we have a tournament schedule, we may reorder the rounds, so we can obtain different schedules. In other words, a given tournament schedule determines a set of $(2m+1)!$ different schedules.

Since p_0 is fixed, a circular permutation of $p_1, p_2, \ldots, p_{2m+1}$ corresponds to a set of tournament schedules. There are $(2m)!$ different circular permutations. By Lemma 1, we know that if we reverse the order of a circular permutation, we will obtain the same set of tournament schedules by using the round-robin algorithm. If a circular permutation is given, for example, a circular permutation corresponds to (1), then a different circular permutation which generates the same set of tournament schedules as (1) is determined by the position of the column $\begin{pmatrix} p_1 \\ p_{2m} \end{pmatrix}$ by Lemma 3. By Lemma 4, for each $k \in \{1, 2, \ldots, m\}$ that is relatively prime to $2m+1$, there is a circular permutation which generates the same set of tournament schedules. There are $\phi(2m+1)/2$ numbers in $\{1, 2, \ldots, m\}$ that are relatively prime to $2m+1$. Combining all these results, we obtain that the number of tournament schedules is equal to $\frac{(2m)!(2m+1)!}{\phi(2m+1)}$. □

Next, we try to answer the question: Is it possible to get the same set of tournament schedules using a different fixed team by the round-robin algorithm? We find that this is only possible for the case that there are 4 teams or 6 teams. Actually, by a straightforward computation, we can see that

$$\begin{pmatrix} p_0 & p_1 \\ p_3 & p_2 \end{pmatrix} = \begin{pmatrix} p_3 & p_1 \\ p_0 & p_2 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} p_0 & p_1 & p_2 \\ p_5 & p_4 & p_3 \end{pmatrix} = \begin{pmatrix} p_5 & p_2 & p_1 \\ p_0 & p_3 & p_4 \end{pmatrix}.$$

Therefore, the numbers of schedules generated by the round-robin algorithm for four and six teams are obtained by Theorem 1, that is, six schedules for four teams and 720 schedules for six teams.

If there are more than six teams, then we have the following theorem.

Theorem 2. *Let $p_0, p_1, \ldots, p_{2m+1}$ represent $2m+2$ teams. If we use the round-robin algorithm to generate tournament schedules with different fixed teams, for example, p_0 and p_{2m+1}, then the resulting schedules are different for $m \geq 3$.*

Proof. We assume that the tournament schedules are generated by

$$\begin{pmatrix} p_0 & p_1 & \cdots & p_m \\ p_{2m+1} & p_{2m} & \cdots & p_{m+1} \end{pmatrix}. \tag{15}$$

Suppose to the contrary that the schedules can be generated by the round-robin algorithm using a different fixed team. We may assume that the schedules are generated by

$$\begin{pmatrix} p_{2m+1} & x_1 & p_i & x_3 & \cdots & x_m \\ p_0 & x_{2m} & p_{2m-i+1} & x_{2m-2} & \cdots & x_{m+1} \end{pmatrix} \tag{16}$$

with $1 \leq i \leq m$ being a fixed index and x_j, $1 \leq j \leq 2m$, $j \neq 2$, and $j \neq 2m-1$, to be determined.

Consider the round when p_0 plays p_{2m-i+1}. Using (15) we obtain

$$\begin{bmatrix} p_0 & p_{2m-i+2} & p_{2m-i+3} & \cdots & p_{2m} & p_{2m+1} & p_1 & \cdots & p_{m-i} \\ p_{2m-i+1} & p_{2m-i} & p_{2m-i-1} & \cdots & p_{2m-2i+2} & p_{2m-2i+1} & p_{2m-2i} & \cdots & p_{m-i+1} \end{bmatrix}. \quad (17)$$

We should point out that when $m - i = 0$, (17) should be understood as

$$\begin{bmatrix} p_0 & p_{m+2} & p_{m+3} & \cdots & p_{2m} & p_{2m+1} \\ p_{m+1} & p_m & p_{m-1} & \cdots & p_2 & p_1 \end{bmatrix}.$$

Using (16), we have

$$\begin{bmatrix} p_{2m+1} & p_0 & x_1 & p_i & x_3 & \cdots & x_{m-1} \\ x_{2m} & p_{2m-i+1} & x_{2m-2} & x_{2m-3} & x_{2m-4} & \cdots & x_m \end{bmatrix}. \quad (18)$$

Comparing (17) with (18), we obtain that $x_{2m} = p_{2m-2i+1}$, which in view of (15) and (16) further implies that $x_1 = p_{2i}$. Now we will try to find x_{2m-2} and x_{2m-3}. It depends on where p_i and p_{2i} are located in (17). We consider the following cases, which list all the possible locations of p_i and p_{2i} in (17).

Case 1: $i \le m - i$ and $2i \le m - i$. In this case, we obtain that $x_{2m-2} = p_{2m-4i+1}$ and $x_{2m-3} = p_{2m-3i+1}$ by comparing (17) and (18). Consider the round that p_0 plays $p_{2m-3i+1}$. Since $2m - 3i + 1 \ge m + 1$ due to the assumption that $2i \le m - i$, $p_{2m-3i+1}$ is in the second row of (15). By the round-robin algorithm and using (15), we obtain that

$$\begin{bmatrix} p_0 & p_{2m-3i+2} & \cdots & p_{2m} & p_{2m+1} & p_1 & \cdots & p_{m-3i} \\ p_{2m-3i+1} & p_{2m-3i} & \cdots & p_{2m-6i+2} & p_{2m-6i+1} & p_{2m-6i} & \cdots & p_{m-3i+1} \end{bmatrix}. \quad (19)$$

Using (18), we have

$$\begin{bmatrix} p_{2m+1} & p_{2m-2i+1} & p_0 & p_{2i} & p_i & \cdots & x_{m-2} \\ p_{2m-i+1} & p_{2m-4i+1} & p_{2m-3i+1} & x_{2m-4} & x_{2m-5} & \cdots & x_{m-1} \end{bmatrix}. \quad (20)$$

Because the games determined by (19) and (20) are the same, we know that $p_{2m-i+1} = p_{2m-6i+1}$, which implies that $2m - 6i + 1 = 2m - i + 1$. It is impossible.

Case 2: $i \le m - i$ and $2i \ge m - i + 1$, that is, $\frac{m+1}{3} \le i \le \frac{m}{2}$. In this case, we know p_i is in the first row and p_{2i} is in the second row of (17) and $2i \le 2m - 2i$. By the same argument as in Case 1, we obtain that $x_{2m-2} = p_{2m-4i+1}$ and $x_{2m-3} = p_{2m-3i+1}$. Since $2m - 3i + 1 \le m$ due to the assumption that $2i \ge m - i + 1$, $p_{2m-3i+1}$ is in the first row of (15). Consider the round that p_0 plays $p_{2m-3i+1}$. Using (15), we obtain

$$\begin{bmatrix} p_0 & p_{2m-3i+2} & \cdots & p_{4m-6i+1} & p_{4m-6i+2} & p_{4m-6i+3} & \cdots & p_{3m-3i+1} \\ p_{2m-3i+1} & p_{2m-3i} & \cdots & p_1 & p_{2m+1} & p_{2m} & \cdots & p_{3m-3i+2} \end{bmatrix}. \quad (21)$$

Using (18), we obtain (20). Comparing (20) with (21), we have $p_{4m-6i+2} = p_{2m-i+1}$, which shows that $4m - 6i + 2 = 2m - i + 1$. So $2m + 1 = 5i$. Therefore, $x_{2m-2} = p_{2m-4i+1} = p_i$, which is impossible due to (16) unless $m = 2$.

Case 3: $m-i+1 \le i \le 2m-2i$. In this case, we also have $2m-2i+1 \le 2i \le 2m-i+1$. It shows that p_i and p_{2i} are in different portions of the second row separated by p_{2m-2i} in (17). We obtain that $x_{2m-2} = p_{4m-4i+2}$ and $x_{2m-3} = p_{2m-3i+1}$ by comparing (17) and (18). Consider the round that p_0 plays $p_{2m-3i+1}$. From $2m-2i+1 \le 2i$, we know $2m-3i+1 \le i \le m$. Now using (15) we obtain (21). Using (18), we have

$$\begin{bmatrix} p_{2m+1} & p_{2m-2i+1} & p_0 & p_{2i} & p_i & \cdots & x_{m-2} \\ p_{2m-i+1} & p_{4m-4i+1} & p_{2m-3i+1} & x_{2m-4} & x_{2m-5} & \cdots & x_{m-1} \end{bmatrix}. \tag{22}$$

Comparing (21) with (22), we obtain that $p_{2m-i+1} = p_{4m-6i+2}$, which implies $2m+1 = 5i$. If $m \ge 3$, then $x_{2m-2} = p_{4m-4i+2} = p_{6i}$, which in view of (15) implies that $x_3 = p_{2m-6i+1}$. Using the fact that $2m+1 = 5i$, we get a negative index $-i$, which is impossible.

Case 4: $2m-2i+1 \le i \le 2m-i+1$ and $2m-2i+1 \le 2i \le 2m-i+1$. In this case, we obtain that $\frac{2m+1}{3} \le i \le \frac{2m+1}{3}$. So we have $2m+1 = 3i$, plugging this into (16) and noting that $x_1 = p_{2i}$, we know $p_{2m-i+1} = p_{2i} = x_1$, which is impossible.

Case 5: $2m-2i+1 \le i \le 2m-i+1$ and $2m-i+2 \le 2i \le 2m+1$. We, therefore, have $\frac{2m+2}{3} \le i \le m+\frac{1}{2}$. We now have $x_{2m-2} = p_{4m-4i+2}$ and $x_{2m-3} = p_{4m-3i+2}$ by comparing (17) and (18). Consider the round when p_0 plays $p_{4m-3i+2}$. Using (15) and in view of the fact that $4m-3i+2 \ge m+1$ (due to the assumption $i \le m$), we have

$$\begin{bmatrix} p_0 & p_{4m-3i+3} & \cdots & p_{2m} & p_{2m+1} & p_1 & \cdots & p_{3m-3i+1} \\ p_{4m-3i+2} & p_{4m-3i+1} & \cdots & p_{6m-6i+4} & p_{6m-6i+3} & p_{6m-6i+2} & \cdots & p_{3m-3i+2} \end{bmatrix}. \tag{23}$$

Using (18), we obtain

$$\begin{bmatrix} p_{2m+1} & p_{2m-2i+1} & p_0 & p_{2i} & p_i & \cdots & x_{m-2} \\ p_{2m-i+1} & p_{4m-4i+2} & p_{4m-3i+2} & x_{2m-4} & x_{2m-5} & \cdots & x_{m-1} \end{bmatrix}. \tag{24}$$

Comparing (23) with (24), we have $p_{2m-i+1} = p_{6m-6i+3}$, which implies $5i = 4m+2$. If $m \ge 3$, then $x_{2m-2} = p_{4m-4i+2} = p_i$, which is impossible since p_i appears twice in (16).

Since Cases 1–5 include all the possibilities of the locations of p_i and p_{2i} in (17), we, therefore, complete the proof of the theorem. □

Using Theorem 2, we have the following result.

Theorem 3. *Suppose there are $2m+2$ ($m \ge 3$) teams, $p_0, p_1, \dots, p_{2m+1}$. Let $T(m)$ denote the number of tournament schedules using the round-robin algorithm. Then*

$$T(m) = \frac{(2m+2)(2m)!(2m+1)!}{\phi(2m+1)}.$$

Proof. For $m \ge 3$, by Theorem 2, we know for different fixed teams, we obtain different sets of tournament schedules. There are a total of $2m+2$ teams. Therefore, $T(m) = (2m+2)n(m)$, which by Theorem 1 indicates that the equality is true. □

5 Does the round-robin algorithm generate all the schedules?

For six teams, both approaches using chromatic polynomials and Theorem 1 give 720 different schedules. Therefore, in this case the round-robin algorithm generates all schedules. For two teams or four teams, we can verify in a straightforward manner that all schedules are generated by the round-robin algorithm. Is the statement still true if there are more than six teams? The answer is no. In fact, two, four, and six are the only numbers with the property that all schedules of the tournament are generated by the round-robin algorithm. We have the following examples for more than six teams. We first consider the case that there are an even number of games in each round. In other words, there are $4p$ teams in the tournament with $p \geq 2$.

Example 3. Suppose there are $4p$ teams in the tournament, which gives an even number of games in each round of the tournament. We divide them into two groups called Group A and Group B. In each group there are $2p$ teams. We use $a_0, a_1, \dots, a_{2p-1}$ and $b_0, b_1, \dots, b_{2p-1}$ to denote the teams in Group A and Group B, respectively. Now we construct a schedule that will be proved not to be generated by the round-robin algorithm. We use

$$\left(\begin{array}{cccc|cccc} a_0 & a_1 & \dots & a_{p-1} & b_0 & b_1 & \dots & b_{p-1} \\ a_{2p-1} & a_{2p-2} & \dots & a_p & b_{2p-1} & b_{2p-2} & \dots & b_p \end{array} \right) \tag{25}$$

to denote $2p-1$ rounds of games concatenated by the games generated by the round-robin algorithm individually in Group A and Group B. In other words, these $2p-1$ rounds of games are as follows:

$$\begin{bmatrix} a_0 & a_1 & \dots & a_{p-1} & b_0 & b_1 & \dots & b_{p-1} \\ a_{2p-1} & a_{2p-2} & \dots & a_p & b_{2p-1} & b_{2p-2} & \dots & b_p \end{bmatrix},$$

$$\begin{bmatrix} a_0 & a_{2p-1} & \dots & a_{p-2} & b_0 & b_{2p-1} & \dots & b_{p-2} \\ a_{2p-2} & a_{2p-3} & \dots & a_{p-1} & b_{2p-2} & b_{2p-3} & \dots & b_{p-1} \end{bmatrix},$$

$$\dots\dots\dots$$

$$\begin{bmatrix} a_0 & a_2 & \dots & a_p & b_0 & b_2 & \dots & b_p \\ a_1 & a_{2p-1} & \dots & a_{p+1} & b_1 & b_{2p-1} & \dots & b_{p+1} \end{bmatrix}.$$

We use

$$\left(\begin{array}{cccc} a_0 & a_1 & \dots & a_{p-1} \\ b_{2p-1} & b_{2p-2} & \dots & b_p \end{array} \right] \begin{array}{cccc} b_0 & b_1 & \dots & b_{p-1} \\ a_{2p-1} & a_{2p-2} & \dots & a_p \end{array} \right) \tag{26}$$

to denote p rounds of games concatenated by the games obtained by fixing a_0, a_1, ..., a_{p-1}, b_0, b_1, ..., b_{p-1}, and rotating the sequences $b_{2p-1}, b_{2p-2}, \dots, b_p$ and a_{2p-1}, a_{2p-2}, ..., a_p simultaneously. These p rounds of games are

$$\begin{bmatrix} a_0 & a_1 & \dots & a_{p-1} & b_0 & b_1 & \dots & b_{p-1} \\ b_{2p-1} & b_{2p-2} & \dots & b_p & a_{2p-1} & a_{2p-2} & \dots & a_p \end{bmatrix},$$

$$\begin{bmatrix} a_0 & a_1 & \dots & a_{p-1} & b_0 & b_1 & \dots & b_{p-1} \\ b_{2p-2} & b_{2p-3} & \dots & b_{2p-1} & a_{2p-2} & a_{2p-3} & \dots & a_{2p-1} \end{bmatrix},$$

$$\dots\dots\dots$$

$$\begin{bmatrix} a_0 & a_1 & \dots & a_{p-1} & b_0 & b_1 & \dots & b_{p-1} \\ b_p & b_{2p-1} & \dots & b_{p+1} & a_p & a_{2p-1} & \dots & a_{p+1} \end{bmatrix}.$$

With this notation,

$$\left(\begin{array}{cccc} a_0 & a_1 & \dots & a_{p-1} \\ b_0 & b_1 & \dots & b_{p-1} \end{array} \right] \left. \begin{array}{cccc} a_p & a_{p+1} & \dots & a_{2p-1} \\ b_p & b_{p+1} & \dots & b_{2p-1} \end{array} \right) \tag{27}$$

represents the other p rounds of games. Now, if we put the rounds in (25)–(27) together, we get $4p-1$ rounds of games, which give us a schedule of the tournament with $4p$ teams. We should also point out that for each round in (25) teams in Group A only play teams in Group A and teams in Group B only play teams in Group B, and for each round in (26) and (27) teams in Group A only play teams in Group B and teams in Group B only play teams in Group A. Now we prove that the round-robin algorithm does not generate this schedule.

Suppose to the contrary that the round-robin algorithm generates this schedule. Without loss of generality, we may assume that a_i is the fixed team in the setting of the round-robin algorithm. In the circular permutation which corresponds to the setting of the round-robin algorithm, if at least two consecutive teams are from group A, for example, a_j and a_k, then the round when a_i plays a_j, which is a round in (25), can be written as follows:

$$\begin{bmatrix} a_i & a_k & x_{4p-3} & \dots & x_{2p} \\ a_j & x_1 & x_2 & \dots & x_{2p-1} \end{bmatrix}.$$

Because teams in Group A only play teams in Group A in all rounds in (25), we should know that x_1 should be a team in Group A. By rotating the sequence $a_j, a_k, x_{4p-3}, \dots, x_1$ clockwise one position, we have a round with a game that a_i plays x_1. Since x_1 is in Group A, we know that this round is in (25). However, in all rounds in (25), teams in Group A play teams in Group A, we obtain that x_2 and x_3 are in Group A, too. In a similar way, we obtain that all x_i, $1 \le i \le 4p-3$ are in Group A. Of course, this is impossible because no teams in Group B are presented in the setting of the round-robin algorithm. Therefore, we cannot have two consecutive teams from Group A in a setting of the round-robin algorithm in order to generate a schedule determined by (25)–(27).

Now, suppose there is a setting of the round-robin algorithm that generates a schedule determined by (25)–(27). Then in the circular permutation corresponding to the setting of the round-robin algorithm, teams in Group A should be separated by teams in Group B. In other words, no two consecutive teams are from Group A. Since a_i is fixed, we only have $2p-1$ teams in Group A and $2p$ teams in Group B in the circular permutation. Therefore, there exists one and only one subsequence with two consecutive teams from Group B, namely b_j, b_k. We now consider a round that is as follows:

$$\begin{bmatrix} a_i & \dots & b_j & b_k & \dots \\ z & \dots & x & y & \dots \end{bmatrix}.$$

In this round, b_j plays x and b_k plays y. Since no two consecutive teams are from Group A and b_j, b_k is the only subsequence with two consecutive teams from Group B, we obtain that one of x and y is in Group A and the other one is in Group B. Therefore, either the game that b_j plays x or the game that b_k plays y is a game between two teams in Group B and the other game is one between a team in Group A and a team in Group B. This is impossible because for each round in (25) teams in Group A only play teams in Group A and teams in Group B only play teams in Group B, and for each round in (26) and (27) teams in Group A only play teams in Group B and teams in Group B only play teams in Group A. Therefore, the round-robin algorithm does not generate the schedule which is put together by (25)–(27).

Example 4. Suppose there are an odd number of games in each round of the tournament. We may assume that there are $4p-2$ teams with $p \geq 3$. As in the previous example, we divide these $4p-2$ teams into two groups still called Group A and Group B. In each group, there are now $2p-1$ teams. We use $a_1, a_2, \ldots, a_{2p-1}$ to represent teams in Group A and $b_1, b_2, \ldots, b_{2p-1}$ to represent teams in Group B. In order to use the round-robin algorithm within each group, we add a dummy team for each group. We add a_0 to Group A and b_0 to Group B. Now we construct a schedule that will be proved not to be generated by the round-robin algorithm. We use

$$\left(\begin{array}{cccc} a_0 & a_1 & \ldots & a_{p-1} \\ a_{2p-1} & a_{2p-2} & \ldots & a_p \end{array} \Bigg\| \begin{array}{cccc} b_0 & b_1 & \ldots & b_{p-1} \\ b_{2p-1} & b_{2p-2} & \ldots & b_p \end{array}\right) \tag{28}$$

to denote $2p-1$ rounds of games concatenated by the games generated by the round-robin algorithm individually in Group A and Group B. We note that if a_0 plays a_i and b_0 plays b_j, because a_0 and b_0 are dummy teams, we should understand this is equivalent to the game that a_i plays b_j. With this in mind, the set of rounds in (28) consists of the following rounds

$$\left[\begin{array}{ccccccc} a_{2p-1} & a_1 & \ldots & a_{p-1} & b_1 & \ldots & b_{p-1} \\ b_{2p-1} & a_{2p-2} & \ldots & a_p & b_{2p-2} & \ldots & b_p \end{array}\right],$$

$$\left[\begin{array}{ccccccc} a_{2p-2} & a_{2p-1} & \ldots & a_{p-2} & b_{2p-1} & \ldots & b_{p-2} \\ b_{2p-2} & a_{2p-3} & \ldots & a_{p-1} & b_{2p-3} & \ldots & b_{p-1} \end{array}\right],$$

$$\ldots\ldots\ldots$$

$$\left[\begin{array}{ccccccc} a_1 & a_2 & \ldots & a_p & b_2 & \ldots & b_p \\ b_1 & a_{2p-1} & \ldots & a_{p+1} & b_{2p-1} & \ldots & b_{p+1} \end{array}\right].$$

We use

$$\left(\begin{array}{cccc} a_0 & a_1 & \ldots & a_{p-1} \\ b_{2p-1} & b_{2p-2} & \ldots & b_p \end{array} \Bigg[\Bigg] \begin{array}{cccc} b_0 & b_1 & \ldots & b_{p-1} \\ a_{2p-2} & a_{2p-3} & \ldots & a_{2p-1} \end{array}\right) \tag{29}$$

to denote p rounds of games concatenated by the games obtained by fixing a_0, a_1, …, a_{p-1}, $b_0, b_1, \ldots, b_{p-1}$, and rotating the sequences $b_{2p-1}, b_{2p-2}, \ldots, b_p$ and a_{2p-2}, a_{2p-3},

..., a_p, a_{2p-1} simultaneously. In other words, the set of the rounds in (29) consists of the following rounds

$$\begin{bmatrix} a_{2p-2} & a_1 & \dots & a_{p-1} & b_1 & \dots & b_{p-1} \\ b_{2p-1} & b_{2p-2} & \dots & b_p & a_{2p-3} & \dots & a_{2p-1} \end{bmatrix},$$

$$\begin{bmatrix} a_{2p-3} & a_1 & \dots & a_{p-1} & b_1 & \dots & b_{p-1} \\ b_{2p-2} & b_{2p-3} & \dots & b_{2p-1} & a_{2p-4} & \dots & a_{2p-2} \end{bmatrix},$$

$$\dots\dots\dots$$

$$\begin{bmatrix} a_{2p-1} & a_1 & \dots & a_{p-1} & b_1 & \dots & b_{p-1} \\ b_p & b_{2p-1} & \dots & b_{p+1} & a_{2p-2} & \dots & a_p \end{bmatrix}.$$

We note that for each round in (28), there is only one game between a team in Group A and a team in Group B which can be described as a_i plays b_i. All other games are played within their groups. We also find that for each round in (29), there is no game between a team in $\{a_1, a_2, \dots, a_{p-1}\}$ and a team in $\{b_1, b_2, \dots, b_{p-1}\}$, and there is only one game between a team in $\{a_p, a_{p+1}, \dots, a_{2p-1}\}$ and a team in $\{b_p, b_{p+1}, \dots, b_{2p-1}\}$ which is either a game that a_i plays b_{i+1} for $p \le i \le 2p-2$ or a game that a_{2p-1} plays b_p. Therefore, the following $p-2$ rounds of games

$$\left(\begin{matrix} a_1 & \dots & a_{p-2} & a_{p-1} \\ b_2 & \dots & b_{p-1} & b_1 \end{matrix} \right]\left] \begin{matrix} a_p & a_{p+1} & \dots & a_{2p-3} & a_{2p-2} & a_{2p-1} \\ b_{p+2} & b_{p+3} & \dots & b_{2p-1} & b_p & b_{p+1} \end{matrix} \right) \tag{30}$$

obtained by fixing $a_1, a_2, \dots, a_{p-1}$ and $a_p, a_{p+1}, \dots, a_{2p-1}$, and rotating the sequences $b_2, b_3, \dots, b_{p-1}, b_1$ and $b_{p+2}, b_{p+3}, \dots, b_{p+1}$ simultaneously $p-3$ times, can be added to (28) and (29) to form a schedule of the tournament.

We should point out that only rounds in (28) have games between teams within Group A or Group B and if a game that a_i plays a_j in a round in (28), then there is also a game between b_i and b_j for that round. Now we prove that the round-robin algorithm does not generate this schedule formulated above.

Suppose to the contrary that the round-robin algorithm generates this schedule. Without loss of generality, we may assume that a_i is the fixed team in the setting of the round-robin algorithm. In the circular permutation which corresponds to the setting of the round-robin algorithm, if at least two consecutive teams are from Group A, for example, a_j and a_k, then we may assume a subsequence a_j, a_k, b_l in the circular permutation. We consider the round when a_i plays a_k, which is a round in (28) and can be written as follows:

$$\begin{bmatrix} a_i & b_l & x_{4p-6} & \dots & x_{2p-2} \\ a_k & a_j & x_1 & \dots & x_{2p-3} \end{bmatrix}. \tag{31}$$

Because a_j and b_l are in Group A and Group B, respectively, due to a property of the rounds in (28), we obtain that $l = j$. Now we rotate the sequence $a_j, a_k, b_l, x_{4p-6}, \dots, x_1$ clockwise one position and consider the round that a_i plays a_j that is also a round in (28), we have the following games

$$\begin{bmatrix} a_i & a_k & b_l & x_{4p-6} & \dots & x_{2p-2} \\ a_j & x_1 & x_2 & x_3 & \dots & x_{2p-3} \end{bmatrix}. \tag{32}$$

Now if x_1 is in Group B, then $x_1 = b_k$ by (32), which by (31) further implies that $x_{4p-6} = b_i$, which by (32) again implies that $x_3 = b_j$, which is a contradiction since $b_l = b_j$ already appeared in the setting. If x_1 is in Group A, by (31) we know that x_{4p-6} is also in Group A. Now rotating the sequence $x_1, x_2, \ldots, x_{4p-6}, b_l, a_k, a_j$ in (31) counterclockwise one position, we have

$$\begin{bmatrix} a_i & x_{4p-6} & x_{4p-5} & \ldots & x_{2p-3} \\ b_l & a_k & a_j & \ldots & x_{2p-4} \end{bmatrix}. \tag{33}$$

Since x_{4p-6} is in Group A, a game between teams in Group A in the round (33), that is the game that x_{4p-6} plays a_k, shows that the round (33) must be in (28). Therefore, $b_l = b_i$, which shows that $l = i$. However, we already know that $l = j$. We have a contradiction.

Now, suppose there is a setting of the round-robin algorithm that generates a schedule determined by (28)–(30). Then in the circular permutation corresponding to the setting of the round-robin algorithm, teams in Group A should be separated by teams in Group B. Therefore, there exists one and only one subsequence with two consecutive teams from Group B, namely b_{j_1}, b_{j_2}. We now consider a round that is as follows:

$$\begin{bmatrix} a_i & \ldots & a_{i_1} & b_{j_1} & b_{j_2} & a_{i_2} \\ z & \ldots & a_{j_5} & b_{j_4} & a_{i_4} & b_{j_3} \end{bmatrix}. \tag{34}$$

In this round, b_{j_1} plays b_{j_4}. Since only the rounds in (28) have games between teams in Group B, we know that the round (34) must be in (28). However, for all rounds in (28), there is only one game between a team in Group A and a team in Group B and there are two such games in the round (34). We, therefore, get a contradiction.

Therefore, there does not exist a setting of the round-robin algorithm which generates a schedule put together by (28)–(30).

6 Conclusion

We have proved an equality for the number of schedules generated by the round-robin algorithm. As Examples 3 and 4 show, some tournament schedules may not be generated by the round-robin algorithm. It might be interesting to develop a practical approach to find the number of all tournament schedules for n teams. Though chromatic polynomials can be used to describe the total number of schedules for n teams, it is not easy to compute chromatic polynomials even for a small number of teams, for example eight teams. So we further ask if an equality or an inequality exists for the number of all tournament schedules for n teams. It seems to us that this is not an easy problem. Our future work will try to answer this question.

Acknowledgment

The authors would like to thank the referee for the comments that helped improve the paper. Specifically, the comment that motivates us to construct counterexamples in Section 5, is greatly appreciated.

References

[1] Briskorn, D., Combinatorial properties of strength groups in round-robin tournaments, European Journal of Operational Research, 192(2009), 744–754.

[2] Briskorn, D.; Drexl, A., A branch-and-price algorithm for scheduling sport leagues, Journal of the Operational Research Society, 60(2009), 84–93.

[3] Freund, J.F., Round robin mathematics, The American Mathematical Monthly, 63(1956), No. 2 , 112–114.

[4] Harris, J.M., Hirst, J.L., and Mossinghoff, M.J., Combinatorics and Graph Theory, 2nd edition, Springer, 2008.

Alyssa Cherry
University of Missouri
Department of Mathematics
Columbia, MO 65211, USA
e-mail: alyssa.cherry@mail.missouri.edu

Drake Olejniczak
Department of Mathematics
Western Michigan University
Kalamazoo, MI 49008, USA
e-mail: Drake.P.Olejniczak@wmich.edu

Qinghong Zhang (Corresponding author)
Department of Mathematics and Computer Science
Northern Michigan University
Marquette, MI 49855, USA
e-mail: qzhang@nmu.edu