Noah Molder is a double major in Mathematical Sciences and Film & Video Production at the University of Memphis, graduating in the spring of 2024. He has earned the University Honors with Thesis, Honors in Mathematical Sciences, and Undergraduate Research Scholar designations, among other awards and honors. Following graduation, he will stay at the University of Memphis to complete his second year of the ABM program to earn an MS degree. He is incredibly thankful to Dr. Thomas Hagen, under whom he completed this research, for all the wonderful opportunities and advice he has offered.

Noah Molder

The Nontransitive Dice: New Results for a Statistical Paradox in a Game of Chance

Faculty Sponsor
Dr. Thomas Hagen

Abstract

Given three dice, can we place the numbers 1 through 6 on the dice, allowing repetitions, so that after a first player chooses one die of the three, the second player can always choose a die with a higher probability of rolling a greater number from the remaining two? If so, is it possible to optimize the chances of winning? Here, we have discovered the "best" set of so-called nontransitive dice under our conditions and using our specific choice of meaning for "best," which indeed optimizes winning chances. This research project draws on techniques from probability theory, combinatorics, complexity theory, game theory, and scientific computing. The topic falls in the category of nontransitive games, a research area in Mathematics and Economics, and combines theoretical and practical methods. This research was completed in collaboration with Dr. Thomas Hagen and Tyler Owens.

The Game

Imagine you and an opponent are playing a game consisting of three 6-sided dice. To begin, you will place the numbers 1 - 6 on the three dice in any way you like, allowing repetition. For example, you could create a die that is all ones, denoted as (1,1,1,1,1,1), or perhaps just the even numbers, (2,2,4,4,6,6), or even a random mix, maybe (2,3,3,4,4,5).

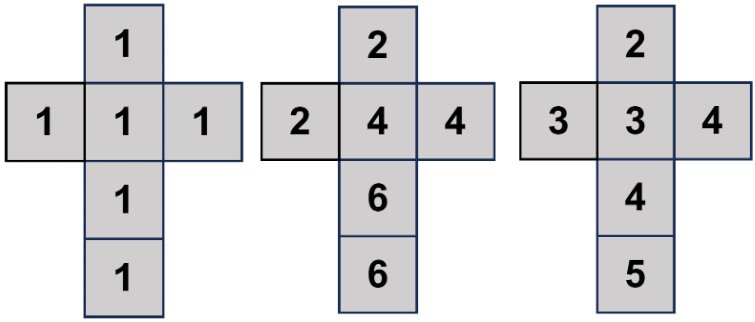

Figure 1. Example Dice Nets

It is important to note here that the placement of these numbers on a die does not matter. For example, the two dice (2,2,3,3,3,6) and (3,6,2,3,2,3) are equivalent in our game. To avoid double counting, we only accept dice whose sides are labeled in non-decreasing order. This means we will only accept the first of these two equivalent dice because the numbers do not decrease from left to right as they do in the second. The four dice we have named so far, as well as the other 458 dice that are possible under these conditions, are perfectly valid for the game. Now that you have created the three dice, your opponent will choose one of them, and you will choose one of the remaining two. You each roll your die, and whoever rolls a higher number is the winner.

The game seems quite simple at first, but let us put some money on it. Let us say that you and your opponent both wager a dollar each time you roll the dice. If you just want to make sure you do not lose, you could make all of them normal 6-sided dice, (1,2,3,4,5,6), and things would remain relatively even. But, if you want to really win, you will have to employ some sort of strategy.

So, what might you try? Perhaps you try to create one die that is stronger than the other two. This could work as long as your opponent does not choose the die that is obviously better, which would be unusual. So, perhaps you make two dice that are strong and leave one to be weak. Then you and your opponent both have strong dice, and things are back to being equal again. You may begin to wonder if there is any way you can guarantee that, over time, you will win big or if this all must be left to chance. As it turns out, you can ensure that you will win, no matter the choice of your opponent, thanks to a paradox of probability known as "nontransitive dice".

Exploring Nontransitivity

Before we take a look at nontransitivity and nontransitive dice, we should first understand transitivity. Transitivity is present in a majority of the relations we see day-to-day, which means it is very intuitive and, by association, is the reason that nontransitivity tends to be counterintuitive. As an example, let us take three people named Adam, Bob, and Charlie, and look at their heights. When we stand Adam next to Bob, we see that Adam is taller. When we stand Charlie next to Bob, we see that Bob is taller. By taking just these two comparisons, we know who of Adam and Charlie will be taller without ever standing them next to each other. Adam is taller than Bob who is taller than Charlie. So, Adam must be taller than Charlie. This is true because "taller than" is what is called in mathematics a 'transitive relation'. If "taller than" were nontransitive, we would not be able to say anything about Adam and Charlie solely through their comparisons with Bob. You may now be thinking, well then, is not every relation transitive? They are not, but to see an example, we will have to do a bit of abstraction.

Considering that we are working with a game of dice, try to think of some sort of game that could be nontransitive. Perhaps a sport comes to mind. Let us take three baseball teams: the Angels, the Braves, and the Cubs. They play three games: the Angels beat the Braves in the first, and the Braves beat the Cubs in the second. We will use the symbol "$\succ$" to denote that one team beats another; that is to say that "$i \succ j$" will mean that i beats j. Referring to each team by the first letter of its name, the results of the first two games will look like $A \succ B$ and $B \succ C$. If this relationship were transitive then the Angels would certainly beat the Cubs in the third game, but this is not necessarily the case. $A \succ B \succ C$ may be true, but it does not have to be.

But, there are many variables involved in a sports game. So, let us try something a bit more concrete and familiar. How about a game of Rock, Paper, Scissors? As we all know, rock beats scissors, $R \succ S$, and scissors beats paper, $S \succ P$. If transitivity were true for this relation, we would have $R \succ S \succ P$, but that is not the case (if it were, you could never lose by choosing rock every round). $P \succ R$ is true, meaning that Rock, Paper, Scissors is an example of a nontransitive game. The winning results in the game form a cyclical relationship, which we refer to as a nontransitive loop. This is the relationship we want to replicate with our dice.

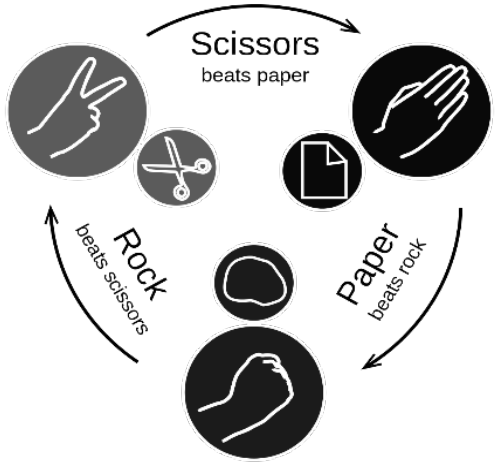

Figure 2. Rock, Paper, Scissors Nontransitive Loop

From Enzoklop, [1]

Nontransitive Dice

While our dice game can be modeled closely after Rock, Paper, Scissors, there is one crucial difference. In Rock, Paper, Scissors, you and your opponent must make your choices at the same time, but in our dice game, this is not how we play. Recall that your opponent will select a die first, and then, knowing what they have chosen, you make your choice. Imagine knowing that your opponent has chosen rock and then getting to pick second. It is up to you to choose paper (in that example), but knowing your opponent's choice should ensure that you never lose. In our game, you will know your opponent's choice. So, as long as you choose your die correctly, you will always come out on top.

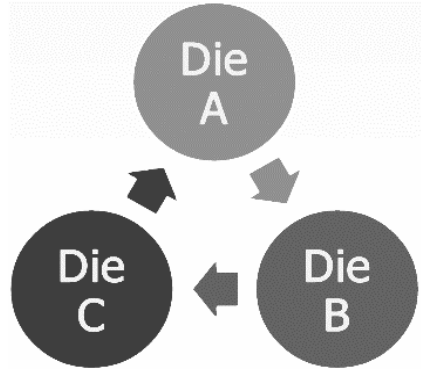

Figure 3. Three Dice Nontransitive Loop

With this in mind, we can try to create a set of nontransitive dice. Through the simple but lengthy process of trial and error, the set that I found when the problem was first posed to me contains dice represented by: A - (2,2,3,5,5,6), B - (2,3,4,4,5,5), and C - (3,3,3,4,5,5).

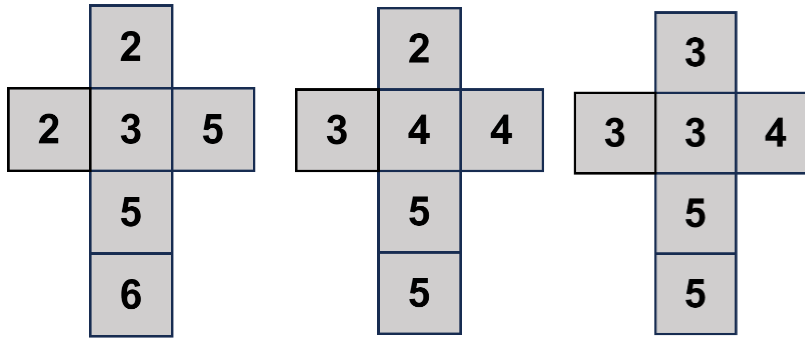

Figure 4. Nets of Nontransitive Dice Set

To check that these three dice truly form a nontransitive set, we have to compare each die with every other die. In each comparison, since there are 6 ways one die could land and 6 ways the other die could land, we have 6 x 6 = 36 possible outcomes. We will check what happens in each one of these cases, as in, we will compare one face of a die with each face of another die to see when it wins (when its number is greater than the number on the other die) and then repeat with each face of the first die. The number

of times a face beats a face of the other die will be that face's score, and we will add all of these scores together to get the die's total score. Score here is the same as the number of times out of 36 possibilities that a die will roll a higher number; thus, when we have compared two dice with each other, the one with the higher score will be declared the winner.

In general, the process of comparing two dice is as follows: Consider dice A and B, A with sides a_i and B with sides b_j where $1 \leq i \leq 6$ and $1 \leq j \leq 6$. Compare each a_i with each b_j. If $a_i > b_j$: Score(A) increases by 1. If $a_i < b_j$: Score(B) increases by 1. If $a_i = b_j$: no score increases. Then, compare Scores. If Score(A) > Score(B): A wins. If Score(A) < Score(B): B wins. If Score(A) = Score(B), neither wins. It is also true that if Score(A) > 18 or Score(B) > 18: that die will win (since $18 = \frac{36}{2}$, meaning the die wins in over half the cases).

Looking back at my dice set, let us begin with checking A and B. If a 2 is rolled on A, it cannot beat B no matter what happens since there are no 1s on B. The same is true for both 2s on A, so these faces earn scores of 0. If A rolls a 3, it wins in the one case that B rolls a 2, so it receives a score of 1. The two 5s on A beat the 2, 3, and 4s on B, so they get scores of 4. Lastly, since there are no 6s on B, the 6 on A beats all six faces of B, so it gets a score of 6. Adding these scores together: 0+0+1+4+4+6=15, so in the match of A vs. B, A scores a 15.

Now let us see how B stands against A. B's 2 face scores 0, the 3 face scores 2, the 4 faces score 3, and the 5 faces score 3. So, 0+2+3+3+3+3=14, which means B's score is one less than A's score in A vs. B. So, $A \succ B$. Now we have to check B vs. C and A vs. C. The scores from here on will be left for the reader to verify. In B vs. C, B scores a 14 and C scores a 13, so $B \succ C$. In A vs. C, A scores a 14 and C scores a 15, so $C \succ A$. Let us assume this set is transitive and see if it holds. Taking the first two results, $A \succ B$ and $B \succ C$, we would be able to say that $A \succ B \succ C$ which would mean $A \succ C$, but we know that $C \succ A$, so this set is nontransitive.

Another set of nontransitive dice, found by Tyler Owens, consists of A - (1,1,4,4,5,5), B - (2,2,3,3,3,4), and C - (1,2,2,2,6,6). Our verification process for this set will be exactly the same. In A vs. B, A scores a 20 and B scores a 12, so $A \succ B$. In B vs. C, B scores an 18 and C scores a 12, so $B \succ C$. In A vs. C, A scores a 16 and C scores an 18, so $C \succ A$. Again we see that $A \succ B$, $B \succ C$, but $C \succ A$. So, this set is, in fact, nontransitive. While these sets are both nontransitive, they are not equal in all regards. You may notice, the scores in each match are much closer in my set than

in the Owens set. The difference between the two scores in a given match will be called the "win difference". All of the win differences in my set are 1, while the Owens set has win differences of 8, 6, and 2. In theory, playing with either set will result in you winning over time, but my set does so at a far slower rate.

When looking in terms of these win differences, it is clear that the Owens set is better than my set. To say that one set is better than another – and assuming that there are more sets of nontransitive dice under our conditions than the two we have found – raises the question: is there a best solution, and, if so, what is it?

Searching for the Best Solution

To begin, we must note that there is no one way to define a "best solution," so we need to decide what it is exactly that we are looking for. We chose to look for the solution with the highest sum of win differences as our "best solution". Now, with that in mind, the simplest route forward is to check all the solutions and choose the best one. To find all of the solutions, we will check all of the possible combinations of dice. This will certainly require the aid of a computer program, but even then, there are not unlimited capabilities. We first checked the number of combinations we would have to deal with. Given our conditions, each of the 6 faces of a die can have any one of 6 numbers on it. This means there are $6^6 = 466{,}566$ ways to create one die. Since our game has 3 dice, we compare all possible sets of 3 dice. This makes a grand total of $(6^6)^3 = 101{,}959{,}596{,}668{,}416$ possible games that we would have to check. That is over 100 trillion combinations, which is far too much to run even via program, so, we found a way to simplify.

When using to calculate the number of dice, we indeed counted them all, but we also counted every permutation of every die. This means that the valid die called (1,1,1,1,1,2) has also been counted under the names (1,1,1,1,2,1), (1,1,1,2,1,1), (1,1,2,1,1,1), (1,2,1,1,1,1), and (2,1,1,1,1,1).

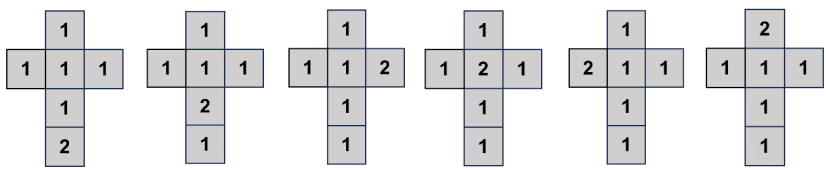

Figure 5. Nets of Set of Die Permutations

Recall that only one of these dice, the one with labels in nondecreasing order, is valid. So, we need another way to construct these dice that includes the ones we want to consider and does not include these extraneous permutations.

We can start by setting the first number on a die, denoting the nth number on the die as D(n). When the first number, D(1), is set, we have limited possibilities for the second number, D(2); it must be greater than or equal to D(1). In general, D(n+1) must always be greater than or equal to D(n). Let us use a die that starts with a 4, for example. For D(1) = 4, D(2) could be 4, 5, or 6. If D(2) = 4, then again the next number, D(3), could be 4, 5, or 6, but if D(2) = 5, then D (3) can only be 5 or 6, and if D(2) = 6 then D(3) must be 6. So now, deciding just the first three sides of a die with D(1) = 4, we have 6 possibilities. If we continue this process to its conclusion with D(6), we will see that there are 21 possible dice where D(1) = 4. We must also execute this process for D(1) = 1,2,3,5,6, and then we will add all of these possibilities together. When we do this, we produce 462 total possible valid dice. This means there are only $462^3 = 98{,}611{,}128$ combinations to check. This number (just under 100 million), while still very large, is much more manageable.

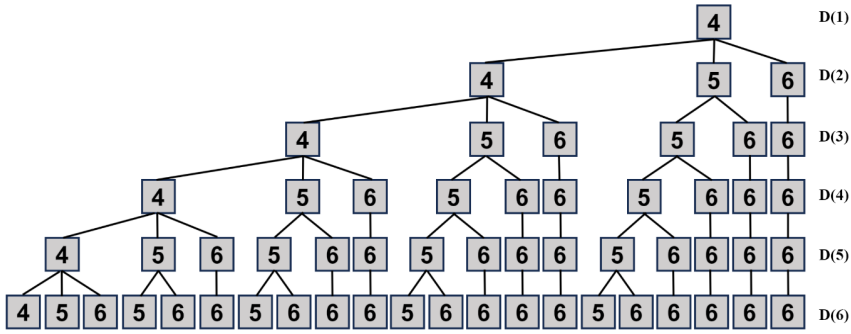

Figure 6. Branch Diagram for Dice with D(1) = 4

Next, we needed a strategy for computing the best solution. The first step in writing the program, which can be found in the appendix, was to create the dice. While the way we have named the dice is easy for us to understand, it is not as easy or efficient to program them in the same way. So, instead of naming the dice by the numbers that appear on the faces of the die, we name them by the frequency which each number appears. Because there are 6 numbers that can appear on these dice, each die will become

an array of 6 components, each representing a different number 1 - 6. For example, we will take a die from my set, the one named (2,2,3,5,5,6), and represent it in our new array form. We can start with a blank array (notice the bracket notation instead of parentheses): [-,-,-,-,-,-]. The first component will represent the frequency of ones. This particular die has 0 ones, so we will fill that spot in with a 0: [0,-,-,-,-,-]. For the second component, we note that the die has 2 twos, so we fill in: [0,2,-,-,-,-]. Then 1 three: [0,2,1,-,-,-], and so on until we get the completed array, which in this case is [0,2,1,0,2,1]. Because we are using dice with 6 sides, any die that we rename in this new array form will have the total of its components equal to 6. This allows for much simpler programming.

We wrote the program using a series of "for" loops to give us the dice. Essentially, we loop through the frequencies of each number from 1 to 6, which will result in all of the dice forming in a specific order, which will be useful later. For Die 1, we start with a 6 in the sixes place: [0,0,0,0,0,6], or (6,6,6,6,6,6). Then for Die 2, we have one less six and add a five: [0,0,0,0,1,5], or (5,6,6,6,6,6), followed by having 2 to 6 fives: [0,0,0,0,2,4], [0,0,0,0,3,3], [0,0,0,0,4,2], [0,0,0,0,5,1] and [0,0,0,0,6,0]. Then we add 1 four, beginning with [0,0,0,1,0,5] and looping through the different numbers of fives and sixes until we end with [0,0,0,1,5,0]. We then do the same for 2 to 6 fours and then all possibilities of numbers in this fashion, ending on Die 462: [6,0,0,0,0,0], or (1,1,1,1,1,1). This process outputs a matrix we named "DiceFreq", which is a 462-row matrix (one row for each die) with 6 columns for the different frequencies. Each die array is now a row in this matrix.

We can then use DiceFreq to create a matrix of win differences. It is, in this case, a 462 x 462 matrix where each die is compared with each die, including itself, and the net wins, or win difference, is stored in that position of the matrix. That is to say, in position (i, j) of the matrix (row i and column j), the entry, $a_{i,j}$, is net wins of Die i over Die j. We do this by multiplying the number of sixes on Die i by the number of faces less than Six on Die j (because each six will win in each of these cases). We add this to the number of fives on i times the number of faces less than 5 on j, and so on until we have added the 5 products (we can ignore ones since they never win). This is why it is so beneficial to store the dice by their frequencies as we did in DiceFreq. To do these win difference calculations, we need to know the frequency which each number appears, which is exactly what we have. DiceFreq(m, n) is the number of faces labeled

"n" on Die m. For example, DiceFreq(289,4) is the number of fours on Die 289. So, when comparing Die 289 with Die 46, we have the fours step in calculating $a_{289,46}$:

DiceFreq(289,4)(DiceFreq(46,3)+DiceFreq(46,2)+DiceFreq(46,1)).

This serves mostly as an intermediary step towards our solution, but there are a couple of interesting things to note. When $i = j$, also known as the entries on the diagonal of the matrix, $a_{i,j} = 0$. This is because we will be comparing a die with itself when $i = j$, and no die will have any advantage when rolled against itself. It is also worth noting that this matrix is skew-symmetric. This means that, $a_{i,j} = -a_{j,i}$, or for example, $a_{64,351} = -a_{351,64}$. This occurs because, in these cases, we are comparing the same two dice; we are just counting the net wins from different perspectives. Taking any dice A and B, if A beats B 12 times, then it must be the case that B beats A -12 times; the negative is also interpretable as denoting net losses.

The final step is writing code that will analyze this win matrix and tell us where any 3 dice – A, B, & C – have the relationship that we are looking for: $A \succ B$, $B \succ C$, $C \succ A$. This will show us all solutions to our problem. From there, we can narrow the results further to find whether there is a best solution.

Results

We learned several things from running this program. They are as follows: Using only 3 different numbers on the dice (or 1 or 2 different numbers, for that matter, but these are trivial) is not enough to create a set of nontransitive dice. 4 different numbers are the minimum required. Out of the nearly 100 million possible sets, only 121,998 sets are nontransitive. Because of the way we wrote our program, each set appeared in our results 3 times under different permutations, so, as before, we eliminate these extras. Dividing the total by three gives the true number of different sets of nontransitive dice, leaving us with 40,666. Now we look at the win differences of these sets to determine the best. We found that three sets contained the highest minimum win difference, meaning the least of their three win differences was higher than the least win difference in any other set, which was 6. Two of these sets: {(3,3,3,3,4,6), (2,2,2,5,5,5), (1,4,4,4,4,4)} & {(1,3,4,4,4,4), (3,3,3,3,3,6), (2,2,2,5,5,5)}, have win differences summing to 21. The remaining set has the greatest sum of win differences, 26, meaning that, by our definition of "best," this will be the singular, best solution.

The three dice in the best set are A - (3,3,3,3,3,6), B - (2,2,2,5,5,5), & C - (1,4,4,4,4,4).

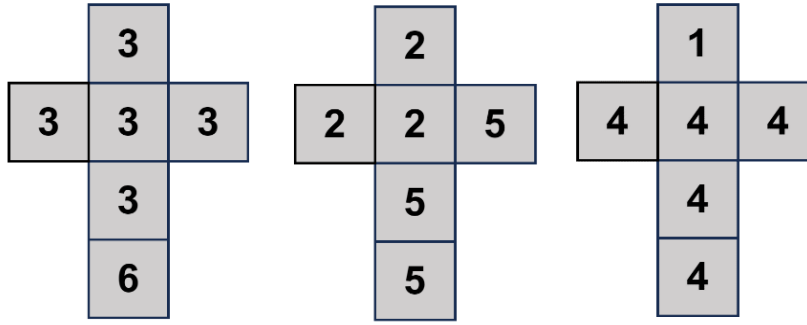

Figure 7. Nets of Best Set of Nontransitive Dice

Upon a quick glance at the dice in this set, you will notice a striking symmetry. Picturing the faces as dots above a number line, if we reflect A over a line at 3.5, we get C (and vice versa). B itself has perfect symmetry when reflected over this line at 3.5; it is identical to its own reflection. These symmetries were unexpected, but they are intriguing to explore.

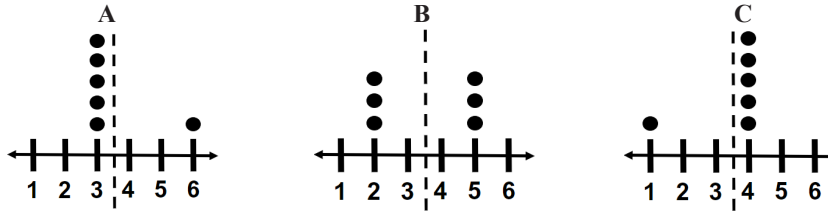

Figure 8. Number Line Constructions of Best Set

Looking at the matchups of this set, A ≻ B and B ≻ C both with win differences of 6, and C ≻ A with a win difference of 14. These win differences, particularly in the last case, are far greater than those in the original two sets that we discovered. With this set, if your opponent is unwise enough to choose Die A and you make the right choice of Die C, you will win more than twice as often as them. Even if they make the better choice of Die B or Die C, you will still have all their money in no time.

Additional Research

Much research has been done on this topic, exploring many related questions. Some examples are: “When can these sets of nontransitive dice exist?”, “How large can the sets be?”, and “Can you create a game for more than two players?” More information and answers to these questions can be found in [2], [3] & [4], and [5], respectively.

References

[1] Enzoklop.https://commons.wikimedia.org/wiki/File:Rock-paper-scissors.svg

[2] Schaefer, A., & Schweig, J. (2017). Balanced nontransitive dice. *The College Mathematics Journal*, *48*(1), 10–16. https://doi.org/10.4169/college.math.j.48.1.10

[3] Gardner, M. (1970). Mathematical games. *Scientific American*, *223*(6), 110–115. http://www.jstor.org/stable/24927686

[4] Angel, L., & Davis, M. (2017). A direct construction of nontransitive dice sets. *Journal of Combinatorial Designs*, *25*(11), 523–529. https://doi.org/10.1002/jcd.21563

[5] Grime, J. (2017). The bizarre world of nontransitive dice: Games for two or more players. *The College Mathematics Journal*, *48*(1), 2–9. https://doi.org/10.4169/college.math.j.48.1.2

Appendix

```
% Create all the dice by the frequency of their labels
DiceFreq=zeros(462,6);
counter=0;
for ones=0:6
     for twos=0:(6-ones)
          for threes=0:(6-ones-twos)
               for fours=0:(6-ones-twos-threes)
                    for fives=0:(6-ones-twos-threes-fours)
                         counter=counter+1;
                         DiceFreq(counter,1)=ones;
                         DiceFreq(counter,2)=twos;
                         DiceFreq(counter,3)=threes;
                         DiceFreq(counter,4)=fours;
                         DiceFreq(counter,5)=fives;
                         DiceFreq(counter,6)=6-fives-fours-threes-twos-ones;
                    end
               end
          end
     end
end
% Compare all dice with matrix of Win Differences
W=zeros(462,462);
for i=1:462
     for j=(i+1):462
W(i,j)=DiceFreq(i,2)*DiceFreq(j,1)+DiceFreq(i,3)*(DiceFreq(j,2)+DiceFreq(j,1))+Dice
Freq(i,4)*(DiceFreq(j,3)+DiceFreq(j,2)+DiceFreq(j,1))+DiceFreq(i,6)*(DiceFreq(j,5)+
DiceFreq(j,4)+DiceFreq(j,3)+DiceFreq(j,2)+DiceFreq(j,1))+DiceFreq(i,5)*(DiceFreq(j,
4)+DiceFreq(j,3)+DiceFreq(j,2)+DiceFreq(j,1)) -
(DiceFreq(j,2)*DiceFreq(i,1)+DiceFreq(j,3)*(DiceFreq(i,2)+DiceFreq(i,1))+DiceFreq(j
,4)*(DiceFreq(i,3)+DiceFreq(i,2)+DiceFreq(i,1))+DiceFreq(j,6)*(DiceFreq(i,5)+DiceFr
eq(i,4)+DiceFreq(i,3)+DiceFreq(i,2)+DiceFreq(i,1))+DiceFreq(j,5)*(DiceFreq(i,4)+Dic
eFreq(i,3)+DiceFreq(i,2)+DiceFreq(i,1)));
W(j,i)=-W(i,j);
     end
end
% Find all Nontransitive sets
counter=0;
for i=1:462
     for j=1:462
          for k=1:462
               if (W(i,j)>0) && (W(j,k)>0) && (W(k,i)>0)
                    counter=counter+1;
                    Result(counter,:)=[i,j,k];
               end
           end
      end
  end
  % Calculate all minimum Win Differences
  for index=1:counter
       i=Result(index,1);
       j=Result(index,2);
       k=Result(index,3);
       WinDist(index)=min([W(i,j),W(j,k),W(k,i)]);
  end
  MaxDist=max(WinDist);
  WinLoop=find(WinDist==MaxDist);
  Result(WinLoop,:)
```