Winning Strategies in Non-Transitive Games: Applying Graph Theory and Combinatorics to investigate winning strategies.

Cameron Rutherford
MA 430W Paper

Abstract

This paper explores the strategy behind playing Penney's game with more than two players. Penney's game in its simplest form involves two players, A and B, who bet on a pattern. A player wins when their pattern occurs in a randomly generated sequence (by a dice or coin for example). When considering the overlapping words paradox, one strategy seems to have a clear advantage over another in a given situation. There is much established research in this area, with proof that the game does not express transitive properties given various game parameters (such as an alphabet of two, and a pattern length of 3). This paper will attempt to analyze the problem through graph theory and combinatorics in order to gain an intuition as to why some strategies are so much better than others. This paper will also investigate how the strategies perform when the game is extended to include more than two players, with the investigated format being a single elimination tournament.

Contents

1 Penney's Game

1.1 Introduction to the N = 3 situation

Penney's game is a probability game that deals with sequences of coin flips discovered by Walter Penney in 1969 [10]. The game is played by two players (Player A and Player B) selecting a unique sequence of heads and tails of length N, flipping the coin repeatedly, and waiting until one of the sequences that the players selected appears. Consider an example game with $N = 3$ where player A selects the sequence THH (tails-heads-heads), and player B selects HHH (heads-heads-heads). The sequence of coin flips observed is as follows: THTTHH... After the final H that appears in the sequence is seen, player A would be declared the winner.

While this initial situation may seem trivial, when examining the odds of each player winning some interesting behaviour emerges. Considering the same two strategies above for Players A and B, given that the coin being flipped is fair, it would be assumed that the game itself would also be fair. However, instead of there being a roughly 1/2 chance of each player winning, the odds that player A beats player B in this situation is actually 7/8. In this following subsections we will examine why this is the case, and try to analyze which strategies are the most effective.

1.2 Calculating the odds

1.2.1 Probabilistic Approach

For ease of reference we will continue using strategies A (THH) and B (HHH) for our examples. To calculate the probability of player A winning, let us first examine the two strategies a little more closely. It turns out that in this situation, as soon as a tail is flipped, B no longer has any chance of winning the game. Thus, for B to win all three of the first three flips must all be heads. The inverse of this would then give us the probability that A will win:

$$P(\text{A wins}) = 1 - P(\text{HHH before THH}) = 1 - \left(\frac{1}{2}\right)^3 = \frac{7}{8}$$

A similar analytic approach can be used to calculate other odds of different strategies competing against each other, such as the odds of TTH winning against HTH. Instead of manually calculating each of these odds, instead let's look at a different approach in calculating the winning odds using an algorithm developed by John Conway.

1.2.2 Conway's Algorithm

To understand exactly how Conway's Algorithm works, we will simply apply the algorithm to our current working strategies of A (THH) and B (HHH) for players A and B respectively. Before calculating the odds, the first step that must be performed is a calculation of different Conway Numbers for the patterns. Conway Numbers, named after their inventor, are indicators of the amount of repetition of a given pattern in relation to another preceding pattern. For example the Conway Number XY (for pattern X against pattern Y), would represent the degree of overlap of pattern X with regards to another pattern Y. From now on we shall use notation of the following form: C_{XY} would be the Conway Number XY, and so C_{AB} would be the Conway Number AB etc.

Calculating C_{BB}: Table 1 has the calculated digits of C_{BB} in the first row, with column [0], [1] and [2]'s data coming from rows [0], [1] and [2] respectively. A_0 denotes the pattern A in the first position of the Conway number C_{BB}, and B_1 the second.

Calculating $C_{BB}[0]$: Take B_0 and align the first entry in the sequence with that of B_1 (as shown in rows B_0 and $B_1[0]$). If these two sequences match, then you place a 1 in location $C_{BB}[0]$, 0 if they do not match. So there is a 1 in $C_{BB}[0]$ as ***HHH*** matches with ***HHH***.

	$C_{BB}[0]$	$C_{BB}[1]$	$C_{BB}[2]$
C_{BB}	1	1	1
B_0	***H***	***H***	***H***
$B_1[0]$	***H***	***H***	***H***
$B_1[1]$	–	***H***	***H***
$B_1[2]$	–	–	***H***

Table 1: Table calculating C_{BB}

Calculating $C_{BB}[1]$: Take B_1 and align the first entry in that sequence with the second entry of B_0 (as shown in rows B_0 and $B_1[1]$). If the remaining sequences still match (ignoring the first "hanging" entry of B_0), then you place a 1 in location $C_{BB}[0]$, 0 if they do not match. So there is a 1 in $C_{BB}[1]$ as (H)***HH*** matches with (–)***HH***

Calculating $C_{BB}[2]$: Take B_1 and align the first entry in that sequence with the second entry of A_0 (as shown in rows B_0 and $B_1[1]$). If the remaining sequences still match (ignoring the first "hanging" entry of B_0), then you place a 1 in location $C_{BB}[0]$, 0 if they do not match. So there is a 1 in $C_{BB}[1]$ as (HH)***H*** matches with (—-)***H***

Since Conway numbers are decimal numbers that have been derived as binary numbers, the resulting Conway number is $C_{BB} = 111 = 7$. However C_{BB} just one of four Conway numbers ($C_{AA}, C_{AB}, C_{BA}, C_{BB}$) needed to input into Conway's formula:

C_{AA}	=	1	0	0	=	4	C_{AB}	=	0	1	1	=	3
A	–	T	H	H			A	–	T	H	H		
A	–	T	H	H			B	–	H	H	H		

C_{BB}	=	1	1	1	=	7	C_{BA}	=	0	0	0	=	0
B	–	H	H	H			B	–	H	H	H		
B	–	H	H	H			A	–	T	H	H		

Table 2: All the needed Conway numbers: $C_{AA} = 4, C_{AB} = 3, C_{BA} = 3, C_{BB} = 7$

Calculating the Probability of A beating B ($P(A,B)$): Table 2 now gives us all the Conway values $C_{AA} = 4, C_{AB} = 3, C_{BA} = 0, C_{BB} = 7$, which can be put into the following equation:

$$P(A,B) = \frac{C_{BB} - C_{BA}}{C_{AA} - C_{AB}} = \frac{7-0}{4-3} = 7$$

.

Therefore $P(A,B) = 7$, meaning A will win with 7 to 1 odds. i.e. The probability of A winning against strategy B is $7/8$. Not only is Conway's Algorithm excellent for generating all the probabilities of all the different match ups as in Table 3, but it can also be extended to larger N values if needed. We can also take all of the best probabilities for each pattern and create a table of the most effective pattern B against any given pattern A with the associated probabilities (Table 4).

From tables 3 and 4, we can see that some strategies are clearly outmatched by others. Strategy HHH and TTT for example never do better than a 1/2 chance when coming up against the other strategies. On the other hand both HTT and THH only have a less than 1/2 chance of winning against one other strategy.

		A							
		HHH	HHT	HTH	HTT	THH	THT	TTH	TTT
B	HHH		1/2	2/5	2/5	1/8	5/12	3/10	1/2
	HHT	1/2		2/3	2/3	1/4	5/8	1/2	7/10
	HTH	3/5	1/3		1/2	1/2	1/2	3/8	7/12
	HTT	3/5	1/3	1/2		1/2	1/2	3/4	7/8
	THH	7/8	3/4	1/2	1/2		1/2	1/3	3/5
	THT	7/12	3/8	1/2	1/2	1/2		1/3	3/5
	TTH	7/10	1/2	5/8	1/4	2/3	2/3		1/2
	TTT	1/2	3/10	5/12	1/8	2/5	2/5	1/2	

Table 3: Probabilities of winning with B against A

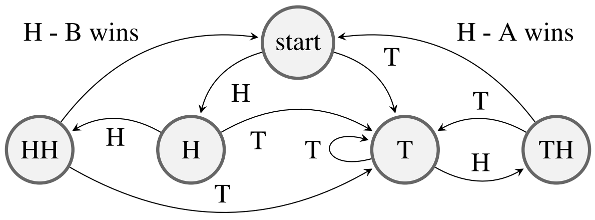
Pattern A	Pattern B	Probability that B Wins
HHH	THH	7/8
HHT	THH	3/4
HTH	HHT	2/3
HTT	HHT	2/3
THH	TTH	2/3
THT	TTH	2/3
TTH	HTT	3/4
TTT	HTT	7/8

Table 4: Best pattern (B) to chose against a given pattern (A) and the winning odds

1.3 Application of Graph Theory and Combinatorics

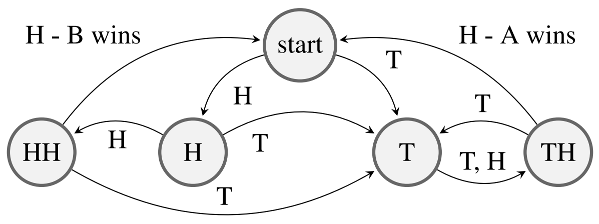
When first investigating this phenomena of lop-sided match ups, it can be hard to develop an intuition as to why one strategy is so much better than another one. By approaching this problem through the lens of Graph Theory and Combinatorics, we can hopefully gain some insights as to why some strategies are so much better than others.

Start by drawing a directed graph where each node represents the current game state, with edges connecting each node based on the ability to get from one game state to another via a heads or a tails. This graph will represent the transitions between game states after each coin flip.

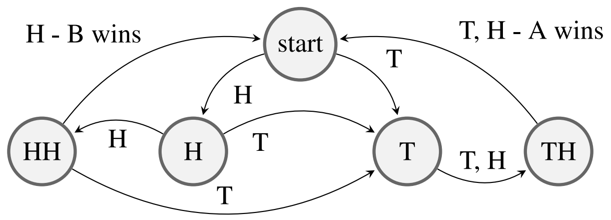


For example, by using this graph we can simulate a game being played between our two favorite patterns A (THH) and B (HHH). Starting at node start, we can simulate either a H or a T flip by using the corresponding edge to travel to the next node. We can then keep on traversing the graph (and in effect traversing through all of the possible game states) until we arrive back at the starting node. Depending on how we arrived back at the starting node, we can determine which pattern won the game.

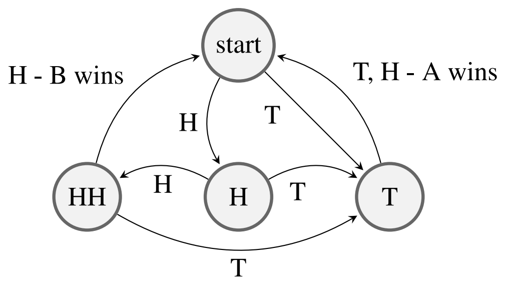
Using the graph to understand the probabilities: The aim of the following steps is to simplify the given graph as much as possible. By doing so we can arrive at a simpler representation of the game, and hopefully get an intuition as to which strategy is most likely to win. First, we must remove any immediate loops in the graph, leaving only one path out of the node with the removed loop. For example, we shall remove the edge with a T label looping into T and instead re-label the edge into TH as T,H to represent the removed loop.



Now that we have eliminated all the immediate loops, we need to repeat the process with any less obvious loops. The example in this graph would be removing the edge with a T label from TH to T, and instead labeling the edge from TH to start accordingly. On different graphs you may need to perform this multiple times until the desired result is achieved.



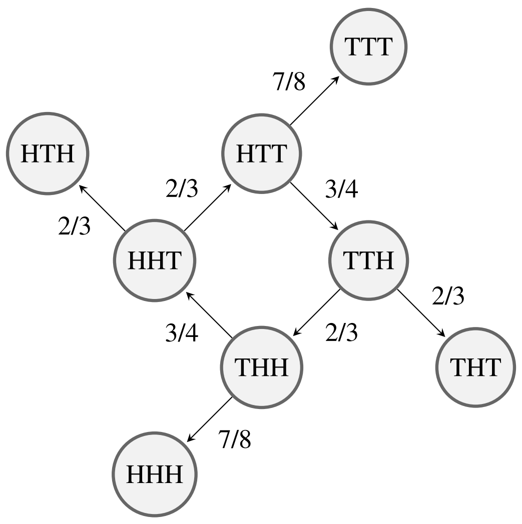
Finally you should collapse all paths that go through extra nodes. In this example once you are at node TH, you are forced to travel to the start. As such, we can remove node TH from the graph completely, and connect node T itself to the starting node.



Now that we have a simplified representation of the game where strategy A (THH) and B (HHH) are competing against each other, it is easier to see why A has the significant advantage. The only way that B can win is if HHH are the first three flips of the coin - something that is easily missed in the starting graph. Whilst this approach is less concrete at calculating exact odds of winning, it is very useful in developing an intuition about the problem.

1.4 Elementary Analysis of Best Strategy

Now that we have established the various probabilities of different patterns competing against others, we can perform an elementary analysis of what we think the best strategy might be. Without establishing exactly how these patterns will be competing against each other, let's investigate the relationship between all the patterns, and the other patterns that they are most effective against. A graph with nodes representing patterns, and edges representing the strategies the pattern is most effective against will serve as a good starting point, and can be produced using Table 5.



Based on this graph, it would seem as though HHH and TTT are not the strongest patterns, with THH and HTT being the dominant two strategies compared to the two in the cycle of length four in the middle. We could also investigate the expected value of each strategy against all the other strategies using the data in Table 3.

Pattern	Average Probability	Percentage
HHH	317/840	37.74%
HHT	67/120	55.83%
HTH	407/840	48.45%
HTT	487/840	57.98%
THH	487/840	57.98%
THT	407/840	48.45%
TTH	67/120	55.83%
TTT	317/840	37.74%

Table 5: The Average Probability of a Pattern Winning against all other Patterns

Table 5 gives us the averaged summation of all the probabilities that the pattern will win against every other pattern individually. The data then shows us that THH and HTT are indeed the most statistically likely teams to succeed with an average 58% chance of winning. However these numbers are based on a system where every pattern competes against ever other pattern about once, and is a fine place to leave the elementary analysis at.

In the next sections we will examine different methods of having the patterns compete against each other, and see if we can get some other, interesting probabilities of winning that differ from Table 5. By analyzing what patterns are more likely to win in different systems (such as a single elimination bracket), we could also gain insight into the "fairness" of these systems by cross checking the results of those analysis against Table 5. If some teams are misrepresented heavily, and the results are even more skewed then it could be stated that the systems are just making the situation more unfair. On the other hand, if the system normalizes the probabilities then maybe it is a better system overall.

2 Extending the game to include more than two players

2.1 Why Brackets?

When extending a two player game such as chess or Penney's game to include more and more players, it often becomes difficult to find the best format to do so in. Some definitions from [7] that could be used to qualify a competition as the "best" would be:

- Highest probability of selecting the strongest contestant.
- Highest probability of the two strongest contestants meeting in the final.
- Fewest rounds needed to select a winner
- etc.

These definitions however already seem to present some issues for the situation at hand. To start with we are not completely sure what the best strategy is in the first place, and from table 5 we can see that two patterns are tied for best making things even more difficult. This, in addition to the non-transitive nature of the game (that being that if A beats B, and B beats C, A does not necessarily beat C), means that making a competition that is the best for this situation seemingly impossible.

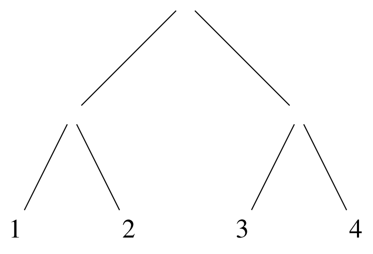
As such, it makes the most sense to start with the simplest situation first. Whilst a round robin could be considered simpler than a single elimination tournament, round robins are rarely used in reality due to the cost of playing real matches. In a round robin tournament of t teams, the number of matches played is $\binom{t}{2}$, where as in a single elimination tournament only $(t-1)$ games are played. In addition to the number of games decreasing in a single elimination tournament, this format also offers more excitement in each game. Especially in this situation where some strategies are simply much worse off against others, a single elimination tournament would hopefully give the worse strategies more of a hope of winning, whereas in a round robin the higher number of games may result in the odds being stacked against some patterns a little too much.

Double elimination tournaments could also be considered as a potential format for competition. Double elimination would probably make the tournament "better" in the sense that the better teams are more likely to come out on top, however more games are needed to be played. In addition to this it would further hinder the chances of the less effective strategies, and as such it will not be considered in this paper.

2.2 Analyzing Brackets For the Best Strategy

Now that single elimination brackets have been chosen as the format for competition, we can examine how the different strategies fare in the format. Before we examine the situation where all 8 strategies compete in the bracket, instead we shall examine a simpler situation and determine the likelihood of all the teams winning the tournament.

Four team bracket: For this example we will label the contestants 1, 2, 3 and 4, with all contestants having a 1/2 chance of beating the others. An example bracket:



Whilst one could simply say that each team has a 1/4 chance of winning the tournament without doing any analysis, it will be good to understand exactly why this is the case before trying to extend the analysis to more complicated situations.

Odds of each team winning: First we need to determine the probability that each of the four teams makes it out of the first round. This is easy enough as each team has a 1/2 chance of defeating their first opponents. Now to determine the probability that a team wins in the second round is slightly more difficult. To do this, we need to determine what the possible opponents are for said team in the second round, the probabilities that these opponents will actually make it to the second round, and the probabilities that the team actually beats these opponents in the second round. Consider this example of calculating the odds of 1 winning in various rounds:

Odds of 1 winning in the first round: Since 1 only has one possible opponent in the first round, and 1 is also guaranteed to play in the first round:

$$P(1 \text{ wins in round } 1) = P(1 \text{ beats } 2) = 1/2$$

Odds of 1 winning in the second round: There are two possible opponents for 1 in the second round (3 and 4), and both opponents have a 1/2 chance of making it into the second round. From our previous calculations we also know that 1 has a 1/2 chance of making it into the second round:

$$\begin{aligned} P(1 \text{ wins in round } 2) &= P(1 \text{ wins in round } 1) * [P(3 \text{ wins in round } 1) \\ &\qquad * P(1 \text{ beats } 3) + P(4 \text{ wins in round } 1) * P(1 \text{ beats } 4)] \\ &= 1/2[1/2 * 1/2 + 1/2 * 1/2] \\ &= 1/4 \end{aligned}$$

A similar pattern can be applied to bigger and bigger tournaments, with the calculations becoming more and more cumbersome to do by hand as the tournaments start becoming bigger and as the probabilities stop being so nice to calculate. Because of this difficulty in calculation, I have written some code in python (see listing 1) that will calculate the odds of each team winning the tournament in any given tournament arrangement. In addition to this, the code records the minimum and maximum odds of a team winning a tournament over the course of all possible tournament arrangements, as well as keeping track of which strategies had the best chance of winning each given tournament.

Pattern	Lowest Odds	Highest Odds	Tournaments "won"
HHH	7.63%	1.90%	0
HHT	23.17%	8.22%	9424
HTH	14.54%	7.80%	0
HTT	29.18%	11.10%	13072
THH	29.18%	11.10%	13072
THT	14.54%	7.80%	0
TTH	23.17%	8.22%	9424
TTT	7.63%	1.90%	0

Table 6: Results from running listing 1

Notes about the code: The Lowest odds column indicates the worst odds that a given pattern has in all possible tournament arrangements, and the highest odds indicates the best odds. The tournaments "won" column indicates the number of times a pattern had the best chance(s) of winning the tournament (there may have been ties in certain situations). Since I was unable to accurately generate all the unique tournament seeding that are possible, I instead just generated all the permutations of the different patterns and used that as the seeding for the brackets. This may have thrown off the numbers in Table 6's tournaments "won" column, however it should give a good rough idea of the "best" strategies.

2.3 Analysis of Best Strategy

It seems as though HTT and THH have the best, and most consistent results in both the 2 player situation when taking the average of all the probabilities, as well as when a single elimination tournament is being played with random seeding. It should also be noted that TTH and HHT also perform well in both settings, with a significant amount of single elimination brackets resulting in both patterns being the most effective one.

In the single elimination bracket it also appears that TTT and HHH are significantly discriminated against, with there being no realistic chance for them to win the tournament (their chances being as low as 1.9%). This suggests that in nearly all situations, TTT and HHH are an ineffective pattern to pick.

Finally, THT and HTH look to be the middle of the road strategies. They did not perform the worst in any given situation, but at the same time could not compete with the four main strategies of HTT, THH, TTH and HHT.

3 Conclusions and Future Research

In the situation where the pattern is of length 3, Penney's seems to have two most effective strategies: HTT and THH. Both Conway's Algorithm and analysis through graph theory and combinatorics can help understand the situation and allow you to calculate the various probabilities of different strategies winning, as well as giving you an intuition as to why this is the case. In the future, investigation should be done as to how different pattern lengths can affect these results, as well as how different competition formats can influence what the most effective strategy is (such as double elimination, swiss tournaments etc.). It may also be interesting to investigate what happens when you add more letters to the alphabet for use in pattern choosing (i.e. using a "3 sided coin"), as the discrepancies in success of the different patterns may end up being more moderate.

In addition to this, it would be interesting to consider what strategies are most effective when people change their strategies over time. For example, consider a 64 person tournament where each player has to select a pattern before the tournament begins. This would mean that the most effective pattern would depend on the distributions of the other patterns in the tournaments. One way of modeling this situation would be through the use of a genetic algorithm. In the genetic algorithm a tournament selection scheme would be most appropriate, as you can re-distribute the different patterns among the players based on the success that the various patterns had in the tournament. It would also be interesting to see how the population would change over time, and if there is a potential equilibrium position to be found where some strategies reign supreme over others.

The reason why such a modeling could be useful is that you could model many sports leagues in such a fashion, particularly where you know the likelihoods of each strategy beating each other strategy beforehand (consider many online games where you have hundreds of thousands of games to pull data from). By using a genetic algorithm to model such sports, you could predict the strategy distributions of the player base over time. This could in turn give you insight into the balance of different games, giving you information on how you could tweak such a game in the future to encourage more diversity in the player base. For example if you could tweak Penney's game to give HHH and TTT more success in the various tournaments, then it may be beneficial to do so as it would encourage more diversity within the playerbase.

References

[1] Christoph Adami and Arend Hintze. “Evolutionary instability of zero-determinant strategies demonstrates that winning is not everything”. In: *Nature communications* 4 (2013), p. 2193.

[2] Roland Backhouse. “First-past-the-post games”. In: *Science of Computer Programming* 85 (2014). Special Issue on Mathematics of Program Construction 2012, pp. 166–203. ISSN: 0167-6423. DOI: `doi.org/10.1016/j.scico.2013.07.007`.

[3] E.R. Berlekamp, J.H. Conway, and R.K. Guy. *Winning Ways for Your Mathematical Plays*. v. 2. Taylor & Francis, 2003. ISBN: 9781568811420.

[4] C.M.L.E.T.C.C. Bicchieri et al. *The Dynamics of Norms*. Cambridge Studies in American Literature and Culture. Cambridge University Press, 1997. ISBN: 9780521560627.

[5] J.H. Conway. *On Numbers and Games*. Ak Peters Series. Taylor & Francis, 2000. ISBN: 9781568811277.

[6] Maxime Crochemore, Lucian Ilie, and Wojciech Rytter. “Repetitions in strings: Algorithms and combinatorics”. In: *Theoretical Computer Science* 410.50 (2009). Mathematical Foundations of Computer Science (MFCS 2007), pp. 5227–5235. ISSN: 0304-3975. DOI: `doi.org/10.1016/j.tcs.2009.08.024`.

[7] Christopher Todd Edwards. “The combinatorial theory of single-elimination tournaments”. PhD thesis. Montana State University, 1991.

[8] David E. Goldberg and Kalyanmoy Deb. “A Comparative Analysis of Selection Schemes Used in Genetic Algorithms”. In: ed. by Gregory J.E. Rawlins. Vol. 1. Foundations of Genetic Algorithms. Elsevier, 1991, pp. 69–93. DOI: `doi.org/10.1016/B978-0-08-050684-5.50008-2`.

[9] L.J Guibas and A.M Odlyzko. “String overlaps, pattern matching, and nontransitive games”. In: *Journal of Combinatorial Theory, Series A* 30.2 (1981), pp. 183–208. ISSN: 0097-3165. DOI: `doi.org/10.1016/0097-3165(81)90005-4`.

[10] Yutaka Nishiyama. “Pattern matching probabilities and paradoxes as a new variation on penney’s coin game”. In: *International Journal of Pure and Applied Mathematics* 59 (Jan. 2010), pp. 357–366.

4 Appendix

Listing 1: Generating probabilities of winning brackets in python

```
import math
import itertools
import copy


# Compute the Conway number C_ab
def c(a, b):
    result = 0
    for i in range(len(a)):
        if a[i:] == (b[: i] if i != 0 else b):
            result += 2 ** (len(a)   (1 + i))
    return result


# Compute the probability that a beats b
def p(a, b):  # Calculates the probability of pattern a beating pattern b
    if a == b:
        return 0.5
    n = (c(b, b)   c(b, a)) / (c(a, a)   c(a, b))
    return n / (n + 1)


# Input:    list of patterns that represents the seeding (i.e. [HHH, TTT,...])
# Outputs:  dict with the probabilities of each team winning the tournament
def tournament_odds(patterns):
    num_rounds = 1 + int(math.log(len(patterns), 2))

    curr_round = {key: 1.0 for key in patterns}
    next_round = {key: 1.0 for key in patterns}

    for j in range(num_rounds   1):
        groups = [patterns[i: i + 2 ** j] for i in range(0, len(patterns),\
                                                          2 ** j)]
        pairings = [groups[i: i + 2] for i in range(0, len(groups), 2)]

        for group1, group2 in pairings:
            # Deal with the first group
            for x in group1:
                result = 0
                x_current = curr_round[x]
                for y in group2:
                    result += x_current * curr_round[y] * p(x, y)
                next_round[x] = result
            # Deal with the second group
            for x in group2:
                result = 0
                x_current = curr_round[x]
                for y in group1:
                    result += x_current * curr_round[y] * p(x, y)
                next_round[x] = result

        curr_round = copy.deepcopy(next_round)

    return curr_round


Patterns = [''.join(x) for x in itertools.product('HT', repeat=3)]
Record = {key: [] for key in Patterns}
```

```
Winning = {key: 0 for key in Patterns}
for seeding in itertools.permutations(Patterns):
    probabilities = tournament_odds(seeding)
    best_value = 0
    for key, value in probabilities.items():
        Record[key].append(value)

        if value > best_value:
            best_value = value

    for key, value in probabilities.items():
        if value == best_value:
            Winning[key] += 1

for pattern, record in Record.items():
    print("{}:␣best␣odds␣=␣{:.2f}%,␣worst␣odds␣=␣{:.2f}%,␣\
␣␣␣␣␣␣␣␣␣␣␣\"wins\"␣{}␣time(s).".format(pattern,\
                                              max(record) * 100,\
                                              min(record) * 100,\
                                              Winning[pattern]))
```