T-Bomb Pairing Method For T9A Team Events

Trent Barnard

1 Abstract

I have developed a method that I believe to be the best possible way to perform the pairing process for T9A team tournaments. In this paper I will justify this claim and talk about possible extensions and future work. This project was specifically made with the 'No I in Team' UK tournament in mind and therefore is applicable only to 4 man team events, however, it can quite easily be modified for any size team.

2 Introduction

In order to explain how this works, it is important that we define how the pairing process works. There are 4 steps to the pairings process:

1. Both team captains select one army from their team to put forward to play and place the card for this army face down.
2. Once both teams have selected an army, turn over the cards to reveal which armies are chosen.
3. Both team captains now select two armies from the remaining three in their hand to face the opposing army revealed in the Step 2. The armies put forward by each team are kept secret and revealed at the same time.
4. The captain of each team then chooses which out of a) the army placed in Step 1 and b) the one army still in their hand, plays against which of the two armies the opponent placed in Step 3.

Generally, teams will create a 'pairings matrix' to determine how favourable particular matchups are. This is an $N \times N$ matrix, M where N is the number of players in a team and index M_{ij} is some metric relating to a predicted score range for a game between player i from your team against player j from the opponents team. The prediction metric used in this paper is a 5 point system shown in table 1.

An alternative, commonly used 'traffic light' system also exists with red, amber and green indicating a loss, draw, and win respectively is used by many teams. For this particular pairing method, a 5 point scoring metric is suggested as it offers a more detailed view of predictions. Due to the high level of variance and unpredictability in T9A games, I would not recommend using more than 5 point brackets.
The T-Bomb pairing method employs a probabilistic algorithm that gives a score for each possible selection that the team captain has to make. This score is an average of the future outcomes and will be explained in more detail in section 3.2.

Score	Prediction
1	0-3
2	4-7
3	8-12
4	13-16
5	17-20

Table 1
Scoring metric used for predictions.

3 Method

3.1 Pairing matrix

As previously mentioned, teams create a pairing matrix to determine which matchups to go for. This matrix is highly subjective and adds a challenging aspect to the game theory of pairings. It is extremely unlikely that two competing teams will have identical matricies, so this leads to a level of complexity and is the basis of my justification for using a probabilistic model as opposed to a deterministic or predictive model.

It is of paramount importance that the matrix is as accurate as possible, and unfortunately this is not something that the T-Bomb pairing method can do for you, however I have come up with a few ways to potentially increase the accuracy of your matrix:

- Each player in the team creates an independent $N \times N$ matrix and a final matrix is calculated using the average score suggested by each player. This reduces the inherant bias for predicting your own matchup.
- Get another team to complete your matrix for you and combine this with the other method to further reduce bias. Obviously there are issues with this method, but in theory it will provide more accurate predictions.

As this algorithm is of no use for generating the matrix, I can not offer any additional help with this, it is up to the team to provide a matrix as accurate as possible in order for the T-Bomb pairing method to work.

3.2 Algorithm

Once a team has generated their pairings matrix, it is easy to see the optimal final pairings, this is usually trivial to compute for $N = 4$, and for larger N the Hungarian Maximum Matching Algorithm can be employed. Due to the nature of the pairing process it is not possible to choose every matchup and therefore extremely unlikely that the optimal pairing can be attained.

The way the T-Bomb pairing algorithm works is probably best explained with an example, so Table 2 is a pairings matrix that we used for one of our games. Once the pairings matrix has been given to the algorithm, it will predict the expected scores for each player if they were to be put forward in step one. Then, after the algorithm has been told who was put forward by both team captains, it will give an expected score for each pair of players to be put forward in step two. The expected scores are averages of every possible outcome. Now, I understand this may be of some concern to some people because they think that not every matchup is equally likely, I think this is debatable.

Firstly, the chances that your matrix match your opponents are very slim, your predictions are not an accurate representation of theirs. This means that you can not reliably predict what the opposing team will do in a given situation. This is the first justification for a probabilistic model, although I accept that your predictions are potentially more representative of the opponents selection than just assuming an equal distribution of likelihoods. This leads me to my second justification, the pairings matrix for both teams is subjective and not open information, there is unlikely to be a Nash Equilibrium and therefore it is extremely unlikely that there exists an optimal strategy. Furthermore, the strategy you use is dependant on your opponents pick which is made simultaneously to your teams. This means that there is an element of 'rock, paper, scissors' and it then becomes a strategy to predict what your opponent will do. This means that there is generally a justification for each possible selection. This coupled with the fact that your opponents matrix is potentially very different to yours means that the 'safest' strategy is to assume that your opponent will make each selection with some equal probability. This is not to say that it is always the best option, the T-Bomb pairing method is used as more of an indicator as to what options are favourable for your team, you should interpret the results from the algorithm as advice instead of ground truth.

	ID	SE	OK	SA
OnG	2	4	2.75	4
DL	2.75	3	4	4
KoE	1.5	3.25	3.5	3.25
SA	1.5	3.25	3.75	3

Table 2
Example pairings matrix.

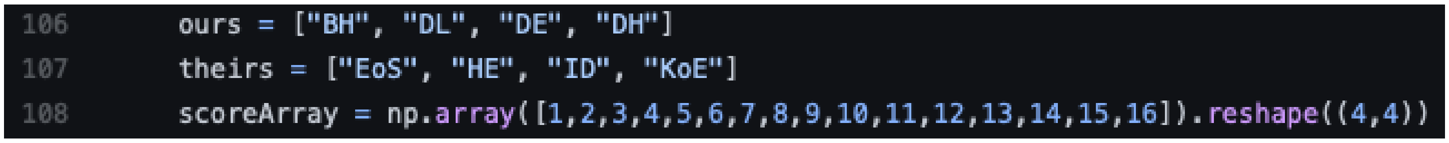
To explain the details of how the T-Bomb pairing method works it is explained best starting from step 4 of the pairing process. Step 4 is a deterministic step where each team captain has two players from their team and two players from the opponents team. The only logical result here is that each captain chooses the two matches that have the highest combined predicted score. There is no other sensible outcome. So for example if our team put forward OnG in step 1 and DL and KoE in step 2, whilst our opponent put ID in step 1 and SE and OK in step 2. This leaves two choices for us, either OnG vs SE and SA vs OK which yeilds a total combined score of 7.75 or OnG vs OK and SA vs SE with a combined score of 6. So we choose the matchup with the highest combined score while our opponent chooses the matchup with the lowest combined score (from our perspective). This is the basis for how the average scores are calculated, it is the sum of the 4 matchups that will be selected during step 4.

If we now go back to step 3, assume that step 1 has been completed, so both teams have put down their first player. Each team captain is now left with $\binom{N-1}{2}$ choices. This means that each selection has $\binom{N-1}{2}$ possible outcomes where the opposing team captain can choose a different combination of players in step 3. Once these selections have been made the outcome is deterministic as explained in the previous paragraph. The algorithm calculates the score for each possible outcome for each pair of players and returns the average score. So to use our example, if we had put down OnG in step 1 while the opponent put down ID, for step 3 of the pairing process we have 3 options (DL, KoE), (DL, SA) or (SA, KoE) while the opposing team can choose out of (SE, OK), (SE, SA) and (SA, OK). I won't bother to show the working here because it was already explained in the previous paragraph, but if we chose (DL, KoE), the combined predicted scores would be 12.5, 12.75 and 12.25 if the opponent selected (SE, OK), (SE, SA) and (SA, OK) respectively. This is an average of 12.5 and the algorithm would therefore give us a score of 12.5 for the selection of (DL, KoE) in step 3 of the pairing process.

Finally we can now look at step 1, this is essentially the same as the previous paragraph. For each of your N players, the average of each possible pair you can put forward (calculated as explained in the previous paragraph) is returned. This gives an expected score for each player in your team if they were to be selected in step 1 of the pairing process.

4 Code and its usage

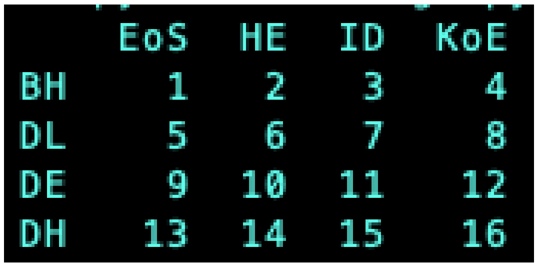
The algorithm is available at https://github.com/tBomb-t9a/T-Bomb-Pairing-Method.git. The code is written in Python and requires a few dependencies (just look at the top of the Pairing.py file to see these). The only modifications needed are on lines 106-108 where the team members and predictions need to be input. I will eventually update this code to read this from a .csv file at some point.



```
106        ours = ["BH", "DL", "DE", "DH"]
107        theirs = ["EoS", "HE", "ID", "KoE"]
108        scoreArray = np.array([1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16]).reshape((4,4))
```

Figure 1
Lines of code that need to be changed

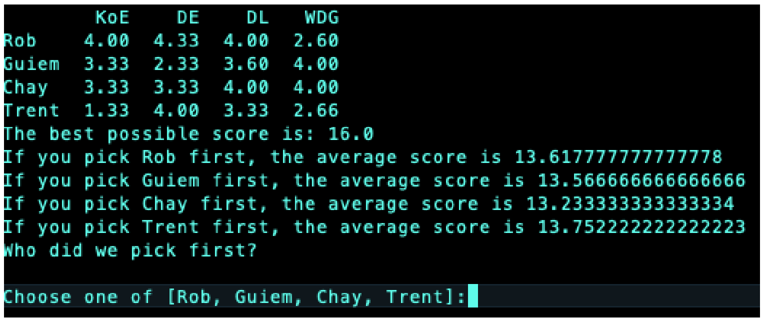
As shown in figure 1, the 'ours' variable should be a list of team members/armies in your team. 'theirs' is a list of team members/armies in the opponents team and 'scoreArray' is an array of all the predicted scores where the first entry is your first player vs their first player, the second entry is your first player against their second player. The output of this particular case is shown in Figure 2 below.



```
     EoS  HE  ID  KoE
BH     1   2   3    4
DL     5   6   7    8
DE     9  10  11   12
DH    13  14  15   16
```

Figure 2
Output using values from Figure 1

Once this is done the code is ready to use. Its usage is fairly intuitive, when you run the code the output is shown in Figure 3

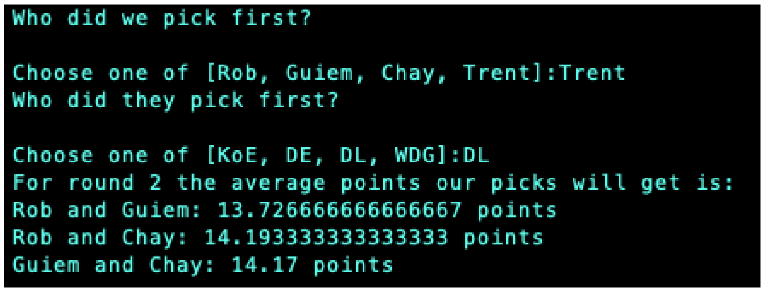


```
        KoE    DE    DL   WDG
Rob    4.00  4.33  4.00  2.60
Guiem  3.33  2.33  3.60  4.00
Chay   3.33  3.33  4.00  4.00
Trent  1.33  4.00  3.33  2.66
The best possible score is: 16.0
If you pick Rob first, the average score is 13.617777777777778
If you pick Guiem first, the average score is 13.566666666666666
If you pick Chay first, the average score is 13.233333333333334
If you pick Trent first, the average score is 13.752222222222223
Who did we pick first?

Choose one of [Rob, Guiem, Chay, Trent]:
```

Figure 3
Example output of the code

So at the top is the score matrix given in line 108 of the code, you should check this is correct. Under that it tells you the best possible score, this is calculated using the Hungarian Matching Algorithm (modified to maximise instead of minimise values), it is very unlikely that you will be able to achieve this score due to the nature of the pairing process, but it is nice to know. Underneath that, the results are shown, so for this example the algorithm suggests that picking Trent to go down in step 1 of the pairings is the best option. So all you have to do is type the name of the person who you decide to pick first and press enter. Figure 4 shows what happens next.



```
Who did we pick first?

Choose one of [Rob, Guiem, Chay, Trent]:Trent
Who did they pick first?

Choose one of [KoE, DE, DL, WDG]:DL
For round 2 the average points our picks will get is:
Rob and Guiem: 13.726666666666667 points
Rob and Chay: 14.193333333333333 points
Guiem and Chay: 14.17 points
```

Figure 4
Example output of the code (continued)

The code then prompts us to enter who the opposing team chose first. In this example we type in 'DL' and hit enter. The code then tells us the average score for the two players we should put forward in the next phase of pairing. In this case it suggests Rob and Chay. Once that is done, the code will exit. It is then up to the user to choose the two matchups with the highest combined predicted score.

5 Final words

Thank you for reading this, any comments/criticism would be appreciated. You can message me on Discord: 'Trent#4975' or on the T9A forum @Uwedxot if you would like to get in touch. I am also happy to help anyone who isn't experienced with Python and would still like to use the algorithm. Again, this code only works for teams of 4 at the moment. In the future I will extend this to teams of up to 8, I will also be adding a better way of inputting the matrix so the code does not need to be modified by the user.