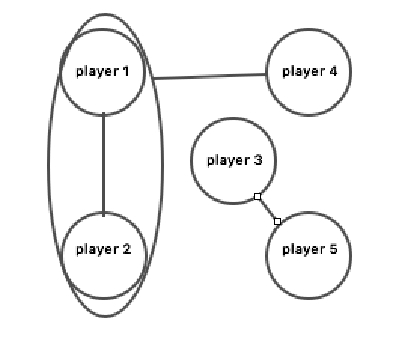
Final project report

##### Body of analysis 1

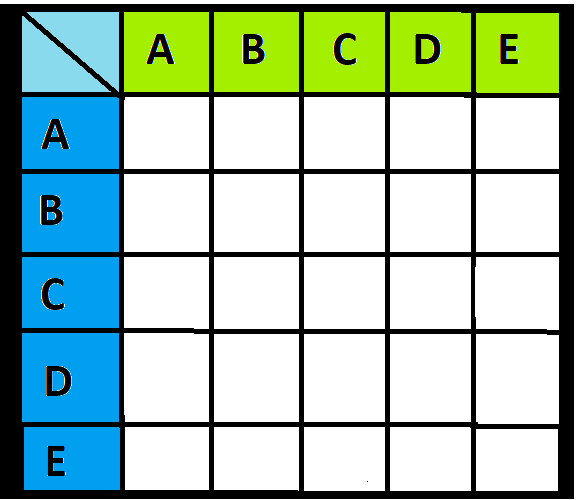
As a basic team-play system , each player will choose 5 famous basketball player to combine as their fantasy team to against the other player, as the main point mentioned before this project will focus on rating the fantasy in a different way which could figure out the relation between specific group and the change of rating. Therefore, there may have some potential relation and different combine which could impact the team rating.



This is one example which the team relation

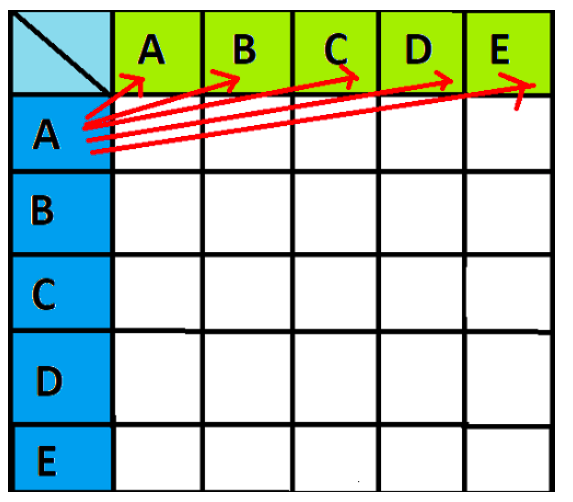
As the graph shown above , there will have a lot of different combines between the player which selected from the database. It is necessary to figure out the powerset of the team combine, also the range of powerset should less than 5 as the tester only can choose 5 players combine as a team.

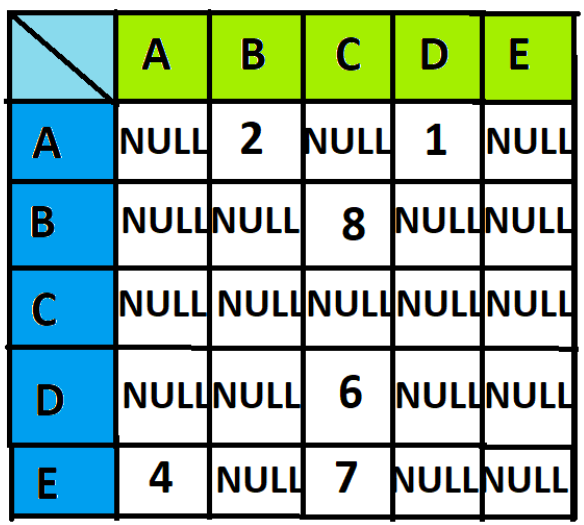
The graph is used to be design as the model to show the relationship combine between each player. A Graph is a non-linear data structure consisting of nodes and edges. The nodes are sometimes also referred to as vertices and the edges are lines or arcs that connect any two nodes in the graph. And it is consists of a finite set of vertices(or nodes) and set of Edges which connect a pair of nodes The Adjacency Matrix is the main factor which give us a way to represent our player relation graph in an efficient and structured procedure. It is easy to represent nodes and edges by creating a matrix table as the graph show below:



-empty matrix table

In this project, both of the horizontal and vertical node will be the power set, which is the power set of combine of players , stand for the function could be shown as P(p) \* P(p) = Graph. And for the edge in this project is used to show the relation between each power set if there are no relation between the group the edge will be input null, otherwise will be input 1 to show there do exist the relation between the chosen power sets.





-the relation show in details e.g.

The assumption data structure for player database is attempt to use the a list of list. [Name, combination group] all of the players data will be saved in this form and will be concluded in a list.The main point for this project is that try to rating the random players combine as a new team. Therefore, some problems will need to tackle. This project will also consider are there existed some relation between each player. The relation are mainly concluded as two type (positive or no relation), in the data coding process it will use 1 or 0 to stand for the relation. If there are two players have a positive relation that will cause the rating of both will increase due to the team combination and, the rating of the team could also be improved. Furthermore, this project will attempt to use some algorithm to figure out is there existed the negative relationship which may have a negative impact on the player, also if there are two player in the team have negative relation will decrease the overall rating for this team.

The main algorithm used to calculate the final rating of a team is memoization, the formula is

Here the Strength is used to describe the rating of a player or team, S is the total number of the player and t is the player which is a number of T. Here is a graph to show the function in details:



As the graph shown, the goal is attempt to get the rating of three player team (Player1,2,3), in order to calculate the rating of 3 players combine, it is necessary to consider the possible combine for two player and find out the potential relation could affect the group rating.

In this case ,the formula could be understand easily as:

Strength(player1,2)= Strength(player1)+Strength(player2)+K,

Which the k is how their relation could impact their rating.

The first method to solve this condition is attempt to use the Fibonacci. In details, the Fibonacci sequence is a sequence in which each number is the sum of the first two numbers. The number at a specific position in the Fibonacci sequence can be obtained using recursive methods.

The pseudocode for fib can be conclude as:

Define Fib(n)

If n = 0

Return 0

If n = 1

Return 1

Else

Return Fib(n-1)+Fib(n-2)

As it is rely on the way of recursion, it is necessary to set up the base case, so for this project’s fib recursion is if the number of set one, it will straight return that one player rating.

Define Strength(set)

If num == 1

Return Rating(set[0])

Else

strength = 0;

For each subset in itertools.combination(set, len(set)-1)

Strength += Strength(subset)

Strength / (len(set)-1)

Return Strength()

However, It will cost a lot of space to calculate, for example like the graph to calculate the overall rating of (player1,player2,player3). The algorithm will calculate the rating of (player1,player2) and (player1,player3) and (player2,player3) then combine their rating to get the rating of three player, if there are request to rating a player of 4 group it will do a recursion until find all factor it need to calculate, In this case, the recursion will do a lot of redundant work and have a low efficiency. So I try to find the more efficient way to solve the calculation (improve the fib recursion), which is the Memoization, the basic idea for memoization, is attempt to save all of the result that obtain from the recursion. And take out the result directly when we use it to calculate further information. Therefore, an array is defined to store the calculated data, and then query from the array when needed, which will save unnecessary calculations and improve the efficiency of the program.

Here is the pseudocode for Memoization:

Define Strength(set)

Memo = keys:powerset5 -> values: zeros(len(powerset5(players)))

If num == 1

Memo[set] = Rating(set[0])

Return Rating(set[0])

Else

If (Memo[set] == 0)

strength = 0;

For each subset in itertools.combination(set, len(set)-1)

strength += Strength(subset)

strength / (len(set)-1)

Memo[set] = strength

Return Memo[set]