Android Based Football Matchmaking Application

Anushil Mahajan[1] Aniket Kale[2] Dhananjay Ashok[3] Prof. Nita S Patil[4]
[1,2,3]B.Tech Student [4]Assistant Professor
[1,2,3,4]Department of Computer Engineering
[1,2,3,4]Datta Meghe College of Engineering, Airoli

Abstract— Football, being a global sport, has started to gain popularity in India in the past few years. Playing football in football arena, also known as "turfs", has now become a growing trend. Every football player knows that how difficult it is to find an opponent team to play. The conventional method is to make your own team by calling up a few friends, book the slots on the phone and also search for an opponent team which is a very tedious task. Finding a good opponent team is the biggest problem that the players face. Till date the whole process is manual from forming the teams to booking the turf. This is where the Derby brings in a major change. "Derby" is a football app which provides features such as match making, challenging other teams, booking turfs, and creating and managing your own team. This app provides each user to create their own clan i.e. team and challenge the opponents based on their preferences. Users can also search and book turfs.

Key words: Android, Match Making, Turfs, Weights, Challenge

I. INTRODUCTION

Derby is a Football based Android application designed for a structured team management and matchmaking process. Derby means games between two rivals of close geographical proximity, Hence, the name derby of our application. The team management feature will provide the player an interface to manage their own teams and also set their preferences. Preference includes location, capacity of the turf team wants to play on, and preferred time slots and the days of the week preferred. The challenging feature comes with two options for the user. One where the teams can manually search for the team they want to play with. Second, where the users can ask the system to randomly display the hits based on the calculated weights. In both the cases teams are obliged to put down a challenge in the challenge pool. This matchmaking feature is the key highlight feature we present in this paper. After opting for random matchmaking, the deployed system calculates the weights of each preference assigned based on the fuzzy logic and accordingly display the hits based on the maximum calculated weights. The users can then challenge or reject the team. The two kinds of users are the captain and the team member. Only the captain can create the team and challenge the teams. The team member can only accept the team invites. Booking turf is being totally changed from the conventional way. Players are made aware of the available slots so that they can play their games accordingly. Players can explore their level moving to various zones throughout the city. Also the players can rate the turf depending on their experience while playing. For better understanding "Derby" also provides the feature of posting reviews for the turfs.

II. LITERATURE REVIEW

There are multiple parts to implementing random matchmaking algorithm. You need a way to determine the similarity between users and a way of rating this similarity. Also, you need a way of performing the algorithm, for instance: do all the users go into a pool and the server makes the matches or do you separate the users and one performs the operation while the other waits to be matched.

Similarly rating depends on the specific algorithm you choose to use but a weighted value is one way to do it. In this case, each attribute a user contains is assigned a weighted value which signifies the importance of that value to the algorithm. The higher the weighted value the more important it is. The weighted value you choose by which you deem more important. This value is meaningless on its own but is useful relative to the algorithm. Similarly, the total computed similar value assigned to the user (outcome value of the whole algorithm with the weights applied) is meaningless on its own but is useful when compared to other users with a computed value. [1]

Once these values are obtained, you can simply sort the users (highest value being most similar) and then connect. But how exactly do you determine similarity between users? One way would be to look at the attributes and see if they equal.

Now, if we know all the "types" of the users attributes (exact, set, or range), we can use the correct formula to get the value, V, multiply by its weight, W, and get the total match value between two users, M:

$$M = (V1 * W1) + (V2 * W2) +, ..., + (Vn * Wn)$$

III. BLOCK DIAGRAM OF THE PROPOSED SYSTEM

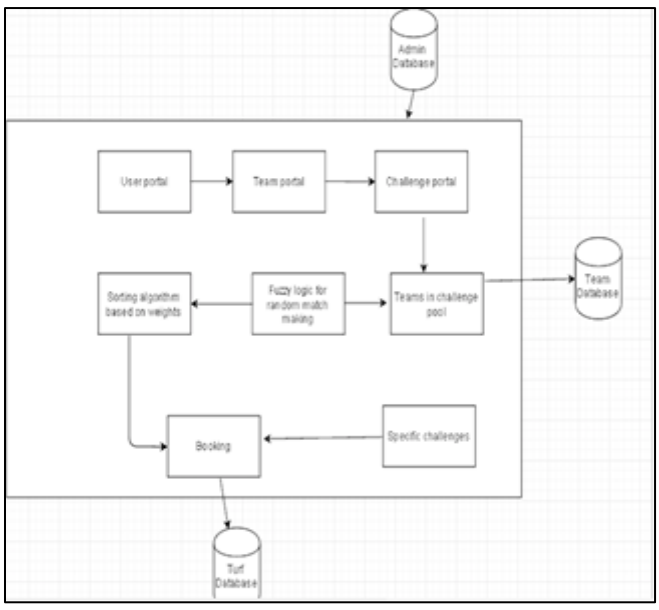

Fig. 1:

With the application users will be able to join teams, challenge other teams and also generate random match requests to play with random unknown teams. Turf owners will be able to add their turfs to the application. To maintain the quality standards of the turf, there will be feature of giving ratings and reviews to the turfs. Also users would be able to post the events as advertisements on the app. Admin will approve the publishing of new turf to the database, also he can ban players or turf owners from the app. There will be a unique id for players, turf owners and turfs.

IV. SYSTEM DESIGN AND WORKING

The process of random match making uses fuzzy logic to maximize the weight associated with the parameters and displays the healthiest results out of the pool. Users put a "Challenge" in the Challenge pool before searching or matching random teams. The user when clicks on random match making, the random match making algorithm is deployed. By comparison with the challenge and user preferences the healthiest challenge matches are selected according to the weights associated. Then the user can select a challenge of their choice and accepts the challenge. Then the system notifies the challenger that a viable challengee has accepted the challenge. After confirmation from both ends the match is scheduled. Then they are redirected to pay for the turf slot. The entire match making process takes place in accordance of the following phases:

A. Phases of Match making Algorithm:

Phase 1:

Before random match making, the user has to create a team he is going to play with and also he has to set his preferences. The team might consist of a maximum of 5 players where the user might be the team leader(captain). The preferences include information like the location, timing, type of team and the days preferred. These preferences are the basis of selection of challenges. Also the user must mandatorily set a challenge which goes down in the challenge pool.

Phase 2:

The main functioning of the match making algorithm is by calculation of weights. The derivation of this algorithm is loosely based on the fuzzy logic as the improper data is fed into the algorithm and the resultant weight computed W is the crisp value which is used to arrange the match cards in the descending order of the computed result. We use the aforementioned idea of fuzzy weight calculation in the literature review to derive our own computation formula for the match making process. These weights are calculated for each and every challenge in comparison with the user preferences. The parameters location, rating and timing are referred from a predefined set of values. The range of the values is from 0 to 1. It is based on three values represented as T =timing, R = rating , L = location.

Computation of Location value(L): For example, if the distance between the two turfs is less than 10kms then a perfect value of L=1 is assigned, for less than 20kms a value is L=0.9 and so on. A location matrix is used to compare the preferences of the two teams. The step ranges of these values are predefined and are just referred to calculate the weight. The original distance between the turfs is processed and represented using a Directed Graph. Once the distance between two turf locations is compared then the values between the range 0 to 1 are assigned for the distance values between each node in the directed graph.

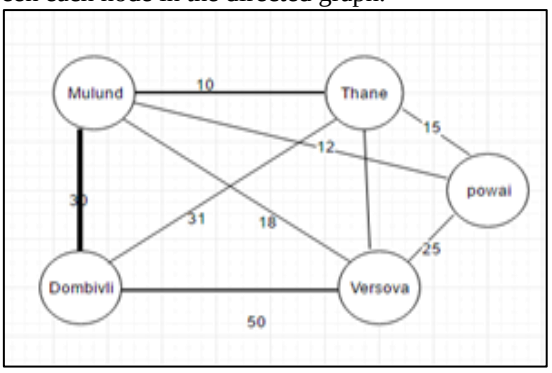

Fig. 2: Weighted Graph

Similarly, computation for calculating the value of timings set by two teams is done by comparing the time preference set by the teams. For example, if the challenger has the time range of 1 p.m. to 3 p.m. where he is available to play and the challengee has the range 2 p.m. to 4 p.m. then a value of 0.5 is assigned, as the probability of time match is half between the two teams. A value of t=1 if the timings match perfectly and 0.1 if don't match at all. This values will be predefined by us in the algorithm.

Similarly, for computing the value of rating firstly the user has to select what kind of team they are out of the three options:

1) Average team
2) Intermediate team
3) Professional team.

If both the teams are of the same level then a perfect score of t=1 is assigned, t=0.75 if the match is between the average team and intermediate or the intermediate and professional. And, t=0.5 for a match between average and professional team.

Phase 3:

The working of the match making algorithm is as follows:

- Step 1: Initialize the weights according to the user preferences parameters with comparison to the challenge preferences.
- Step 2: Set parameters by comparison (value: 0-1) as T =timing, R = rating , L = location
- Step 3: Using these parameters calculate the weights by-

$$W = T*R*L$$

- Step 4: Calculate the weights for all the challenges in the challenge pool up to a threshold value of weights.
- Step 5: Group the challenges according to the value of maximum weights.
- Step 6: Display the match cards in the descending order of the calculated weights.

Hence the match card with the highest computed value of W is displayed first and so on. Hence, our system provides the team the best matches according to the professionalism, location and timing set by the team. If they do not want to challenge a particular team then they can reject and move on to the next team.

Phase 4:

After selecting the match, the challenger is notified and then he is asked to confirm the match. After confirmation the match is scheduled and both the users are redirected to payment gateway for booking the turf time slot.

The booking of turfs is carried out in a very simplistic manner. First the user searches for the turf by filtering various factors like location, capacity, name and so on. Then the user selects the turf as per their choice. Now they can select the time slots according to the available slots and reserve it. For confirmation the user has to pay the respective amount. After payment the turf time slot is confirmed and the acknowledgement is sent to the user.

V. RESULTS

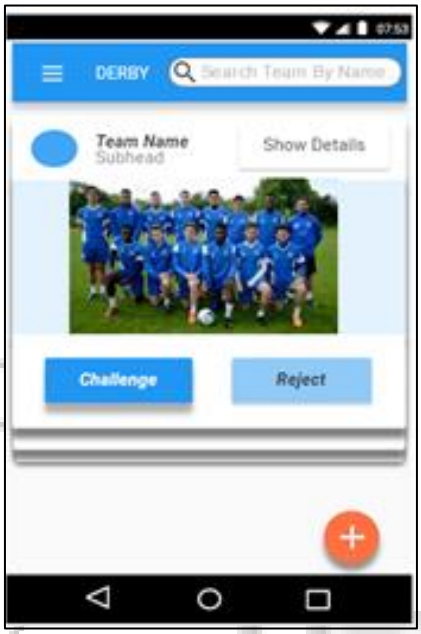

Fig. 3: Random Match Cards

Fig. 4: Home Page

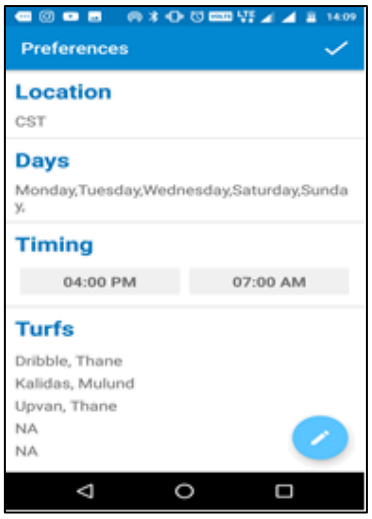

Fig. 5: Manage Preferences

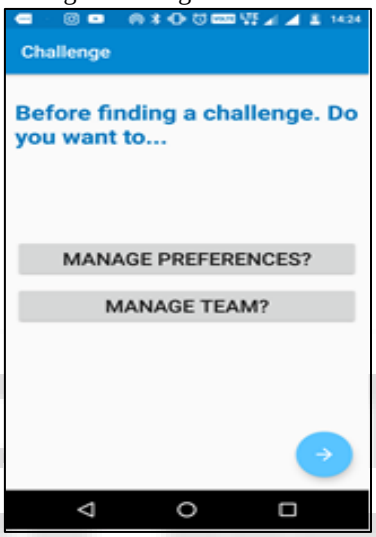

Fig. 6:

VI. CONCLUSION AND FUTURE SCOPE

In this paper, we presented a fuzzy based weighted algorithm that could calculate the probabilities of each match and display the results accordingly in the descending order of calculated final values. The user could challenge any of the teams from the displayed results in the challenge pool. The weights of each attribute will be predefined by the admin. Based on this derived formula, the system computes the health of each match.

Fuzzy sets were used to model the uncertainties arising due to the varying preferences. The consistency of each preference's decisions are determined and used to calculate the final weights for the decision making process. The system also presents another process that is challenging the team by searching the name. Derby also provides an interface for easing the team management process with features such as creating and managing teams, adding preferences. It can also be used to review and book the turfs. Organizers can post the latest events. These events would be authenticated by the admin and then notified to all the active users.

REFERENCES

[1] http://stackoverflow.com/

