A players clustering Method to Enhance the Players' Experience in Multi-Player Games

Yannick Francillette
LIRMM
University of Montpellier
France, CNRS
yannick.francillettee@lirmm.fr

Lylia Abrouk
Le2i
University of Bourgogn
France
Lylia.abrouk@u-bourgogne.fr

Abdelkader GouaÏch
LIRMM
University of Montpellier
France, CNRS
gouaich@lirmm.fr

Abstract - Online multi player games are a kind of video game where players have to interact together in a same game environment through an Internet connection. In these games, the players are grouped in different sessions where they can only interact with the member of the session. In order to maximize the player's experience, game designers have to solve different issues. The first one is to maximize the number of players in the different game sessions. As these game are designed to provide the better game experience when the maximum number of players are in the session. It is important to avoid session with few players. The second one is to create session where the players have the same skill level. A too large difference between the level skills of the players can create frustration.

In this short paper we focus on playerÕs skill and present an approach in development to automatically detect communities of players in order to create game session. Our approach is based on game play component, user profile and player interaction with the different game play component. We describe the experiment scheme that has been designed in order to evaluate the impact of the proposition on player satisfaction.

I. INTRODUCTION

Multi-players video games are a kind of video game that evolve several players in a same game session. These games tend to promote social interaction between players, in contrast to single player game where the player interact with a set of entities that are controlled by an artificial intelligence (AI). In multi-players games the player interacts with a mix of entities that are controlled by an AI, and by other player. We can divide multi-player video games in two groups: networked multi-player games that link several users divide through a network and local multi-player games where the players play in the same computer.

Online multi-player games are a sub-category of networked multi-player that use the Internet as a network. Thanks to the Internet a player can potentially play with thousands of players. One design issue in these games is to help the player to find a game session. Indeed, while in local multi-player games the players are already grouped, in online games the player has to find a group of player in order to begin the game. In order to deal with this issue, game designers give to the players two kind of tools: (i) a program can provide to the player the list of game session and information about these sessions. Then, the player choose the session that he wants to join. (ii) a program is charged to find a connect the player to a "correct" game session. This program is called a matchmaking mechanism. With this solution the player does not have any effort to do in order to play.

In the mobile gaming context, the matchmaking mechanism is an important tool to provide a good game experience. In these supports, it is difficult for the player to browser between the different session in order to choose one. On the one hand, the screen and the input sizes are not adapted to perform this task. On the other hand, in this context the player is outside and he wants to quickly play a game before he has to leave the place. So, he does not have time to spend in consulting game sessions information in order to make a choice.

When designing this matchmaking mechanism the game designer has to deal with this question: "What is a correct game session for a player?" A correct game session can be defined regardless of the skills of the player and it can be based on characteristic such as the number of player or the latency. But, more and more the skills and the player style are considered in this process. These characteristics are considered because they can affect the players' experience. The player's experience refers to the ensemble of sensations, thoughts, feeling, actions applied on the player during a playing session [1]. So, according to the author, the player experience is not a property of the game, but a player's state that emerges during the interaction between the player and the game. And in online multi-player games, the people are as important as the game mechanics and for example, the players can leave the game because of the attitude and actions of other players in the game [2]. Another example is that, put in the same session a beginner and a expert can result in a bad player's experience because the expert will bored to play with a player which discover the game mechanics. And the beginner will be frustrated to play with a player which does not give him the time to discover these game mechanics.

In [3], Bartle highlights the problem of how the player's experience can be affected by the behaviour of other players. He studies the different players' style in multi-user dungeon games. He extracts four archetypes of player and he notes that influencing the number of players of a particular archetype affects the number of players of another archetype. More recently, Riegelsberger et al. suggest in [4] that the matching of players according to their behavioural preferences can help to reduce undesired behaviour in game environments. The reducing of undesired behaviours is a way to provide a good game experience.

The goal of this short paper is to present our ongoing work to implement a matchmaking mechanism for a multi-player level generator which takes account of the different level of skills and players' preferences.

This proposition is based on the modelling of a video game through a set of resources named game play component [5]. We observe how the player interacts with these components in order to build the player's profile. Then we apply community detection methods to group the players. This document focuses on the description of our clustering method.

The rest of this document is organized as follows: The next section presents an overview of the architecture of our proposition. The section III introduces the reader with the gameplay components concept. In section IV we present how we build the player's profile. The section V explains our clustering methods. The related work is presented in the section VI. Finally, we conclude this article in section VI.

II. Overview of the Architecture

The solution aims to find the game mechanic that are the more used by the players and what is the level of mastery of these game mechanics. In order to achieve this objective the proposition is divided in four main parts. Figure 1 shows the stages through which the proposition passes to reach its goal.

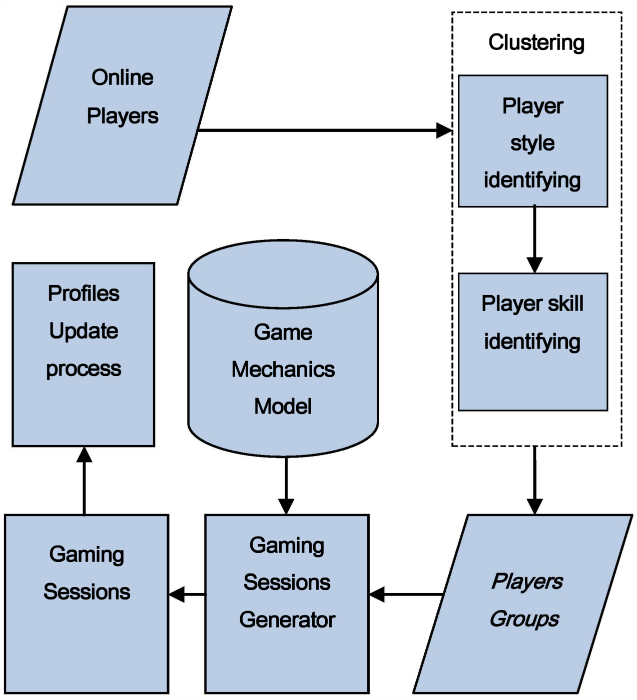


Fig. 1 Scheme of the main architecture of the proposition

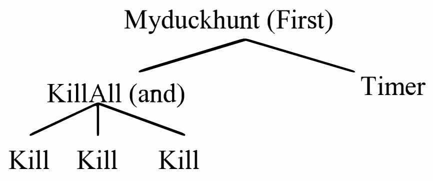
1) **Game mechanics modelling** is an initialization stage. The game mechanics are structured as a set of resource within which the players aim to interact. These mechanics are defined by the game designer.
2) **Observing the behaviour of the players** is the first stage of our matchmaking system. This stage is the watching of the mechanics that are most used by a player during its different gaming sessions and what is his level of mastery of each mechanics. This information are recorded in a document called player profile.
3) **Player's style based clustering** is the first step of the clustering stage. It deals with the identifying of the different player's style and it groups in the same cluster the players which have the same playing style.
4) **Player's skill based clustering** is the last step of the clustering stage. It deals with the identifying of the different player's skill and it groups in the same clustering the players which have the same level of skill.
5) **Creation of the game sessions** builds the different game sessions for each cluster.

III. Game play Component

The game play components are a way to model the mechanics of a game presented in [5]. The game mechanics refer to the rules and the objective of a game. The game play component model is inspirited from the game loop concept which is presented by Albinet in [6]. A game loop is a triplet by an objective, a challenge and a reward. The objective refers to what the player strives for. The challenge is the set of elements that prevents the player to easily reach the objective and that creates the fun. The reward is the gain that the player gets if he reaches the goal. According to Albinet, in a game the player is involved in a set of game loop that he try to resolve. A game play component is an entity which is added in a gaming scene in order to create the fun. The gameplay component contains and adds to the scene all the elements that define the objective the challenge and the reward. The gameplay component concept is built on two elements. The atomic gameplay component represents atomic objectives. The operators link several gameplay components in order to create a more complex objective. With these two kind of objects a game is structured as a tree that the player has to reduce in order to win the game.

In order to illustrate how the game play components work, we will model the game *"My Duck Hunt"*. This game is a shooter game that can be described by the next rules:

- The player must kill the tree ducks that appear in the screen.
- The player has a limited time to kill the ducks. If the time is over, he loses.
- The player gains some points for each dead ducks.
- The game can be modelled by the following game tree :



In this tree, the component kill refers to the objective of the player to kill a duck. This component adds a duck in the scene and control if the player kills this duck. This component control the duck move in order to create the challenge. When the duck is killed the component gives some points to the player. Three components kill are linked with an operator *and* in order to create the objective to kill three ducks. The operator *and* means the player has to complete all the components which are linked. The component time define the time constraints. It adds a timer and check if this timer is not over. This component timer is linked with the component killAll with the operator *first.* This operator means the game is lost if the component timer is complete before the killAll component.

IV. PLAYER'S PROFILE

The player's profile is the document that stores the knowledge about the player. The player's profile update task deals with the collecting and the updating of the information that are contained in this document. The information can be collected by two methods. The explicit methods ask the player to give the information for example with a form. The implicit methods computed the information by observing the behaviour of the player. Our proposition uses implicit methods.

The figure 2 shows the set of entities that are involved in our problem. This figure shows also what the interactions between them are. These entities can be classified in three ensembles: Players, Games, and Game play components.

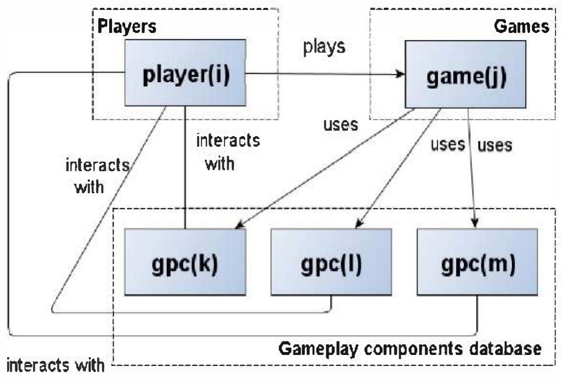


Fig. 2 Scheme of the set of elements and interaction involved in the system

The interaction between the different elements is the following: a player plays a game. The game is composed by several GPC. Through the game the player interacts with the GPC that are provided by the game.

We get the number of success and the trials numbers of each GPC. This information can be obtained by monitoring ending state of GPC. A GPC ends in one of the two following state: *success* means the player has reached the objective of the GPC; *failed* means the player has not reached the objective. With these pieces of information, it is possible to compute a success rate for each GPC.

The skills of a player in a game is modelled by the vector of real $U = u\backslash,...u_n$ where n is the number of GPC of the game. An element of U is defined on the value interval [0,1]. Each GPC require that the player use one or several skills to reach the goal. We get the ratio number of success per the number of try in order to estimate the level of skill of the player. For example, if the player presents a success rate equals to 2/10 for a GPC, we can conclude that the player does not master already the skill to easily complete the GPC. If the player presents a success rate equals to 8/10, we can conclude that the player masters the skill to easily reach the goal of the GPC. And more, we can compare two player by analyzing the success rate of the GPC. For example, if for a GPC g1 the *player 1* has a success rate of 2/10 and the *player 2* has a success rate of 8/10, we can conclude that *player 2* is more skilled that *player 1.*

V. CLUSTERING

User clustering deals with the identifying of users that share commons characteristic. In our context this task deals with group of players that prefer the same game play components and sub divide these group according to the level of skill. Our clustering process is divided in two parts.

A. Player's thematic clustering

The objective of the thematic clustering is to identify and create the different thematic groups. A thematic group is a group where all the members have interacted on the same resources. We focus on the interaction between the player and the GPC and not the player and the game. As one GPC can be used by different game, this strategy allows two player that interact with the same GPC but that do not play at the same game to be potentially in the same group. As input parameters, we use a vector which contains the use rate of each GPC. We record each time the player is involved with a component to compute this rate.

We can find different methods to cluster data in the literature [7]. These algorithms use different methods and input parameters to perform the clustering task. In order to solve this problem, we have chosen the Density-Based Spatial Clustering of Applications with Noise algorithm (DBSCAN) [8]. DBSCAN is a density-based clustering algorithm. This category of clustering algorithm computes clusters by analyzing the density of points in a region. The objective of these algorithms is to find a set of clusters where each cluster that has a neighbourhood defined by a given radius must contain at least a minimum number of points.

The DBSCAN algorithm proceeds as follows: from a starting node it gets the entire node with are near this point. The proximity between the nodes is computed by the proximity function that is used by the algorithm. The Euclidean distance is one of the more used proximity function. This is the proximity function used in this stage. Then, the algorithm starts from the nodes which have been found in the previous step and it looks for other nodes. And the end of this process, if the group contains a minimum number of nodes, this group is considered as a cluster. Then the algorithm selects another starting point outside the clusters which have already been defined to detect other clusters.

We have chosen DBSCAN because this algorithm takes as input parameter the radius and the minimum number of points. These two parameters can be matched on two requirement of our system. The minimum number of point parameter can be matched on the minimum number of player that a game session must have. The radius can be adjusted in order to include or exclude more different profile. Moreover we are able to adapt the values of these parameters according to the number of available players.

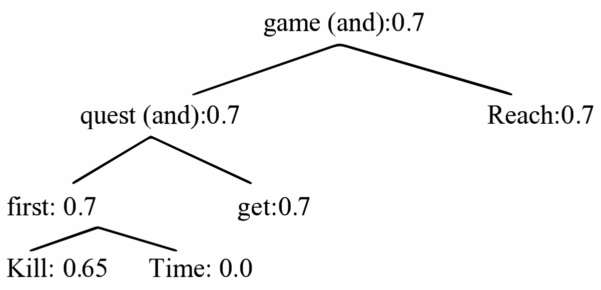
B. **Player's skills clustering**

The playersÕskills based clustering deals with players that are gather in the same thematic groups and that show the same ability. To perform this task we work from the success rate of GPC that is contained in the player profile. We also use the tree structure of a game that is defined by GPC.

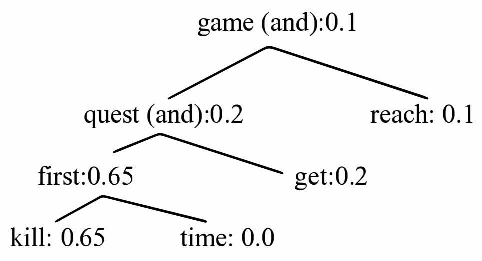
In order to group the players according to their skills we have to define a method to compute the similarity between two players. The algorithm that computes this similarity is described below. As the game has a tree structure, the algorithm travels the tree from the root the leaves. For each node, we compare if the success rate of the two players is near. If it is true, it means that for this GPC the two players have the same skills level. If it is false, we keep on travelling to the sons looking for node where the players are similar.

The two trees that are presented below show the success rate of two players on the same game tree. At each node the number indicates the success rate of the node.

a) Game tree with the success rate of the player 1:



b) Game tree with the success rate of the player 2:



These trees define a game where the players have to kill an entity in a given time; get an object in the environment and reach a position in order to win the game.

We perform the algorithm with the variable successRateDelta = 0:1. At the root we can see that the two players are different because player 1 has a success rate tof 70% where player 2 has only 10%. From this node, the next step is to see if the two players have some node where they have the same success rate. The node quest and reach do not have the same success rate, so for these nodes the players are not similar. For the node first the two players are similar because the condition abs (0:7 - 0:65) < 0:1 is true. The algorithm saves one node where the players have the same skills. For the node get the players are not similar. So, the result of the similarity computing is 1/5 where 5 is the number of visited node. In order to define the skills based clustering, we use the DBSCAN algorithm again. In difference to the previous stage, we use our similarity algorithm to compute the proximity between the players. The proximity between two players is the mean of the similarity for the entire game tree played by the two players.

Algorithm: Similarity

```
Require: player 1 profile p1, player 2 profile p2, maximum
         delta between success rates successRateDelta
1: gpcCounter Ñ 0 {gpcCounter counts the number of
   visited gpc node }
2: simGpcCounter Ñ 0 {simGpcCounter counts the number
   of gpc node with similar success rate}
3: for all (g e G1) do {Gi is the set of GPC tree shared by
   the two players }
4: if abs(p1g(i) — p2g(i)) < successRateDelta then
5: simGpcC ounter — simGpcC ounter + 1
6: gpcCounter — gpcCounter + 1
7: getNextGpTree() {we compute similarity for another
         GPC tree which has not been evaluated}
8: else
9: while currentNode has brother or (currentNode
         has son and players are not similiar) do
10: if abs(p1g (i)—p2g(i) < successRateDelta then
11:              set players are similar
12:              simGpcCounter — simGpcCounter + 1
13:          end if
14:          gpcC ounter — gpcC ounter + 1
15: end while 16: end if
17: return simGpcCounter/gpcCounter 18: end for
```

VI. **RELATED WORK**

We can find in the literature some work about players' style and players' skill. In this section we evaluate the works that analyze game data to cluster players and we exclude questionnaires based methods such as the Bartle test [9].

True Skill [10] Herbrich et al. present a Bayesian approach that is derived from ELO rating system [11]. The main idea is to attribute a score to the player, the higher the score is the more skilled the player is. This score is computed according to the level of the two player and the predicted outcome of the game. The idea is that a player with a higher score should win the game. If it is right the player gain fewer point. If the player loses the game he loses many points. These technical are based on the final outcome of the game; for example, a score or if the game is won or lost. Dealing with the outcome of the game gives a global indication about the player's skill. But this information does not allow to detect which are the sections of the game the player mastered or not. We can only see in which proportion the player can win or lose. The previous data cannot be used to predict the outcome of a new game.

Other methods are based on the computing of data which are extracted from some game mechanics such as the accuracy, the walking time or the number of death. In [12], Ramirez- cano et al. presents a user clustering approach in video game. In their approach, they split a game characteristic in three layers: action/skill, preference and social data. These layers represent different game variables that are recorded and computing by a clustering algorithm. For example, they record the shooting accuracy, the number of picture taken and the displacements in order to compute the preferred locations. They apply a different classification algorithm to each layers in order to find some similarity between the player. In [13] Drachen et al. try to clustering players in two AAA video game *(Tera* and *Battlefield 2: Bad Compagny 2)*. They use k-means clustering and the Simpley Volume Maximization (SIVM) which is Archetype Analysis technique. The Archetype Analysis is an alternative clustering methods. This method defines a set of entity with some extreme characteristics, these entities are called archetype. And others entities are considered as a mixture of these archetypes. Drachen et al. apply both methods on data such as kills, death, accuracy, the number of quests completed according to the kind of game. They highlight that k-means is a good technique to identify the players that do not use particular features of the game and it can help to detect players with low performance. On the other hand, SIVM is a good method to detect cheaters or bot. Drachen et al. proposes in [14] to use Selft-Organizing Map (SOM) to classify the players according to their behaviors. A SOM is a category of artificial neural network (ANN). This ANN uses as input parameters a 6 dimensional vector that is composed by data such as total number of deaths, the total completion time, or the different causes of death.

In contrast of final outcome based methods, the methods based data extract from game mechanics allows to detect more accurately the preferences and the skill of the player. The working on the data provided by the game mechanics has several weaknesses. The first one is the game developer has to identify the set of variables that are interesting for each new games.

Actually, it is difficult to identify which variable will be interesting to compute. And the more the game is complex the more there are different variables. The second weakness is the difficulty to identify a set of variable that could be used for several kind of games. For example, the number of kill in a First Person Shooter and in a Real Time Strategy game can not be computed as the same way because, the entities in both games are not killed the same way. Our method monitors the game mechanics themselves and these data are located between the final outcomes of the game layer and the game based datasets. As game play component can be shared between several kinds of game, we are able to built a generic preference and skill player profile. This allows us to generate a multi player level predicted as interesting for a group of players according to their old multi player or mono player gaming sessions.

VII. **CONCLUSION**

In this paper we have presented our ongoing work on a matchmaking system that group player according to their players' style and their players' skill. The proposition consists of the using of gameplay components to model the game mechanics of a video game as a set of resources with which the players interact.

By studying the way the players interact with the components we construct the player's profile. We record the interact frequency with the different components and the rate of success for each component. Then we apply the DBSCAN clustering algorithm in order to group the players according to their skill and the resources they like. We choose this algorithm because it is a robust clustering algorithm and it has the advantage that it need not to give the number of cluster as input parameter.

As future work, we plan to experiment the impact of the proposition on the player's experience. The experiment is defined to follow the repeated-measures experimental design. All the candidates will play two versions of a multi-player game. One with a matchmaking system that select randomly the player and the other with our proposition. The objective is to invalid the hypothesis: There is no difference concerning the average player's experience when the game session is built with our proposition and when the game is built from random players.

REFERENCES

[1] S. De Castell and J. Jenson, *Worlds in play: international perspectives on digital games research,* ser. New literacies and digital epistemologies.

[2] R. A. Bartle, "Developing multi-player games from scratch versus adapting existing single-player games for the multi-player environment," http://www.mud.co.uk/richard/nmg96.htm, 1996, [Online; accessed 15- July-2013].

[3] , "Hearts, clubs, diamonds, spades: Players who suit muds," http://www.mud.co.uk/richard/hcds.htm, 1996, [Online; accessed 15- July-2013].

[4] J. Riegelsberger, S. Counts, S. D. Farnham, and B. C. Philips, "Personality matters: Incorporating detailed user attributes and preferences into the matchmaking process," in *Proceedings of the 40th Annual Hawaii International Conference on System Sciences,* ser. HICSS '07, 2007.

[5] Y. Francillette, A. Gouaich, N. Hocine, and J. Pons, "A gameplay loops formal language," in *Computer Games (CGAMES), 2012 17th International Conference on,* 30 2012-Aug. 1, pp. 94-101.

[6] M. Albinet, *Concevoir un jeu video: Les methodes et les outils des professionnels expliques a tous !,* ser. Entreprendre: Developpement professionnel. Fyp editions, 2011. [Online]. Available: http://books.google.fr/books?id=iwOFZwEACAAJ

[7] S. B. Kotsiantis and P. E. Pintelas, "Recent advances in clustering: A brief survey," *WSEAS Transactions on Information Science and Applications,* vol. 1, pp. 73-81, 2004.

[8] M. Ester, H. peter Kriegel, J. S, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise." AAAI Press, 1996, pp. 226-231.

[9] GamerDNA, "Bartle test of gamer psychology," http://www.gamerdna.com/quizzes/bartle-test-of-gamer-psychology, 1996, [Online; accessed 15-July-2013].

[10] R. Herbrich, T. Minka, and T. Graepel, "TrueSkill(TM): A Bayesian Skill Rating System," in *Advances in Neural Information Processing Systems 19,* B. Scholkopf, J. Platt, and T. Hoffman, Eds. Cambridge, MA: MIT Press, 2007, pp. 569-576.

[11] A. E. Elo, *The rating of chessplayers, past and present.* New York: Arco Pub., 1978.

[12] D. Ramirez-cano, S. Colton, and R. Baumgarten, "Player classification using a meta-clustering approach," 2010.

[13] R. S. Anders Drachen, "Guns, Swords and Data: Clustering of Player Behavior in Computer Games in the Wild," 2012. [Online]. Available: http://geneura.ugr.es/cig2012/papers/paper87.pdf

[14] A. Drachen, A. Canossa, and G. N. Yannakakis, "Player modeling using self-organization in tomb raider: underworld," in *Proceedings of the 5th international conference on Computational Intelligence and Games,* ser. CIG'09, 2009, pp. 1-8.