An Approach for Team Composition in League of Legends using Genetic Algorithm

Lincoln Magalhães Costa, Alinne C. Corrêa Souza, Francisco Carlos M. Souza
Software Enginnering Coordination
University of Technology - Paraná
Dois Vizinhos, Paraná, Brazil
lincolncosta@alunos.utfpr.edu.br, alinnesouza@utfpr.edu.br, franciscosouza@utfpr.edu.br

***Abstract*—In recent years, Multiplayer Online Battle Arena and League of Legends are the most played genre and game. This is a genre which competitors are separated into two teams facing each other in a common objective, that generally involves destroying the opponent base. One of the most important steps in LoL is the selection of the champions who will be used in the match, since this involves many factors and variables that are important to the match such as attributes and abilities of each character. Due to several variables, team composition is considered a complex problem and can be handled with search-based algorithms. This paper proposes an automated team composition approach in League of Legends by combining attributes of the champions and game strategies via a Genetic Algorithm. It aims to generate teams from total set options in the game by focusing on satisfying constraints contained in real game strategies. Thus, this study also presents a fitness function to assess the adequacy of the generated teams. Finally, the paper reports empirical results regarding the effectiveness and efficiency of the proposed approach on three game strategies. Our results demonstrate that the proposed approach is useful for this context since the generated teams achieve high adequacy regarding the objective functions in a low time.**

***Keywords*-league of legends; team composition; genetic algorithm;**

I. Introduction

Quite recently, considering the constant evolution of the electronic gaming industry, the growth of the Multiplayer Online Battle Arena (MOBA) in the world of games is very remarkable [1]. MOBA categories are cooperative games that connect players to achieve a common goal. Due to their growing popularity, different MOBA games have emerged such as Defense of the Ancients (DotA), League of Legends (LoL) and Heroes of the Storm.

Currently, we find several championships of the modality as DotA 2 and League of Legends with millionaire awards. According to the 2018 Year In Review published by SuperData, the revenue of the League of Legends (LoL) in 2018 was about U$1.4 billion dollars, placing the game in the third place of the most profitable of the year. In addition, the channel of Riot Games (League of Legends producer) is on the list of the most watched by Twitch, an online broadcasting platform.

In general, a MOBA game consists of two teams with five players, each team against the other into a map. The teams must be selected before the match begins, the players select a character (also known as a Champion) to play from a set of 141 that will represent each of them during the game. Each champion has its specific skills, advantages, and disadvantages. It is important that every choice be made in a strategic way, according to the role of every player.

The success and competitiveness of a team depend on players experience, knowledge of champions features, own ability and mainly, strategic planning to compose a team of characters. The last point is absolutely necessary to make combined actions and team strategies to advance in the game or to counterattack to the opposing team's action.

The choice of champions can be considered a complex activity since the players must consider different aspects such strengths and weaknesses of each character, the pick of the opposite team and a strategy defined by the Coach. An adequate team is one that can minimize their weaknesses and maximize the resources for cooperation with each other champion [2].

In this context, the team composition can be addressed as an optimization problem. Most optimization problems involve huge amounts of solutions and complex constraints that turn an activity impossible to perform manually, but it can be treated using search-based techniques or metaheuristics. Metaheuristics are methods of Artificial Intelligence that operate in a set of solutions through local improvement procedures and higher level strategies to produce a process capable of escaping from local optima and performing a robust search of a solution space [3].

One of the major challenges in obtaining an adequate team is to combine the champions considering different aspects. This challenge consists of identifying a set of five champions that maximize their attributes for a specific game strategy. The fundamental problem is to identify the characters (champion1, champion2, ..., champion5), so that they are not selected more than once, their attributes do not overlap and they do not have many weaknesses.

This paper introduces a novel automated approach for team composition in League of Legends by combining attributes of the champions and game strategies via a metaheuristic technique. This technique is known as Genetic Algorithm (GA) and is guided by an objective function for assessing the solution candidates.

Overall, the contributions of the present work can be summarized into the following points: *(i)* an automated approach for team composition in LoL; *(ii)* an objective function to support search-based techniques for generating teams in MOBA games; *(iii)* an empirical assessment of the effectiveness and time of our proposed approach.

This paper is organized as follows: Section II details the background. Section III presents the related work. Section IV reports the proposed approach. Section V reports research design of experiments conducted. Section VI analyzes the results obtained and discusses the benefits, relevance, and limitations of the proposed approach. Finally, Section VII makes the concluding remarks and future directions are discussed.

II. Background

This section presents the main concepts about LoL particularities and GA that should be considered during team composition.

A. League of Legends

League of Legends (LoL) is a Multiplayer Online Battle Arena (MOBA) game developed and published by Riot Games company since 2009, year in which it was released. Most MOBA games have the same characteristics where two teams fight to destroy each other's bases. After a few minutes of the beginning of the game, some creatures (minions) spawn from each base to help the players win the game. Those creatures cannot be controlled by anyone and usually can be easily killed. By doing this, players earn gold pieces.

Each match has two teams of five players fighting across three lanes and an expansive jungle that holds buffs and neutral objectives. The main objective of the game is to destroy the enemy nexus, a structure localized on the base of each team, while simultaneously defending yours.

After starting a game, the first step is to select the character (also called champion) that will represent the player in the match according to the role you were selected to play. League of Legends has many roles e.g Attack Damage Carry (ADC) and Support. Carries main task is to eliminate specific targets. Because they attack from a distance, they always need to care the right position, so as not to suddenly find themselves on the line of enemy fire.

Initially, they are carried by their teammates, so they could freely collect gold needed for essential items. Supports are in charge of controlling vision, protecting the ADC, and just doing all the little things to help their team win. During the game, players evolve their champions, buy items and learn new skills to achieve victory. League of Legends also has secondary objectives such as neutral monsters that buff your character and facilitate the victory.

As said before, when entering the game players have to decide which champion will represent him during the match. Champions are player-controlled characters with unique abilities and attributes and cannot be selected by two players in a game. Champions and their abilities are unique, providing a different gameplay every time the player selects another champion. In addition to the attributes, champions are divided in classes and roles according to their responsibility in the match.

Each champion has statistics, that is a number indicating how well he can do a certain thing. These help defining the assets of a champion. There are 24 statistics, divided into 4 categories: Offensive, Defensive, Utility and Other. Those attributes are really relevant and must be considered when choosing the champion you will play with and the composition.

Team Compositions can literally refer to the five champions that a team chooses to play in a game, but they can also refer to overarching classes of team comps. Depending on the champions chosen, a team will have different strategies that they employ in order to win the game. The roles in LoL are Top, Jungle, Mid, Bot, and Support as presented in Fig. 1. Since there are three lanes in the game, usually the roles are divided between Support and Bot on the bottom lane, Mid go to the middle lane and Top to the top lane. The Jungle kills neutral monsters located between the three lanes and also helps their teammates lanes by ambushing the enemies.

Figure 1. Map and positions [4].

Cooperation is the essential component of LoL. There are champions that together combine to augment the abilities of their partners. On a larger scale, full team compositions can often merge multiple individuals into a huge singular force. Beyond synergies between individual champions, players should plan the whole team's strategy. LoL has many team compositions but some appear more frequently:

- Team Fight compositions involves champions with high attack damage and Area of Effect (AoE) abilities with burst damage in order to kill multiple enemies at once.
- Poke (also called Pusher) strategy excels at long-range combat, surrounding structures and objectives

and keeping the enemy team at a safe distance.
- Hard Engage compositions are designed to stop the enemy in their tracks, forcing decisive team fights or focusing to kill 1 or 2 enemies.

Considering all the variables described earlier, we have the problem examined in this study. Champions, their skills and unique attributes, synergy between them and team compositions are relevant factors that must be considered when beginning a League of Legends game and are factors that were taken into account during the development of this study.

B. Genetic Algorithm

Overall, complex problems tend to be very intricate due to the types of data, the large size of the domain of solutions and cost to find a solution. In such problems, the objective is to find the optimal of all possible solutions i.e. minimizes or maximizes an objective function. The objective function (also called fitness function) is a function to measure the quality of a candidate solution, and the set of all candidate solutions for a given problem is described as search space [5].

Solving optimization problems consists in finding values of the variables to give the best solution (minimum or maximum), i.e., the global optimum. A global optimum is an optimum of the whole solution domain D_s while a local optimum is an optimum of only a subset of D_s [6].

This type of problems can be solved through metaheuristic. It combines objective functions or heuristics in an abstract and hopefully efficient way, usually without utilizing deeper insight into their structure, i. e., by treating them as black-box-procedures [7].

Several search-based techniques and meta-heuristics including, for example, hill climbing [8] and tabu search [9] have been applied to solve optimization problems in different domains. However, methods like Genetic Algorithm, may be more specialized and work with predefined search spaces and search operations.

Genetic Algorithm is a probabilistic search technique based on the theory of natural evolution proposed by Charles Darwin [10]. GA consists of an iterative process aiming to identify the best solution from a population of solutions, known as individuals for a given problem.

The GA process starts with the population of candidate solutions typically randomly generated. Then, the individuals are evaluated with a fitness function predefined, and those that are more suitable to achieve the solution of the problem are sent to reproduction and consequently, generate new individuals for the population. Throughout successive generations, GA performs a set of stages to improve and evolves the population on each iteration until the algorithm finds the most suitable solution or reaches a stopping criterion, such as a fixed number of generations.

In this context, the evolution occurs through the use of two genetic operators: *(i)* crossover; and *(ii)* mutation. These operators allow an initial population to move through a given number of generations and succeed in generating successive populations with new individuals with genes more adapted than the previous generation [11]. Crossover is the process of concatenating two chromosomes, called parents, to generate two new chromosomes by switching genes. The input of this process is two chromosomes while its output is two different chromosomes. The reason for such an operator is that both chromosomes might represent successful parts of solutions that when combined even outperform their parents [12].

The mutation is the process of randomly changing the value of one gene in a chromosome so as to increase the structural variability of the population. The role of mutation is that of restoring lost or unexplored genetic material into the population to prevent the premature convergence of GA to optimal solutions. It ensures that the probability of reaching any point in the search space is never zero. Each position of every chromosome in the population undergoes a random change according to a probability defined by a mutation rate.

The GA flexibility makes them attractive for many optimization problems in practice. In this context, several studies have demonstrated the GA success in different domains including music [13], games [14], biomedical domain [15], among others.

III. Related Work

Few publications have appeared in recent years documenting different techniques for MOBA games team composition. Some of these studies address the use of different techniques such as deep neural networks [16], MinMax algorithm and linear regression [2] and logistic regression and K-Nearest Neighbors [17]. However, these researches did not addressed the use of battle strategies and genetic algorithm, and only one study [2] is for LoL games.

The work of Sapienza et al. [16] aims to analyze the complex interplay between cooperation, teams and teammates' recommendation, and players' performance in Dota 2 game. It was built a co-play network of players using a Deep Neural Network (DNN), with weights representing a teammate's short-term and long-term influence on player performance. The results indicate that skill transfer and performance improvement can be predicted. Additionally, the skill transfer can influence teammates to have in increasing or decreasing the actual player's skill level throughout matches.

Oliveira et al. [2] proposed a computational model using MinMax algorithm and linear regression for the LoL game. This model can automatically suggest to the player a team composition that aims to decrease the opponent's gains and widen the advantage of his team. They used MinMax to model and generate possible good teams that can be champions. Additionally, Linear Regression was applied to

Table I
SUMMARING S

Studies	Techniques	Features	Games
[16]	DNN	player's performance	Dota 2
[2]	MinMax, Linear Regression	every champion picked for each game role and the result of the match, such as victory or defeat	LoL
[17]	KNN, Logistic Regression	which heroes were chosen for each team, how those heroes performed over the course of the game, and which team ultimately won the game	Dota 2
Our approach	GA	champions attributes, game strategies	LoL

evaluate this classification in order to identify the team with the highest chance of victory.

Conley and Perry [17] aims to recommend heroes that will perform well against an opposing team of heroes as well as predicting match outcomes for the Dota 2 game. In their study, it has been developed a recommendation engine that, using real data from DOTA 2 matches, computes the probability of victory in a match between two specific teams. The algorithms used for this purpose were Logistic Regression and K-Nearest Neighbors (KNN).

In our study we developed an automated approach for LoL team composition by combining attributes of the champions and game strategies using GA, which is guided by an objective function. The difference between our approach and the others is presented in Table I.

Based on the studies identified in the literature, we can notice that no study builds teams as in the proposed work. Most of them employ statistics and past games to recommend a team or only predict chances of victory. The advantages of our approach are primarily the generation of the best teams based on a battle strategy. In addition, it can be extended based on other features or even past game statistics, treating this as a search problem that can bring new opportunities for AI techniques and algorithms applications as it is used in our approach.

IV. APPROACH DESCRIPTION

The proposed approach attempts to generate a team composition in MOBA games and uses search algorithms to achieve this. The idea of the approach is to generate a League of Legends team with genetic algorithm using a fitness function based on each champion attributes.

Team composition or team generation is a classic search problem since it cautiously identifies a set of champions from a set that has all possible options to a specific formation.

As show in Fig. 2, our approach employs champions attributes into a mathematical function to evaluate the generated teams. The data representation used in this approach is a json file with information of all the 141 champions. Each one is represented as an object containing his name, description, roles, identifier (an ordered number between 1 and 141), icon and, most important, his attributes like armor, health point per level, movement speed, attack damage, health points and others. Champions attributes such as attack damage and movement speed are used to evaluate the adequacy of the generated team and lead the searching process to an optimum in the search space. Additionally, we have created a validation to guarantee that a champion will not repeat inside a team.

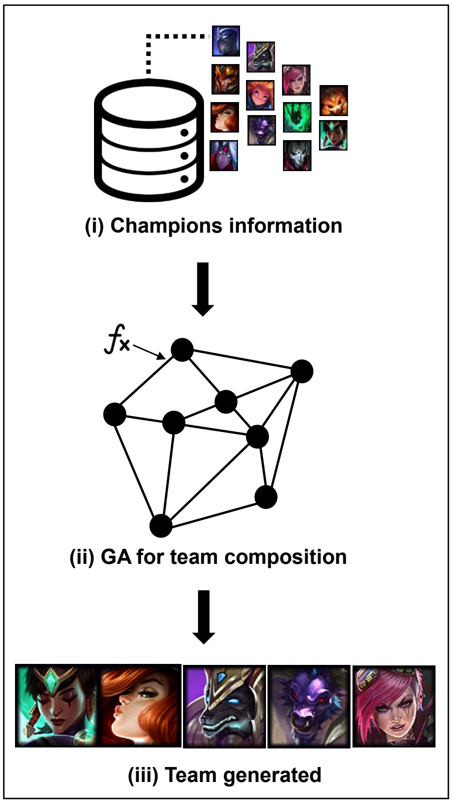

Figure 2. Approach for team composition and generation.

A. Team composition in LoL

Many online games use cooperation, a mechanism present in real-world systems in different environments and at various scales to improve the player experience, giving him the chance to play and cooperate with players all around the globe. This relation is in focus in games like LoL, which the players have to play thinking in themselves and in their teammates since there are many roles with specific responsibilities on the match.

In League of Legends, this cooperation occurs by combining attributes and abilities that can be complementary between the champions. For instance, we can consider a character with excellent attributes in battle. This kind of champion can propel a team to victory, nevertheless, without a support champion, it becomes vulnerable against an opponent.

In general, the team composition is more complex than the union between roles, mainly if we consider the map position of champions since they must have unique features or players with high skills. For this reason and for we do not want to create an undecidable problem, in this paper, we define two constraints.

Thus, the first major constraint in team composition in MOBAS games like LoL and DOTA can be described as: a) Any carry champion must have with him his support champion. This means, a team must be composed of at least one carry champion and one support, the rest of the characters may be of other positions.

The second main constraint is that a successful team can not be formed without a battle strategy (for instance, hard engage team, team fight, etc.). Combining attributes requires a common goal, therefore, we can define the second constraint as: b) Any team needs a battle strategy.

Formally, the problem in team composition consists of identifying a set of champions that maximize the number of their attributes respecting the constraints (a and b) presented in (1), (2) and (3).

Maximize

$$\sum_{i=1}^{n} attrib_1 + attrib_2 + attrib_m \tag{1}$$

subject to

$$strategy \in \{hardengage, teamfight, poke\} \tag{2}$$

$$team = (carryC_1, supportC_2, C_3, ..., C_5) \tag{3}$$

where C_i is the champion, n is the size of the team and this is equal to 5 and $attrib_m$ are attributes of each champion selected.

B. GA for team composition

For several years a great effort has been devoted to the study of Genetic Algorithms and this technique has been achieving great results in different contexts [18] [19] [20]. Even that team composition is not one of them, the results obtained on other applications demonstrate that this technique can be a good choice to optimize the generation of team composition.

The search starts by selecting random champions for the initial population, then, the teams in the population are evaluated. After that, GA starts the improvement and evolution process in the population of through genetic operators as presented in Pseudo-code 1. This step is performed until achieving the maximum number of generations.

Fig. 3 presents an example of crossover in two teams. Each gene in the individual is one different champion, a

Pseudo-code 1 GA for Team Composition

```
procedure AG
    Initialize the team population;
    Evaluate the teams in population;
    while not reach number of generations do
        Select individuals;
        Apply crossover and mutation;
        Evaluate new population;
    end while
end procedure
```

crossover works through exchange between genes in the solutions. After this process, two new individuals are created containing information of Individual 1 and Individual 2.

Figure 3. Crossover.

The mutation operator has the purpose of providing a diversification in the population. It works by a mutation in a individual, Fig. 4 shows this process in a solution. A gene into the chromosome is selected and changed by another random champion from the database.

Figure 4. Mutation.

In this paper, we implemented a traditional GA and handled the champions using a data representation called json which contains their information. The proposed approach uses a novel fitness function that incrementally guides the process to generate teams regarding a specific battle strategy and this is presented in the following subsection.

C. Fitness Function

An adequate objective function is considered the most important part of search-based techniques since it is which guides the exploration process towards the promising areas of the search space. Thus, more effective functions lead to significantly better results i.e., a metaheuristic can find an optimal team. However, the main problem is how to define a fitness function, because it must be created according to problem features. In other words, the objective is represented by a specific element from the problem capable of evaluating candidate solutions, in terms of their goodness and suitability [21].

LoL has many teams compositions and strategies that can lead a team to the victory. To maintain the balance, each one of them has their weaknesses and strengths. As example, we can cite a siege composition, where the objective is to pick champions with long range abilities to take down objectives easily and corner the adversary team. The main weakness of teams with this strategy is a composition focused on hard initiation, since the players will not be able to dealt the expected damage before getting into a team fight.

Based on that context, this approach focused on the three team compositions previously mentioned in subsection II-A: Team Fight (I), Poke (II) and Hard Engage (III). All the attributes used in the three fitness functions are presented in and their relation with each strategy is presented in Table II. It is important to consider that the maximum value not always will be the only one present in a team since it is not possible to have the same champion twice in a composition.

Table II
CHAMPIONS' ATTRIBUTES AND STRATEGIES

Attributes	**Maximum value**	**Strategies**
Attack damage (AD)	70	Team Fight, Poke and Hard Engage
Attack damage per level (ADL)	5	Team Fight
Attack range (AR)	650	Poke
Attack speed per level (ASL)	6	Poke
Health points (HP)	800	Team Fight
Movement speed (MS)	355	Hard Engage

Equation (4) defines the highest value that can be obtained by a Hard Engage strategy in the fitness function, according to the team size (TS) and highest MS and AD in the dataset.

$$hardEngage = (\max(AD) + \max(MS)) * TS \qquad (4)$$

The equation that expresses the maximum value achieved by a Poke strategy is presented in (5).

$$poke = (\max(AD) + \max(AR) + \max(ASL)) * TS \qquad (5)$$

The highest value reached by a Team Fight composition is expressed in (6). This strategy uses statistics like attack damage, attack per level and health points to calculate the maximum evaluation value.

$$tfight = (\max(AD) + \max(ADL) + \max(MS)) * TS \qquad (6)$$

V. EXPERIMENTAL STUDY

We conducted an experiment to analyze and evaluate the effectiveness of the proposed approach for team composition in League of Legends by combining attributes of the champions and game strategies. We are interested in measuring the effectiveness in terms of the fitness function and time. In this study the guidelines recommended by Wholin et al. [22] were used. The experiment was performed through a laptop with Intel Core i7 2.4GHz CPU, 8GB memory in the Windows 10 Pro operating system.

A. Experiment Definition

We used the Goal-Question-Metric (GQM) model [23] to set out the objectives of the experiment that can be summarized as follows:

*"Analyse **proposed approach** for the purpose of **evaluation** with respect to **fitness function and time** from the point of view of **experimenters** in the context of **the teams' composition in League of Legends**."*

For achieving the goal, we seek to investigate the following Research Questions (RQs):

RQ_1**: How effective is the proposed approach for composing a team for League of Legends based on their strategies?**

The effectiveness of the approach was measured using the fitness function for composing a team. The fitness value was computed for each setting and game strategy. We also performed this experiment 30 times and computed the average of fitness value. We have defined the following hypotheses for this research question:

$H1_0$**:** There is no difference on fitness value between the setting 1 (μst_1), setting 2 (μst_2) and setting 3 (μst_3), thus:

$$H1_0\text{: } \mu st_1 = \mu st_2 = \mu st_3$$

$H1_1$**:** The fitness value achieved by setting 1 (μst_1), setting 2 (μst_2) and setting 3 (μst_3) are different, thus:

$$H1_1\text{: } \mu st_1 \neq \mu st_2 \neq \mu st_3$$

RQ_2**: How efficient is the proposed approach for composing a team for League of Legends based on their strategies?**

The efficiency was measured using time. The time was computed only for the configuration that obtained the best fitness value considering all three strategies. We also performed this experiment 30 times and computed the time average. The time was computed in milliseconds. We defined the following hypotheses for this research question:

$H2_0$**:** There is no difference on time between the *hard engage* ($\mu HEngage$), *team fight* ($\mu TFight$) and *poke* ($\mu Poke$) strategies, thus:

$$H2_0\text{:}\ \mu HEngage = \mu TFight = \mu Poke$$

$H2_1$**:** The time achieved by *hard engage* ($\mu HEngage$), *team fight* ($\mu TFight$) and *poke* ($\mu Poke$)) strategies are different, thus:

$$H2_1\text{:}\ \mu HEngage \neq \mu TFight \neq \mu Poke$$

B. Experiment Design

In this study, two different experiments (E = e_1; e_2) were carried out. The first experiment (e_1) answered RQ_1 and the second (e_2) was conducted to answer RQ_2. For answering the RQs, this empirical study manipulated an independent variable: team generation; and four dependent variables were measured:

- **Strategy (**S**):** represents the play styles that are essential to ensure the team's win. We used three different game strategies: hard engage, team fight and poke.
- **Number of population (**P**):** represents the number of candidate solutions. The population is composed by p individual, where p is represented by three different parameters (P = 10, 20, 30);
- **Mutation Rate (**MR**):** The MR contains three different parameters (MR = 0.3, 0.5, 0.7);
- **Number of generations (**G**):** represents the number of generations in a genetic algorithm. We used three different parameters for generation (G = 10, 100, 1000).

For both experiments, we define three settings (ST = st_1 st_2, st_3), which are a combination of the parameters population, mutation rate, and generations, respectively. For each strategy, all three settings were used. A design overview of these experiments is presented in Table III.

Table III
EXPERIMENTS DESIGN

Strategies	Settings	Parameters		
		P	MR	G
Hard engage, Team fight, Poke	st_1	10	0.3	10
	st_2	20	0.5	100
	st_3	30	0.7	1000

C. Procedure of Experiment

To answer the RQs, we carried out the experiments as follows: (i) composing and improving team using the proposed approach and to measure fitness value and time; and (ii) comparison of the fitness value and time obtained in each strategy. These experiments were performed in four steps:

1) We generated different teams (T) as experimental subjects. These teams varied according to strategies.
2) We evaluated each team using the proposed fitness function. The fitness function set directly indicates the quality of the generated team.
3) We computed the fitness value for each team from the selected strategy.
4) The time is computed in milliseconds for each team from the selected strategy.

VI. Results and Discussion

The results of the experiment are shown following in separate subsections according to each research question.

A. Effectiveness of the proposed approach (RQ_1)

In this research question, the fitness value was computed according to each setting and strategy.

In Fig. 5 the lines represent the fitness value obtained for each generation and strategy using setting st_1. The results indicate that the hard engage strategy, on average (mean), achieved the best fitness value was 79.10%. With team fight strategy was achieved 77.23% and 73.30% with poke. Therefore, the fitness value average obtained in this setting was around 76.53%.

Fig. 6 shows the fitness value obtained for each strategy in 100 generations. We observed that the fitness value improves using this setting (st_2) in all strategies. Therefore, as can be seen in Fig. 6, it was possible to increase the fitness value by 15.23% for the hard engage strategy, 13.98% for team fight, and 19% for poke.

In the last setting (st_3), the fitness value achieved by hard engage strategy was 97.48%, 95.10% for team fight and 94.62% for poke. It is natural to expect that the larger the population and number of generations, the greater is the fitness value, mainly for a huge search space like in our study.

We noticed in the results that the null hypothesis ($H1_0$) was rejected i.e, st_1, st_2 and st_3 are not able to achieve the same value fitness when executed for the strategies. In Fig. 7, the team adequacy average achieved by st_3 is 19.23% best than st_1 and 3.13% best than st_2. Therefore, the proposed approach is more effective using the setting st_3 with 95.73%.

As explained in Section IV-C, the maximum fitness value is obtained by the sum of the highest values of each attribute. Thus, higher fitness value means better compositions according to the chosen strategy. This GA tries to maximize the fitness function to provide a population consisting of the fittest individual.

The results thus obtained are compatible with the strategies expectations in the mathematical and empirical context. With st_3 the fitness value was higher than 94% and the compositions could be used in a LoL game because the champions have synergy and fit the proposed strategy.

Fig. 8, 9 and 10 show a team composition generated by the approach using the team fight, poke and hard engage strategies, respectively. Despite being rated with a high

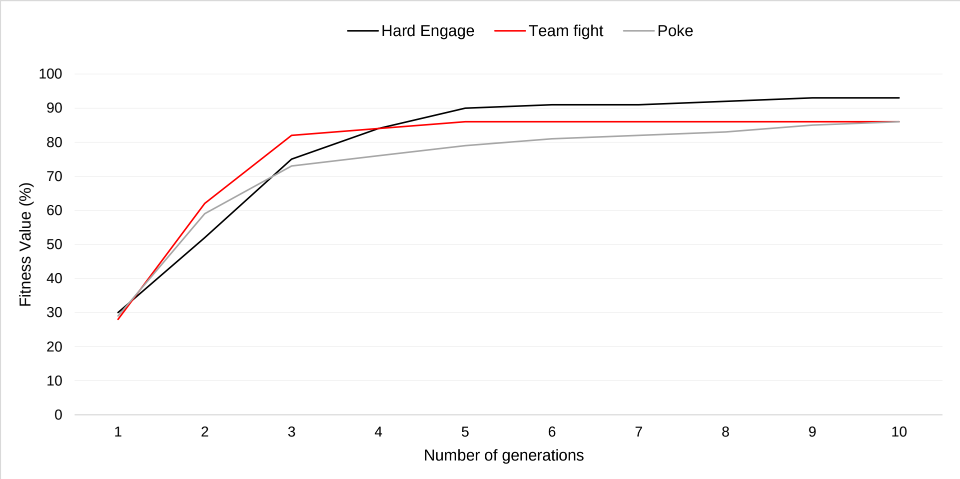

Figure 5. Fitness value achieved using the st_1 setting for each strategy.

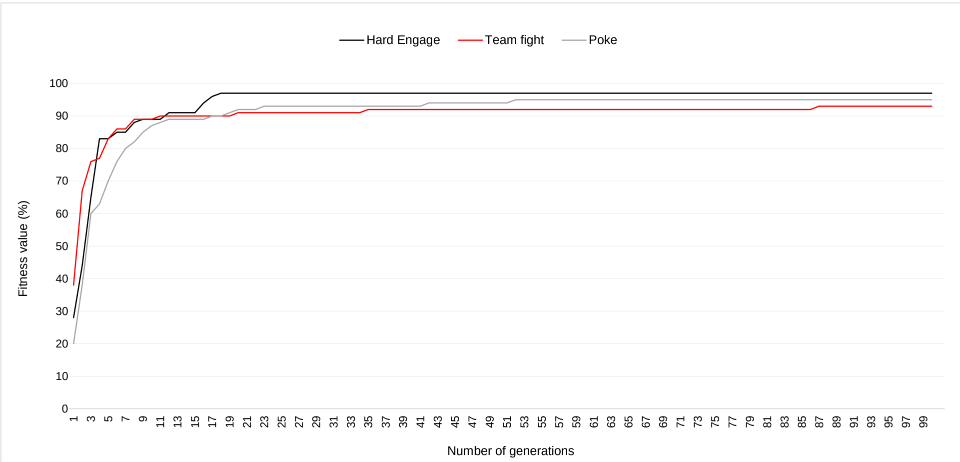

Figure 6. Fitness value achieved using the st_2 setting for each strategy.

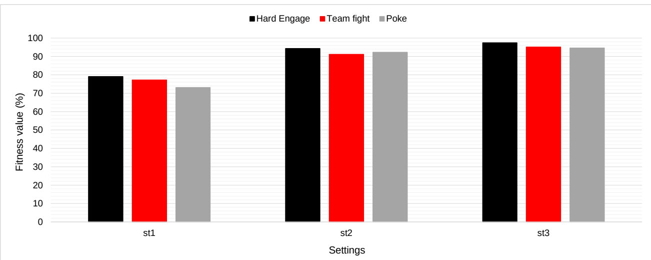

Figure 7. Average fitness value according to settings for each strategy.

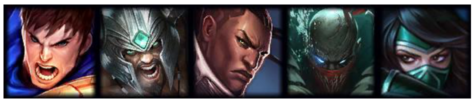

Figure 8. Team generated using team fight strategy.

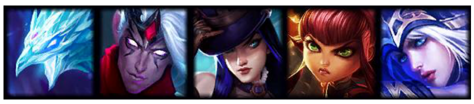

Figure 9. Team generated using poke strategy.

fitness, the team generated for the poke strategy would hardly be seen in a game by having three attack damage carries.

The champions selected by the algorithm have high attack damage and great movement speed, indispensable attributes for a Hard Engage composition. They also have abilities that improve their movement speed and attributes like armor and attack damage. Also the obligatoriness of having a Support champion (the fourth in Fig. 10) and an Attack Damage Carry (the second in Fig. 10) champion in the composition

Figure 10. Team generated using hard engage strategy.

was fulfilled.

The proposed approach can be seen as an option to be used in this context since most of the approaches and tools utilize prediction for suggesting teams. Moreover, a search-based approach with an adequate fitness function provides a high potential to satisfy even very hard constraints.

Overall, search-based techniques combined with an adequate fitness function can reach good solutions. Nevertheless, for achieving the best solutions in a huge search space the number of generations i.e., the number of iterations that the genetic algorithm will be executed should be large.

Search-based algorithms with few iterations have a premature convergence risk or impediments for the search to make substantial progress. It means that in some cases the adequate team for a specific strategy can not even be found. However, a high number of iterations leads the algorithm to high time and computational costs [21]. Thus, ideally, we must find a trade-off between the number of execution and a fitness value admissible.

B. *Efficiency of the proposed approach* (RQ_2)

To answer this research question, this second experiment computed the time average for each strategy using the best setting (st_3). Fig. 11 reports the results from a millisecond experiment based on time and number of executions.

We noticed in the results that there was a significant difference in the each strategy time, thus rejecting the null hypothesis ($H2_0$).

The results indicate that the poke strategy, on average, obtained the shortest time, 715 milliseconds, in the team generation. On the other hand, team fight strategy was the most time consuming with 794 milliseconds. Considering all strategies the sixteenth execution was the fastest one with the 471 milliseconds and the fourth was the slowest, with 908 milliseconds.

In a empirical context, the teams generated for fight strategy and hard engage strategy were adequate. They fit perfectly for their respective strategy and could be used in a game. The experiment using poke strategy didn't returned an usual composition, since there are four Attack Damage Carries and a normal team has only one.

VII. Conclusion

We have proposed an automated approach for team composition in League of Legends. Our approach consists of a Genetic Algorithm guided by different fitness functions to generate teams based on three game strategies. The fitness functions were inspired in real strategies utilized into the LoL and can be applied in training and championships.

Our approach works from a dataset of champions containing their information such as names, roles, attributes. Thus, the GA generates and improves iteratively the teams through fitness functions that assess the quality of them. The capability of metaheuristics as GA to solve high complexity problems makes possible to generate teams that satisfy any constraints and it from a huge amount of champions.

We evaluated our approach through an experiment to analyze the adequacy of teams generated from GA for each fitness function i.e., Hard Engage, Team Fight and Poke. We carry out three experiments considering the strategies of the game and computed the fitness value achieved and the execution time. In addition, We have also considered different GA parameters in an attempt to minimize execution costs, such as time and use of the CPU.

Summing up the results, it can be noticed that for all experiments the quality of teams achieved between 76 and 95 percent. It shows that a GA combined with an appropriate fitness function can raise the team composition for another level since the algorithm can suit any strategy. This can provide a valuable tool for players and mainly for coaches of LoL teams.

This study demonstrates the feasibility of an intelligent tool that can create and suggest teams in LoL, this suggestion can be generated by different features such as champions attributes which were the focus in our work or even combine skills of players with victory statistics.

Therefore, the present work may lead to the development of more robust approaches to assist in the selection process of teams. We employed our approach only for LoL game, however, we believe that it can be used to other MOBA games.

Future work is directed towards the following topics: *(i)* development of a tool in order to perform experiments with players and coaches; *(ii)* extend the approach to other MOBA games; *(iii)* improvement of the fitness functions; and *(iv)* experiment using different search algorithms such as Hill Climbing, Tabu Search and Particle Swarm Intelligence.

References

[1] NewZoo, "Most popular core pc games — global," Tech. Rep., 2019. [Online]. Available: https://newzoo.com/insights/rankings/top-20-core-pc-games/

[2] V. d. C. Oliveira, B. J. Placides, M. d. F. O. Baffa, and A. F. d. V. Machada, "A hybrid approach to build automatic team composition in league of legends," in Proceedings of the 16th Brazilian Symposium on Computer Games and Digital Entertainment, ser. SBGames'17, 2017, pp. 630–633.

[3] M. Gendreau and J.-Y. Potvin, Handbook of Metaheuristics, 2nd ed. Springer Publishing Company, Incorporated, 2010.

Figure 11. Average time according to st_3 for each strategy.

[4] G. Gao, A. Min, and P. C. Shih, "Gendered design bias: Gender differences of in-game character choice and playing style in league of legends," in Proceedings of the 29th Australian Conference on Computer-Human Interaction, ser. OZCHI '17. ACM, 2017, pp. 307–317.

[5] S. J. Russell and P. Norvig, Artificial Intelligence: A Modern Approach, 2nd ed. Pearson Education, 2003, chapter 4.

[6] T. Weise, Global Optimization Algorithms – Theory and Application, 1st ed., 2008.

[7] F. G. Teofilo, Handbook of Approximation Algorithms and Metaheuristics, 1st ed. Chapmann Hall/CRC Computer and Information Science Series, 2007. [Online]. Available: http://books.google.de/books? id=QK3$_VU8ngK8C$

[8] D. LaSalle and G. Karypis, "A parallel hill-climbing refinement algorithm for graph partitioning," in Proceedings of the 45th International Conference on Parallel Processing, ser. ICPP'16, 2016, pp. 236–241.

[9] K. Z. Zamli, B. Y. Alkazemi, and G. Kendall, "A tabu search hyper-heuristic strategy for t-way test suite generation," Applied Soft Computing, vol. 44, pp. 57–74, 2016.

[10] D. E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, 1st ed. Addison-Wesley Longman Publishing, 1989.

[11] J. H. Holland, "Genetic algorithms," Scientific American, vol. 267, pp. 66–73, 1982.

[12] O. Kramer, Genetic Algorithm Essentials. Springer, Cham, 2017.

[13] M. Scirea, J. Togelius, P. W. Eklund, and S. Risi, "Metacompose:acompositional evolutionary music composer," in Proceedings of the Evolutionary and Biologically Inspired Music, Sound, Art and Design, ser. EvoStar'16, 2016, pp. 202–217.

[14] D. Lessin and S. Risi, "Darwin's avatars: A novel combination of gameplay and procedural content generation," in Proceedings of the Genetic and Evolutionary Computation Conference, ser. GECCO'15, 2015, pp. 329–336.

[15] A. Monteagudo and J. S. Reyes, "Evolutionary optimization of cancer treatments in a cancer stem cell context," in Proceedings of the Genetic and Evolutionary Computation Conference, ser. GECCO'15, 2015, pp. 233–240.

[16] A. Sapienza, P. Goyal, and E. Ferrara, "Deep neural networks for optimal team composition," ArXiv, vol. abs/1805.03285,
pp. 1–10, 2018.

[17] K. E. Conley and P. D., "How does he saw me ? a recommendation engine for picking heroes in dota 2," pp. 1–4.

[18] C. Zhamri, Z. Che Ani, A. Yasin, M. Z. Husin, and Z. Hamid, "A method for group formation using genetic algorithm," International Journal on Computer Science and

[19] F. Zhang, X. Cao, and D. Yang, "Intelligent scheduling of public traffic vehicles based on a hybrid genetic algorithm," Tsinghua Science and Technology, vol. 13, no. 5, pp. 625–631, 2008.

[20] D. M. Deaven and K. M. Ho, "Molecular geometry optimization with a genetic algorithm," Phys. Rev. Lett., vol. 75, pp. 288–291, 1995.

[21] F. C. M. Souza, M. Papadakis, Y. Le Traon, and M. E. Delamaro, "Strong mutation-based test data generation using hill climbing," in Proceedings of the 9th International Workshop on Search-Based Software Testing, ser. SBST '16. ACM, 2016, pp. 45–54.

[22] C. Wohlin, P. Runeson, M. Host, M. C. Ohlsson, B. Regnell, and A. Wesslen, Experimentation in Software Engineering: An Introduction, 1st ed. Springer-Verlag Berlin Heidelberg, 2012.

[23] V. Basili and D. Weiss, "A methodology for collecting valid software engineering data." IEEE Transactions on Software Engineering, vol. 10, no. 6, pp. 728–738, 1986.