Developing Competitive Strategies for Legends of Runeterra using Genetic Algorithm

Ricardo A. Balbinot[1], Lincoln M. Costa[2], Alinne C. Corrêa Souza[1], Franscisco Carlos M. Souza[1]

[1]Federal University of Technology (UTFPR)
Dois Vizinhos – PR – Brazil

[2]Federal University of Rio de Janeiro (UFRJ)
Rio de Janeiro – RJ – Brazil

rab.balbinot@gmail.com, costa@cos.ufrj.br

alinnesouza@utfpr.edu.br, franciscosouza@utfpr.edu.br

Abstract. *Card games have long served as recreational tools across various cultures. This genre has also followed the trends of the digital gaming industry, with numerous games now featuring competitive scenes with substantial prizes, such as Legends of Runeterra. This paper proposes an automatic generator for competitive deck lineups in Legends of Runeterra, assessing extensive match databases using a Genetic Algorithm that evaluates candidate solutions based on three different generation strategies. This study introduces an evaluation function and presents empirical results concerning the effectiveness and efficiency of the approach. The findings indicate that the proposed method is useful and representative in a competitive context for all three addressed generation strategies.*
Keywords *Legends of Runeterra, Digital Card Games, Genetic Algorithm, Strategy Composition*

1. Introduction

Video games are integral to people's lives, and beyond their cultural and social significance, the video game industry is regarded as the foremost entertainment and media sector of the 21st century. According to an article by Valor Econômico [Laurence 2019], this industry is expected to generate approximately $187.7 billion in 2023, marking an annual increase of 2.6%. This growth has elevated games from mere leisure platforms to a medium of media and sports, giving rise to competitions with million-dollar prizes globally.

Card games are prevalent across various cultures, serving as leisure tools or even representations of cultural beliefs. The traditional deck of cards used today, featuring four suits, was invented by the Chinese in the ninth century and introduced to Europe by the Arabs towards the end of the fourteenth century. From there, numerous versions were developed across different cultures, with the French version becoming the most widespread and popularized globally. Today, card decks feature in various games and competitions worldwide, leading to the emergence of more unique card game versions.

Created in 1993 by Richard Garfield and published by Wizards of the Coast, Magic: The Gathering (MTG) was the first game to introduce the modern concept of a collectible card deck. Players battle with cards representing units and spells to deplete the opponent's life points in this game. While retaining the classic deck concept with sixty cards per match, it launched with about 300 distinct cards, each offering unique strategic possibilities. This added complexity and diversity of combinations not seen before in conventional 52-card decks.

Over time, classic collectible card games in physical formats, like MTG or the Yu-Gi-Oh! Trading Card Game, published by Konami in 1999, showed interest in transitioning into the digital gaming sphere, a much more manageable environment than printed cards. Besides these, exclusively digital card games have been created, such as Hearthstone and, in 2020, Riot Games released Legends of Runeterra (LoR), a digital card game set in the League of Legends universe.

Keeping pace with electronic gaming trends, competitions involving digital card games are becoming increasingly common. The formats of these tournaments vary, with the most well-known involving best-of-three matches with three different decks. LoR introduced its own three-deck format with unique rules for both deck construction and composition, creating a playing field with a myriad of possible combinations for players. In a segment where player skill and strategic synergies blend with the inherent luck of any card game, the challenge is to navigate this delicate balance, where maximizing synergistic advantages can be crucial in determining the outcome of a match.

In this context, combining three decks, known as a lineup, can be viewed as a scenario employing search-based or meta-heuristic techniques. Meta-heuristics are Artificial Intelligence (AI) methods that perform robust searches in a space with many solutions, utilizing local improvement procedures to find local optima within that scope. One of the primary challenges in achieving an appropriate lineup is to mix decks considering various aspects. The ideal goal is to find a combination of three unique decks that adhere to assembly restrictions and optimize their statistics to the maximum according to the strategy chosen by the player to minimize possible weaknesses.

This article introduces an automated proposal for generating competitive lineups in Legends of Runeterra by comparing each deck's individual and relative performance rates through meta-heuristic techniques. The Genetic Algorithm (GA) technique uses an evaluation function to iterate and select candidates among possible solutions.

The paper is organized as follows: Section 2 details the theoretical framework; Section 3 presents related works; Section 4 describes the proposed approach; Section 5 reports the research structure and experiments conducted; Section 6 analyzes the results obtained and discusses the benefits, relevance, and limitations of the proposed approach; and finally, Section 7 offers concluding remarks and discusses next steps.

2. Background

This section introduces the main concepts about Legends of Runeterra and GA that were considered in the proposed deck composition.

2.1. Legends of Runeterra

Legends of Runeterra (LoR) is a digital collectible card game developed and released by Riot Games, the same company behind League of Legends (LoL). Drawing heavy inspiration from Magic: The Gathering (MTG), LoR features characters from the LoL universe as its cards, along with their backgrounds, narratives, and characters that enhance their interconnectedness.

The game adopts the battle concept pioneered by MTG, where two players compete, and the winner is the one who reduces their opponent's life points to zero first. Each player can perform game actions turn by turn, but only one player may declare an attack each turn. The turn ends when both players have completed their actions. Players have three different types of cards at their disposal: units, spells, and landmarks. Each card has a cost to be played, paid with a resource known as mana. Each player starts with one mana, which increases by one with each turn [Games 2024].

Figure 1 illustrates a unit card, Miss Fortune. The blue value in the top left corner indicates the mana cost, accompanied by its attack value in the bottom left corner and health points in the bottom right corner. In the top right corner is an emblem corresponding to the region to which the card belongs. When two cards engage in combat, each inflicts damage equivalent to its attack value on the other card's health points. Spell cards and landmark cards also have a similar structure with mana cost and region displayed, but they do not have attack or health points and only produce various effects.

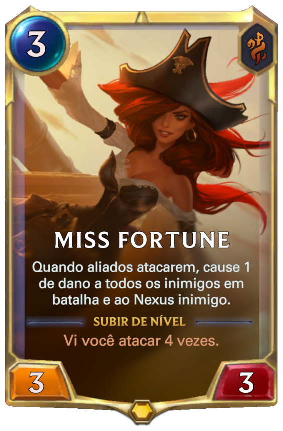
3
MISS FORTUNE
Quando aliados atacarem, cause 1 de dano a todos os inimigos em batalha e ao Nexus inimigo.
SUBIR DE NÍVEL
Vi você atacar 4 vezes.
3
3

Figure 1. Legends of Runeterra card [Games 2024]

The yellow pendant, centered at the card's bottom edge, signifies this is a champion card. Champion cards possess unique and powerful abilities, described beneath their name. They also come with a leveling condition that, once met, transforms the champion card into a stronger version of itself. Standard unit cards are generally weaker than champions and do not have a leveling condition.

A deck in LoR consists of 40 cards, with up to three copies of the same card allowed but not exceeding six champion cards. Another assembly condition involves the region emblem to which each card belongs. A deck cannot contain cards from more than two different regions. Champions are powerful cards, and the combination of champion cards and the areas of a deck often define the gameplay style it will adopt. This combination is also known as an archetype, an example depicted in Figure 2.

The concept of archetype is very important when discussing the competitive tournaments of the game. Riot Games has implemented a proprietary format for

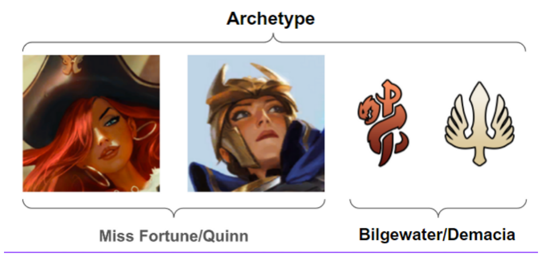

Figure 2. Representation of an archetype [Games 2024]

assembling a lineup of three distinct decks (or archetypes) called Riot Lock. This format proposes that players only combine decks where: (*i*) the champion present in one deck does not repeat in any other in the lineup, (*ii*) the same combination of regions from one deck does not repeat in the lineup, and (*iii*) the player does not have more than one deck without the presence of any champion card among the 40 cards.

In tournaments with the Riot lock rule, players must have three valid decks (d_1, d_2, d_3) and face their opponent in a best-of-three Conquest format. In this format, each player can ban an opponent's deck, preventing them from playing it in that matchup. After this, the format requires that each player must win against their opponent with their two remaining decks, as when they win using deck d_1, for example, that deck is also locked for the remainder of the matchup, and they must win their last match with deck d_2 if deck d_3 is banned by the opponent. This is the official competition format adopted by LoR in its official tournaments (continental and world). Considering all the possibilities of matchups and synergies that the format proposes, some strategies are used by competitive players to generate cohesive lineups. The main ones are:

- E_1 **By average win rate:** This involves manually selecting three individually strong archetypes based on recent performance, considering their average win rate;
- E_2 **By win rate excluding a fixed ban:** This strategy also considers the selection based on win rate. However, it disregards any matchup against a deck d_x, which corresponds to a deck the player always intends to ban;
- E_3 **By win rate focusing on a specific deck:** This involves the selection of three archetypes that have the best matchups against a specific deck. This strategy is common when there is a dominant archetype in the game, and the player wants to create a lineup that has a very high chance of defeating it, considering that with a high play rate, they would have a high chance of facing it even in a random spectrum.

Considering all the variables involved in the competitive process described above, the importance of optimizing the cohesive generation of a competitive lineup in an environment that already has other factors that can affect a player's victory, such as luck, becomes apparent. Thus, the restrictions for generating archetypes, restrictions on combinations between archetypes in competitive lineups, and the strategy of creation will be factors considered during this study's development.

2.2. Genetic Algorithm

Optimization problems tend to be complex due to their large search spaces and enormous number of potential solutions, making manual attainment impractical. Identifying such solutions involves finding an ideal solution, which may be considered a global optimum if it represents the best solution within the domain D; or a local optimum, when it is the best solution within a specific part of domain D [Weise 2008]. To handle many possible solutions, it is common to employ search-based techniques or meta-heuristics, such as well-known methods like hill climbing [LaSalle e Karypis 2016] and tabu search [K. Z. Zamli e Kendall 2016]. However, when seeking greater efficiency in an attempt to find global optimism in extensive search spaces, genetic algorithms may be the preferred choice.

Genetic Algorithms are a search technique based on the principles of evolution by natural selection proposed by Charles Darwin [Darwin 2003]. This method aims to discover a better solution for a given problem by iteratively analyzing a set of solutions known as a population. The workflow of a GA is characterized by several main steps. The algorithm begins with a population of candidate solutions to the problem, typically generated randomly. Each individual is then assessed by a fitness (or objective) function, where, based on the evaluation parameters, the fittest are selected to reproduce and create a new generation of candidate solutions. This process is repeated until the algorithm reaches a predefined number of generations or an optimal evaluation point.

For reproduction to occur, the technique involves two processes: i crossover and ii mutation. Crossover involves taking two parent individuals and generating new offspring from their characteristics. This process disseminates the traits of the best individuals to subsequent generations to evolve them. Mutation, on the other hand, occurs afterward, aiming to increase the diversity of the new generation by altering a random element of the generated individual. Moreover, mutation attempts to prevent the population from prematurely converging on optimal solutions and leaving unexplored gaps in the search space [Kramer 2017].

The techniques and processes involved in Genetic Algorithms are straightforward and offer significant flexibility regarding the size of the search space and the evaluation of individuals. These factors have contributed to the popularity of the method and its application across various research fields, such as graphic arts [Egan e Picton 1994], musical composition [M. Scirea e Risi 2016], and procedural generation in digital games [Mitsis et al. 2020].

3. Related works

After searching sources such as IEEE, ACM Digital Library, Springer, and Google Scholar, only the latter returned papers related to the game Legends of Runeterra. However, these papers focused on something other than technological implementations, which is an analysis of the translation of the card texts into different languages. The study proposed by chu examines the quality of linguistic and cultural techniques used in translating the names of the cards from English to Spanish. Meanwhile, a study from an Indonesian university by hasibuan explores similar techniques for translation from English to Japanese.

Expanding the search to other digital card games, relevant studies were found related to MTG, previously mentioned, as well as Hearthstone. In the located studies, costa proposed implementing a Genetic Algorithm with a structure similar to that proposed in this paper, but for team compositions within the League of Legends.

The study of chia proposed a solution using Genetic Algorithms in conjunction with the Monte Carlo method applied to the game Hearthstone, with the implementation carried out on an external simulator where the proposed solution analyzed the game board and made the next move for the player. Using the Monte Carlo method and search techniques, ward proposed a solution for real-time move selection, aiming to create a bot for the MTG game.

Many of the identified studies dealt with strategies for creating decks and defining the cards composing the deck. The authors chen and garcia propose very similar solutions for deck generation with Genetic Algorithms for the game Hearthstone, using the same simulator employed by chia for result validation. The work of yang also uses Genetic Algorithms for deck creation but focuses on the game MTG, proposing three algorithm variations and testing their effectiveness.

Unlike previous works, betley utilized the DBSCAN algorithm for victory prediction based on different archetypes of the game Hearthstone. Although they use techniques similar to those proposed in this paper, Legends of Runeterra differs from other digital card games due to its unique deck and lineup generation restrictions.

Statistical information is not readily available to all players. Therefore, experienced players generate competitive lineups manually, as it requires a high empirical knowledge of the game. The manual process starts with a solution based on that player's comfort and is validated against another player in an isolated environment. It undergoes changes as it covers different matchups, making it a slow and tiring process. In our study, an automated proposal for generating competitive lineups for the game Legends of Runeterra was developed, combining the required lineup assembly strategies, statistics extracted from each archetype with Genetic Algorithm techniques, guided by an objective function. The differences between the proposal in this paper and those discussed in this section are presented in Table 1.

Table 1. Relationship between the proposed approach and related work

Studies	Technique	Application	Game
[Chia et al. 2020]	GA/Monte Carlo	Make moves automatically	Hearthstone
[Chen et al. 2018]	GA	Deck generation	Hearthstone
[García-Sánchez et al. 2016]	GA	Deck generation	Hearthstone
[Betley et al. 2018]	DBSCAN	Victory prediction	Hearthstone
[Yang et al. 2021]	GA	Deck generation	MTG
[Ward e Cowling 2009]	Monte Carlo	Make moves automatically	MTG
Our approach	GA	Competitive lineup generation	LoR

4. Proposed Approach

The approach presented in this work proposes to generate competitive lineups involving different archetypes from Legends of Runeterra. This approach utilizes a genetic algorithm to evaluate various solutions through an objective (or fitness) function.

Currently, no technical approaches are dedicated to composing lineups that Legends of Runeterra (LoR) players can utilize. This task is carried out manually and is largely based on empirical knowledge of the game or each player's comfort zone. Only highly experienced players with a comprehensive understanding of all the game's cards can create concise lineups with significant results. Less experienced players often replicate lineups that have defeated them or those made by well-known, skilled players.

There is a natural tendency among players to adopt strategies crafted by another player. However, this can be risky, as it may involve a strategy or adaptation that is highly personal to that player and may not perform equally well when used by different players. This risk underscores the need for a more personalized and adaptable approach to lineup generation. Additionally, comprehensive statistical data about the archetypes are not readily accessible, which further limits the creative process for both novice and veteran players.

Thus, in addition to presenting a proposal that includes more comprehensive statistical methods to determine the quality of a lineup, this approach will provide players of various skill levels with a support tool. This tool will generate competitively relevant lineups optimized by an AI technique, empowering players to compete at a higher level regardless of their experience. Following this objective, the results of the experiments in the Results and Discussion Section were also analyzed based on topics discussed in the work of [Costa et al. 2023], which examines human-centered AI, focusing on how the results generated by the proposed approach can help players prepare for the competitive scene in LoR without having to copy lineups based on personal preferences and/or seek hard-to-access individual information for each archetype.

As depicted in Figure 3, the proposal involves using a database with many matches involving various archetypes over a period p_x of time, also referred to as a patch. A game patch refers to the period between two game balance updates, whether by adding new cards or balancing existing ones. This period typically ranges from two to three months. For this study, all matches found between September and October 2023, involving only the highest-ranking players' matches, will be considered. These matches are derived from the public database of the game's owner, Riot Games, with the search process explained in the following section.

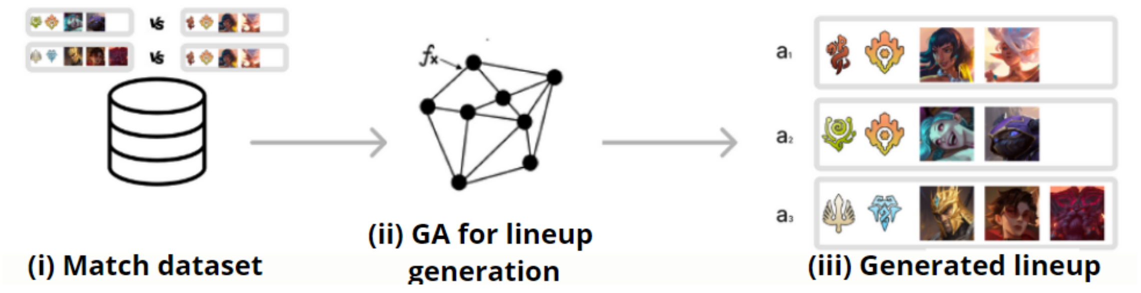

Figure 3. Approach for competitive lineups generation

Once the database is established, it will undergo preprocessing to identify the archetypes used by both players in each match. Subsequently, all archetypes will be grouped by calculating the win rate and play rate within the defined search space. This preprocessed database will then be subjected to the GA, which will cross-reference these data to generate an optimized lineup for competitive tournaments.

4.1. Database definition

As it is a game released in 2020, there are few applications and scientific studies focused exclusively on LoR, as seen in the previous section on related works. This means that there are no public databases available for studies or implementation of functionalities external to the game, with the only resource available to developers being a public API from Riot Games (*developer.riotgames.com/apis*). However, the API does not solve the problems by itself, as the functionalities intended for LoR involve endpoints that return only the last twenty matches of a given player, making it difficult to formulate a database for analysis.

To build a complete and representative database, a *TypeScript* application with *Node.js* was developed as a support activity for this work, intended for collecting LoR match records. Given the limitations encountered in finding a complete match history, the application operates on a continuous search strategy for matches. This search started, initially, from a top-ranked player J_1. The public API is queried for the last twenty matches of this player, and the application adapts by attempting to find matches of all discovered players J that have not yet been saved, in addition to discovering and saving new opponent players found in that iteration.

By creating a routine of periodic execution, the application was able to collect 1,146,966 matches within the highest skill ranking of LoR. This was further enhanced by the generous contribution of another independent developer from the game community, who provided data related to matches from the month of September, thereby enriching the quality of this study.

In each match between players J_1 and J_2, we have, in addition to player details, noteworthy information for this work: *(i)* list of J_1's cards encoded in a hash; *(ii)* list of J_2's cards encoded *(iii)* who won the match J_1 or J_2. The number of matches in the current patch in the database for analysis is **1,146,966 games**. We carried out two activities to prepare this database for the execution of the GA: a_1 Python code that decodes the card lists of each player and defines the archetype of that list based on the concept described in Section 2.1; a_2 Python code to group the data by archetypes, defining their respective win rates and match play rates in that base.

Activity a_2 above will be specific to the three strategies described in Section 2.1. Strategy E_1 proposes generating lineups by average win rate. Therefore, it will be based on pre-processing that considers the win rate of that archetype across the entire match base. In strategy E_2, the generation of the lineup is based on the right that the *Riot lock* format gives the player to ban an archetype from their opponent, considering in the pre-processing the win rates of the archetypes only in matches that do not involve the banned archetype. The last strategy, E_3, aims for a matchup advantage against a specific archetype, therefore calculating the win rates of all archetypes only considering the matchups against the target archetype.

4.2. GA for lineup generation

Considering the large number of possible combinations and the level of constraints imposed for the generation of archetypes and lineups, the choice of GA for manipulating and generating results proves to be a viable choice [Kramer 2017]. The execution starts by selecting the previously pre-processed data depending on the strategy for generating the lineup. After that, the GA will start its optimization until it reaches the maximum number of generations defined, as described in the Pseudo-code 1.

Algorithm 1 Genetic Algorithm for Lineup Optimization

```
0: Initializes lineup population
0: Evaluate the created lineups
0: while not reach number of generations do
0:     Select best individuals
0:     Apply crossover and mutation
0:     Evaluate new population
0: end while
0: return Best solution found =0
```

Figure 4 illustrates how individuals' crossover and mutation stages work. The crossover is performed by a pair of parent individuals ap_i and ap_j, where each pair will enter the crossover function twice, swapping their order after the first iteration, thus generating two new offspring per pair of parents. The definition of the offspring is given by: ap_i in the first position; ap_j in the second position; and ap_i or ap_j in the third position (50% chance). After that, the mutation function will be applied to a random position of the individual, respecting the probability relative to the provided mutation rate m.

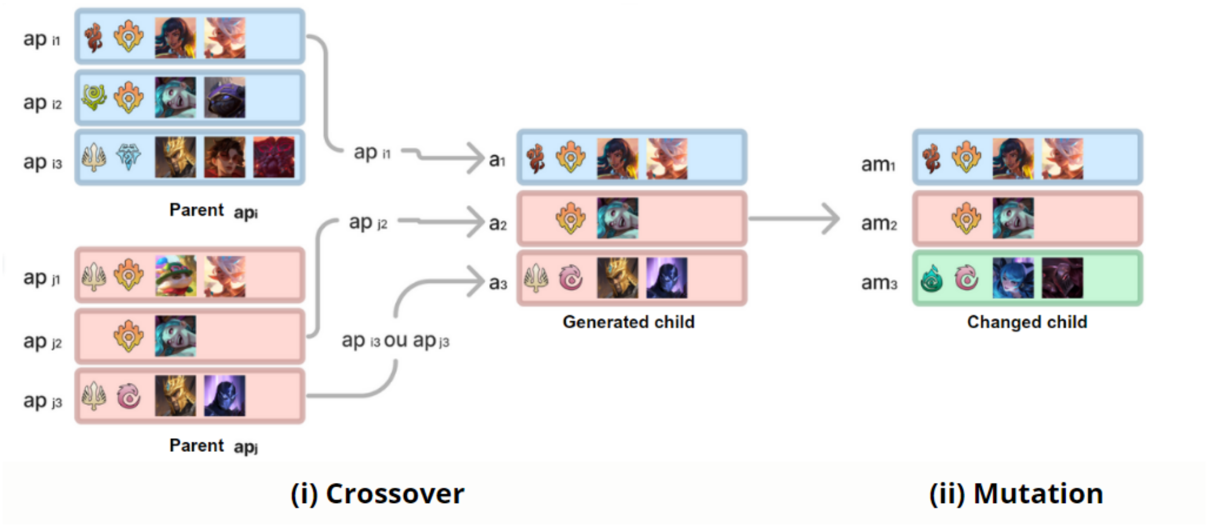

Figure 4. Crossover and mutation steps

A mathematical equation was generated to guide the crossover and mutation processes in the generations of individuals and to be used as a fitness function to evaluate the candidate solutions. To achieve the objectives of the GA, the Python programming language version 3.11.6 was used for its construction, along with the Pandas library for database manipulation. Other auxiliary Python modules were used for data manipulation and the definition of the fitness function, such as the *csv*, *json*, *math*, and *random* modules.

4.3. Fitness function

An objective function suitable for the problem is the most crucial step in search problems. This is because the entire optimization process is based on it, being the main factor associated with the outcome obtained. The function should always be created with the problem the proposal aims to solve, as it will be responsible for evaluating the candidate solutions [F. C. M. Souza e Delamaro 2016].

Unlike what was used by [Chen et al. 2018] and [García-Sánchez et al. 2016] in the game Hearthstone, LoR currently does not have a way to simulate results between matches involving completely random archetypes or card lists. For this reason, the database used in this work and its fitness function was created based on the historical performance of that archetype in a spectrum of recent matches, cross-referencing its statistical information to define the value of each solution.

It is essential to explain the function used to describe the concept adopted in the game as a metagame. This concept reflects the dominant archetypes at a given moment within the game. There are several empirical ways to define the metagame. The approach adopted by this work is based on the value referring to the play rate of the archetypes, thus being a list of the top twenty archetypes in this regard.

The functions are composed of two win rate variables, TV and TVM. The first refers to the win rate value of that archetype in a general spectrum. At the same time, TVM reflects the win rate of an archetype against the metagame archetypes, which better represents an archetype's strength against the most played ones. In the equations, the average win rates of the three lineup archetypes are normalized based on TV_{base}, which is set to 45 in the experiment. This is because a margin of error of 5%, for both higher and lower, was used to define the balance between opposing win rates. Therefore, average values below 45% will assume negative values and be penalized in their evaluation.

The play rate TJ is given by the base two logarithm ($log_2 n$) of the average play rate of the three archetypes. This strategy was used for an enhanced evaluation of lower play rates, such as those below 1%. These TV, TVM, and TJ rates acquire different weights for f_1 and f_2. f_1 uses 30% weight for TV, 50% weight for archetype performance against the metagame (TVM), and 20% for TJ. f_2, used by strategy E_3, changes these weights to give greater relevance to the TV variable, as it will represent the performance against the archetype it aims to face.

Based on the different lineup generation strategies (E_1, E_2, and E_3), two different fitness functions were created. Strategies E_1 and E_2 use the same fitness function f_1 below:

$$TV = \left(\frac{TV_1 + TV_2 + TV_3}{3} - TV_{base}\right) \times 0.3 \tag{1}$$

$$TVM = \left(\frac{TVM_1 + TVM_2 + TVM_3}{3} - TV_{base}\right) \times 0.5 \tag{2}$$

$$TJ = \left(\log_2\left(\frac{TJ_1 + TJ_2 + TJ_3}{3}\right) \times 0.2\right) \tag{3}$$

$$f_1 = (TV + TVM + TJ) \tag{4}$$

While the E_3 strategy will use the f_2 fitness function, which has different weights for the TV and TJ variables:

$$TV = \left(\frac{TV_1 + TV_2 + TV_3}{3} - TV_{base}\right) \times 0.5 \tag{5}$$

$$TVM = \left(\frac{TVM_1 + TVM_2 + TVM_3}{3} - TV_{base}\right) \times 0.4 \tag{6}$$

$$TJ = \left(\log_2\left(\frac{TJ_1 + TJ_2 + TJ_3}{3}\right) \times 0.1\right) \tag{7}$$

$$f_2 = (TV + TVM + TJ) \tag{8}$$

5. Experimental Study

We conducted an experiment to analyze and evaluate the effectiveness of lineup generation. We are interested in measuring the effectiveness of the fitness value and computational cost in time. This study used the guidelines recommended by Wohlin12. The experiments were executed on a MacBook with an Apple M1 chip, 8GB of RAM, and the MacOS operating system.

5.1. Experiment Definition

We used the Goal-Question-Metric (GQM) model [Basili e Weiss 1984] 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 computational time** from the point of view of **experimenters** in the context of **lineup composition in Legends of Runeterra**"*. For achieving the goal, we seek to investigate the following Research Questions (RQs):

RQ_1: **How effective is the proposed approach for generating competitive lineups in Legends of Runeterra based on the chosen strategy?** The effectiveness of the approach was measured through the fitness function during lineup composition. The fitness value was computed for each configuration $config$ present in Table 2. This experiment was executed 30 times for each combination of $config$ with E, where the average fitness value of the results was recorded.

RQ_2: **How efficient is the proposed approach for generating competitive lineups in Legends of Runeterra based on the chosen strategy?** The method chosen to measure efficiency was execution time. This time was computed based on the $config$ that obtained the best fitness value considering the three generation strategies. This experiment was executed 30 times for each configuration $config$ present in Table 2, with the average execution time computed in milliseconds.

RQ_3**: How representative are the results of the proposed approach for generating competitive lineups in Legends of Runeterra?** The representativeness of the main generated lineups was measured empirically. The presence of the archetypes generated by the three strategies was checked based on the $config$ that obtained the best fitness values in the game's largest competitive tournament, which took place on October 29, 2023.

5.2. Experiment Design

We conducted empirical experiments to address the RQs above, manipulating the following variables:

- **Strategy** (E): representing the different strategies for creating lineups, which can be focused on average performance, performance by banning a fixed deck, or aiming to counter something specific;
- **Number of solutions** (P): number of candidate solutions in the experiment. where the number of solutions P is represented by 3 values: (P = 10, 20, 30);
- **Mutation rate** (TM): probability of changing a gene of the generated child, being used (TM = 0.3, 0.5, 0.7);
- **Generations number** (G): number of iterations that the GA will execute, using the parameters (G = 10, 50, 100);

We defined three configurations ($conf$ = $conf_1$, $conf_2$ and $conf_3$) to execute the experiments. These configurations were used for each E experiment, referring to three combinations of the above parameters, as seen in Table 2.

Table 2. Representation of experiment configurations

Config	P	TM	G
$conf_1$	10	0.3	10
$conf_2$	20	0.5	50
$conf_3$	30	0.7	100

The results obtained in the previous experiments were used to address RQ3. For this, we made an empirical comparison with the archetypes in the final phase of the continental LoR tournament (Runeterra Open) held on October 29, 2023. These data are publicly available on the website of one of the independent community developers (`https://lor.dne.to/`).

Given that the different strategies use different proportions to formulate their fitness value and employ different crossovers between the base matches, such as removing an archetype a_1 from the equation (E2) or focusing solely on the matchup table of an archetype a_2 (E3), a maximum achievable fitness value was defined for each strategy to provide a comparison point between them.

To define this value, the fitness function was applied using the data of the archetype with the highest average win rate ($max(TV)$), the best archetype regarding win rate against the metagame ($max(TVM)$), and the leading archetype in play rate

($max(TJ)$). We defined these values based on the pre-processed data for each strategy, and the respective weights for their fitness functions were applied. The maximum fitness values for comparison are: (i) E1: 10.2603; (ii) E2: 11.499; and (iii) E3: 13.626.

To execute the proposed experiments, a subdivision within the concept of archetype was created: the competitive archetype. Interviews with the top-ranked Brazilian players revealed that since each archetype can have several minor variations in the 40 cards that make up the deck, varying according to the player's personal preference, a minimum play rate TJ must be considered to define a diversified and significant archetype, set at $TJ \geq 0.1\%$ of the total matches. This value helps to explain the pre-processed archetypes that will be used to initialize the GA population.

5.3. Procedure of Experiment

The procedure followed during the execution of the experiment consists of the following steps:

1. Generating different competitive lineups as experimental subjects;
2. Evaluating each generated lineup by GA. The fitness function set directly indicates the quality of the generated lineup;
3. Computing the fitness value for each lineup generated by GA;
4. Computing the time in seconds for each lineup obtained from the GA;
5. Comparison between individual and relative performance rates of each deck obtained from the GA;
6. Performing an empirical analysis to identify the representativeness of the main generated lineups.

6. Results and Discussions

The results of the experiments are discussed according to each RQ.

6.1. Effectiveness of the proposed approach (RQ_1)

In this research question, the average fitness values were computed according to each configuration and generation strategy. The analysis of the generation strategies comprised the average of the best fitness and execution time after 30 different generations for each defined configuration and for each proposed strategy.

According to the variables used in the fitness function, an optimal solution is not just the one that presents the best average win rate. A good competitive lineup also needs to have a good win rate against the metagame, which are the most likely opponents in a real competitive scenario. The play rate also needs to be considered, as decks played more frequently have a larger and more reliable sample size. All these factors need to be used together to avoid false positives in the ideal solutions.

In Figure 5, the lines represent the progression of the best fitness obtained in each generation for the three strategies using $conf_3$. During the experiment, we observed that strategy E_2 achieved the best percentage, reaching 91.39% of fitness value in its best execution. This graph highlights an interesting point regarding the parameter G related to the number of generations, as in rare cases, the best value was found before $G = 50$. Still, there are fragile values related to G = 10.

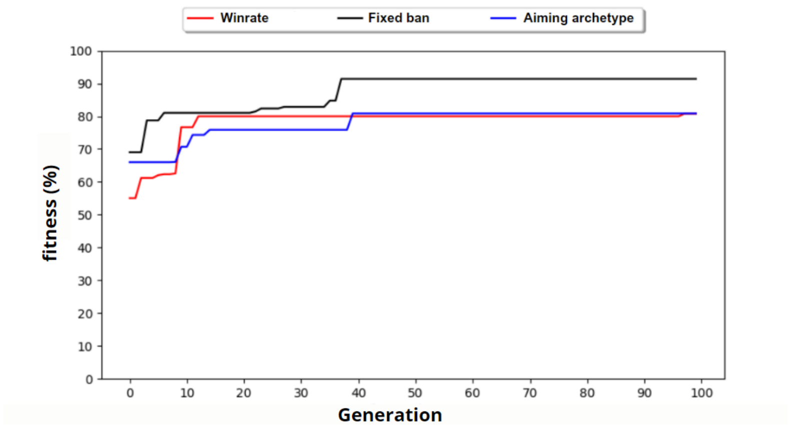

Figure 5. Fitness value achieved by each strategy using the configuration $conf_3$

Figure 6 shows considerable differences in the effectiveness of their fitness value relative to the optimal value using the configurations. It can also be observed that higher numbers of generations G improve the quality of the result, reaching a value of 88.10% by strategy E_2 with configuration $conf_3$, compared to 67.30% effectiveness in the value of strategy E_1 with $conf_1$. This also shows that $conf_3$ was by far the most effective. It achieved values of 80.18% by strategy E_1, 88.10% by strategy E_2, and 79.72% by strategy E_3.

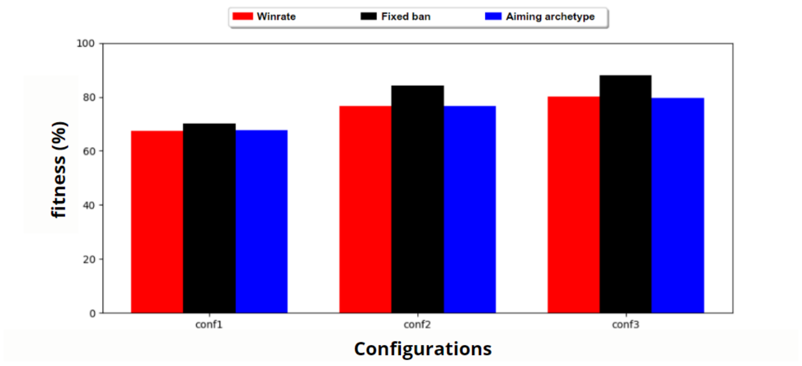

Figure 6. Configuration average fitness for each strategy

The results are consistent with the expectations regarding the effectiveness of the configurations, showing good fitness performance for G ¿= 50. The lower fitness performance of strategies E_1 and E_3 compared to E_2 suggests that a more refined fitness function for these strategies might yield even better results. Additionally, the low performance of early generations might be linked to a premature convergence problem, preventing the progress of the solution. However, Figure 5 shows that many generations may need to provide the expected benefits. Therefore, the correct choice of configurations that balance the cost-benefit between time and results is of utmost importance.

6.2. Efficiency of the proposed approach (RQ_2)

In this RQ, we computed the average execution time for the three strategies using the parameters of $conf_3$. Figure 7 presents the results of this experiment. Each execution on the X-axis corresponds to the average execution time of the approach over 30 different runs. Analyzing the results, the variation between the average values after the 30 execution cycles was negligible, less than four milliseconds. The results also show that Strategy E_3 used more processing time, despite the low variation, with an average of 191.94 milliseconds. The lowest average processing time was for Strategy E_2, with 188.24 milliseconds. Considering the executions of all strategies, the maximum execution time in one cycle was achieved by E_3, with 196.27 milliseconds.

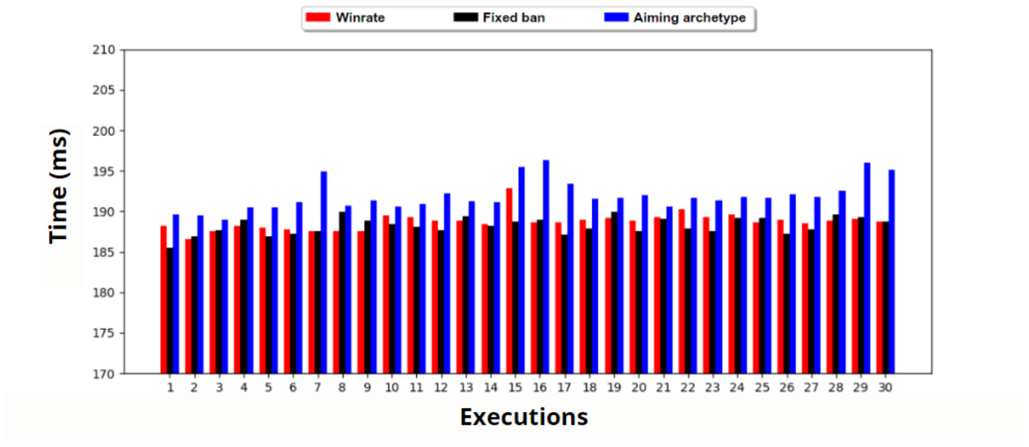

Figure 7. Average execution time using $conf_3$ for each strategy

The time required to define a lineup manually is relative. It will follow two main paths: (i) defined personally by empirical knowledge or (ii) copying the strategy of a well-known player in the competitive scene. Approach (ii) is more straightforward but risky, as that strategy will be adapted to the play style of the person who created it. Since everyone in strategy cannot need help finding play rate and win rate information (i), the player must validate their solutions against another high-level player. This process can take several hours or even a few days.

In an empirical analysis of the lineups generated in this experiment, the generation strategies by win rate (E_1) and aiming for a fixed ban (E_2) presented an adequate and competitively viable result. Since it is not an exact solution in all executions, some outputs of Strategy E_3 may be less effective in high-level tournament scenarios, as their advantages tend to perform better against a specific archetype and not necessarily as well against other strong decks.

6.3. Representativeness of the proposed approach (RQ_3)

The approach generates three competitive lineups, one for each strategy. Strategies E_2 and E_3 used the Janna/Nilah archetype (Bilgewater/Piltover), representing over 135,000 games in the current database. In E_2, the strategy is to remove matchups against this archetype, while in E_3 it aims to exploit its weaknesses.

Analyzing Figures 8 and 9, we can see very close results, as the Jinx/Kennen (Bandle City/Piltover) and Shen/Jarvan IV (Demacia/Ionia) archetypes are very consistent options. The exciting feature is the single substitution made in Figure 9, a great alternative significantly benefiting from the banned option.

Conducting an empirical analysis of the data from the continental Runeterra Open tournament held on October 29, 2023, available at (https://lor.dne.to/), it can be

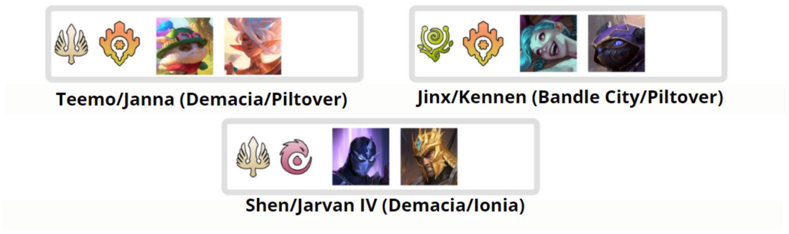
Teemo/Janna (Demacia/Piltover)
Jinx/Kennen (Bandle City/Piltover)
Shen/Jarvan IV (Demacia/Ionia)

Figure 8. Lineup generated by E1 strategy

Miss Fortune/Quinn (Demacia/Bilgewater)
Jinx/Kennen (Bandle City/Piltover)
Shen/Jarvan IV (Demacia/Ionia)

Figure 9. Lineup generated by E2 strategy

seen that the Miss Fortune/Quinn archetype (Demacia/Bilgewater) is the second most present among the Top 64 players, being used by 22 of the 64 players. This shows the effectiveness of strategy E_2, as it is an excellent option by removing the Janna/Nilah archetype (Bilgewater/Piltover), which had the highest presence in the tournament, being used by 36 of the 64 players. All presence indices are available in Table 3.

Table 3. Representation of archetypes in the continental tournament

Archetype	No. of players using
Miss Fortune/Quinn (Demacia/Bilgewater)	22
Shen/Jarvan IV (Demacia/Ionia)	11
Teemo/Jannah (Demacia/Piltover)	6
Galio/Udyr (Demacia/Freljord)	4
Teemo/Caitlyn (Freljord/Piltover)	3
Jinx/Kennen (Bandle City/Piltover)	2

The analysis of Table 3 underscores the significance of the generated lineups, which were found to be highly relevant among the top players during the largest competitive tournament of the studied period. Notably, only one archetype returned by strategy E_3 was not utilized among the top 64 players of the period.

The most relevant empirical analysis in RQ3 is that, besides the approach generating solid and consistent options for strategies E_1 and E_2, 2/3 of the archetypes generated by strategy E_3 were used in the tournament. This is justified by the natural tendency to have many appearances of the Janna/Nilah archetype (Bilgewater/Piltover), and these two archetypes being favorable. The data also shows a trend in the distribution of strategies, divided between (i) the most robust strategy and (ii) strategies that attempt to exploit its weaknesses.

Our analysis revealed that certain archetypes, although not prevalent in the October Runeterra Open statistics, could potentially be harnessed in a new lineup generation strategy. These archetypes, while characterized by weak overall statistics and low representation in the number of matches, perform exceptionally well in the hands of

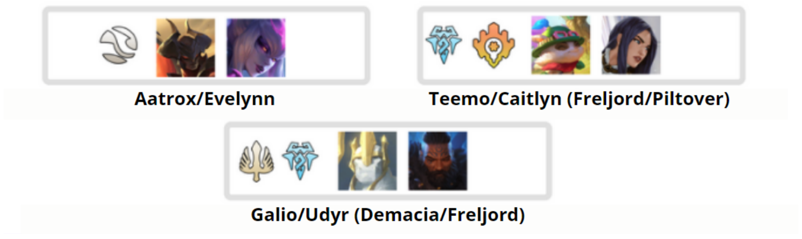
Aatrox/Evelynn
Teemo/Caitlyn (Freljord/Piltover)
Galio/Udyr (Demacia/Freljord)

Figure 10. Lineup generated by E3 strategy

specific players. This discovery paves the way for a new strategy: one that is based on the player's personal preferences. In this approach, the weight of archetypes that the player excels with would carry more influence than their general statistics, offering a promising avenue for future exploration.

The strategy exceeded expectations in the empirical analysis of competitive representativeness. Apart from extremely strong options present in the E_1 and E_2 generations, extreme competitive environments, such as the final stages of continental tournaments, tend to consist of several strictly personal options adapted to the player's specific playing style, often deviating from the natural metagame choices. However, even in this scenario, the approach proved effective in all three proposed strategies.

7. Conclusion

This work proposes an automatic approach for generating competitive lineups in Legends of Runeterra. The proposal involves a Genetic Algorithm guided by two fitness functions to create combinations of archetypes with three different generation strategies. We developed the fitness function based on the analysis of recent match data. Since we do not have any tool available for simulating matchups, we must use search strategies in past search spaces to predict and optimize better combinations.

We evaluated the approach through experiments to analyze how cohesive the GA generations were for average win rate, fixed ban, or focusing on an opponent's strategy. Experiments were conducted, divided into analyses of fitness values and the algorithm's execution time and an empirical analysis of the compatibility of the generated lineups with real tournaments that took place during the studied period.

This study demonstrates the flexibility of using AI techniques for solving search and optimization problems, opening up possibilities for various other strategies that can be used to further assist LoR players and other games in the genre, such as using statistics and comfort zone choices of each player to generate optimized lineups for them.

As a pioneer in the field of research and technical implementation, particularly in the context of Legends of Runeterra, this work has the potential to significantly impact the gaming community. By harnessing the vast creative potential inherent in the subject, our approach, tailored to the specific constraints of LoR, can pave the way for future innovations. Moreover, it can be adapted to suit other digital card games, thereby extending its reach.

Future implementations related to the topic are listed below: (i) development of a public website for using lineup generation tool; (ii) modifying the database and fitness functions to consider choices adapted to the player's style; (iii) improving the fitness functions; (iv) comparing the proposed approach with other search strategies, such as PSO and tabu search; and (v) extending the GA approach to generate all the 40 cards of an archetype.

References

Basili, V. e Weiss, D. (1984). A methodology for collecting valid software engineering data. *IEEE Transactions on Software Engineering*, 10(6):728–738.

Betley, J., Sztyber, A., e Witkowski, A. (2018). Predicting winrate of hearthstone decks using their archetypes. In *2018 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pages 193–196.

Chen, Z., Amato, C., Nguyen, T.-H. D., Cooper, S., Sun, Y., e El-Nasr, M. S. (2018). Q-deckrec: A fast deck recommendation system for collectible card games. In *2018 IEEE Conference on Computational Intelligence and Games (CIG)*, pages 1–8.

Chia, H.-C., Yeh, T.-S., e Chiang, T.-C. (2020). Designing card game strategies with genetic programming and monte-carlo tree search: A case study of hearthstone. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 2351–2358.

Costa, L. M., Drachen, A., Souza, F. C. M., e Xexéo, G. (2023). Artificial intelligence in moba games: A multivocal literature mapping. *IEEE Transactions on Games*, pages 1–23.

Darwin, C. (2003). *A Origem das Espécies*. Editora Hermus, São Paulo.

Egan, T. e Picton, P. (1994). Behaviour of a simple genetic algorithm searching for bright and edge pixels in an image. In *IEE Colloquium on Genetic Algorithms in Image Processing and Vision*, pages 2/1–2/6.

F. C. M. Souza, M. Papadakis, Y. L. T. e Delamaro, M. E. (2016). *Strong mutation-based test data generation using hill climbing,*. Proceedings of the 9th International Workshop on Search-Based Software Testing, ser. SBST '16. ACM.

Games, R. (2024). Legends of runeterra.

García-Sánchez, P., Tonda, A., Squillero, G., Mora, A., e Merelo, J. J. (2016). Evolutionary deckbuilding in hearthstone. In *2016 IEEE Conference on Computational Intelligence and Games (CIG)*, pages 1–8.

K. Z. Zamli, B. Y. A. e Kendall, G. (2016). *A tabu search hyper-heuristic strategy for t-way test suite generation*. Applied Soft Computing, vol. 44.

Kramer, O. (2017). Genetic algorithm essentials. *Springer*.

LaSalle, D. e Karypis, G. (2016). *A parallel hill-climbing refinement algorithm for graph partitioning,*. Proceedings of the 45th International Conference on Parallel Processing, ser.ICPP'16.

Laurence, F. (2019). Receita de games deve atingir us 187 bi no mundo em 2023.

M. Scirea, J. Togelius, P. W. E. e Risi, S. (2016). *Metacompose: acompositional evolutionary music composer*. Proceedings of the Evolutionary and Biologically Inspired Music, Sound, Art and Design, ser.EvoStar.

Mitsis, K., Kalafatis, E., Zarkogianni, K., Mourkousis, G., e Nikita, K. S. (2020). Procedural content generation based on a genetic algorithm in a serious game for obstructive sleep apnea. In *2020 IEEE Conference on Games (CoG)*, pages 694–697.

Ward, C. D. e Cowling, P. I. (2009). Monte carlo search applied to card selection in magic: The gathering. In *2009 IEEE Symposium on Computational Intelligence and Games*, pages 9–16.

Weise, T. (2008). Global optimization algorithms – theory and application, 1st ed.

Yang, Y., Yeh, T., e Chiang, T. (2021). Deck building in collectible card games using genetic algorithms: A case study of legends of code and magic. In *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 01–07.