Using Monte Carlo Tree Search and Google Maps to improve Game Balancing in Location-based Games

Luís Fernando Maia  
Federal Institute of Education - IFMA  
Caxias-MA, Brazil  
Email: luis.maia@ifma.edu.br

Windsor Viana  
GREaT - MDCC  
Federal University of Ceará - UFC  
Fortaleza-CE, Brazil  
Email: windson@great.ufc.br

Fernando Trinta  
GREaT - MDCC  
Federal University of Ceará - UFC  
Fortaleza-CE, Brazil  
Email: fernandotrinta@great.ufc.br

Abstract—Location-Based games (LBGs) are a subtype of digital games that uses the location of players as a key component for playability, including changes to the game state. However, a significant challenge that threatens the development and popularization of LBGs is the game balancing. Since LBGs rely on players’ location, it is hard to manually design interactions, challenges, and game scenarios for each part of the world. Thus, the same LBG is likely to present varying difficulty levels depending on the player’s location due to differences in terrain, distance, and transport availability. As a result, even modern LBGs show huge balancing differences between regions and they do not explore competition between players like other game genres. In this paper, we present measurements to estimate game balancing in modern LBGs and introduce a method that uses Monte Carlo Tree Search (MCTS) to automatically edit instances of these games to minimize differences in game balancing. Additionally, we present a study detailing the improvements in game balancing when using the proposed method in today’s two most popular LBGs (Ingress and Pokémon Go).

Keywords—Location-based Games, Game Balancing, Monte Carlo Tree Search, Google Maps.

I. INTRODUCTION

Mobile devices are best known for their mobility and for providing quick access to the Internet. However, smartphones have impacted the way people play games and have boosted the game industry due to its immense potential as a gaming platform. Previously, games were played in specific places using devices such as keyboard, mouse, joystick, PCs, and consoles [1]. Nowadays, smartphones have enabled games to be played anytime and in multiple locations. Consequently, many games have been ported or developed specifically to target the mobile market. To illustrate the importance of mobile devices to the game industry, the Global Mobile Game Confederation has released a report showing that mobile games are expected to hold 38% of the game market in 2017 [2].

Additionally, the increase in processing power and the popularization of sensors in modern smartphones (e.g., accelerometer, compass, camera, GPS tracker, gyroscope, among others) have allowed the implementation of new game genres, such as Pervasive Games. These games use smartphones’ sensors to infer contextual information about the player’s location. Pervasive Games are capable of providing a mix of virtual and real environments by using mobile, ubiquitous, and embedded digital technologies [3]. This research focuses on a popular subtype of Pervasive Games called Location-Based Games (LBGs). LBGs use the players’ location to modify the game state during runtime. As a result, players have to physically move to progress in the game, thus creating a link between virtual and real worlds.

Lately, both industry and academia have focused efforts in the development of LBGs. Games like Parallel Kingdom (released in 2008) and Ingress (released in 2012) have reached more than 1 million users and 7 million downloads, respectively. These numbers indicate a considerable potential for growth in the market. More recently, the release of Pokémon Go in 2016 confirmed this potential, and the game reached 45 million users within a few weeks.

Nevertheless, developing LBGs is challenging due to the inclusion of features and issues that are not present in most mobile games. For instance, developers have to create games to be available in a vast number of places, cope with tracking issues, and implement game balancing in different regions. A common strategy employed to make LBGs widely available is to use a database containing points-of-interest (POIs) scattered around the globe.

Usually, LBGs map POIs to virtual elements of the game, and players have to move to a POI before interacting with it in the game. Therefore, an LBG can be played virtually everywhere, provided there are POIs nearby the player. However, the time and effort to physically move between POIs is a crucial factor to game balance. The main drawbacks of this approach are the complexity of building such a large database of POIs and the difficulties of balancing the game for multiple regions. Consequently, most LBGs present great inequality between countries, cities, and even neighborhoods.

Today, the two most popular LBGs are Pokémon Go and Ingress, and despite their enormous popularity and broad availability, players of both games have experienced issues regarding the distribution of POIs. These POIs are known as portals in Ingress or PokeStops and Gyms in Pokémon Go. Millions of players have downloaded these games before noticing there were a few or no POIs in their region. This issue is so evident that an American newspaper compared...
The scenario as mentioned earlier is the key motivation for this research. To reduce game unbalancing, we need to incorporate new places in regions lacking POIs and redistribute POIs in areas where they are unevenly scattered. In this work, we propose an approach that uses Monte Carlo Tree Search (MCTS) and information from Google Maps API to improve game balancing in LBGs by adding or replacing POIs. Additionally, we introduce an approach to compare aspects of game balancing in LBGs that use a database of POIs, such as Ingress and Pokémon Go.

The rest of this paper is organized as follows: Section II presents the background related to this work. Section III introduces the measurements designed to estimate game balancing. In section IV, we detail the steps that comprise the proposed method. Section VI presents an evaluation comprising the performance of the method and the improvements in game balancing. Moreover, in section VII, limitations about this research are discussed. Finally, section VIII summarizes the paper and points some directions for future exploration.

II. BACKGROUND

A. Location-Based Games

Recently, due to the widespread usage of mobile devices and the release of popular games, LBGs have experienced great popularity. However, the first LBGs were developed in early 2000’s despite the technological limitations of the period. Games like Pirates!, Can You See Me Now?, ARQuake, and Human Pacman pioneered the use of location in the gameplay [4]. The term LBG hasn’t been consolidated by the time of their development, therefore authors of these early games classified them as Pervasive Games.

In the last years, companies and universities have developed LBGs with diversified purposes[5], [6]. In general, LBGs have been applied to areas such as tourism [7], education [8], health [9], [10], among others. This widespread use of LBGs corroborates to their potential for growth and highlights their ability to be used in many fields. Nevertheless, there are challenges in the development of these games that haven’t been addressed.

B. Game Balancing

A great issue developers face when designing LBGs is the need to balance the game throughout the regions it can be played. Game balancing relates to the difficulty level faced by players. It is a key factor in the development of modern games due to its importance to the gaming experience [11]. [12] considers game balancing as one of the three quality factors responsible for the engagement in games. In general, too hard games are deemed frustrating and too easy games lead to boredom [13].

Usually, balancing games is a difficult and time-consuming task that requires extensive testing and calibration. Moreover, game balancing is presumably harder in more sophisticated games because even minor changes can affect other areas of the game [14]. Consequently, many efforts have been devoted to the design of automatic and dynamic game balancing techniques, ranging from the use of artificial neural networks [15] to reinforcement learning algorithms [13].

Balancing LBGs is especially complex due to the inherent link between the game and real world. This connection gives rise to countless factors and options that can be regarded as game balancing components. For instance, terrain topology, transport availability, and public safety are just a few examples of real world features that can influence the gameplay, and hence the balancing of an LBG. Although the method presented here is general, in this work we selected distance and time-to-walk as sole real-world features for improving game balancing.

Furthermore, there are works applying automatic game balancing approaches to multiple genres of games, including serious games [16], puzzles [17], real-time strategy games [15], fighting games [13], etc. However, game balancing for LBGs composed of POIs has not been an important subject in many studies, and the few works that mention it do not specifically address the issue [18], [19]. As a result, even modern LBGs present significant disparity in game balancing between different regions.

C. Monte Carlo Tree Search

Modern LBGs can be played virtually everywhere, and hence each place of execution corresponds to a different game setting that needs to be balanced. This myriad of possibilities and configurations makes manual game balancing an impractical job. Therefore, automatic approaches to improving game balancing in LBGs is needed.

This work builds on Monte Carlo Tree Search (MCTS) to create an automatic method to improve game balancing in LBGs composed of POIs. Despite its potential as an AI technique capable of competing at an expert level in Go [20], MCTS can also be used to find near optimal solutions in large state-space Markovian Decision Problems [21]. Over the years, MCTS has successfully been used in different domains, such as optimization, real-time strategy games, general game playing, and complex real-world planning [22].

We designed this approach to adjust the game balancing on demand -according to the player’s location- and to cope with LBGs that have varying amounts of POIs. Therefore, the number of possible solutions is big, since POIs can be shifted anywhere. Consequently, deterministic approaches are unsuited to solve this challenge, especially considering that the number of selected POIs varies according to each region and game. In this case, non-deterministic methods are deemed more viable due to their ability to deliver good results under a predefined computational budget.

Nevertheless, selecting appropriate non-deterministic approaches is fundamental to achieve good results. In this work, we used MCTS for its potential to optimize exploration using information collected on previous searches. MCTS focuses on promising solutions while devoting minor efforts to portions of the search space that generate bad outcomes.

III. GAME BALANCING IN LBGs

This research focuses on LBGs composed of POIs. Therefore, we generalize the difficulty level an LBG presents in a determined location as the total effort to move between the POIs nearby. In this case, we represent LBGs as a directed weighted graph \( G = (V, E, W) \), where nodes \( (V) \) specify the collection of POIs that composes the game, edges \( (E) \) indicate the existing paths between two POIs, and weights \( (W) \) quantify the effort to move between POIs, with each weight \( w \in \mathbb{R}^+ \).

This representation is flexible as edges can be added or excluded to map custom game flows and weights can be associated with real world features that are deemed influential to the game balancing. For instance, LBGs that request players to visit an ordered sequence of POIs can be symbolized as a graph with a single chain of edges reaching all the nodes. Moreover, healthy LBGs can use footsteps, calories or heartbeats as weights, whereas competitive games can use time, distance, etc.

We have defined two aspects of evaluation to gauge aspects of game balancing in LBGs. The first focuses on examining the internal balancing, i.e., it estimates the effort for players to move to the closest POIs, and also checks whether POIs are evenly distributed in the region. In this case, unbalanced LBGs are the ones that have POIs too easy or too hard to reach. Therefore, we called Internal Difficulty Level (3) the average estimated effort to reach every POI in a determined location, as shown by 1.

\[
J = \sum_{x=1}^{N_V} \frac{w_x}{N_V} \tag{1}
\]

where \( w_x \) is the effort to reach POI \( x \), and \( N_V \) is the number of POIs within a certain range. Additionally, the standard deviation \( \sigma_J \) for the efforts is calculated to evaluate the uniformity of POIs distribution.

The second analysis was designed to highlight dissimilarities in game balancing between two areas. It establishes a direct comparison between distinct LBGs or between the same LBG played in distinct regions. To make such comparison, it is necessary to calculate the minimum difference in game balancing considering each path that forms the games. In this case, since LBGs are mapped to weighted graphs, the analysis is equivalent to the graph matching presented in [23]. This process extracts the best similarity between the paths of both games by minimizing their differences. This measurement is called Minimum Balancing Difference (\( \mathfrak{M} \)) and is given by the sum of the differences between all the corresponding paths in the best similarity case, as expressed by 2. Next, we detail how \( \mathfrak{M} \) is calculated.

Consider an LBG that can be played in two different areas, thus giving rise to two distinct game configurations \( (G_A \text{ and } G_B) \). \( \mathfrak{M} \) is obtained by matching a set of POIs \( A \) of \( G_A \) to a set of POIs \( B \) of \( G_B \) so that:

\[
\mathfrak{M} = \min_{x=1}^{N_A} \sum_{y=1}^{N_B} \left| w_{A_{xy}} - w_{B_{xy}} \right| \tag{2}
\]

where \( w_{A_{xy}} \) and \( w_{B_{xy}} \) represent the weights of paths connecting POIs \( A_x \) to \( A_y \) and \( B_x \) to \( B_y \), respectively.

For the sake of simplicity, the following example demonstrates how to calculate \( \mathfrak{M} \) for LBGs containing only three POIs. Figure 1 depicts POIs in different areas and the estimated effort to move between them. Despite having only three POIs, there are some possible matches between these graphs. For instance, the mapping \( (A/X, B/Y, C/Z) \) presents the following difference between paths \( |3 - 7| + |2 - 3| + |5 - 1| = 9 \), however the best match is \( (A/Z, B/X, C/Y) \), thus the Minimum Balancing Difference is 3 (\( \mathfrak{M} = |3 - 3| + |2 - 1| + |5 - 7| \)).

To summarize, \( \mathfrak{M} \) indicates whether players in different locations can compete more fairly since it shows how unbalanced the games are considering each available path. Therefore, if \( \mathfrak{M} = 0 \) for two distinct games, players in both areas should experience equivalent game balancing because the estimated effort to move between each POI is equal in both games. In section IV, \( \mathfrak{M} \) is used as the reference to an automatic approach capable of improving game balancing in a specific area.

This section presented two distinct measurements designed to estimate game balancing in LBGs. While \( J \) concerns the overall difficulty to play the game in a specific location, \( \mathfrak{M} \) focuses on gauging the existing disparity in game balancing between distinct areas. Section V presents a case study investigating the differences in some aspects of game balancing for Ingress and Pokémon Go using \( J \) and \( \mathfrak{M} \).

IV. METHOD

In order to improve game balancing in LBGs, we present a method that uses MCTS to find near optimal solutions in a set of candidate POIs. As shown in section III, LBGs can be represented as directed weighted graphs where weights represent the estimated effort to move between POIs. Additionally, we introduced a measurement that compares aspects of game balancing between two areas considering differences in their weights (\( \mathfrak{M} \)). Based on this feature, we designed a method capable of editing POIs in LBGs to minimize differences in game balancing.

Despite the intuition of the process, replacing POIs is a complex task that affects all the paths between them. As a result, shifting a POI can minimize differences between weights for a set of paths while maximizing others. Additionally, the need to assess many POIs for every unbalanced path can lead to huge search spaces. For instance, if an LBG is composed of 10 POIs and there are 20 candidates to improve the game, the total
amount of tests is \(20!/10!\) (permutation without repetition). In fact, this challenge is similar to the graph matching, a well-known problem present in many fields [23]. In this work, we use MCTS to explore the search space and find improved solutions under a specified budget.

The proposed method is composed of four steps: (i) Convert LBGs to Weighted Graphs, (ii) Select POIs, (iii) MCTS Computation, and (iv) Improvement Analysis. Figure 2 details the pipeline of execution. The output of the method is a version of the LBG that may have a few or all of the original POIs replaced, depending on the game balancing differences between the original game and the target LBG. Each one of these steps is better detailed in the following subsections.

A. Convert LBGs to Weighted Graphs

In this step, the LBGs are converted to directed weighted graphs in the following format \(G = (V, E, W)\). The conversion is straightforward and consists of a direct mapping from POIs to nodes \((V)\), paths to edges \((E)\), and estimated efforts to weights \((W)\). Nevertheless, a key part of this process is to estimate the effort to move between POIs since this information differs depending on each game.

In theory, every feature capable of affecting the game balancing should be employed. However, it is impractical to define and assess every real-world feature. Thus, the most significant aspects must be selected. In our experiments, we used Google Maps API to estimate distance and time-to-walk between POIs, although other tools and features can be used.

B. Select POIs

The method can use a variable number of POIs depending on the region the game is played. In theory, all POIs nearby are eligible, but the high complexity demanded to both define weights for each path and find the optimal solution in greater search spaces prevents this practice. Usually, urban centers have a wide range of locations, thus demanding a selection. In our tests, we used the longest path in the target game as the maximum distance a candidate POI should be.

Additionally, POIs can be selected to conform to the gameplay or for specific purposes. For example, an LBG may associate virtual shops to real markets and educational games may select historic places as POIs. After selection, it is necessary to add the POIs to the graph that represents the game. We used Google Places API to query POIs and selected them arbitrarily. Then, we used Google Maps API to estimate weights before adding them to the game graph. As a result, this step outputs a larger weighted graph composed of POIs from both the original game and Google Places.

C. MCTS Computation

Previously, we discussed the complexity involved when replacing a POI to improve game balancing. To address this issue, we rely on MCTS’ capacity to take random samples in the search space and use this information to guide further exploration for better solutions. The proposed method receives two weighted graphs as input - the target graph and a larger graph composed of original and selected POIs -, and process these data by searching for matches between POIs that minimize \(\mathcal{M}\). In this case, searching for POIs corresponds to the selection and expansion process of MCTS, and calculating \(\mathcal{M}\) represents simulation.

Furthermore, it is necessary to backpropagate a reward based on the results of simulation. In this instance, we decided to use a limiting threshold \((\tau)\) to ensure the method returns solutions within a minimum similarity in relation to the game balancing of the original LBG. Therefore, \(\tau\) is compared to the ratio between \(\mathcal{M}\) and the target graph \((\sum_{x,y=1}^{N_{V}} w_{xy})\). When expression 3 is true, the method backpropagates a win, otherwise a loss is returned.

\[
\frac{\mathcal{M}}{\sum_{x,y=1}^{N_{V}} w_{xy}} < \tau \tag{3}
\]

As a result, MCTS will allocate more resources to exploit branches of the solution tree that yield results within a satisfying threshold \(\tau\), while eventually exploring branches that appear to generate sub-optimal matches. Consequently, the method relies on specifying a suitable value for \(\tau\) to deliver good performance. High values of \(\tau\) can cause MCTS to back-propagate wins for every case, and low values can continually lead to loss. In these cases, MCTS fails to select branches for exploitation and hence operates choosing solutions randomly.

The outcome of this step is the solution presenting lower \(\mathcal{M}\) evaluated during simulation. In this case, a set of POIs matched to each node of the target graph. In the next step, the solution is analyzed to assess improvements in game balancing.

D. Improvement Analysis

The main goal in this phase is to ensure the new solution presents improved game balancing in relation to the original game. In this regard, an analysis based on \(I\), \(J\) and \(\sigma_{3}\) is conducted to compare both the original JBL and the generated solution.

First, it is necessary to verify if \(\mathcal{M}\) for the generated solution has lowered in relation to the original solution. In case MCTS provides a better result, \(\mathcal{M}\) for the generated graph is usually lower, therefore \(I\) and \(\sigma_{3}\) are likely to be similar to the target game. Conversely, if MCTS fails to improve the original game, \(\mathcal{M}\) tends to be higher than the original LBG. In case \(\mathcal{M}\) is similar to both new and original games, \(I\) and \(\sigma_{3}\) are used to choose the best one.

In this section, we detailed the method used to improve game balancing on LBGs. The proposed approach supports multiple types of LBGs, as long as they are composed of POIs. In section VI, we present an evaluation consisting of experiments conducted using POIs from Ingress and Pokémon Go to assess the proposed method.
V. ANALYSIS OF GAME BALANCING IN INGRESS AND POKÉMON GO

As mentioned in section I, Ingress and Pokémon Go were released in many countries and are currently the most popular LBGs worldwide. However, both games share the same database of POIs and hence present similar issues with game balancing. This section shows an analysis of some aspects of game balancing in these games using the evaluation aspects presented in section III.

Before conducting this investigation, we compiled a set of POIs using collaborative portals that map Pokémon Go Gyms and Gyms in many countries. However, to ensure the authenticity of this data, we validated each POI using the official Ingress website 7, that shows information about POIs of the game.

Furthermore, we used Google Maps API to estimate the effort to move between POIs by querying routing information. Google Maps API provides live information about traffic and offers a series of options including traffic modes, time of departure and route settings. However, we conducted evaluations considering distance and time to walk as effort since these are essential features in the games. For instance, Pokémon Go demands players to walk by posing speed limits.

In order to provide a broad investigation about game balancing in the games, a set of locations in different continents were used as sample. The places enabled us to compare aspects of game balancing between cities of different countries and distinct neighborhoods in the same city. Therefore, it is possible to notice how game balancing varies according to each region. Table I shows the selected locations, their geographic coordinates, and, in the leftmost column, the corresponding region. The data indicate a significant disparity in POIs distribution and hence POIs are unevenly distributed.

To provide a better comprehension about the generated data, we present $\mathcal{M}$ as a percentage in relation to the target game. Table II shows the minimum difference in game balance considering the distance to walk ($\mathcal{M}_d$). The data show correlations between areas of distinct cities and highlights differences in game balancing within the same city.

In general, $\mathcal{M}$ provides key information to assess whether players in different location face similar challenges. This study emphasizes that game balancing in LBGs is still an issue even when comparing great cities. In this context, players can have different challenges to complete the game.
significant benefits or disadvantages according to the region they are playing or even the country they live. Consequently, this great disparity in game balancing can be regarded as a key challenge to their popularization in some areas. Moreover, it prevents LBGs to incorporate competitive features even for players living in the same city.

The evaluation shows great inequality in game balancing between some areas of distinct cities and within the same city. This indicates that both Ingress and Pokémon Go deliver varying gaming experiences depending on the city and the neighborhood the games are played. To address this issue, section VI uses the proposed method to reduce unbalancing in those areas.

VI. EVALUATING THE METHOD

The analysis conducted in section V highlighted great differences in game balancing between distinct locations. This issue has been reported repeatedly by players around the globe, and in section IV we presented an approach developed to overcome this problem. Therefore, to validate the performance of our method we conducted a series of tests involving the same data analyzed in section V. The evaluation consists of applying the method to improve game balancing among distinct locations and drawing a comparison to assess the results.

The proposed approach to improve game balancing edits an original game based on the game balancing of a target LBG. Therefore, it is necessary to select a game with the desired difficulty, although a synthetic game can also be used. In our tests, NY1, TK1, and BL1 were used as target games. These locations were selected due to their low and σ, therefore POIs in these locations are regularly distributed and easy to reach.

As shown in Table II, $M_d$ indicates NY1 and TK1 present remarkable disparity in game balancing when compared to other locations, specially RJ2, JOH1, and BL2. In all these cases, measurements indicate players in NY1 and TK1 are likely to have a significant advantage. Consequently, we applied the method to improve game balancing for RJ2, JOH1, and BL2 based on features of NY1 and TK1. In addition, we improved game balancing on BL2 considering BL1 as the target, since these locations present the highest difference in $M_d$ within the same city.

As a result, Figure 4 depicts how $M_d$ reduced in relation to the original games. Notice the method was able to reduce $M_d$ to below 45% for all JBLs, even in cases like BL2-TK1(43.7%), BL2-NY1(39%), and RJ2-TK1(29.8%), that originally presented $M_d$ around 100%.

Furthermore, we also compared $\mathcal{J}_d$ and $\sigma\mathcal{J}_d$ between original, target and improved games. Results show the proposed method provoked significant changes to $\mathcal{J}_d$ and $\sigma\mathcal{J}_d$. In fact, improved LBGs present distinct features to the original version and similar to the target game, as depicted in Figure 5. In many cases, values of $\mathcal{J}_d$ for improved games were lower than their target counterpart. This occurrence is linked to the range used to select POIs from Google Maps. The majority of places selected were situated near to the player, thus forming a cluster of candidate POIs. We believe better results can be achieved after improving the step to select POIs.

Figure 6 presents images of POIs distribution for the target, original and improved JBL. POIs are represented by black dots on the map, and the red circle marks the maximum range (40 meters) Ingress and Pokémon Go allow players to interact with a POI. In most cases, the method replaced all the POIs in the original game due to the considerable differences between original and target LBGs.

In this section, we presented a study conducted to assess and validate the proposed method to improve game balancing in LBGs. Results indicated the method presented good performance, especially for cases with great difference in game balancing. Additionally, the analysis pointed out aspects for improvement and potential flaws that are discussed in section VII.
A. Material and Methods

The MCTS algorithm implemented in this work used UCB1 [21] as selection policy. We defined 3000 milliseconds as the computational budget during MCTS computation because a recent global research report showed that 68% of consumers demanded a loading time of less than six seconds - and nearly half of those respondents demand a load time of less than three seconds [24]. Nevertheless, in further tests we could achieve similar results with half the budget, depending on the specified \( \tau \). In all tests, we limited the amount of POIs to 10 for target and original JBL, and for candidate selection. Therefore, the method explored improved solutions by choosing 10 POIs among 20 candidates.

VII. DISCUSSION AND LIMITATIONS

This work introduces a method to improve aspects of game balancing for LBGs composed of POIs. The method was designed to be general and capable of handling many types of gameplay and balancing features. However, some LBGs are not composed of POIs. In these cases, the proposed method can be unsuited or demand multiple adaptations. This limitation must be evaluated and addressed in future works.

Another drawback of the presented approach is the need to provide better matching between virtual and real places. In some LBGs, this matching is key to the gameplay. Although it is possible to select POIs by their type, as shown in section IV-B, the method does not allow individual matching of POIs. A potential solution to this problem is to add a step responsible for managing the matches before MCTS computation, thus the search space will be composed only by wanted matches.

The evaluation shows the majority of improved JBLs replaced all their original POIs and, in many cases, they presented lower \( I \) than the target JBL. This issue is deeply connected to the selection of POIs, since no process was implemented to ensure the uniformity of their geographic distribution. Consequently, if target and original games present great differences in game balancing, MCTS will probably find better solutions within a cluster of POIs, as depicted in Figure 6 (i).

The results of the evaluation indicate that the proposed method presents satisfying performance in most cases. However, we believe significant enhancements can be achieved by adjusting the selection of POIs to feed MCTS with better candidates. Additionally, further analysis must be conducted to define an optimal threshold for each set of POIs processed by MCTS, and a set of new tests must evaluate the performance of the method with varying number of POIs.

A. Threats to Validity

We regard weights as key components to measure and improve game balancing. Therefore, any imprecision when defining their values can lead to bad results. In addition, we rely on Google Maps to estimate values for weights, but the queried data vary with time. As a result, the information used in this work may not always represent real game balancing. Consequently, the presented approach should be used immediately before playing the games.

Additionally, a key pitfall to this work is the need to cope with more balancing features. In practice, many aspects can affect game balancing, especially in LBGs due to their connection to the real world. However, only two aspects (distance and time to walk) were used to gauge and improve game balancing in this investigation.
VIII. Conclusion

Improving game balancing in LBGs is a complex task due to the inherent connection between these games and the real world. Despite the advances in the development of LBGs, this challenge remains open and affects modern games. In this work we introduced a method to improve aspects of game balancing in LBGs composed of POIs. Additionally, we presented two distinct measurements to assess game balancing in LBGs composed of POIs. These measurements were designed to evaluate the effort to reach POIs within an area and to compare game balancing between two distinct LBGs.

The proposed method to improve game balancing replaces POIs of an LBGs to minimize disparity in game balancing. This process relies on information queried from Google Maps to select candidate POIs and to estimate the effort to move between locations. Moreover, the method uses MCTS to find solutions with better game balancing under a specified budget.

An analysis of game balancing was conducted with the two most popular LBGs (Pokémon Go and Ingress). In this case, a set of locations in different countries were used as sample. Results showed significant differences in game balancing between distinct cities and within the same city.

Furthermore, a comparison showing differences in game balancing was drawn to assess the performance of the proposed method. Results indicated the approach improved aspects of game balancing in many locations, especially in games with greater inequality. Additionally, the evaluation indicated potential flaws and room for advances.

In the future, we intend to refine the quality of results by designing a better selection of candidate POIs and allowing the method to handle more aspects of game balancing. In addition, a study must be conducted to determine the best limiting threshold to be used in MCTS computation.

Acknowledgment

The authors would like to thank the Brazilian research-funding agencies FUNCAP (State of Ceará) and FAPEMAS - SECTI (State of Maranhão) for partially funding this work.

References