AI as Evaluator: Search Driven Playtesting of Modern Board Games

Fernando de Mesentier Silva and **Scott Lee** and **Julian Togelius** and **Andy Nealen**
Game Innovation Lab - New York University - Tandon School of Engineering
5 Metrotech Center
Brooklyn, New York 11201

Abstract

This paper presents a demonstration of how AI can be useful in the game design and development process of a modern board game. By using an artificial intelligence algorithm to play a substantial amount of matches of the Ticket to Ride board game and collecting data, we can analyze several features of the gameplay as well as of the game board. Results revealed loopholes in the game's rules and pointed towards trends in how the game is played. We are then led to the conclusion that large scale simulation utilizing artificial intelligence can offer valuable information regarding modern board games and their designs that would ordinarily be prohibitively expensive or time-consuming to discover manually.

INTRODUCTION

Modern board games pose interesting research challenges with their varying levels of strategic complexity, multiplayer aspect and usage of stochasticity, hidden information and feedback loops. As board games continue to increase in popularity (Griepp 2016) an increasing number of games are being released every year, and for many of these games, the design relies on ensuring that the gameplay experience feels fair for all players. Dominating strategies, an unbalanced deck of cards or a board that allows for situations not covered by the rules are then elements that impact the reception of a game. As such, a substantial portion of a game's design process is spent making changes to repair and avoid these unwanted scenarios.

Testing and experimentation are essential aspects of game design. Designers can only test for scenarios that they can think of, and design flaws are inevitable in any system of considerable complexity. Ideas and mechanics that work in theory or on paper can fall apart once the playerbase finds a weakness it can exploit. Such scenarios are often so obscure and ultimately impractical that they are harmless to the state of the game, but they can occasionally give way to optimal strategies or undesirable gameplay. It is often in a designer's best interest to minimize the number and severity of these weaknesses or loopholes in the game's logic. To this end, AI can accelerate the process of playtesting by exploring the game state space to find exploits and scenarios that the designer has yet to anticipate.

Another factor that designers look for in their games is competitive balance. Competitive games that reward skilled play result in players feeling more fulfilled, while simultaneously motivating players to improve further (Kraaijenbrink et al. 2009). Additionally, a lack of balance is problematic. Games with widely known optimal strategies tend to have its playerbase converging on a single playstyle or emphasizing a single game mechanic, condemning divergent playstyles as ineffective or extraneous mechanics as worthless. By having AI that can simulate different powerful strategies under a reasonable amount of time, we can evaluate their performances over a large number of simulations, address more obvious imbalances and see how certain changes would affect gameplay before having human players experience it.

It is important to note that as the complexity of a game increases, the difficulty of maintaining competitive balance and game integrity quickly escalates. Each variable in a game needs to be inspected for its impact on the system as a whole. Ensuring that a game is balanced, or its logic is sound, therefore requires a great deal of testing, and even then, individually accounting for each and every possible scenario is a herculean effort. This serves to reiterate our proposal to use AI to lighten the workload from fine tuning game features. As opposed to more traditional games such as Chess and Go, which are exclusively 2-player games, many modern board games are tailored to work with a varying number of players. Changing the number of players usually affects different aspects of the game. The same can be said for changing a component or rule. Those changes, for instance, may impact the number of turns, and so the length of the game, as well as how effective strategies are. With the use of AI agents we can simulate the game in different scenarios and account for the affect they have on gameplay.

This paper is focused on exploring the potential uses of artificial intelligence algorithms to aid in the design process of modern board games. For this work we use the game Ticket to Ride and several of its expansions as test cases. For such, we describe 2 gameplaying agents based on common strategies used by players. We show how AI agents are able to find scenarios that the rulebook of the game doesn't cover. We also compare the performance of the agents over 7 different board and deck packs, analyze their characteristics and how they impact the agents strategies. Then we propose future work for this project and other ways in which AI can have

an impact on the design process of modern board games.

RELATED WORK

Modern board games are not new to AI research. Guhe et al. presented a framework to compose agents for the game Settlers of Catan (Guhe and Lascarides 2014). Previous work has demonstrated that it is possible to tailor an agent's strategy to a game and evaluate its behavior in comparison to human players. Szita et al. and Chaslot et al. both present Monte Carlo Tree Search (MCTS) agents for playing Settlers of Catan and compare their performance to heuristic driven AIs (Szita, Chaslot, and Spronck 2009; Chaslot et al. 2008), and Pfeiffer presents a reinforcement learning approach to playing the game (Pfeiffer 2004). The agent presented is tailor-made using both learning and prior knowledge. Other work compares multiple approaches of agents to one another in the game Carcassonne (Heyden 2009). The work focuses on the 2-player variant of the game and discusses variations of MCTS and Minimax search for playing the game. Robilliard et al. demonstrates the study of using MCTS to play the game 7 Wonders (Robilliard, Fonlupt, and Teytaud 2014). Huchler presents how different enhancements made to MCTS can improve its performance when playing Ticket to Ride (Huchler 2015). Different player agents play against a cheating agent, one that has access to all hidden information, to benchmark their results.

The idea of algorithmically optimizing balance in classic board games and card games has been explored. Hom et al. explored AI techniques to design balanced abstract board games by altering game rules by using a genetic algorithm to search for possible rules in the game space (Hom and Marks 2007). The work defines a good balanced game in terms of the advantage of moving first and how often agents draw. Krucher showed AI that had the ability to generate more interesting and complicated elements of the game (Krucher 2015). In this work the author algorithmically balances his collectible card game implementation via rating and modifying cards through iterations of an AI agent playing it. These ratings were then used by the agent when deciding what action to perform. Jaffe et al. uses a set of balance metrics to find the importance of various features of the game (Jaffe et al. 2012). The authors apply these to a perfect information educational card game where they automatically track and measure these features. Dormans presents a framework that can describe the flow of resources throughout a game, describing its internal economy and through simulations enabling balance when analyzing the efficiency of various strategies on the early stages of a game's design (Dormans 2011).

Mahlmann et al. discusses balance evaluation on the card game Dominion (Mahlmann, Togelius, and Yannakakis 2012). This work utilizes three different AI agents, each with different fitness functions and skill levels to determine a balanced card set. Artificial neural networks (ANN) were used for evaluating the state of nodes in the MCTS, as well as the general state of the game board. It was found out that certain cards were present in the winning sets of all three AI agents, and thus it was concluded that these cards made the game more balanced independently of gameplay style. This work demonstratres that this method of game balance evaluation has credence in designing and balancing other games.

Video game research has tackled the idea of having AI and Machine Learning algorithms act as a co-designer, giving inputs and suggestions during the process of development. This research field is called mixed initiative design (Yannakakis, Liapis, and Alexopoulos 2014). Liapis et al. present Sentient Sketchbook (Liapis, Yannakakis, and Togelius 2013), a tool where users create maps for Real Time Strategy games and have suggestions offered by the system. Smith et al. describes a system for developing 2D platformer levels (Smith, Whitehead, and Mateas 2010). In it the user edits key aspects of the level and the tool fills the rest of the level while guaranteeing that it is playable. Shaker et al. show a tool for designing levels for the video game Cut the Rope (Shaker, Shaker, and Togelius 2013). It is capable of checking a level for playability, completing a level after a partial design input from the user or automatically generating a new level.

There are also other approaches to how AI can aid the process of game design. Browne et al. explore this avenue by using an evolutionary algorithm to design games (Browne and Maire 2010). By evaluating and automatically measuring the quality of a game's concept (Browne 2008), the algorithm was able to create a novel and interesting abstract board game that was later published. Salge et al. uses an adaptive AI to model the concept of Relevant Information and indicate how it can relate to a game's design (Salge and Mahlmann 2010). Smith et al. introduces a game engine that can generate gameplay traces to identify the underlying behavior of a game (Smith, Nelson, and Mateas 2010). Nelson argues that not all information extracted from a game needs to derive from empirical playtesting (Nelson 2011). The author demonstrates seven other strategies to extract information from games. Nielsen et al. investigates the idea of characterizing a game's quality through the relative performance of multiple general game playing algorithms (Nielsen et al. 2015). Isaksen uses automatic play testing to explore the game space of the video game Flappy Bird and find interesting variants of the game that alternate in game feel and difficulty among other features (Isaksen, Gopstein, and Nealen 2015; Isaksen et al. 2015). AI and Machine Learning is also used by de Mesentier Silva et al. to search for simple and effective novice level gameplaying heuristics for the card game Blackjack (de Mesentier Silva et al. 2016).

Although work has been done in AI agents for modern board games, in balancing video games, card games and classical board games and in tools for AI co-authored design, no research, to our knowledge, has tried to explore balancing, feature analysis and automated playtesting for the purpose of aiding the design of a modern board game.

TICKET TO RIDE

Ticket to Ride is a two to five player competitive board game designed by Alan R. Moon, published by Days of Wonder in 2004. The game has won multiple awards and sold over 3 million copies by 2014 (Days of Wonder 2004). Multiple versions of the game have been released since. The results shown in this paper use the standard game and four

of its variants/expansions. Out of the expansions used, three change the board and destination deck with which the game is played and add a couple new rules to the ruleset, they are Europe, India and Nordic Countries expansions, and the other changes the original destination deck in three different ways creating three new variants while using the board of the standard game, the USA 1910 expansion. The rules described below are the ruleset for the standard game, expansions might change details in the rules or add new ones.

In Ticket to Ride, players are trying to score the most points. The game is composed of 4 elements: Train Cards, Destination Cards, Train Tokens and the Board. To score points, players collect Train Cards that allow them to use their Train Tokens to connect different cities on the Board.

The deck is comprised of 110 Train Cards: 12 for each of the basic colors, Red, Blue, Green, Yellow, Pink, Orange, White and Black, and 14 locomotive, or wild, cards. The wild cards are unique in that they can take the place of any other color of Train Card. While each player has their own pool of Train Tokens, the deck of unique Destination Cards and the Board are shared by all players. Each player has their own hands, which are hidden from other players, where they keep the Train Cards and Destination Cards they collect. Each player's Train Token pool size remain public throughout the game.

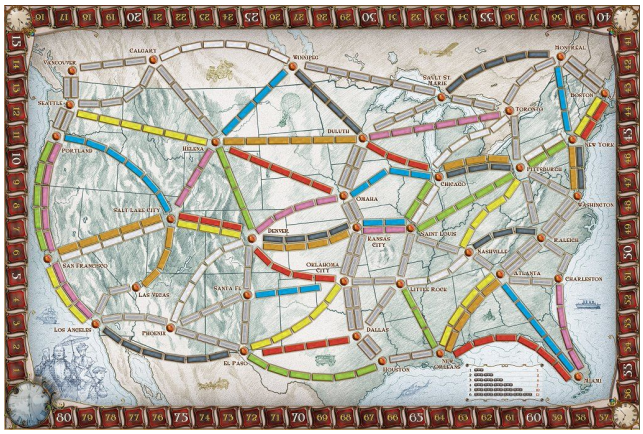

Figure 1: The USA map which portrays the board used in the original game. Each city is represented by a circle. Each route is a sequence of rectangles. The number of rectangles is equal to the route size and the color of the rectangles portray the route's color.

The board represents the cities in the game and their connections, called routes. Two cities connected by a route are called adjacent cities. There can be either 1 or 2 routes connecting the same 2 cities. Each route has two attributes: color and size. The route's color can be any of 9 different possible, either one of the 8 basic colors of the game or a special color: Grey. The basic colors are represented by the different Train Cards, so to claim a route of a basic color, Train Cards of said color are needed. Meanwhile, a Grey colored route can be claimed with any color of Train Card, as long as all the cards used are of the same color. The size of route determines how many of a specific Train Card a player must have to claim that route. Figure 1 shows the board used to simulate games in this paper.

Claiming a route between two cities will score the player points, although much more score can be achieved through connecting two specific non-adjacent cities required by a Destination Card. Players complete a Destination Card by claiming multiple routes that together connect the two cities listed on the card. The Destination Cards each player has are kept hidden for the others until the game has ended, when they are revealed and their points are tallied. If at the end of the game the player has managed to connect the two cities with only routes she claimed he gains the number of points written on the card. If she fails to do so, she loses that same amount of points.

One player is selected at random to be the first player. From them on, each player will then execute one of the 3 possible moves when their turn comes. Once they are done, the play proceeds to next player in clockwise fashion. The possible moves a player can take on their turn are: Draw Train Cards, Draw Destination Cards or Claim a route.

When drawing Train Cards, a player can choose to pick one of the cards currently face up or to draw the card at the top of the deck to add to her hand. If she chose a face up card, that card gets immediately replaced by revealing the next top card of the Train deck before proceeding. When drawing Train Cards, the player will take 2 cards on their turn, with the exception of only drawing 1 in case he chooses to draw a face-up wild card. If a card would have to be drawn, but the Train deck is depleted the Train Card discard pile is shuffled and becomes the new Train deck.

When drawing Destination Cards the player gets the top 3 cards of the destination deck. The player then looks at the cards she drew and chooses to keep 1, 2 or all the cards drawn, which are added to their hand. She then returns the cards she decided not to keep.

To claim a route, a player must have Train Tokens equal to or higher than the size of the route they want to claim, and that route must have not been claimed by anyone. She must then discard from her hand a number of Train Cards equal to the size of the route and whose color matches the color of such route. She then moves on to placing her Train tokens on the route to mark it as have been claimed by her. The player also immediately scores points proportional to the size of the route claimed: A route of size 1 rewards 1 point, size 2 rewards 2 points, size 3 rewards 4 points, size 4 rewards 7 points, size 5 rewards 10 points and size 6 rewards 15 points.

If after claiming a route a player is left with 2 or less Train Tokens in her pool, she announces the last round of the game. Then, every player, including the one that announced, gets to make one more move. After everyone is done, the game ends, every player reveals their Destination Cards and the points are tallied.

Points obtained, or lost, from the Destination Cards are added to the ones each player got from claiming routes. In addition to those, the player with the longest continuous route of unique train tokens is awarded an additional 10 points. After adding all the points, the player that achieved

the highest amount of points wins the game.

PLAYER AGENTS

To evaluate different aspects of Ticket to Ride, we decided to implement multiple player agents that would approach game strategies differently. In addition to tree-search based algorithms A* and MCTS agents, we decided to create two agents developed specifically for playing this game, based on strategies commonly used by players. The use of such agents are justified by several reasons. Coming closer to replicating the human play styles would give a more interesting reflection of a common play of the game, with our objective being aiding the designer to tailor the game which ultimately is going to be played by humans. The two agents described are much cheaper in terms of computational time when compared to our other experiments using A* and MCTS agents. The agents shown are also the most well-performing agents we know of, outperforming A* agents that search the game states for actions and regular MCTS agents, a pattern that we believe to be a consequence of the state space size for Ticket to Ride. The game has a very high branching factor which makes it difficult to explore enough states to choose a good move in a reasonable amount of time. More than that, it is hard to formulate a heuristic that is capable of evaluating a game state to achieve good play. That being said, we believe that it is possible to implement modifications to MCTS to make it a more effective player, similar to what was done for Super Mario Bros in (Jacobsen, Greve, and Togelius 2014) and to Ticket to Ride in (Huchler 2015); alternatively, it might be possible to use evolutionary computation to play the game well, as we have seen promising results with this approach in other games with high branching factors (Justesen, Mahlmann, and Togelius 2016).

The two agents that we are using are described below. They represent very different approaches to the game, with the first agent playing the game more conservatively, trying only to complete the routes is has been assigned, whereas the second plays the game more aggressively and proactively draws new destination cards. We believe these two agents span a very important axis of Ticket to Ride playstyles.

Route Focus Agent (RFA)

This agent tries to fulfill the Destination Cards that it receives at the beginning as fast as it can while also try to take long routes on the board to score additional points. At the start of the game, the agent keeps all of the Destination Cards it was dealt during the setup of the match. Then, in every subsequent turn, it evaluates all Destination Cards it has not completed yet. For each Destination, it looks for the shortest path between the two cities, considering the routes it has already claimed. It then puts the routes that make up that path into a priority queue, where their priority is defined by the function:

$P = R_{value} + 2 * DR_{score}$

where R_{value} is the amount of points the player scores just for claiming that route. DR_{score} is the point value of the Destination Card that route belongs to. Here the points related to the Destination Card are doubled to reflect the fact that if that objective is not completed the player will lose that amount of points.

If all of its Destination Cards have been completed, it looks at all of the free routes and puts them in the queue using the only R_{value} as their priority.

After the agent fills the priority queue, it goes to every route in the queue and checks if it has the Train Cards to claim it, and if it does, it returns that as a move. If it can't take any of the routes, it checks the color of the highest priority route. It chooses to draw the face up Train Card that matches that color, and if that color is not available, it draws the top card of the deck or a face up wild card.

Destination Hungry Agent (DHA)

This agent focuses on trying to score a large amount of points by accumulating a large number of destinations to complete. After selecting the Destination Cards it will keep from the setup the AI will spend a sequence of turns drawing more Destination Cards in order to build on the prospective points it will score by the end of the game. It will evaluate how to complete Destination Cards by checking the shortest route between cities, where the shortest route will be the one that needs the least amount of Train Tokens.

When deciding which Destination Cards to keep, the algorithm looks at all combination of cards it could keep. It them puts all cities present in the candidate cards into a pool together with the cities in the Destination Cards it already owns. It computes shortest paths between all the cities in these cards until it finds a list of routes that connects them all. It then computes how many trains are needed to claim all these routes, N_{trains}, and how many points the agent will score by claiming them, P_{Rvalue}. If N_{trains} is bigger than the number of trains the player has, it will discard selecting these as a viable option. The selection of the cards are based on a fitness function. The fitness function tries to reward choosing cards that have cities closer together and punishes making a choice for which take a lot of trains to complete in comparison to the number of points it awards. The fitness function is as follows:

$F = \frac{D_{score}+P_{Rvalue}}{N_{trains}}$

where D_{score} is the aggregate score of the Destination Cards. The agent then calculates how many of each color of Train Card it will need, to guide the decision of which to draw. It stops drawing new Destinations when the current set it holds already meets a certain threshold of Train Tokens needed. The threshold allows for a high number of Tokens, but also guarantees that the algorithm can have spare trains in case it needs to change the routes it will take in consequence of another player claiming a route it wanted.

Then, at every turn it will check if it can claim a route it needs. If it can claim more than one route it wants, it will prioritize routes that are of the color it needs the most to complete the objective set. If it can't claim any route, it will choose to draw Train Cards. It will decide which color to draw based on which it needs the most overall. If it already claimed all routes it wanted, the agent will look for claiming routes that will make it trigger the end of game condition the fastest.

Table 1: Data from the Destination Hungry Agent (DHA) versus Route Focus Agent (RFA) games. Ties are not shown for the Win Ratio and are the reason totals don't sum up to 1.

Variant	Agent	Win Ratio	Avg. Score	Normalized mean score
Standard	**DHA**	0.75	102.271	0.7284
	RFA	0.245	75.311	0.5486
USA 1910	**DHA**	0.767	109.97	0.7537
	RFA	0.226	79.016	0.5642
Big Cities	**DHA**	0.816	97.13	0.6521
	RFA	0.181	58.334	0.4741
Mega Game	**DHA**	0.853	116.013	0.7749
	RFA	0.146	55.204	0.4798
Europe	**DHA**	0.585	82.883	0.7409
	RFA	0.407	76.585	0.6312
Nordic Countries	**DHA**	0.631	73.935	0.6129
	RFA	0.365	51.688	0.5753
India	**DHA**	0.523	86.848	0.7376
	RFA	0.464	87.744	0.6146

ANALYSIS OF THE GAME

The analysis of the game was performed by having the RFA play against the DHA in 1,000 games in each variant of the game. Overall we have 7 variants of the game: The standard game, USA 1910, Mega Game, and Big Cities use the USA board that comes with the standard game, whereas the Europe game, the Nordic Countries game and the India game, all have their own specific boards and Destination Cards.

AI Match-up

In order to evaluate how our agents fare when playing the game, we looked at the simulations in all the variants. We compiled the most interesting aspects in Table 1.

From looking at the data we can see that DHA has a higher Win Ratio in every variant of the game, but while it is dominant in every variant played on the Standard board (Standard, USA 1910, Big Cities and Mega Game), the win ratios between the two players are much closer in the variants that use other boards (Europe, Nordic Countries and India). We can try to answer two questions at this point: Do we have enough confidence to claim that DHA has a dominant strategy over RFA? Does the board impact the outcome of the match?

In terms of the dominance of DHA over RFA, we can turn to a binomial test to test the statistical significance of this hypothesis. Through the binomial test we can verify $p < 0.0001$, that gives us a high confidence to assure that DHA does in fact execute a dominant strategy over RFA.

As to the impact the board has on the outcome of the match, we use the Chi-square test to try and refute the null hypothesis that our data is independent from the board the game is played in. The Chi-square test reveals $p < 0.0001$. We can then make a strong argument that the board holds significant influence over the outcome.

The difference in win ratio among the maps gives us room to analyze what aspects make them different. A mechanic common to all boards but the standard is the underground routes. These add more randomness to the game: when claiming one of these routes, a player must turn the top 3 cards of the Train Deck face-up and depending on what he draws, the number of cards required to claim that route may temporarily increase. This stochastic event may be responsible for evening out the odds between the two strategies implemented by the agents. The boards other than the standard are also known to promote more adversarial play. This could also point as to why DHA is more dominant in the variants played on the standard map.

Scenarios not covered by the rules

Over the course of the experiments, the Agents encountered two scenarios that are not described on the rules of the game. Both scenarios created situations in which one or both players could not make any moves.

No Train Cards This scenario arises when one player exclusively draws Train Cards, and the other player fails to finish the game before all Train Cards are depleted. If all of the Train Cards have been drawn, but the player lacks the necessary Train Cards to claim any routes, then they are forced to draw Destination Cards, forcing them to decrease their own score. Once the Destination Cards have been depleted, the player is no longer able to draw any more cards, nor is he able to claim any routes.

3 Wild Loop This scenario involves a rule involving face up Train Cards. When 3 or more of the face up Train Cards are wild cards, the face up cards are discarded, and 5 new cards are added. If both players exclusively draw non-wild Train Cards, then at some point, the deck will consist exclusively of wild Train Cards. At some point, the wild cards will appear in the face up set, which will prompt a reshuffle. However, if there are no other cards in the Train Card deck, then the wild cards will reappear in the face up set, prompting another reshuffle. The reshuffle cannot end because there are no other cards in the deck.

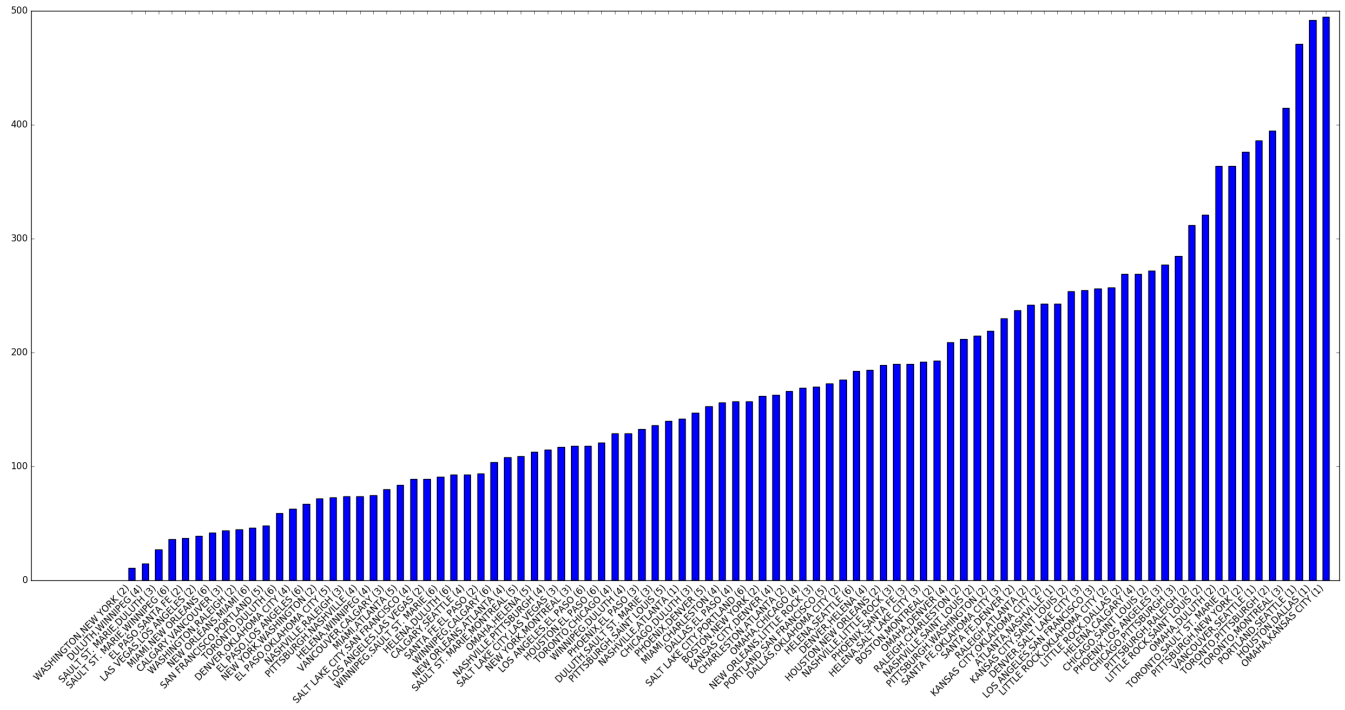

Figure 2: The bar chart for the frequency in which the winner of a match on the standard board takes a specific route. On the y-axis are the number of games in which the winner took the route, out of 1,000 games. The labels on the x-axis show the routes, named after the two cities it connects, with its length in parenthesis.

Winning Routes

Fig. 2 shows the distribution of routes claimed by winning players over the course of the simulations on the standard board. As can be seen, there is substantial variation between the routes, indicating that as a whole, winning agents favored certain routes over others. Several plateaus are also visible in the distribution, implying a tiered nature to the value of connections in the game board. There are two interesting features to note about the 9 connections that make up the highest plateau. First, routes of length 1 are disproportionately represented in this cluster relative to its quantity on the game board. Roughly 6 percent of all routes on the board are cost 1, yet it makes up 44 percent of the highest tier of routes. While there are 4 routes of cost 2, they are the most common type of route on the board, and so the representation is approximately proportionate.

Also of note in this cluster is the set of cities the routes connect. Toronto, Pittsburgh, and Seattle each have at least two connections in the top cluster. These three cities are notable for having a substantial number of connections. However, Helena, Denver, and Atlanta all have more connections than Seattle, yet no connections to them appear in the top cluster. The reason for this is currently unknown and bears further investigation, but the data indicates that these locations hold some particular value in gameplay.

Undesirable Cities

Fig. 3 is a heatmap displaying the frequency with which cities were ignored over the course of the experiment. For a city to be counted as ignored, all of its connections must be unclaimed. The implication behind an unclaimed city is that it is neither a destination nor a city through which a destination can be reached optimally. While unclaimed cities can change based on Destination Cards, a large number of simulations can reveal trends in the overall value of the city and its connections.

Particularly surprising is the frequency with which Washington is ignored over the course of the experiment. The values for other undesirable cities such as Las Vegas and Miami are understandable, because the former has only 2 connections and the latter is expensive to reach. However, Washington has 3 connections, all of which are inexpensive routes to reach key cities such as New York and Pittsburgh. One possible explanation is the fact that Washington is not on any of the Destination Cards, meaning that at most, it will be used as a passage to other cities. It is also worth noting that Pittsburgh is a more connected alternative to Washington for players attempting to connect Raleigh to New York. Under this assumption, it makes sense that Washington is undesirable, because Pittsburgh is simply a more attractive alternative in any given situation.

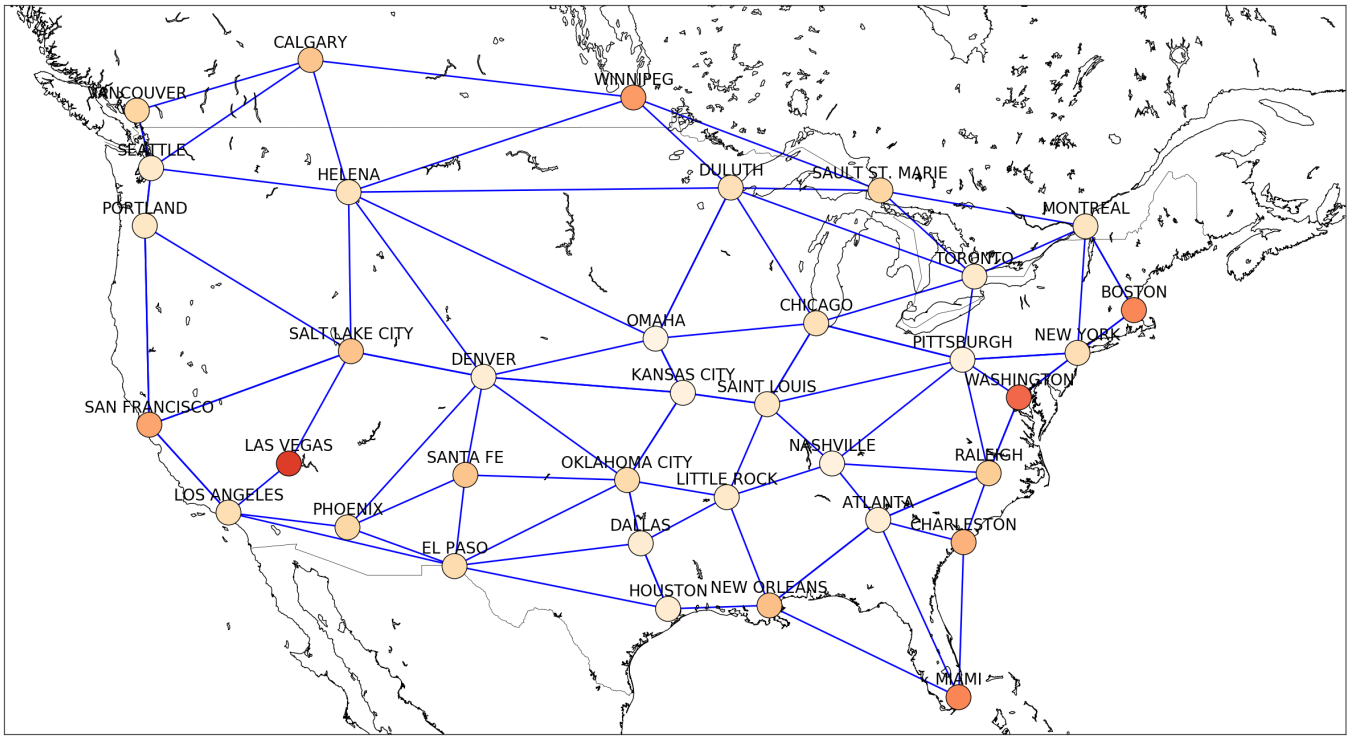

Figure 3: A color map of how often a city is claimed by players. A claimed city is one that has at least one route connecting it to an adjacent city claimed by the end of the game. On the map, the darker the city color is, the less desirable it is, meaning, there were more games where no routes connecting to it were claimed.

DISCUSSION AND CONCLUSION

With this work we were able to identify and analyze key aspects of the very popular modern board game Ticket to Ride and several of its variants. By emulating strategies through intelligent agents, we were able to simulate a concrete number of matches that allowed us to investigate the intricacies behind the game's dynamics. The results demonstrate that large scale AI-based testing and exploration has the potential to be useful in analyzing concrete loopholes in game rules, assessing features of its components and investigating more abstract strategic biases. This is potentially useful because it gives the designers the ability to promote certain strategies by testing to see how well it performs in the game they are creating, and tuning the game accordingly. This provides a great deal of freedom to designers who have a specific vision for their game's strategic landscape.

So far, we were only able to generate sizable amounts of data for the Destination Hungry and Route Focus Agents. Because of this, the data is biased toward those playstyles. We feel that the optimization requirements are general enough that the effect of this bias is not too severe, but we would ideally have a wider variety of algorithms, to confirm that these values would persist.

FUTURE WORK

We intend to investigate the possibility of collecting large amounts of data from a wider variety of algorithms. We also have reason to believe that this methodology could act as an effective evaluation function for an evolutionary algorithm designed to tune values in a game. This would work to automate a tedious but still essential step in the game design process. In addition to testing for fairness and rule integrity, designers could also use AI to promote certain playstyles, by tuning a game such that certain strategies play more effectively than others.

We believe that the work presented here may be useful for developing a more robust system that aids modern board game designers. We wish to build on the work presented to construct a system that explores the flexibility and constraints of the design. The designer could test how the system would react to a new constraint, such as limiting the number of Train Cards one player can have in her hand. The designer could also use it to tailor the follow of the game in a specific way, for example, he could evaluate that the game is taking more turns than he would like and so ask the system to change thresholds and make new simulations looking for a shorter play time.

Another future project we would like to explore is using our analysis to generate content or change the rules of the game. By having a set of components and rules that satisfies the designer, a system could be trained to generate new variants and expansions, for instance a new board. Another way of generating a variant would be to have the designer input a current board and ask the system to test changing its features toward a certain direction. For instance, he could give

the system a board and a Destination Deck and ask for it to be modified in such a way that would force players to claim routes that would be needed by others, and as such, increase the tension and interpersonal interactivity of the game.

ACKNOWLEDGMENTS

Authors thank the support of CAPES, Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brazil.

References

Browne, C., and Maire, F. 2010. Evolutionary game design. *IEEE Transactions on Computational Intelligence and AI in Games* 2(1):1–16.

Browne, C. 2008. *Automatic generation and evaluation of recombination games*. Ph.D. Dissertation, Queensland University of Technology.

Chaslot, G.; Bakkes, S.; Szita, I.; and Spronck, P. 2008. Monte-carlo tree search: A new framework for game ai. In *AIIDE*.

Days of Wonder. 2004. *Ticket to Ride*. https://en.wikipedia.org/wiki/Ticket_to_Ride_(board_game) Accessed: 2016-05-15.

de Mesentier Silva, F.; Isaksen, A.; Togelius, J.; and Nealen, A. 2016. Generating heuristics for novice players. *2016 IEEE Conference on Computational Intelligence and Games*.

Dormans, J. 2011. Simulating mechanics to study emergence in games. *Artificial Intelligence in the Game Design Process* 2(6.2):5–2.

Griepp, M. 2016. *Hobby Games Market Nearly 1.2 Billion in 2015*. http://icv2.com/articles/news/view/35150/hobby-games-market-nearly-1-2-billion Accessed: 2016-10-29.

Guhe, M., and Lascarides, A. 2014. Game strategies for the settlers of catan. In *2014 IEEE Conference on Computational Intelligence and Games*, 1–8. IEEE.

Heyden, C. 2009. *Implementing a computer player for Carcassonne*. Ph.D. Dissertation, Maastricht University.

Hom, V., and Marks, J. 2007. Automatic design of balanced board games. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*, 25–30.

Huchler, C. 2015. An mcts agent for ticket to ride. Master's thesis, Maastricht University.

Isaksen, A.; Gopstein, D.; Togelius, J.; and Nealen, A. 2015. Discovering unique game variants. In *Computational Creativity and Games Workshop at the 2015 International Conference on Computational Creativity*.

Isaksen, A.; Gopstein, D.; and Nealen, A. 2015. Exploring game space using survival analysis. *Foundations of Digital Games*.

Jacobsen, E. J.; Greve, R.; and Togelius, J. 2014. Monte mario: platforming with mcts. In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*, 293–300. ACM.

Jaffe, A.; Miller, A.; Andersen, E.; Liu, Y.-E.; Karlin, A.; and Popovic, Z. 2012. Evaluating competitive game balance with restricted play. In *AIIDE*.

Justesen, N.; Mahlmann, T.; and Togelius, J. 2016. Online evolution for multi-action adversarial games. In *European Conference on the Applications of Evolutionary Computation*, 590–603. Springer.

Kraaijenbrink, E.; van Gils, F.; Cheng, Q.; van Herk, R.; and van den Hoven, E. 2009. Balancing skills to optimize fun in interactive board games. In *Human-Computer Interaction–INTERACT 2009*. Springer. 301–313.

Krucher, J. 2015. Algorithmically balancing a collectible card game. Bachelor's thesis, ETH Zurich.

Liapis, A.; Yannakakis, G. N.; and Togelius, J. 2013. Sentient sketchbook: Computer-aided game level authoring. In *FDG*, 213–220.

Mahlmann, T.; Togelius, J.; and Yannakakis, G. N. 2012. Evolving card sets towards balancing dominion. In *Evolutionary Computation (CEC), 2012 IEEE Congress on*, 1–8. IEEE.

Nelson, M. J. 2011. Game metrics without players: Strategies for understanding game artifacts. In *Proceedings of the First Workshop on AI in the Game-Design Process*, 14–18.

Nielsen, T. S.; Barros, G. A.; Togelius, J.; and Nelson, M. J. 2015. General video game evaluation using relative algorithm performance profiles. In *European Conference on the Applications of Evolutionary Computation*, 369–380. Springer.

Pfeiffer, M. 2004. Reinforcement learning of strategies for settlers of catan. In *Proceedings of the International Conference on Computer Games: Artificial Intelligence, Design and Education*.

Robilliard, D.; Fonlupt, C.; and Teytaud, F. 2014. Monte-carlo tree search for the game of 7 wonders. In *Computer Games*. Springer. 64–77.

Salge, C., and Mahlmann, T. 2010. Relevant information as a formalised approach to evaluate game mechanics. In *Computational Intelligence and Games (CIG), 2010 IEEE Symposium on*, 281–288. IEEE.

Shaker, N.; Shaker, M.; and Togelius, J. 2013. Ropossum: An authoring tool for designing, optimizing and solving cut the rope levels. In *AIIDE*.

Smith, A. M.; Nelson, M. J.; and Mateas, M. 2010. Ludocore: A logical game engine for modeling videogames. In *Proceedings of the 2010 IEEE Conference on Computational Intelligence and Games*, 91–98. IEEE.

Smith, G.; Whitehead, J.; and Mateas, M. 2010. Tanagra: A mixed-initiative level design tool. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games*, 209–216. ACM.

Szita, I.; Chaslot, G.; and Spronck, P. 2009. Monte-carlo tree search in settlers of catan. In *Advances in Computer Games*, 21–32. Springer.

Yannakakis, G. N.; Liapis, A.; and Alexopoulos, C. 2014. Mixed-initiative co-creativity. In *Proceedings of the 9th Conference on the Foundations of Digital Games*.