The Implementation of Greedy Algorithms to Play Avatar Duel

Muhammad Ravid Valiandi (13518099)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
13518099@std.stei.itb.ac.id

***Abstract*—Avatar Duel is a card game developed as a project by a group of students of the Banding Institute of Technology (ITB). It's design is based on trading card games such as Yu-Gi-Oh and Magic: the Gathering. The cards and factions in the game are based on the animated series Avatar: the Last Airbender and Avatar: the Legend of Korra. The game is meant to be played by two players at a time. In similar computerized games, there is usually a computer player of some kind. The computer player uses an algorithm in order to determine its actions and play competently against a human player. Some algorithms are sophisticated enough to defeat skilled players. One type of algorithm commonly used for such computer players, is the Greedy Algorithm. This paper is going to discuss the implementation of a Greedy Algorithm to play Avatar Duel, which can be used as the basis of a bot or to help play the game.**

***Keywords*— Avatar Duel, Greedy Algorithm, Aggro, card game, player.**

I. Introduction

Avatar Duel is a two player card game designed by the students of the ITB class *Object-Oriented-Programming*. Its concept and rules are thought of by the lecturers of ITB, who then assigned it as a project for the students. The students were split into several groups, each one tasked to implement Avatar Duel using the Java programming language over the course of three weeks.

The game is designed to be playable by two players on the same laptop or personal computer. The players pass the computer between them upon changing turns. The game takes some inspirations from other card games before it, such as Magic: the Gathering and Yu-Gi-Oh!, though it is noticeably simpler than those two games when it comes to mechanics and card effects.

This paper will contain a discussion about the implementation of a basic greedy algorithm for the purpose of playing, and winning, a game of Avatar Duel using an aggressive strategy. This greedy algorithm can be used for the sake of planning in a game, or to program a computer player for Avatar Duel.

As there are several versions of Avatar Duel, each implemented by a different group of students, this paper will refer only to the version of Avatar Duel implemented by Group 8. Although the concepts discussed in this paper can be used in different versions of Avatar Duel, due to their identical basic rules, the implementation of this algorithm will inevitably differ.

II. Theory

A. *Avatar Duel*

Avatar Duel is a card game based on the animated series Avatar: the Last Airbender and Avatar: the Legend of Korra. Various concepts and characters from those works of fiction are represented in the game as cards or gameplay mechanics. The game uses art taken directly from both the shows and comics as illustrations for the various cards.

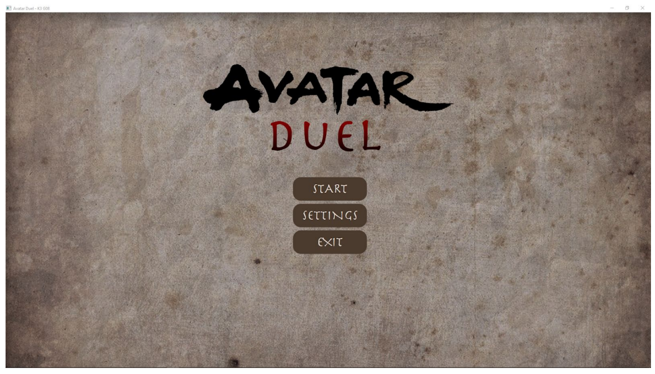

Image 1*: Start screen for Avatar Duel*

The board is split into two halves, with one player's side being on the lower half of the screen, and the other on the higher half of the screen. Each player's side of the board displays a number of things relevant to the game.

1. *A deck*. A collection of 40-60 randomized cards. When a player draws a card, they remove one card from the deck and place it in their hand. If a player's deck runs out of cards, they lose.

2. *A hand*. The cards a player can use. A player starts with seven cards in their hand, and draws an additional card each turn. There is no limit to the amount of cards a player can have in their hand at a time.
3. *The board*. The area summoned Characters and their attached Skills are placed. When a Character is summoned, the card is placed on the board. When a Character is destroyed, they are removed. Each player can have up to six Characters on the board at a time.
4. *Power*. The amount of Power that the player has, and may spend each turn. Power is divided among the five Element types. Each player begins with 0 Power in all Elements, but may increase the amount they can spend each turn by using Land cards.
5. *Health Points*. The amount of health points that each player has. Both players begin with 80 health points. The amount of health points a player is reduced when they are attacked by the opposing player's Characters. When a player's health points reach 0, that player loses.

The way the game works is notably similar to other card games, such as Magic: the Gathering and Yu-Gi-Oh!, with each player able to summon Characters and use Skills to enhance or destroy enemy Characters. Each player uses their characters to defend themselves or to attack the opposing player.

Each card is of one of five Elements: Air, Earth, Energy, Fire, and Water. Though the Elements do not have much differences mechanics wise, they do restrict the type of Power that can be spent for Character and Skill cards.There are three types of cards.

1. *Character cards* are cards that can be put on the field to attack the opposing player. When a Character card is used, it costs a certain amount of Power of the same Element as the character's. A Character possesses an Attack value, a Defense value, and a Power value. Their Attack value determines how much damage they deal to the opposing player or to a defending character, and their Defense value determines how much damage they can withstand each turn. Finally, their Power value determines how much Power must be used to summon the character. A summoned Character can be set to either attack the opposing player or defend their own, though characters are unable to attack on the same turn they are summoned. A player can have up to six Characters summoned at a time.
2. *Land cards* are the source of a player's Power. When used, the card adds one permanent Power of the card's Element to the player's total. This total is the amount of Power a player can spend in a turn. A Land card does not cost anything to use, but a player can only use one Land card each turn.
3. *Skill Cards* are cards that can be used on any Character on the board. They possess a Power value, which determines how much Power of their Element must be spent in order to be used. Skill cards come in three types, *Aura*, *Power Up*, and *Destroy*.
   a. *Aura* Skills modify the attack and defense of the Character they are attached to, either increasing or decreasing them. This is the most common type of Skill in the game. A Character can only have one Aura or Power Up Skill attached to them at a time, but a player may discard any Skill they control during the main phase.
   b. *Destroy* Skills outright destroys the Character they are used on. The targeted Character and any Skills attached to them are removed from the board.
   c. *Power Up* skills allow a Character to damage a player even through a defending Character. If the attacker's Attack value is higher than the defender's Defense value, any leftover damage is inflicted upon the opposing player. A Character can only have one Aura or Power Up skill attached to them at a time.

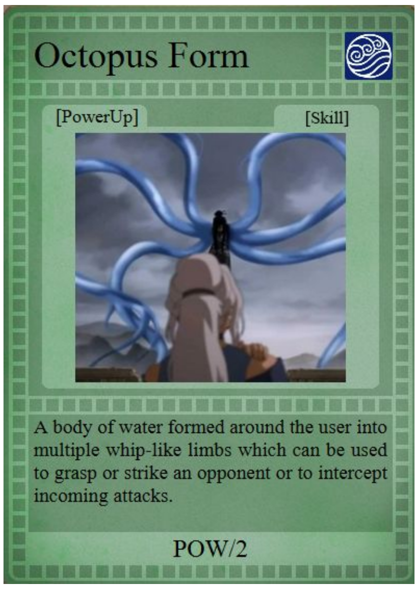
Octopus Form
[PowerUp]
[Skill]
A body of water formed around the user into multiple whip-like limbs which can be used to grasp or strike an opponent or to intercept incoming attacks.
POW/2

Image 2*: An example Avatar Duel skill card*

Avatar Duel is a turn based game, with each player switching between one another to take their turns. Each turn consists of four phases:

1. *The draw phase is* when a player draws a card from the deck and resets their available Power to its maximum value.
2. *The main phase* is when a player can use their cards. They may use up to one Land card each turn, and may use any number Character or Skill cards so long as they can afford the cost in Power. They may also remove any Skill card they controlled from the character they are attached to.

3. *The battle phase* is when all Characters the player has set to attack mode attacks. They either attack the opponent's Characters or the opposing player directly. Only Characters set to attack mode can attack, and each Character can only attack once per turn. A character cannot attack during the same turn they are summoned on. When a character attacks, a few things need to be considered:
    a. If there are no opposing Characters in defense mode, attacking Characters can attack the opposing player directly.
    b. If there are defending Characters, the attackers must destroy them first.
    c. If a Character attacks the opposing player, they reduce their health points by an amount equal to the attacker's Attack value.
    d. If a Character attacks another Character, the defending character and any Skill attached to them are destroyed and removed from play.
    e. A Character cannot attack a defending Character whose Defense value is higher than their Attack, or a non-defending Character with an Attack value higher than theirs.
4. Finally, *the end phase* is when the turn wraps up and ends. After this, it becomes the opposing player's turn.

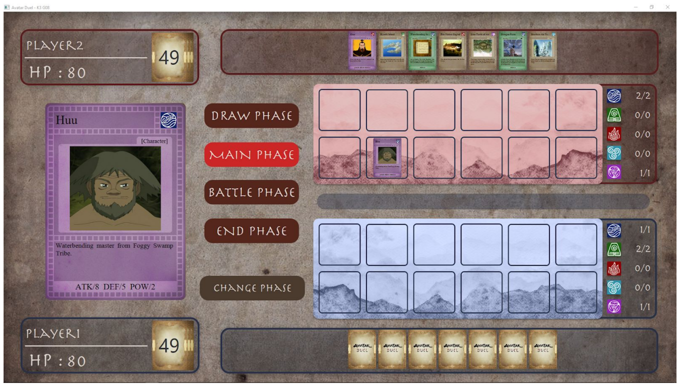

***Image 3**: Avatar Duel gameplay*

To summarize, the way Avatar Duel works is quite simple. Both players are given a deck of cards to use, which can have between 40-60 cards depending on the setting of the game. Each player begins with 80 health points, 0 Power for all Elements, and seven cards in their hand.

Players take turns to play the game. Each turn, they draw a single card from their deck and may use a land card to accumulate Power. Power can be used to summon Characters to attack the opposing player or to use Skill cards to affect the Characters that are currently on the board.

Characters are used to deal damage to the opposing player, which will reduce their remaining health points. The first player to reach zero health points or to run out of cards in their deck, loses.

B. *Aggro*

Aggro, short for "aggressive", is a major strategy archetype used in trading card games like Magic: the Gathering and Yu-Gi-Oh [2]. The strategy is based on the brute force method when it comes to winning the game. Rather than any complex plans or combos, Aggro focuses on dealing as much damage as possible to the opposing player as quickly as possible.

Aggro decks have a tendency to eskew defense in favor of pure offense. The concept of the strategy is that any damage the player taks doesn't matter so long as the opponent hits 0 health points first.

An Aggro strategy uses massive amounts of damage in order to quickly overwhelm the opposing player before they can mount a proper response. This makes Aggro an effective counter to more complicated and drawn out strategies that can be employed by other players. It also makes Aggro a very predictable strategy, as its entire plan boils down to attacking the opposing player over and over until their health points drop to zero.

In games like Magic: the Gathering, Aggro decks would rely on summoning as many creatures as possible as quickly as possible and using various damaging spells to finish off whatever health points the opponent has left [3]. Similarly, in Avatar Duel an Aggro strategy would depend on summoning Characters as quickly as possible. Skills are used either to enhance the player's Characters or to remove defending Characters. Though the lack of any directly damaging Skills would make it harder to deal direct damage to the opponent.

C. *Computer Players*

In games there is often the option of playing with only a single player, even if the game is usually played by multiple. This is possible due to *computer players* or *bots*. Computer players are players that are controlled by the computer using algorithmic and responsive behaviour, though not necessarily true artificial intelligence.

Computer players can be used to play against human players to improve their gameplay experience by acting as an engaging opponent. The more sophisticated a bot and its programming is, the more engaging an opponent it becomes for the player. Bots may also be used to play against other computer players for research or for competitive programming [4].

The intelligence of a computer player depends on its programming. The more complex and intricate a computer player's programming becomes, the more intelligent its actions. Some advanced bots can make use of machine learning in order to learn the patterns of their opponents and slowly improve over time. Other bots simply rely on a small script of actions, making them much more predictable.

The computer bots proposed by this paper lies somewhere in between machine learning and a static script in terms of complexity. Rather than dynamically learn the patterns of opposing players, the bot discussed in this paper will use a greedy algorithm in order to analyze the gamestate and determine the most optimal action at each step of the process.

D. Greedy Algorithms

A greedy algorithm is an algorithm designed to optimize results, either minimizing or maximizing it depending on the program's purpose. The problem the algorithm tries to solve is split into various steps, and at each step the algorithm compares all of its options and always takes the most optimal decision at that individual step. The idea of the algorithm is that, if it takes the optimum choice for each local step, the result would be the most optimum choice overall.

A greedy algorithm can be programmed in a number of ways for any given problem. Generally, this type of algorithm is made up of five components:

1. *A candidate set*, which consists of the elements that can form a solution;
2. *A solution set*, which contains the elements that are chosen to contribute to a solution;
3. *A selection function*, which chooses the most optimum option available at each individual step of the process;
4. *A feasibility function*, which determines if a candidate can be used to contribute to a solution;
5. *An objective function*, which assigns value to a solution; and

The process of a Greedy algorithm is as follows:

1. The algorithm considers each element in the candidate set using the *feasibility function*.
2. The *selection function* chooses the element with the highest or lowest value, as determined by the algorithm.
3. The chosen element is added to the *solution set*.
4. Repeat 1-3 until the *solution set* is completed.

A greedy algorithm takes the most optimum choice available at a step in the program, before moving on to the next step. It only ever considers the available choices and the choices that have been made so far. It does not consider the choices that will be made in the future, and it never reconsiders a decision after it has been made. A greedy algorithm is not an exhaustive search, and is therefore liable to miss something.

Additionally, for each implementation of this algorithm there can be numerous possible *selection functions*, as there can be multiple ways to count an element's value. This means that it may not return the best result. In some cases, the optimum local solutions do not form the most optimum solution overall, and instead return a sub-optimum or *pseudo-optimum* solution.

This is the main difference between greedy algorithms and dynamic programming. Where greedy algorithms will only make one choice after another, dynamic programs will exhaustively consider all options until it finds the most optimum solution. While greedy algorithms can not reconsider options after making a choice, a dynamic program will always do so.

A greedy algorithm is better than more complex algorithms like dynamic programming when the absolute best solution is not required, and an approximation of the optimum solution would function just as well. Greedy algorithms are often used when creating bots or computer players for games. Greedy algorithms are well suited to playing games, as the situation in games are often constantly changing and most games do not allow an algorithm to backtrack or reconsider a move after it has been made.

Such AIs can be made more intelligent by using more complex *selection functions* in order to make their moves seem more thought out. Though of course, more complex algorithms are harder to implement.

III. Implementation and Discussion

As explained previously, a greedy algorithm is an algorithm that makes the most optimum decision at each step in order to make the most optimum solution possible. When used in a game like Avatar Duel, this means that the algorithms would choose which cards to use each turn in order to win.

1) Implementing Aggro Strategy

The following algorithm implements an Aggro strategy in order to win the game as quickly as possible. For clarity's sake, the player who is using the algorithm is simply referred to as the player, while the player playing against them is referred as the opposing player.

The algorithm consists of a few parts:

a) Candidate Set (C)

The algorithm would require two candidate sets, one for the main phase and one for the battle phase. The main phase's candidate set would consist of the player's hand, with each card being an element to be considered. The battle phase's candidate set would consist of any of the player's Characters that are able to attack during that phase.

b) Solution Set (S)

The solution set of the algorithm would keep track of the gamestate. It would track both player's health points, as well as their summoned characters, their Attack and Defense values, and any skills attached to them. The solution set is considered complete once a player has met a loss condition, either having 0 health points left or by having no cards left in their deck.

c) Selection Function

Due to the way the game works, there would need to be two selection functions. One for the main phase and one for the battle phase.

The main phase selection function would choose which cards to use. The selection function would prioritise using a land card, if the player has one in their hand, followed by character cards, and finally skill cards.

The following table is the ladder that prioritises which cards are used during the main phase. Cards that meet criteria higher on the table are given a higher value compared to cards that meet criteria placed lower on the table.

Card Type	**Criteria**
Land	Land's Element matches that of the lowest cost Character or Skill card in the player's hand.
	Land's Element matches that of another Character or Skill card in the player's hand.
	Any Land card in the player's hand.
Character	Character with a high Attack value.
	Any Character in the player's hand
Skill	Aura skill that increases Attack value.
	Power Up skill.
	Destroy skill.
	Aura skill that increases Defense value.
	Aura skill that decreases Attack or Defense.

Table 1: Main phase priority ladder

Depending on what kind of card is selected, the algorithm would act somewhat differently:

1. *Land card*. The algorithm always selects a Land card first during each turn. If the candidate set contains multiple Land cards, it would choose whichever one had an Element that matches another Character or Skill card in the candidate set, with priority given to the Character or Skill card with lower costs.
2. *Character card*. The algorithm would always prioritise summoning Characters with high Attack values as quickly as possible.
3. *Skill card*. The algorithm prioritises using Aura skills that enhance Attack value and Power Up skills on the player's Characters. Slightly less priority is given to using Destroy skills on opposing Characters and using Aura skills that increase Defense on the player's Characters. The algorithm does not prioritise using Aura skills that decrease Attack or Defense values on an opposing Character.

The battle phase selection function is used to determine which characters attack what target. The function always prioritises attacking the opposing player directly when there are no defending characters. If there is a defending character in the way, the function would prioritise attacking Characters with the highest Defense value and attacking them with the weakest available attacker that can still destroy them.

The following table is the ladder that prioritises which actions are taken during the battle phase. Actions that meet criteria higher on the table are given a higher value compared to actions that meet criteria placed lower on the table.

Target	**Criteria**
Opposing Player	There are no Characters defending the opposing player.
Defending Character	Defending character's Defense value is high.
	Attacking Character's Attack value is equal or near to defending Character's Defense Value.
	Any attacking character with an Attack value higher than the defending Character's Defense value..

Table 2: Battle phase priority ladder

d) Feasibility Function

The algorithm's feasibility function would ensure that the player's actions are within the constraints of the game's rules. When the selection function considers an option, the feasibility function would disqualify options such as:

1. Using a Land card more than once per turn.
2. Using a Skill or Character card that costs more Power of a certain Element than the player has available.
3. Using a Skill card on a character that already has a Skill attached to them.
4. Attacking a defending Character whose Defense value is higher than the attacker's Attack value.
5. Attacking with a Character more than once per turn.
6. Attacking with a Character that was summoned on the same turn.

e) Objective Function

The objective function of the algorithm would determine if the game has resulted in a win, a loss, or if the game continues on. It tracks both player's health, as well as how many cards each deck has remaining. Once one of the players has reached a loss condition, either having zero health points left or running out of cards, the game has reached its conclusion.

f) Algorithm

The way the algorithm works is that each turn is considered one step in the process, with the process only ending once one of the player's health points reaches 0 or one of their decks runs out of cards.

Generally, the process of the greedy algorithm for this strategy is as follows:

1. The game begins, both players start with their deck, 80 health points, and 0 Power in all Elements. Each player draws seven cards.
2. The player's turn begins, they enter their draw phase. They draw one card and refresh their available Power.
3. The player enters their main phase. The algorithm plays their cards, with those with higher priority being used first. This repeats until there are no more cards that are feasible to use.
4. The player enters their battle phase. The algorithm attacks the opposing player and any defending Characters the opponent has. This repeats for each attacking CCharacter the player has.
5. The player's turn ends, and the opposing player takes their turn.
6. Repeat 2-5 until one of the players either loses all of their health points or they have no cards left in their deck.

2) *Implementing Other Strategies*

Other strategies can be implemented similarly to Aggro, it would only require modification of the *selection function* to do so. Though how much modification is required depends on how complex the implemented strategy is.

Because Aggro is a relatively simple strategy, the *selection function* required to simulate it is relatively simple as well. Implementing more complex strategies, ones that make better use of Skills or the defense mechanics, is possible. Although it would take a far more complex algorithm in order to implement them

There is also the possibility of using different types of algorithms to implement a bot that uses an Aggro strategy. For example, a machine learning algorithm can practice against other opponents using this strategy over the course of multiple matches, improving its skill in the game over time.

.

IV. Conclusion

A number of conclusions can be drawn from this paper:

1. Avatar Duel is a card game based on Avatar: the Last Airbender and Avatar: the Legend of Korra. It is based on similar card games such as Magic the Gathering and Yu-Gi-Oh!.
2. An Aggro deck is a strategy in card games wherein the player focuses on dealing as much damage to their opponent as possible, as quickly as possible. Aggro strategies can be quite effective, as they can end the game before their opponent can complete any long-term plans.
3. A bot or computer player is often used in games in order to enhance the player's experience. The bot provides an engaging opponent to the player, allowing them to test their skill against the program.
4. Some bots are made using machine learning algorithms, allowing them to improve their strategies over time. Other bots are more static, relying on algorithms to determine the most optimal course of action at any time.
5. A Greedy Algorithm can be used in the creation of game computer players and bots, due to the way that a game is normally run. By using a complex *selection function*, it is possible to create AI opponents that can seem like they actually think their moves through. Though such complex AIs can be quite difficult to actually implement.
6. A Greedy Algorithm can be implemented to create a bot that can play Avatar Duel using a pure Aggro strategy. Other bots that make use of other strategies can be made as well, though it is likely that their complexity compared to Aggro would mean that a more complicated algorithm is required.

Video Link at YouTube

This paper is created along with a YouTube video explaining the concepts here, such as Greedy Algorithms, bots, and Aggro strategies, and various other strategies that can be used in a card game like Avatar Duel.

For further information, click this link in order to watch the video: https://youtu.be/0v2pJWbAesg.

Acknowledgment

Saya ingin mengucapkan terimakasih kepada Tuhan Yang Maha Esa, untuk membuat proses pembuatan makalah ini lancar sampai penyelesaiannya. Saya juga ingin memberi terimakasih ke orang tua saya, untuk memasukkan saya di kuliah ITB dan untuk dukungan mereka dalam meraih mimpi saya.

Saya juga ingin memberi terima kasih ke Dr. Ir. Rinaldi Munir, MT, dosen Strategi Algoritma saya, untuk membuat presentasi Algoritma Greedy yang sangat membantu dalam pembuatan makalah ini, dan untuk mengajari saya walaupun susah berkonsentrasi di kelas.

Akhirnya, saya ingin memberi terima kasih ke teman-teman sekelas saya, untuk membantu saya mendapat ide makalah ini dan untuk membantu saya membuat game Avatar Duel untuk Tugas Besar sebelumnya.

References

[1] Munir, Rinaldi. Algoritma Greedy (2020).

[2] https://www.channelfireball.com/all-strategy/articles/magic-the-gathering-has-three-major-archetypes-aggro-combo-and-control/, Accessed on the 3rd of May 2020.

[3] https://magic.tcgplayer.com/db/article.asp?ID=15602, Accessed on the 3rd of May 2020.

[4] GameBots: A Flexible Test Bed for Multiagent Team Research, Accessed on the 4th of May, 2020

[5] https://cai.tools.sap/blog/bot-in-video-games/, Accessed on the 4th of May 2020

[6] https://www.newscientist.com/article/mg22329884-300-buttonmasher-gaming-bots-learn-from-how-we-play/, Accessed on the 4th of May 2020

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 29 April 2020

[illegible]

Muhammad Ravid Valiandi