JRPG Battle Tactic Development with Depth First Search Algorithm

Aristoteles Swarna Wirahadi (13516146)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13516146@std.stei.itb.ac.id

***Abstract*—The topic that would be discussed in this paper is how Depth First Algorithm could be applied in battle tactic formulation for video games, especially Japanese role-playing game or JRPG. The experiment for this topic was done by taking a sample boss battle from one of the games of the mentioned genre. The sample was then tweaked a little and used as a simulation for the concept that will be discussed. The result was enough to prove that the DFS approach in battle tactic formulation is feasible. After reading this, readers should have had some grasp on the concept and are able to independently design their own set of tactics to beat bosses in JPRGs of their choice without consulting sources from the internet, especially the so-called walkthroughs or game guides.**

***Keywords*—DFS, tactic, boss, hero, battle, stage, BTD**

I. Introduction

Japanese Role-playing Game (JRPG) is a genre of video games that started to exist in the 1980s. Some of the first titles that were published were Final Fantasy and Dragon Quest. These games are popular for their unique battle systems, mostly compelling plots, and amazing soundtracks. One of the main features of the genre is having boss fights or battles ranging from easy to arduous ones. Some battles might be emotional because the boss itself may someone who used to fight alongside the protagonist. This is what makes Japanese role-playing games interesting. They never cease to surprise the players with a lot of plot twists that blow the players away.

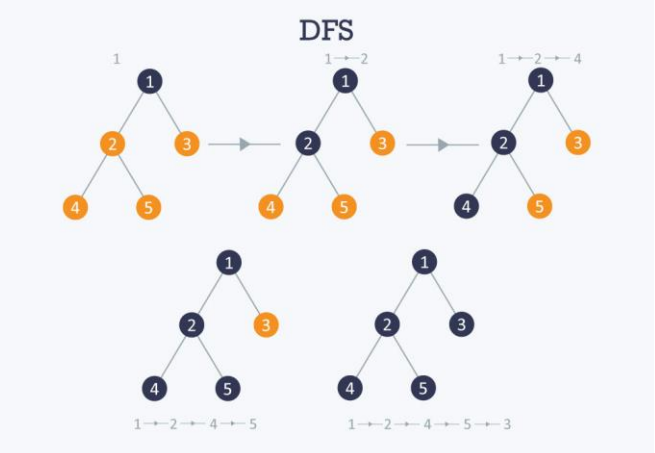
Typically, some end-game boss battles include several stages. For each stage, the boss has a certain amount of health points (HP). After the HP is depleted during that stage, the battle is not over. Instead, it proceeds to the next one. In addition to that, the boss usually becomes stronger and deadlier. Players not expecting this to happen might be overwhelmed by the fact if their heroes have so much strength left before the boss annihilates them with its special attack in the next stage. Consequently, players are stuck on that point and are unable to progress in the game.

Nowadays, this is easily overcome with the help of the internet. Players could easily find forums or walkthroughs that contain several different ways to defeat particular bosses with or without difficulty. Forums are pages containing threads posted by users from around the world. These users express their opinions by posting threads in these pages. Looking at this fact, forums could be seen as discussion pages. The reason is that a forum mainly focuses on one topic. Some players also utilize walkthroughs, which are text-based game guide. They have different formats and focuses on different aspects of the games they are covering. Today, tons of them are available online. With the prevalence of walkthrough, playing any JRPG will be a breeze. However, there might be a possibility that players become too dependent on game guides if players use them too much. Their mindset might change to something that sounds like this: “I couldn’t get past this point, I’ll just immediate Google the walkthrough, I’m too lazy to try and find a way to get past this myself.” This is a minor problem, but a problem nonetheless. Those players might make their own gaming experience less rewarding, which takes away the fun.

This paper will be covering the use of depth-first search (DFS) algorithm in developing tactics for JRPG boss battles. The purpose of this paper is to encourage players to rely less on walkthroughs and give them a better and much more rewarding experience when playing JRPGs.

II. Fundamental Concept and Theory

Depth First Search (DFS)



State Tree Diagram
Source: https://he-s3.s3.amazonaws.com/media/uploads/9fa1119.jpg

Given the diagram above, the circles and the numbers inside them are the nodes and values respectively. Simply put, we traverse through the nodes from the root (node with the value 1) to the deepest level possible and then backtrack by moving back to the previously traversed node and visit (traverse to) unvisited nodes. If on the current level we don't find any unvisited nodes, we again execute backtracking. The traversal direction depends on the conditions of a given case. When the goal or solution node is visited, the DFS algorithm terminates.

To give a better understanding of the algorithm, refer to the diagram above as an example.

The value of the node will be referred to as the number of the node for the rest of this paper. For example, a node with value 11 is referred to as node 11. Let's assume that the direction of traversal is always left if possible and let the goal node be node 3. The starting point will be the root of the tree, which is node 1. First, node 1 will be visited. The traversal will proceed to node 2 and then node 4. Now, the traversal has reached the deepest level of the tree. Backtracking is initiated. We move back to node 2, and then we visit the unvisited node 5. Again, backtracking is executed. We now move back to node 2 again, then node 1 which is where we started. What is the remaining unvisited node here? The answer is node 3. We visit the node, 'find' the goal node and terminate the DFS algorithm.

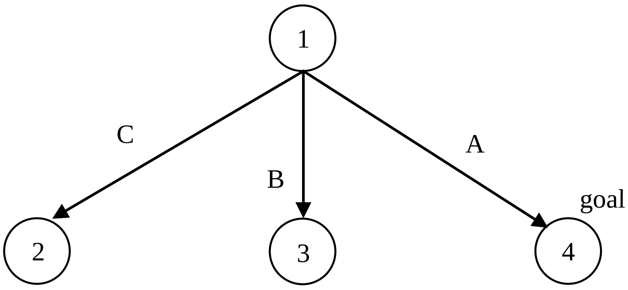
III. Trial-and-error Concept in Tactics Development

Before trying to understand what the trial-and-error concept is, try understanding the fundamentals of a typical, generic boss battle found in most JRPGs.

A boss battle is a battle between two opposing sides. Let's give them names. The good guys are called 'heroes' and the enemy individual is called 'boss.' The hero's objective is usually to defeat the boss by any means available. The two opposing sides switch turns every time each side performs an action. The boss is usually stronger and more durable than each of the heroes, making it a more challenging battle. The heroes and the boss would trade blows until one of the two sides is defeated, then the battle ends – either the game shows a Game Over screen or it shows the aftermath.

If the battle ends with a Game Over screen, players know that their previous tactics don't lead the heroes to victory. They have to try new tactics to see if they could change the outcome. That is where the trial-and-error comes in. This can be done by applying the Depth First Search algorithm. The example below will illustrate how DFS algorithm simulates the trial-and-error concept.

For example, there is a one-on-one battle between the hero and the boss. The hero goes first, then the boss. They both switch turns every action. The boss will always attack the hero whenever he gets his turn. Let's assume the boss battle lasts only one stage. When the boss' health point is reduced to zero, the battle ends and the player wins. The methods used to beat the boss can be various. Let's say the tactics available to the players are: "Attack every turn", "Boost attack every turn", and "Choose Heal every turn". The tactics will be represented by edges (A, B, and C respectively) that stem from the root of the tree, assuming that we can only use one tactic for the whole battle. The nodes here are the current states during the battle, which won't be focused on for this example.



Example Diagram

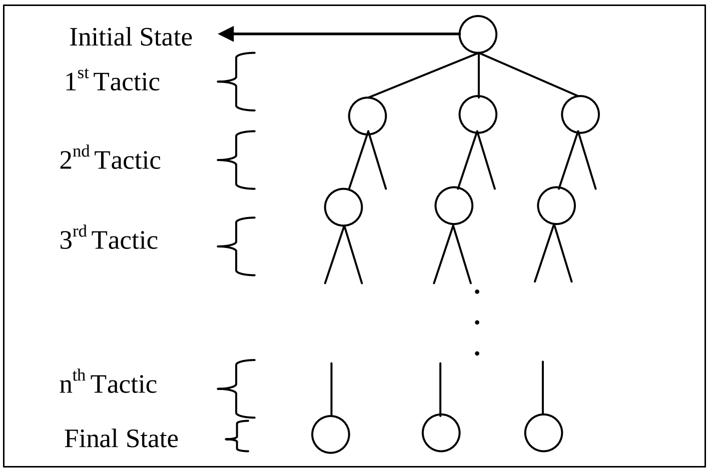
This diagram represents the experience of a certain player, where he developed his tactics based on the mistakes he made when using particular tactics.

Now, let's analyze the diagram above with some common sense and with the perspective of DFS. The starting point would be node 1, the state in this node is where the battle had just begun. The player tried using tactic 'C', which is "Choose Heal every turn". Now, think about it, can the boss be defeated if the hero heals himself every turn while the boss attacks him? The answer is no, because the boss' HP won't be depleted any time soon. That is why node 2 is not the goal node. The player realized his mistake and backtracked to node 1. He then tried tactic 'B', one of the remaining tactics, which is "Boost attack every turn". Again, node 3 is not the goal node because of the same reason as the previous tactic. Realizing his second fault, the player tried the remaining tactic, tactic 'A' ("Attack every turn"). With a little effort, the player finally beat the boss fair and square. Hooray!

The example shows how similar the DFS algorithm is to the trial-and-error concept. When we make a mistake (don't find the goal node), we learn from it. After that, we try again (backtrack) with a different input (visit unvisited nodes) to see if the output could be different too (is the possibly deepest level node the goal node?). However, this illustration only covers a small part of the real battle tactics development concept. This is only the basics. The elaboration of the whole concept will be available in the next section.

Source: https://img00.deviantart.net/6426/i/2010/280/9/b/versus_the_dragon_by_arkangel_wulf-d30aem1.jpg

IV. Battle Tactics Development (BTD) Model



Fundamental Battle Tactic Development Model

The diagram above represents the basic structure needed to develop battle tactic, in this case, for boss battles. Players build this structure as they play, either mentally (in their minds) or physically (they write it down on a piece of paper). This model represents one stage of a boss battle. A stage starts from the initial stage and ends at the final state.

The initial state is the state where the boss has 'full' health. The final state is where either the boss is 'defeated' or the heroes are unable to continue fighting. Notice the apostrophes on the words 'full' and 'defeated'. The reason for that is because there are boss battles with more than one stage, where the boss doesn't die immediately after its HP reaches zero. Instead, the boss might change forms and become deadlier than the previous form, resetting its HP to a certain maximum. This type of boss is mainly encountered near the end of a JRPG.

There will be many nth tactic edges rather than just one or two. It all depends on how many tactics the player could think of at the moment. The BTD structure is going to be dynamic. This means that it is always changing through time. When the players think of a possibly new tactic, they add it into the structure. This increases the combinations of tactics to choose from.

Tactics are divided into 2 categories. The first category is the "main tactic". It is the most important tactic and is usually dependent on the objective. For example, the objective is to defeat the enemy. The main tactic would be trying to attack the enemy as many times as possible until it is defeated, hence the name.

The second category is the "supporting tactic". The supporting tactic is the one that helps the execution of the main tactic, the one that increases the chance of winning. In other words, it increases the success rate of the main tactic. For example, let's say that our main tactic is to attack the enemy as many times as possible. Now, every time the enemy suffers damage from the player's party, it accumulates power. When the enemy has enough power accumulated, it will release a very powerful attack at the party, which will almost kill them all. Knowing this fact, the supporting tactic may be something like, "Deploy a shield when the enemy is about to release the powerful attack". This way, the player could still stick to the plan, constantly executing the main tactic mentioned before. If it is not possible to win with the current main and supporting tactic, the player would have to alter either the main, supporting or both tactics. If all else fails, the player might have to power up the hero party, generally by grinding (defeating low-level enemies to easily gain experience points a.k.a. 'exp' to increase the strength level of the party)

V. Final Fantasy X

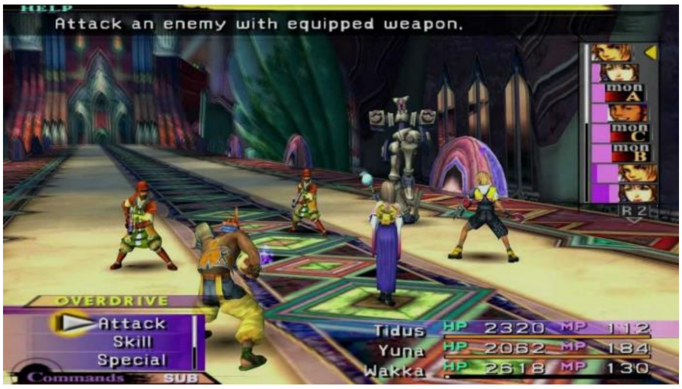
Before jumping into the real application of the model discussed so far, let's have a look at one of the author's JRPG of choice, Final Fantasy X.

Final Fantasy X DVD Cover
Source:
https://upload.wikimedia.org/wikipedia/en/thumb/a/a7/Ffxboxart.jpg/220px-Ffxboxart.jpg

As seen in the video game cover above, Final Fantasy X is a Japanese role-playing game released for the Playstation 2 home console. There are also re-releases for the Playstation 3 and Playstation Vita, Playstation 4 and finally, Microsoft Windows (in year 2013, 2015, and 2016 respectively). The game is developed by Squaresoft, with Yoshinori Kitase as the director, Nobuo Uematsu as the music composer, and Tetsuya Nomura as the character designer. For your information, Tetsuya Nomura is also the director of the upcoming Disney and Final Fantasy crossover game, Kingdom Hearts III (said to be released this year – 2018).

"Listen to my story. This… may be our last chance" – Tidus

The game follows the story of Tidus, a professional blitzball (you could say it is like American football but played inside a sphere of water) player in a city called Zanarkand. One day, while he was in a blitzball match, a giant monster – called Sin – arrived at Zanarkand and started wreaking havoc to the city. As a result, the citizens entered a state of panic and start scrambling around Zanarkand while Sin continues to crush the buildings. The blitzball match was cancelled. Tidus then exited the building he was in and started to process what was happening at the moment. Not long after, a man who seemed to be acquainted to Tidus – Auron – came to him and together, they tried to retreat to a safe place while cutting down obstacles with their swords. Finally, they came to a dead end with monsters chasing behind them. Suddenly, a bright light shined from the sky. Tidus and Auron were both sucked into the light and transported a thousand years into the future, to a place called Spira. This is the place where Tidus reunited with Auron, met different people and befriended them, leading him to befriending a female summoner named Yuna. A summoner is the one with the responsibility to free the whole world from Sin. Tidus and Auron journeyed together with Yuna and her friends to defeat Sin, and fulfill Yuna's duty.



HELP

Attack an enemy with equipped weapon.

Regular Battle Screenshot
Source: https://i.ytimg.com/vi/-NKxaTpEsv8/maxresdefault.jpg

Referring to [2], this game features the use of Conditional Turn-Based Battle system. In this system, the actions taken by an individual character determines when he / she will get his / her turn next. The turns of each individual, be it enemy or ally, could be seen in the window on the right-upper corner. The next turn would be of the individual on the top of the window, the one pointed by a yellow triangular arrow. The order of the turn may change according to the action taken by one of the characters in battle.

Each character becomes stronger by spending Ability Points (AP), which is gained through victory in battles, in the Sphere Grid. A Sphere Grid is a grid full of skills for different characters to be unlocked using AP. In addition to these new features, players can switch one of the characters in the active party (in battle) with one of those in the inactive party (not in battle).

Every playable character in the game has a skill called Overdrive. This skill enables the character to perform attacks deadlier than their normal attacks. The requirement to achieve this is that the Overdrive meter be full (notice the bar below every character's Health Points (HP) and Magic Points (MP)). Full Overdrive meters are indicated by their orange color. There are several ways to fill the Overdrive meter. By default, the character fills the meter by taking damage.

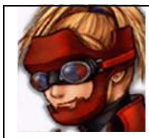
VI. Application of the Battle Tactic Development Model

The application of the Battle Tactic Development model will be simulated with a modified sample boss battle taken from Final Fantasy X. One thing to keep in mind is that the outcomes of the battles are determined intuitively for illustration purposes. The outcomes stated here aren't necessarily true.

The following are the abilities and the roles of each individual participating in the boss battle. The abilities listed below are only the ones that will be discussed.

 TROS (boss)	Attack
	Nautilus Charge
	Retreat
	Picture Source:[5]

 TIDUS (hero 1)	Attack
	Potion
	Picture Source:[6]

 ???? (hero 2)	Potion
	Steal
	Grenade
	Picture source:[7]

Auron
Source:
https://cdn.mos.cms.futurecdn.net/pazFN5Wf5GcbX8bDzVnHLd.jpg

The descriptions of all the abilities from each individual presented above are as follows:

1. Attack : Deal damage to the target
2. Nautilus Charge : TROS' high-damage attack that is executed one turn after it performs 'Retreat'
3. Retreat : Move out of range of the player's party
4. Potion : Heal / Restore HP of the target
5. Steal : Steals an item from the target (in this case: Grenade), not always successful
6. Grenade : Deal high damage to the target with a grenade

From the heroes' abilities, a few tactics could be generated. Keep in mind the definition of 'tactic' here is an action performed almost repeatedly to achieve the completion of the objective. Main Tactics (MT) should always come before Supporting Tactics (ST). Main tactics may act as Supporting Tactics, but not vice versa. The following are the generated tactics. The format is {<executor><ability><target>}.

1. TIDUS Attack TROS (MT)
2. ???? Steal + Grenade TROS (MT)
3. TIDUS Potion ???? (SS)
4. ???? Potion TIDUS (SS)

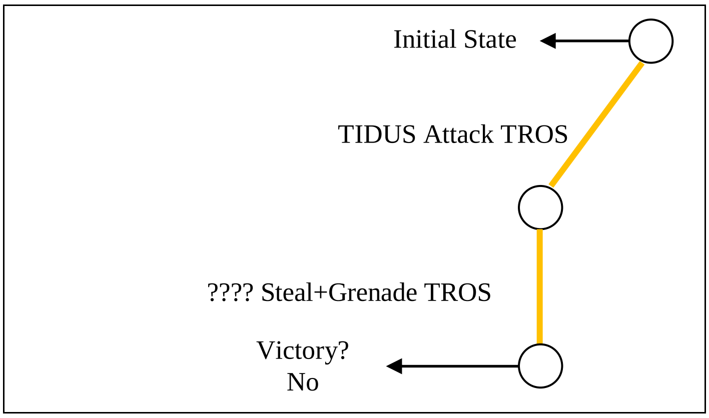
For this simulation, assume that the solution combination of tactics is: {???? Steal + Grenade TROS, TIDUS Potion ????}

Attack an enemy with equipped weapon.

Boss Battle Screenshot

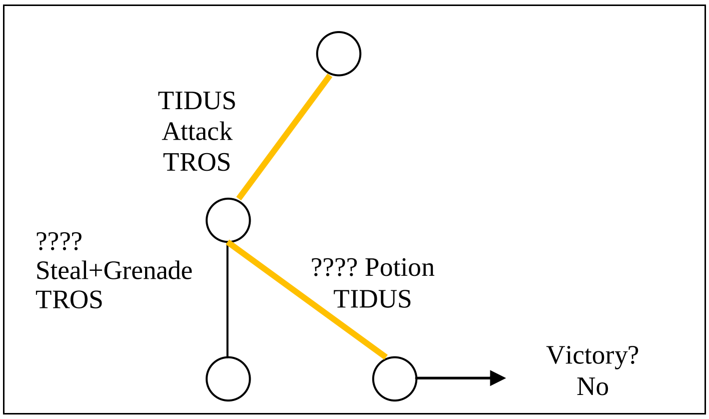
Source: https://lparchive.org/Final-Fantasy-X-(by-The-Dark-Id)/Update%2010/37-shipwreck55.jpg

Since there are 2 people in the heroes' party, at least 2 tactics have to be used. Assume the party is at a high enough strength level to defeat the boss. Also, assume each party member can only use one of their own abilities in each tactic combination. Initially, the first tactic combination is presented in the diagram below.



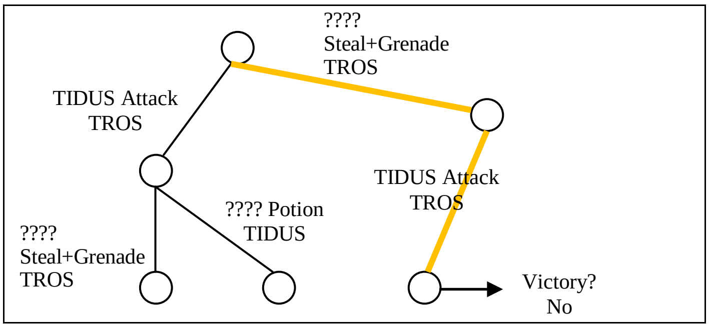
Initial / First Combination

Looking at the situation displayed in the diagram above, the combination of two tactics above would not work because TROS will always be attacking the party members every turn and has the chance to deal high damage with the Nautilus Charge. Without any healing the party would have been wiped out before they could defeat TROS. To counteract this situation, how about we have ???? heal TIDUS every turn?



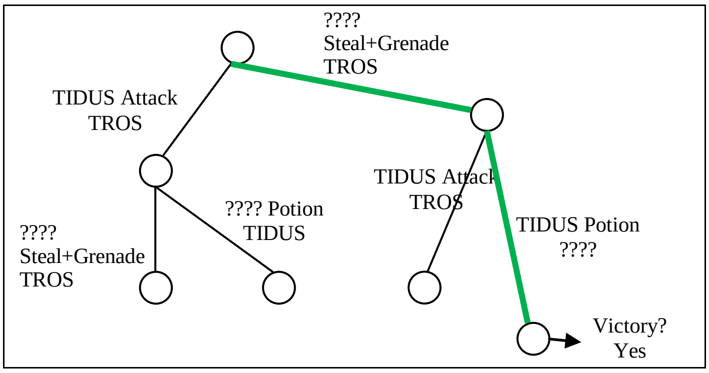
Second Combination

The current tactics are still not working. The reason is that TIDUS' regular attack quite weak and potions are very limited. Healing constantly would lead to the initial situation where the effort to heal party member is absent. This leads to the same outcome as before, which is the demise of all the party members. Try another possible combination.



Third Combination

This combination is no different than the initial combination. Try again.



Final / Fourth Combination

By having ???? deal high damage to TROS while TIDUS heals her, the boss would surely be defeated in time before the HP of each party member drops to zero due to the damage dealt by TROS' Nautilus Charge.

The final solution combination for defeating the boss would be {???? Steal + Grenade TROS, TIDUS Potion ????}, indicated by paths highlighted in green. The conclusion of this simulation*: It is possible to develop battle tactics with Depth First Search algorithm.*

Conclusion

There are many ways to discover the different methods to beat any video game. One of them is by trial-and-error. The better trial-and-error approach is by applying Depth First Search algorithm into it. If the current tactics don't succeed in winning the game, try changing one or two and try again. With the help of intuition, the execution of DFS algorithm may be shorter in terms of execution time. This kind of approach might not be perfectly effective, but it is the basis of good trial-and-error.

Acknowledgment

The author thanks Dr. Nur Ulva Maulidevi for giving him the knowledge he needs to become a better program designer and for being such a kind and caring figure for class K-02. The author also thanks Dr. Rinaldi Munir for the opportunity to express his opinions and thoughts through this paper. Dictionary.com, GameFAQs, and Youtube are also credited for providing the information to support the writing of this paper.

References

[1] Depth First Search, https://www.hackerearth.com/practice/algorithms/graphs/depth-first-search/tutorial/, accessed: May 11th, 2018

[2] http://finalfantasy.wikia.com/wiki/Final_Fantasy_X, accessed: May 11th, 2018

[3] https://gamefaqs.gamespot.com/ps2/197344-final-fantasy-x/faqs/16895, accessed: May 12th, 2018

[4] https://www.youtube.com/watch?v=6SS1hJ02EWo, accessed May 12th, 2018

[5] https://vignette.wikia.nocookie.net/finalfantasy/images/7/73/Tros_artwork.jpg/revision/latest/zoom-crop/width/240/height/240?cb=20100815175356, May 11th, 2018

[6] https://vignette.wikia.nocookie.net/finalfantasy/images/5/5d/Tidus_Avatar_PS2.png/revision/latest?cb=20140123191940, accessed: May 11th, 2018

[7] https://vignette.wikia.nocookie.net/finalfantasy/images/b/bd/Rikku_Diving_Suit_Avatar_PS2.png/revision/latest?cb=20140123191642, accessed: May 11th, 2018

DECLARATION

I hereby declare that this paper is of original work, neither an adaptation, nor a translation of any existing paper, and not an act of plagiarism

Bandung, May 12th 2018

Aristoteles Swarna Wirahadi / 13516146