Application of Graph Theory in Designing Video Games

Rionaldi Chandraseta - 13515077
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13515077@std.stei.itb.ac.id
rionaldi.chandraseta@gmail.com

***Abstract*—Video games have been around since the 1940s, and they have changed a lot since then. Modern video games are grouped into different genres, such as Strategy, Adventure, Role-Playing, etc. Graph theory is actually applied in a lot of aspects of most, if not all, modern video games. This paper will give a perspective of how graph, including its derivation such as tree, is used by game developers in their games.**

***Keywords*—Features, Graph, Tree, Video Games**

I. Introduction

Video games started off with a humble beginning, with a seemingly simple game such as *Pong* being so popular when its first released back in 1972. Pong simulates a table-tennis game between the player and an AI (Artificial Intelligence). The game features a 2D graphic, a completely black background with two paddles, one controlled by the player and one by the AI, and a single ball. The success of Pong was followed a few years later by the birth of *Space Invader*. The Space Invader were very popular in the arcades as stated in [1], despite having a simple gameplay and interface like the Pong.

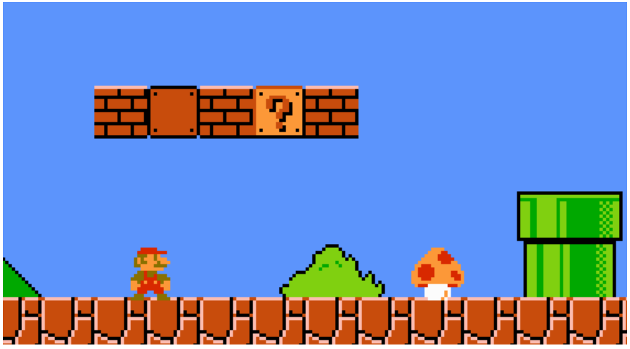

Fig. 1: The first Super Mario Bros. game screen. *Source: www.nintendonews.com*

The rise of NES (Nintendo Entertainment System) in 1983 marked an end to the popularity of arcades, and opened the market for consoles. The popularity of NES was thanks to one of the games available for it. That game is the legendary *Super Mario Bros*, a 2D side-scrolling platformer. Contradictory to Pong and Space Invader, Super Mario Bros. features a fully-colored screen shown in Figure 1. It is also one of the first games to implement graph theory in the design. Player could use some of the green pipes in the game to go to a secret area in the current "World". It requires the use of a graph to navigate both to and from the secret area.

Throughout the years, video game industry has grown massively. In 2016, almost every video games utilize the graph theory in its design. Graph could be used in so many different features of a video game. Map design, story progression, unlockable skills, and even the achievement system use a graph to be implemented in a game.

II. Theory

A. Graph

According to [2], a graph G is defined as a tuple (V, E), where:

V is a non-empty set of vertices $\{v_1, v_2, ..., v_n\}$.

E is a set of edges $\{e_1, e_2, ..., e_n\}$.

Notated as G = (V, E).

Based on the orientation of the edges, a graph could be classified into:

1. Undirected Graph

   An undirected graph's edge does not differentiate the sequence of *to-and-from* between nodes. In other words, (v_j, v_k) and (v_k, v_j) indicates the same edge.
2. Directed Graph

   A directed graph, on the other hand, does take into account the sequence of nodes. The edge of a directed graph is commonly called an *arc*. So, (v_j, v_k) and (v_k, v_j) actually represent different arcs. In arc (v_j, v_k), v_j is called the *initial vertex*, and v_k is called the *terminal vertex*.

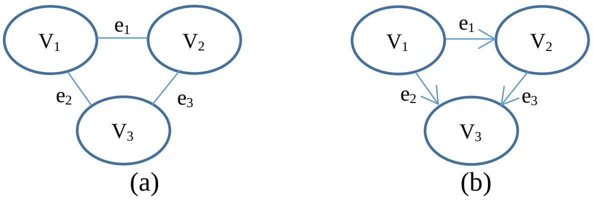

Fig. 2: (a) Undirected graph, (b) Directed graph.

A *simple graph* is a graph that contains no double-sides nor loops. The graphs in figure 2(a) and 2(b) are simple undirected graph and simple directed graph, respectively.

The following are some special types of simple graph:

1. Complete Graph

   Complete graph is a simple graph with every single node has a side connecting it to all other nodes in the graph.
2. Wheel Graph

   Wheel graph is a simple graph with each node having a degree of 2.
3. Regular Graph

   Regular graph is a simple graph with each node having the same degree.
4. Bipartite Graph

   Bipartite graph is a graph whose set of nodes V could be separated into two sub-sets V_1 and V_2, such that every side in G connects a node in V_1 to a node in V_2.

B. Tree

Reference [2] stated that a tree is an undirected, connected graph that contains no circuit. In most applications of tree, a certain node would be chosen as the *root*. The root node is basically the base of the tree. Every single node could be accessed by navigating through the root node.

An ordered tree is a special rooted tree where the order of its nodes matters, as in [2]. The use of ordered tree in computer programs is crucial to make data retrieval easier. A common way of naming the nodes in an ordered tree is by giving numbers as in books – with chapter and subchapter number. An example of ordered tree is shown in figure 3.

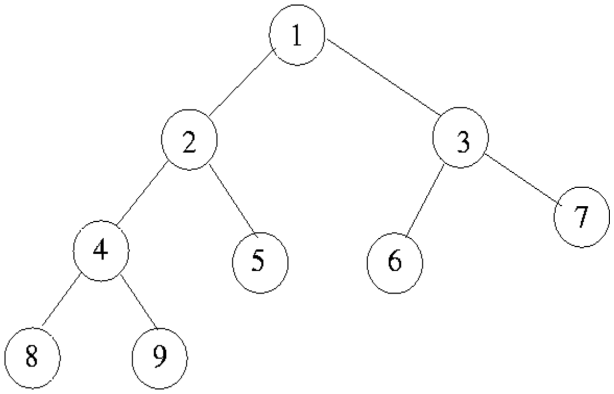

Fig. 3: An example of ordered tree. *Source: www.stackoverflow.com, Credits: Siddhant.*

Another type of tree is called the *n-ary tree* – a tree with every node having at most N children. It is called *full* or *ordered* if every node has M children, where M ≤ N. N-ary tree is widely used in parsing trees and computer file system. When N = 2, the tree would form a *binary tree*; another special type of tree; the children of which are divided into *left child* and *right child*.

III. Application of Graph in Video Game

A. Map Design

If each area of the map is treated as a node and a line is drawn between two adjacent nodes, then the map would produce a graph that could be analyzed.

The map design of any game is crucial for the user experience. Both the map design for an *open-world* game, such as *The Witcher Series*, or an *arena* game, such as *Team Fortress 2*, need to be designed well to deliver the desired experience for the user.

Open-world game would feature a single large map that the player could explore. Figure 4 shows the map for one of the Pokémon game, namely the Ruby/Sapphire/Emerald version. The map could be divided into smaller areas with, usually, different purposes per area. In Pokémon world, usually the areas are consisted of towns, caves, and forests.

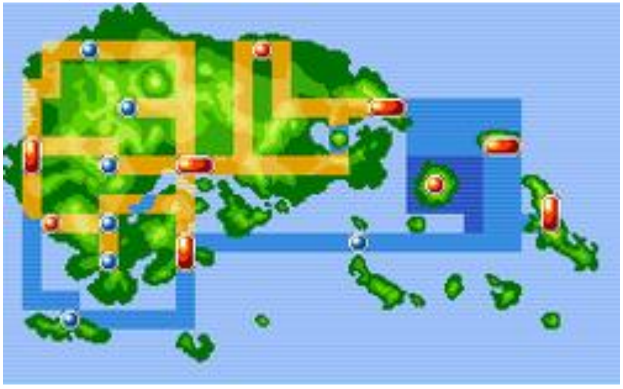

Fig. 4: Map of Hoenn region from Pokémon R/S/E. *Source: www.serebii.net*

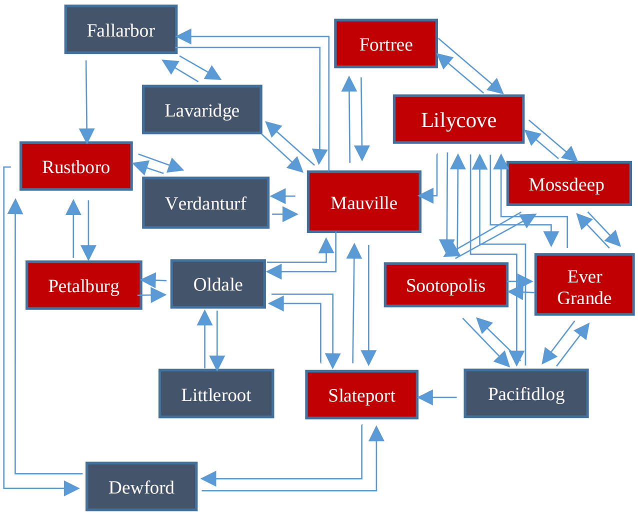

Fig. 5: Graph of Hoenn region map.

Figure 5 is a graph made from the map in figure 4. The nodes represent either cities or towns in the game, and graph is colored the same way as in the map for convenience. As you may have noticed, most of the areas in the map are connected with arcs that go both ways, but some are not. The routes from Fallarbor to Rustboro, Lilycove to Mauville, and Pacifidlog to Slateport are only one direction. This is due to obstacles in the map design that prevent player from going the other way in those areas. The route from Rustboro to Fallarbor and Mauville to Lilycove is blocked by ledges, which could only be traversed one-way, and the route from Pacifidlog to Slateport is littered with currents in the sea, which would stop the player from going the other way. Later on, in the game of Pokémon R/S/E, the player would acquire the ability to *Fly* with a suitable Pokémon. This ability simply let the player teleport from any location to previously visited city. From that point onward, the game map would practically form a *complete graph*, as the player could travel from one city to other cities that have been visited beforehand.

In an open-world game, the ideal game map would be one that is designed so that the player would always feel challenged. This could be done by placing weaker enemies in the early parts of the map and stronger enemies in the later parts of the map, such as *The Witcher 3* and *The Elder Scrolls V: Skyrim*. Some games, however, practically forbid the player from accessing parts of the map before the player progressed through the story. For example, the newer *Pokémon X* has brought back the running gag from the first-generation Pokémon where a *Snorlax* (a type of pokémon) would sleep on the map and block your way, and *Grand Theft Auto IV* implemented an algorithm where the player would get a 5-star wanted level after crossing the bridge before completing a certain part of the story. All of these are the developer's way to keep the player from straying too far from the main story.

An arena based game would need a different approach in designing the map. Most arena game are designed for multiplayer system, which means apart from an interesting map, the balance between teams would also need to be thought over. An arena map is also way smaller compared to an open-world map. That is why arena games such as *Team Fortress 2* (TF2) and *Overwatch* have multiple maps available. Each of these maps were designed in such a way that both teams would have about the same chance to win. For example, the spawn area of the attacking team would be further from the objective, but the spawn time would be faster. Another example is that attacking team has a flank route to slip past the defending team front line, but the defending team also has choke points to fend off the attacking team.

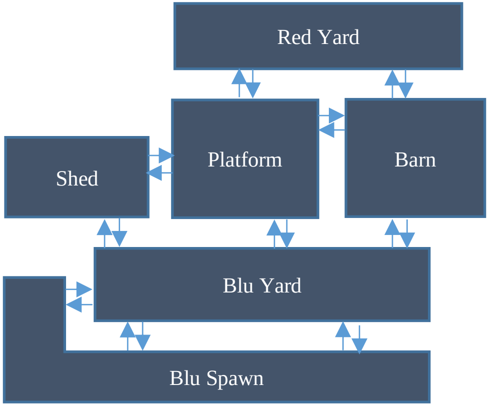

Fig. 6: Graph of first point in pl_barnblitz of TF2.

Figure 6 shows the graph of a map in the game TF2. The map features a payload that the *Blu* (attacking) team has to push all the way to the final point, while the *Red* (defending) team tries to stop the payload. The first point is located in the *Barn* in figure 6. Notice how the *Barn* could be flanked from two sides, the *Platform* and the *Red Yard*. By designing the map this way, the Red team needs to defend not only the *Barn*, but also the *Platform* to prevent flanking. Meanwhile, the Blu team has to decide which area to attack for biggest impact to Red team's defense. In the end, it all comes down to the team members. An experienced team would know where to set up defense and how to bypass it.

B. Achievement System

The reason behind the popularity of in-game achievement system is *Steam*, a digital distribution platform made by Valve Corporation. Steam provides its users with a unique page that contains information about their friend list, groups, game library, achievements, and other social features, as stated in [3].

Every game has different achievement depending on the genre of the game and, of course, the developers. The achievement system itself is very simple; it gives a notification to player if the condition(s) to unlock the achievement has been fulfilled. Once unlocked, the achievement would be displayed on the player's profile. This would motivate people to unlock all of the achievements, thus spending more time in the game. There are people known as the *achievement hunters*. Their goal would be to unlock every achievement available in the game.

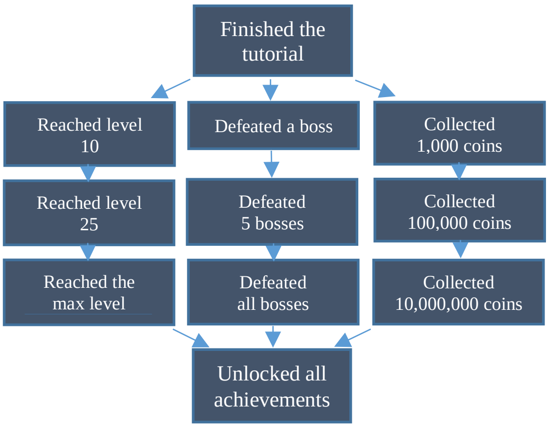

Fig. 7: An example of the achievement system.

The achievement system could actually be a tree instead of a graph, but most video games give the player an achievement right in the start and has a special achievement for unlocking every other achievement. The general idea is shown in figure 7. A tree could not be directed nor contain any circuit, hence the achievement system is a graph.

C. Artificial Intelligence

Every video game that has NPC (Non-playable characters), or an option to play against CPU, also has something called *artificial intelligence* (AI). According to [4], the term AI is used when the machine could mimic the human brain in processes such as learning and problem solving. The implementation of AI heavily uses the graph theory, because it would be programmed in *states*. The states, transitions, and effects depend solely on the programmers of the game.

Let us take a look at a game called *The Sims* by *Electronic Arts*. This particular game is a life simulation game, and one of the most popular in the genre. In *The Sims*, player could build houses, create *Sims* – a digital character, and actually control the created sim(s). While controlling a sim, the game would show the basic needs such as hunger, hygiene, bladder, and so on. The player needs to keep the basic needs fulfilled to make the sim happy. For example, eating to keep the hunger bar up, showering to raise the hygiene, etc. There is no real stated goal for this game, so the player could make their own goal, just like real life.

What makes the game interesting is the option to turn on the AI. When turned on, the members of the active household would act on their own. They could fulfill their own needs without needing any input from the player. This AI actually uses a graph to map the states they are in, and decides what to do based on the needs of individual sim and its personality.

The Sims has developed throughout the years and has also improved the AI used in the game. *The Sims 2* introduced a new “Wants and Fears” mechanic; the AI would try to do the acts listed under “Wants” and also avoid the acts listed under “Fears”. The next installment of the game called *The Sims 3* brings a new attribute called *Traits* in sim creation. Moreover, the game also adds a “Moodlet” feature. The game would process current moodlets of the sim to determine its mood. The AI would take into account each sim’s mood as well as traits in determining its actions.

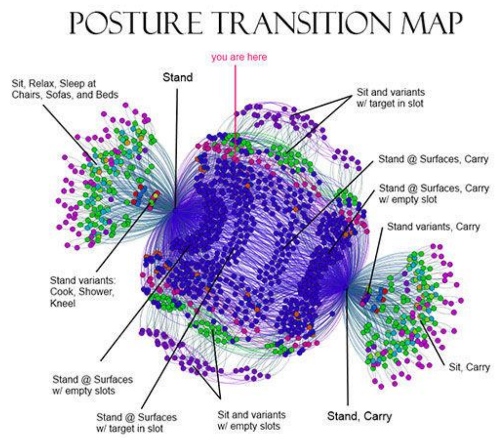
POSTURE TRANSITION MAP
you are here
Sit, Relax, Sleep at Chairs, Sofas, and Beds
Stand
Sit and variants w/ target in slot
Stand @ Surfaces, Carry
Stand @ Surfaces, Carry w/ empty slot
Stand variants, Carry
Stand variants: Cook, Shower, Kneel
Sit, Carry
Stand @ Surfaces w/ empty slots
Stand @ Surfaces w/ target in slot
Sit and variants w/ empty slots
Stand, Carry

Fig. 8: The Sims 4 Posture Transition Map. *Source: www.popularmechanics.com*

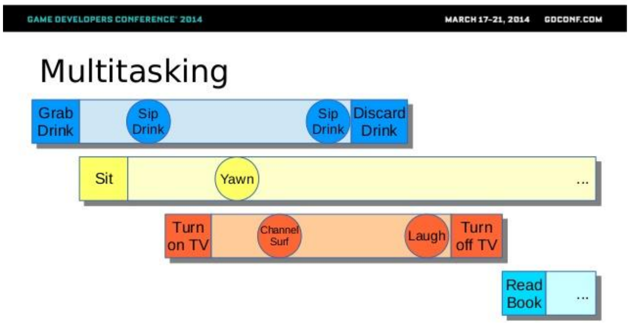

Fig. 9: The Sims 4 Multitasking mechanic representation. *Source: www.gdcvault.com*

The most recent installment, *The Sims 4*, comes the closest in replicating real life people. Figure 8 shows the posture transition map of a sim in the game, including standing, sitting, standing while carrying object, and so on. To add to that, a new mechanic is yet again added to the game. As stated in [5], the mechanic is called *concurrent interaction*. It basically lets sim do multitasking, just like humans usually do, represented in figure 9. This mechanic still uses the concept of states and determining what actions to take based on the current state.

IV. Application of Tree in Video Games

A. Story Branches

Some video games are heavily story-driven, where the storyline changes depending on player’s previous choice. In fact, a game company called *Telltale Games*, founded in 2004, specialized in making a story-telling games. They have made numerous game adaptations from comic books (e.g. *The Walking Dead*), TV series (e.g. *Game of Thrones*), movie series (e.g. *Back to the Future*), and many more. The fact that the company is still going strong by the time of writing shows that this type of game is quite a success.

Another company called *Quantic Dream* also uses the basic principle of branching storylines. One of the famous game made by *Quantic Dream* is called *Beyond: Two Souls*, and it has 24 different endings. The game also stood out because it uses real actors (*Willem Dafoe* and *Ellen Page*) and motion capture in the game scenes. The result is an interactive game with lifelike characters, with sales reaching over one million copies just two months after the game is released.

Fig. 10: Butterfly Effect in Until Dawn. *Source: inthedarkair.wordpress.com*

Nevertheless, one of the best games to feature multiple storyline are not made by *Telltale Games* or *Quantic Dream*. Developed by *Supermassive Games*, *Until Dawn* has succeeded in pushing the tree of story branches to the next level. The game was developed with the "Butterfly Effect" as its core. The Butterfly Effect states that any player decision could cause unpredictably large effect on later part of the game. Moreover, the amount of choices is unbelievable, reference [6] points out that there are hundreds of possible endings, and thousands of available storylines in the game.

The path of available storylines is represented in figure 10. It is a picture of a butterfly, with numerous lines starting from the center of the body and branching all the way to the tip of the wing. There are 8 characters in the game, with the possibility of each and every one of them dying through the player's choices. That alone gives a staggering $2^8 = 256$ total ending combinations. Not to mention the huge amount of choices that could be made throughout the game; this includes everything from deciding what to say until deciding what to do.

One of the advantages of these games with multiple storyline and endings is the *replay value*. Most video games feature only a single storyline and a single ending, which means once finished, the story would never change no matter how many times the game is played. However, in games featuring story branches, each playthrough could bring new, previously-unknown plot and ending for the player. That way, people could spend more time playing the same game without feeling bored because they knew what would come. This does not imply that people do not replay a game without branching story though. Some games, such as *The Last of Us*, have really good storyline that people would play it over and over again, even though the plot is always the same.

It is not uncommon that in-game achievement requires the player to reach all possible final endings. Achievement hunter, as explained in Achievement System, tend to save the progress just before making a choice that would change the ending of the game. This is actually the same as finding the *least common ancestor node* of the two endings. By saving the game right at the least common ancestor node, the player could skip all the previous storyline and continue playing the previously unvisited nodes.

B. Skill Tree

Some video games, especially RPG (Role-Playing Games), focused on character development throughout the game. One of the most famous feature in an RPG is the *skill tree*. A skill tree is a tree containing available skills that could be unlocked by the player using a certain mechanic. Popular RPGs, such as *The Witcher Series*, *The Elder Scrolls Series*, and *Final Fantasy Series*, implemented a skill tree system in the game. A game does not necessarily have only one skill tree, in fact most of modern games have multiple skill trees for one character. Figure 11 shows an example of the skill trees in *The Elder Scrolls V: Skyrim*.

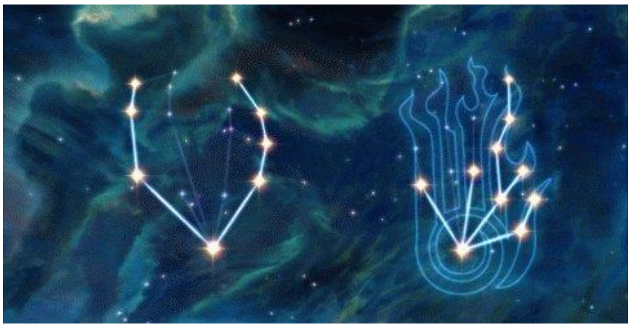

Fig. 11: Some of the skill trees in The Elder Scrolls V: Skyrim. *Source: www.carlsguides.com*

A skill tree could be a *unary tree*, a type of N-ary tree with N=1, or a regular N-ary tree, where N > 1. A unary skill tree would usually be already categorized into different specialization, such as in *The Witcher 3*. Meanwhile, a regular N-ary tree usually starts off with general skill and give the player a choice to focus on specific things down the line, just like the one in *Final Fantasy*. Usually, in both trees, the further the node is from the root, the more powerful the skill is.

Every game also has different ways of unlocking a skill in the skill tree. Most of the time, a game would use a *skill point* system, where the player could unlock the skill when the required amount of skill point is fulfilled. Another way is by restricting the available skills by player level, so that the skills could only be unlocked after certain level. Sometimes, a game would implement both of them, making a skill unavailable before reaching a determined level and also needing the player to have enough skill points to be able to unlock it. However way it is implemented, there is one requirement that does not change. A skill could not be unlocked if the preceding skill(s) are not yet unlocked. It is called a skill tree after all.

V. Conclusion

Most modern video games used both graph and tree in many features inside the game. Understanding the concept of graph theory is extremely important for people who want to pursue a career in video game industry.

VII. Acknowledgment

The author would like to thank Dr. Ir. Rinaldi Munir, M.T. for the comprehension of discrete mathematics through all the lectures in this semester. The author would like to say thank you to all the family members and friends for their support throughout the making of this paper.

References

[1] S. L. Kent, *The Ultimate History of Video Games: From Pong to Pokémon*, Three Rivers Press, 2001.

[2] R. Munir, *Diktat Kuliah IF2120 Matematika Diskrit*. Institut Teknologi Bandung, 2006, ch. 8-9.

[3] W. Greenwald, *Steam Review & Rating*, *PC Magazine*, *http://www.pcmag.com/article2/0,2817,2392198,00.asp*, accessed 3 December 2016.

[4] S. J. Russell, P. Norvig, *Artificial Intelligence: A Modern Approach (3rd ed.)*, Upper Saddle River, 2009.

[5] P. Ingebretson, *Concurrent Interactions in The Sims*, *GDC Vault*, *http://www.gdcvault.com/play/1020190/Concurrent-Interactions-in-The-Sims*, accessed 4 December 2016.

[6] D. North., *Until Dawn has hundreds of endings and thousands of branches*, *Destructoid*, *https://www.destructoid.com/until-dawn-has-hundreds-of-endings-and-thousands-of-branches-279599.phtml*, accessed 4 December 2016.

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, 4 December 2016

Rionaldi Chandraseta - 13515077