Victory Ratio Specifications

Overview

The purpose of this project is to create an educational game to augment student's curriculum studying fractions, ratios, and probability in school. The goal of any game is first to be fun, and so we will seek to create an intuitive, engaging, and satisfying learning environment that will give the players an experience to draw from while moving forward in their curriculum.
It will be set in medieval times with control of set units over a grid. Core objective is to take all enemy units.

Intended Use-Case

This should be a strategy game that is simple enough for children around the age of ten, but engaging enough for adults. The use-case of any game is fun, but as the origins of this idea is based in the value of transferability of skills from playing games, it should have benefits for the player much like a player who engages in any sort of competitive strategy game, with transferable benefits in mathematics by being mindful of the simple calculations that the game operates on, advanced reasoning skills that come with anticipating the move of an opponent (much like chess), and the social aspect of engaging in friendly competition with peers.

Mechanical Design

The design of the combat will be a mixture of games such as Fire Emblem, Advanced Wars, and Civilization. A weapons triangle will give units a 50% strength bonus over the unit they have advantage against (eg. Swords are effective against axes, axes against spears, spears against swords). A unit's strength is represented by the number of soldiers in that unit. Terrain can also present advantages (soldiers in the woods have a 25% strength increase against soldiers in an adjacent plane while being attacked).
Armies will start on opposite sides of an X by Y grid. Each unit will have a maximum allowed movement, and will be able to attack units in adjacent cells. If archers will be able to attack an extra cell away, but have disadvantage if the enemy is in the adjacent cell.

When units are clicked, details such as unit type, remaining strength, and possible movement will be presented to the player. The player will also have the option to split groups of units or combine groups of the same type into stacks of x (current plan is 9, but if I settle on a pleasant way to render greater numbers than that, it can and maybe should be done).
Pending on if the player is going against a guided “story” mode or a physical player 2, turn order may be changed. This is addressed in “Curriculum/Story Design”.
To enhance educational benefits of the game, I am proposing that the player have an option to predict the odds of victory, if correct, the player’s units “gain confidence” in the player’s ability and get a slight bonus to either damage dealt or chance of victory.
Combat flow is currently TBD. Originally combat was calculated as each unit attacking had a chance of their strength vs the strength of one enemy unit (so if equal, 50/50) of destroying a unit. This calculation was done for each attacking unit. Afterwards, defending units would strike back with the same calculation for the number of units they have remaining. While a fine way mechanically to do combat, it may not be the most useful for an educational game. So I need to review some middle-highschool aged math to determine a better way to calculate this, probably separating victory calculations from damage calculations. It may be good to research how the Civilization series calculates combat, as it presents the player with their percent chance of winning when engaging enemy units, which is separate from how damage is calculated. Ideally, I want to have all data presented to the player to get the percentage that Civilization gives the player, without giving them that answer.

Curriculum/Story Design

The story mode of this game will be a 4 or 5 map series of missions where you play as a young strategist being advised by his mentor on how to repel an invading kingdom. Originally I was going to put the tutorial as separate from the story mode, but it may make for better ability to scaffold the learning objectives to have mechanics introduced split across the missions.
The game should take anywhere between 30 and 60 minutes to complete, as it should be done in a single classroom sitting. A “hotseat” multiplayer mode may also be added for students who may wish to continue to play either because they have finished or because they desire the competition.
The original prototype was a 2 player game, in which players take turns moving units until all units are moved and the round resets. This was done to make it feel more fair in a small map with 2 players. For the “story” mode at least I would like to have it be that the player moves all their units then the enemy moves all of their units, and so forth until one side has lost all units.

Code/Asset Design

Victory Ratio shall be made in Unity using it's 2D framework. While the original project was done in Python, the tools available in Unity will make it easy to reproduce up to that point in a short period of time, and has resources such as particle systems, smart tiles for map creation, and a massive community to offer support, meaning many of the harder programming questions for this project are more or less already solved.
Art assets will be created in Gimp, as it is familiar, free, and a popular sprite art tool. Sprite art will be lower detailed, but inspired by games such as Advance Wars and Fire Emblem.

Advance Wars battle animation from Amazon.com

Fire Emblem from Wikipedia.org

Goals

Minimal Goal

- Single Player
- 30-60 minutes of gameplay
- 5 Unit types (sword, spear, axe, archer, mounted)
- 4 Terrain Types (plane, forest, water, castle wall)
- 4-5 maps with predetermined enemy types and placements
- 3 or 4 lightly animated characters with dialogue.
- Ability to stack and split units
- AI opponents with A* pathfinding

Expanded Goals

- Procedurally generated maps for multiplayer
- 3 Different AI types or difficulties
- Online Multiplayer (ain't happenin')

Technical Feasibility

Unity takes a lot of guesswork out of this, as it has systems in place to build to WebGL as well as Windows deployments. It has more, but likely those will not be utilized. Ideally, it will be deployed on my own webspace, but Itch.io is a functional fallback, as it hosts both as download and in browser for free and I have used this in the past for the Ludum Dare game jam. In the past, people have ran into problems with WebGL deployments, but it sounds like these mostly deal with browser RAM allocation, and this should not be resource intensive enough to worry much about it, but I will still want to have early builds up and tested just so we can see if it runs into problems.

Early on, I will want to have a build similar to the summer project done, and from there be testing and figuring out how to augment the smart tile system to incorporate terrain effects. This is something that people definitely do, but I have not done, and it does require some tweaking to get it to work.

The next thing I will want implemented is A* for showing valid movements and controlling enemy AI, from there I can start building decision trees for AIs.
Once these are working, Unity's toolset will make it fairly easy to build the maps and design encounters, and I can focus my energy on creating a pleasant UI experience.

Flow/Gameplay Loop

This was the loop created for the original 2 player game created as a summer project.

1. Start Software
2. Enter Player 1 Name
3. Enter Player 2 Name
4. Players Take turns placing allotted units on their side of the field
5. Begin Match/Core game loop
    a. Begin round
        i. Each player takes turns moving a single group (stack of units). A group can attack or be moved once per round. The player can attack first then move, or move then attack. Choosing to hold position counts as a movement.
        ii. Groups can move past ally or enemy units, but not through enemy units. (likely have the ability to only move 3 or 4 spaces a move, as the map is small).
        iii. Current idea for fighting is attacking units get to hit first, so the player has incentive to be the first to strike. After taking damage, what remains of an enemy group attacks. Each unit individually can take out one enemy unit, based on their relative strength, so a unit with advantage has a higher chance of killing a unit. If the units are the same, they have a 50/50 chance of killing a unit. A smaller group of units can kill at most the number of units they have in their group.
    b. Repeat until all units have been moved or instructed to hold position
    c. Continue to the next round until all of one player's units have been defeated.
6. Victor Declared

For a single player implementation it would be as follows.

1. Start Software

2. Enter Player 1 Name
3. Players Places units on their starting area of the map.
4. Begin Match/Core game loop
    a. Begin round
        i. Player begins moving each unit. A group can attack or be moved once per round. The player can attack first then move, or move then attack. Choosing to hold position counts as a movement.
        ii. At start of movement, a player can split a group into multiple units. Only the units that were moved count as having been moved.
        iii. The moved unit can then be added to another group of the same type of unit if it is within range, but if that group has not been moved, it's movement will be counted as spent.
        iv. Groups can move past ally or enemy units, but not through enemy units. Range of movement will depend on the finalized size of the maps. Original product only allowed the movement of 3 spaces per move.
        v. Original implementation for fighting is attacking units get to hit first, so the player has incentive to be the first to strike. After taking damage, what remains of an enemy group attacks. Each unit individually can take out one enemy unit, based on their relative strength, so a unit with advantage has a higher chance of killing a unit. If the units are the same, they have a 50/50 chance of killing a unit. A smaller group of units can kill at most the number of units they have in their group. This is subject to change after more research.
    b. Repeat until all units have been moved or instructed to hold position
    c. Continue to the next round until all of one player's units have been defeated.
5. Victor Declared.
6. Next level.

This continues until story mode is completed.