

# Pokémon Go Raid Battle Simulator

CS400 x4 Project Proposal

By Nicholas Beninato

## Problem

Pokémon GO is a mobile game that was released in the summer of 2016. Although the hype has died down since launch, there still is a very large and active community playing it. One feature of the game is the ability for you to fight a very powerful Pokémon (called a raid boss) in a battle called a raid. The Pokémon combat system involves relationships between 18 different "types", each of which is strong or weak against other types. Every individual Pokémon can have 1 or 2 base types, and 2 or 3 attacks, each of which has its own type. For example, a Fire type Pokémon is weak to Water type attacks. Because there is so much complexity involved in choosing which Pokémon, it can be difficult for players to know which Pokémon to use to do the highest possible amount of damage.

## Primary Stakeholder

The primary purpose of this program is to assist Pokémon GO players. It will help players find optimal counters against a raid boss. By doing more damage, a player benefits from more rewards at the end of a raid, and defeating a raid boss in a shorter amount of time, which can be useful when you only have a few seconds of spare time at the end. There are multiple different ways to assist players, and each method will require different amounts of data. For example, here are some ways to help a player, ordered from least data required to most data required. The only user input required is the name of the raid boss.

- **Example 1:** Listing the types that do the most damage against a raid boss
  - Stored data: see Example 1 in the Data section
- **Example 2:** Listing the Pokémon that theoretically do the most damage against a raid boss
  - Stored data: see Example 2 in the Data section
- **Example 3:** Listing the Pokémon a player has that would do the most damage against a raid boss
  - Stored data: see Example 3 in the Data section

Each of these methods will benefit a player in a different way, but they all serve the overall goal of increasing the damage per second (DPS) of a player in the raid in order to reduce the overall time spent battling and increase the rewards for damage dealt after a raid.

## Graphical User Interface (GUI)

Input a Pokémon

Name

Bulbasaur ✓

VINE\_WHIP SLUDGE\_BOMB SEED\_BOMB

Level

29

Attack IV

15

Defense IV

10

Stamina IV

12

Confirm

In this input menu, a user will input their Pokémon. Once a Pokémon name is determined to be valid, a list of moves for fast, charged 1, and charged 2 will be accessible through a drop down menu. The user can enter the IVs and Level of their Pokémon, then hit confirm to write to a JSON file like the one seen in Example 3 of the Data section




Selected Raid Boss:

Mewtwo ✓

Mewtwo is Psychic Type, which is weak to Bug, Dark, and Ghost Types

What would you like to run in the simulation? (All Pokemon or Your Pokemon)

All Pokemon

|                                                                                     |                                                                                     |                                                                                     |
|-------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|
| 1: Chandelure                                                                       | 2: Darkrai                                                                          | 3: Giratina                                                                         |
|  |  |  |

In this mock up of the GUI, the user selects a Raid Boss, and once it is verified to be true, the program will display which types are super effective against it. The user can then select to run a simulation to find the Pokémon with the highest DPS using their Pokémon or All Pokémon.

## Data

Depending on what the user requires, different types of data will need to be used. For each of the examples mentioned in the Primary Stakeholder section, here is a sample of the data needed, in JSON format. A ". . ." means the data continues in the same pattern.

- Example 1

- A dictionary with keys being a Pokémon's name and values being a list of the key's type(s)

```
{
    "Bulbasaur":["Grass", "Poison"],
    "Charmander":["Fire"],
    "Squirtle":["Water"],
    ...
}
```

- A dictionary with keys being a type and values being a dictionary with keys being a type and values being how effective the first key is against the second key

```
{
    "Grass":{
        "Grass":0.625,
        "Fire":0.625,
        "Water":1.6,
        ...
    },
    "Bug":{
        "Grass":1.6,
        "Fire":0.625;
        "Water":1,
        ...
    },
    ...
}
```

- Example 2

- A dictionary with keys being a Pokémon's name and values being a dictionary of the key's base stats, type(s), fast moves, and charged moves.

```
{
  "Bulbasaur": {
    "type": ["Grass", "Poison"]
    "attack": 118,
    "defense": 111,
    "stamina": 128,
    "fast": ["VINE_WHIP", "TACKLE"],
    "charged": ["SLUDGE_BOMB", "SEED_BOMB",
"POWER_WHIP"],
  },
  ...
}
```

- A dictionary with keys being a move's name and values being a dictionary of the energy cost, cooldown, damage, and the type of the key.

```
{
  "ROCK_SLIDE": {
    "damage": 80,
    "type": "Rock",
    "energyCost": -50,
    "cooldown": 2700
  },
  ...
}
```

- Example 3
  - The user needs to input the stats for each Pokémon they wish to use in the simulator. So this version would require the same data as Example 2, and also an array of timestamps for each time the user inputs a Pokémon, and a new dictionary with keys being a timestamp for when a Pokémon is entered by the user and

```
[1572667461.198138, 1572667489.736955, 1572667490.483094, ...]
```

```
{
  1572667461.198138: { "name": "Bulbasaur",
                      "level": 29,
                      "attackIV": 15,
                      "defenseIV": 10,
                      "staminaIV": 12,
                      "fast": "VINE_WHIP",
                      "charged": ["SLUDGE_BOMB"]
                    },
  ...
}
```

Overall, I think these JSON objects are all the data that I would need to read into my program to start helping Pokémon GO players find optimal raid counters. I think the most common object will be a JSON object, and I don't see myself having to use any of the ADTs that we went over in class, such as a HashTable or BST. I might use a BST if I want to display every Pokémon sorted from best to worst, which I could do by adding the results in terms of DPS against a raid boss as the key for a BST, with the value being the Pokémon's name.