Noah Lichtenstein, Nicholas Seidl Monday, November 5, 2018

Final Project Proposal: Applying Search, CSP, MDP to Pokémon Go Gym Battles

Pokémon Go is a mobile game played by millions of people. One of the core features is a "gym battle." A gym consists of defending Pokémon. A player attacks a gym by selecting some of their Pokémon to attack. The winner of the battle is whoever has the last Pokémon standing.

This informal problem statement gives rise to the question: what is the optimal choice of attacking Pokémon to defeat the defenders? There are over 300 uniquely composed Pokémon, so it is non-trivial to compute the ideal combination of attackers, given the defenders. We propose to formulate this problem in three different ways: as a search problem, as a constraint satisfaction problem, and as a reinforcement learning markov decision problem. We hope to show that this problem is solvable with varying degrees of success, dependent on the algorithm.

We will use a common base formulation for all algorithms. A "Pokémon" has a name, base statistics (stamina, attack, defence, from 1 to 600), individual values (IVs: stamina, attack, defence, from 0 to 15), combat power (CP: from 1 to 5000), level (from 1 to 40)[1], type(s) (one or two), and moves (one fast and one charge)[2]. All Pokémon share the same level (30) and the same IVs (15, 15, 15). All "Bulbasaur" will share the same base statistics, CP, type(s), and moves.

A state in the state space is the combination of an attacking set A of six Pokémon a_i and a defending set D of six Pokémon d_j. The order of Pokémon within a set matter: same Pokémon used in a different order warrants a different state. The defending set D is set constant. The attacking set A may consist of multiple "Bulbasaur". A transition between a state is the change of a single a_i.

States will be evaluated by simulating a gym battle. The winner of a gym battle is whomever has the last Pokémon standing. Fights are played sequentially, starting with the fight a_1 against d_1. After each fight, one Pokémon dies, and one Pokémon lives. The winning Pokémon plays the (chronologically) next opponent after being healed to maximum health. For example, if a_1 beats d_1, the gym battle fight sequence may begin $a_1>d_1$, $a_1>2_2$, $a_1<d_3$, $a_2>d_3$, etc. Each state is guaranteed to have a winner because each team has a finite number of Pokémon, and each fight has a winner or loser. The evaluation function for a 1 vs. 1 Pokémon fight is well known[3].

Formulating this problem as a search problem will provide a basic proof of concept, by showing that our state space and transition function are appropriate. The search problem formulation should provide a simple baseline for comparison against our other algorithms. In this formulation, each state is a goal (attacker wins) or non-goal (attacker loses) state. The algorithm produces the first goal state found. The outcome of this algorithm is expected to be valid but non-optimal, as it lacks the expressiveness to judge states based on their features.

We will then implement the problem as a CSP. This formulation will contain extra constraints on what a goal state is beyond the winner. Proposed constraints include ensuring no individual matchups are unfavorable (due to typeset), and that no defensive oriented Pokémon are selected as attackers. We expect this formulation to provide a more optimal answer, because it is able to discern between states with more specificity. The tighter we constrain the problem using knowledge of the game mechanics, the more optimal we expect our solutions to be.

Finally, we will implement a reinforcement learning MDP, because it quantifies how winning a state is. In this formulation, we will determine the reward of each state based on a linear combination of its features. The primary feature is if the attacker is the winner. Secondary features may include the number of Pokémon alive for each team, the ideality of the matchups for each fight, and the efficacy of each selected attacking Pokémon. Using an online learning algorithm like Q-learning will allow us to episodically train our algorithm on varying gym matchups. After enough time, we expect the algorithm to reach a near-optimal solution.

Many of the pieces for this assignment (battle outcome simulation, state representation, datasets) are shared between all three algorithms, and as such will be completed jointly. Since we believe the search problem is relatively simple, that algorithm will be implemented jointly as well. Nick will research and implement the CSP algorithm, using backtracking and forward checking. Noah will research and implement the MDP algorithm, using Q-learning and reinforcement learning.

In our research for this project, we have discovered several works that explore the application AI to Pokémon Go, as well as computational methods to determine the outcome of a Pokémon battle[4]. Current tools exist which are able to determine the outcome of a single Pokémon battle. Other tools exist to evaluate other game features, like the optimal Pokémon to select for "raid" encounters[5]. However, no work seems to have been conducted in order to extend existing ideas to a full six on six battle. Further, AI has been applied in other areas of the game, such as using IBM's Watson to play Pokémon Go[6], although its application is not relevant to this paper.

In conclusion, this proposal outlines three ways to solve the "gym battle" problem in Pokémon Go. We believe this problem is an appropriate example of how AI algorithms of can be applied to the real world. In the case of Pokémon Go, we hope create the real world benefit of providing members of the Pokémon Go community with a way to optimize their gym battles.

Sources:

1. Pokémon Go Gamepress Stat Data, https://pokemongo.gamepress.gg/pokemon-list
2. Poké Assistant Moveset Data, https://pokeassistant.com/movesets
3. Pokémon Go Gamepress Damage Mechanics, https://pokemongo.gamepress.gg/damage-mechanics
4. Pokebattler Gym Battle Simulator, https://www.pokebattler.com/fights
5. Pokebattler Raid Counter, https://www.pokebattler.com/raids
6. World's Most Powerful AI Used to Catch Pokémon, https://www.newsweek.com/worlds-most-powerful-ai-watson-ibm-used-catch-pokemon-489063