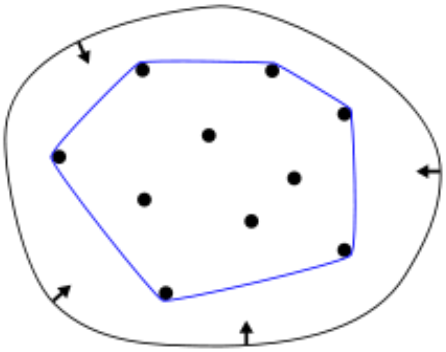
Name: Mohammad Ibrahim Abu-Amara
ID: 20180358

WildStella

A General Approach To Conquer Dynamic Real Time Strategic Situations

Idea and Thought Process



The idea first came up from trying to solve the wild fires problem, which is basically finding a way to prevent them, as well as trying to fight it through a sub-optimal way. It first looked like an algorithmic problem, where finding a simple convex hull around the fire would get the job done and put that fire only inside the convex hull and cut it off.

However, that is not that case in many wildfires as they can get out of hand very quickly, in addition to the fact that a convex hull will damage everything inside it; we're only fighting the fire on the side of the convex hull.

So, finding a strategic plan to help firefighters navigate through the wildfires, while maintaining their own health, the health of others, and their assets such as water capacity, equipment, helicopters and other vehicles.
This can be thought of as a game, where there are two teams, the wildfire team, which consist of the fire itself, the assets that the fires burn (trees, houses, grass, …, etc), and the firefighters teams, which consists of the firefighters themselves, their equipment, and any assets they rescue from the other team. So, basically this is a strategic game of who can control more assets first, furthermore, who can survive in this.

Moreover, this game can be implemented in any two or more team's games, where the goal is to handle or find a strategic way to protect/use some asset(s) in real-time, and many real-life situations can be abstracted to match this criteria.

Examples:

- The electric grid problem, where we need to optimize the energy production and conception dynamically and in real-time, where the players are the producers and consumers, and the asset is the electricity itself.
- Managing a set of bargaining business assets.
- Other natural disasters where the human factor can help.

and many more, this is where WildStella got its name, WildStella is latin for WildStar = Wild*.
It is a strategic planning algorithm to handle different general situations that learns on its own with zero data, and through continuous development while playing the game ;D.

Problem Statement

Given a set of players denoted by P, a set of assets denoted by A, a surjective function that maps each player to a team denoted by PT, and a injective function that maps each team to an asset to a team denoted by AT.
Example:
$P = \{1, 2, 3, 4, 5\}$
$A = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$
$PT = \{(1, A), (2, A), (3, B), (4, C), (5, A)\}$
$AT = \{(A, \{1, 3, 5, 7, 8, 9\}), (B, \{2, 6\}), (C, \{4\})\}$

find an optimal or suboptimal dynamic real time strategy for a given team to at least control it's desired assets.

Solution Proposed

Use Artificial Intelligence (Reinforcement Learning) and game reasoning (Game Theory) in order to let virtual agents play the game, and play against each other in order to achieve better strategies.

Deliverables And Success Criteria

1. At least human like intelligence strategies (game reasoning)
2. Real time strategies and continuous learning (Adaptivity)
3. Create more efficient strategies than human like intelligence (Creativity)

Plan/Approach Followed

The steps to achieve the success criteria are in this order

1. Create a virtual environment that can parse the agents (the players), the assets, the teams, and the goal(s). This environment should be treated purely as a game, with an appropriate reward system that can fairly distribute different player behaviours (not always a uniform distribution), in addition to that we need it to easy to compute and check against certain benchmarks that tells certain states of the game e.g: goal achieved, goal is impossible to achieve, player x is moving through their path to a nash equilibrium or a stable state.
2. Create agents that can play the game with no prior experience (ZERO DATA), but it can use some properties to be able to compute some game reasoning properties. Also, they should be able to work in parallel and also learn from other agents to improve their learning rate faster, more accurately, and most importantly more efficiently. The agent should also have an action space and a set of "sensors" that can "see" a reward in an action.