COMPSCI 6TB3 Project Plan

Brooks MacLachlan

March 13, 2019

1 Project Description

The goal of this project is to develop a grammar and DSL for describing game structures. For the purposes of this project, a "game" is simply an activity involving one or more players with some concept of "winning".

Variabilities between games include the number of rounds, number of players, player affiliations (are there teams, or is everyone playing individually?), the win condition, and game "events" that dictate how the game progresses. Some examples of potential events are updates to a player's affiliation or score, and events could be triggered by results of player performance-based challenges or player decisions. The language should be expressive enough to describe all of the following examples of game structures:

- Tennis: a game consisting of 7 rounds between 2 players where the winner is the player who wins the majority of rounds.
- Baseball: a game consisting of 9 rounds between 2 teams where the winning team is the one with the greatest cumulative score across all 9 rounds.
- Survivor (based on the television show with the same name): a game with many players consisting of a series of rounds where a player is eliminated in each round by majority vote and the winner is the last player standing (many different variations of this formula would be possible and expressible by the grammar)
- Many more

Note that the language is describing only the game structure, not the details of how the game works. For example, the mechanisms behind performance-based challenges are not of concern here. What matters is how the results of those challenge impact how the game progresses. The details of *how* a player wins a round of tennis do not matter, but the result of winning a round of tennis is that the player moves closer to winning the entire game, and that is the kind of structure that will be defined by the language.

2 Resources

The compiler for the DSL will be written in Haskell. Tests will be written with the assistance of HSpec, a testing framework for Haskell. Haddock will be used to generate documentation for the software. The DSL will be compiled into Python 3 code. Various television programs will be used as inspiration for game structures to use as examples to assist in development of the language. These include Survivor, Big Brother, The Genius, and sports.

3 Division of Work

Brooks MacLachlan is solely responsible for completing the project.

4 Weekly Schedule

Week	Tasks to complete
March 18-24	Fully define the grammar and write examples
March 25-31	Write Python programs for some of the examples, to get an idea of what the final, compiled product should look like. Develop the compiler in Haskell, writing tests and documentation along the way.
April 1-7	Design and complete poster
April 8-15	Final touches and print poster
April 16-17	Poster presentation