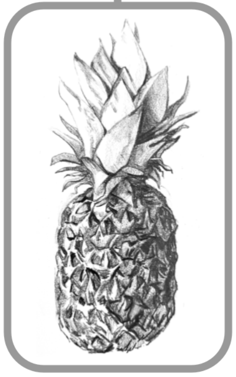
Part 1



4

Managing a Rugby match

Concept. The idea here is

4.1 Introduction

Since the basic game engine will run using the stochadex sampler, the novelties in this project are all in the design of the rugby match model itself. And, in this instance, we're not especially keen on spending a lot of time doing detailed data analysis to come up with the most realistic values for the parameters that are dreamed up here. Even though this would also be interesting.

One could do this data analysis, for instance, by scraping player-level performance data from one of the excellent websites that collect live commentary data such as rugbypass.com or espn.co.uk/rugby.

This game is primarily a way of testing out the interface of the stochadex for other users to build projects with. This should help to both iron out some of the kinks in the design, as well as prioritise adding some more convenience methods for event-based modelling into its code base.

4.2 Designing the event simulation engine

We need to begin by specifying an appropriate event space to live in when simulating a rugby match. It is important at this level that events are defined in quite broadly applicable terms, as it will define the state space available to our stochastic sampler and hence the simulated game will never be allowed to exist outside of it. So, in order to capture the fully detailed range of events that are possible in a real-world match, we will need to be a little imaginative in how we define certain gameplay elements when we move through the space.

The diagrams below sum up what should hopefully work as a decent initial approximation while providing a little context with specific examples of play action.

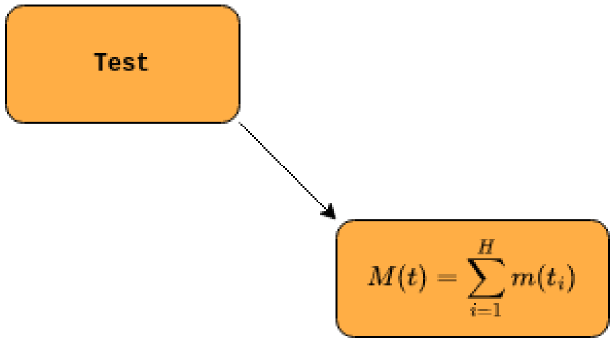


Figure 4.1: Simplified event graph of a rugby union match - replace with drawio.

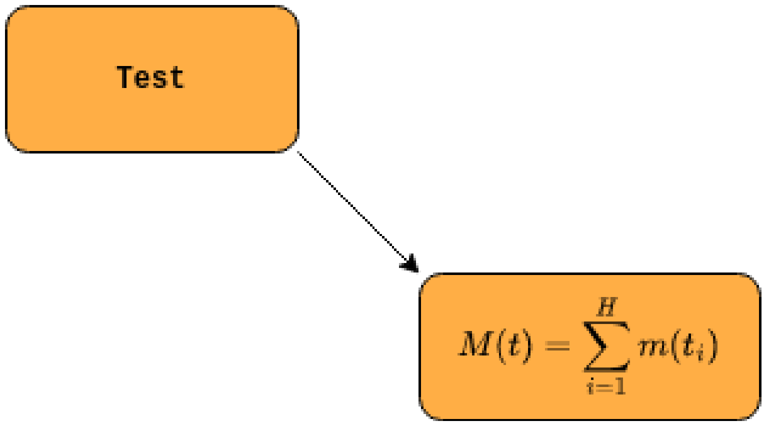


Figure 4.2: Optional model ideas - replace with drawio.

4.3 Linking to player attributes

4.4 Deciding on gameplay actions

4.5 Writing the game itself

Bibliography