Musical Games: Exploring Improvisational Composition Through Game Theory

Caroline Marcks, Andrew Mendelsohn, Jayme Woogerd

1 Introduction

In *A Model of Performance, Interaction, and Improvisation* [2], Paul Hudak outlines a formal model of music performance and improvisation based on the idea of *mutually recursive processes*.

This model lends itself to an form of algorithmic composition using game theory. In this model, we treat engaged processes as players in a game. The payoff for each player is quantified by a player's notion of aesthetically pleasing music. The rules of the game specify the allowable moves of a player given all possible game situations. Finally, a player's strategy is a (non-deterministic) algorithm for playing the game.

We plan to implement a simple, two-player musical game using the embedded language Hagl, a domain-specific language for defining and exploring game theoretic experiments. We will use infrastructure provided by the Haskore module to define what is meant by a well-formed move in a musical game. In general, these moves will be time-stamped musical events.

2 One-sentence description

This project will model and implement interactive music composition as a two-player cooperative game.

3 Project Type

Originality (software artifact)

4 Next Steps

1. Download and become familiar with Haskore/Euterpea and Hagl. Experiment with music generation using Paul Hudak's generative grammars and tools.
2. Produce a well-defined (coded) definition for moves (i.e. sequence of time-stamped musical events). In this case, r is a sequence of time-stamped musical events, which is defined through mutual recursion given a player's interpretation of a score and the most recent moves made by each player.

   $\mathbf{r_1 = instru_1\ (player_1\ s_1\ r_1\ r_2)}$

   $\mathbf{r_2 = instru_2\ (player_2\ s_2\ r_2\ r_1)}$

   To simplify, we will treat the score as a fixed length set of empty measures.
3. Produce a well-defined (coded) definition for a (Hagl) Game that models a simple, two-player game.
4. Using domain knowledge in music theory define a simple quantification scheme for the payoff of a move (i.e. quantify musical aesthetic).
5. Develop a few basic strategies for players. For example, try to match the note of the other player, try to create the best harmony, try to play dissonance, etc.
6. Experiment with payoffs and strategies to find those that produce nice sounding music.

5 Related Publications

List major publications that are most relevant to this project, and how they are related.

1. Paul Hudak and Jonathan Berger. "A Model of Performance, Interaction, and Improvisation". In: *Proceedings of International Computer Music Conference*. Int'l Computer Music Association. 1995

2. Paul Hudak et al. "Haskore Music Notation – An Algebra of Music". In: *Journal of Functional Programming* 6.3 (May 1996), pp. 465–483

3. Eric Walkingshaw and Martin Erwig. "A Domain-Specific Language for Experimental Game Theory". In: *Journal of Functional Programming (JFP)* 19 (2009), pp. 645–661

4. Eric Walkingshaw. "Domain-Specific Language Support for Experimental Game Theory". MA thesis. Oregon State University, 2011

5. Paul Hudak. *The Haskell School of Music – From Signals to Symphonies.* (Version 2.5), 2013