Generating Concurrent Musical Accompaniment using Artificial Intelligence

Gregory Hughes; Mardigon Toler
Dr. Sanish Rai, Advisor
WVU Tech Department of Computer Science and Information Systems

Introduction

- Driving Question: Can an algorithm be used to create real-time musical accompaniment?
- Artificial Intelligence may provide possible solutions.
- Goal: Create an AI agent that responds to real-time musical input with real-time musical output
- This agent, GenegleBot, plays accompaniment in real time while also following the most recently established pattern when no input is present.
- We approach this problem with a genetic algorithm:
  - an AI Technique for searching for solutions to problems inspired by evolutionary processes [1]

Methods

- Input & output are captured and transmitted as MIDI control messages to/from a keyboard
- We represent user input with a *queue* data structure of finite size
  - A histogram, is computed from the queue
- The Genetic Algorithm in Fig. 3 is comprised of *individuals*: possible histograms that will be ranked based on their similarity to user input.
  - Ranking is based on a fitness function (see figure 2) inspired by signal correlation [2].
- The algorithm relies on a control message for determining the tempo of the performance:
- A hardware/software clock source is polled for: **0xF8, 0xFA,** and, **0xFC** messages to determine tempo events, and messages are masked with **0xF0** and masked with **0x90** to determine when notes are input. Note data is extracted from the next byte of the control message.
- At each generation of the genetic algorithm (Figure 2 and Figure 3), the “best” individual (a histogram) is selected, which will be sampled for output notes.
- An example of the relationship between the *queue* and its associated histogram is shown below. Note that the advantage of using a queue is that this allows only the most recent part of a performance to be considered by the genetic algorithm

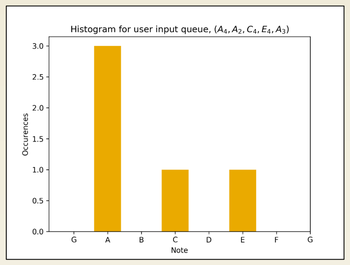

Fig. 1 Example histogram

$$Fit(\vec{v}) = \vec{v} \cdot \vec{y} = \sum_{i=1}^{m} \vec{v}_i \vec{y}_i$$

Fig. 2 Fitness function

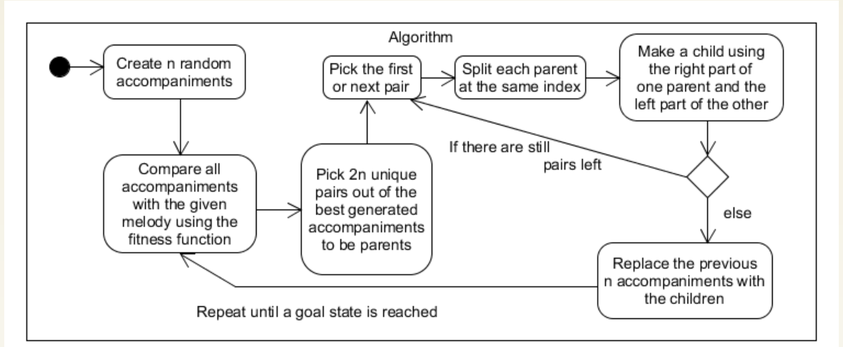

Fig 3. Simplified genetic algorithm

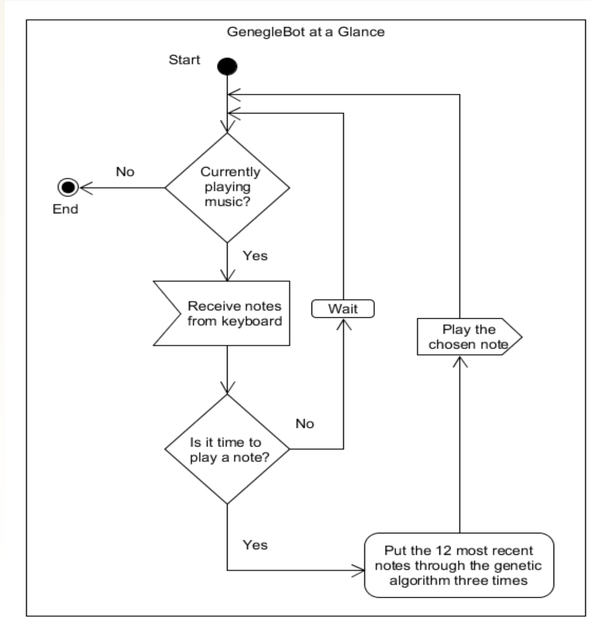

Fig. 4 Simplified Agent Algorithm

Results

- Exploring the performance of this approach to the problem revealed that the algorithm’s effectiveness relies on the monotony of the user input. Using a population size of 144 individuals with 3 generations of the genetic algorithm each time an individual is chosen for sampling notes, the fitness of each chosen individual occurred with frequencies shown in Figures 5, 6, and 7. For a chromatic scale as input, no individual ever exceeded a fitness of 3. When input consisted of a slightly monotonic melody in C, there was more variance in the distribution of individuals’ fitness. Finally, an input of an extremely monotonous D minor arpeggio resulted in many more highly fit individuals being produced.

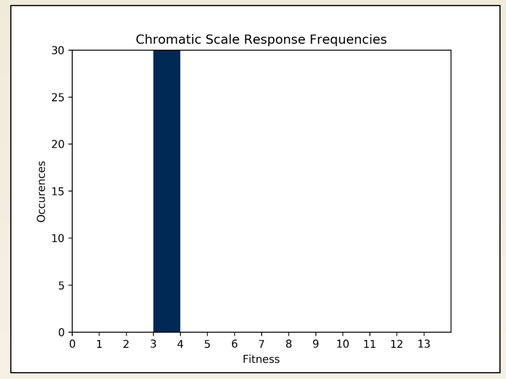

Fig. 5

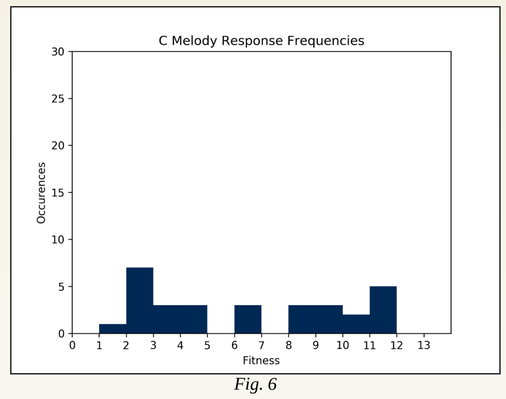

Fig. 6

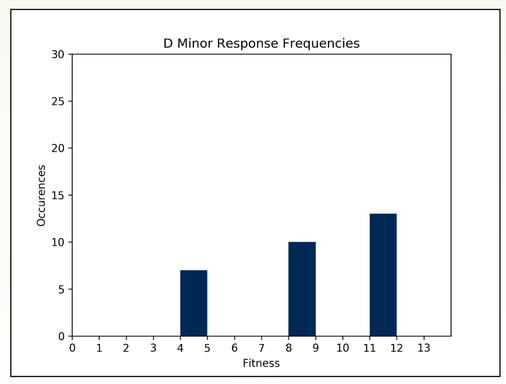

Fig. 7

Discussion

- Most software for generating musical accompaniment will generate an accompaniment for a prerecorded piece.
- This AI is rare in that it generates accompaniment in real time.
  - These techniques are a possible candidate for creating music software that will be helpful for any musician in need of an agent to play an extra part to a song.
- This genetic algorithm has still more potential if it is adapted to respond intelligently to rhythm.

References
[1] Whitley, Darrell. "A genetic algorithm tutorial." Statistics and computing 4.2 (1994): 65-85.

[2] Smith, Steven W. The scientist and engineers guide to digital signal processing. San Diego, CA: California Technical Publ., 1999.