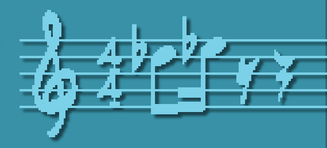
Algorithmic Composition of Jazz

Richard Harris, Dept. of Computer Science, University of Bath



Abstract

JazzGen *generates piano jazz improvisations over a user-supplied chord progression; melodies are evolved with a genetic algorithm based on both common sense and models of melodic and rhythmic expectancy.*

Introduction

Finding "good jazz melodies" in a problem space that has 10^{31} possibilities *per bar of music* requires an adaptive and intelligent search algorithm. Genetic algorithms (GAs) rapidly converge to *good* solutions and a random element that results in less predictable, more organic melodies.

JazzGen is highly configurable and extendable. The fitness function is composed of multiple routines which can be switched on or off: while music is being generated, these routines are *randomly* enabled, disabled and reweighted to change the style of the output.

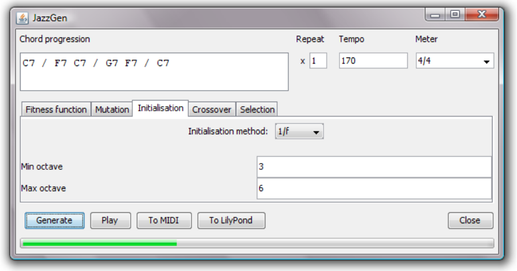
Background

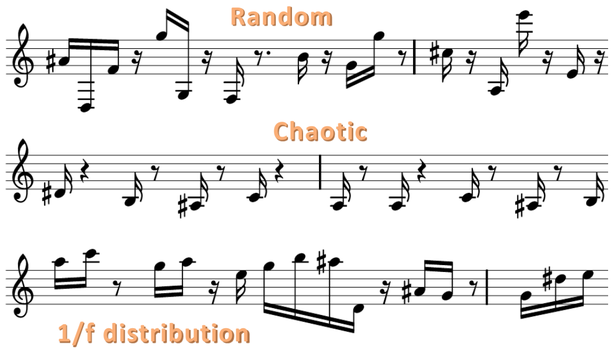
Biles (1994) famously produced *GenJam*, which relies on a human supervisor to tell it what sounds good or bad. Other systems utilise GAs to evolve harmonies or bridges between two distinct melodies. JazzGen requires no human intervention because its idea of "what's good" is based on hard-coded rules and statistics.

JazzGen also uses Narmour (1992)'s *"Implication-Realization"* model of *melodic expectancy* to evolve music that is not predictable, but conversely not too random or disjointed.

The Genetic Algorithm

1 **User inputs a chord progression, tempo and parameters for the genetic algorithm**





2 **Create initial candidates**

Several methods are employed. These "first tries" are worked on and improved by the genetic algorithm over thousands of generations.

3 **Assess *fitness* of each candidate in the population**

4 **Fight to the death**

Groups of melodies are randomly picked; only the *fittest* survive.

5 **Reproduction**

The best melodies are combined and subjected to random mutations. These form the new population, and we repeat.

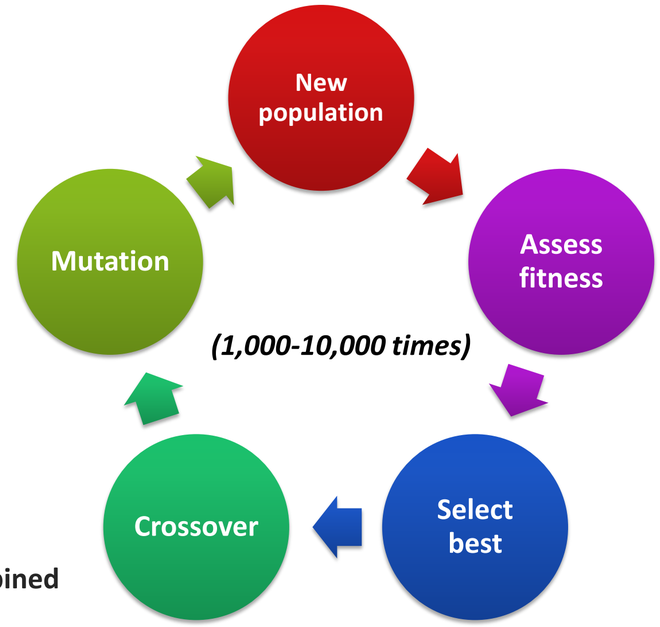


Figure 1 *The evolution process*

The **crossover** stage tries to combine the best parts of two parents into their offspring, while **mutation** stops the algorithm from converging on only one solution.

The Fitness Function

The evolving musical output is shaped by several factors:

pitch interval size	**melodic expectancy (keep close to ~75%)**
note lengths	**number of notes**
notes on/off beat	**triads (chord notes)**
note octave	**repetition**
syncopation	

Each test has a (modifiable) importance in determining the overall fitness. Tests can be turned on or off dynamically to change the melodic archetype of the output.

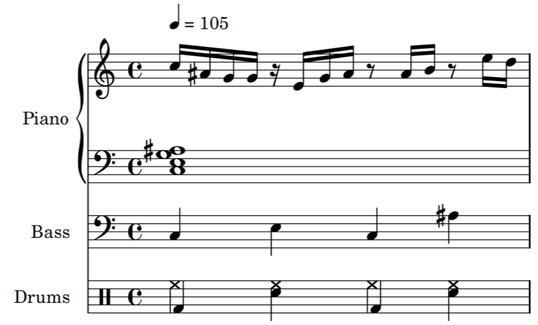


Figure 2 *Sample musical output*

Conclusion

So far, results are encouraging but basic and somewhat repetitive. Once the JazzGen UI is complete, extensive fitness function experimentation will be possible, which should lead to more finely-tuned behaviour and more aesthetically pleasing melodies.

References

Biles, J.A. (1994). *GenJam: A genetic algorithm for generating jazz solos.* Proceedings of the 1994 International Computer Music Conference

Horner, A. and Goldberg, D.E. (1991). *Genetic algorithms and computer-assisted music composition.*

Narmour, E. (1992). *The Analysis and Cognition of Melodic Complexity: The Implication-Realization Model*. Chicago: University of Chicago Press.