Life with GenJam: Interacting with a Musical IGA

John A. Biles
Information Technology Department
Rochester Institute of Technology
Rochester, NY 14623-5608 USA
http://www.it.rit.edu/~jab
jab@it.rit.edu

ABSTRACT

GenJam, short for ***Gen***etic ***Jam***mer, is an interactive genetic algorithm (IGA) that models a jazz improviser and performs as a featured soloist in the author's Virtual Quintet. GenJam evolves populations of melodic ideas under the guidance of a human mentor, whose feedback provides the environment under which individual ideas either survive to breed or die off. GenJam also uses its genetic algorithm machinery as a real-time melodic development paradigm to evolve phrases played by a human into its improvised responses in chase choruses. This paper provides an overview of the GenJam architecture and focuses on interface issues for three classes of users: mentors who train GenJam individually, audiences who train GenJam collectively, and performers who interact with GenJam in real-time performance situations.

1. INTRODUCTION

A common problem in applying evolutionary computation techniques to artistic domains is the difficulty of deriving formal fitness functions to evaluate the individuals in a population. Interactive genetic algorithms (IGAs) circumvent this problem by relying on a human *mentor* to experience a population and provide a subjective fitness for each individual. While this approach eliminates the need to formalize artistic merit, it does so at the cost of requiring the mentor to consider every individual in each generation, a task that takes considerable time and effort. This *fitness bottleneck* is particularly narrow in time-based domains like music, where individuals must be presented to the mentor singly, in real time[1]. This presents challenges, both for the design of genetic representations and operators, and for the design of mentor interfaces. The large populations and numerous generations commonly used in "standard" genetic algorithms (SGAs) simply are not feasible for IGAs. This means that genetic operators used in IGAs often must change individuals more radically and more intelligently than those used in SGAs so that those changes are more likely to result in noticeable improvements. For example, mutation operators typically must do more than flip an occasional bit to insure that they make discernible differences in individuals.

The mentor's interface is another critical concern in designing IGAs. Since mentors need to concentrate on the material they are evaluating, the interface must be efficient, unobtrusive and easy to use. This is especially critical in time-based domains like music, where material often must be presented at a fixed rate, which requires that the mentor process and respond to it appropriately in real time. A well designed interface can help reduce the fatigue experienced by mentors who must listen to and/or view an often monotonous stream of less-than-successful attempts while remaining alert for those that show promise [2].

GenJam is a time-based IGA that searches for pleasing melodic ideas that it uses to construct jazz improvisations. It can improvise full-chorus solos autonomously, and it can interact with a human performer in real time on chase choruses, where it listens to four-bar phrases played by a human, maps what it heard to its chromosome structure, mutates the chromosomes, and immediately performs the result as its response in the next four bars of the tune. GenJam models an active improviser, not a specific improvisation, so mentors who train a GenJam soloist are training a performer, not creating a specific performance. GenJam's interface for mentors and performers is the subject of this paper. After presenting an overview of GenJam's architecture, chromosome representations and genetic operators, the focus will shift to training GenJam as a mentor, experiencing GenJam in an audience, and working with GenJam as a performer.

2. GENJAM ARCHITECTURE

GenJam's architecture is summarized in Figure 1, where the rectangles represent files that GenJam accesses, and the ovals represent humans with whom GenJam interacts. Most of the files are simply read by GenJam and provide information about the tune to be performed.

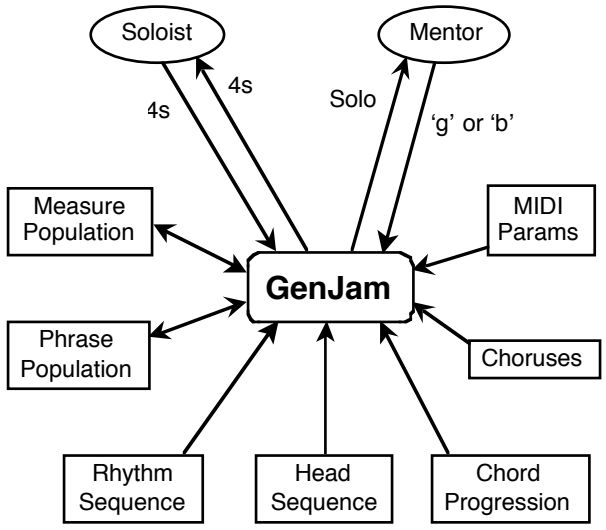

Figure 1. GenJam Architecture

The Chord Progression file contains the chords that define the harmonic progression for the tune, along with the tempo and the octaves in which GenJam should improvise. The Choruses file tells GenJam what it should do for each chorus of the form

defined in the chord progression (for example, improvise a full-chorus solo, trade fours with the human performer in a chase chorus, or rest while the human solos for a chorus). The MIDI Params file sets synthesizer parameters for the various parts, including loudness, instrument, location in the stereo field, and several more esoteric settings.

The two sequence files are standard MIDI sequences for the rhythm section and for the pre-arranged heads. The Rhythm Sequence file contains three to five parts played by the rhythm section to accompany the soloists (for example, piano, bass, drum, and possibly guitar and/or strings). The Head Sequence typically contains a harmony part for GenJam to play on the first and last choruses of the tune, which usually state the original melody. Both of these files are simply played back during the performance of a tune as GenJam and its human counterpart improvise. The desired "illusion" in performance is a jazz quintet with two front-line soloists (GenJam and the author) and a rhythm trio, hence the billing of the group as the Virtual Quintet [3].

The two population files, which are both read and updated by GenJam, contain hierarchically interrelated populations of individuals that represent melodic ideas. The population sizes are 64 individuals for the measure population and 48 for the phrase population. Individuals in both populations consist of a fitness value (an integer) and a chromosome (a bit string). An individual chromosome in the Measure Population decodes to a measure of eighth-note-length events that are mapped to actual MIDI messages in real time during performance. An individual chromosome in the Phrase Population decodes to a sequence of pointers to four individuals in the measure population. A GenJam solo, then, consists of a random sequence of phrases, which indirectly accesses a sequence of measures, which results in the sending of a sequence of MIDI note-on and note-off events to the synthesizer in real time.

35	25	23	14	59	8

23	7	14	14	13	14	13	12	12	11
14	26	11	11	10	11	15	12	13	14
59	-9	15	0	12	11	10	9	15	0
8	18	7	8	9	10	11	9	8	7

Figure 2. Example phrase and measure individuals

Figure 2 shows an example phrase individual and the four measure individuals it references. This phrase individual happens to be at offset 35 in the phrase population. Its fitness is 25, and the chromosome, which is actually a 24-bit string, is presented in segmented decimal form to show the measure offsets more clearly. The four measure individuals are presented in the order in which they occur in the example phrase instead of their actual locations in the measure population. For example, the third measure individual listed, offset 59 in the measure population, has a fitness of –9. The chromosome, which is actually a 32-bit string, is once again presented decimally and segmented to show the eight events more clearly.

Each eighth-note-length event is represented by four bits in the chromosome. A 0 decodes to a *rest* event, which is performed by sending a MIDI off message. A 15 decodes to a *hold* event, which causes no MIDI activity (the previous event is held). A 1 through 14 decodes to a *new-note* event, which is performed by sending a MIDI off to end any previous note, followed by a MIDI on to begin the new note. The pitch of the new note is determined by using the event as an index into a roughly two-octave scale of notes determined from the chord being played by the rhythm section at the time the event is performed. GenJam currently understands 17 different chord types [4], and the chord information for a tune is provided in the previously mentioned Chord Progression file. Figure 3 shows the phrase from Figure 2 as it would be performed against the chords from measures 29-32 of the jazz standard *All the Things You Are*.

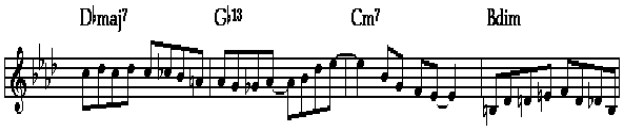

Figure 3. Phrase from Figure 2 as performed over progression

GenJam's chromosome representations efficiently integrate pitch and rhythmic structures, successfully represent a three-level hierarchy (notes, measures and phrases), and provide a huge search space of potential melodic material (slightly less than 2^{32} different measure individuals). On the down side, GenJam can play only eighth-note multiples, so its rhythmic sophistication is not great. It also cannot play theoretically wrong pitches, which insures competent improvisations but prohibits the creative rule breaking the marks master improvisers. In other words, GenJam plays it safe.

When creating a new soloist, both populations are initialized with randomly generated individuals. Fitnesses are initialize to zero, phrase chromosomes are initialized with a uniform random number generator, and measure individuals are initialized with a fractal generator biased toward smaller intervals between adjacent new-note events and an increased probability of rests and holds. As a mentor trains a soloist, the fitness values of the individuals in both populations diverge, and new individuals are bred. Breeding is accomplished using tournament selection, single-point crossover, and musically meaningful mutation. A tournament of four individuals is selected at random. The two fittest individuals in the tournament then are selected to be the parents; a simple single-point crossover is performed at the bit level to create two children; one of the children is mutated; and the two children replace the two non-parents from the tournament in the population. In each generation, 50% of the individuals in both populations are replaced.

GenJam's musically meaningful mutation operators perform radical changes to individuals and represent GenJam's knowledge of melodic development. The mutation operators on measures operate at the note level rather than the bit level and include such compositional devices as transposition, retrograde, inversion, retrograde inversion, rotation, sorting, and smoothing. The phrase mutations operate at the measure-pointer level rather than the bit level and include rotation, reversal, repetition, and several operators that insure diversity. The diversity-oriented operators counteract the tendency of the GA machinery to converge on a small number of highly fit individuals. While premature convergence can be a problem for GAs in general, convergence of any kind is a problem for GenJam. If the populations converge to a set of variations on a single theme, the resulting solos will consist of that theme repeated over and over, which is clearly not desirable. Again, the genetic search in GenJam is for an entire population that works well together, not for a single individual.

GenJam's real-time interaction during chase choruses is probably its most convincing performance feature. Trading

fours, where soloists take turns improvising on successive four-bar segments of a tune, is standard practice in jazz performance, particularly at jam sessions. The term chase chorus refers to the way soloists will chase one another by quoting and developing each other's melodic ideas in their own fours. GenJam uses its genetic representations and mutation operators to accomplish this.

Using a Roland GI-10 pitch-to-MIDI converter, GenJam listens to four measures of the human soloist's improvisation and maps the incoming MIDI events to four measure chromosomes and one phrase chromosome. While these chromosomes use the same format as described previously, they are not part of the measure and phrase populations that constitute a trained soloist. During roughly the last thirty-second note of the human's phrase, GenJam stops listening to the human, applies a random selection of its mutation operators to the phrase and measure chromosomes, and then performs the result as its next phrase. Since the target of the pitch tracking is GenJam's chromosome structure, the inevitable errors made by the pitch tracker are not a problem because whatever chromosomes GenJam ends up with will be playable, and the notes will always follow the chord changes (unlike the human soloist, on occasion!).

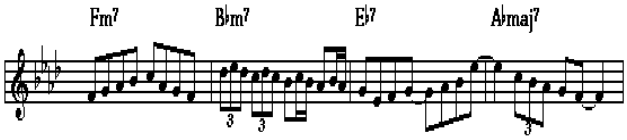

Figure 4. Human phrase mutated into phrase in Figure 3

Figure 4 shows a phrase played by the author over measures 25 to 28 of *All the Things You Are*. GenJam's actual response to this phrase after mutation was shown in Figure 3. The two phrases are different in many respects, partly due to the mapping process from the human's four to GenJam's chromosome structure and partly due to the mutations that were applied. The result, however, illustrates that evolutionary techniques applied in real time can produce spontaneous behavior in an interactive system. We turn now to the interface issues for mentors, audiences and performers.

3. MENTOR INTERACTION

The mentor's task is to listen critically to GenJam improvise and provide immediate, ongoing feedback on the perceived quality of GenJam's improvisation. As indicated in Figure 1, the mentor registers feedback by simply typing 'g' for good, 'b' for bad, or nothing for neutral. Each time the mentor types 'g', the fitness values for the measure and phrase individuals currently being performed are incremented by one. Each time the mentor types 'b' the fitness values are decremented by one. The letters 'g' and 'b' were chosen initially because they clearly stood for "good" and "bad," thereby providing an obvious mapping for mentors. After using the interface for a while, it became clear that 'g' and 'b' also possessed ergonomic benefits because mentors could use their left hands, placing the index finger on the 'b' and the middle finger on the 'g'. This is relatively comfortable and keeps the mentor's right hand free for sipping coffee, a definite plus during extended training sessions!

To allow time for the mentor to react, delays have been built into the feedback mechanisms so that the feedback window for measures is shifted two beats late and the window for phrases one measure late. This means that when a 'g' or 'b' is typed during the playing of beats three or four of a measure, the fitness for that measure is incremented or decremented. Feedback typed during beats one or two will affect the previous measure. Similarly, feedback occurring in the first measure of a phrase applies to the previous phrase, while feedback in measures two, three or four affect the current phrase.

The measure delay was derived empirically by seeding an otherwise random measure population with quotes from recognizable tunes, asking mentors to listen and respond with a 'g' when they heard something they recognized, varying the measure delay in several trials at various tempi, and evaluating the proportion of feedback that landed in the appropriate measure windows. Beat-oriented delays, which are tied to tempo, worked better than fixed-time delays. It seems that mentors respond more quickly on faster tunes, or, in other words, the tempo of their feedback seemed to correlate with the tune's tempo.

The delay for phrase individuals was set at one measure, which in 4/4 time is twice the length of the measure delay. This was done after failed attempts to design an interface that allowed mentors to provide distinct feedback for measures and phrases. For example, one approach used 'G' and 'B' for phrase feedback and 'g' and 'b' for measures. The intended mapping was big letters for phrases, small letters for measures. It became clear very quickly that this interface was unusable because mentors simply couldn't decide fast enough which level their opinions applied to, much less push the right buttons. However, it was clear that opinions at the phrase level took longer to form, so the one-measure delay for phrase feedback was adopted as a reasonable solution. Another simplification to the interface was the elimination of any visual feedback from GenJam when the mentor types 'g' or 'b'. This would seem to contradict conventional interface design principles, which stress the importance of system feedback for user actions [5], but many mentors report that visual stimuli are distracting. Musicians in particular tend to listen to GenJam with their eyes closed to focus on what they are hearing.

Even with GenJam's simplified interface, which requires only that the mentor type 'g's or 'b's whenever so moved, the mentor's task is unnatural for most people. To study mentor behavior, the author has conducted informal studies at an annual technology exposition produced by the Information Technology Students Organization at RIT (the ITSO Expo). The author sets up GenJam in a booth that allows passersby to sit down and train a new soloist for as many generations as they wish. The author fills the time between these impromptu training sessions by performing with the Virtual Quintet, which tends to draw potential mentors to the booth. While this mentor selection process is clearly biased toward people who find the music appealing enough to stop and find out how it works, the Expo attendees nonetheless represent a diverse mix of ages, musical abilities and computer experience.

The anecdotal findings from these and other mentoring sessions are interesting, if not terribly rigorous. Mentors who feel most comfortable in the mentoring role possess at least one of the following three attributes: they either 1) like jazz, 2) are a musician, or 3) are active in the computer field. People who lack all three of these attributes tend to be intimidated by the prospect of serving as a mentor and seldom spend more than one tune trying it, if they can be cajoled into trying it at all. This seems to be due to their inability to form a mental model of what the mentor is supposed to do. They literally lack enough

knowledge of music in general and jazz in particular to be able to form opinions about GenJam's improvisations, and they often feel intimidated at having to render an opinion on something they don't really understand.

Mentors who possess one or two of the above attributes still may stumble over the intensity of listening required in the mentoring task. Most people listen to music passively, often while doing other things. Occasionally they focus primarily on what they are hearing, but most of the time, the music is in the background, particularly if the music has no vocals, as is the case with instrumental jazz. On the other hand, the mentoring task requires active listening; the mentor must focus closely on the music and constantly be in touch with how much they like it. Even professional music educators, who listen critically to music for a living, have trouble with the relentless requirement that the mentor express opinions in real time during a performance rather than after the performance has concluded.

Mentors who possess all three of the above attributes still can have trouble with the mentoring task, often because they build overly complex mental models of how GenJam works. This is particularly true of mentors who study or work in computing-related fields. These mentors often have trouble with the simple system image presented to them and make assumptions about how GenJam "really" works. This kind of behavior speaks to the system images of many software products, where knowledge of the underlying architecture of the system is helpful, if not necessary, for its successful use. Computing professionals are used to finding out or guessing how a software product works "under the hood," and they often develop techniques to exploit that knowledge.

All of this leads to an interesting diversity of coping strategies that help mentors impose a structure on the task. Some mentors tap in time to the music, changing from 'g' to 'b' whenever so moved, but always typing something on the beat. Other mentors use a regular but slower tempo, making a single tap at the beginning of each measure. Another, more nervous approach is to tap very rapidly at all times, usually not in time to the music. At the other extreme are mentors who make a single deliberate tap only occasionally, maybe once every several measures or so.

Regardless of the approach, mentors tend to report that the system is responsive and that successive generations are better. The fact that such diverse behavior patterns all tend to work is a tribute to the robustness of GAs in the face of noisy objective functions. IGAs tend to be noisier than most GAs, and GenJam is noisy for an IGA, but as long as the fitness values differ in the right direction, whether it be by one or by 30 units, appropriate selection occurs, and the paradigm works.

4. AUDIENCE INTERACTION

Audiences have interacted with GenJam both as a collective mentor in audience-mediated performance situations and in more traditional performances by listening and (hopefully) applauding. Even in traditional settings the audience can be conceptualized as the "users" of the performance. This user class brings varying levels of familiarity with computing, music in general and jazz in particular, and this knowledge informs their mental models of the Virtual Quintet and GenJam. Most people who experience GenJam without any information on the technology believe that the music coming from the sound system is a CD and don't realize that it is a live, interactive performance. Even listeners who identify GenJam as a MIDI system tend to believe that GenJam's solos are simply prerecorded MIDI sequences. From a performance standpoint, the biggest problem is that the interaction is only at an aural level and is not visual. Audiences at jazz performances expect to see the interaction between musicians, not just hear it.

To help inform and engage audiences, the author has experimented with *audience-mediated performances* [6] in several concerts. The usual scenario begins with the Virtual Quintet playing from two tunes to a 45-minute set using pre-trained soloists in order to acquaint the audience with what GenJam sounds like. Audience members then are given feedback paddles, which are green on one side and red on the other. They are instructed to show the green side to the stage if they like what GenJam is playing during a solo, show the red side if they don't like what GenJam is playing, and hide the paddle if they have no strong feeling either way. The author then plays from two to four tunes, arranged so that GenJam improvises full-chorus solos for each chorus, and collects the audience feedback by judging the relative amounts of red and green visible and typing 'g's and 'b's as appropriate. The soloist used is created randomly for the first tune in the training set, and each tune represents the next generation of this new soloist. When the author judges that the audience is beginning to get restless with the training regimen, he ends the training session, and the entire Virtual Quintet then plays one or more tunes using the audience's soloist.

A major difference between mentor behavior in this collective setting and mentor behavior in an individual setting is that people who are reluctant to serve as an individual mentor usually give it a try in the group setting. This is likely due to behavior modeling of the more enthusiastic audience members, dilution of the perceived impact of "wrong" feedback, and/or peer pressure to perform. One delightful byproduct of this scenario is the audience's use of the paddles to provide feedback to the human's first solo following the training set. This is always accompanied by broad grins, and the predominant color tends to be green (thankfully!).

5. PERFORMER INTERACTION

Performing with GenJam in the Virtual Quintet is an unusual experience that offers both advantages and disadvantages, compared to playing with people. On the negative side, the Virtual Quintet can be somewhat stiff. Tempi never vary; the rhythm section always plays the correct chord changes; and everybody (except the human) always plays in tune and articulates notes flawlessly. Since the human is the only visible performer, he is the primary visual component of the group's interface with the audience and is always on stage, even when GenJam is playing a full-chorus solo. This leads to a central performance issue – what does the human do while GenJam is taking a full-chorus solo? In an all-human jazz group, musicians can watch the soloist, offer encouragement, and otherwise provide indirect input to the performance. In the Virtual Quintet, such actions are somewhat forced, if not downright silly. One simple solution to this would be to write arrangements where GenJam never solos; however, this not only defeats the purpose of GenJam, but it also is not feasible for the human soloist, who would be playing on every chorus of every tune. Unlike GenJam, the human needs a break!

Advantages of the Virtual Quintet include working with a drummer who doesn't rush, writing arrangements for any combination of instruments that will be played as written, and working with soloists that can be replaced if their ideas get stale.

Indeed, the author uses a collection of about a dozen different soloists that have been trained on tunes of different musical styles and are evolved or replaced regularly when they become too predictable.

The major advantage to playing with GenJam, though, is its facility on chase choruses. In many ways, GenJam is the most formidable "opponent" the author has encountered in the thousands of jam sessions and gigs he has played over the last 25 or so years. GenJam is a master at hearing the previous soloist's four and developing it in interesting ways. The musical conversations that result are highly stimulating for the human soloist and help make gigs with GenJam very enjoyable to play.

6. CONCLUSION

One way to think of evolutionary computation in general and genetic algorithms in particular is as a classic generate-and-test strategy. The success of such a strategy certainly depends on the quality of the test, but the efficiency often depends on the generators. If the generators generate better individuals, then the search for individuals that pass the test should be briefer. For GAs the test is, of course, the fitness, which, as this paper demonstrates, can present problems in an IGA. The generators in a GA are the randomization of the initial population, along with the genetic operators used to breed new individuals. GenJam demonstrates that making these operators more intelligent may ease the IGA fitness bottleneck.

Taking it one step further, can the fitness bottleneck be eliminated altogether by eliminating the need for determining fitness? In other words, can the generators be made so good that fitness is not necessary? To begin to address this question, two informal experiments were conducted with GenJam, the first during its initial development in 1994 and the second recently. In both cases, GenJam was trained for several generations with no feedback from a mentor, which resulted in all individuals retaining a fitness of zero. These uniform fitness values result in GenJam selecting parents randomly, so any improvement in the populations would be due primarily to the mutation operators. When this was done in GenJam's infancy, the resulting soloists sounded little better after dozens of generations than they sounded in the initial generation. The conclusion drawn was that although the mutation operators were intelligent, they were not powerful enough by themselves to generate good individuals, and that fitness was an important component of successful breeding.

During the most recent ITSO Expo, a musically capable mentor remarked that it seemed to him as though GenJam were responding to his feedback and that he thought it was beginning to sound the way he likes to hear jazz played, although it still surprised him, albeit pleasantly. In other words, he liked what he heard, but he wasn't convinced he was responsible for the improvement. The author's interpretation of this episode was that the improvements made over the last five years in the mutation operators and the introduction of the fractal measure initialization might have more to do with GenJam's success than the feedback from mentors.

To test this conjecture, the no-feedback experiment was tried again, and this time the results were different. First, the initial generation sounds better now than it did 1994, a tribute to the fractal initialization routine. The second result was that successive generations also sounded good, although not as good as a well trained soloist. The difference seems to be that training with no feedback produces soloists that don't sound awful, but these soloists seldom come up with the really nice moments that tend to emerge in well trained soloists.

As a final test, several soloists were trained to sound bad rather than good, and the result was that mentors were indeed able to overcome the mutation operators quite easily and produce soloists that got progressively worse in later generations. The conclusion, then, is that the fitness remains a critical issue in IGAs, and, from this final test, that there is no accounting for taste.

7. REFERENCES

[1] J. A. Biles. "GenJam: A Genetic Algorithm for Generating Jazz Solos," In *Proceedings of the 1994 International Computer Music Conference*, ICMA, San Francisco, 1994.

[2] H. TAKAGI, "Interactive Evolutionary Computation – Cooperation of computational intelligence and human KANSEI," 5th Int'l Conf. On Soft Computing (IIZUKA'98), pp. 41-50, World Scientific, Iizuka, Fukuoka, Japan, (Oct. 16-20, 1998).

[3] Al Biles Virtual Quintet. *GenJam*. Compact disk DRK-144. Dynamic Recording Studio, Rochester, NY, 1995.

[4] J. A. Biles. Interactive GenJam: Integrating Real-time Performance with a Genetic Algorithm. In *Proceedings of the 1998 International Computer Music Conference*, ICMA, San Francisco, 1998.

[5] D. A.. Norman. *The Design of Everyday Things*. Doubleday, New York, 1988.

[6] J. A. Biles and W. Eign. GenJam *Populi*: Training an IGA *via* Audience-Mediated Performance. In *Proceedings of the 1995 International Computer Music Conference*, ICMA, San Francisco, 1995.