AI and Music
From Composition to Expressive Performance

Ramon Lopez de Mantaras and Josep Lluis Arcos

■ In this article, we first survey the three major types of computer music systems based on AI techniques: (1) compositional, (2) improvisational, and (3) performance systems. Representative examples of each type are briefly described. Then, we look in more detail at the problem of endowing the resulting performances with the expressiveness that characterizes human-generated music. This is one of the most challenging aspects of computer music that has been addressed just recently. The main problem in modeling expressiveness is to grasp the performer's "touch," that is, the knowledge applied when performing a score. Humans acquire it through a long process of observation and imitation. For this reason, previous approaches, based on following musical rules trying to capture interpretation knowledge, had serious limitations. An alternative approach, much closer to the observation-imitation process observed in humans, is that of directly using the interpretation knowledge implicit in examples extracted from recordings of human performers instead of trying to make explicit such knowledge. In the last part of the article, we report on a performance system, SaxEx, based on this alternative approach, that is capable of generating high-quality expressive solo performances of jazz ballads based on examples of human performers within a case-based reasoning (CBR) system.

AI has played a crucial role in the history of computer music almost since its beginning in the fifties. However, until recently, most efforts had been on compositional and improvisational systems, and little efforts had been devoted to performance systems. Furthermore, AI approaches to performance tried to capture the performer's *touch*, that is, the knowledge applied when performing a score by means of rules. This performance knowledge concerns not only technical features but also the affective aspects implicit in music. Humans acquire it through a long process of observation and imitation (Dowling and Harwood 1986). For this reason, AI approaches, based on following musical rules trying to capture interpretation knowledge, have serious limitations when attempting to model humanlike music performance. The last sections of this article describe the SaxEx system (Arcos and Lopez de Mantaras 2001; Arcos, Lopez de Mantaras, and Serra 1998).[1] SaxEx directly uses the performance knowledge implicit in examples extracted from recordings of human performers instead of trying to make that knowledge explicit by means of rules. With this approach, closer to the observation-imitation process observed in humans, SaxEx is capable of generating high-quality humanlike monophonic (one single melodic instrument) performances of jazz ballads based on examples of human performers.[2]

SaxEx is based on a type of analogical reasoning called CBR. In CBR (Aamodt and Plaza 1994; Kolodner 1993; Leake 1996), problems are solved by reusing (often through some sort of adaptation process) the solutions to similar, previously solved problems. It relies on the reasonable assumption that similar problems have similar solutions. CBR is appropriate for problems where (1) many examples of already solved similar problems are available and (2) a large part of the knowledge involved in the solution of problems is *tacit,* that is, difficult to verbalize and generalize (as is the case for knowledge involved when playing expressively). An additional advantage of CBR is that each new solved problem can be revised by a human and memorized, and therefore, the system can improve its problem-solving capabilities by experience; in other words, it learns by experience. It is worth noticing here that although the use of CBR was motivated by the observation-

imitation process observed in humans, we do not claim that SaxEx entirely reproduces what musicians do when performing music.

This article starts by briefly surveying a set of representative work on composition, improvisation, and performance. Then, through the description of SaxEx, we look in more detail at the problem of endowing the computer-generated performances with the expressiveness that characterizes human performances. Indeed, as described in the AI and Music section, existing approaches typically deal with two expressive resources (such as dynamics and rubato, rubato and vibrato, or rubato and articulation). In contrast, with the approach used in SaxEx, we are capable of dealing with all the main expressive resources: *dynamics* (loudness), *rubato* (variations in note duration and attack time), *vibrato* (repeated small variation in pitch's frequency and amplitude), *articulation* (variations in the transition time between notes), and *attack of the notes*. Another distinctive aspect of SaxEx is that it works with non-MIDI sound files, which makes the problem much more difficult but also more interesting because the resulting performances are much more humanlike. We end the article by describing the experiments performed, the results obtained, and presenting a discussion on the limitations and conclusions.

AI and Music

In this section, we survey a set of representative computer music systems that use several AI techniques. It is organized into three subsections: The first is devoted to compositional systems, the second describes improvisation systems, and the third is devoted to performance systems. These are the subfields of computer music in which AI techniques have mostly been applied. In the performance subsection, we raise the important problem of generating humanlike expressive performances. An example of a successful step toward humanlike performance is described in detail in the last sections of the article.

Compositional Systems

Hiller and Isaacson's (1958) work on the ILLIAC computer, is the best-known pioneering work in computer music. Their chief result is the Illiac Suite, a string quartet composed following the generate-and-test problem-solving approach. The program generated notes pseudorandomly by means of Markov chains. The generated notes were next tested by means of heuristic compositional rules of classical harmony and counterpoint. Only the notes satisfying the rules were kept. If none of the generated notes satisfied the rules, a simple backtracking procedure was used to erase the entire composition to that point, and a new cycle was started again. The goals of Hiller and Isaacson excluded anything related to expressiveness and emotional content. In an interview (Schwanauer and Levitt 1993), Hiller and Isaacson said that before addressing the expressiveness issue, simpler problems needed to be handled first. We believe that this was a very correct observation in the fifties. After this seminal work, many other researchers based their computer compositions on Markov probability transitions but also with rather limited success, judging from the standpoint of melodic quality. Indeed, methods relying too heavily on Markovian processes are not complete enough to produce high-quality music consistently.

However, not all the early work on composition relies on probabilistic approaches. A good example is the work of Moorer (1972) on tonal melody generation. Moorer's program generated simple melodies, along with the underlying harmonic progressions, with simple internal repetition patterns of notes. This approach relies on simulating human composition processes using heuristic techniques rather than on Markovian probability chains. Levitt (1993) also avoided the use of probabilities in the composition process. He argued that randomness tends to obscure rather than reveal the musical constraints needed to represent simple musical structures. His work is based on constraint-based descriptions of musical styles. He developed a description language that allows expressing musically meaningful transformations of input, such as chord progressions and melodic lines, through a series of constraint relationships that he calls *style templates*. He applied this approach to describe a traditional walking jazz bass player simulation as well as a two-handed ragtime piano simulation.

The early systems by Hiller and Isaacson and Moore were both based also on heuristic approaches. However, possibly the most genuine example of the early use of AI techniques is the work of Rader (1974). Rader used rule-based AI programming in his musical round (a circle canon such as "Frère Jacques") generator. The generation of the melody and the harmony were based on rules describing how notes or chords can be put together. The most interesting AI components of this system are the *applicability rules,* which determine the applicability of the melody and chord-generation rules, and the *weighting rules,* which generate the likelihood of application of an applicable rule by means of a weight. We can already appreciate the use of metaknowledge in this early work.

AI pioneers such as Herbert Simon or Marvin Minsky also published works relevant to computer music. Simon and Sumner (1968, p. 83) described a formal pattern language for music as well as a pattern induction method to discover patterns more or less implicit in musical works. Simon and Sumner (1968, p. 83) describe one example of a pattern that can be discovered: "The opening section is in C Major, it is followed by a section in dominant and then a return to the original key." Although the program was not completed, it is worth noticing that it was one of the firsts in dealing with the important issue of music modeling, a subject that has been, and still is, largely studied. For example, the use of models based on generative grammars has been, and continues to be, an important and useful approach in music modeling (Lerdahl and Jackendoff 1983).

Marvin Minsky (1981) in his well-known paper Music, Mind, and Meaning addresses the important question of how music impresses our minds. He applies his concepts of agent and its role in a society of agents as a possible approach to shed light on this question. For example, he hints that one agent might do nothing more than noticing that the music has a particular rhythm. Other agents might perceive small musical patterns such as repetitions of a pitch and differences such as the same sequence of notes played one fifth higher. His approach also accounts for more complex relations within a musical piece by means of higher-order agents capable of recognizing large sections of music. It is important to clarify that in this paper, Minsky does not try to convince the reader about the question of the validity of his approach; he just hints at its plausibility.

Among the compositional systems, there are a large number dealing with the problem of automatic harmonization using several AI techniques. One of the earliest works is that of Rothgeb (1969). He wrote a SNOBOL program to solve the problem of harmonizing the unfigured bass (given a sequence of bass notes, infer the chords and voice leadings that accompany these bass notes) by means of a set of rules such as "if the bass of a triad descends a semitone, then the next bass note has a sixth." The main goal of Rothgeb was not the automatic harmonization itself but testing of the computational soundness of two bass harmonization theories from the eighteenth century.

One of the most complete works on harmonization is that of Ebcioglu (1993). He developed an expert system, CHORAL, to harmonize chorales in the style of J. S. Bach. CHORAL is given a melody and produces the corresponding harmonization using heuristic rules and constraints. The system was implemented using a logical programming language designed by the author. An important aspect of this work is the use of sets of logical primitives to represent the different viewpoints of the music (chords view, time-slice view, melodic view, and so on). These sets of logical primitives allowed tackling the problem of representing large amounts of complex musical knowledge.

MUSACT (Bharucha 1993) uses neural networks to learn a model of musical harmony. It was designed to capture musical intuitions of harmonic qualities. For example, one of the qualities of a dominant chord is to create in the listener the expectancy that the tonic chord is about to be heard. The greater the expectancy, the greater the feeling of consonance of the tonic chord. Composers can choose to satisfy or violate these expectancies to varying degrees. MUSACT is capable of learning such qualities and generate graded expectancies in a given harmonic context

In HARMONET (Feulner 1993), the harmonization problem is approached using a combination of neural networks and constraint-satisfaction techniques. The neural network learns what is known as harmonic functions of the chords (chords can play the function of tonic, dominant, subdominant, and so on), and constraints are used to fill the inner voices of the chords. The work on HARMONET was extended in the MELONET system (Hörnel and Dagenhardt 1997; Hörnel and Menzel 1998). MELONET uses a neural network to learn and reproduce higher-level structure in melodic sequences. Given a melody, the system invents a baroque-style harmonization and variation of any chorale voice. According to the authors, HARMONET and MELONET together form a powerful music-composition system that generates variations whose quality is similar to those of an experienced human organist.

Pachet and Roy (1998) also used constraint-satisfaction techniques for harmonization. These techniques exploit the fact that both the melody and the harmonization knowledge impose constraints on the possible chords. However, efficiency is a problem with pure constraint-satisfaction approaches.

In Sabater, Arcos, and López de Mántaras (1998), the problem of harmonization is approached using a combination of rules and CBR. This approach is based on the observation that purely rule-based harmonization usually fails because in general the rules don't make the music; it is the music that makes the rules. Then, instead of relying only on a set of imperfect rules, why not making use of the source of the rules, that is, the compositions

themselves? CBR allows using examples of already harmonized compositions as cases for new harmonizations. The system harmonizes a given melody by first looking for similar, already harmonized, cases; when similar cases cannot be found, it looks for applicable general rules of harmony. If no rule is applicable, the system fails and backtracks to the previous decision point. The experiments have shown that the combination of rules and cases results in much fewer failures in finding an appropriate harmonization than using either technique alone. Another advantage of the case-based approach is that each newly correctly harmonized piece can be memorized and made available as a new example to harmonize other melodies. That is, a learning by experience process takes place. Indeed, the more examples the system has, the less often the system needs to resort to the rules, and therefore, it fails less. MUSE (Schwanauer 1993) is also a learning system that extends an initially small set of voice-leading constraints by learning a set of rules of voice doubling and voice leading. It learns by reordering the rules agenda and chunking the rules that satisfy the set of voice-leading constraints. MUSE successfully learned some of the standard rules of voice leading included in traditional books of tonal music.

Morales-Manzanares et al. (2001) developed a system called SICIB capable of music composition using body movements. This system uses data from sensors attached to the dancer and Prolog rules to couple the gestures with the music. The architecture of SICIB also allows real-time performance.

Certainly the best-known work on computer composition using AI is David Cope's (1990, 1987) EMI project. This work focuses on the emulation of styles of various composers. It has successfully composed music in the styles of Cope, Mozart, Palestrina, Albinoni, Brahms, Debussy, Bach, Rachmaninoff, Chopin, Stravinsky, and Bartok. It works by searching for recurrent patterns in several (at least two) works of a given composer. The discovered patterns are called *signatures*. Because signatures are location dependent, EMI uses one of the composer's works as a guide to fix them to their appropriate locations when composing a new piece. To compose the musical motives between signatures, EMI uses a compositional rule analyzer to discover the constraints used by the composer in his/her works. This analyzer counts musical events such as voice- leading directions and the use of repeated notes and represents them as a statistical model of the analyzed works. The program follows this model to compose the motives to be inserted in the empty spaces between signatures. To properly insert them, EMI has to deal with problems such as linking initial and concluding parts of the signatures to the surrounding motives avoiding stylistic anomalies, maintaining voice motions, and maintaining notes within a range. Proper insertion is achieved by means of an augmented transition network (Woods 1970). The results, although not perfect, are quite consistent with the style of the composer.

Improvisation Systems

An early work on computer improvisation is the FLAVORS BAND system of Fry (1984). FLAVORS BAND is a procedural language, embedded in Lisp, for specifying jazz and popular music styles. Its procedural representation allows the generation of scores in a prespecified style by making changes to a score specification given as input. It allows combining random functions and musical constraints (chords, modes, and so on) to generate improvisational variations. The most remarkable result of FLAVORS BAND was an interesting arrangement of the bass line, and an improvised solo, of John Coltrane's composition "Giant Steps."

GENJAM (Biles 1994) builds a model of a jazz musician learning to improvise by means of a genetic algorithm. A human listener plays the role of fitness function by rating the offspring improvisations. Papadopoulos and Wiggins (1998) also used a genetic algorithm to improvise jazz melodies on a given chord progression. Contrary to GENJAM, the program includes a fitness function that automatically evaluates the quality of the offspring improvisations rating eight different aspects of the improvised melody, such as the melodic contour, note duration, and intervallic distances between notes.

Franklin (2001) uses recurrent neural networks to learn how to improvise jazz solos from transcriptions of solo improvisations by saxophonist Sonny Rollins. A reinforcement learning algorithm is used to refine the behavior of the neural network. The reward function rates the system solos in terms of jazz harmony criteria and Rollins style.

The lack of interactivity, with a human improviser, of these previous approaches has been criticized (Thom 2001) on the grounds that they remove the musician from the physical and spontaneous creation of a melody. Although it is true that the most fundamental characteristic of improvisation is the spontaneous, real-time, creation of a melody, it is also true that interactivity was not intended in these approaches, but nevertheless, they could generate very interesting improvisations.

Thom (2001) with her BAND-OUT-OF-A-BOX (BoB) system addresses the problem of real-time interactive improvisation between BoB and a human player. In other words, BoB is a "music companion" for real-time improvisation. Thom's approach follows Johnson-Laird's (1991) psychological theory of jazz improvisation. This theory opposes the view that improvising consists of rearranging and transforming prememorized "licks" under the constraints of a harmony. Instead he proposes a stochastic model based on a greedy search over a constrained space of possible notes to play at a given point in time. The very important contribution of Thom is that her system learns these constraints and, therefore, the stochastic model, from the human player by means of an unsupervised probabilistic clustering algorithm. The learned model is used to abstract solos into user-specific playing modes. The parameters of this learned model are then incorporated into a stochastic process that generates the solos in response to four-bar solos of the human improviser. BoB has been very successfully evaluated by testing its real-time solo tradings in two different styles, that of saxophonist Charlie Parker and that of violinist Stephane Grapelli.

Another remarkable interactive improvisation system was developed by Dannenberg (1993). The difference with Thom's approach is that in Dannenberg's system, music generation is mainly driven by the composer's goals rather than the performer's goals. Wessel's (Wessel, Wright, and Kahn 1998) interactive improvisation system is closer to Thom's in that it also emphasizes the accompaniment and enhancement of live improvisations.

Performance Systems

The compositional and improvisation systems described to this point, even though they generated some audible output in many cases (very early systems by electronic means and later systems by MIDI instruments), did not specifically address the problem of expressiveness. However, for performance systems, *expressiveness* is a mandatory concern because the brain is mainly interested in change. Auditory neurons, like the other neurons in the brain, fire constantly even in silent environments. What really matters is not the firing rate but the changes in firing rate. There are auditory neurons whose firing rate changes only when the sound frequency or the intensity increase or decrease. Other neurons react similarly when a sound repeats. Conversely, most of the primary auditory neurons also exhibit what is known as *habituation* (Baars 1998). When neurons repeatedly receive the same stimulus, their firing rate, instead of remaining constant, decreases over time, which means that we deafen to a sound unless it manifests some form of novelty or renewal in its characteristics. Therefore, it is not surprising that music becomes more interesting when it is not too repetitive; that is, when it contains alterations in dynamic, pitch, and rhythm. This pack of alterations might partly explain why synthesized music is much less interesting than human-performed music: A real instrument sends to the auditory cortex more stimuli to react to than synthesized music. Indeed, the so-called expressive resources provide an extremely rich source of changes to our brains.

In the remaining sections of this article, we focus on the use of AI techniques for generating expressive performances. It is worth noticing that there has been much less work on AI techniques for performance than on AI for composition and improvisation and that all the existing work is recent.

One of the first attempts to provide high-level musical transformations using a rule-based system is that of Johnson (1992). She developed an expert system to determine the tempo and the articulation to be applied when playing Bach's fugues from "The Well-Tempered Clavier." The rules were obtained from two expert human performers. The output gives the base tempo value and a list of performance instructions on note duration and articulation that should be followed by a human player. The results very much coincide with the instructions given in well-known commented editions of "The Well-Tempered Clavier." The main limitation of this system is its lack of generality because it only works well for fugues written in a 4/4 meter. For different meters, the rules should be different. Another obvious consequence of this lack of generality is that the rules are only applicable to Bach fugues.

The work of the KTH group from Stockholm (Bresin 2001; Friberg 1995; Friberg, Sunberg, and Fryden 2000; Friberg et al. 1998) is one of the best-known long-term efforts on performance systems. Their current DIRECTOR MUSICES system incorporates rules for tempo, dynamic, and articulation transformations constrained to MIDI. These rules are inferred both from theoretical musical knowledge and experimentally from training, especially using the so-called analysis-by-synthesis approach. The rules are divided into three main classes: (1) *differentiation rules,* which enhance the differences between scale tones; (2) *grouping rules,* which show what tones belong together; and (3) *ensemble rules,* that synchronize the various

voices in an ensemble.

Another long-term effort is the project led by G. Widmer (2001, 1996). It is another example of the use of rules for performing transformations of tempo and dynamics. The approach taken in this project was first to record and preprocess a large amount of high-quality musical performances (containing more than 400,000 notes). The large amount of examples is used to induce performance rules by means of machine learning techniques. The project has already produced very promising results.

Canazza et al. (1997) developed a system to analyze how the musician's expressive intentions are reflected in the performance. The analysis reveals two different expressive dimensions: (1) one related to the energy (dynamics) and (2) the other one related to the kinetics (rubato) of the piece. The authors also developed a program for generating expressive performances according to these two dimensions.

The work of Dannenberg and Derenyi (1998) is also a good example of articulation transformations using manually constructed rules. They developed a trumpet synthesizer that combines a physical model with a performance model. The goal of the performance model is to generate control information for the physical model by means of a collection of rules manually extracted from the analysis of a collection of controlled recordings of human performance.

Another approach taken for performing tempo and dynamics transformation is the use of neural network techniques. In Bresin (1998), a system that combines symbolic decision rules with neural networks is implemented for simulating the style of real piano performers. The output of the neural networks express time and loudness deviations. These neural networks extend the standard feed-forward network trained with the back-propagation algorithm with feedback connections from the output neurons to the input neurons.

We can see that except for the work of the KTH group that considers three expressive resources, the other systems are limited to two resources such as rubato and dynamics or rubato and articulation. This limitation has to do with the use of rules. Indeed, the main problem with the rule-based approaches is that it is difficult to find rules general enough to capture the variety present in different performances of the same piece by the same musician and even the variety within a single performance (Kendall and Carterette 1990). Furthermore, the different expressive resources interact with each other. That is, the rules for dynamics alone change when rubato is also taken into account. Obviously, because of this interdependency, the more expressive resources one tries to model, the more difficult it is to find the appropriate rules.

As mentioned in the introduction, in the SaxEx system, we directly use the interpretation knowledge implicit in examples from monophonic recordings of human performers instead of try to make this knowledge explicit by means of rules. This approach allows us to deal with the five most important expressive resources: (1) dynamics, (2) rubato, (3) vibrato, (4) articulation, and (5) attack of the notes. Furthermore, we do it in the context of an expressively richer (non-MIDI) instrument such as a tenor saxophone. Besides, ours was the first attempt to apply CBR to music as well as the first attempt to cover the full cycle from a non-MIDI inexpressive input sound file to an expressive output sound file. Working directly with sound files is much more complex than working with MIDI because sound files require automatically extracting the higher-level parameters that are relevant for expressiveness. Besides, it also complicates the expressive transformation process and the synthesis. However, with our approach, we achieve expressive performances that are much closer to human performances. The remaining sections describe our system.

SaxEx: A Case-Based Reasoning Approach to Expressive Performance

The task of SaxEx is to infer a set of expressive transformations to apply to every note of an inexpressive monophonic input so that the output sounds expressive. In other words, what SaxEx does is add expressiveness to a given flat performance. To do so, SaxEx uses a case memory (figure 1) containing examples of expressive human performances, analyzed by means of spectral modeling techniques (Serra et al. 1997), and musical knowledge. SaxEx also knows the score of the performance. The heart of the method is to analyze each input note determining its "role" in the musical phrase it belongs to, identify and retrieve notes with similar roles in the expressive cases, and transform the input note so that its expressive properties (dynamics, vibrato, rubato, articulation, and attack of the notes) match the properties of the retrieved similar notes.

Two types of analysis are performed on the expressive human performances. First, using spectral modeling techniques (Serra et al. 1997), the SMS subsystem computes, for each of the five expressive resources, a qualitative value representing how each note, in the expres-

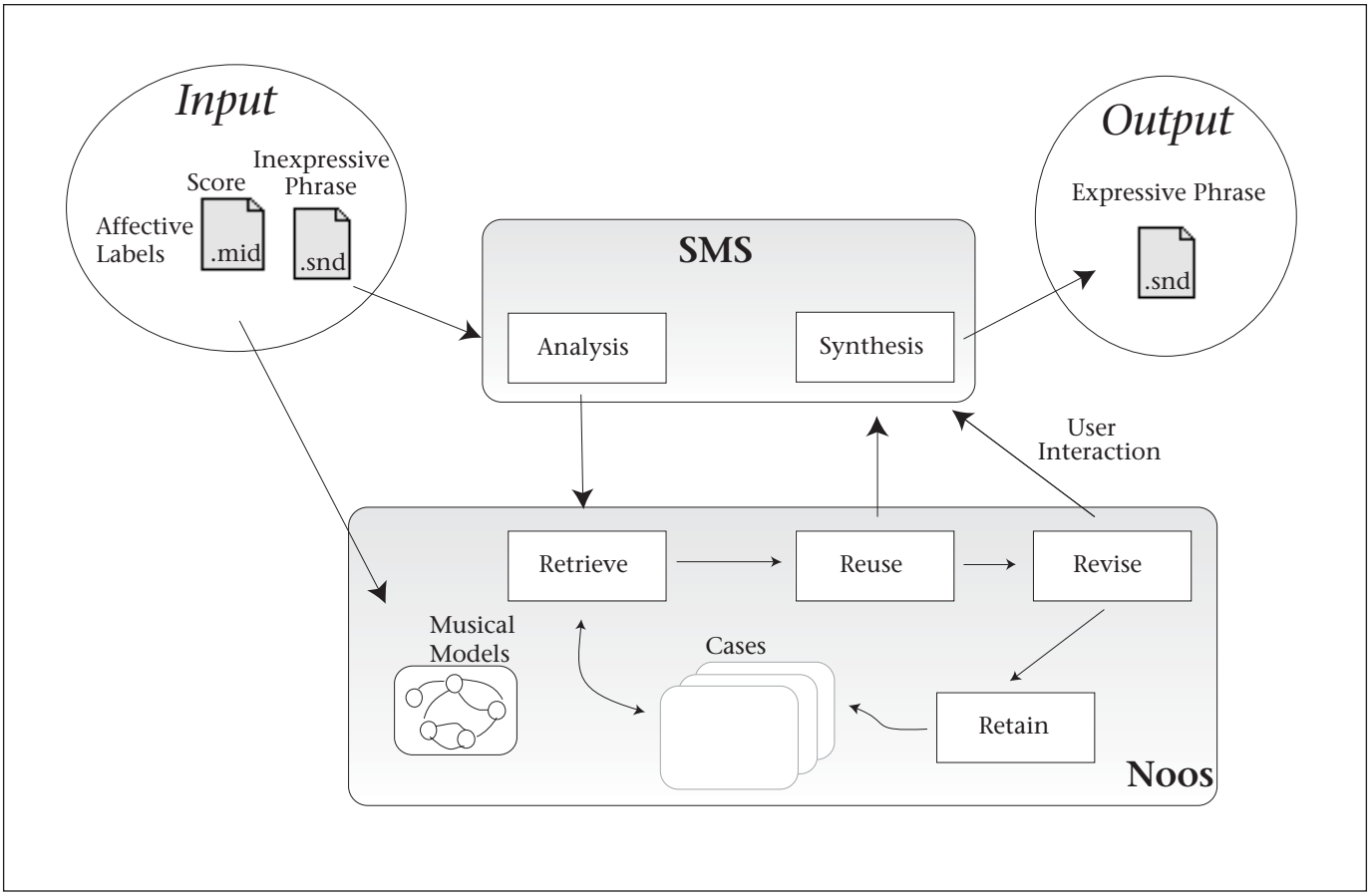


Figure 1. SAXEX Schema.

sive examples, has been played by the human performer. This value is stored with each note in the case memory and is available to SAXEX. The possible qualitative values are very low, low, medium, high, and very high. To compute them, SMS computes the average quantitative value of the whole performed piece for each one of the expressive resources and then compares how each individual note has been played in relation with these averages. For example, if the dynamics (in decibels) of a note is much higher than the average dynamics of the piece, then the symbolic value "very high dynamics" is associated with this note. The same occurs for the other values and the other expressive resources.

The second analysis applies the generative theory of tonal music (Lerdahl and Jackendoff 1983) and Narmour´s (1990) music cognition model to determine the role of each note in the musical phrase. For example, a note might be the last of an ascending melodic progression, have long duration, be metrically strong, be harmonically stable, and be the most prominent note in the bar.

As a result of these two analyses, each note in the stored expressive performance is annotated with its role in the musical phrase it belongs to and its qualitative expressive values. These annotations are represented in the cases of the system. It is worth clarifying here that the GENERATIVE THEORY OF TONAL MUSIC (GTTM) and Narmour's analyses introduce important context information on the notes (because the role of each note in the phrase depends on its relation to the remaining phrase notes as well as on the chord); therefore, SAXEX cases do not contain information only on single notes but include contextual knowledge at the phrase level.

To generate the expressive performance, SAXEX applies the second analysis to the inexpressive input performance to identify the role of each note. It then searches for *similar notes* (that is, notes having similar roles within the musical phrase) in the stored expressive performances. When a similar note is selected, SAXEX applies the set of expressive values that will make the inexpressive note sound like the retrieved expressive one. The rationale behind this way of proceeding is that similar expressive nuances (similar solutions in CBR terms) should be applied to similar musical phrases

Narmour's Implication-Realization Model

An intuition shared by many people is that appreciating music has to do with expectation. That is, what we have already heard builds expectations on what is to come. These expectations can be fulfilled or not by what is to come. If fulfilled, the listener feels satisfied. If not, the listener is surprised or even disappointed. Based on this observation, Narmour proposed a theory of cognition of melodies based on a set of basic grouping structures (figure A). These structures characterize patterns of melodic implications (or expectations) that constitute the basic units of the listener's perception. Other resources such as duration and rhythmic patterns emphasize or inhibit the perception of these melodic implications. The use of the implication-realization model provides a musical analysis of the melodic surface of the piece. The basic grouping structures are (figure A) the *P structure* (a pattern composed of a sequence of at least three notes with similar intervalic distances and the same registral direction), the *ID structure* (a sequence of three notes with the same intervalic difference and different registral direction), the *D structure* (a repetition of at least three notes), the *IP structure* (a sequence of three notes with similar intervalic distances and different registral direction), the *VP structure* (a sequence of three notes with the same registral direction; the first intervalic distance is a step, and the second is a leap), the *R structure* (a sequence of three notes with a different registral direction; the first intervalic distance is a leap, and the second is a step), the *IR structure* (a sequence of three notes with the same registral direction; the first intervalic distance is a leap, and the second is a step), and the *VR structure* (a sequence of three notes with different registral direction; both intervals are leaps). In figure B, the first three notes form a P structure, the next three notes an ID, and the last three notes another P. The two P structures in the figure have a descending registral direction, and in both cases, there is durational cumulation (the last note is significantly longer).

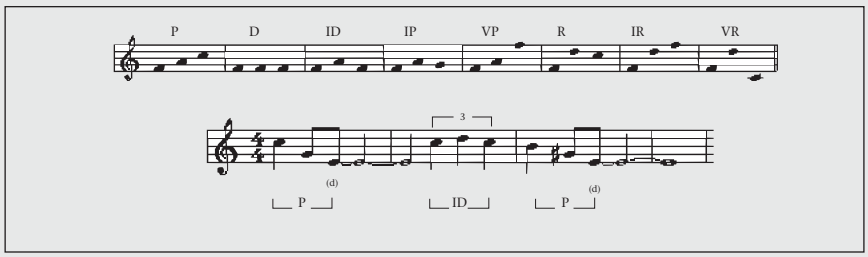


Figure A (top). Narmour's Basic Structures.

Figure B (bottom). Narmour's Parsing of the Beginning of "All of Me."

Looking at melodic groupings in this way, we can see how each pitch interval implies the next. Thus, an interval can be continued with a similar one (such as P or ID or IP or VR) or reversed with a dissimilar one. That is, a *step* (small interval between notes) followed by a *leap* (large interval between notes) in the same direction would be a reversal of the implied interval (another step was expected, but instead, a leap is heard) but not a reversal of direction. Pitch motion can also be continued by moving in the same direction (up or down) or reversed by moving in the opposite direction. The strongest kind of reversal involves both a reversal of interval and a reversal of direction. When several small intervals (steps) move consistently in the same direction, they strongly imply continuation in the same direction with similar small intervals. If a leap occurs instead of a step, it creates a continuity gap, which triggers the expectation that the gap should be filled in. To fill it, the next step intervals should move in the opposite direction from the leap, which also tends to limit pitch range and keeps melodies moving back toward a center. Basically, *continuity* (satisfying the expectations) is nonclosural and progressive, whereas *reversal of implication* (not satisfying the expectation) is closural and segmentive. A long note duration after reversal of implication usually confirms phrase closure.

Any given melody can be described by a sequence of Narmour structures. SaxEx performs an automatic parsing of the melody to extract the Narmour structures (figure B). Then, SaxEx uses these structures to search and retrieve notes in the case memory that are similar to the input inexpressive notes. To do this retrieval, it takes into account the kind of structure it belongs to and its position within the structure (first note, inner note, or last note). To select the most preferred notes among the retrieved ones, it uses the registral direction (notes having the same registral direction as the input note will be preferred), and the durational cumulation (if the input note has a long duration and is the last note of the structure, retrieved notes having this properties will be the preferred ones). The rationale is that two notes that share these Narmour properties should be played similarly. This rationale agrees with the main case-based reasoning assumption that similar problems have similar solutions.

(similar problems in CBR terms). For example, if the dynamics of the expressive note is high, the rubato is low, the vibrato is very high, the articulation is medium, and the attack is very noisy, SaxEx will apply these same values to the inexpressive input note. These expressive values are then given to the SMS synthesizer, which generates the output sound file (see footnote 2) by transforming the inexpressive sound file given as input.

The SMS Subsystem

Sound analysis and synthesis techniques based on spectral models are useful for extracting high-level parameters from sound files, transforming them and synthesizing a modified version of such sound files. Our system uses the SMS subsystem (figure 1) to extract key information related to the five expressive resources we deal with. Furthermore, we use the SMS synthesis component to generate the expressive interpretations inferred by the CBR subsystem. SMS is based on decomposing the original sound into sinusoids plus a spectral residual (noise). From the sinusoidal representation, SMS extracts high-level attributes such as attack and release times of the notes, formant structure, vibrato, and average pitch and amplitude when the sound is monophonic. These attributes can be modified according to the expressive values inferred by the CBR subsystem (see next section), and the result can be added back to the spectral representation without loss of sound quality. In summary, this sound analysis and synthesis software is ideal as a preprocessor that provides to the CBR subsystem a high-level musical description of the sound files and as a postprocessor that adds the expressive transformations inferred by the CBR subsystem to the inexpressive input sound file.

The CBR Subsystem

Solving a problem in the SaxEx system involves three parts: (1) the analysis, (2) the reasoning, and (3) the synthesis. The analysis and the synthesis are done using the SMS subsystem as described earlier. The reasoning part is achieved by means of the CBR subsystem (figure 1) and consists of inferring a set of expressive transformations to apply to every note of an inexpressive input so that the output sounds expressive. To do this inference, the CBR subsystem has to be able to retrieve, from a memory containing expressive interpretations, those notes that are, musically speaking, similar to the input inexpressive notes. The retrieved notes are candidates to be imitated. The next four subsections describe the necessary CBR steps performed by SaxEx.

The Retrieval Step

From the overall description section, we can see that the key problem for SaxEx is finding the appropriate note to imitate or, in other words, to figure out when two notes belonging to two different musical pieces play a similar role in their respective musical phrases. When musicians compare two music scores, they use their musical knowledge to solve this problem. For example, they take into account the following five elements:

First is the *metrical strength* of the notes (the metrical strength of notes played on strong beats is higher than the metrical strength of notes played on weak beats).

Second is the *harmonic stability* of the notes according to jazz harmony (position of the note within its underlying chord and the role of the note in the chord progression),

Third is the *notes duration*.

Fourth is the *hierarchical relations* of the note with respect to the rest of the notes in the phrase according to the GTTM (Lerdahl and Jackendoff 1983). For example, some notes elaborate other notes (some notes are essential, and others are ornamental), some notes create tensing and relaxing relations with respect to others (some notes give a feeling of finality or settledness; others feel unstable, and when they are played, the listener feels a tension that is resolved when the piece returns to a more stable note). (see sidebar 2 for further details).

Fifth is the *position of notes* in Narmour's (1990) basic musical grouping structures, such as melodic progressions, repetitions, and changes in registral directions. Narmour, in his implication-realization model of cognition of melodies, identified a small number of basic structures that constitute the basic units of the listener's perception (see sidebar 1 for further details).

SaxEx represents, in its case memory, all this knowledge about metrical strength, harmonic stability, note duration, hierarchical relations, Narmour structures, and so on, by means of the NOOS object-oriented language (Arcos 1997) and uses this knowledge to retrieve the appropriate notes to imitate in the case memory of expressive notes. That is, SaxEx retrieves notes that are musically similar to the inexpressive input notes. For example, assume that the inexpressive input note is the last of a melodic progression, is metrically strong, and is harmonically stable. Then, SaxEx searches for notes with similar musical properties in the case memory because notes with these properties would clearly be good candidates to imitate. Assume for the moment that only one similar note is found. In this simple situation, this note would immediately be retrieved for imitation. In other words,

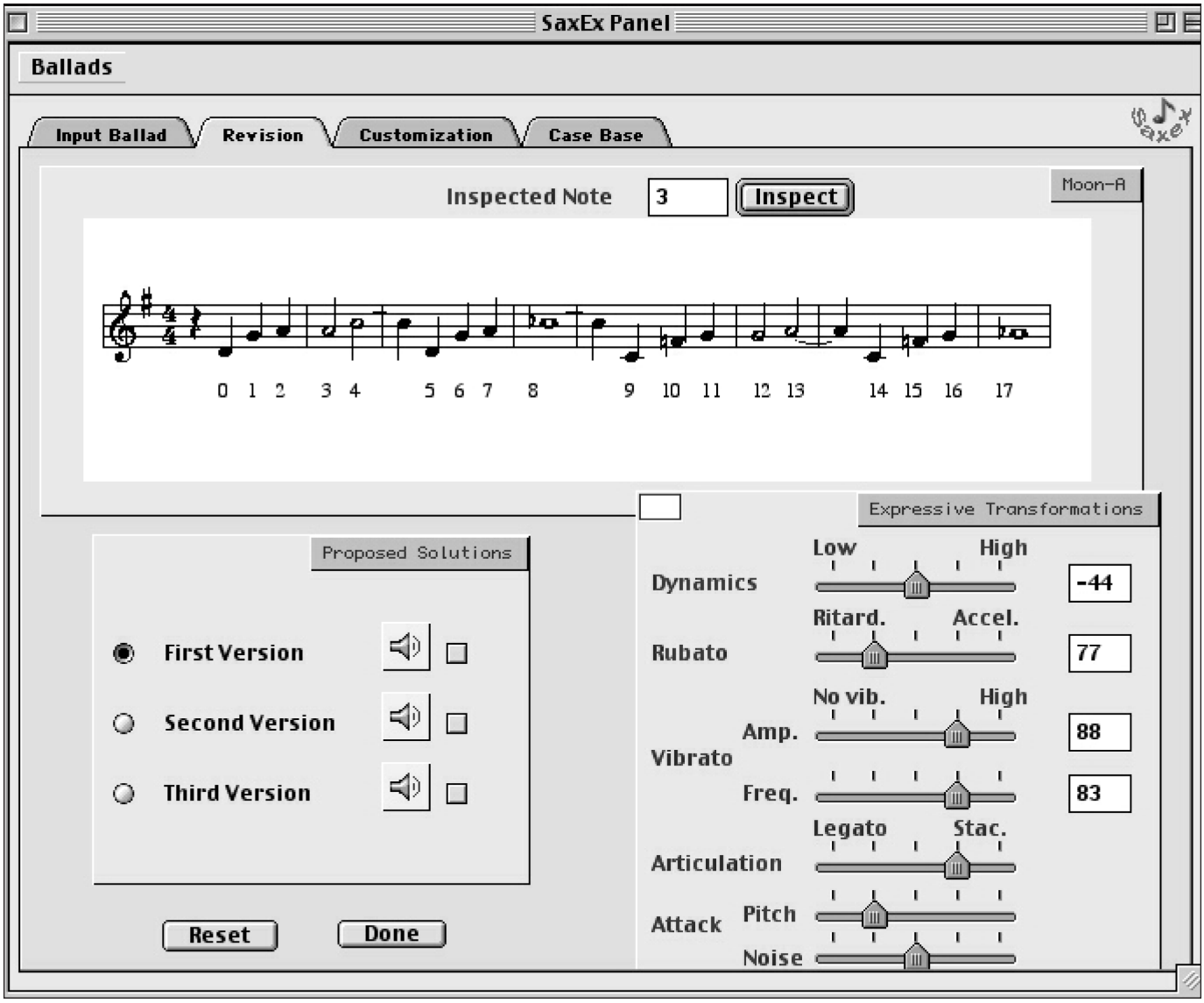


Figure 2. Interactive Revision Window.

SaxEx would decide to apply, to the inexpressive input note, the set of expressive qualitative values that will make that note sound like the retrieved expressive one. For example, if the dynamics of the retrieved expressive note is high, the rubato is low, the vibrato is very high, the attack is very noisy, and the articulation is medium, SaxEx will apply these values to the inexpressive input note.

When several similar notes are found, an additional step is carried out to select the most preferred one using additional musical criteria such as the registral direction (ascending or descending) and the duration of the note (see sidebars for further details).

The Reuse Step

In this step, the selected expressive note is imitated; that is, SaxEx decides to apply expressive transformations to make the inexpressive input note sound like the selected expressive one. In the case where the preference criteria cannot single out a note, and therefore, several are equally appropriate according to all the available musical criteria, a fuzzy aggregation operator similar to a weighted average (Arcos and Lopez de Mantaras 2002) is applied to compute a set of expressive values that combines all the expressive values of the selected notes. The user can also preselect one of the following alternatives to replace the aggregation operator:

First, the *majority rule* chooses, for each expressive resource, the value that was applied in the majority of the selected expressive notes.

Second, the *strict majority rule* chooses the

Lerdahl and Jackendoff's Generative Theory of Tonal Music

According to the GENERATIVE THEORY OF TONAL MUSIC (GTTM), music is built from an inventory of notes and a set of rules. The rules assemble notes into a sequence and organize them into hierarchical structures of music cognition. To understand a piece means to assemble these mental structures as we listen to the piece. It proposes four types of structures associated with a piece: (1) grouping, (2) metric, (3) time-span reduction, and (4) prolongational reduction. The *grouping structure* describes the hierarchically related segmentation units that listeners can establish when hearing a melody: motifs, phrases, and sections. The listener feels that notes group into motifs, which, in turn, are grouped into phrases, which are grouped into sections, which are finally grouped into full pieces. The metric structure describes the rhythmical hierarchy of the piece. It assigns a weight to each note depending on the beat in which it is played in such a way that the metric strength of notes played on strong (down) beats is higher than the metric strength (accent) of notes played on weak (up) beats. In figure A, the metric structure is illustrated by the dots under the notes: the more dots, the stronger the accent on that note. The *time-span reduction* is a hierarchical structure describing the relative structural importance of notes within the audible rhythmic units of a phrase (figure A). It differentiates in the melody the essential parts from the ornaments. The essential parts are further dissected into even more essential parts and ornaments on them. The reduction continues until the melody is reduced to a skeleton of the few most prominent notes. There are two types of nodes in this hierarchical structure: (1) left elaboration and (2) right elaboration. For example, in figure A, the first quarter note of the second bar (C) belongs to a left leaf in a right-elaboration node because the following two notes (D and C) elaborate the first note. In turn, these two notes belong to a left-elaboration node because the second note (D) elaborates the third (C). In figure A, we can see then that the C of the first bar is the most prominent note; the next most prominent notes are the first C of the second bar and the B of the third bar, and so on. The prolongational reduction is a hierarchical structure describing tension-relaxation relations among notes. There are two basic types of nodes in this structure: (1) tensing and (2) relaxing. Besides, there are three possible modes of branch chaining: (1) *strong prolongation*, in which events repeat maintaining sonority (for example, notes of the same chord such as the notes of the first and third measures in figure A); (2) *weak prolongation,* in which events repeat in an altered form (for example, from an I chord to a I6 Chord); (3) jumps in which two completely different events are connected (for example from an I chord to a dominant chord such as the jump from the Cmaj7 chord of the second measure to the E7 chord of the third measure in figure A). This structure captures the sense of musical flow across phrases, that is, the buildup and release of tension within longer and longer passages of the piece, until a feeling of maximum repose at the end of the piece. Tension builds up as the melody departs from more stable notes to less stable ones and is discharged when the melody returns to stable notes. For example, in figure A, the D of the second bar creates a tension (a D is not a note of the underlying Cmaj7 chord) that is immediately resolved in the next note (C). Tension and release are also felt as a result of moving from dissonant to consonant chords, from nonaccented to accented notes, from higher to lower notes, and from prolonged to nonprolonged notes. SAXEX uses this model to search and retrieve notes from the case memory whose metrical strength is similar to the input inexpressive note. However, the role (prominent note, ornamental note, tensing note, relaxing note) of the retrieved notes in the time-span and prolongational reduction trees are used by SAXEX as preference criteria to prefer some notes and discard others. Those notes whose roles are similar to the roles of the input note will be the preferred ones. Again, the rationale is that similar notes (in the sense of the GTTM theory) should be played similarly.

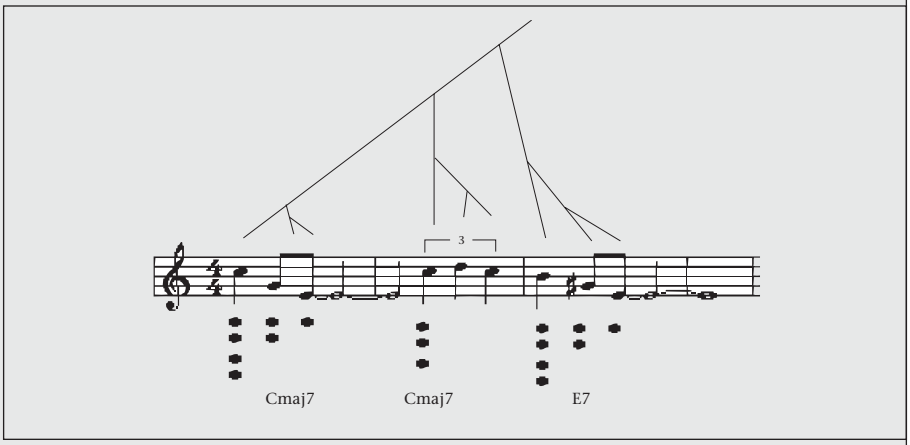


Figure A. Metrical Strength and Time-Span Tree of the Beginning of "All of Me."

value that was applied in at least half of the selected expressive notes.

Third, is the *minority rule* (the dual of the first criterion).

Fourth, the *strict minority rule* chooses the value that was applied in, at most, one of the selected expressive notes.

Fifth, *continuity* selects the values of the note that comes from the same musical subphrase as the previously selected note.

Sixth is *discontinuity* (the dual of the fifth criterion).

If, after applying any of these criteria, more than one note remains, a random choice takes place.

From this description, it should be noticed that the default strategy of the system is to apply the fuzzy aggregation operator. Nevertheless, because the generation of expressive performances is a creative process influenced by the user's personal preferences, the alternative *reuse* criterion can be used to tailor the system according to such personal preferences. For example, the strict minority and the discontinuity criteria would force the system to produce expressive performances containing less usual combinations of expressive effects.

The Revise Step

The user can listen to the obtained results and, through an interactive revision window (figure 2), has the option to select any note and manually modify its expressive values (decreasing vibrato, increasing dynamics, and so on) if he/she thinks that some of the solutions decided by the system should be corrected. Additionally, if several expressive versions of the same piece have been generated, the user can take the best partial solutions of each one of them to obtain a combined best performance, for example, by taking the first phrase of the second version followed by the second phrase of the first version, and so on.

The Retain Step

The performance resulting from the revision step is automatically added to the case memory. This added performance is therefore available for retrieval when solving future problems. It is worth noticing that only positive feedback is given to the system, that is, only expressive performances that have been judged as good ones or that have been improved at the revision step, are retained. This step provides the (knowledge-intensive) learning dimension to the system.

The Use of Affective Labels

Music clearly evokes emotions. This phenomenon has been studied quite extensively both from the psychological point of view (Davis and Thaut 1989; Dowling and Harwood 1986; Goldstein 1980; Krumhansl 1997; Robazza, Macaluso, and D'Urso 1994; Sloboda 1991) and, more recently, from the neurological point of view (Boold et al. 1999). SaxEx also gives the user the possibility to bias the results along several affective dimensions. Thus, the expressive performances in the case memory also include information related to their affective content. This information is represented by a sequence of *affective regions*. Affective regions group sequences of notes with common affective expressiveness along three dimensions: (1) tender-aggressive, (2) sad-joyful, and (3) calm-restless. These dimensions are graded by means of five ordered qualitative values expressed by means of linguistic labels. The middle label represents no predominance (for example, neither tender nor aggressive in the tender-aggressive dimension); lower and upper labels represent, respectively, predominance in one direction or the other (for example, very calm is the lowest label in the calm-restless dimension). These affective regions were manually labeled. The user can bias the performances synthesized by SaxEx by means of indicating an affective label at the retrieval step. For example, if the user declares that he/she wants a calm and very tender performance, SaxEx will first look for notes in the case memory that belong to calm and very tender affective regions (most preferred). If none is found, it will look next for notes belonging to calm and tender or very calm and very tender regions (both as second choices because these are the next-closest options). Obviously, the selected notes will also have to be similar to the input nonexpressive note, according to the criteria based on the musical knowledge explained previously. In the sound examples (see footnote 2), we have included two performances with different affective values.

Experimentation

We limited our experiments to monophonic tenor saxophone interpretations of standard jazz ballads (such as "All of Me," "Autumn Leaves," and "Misty"). The expressive examples, as well as the flat performances for each ballad, were recorded in the audio recording studio of the Audiovisual Institute in Barcelona by a tenor saxophone player.

We have performed two sets of experiments combining the different jazz ballads that were recorded. The first set of experiments consisted of using the expressive examples from the first phrases of a given ballad for generating the

expressive version of the remaining phrases. This group of experiments revealed that SaxEx clearly identified the relevant precedent cases to be imitated, and appropriate expressive transformations were applied.

The second set of experiments consisted of using examples of expressive performances of some pieces to generate expressive performances of other pieces. This second group of experiments revealed that SaxEx also identified relevant precedent cases to imitate, and appropriate expressive transformations were applied. One can notice, for example, very appropriate crescendos in increasing melodic progressions, low-frequency vibratos in long notes in sad performances and higher-frequency vibratos in joyful ones, accentuated legato in sad performances and staccato in joyful ones, "swingy" rhythmic variations, and attacks from a lower pitch in the sad interpretations versus noisy attacks in the joyful ones. These expressive resources are often applied in the same way by human performers. We also received additional feedback from dozens of individuals, musicians, and nonmusicians who accessed the sound examples posted on the web and judged the results as humanlike. Indeed, the "playing style" of SaxEx very much resembles the style of the human saxophonist that provided the expressive examples. Readers are encouraged to judge the quality of SaxEx performances by accessing our demos (see footnote 2) and giving us additional feedback.

Concluding Remarks

In the first part of this article, we presented a survey of representative computer music systems that, to a greater or lesser extent, use AI techniques.[3] The survey covered the three major types of systems: (1) compositional, (2) improvisational, and (3) performance. In the second part, we emphasized the importance of dealing with the problem of generating expressive performances and described SaxEx, a computer system based on CBR techniques that is capable of performing expressive, monophonic music that resembles human performance. CBR has proven to be a very powerful technique to directly use the interpretation knowledge implicit in examples extracted from recordings of human performers rather than try to make this knowledge explicit. Although at present our system is limited to the style of jazz ballads, this is not a strong limitation because with not much effort, our system could generate expressive performances in other styles provided that we introduce the pertinent cases in the case base. As a matter of fact, jazz ballads, because of their slow tempo and the importance of the melody, are more sensitive than other jazz styles to the inadequate use of expressive resources. We believe that performing expressively in styles such as be-bop, or hard-bop, would actually be easier. It possibly would also be easier to deal with popular or folk music because it is generally structurally simpler than jazz. The bigger problems would be with classical music because it would not be enough to have the appropriate example cases. Indeed, we would also need to change the background musical harmonization knowledge of the system.

Although at present SaxEx works with tenor sax sound, it would not be difficult to change to other instruments. We work with a sax because it is a very rich instrument in terms of expressiveness. Many other instruments would actually be easier to deal with. Finally, the problem of dealing with polyphonic music is that of source separation, that is, isolating each instrument (and, particularly, the soloist) from the rest. This problem has not yet been solved. Therefore, unless each instrument is recorded in a different track, we cannot yet deal with polyphonic music.

Our agenda for the near future includes, among other minor improvements, developing a real-time version as well as adding improvisation capabilities because the current version is restricted to perform the notes that are written in the score; that is, it cannot change, add, or remove notes. We believe that truly humanlike performance has to cope, among other things, with at least some basic improvisation resources, such as not playing some notes and adding ornamental notes. We intend to build on existing work on improvisation, in particular on the approach being developed within our group by Grachten (2001) that is based on a cognitive model for jazz improvisation (Pressing 1988).

Acknowledgments

We thank David Leake and the anonymous reviewers for their helpful and stimulating suggestions to improve the article. Our research on AI and music is partially supported by the Spanish Ministry of Science and Technology under project TIC 2000-1094-C02 "TABASCO: Content-Based Audio Transformation Using CBR."

Notes

1. We received the best paper award for an early version of SaxEx at the 1997 International Computer Music Conference.

2. Listen to some examples at www.iiia.csic.es/~arcos/noos/Demos/Aff-Example.html and www. iiia.csic.es/

~arcos/noos/Demos/Example.html. Please make sure your browser has a plug in to process wave sound files.

3. Readers can also find information on AI and music at the American Association of Artificial Intelligence AI Topics web site www.aaai.org/AITopics/html/music.html.

References

Aamodt, A., and Plaza, E. 1994. Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. *AI Communications* 7(1): 39–59.

Arcos, J. L. 1997. The NOOS Representation Language. Ph.D. dissertation, Computer Science Department, Universitat Politècnica de Catalunya.

Arcos, J. L., and Lopez de Mantaras, R. 2002. Combining Fuzzy and Case-Based Reasoning to Generate Humanlike Music Performances. In Proceedings of the Information Processing and Management of Uncertainty Congress (IPMU-2000). Studies in Fuzziness and Soft Computing, Volume 89, 21–31. New York: Springer-Verlag. Forthcoming.

Arcos, J. L., and Lopez de Mantaras, R. 2001. An Interactive Case-Based Reasoning Approach for Generating Expressive Music. *Applied Intelligence* 14(1): 115–129.

Arcos, J. L.; Lopez de Mantaras, R.; and Serra, X. 1998. SAXEX: A Case-Based Reasoning System for Generating Expressive Musical Performances. *Journal of New Music Research* 27(3): 194–210.

Baars, B. 1998. *A Cognitive Theory of Consciousness*. New York: Cambridge University Press.

Bharucha, J. 1993. MUSACT: A Connectionist Model of Musical Harmony. In *Machine Models of Music,* eds. S. M. Schwanauer and D. A. Levitt, 497–509. Cambridge, Mass.: The MIT Press.

Biles, J. A. 1994. GENJAM: A Genetic Algorithm for Generating Jazz Solos. In Proceedings of the 1994 International Computer Music Conference, 131–137. San Francisco, Calif.: International Computer Music Association.

Boold, A. J.; Zatorre, R. J.; Bermudez, P.; and Evans, A. C. 1999. Emotional Responses to Pleasant and Unpleasant Music Correlate with Activity in Paralimbic Brain Regions. *Nature Neuroscience* 2(4): 382–387.

Bresin, R. 2002. Articulation Rules for Automatic Music Performance. In Proceedings of the 2001 International Computer Music Conference 2001. San Francisco, Calif.: International Computer Music Association. Forthcoming.

Bresin, R. 1998. Artificial Neural Networks–Based Models for Automatic Performance of Musical Scores. *Journal of New Music Research* 27(3): 239–270.

Canazza, S.; De Poli, G.; Roda, A.; and Vidolin, A. 1997. Analysis and Synthesis of Expressive Intention in a Clarinet Performance. In Proceedings of the 1997 International Computer Music Conference, 113–120. San Francisco, Calif.: International Computer Music Association.

Cope, D. 1990. Pattern Matching as an Engine for the Computer Simulation of Musical Style. In Proceedings of the 1990 International Computer Music Conference, 288–291. San Francisco, Calif.: International Computer Music Association.

Cope, D. 1987. Experiments in Music Intelligence. In Proceedings of the 1987 International Computer Music Conference, 174–181. San Francisco, Calif.: International Computer Music Association.

Dannenberg, R. B. 1993. Software Design for Interactive Multimedia Performance. *Interface* 22(3): 213–218.

Dannenberg, R. B., and Derenyi, I. 1998. Combining Instrument and Performance Models for High-Quality Music Synthesis. *Journal of New Music Research* 27(3): 211–238.

Davis, W. B., and Thaut, M. H. 1989. The Influence of Preferred Relaxing Music on Measures of State Anxiety, Relaxation, and Psychological Responses. *Journal of Music Theory* 26(4): 168–187.

Dowling, W. J., and Harwood, D. L. 1986. *Music Cognition*. San Diego, Calif.: Academic.

Ebcioglu, K. 1993. An Expert System for Harmonizing Four-Part Chorales. In *Machine Models of Music,* eds. S. M. Schwanauer and D. A. Levitt, 385–401. Cambridge, Mass.: MIT Press.

Feulner, J. 1993. Neural Networks That Learn and Reproduce Various Styles of Harmonization. In Proceedings of the 1993 International Computer Music Conference, 236–239. San Francisco, Calif.: International Computer Music Association.

Franklin, J. A. 2001. Multi-Phase Learning for Jazz Improvisation and Interaction. Paper presented at the Eighth Biennial Symposium on Art and Technology, 1–3 March, New London, Connecticut.

Friberg, A. 1995. A Quantitative Rule System for Musical Performance. Ph.D. dissertation, Department of Speech, Music, and Hearing, Royal Institute of Technology.

Friberg, A.; Bresin, R.; Fryden, L.; and Sunberg, J. 1998. Musical Punctuation on the Microlevel: Automatic Identification and Performance of Small Melodic Units. *Journal of New Music Research* 27(3): 271–292.

Friberg, A.; Sunberg, J.; and Fryden, L. 2000. Music from Motion: Sound Level Envelopes of Tones Expressing Human Locomotion. *Journal on New Music Research* 29(3): 199–210.

Fry, C. 1993. FLAVORS BAND: A Language for Specifying Musical Style. In *Machine Models of Music,* eds. S. M. Schwanauer and D. A. Levitt, 427–451. Cambridge, Mass.: The MIT Press.

Goldstein, A. 1980. Thrills in Response to Music and Other Stimuli. *Physiological Psychology* 8(1): 126–129.

Grachten, M. 2001. JIG: Jazz Improvisation Generator. In *Proceedings of the MOSART Workshop on Current Research Directions in Computer Music,* 1–6. Barcelona, Spain: Pompeu Fabra University Publishers.

Hiller, L., and Isaacson, L. 1993. Musical Composition with a High-Speed Digital Computer. In *Machine Models of Music,* eds. S. M. Schwanauer and D. A. Levitt, 9–21. Cambridge, Mass.: MIT Press.

Hörnel, D., and Degenhardt, P. 1997. A Neural Organist Improvising Baroque-Style Melodic Varia-

tions. In Proceedings of the 1997 International Computer Music Conference, 430–433. San Francisco: International Computer Music Association.

Hörnel, D., and Menzel, W. 1998. Learning Musical Structure and Style with Neural Networks. *Journal on New Music Research* 22(4): 44–62.

Johnson, M. L. 1992. An Expert System for the Articulation of Bach Fugue Melodies. In Readings in Computer-Generated Music, ed. D. L. Baggi, 41–51. Washington, D.C.: IEEE Computer Society.

Johnson-Laird, P. N. 1991. Jazz Improvisation: A Theory at the Computational Level. In *Representing Musical Structure,* eds. P. Howell, R. West, and I. Cross. San Diego, Calif.: Academic.

Kendall, R. A., and Carterette, E. C. 1990. The Communication of Musical Expression. *Music Perception* 8(2): 129.

Kolodner, J. 1993. *Case-Based Reasoning*. San Francisco, Calif.: Morgan Kaufmann.

Krumhansl, C. L. 1997. An Exploratory Study of Musical Emotions and Psychophysiology. *Canadian Journal of Experimental Psychology* 51(4): 336-352.

Leake, D. 1996. *Case-Based Reasoning: Experiences, Lessons, and Future Directions*. Menlo Park, Calif.: AAAI Press.

Lerdahl, F., and Jackendoff, R. 1983. An Overview of Hierarchical Structure in Music. *Music Perception* 1(2): 229–252.

Levitt, D. A. 1993. A Representation for Musical Dialects. In *Machine Models of Music,* eds. S. M. Schwanauer and D. A. Levitt, 455–469. Cambridge, Mass.: The MIT Press.

Minsky, M. 1993. Music, Mind, and Meaning. In *Machine Models of Music,* eds. S. M. Schwanauer and D. A. Levitt, 327–354. Cambridge, Mass.: MIT Press.

Moorer, J. A. 1993. Music and Computer Composition. In *Machine Models of Music,* eds. S. M. Schwanauer and D. A. Levitt, 167–186. Cambridge, Mass.: The MIT Press.

Morales-Manzanares, R.; Morales, E. F.; Danenberg, R.; and Berger, J. 2001. SICIB: An Interactive Music Composition System Using Body Movements. *Journal of New Music Research* 25(2): 25–36.

Narmour, E. 1990. *The Analysis and Cognition of Basic Melodic Structures: The Implication-Realization Model*. Chicago: University of Chicago Press.

Pachet, F., and Roy, P. 1998. Formulating Constraint-Satisfaction Problems on Part-Whole Relations: The Case of Automatic Harmonization. Paper presented at the ECAI'98 Workshop on Constraint Techniques for Artistic Application, 23–28 August, Brighton, UK.

Papadopoulos, G., and Wiggins, G. 1998. A Genetic Algorithm for the Generation of Jazz Melodies. Paper presented at the Finnish Conference on Artificial Intelligence (SteP'98), 7–9 September, Jyvaskyla, Finland.

Pressing, J. 1988. Improvisation: Methods and Models. In *Generative Processes in Music: The Psychology of Performance, Improvisation, and Composition,* ed. J. Sloboda, 129–178. Oxford, U.K.: Oxford University Press.

Rader, G. M. 1993. A Method for Composing Simple Traditional Music by Computer. In *Machine Models of Music,* eds. S. M. Schwanauer and D. A. Levitt, 243–260. Cambridge, Mass.: The MIT Press.

Robazza, C.; Macaluso, C.; and D'Urso, V. 1994. Emotional Reactions to Music by Gender, Age, and Expertise. *Perceptual Motor Skills* 79:939–944.

Rothgeb, J. 1993. Simulating Musical Skills by Digital Computer. In *Machine Models of Music,* eds. S. M. Schwanauer and D. A. Levitt, 157–164. Cambridge, Mass.: The MIT Press.

Sabater, J.; Arcos, J. L.; and López de Mántaras R. 1998. Using Rules to Support Case-Based Reasoning for Harmonizing Melodies. Paper presented at the AAAI Spring Symposium on Multimodal Reasoning, 27 July, Stanford, California.

Serra,, X.; Bonada, J.; Herrera, P.; and Loureiro, R. 1997. Integrating Complementary Spectral Methods in the Design of a Musical Synthesizer. In Proceedings of the 1997 International Computer Music Conference, 152–159. San Francisco, Calif.: International Computer Music Association.

Simon, H. A., and Sumner, R. K. 1993. Patterns in Music. In *Machine Models of Music,* eds. S. M. Schwanauer and D. A. Levitt, 83–110. Cambridge, Mass.: MIT Press.

Sloboda, J. A. 1991. Music Structure and Emotional Response: Some Empirical Findings. *Psychology of Music* 19(2): 110–120.

Schwanauer, S. M. 1993. A Learning Machine for Tonal Composition. In *Machine Models of Music,* eds. S. M. Schwanauer and D. A. Levitt, 511–532. Cambridge, Mass.: MIT Press.

Schwanauer, S. M., and Levitt, D. A., eds. 1993. *Machine Models of Music*. Cambridge, Mass.: The MIT Press.

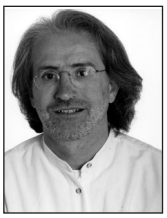
Thom, B. 2001. BoB: An Improvisational Music Companion. Ph.D. dissertation, School of Computer Science, Carnegie-Mellon University.

Wessel, D.; Wright, M.; and Kahn, S. A. 1998. Preparation for Improvised Performance in Collaboration with a Khyal Singer. In Proceedings of the 1998 International Computer Music Conference, 497–503. San Francisco, Calif.: International Computer Music Association.

Widmer, G. 2001. The Musical Expression Project: A Challenge for Machine Learning and Knowledge Discovery. In Proceedings of the Twelfth European Conference on Machine Learning, 603–614. Lecture Notes in Artificial Intelligence 2167. New York: Springer-Verlag.

Widmer, G. 1996. Learning Expressive Performance: The Structure-Level Approach. *Journal of New Music Research* 25(2): 179–-205.

Woods, W. 1970. Transition Network Grammars for Natural Language Analysis. *Communications of the ACM* 13(10): 591–606.



Ramon Lopez de Mantaras is research full professor and deputy director of the Artificial Intelligence Institute of the Spanish Scientific Research Council. He received a Ph.D. in applied physics in 1977 from the University of Toulouse (France), an M.Sc. in engineering from the University of California at Berkeley, and a Ph.D. in computer science in 1981 from the Technical University of Catalonia. He was the recipient of the Swets and the Zeitlinger Distinguished Paper Award (corecipient) of the International Computer Music Association in 1997. He is a fellow of the European Federation of National Artificial Intelligence Associations. He is presently working on case-based reasoning, multiagent approaches to autonomous robots' qualitative navigation, and AI applications to music. His e-mail address is mantaras@ iiia.csic.es.

Josep Lluís Arcos is a tenured scientist at the Artificial Intelligence Institute of the Spanish Scientific Research Council (IIIA-CSIC). He received an M.S. in musical creation and sound technology from Pompeu Fabra Institute in 1996 and a Ph.D. in computer science from the Universitat Politècnica de Catalunya in 1997. He was the corecipient of the Swets and Zeitlinger Distinguished Paper Award of the International Computer Music Association in 1997. He is currently working on representation languages for integrating problem solving, case-based reasoning, and learning; the integration of software agents with learning capabilities; and AI applications to music. His e-mail address is arcos@iiia.csic.es.