Pachet, F. and Roy, P. Musical harmonization with constraints: A survey, Constraints Journal, Kluwer Publisher, 6(1):7-19 2001

Musical Harmonization with Constraints: A Survey

FRANÇOIS PACHET

pachet@csl.sony.fr

SONY CSL-Paris, 6 rue Amyot, 75005 Paris, France

PIERRE ROY

Pierre.Roy@lip6.fr

INRIA, Domaine de Voluceau, Rocquencourt, France

Abstract. We survey works on the musical problem of automatic harmonization. This problem, which consists in creating musical scores which satisfy given rules of harmony, has been the object of numerous studies, most of them using constraint techniques in one way or another. We outline the main results obtained and the current status of this category of problems.

Keywords: harmony, computer music, temporal constraints.

1. From Harmonization to Problem Solving

Computer scientists have long been considering music as a particularly interesting art. Indeed, the history of computer music is almost as long as the history of computer science. Programs to compose music, or to "make music" at various levels of the composition process, have been designed since the 50s. Some of them achieved spectacular results, such as the Illiac Suite (Hiller & Isaacson, 1993) or David Cope's symphony in the style of Mozart (Cope, 1995).

This is not the case with other arts, such as graphical arts, painting, dance, or architecture. The reason is probably that music has long been the subject of rigorous formalization, from the early stages of its evolution. In particular, so-called "tonal music", based on the idea of tonality, has developed into a formal framework which is particularly well adapted to computer modeling and problem solving. This paper surveys the main approaches and results obtained in the field of *automatic harmonization*, i.e. the problem of producing musical arrangements (scores) from given melodies, and focuses on the most widely used techniques to do so: constraints.

1.1 Tonal Music and Treatises of Harmony

In this survey we focus on tonal music, not because of some aesthetic preference, but because tonal music has been formalized in a particularly mathematical fashion, as opposed to other kinds of music.

Tonal music encompasses most of Western music produced and listened to today: from Baroque to Classical and Romantic music, to Jazz, Rock and Blues. Tonal Music is based on the notion of tonality, which consists in organizing sounds in groups of notes called *scales*, themselves organized in a hierarchical fashion within a piece of music. Usually one

tonality is considered the main tonality of the piece, and therefore notes of the corresponding scale are considered more important than notes outside the scale. The art of tonal music consists precisely in arranging notes in such a way that tonal centers are suggested, prepared, enforced, or deceived. It is the organization of notes into related tonalities that makes this interplay possible and meaningful. Tonal music is usually opposed - at least from a musicologist viewpoint - to "atonal music", such as *serial music*, invented by Schoenberg at the beginning of the century, which consists in giving each note of the 12-tone scale an equal importance. Other kinds of non-tonal music exist, such as minimalist music (represented by composers such as John Cage or Steve Reich), traditional ethnic music in general (for instance Balinese or Japanese music), or so-called "modal" music, although it can be considered as an early stage of tonal music (for instance Gregorian songs, or some varieties of Jazz-Rock).

Numerous treatises of tonal music have been written through the centuries. One of the most influential treatises is probably Johan J. Fux's *Gradus ad parnassum* written in 1725 (Fux, 1965), which was the first to propose precise rules for counterpoint, the art of combining different melodies together (Bach and Haydn are said to have studied composition with this treatise). The actual formalization of tonality, and in particular the notion of *fondamental bass* was invented by Jean-Philippe Rameau and developed in his famous treatises in 1722 (Rameau, 1985). At the other end of the spectrum, the treatise of harmony by Arnold Schoenberg (1983), is one of the most accomplished treatise of tonal music (although it also includes descriptions of his later atonal theory). More than 1000 treatises of harmony have been written, as virtually all composers of tonal music have written their own treatise. Today, it is usually considered that the evolution of tonal music - at least from an harmonic viewpoint - is now finished.

Most of the treatises of harmony are organized as a collection of rules that specify how notes can be - or cannot be - arranged together. In particular, rules specify relations that should be satisfied on simultaneous notes - called chords - or sequences of chords. We will describe these rules in the next section.

Several remarks should be made at this point. First, all treatises on harmony do not contain exactly the same rules. In fact, rules have evolved along with music. However, a set of basic rules are considered valid throughout the evolution of tonal music, and therefore always included, such as the parallel fifth rule illustrated below. Second, these rules are not intended to constitute a fully specified algorithm for composing music. They are mainly specifications of forbidden combinations of notes, these combinations being considered as dissonant or not pleasing for the ear. Of course, the composer still has a great amount of freedom in choosing the notes. Some composers even argue - and this is a general idea throughout history of arts - that the freedom of composition precisely stems from the constraints imposed by the rules. Finally, respecting the rules does not guaranty that the result will be musically meaningful or interesting. To make an analogy with language, harmonic rules can be considered as merely syntactic.

1.2 Rules of Harmony

We will now describe typical examples of rules of harmony. We will take the case of four-voice music, i.e. music composed of four simultaneous melodies or *parts*, traditionally sung by four singers (the soprano (highest), alto, tenor and bass (lowest)). Rules of harmony can be classified according to their scope in the score. We do not intend to draw a complete list of these rules (this would be equivalent to producing yet another treatise of harmony), but rather to introduce typical examples of the rules to give the flavor of the whole problem.

The simplest rules simply states that each melody has a given range, which corresponds to the voice range of the corresponding singer. For instance, the soprano spans two octaves

from C_1 (middle *C* on the piano) to C_3 (that is, 25 notes). Additionally, a rule states that voices should never cross each another.

Horizontal rules hold on successive notes of a given voice. For instance, several intervals between notes are forbidden, such as the *tritone* interval (see Figure 1), and in general all intervals considered dissonant in the given style of the composition.

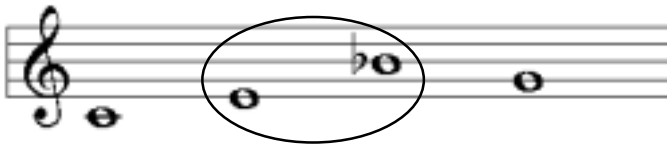

Figure 1. The two notes in the oval violate the rule about forbidden horizontal intervals (here, an ascending tritone interval between *E* and *B flat*). The two other intervals are legal (ascending major third and descending minor third).

Other rules concern simultaneous notes or chords (vertical rules). For instance, in early Baroque harmonization, only three different pitches are allowed in each chord, so one pitch must be repeated (usually with a different octave). Similarly, for certain chord types, repetition of certain notes is forbidden. For instance, for so-called "six" chords ("6"), the bass cannot be repeated (see Figure 2).

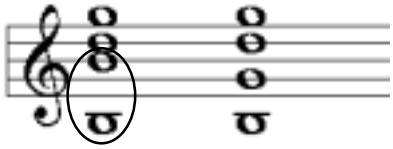

Figure 2. The two notes of the first chord in the oval violate the rule about forbidden repetition of the bass of a "6" chord. The next chord (also a "6" chord) is legal.

Finally, the most complex rules concern sequences of chords. A typical rule of the kind is the *parallel fifth* rule. This rule states that between two successive chords there should not be a parallel fifth. More precisely, if there is a fifth interval between any two notes of the first chord, then there should not be also a fifth between the two corresponding notes of the next chord. This rule is illustrated in Figure 3. Other similar rules apply to parallel octaves or unison intervals.

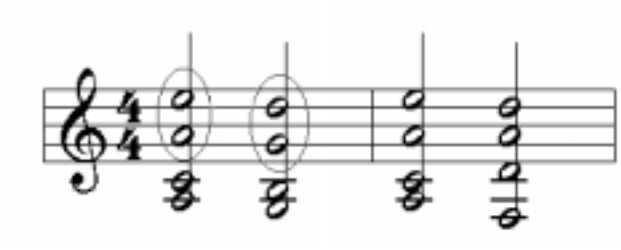

Figure 3. The first two chords violate the *parallel fifth rule:* the two fifth intervals are circled. The two last chords satisfy the parallel fifth rule.

Finally, another important element of harmonization is the so-called chord *figuring*. To extend the analogy with language, the figure represent some sort of semantic information on the harmony Since Baroque music, various kinds of figures have been used to characterize or represent extra information on chords in the score. These figures were used initially as a shortcut for chords: instead of writing the four actual notes making up a chord, only the bass would be indicated, plus a figure. The figure allows in principle the performer to reconstruct

the original chord, or a similar one. Various figuring systems have been developed during the evolution of music. Today, two main systems exist: the Baroque system, in which figures indicate the intervals making up the chord. For instance "6/3" means that the chord is made up of a sixth interval and a third interval, starting from the bass note. The other system, used in analysis and also in Jazz, is the functional notation: figures indicate the harmonic function of the chord, according to an assumed tonality. For instance, II means the chord of the second degree of the current tonality. If the two system are quite different in their role and usage, they can be considered in any case as semantic information on the chord. In practice they give useful extra information to find the notes making up the chord. In most of the cases, there are only a few possibilities for *realizing* a chord (i.e. finding the actual notes) according to a given figure.

As mentioned above, there is no official, fixed set of rules which apply for all musical styles. There is, however, an implicit consensus for certain rules, considered as applicable to a wide range of musical styles (from Baroque to Classical, for instance the parallel fifth rule). Other rules are more specific to a style (for instance the limitation of chords to three pitches typically applies to Baroque music). Complete sets of rules can be found in a natural format in any of the treatises mentioned above. A more digested rule set (20 rules) can be found in Tsang & Aitken (1991). Ballesta (1994) gives a list of all the rules for Baroque music in constraint format. In all cases, it should be stressed that the rules can be expressed precisely and locally, i.e. as predicates (usually negative properties) holding on neighboring notes (either vertical or horizontal) and intervals they entertain with each other.

1.3 Automatic Composition

One of the first uses of harmonic rules in computer music was for composing music. In this scheme, the harmonic rules are compiled into a finite state automata. The transitions correspond to legal continuations for a given state. With probabilities associated to each transition, the result is a Markov model, as used for the first time by Hiller & Isaacson (1993) in their famous Illiac suite, composed in 1958 entirely by this process. In this scheme, rules are used imply as passive checks. A short-sighted generate and test procedure is used, and the system does not perform any real search.

1.4 The Combinatorial Problem

The combinatorial problem arises when one has to produce harmonizations with an imposed, fixed voice. In this case, the harmonization problem naturally becomes a finite domain constraint satisfaction problem: each note to find may be considered as a variable whose domain is the range of the corresponding voice (soprano, alto, tenor, bass). The harmonic rules may then be considered as constraints holding on these variables, in the sense of constraint satisfaction, i.e. relations that must be satisfied in all solutions.

Several variations of this basic problem exist in the practice of musicologists, depending on what information is actually imposed. Traditional exercises in composition classes include:

- *Melody imposed* (soprano), compose the three other voices.
- *Unfigured bass*: the bass is imposed; compose the three other voices. In this case, there is only a minimum amount of information given. The problem of unfigured bass has been long debated in music theory, and studies have shown that it is in practice largely under-specified (see Ph.D. thesis of Rothgeb (1968)). Figure 4 shows a solution for a four-voice unfigured bass problem.
- *Figured bass*: the bass part is imposed, as well as figures indicating the nature of the chord on top of each note of the bass. This information reduces considerably the possible notes. It used to correspond to an actual practice of harpsichord players, who

had to play - in real time - chords that satisfied the bass voice, the figures, and a sung melody.

- *Two-voice* problems. A melody is imposed, find the bass only. In this case, only a subset of the rules - or simplified rules - is applied.

It is important to note that the first two problems naturally induce a combinatorial viewpoint. In the last two problems, because of the extra information given, the combinatorial aspect is less important. Somehow, the intentional aspect of composition and the combinatorial aspect are mutually exclusive. Consider Figure 3 for instance: a correct solution (two last chords of the figure) produces a quite different tonal progression (the last chord is a D minor chord) than the first - incorrect progression (the second chord is a G major chord).

However, the strict combinatorial problem is interesting as such from the constraint viewpoint. The following section reviews the main approaches in solving this problem with constraints.

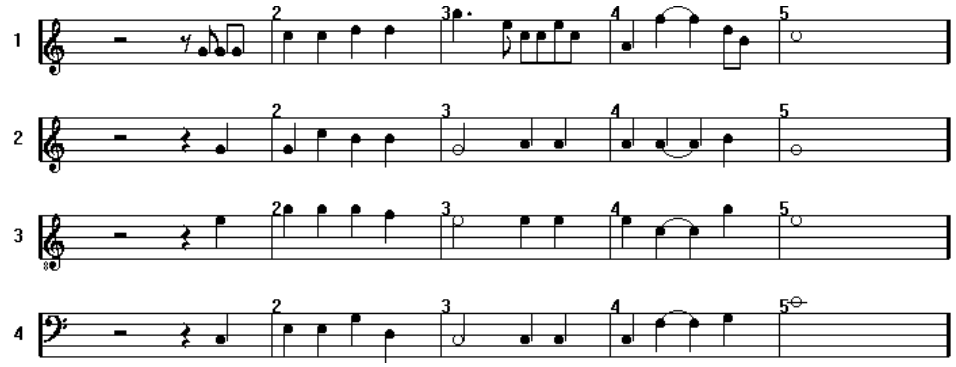

Figure 4. A four-part harmonization in the Baroque style. The first voice (soprano) is imposed. The three other voices (2 to 4) must be composed to satisfy the rules of Baroque harmony (from Roy, 1998).

2. Automatic Harmonization with Constraints

This section reviews the main works in trying to solve one of the automatic harmonization problems, and focuses on works that are based on the notion of constraint.

2.1 First Attempts at Modeling the Problem with Constraints

The first works were pioneering attempts at using some sort of declarative means for representing musical rules. No constraint satisfaction algorithm was used, and the main concern was about mastering the combinatorial explosion, by putting more knowledge in the solver.

The first attempt at representing musical rules as constraints, that is as declarative relations posed on musical representations is probably Steels (1979). In this work, Steels proposes to use constraints to create passing chords, i.e. chords that may be inserted between two given chords. These passing chords must satisfy some musical constraints, such as interval relations between the roots of the first, passing, and last chords. Steels uses essentially a frame system, augmented with a bread-first search. The combinatorial explosion is not explicitly handled, apart from a clever lazy evaluation mechanism.

The same author experimented later with a more sophisticated musical system (Steels, 1986) which aimed at solving the four-voice harmonization problem in its full generality. This system has the interesting characteristic of combining brute exploration (with no consistency procedure) with heuristic search. One of its aims is to actually learn automatically heuristics from solutions.

Courtot (1990) describes a Prolog-II system for defining musical structures and constraints relating them, in order to create polyphonies. The system - Clara - aims at representing composer's models, and is based on an extendable type system. Although the notion of constraint is explicitly introduced to define types, emphasis is made on type induction rather than on search space exploration.

Daniel Levitt describes in its thesis (Levitt, 1981; 1993) a music constraint language to define stylistic properties of harmonic accompaniments. This language allows to specify incrementally constraints, starting from simple voice ranges, to the motion of chord roots. The syntax of the language allows to specify basic arithmetic relations between pitches and chords, which is shown to be enough to specify piano pieces in the Ragtime genre. The solver, apparently, uses a basic backtracking procedure, so its ability to cope with combinatorial exploration is limited. Therefore, it only works for small or easy problems, such as under-constrained ones.

2.2 Using Constraint Satisfaction

This section outlines the works in automatic harmonization using fully-fledged constraint satisfaction, under various forms.

Schottstaedt (1984) describes a complete four-part harmonization system based on Fux's treatise (Fux, 1965). Schottstaedt makes in particular a detailed analysis of Fux's theory, and classifies the rules according to their importance, represented by a penalty: Prohibitions (infinite penalty), Very bad infractions (penalty of 200), Bad infractions (penalty of 100) and other rules, with smaller penalties. The parallel fifth is an example of prohibition (i.e. no exception to this rule is accepted). The range rule (i.e. voices should remain in their range), is considered a very bad infraction. The system uses a best first backtracking strategy. To avoid being stuck in a local extremum, and to reduce computation time, the algorithm is forced to abandon the current branch of the tree when it finds one solution. The resulting algorithm is therefore incomplete, but is able to find good solutions quickly.

Ebcioglu (1987) is probably the first to have completed a system able to produce high-quality four-part music entirely automatically. His system, the result of an impressive amount of work, contains various related modules, dealing with different aspects of the composition process, including harmonization, melody generation, melody analysis, passing note generation. Concerning the strict problem of harmonization, Ebcioglu's system contains about 350 rules, representing the style of Johan Sebastian Bach as analyzed by Ebcioglu. These rules are represented in a constraint language specifically designed for the task, called BSL. BSL is essentially a constraint logic programming language implementing backjumping, a technique considered by the author as necessary for coping with the combinatorial search specific to music harmonization. Ebcioglu's system is interesting because it is the first to be able to solve a difficult musical problem, and because it produces high quality results. The system is, however, difficult to understand and assess, because it is implemented in a proprietary language, and contains lots of interrelated modules. An attempt at untangling the system from the viewpoint of constraint processing was made by Dellacherie (1998), who extracted the rule system and reimplemented it in Eclipse, although with poor results, due to the lack of expressivity of the target constraint language (Eclipse does not handle full arc-consistency on finite domains). As showed by Ovans, full arc-consistency is necessary in musical harmonization, so bound consistency is not sufficient to achieve good performance.

Ovans and Davidson (1992) (see also Ovans, 1992) were the first to emphasize the deep combinatorial aspect of fully automatic harmonization, and advocate the use of arc-consistency. They implemented a system for solving two-voice harmonization using Fux's rules, and compared several filtering techniques, such as standard backtracking, forward-checking, and full arc-consistency on the same problem. Although no method appears always better than the others, Ovans makes a point that arc-consistency should always be used for solving harmonization problems, and criticizes Ebcioglu's system on this respect, arguing that his system would have benefited much more from arc-consistency than backjumping. It is interesting to note that these conclusions are today considered true in general in constraint programming. However, Ovans's system produces only two-part harmonies, so its results are not fully convincing.

The first system to produce four-part harmonization using conventional logic programming techniques is described in Tsang & Aitken (1991). This system performs four-part harmonization using a set of 20 rules, and is implemented in CLP (R) (Jaffar & Lassez, 1987). They use a straightforward representation of musical objects (pitches, intervals and chord types). The result is a very slow and space consuming system (70 Megabytes of memory to harmonize a 11-note melody). However, because of the simplicity of the rule set, it is often used as a reference work to be compared against.

A very detailed study of four-part harmonization using conventional constraint satisfaction techniques is the Ph.D. thesis of Philippe Ballesta (Ballesta, 1994). Ballesta implemented a full set of harmonization rules using Ilog's *PECOS* system, the ancestor of Ilog Solver. The problem solved is *Figured bass*, and the work includes detailed descriptions of the representation of musical objects and of the constraints. The solutions produced by the system are considered relatively good musically. The performance of the system, however, is not so satisfying, mostly because it uses constraints for representing all musical relations, and not only the harmonic rules. For instance, the relation between a chord and its bass note is represented as a constraint, as well as the relation between two notes and the interval between them. This radical choice produces an elegant and expressive system, which, however, is limited in its capacity to handle very complex or long melodies.

2.3 Problem Structuring

The various attempts at solving the harmonization problem with constraint techniques *stricto sensu* outlined above have showed that constraints were a useful paradigm, but are still limited, and basically unable to cope with realistic melodies in reasonable time. Other approaches have tried to improve upon these works, in particular by trying to find better representations of the problem. We outline two of these approaches here.

The first one consists in exploiting chord structures in a more active way. In the preceding approaches, chords have always been treated as simple groups of notes (which they are in some sense). However, from the constraint viewpoint, this is a mistake. Indeed, some important harmonic rules (such as the parallel fifth rule, see Section 1.2), concern explicitly chord entities, and not notes. Of course, it is possible to represent these rules as holding on note objects, thereby squashing chords out of the problem. This is in particular the approach taken by Ovans, Ballesta or Tsang & Aitken. We have shown in (Pachet & Roy, 1995) that by considering explicit chord variables on top of the basic note variables, one could dramatically divide the theoretical complexity of the problem. More precisely, the complexity can be divided by a factor equivalent to the density of chords in the space of note quadruplets. This scheme is realized by actually splitting the CSP in two parts: one part dealing only with notes and note constraints, which results in the construction of domains for chord variables. The second part handles the chord constraints (such as the parallel fifth) and produces the solutions. In practice, the scheme allows to compute the solutions of unfigured four-part exercises practically in real time. Details on the design and

implementation can be found in (Roy, 1998). An important fact to note is that these results are obtained without adding any knowledge to the system.

Another approach to reduce the complexity of the harmonization problem has been proposed by (Henz et al, 1996; Zimmerman, 1999). It consists in building prior to the resolution a harmonization plan, which contains information on the harmony to be produced. This plan represents the harmonic intention of the piece, this very intention that has disappeared with the combinatorial problem (see section 1.4). The plan contains figure information such as the degree (e.g. I, II, see Section 1.2). This information reduces dramatically the number of possible notes to be used, since it determines the pitch classes of each chord (e.g. C, E, G), and leaves open only the actual octaves of these pitch classes (e.g. C0, E2, G2, C2). Once the harmonic plan is produced, a straightforward constraint system (written in Oz) is used to compute the actual notes (this is called the "realization" of the chords). This second problem is much simpler than the full unfigured four-part problem.

2.4 *Other Approaches*

A variety of constraint solvers designed for musical problems have been built at IRCAM - a French research center specialized in computer music - and used by composers of contemporary music. *Situation*, the latest one, allows to specify polyphonies with a set of built-in specific constraints (Rueda et al. 1998). It is not aimed at solving exercises of tonal music, but rather as a real composition tool. An interesting characteristic of the engine is its capacity to state constraints in an abstract manner: on "points" (e.g. notes, chords, rhythms values) and "distances" (e.g. pitch intervals, time intervals, chord intervals, etc.). This abstract layer allows to use the solver in a large number of situations, not only in the pitch domain, as the user can choose how to map these "points" and "distances" to any musical entities.

From a technical viewpoint, the engine of Situation is based on a forward-checking mechanism, enhanced with lazy evaluation (Rueda and Valencia, 1997), and uses a flat representation of the problem (i.e. variables represent notes or note attributes). Experiments have been conducted to define constraints in Situation corresponding to rules of tonal harmony (Guenego, 1998), and have shown the same limitations than in the work of Ballesta. However, the built-in constraints allow to find solutions quickly, and are particularly well adapted to the needs of contemporary music composers.

The work of Ramirez and Perlata (1998), although not directly related to four-part harmonization, is worth mentioning here: they address the issue of generating a sequence of chords (actually a sequence of chord names) that satisfactorily harmonizes a given melody. They limit the chords to simple triads (three-note chords). The interesting aspect of this work is that they try to maximize the number of well-known chord progressions, which are stored in a dictionary. A simple algorithm is proposed, which is not based on arc-consistency techniques, but which allows to solve the problem. In this case, fully-fledged constraint satisfaction is not required, since the combinatorial search is smaller than with the four-part harmonization problem. However, the system does solve a useful harmonization problem.

Finally, comparisons between a rule-based approach, which consists in representing explicitly symbolic knowledge, and approaches based on genetic algorithms have been proposed in (Phon-Amnuaisuk & Wiggins, 1998), concluding on the superiority of the former in terms of the quality of the solutions found. Of course, entering explicitly the rules of harmony is a hard task. Recent works have addressed this issue by trying to induce automatically rule sets or grammars from various corpus of music scores using statistical techniques (Ponsford et al., 1999).

Pachet, F. and Roy, P. Musical harmonization with constraints: A survey, Constraints Journal, Kluwer Publisher, 6(1):7-19 2001

2.5 Available Systems

There is no - to our knowledge - commercially available software system for harmonization based on constraint techniques. This is probably because although harmonization problems - especially in the style of tonal music - are interesting exercises, they have limited practical applications. In practice, there are a lot of synthesizers which propose basic harmonization features (such as one-touch chords on some electronic keyboards). These harmonizations are usually performed on a note-to-chord, reactive basis, i.e. without taking into account the context (preceding notes of the melody). Real time requirements forbid anyway the use of backtracking for such systems !

The *Tonica* system (distributed by Software Partners) is a very good harmonization system which uses neural network techniques. It produces harmonization of given melodies and provides lots of flexibility in the user interface. The system proceeds in two phases. First it produces a chordal analysis of the melody (equivalent to figures). Then it produces an actual harmonization from the melody and the chordal information. Since there are many possible chordal analysis for a given melody, the system does not actually perform any deep search. The user has the responsibility of choosing a chordal analysis which suits his intention. In practice, this approach is much more intuitive for musicians, so the resulting system is useful. The system does not propose any new solution however to harmonization problem, from the viewpoint of problem solving. Note also that several approaches in solving the harmonization problem using learning techniques such as neural networks have been proposed, for instance in (Hild et al., 1992), and shown to yield excellent results.

3. Conclusion: Is the Problem Solved Now ?

The technical problem of four-voice harmonization may now be considered as solved, using constraint satisfaction techniques based on arc-consistency, and an adequate structuring of the problem to handle chord variables properly. This result comes after several years of trials and errors, starting from brute force approaches (e.g. Schottsdtaedt), to proprietary constraint languages (Ebcioglu), to arc-consistency techniques (Ovans), augmented with adequate problem structuration (Pachet & Roy).

However, what remains unsolved is the problem of producing musically nice or interesting melodies. There are several open issues concerning what makes melodies interesting:

First, the rules that make melodies interesting are not known, if any. As we saw, the formalization of music has mainly focused on establishing “syntactic rules”. Although some attempts were made to formulate explicit properties of melodies which sound “nice, or “balanced” (see for instance the statistical approaches and works about 1/f noise in music (Voss & Clarke, 1978)); these formulations have not reached a point where they could be turned into operational specifications.

Second, musical interest probably results from the interplay between incompatible criteria. For instance, it is well-known that conjunct movements sound nice (i.e. voices going in the same direction, up or down). But this goes in the way of the various rules against parallel movements (parallel fifth or octave). How exactly do these criteria coexist ? Ballesta’s work (outlined above) includes studies of optimization criteria, to choose among the various solutions. Ballesta proposes several criteria to describe solution melodies, and implements them using the branch and bound algorithm of the solver. For instance, a nice melody can be described by the following properties:

- The soprano part should not change direction too often,
- The soprano and bass should be in contrary motion as much as possible,
- The soprano part should move stepwise as much as possible,
- The intermediary parts should be kept as close as possible.

This multi criteria optimization problem is however, very hard; and no systematic experiments have yet been done on realistic melodies. However, the issue definitely deserves more attention.

Finally, nice melodies are melodies which contain well-known patterns, harmonic progressions, or typical harmonic "solutions". Not mentioning the problem of building a catalogue of well-known patterns, trying to maximize the number of well-known patterns in solutions turns the problem into a very difficult one indeed. The works of Ramirez and Peralta constitute a first step in this direction.

Automatic harmonization has continuously been raising important issues in the field of constraint programming, and has prompted researchers to find solutions to these issues, thereby increasing both our knowledge of musical structures and constraint satisfaction problems in general. As claimed here, this problem - in fact this category of problems - still yields challenging issues for constraint research.

4. References

Ballesta, P. (1994) "Contraintes et objets: clefs de voûte d'un outil d'aide à la composition", Ph.D., Université du Maine. Also published in *Informatique Musicale, Recherche et Applications*, Chemillier and Pachet Eds, Hermes, 1998.

Brown, F. Mouton, R. (1994) "L'automate musical en temps réel Kantor", First Journées d'Informatique Musicale, Labri, Bordeaux (France).

Cope, D. (1987) "An Expert System for Computer-Assisted Music Composition," Computer Music Journal, 11 (4), pp. 30-46.

Courtot, F. (1990) "A Constraint Based Logic Program for Generating Polyphonies", *International Computer Music Conference*, Glasgow. See also "Understanding Music with AI", Balaban M. et al. Eds, AAAI Press, 1992, pp.156-181.

Dellacherie, P. (1998) Modélisation et optimisation d'un harmoniseur automatique de chorals en PLC (FD). Rapport de DEA, Université de Caen.

Ebcioglu, K. (1987) "Report on the Choral Project: An Expert System for Harmonizing Four-Part Chorales", IBM technical report RC 12628.

Ebcioglu, K. (1993) "An Expert System for Harmonizing Four-Part Chorales", *Machine Models of Music*, S. M. Schwanauer and D. A. Levitt, MIT Press, pp. 385-401.

Fux, Johan Joseph (1965) *The Study of Counterpoint from Johann Joseph Fux's Gradus ad Parnassum*, translated and edited by Alfred Mann, W.W. Norton & Company, New York.

Guénégo, J.-L (1998) "Une application de résolution de contraintes en harmonie tonale", Ircam / Université Paris VI, 1998. Available at http://www.ircam.fr/equipes/repmus/Rapports/JLGuenego98.

Henz, M., S. Lauer, et al. (1996) "CompoZe- Intention-Based Music Composition Through Constraint Programming", *8th IEEE International Conference on Tools with AI*, Toulouse (France).

Hild, H. Feulner, J. Menzel, W. (1992) HARMONET: A Neural Net for Harmonizing Chorals in the Style of J.S. Bach, in Advances in Neural Information Processing 4 (NIPS 4), pp. 267-274, R.P. Lippmann, J.E. Moody, D.S. Touretzky (eds.), Morgan Kaufmann.

Hiller, L. and Isaacson, L. (1993) "Musical Composition with a High-Speed Digital Computer", in *Machines Models of Music*, Schwanauer, M. And Levitt, D. Eds, MIT Press, pp. 9-21. Reprint of original article in Journal of Audio Engineering Society, 1958.

Jaffar, J. and Lassez, J. L (1987) "Constraint Logic Programming", IEEE 4th International Conference on Logic Programming, Melbourne, 25-29 May.

Pachet, F. and Roy, P. Musical harmonization with constraints: A survey, Constraints Journal, Kluwer Publisher, 6(1):7-19 2001

Levitt, D. A. (1993) "A Representation of Musical Dialects", *Machine Models of Music*, S. M. Schwanauer and D. A. Levitt, MIT Press**,** pp. 456-469.

Levitt, D. (1981) "A Melody Description System for Jazz Improvisation", Masters Thesis, MIT Artificial Intelligence Lab, Cambridge, MA.

Ovans, R. and R. Davison (1992). An Interactive Constraint-Based Expert Assistant for Music Composition. Ninth Canadian Conference on Artificial Intelligence, University of British Columbia, Vancouver, pp. 76-81.

Ovans, R. (1992). Efficient Music Composition via Consistency techniques, Technical Report CSS-IS TR 92-02, Simon Fraser University, Canada.

Pachet, F. Roy, P. (1995) Integrating constraint satisfaction techniques with complex object structures. *15th Annual Conference of the British Computer Society Specialist Group on Expert Systems, ES'95*. Cambridge, pp. 11-22.

Pachet, F. Roy, P. (1995) Mixing constraints and objects: a case study in automatic harmonization. *TOOLS Europe '95*. Prentice-Hall. pp. 119-126.

Phon-Amnuaisuk S. & Wiggins, G. (1998) "The Four-Part harmonization problem: A comparison between genetic algorithms and a Rule-Based system", Proceedings of the AISB'99 Symposium on Musical Creativity, AISB Symposium, Edinburgh.

Ponsford, D. Wiggins, G. Mellish, C. (1999) Statistical Learning of Harmonic Movement. *Journal of New Music Research*, 28:2, pp. 150-177.

Rameau, Jean-Philippe, (1722) Traité d'Harmonie, reprinted as *Treatise on Harmony*, Dover, March 1985.

Ramirez, R. and Peralta, J. (1998) A constraint-based melody harmonizer, ECAI'98 Workshop on Constraints and Artistic Applications, Brighton.

Rothgeb, J. (1968). "Harmonizing the Unfigured Bass: a Computational Study," Ph.D., Indiana University.

Roy, P. (1998), Satisfaction de contraintes et programmation par objets, Ph.D Thesis, University of Paris 6.

Rueda, C. Lindberg, M. Laurson, M. Bloch, G. Assayag, G. (1998) "Integrating Constraint Programming in Visual Musical Composition Languages", ECAI Workshop on Constraints for Artistic Applications, Brighton.

Rueda, C. Valencia, F. (1997) "Improving forward-checking with delayed evaluation", in Proceedings of CLEI'97, Santiago, Chile.

Schoenberg, A (1993) *Theory of Harmony*, University of California Press.

Schottstaedt, B. (1984) Automatic species counterpoint. Tech. Rep. STAN-M-19, Stanford University CCRMA. A short report with source code appeared in *Current Directions in Computer Music Research*, Max V. Mathews and John R. Pierce (eds.), MIT Press, 1989.

Steels, L. (1979). Reasoning modeled as a Society of Communicating Experts, MIT AI Lab technical report.

Steels, L. (1986). Learning the craft of musical composition, ICMC, The Hague (Netherlands).

Tsang, C. P. and M. Aitken (1991). Harmonizing music as a discipline of constraint logic programming. ICMC, Montréal.

Voss, R. Clarke, J. (1978) 1/f noise in Music, *Journal of the Acoustical Society of America*, 63, 1978, pp. 258-263.

Wiggins, G. (1998) The Use of Constraint Systems for Musical Composition, ECAI'98 Workshop on Constraints and Artistic Applications, Brighton.

Zimermann, D. (1999). Modelling Musical Structures, *Constraints*, this issue.