Harmonizing Music as a Discipline of Constraint Logic Programming

C P Tsang and M Aitken
Department of Computer Science
University of Western Australia
Nedlands, Western Australia 6009.
(tsang@cs.uwa.oz.au)

ABSTRACT

Constraint Logic Programming (CLP) is a new logic programming language which extends the concept of unification to include constraint solving. This is particularly useful for the specification of constraints inside a first order logic frame work. Harmonization of music is in essence a combinatoric selection as well as a constrained solving process. In this paper, we report the design of a simple harmonization system for four part choral music and illustrate the natural declarative style of harmonization rules using CLP. The system is very lucid and small. This simple system was tested against some of the examples in harmonization texts, and demonstrated to be useful for simple composition as well as for automated tutoring purposes.

We conclude by showing the convenience and advantages of CLP in expressing all type of harmonization rules.

1. Introduction

Since the mid 1950, computers has been used to assist composers to harmonize music [Hil97]. Many computational models has been applied to the construction of these systems. They include grammatical rules [Hol80, Ro85], and more recently 1st order logic [Ed87, Ed88]. While the grammatical model is appropriate for composing serial music, it is not that simple to handle polyphonic music. Furthermore, the selection of the grammatical rules is based on pattern matching, and only one grammar rule is applied at any one instance with non-determinacy generally not possible. In order to capture all the possible situations, usually the grammatical method results in multiple distinct but related rules. This difficulties can be related to the forward chaining mechanism in automated reasoning. The next advance is the use of logic programming models [Ed87, Ed88] where unification and backtracking are used to select alternative solutions. The main advantage here is that a rule is applied on successful unifications of the heads, and the most general unifier is used to propagate the inference. The advantages is that a combination of more than one rules can be enforced by the "and", and alternative solutions are possible by means of the "or" construct. However, the result is that there are many solutions and combinatoric explosion happens to the search. The harmonization process in this case is similar to that of generation and test. To overcome these problems, one way is to improve the speed by some specialised tool which can generate and test at high speed [Ed87].

In recent years, through the theoretical investigation of logic programming Jaffar and Lassez designed Constraint Logic Programming (CLP) [JL87]. CLP generalizes the concept of unification to that of constraint solving, and it permits not only the combination of rules by also the combination of multiple constraints by the "and" construct. Thus CLP allows a very general representation of the harmonization alternatives, reduce the
amount of backtracking, and allow a very compact, lucid and simple representation of the rules. In this paper we demonstrated how to program harmonization rules in CLP and investigated into the efficiency as well as possibility of using Constraints to construct some large scale automated composing tools.

2. Preliminary of Constraint Logic Programming

Constraint Logic Programming was first proposed by Jaffar and Lassez [JL87]. We assume that the reader is familiar with the semantics of a logic programming language such as Prolog. In that context, CLP is an extension of PROLOG by extending the meaning of unification. In Prolog, unification is only applied to 1st order terms. Jaffar and Lassez observed that by finding a constraint system where the solutions are decidable, one can generalise the concept of unification to that of finding solutions for constraints. By this extension, the arithmetical statements are no longer treated as imperative assignments, but as constraints. Some examples of such constraint domains are real number linear constraints, and boolean constraints. The CLP that is currently available to us is the one which uses only real number domain [HLN87]. Although real number is not the most convenient constraint system to use, it serves the purpose of our initial investigation.

3. Music Harmonization and its Rules

In this paper, we address the harmonization of 4 part choral music. A piece of choral music is composed of four voice parts. They are soprano, alto, tenor and bass. Each one of these voices has its own range. Usually each range covers a bit less than 2 octaves. The characteristic of these parts is that each one of them should be a reasonably nice melody by itself. However, when they are sung together, they harmonize together to give a pleasant effect. Harmony is governed by the relative intervals of the notes which is defined by the chord. The parts of a piece of music is not only constrained by the chord, but it is also constrained by some rules of progression of the chords which are called Cadences. By the harmonization of music, we mean the choice of notes for the alto, tenor and base parts, after the soprano part is given.

From the above description, we already see that there are many different constraints governing the choice of notes to the parts. In fact these basic rules are found in standard harmony text books such as [KP84, La82, Pl65, Lo81]. By consulting these books, we come up with two sets of rules. They are the harmony rules and the voice selection rules. These rules are given as follows:

**Harmony rules.**

1. The first chord selected must be the tonic chord.
2. At the end of a phrase a cadence must be selected. It may be an imperfect, interrupted, plagal or perfect cadence.
3. The last cadence selected must be a perfect or plagal cadence.
4. No cadence should be immediately preceded by its second chord.
5. No cadence should be immediately preceded by its first chord.
6. Avoid consecutive chords which are the same.
7. Avoid such progressions as Chord II followed by Chord I, Chord IVb followed by Chord IV, Chord V followed by Chord IV, Chord IVb followed by Chord V.
8. Avoid second inversions unless at cadence points.
9. When there is an equal choice between a first inversion and root position, the first inversion is chosen.

**Constraints governing note selection.**

1. Voices are limited to their vocal ranges.
2. Voices may never cross.
3. Next voices should be kept close.
4. Contrary motion between soprano and bass is good.
5. No consecutive fifths, octaves or unisons or hidden fifths or octaves within chords.
6. No illegal leaps within voices. (there are many cases of illegal leaps)
7. Whenever possible a voice should move to the nearest available note.
8. The leading note must always rise to the tonic.
9. The supertonic note should always try to fall to the tonic.
10. A note occurring in two consecutive chords should be kept in the same voice.
11. For a chord in root position it is best to double the root. etc.

4. Representing the Harmonization rules in CLP

We represent of a musical note in CLP by a term called pulse. A pulse is a first order term containing the following pieces of information: length in time, soprano pitch, alto pitch, tenor pitch, base pitch, chord identity. A pitch is denoted by a number. Middle C is the pitch number 60. A pitch class is of simply pitch number modular 12 so that any pitch can be classified by the 12 basic tones. An interval is the difference between two pitch. For every melody to be harmonized, the scale note class list and the triads list are first generated for the given key of the music. The triads are identified by the pair (number, inversion). Number is the scale degree on which the triad is to be built, an the inversion specify the different allowable triads. For example the allowable triads for major keys are:

\[ I_a, I_b, I_c, I_i, I, IV_a, IV_b, IV_c, V, V_a, V_b, VI, V_Ib \]

A piece of music is a list of pulses. Before harmonization, the variables in the term pulse are only partially specified. As we are given the melody of the soprano, the length of the pulse and the soprano pitch are the only variables instantiated. On successful termination of the harmonize query all the variables are fully instantiated giving a complete piece of music.

The program is the set of rules in section 3 represented as the constraints and rules. Most of the rules are representable by constraints between the variables of the terms. Typical numeric constraints are the voice selections, contrary motions, consecutives, cadence selections and so on. All the above 20 rules are defined in less than 9 pages of CLP code, and the full system was implemented in less than 15 pages of CLP code.

5. Result and Conclusion

We originally constructed our harmonizing system on a SUN-3/60. Recently, we have converted the system onto a SUN-SPARC workstation. We have experimented with some of the examples from the AMEB music examinations and the results are pleasing. On a SPARC-1, to harmonize a melody phrase with 11 notes, it took 5 minutes to generate the first solution. Example of a solution is shown in the diagram. However, it does require a swap space of up to 70 Mbyte and for larger problems it requires up to 100Mbyte of swap space. We believe that this excessive requirement of swap space is due to the very inefficient space utilization of our implementation of CLP as well as the inherent cost of constraint propagation. A constraint system with efficient space utilization specially designed for music is our current goal. We believe that substantial performance improvement is achievable. CLP can also generate all possible solutions, but it will take much longer. This harmonization system not only can be used to compose, it can also be used (with no change to program) to check whether a given piece of music follows the required rules. If the list of pulses are fully instantiated before it is passed to the harmonize query, the system will answer whether the given piece of music is correctly harmonized or not. When it is executed in this mode, the system is suitable for tutoring purposes. Failure explanation may also be generated to identify the error. We have used this mode of operation to verified some of the examples in the AMEB examination solutions.

Harmonization is inherently a constraint solving paradigm. This is demonstrated by the ease and natural manner that all the rules are expressed in term of CLP. Furthermore the constraints that are required are
basically linear. This means that it is solvable by the Simplex algorithm. The use of constraints for harmonization has been studies before [Ro85] and more recently by [Ov91]. However, those constraint systems are static while the set of constraints in CLP is dynamic. An example of this situation, is that usually the triad Vc is not allowed, but it is OK as a passing between IV and VI. This dynamic alternatives are possible via the backtracking mechanism. Another advantage of constraints is that negation is usually by exhaustive searches in the logic programming paradigm. However, negative constraints are simply a combination of two inequality constraints. Thus avoided the time consuming searches. In CLP constraints can be selected and combined with many different possible combinations automatically, and one can specify with all the possible alternatives by a concise specification. The system can construct a possible set of constraints dynamically. Further the CLP also provide a powerful specification method using 1st order terms and 1st order logic. Thus CLP is suitable for combinatoric selection together with constraint solving.

While CLP is probably one of the better models for harmonization, it is very slow and large. There are two ways to improve this. Firstly, one should look for specific constraint domain for music such that a faster constraint solving algorithm may be invented. Secondly, the deterministic left-most depth first search of CLP may not be suitable for music search. One should try other types of rule based or heuristic based tree search strategies. This is shown by the very high density of solutions in some branches of the SLD tree. If one can identify these branches, the search can be improved substantially. These are the problems we are addressing now.

References

/Lo81/ Lovelock William, First Year Harmony, Hammond.
/[Pi55/ Pilling Dorothy: Harmonization of Melodies at the Keyboard, Forsyth Bros, 1955.

ICMC 64