Adaptive User Modeling in a Content-Based Music Retrieval System

Pierre-Yves ROLLAND
Laboratoire d'Informatique de Paris 6 (LIP6)
and Atelier de Modélisation et de Prévision, Université d'Aix-Marseille III,
15/19 allée Claude Forbin, 13627 Aix en Provence cedex 1, France
Tel: +33 61 19 19 19 7
P_Y_Rolland@yahoo.com

1. INTRODUCTION

While it is clear that user modeling could be valuable in many music retrieval contexts [1], the focus in this paper is on content-based music retrieval as in the WYHIWYG (*What You Hum Is What You Get*) paradigm [5], also referred to as Query-by-Humming. Desirable data to include in such a user model include:

- Musical preferences, expressed by the user by answering the system's questions (examples can be found in [1]).
- History-oriented information computed by the system, e.g. the music genre most often retrieved by the user *recently* (or up to now).

Such data can indeed be utilized by a CBMR system to bias its search results, hopefully making the latter more accurate. Based on user data the system builds expectations which are used to filter candidate results. For instance, if a song belonging to the user's most usually retrieved music genre matches the user's query, it would be advantaged by the system over a matching song of a genre the user never retrieved before. Similarly for songs belonging or not to a genre declared by the user as being among his/her favorite. A number of methods have been proposed for collecting, representing and updating these more traditional kinds of user data. These are out of the scope of this paper.

In this paper the focus will be on presenting concepts and techniques for **modeling a user's sense of musical similarity**, which I see as absolutely central. However the similarity model which will be proposed throughout the rest of this paper is seen as part of a larger user model including the more 'traditional' data types mentioned above. Whatever the user modeling paradigms used, I suggest that user models in CBMR should be **adaptive**. This means that the model is continuously enhanced throughout the successive interactions with the user. While it can be desirable to initialize user X's model by asking X a series of questions, the adaptive paradigm allows to instead initialize the model of every new user to a default and then automatically, incrementally personalize it based on the user's feedback.

In the next section I explain why modeling of a user's sense of musical similarity is seen as central in a CMBR system.

2. RATIONALE

The rationale behind this work lies in the following points:

- **A.** Most — if not all — content-based retrieval systems for music use similarity searching.
- **B.** Similarity search is based on an explicit, or sometimes implicit, **formal model of musical similarity as it is perceived by human listeners**. In this paper, to avoid any confusion between 'musical similarity model' and 'user model' the former will be designated by the more restrictive term 'melodic similarity *function*'. Such a mathematical function is designed to automatically compute the similarity between two melodic pieces or passages, often entailing a whole algorithm such as a dynamic programming one. Many different such functions have been proposed. Each is underlied by a sequence comparison scheme that is either exact or approximate (strict vs. error-tolerant matching), binary or gradual (boolean vs. gradual similarity function), etc. — see e.g. [4] for a review
- **C.** Human judgments of musical similarity are multidimensional. This means that the perceived degree of similarity of, say, two passages not only derives from the absolute pitch and duration of the notes heard but generally from a far larger set of musical characteristics of the two passages.
- **D.** The relative importance of descriptions in the overall similarity judgment can be different from one description to another. As a well-known example, it has been established that two isochronous passages having exactly the same underlying interval sequence are often judged more similar than two isochronous passages having mostly the same absolute pitches sequence but with several mismatches.
- **E.** Last but not least, **the relative importance of one given description can vary from one individual to another.** For instance, for certain human subjects rhythmic aspects contribute more strongly to similarity than pitch aspects, and vice-versa.

For all these reasons I suggest it is desirable that CBMR systems should use not only general user models, but also models of users' sense of musical similarity. Since there is no way to (entirely) predict the parameter values of such a similarity model beforehand — i.e. based on general user characteristics of the user — the model should be **adaptive.** In other words, the system adjusts the similarity model based on user feedback received during successive interactions with the user (search sessions).

There is a trade-off between search speed and search quality. Very fast search techniques have been developed for CBMR (e.g. [2]), which is convenient for allowing the user to carry out for instance a broad 'initial screening' of a music content database. However these techniques easily lack recall (or even precision) because their time efficiency relies on the simplicity of musical similarity models and, correlatively, on the strictness of match criteria. Similarly to standard text-based search engines, it is seen as important that CBMR systems should also offer 'advanced' or 'specialized' search modes based, among others,

on richer musical similarity models and more flexible match criteria. The ideas and techniques presented in this paper should prove even more useful for these slower search paradigms.

3. USER MODELING PARADIGM

The proposed user modeling paradigm is intrinsically linked to the CBMR framework in general, and to that of melodic similarity assessment schemes it uses. These frameworks are presented in the first three subsections.

3.1 CBMR Framework

The CBMR paradigm in which we place, viz. that of the Melodiscov system [5] will now be briefly described. Schematically, pattern-matching techniques are used to match the user's query against the target collection of music pieces (see Figure 1). The underlying similarity function (see 3.2) being *gradual*, search results are returned under the form of a ranked list of matches, by decreasing order of match quality. In the typical querying mode, the user can hum, sing (with lyrics), whistle or play an acoustic instrument, in which case Melodiscov's *transcription* module transforms the audio query into a MIDI-like music structure called *raw symbolic query*. (Here the adjective 'symbolic' is used to distinguish between direct content, viz. digital audio signal, and abstract content such as MIDI or score representations).

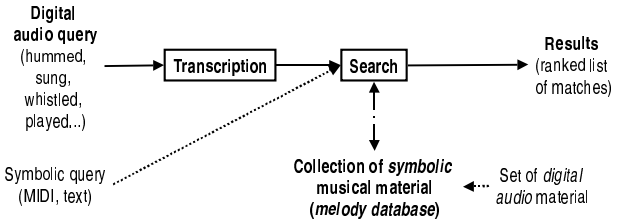

Figure 1. CBMR paradigm (Melodiscov system)

Although some of the concepts and techniques proposed in this paper would apply to other kinds of CBMR schemes, it is assumed here that the target content database is a collection of musical works such as MIDI songs, called *melodic database*. In that phrase as well as in the rest of the paper, 'melodic' is used to distinguish from other kind of musical content, e.g. harmonic (chord) sequences. However 'melodic' does not imply monophonic material or mere pitch sequences with no rhythmic information.

3.2 Melody Representation

In Melodiscov music is represented using multiple characteristics (called *descriptions* henceforth). These are derived from the immense body of work that has been carried out in the areas of music psychology, music perception and cognition, and music theory at large. The various descriptions for melodic material are categorized according to their horizontal span (examples given below do not necessarily fit all melody retrieval contexts — a far more complete list can be found in [3]):

- *Individual* descriptions correspond to individual notes (or rests, or chords). Examples: 'absolute pitch', 'forward interval direction', 'backward chromatic interval', 'backward duration ratio', 'forward metric variation'…
- *Local* descriptions correspond to groups of notes (or rests/chords). Examples: 'ascending pitch contour', 'gap-fill', 'phrase-based grouping'…
- *Global* descriptions correspond to a whole melody (viz. one song in the searched database or the hummed/sung/whistled/… query). Examples: 'pitch histogram', 'average note duration' 'time signature', 'overall tonality'.

The raw symbolic query output by the query transcription module is, roughly speaking, a MIDI melody. Similarly, currently in Melodiscov the melody database is initially made of standard MIDI files. This initial, MIDI-type, representation comprises only three descriptions: the individual descriptions absolute pitch (or Midipitch 0-127), relative duration (in number of beats) and absolute amplitude (0-127). An algorithmic step is required to compute the *final* representation from the *initial* one. An automated *representation enrichment phase* is inserted in the CBMR algorithmic scheme. In an incremental process, descriptions are derived one after the other from basic descriptions and/or already derived descriptions, in a specific order (see [3] for more details).

3.3 Melodic Similarity Assessment

3.3.1 Overview

The melodic similarity function used in Melodiscov is based on the Multidimensional Valued Edit Model or MVEM (see e.g. [4]). MVEM has been designed to accommodate the multiplicity of musical descriptions, each possibly with a different horizontal span. MVEM generalizes the basic string edit distance framework and allows to carry out soft matching, i.e. allows a level of discrepancy between the query and a candidate passage in a target music work. Such error tolerance is fundamental in CBMR systems because errors in CBMR result from many possible causes:

- Users' inaccurate remembering of the searched melody
- Monophonic rendering, by users, of inherently polyphonic queries (think for instance of a theme with some bi-phonic passages)
- Transcription process, either because the query is physically too inaccurate (wrong pitches and/or rhythm) or because of technical shortcomings in the transcription algorithm.

3.3.2 MVEM in greater detail

MVEM will now be described in more technical terms, using as an example the similarity computation between the two passages shown in Figure 2. These two passages illustrate typical ornamentation cases that can be encountered in CBMR.

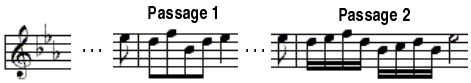

Figure 2. Two melodic passages (Joseph Haydn, Concerto for Trumpet in Eb major)

To compare two passages, the optimal correspondence scheme between their respective elements is determined. A 'correspondence scheme', called *alignment*, is a series of *pairings*, with each pairing meaning that two groups of notes and/or rests are put in correspondence (see Figure 3 and Figure 4). For instance, the second pairing in Figure 3 puts in correspondence notes 2 and 3 of passage 1 with notes 2, 3, 4 and 5 of passage 2. This is depicted in gray on the figure as two ellipses connected by an oriented link. I have introduced the notion of pairing to define alignments as it provides a richer and more flexible formalism than the traditional 'edit operations' formalism. Each group in a pairing may contain only one note or even zero note (see below). The succession of pairings forming an alignment between the two passages is interpreted as a *transformation* of passage 1 into passage 2. For instance, in Figure 4 it is considered that the final

Eb (quarter note) in passage 1 is *replaced* by the final Eb (half note) of passage 2. Similarly, in passage 2 the second Eb is said to be *inserted.*

For any pairing the number of notes in each group determine the *pairing type.* Let the *signature* of a pairing be the integer pair (#G1,#G2) where #G1 (resp. #G2) is the number of notes and/or rests in the pairing's first group (resp. second group). In the above example, the pairing's signature is (2,4).

- Pairings with a signature of that form, i.e. (r, s) where r > 1 and s > 1 are called generalized replacements.
- Pairings with signature (1,1), such as the first pairing in Figure 3, are called [individual] replacements.
- (0,s) pairings, where s > 1, are called generalized insertions.
- (r,0) pairings, where r > 1, are called generalized deletions.
- (0,1) pairings are called [individual] insertions.
- (1,0) pairings are called [individual] deletions.

As can be seen, such important musical notions as ornamentation or variation can be neatly dealt with in this framework. One other powerful feature of MEVM is that it can *elicit* (or explain) the similarity between two passages P and P', but this is out of the scope of this paper.

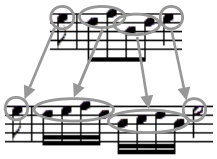

Figure 3. One possible alignment between the two passages, using 2 individual replacements and 2 generalized replacements

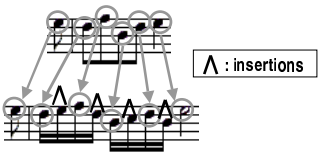

Figure 4. Another possible alignment between the two passages, using 6 individual replacements and 4 insertions

3.4 Valuation

In a **valued** edit model, a *similarity contribution function* (in short 'contribution function') is associated to each pairing type. Every pairing *p* in an alignment gets a numerical evaluation *contrib(p)* reflecting its individual contribution to the overall similarity. The contribution may be positive or negative. The various descriptions in the representation are simultaneously taken into account in contribution functions using a weighted linear combination paradigm, as shown in Equation 1. *contrib(p)* is the sum, over all descriptions belonging to the music representation *R*, of terms $w_D \times contrib_D(p)$, where:

- w_D is the weight attributed to description *D*, a real number in [0;1]. A weight has value zero iff the associated description is not taken into account in the model, at least at that particular moment.
- $contrib_D(p)$ is the contribution of *p* seen only from the point of view of description *D.* Suppose for example that p is the replacement of the final Eb (quarter note) in passage 1 by the final Eb (half note) of passage 2 in Figure 4. Consider the basic descriptions *D1*: 'degree of note in overall tonality' and *D2*: 'relative duration of note in beats'. We can expect $contrib_{D1}(p)$ to have a strong positive value because the degree is the same for both notes. Conversely, $contrib_{D2}(p)$ can be expected to have a (moderately) negative value because the duration of the 'replacing' note is double that of the 'original' note.

The *value* of an alignment is the sum of all of its constitutive pairings contributions (Equation 2). Finally, the **similarity** between passage 1 and passage 2 is defined as being the **greatest value of all possible alignments between the two passages**. There are several techniques, based on dynamic programming, for computing that greatest value as well as, if needed, the corresponding alignment(s). In the case of CBMR, the *matching quality* (or 'matching score', etc.) between a query and a music work is the greatest value of all possible alignments between the query and a passage of the work. The search results are made of the list of works, ordered by decreasing matching quality, whose matching quality is above a predefined threshold.

$$contrib(p) = \sum_{D \in R} w_D \times contrib_D(p)$$

Equation 1

$$Value(A) = \sum_{p \in A} contrib(p)$$

Equation 2

3.5 Model Representation and Adaptation

3.5.1 Description weight vector

A user's sense of melodic similarity is modeled using a scalar vector which we call *description weight vector* (DWV). The DWV contains the weight w_D of each description D in the music representation, each weight being able to vary throughout user interaction — primarily search sessions. This user model underlies a melodic similarity function that emulates as closely as possible the user's sense of melodic similarity.

The numeric vector format of the user model allows it to be shared by different applications or agents. In fact, a musical similarity function is inherently modular, other musical software such as navigational interfaces, pattern extraction programs and so on can directly reuse it for the omnipresent purposes of melodic comparison. Also, a DWV can directly be merged with a sharable user model such as the one suggested in [1].

3.5.2 Initialization and interaction

The first time a user uses Melodiscov, the DWV is initialized by setting all of its components (description weights) to a default value of 0.5.

Every time the user is presented with a ranked list of search results, s/he gives feedback to the system in two possible fashions:

- In the simplest interaction mode, *single match feedback*, s/he just tells the system which of the found matches is correct, i.e. corresponds to the music work actually looked for. In case that music piece does not appear in the system's list of matches, the user may request that lower quality matches, if any, should be displayed. These are matches whose similarity scores are below a given constant threshold specific to the CBMR system.
- In a more complex one, *ranked match feedback*, s/he gives her/his own ranking of some or all of the matches. Think for instance of a content database containing several variations of a target song; these variations could be designated by the user to the system as 'reasonable secondary matches'.

3.5.3 Model update

Unless the user has confirmed the system's best match (*single match feedback*) or best matches (*multiple match feedback*), the

user's DWV is updated in the following manner. (For the sake of simplicity it will be assumed that *single match feedback* mode has been used; feedback in the other mode is dealt with similarly).

The system's ranked list $\{M_1,\ldots, M_m\}$ of matches is separated in three groups (in decreasing order of match quality as computed by the system) :

- the group $FP=\{FP_1, ..., FP_f\}$ of all the 'false positive' matches (i.e. all matches reported by the system with a better matching score than the correct match. In other words, if the user selects Mi the list's then $FP=\{M_1..M_{i-1}\}$)
- the correct match C
- the group S of all subsequent matches proposed by the system

For each, the contribution of each description to the matching score is computed. The weights of some, if not all, weights in the user's DWV are then adjusted in such a way that, after adjustment, the new ranking gets closer to what it should be, i.e. C should be ranked first. The actual update algorithm in detail is not important here, what is key is the underlying idea. The latter will be presented through two characteristic cases:

- For each description D such that the term $v_{C,D}$ is greater than its vertically homologous terms in FP_1, … FP_f (viz. $v_{FP1,D}$, … $v_{FPf,D}$), W_D is increased in the following manner :

  $$w_D = (1+k)w_D$$

  Intuitively, this is based on the observation that, if a term $v_{C,Di}$ contributes more to the similarity in the correct match than in the false positive matches, it should be reinforced via an increase of its weight.

- Conversely, for each description D such that the term $v_{C,D}$ is lesser than its corresponding terms $v_{FP1,D}$, … $v_{FPf,D}$:
  W_D is decreased in the following manner :

  $$w_D = w_D / (1+k)$$

  This is based on the observation that, if a term $v_{C,D}$ contributes more to the dissimilarity in the false positive matches than in the correct match, it should be attenuated via a decrease in its weight.

The positive real number k is called *update rate*. Of course, k values close to 0 induce weak updates while higher values induce more drastic updates. In order to force model convergence (stabilization), k can also be made a decreasing function of time, tending to zero. This is similar to the temperature function used in simulated annealing algorithms.

What has just been presented is the normal, 'ongoing' interaction scenario: model update occurs based on feedback the user gives throughout successive search sessions. In addition, the user can, initially but also at any time, enter *system learning sessions* that are directed toward fast user model learning. In these sessions, instead of carrying out CBMR searches the user makes direct similarity judgments about melodic material presented by the system. In the simplest setting, the user is presented with melodic passages A, B and B' and must tell the system which of the pairs A-B and A-B' is more similar. Of course, every (A,B,C) triplets is chosen (by the system's designer) so as to emphasize the contrast between two particular descriptions. The results of the successive similarity rankings made by the user during such a learning session are finally aggregated, resulting in an appropriate update in the user's DWV.

As can be observed, the current weight update scheme uses a fixed strategy similar to error gradient feedback in neural networks. It should be remarked that other strategies such as evolutionary algorithms could be other appropriate candidates.

4. RELATED AND FUTURE WORK

The techniques proposed in this paper are currently being implemented within the Melodiscov system. Melodiscov uses a core set of object-oriented classes and methods allowing to represent music with multiple, individually weighted descriptions. Using that same representation platform, the influence of description weight adjustment has been experimented in the context of automated musical pattern extraction [3], a problem area directly connected to that of CBMR. Additionally, the outcome of the work presented in [6] may be very useful.

The current priority is on completing implementation and experimenting with the system. One future work direction concerns model initialization. Currently the initial user model is a default that is the same for every user. I wish to investigate whether more appropriate initial models could be generated for every user based on the characteristics recorded in a standard, 'general' user model. For instance, suppose that user X's general model says that X *has received significant musical education*. Stronger weights would then be given to the more abstract descriptions in user X's initial model (e.g. harmony-oriented ones) than if X was known to *have received no musical education*. While this simple example relies on common sense, more probabilistically accurate initialization strategies for user models could be designed based on statistical analysis of evolved and stabilized user models of perceived musical similarity. Another future work direction is investigating aggregation/fusion strategies for synergetically mixing the DWV-based model with more 'traditional' user models such as the ones mentioned in the first section of this paper.

5. REFERENCES

[1] Chai, Wei and Vercoe, Barry. 2000. Using user models in music information retrieval systems. Proc. International Symposium on Music Information Retrieval, Oct. 2000.

[2] Lemstrom, K. and S. Perttu. 2000. SEMEX - An efficient Music Retrieval Prototype. Proceedings ISMIR'00.

[3] Rolland, P.Y. 1999. Discovering Patterns in Musical Sequences. *Journal of New Music Research* 28:4, December 1999.

[4] Rolland, P.Y., Ganascia, J.G. 1999. Musical Pattern Extraction and Similarity Assessment. In Miranda, E. (ed.). *Readings in Music and Artificial Intelligence*. New York and London: Gordon & Breach - Harwood Academic Publishers.

[5] Rolland, P.Y., Raskinis, G., Ganascia, J.G. 1999. Musical Content-Based Retrieval : an Overview of the Melodiscov Approach and System. In Proceedings of the 7th ACM International Multimedia Conference, Orlando.

[6] Shmulevich, I., Yli-Harja, O., Coyle, E., Povel D., Lemström, K. 2001. Perceptual issues in music pattern recognition. In Rolland, P.Y., Cambouropoulos, E., Wiggins, G. (editors). Pattern Processing in Music Analysis and Creation. *Computers and the Humanities* 35(1), journal special issue.