MONOGRAFIES DE L'INSTITUT D´INVESTIGACIÓ EN INTEL·LIGÈNCIA ARTIFICIAL

Expressivity-Aware Tempo Transformations of Music Performances Using Case-Based Reasoning

Maarten Grachten

Consell Superior d´Investigacions Científiques

MONOGRAFIES DE L'INSTITUT D'INVESTIGACIÓ
EN INTEL·LIGÈNCIA ARTIFICIAL
Number 29

Institut d'Investigació
en Intel·ligència Artificial

Consell Superior
d'Investigacions Científiques

Expressivity-Aware Tempo Transformations of Music Performances Using Case-Based Reasoning

Maarten Grachten

Foreword by Josep Lluís Arcos

2007 Consell Superior d'Investigacions Científiques
Institut d'Investigació en Intel·ligència Artificial
Bellaterra, Catalonia, Spain.

Series Editor
Institut d'Investigació en Intel·ligència Artificial
Consell Superior d'Investigacions Científiques

Foreword by
Josep Lluís Arcos
Institut d'Investigació en Intel·ligència Artificial
Consell Superior d'Investigacions Científiques

Volume Author
Maarten Grachten
Institut d'Investigació en Intel·ligència Artificial
Consell Superior d'Investigacions Científiques

Institut d'Investigació
en Intel·ligència Artificial

Consell Superior
d'Investigacions Científiques

NIPO: 978-84-00-08571-1
ISBN: 653-07-094-0
Dip. Legal: B.50668-2007

Ordering Information: Text orders should be addressed to the Library of the IIIA, Institut d'Investigació en Intel·ligència Artificial, Campus de la Universitat Autònoma de Barcelona, 08193 Bellaterra, Barcelona, Spain.

Contents

List of Figures

List of Tables

Foreword

This monograph reports an investigation into the use of case based reasoning for expressivity-aware tempo transformation of audio-recorded performances of melodies. This specific task is illustrative of a wider application domain that has emerged during the past decade: content-based multimedia processing. Large scale availability of image, video, and audio information in digital format requires new ways of managing and transforming information. The work presented in the monograph is an example of such content-based transformation. The monograph investigates the problem of how a musical performance played at a particular tempo can be rendered automatically at another tempo, while preserving naturally sounding expressivity.

The work presented raises a number of challenging topics. First there is the question of data modeling. It is an open issue how expressivity information can be extracted from the performance and appropriately represented, and what aspects of the melody should be explicitly described for content-based manipulation of performed music. Secondly, from the case based reasoning perspective tempo-transformation of performed melodies is an interesting problem domain, since the problem and solution data are composite structures of temporal nature, and the domain expertise (expressively performing music) is almost entirely a tacit skill. Thirdly, since problem solving in case based reasoning is based on the reuse of previous problems, similarity measures for melodies and performances play a central role. This creates an overlap with the field of music information retrieval. Lastly, this research raises the question of how the quality of transformed performances can be evaluated. The evaluation of models for expressive music performance is an important unsettled issue that deserves broad attention.

Bellaterra, October 2007

Josep Lluís Arcos
IIIA, CSIC
email: arcos@iiia.csic.es
http://www.iiia.csic.es/~arcos

Abstract

This dissertation is about expressivity-aware tempo transformations of monophonic audio recordings of saxophone jazz performances. It is a contribution to content-based audio processing, a field of technology that has recently emerged as an answer to the increased need to deal intelligently with the ever growing amount of digital multimedia information available nowadays. Content-based audio processing applications may for example search a data base for music that has a particular instrumentation, or musical form, rather than just searching for music based on meta-data such as the artist, or title of the piece.

Content-based audio processing also includes making changes to the audio to meet specific musical needs. The work presented here is an example of such content-based transformation. We have investigated the problem of how a musical performance played at a particular tempo can be rendered automatically at another tempo, while preserving naturally sounding expressivity. Or, differently stated, how does expressiveness change with global tempo. Changing the tempo of a given melody is a problem that cannot be reduced to just applying a uniform transformation to all the notes of the melody. The expressive resources for emphasizing the musical structure of the melody and the affective content differ depending on the performance tempo. We present a case based reasoning system to address this problem. It automatically performs melodic and expressive analysis, and it contains a set of examples of tempo-transformations, and when a new musical performance must be tempo-transformed, it uses the most similar example tempo-transformation to infer the changes of expressivity that are necessary to make the result sound natural.

We have validated the system experimentally, and show that expressivity-aware tempo-transformation are more similar to human performances than tempo transformations obtained by uniform time stretching, the current standard technique for tempo transformation. Apart from this contribution as an intelligent audio processing application prototype, several other contributions have been made in this dissertation. Firstly, we present a representation scheme of musical expressivity that is substantially more elaborate than existing representations, and we describe a technique to automatically annotate music performances using this representation scheme. This is an important step towards fully-automatic case acquisition for musical CBR applications. Secondly, our method reusing past cases provides an example of solving synthetic tasks with multi-layered sequential data, a kind of task that has not been explored much in case based reasoning research. Thirdly, we introduce a similarity measure for melodies that computes similarity based on an semi-abstract musical level. In a comparison with other state-of-the-art melodic similarity techniques, this similarity measure gave the best results. Lastly, a novel evaluation methodology is presented to assess the

quality of predictive models of musical expressivity.

This research was performed at the Artificial Intelligence Research Institute (IIIA), Barcelona, Spain, and was funded by the Spanish Ministry of Science and Technology through the projects TIC 2000-1094-C02 TABASCO and TIC 2003-07776-C02 PROMUSIC.

Acknowledgments

I think behind every dissertation there is a host of people, all vital in some sense or another. The word has more than one sense:

vital adjective

1. critical, vital (urgently needed; absolutely necessary) "a critical element of the plan"; "critical medical supplies"; "vital for a healthy society"; "of vital interest"
2. vital, life-sustaining (performing an essential function in the living body) "vital organs"; "blood and other vital fluids"; "the loss of vital heat in shock"; "a vital spot"; "life-giving love and praise"
3. full of life, lively, vital (full of spirit) "a dynamic full of life woman"; "a vital and charismatic leader"; "this whole lively world"
4. vital (manifesting or characteristic of life) "a vital, living organism"; "vital signs"

(WordNet 2.1, `http://wordnet.princeton.edu/`)

For this dissertation, many people were vital in various of those senses, and I wish to express my appreciation and gratitude to all of them. First and foremost Josep Lluís Arcos, who was my primary guide and source of feedback and motivation for this research. He and Ramon López de Mántaras have given me the opportunity to work at the Artificial Intelligence Research Institute (IIIA) and suggested that I took up the Ph.D. in the first place. They have helped me a lot by suggesting or evaluating new ideas, tirelessly revising texts, having me participate in conferences, and expressing their faith in me. However, I wouldn't even have known them without Rineke Verbrugge, my M.Sc. supervisor who inspired me to work at the IIIA. The three of them were definitely vital in sense 1.

I am also indebted to Xavier Serra, who has generously given me a place to work as a collaborator at the Music Technology Group. It's a great place to be, like a second home.

Furthermore I wish to thank colleagues, fellow researchers, co-authors and (anonymous) paper reviewers who contributed to my work by their work, by giving feedback, by discussions etcetera. Emilia Gómez and Esteban Maestre were of great help, they provided audio analysis/synthesis tools and melody descriptions. Amaury Hazan, Rafael Ramirez, Alejandro García, Raquel Ros, Perfecto Herrera, Henkjan Honing, Gerhard Widmer, and Roger Dannenberg provided feedback (at some point or throughout the years) and made me rethink and refine ideas. I thank Rainer Typke, Frans Wiering, and the IMIRSEL crew (especially Xiao Hu and Stephen Downie) for the great amount of work they put in the MIREX '05 Symbolic Melodic Similarity contest. David Temperley, Emilios Cambouropoulos, Belinda

Thom, Mikael Djurfeldt, Lars Fabig, Jordi Bonada and Santi Ontañon provided programs and source code I used. Finally, I owe respect to Eugene Narmour, his work has been of great importance in my research.

Another group of people I hardly know but whose work I appreciate very much are the free software developers. Throughout the years I have used GNU/Linux as a free operating system, and grew to love it to an almost evangelical degree (to the despair of some). I should mention especially Richard Stallman, the godfather of free software, and the Debian/Ubuntu developers in general. Also, I got a lot of help from the developers of Guile.

I wish to thank my parents Harma Vos, and Henri Grachten, and my sister Gerdi Grachten. They have shown me love, kindness and generosity throughout my life. And also my grandparents and relatives, particularly Elly Tomberg, Henk van Rees, Willemien van Heck, and Wim Mulder.

And more people have been important to me, as friends, by being there and being vital (sense 3) every day, just now and then, or on a single occasion: Gabriela Villa, Hugo Solís, Áurea Bucio, Alfonso Pérez, Sylvain le Groux, Amaury Hazan, Noemí Perez, Stephanie Langevin, Raquel Ros, Alejandro García, Dani Polak, Catarina Peres, Daniele Montagner, Patrick Zanon, Ping Xiao, Fabien Gouyon, Rui Pedro Paiva, Leandro Da Rold, Hendrik Purwins, Mary DeBacker, Helge Leonhardt, Michiel Hollanders, Sjoerd Druiven, Volker Nannen, Marten Voulon, and (not least) Lieveke van Heck.

Chapter 1

Introduction

This dissertation is an investigation into the use of case based reasoning for expressivity-aware tempo transformation of audio recorded performances of melodies. This specific task is illustrative of a wider application domain of science and technology that has emerged during the past decade, and the impact and importance of which is only recently being realized: content-based multimedia processing. Large scale access to the world wide web, and the omnipresence of computers have lead to a strong increase of image, video, and audio information available in digital format, and it is only realistic to assume that in the future the majority of information will be distributed in the form of multimedia. This shift toward non-text media asks for new ways of managing information. In parallel with the technology that aims at automatic semantic description of image, video, and audio information to make its content accessible, there is a need for technology that enables us to handle such information according to its content. This includes for example content based information retrieval, but also content based transformation, where previous information is reused for new purposes, often requiring non-trivial adaptations of the information to its new context.

Both in content extraction and manipulation progress has been made (see Aigrain [1999] for an overview). Nowadays high-quality audio time stretching algorithms exist (e.g. [Röbel, 2003; Bonada, 2000]), making pitch-invariant temporal expansion and compression of audio possible without significant loss in sound quality. Such algorithms perform *low level* content based transformation, i.e. by segmenting the audio signal into transient and stationary parts based on spectro-temporal content and stretching the audio selectively. The main goal of those algorithms is to maintain *sound* quality, rather than the *musical* quality of the audio (in the case of recorded musical performances). But as such, they can be used as tools to build higher level content based audio transformation applications. A recent example of this is an application that allows the user to change the swing-ratio of recorded musical performances [Gouyon et al., 2003]. Such audio applications can be valuable especially in the context of audio and video post-production, where recorded performances must commonly be tailored to fit specific requirements. For instance, for a recorded musical performance to accompany video, it must usually meet tight constraints imposed by the video with respect to the timing or the duration of the recording, often requiring a tempo transformation.

In order to realize a tempo transformation that maintains the musical quality of the musical performance, higher level content of the audio must be taken into account (as we will

argue in the next section). Content based transformation of music performances inevitably demands a thorough grip on musical expressivity, a vital aspect of any performed music. We use the term musical expressivity to refer to the deviations of the music as it is performed with respect to some norm, for example the score. This phenomenon is notoriously complex. Changes in the musical setting (for instance changes of tempo), often lead to subtle changes in expressivity that may nevertheless be indispensable to maintain a feeling of musical correctness. Through the use of case based reasoning as a state-of-the-art AI problem-solving methodology that has proved its merits in a variety of tasks, we try to realize tempo transformations of musical performances that are expressivity-aware. This means that apart from changing the rate at which the performance is being reproduced, changes to the expressive character of the performance are made to the extent that a human musician would change her way of performing to make the performance sound good at the new tempo.

The task and chosen approach raise a number of challenging topics. First there is the question of data modeling. It is an open issue how expressivity information can be extracted from the performance and appropriately represented, and what aspects of the melody should be explicitly described for content based manipulation of performed music. Secondly, from the case based reasoning perspective tempo-transformation of performed melodies is an interesting problem domain, since the problem and solution data are composite structures of temporal nature, and the domain expertise (expressively performing music) is almost entirely a tacit skill. Thirdly, since problem solving in case based reasoning is based on the reuse of previous problems, similarity measures for melodies and performances play a central role. This creates an overlap with the field of music information retrieval. Lastly, this research raises the question of how the quality of transformed performances can be evaluated. The evaluation of models for expressive music performance is an important unsettled issue, that deserves broad attention.

In the remainder of this chapter, we explain the problem of expressivity-aware tempo-transformation in more detail (section 1.1). Then we will outline the scope and the specific problems addressed in this dissertation (section 1.2). Finally, we give a brief overview of the structure of the dissertation (section 1.3).

1.1 Musical Expressivity and Tempo

It has been long established that when humans perform music from score, the result is never a literal, mechanical rendering of the score (the so called *nominal performance*). Even when musicians intentionally play in a mechanical manner, noticeable differences from the nominal performance occur [Seashore, 1938; Bengtsson and Gabrielsson, 1980]. Furthermore, different performances of the same piece, by the same performer, or even by different performers, have been observed to have a large number of commonalities [Henderson, 1937; Seashore, 1938]. Repp [1995b] showed that graduate piano students were capable just as well as professional piano players, of repeatedly producing highly similar performances of the same piece.

Given that expressivity is a vital part of performed music, an important issue is the effect of tempo on expressivity. It has been argued that temporal aspects of performance scale uniformly when tempo changes [Repp, 1994]. That is, the durations of all performed notes maintain their relative proportions. This hypothesis is called *relational invariance* (of timing under tempo changes). Counter-evidence for this hypothesis has also been provided

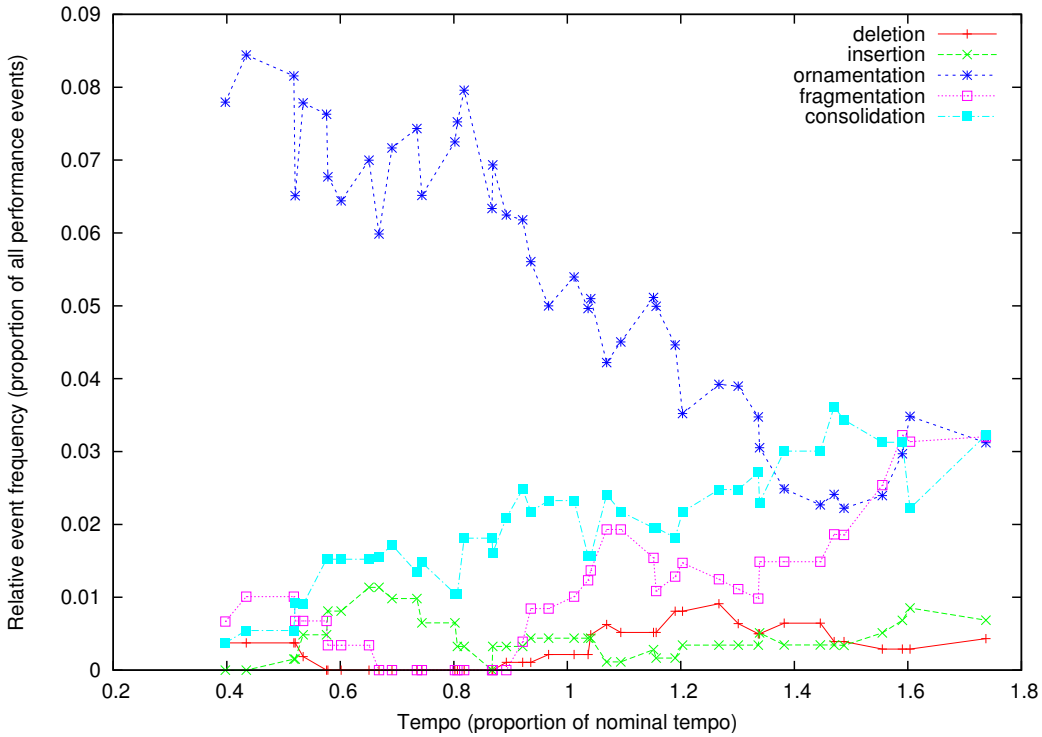

Figure 1.1: The frequency of occurrence of several kinds of performance events as a function of global performance tempo

however [Desain and Honing, 1994; Friberg and Sundström, 2002; Timmers et al., 2002], and a recent study shows that listeners are able to determine above chance-level whether audio-recordings of jazz and classical performances are uniformly time stretched or original recordings, based solely on expressive aspects of the performances [Honing, 2007].

A brief look at the corpus of recorded performances we will use in this study (details about the corpus are given in subsection 3.2) reveals indeed that the expressive content of the performances varies with tempo. Figure 1.1 shows the frequency of occurrence of various types of expressivity, such as *ornamentation* and *consolidation*, as a function of the nominal tempo of the performances (the tempo that is notated in the score). In subsection 4.3 we will introduce the various types of performance events as manifestations of musical expressivity in detail. Note that this figure shows the occurrence of discrete events, rather than continuous numerical aspects of expressivity such as timing, or dynamics deviations. The figure clearly shows that the occurrence of certain types of expressivity (such as ornamentation) decreases with increasing tempo, whereas the occurrence of others (consolidation most notably) increases with increasing tempo.

Figure 1.2 shows how various expressive parameters change systematically with tempo. The points in the figure represent comparisons of performances of the same phrase at differ-

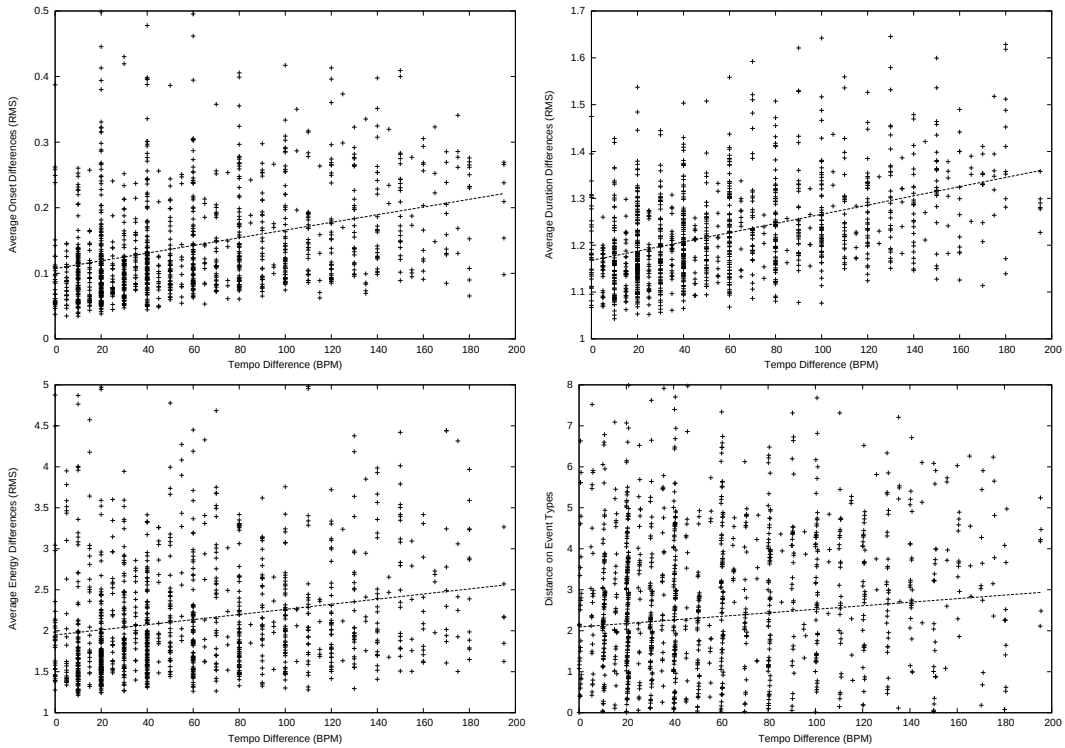

Figure 1.2: Dissimilarities between performances (of the same phrase) vs. their difference in tempo

ent tempos (tempos are specified in the number of *beats per minute*, or *BPM*). The x-axis shows the difference in tempo between the two performance. In the top left figure, the y-axis shows the *root mean square* (RMS) value of the pairwise absolute difference in note onset. The top right figure shows the RMS value of the pairwise duration difference (proportion). The bottom left figure shows the RMS value of the pairwise energy difference (proportion). The bottom right figure shows the distance value between the performances as sequences of performance events. The distance increases when the sequences contain different performance events. In all four expressive parameters, values tend to differ more when the tempos of the compared phrases increases. In some parameters the change as a function of tempo seems only small, but it must be kept in mind that the actual performances are a result of the combination of all parameters. The effects in the individual parameters are therefore cumulative.

The above observations amount to the belief that although in some circumstances relational invariance may hold for some aspects of expressivity, in general it cannot be assumed that all aspects of expressivity remain constant (or scale proportionally) when the tempo of the performance is changed. In other words, tempo transformation of musical performances involves more than *uniform time stretching* (*UTS*).

Throughout this dissertation, we will use the term UTS to refer to the scaling of the *temporal* aspects of a performance by a constant factor. For example, dynamics, and pitch will be left unchanged, and also no notes will be inserted or removed. Only the duration and onsets of notes the will be affected. Furthermore, we will use the term UTS in an abstract sense. Depending on the data under consideration it involves different methods to realize it. For example, it requires non-trivial signal-processing techniques to apply pitch-invariant UTS to the audio recording of the performance. In symbolic descriptions of the performance on the other hand, UTS consists in a multiplication of all temporal values by a constant. Note that this holds if the descriptions measure time in absolute units (e.g. seconds). When time is measured in score units (e.g. beats) UTS makes no sense, since changing the tempo of the performance only changes the translation of score time units to absolute units of time.

1.2 Problem Definition, Scope and Research Objectives

In this section we describe the main problem we address in this dissertation. We define the scope of the project with regard to the type of musical data and the level of processing. After that we will list the secondary objectives that derive from the main problem.

1.2.1 A System for High-level Content-Based Tempo Transformation

As mentioned at the beginning of this chapter, the primary objective of this research is to develop an expressivity-aware system for musical tempo transformations of audio recorded performances, that maintains not only the *sound quality* of the recording, but also *musical quality* of the performance. There are several ways to concretize the criteria for success of the system. For instance, we can say the tempo-transformation of a recorded performance is successful if:

- its expressive characteristics (statistically speaking) are in accordance with human performances at that tempo (more than with human performances at other tempos);
- it is preferred by human listeners over a tempo-transformed performance that was obtained by UTS;
- it is not recognized by human listeners as being a manipulated performance; they regard it as an original recording of a human performance;

Although there is not a strictly logical relation between the above criteria, we feel that in practice, each former criterion is implied by the subsequent criteria. That is, they are ordered from weak to strong. In subsection 2.4.6 we review evaluation paradigms for expressive music prediction models in general, and in subsection 6.3 we propose a hybrid evaluation approach that combines human judgment with a quantitative assessment of tempo-transformed performances.

1.2.2 Scope

In order to optimize the conditions for investigation of expressivity aware tempo transformation and to make the problem feasible as a project, the scope of research must be chosen appropriately. We work with a corpus of recorded performances that has some rather specific characteristics, that make it suitable for this research. Firstly, the recordings are performances of jazz standards, taken from The Real Book [2004]. Jazz is a good genre for studying expressivity because of its emphasis on liberal and expressive performance rather than precise reproductions of notated melodies. Secondly, the performances are played by a saxophone, an instrument that offers a very broad range of sound-qualities that allow a skilled performer to perform expressively in an elaborate manner. Thirdly, the melodies are monophonic, which relieves the need for voice separation, and lets us focus directly on the expressivity in the performance. Finally, the type of expressivity in the recording is what we call 'natural': the melodies are performed as the performer (a professional musician) thinks they should without any explicit intentions of expressing a particular mood or affection. See section 3.2 for more details on the musical corpus.

In addition to the focus on a specific type of musical data, we focus on a particular level of data processing. As explained before, we will work within the paradigm that separates content extraction from content analysis/manipulation, as opposed to for example purely signal processing approaches to audio-transformation[1]. The research presented in this dissertation exclusively addresses content analysis/manipulation. For the content extraction (audio analysis) and reconstruction of audio from content descriptions (audio re-synthesis), we rely on an external system for melodic content extraction from audio, developed by Gómez et al. [2003b,a].

1.2.3 Representation of Expressivity

One of the secondary objectives is the development of a suitable representation scheme for the expressivity in the data that we work with. This will require a study of the expressive phenomena encountered in the musical corpus.

1.2.4 Melodic Retrieval Mechanisms

The case based approach to the tempo transformation problem implies the need for a retrieval mechanism for cases that is based on the melodic material contained in the cases. We will investigate and compare several approaches to melodic similarity.

1.2.5 Performance Model Evaluation

In order to evaluate the proposed tempo transformation approach we need an evaluation methodology that assesses the quality of tempo transformed performances. We will discuss common evaluation paradigms.

[1] This two-level paradigm has been argued for by e.g. Scheirer [1995]

1.3 Outline of the Dissertation

In chapter 2, we provide the relevant background of the work presented in this dissertation. We discuss the principal concepts involved, notably musical expressivity, performance models and their evaluation, melodic similarity, and case based reasoning. Along with this survey we will review related research, methodologies, and notable findings.

In chapter 3 we propose a system architecture for the tempo transformation system. We define the terms that will be used, and summarize the goal and functionality of the different components. Furthermore, we detail the musical corpus used for experimentation, we present various representation schemes of the musical data, and discuss their value for the current application.

In chapter 4 we explain in detail the methods we have developed to process the input data, in order to form cases that contain not just the raw input data, but provide higher level knowledge-enriched data descriptions that interpret and interrelate the input data, and are used at various stages throughout the case based reasoning process.

Chapter 5 is devoted to the problem solving process, that consists in applying case based reasoning to construct a tempo transformed performance of the input phrase.

Chapter 6 describes a set of experiments that validate the different components of the system, notably knowledge acquisition (performance annotation), and abstract melody representation, and melody retrieval. The final section of the chapter describes an experimental validation of the system as a whole.

In chapter 7 we summarize the research presented in the dissertation. We list the main contributions of our work, and identify future work.

The appendices respectively contain abbreviations/notational conventions, annotated scores of the phrases used as musical corpus to form the case base, a list of songs used for a melodic similarity comparison experiment, and a list of publications by the author.

Chapter 2

Preliminaries and Related Work

In this chapter we provide the context in which the present research is situated. The first part of the chapter is dedicated to musical expressivity, providing a description of the phenomenon of expressivity, its communicational functions, and alternative definitions (sections 2.1, 2.2, and 2.3). Then, we discuss various methodologies for developing computational models of expressivity, and alternative ways of evaluating such models (section 2.4). After that, we briefly survey research on melodic similarity and different computational approaches to assess melodic similarity (section 2.5). In section 2.6 we introduce case based reasoning and give a characterization of CBR with respect to other learning approaches. We conclude the chapter with some observations and concluding remarks on the reviewed work (section 2.7).

2.1 Expressivity in Musical Performances

Investigations into the performance of music dates from at least the end of the nineteenth century and early twentieth century. For example, around 1896, Binet and Courtier [1896] prepared a grand piano to display the dynamics of the keys pressed on a paper. In 1913, Johnstone [1913], observed that in piano performances, the notes that belong to the melody are often played slightly earlier than chord notes at the same metrical position. Around 1930, extensive research on music performance was carried out and reported by a group of researchers led by Seashore [1938]. This research was mainly concerned with singing, violin, and piano performances.

The performance of a musical piece is determined by several factors. Firstly, physical conditions of the musician and her instrument are of influence. Obviously, the type of instrument determines to a large extent the character of the performance. Also, physiological conditions of the musician (such as fatigue, or state of health) can play a role. Secondly, the motor skills of the musician are of importance. This becomes clear when comparing the performances of a novice to those of a skilled musician. With practice, the musician trains her motor speed and accuracy, reducing the amount of unintentional deviation from performance to score. A third factor consists of the cognitive, and affective aspects of the musician. It has been shown by Sloboda [1983] that performers deviate systematically from the score when they play variations of the same score that consist of exactly the same sequence of notes

(only their placement within the meter was changed). This result rules out the possibility of deviations due to motor incapacities and shows the influence of meter on the performance of a score. Other studies have shown systematic deviations in performances that were played with different moods or expressive intentions [Rigg, 1964; Gabrielsson, 1995; De Poli et al., 2001].

2.2 Functions of Expressivity

As far as performance deviations are intentional (that is, they originate from cognitive and affective sources as opposed to e.g. motor sources), they are commonly thought of as conveying *musical expressivity*. But what is it that is being expressed? Two main functions of musical expressivity are generally recognized. We will address both functions.

2.2.1 Expressivity and Musical Structure

Firstly, expressivity serves to clarify the musical structure (in the broad sense of the word: this includes metrical structure, but also the phrasing of a musical piece, harmonic structure etc.). Sloboda [1983] showed the influence of metrical structure on performances by having pianists perform the identical sequences of notes, differing in the position of the bar lines. For instance, the pianists varied their performances such that the notes at the beginning of measures were played louder and more legato than other notes. Furthermore, he observed that the more advanced the performers were, the more they utilized this kind of expressivity, and the better listeners were able to transcribe the performed music correctly. This is a clear indication that expressivity clarifies metrical structure.

Phrase structure also has a salient effect on performance. Phrases were found to start and end slow, and be faster in the middle [Henderson, 1937]. Moreover, Todd [1985, 1989] invented a model that predicts the level of rubato of a given musical piece, given a hierarchical grouping structure of the piece. The predictions of this model are similar to the rubato patterns in professional performances of the piece. Gabrielsson [1987] found that pianists performing Mozart's Piano Sonata K. 331, tended to lengthen note durations considerably at the end of phrases. Similarly, he found the tones to be relatively loud in the middle of the phrases and relatively soft at the beginning and end of the phrase.

Another form of structure that influences performance is harmonic and melodic tension. Harmonic and melodic tension are commonly defined by reference to the circle of fifths (where the tension is low for notes or chords that are close together on the circle of fifths and high for those that are far apart) [Lerdahl, 1996]. Palmer [1996] calculated a positive correlation between note lengthening and tension. Contrastingly, no correlation was found between note intensity and tension. In the same article, Palmer showed that *melodic expectancy*, the extent to which an implied continuation of a melody is actually realized, did correlate with note intensity (unexpected notes were played louder), but not with note lengthening. As an explanation of the fact that the expression of tension-relaxation and melodic expectancy are realized in unrelated ways, Palmer observes that the two phenomena manifest themselves on different time scales; tension-relaxation is a phenomenon at the level of phrases and sub phrases (that is, a large time scale), whereas melodic expectancy is manifested from note to note, i.e. on a smaller time scale.

In a general survey of the relation between expressivity and musical structure, Clarke [1988] proposes the interesting view that expressivity is tied to structure by a limited set of rules (like the rules proposed by Sundberg et al. [1991]; Friberg [1991], see subsection 2.4.3). Hence, the diversity of ways in which a piece can be played is not due to ambiguous rules for expressivity, but due to the diversity of ways in which the music can be structurally interpreted (i.e. ambiguous musical structure). In this context, Clarke notes the practice of the live performances of jazz standards. In this practice, the music that is played belongs to a widely known and fixed repertoire. Therefore the audience is usually acquainted with the music, and the expressiveness of the performance is not constrained by the requirement that it should clarify the basic musical structure to the listeners. The musicians can thus freely vary their expressiveness to surprise the audience through an unexpected musical interpretation of the piece.

Another structural aspect of music that has been found to influence musical expressivity is the melody. In ensemble performance (but also polyphonic piano performances), the voice that plays the melody tends to be slightly (by around 20–30 ms.) ahead of the accompaniment [Rasch, 1979; Palmer, 1988]. The purpose of *melody lead* is presumed to be the avoidance of masking of the melody by the accompaniment, and to facilitate voice separation in human perception.

In a study of performances of jazz melodies by well-known jazz musicians, Ashley [2002] has shown that the patterns of rubato tend to be related to motivic structure. He also found that note displacement was related to the underlying harmony (chord tones are more displaced than non-chord tones).

For more extensive surveys of music performance research in relation to structural aspects see [Friberg and Battel, 2002; Clarke, 1991; Palmer, 1997].

2.2.2 Expressivity and Emotional Content

Secondly, expressivity serves as a way of communicating, or accentuating affective content. Langer [1953] proposed the view that the structure of music and the structure of moods or feelings are isomorphic. Langer observes that similar properties are ascribed to music and emotional life (such as 'excitation' and 'relief'). Another influential theory about music and meaning (which subsumes emotion) is from Meyer [1956]. He states that meaning (be it emotional, or aesthetic) arises in music when expectations raised by the music are not realized. Early work investigating the relation between emotional character and musical structure is reviewed by Rigg [1964]. Some typical regularities were found, e.g.: solemn music is often played slow, low pitched and avoiding irregular rhythms and dissonant harmonies; happy music is played fast, high pitched, it contains little dissonance and is in major mode; exciting music is performed fast and loud and apt to contain dissonance [Gabrielsson, 1995] (another mapping between emotional characters and musical cues can be found in [Juslin, 2001]). Gabrielsson [1995]; Lindström [1992] have studied the relation between emotional intentions and micro structure in music (e.g. timing deviations, intensity changes and articulation). They compared versions of "Oh, my darling Clementine", played with emphasis on different emotional characters ('happy', 'sad', 'solemn', 'angry', 'soft', 'expressionless'). Their results with regard to the overall properties tempo and loudness were mainly in accord with previous results: angry and happy versions were played faster than sad, soft, and solemn versions; angry versions were played loudest. With respect to the micro structure they also found

clear differentiation between the different versions, Notably the variance of articulation and loudness. Duration ratios in the rhythm were also different for different emotional characters (the performers were given great freedom of performance, only the pitch sequence was to be kept intact).

Similar results have been found by Canazza et al. [1997a], who have studied how physical parameters of recorded performances (e.g. timbre, and temporal parameters like articulation and global tempo) affected by varying expressive intentions of the musician. The performer was told to communicate a particular mood through his performance, that was described by a sensorial term, e.g. 'heavy','light','bright', or 'dark'. The sonological analysis of the recordings made it possible to tie particular parametric values to particular moods (for instance a light performance turned out to be in fast tempo, with shortened note durations and soft attacks). The results were validated by performing listening tests on synthetic performances that were generated with the physical parameter values corresponding to particular moods. The subjects were able to recognize the intended mood in the synthesized performances.

In a related perceptual analysis of listener judgments of expressed moods in musical performances [Canazza et al., 1997b], Canazza et al. found that there is a large degree of consistency between the mood ratings of the listeners. They concluded that tempo and note attack time are two important factors by which listeners rank the sensorial terms (as those mentioned above). Moods like 'heavy' were opposed to 'light' and 'bright' on the scale of tempo, whereas 'soft', 'dark' and 'hard' were distinguished from each other on the scale of attack time.

See [Gabrielsson, 1999, 2003] for more general reviews on music performance research, including the role of expressivity in the communication of emotions.

2.3 Definitions of Expressivity

Until now, we have used the term expressivity in performances loosely as deviations or irregularities that occur when a score is performed by a musician. But deviations or irregularities with respect to what? Several definitions have been proposed [Timmers and Honing, 2002]:

Expressivity as Microstructure (EM) Firstly, there is the definition of 'expressivity as micro structure' [Repp, 1990; Clynes, 1983]. This definition conceives of expression as everything that is left unspecified in the score. This definition does not view expressivity strictly as deviations from a standard. Rather, it holds that the score only specifies the macro structure of the music, leaving undecided how the low level attributes of the macro elements should be realized. To quantify this microstructure however, at least with regard to the timing of notes, the difference with nominal values is computed [Repp, 1995a]. As a result, although conceptually different, this definition is in practice equivalent to the following definition of expressivity.

Expressivity as Deviation from the Score (EDS) The second, and most common definition is 'expressivity as deviation from a musical score'. This definition was employed already in early music performance research [Seashore, 1938], and was stated more explicitly by Gabrielsson [1987]. In this definition expressivity is regarded as the deviation from the norm dictated by notation, in terms of intrinsic note attributes like pitch, timbre, timing, and dynamics.

In spite of its intuitive appeal, this definition has been criticized [Huron, 1988] for stemming from a exaggerated distinction[1] between content and form. Another criticism is that knowing the score is not indispensable for listeners to appreciate expressiveness [Desain and Honing, 1991].

Expressivity as Deviation within the Performance (EDP) As an alternative to the EDS definition Desain and Honing proposed to define expressivity as deviation within a performance [Desain and Honing, 1991]. More precisely, 'expression is the deviation of a lower order unit from the norm as set by a higher order unit' [Timmers and Honing, 2002]. This definition thus assumes a hierarchical description of the musical piece. For example, the deviation of a beat duration can be related to the deviation of the duration of the enveloping bar.

Although we do acknowledge the problems with EDS, we believe their impact is limited. For example, many popular and jazz music has a fixed tempo accompaniment, that steadily marks the beat. This means that the timing deviations in the melody will be perceived with respect to this beat, and effectively, with respect to the nominal score. In other words, in restricted contexts, particularly where the melody has a regular background, and the (performance) style is familiar to the listener, the listener's conception of the melody she hears may correspond closely to the score. As a consequence, the definition of expressivity as deviations from the score may (in such contexts) be an acceptable approximation of a more sophisticated definition of expressivity.

However, we signal another problem with the usual interpretation of EDS in the research that we have encountered. The problem is that the deviations with respect to the score are defined on a note-to-note basis. We have found this interpretation of expressivity to be too limited to accurately represent the deviations encountered in actual musical performances that we use in the research presented in chapters 3 to 6. Therefore, we propose an extension to the standard note-to-note fashion of computing performance deviations from the score (section 3.6).

2.4 Computational Models of Expressivity

A natural step in the study of expressive music performance research is the development of computational models of expressivity. In a general manner of speaking such models are intended to express and clarify the structure and regularities that occur in musical expressivity. In particular, they seek to establish a relation between the form that expressivity takes and the musical context. These aspects include characteristics of the melody such as phrase structure and meter. A number of such models have been proposed in the past decades. Due to the richness and complexity of the subject, the proposed models are typically limited to a few aspects of expressivity, or to a particular musical style. Apart from differences in area of focus, there is also divergence in methodologies. In the rest of this section we briefly discuss various methodologies, and highlight their corresponding research and models.

[1] Huron argues that this strict separation may arise from the Western tradition of division of labors between composers and performers.

2.4.1 Common Prerequisites

Every methodology for constructing models of expressivity has certain requirements, especially with regard to data:

Recordings of Performances The most essential part of nearly all methodologies is the availability of recorded performances. Such data are commonly gathered through one of the following methods:

Audio Recording The most straight-forward approach to gathering performance data is to record the acoustical signal that the instrument emits when the musician plays. The result is a (discretized) continuous audio signal.

MIDI Recording Performance data can also be gathered in MIDI format, when the musician plays a MIDI instrument. The output of the digital instrument is a non-continuous series of events symbolically describing the states of the instrument.

Recording of Augmented Instruments When there is no satisfactory MIDI implementation of the instrument, a solution is to augment the acoustical instrument with sensors that capture for example finger or lip pressure (depending on the type of instrument). In this way, detailed and multi-channel information can be gathered in a non-intrusive way[2].

Recorded performances are not strictly necessary for the analysis by synthesis approach, which in turn relies strongly on synthesis of performances. See [Goebl et al., 2005] for a more complete survey of performance data acquisition methods.

Annotation / Extraction of Expressivity It is uncommon that recorded performances are used directly for the modeling of expressivity. Usually, an extensive and often manual processing of the data is necessary to annotate the performed data so that the information of interest (e.g. dynamics and timing deviations per note) is easily accessible. This step requires most effort when the performance is recorded as audio. Automatic pitch and onset detection algorithms (e.g. [Goto, 2004; Gómez et al., 2003b; Klapuri, 1999]) can be of help to obtain a description of the performed melody, after which a symbolic performance-to-score alignment can be performed [Desain et al., 1997; Arcos et al., 2003]. Integrated melodic extraction and score alignment are also common (e.g. [Dannenberg and Hu, 2003; Orio and Schwarz, 2001]).

Musical Expertise Especially the analysis by synthesis relies on the a priori postulation of hypotheses in the form of rules for applying expressivity based on the score. Since the space of possible hypothesis is so large, musical intuition and experience is indispensable to guide the generation of hypothesis. In other approaches the role of musical expertise is less explicit, but the choice of representation schemes for score and performance/expressivity is critical for a successful model and requires a certain degree of musical knowledge as well.

[2] Very few professional musicians feel comfortable with a MIDI implementation of their instrument, since the tactile characteristics, the 'feel' of the instrument is very distinct. This holds notably for grand piano vs. MIDI keyboard.

2.4.2 Analysis by Measurement

The analysis by measurement approach in essence tries to construct models for expressive phenomena that are revealed by measuring the expressivity from a set of recorded musical performances. A common way to construct models is to hypothesize a particular relation between some aspects of the score and some type of deviation (in for example the timing of the notes) and test the statistical significance of the hypothesis on the data.

Many expressive effects have been addressed individually. Perhaps most widely studied are expressive deviations of performed notes in terms of timing[3], dynamics, and to a slightly lesser degree articulation. Todd has proposed a simple model for the relation between timing and dynamics stating that notes are played 'the faster, the louder' and vice versa [Todd, 1992]. Subsequent research has proved that this relation is too simplistic to account for a wider range of observations [Windsor and Clarke, 1997]. In [Drake and Palmer, 1993], timing, dynamics and articulation were found to accentuate melodic turns (peaks or valleys in the melodic contour), and rhythmic grouping. Accentuation through these variables is obtained by playing louder, delaying notes, and playing staccato, respectively. In a study of Beethoven Minuet performances by 19 famous pianists, Repp found a large amount of common variation in expressive timing, in part as a response to phrase ending, and melodic inflections [Repp, 1990]. He provided (weak) counter-evidence for the existence of a 'composer's pulse' [Clynes, 1983] (see next subsection).

Special attention has been paid to the timing of musical ornaments (that is, notated ornaments in classical music). It was shown that the timing of ornaments is of a different nature than the timing of regular notes [Timmers et al., 2002]. In particular, the timing of grace notes scales to a lesser degree with tempo than ordinary notes.

A notable case of analysis by measurement is a mathematical model for rubato [Todd, 1985, 1989]. In accordance with other studies [Palmer, 1989; Repp, 1995a], Todd argues that rubato, the pattern of speeding up and slowing down during a performance, is at least partly determined by the phrase structure of the music. To demonstrate this, he designed a model that predicts a rubato curve based on the time-span reduction of a musical piece [Lerdahl and Jackendoff, 1983] (see subsection 3.4.1). The system maps parabolic functions to the different subtrees of the global time-span reduction tree, and adds these functions to obtain a final rubato curve. Apart from phrase structure, other contextual factors of the music also have an effect on rubato, such as metrical structure and the presence of a second melody [Timmers et al., 2000].

A specific form of rubato is the *final ritard*, the slowing down that frequently occurs at the end of a piece. Several studies [Todd, 1995; Friberg and Sundberg, 1999] establish an analogy between the final ritard and the motion of physical bodies, arguing that their dynamics is very similar. This kinematic model of the final ritard has been criticized for not taking into account any musical context, in particular rhythm [Honing, 2003]. Honing et al. argue that a model for the final ritard might rather take the form of a set of constraints than that of a mechanical model, despite its elegance.

Timmers has fit a regression model to predict the subjective quality ratings of continued performances based on factors including the rubato, dynamics, and articulation of the initial part of the performance [Timmers, 2003].

[3]It should be noted that the term timing in some studies is used to refer either to incidental changes of the onset/offset of individual notes, to the local tempo at which the melody is performed, or to both.

2.4.3 Analysis by Synthesis

One approach to automatic performance of music is to construct a set of performance principles allowing for the reconstruction of real expressive performances. That is, a grammar that describes the structure of musical expression in terms of the musical score. This approach has been taken by Friberg [1991]; Sundberg et al. [1991]. They have defined a set of context-dependent rules. These rules prescribe small deviations for timing and dynamics of notes, based on their musical context. They can act in ensemble to sequentially process a sequence of notes, to synthesize an expressive performance. In this way, the validity of the rules can be checked, either by judging the musical acceptability of the synthesized performance, or by rating the quantitative difference of expressive deviations from a human performance of the same piece of music. This approach is called *analysis-by-synthesis*, referring to the procedure of analyzing musical expressivity by synthetically mimicking it. In this approach, the factors contributing to musical expressivity are validated by judging their effects on performances. The rules have been incorporated into a software application called *Director Musices* (*DM*), which processes MIDI files to generate expressive performances.

The rules were developed with the help of a musical expert (L. Frydén, a music performer and music teacher). The method of rule development is as follows: An intuition by the expert about the effect of a particular aspect of a musical score on the performance of that score, is translated into an initial rule. This rule is used to synthesize an 'expressive' MIDI performance of the score. The expert judges the result and adjusts his instructions if the result is unsatisfactory. Thus, performance rules evolve in an iterative process of synthesis, judgment, and adjustment. The rules affect the duration, overall sound level (dynamics), time envelope of the sound level (articulation), vibrato speed and depth, and fine tuning of the pitch. The rules have a conditional part that specifies the target notes to which the rule applies. Furthermore, each rule has a corresponding quantity parameter, specifying how large the effect of the rule should be (i.e. a scaling factor for the deviations prescribed by the rule).

An interesting hypothesis that has lead to further experimentation is that emotional colorings of performances may correspond to specific parameter settings for the rule set. Subjects were able to recognize intended emotions by listening performances that were generated with a specific 'rule palette' to suggest a particular emotion (e.g. anger, tenderness, sadness etc) [Bresin and Friberg, 2000].

Clynes has argued that timing of classical pieces is in part dependent on the composer. For a number of famous composers from the Classical and Romantic periods (e.g. Beethoven and Mozart), he established a 'composer's pulse', a pattern of timing and dynamics deviations that was characteristic for authentic performances of the works of that composer [Clynes, 1983, 1995]. He applied an analysis by synthesis approach to evaluate different combinations of pulses and composers.

2.4.4 Automatic Rule Induction

The development of machine learning techniques for discovery of knowledge from data [Mitchell, 1997] has also had an impact on expressive music performance research. Techniques such as *inductive logic programming* [Muggleton, 1991; Quinlan, 1990], and *decision tree learning* [Quinlan, 1986; Mitchell, 1997] have proved useful for deriving predictive, and

to some extent explanatory, models for music performance [Widmer, 2000]. Widmer has derived a number of performance principles from a large set of piano performances of Mozart sonatas. The models are typically in the form of rules that predict expressive deviations based on local score context. An advantage of this method over analysis by synthesis is that no prior hypotheses have to be specified, rather the hypotheses are extracted from the data. This circumvents a possible bias in the generation of hypotheses due to limitations in the ability of musical experts to verbalize their musical knowledge. On the other hand, another kind of bias may arise, namely bias due to the form of representing the musical data, and the representation language of extracted hypothesis. Another drawback of this method is that score/expressivity regularities are only detected if they are statistically significant, although infrequent (and thus statistically insignificant) expressive phenomena may still be meaningful[4].

Model generation using inductive algorithms is subject to a trade-off: on the one hand *predictive* models can be derived to maximize predictive power. Such models provide a high predictive accuracy due to the induction of a large set of highly complex rules, that are generally unintelligible. On the other hand, the induction of *descriptive* models involves a conservative approach to the introduction of rules, trying to keep the complexity of rules as low as possible, at the cost of reduced accuracy.

A specialized algorithm for learning rules from data, *PLCG*, was proposed by Widmer [2003]. This algorithm is oriented towards finding simple and robust classification rules in complex and noisy data. This is achieved by first learning many rules that cover a subset of the data, and clustering the learned rules so that similar rules (syntactically and semantically) are in the same cluster; then for each cluster of rules a new rule is formed, generalizing the rules in the cluster. From the resulting set of rules, the rules that have the highest percentage of correct predictions are selected. In this way the obtained rule set consists of a modest number of rules with a reasonable degree of generality.

Ramirez and Hazan [2004] have compared k-nearest neighbor, support vector machines, and tree-learning (C4.5) techniques for predicting the durations of notes in jazz performances (using the classes 'lengthen', 'shorten', and 'same'). They used the same set of performances as used in this dissertation, and also included the Implication-Realization analysis of the melody (see subsection 3.4.2) as part of the data instance representation. Some rules with general validity were induced that linked the timing of notes to the I-R structures. Structural descriptions of the melodic surface as part of the instance representation were also used for rule induction by Widmer [1996]. In another study, Hazan et al. [2006a,b] have proposed an evolutionary generative regression tree model for expressive rendering of melodies. The model is learned by an evolutionary process over a population of candidate models. The quality of rendered melodies was computed both as the root mean square error and tuned edit-distance to a human target performance of the melody.

A machine learning approach has also been adopted in an attempt to classify the expressive style of various professional pianists [Goebl et al., 2004; Widmer, 2002]. Performances were first represented as curves in the local tempo/dynamics plane. Those curves were segmented and the segments were clustered. The curve segments of the most important

[4]This limitation of inductive approaches in the context of expressive music performance research was mentioned by Johann Sundberg, during the Music Performance panel, held at the MOSART workshop, Barcelona, 2001

clusters were collapsed into prototype curve-segments. This yielded an alphabet of characteristic curve-forms that was used to represent a performance as a string of symbols from the alphabet. Standard pattern matching techniques are employed subsequently to discover commonalities in the performances of the same pianist.

2.4.5 Case Based Reasoning

An alternative technique to the automatic rule induction technique described above is case based reasoning (CBR). It has in common with rule induction that it uses example data to predict expressivity for unseen melodic material, but CBR employs a different strategy: Instead of generating a set of rules in an analysis phase, it uses the most similar examples (called cases) directly to process new melodic material. We explain the case based reasoning process in more detail in section 2.6. Here we present related case based reasoning approaches to the generation of musical expressivity.

SaxEx [Arcos et al., 1998] is a CBR system that generates expressive music performances by using previously performed melodies as examples. The work presented in this document springs from and extends this previous work. Hence, we will address some of the topics involved (especially CBR and musical models) in more detail in subsequent sections, and in the following overview of SaxEx, we will mention those topics only briefly, referring to the relevant sections for further details.

The goal of SaxEx was to transform inexpressive performances into expressive performances, allowing user control over the nature of the expressivity, in terms of expressive labels like 'tender', 'aggressive', 'sad', and 'joyful'. The input to the system is a musical phrase in MIDI format, and the audio recording of inexpressive performance of the phrase to be played expressively. To obtain appropriate values for the expressive features of the performance, case based reasoning (see section 2.6) is used. Thus, a case base is employed, containing examples of expressively played phrases. For each note of the input phrase, a set of similar notes is retrieved from the case base. This retrieval step employs different *perspectives* on the phrases. One perspective used in SaxEx refers to the musical analysis of the phrase, using music theories such as the Implication-Realization model and GTTM (see subsection 3.4.2). Such analyses are useful to determine the role of notes in their musical contexts and can be used to assess similarities between notes. Another perspective is constructed using the desired expressive character of the output performance. This requirement is specified in advance by the user, on three bipolar scales (tender-aggressive, sad-joyful, and calm-restless, respectively). Each pole on these scales is associated with particular expressive values. In this way, the affective requirement perspective is used to bias the performances in the case base that meet those requirements.

This inexpressive performance sound file is analyzed using spectral modeling synthesis techniques (SMS) [Serra, 1997] yielding a parametric description of the audio. The parameters, that correspond to expressive features like note-onsets, dynamics, and vibrato, are changed according to the expressive feature values derived from earlier examples by CBR and the sound is re-synthesized with the new parameter-values using SMS, resulting in an expressive performance.

Note that although strictly speaking, SaxExperforms a *transformation* of a performance, it uses the input performance only as a basis for the re-synthesis. The CBR process does

include expressive features as a part of the case descriptions. On the *expressive* level the task of SaxExis therefore performance *generation* rather than *transformation*.

Another example of a CBR system for expressive music performance is *Kagurame*, proposed by Suzuki [2003]. This system renders expressive performances of MIDI scores, given performance conditions that specify the desired characteristics of the performance. *Kagurame* employs the edit-distance for performance-score alignment, but it discards deletions/insertions and retains just the matched elements, in order to build a list of timing/dynamics deviations that represent the performance. Furthermore, its score segmentation approach is a hierarchical binary division of the piece into equal parts. The obtained segments thus do not reflect melodic structure. *Kagurame* operates on polyphonic MIDI, and the expressive parameters it manipulates are local tempo, durations, dynamics, and chord-spread.

Tobudic and Widmer have taken a k-nearest neighbor approach (akin to CBR) in order to predict timing and dynamics deviations for Mozart sonatas [Tobudic and Widmer, 2003]. They constructed a hierarchical phrase analysis for every melody and, for every level in the hierarchy, a polynomial function is fitted to the timing and dynamics curves. The lower level polynomials are fit to the residual curves of fitting higher level polynomials. Training instances are defined to contain phrase constituents of the melody, with attributes describing overall attributes of the melody fragment, for example the phrase length and start-end pitch interval. The distance between a training and a target instance is defined as the inverse of Euclidean distance between the attribute vectors, and the solution (the parameters of the polynomial tempo and dynamics curve). The initially poor prediction results could be improved by averaging the solutions over 10 nearest neighbors instead of adopting the solution from the single nearest instance as is. The results were also improved by taking into account only the lowest three levels. Finally, results were improved by additionally employing the PLCG (see subsection 2.4.4) algorithm to derive rules that account for the residual dynamics and timing curves after subtracting all polynomials. Tobudic and Widmer have extended this approach by replacing the attribute-value instance representation scheme to a first order predicate representation scheme [Tobudic and Widmer, 2004]. The distance measure for instances was a maximal matching set distance. In particular, the distance between two literals was computed as before (i.e. the Euclidean distance between the arguments of a predicate), and sets of literals were compared by computing the maximal match between literals and adding a constant cost for each unmatched literal in either of the sets.

2.4.6 Evaluation of Expressive Models

An important question in computational modeling of musical expressivity is how models should be evaluated. Clearly, the evaluation depends on the aims of the model.

Descriptive or explanatory models are intended to explain expressive variations of the performance by hypothesized principles. In accordance with the criteria for evaluating scientific theories in general, they should have explanatory power (account for as much variation as possible) and be parsimonious at the same time. As mentioned before, expressivity is thought to arise from many different factors, such as communication/expression of emotions/affects, and various forms of musical structure. Many existing models cover only a single aspect. This has lead Desain and Honing to propose an evaluation technique that consists in fitting complementary partial models for expressivity jointly to performance data

[Desain and Honing, 1997]. Alternative (sets of) models can then be compared by the amount of variation in the performance data they explain after optimization. Although theoretically appealing, this approach relies on some strong assumptions, like the assumption that the partial models jointly represent all sources of expressivity, and that a poor explanation of variance by a model is not caused by sub-optimal parameter settings.

Perceptual or cognitive models, on the other hand, describe the cognitive processes involved in listening to expressive performances, such as the emergence of perceptual categories, for example in rhythm perception, or the mental association of rubato to physical motion. Such models may be validated by testing hypotheses derived from the model. Hypotheses could for example predict subject response times to recognition tasks or particular phenomena in EEG signals.

Most relevant for this dissertation however is the evaluation of predictive models, that predict deviations of note attributes such as timing and/or dynamics, and rubato, given the score. The evaluation of such models is most naturally based on the prediction result of the model. Generally two distinct evaluation approaches can be identified. Firstly, one can measure how different the predicted performance is quantitatively from a target performance. The prediction accuracy is thus used as an indicator of the model quality. In this apparently simple approach, some issues must be addressed:

What is the right target performance? Although expressive performance research throughout the years has revealed some clear regularities in the form of expressivity, it is generally acknowledged that musical expressivity is not completely determined by such regularities, and that there is a considerable degree of freedom for the performer to play a melody in a variety of musically acceptable ways, that can differ substantially in quantitative terms. To evaluate a predicted performance for a given score by comparing it to just a single target performance can therefore be overly restrictive. An accurate prediction for this particular target performance is of course a satisfactory result, but when the prediction is not accurate, there may still be another (unknown) target performance that it resembles.

How to evaluate partial models? Another problem when assessing models by comparing their results to a real performance is that most models of expressivity only model a small subset of the possible sources of expressivity. That is, the predicted performance contains only the expressive deviations that arise from the modeled sources (e.g. metrical structure). However, any performance played by a human musician is bound to contain expressive deviations that arise from a much wider range of sources. The resulting pattern of expressive deviations is constituted by overlaying the patterns due to the individual sources.

A second approach to evaluating predictive expressive performance models is to rate how well generated performances sound in itself. The advantages of such an approach are that the arbitrariness of choosing a target-performance is avoided, and also that the evaluation does not depend on a numerical comparison between expressive performances that may not be representative for human perception. It does introduce new questions however:

Subjectivity The rating of the 'quality' of a performance is inherently a subjective task. The rating subjects may have different notions of the concept of quality, and it is unclear how to interpret lack of consensus in the ratings of different subjects. A panel of subjects who discuss their opinions to form consensus may resolve the latter problem.

Which subjects? A related question is which subjects should be chosen. There is evidence that at least for some musical listening tasks, results diverge for trained and untrained

musical listeners [Lamont and Dibben, 2001; Canazza et al., 1997b; Granot and Donchin, 2002]. Differences in rating may arise from different levels of musical background knowledge, or differences in selectivity of perception.

The Rendering Contest (Rencon) [Hiraga et al., 2004] is an annual event started in 2002 that strives to evaluate computer generated performances by human judgment. Contestants participate by rendering a given musical piece (semi) automatically using a predictive performance model. In a loose sense the contest employs a *Turing test* paradigm. That is, a successful system is one that renders performances in such a way that it is believed to be a human performance. In practice, since most automatically rendered performances are still easily discerned from human performances, the contestant performances are ranked according to listener preference. The judgments are made by the audience of the event. The ranking of the contestant performances is established by voting. The contest has a compulsory section, where a small number of fixed musical pieces (piano pieces by Mozart and Chopin) must be performed. The rendered expressive performance is delivered in MIDI format, which is synthesized to audio using a high quality sampled piano instrument. An open section was also added to the contest to allow for the participation of a wider range of models, for example models that focus on the rendering of monophonic melodies, or that render expressivity for other instruments than piano, e.g. human voice or wind instruments. The comparison of contestant performances of different pieces may be problematic because the listener's aesthetic perception of the performance is likely to be influenced mainly by the piece rather than the way it is performed.

To summarize, the evaluation of expressive performance models is a complex topic, that allows for different perspectives. We have pointed out two major evaluation strategies and some problems that they must address. In part these are practical problems such as the problem of gathering reference data, and the incompleteness of models. Although rendering systems exist for diverse types of music and instrumentation, the number of participating systems in the Rendering Contest is not yet large enough to establish fixed compulsory sections for different musical styles and instruments. Other problems are more conceptual. For example, should models be quantitatively or qualitatively assessed, and if they are assessed quantitatively, what is the point of reference for comparison?

2.5 Melodic Similarity

The assessment of similarity between melodies is a topic that has wide interest, partly due to its commercially relevant applications such as music recommender systems, composer attribution, and plagiarism detection. In music analysis, and music processing systems that involve comparison of musical material, such as the one presented in this dissertation, melodic similarity is also an important issue.

Despite its acknowledged importance, the concept of melodic similarity is hard to pinpoint, firstly because it is of an intrinsically subjective nature, and secondly because melodic similarity is strongly multi-faceted, implying that two melodies can be similar in one respect, but dissimilar in another. Because of this it is difficult to speak of melodic similarity in a universal and unambiguous manner.

Nevertheless, useful research has been done on the topic, including theoretical, empirical, and computational studies. For example, Deliège [1997] proposed a theory of the cognitive

processes underlying human similarity judgments of music. She states that when humans hear music, they abstract from what they hear by means of emphcues. She defines a clue as 'a salient element that is prominent at the musical surface' (Deliège [2001], page 237). This concept is akin to Gestalt psychology, since it conceives of the perception of music as having a background (the normal flow of music) and a foreground (salient elements in this flow). Memorization is realized through the cues, which are the parts of the music that are stored in the listener's long-term memory. The cues serve as a 'summary' of the music.

The similarity judgment between two fragments of music, according to Deliège, is based on the similarity (or identity) of clues. This is implied by the fact that music is memorized as clues, since similarity judgments involve the listeners memory of the music (the fragments of music compared are not heard at the same time, and may not be heard at the moment of judgment at all). In particular, the perception of different musical fragments as *variations* of each other, is thought to be caused by the fact that the fragments give rise to identical clues. In this way, the concept of clues can form the basis of a theory of categorization of music, where musical motifs can belong to the same category based on the perceptual cues they provide, with varying degrees of prototypicality.

Lamont and Dibben [2001] have performed listening experiments to investigate melodic similarity from a perceptual point of view. They are interested in questions such as: Do listeners perceive similarity while listening? Is perceived similarity based on deep, thematic relations between the musical fragments, or rather on shared surface attributes of the music? Do listeners employ the same similarity criteria across different styles of music? Are the employed criteria the same for listeners with different levels of musical experience? The listening experiments involved both trained musicians and non-musicians as subjects. They were asked to rate the similarity of pairs of musical extracts. Based on the results, Lamont and Dibben concluded that similarity perceptions in general (i.e. across musical styles) are based on surface features such as contour, rhythmic organization, texture, and orchestration. Remarkably, they concluded that 'deep' structural relationships, that are often regarded as determining for the similarity of musical motifs, do not seem to have a significant impact on listeners judgments on similarity of musical motifs.

Scheirer [2000] has measured perceived similarity for short polyphonic audio fragments. He uses a multiple regression model to predict listeners' similarity ratings based on psychoacoustic features of the audio. In contrast to the previous experiment, this experiment also takes into account non-melodic features of the music, such as orchestration.

A variety of computational methods exist to quantify the degree of similarity between two melodies. We will summarize a selection of methods:

Edit-Distance A method based on string-matching techniques used in other fields such as genetics, and text-mining (see section 4.2). The edit-distance has proved useful in a variety of computer music applications such as score-following [Dannenberg, 1984], performance error-detection [Large, 1993], and pattern-detection [Rolland, 1998]. It has also been used to compute melodic similarity, e.g. for melody retrieval or melodic clustering [Mongeau and Sankoff, 1990; Smith et al., 1998; Lemström and Ukkonen, 2000; Grachten et al., 2005a, 2004b].

Earth-Mover-Distance Typke et al. [2003] apply the Earth-Mover-Distance (EMD) [Rubner et al., 1998] to compare melodies. The EMD can be understood metaphorically as conceiving the notes of two melodies under comparison as heaps of earth and holes respectively, where the size of the heaps and holes represents note attributes like duration and pitch. The distance is proportional to the minimal effort it takes to move the heaps of earth into the holes.

n-Grams n-Grams based melody comparison rates the distance between melodies by segmenting both of the melodies under comparison into overlapping segments of n notes (n-grams), and counting the number of segments the two melodies have in common. The distance is proportional to this number [Doraisamy and Rüger, 2003; Orio, 2005; Suyoto and Uitdenbogerd, 2005].

Compression Cilibrasi et al. [2003] have introduced a distance measure to the area of musical similarity that is based on compression. The core idea is that two musical fragments are similar to the extent that one fragment can be efficiently compressed by using the other fragment. Ideally, the efficiency of compression would be measured by the *Kolmogorov complexity* of the fragments. This quantity is equal to the length of the smallest string of bits that allows exact reconstruction of one melody from the other. Because it is effectively uncomputable, existing compression algorithms such as 'bzip2' are used as an approximation.

Hybrid Approaches Different existing melodic similarity algorithms can be employed in ensemble to form a final distance measure. Frieler and Müllensiefen [2005] linearly combine over 50 melodic distances, and tune the linear weights of each of the distances according to ground truth data.

2.6 Case Based Reasoning

Case based reasoning (CBR) is the process of problem solving using the solutions of previously solved problems. It refers both to the cognitive process of problem solving, and to a formal method that can be used for a variety of tasks such as supervised learning (e.g. classification, or regression tasks), design, and planning. The basis for CBR as a formal method is a model of dynamic memory by Schank [1982]. Henceforth, we will use the term CBR in the latter sense, unless otherwise specified.

Problem solving by CBR is achieved by retrieving a problem similar to the problem that is to be solved from a case base of previously solved problems (the cases), and using the corresponding solution to obtain the solution for the current problem. From this it becomes clear that CBR is based on the general assumption that if two problems are similar, their solutions will also be similar. In many domains this is a reasonable assumption, since solutions are often related to the characteristics of problems in some systematic way. Of course this does not imply that the relation between problem and solution is easily captured explicitly e.g. in terms of rules.

CBR can be regarded as a special form of *instance based learning* (IBL), a 'lazy' machine learning technique where the problem solving task is performed by using individual instances

of training data, as opposed to 'eager' learning techniques that induce general knowledge from the complete set of training data and use that knowledge to solve problems [Mitchell, 1997]. We will go into this distinction in more detail in subsection 2.6.2.

Instance based learning is typically used for classification and regression tasks, and in such applications, instances are usually represented as a vector of numerical features, where the distance between instances is often the (weighted) sum of differences between features. CBR on the other hand is characterized by the use of a richer representation of the instances. Features may have non-numeric attributes, such as symbols, text, or references to other cases, and cases may contain more than just the information that defines a problem and its solution. Meta-information may also be included, for example an indication of the quality of the solution, time-stamps (in case the domain is dynamic, and valid solutions to problems at one time are not necessarily valid at another), a description of the steps that were taken to arrive at the solution, or a justification of the solution in terms of domain-knowledge. As a result, similarity computation between cases can involve complex comparisons. In this comparison, general domain-knowledge may also be used (for example in the form of rules). Due to its representational richness, CBR lends itself for other tasks than classification, such as query/answer systems [Kolodner, 1983], structural design [Hinrichs, 1992], diagnosis [Koton, 1988], and legal reasoning [Ashley, 1990].

CBR systems can be categorized according to the type of problem solving task they address: *analytical*, or *synthetical* tasks [Plaza and Arcos, 2002]. In analytical tasks there is a limited number of solutions and solutions are simple, non-aggregate entities (classification/diagnosis is a typical analytical task). In synthetic tasks, the solutions have a composite structure, and as a result the number of possible solutions is usually very large. A typical example of a synthetic task is structural design.

The concept of problem solving using CBR is illustrated in figure 2.1, where problem solving is conceived of as mapping points on a problem plane to points on a solution plane. The transition from problem to solution plane is made via a (possibly multiple) neighboring problem, for which the solution is known. Relating the unsolved problem to a nearby solved problem is referred to as retrieval, and proposing a solution based on the nearby solution is referred to as reuse, as explained in the next subsection.

2.6.1 The CBR Cycle

The CBR problem solving process is commonly described as a cycle of four principal steps [Aamodt and Plaza, 1994]:

Retrieve Pick earlier solved problems from the case base, that are similar to the problem to be solved currently.

Reuse Adapt (or construct) an initial solution for the current problem, based on the solutions of the retrieved problems.

Revise Test the proposed solution for adequacy (typically by user feedback) and if necessary revise the solution to meet the requirements.

Retain Store the problem, together with its revised solution, in the case base for future use.

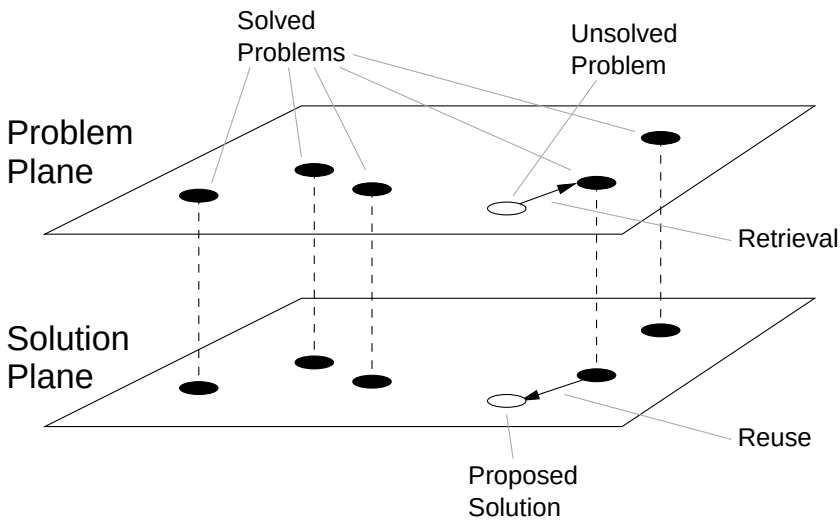

Figure 2.1: Problem solving in CBR.

Figure 2.2 shows how the four CBR tasks are organized within the problem solving process. For generating a solution, a CBR system must at least provide methods for the Retrieve and Reuse tasks.

In the retrieve step the input problem is compared to the cases in the case base (which contain a problem and a solution). This requires a distance measure that ranks cases according to their similarity to the input problem. Typically a similarity threshold is established in advance to filter out all cases that are not similar enough to serve as examples for solving the input problem. The distance measure is a crucial component of the CBR system, because it should ensure that the 'similar problems have similar solutions' assumption holds. To achieve this, a distance measure can be chosen and/or optimized in order to fit a known 'real' similarity relation between cases, or alternatively, to fit a *utility* function that expresses how well one case serves to solve another case [Smyth and Keane, 1998; Stahl, 2004]. Both the distance measure and the threshold value usually require domain-specific knowledge. The sensitivity of the distance measure to problem features should be proportional to the influence of the features on the form of the solution. The threshold value comprises knowledge of the range of variation that is tolerable between cases without their solutions becoming incomparable.

The objective of the reuse step is to propose a solution for the input problem based on the solutions of one or more retrieved problems. More specifically, the task is to determine how the solution changes as a function of (relatively small) changes in the problem. Like the retrieve step, the reuse step usually involves knowledge of the problem domain. Depending on the problem solving task, adaptation can range from very simple to very elaborate operations. CBR systems for classification usually employ a simple reuse strategy, such as a majority voting among the retrieved cases to predict the class of the input problem. In synthetical tasks on the other hand, the solution is not a nominal value, but rather a composite structure.

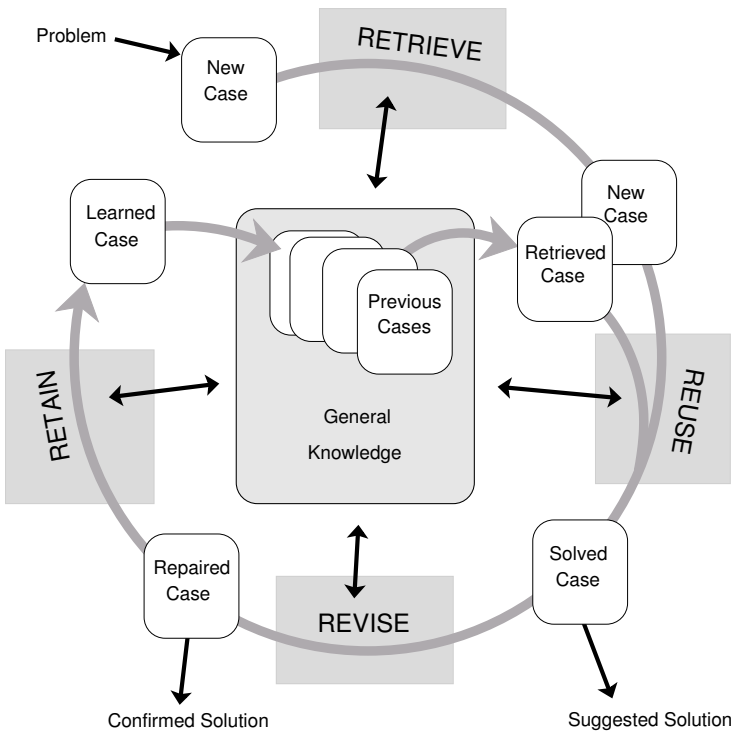

Figure 2.2: The Case Based Reasoning cycle, which shows the relations among the four different stages in problem solving: Retrieve, Reuse, Revise, Retain.

Reuse in such tasks may consist in adjustment of parameters in the structure, or substitution of sub-structures. Like the retrieve step, reusing retrieved cases typically requires domain knowledge, for example in the form of adaptation rules that specify how to modify a solution given certain conditions of the input problem and the retrieved problem.

The solution derived by the CBR system through retrieval and reuse is suggested as an initial solution to the input problem. This solution is then assessed for its effectiveness, normally by the user that queried the CBR system with the input problem. Testing effectiveness can be done by trying out the solution in the real world, although in some domains this may be too dangerous or costly (e.g. in decision support for oil well drilling [Skalle and Aamodt, 2004]). Based on this feedback, revisions are made to the case to improve the quality of the solution. This is the revise step.

Finally, in the retain step, if the revision of the solution is successful, that is the revised solution is of sufficient quality, the solution is stored together with the input problem as a new case in the case base. In this way the case can be used for solving future problems. One aspect of case retention is the actual storage of the case, possibly including case indexing for faster retrieval. Another aspect is case base maintenance. Depending on the actual and desired characteristics of the case base, a case may be either stored or not. For example, if the case base size must be minimized due to memory constraints, a case that introduces redundancy will be left out, implying that the solution must be derived again if the system is queried with same problem. If the cost of adaptation is high (e.g. due to computational complexity or time constraints), the case base redundancy introduced by the new case may be an acceptable trade-off.

The revise and retain steps enable the CBR system to learn incrementally. That is, the system can learn to solve new problems one at the time, without the need to have all training data available at the same time.

2.6.2 CBR Characterization

As mentioned before in this section, CBR is a special form of IBL. This approach, which is called *lazy learning*, is different from *eager learning* approaches such as inductive logic programming or decision-tree learning, in that the induction step from seen instances to unseen instances is postponed until a new unseen instance is presented, whereas eager-learning approaches generalize from the training data without regard to any particular problems to be solved. The advantage of postponing generalization is that the training data remains accessible with all of its detail. If a new problem must be solved that is very different from the seen data, inevitably a strong generalization must be made that may result in a low accuracy, but if on the other hand a problem similar to the seen data must be solved, little generalization is required and the system may benefit from the detailed data. Contrastingly, an eager learning system a priori determines a single generalization of the data. In order for this generalization to be broad enough to cover problems that are different from the seen data, accuracy is sacrificed for problems that are similar to the seen data. Stated otherwise, lazy learning approaches construct many local approximations of the target function[5], rather than a single global approximation. This generally allows for more complex target functions to be learned.

[5] the function that maps problems into solutions

Knowledge Intensiveness	
Low	High
No explicit general knowledge	Substantial general knowledge
Many cases	Few cases
Cases are data records	Cases are user experiences
Simple case structures	Complex case structures
Global similarity metric	Sim. assessment is explanation
No adaptation	Knowledge-based adaptation

Table 2.1: Characterization of IBL/CBR systems in terms of knowledge intensiveness (adapted from Aamodt [2004])

Aamodt and Plaza [1994] note that in CBR, as opposed to most other problem solving approaches, it is possible to employ both specific knowledge about individual cases, and general knowledge. The degree to which general domain knowledge is employed may vary, leading to a scale of knowledge-intensiveness on which CBR approaches can be located [Aamodt, 2004]. Characterizations of the low and high ends of the scale are given in table 2.1.

A common critique is that problem solving using CBR is opaque, that it does not *explain* the obtained solution to the user. It is argued that learning techniques that build general models on the other hand do explain the solution, since the model (e.g. a decision tree), allows the user to trace how the solution was obtained. We feel that this argument is not convincing. Firstly, we believe that like induced general models, cases can provide the user with an intuition of why a particular solution is predicted. Just like learning a new task is often easier when examples are given than when just the rules are summed up, providing a case to the user that is known to be related to the input problem, may give her insight into which aspects of the problem (by comparing input and retrieved problem) are relevant for the solution, and where the solution differs as a function of the differences between the problems. Secondly, the level of insight the user gets from a model is likely to be related to the complexity of the model, rather than the type of model. Tracing how an expressive performance was obtained through a decision tree that has thousands of branches will not in any way clarify the principles that underlie expressive performance. Neither will a CBR system with overly specific cases. On the other hand, models that have an adequate instance representation, and are not over-specified, can be transparent both as eager and as lazy learners.

2.7 Conclusions

From the review of the different types of models of expressivity in music performance we observe that most modeling approaches belong to either of two opposed approaches. One approach can be roughly described as top-down. It starts from the concepts that are assumed to (partly) determine the expressivity (like phrase structure, or emotions/affects), and investigates how these concepts manifest themselves through expressivity. The other approach can be called bottom-up. It does not interpret expressive information primarily as conveying underlying concepts, but rather takes it as data of which the syntactical form is to be discovered. Both approaches have their merits and limitations. The top-down approach,

although it may give more insight to the nature of expressivity, is unlikely to account for detailed and low level aspects of expressivity. It typically derives more general rules such as: 'Sadness is conveyed by playing softly and legato' [Juslin, 2001]. On the other hand, the bottom-up approach typically accounts for detailed forms of expressivity without putting them in an interpretative framework (e.g. 'Ascending melodic lines are played faster than descending lines' [Friberg, 1991]). In order to provide a more complete picture of expressivity, existing models may have to be integrated into a general model.

It also becomes clear from the survey of related work is that most studies are devoted to expressive performance *generation* rather than performance *transformation*. In generation the focus is on how expressive deviations are related to the score alone (or other forms of musical structure), whereas performance transformation deals with an initial performance that already contains expressive deviations. These deviations may play a role when deciding how to play the new performance.

Furthermore, although research on jazz music performance exists, many of the predictive models are based on classical music rather than jazz. This, together with the performance practice in classical music that tends to follow the score more strictly than jazz performance, may account for the fact that many predictive models are restricted to predicting dynamics and tempo curves. In jazz performance practice it is common that the score is interpreted loosely, as an outline of the melody rather than a complete specification of it. Even in performances of melodies that would not be classified as improvisations or variations on the melody (which depart even more from the score), phenomena like note ornamentations, fragmentations, and consolidations frequently occur, as pointed out in section 1.1. To deal with such forms of expressivity, a richer representation of expressivity is required. In section 3.6 we propose a representation scheme for these forms of expressivity.

We have also concluded that the evaluation of predictive models is a complex topic and that both of the approaches explained have their drawbacks. Although we do not pretend to solve all evaluation questions, in this dissertation we present a novel hybrid approach to evaluating performance models that combines human subject ratings with a computational distance measure between performances (see section 6.3).

As for the method of modeling expressivity, we consider case based reasoning a promising approach for application for several reasons. Firstly, expressive music performance apparently involves a large amount of background knowledge, but musicians hardly employ the knowledge in an explicit form (e.g. as rules). CBR is an appropriate problem solving method for such domains as it applies cases directly, without the need to infer explicit knowledge from them. Secondly, an advantage over analysis-by-synthesis that CBR shares with inductive learning approaches is that the range of expressive effects is not limited to a set of rules that human musical experts can verbalize (as discussed in subsection 2.4.4) . Rather it will use the expressive effects that actually occur in the training data. The other side of the coin is that for inductive learning approaches to pick up a certain expressive regularity, it must be statistically significant in the training data, whereas Sundberg (see footnote 4) argued that this is not a necessary condition for an expressive effect to be musically significant. CBR on the other hand does not suffer from this drawback.

Chapter 3

The TempoExpress System Architecture and Data Modeling

In this chapter we present TempoExpress, a case based reasoning system for applying global tempo transformations to monophonic audio recordings of saxophone jazz recordings. The system deals with tempo transformations as problems to be solved using a set of previously solved tempo transformations. In line with the case based reasoning process discussed in the previous chapter, the previously solved problems are not used to learn a general model for applying tempo transformations, but rather they are stored as cases in a case base and only relevant cases are retrieved and used to solve particular problems.

The focus of the problem solving is on the musical aspects involved in the tempo transformation, as opposed to the signal processing aspects required to render the tempo-transformed performance into audio.

In chapters 4 and 5 we will present the main components of TempoExpress in more detail. The present chapter is intended to give an overview of the system, introduce the key concepts involved, motivate design decisions, and present the models to represent the input data, such as scores and performances. The chapter also includes a section that provides details about the musical corpus we used to construct cases.

3.1 A Global View of the System

The structure of the system is shown schematically in figure 3.1. The complete process of tempo-transformation of audio is divided into an audio analysis/synthesis part and a content manipulation part. The audio analysis/synthesis part is responsible for the derivation of a melodic content description from audio, and vice versa. This part falls outside the scope of this dissertation. The content-manipulation part is the part that is embodied by the CBR system we have named TempoExpress. Its task is to revise the content description of the source audio in such a way that the audio, when resynthesized according to this revised description, will be at the target tempo and have expressive features that are in accordance with that tempo.

The input data of TempoExpress consists of:

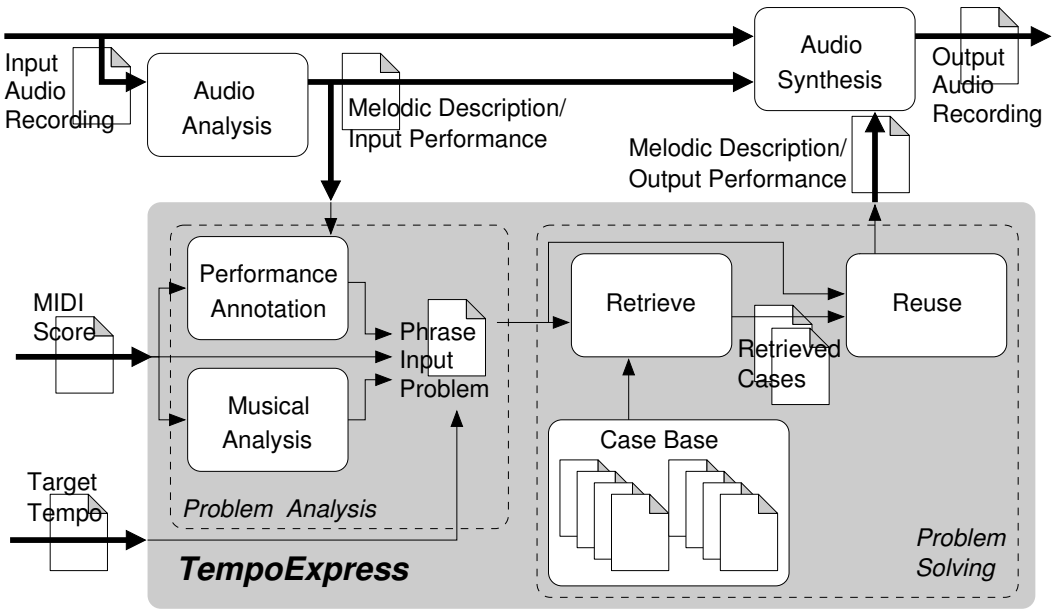

Figure 3.1: Diagram of the major components of TempoExpress

1. a *melodic description* of the input audio recording of which the tempo is to be transformed;
2. a *MIDI score* with the nominal melody that is played;
3. a *target tempo*, the tempo (in BPM) at which the recorded performance should be rendered;

The *melodic description* (MD) is a formatted description of the melody as it is detected in the input audio recording, containing both a frame-by-frame description of the audio (with descriptors like fundamental frequency candidates, and energy), and a segment-by-segment description of the audio. The audio segment descriptions correspond to the individual notes detected in the audio, and apart from their begin and end time (i.e. note onset and offset) they include mean energy (dynamics), and estimated fundamental frequency (pitch) as descriptors. The method for deriving MD's from monophonic audio recordings is described in more detail in [Gómez et al., 2003a; Maestre and Gómez, 2005].

The result of transformation is a revised MD of the *output audio recording* to be generated. The difference between the revised MD and the original MD define the changes that must be made to the original audio to obtain the output audio.

3.1.1 Problem Analysis

This subsection provides an overview of the topics that will be covered in more depth in chapter 4.

Within TempoExpress, again two major sub tasks can be identified, in the figure indicated by the boxes 'problem analysis', and 'problem solving'. The first task is to build a *phrase problem specification* from the given input data. This is a data structure that contains all information necessary to define a tempo transformation task for a musical phrase, and additional information that will improve or facilitate the problem solving. A phrase problem specification contains the following information:

1. a *MIDI score*, the score as a sequence of notes;
2. a *musical analysis*, an abstract description of the melody;
3. a *source tempo*, the tempo (in BPM) of the input performance;
4. a *target tempo*;
5. an *input performance*, the performance as a sequence of performed notes;
6. a *performance annotation*, a description of the expressivity in the performance; and
7. a list of *segment boundaries*, that indicates how the score of the phrase is divided into segments.

As figure 3.1 shows, only items (1) and (4) of this list are directly provided by the user. The musical analysis (2) is derived from the MIDI score and contains information about various kinds of structural aspects of the score, like metrical structure, an analysis of the melodic surface, and note grouping. The phrase segmentation (7) is also derived from the MIDI score, and is intended to capture the musical groupings inherent in the phrase.

The performance annotation module takes the MIDI score and the MD of the input audio recording as input and provides the source tempo (3), the input performance (5), and the performance annotation (6). The source tempo (3) is estimated by comparing the total duration of the audio (time in the MD is specified in seconds) and the duration of the MIDI score (which specifies time in musical beats). Although this way of estimating the global tempo is simple, it works well for the data we used [1]. The input performance (5) is a symbolic representation of the performed notes, with MIDI pitch numbers (estimated from the fundamental frequency), duration, onset, and dynamics information. This information is readily available from the MD. To facilitate comparison between the performed notes and the MIDI score notes, the duration and onset values of the performed notes are converted from seconds to beats, using the computed source tempo. Finally, the performance annotation (6) is computed by comparing the MIDI score and the input performance.

We will refer to the phrase problem specification that was built from the input data as the *phrase input problem*, being the problem specification for which a solution should be found. The solution of a tempo transformation will consist in a performance annotation. The performance annotation can be interpreted as a sequence of changes that must be applied to the score in order to render the score expressively. The result of applying these transformations is a sequence of performed notes, the *output performance*, which can be directly translated to an MD at the target tempo, suitable to be used as a directive to synthesize audio for the transformed performance.

[1] Tempo estimates computed for 170 performances have a mean error of 0.2 BPM and a standard deviation of 1.1 BPM

3.1.2 Problem Solving

This subsection provides an overview of the topics that will be covered in more depth in chapter 5.

In a typical CBR setup, the input problem is used to query the case base, where the cases contain problem specifications similar in form to the input problem, together with a solution. The solution of the most similar case is then used to generate a solution for the input problem as a whole. In the current setting of music performance transformation however, this approach does not seem the most suitable. Firstly, the solution is not a single numeric or nominal value, as in e.g. classification, or numeric prediction tasks, but it rather takes the form of a performance annotation, which is a composite structure. Secondly, melodies are usually composed of parts that form wholes in themselves (a phrase is typically composed of various motifs). The first observation implies that solving a problem as a whole would require a huge case base, since the space of possible solutions is so vast. The second observation on the other hand suggests that a solution may be regarded as a concatenation of separate (not necessarily independent)[2] partial solutions, which somewhat alleviates the need for a very large case base, since the partial solutions are less complex than complete solutions.

This has lead us to the design of the problem solving process that is illustrated in figure 3.2. The phrase input problem is broken down into phrase segment problems (called *segment input problems*, or simply *input problems* henceforward), which are then solved individually. The solutions found for the individual segments are concatenated to obtain the solution for the phrase input problem. However, a preliminary retrieval action is performed using the problem at the phrase level. The goal of this preliminary retrieval is to set up the case base for the segment-level problem solving, from what we will call *proto cases*. Proto cases are information units that contain phrase related information like the MIDI score, the musical analysis, the segmentation boundaries, and all performances (with varying global tempos) available for that phrase. The case base is formed by pooling the segments of the selected proto cases, hence the number of cases M it will contain depends on the selectivity of the preliminary retrieval, and the number of segments per phrase: If C is the subset of proto cases that were selected during preliminary retrieval, and s_i the number of segments in the i^{th} proto case from C, then the case base size is:

$$M = \sum_{i=1}^{|C|} s_i$$

The case base obtained in this way contains *cases*, consisting of a segment problem specification and a solution at the segment level. The cases contain the same type of information as the input problem specifications and solutions at the phrase level, but they span a smaller number of notes. Solving the phrase input problem is achieved by searching the space of partially solved phrase input problems. A partially solved phrase input problem corresponds to a state where zero or more segment input problems have a solution. A complete solution is a state where all segment input problems have a solution. Solutions for the segment input

[2]For example, the way of performing one motif in a phrase may affect the (in)appropriateness of particular ways of playing other (adjacent, or repeated) motifs. Although such constraints are currently not defined in TempoExpress, we will explain in section 5.3 how the reuse infrastructure can easily accommodate for this.

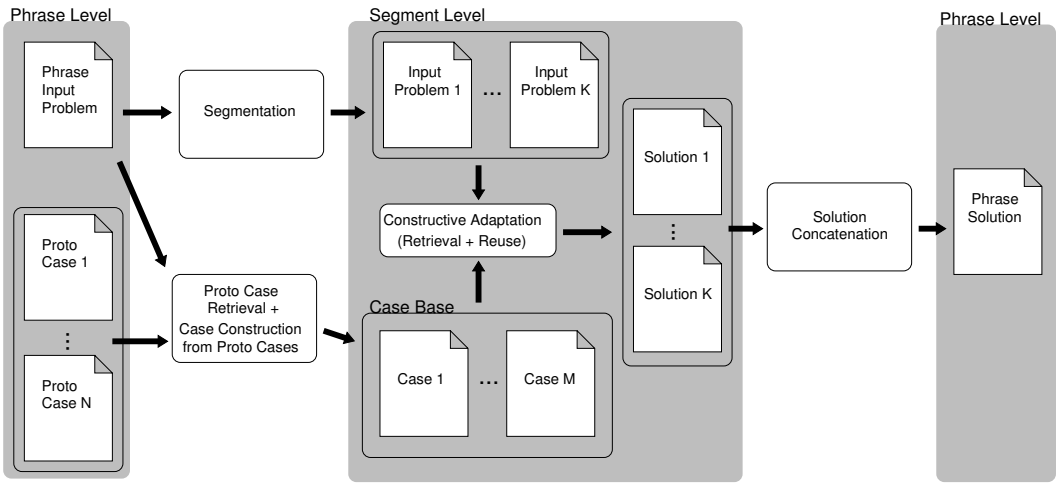

Figure 3.2: The problem solving process from phrase input problem to phrase solution

problems are generated by adapting retrieved (segment) cases. This technique for case reuse is called *constructive adaptation* [Plaza and Arcos, 2002].

The expansion of a state is realized by generating a solution for a segment input problem. To achieve this, the retrieve step ranks the cases according to similarity between the MIDI scores of the segment input problem and the cases. The reuse step consists of mapping the score notes of the retrieved case to the score notes of the input problem, and using this mapping to 'transfer' the performance annotation of the case solution to the input problem.

3.2 Musical Corpus

In this section we present the musical material that was used for the tempo-transformation experiments reported in this dissertation. We had at our disposal a set of monophonic saxophone recordings, which were made in the context of Tabasco[3], a research project focusing on content-based audio transformation.

Four different songs were recorded using professional recording equipment. The performing artist was a professional jazz saxophone player. Every song was performed and recorded at various tempos. One of these tempos was the nominal tempo. That is, the tempo at which the song is intended to be played. This is usually notated in the score. If the nominal tempo was not notated, the musician would determine the nominal tempo as the one that occurred most natural to him. The other tempos were chosen to be around the nominal tempo, increasing and decreasing in steps of 5 (in slow tempo ranges) or 10 BPM (in faster tempo ranges). About 12 tempos per song were recorded. The musician performed on his own, accompanied by a metronome indicating the global tempo of the piece. In total 170 interpretations of phrases were recorded, amounting to 4256 performed notes. Table 3.1

[3]CICYT Grant TIC 2000-1094-C02 28-12-2000/27-12-2003

Title	Composer	Song Structure	Tempo Range (BPM)
Body and Soul	J. Green	A A B1 B2 A	35 – 100
Like Someone in Love	Van Heusen/Burke	A B1 B2	65 – 260
Once I Loved	A.C. Jobim	A B C D	55 – 220
Up Jumped Spring	F. Hubbard	A1 A2 B	90 – 270

Table 3.1: Songs used to populate the case base

shows the top level phrase structure of the songs (determined manually from the score) and the tempo range per song.

The musician was instructed to perform the music in a way that seemed natural to him, and appropriate for the tempo at which he was performing. Note that the word natural does not imply the instruction play in-expressively, or to achieve 'dead-pan' interpretations of the score (that is, to imitate machine renderings of the score). Rather, the musician was asked not to strongly color his interpretation by a particular mood of playing.

The songs were manually split up into phrases [4] (in the case of jazz standards, the phrase structure of a song is often easily determined by looking at the score, if not annotated explicitly), and the recordings of each song were accordingly split up into phrases.

3.3 Melody Representation

The history of computer music comprises a wealth of melody representation schemes [Selfridge-Field, 1997], some being oriented towards graphical notation, others to for example efficient data storage, or real-time usability. According to their purpose, they represent different aspects of the music. For example, they may or may not convey information about timbre, dynamics, or articulation. The bare minimum of information that will always be present in some way or another (it can be considered as the essence of the melody), is the information conveyed by *piano-roll notation*: pitch, duration, and temporal position of the notes. The temporal position may sometimes be implicitly defined as the summed duration of the previous notes/rests. This is also the principal information present in traditional Western music notation, where pitch is conveyed by the vertical position of the note heads on the staff, and duration by the graphical form of the note. Another aspect of the melody that is represented in traditionally notated music is meter. The melody is normally preceded by a time signature, and the notes of the melody are accordingly separated by barlines indicating the beginning/end of each bar. Because time signatures by convention have a fixed pattern of accents, this implicitly defines the metrical strength of each note in the melody.

The notation of classical music usually contains additional hints for performance. For example, *crescendo/decrescendo* signs may indicate dynamical evolution, and articulation may be indicated by slurs and dots. In jazz however (notably in The Real Book [2004], the primary resource of notated jazz music), such expressive directions are usually absent. Another difference with the notation of classical music is that the melody is typically tran-

[4] see appendix B for score representations of the individual phrases

scribed monophonically, and harmony is not notated using notes on the staff, but is instead annotated using chord-labels that describe the degree and tension of the chords. This alleviates the problem of defining/separating melody and harmony in polyphonically transcribed music.

Our focus in this dissertation will be on jazz music. Hence, given the above observations, we will assume that a melody representation in the form of a sequence of note elements that includes pitch, duration, onset, and metrical information for each note captures all essential of the score. The reason for not including further expressive hints into our melody representation scheme is that the task of the system we propose is performance *transformation* rather than performance *generation*. That is, the aim of the system is not to generate an expressive performance from a nominal melody representation, but rather to manipulate expressive effects present in one performance, in order to obtain a performance with different characteristics (in our case a different tempo). The source of the expressivity is therefore the input performance rather than expressive hints in the score.

3.4 Musical Models

The melody representation presented above is commonly used in computational systems dealing with music, probably because it is the most obvious and straight-forward scheme. Although this representation conveys the most essential melodic information, it leaves implicit a number of aspects of melody that will be of importance in musical tasks, such as melody comparison, pattern finding, and segmentation of melodies. Aspects not present in the note sequence representation are for example: metrical structure, (sub) phrase structure, and the development of harmonic tension. These aspects are all emergent properties of melody as a perceived whole, rather than a sequence of notes.

3.4.1 GTTM

A number of theories exist that describe how such higher level constructs arise from sequences of notes. The most well-known of these is GTTM [Lerdahl and Jackendoff, 1983], a theory that models the evolution of harmonic tension (*prolongational reduction*) and the relative importance of notes (*time-span reduction*) within a melody. This theory is based on systematic musicology rather than cognitive studies. A more recent approach provides a framework for extracting a variety of musical structures like meter, phrase structure, and key [Temperley, 2001]. The theory is accompanied by *Melisma*[5], a set of algorithms that implement the theoretical framework.

3.4.2 The Implication-Realization Model

Another, strongly cognitively motivated model for musical structure is the Implication-Realization (I-R) Model [Narmour, 1990, 1992]. This model tries to explicitly describe the patterns of expectations generated in the listener with respect to the continuation of the melody. It applies the principles of *Gestalt Theory* to the melody perception, an approach

[5]URL: `http://www.link.cs.cmu.edu/melisma/`

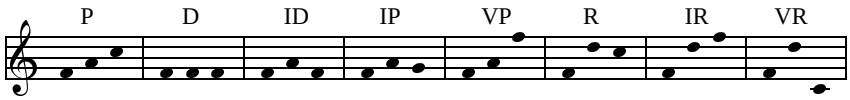

Figure 3.3: Eight of the basic structures of the I/R model

introduced by Meyer [1956]. The model describes both the continuation implied by particular melodic intervals, and the extent to which this (expected) continuation is actually realized by the following interval.

According to the I-R model, the sources of the listeners' expectations about the continuation of a melody are two-fold: both innate and learned. The innate sources are 'hard-wired' into our brain and peripheral nervous system, according to Narmour, whereas learned factors are due to exposure to music as a cultural phenomenon, and familiarity with musical styles and pieces in particular. The innate expectation mechanism is closely related to the *gestalt theory* for visual perception [Koffka, 1935; Köhler, 1947]. Gestalt theory states that perceptual elements are (in the process of perception) grouped together to form a single perceived whole (a 'gestalt'). This grouping follows certain principles (*gestalt principles*). The most important principles are *proximity* (two elements are perceived as a whole when they are perceptually close), *similarity* (two elements are perceived as a whole when they have similar perceptual features, e.g. color or form, in visual perception), and *good continuation* (two elements are perceived as a whole if one is a 'good' or 'natural' continuation of the other). Narmour claims that similar principles hold for the perception of melodic sequences. In his theory, these principles take the form of *implications*: Any two consecutively perceived notes constitute a melodic interval, and if this interval is not conceived as complete, or closed, it is an *implicative interval*, an interval that implies a subsequent interval with certain characteristics. In other words, some notes are more likely to follow the two heard notes than others. Two main principles concern *registral direction* and *intervallic difference*. The principle of registral direction (PRD) states that small intervals imply an interval in the same registral direction (a small upward interval implies another upward interval, and analogous for downward intervals), and large intervals imply a change in registral direction (a large upward interval implies a downward interval and analogous for downward intervals). The principle of intervallic difference (PID) states that a small (five semitones or less) interval implies a similarly-sized interval (plus or minus two semitones), and a large intervals (seven semitones or more) implies a smaller interval.

Based on these two principles, melodic patterns can be identified that either satisfy or violate the implication as predicted by the principles. Such patterns are called *structures* and labeled to denote characteristics in terms of registral direction and intervallic difference. Eight such structures are shown in figure 3.3. For example, the P structure ('Process') is a small interval followed by another small interval (of similar size), thus satisfying both the registral direction principle and the intervallic difference principle. Similarly the IP ('Intervallic Process') structure satisfies intervallic difference, but violates registral direction.

Another principle that is assumed to hold concerns the concept of *closure*. Closure refers to the situation where a perceived interval does not imply a following interval, that is the listener's expectation is inhibited. Occurrences of this kind of closure might coincide with those of a different, more commonly used concept of closure in music theory that refers to the

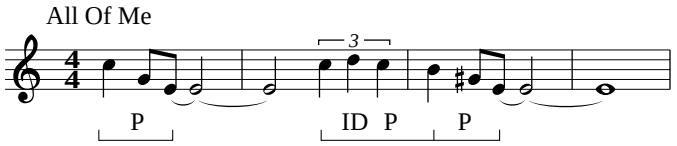

Figure 3.4: First measures of All of Me, annotated with I/R structures

Structure	Interval sizes	Same direction?	PID satisfied?	PRD satisfied?
P	S S	yes	yes	yes
D	0 0	yes	yes	yes
ID	S S (eq)	no	yes	no
IP	S S	no	yes	no
VP	S L	yes	no	yes
R	L S	no	yes	yes
IR	L S	yes	yes	no
VR	L L	no	no	yes

Table 3.2: Characterization of eight basic I/R structures; In the second column,'S' denotes small, 'L' large, and '0' a prime interval

finalization, or completion of a musical whole, such as a phrase. In the I-R model, closure, that is the inhibition of expectations, can be evoked in several dimensions of the music: when the melody changes in direction, or when a small interval is followed by a large interval. Additional factors that can add to the effect of closure are metrical position (strong metrical positions contribute to closure), rhythm (notes with a long duration contribute to closure), and harmony (resolution of dissonance into consonance contributes to closure). The degree to which the listener's expectations are inhibited is dependent on the accumulated degrees of closure in each dimension. Points of closure mark the boundaries of I-R structures. When there is closure, but only to a slight degree (weak closure), the end-note of one I-R structure also marks the beginning of the next I-R structure, causing the structures to share one note. When strong closure occurs, the next I-R structure begins on the note after the strong closure point. On the other hand, at the notes where no closure occurs, or where closure in some dimensions is overruled by continuing processes (such as repeated syncopation, or ritardando) in other dimensions, linked I-R structures occur. That is, the middle note of one I-R structure is the first note of the next I-R structure, causing the structures to share an interval. In this way two or more I-R structures maybe chained together. An example of this are the combined ID and P structure in figure 3.4.

3.5 Melody Segmentation

As we argued in section 3.1, working with musical chunks smaller than complete phrases is a natural step, because melodies by their nature are composed of several smaller note groups. Furthermore, segmentation likely to improve the utility of the case base, since it relaxes the constraint that in order to reuse the solution of a case, the complete phrases of the input problem and the retrieved case must match. Rather, it allows for partial reuse

	overlapping	non-overlapping
musical	I-R (weak closure)	Melisma, LBDM, I-R (strong-closure)
non-musical	n-grams, binary (hierarchical)	binary (flat)

Table 3.3: Classification of melodic segmentation strategies

of the musical examples in the case base, so that only the parts of the retrieved case that match the problem are reused (and only for the part of the problem that matches).

Given that cases should contain subphrase chunks of musical information, the next question is how the musical material should be segmented. In this section we will present some melody segmentation strategies, and discuss their characteristics and advantages/disadvantages.

We use the term melody segmentation to refer to a partition of a phrase as a sequence of notes into subsequences. Most automatic melody segmentation approaches are situated in the context of modeling note grouping according to human perception, and therefore conceive of melody segmentations as a complete and non-overlapping partitioning of the sequence of notes. Here however, there is no a priori reason to limit the concept to non-overlapping segments, since the aim of segmentation is not primarily to identify segments as 'perceptual wholes', but just to extract melodic material at a short time scale from phrases.

In order to clarify how different segmentation methods relate to each other we propose to classify melodic segmentations along two major dichotomies: overlapping vs. non-overlapping, and musical (that is, reflecting musical structure) vs. non-musical segmentations. In the following subsections, we will review various melodic segmentation techniques, exemplifying the various approaches we have discussed above. They can be classified as shown in table 3.3.

3.5.1 Melisma

The Melisma Music Analyzer [Temperley, 2001] provides a set of tools for various types of musical analysis. One of its components, the *Grouper*, is a tool that segments melodies into smaller units. It is flexible in the sense that it can operate on various time scales. For example, it can segment melodies into phrases, but also into smaller segments like motifs. It works in a straight-forward way: First it calculates a *gap-score* for every consecutive pair of notes. This gap-score is the sum of the interonset interval and the offset to onset interval. The score for a group of notes is proportional to the gap-scores at its boundaries. A penalty is applied proportionally to the difference between the length of the group and a specified optimal length. Additionally, a penalty is applied if the metrical position of its beginning is different from the metrical position of the beginning of the previous group.

3.5.2 Local Boundary Detection Model

The Local Boundary Detection Model (LBDM) [Cambouropoulos, 2001] is based on two Gestalt-related principles: *identity/change* and *proximity*. The model focuses on the degree of change in several parameters of melody, such as pitch, IOI, and rests. The change in these parameters is represented by taking the intervals (absolute differences) between the values for consecutive notes. Based in this, a boundary strength is calculated per interval that is proportional to the change in consecutive intervals. The model thus computes the second order derivative of the parameter sequence. Local maxima in the (weighted sum of the) second order derivatives are postulated by the model as local group boundaries in the melody. Like Melisma the LBDM model has several parameters to control its behavior. The most important of these is a threshold for the local maxima values to be recognized as a boundary. By increasing the threshold, longer segments are found.

3.5.3 I-R based Segmentation

Since I-R structure boundaries are determined by the level of perceived closure, it is not surprising that the boundaries of musical constructs like motifs often coincide with boundaries of I-R structures. This is not to say the converse, that every I-R structure boundary corresponds to a motif boundary. A motif may for example contain a weak closure (inducing an I-R structure boundary). If no closure occurs within a motif, the motif may be formed by multiple chained I-R structures.

Several melodic segmentation strategies can be based on the I-R analysis, depending on the 'closure threshold' that is adopted. For example, one can define a segment as the sequence of I-R structures that occur between two strong closure points. As mentioned in sub section 3.4.2, no overlap between I-R structures occurs on strong closure points, so this approach yields non-overlapping segmentations. When weak closure points are also taken to define segment boundaries, the segments may overlap by at most one note. A third, and less musically meaningful alternative is to define a segment for each I-R structure, even when they are chained. This may result in segments that overlap by two notes.

A drawback of I-R segmentation is that the above mentioned parameter to control segment-overlap provides only moderate control over the resulting segment length. Especially segmentation into non-overlapping segments may result in segments that vary greatly in length, depending of the melody. This can be seen in appendix B. Non-overlapping I-R segmentation results in segments that correspond to the joined horizontal brackets above the I-R labels. For example, Up Jumped Spring, phrase B would result in a single segment containing all 19 notes in the phrase, whereas phrases A1 and A2 of the same song contain segments of only 3 notes. Choosing weakly overlapping segments leads to segments of more constant length.

3.5.4 n-Grams Segmentation

n-Grams segmentation is commonly used in melody retrieval [Melucci and Orio, 2002; Uitdenbogerd and Zobel, 2002; Doraisamy and Rüger, 2003]. An n-gram is a subsequence of n items from a given sequence of items. By an n-gram segmentation of a melody we refer to

the ordered set of all n-grams that can be extracted from that melody. This is a very simple overlapping segmentation technique that employs no musical knowledge at all.

3.5.5 Binary Segmentation

In Kagurame Phase-II [Suzuki, 2003] Suzuki employs a form of binary segmentation of the melody that recursively divides the phrase into two parts of equal duration, to a desired level of granularity, for example measures, or beats. Because this method refers to measures and beats to divide the melody, it is not clear how situations should be handled where the number of measures, or beat is uneven. However, we can easily define a variation of Suzuki's method for binary segmentation that circumvents this problem, by dividing the melody by duration (rather than the number of measures). In this setup, the melody is recursively split in two, by finding its middle position (the position that is the average of the onset of the first note and the offset of the last note), and defining the first group as all notes whose onsets are before the middle position, and the second group as all notes whose onsets are after the middle position. The depth of the recursion can be used as a parameter to control the number of segments.

A notable difference with n-grams segmentation is that n-grams defines segments of equal numbers of notes whereas binary segmentation defines segments of (approximately) equal duration.

When the higher level segments of the recursive segmentation are included in the segmentation, obviously those segments will overlap with their child segments. When only the lowest level of segments is retained, a non-overlapping segmentation is obtained.

3.5.6 Characterization Segmentation Methods

Each segmentation method has advantages and disadvantages. Depending on the application context of the segmentation, these (dis)advantages may be relevant or less relevant. We discuss the different methods in the light of our current context, melody retrieval for tempo transformation.

The main advantage of non-overlapping segmentations for reuse of performance segments to transform an input performance in the current setting of tempo transformation, is that the final phrase result can be obtained straightforwardly, by concatenating the results of consecutive segments. If the problem solving would be done segment-wise using overlapping segments, the solutions (performances) to each segment could not be concatenated because the notes belonging to overlapping segments would have multiple solutions. In this case, an additional strategy is needed to decide which solution is preferred for these notes.

However, this apparent drawback of overlapping segments can also be regarded as an opportunity to evaluate the continuity of two consecutive segment performances. Since the performances are in part performances of the same notes, the similarity between these two performances can be taken as an indicator of continuity of the performance across segments.

Another advantage of overlapping segmentations is that it potentially allows for the creation of larger sets of segments from the original data set. Of course this introduces redundancy between segments, but the number of non-identical melodic segments in the case base is larger, and therefore case retrieval may offer useful results for a wider range of input problems, than with non-overlapping segmentations.

Taking into account the empirical fact that a melodic phrase is usually composed of short motifs, and that part of what defines musical styles is the length and melodic form of motifs (this seems to hold at least for jazz music), motifs of melodies that belong to the same style are likely to show recurring patterns. This means that segments that coincide with musical motifs are more likely to be found in other melodies of the same style than segments that intersect musical grouping boundaries, and are thus more useful as cases. To clarify this, a loose analogy might be made to spectral analysis of signals: by using an adaptive basis, wavelet analysis may yield both more accurate and more sparse representations of the spectral content of the signal than traditional Fourier analysis.

Suzuki [2003] argues that when the melodic segmentation is used just for case retrieval, it is not necessary that the segmentation reflects the musical structure. Even segments that contain musical boundaries may be useful, although the expected utility of such segments is lower, requiring a larger case base.

An additional observation regarding the class of musical grouping-aware segmentation strategies is that musical (grouping) structure can be intrinsically ambiguous, and due to its perceptual motivation is partially a subjective matter. Different segmentation approaches, even if they all intend to capture musical structure, may therefore yield different results.

We can summarize the above as follows: 1) The expected utility is higher for segments that correspond to musical groups, than for segments that do not correspond to such groups. This reduces the need for a large case base. 2) For a given set of melodies, overlapping segments result in a larger case base than non-overlapping segments. However joining the tempo-transformation results of overlapping segments into a tempo-transformation of the entire phrase is non-trivial. Although this creates an opportunity for deriving more consistent expressivity at the phrase level, exploiting this opportunity requires expressivity comparison criteria that we do not have at present. In conclusion, the most appropriate segmentation approach in our current situation seems to be non-overlapping segmentation that coincides with musical grouping. Therefore we employ Melisma as a segmentation algorithm in our current setup. Note that the LBDM and I-R (weak-closure) algorithms may be used as valid alternatives.

3.6 Performance Representation

It has been widely acknowledged that human performances of musical material are virtually always quite different from mechanical renderings of the music. These differences are thought to be vital for the aesthetic quality of the performance. Although it is not the only proposed definition of musical expressivity (see section 1.1), it is common to define expressivity as the deviations of the performance from the (mechanical rendering of) the score. This is a rather general definition in the sense that it does not specify the dimensions in which the difference should be measured. Most studies on expressive performance however focus on two or three dimensions, typically a subset of dynamics, timing (in onset and duration of notes), and local tempo changes (rubato) [Canazza et al., 1997a; Desain and Honing, 1994; Repp, 1995c; Widmer, 2000]. The paradigmatic representation of the expressivity is a list of expressive values for each note, and for each dimension, where the values represent the deviation between the nominal values as notated in the score, and the observed values of the corresponding note in the performance. In this setting, spontaneous insertions or deletions of

notes by the performer are often discarded as artifacts, or performance errors. This may be due to the fact that most of this research is focused on the performance practice of classical music, where the interpretation of notated music is rather strict. Contrastingly, in jazz music performers often favor a more liberal interpretation of the score, and as a consequence expressive variation is not limited to variations in timing of score notes, but also comes in the form of e.g. deliberately inserted and deleted notes. Thus, research concerning expressivity in jazz music should pay heed to these phenomena and in addition to capturing the temporal/dynamical variations of score notes, the expressive behavior of the performer should be described in terms of note insertions/deletions/ornamentations etcetera.

In this section we propose an annotation scheme for performances that provides such a deliberate representation of expressive phenomena. In the next chapter (section 4.2) we describe how such an annotation can be automatically derived using the edit-distance, and in chapter 6 (section 6.1), we show how this mechanism can be tuned to reduce annotation errors.

3.6.1 Describing Expressivity through Performance Events

In the present context, we use the word 'performance' to refer to a sequence of performed notes, the notes that are played by the musician when she interprets a musical score. The expressivity, defined as the deviations of the performance from the score (see section 1.1 on page 2), can be conceived of as a sequence of events (we will call them *performance events*) that describe how the notes in the score are translated to the notes present in the performance. The first question that must be addressed is which types of performance events we distinguish. On a very abstract level, performance events are *reference events*: they refer either to notes in the performance, or to notes in the score, or both. When viewing reference events as a class, we can thus distinguish two sub-classes: the events that refer to score elements (*score reference events*), and the events that refer to performance elements (*performance reference events*). Events that refer to both are called *correspondence events*, forming a sub-class of both score reference event and performance reference event. This taxonomy of performance events provides a framework in which more specific kinds of performance events can be defined. Figure 3.5 shows this hierarchy, including more specific types of performance events that will be discussed below.

Note that the choice of performance events is an important, but fundamentally subjective decision. Since the performance of a melody by a human musician is not a causal process, there is no way to systematically deduce which performance elements are the manifestation of a particular score element. There may be situations where there is ambiguity as to the way of representing a particular part of the musician's interpretation through performance events. In practice the majority of notes in a performance resemble their score counterparts close enough in terms of pitch, metrical position, and duration in order to unambiguously establish the association[6]. Such occurrences of performed notes that clearly correspond to a score note are interpreted as instances of the class of correspondence events mentioned above. Within a certain range of perceivable sameness, the onset timing, pitch, duration of the performed note may deviate from the score note. These deviations are respectively stored

[6]One can argue that in cases where it is very difficult to recognize the score melody in the performance, the performance should be conceived of as a variation, or improvisation on the melody, rather than an expressive interpretation of it

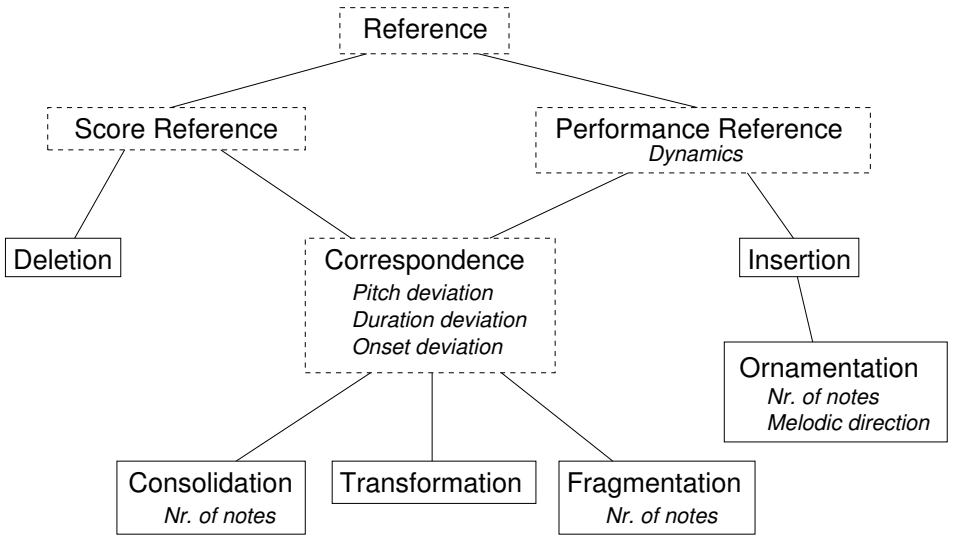

Figure 3.5: A taxonomy of performance events for performance annotation

in the attributes *onset deviation*, *pitch deviation*, and *duration deviation*, of correspondence events. When a correspondence occurs between a single score note and a single performance note, we represent this by a sub-class of correspondence events, *transformation events* (the attributes of the score note are *transformed* in the performance).

The necessity of the various sub-classes of the correspondence event class to represent the musician's performance behavior is trivial, but it is less obvious what other kinds of performance events will be necessary for a complete account of the performance. We have analyzed the musical material that used to populate our case base (see section 3.2), focusing on the situations where a one-to-one correspondence could not be made between score and performance notes. In the following paragraphs, we will define various classes of performance events that account for the different situations we encountered where transformation events cannot describe the performance, and illustrate each performance event class with an example found in the real performances.

The first situation is where a performance note occurs without a clear counterpart in the score, whereas the score counterparts of surrounding notes are unambiguous. In other words, the musician inserts a note that is not present in the score. We account for this situation with an *insertion event*, which is a sub-class of performance reference event. It is not a sub-class of score reference event, since the inserted note has no counterpart in the score. An example of an insertion event from the real performances described in section 3.2 is shown in figure 3.6a. The figure shows the score that was performed, together with the notes that were performed, shown as boxes. The vertical positions correspond to the pitch of the performed notes (the higher the pitch, the higher the position). The widths of the boxes are proportional to the durations of the notes, and their horizontal positions are proportional to onset times of the notes. This way of representing melodies is common in MIDI editing software and is usually called *piano roll* notation. The letters in between the score and the

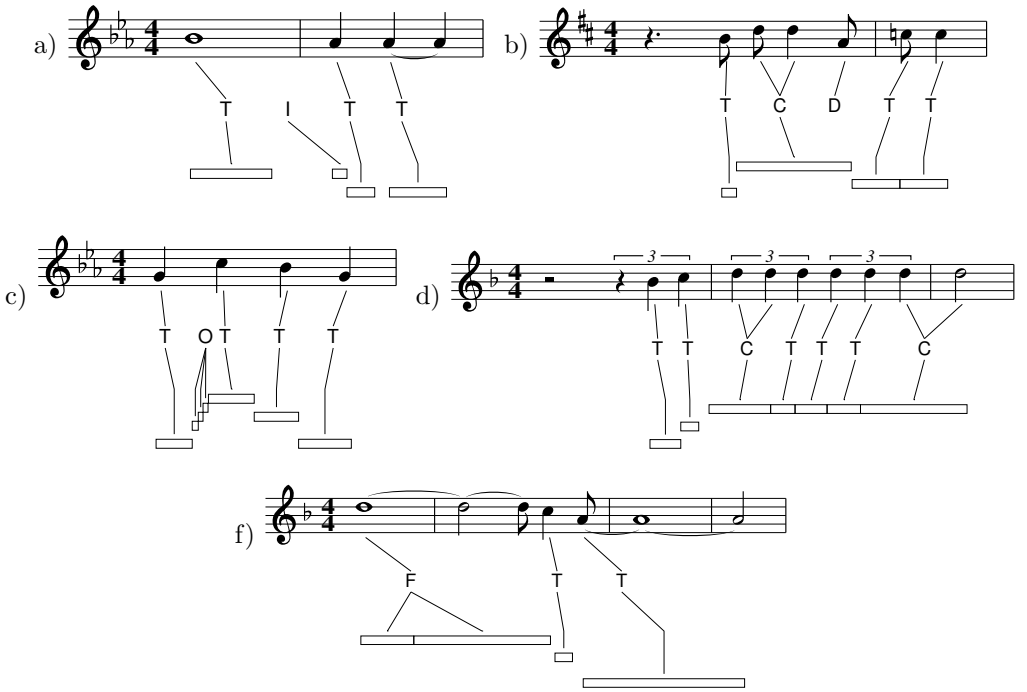

Figure 3.6: Examples of performance event classes encountered in real performances. The boxes below the score represent the performance in piano roll notation. The letters denote performance events (T = Transformation; I = Insertion; D = Deletion; O = Ornamentation; F = Fragmentation; C = Consolidation)

performance represent the performance events that establish the relation between score and performance elements.

The second situation is the opposite of insertion: deletion. This occurs when a note in the score is not played by the performer, and consequently has no counterpart in the performance. Hence, *deletion events* are a sub-class of score reference events only. An example case of deletion is shown in figure 3.6b (denoted by the letter 'D').

Thirdly, a special kind of insertion that occurs very frequently, is *ornamentation*. Although ornamentation also involves performed notes that have no counterpart in the score, *ornamentation events* are different from ordinary insertion events in several respects: Firstly, the inserted notes are usually very short (typically about one tenth of a second). Secondly, they often come as a duple or triple of very short notes. Thirdly, the inserted notes virtually always form a chromatic (occasionally diatonic) ascending progression with the note that follows the ornamentation notes. Ornamentation event are as such a sub-class of insertion events. An example is shown in figure 3.6c. Note that the triple of ornamentation notes in the example are treated as a single ornamentation event. We argue that this is a more appropriate conception of ornamentation events than viewing each inserted note as a separate

ornamentation. Because of their pitch relationship, and closeness in time, the ornamentation notes form a very strong perceptual whole. They are perceived as a single musical gesture. Annotating this situation with three distinct ornamentation events suggests that each of them could have occurred individually, which is clearly not the case.

The fourth kind of performance event occurs when several consecutive notes from the score are played as a single long note. The notes typically have the same pitch, and the duration of the performed note is close to the total duration of the score notes. This many-to-one mapping of notes was described in [Mongeau and Sankoff, 1990] as *consolidation* in the context of comparison of melodies (that is, theme and variations). It is interesting to observe that consolidation of notes is a phenomenon that occurs both as a matter of melody variation (at the level of composition), and in music performance (at the level of musical expressivity). An example is shown in figure 3.6d.

The last kind of performance event described here is *fragmentation*. This is the converse of consolidation, and involves the performance of a single score note as multiple notes of the same pitch, with a total duration that is close to the duration of the score note. This situation (which was also described in [Mongeau and Sankoff, 1990] to relate variations to a thematic melody), was found to occur in a number of performances as well. Figure 3.6e shows such a situation.

The kinds of performance events introduced above were adequate to cover the situations where no obvious one-to-one correspondence could be established between score and performance elements. Once the events have been defined and have been found to occur in particular performances, it is interesting to see the frequency of occurrence of each type of event as a function of the tempo at which the performance was played. Figure 3.7 shows these numbers for each type of performance event (transformation events have been left out, since their mere occurrence does not convey expressivity; rather it is the kind and amount of transformation that bears expressive significance). The tempo is denoted as a proportion of the *nominal tempo* (see section 3.2), that is, the average of all tempos available for a given phrase. The frequency of occurrence of a given kind of performance event is given as the proportion of that kind of event with respect to the total number of performance events (including transformation events), for that tempo. The curves have been slightly smoothed to improve the legibility of the graph.

Some systematic trends can be observed from the figure. The most salient of these is that ornamentation events are by far the most frequent type of event. Furthermore, they occur most frequently at low tempos, and their occurrence steadily diminishes with increasing tempo. This is in accordance with the intuition that since the density of notes is lower at slow tempos, the musician has more opportunity to add detailed forms of expressivity like ornamentation. For the same reason, the general trend that the number of consolidations increases with tempo is not surprising. Consolidation of notes decreases note density and thus compensates for the increased note-density due to the tempo increase. Similarly, insertion events are more frequent at low tempos than at high tempos, and vice versa for deletion event. Only fragmentation events behave contrary to expectation. One would expect that fragmentations (increasing note density) would be more frequent at low tempos than at high tempos, but the opposite is the case. Table 3.4 makes these trends explicit. It shows per event type how the occurrence of that event type is distributed over slow (< 1) and fast (> 1) tempos. The numbers in the second and third columns are proportions of the total

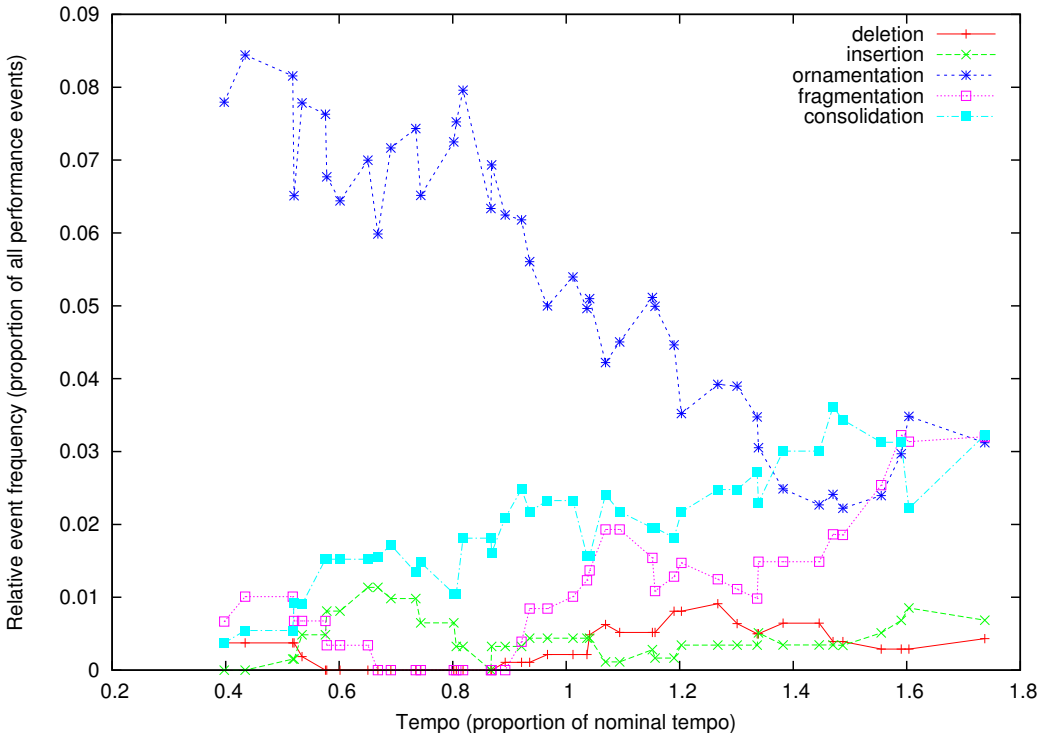

Figure 3.7: The frequency of occurrence of several kinds of performance events as a function of nominal tempo

occurrence of the event types.

We don't want to derive any quantitative conclusion from these numbers, since making robust claims about the occurrence of specific types of events as a function of tempo would likely require more data. However, we do think the figures presented here are strong evidence that expressive phenomena such as ornamentation, and consolidation do not remain constant throughout different tempos. And this justifies our claim that in addition to widely recognized expressive features such as note timing, and dynamics, more complex features like ornamentation, consolidation etcetera, should be taken into account when dealing with expressivity in the context of tempo transformation of performances.

event type	slow	fast
deletion	0.18	0.82
insertion	0.60	0.40
ornamentation	0.67	0.33
fragmentation	0.12	0.88
consolidation	0.39	0.61

Table 3.4: Distribution of kinds of performance events over slow and fast performances (slow = slower than the nominal tempo; fast = faster than the nominal tempo). The numbers express the proportion of events per tempo category for each kind of event

Chapter 4

Knowledge Acquisition

This chapter is devoted to the knowledge acquisition aspects of TempoExpress. Knowledge acquisition refers to the process of obtaining knowledge enriched representations of raw input data. Such representations are in some cases unavoidable (for example, performances must be aligned to their corresponding scores in a knowledgeable way in order to arrive at an adequate description of the performance expressivity). In other cases they form part of the particular strategy we apply to solve tempo transformation problems. For example, we use Implication-Realization analysis to compare melodies during case retrieval. The motivation for this approach is that it is widely recognized that expressivity in music performance is strongly related to musical structure in the melody, (as discussed in chapter 2). Widmer [1996] has shown that structural representations of the melody improve the prediction of expressive performance prediction over simple note-level representations. This suggests that a proper grasping of expressive aspects of music performance involves at least some abstraction from the note-level perspective on the melody.

Knowledge acquisition occurs at two stages that are very much related: case acquisition and input problem specification. Case acquisition refers to the construction of cases to populate the case base, which is done before the CBR system becomes operational. Input problem specification is done at the time of problem solving, and consists in building a problem description containing all the information needed to solve the problem. Because a case is essentially a problem description together with its solution, the task of input problem analysis is in fact subsumed by the task of case acquisition.

Figure 4.1 on the next page shows the TempoExpress system components that will be addressed. Section 4.1 on the following page describes the musical analysis component, in particular, the procedure we developed to obtain the I-R analysis of a melody. In section 4.2 we review the edit-distance, a distance measure and alignment technique for sequential data that we use extensively in this work. Section 4.3 deals with performance annotation. The last section describes how these representations are stored together as cases.

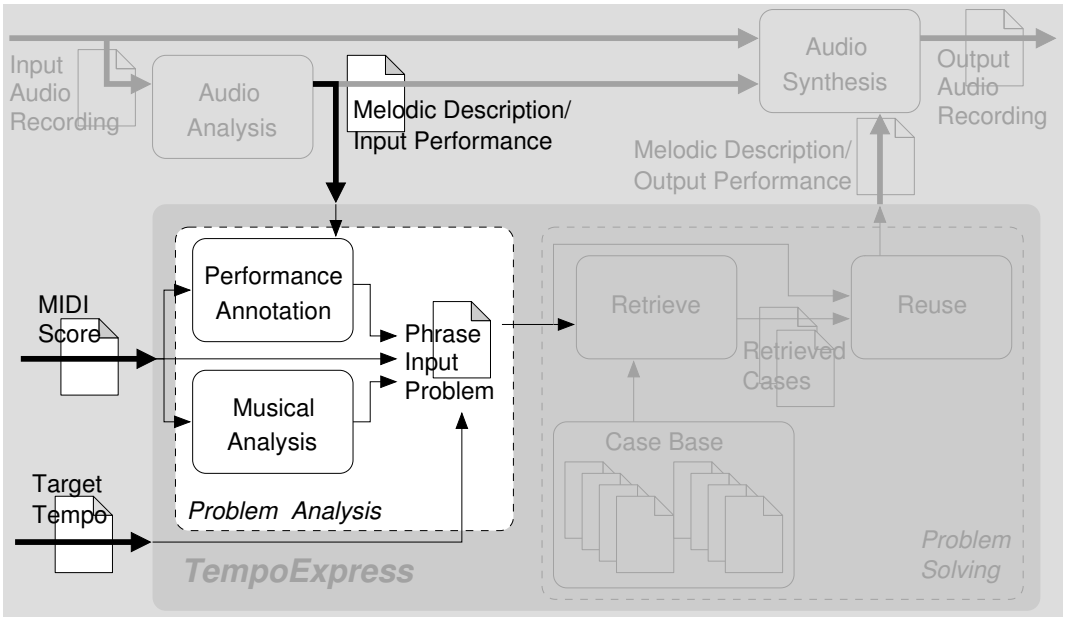

Figure 4.1: The data preparation part of TempoExpress

4.1 The Implication-Realization Parser

The Implication-Realization model provides a way to describe the pattern of implied and realized expectations that a melody generates through its surface form. As such, the model is hypothesized to describe the cognitive processes that constitute the listener's auditory experience of the melody. Typical I-R structures span three notes and may be *chained* across two or three structures (i.e. consecutive structures overlap by one or two notes), depending on particular form of the melody. Rather than reducing the melody as a whole to a single construct (such as a tree), the I-R analysis primarily expresses local structure of the melody. It reflects the temporal and streaming nature of melody, and the sizes of the structures are in accordance with current knowledge of the character human short term memory. This adds to the cognitive plausibility of the model.

The character of the I-R analysis as outlined above, makes it a good candidate representation for melody processing applications. In this dissertation, its major application is melodic similarity computation, a step that is crucial for case retrieval. An empirical study showed that human subjects tend to base similarity judgments of melodic motifs on surface features (like contour, and rhythmic organization), rather than on deep structure [Lamont and Dibben, 2001]. This result implies that for melodic similarity computation, the I-R analysis as a melody representation is favorable over deep structure analyses such as GTTM's time-span reduction trees.

In order to employ the I-R model in TempoExpress, we have implemented a parser for monophonic melodies that generates the corresponding I-R analyses. In this section, we will subsequently describe the features and limitations of the I-R parser (subsection 4.1.1), and

explain the parsing steps involved to derive the I-R analysis from the melody (subsection 4.1.2).

4.1.1 Scope and Limitations

The I-R model has high degree of complexity, and although the core part of the theory is stated explicitly and in great detail, not all aspects of the theory are very amenable to computational modeling. Therefore, it is hard to achieve fully automatic derivation of I-R analysis of melodies, so that they incorporate to all of the aspects the I-R model. In this sense, the I-R parser that we present here is only a partial implementation of the I-R model.

Firstly, the parser implements bottom-up rather than top-down processes. Thus, according to the theory, it captures innate factors that govern expectation, not learned factors. In effect, the parser has no way to identify exceptions to the expectation generation mechanism, as imposed by familiarity with the particularities of the musical piece being parsed (*intra opus style*) and by the embedding in a particular musical culture (*extra opus style*). The I-R model does not specify top-down processes in as much detail as the bottom-up processes, supposedly because they are harder to identify and formalize. Moreover, since the bottom-up processes are claimed to be style, culture, and training independent [Narmour, 1990; Cuddy and Lunney, 1995; Schellenberg, 1996], it seems justified to focus on that part of the model.

The parser can identify 17 different types of I-R structures, namely the basic structures P, D, R, ID, their derivatives VR, IR, VP, and IP, the retrospective structures (P), (ID), (R), (VR), (IR), (VP), (IP), dyadic, and monadic structures. Combinations (two structures linked together) and chains (more than two linked structures) of these structures are also identified.

We designed the parser to operate on monophonic melodies that are represented as sequences of notes having only pitch, onset, and duration attributes. As a consequence, the concept of *closure/ non closure* is implemented for the parameters meter, and rhythm. Closure (and its inhibition) induced by harmony, melodic consonance/dissonance, dynamics, and ritardando/accelerando effects are not taken into account since the necessary information is not present in the representation of the melody.

Lastly, the parser analyzes only the melodic surface. That is, no higher level structures are obtained.

4.1.2 Implementation

In this sub-section we present a parser for constructing I-R analyses from melodies. Melody, rhythm, and meter are processed to produce the I-R analysis as a sequence of I-R structures.

The parser takes as input a melody, represented as a sequence of notes, having pitch, duration and position attributes. Additionally, the meter is known (that is, the metrical strength of each beat can be inferred). The strategy applied to obtain the analysis is roughly to first determine the boundary notes of structures (or chains of structures) by computing the level of (non) closure in various dimensions, and then identify the I-R structures between those boundaries, based on pitch and interval information. Determining non closure is not entirely independent of the I-R structure identification, but this dependency can be solved by 'peeking forward' to determine the I-R structure at the position in question, and does

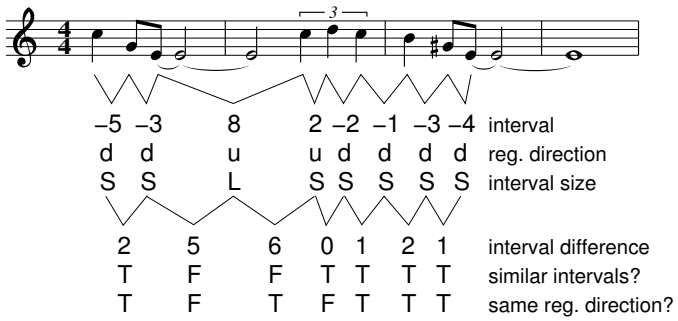

Figure 4.2: Interval information for the first measures of *All Of Me*

not create a bootstrapping problem. More concretely, the parsing process can be divided into the following steps:

1. Gather interval information;
2. Mark level closure at each note position;
3. Mark inhibition of closure;
4. Identify structures between points of strong; closure based on interval information; and
5. Aggregate chained D and P structures.

In the following sub-sections, we will explain these steps in more detail.

Gather Interval Information

The first step is to compute the interval sizes and their directions ('up', 'down', and 'lateral'). In addition to the numerical interval values (measured in semitones), we apply the so-called *syntactic parametric scale* to the intervals, in order to categorize their size as either small ('S') or large ('L'). In general, the I-R model assumes that intervals smaller than 6 semitones are perceived as small intervals, and those larger than 6 semitones as large intervals. Intervals of 6 semitones are ambiguous (and may therefore give rise to multiple I-R analyses of the same melody). To avoid this additional complexity, the syntactic parametric scale in the parser defines intervals of 6 semitones to be large as well. Figure 4.2 illustrates this for the first measures of *All of Me*.

Additionally, the difference in size between pairs of consecutive intervals are computed, and the pairs are labeled with two Boolean labels: *intervallic similarity*, and *registral sameness*. These two features are of primary importance in the classification of a sequence of notes as a particular I-R structure.

Two consecutive intervals are said to be intervallically similar whenever their *intervallic differentiation* (i.e. the difference between these intervals) is less or equal to a minor third. The registral sameness predicate holds if subsequent intervals have the same registral direction, which can be either upward, downward, or lateral (in the case of a prime interval).

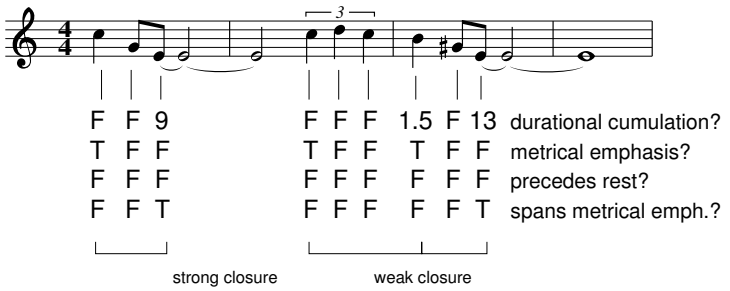

Figure 4.3: Closure information for the first measures of *All Of Me*

Mark Closure

In the I-R model, the term *closure* is used for the inhibition of implication. This inhibition can be established by several conditions. The most prominent are rests, *durational cumulation* (this occurs where a note has significantly longer duration than its predecessor), metrical accents, and resolution of dissonance into consonance. These conditions can occur in any combination. Depending on the number of dimensions in which closure occurs and the degree of closure in each dimension, the overall degree of closure will differ.

The closure detection step in the parser here presented detects four conditions for closure, all related to meter and rhythm:

1. durational cumulation (the duration of a note is substantially longer than its predecessor);
2. the onset of a note falls on a metrical accent;
3. the occurrence of a rest after a note; and
4. a metrical accent occurs while the note is sounding.

For each note in the melody, every condition is evaluated as, illustrated in figure 4.3. The total degree of closure depends on the conditions that are satisfied. Weak closure occurs when any of the closure conditions occur. Strong closure occurs either when condition (3) holds, or when condition (1), and (3) hold, and (1) doesn't hold for the next position.

But in order to determine the final positions where strong or weak closure occurs, one must also take into account factors that may inhibit closure.

Mark Inhibition of Closure

The conditions mentioned in the previous sub section in themselves imply closure. However there are other factors that may inhibit such closure. This causes I-R structures to chain and combine. There are eight conditions that inhibit closure in other parameters. Four of them deal with harmony, melodic consonance/dissonance, dynamics and accelerando/ritardando, respectively. As mentioned before, those factors are ignored in the parser. The remaining factors are:

1. The absence of metrical emphasis;
2. The envelopment of a metric accent by a P process in the context of additive (equally long) or counter-cumulative (long to short) durations;
3. The envelopment of a metric accent by a D process in the context of additive (equally long) durations; and
4. The occurrence of repeated syncopation.

When any of these conditions are satisfied, any metrical closure that was detected in the previous step, is canceled. Notice that conditions (2) and (3) require that the I-R structure is known for the intervals surrounding the note which is investigated for closure inhibition. Although the final I-R structures can only be calculated after the structure boundaries have been identified, we compute the I-R structure of the relevant interval pair regardless of the boundary positions, in order to evaluate the conditions. In the example shown in figures 4.2, and 4.3, the last two triplet notes at the end of the second bar satisfy condition (1), and in addition the last triplet note satisfies condition (3), and as a result, there is no boundary on any of these notes.

Identify I-R Structures

When the final degree of closure is established for each point in the sequence, the next step is the categorization of the interval pairs, based on the information gathered during the first step. The notes on which weak closure occurs will be the terminal note of one I-R structure and the initial note of its successor. The notes where strong closure occurs will be the terminal note of an I-R structure, but its successor structure will begin on the next note. The fact that the interval between these two notes is not part of any I-R structure expresses the fact that this interval is not implied nor implies anything. In between two points of closure, combining and chaining occurs (i.e. consecutive structures share one interval), due to the lack of closure.

The I-R structures are identified by taking every consecutive pair of intervals between closure points, and traversing the decision tree depicted in figure 4.4, based on the interval information that was gathered earlier. Two special situations deserve attention. Firstly, the case were two successive notes both have strong closure. In this situation, a *dyadic* structure applies, which spans only one interval, and is denoted by a number indicating the size of the interval in semitones. The second case occurs when a note occurs in isolation (e.g. separated from the melody by rests). This leads to a *monadic* structure, denoted with the letter M.

Aggregate Chained P and D Structures

In the I-R model, there are two of the basic structures that can span more than three notes, viz. P and D structures. the latter occur when three or more notes form a ascending or descending sequence of similar intervals; The former occurs when three or more notes continue in lateral direction, that is, have the same pitch. A further condition is the absence of closure, for example by durational cumulation.

In the previous step, only structures of three notes have been identified. Yet, possible instantiations of longer P and D structures in this way manifest themselves necessarily as

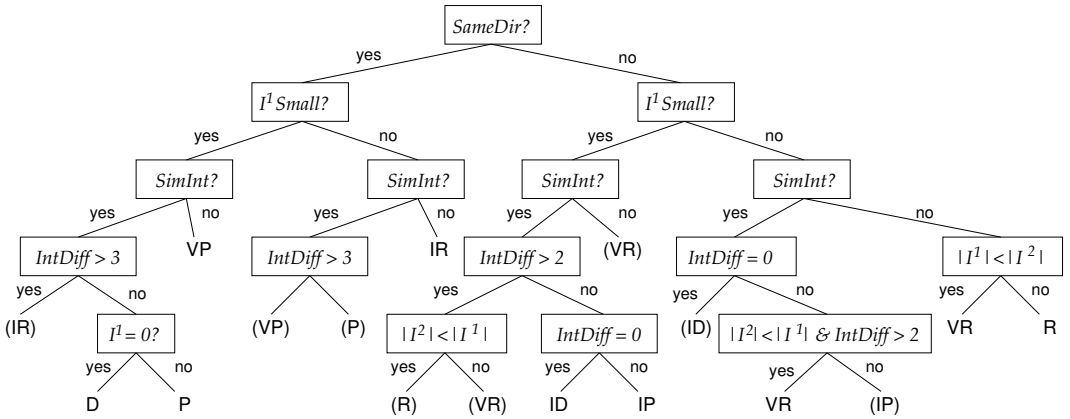

Figure 4.4: Decision tree for labeling pairs of consecutive melodic intervals, I^1, and I^2. *SameDir*: I^1 and I^2 have the same registral direction; *SimInt*: I^1 and I^2 are similar in size; *IntDiff*: $|I^1| - |I^2|$

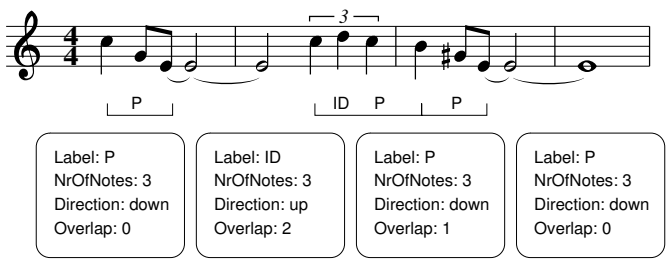

Figure 4.5: Representation of I-R analysis as a sequence of I-R structure objects

combined or chained P and D structures, respectively, Thus, to arrive at a correct I-R analysis, any combined or chained repetition of P or D structures is replaced by a single P or D structure, spanning all of the notes that were spanned by the original structures.

4.1.3 I-R Analysis Representation

The final I-R analysis for the example is shown in figure 4.5. At the bottom, the figure shows the concrete representation of the analysis, as output of the parser. The analysis is a sequence of I-R structure objects, that have as attributes:

- the name of the I-R structure (e.g. P, VP)
- the number of notes the structure spans

- the registral direction (defined as the sign of $pitch_{LastNote} - pitch_{FirstNote}$; if zero then sign of $pitch_{LastNote} - pitch_{MiddleNote}$)
- the number of notes shared with successor structure

In section 5.2.2, we will propose a method for computing melodic similarity that is based on the I-R analysis. This method compares sequences of I-R structures using the edit-distance. In the next section we will give a general overview of the edit-distance and show how it can be extended to allow for context sensitive comparison of sequence elements.

4.2 Flexible Sequence Alignment/Comparison: the Edit-Distance

The *Edit-distance*, or *Levenshtein-distance* [Levenshtein, 1966], is a measure for comparing sequences (or strings) commonly used in a variety of fields and applications that deal with sequential data. Common application domains are for example spell-checking of text, and bio-computing, where it has been used to solve pattern finding, and information retrieval problems. In the domain of music, the edit-distance has been used for computing melodic similarity [Mongeau and Sankoff, 1990; Smith et al., 1998], score following/automatic accompaniment [Dannenberg, 1984; Puckette and Lippe, 1992] and performance transcription [Large, 1993; Pardo and Birmingham, 2002].

The edit-distance is defined as the minimal cost of a sequence of editions needed to transform a source sequence into a target sequence, given a predefined set of edit-operations. The canonical set of operations consists of insertion, deletion and replacement of sequence elements. The cost of a particular edit-operation is defined through a *cost function* w for that operation, that computes the cost of applying that operation to the notes of the source and target sequences that were given as parameters to w. Let $\mathcal{A}$ be the alphabet, i.e. the set of possible symbols occurring in sequences, and let $\mathbf{s}$ and $\mathbf{t}$ be finite sequences over $\mathcal{A}$, of length M and N respectively. Then the typical cost functions associated to insertion, deletion, and replacement are respectively:

$$w(\emptyset, t_j) = 1 \qquad \textit{insertion} \tag{4.1}$$

$$w(s_i, \emptyset) = 1 \qquad \textit{deletion} \tag{4.2}$$

$$w(s_i, t_j) = \begin{cases} 0 & \textit{if } s_i = t_j \\ 1 & \textit{otherwise} \end{cases} \qquad \textit{replacement} \tag{4.3}$$

where s_i $(0 \leq i < M)$ and t_j $(0 \leq j < N)$ are the elements of $\mathbf{s}$ and $\mathbf{t}$, respectively. In case $M \neq N$, insertion and deletion operations are used to accommodate the superfluous elements in the longer of the sequences. The above edit cost scenario is called the *unit cost model*: the cost of edit-operations does not depend on any characteristic of the elements under consideration, apart from their equality or inequality, in the case of replacement. More specialized cost models can be defined that take into account the semantic relations between the symbols of the alphabet. The sequences may also consist of composite structures, such as attribute-value pairs, rather than symbols. Moreover, the set of edit-operations for computing the edit-distance is not necessarily limited to insertion, deletion, and replacement.

Edit-operations may operate on subsequences of arbitrary size from the source and the target sequences. The edit-distance can be defined to deal with any set of edit-operations. To do this, we write $w^{K,L}$ to denote that a cost function w takes a subsequence of length K of the source sequence as its first input parameter, and a subsequence of length L of the target sequence as its second input parameter. The numbers K and L correspond to the number of elements from the source and target sequence w operates on, respectively. For example, a deletion operation would have a cost function $w^{1,0}$. For a given set of edit-operations, let W be the set of corresponding cost functions, and let

$$V_{i,j} = \{w^{K,L} \mid K \leq i \wedge L \leq j\} \subseteq W$$

be the subset of cost functions that accept subsequences with maximal lengths of i and j, respectively. Furthermore, let $\mathbf{s}_{1:i} = \langle s_1, \cdots, s_i \rangle$ and $\mathbf{t}_{1:j} = \langle t_1, \cdots, t_j \rangle$ be the source and target sequences respectively. Then the edit-distance $d_{i,j}$ between $\mathbf{s}_{1:i}$ and $\mathbf{t}_{1:j}$ is defined recursively as[1]:

$$d_{i,j} = \min_{w \in V_{i,j}} \left(d_{i-K,j-L} + w(\mathbf{s}_{i-K+1:i}, \mathbf{t}_{j-L+1:j}) \right) \tag{4.4}$$

where the initial condition is: $d_{0,0} = 0$.

Using equation (4.4), a $M+1 \times N+1$ matrix can be filled with distance values $d_{i,j}$ for $0 \leq i \leq M$ and $0 \leq j \leq N$. The edit-distance value $d_{M,N}$ for the entire source and target sequences appears in lower right corner of the matrix, and the optimal alignment is found by tracing back from cell (M, N) to $(0, 0)$ and recording which was the last applied operation operation that led to the distance value in each visited cell (i, j). This yields (in reverse order) the optimal sequence of edit operations, called the *optimal alignment* between $\mathbf{s}$ and $\mathbf{t}$.

The edit-distance has proved to be a versatile tool in the research presented in this dissertation. Since the musical data we deal with is mainly of sequential form (e.g. melody, performance, I/R analyses), comparison of sequences is a common task, both for distance estimation (as in case retrieval, for example), and sequence alignment (as in performance annotation, and case reuse). We will briefly point out the key advantages of the edit-distance.

Firstly, the edit-distance is *informative*, in the sense that computing the edit-distance yields not only a distance value, but also the optimal alignment between two sequences, that displays which *parts* of the sequences are resembling. In case the primary interest is in the alignment, the distance value is still useful, as it reflects the quality of the alignment.

Secondly, a fundamental characteristic of the edit-distance is that it is *tolerant*: it puts no restrictions on the length of the sequences: sequences of unequal (and arbitrary) length can be compared. Of course, with non-zero insertion and deletion costs, distance values will increase with increasing difference in sequence length.

Thirdly, the set of edit-operations is by no means limited to the canonical set of insertion, deletion, and replacement. Arbitrary edit-operations may be defined. Depending on the domain, or interpretation of the sequences, other edit-operations may be appropriate, such as order inversion, permutation, consolidation, or fragmentation of elements. In this way, the edit-distance can be *customized* to fit the application domain.

[1] See appendix A for notational conventions on denoting sequences

Fourthly, through the adaptation of the costs of edit-operations, the distance measure can be *optimized* to fit a particular (partly) known distance (examples of this can be found in sections 6.1 and 6.2.2). Finding the desired edit-operation costs can be done manually, or by local search methods, like genetic algorithms [Grachten et al., 2004a], or probabilistic methods [Ristad and Yianilos, 1998].

Finally, the use of edit-distance for comparing sequences is not limited to sequences of symbols, nor to sequences of elements of the same type. As long as the cost functions properly define the cost of an edit-operation, the elements of the sequences can be of any type. This feature is of crucial importance in our method of performance annotation, that we will present in detail in section 4.3 on the next page.

4.2.1 Context-Sensitive Cost Functions

An edit-operation always accounts for (or operates on) zero or more elements in the source sequence, and zero or more elements in the target sequence. A complete alignment accounts for every element in the sequences, and each element is accounted for by precisely one operation (e.g. an element cannot be both deleted and replaced). The number of elements an edit-operation accounts for in the source and target sequences is implicitly specified in the definition of d (equation (4.4)). For example, the term $d_{i-1,j} + w(s_i, \emptyset)$ (being the cost of the optimal alignment $d_{i-1,j}$ plus the cost of deleting element s_i), specifies that the deletion operation accounts for one element in the source sequence, and zero in the target sequence, since the deletion extends the alignment from sequences $(s_0, \cdots, s_{i-1})$, $(t_0, \cdots, t_j)$, to $(s_0, \cdots, s_i)$, $(t_0, \cdots, t_j)$.

In equation (4.4), the number of elements passed to w coincides with the size of the operation range (i.e. the number of elements accounted for by the edit-distance). Thus, only the elements accounted for by the edit-operation can figure in the cost function. In this sense, the cost function is not context-sensitive. But there is no reason why the number of arguments to w should coincide with the size of the operation range. In addition to the operation range itself, preceding and succeeding ranges can be included, to determine the cost of the operation based on the elements accounted for inside their context. Figure 4.6 illustrates this. Note that the elements that serve as context for one operation, will be accounted for by neighboring operations in the alignment. Note also that in this more general way of defining cost functions, the cost functions w must be indexed, since the functions for the various edit-operations cannot be distinguished anymore based on their signature, like the functions (4.1), (4.2), (4.3).

Taking this into account, and using the letters A, B, C, D, E, and F to denote the sizes of the operation ranges and contexts as in figure 4.6, we can redefine the edit-distance recursion equation in a more general way like this:

$$d_{i,j} = \min_{w \in V_{i,j}} \left(d_{i-K,j-L} + w(\mathbf{s}_{i-K-K^{pre}+1:i+K^{post}}, \mathbf{t}_{j-L-L^{pre}+1:j+L^{post}}) \right) \tag{4.5}$$

where $0 \leq i - A^k$, $0 \leq j - B^k$, $i + B^k + C^k \leq M$, and $j + E^k + F^k \leq N$ ($0 \leq k \leq n$, and M, and N are the sizes of the sequences s and t respectively).

It can be easily seen that equation (4.5) subsumes the earlier recurrence equation (4.4), since the terms in the latter equation can be obtained by choosing the appropriate values for the operation range and context sizes. For example, the deletion term has an operation

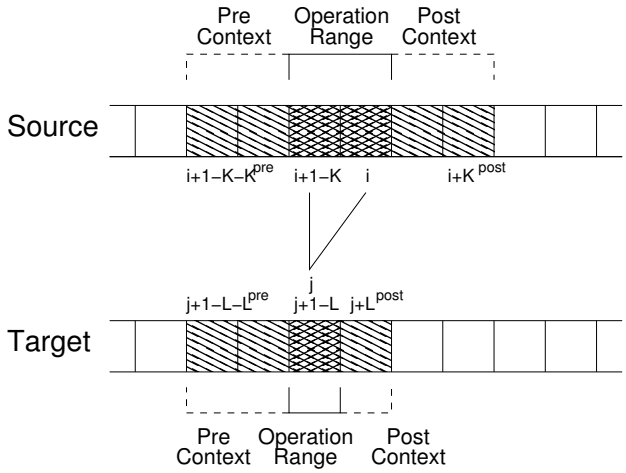

Figure 4.6: An alignment of elements using a hypothetical context-sensitive edit-operation with pre-context sizes K^{pre}, L^{pre}, operation ranges K, L, and post-context sizes K^{post}, K^{post}, respectively for the source and target sequences

range of size one in the source (B), an operation range of size zero in the target (D), and all context sizes zero (A, C, D, and F). In the insertion term, D is equal to one and the other values to zero.

An earlier report of the use of context-sensitivity of cost functions was made by [Lemström and Ukkonen, 2000]. They applied context-sensitive cost functions to enable transposition independent melody comparison using the edit-distance without the need for a preprocessing step to obtain interval encodings of the melodies. We will show in section 4.3.1 how the context-sensitivity of cost functions is employed in the process of performance annotation to detect ornamentations in the performance.

4.3 Automatic Performance Annotation

In subsection 3.6.1, we have described a representation scheme for annotating performances, with the goal of having an explicit description of the performer's behavior in terms of the changes that she applied to the nominal score information to obtain a performance. We will now describe how such performance annotations can be automatically derived from a performance and its corresponding score using the edit-distance.

Although the use of the edit-distance is well established in related fields like melodic similarity [Mongeau and Sankoff, 1990; Smith et al., 1998], score following/automatic accompaniment [Dannenberg, 1984; Puckette and Lippe, 1992] and performance transcription [Large, 1993; Pardo and Birmingham, 2002], not much attention has been paid to its value for the expressive analysis and annotation of musical performances. We will argue that the edit-distance is a very appropriate tool for performance annotation, and show that how a set of parametrized edit-operations can be optimized to bring error rates of automatic performance

annotation down to 2–4%.

An advantage of the edit-distance that explains why it is commonly used in comparisons between scores, or between scores and performances, is that in addition to expressing a distance, it provides an optimal alignment between the sequences to be compared. The alignment provides a detailed account of which element in the source sequence resembles which element in the target sequence (both in terms of similarity and relative sequential order). This information is obviously of interest in score-following where the score-performance correspondence is needed to keep track of the performer in real-time, or in performance-transcription, where it is used to detect the errors in the performance.

Another advantage is that the edit-distance is robust in the sense that its base-line performance is very good. That is, when the source and target sequences have a reasonable degree of similarity, that is, when a human subject would find it easy to establish an intuitive alignment, the edit-distance will be able to align the sequences in that intuitive way, even with a very simplistic cost model like the unit cost model (see subsection 4.2). Still, in specific cases it is possible to improve the alignment quality well beyond the base-line (as we will show in section 6.1 for the case of performance-annotation).

For use in automatic performance annotation, the primary virtue of the edit-distance is that it is generic enough to work with arbitrary sets of (user-defined) edit-operations. This allows us to define a set of performance events that we consider adequate to cover all the types of expressive events occurring in musical performances (as we did in the previous subsection), and define edit-operations for each type of performance event. Using context-sensitive cost functions for edit-operations (subsection 4.2.1), even more complex performance-events such as ornamentation events can be accommodated for by the edit-operations, as will be shown shortly.

In the following subsection we will propose a cost model for the performance annotation scheme. In particular, we will define cost functions for of the edit-operations corresponding to each of the performance events.

4.3.1 Proposed Edit Cost Model

Through the functions that define the cost model, one can control what the optimal alignment between the sequences will be like, and therefore how the performance will be annotated. The performance annotation for a particular performance and score is far from arbitrary, since often listening to the performance while reading the score gives an unambiguous impression of the performance events involved. Of course human listeners will not be able to quantify in detail the deviations from the score, but it is generally easy to determine for example when an ornamentation occurs, or when two notes are contracted (consolidation). In this sense, it is generally unproblematic to speak of 'correct' annotations. Obviously, the aim of the automatic performance annotation process is to derive this correct annotation, and thus, the cost-model should be set up so as to minimize the difference between the generated annotation and the correct annotation.

The main factors that determine which of all the possible alignments between score and performance is optimal, will be on the one hand the features of the score and performance notes that are involved in calculating the cost of applying an operation, and on the other hand the relative costs of the operations with respect to each other.

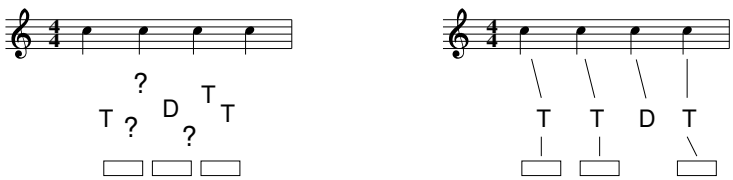

Figure 4.7: A note deletion event in the case of repeated notes with the same pitch forms a problem for mapping, if note positions are not taken into account (left). When note positions are known, the performed notes can be mapped to the nearest score note

Perhaps unsurprisingly, most transformation events (simple matches between one score note and one performance note) are identified correctly even with a relatively simple cost-model, such as presented by [Mongeau and Sankoff, 1990], that measures the difference in pitch and duration when considering a transformation operation, called 'replacement' in the context of comparing scores to scores. In more complicated situations, where e.g. deletions or consolidations occur, this simple model is not always adequate. Improvements can be made by incorporating the difference in position in the costs of the correspondence operations (transformation, consolidation and fragmentation). One situation where the alignment benefits from position information is when one note in a row of notes with the same pitch and duration is omitted in the performance, as illustrated in figure 4.7. Without taking into account positions, the optimal alignment will delete an arbitrary note of the sequence, since the deletions of each of these notes are equivalent when the cost of mapping is based on pitch and duration information only. When position *is* taken into account, the remaining notes of the performance will all be mapped to the nearest notes in the score, so the deletion operation will be performed on the score note that remains unmapped, which is often the desired result.

It is important to note that when combining different features, like pitch, duration and onset into a cost-value for an operation, the relative contribution of each term is rather arbitrary. For example when the cost of transforming one note into another would be defined as the difference in pitch plus the difference in duration, the outcome depends on the units of measure for each feature. The relative weight of duration and pitch is different if duration is measured in seconds, than if it is measured in beats. Similarly, pitch could be measured in frequency, semitones, scale steps, etcetera. Therefore, we have chosen a parametrized approach, in which the relative contribution of each attribute in the cost function is weighted by a constant parameter value (γ for pitch, δ for duration, and ϵ for position).

The other aspect of designing cost functions is the relative cost of each operation. After establishing the formula for calculating the costs of each operation, it may be that some operations should be systematically preferred to others. This independence of costs can be achieved by multiplying the cost of each operation by a factor (α) and adding a constant (β).

The cost functions w for the edit-operations are described below. The expressions $\mathcal{P}(x)$, $\mathcal{D}(x)$, $\mathcal{O}(x)$ respectively represent the pitch (as a MIDI number), duration and onset time of a score or performance element x.

Insertion/Deletion The costs of insertion (4.6) and deletion (4.7) operations both depend only on (and are proportional to) the duration of the inserted or deleted notes, as in [Mongeau and Sankoff, 1990]. This reflects the fact that insertions or deletions usually involve notes of short duration.

$$w^i(\emptyset, p_j) = \alpha^i \cdot \mathcal{D}(p_j) + \beta^i \tag{4.6}$$
$$w^d(s_i, \emptyset) = \alpha^d \cdot \mathcal{D}(s_i) + \beta^d \tag{4.7}$$

Transformation Transformation events involve a score and a performance note. The cost of the corresponding operation should be proportional to the sum of the difference in the note attributes. When a score note and a performance note have very similar attributes, it is very probable that the performance note is a (transformed) version of the score note. When the difference increases, the probability that the performance note is actually a transformation of the score note decreases. As explained earlier, (pairwise differences of) pitch, duration, and position are used to compute the cost of transformation operations (4.8).

$$w^t(s_i, p_j) = \alpha^t \cdot \begin{pmatrix} \gamma \cdot |\mathcal{P}(s_i) - \mathcal{P}(p_j)| + \\ \delta \cdot |\mathcal{D}(s_i) - \mathcal{D}(p_j)| + \\ \epsilon \cdot |\mathcal{O}(s_i) - \mathcal{O}(p_j)| \end{pmatrix} + \beta^t \tag{4.8}$$

Ornamentation The cost of ornamentation (4.9) is determined by the pitch relation of the ornamentation elements and the ornamented element (chromatically ascending sequences are preferred), and the total duration of the ornamentation elements.

Note the competitive relationship of the insertion and ornamentation operations, since they both account for performance elements that have no corresponding score element. We want very short notes with a certain pitch relationship to subsequent notes to be matched as ornamentations. If the duration of the notes exceeds a certain limit, or the pitch relation is not satisfied, the notes should be matched as insertions instead. This can be achieved by setting β^i higher than β^o and α^i lower than α^o. Figure 4.8 shows how the cost values of insertion and ornamentation vary with note duration. Before the point where the lines intersect (shorter durations), the notes will be accounted for as ornamentations; after this point (longer durations) the notes will be accounted for as insertions.

$$w^o(\emptyset, p_j, \cdots, p_{j+L+1}) = \alpha^o \cdot \begin{pmatrix} \gamma \cdot \sum_{l=1}^{L} |1 + \mathcal{P}(p_{j+l}) - \mathcal{P}(p_{j+l-1})| + \\ \delta \cdot \sum_{l=0}^{L} \mathcal{D}(p_{j+l}) \end{pmatrix} + \beta^o \tag{4.9}$$

Fragmentation/Consolidation Fragmentations and consolidations are similar to transformations since they all map score notes to performance notes. In fragmentation, a large score note is performed as several shorter notes. The pitch of the performed notes should thus be the same as that of the score note (c.q. the summed difference should be zero), and

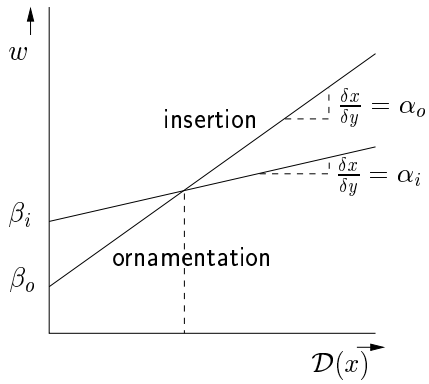

Figure 4.8: Cost values for insertion and ornamentation operations as a function of note duration

the summed duration of the performance notes should be close to the duration of the score note. Furthermore, the onset of the first performance note should be close to the onset of the score note. This is reflected in the cost function (4.10). The cost of consolidation (4.11) is defined conversely, but completely analogous.

$$w^f(s_i, p_j, ..., p_{j+L}) = \alpha^f \cdot \begin{pmatrix} \gamma \cdot \sum_{l=0}^{L} |\mathcal{P}(s_i) - \mathcal{P}(p_{j+l})| + \\ \delta \cdot |\mathcal{D}(s_i) - \sum_{l=0}^{L} \mathcal{D}(p_{j+l})| + \\ \epsilon \cdot |\mathcal{O}(s_i) - \mathcal{O}(p_j)| \end{pmatrix} + \beta^f \quad (4.10)$$

$$w^c(s_i, ..., s_{i+K}, p_j) = \alpha^c \cdot \begin{pmatrix} \gamma \cdot \sum_{k=0}^{K} |\mathcal{P}(s_{i+k}) - \mathcal{P}(p_j)| + \\ \delta \cdot |\mathcal{D}(p_j) - \sum_{k=0}^{K} \mathcal{D}(s_{i+k})| + \\ \epsilon \cdot |\mathcal{O}(s_i) - \mathcal{O}(p_j)| \end{pmatrix} + \beta^c \quad (4.11)$$

The costs of transformation (4.8), consolidation (4.11), and fragmentation (4.10), are principally constituted by the differences in pitch, duration and onset times between the compared elements. In the case of one-to-many matching (fragmentation) or many-to-one (consolidation), the difference in pitch is calculated as the sum of the differences between the pitch of the single element and the pitches of the multiple elements. The difference in duration is computed between the duration of the single element and the sum of the durations of the multiple elements. The difference in onset is computed between the onset of the single element and the onset of the first of the multiple elements. The cost of insertion (4.6) and deletion (4.7) is determined by the duration of the deleted element.

From a musical perspective, it can be argued that the cost of matching two elements with different pitches should not depend on the difference of the absolute pitches, but rather on the different roles the pitches play with respect to the underlying harmonies, or their scale

degree, as these features have more perceptual relevance than absolute pitch difference. This would certainly be essential in order to make good alignments between scores and performances that very liberally paraphrase the score (e.g. improvisations on a melody) and also in the case where alignment is constructed for assessing the similarity between different scores. In our case however, we currently deal with performances that are relatively 'clean' interpretations of the score (rather than improvisations). As such, changes of pitch are very uncommon in our data. Still, it is desirable to have a more sophisticated pitch comparison approach, to accommodate more liberal performances in the future.

4.4 Connection of the Musical Data

In the previous sections we have shown how two secondary representations of the score and the performance can be derived, respectively the I-R analysis and the performance annotation. Together, the primary representations (score, and performance) and the secondary representations (I-R analysis, and performance-annotation) form an enriched description of the musical data. In order to employ the representations to their full extent, it is essential that they are linked together, so as to make clear the structural relations that exist between them.

The linking between the representations will be used for example when deriving cases from proto-cases, based on the melodic segmentations. The constructive adaptation step also depends on links between score and performance-events.

In chapter 3, we discussed the models we use to represent the different kinds of data. In all models, the data representation is of sequential nature, although there is not a one-to-one mapping between the elements of every sequence. For example, a note may belong to two I-R structures at the same time, and inserted notes in the performance have no counterpart in the score. To represent such facts, we define relations between the elements of the various sequences. The result is a vertically layered structure.

Figure 4.9 shows an example of such a structure, for the beginning of a (fictitious) musical phrase. It involves representations of five different kinds of data: the melody, its I-R analysis as produced by the I-R parser described in section 4.1, a performance of the melody, and the annotation of that performance. The sequential nature of the representations is expressed by the *next* predicate that connects subsequent elements of sequences.

The relation between notes and I-R structures is expressed by the *belongs-to* predicate from notes to I-R structures, and the predicates *first-note*, *middle-note*, and *last-note*, that specify which note plays which role in the I-R structure. These relations serve for example to segment the melody by I-R groups, or to identify the melodic context of a note.

Lastly, the performance and the score melody are connected. But rather than connecting them directly through predicates, the performance annotation serves to express the relations between score elements and performance elements. Figure 4.9 illustrates the differences between the various abstract types of performance events (shown in figure 3.5): Score reference events and performance reference events are identified by their relations to score elements, and performance elements, respectively. Correspondence events have both the *score-reference* and the *performance-reference* relations. Note that between the first transformation event and the ornamentation event that precedes it, a relation *ornamented-by* exists, which expresses the dependence between ornamentations and their successor notes

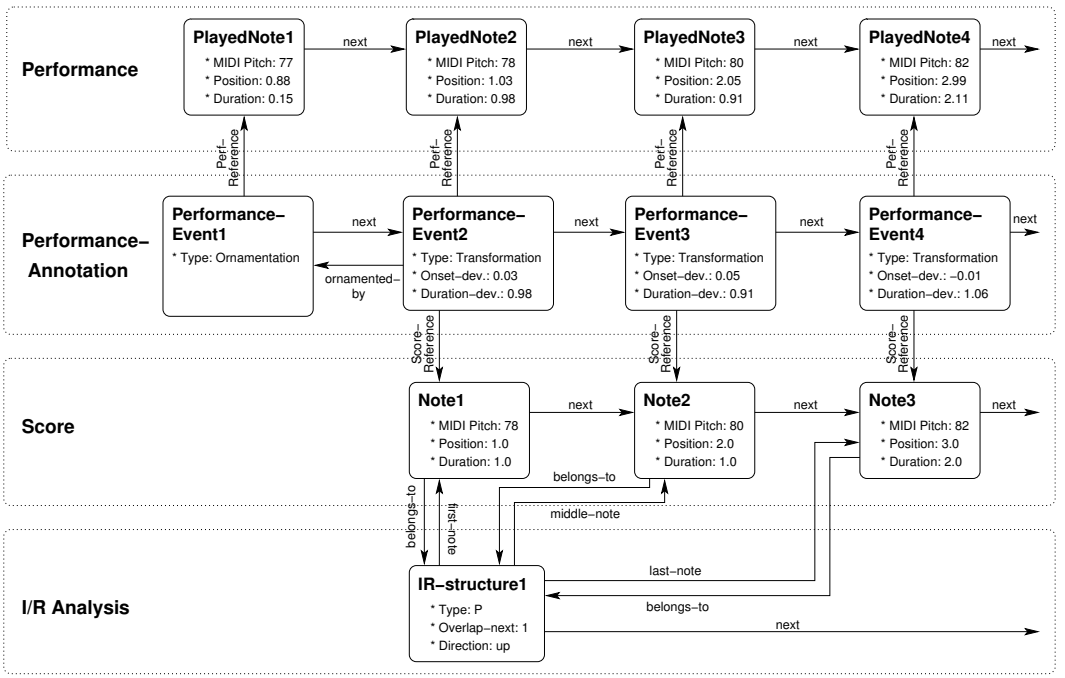

Figure 4.9: An example of interconnected representations of the different kinds of musical data

(ornamentation notes do not occur in isolation; moreover, the pitches of ornamentation notes are usually related to the successor note pitch).

The vertical interconnections between the sequences serve to retrieve the 'vertical musical context' of any element. For example, the context of a particular score note n can be defined to be the set of all sequence elements that are directly or indirectly connected to n, through any relation except the *next* relation. This yields the I-R structure n belongs to, the performance event that refers to n, and (depending on the type of performance-event) the performed-note that represents n in the performance. Any ornamentation events that occur before the performance event corresponding to n will also be included, due to the *ornamented-by* relation.

Chapter 5

Problem Solving

In the previous chapter we have described the knowledge acquisition aspects of TempoExpress. This included the process of deriving knowledge-rich representations of melodies and performances, to be used as a specification of the input problem, as well as forming cases that can be used to solve tempo transformation problems.

In this chapter we explain how this musical information is stored in proto cases and cases (see chapter 3, section 3.1), and we will address the part of the system that uses the obtained cases in order to solve new tempo-transformation problems (see figure 5.1). In particular, we will focus on the retrieval and reuse processes of the CBR approach that is employed.

In chapter 3, section 3.1, we also outlined the problem solving process, illustrated in figure 3.2. It involves an initial retrieval step at the phrase level. In this step, proto cases are selected from the proto case base. The selected proto cases are segmented to form a set of cases that are used in a combined retrieval and adaptation step called constructive adaptation. In this step solutions are constructed for every segment of the input problem and those solutions are concatenated to form a solution for the phrase input problem.

In the following subsections, we will address each of the steps. First we will look at the concept of cases and proto cases in more detail, and how the data is organized in proto cases and cases (section 5.1). Proto case retrieval is explained in section 5.2, and section 5.3 is devoted to the use of constructive adaptation and the transfer of expressive features from retrieved cases to the input problem segments.

5.1 Cases and Proto Cases

Cases in a case based reasoning system are composite units of information that embody the knowledge to solve problems. They describe a particular problem and contain a solution to that problem. Additionally, meta-information of different kinds may be present (see section 2.6). The problem description part of the cases is used for indexing/retrieving cases from a case base, given a problem description for which the CBR system should find a solution (the input problem). The solutions of the retrieved cases are employed in the adaptation step to form a solution to the input problem, either by transforming a retrieved solution (*transformational reuse*) or by constructing a new solution where the retrieved solution is

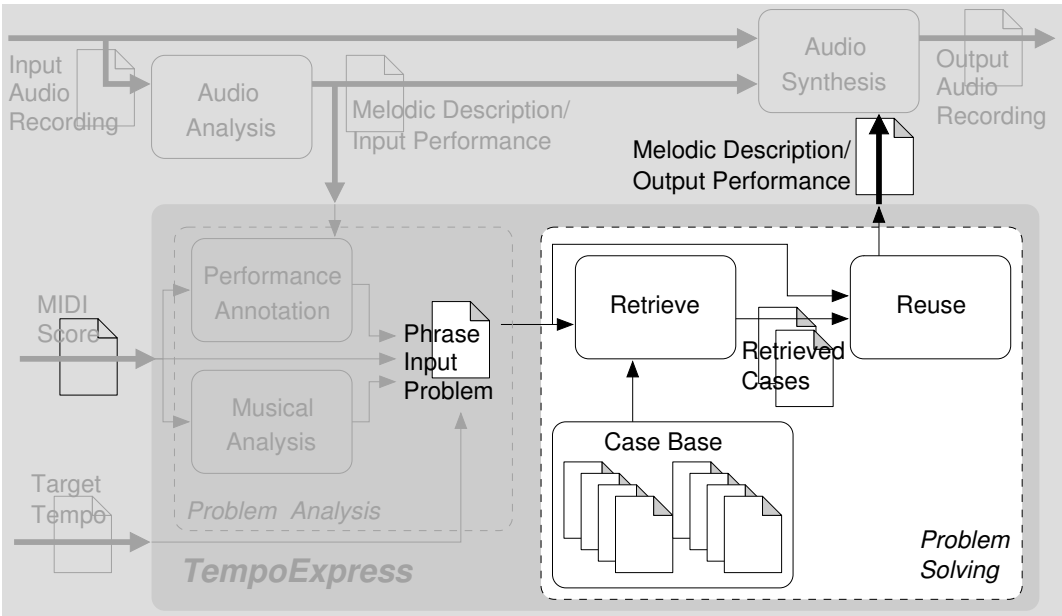

Figure 5.1: The problem solving part of TempoExpress

used as a guide (*generative reuse*). The meta-information can be used for example to guide the retrieval (e.g. in the case of time-stamps), or the adaptation step.

As mentioned above, TempoExpress deals with two different types of cases. The tempo transformation knowledge is stored in the system in a permanent way as proto cases in a proto case base. When the system receives the input that is necessary to perform a tempo transformation, an actual case base is created dynamically from the proto cases.

5.1.1 Proto Cases

One particular characteristic of the problem domain we are considering is that a single musical phrase has several performances, as discussed in section 3.2. Given that many performances pertain to a single phrase, it makes sense intuitively to store them together in a single case. But from the point of view of tempo-transformation this doesn't yield useful cases. Since the tempo-transformation task is to change a performance at one tempo to a performance at another tempo, only performances at (or close to) those tempos will actually serve as precedents for a particular problem. This implies firstly that not all performances for a given phrase will be used for a particular problem. Secondly, many tempo-transformation cases can be constructed using the performances of a single phrase. since any of the available performances is potentially part of a problem description, or a solution, as long as the source and target tempos have not been specified. Thus, a proto case holds the information for more than just one tempo transformation. More precisely, when the proto case contains performances at n different tempos, any of them may occur as an input performance paired with any other as output performance. If we exclude identity tempo transformations (where

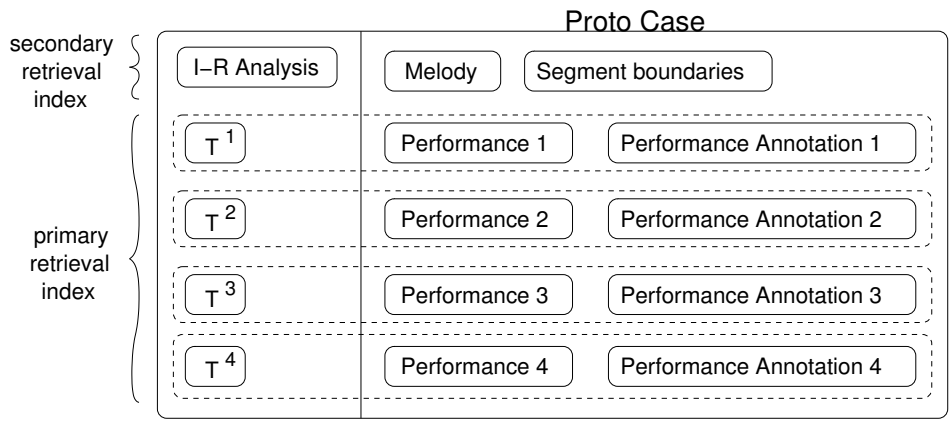

Figure 5.2: The information contained in a proto case

the same performance serves both as input and output performance), this yields $n(n-1)$ possible tempo transformations.

To store the cases for all possible tempo transformations explicitly is inefficient. Firstly because it would involve a vast multiplication of problem description data like the musical phrase and its musical analysis, that are identical for many cases. Secondly, it is known a priori that the usability of a particular case excludes the usability of many other cases, namely all cases that solve tempo-transformations of the same phrase as the usable case, but for source and target tempos that are very different from the source and target tempo of the usable case. For example, when a case with a tempo transformation from 50 to 120 BPM is useful for a particular problem (because these tempos are close to the source and target tempos in the input problem description), it is clear that the inverse tempo transformation, from 120 to 50 BPM will not be relevant, nor other any tempo combinations where the source and target tempos are very different from 50 and 120 BPM respectively. In other words, the case base would be cluttered with many cases of which only a very small number can possibly be used per problem solving episode.

To deal with this problem, the musical information is stored in the system in the form of proto cases. The proto cases are not used directly to solve tempo transformation problems; Instead, they form the material from which a case base is constructed dynamically. Figure 5.2 shows a schematic view of a proto case, with the information that is contained in it. Note that there is no explicit labeling of any data as being part of the problem description or the solution. Rather, the performance information at all available tempos ($T^1 \cdots T^4$) is present. The items of the left side, the tempo-specifications and the I-R analysis, will be used as primary and secondary indices for retrieval, respectively (explained in section 5.2).

5.1.2 Cases

Cases, as opposed to proto cases, correspond to a *specific* tempo-transformation problem, mentioning a source and a target tempo, and contain a solution for that problem, in the form of a performance, and performance annotation at the target tempo.

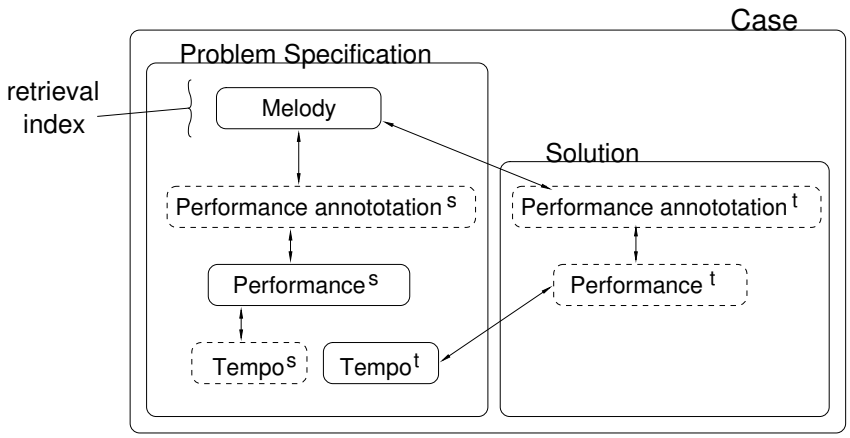

Figure 5.3: Schematic overview of a case, containing a problem description and solution

Another difference with proto cases is that cases provide solutions to tempo-transformations of phrase segments instead of complete phrases.

This case layout is illustrated in figure 5.3. The solid boxes represent information that is provided as input; the dashed boxes represent knowledge that is derived from the input information; arrows denote the presence of structural relations between elements. The names tagged with the superscript s pertain to the source tempo, and the names tagged with t pertain to the target tempo.

Note that the I-R analysis, although present in the proto cases, is not present in the actual cases, because it is used only in the proto case retrieval, not in the retrieval of cases.

5.2 Proto Case Retrieval

In the case retrieval phase, those cases that are expected to be for solving the current problem are retrieved from the case base. Using the retrieved cases, in the reuse phase a solution is constructed for the input problem. The retrieval phase in TempoExpress can be divided into three steps: tempo filtering, melodic phrase comparison, and phrase segment comparison. The motivation for these steps follows from the basic assumption in CBR that similar problems have similar solutions (see section 2.6). This implies that to solve a new problem, the retrieval step should return the cases with the most similar problems to the input problem, since their solutions are assumed to be closest to the desired solution. Input problem specifications in TempoExpress comprise primarily the source and target tempos for the tempo-transformation, and the melody (and its analysis) of the performance to be transformed. Comparing input performances (which are also part of the problem specification) is not useful, since apart from the tempo, the cases do not exemplify any explicitly described way of performing the melodies. Input performances do play a crucial role in the reuse phase however.

Note that retrieval, when it is taken to refer to functional aspects of the CBR process, is not necessarily a single processing phase that is carried out once. Case retrieval may

happen in several non-consecutive steps (possibly interleaved with case reuse). Of the three retrieval steps mentioned above, only the first two steps, tempo filtering and melodic phrase comparison, are executed prior to case reuse, and only once per problem solving cycle (i.e. once per tempo transformation). The last step, phrase segment comparison, is done once for every input segment, and precedes the reuse of that segment.

5.2.1 Tempo Filtering

The most straight-forward step in the case retrieval is selecting the cases with performances at the relevant tempos. It follows logically from the claim that lies at the heart of this work, namely, that performances of the same melody are increasingly dissimilar with increasingly different tempos (see section 1.1 for a more elaborate discussion). It can be concluded that in order for a case to be relevant, it must have an input performance at approximately the source tempo, and an output performance at approximately the target tempo. We will allow for a certain mismatch between the tempos because requiring the tempos to be identical is likely to be overly restrictive: the expressivity of a performance that sound good at a particular tempo will probably also sound good for a tempo about 5 BPM faster or slower. We have defined the tempo tolerance window to be 10 BPM in both upward and downward directions. For example, a tempo transformation from 80 BPM to 140 BPM may serve as a precedent for tempo transformation from 70 BPM to 150 BPM. This particular tolerance range (which we feel may be too nonrestrictive), is mainly pragmatically motivated: In our corpus, different performances of the same phrase are often at 10 BPM apart from each other. Therefore, a $<$10 BPM tempo tolerance will severely reduce the number of available precedents, compared to a $\geq$10 BPM tempo tolerance.

The tempo filtering procedure is applied before the proto-cases (see section 5.1.1) are converted into actual cases. If a proto-case, which contains the performances at every available tempo for a particular phrase, does not have performances within the tolerance region of both the source and the target tempo, it is considered as irrelevant for solving the current problem. That is, no actual case can be constructed from that proto-case, as it should contain a performance close to the source tempo as input-performance and a performance close to the target tempo as output-performance. This procedure is illustrated in figure 5.4. Vertical bars under the phrases represent performances of that phrase, at a tempo that is proportional to the vertical position of the bars.

5.2.2 Melodic Phrase Comparison

After filtering the proto case base by the source and the target tempos of the input problem, a melodic similarity value is calculated between the input phrase and the phrases in the proto cases. The purpose of this rating is to make a rough assessment of the similarity on the phrase level, so as to prevent the reuse of segments that coincidentally match well with an input segment, but belong to a phrase that is radically different from the input phrase as a whole. For example, one can imagine that a very short fragment of a slow ballad might be similar to a fragment of a bebop style melody, but the expressive performance of the fragment will likely be rather different, so the ballad will not make a good precedent for the bebop phrase.

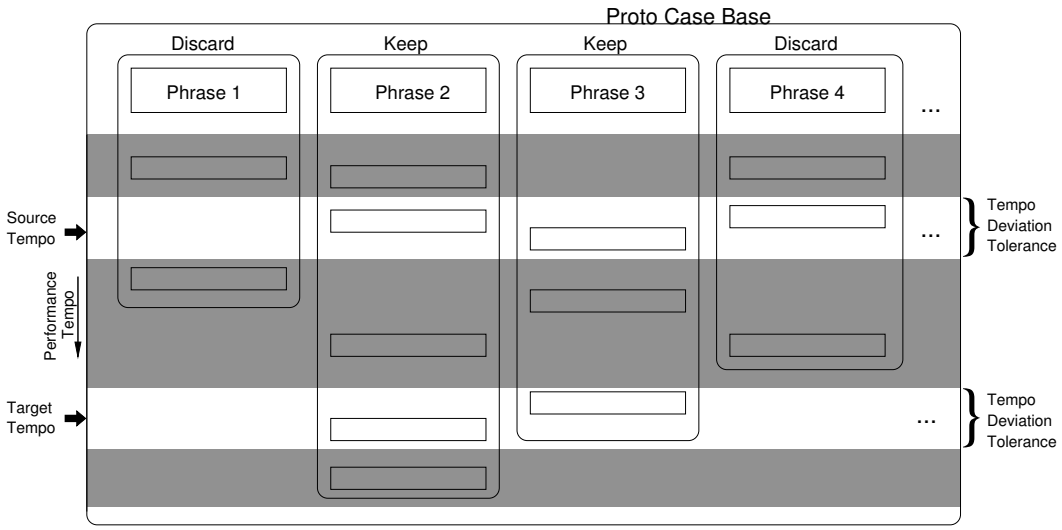

Figure 5.4: Filtering cases using the source and target tempo

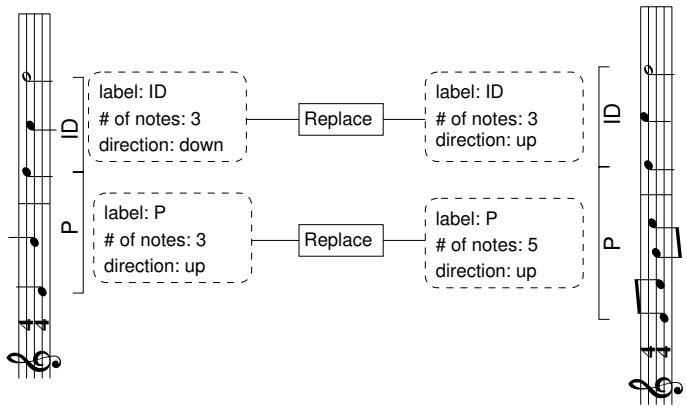

Figure 5.5: comparison of melodies using an I-R based edit-distance

Note that the purpose of making rough and global similarity measurements (e.g. for clustering) requires different characteristics from a melodic similarity measure than for example the typical music information retrieval task of finding the single most similar melody to a given query. For the latter application, the main concern is that the discriminative power of the measure is good in the highest similarity regions (e.g. when the compared melodies are almost identical). On the other hand, for an overall similarity assessment of a set of melodies, the similarity measure should also be discriminative when the melodies are not very much alike (melodies within the same style can be rather different and may share only some global features).

In section 6.2.1 we present a comparison of melodic similarity measures using melody representations with different levels of abstraction. It turns out that there is roughly a trade-off between the two capabilities mentioned in the previous paragraph, and that more abstract melody representations (like the I-R analysis and pitch contour) are more discriminative on a wide range of divergent melodies than concrete representations such as absolute pitch with duration. We have therefore chosen to use a similarity measure that compares I-R analyses rather than the melodies at the note level. Since the I-R analyses are of a sequential nature, the similarity between them can be assessed using the edit-distance. To do this it is necessary to define the edit operations that can be applied to the individual I-R structures, and the functions that compute the costs of such operations. Although specialized operations such as consolidation and fragmentation have been proposed for computing the edit-distance between note sequences [Mongeau and Sankoff, 1990], we decided to use the three canonical edit operations, insertion, deletion, and replacement. Costs of deletion and insertion of I-R structures are proportional to the number of notes spanned by the I-R structure, and replacement is a weighted sum of differences between attributes of the I-R structures, plus an additional cost if the I-R structures under replacement do not have the same label. The latter cost is reduced if the labels are semantically related, that is, one of the structures is the retrospective counterpart of the other.

The cost functions of the edit-operations are parametrized to allow for control and fine-tuning of the edit-distance. They are defined as follows:

$$w(s_i, \emptyset) = \alpha_d \mathcal{S}ize(s_i) \tag{5.1}$$

$$w(\emptyset, s_j) = \alpha_i \mathcal{S}ize(s_j) \tag{5.2}$$

$$w(s_i, s_j) = \alpha_r \begin{pmatrix} \beta \mathit{LabelDiff}(s_i, s_j) + \\ \gamma \,|\, \mathit{Size}(s_i) - \mathit{Size}(s_j) \,| + \\ \delta \,|\, \mathit{Dir}(s_i) - \mathit{Dir}(s_j) \,| + \\ \epsilon \,|\, \mathit{Overlap}(s_i) - \mathit{Overlap}(s_j) \,| \end{pmatrix} \tag{5.3}$$

$$\mathit{LabelDiff}(s_i, s_j) = \begin{cases} 0 & \mathit{Label}(s_i) = \mathit{Label}(s_j) \\ \zeta & \mathit{Label}(s_i) = -\mathit{Label}(s_j) \\ 1 & \mathit{otherwise} \end{cases}$$

where $w(s_i, \emptyset)$ is the cost of deleting I-R structure s_i from the source sequence; $w(\emptyset, s_j)$ is the cost of inserting I-R structure s_j into the target sequence; and $w(s_i, s_j)$ is the cost of replacing element s_i from the source sequence by s_j from the target sequence.

Label, *Size*, *Dir*, and *Overlap* are functions that, given an I-R structure, respectively return its I-R label (encoded as an integer), its size (number of notes spanned), its melodic direction, and its overlap (the number of notes belonging to both the current I-R structure and its successor). *LabelDiff* is an additional function that determines the part of the replacement cost due to difference/equality of I-R labels. The I-R labels are mapped to integer values in such a way that the integer for the retrospective counterpart of a particular label is always the negative of the integer of that label.

The parameters come in two kinds: Firstly there are the parameters that are used to control the relative costs of the operations, α_i, α_d, and α_r. For example by setting α_i and α_d to relatively low values, the optimal alignment is more likely to include insertions and

deletions of elements than replacements. The second kind of parameters, including β, γ, δ, ϵ, and ζ, regulate impact of differences in attributes. For example, a relatively high γ value will make the replacement cost function more sensitive to size differences (leading to higher a cost if the sizes differ).

The parameter values of the cost functions were determined experimentally, by optimizing them to make the edit-distance measure mimic human similarity judgments. This is reported in section 6.2.2.

With this distance measure, the distance to the I-R analysis of the input problem is assessed for the I-R analyses of every proto case. The proto cases closest to the input problem are selected as the basis to form the case base. A parameter defines the proportion of most similar proto cases that is retrieved. The parameter value used by default is 0.5.

5.2.3 Case Creation from Proto Cases

From the final set of proto cases that has been selected, the actual case base is constructed. This consists in partitioning the music representations present in the proto cases, and storing the segments in separate cases. In section 3.5 we have discussed various melody segmentation strategies, and argued that non-overlapping, musical grouping (using the Melisma Grouper) is the most appropriate strategy in our application. The melody segmentation has been performed in the proto case construction/problem description phase, and is stored in the proto case (in the form of note indices that represent the segment boundaries).

The melody segmentation is used as the primary segmentation, from which the segmentations of other representations are derived. These secondary segmentations are derived by employing the structural relations defined between the elements of different representations, as outlined in section 4.4. For example, the performance annotation segment corresponding to a given melodic segment is found by taking the first note and the last note n_i and n_j from a given segment, and finding the corresponding performance-events, that is, the events e_k and e_l for which $\mathit{scoreReference}(e_k, n_i)$ and $\mathit{scoreReference}(e_l, n_j)$ holds, respectively. The performance annotation segment is defined as the subsequence $\mathbf{e}_{k:l}$ of the performance annotation $\mathbf{e}$ of the complete phrase. The corresponding performance segment is delimited by the performed notes p_m and p_n for which $\mathit{scoreReference}(e_k, p_m)$ and $\mathit{scoreReference}(e_l, p_n)$ holds, respectively.

Segmentation of Performance and Performance Annotation

Due to the fact that the performance events that constitute the performance annotation are not necessarily one-one correspondences between the melody and the performance, some non-trivial situations may occur. For example, in case the first performance event of the segment e_k is preceded by any non-score-reference events (ornamentation, or insertion), the first of these events will mark the beginning of the performance annotation segment rather than e_k. This situation is shown in figure 5.6 (left). The reason for joining those events at the beginning of the segment rather than at the end of the previous segment is that from a perceptual point of view, they are normally connected to the note that follows (especially in the case of ornamentation events).

Figure 5.6(right) shows another situation, where the last note of the segment in the performance is played as a consolidation with the first note of the next segment. In such

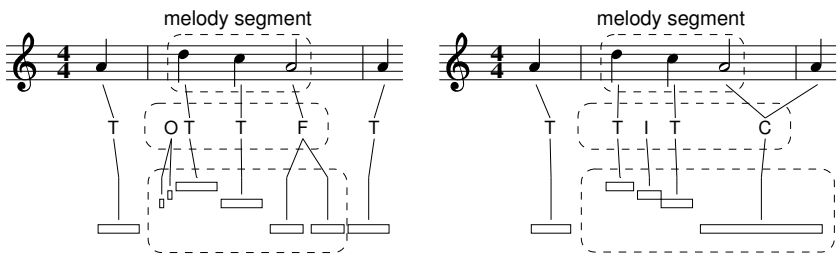

Figure 5.6: Two examples of the derivation of performance annotation and performance segments from melody segments

cases, the latter consolidated note will be included in the melody segment, since otherwise, an inconsistent situation would occur, where a single note is performed as a consolidation. Note that with melodic segmentations that follow musical grouping, this situation is very unlikely, since the musician usually expresses the musical structure of the melody to some extent, e.g. by a short silence. On the other hand, a consolidation event that bridges the segment boundaries effectively masks the musical grouping structure, and thus is improbable.

5.3 Constructive Adaptation

In this step a performance of the input score is generated at the target tempo, based on the input performance and the set of matching cases. Constructive adaptation (CA) [Plaza and Arcos, 2002] is a technique for case reuse that constructs a solution by a search process through the space of partial solutions of the problem. Partial solutions are represented as states. Furthermore, a 'hypothesis generation' function is defined for generating a set of successor states for a given state. The search process is started by defining an initial state that corresponds to the empty solution, the state where no part of the solution is configured. The state space is constructed by applying the hypothesis generation function exhaustively to the initial state and its (direct and indirect) successors. A state is a goal state when it satisfies the constraints defined by the input problem, tested by a 'goal test' function. The order of expansion of states is controlled by a 'hypothesis ordering' function, that orders the states in a best-first manner. The constructive adaptation process is expressed in pseudo code below (adapted from [Plaza and Arcos, 2002]):

```
Initialize OS = (list (Initial-State P))
Function CA(OS)
        Case (null OS) then No-Solution
        Case (Goal-Test (first OS)) then (S2C (first OS))
        Case else
                Let SS = (HG (first OS))
                Let OS = (HO (append SS (rest OS)))
                        (CA OS)
```

The functions HG and HO realize hypothesis generation and hypothesis ordering, respectively. Domain knowledge, as provided by past cases, is typically employed inside these two functions. For example, given the input problem, cases can be retrieved and reused to propose a solution to a part of the problem. They may also be used to determine the order of expansion of states. But the employment of domain knowledge is not limited to cases. Domain-knowledge may also be stated in the form of rules, or otherwise.

The variables *OS* and *SS* are the lists of open states (yet to be expanded) and successor states (the states resulting from the current expansion). The function `Initial-State` maps the input problem description *P* into a state. The function `S2C` maps the solution state (the state for which `Goal-Test` succeeded) into the configuration of the solution. Although the use of domain knowledge is normally most prominent in the functions `HG` and `HO`, the functions `S2C`, `Initial-State`, and `Goal-Test` are also domain-specific.

In TempoExpress states have three components: A list of input problems and a list of corresponding solutions (the input problems and solutions are of the kind that cases contain, see figure 5.3), and lastly, a list of quality values for every solution. The quality values are numbers in the range $[0, 1]$ that indicate the estimated quality of the corresponding solutions. A partial solution to a phrase input problem is defined as a state where zero or more of the (segment) input problems have a corresponding solution and quality. In the initial state, none of the input problems have a solution (and hence no quality estimate).

To expand this state into successor states, an input problem is selected from the set of unconfigured input problems, and solutions are generated for this problem. Different solutions can be generated, using different retrieved cases. The procedure is expressed as pseudo-code below:

```
Function HypothesisGeneration(State)
    NewStates = emptySet
    Problem = (selectUnconfiguredInputProblem State)
    Cases = (retrieveRelevantCases Problem)
    For each Case in Cases
        Let Solution = (solve Problem Case)
        Let Quality = (quality S)
        NewStates = (append (makeState Problem Solution Quality) NewStates)
    NewStates
```

The function `selectUnconfiguredInputProblem` picks the input problem for which solutions will be constructed. The current implementation of the function picks the first unconfigured input problem it encounters in the list. The `retrieveRelevantCases` function uses the case base that was constructed based on the filtered set of proto cases, to find the cases with phrase segments that best match the phrase segment of the input problem. Each of the retrieved cases are then used in the `solve` function to construct a new solution for the current input problem. Section 5.4 explains the functionality of `retrieveRelevantCases` and `solve` in more detail.

The process of state expansion/hypothesis generation is illustrated in figure 5.7. The figure shows states with three input problems (the score fragments), where different states provide different solutions to the problems (the solutions being the performed notes, repre-

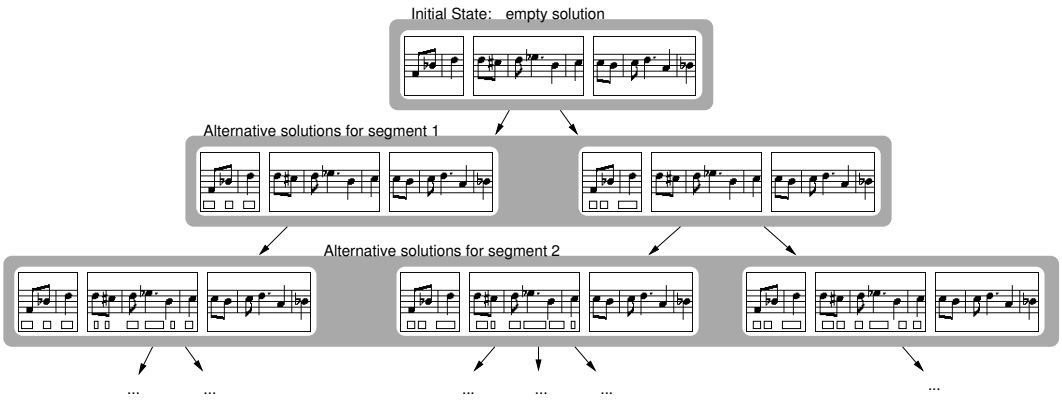

Figure 5.7: Constructive adaptation: The segment-wise construction of the solution through state expansion

sented by the bars underneath the score).

When a state is reached that satisfies the goal state criteria, the solutions are concatenated to form the solution for the phrase problem specification. Otherwise, the state expansion is repeated, by solving one of the remaining unsolved input problems.

The goal state criteria require that a state has a solution for every input problem, and that the overall estimated *solution quality* of the solutions is maximal. The quality of a solution is estimated as the proportion of notes in the problem score segment for which performance events could be inferred based on the retrieved case. This proportion depends on the matching quality between problem score and retrieved score segment, and the availability of a matching adaptation rule, given the performance annotations in the problem and the case.

Independence is assumed between the solution qualities of the input problems, and thus the solution quality of the solution to a phrase input problem is defined as the average quality of the segment solutions, weighted by the segment length. Therefore, a best first search that expands states in the order of their solution quality is guaranteed to find the solution with maximal quality.

Although as of now no constraints have been defined for regulating interdependence of segment solutions, note that such constraints can easily be incorporated through CA. A constraint can take the form of a rule that prescribes a decrease or increase of the overall solution quality, based on some (probably high level) description of two or more segment solutions. Of course this may introduce local maxima in the search space, and the search strategy employed will become more crucial.

5.4 Case Adaptation at Phrase Segment Level

In this section we will explain in detail how a solution is obtained for a segment input problem. This is the process that constitutes the state expansion mentioned before. Figure 5.8 shows an example of the reuse of a retrieved case for a particular input segment. We will

explain the numbered boxes in the figure one by one.

5.4.1 Melody Segment Retrieval/Alignment

Box 1 denotes the comparison of the input score segment to those of the cases in the case base in order to find the case that is most similar to the input problem. The similarity is assessed by calculating the edit-distance between the score segments. The edit-distance now operates on notes rather than on I-R structures, to have a finer grained similarity assessment.

In box 2, a mapping between the input segment and the best matching retrieved segment is made, using the optimal alignment found through the calculation of the edit-distance that was used in box 1 to compute the similarity. The purpose of the edit-distance is thus twofold: To compute the distance between the problem and the cases, and to derive a mapping between the score segment and the retrieved segment. There is a clear relation between the two. The mapping is of prime importance, since the more correspondences can be established between the notes of the input segment and the notes of the retrieved segment, the more likely it is that the case can be reused (the correspondences are necessary, not sufficient conditions for reuse). By configuring the cost of edit-operations in such a way that correspondences are preferred over insertions/deletions, at the same time one ensures that the most reusable cases are rated as closest to the problem.

Of course there is a trade-off between the number of correspondences in the mapping, and the effectivity of reuse, since not all cases are expected to provide good solutions for the current problem (e.g. cases with melodies that are radically different from the input segment). Ideally, correspondences between problem segment notes and retrieved segment notes should only be established when it benefits the solution. Therefore, an edit cost scenario that facilitates replacements too much is not a good option either.

5.4.2 Linking of Performance Events

In box 3, the performance annotations corresponding to the relevant tempos are extracted from the retrieved segment case and the input problem specification (both the source tempo T^s and the target tempo T^t for the retrieved segment case, and only T^s from the input problem specification).

Box 4 shows the linking of the performance events of the retrieved segment at T^s and T^t to the performance events of the input segment at T^s. From the way the performance event scheme was defined it follows that the general form of a performance of a score $\mathbf{m} = (m_0, \cdots, m_t)$ is:

$$(\mathbf{P}_0, \mathbf{S}_0, \cdots, \mathbf{P}_t, \mathbf{S}_t) \tag{5.4}$$

where $\mathbf{P}_i \in \{O, I, \emptyset\}$, and $\mathbf{S}_i \in \{D, T, C, F\}$, $0 \leq i \leq t$. This is convenient because it allows us to group performance events per score note. The performance events corresponding to a note m_i are thus $(\mathbf{P}_i, \mathbf{S}_i)$. That is, a note is always represented in the performance annotation by zero or one (non-correspondence) performance reference event, followed by exactly one score reference event[1].

[1] Actually the scheme does not prohibit multiple insertion events to occur subsequently or after the last score reference event. But since this never occurred in the corpus, we eliminated these possibilities by imposing that an insertion event must always precede a score reference, the constraint that also holds for ornamentation events.

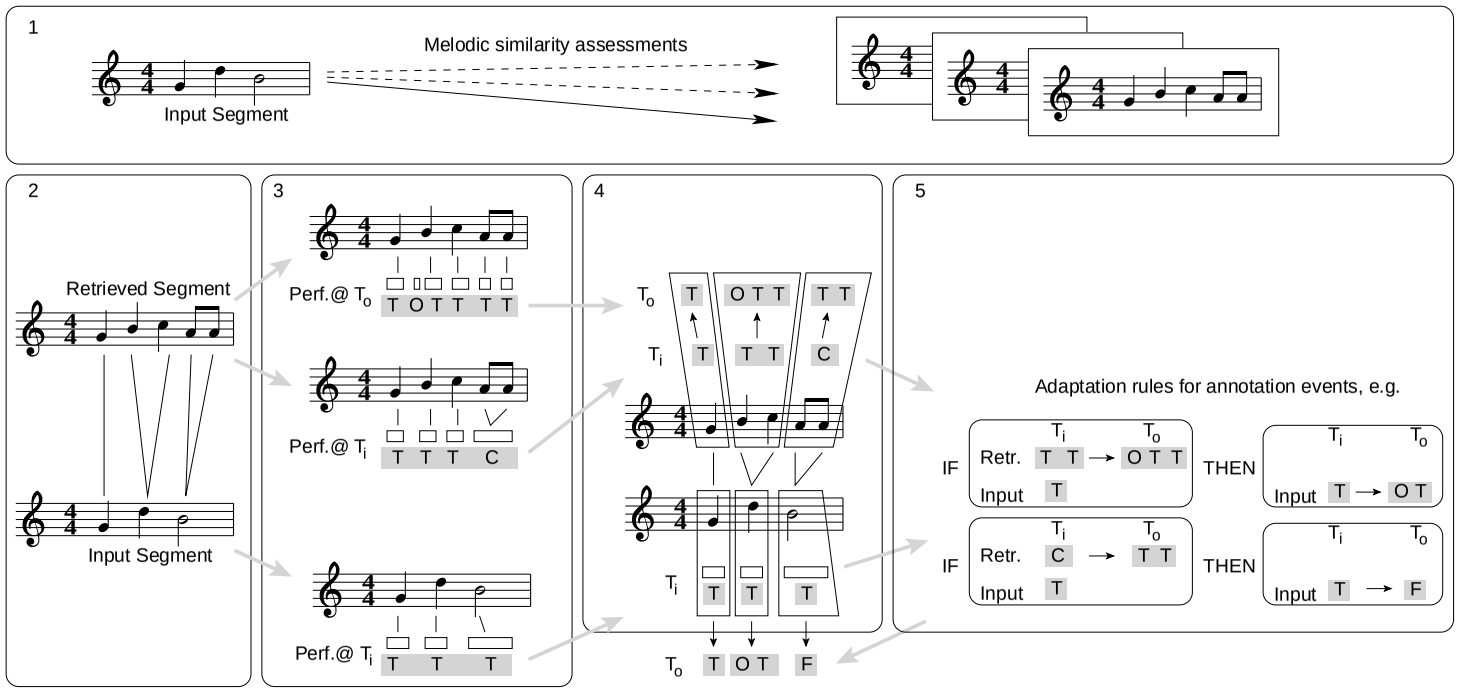

Figure 5.8: Example of case reuse for a melodic phrase segment; T^s and T^t refer to source and target tempo, respectively; The letters T, O, C, and F in the performance annotations (gray bars), respectively represent transformation, ornamentation, consolidation, and fragmentation events

Let $(m_i, \cdots, m_j)$ be the subsequence of a score $\mathbf{m} = (m_0, \cdots, m_t)$, and $P_T(\mathbf{m}) = (p_0, \cdots, p_v)$ be the performance of $\mathbf{m}$ at tempo T, then we define:

Definition 1 *The* slice $P_T((m_i, \cdots, m_j))$ *of* $\mathbf{m}$ *is the unique subsequence* $(p_k, \cdots, p_l)$ *of* $P_T(\mathbf{m})$*, having the form* $(\mathbf{P}_i, \mathbf{S}_i, \cdots, \mathbf{P}_j, \mathbf{S}_j)$ *where* $\mathbf{S}_i$ *is the score reference event referring to* m_i*, and* $\mathbf{P}_i$ *is any performance reference event that precedes* $\mathbf{S}_i$ *or null in case* $\mathbf{S}_i$ *is not preceded by a performance reference event.*

The mapping between the input segment and the retrieved segment is used to determine which performance events from the retrieved case belong to which performance event from the input problem, as shown in box 4 from the figure. This is done as follows: Suppose that the alignment of the input score $\mathbf{m}$ and the retrieved score $\mathbf{n}$ determines that the elements $(m_i, \cdots, m_j)$ correspond to $(n_k, \cdots, n_l)$. The mapping be either one-to-one ($i = j$, $k = l$), one-to-many ($i = j$, $k < l$), or many-to-one ($i < j$, $k = l$). Given a performance of $\mathbf{m}$ at tempo T^s, and two performances of $\mathbf{n}$ at tempos T^s and T^t respectively, we define:

Definition 2 *The* annotation triple *of* $(m_i, \cdots, m_j)$ *and* $(n_k, \cdots, n_l)$ *for tempos* T^s *and* T^t *is the triple of slices* $(P_{T^s}((m_i, \cdots, m_j)), P_{T^s}((n_k, \cdots, n_l)), P_{T^t}((n_k, \cdots, n_l))$.

An annotation triple $\langle P_{T^s}(\mathbf{m}), P_{T^s}(\mathbf{n}), P_{T^t}(\mathbf{n})\rangle$ can be read intuitively as saying: a score fragment (usually just a single note) $\mathbf{n}$ was played as $P_{T^s}(\mathbf{n})$ at tempo T^s, and played as $P_{T^t}(\mathbf{n})$ at tempo T^t, while a melodically similar score fragment $\mathbf{m}$ was played as $P_{T^s}(\mathbf{m})$ at tempo T^s. In order to infer from this how to play $\mathbf{m}$ at tempo T^t, i.e. $P_{T^t}(\mathbf{m})$, two potential difficulties must be overcome. Firstly, it is possible that although $\mathbf{n}$ and $\mathbf{m}$ were similar enough to be matched, the number of notes in $\mathbf{n}$ and $\mathbf{m}$ differs (as occurs in the example in figure 5.8). Secondly, even when $\mathbf{n}$ and $\mathbf{m}$ would be identical, it still may occur that $P_{T^s}(\mathbf{n})$ and $P_{T^s}(\mathbf{m})$ are very different. For example, in one performance a note may be prolonged, and preceded by an ornamentation, whereas it is deleted in the other. This suggests that although the input problem and case are similar with respect to their score, their performances are very different.

In case the mapping is not perfect and a note of the input segment is not matched to any notes of the retrieved segment, that note has no corresponding annotation triple. Such gaps are filled up by resorting to UTS. That is, $P_{T^t}(\mathbf{m})$ is constructed from $P_{T^s}(\mathbf{m})$ by scaling the duration (in seconds) of the events in $P_{T^s}(\mathbf{m})$ by the proportion $\frac{T^s}{T^t}$, leaving all other expressive features unchanged.

5.4.3 Adaptation Rules for Establishing Analogies

To deal with these situations, we have defined a set of *adaptation rules* (exemplified in box 5), that given an annotation triple $\langle P_{T^s}(\mathbf{m}), P_{T^s}(\mathbf{n}), P_{T^t}(\mathbf{n})\rangle$, determine $P_{T^t}(\mathbf{m})$. The rules are intended to establish analogies between the tempo transformations of two matched melodic fragments, based on the *perceptual similarity* of the performances. We will illustrate this using the example adaptation rules that are shown in figure 5.8.

The lower rule infers the fragmentation event (F). This rule states that if you have an annotation triple $\langle T, C, TT\rangle$, you may infer F. The motivation for this is that from a perceptual point of view (ignoring the score), changing a performance from a consolidation event (C) to two transformation events (T) amounts to changing from one performed note to

two performed notes. To obtain this effect when the initial performance is a single performed note (T), a fragmentation event (F) is needed, so that two notes occur in the performance.

The upper rule infers OT, based on the annotation triple $\langle T, TT, OTT\rangle$. The annotation triple indicates that in the retrieved case two notes were performed as two transformation events at tempo T^s and similarly at tempo T^t, but with an ornamentation preceding the first transformation event. The net result is thus the introduction of an ornamentation in front. Since the performance of the input problem at tempo T^s is T, the inferred performance at tempo T^t is therefore OT.

The examples provided above are actually instantiations of abstract rules. The abstract rule system is defined as follows: First, a symmetric and reflexive *perceptually-similar*-relation PS on performance event sequences is defined on the alphabet of score reference events $\mathcal{A} = \{T, C, F, D\}$. We defined PS to be:

$$PS = \{(T, C), (C, T)(TT, F), (F, TT)\} \cup \{(\mathbf{X}, \mathbf{X}) \mid \mathbf{X} \in \mathcal{A}\} \tag{5.5}$$

This relation is then used to specify three abstract adaptation rules that infer an output performance from an annotation triple:

$$\langle \mathbf{X}, \mathbf{Y}, O\mathbf{Y}\rangle \rightarrow O\mathbf{X} \;\Leftrightarrow\; \{\mathbf{X}, \mathbf{Y}\} \in PS \tag{5.6}$$

$$\langle \mathbf{X}, \mathbf{Y}, I\mathbf{Y}\rangle \rightarrow I\mathbf{X} \;\Leftrightarrow\; \{\mathbf{X}, \mathbf{Y}\} \in PS \tag{5.7}$$

$$\langle \mathbf{X}, \mathbf{Y}, \mathbf{Z}\rangle \rightarrow \mathbf{V} \;\Leftrightarrow\; \{\mathbf{X}, \mathbf{Y}\} \in PS \wedge \{\mathbf{Z}, \mathbf{V}\} \in PS \tag{5.8}$$

The first rule governs the introduction of ornamentation events, henceforth called Ornamentation Introduction (OI). The second rule governs the introduction of insertion events, henceforth called Insertion Introduction (II). The last rule allows the substitution of $P_{T^s}(\mathbf{m})$ for any correspondence event that is perceptually similar to $P_{T^t}(\mathbf{n})$, whenever $P_{T^s}(\mathbf{m})$ and $P_{T^s}(\mathbf{n})$ are perceptually similar. We will call this rule Analogy Transfer (AT). Since the rules are based on a general and theoretic conception of perceptual similarity, we believe them to be general, that is, not specific to the domain of jazz music we currently deal with.

When an annotation triple matches none of the adaptation rules, this implies that the performances of the retrieved case is too different from the performance of the input problem to be reused. In this case, UTS will be applied as a default transformation.

The quality of the solution found in this way, is defined as the proportion of notes in the input segment for which a matching adaptation rule was found.

5.4.4 Transfer of Expressive Values

Until now we have focused only on the derivation of the *type* of performance event to form the performance annotation for the output performance. To construct the actual output performance from this performance annotation it is not sufficient to determine the type of the event; Concrete values must also be derived for the attributes of the events, like onset-deviation and duration deviation (in the case of correspondence events), and the dynamics of the performed notes.

The score, performances, and their annotations of the input and retrieved segment are divided into smaller parts by the score alignment between the two segments, leading to

a sequence of annotation triples as explained in subsection 5.4.2. If a given annotation triple $(P_{T^s}(\mathbf{m}), P_{T^s}(\mathbf{n}), P_{T^t}(\mathbf{n}))$ has matched an adaptation rule, this implies that $P_{T^s}(\mathbf{m})$ and $P_{T^s}(\mathbf{n})$ are perceptually similar. The process of determining $P_{T^t}(\mathbf{m})$ depends on the adaptation rule that matched. Nevertheless, all three adaptation rules are based on the transfer of expressive features from $P_{T^t}(\mathbf{n})$ to $P_{T^t}(\mathbf{m})$. We first specify how the transfer is realized for the pairs of performance event sequences that belong to the PS relation. Note that these are all instantiations of the general form specified in equation (5.4), where the performance reference parts $(\mathbf{P}_0, \cdots, \mathbf{P}_t)$ are all empty.

Let $\mathbf{m} = (m_0, \cdots, m_t)$ be the sequence of score notes of input slice $P_{T^t}(\mathbf{m}) = (p_0, \cdots, p_v)$, and let $\mathbf{n} = (n_0, \cdots, n_u)$ be the sequence of score notes of retrieved slice $P_{T^t}(\mathbf{n}) = (q_0, \cdots, q_w)$. Whenever the context implies that either $\mathbf{m}$, $\mathbf{n}$, $P_{T^t}(\mathbf{m})$ or $P_{T^t}(\mathbf{n})$ consists of a single performed note we refer to that note by m, n, p, and q (without subscripts) respectively. The transfer of respective onset, duration, and dynamics values are given for each pair of slices in PS:

(T, T) When both $P_{T^t}(\mathbf{m})$ and $P_{T^t}(\mathbf{n})$ are single transformation events, implying $\mathbf{m} = (m)$ and $\mathbf{n} = (n)$, the transfers of onset, duration, and energy are respectively defined as:

$$\mathcal{O}(p) = \mathcal{O}(m) + \mathcal{O}(q) - \mathcal{O}(n) \tag{5.9}$$

$$\mathcal{D}(p) = \mathcal{D}(m) \cdot \frac{\mathcal{D}(q)}{\mathcal{D}(n)} \tag{5.10}$$

$$\mathcal{E}(p) = \mathcal{E}(q) \tag{5.11}$$

(T, C) When $P_{T^t}(\mathbf{m})$ is a consolidation event and $P_{T^t}(\mathbf{n})$ is a transformation event, the transfers of onset, duration, and energy are defined as:

$$\mathcal{O}(p) = \mathcal{O}(m_0) + \mathcal{O}(q) - \mathcal{O}(n) \tag{5.12}$$

$$\mathcal{D}(p) = \sum_{m \in \mathbf{m}} \mathcal{D}(m) \cdot \frac{\mathcal{D}(q)}{\mathcal{D}(n)} \tag{5.13}$$

$$\mathcal{E}(p) = \mathcal{E}(q) \tag{5.14}$$

(C, T) When $P_{T^t}(\mathbf{m})$ is a transformation event and $P_{T^t}(\mathbf{n})$ is a consolidation event, the transfers of onset, duration, and energy are defined as:

$$\mathcal{O}(p) = \mathcal{O}(m) + \mathcal{O}(q) - \mathcal{O}(n_0) \tag{5.15}$$

$$\mathcal{D}(p) = \mathcal{D}(m) \cdot \frac{\mathcal{D}(q)}{\sum_{n \in \mathbf{n}} \mathcal{D}(n)} \tag{5.16}$$

$$\mathcal{E}(p) = \mathcal{E}(q) \tag{5.17}$$

(F, TT) When $P_{T^t}(\mathbf{m})$ is a pair of transformation events and $P_{T^t}(\mathbf{n})$ is a fragmentation event, the transfers of onset, duration, and energy of each played note are defined as:

$$\mathcal{O}(p_i) = \mathcal{O}(m) + \mathcal{O}(q_i) - \mathcal{O}(n_0) \qquad 0 \leq i \leq 1 \tag{5.18}$$

$$\mathcal{D}(p_i) = \sum_{m \in \mathbf{m}} \mathcal{D}(m) \cdot \frac{\mathcal{D}(q_i)}{\mathcal{D}(m_i)\mathcal{D}(n)} \qquad 0 \leq i \leq 1 \tag{5.19}$$

$$\mathcal{E}(p_i) = \mathcal{E}(q_i) \qquad 0 \leq i \leq 1 \tag{5.20}$$

(TT, F) When $P_{T^t}(\mathbf{m})$ is a fragmentation event, and $P_{T^t}(\mathbf{n})$ is a pair of transformation events, the transfers of onset, duration, and energy of each played note are defined as:

$$\mathcal{O}(p_i) = \mathcal{O}(m_i) + \mathcal{O}(q_i) - \mathcal{O}(n) \qquad 0 \leq i \leq 1 \tag{5.21}$$

$$\mathcal{D}(p_i) = \mathcal{D}(m) \cdot \frac{\mathcal{D}(q_i)}{\sum_{n \in \mathbf{n}} \mathcal{D}(n)} \qquad 0 \leq i \leq 1 \tag{5.22}$$

$$\mathcal{E}(p_i) = \mathcal{E}(q_i) \qquad 0 \leq i \leq 1 \tag{5.23}$$

(D, D) When $P_{T^t}(\mathbf{m})$ and $P_{T^t}(\mathbf{n})$ are both deletion events, no transfer of expressive values is necessary, since the score note will not be performed.

(F, F) When $P_{T^t}(\mathbf{m})$ and $P_{T^t}(\mathbf{n})$ are both fragmentation events, the transfer is:

$$\mathcal{O}(p_i) = \mathcal{O}(m) + \mathcal{O}(q_i) - \mathcal{O}(n) \qquad 0 \leq i \leq 1 \tag{5.24}$$

$$\mathcal{D}(p_i) = \mathcal{D}(m) \cdot \frac{\mathcal{D}(q_i)}{\sum_{q \in P_{T^t}(\mathbf{n})} \mathcal{D}(q)} \qquad 0 \leq i \leq 1 \tag{5.25}$$

$$\mathcal{E}(p_i) = \mathcal{E}(q_i) \qquad 0 \leq i \leq 1 \tag{5.26}$$

(C, C) When $P_{T^t}(\mathbf{m})$ and $P_{T^t}(\mathbf{n})$ are both consolidation events, the transfer is:

$$\mathcal{O}(p) = \mathcal{O}(m_0) + \mathcal{O}(q) - \mathcal{O}(n_0) \tag{5.27}$$

$$\mathcal{D}(p) = \sum_{m \in \mathbf{m}} \mathcal{D}(m) \cdot \frac{\mathcal{D}(q)}{\sum_{n \in \mathbf{n}} \mathcal{D}(n)} \tag{5.28}$$

$$\mathcal{E}(p) = \mathcal{E}(q) \tag{5.29}$$

Ornamentation Introduction

The matching of the OI rule conditions that $P_{T^t}(\mathbf{n}) = (\mathbf{P_n}, \mathbf{S_n})$ where $\mathbf{P_n} = O$, and $\mathbf{S_n} \in \{D, T, TT, C, F\}$. The consequent of the rule states that $P_{T^t}(\mathbf{m}) = (O, \mathbf{S_m})$, where $\mathbf{S_m} \in \{D, TT, C, F\}$ is determined by the input performance at the source tempo $P_{T^s}(\mathbf{m}) = (\mathbf{P_m}, \mathbf{S_m})$. Furthermore, the rule conditions that $(\mathbf{S_m}, \mathbf{S_n}) \in PS$, hence the

transfer of expressive values from $\mathbf{S_n}$ to $\mathbf{S_m}$ can be realized through the corresponding formulae described above.

The ornamentation event is introduced before the score reference event $\mathbf{S_m}$. The expressive values for the ornamentation are adapted from the ornamentation $\mathbf{P_n}$ in $P_{T^t}(\mathbf{n})$. The number of notes that form the ornamentation k is equal to the number of notes in $\mathbf{P_n}$. Since the ornamentation is not a score reference event, there is no score note to compute the onsets based on the onset *deviations* from the score note. To solve this problem we compute the onset deviations of the ornamentation notes relative to the next score note (m_0 and n_0 respectively for the input and the retrieved slices). Just as the onsets, the pitches of the ornamentation note cannot be transferred as absolute values, hence we take same approach to determine the pitches of the ornamentation notes.

$$\mathcal{O}(p_i) = \mathcal{O}(m_0) + \mathcal{O}(q_i) - \mathcal{O}(n_0) \qquad 0 \leq i \leq k \tag{5.30}$$

$$\mathcal{P}(p_i) = \mathcal{P}(m_0) - \mathcal{P}(q_i) + \mathcal{P}(n_0) \qquad 0 \leq i \leq k \tag{5.31}$$

$$\mathcal{D}(p_i) = \mathcal{D}(q_i) \qquad 0 \leq i \leq k \tag{5.32}$$

$$\mathcal{E}(p_i) = \mathcal{E}(q_i) \qquad 0 \leq i \leq k \tag{5.33}$$

Insertion Introduction

The transfer of expressive values for a matching II rule is analogous to that of OI. The difference is that an insertion only refers to a single performance note. Since the II rule conditions that only one insertion event occurs, the transfer functions (5.30), (5.31), (5.32), (5.33) for onset, pitch, duration, and energy can be used, setting $k = 0$.

Analogy Transfer

The AT rule conditions that $(P_{T^t}(\mathbf{m}), P_{T^t}(\mathbf{n})) \in PS$. The transfer of expressive values can therefore be done as in the OI and II rules, by using the appropriate transfer functions (5.9)–(5.29).

Note that, depending on the specific form of $P_{T^t}(\mathbf{n})$, multiple instantiations may be derived for $P_{T^t}(\mathbf{m})$ based on PS. For example, if $P_{T^t}(\mathbf{n}) = (T)$, both $P_{T^t}(\mathbf{m}) = (T)$ and $P_{T^t}(\mathbf{m}) = (C)$ follow from PS. The choice for one of the two instantiations is determined by the alignment between the input score and the retrieved score. In a 1-1 alignment for example, $\mathbf{m} = (m)$, so it is not consistent to derive a consolidation event in that situation, because that would require more than one input score notes.

Preservation of Monophony

The lengthening and displacement of performed notes, as well as the introduction of ornamentation and insertion events may lead to temporal overlap between consecutive notes in the performance. Such situations cannot be rendered by saxophone, as it is a monophonic instrument. Therefore, note overlap is eliminated by shortening the duration of the former of two overlapping notes by the amount of overlap.

This approach is partly justified by findings of Timmers et al. [2002], that ornamental notes tend to steal their time from the previous notes.

Concluding Remarks

We want to make some final remarks on the expressive transfer approach explained above, especially on its current limitations, and possible enhancements and elaborations.

For simplicity fragmentations and consolidations have been limited to two notes. In principle fragmentations and consolidations may involve more than two notes. It is easy to generalize the expressive transfer approach to k notes for consolidation and fragmentation events.

The formalization of the 'perceptually similar' concept by the PS relation is defined on the event level, where events are treated as nominal values. This relation obviously captures perceptual similarity only roughly. For example, it states that any two transformations events (T) are perceptually similar. In reality the perceived similarity greatly depends on the actual deviations in onset, duration, and energy. A more refined interpretation of the concept can be envisioned that also includes a 'perceptually similar' relationship for note attributes. Since the comparison of such quantitative attributes is not a crisp matter, this will add a fuzzy component to the relationship. As a consequence, PS membership will be graded. Case adaptations that were made based on low-valued PS memberships are obviously less trustworthy than those based on high-valued PS memberships. This difference can be taken into account by making adaptation quality (see subsection 5.4.3) proportional to PS membership.

Chapter 6

Experimentation

This chapter describes an experimental analysis and evaluation of the different components of TempoExpress. Section 6.1 deals with the knowledge acquisition component, and describes the results of optimizing of the performance annotation model we have proposed in chapter 3. Section 6.2 covers two experiments on melodic similarity, the most important aspect of the case retrieval step. Firstly, it provides an empirical comparison of several melodic similarity measures that may be used for case-retrieval. Secondly, the results of the MIREX 2005 contest for symbolic melodic similarity are discussed, where our edit-distance based comparison of I-R analyses is compared to several different approaches to melodic similarity computation. Finally, in section 6.3 a systematic test of the proposed problem-solving procedure is carried out in order to assess the quality of the tempo-transformations performed by TempoExpress.

6.1 Optimization of Automatic Performance Annotation

In this section we report experiments done in order maximize the accuracy of the automatic derivation of performance annotations for performances that we proposed in section 4.3. The performance annotation method can be fine-tuned by adjusting the parameters in the definition of the edit-operation cost functions.

We remark that even without optimization (that is, with arbitrary parameter settings), the resulting performance annotations are not wholly incorrect in general. This indicates that the cost model is essentially adequate, in the sense that the terms that figure in the cost functions, like the absolute differences of pitch, and duration, are terms that, when minimized, lead to an intuitive alignment between performed notes and score notes. The parameters in the cost model balance the relative importance of the terms. When the performance is close to the score, the balance is not crucial, and many parameter settings will yield the correct result (a sequence of transformation events). The parts where the balance *is* of importance, are where for example ornamentations or consolidations occur. In such situations, random parameter settings often interpret the performance wrongly, e.g. an ornamentation is interpreted as several insertion events, or consolidations are interpreted as

combinations of deletion and transformation events.

Manually tuning the parameters is possible for a small set of performances, but this becomes unfeasible for larger sets (adjustments that improve the annotation of one performance lead to incorrect annotations of other performances). Therefore, we have employed an evolutionary approach to obtain a good parameter setting. The idea of the evolutionary optimization of the parameter values is simple: an array of the parameter values can be treated as a chromosome. The number of errors produced in the annotation of a set of performances using that set of parameter values, is inversely related to the fitness of the chromosome. By evolving an initial population of (random) chromosomes through crossover, mutation and selection, we expect to find a set of parameter values that minimizes the number of annotation errors, and thus improves automatic performance annotation.

We are interested in two main questions. The first is whether it is possible to find a parameter setting that works well in general. That is, can we expect a parameter setting that worked well for a training set to perform well on unseen performances? The second question is whether there is a single setting of parameter values that optimizes the annotations. It is also conceivable that good annotations can be achieved by several different parameter settings.

6.1.1 Experiment Setup

We have run the genetic algorithm with two different (non-overlapping) training sets, both containing twenty performances from the musical corpus described in section 3.2. The performances were phrases from the songs (*Body and Soul*, and *Once I Loved*), performed at different tempos. For each of the performances, the correct annotation was available (all annotations were manually checked and where necessary corrected). The fitness of the populations was assessed using these annotations as a reference.

The fitness evaluation of a population (consisting of 20 chromosomes) on the training set is a rather time consuming operation. Therefore, it can take a long time before a good solution is obtained, starting the evolution with a randomly initialized population. In an attempt to solve this problem, we initialized the population with solutions that were trained on the individual phrases of the training set (which is a much faster procedure). Assuming that the solution optimized for one phrase may in some cases work for other phrases, this speeds up the time needed to find a good solution for the whole training set.

A new generation is generated from an old generation as follows: From the old generation (consisting of N chromosomes), the k best chromosomes are selected (where k is dependent on the distribution of the fitness across the population); Then, $N - k$ new chromosomes are created by a cross-over of the selected chromosomes; The newly generated chromosomes are mutated (multiplying each parameter value by a random value), and the $N - k$ mutated chromosomes, together with the n (unchanged) chromosomes from the old generation, form the new generation.

6.1.2 Fitness Calculation

The fitness of the chromosomes is calculated by counting the number of annotation errors using the parameter values in the chromosome. For example, assume that the correct annotation of a melodic fragment is (T, T, C, T), and the annotation of that fragment obtained

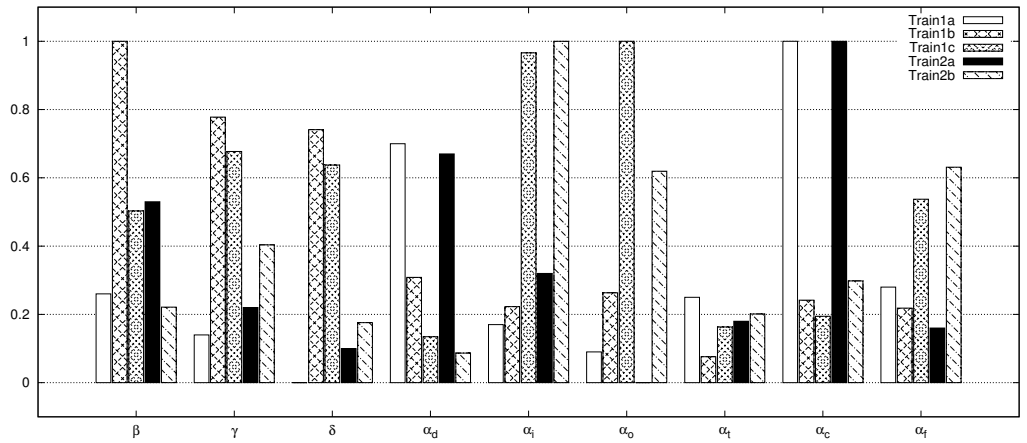

Figure 6.1: Estimated parameter values for two different training sets (Tr1 and Tr2). Three runs were done for each set (a, b, and c). The x-axis shows the nine different parameters of the cost functions (see section 4.3.1). For each parameter the values are shown for each run on both training sets

by using the parameter values of the chromosome is (T, T, T, D, T) (that is, a consolidation operation is mistaken for a transformation and a deletion operation). The C does not match to an element in the second sequence, and the T and D don't match to elements in the first sequence and thus three errors occur. To count the errors between the correct and the predicted annotations (which are represented as sequences of symbols), we use the edit-distance (don't confuse this use of the edit-distance to *compare* annotations with the use of the edit-distance to *generate* annotations).

For a given set S of performances (for which the correct annotations are known), we define the fitness of a chromosome c as:

$$\mathit{fit}(c) = \frac{1}{E(c,S)+1}$$

where $E(c, S)$ is the total number of errors in the predicted annotations for S using the parameter values in c. The fitness function *fit* ranges from zero to one. Obviously, a fitness value of one is the most desirable, since it corresponds to zero annotation errors.

6.1.3 Results

The training sets Tr1 and Tr2 both consist of 20 annotated performances, amounting to a total 488 performance events for Tr1 and 479 for Tr2. For each of the two training sets the evolution algorithm was run three times. The resulting parameter settings are shown in figure 6.1. Table 6.1 shows the number of annotation errors each of the parameter settings produced on the training sets, and on a test set of 35 performances (875 performance events), none of which occurred in Tr1 or Tr2. The first table row shows the number of errors on

	Tr1a	Tr1b	Tr1c	Tr2a	Tr2b	Tr2c
Errors on Train	19 (3.89)	9 (1.84)	10 (2.05)	11 (2.30)	12 (2.51)	11 (2.30)
Errors on Test	19 (2.17)	26 (2.97)	30 (3.43)	19 (2.17)	32 (3.66)	65 (7.43)

Table 6.1: Annotation errors produced by the obtained solutions for three different runs (denoted by the letters a, b, and c) on two different training sets (Tr1 and Tr2) and a test set.

	Tr1a	Tr1b	Tr1c	Tr2a	Tr2b	Tr2c
Tr1a						
Tr1b	-0.32					
Tr1c	-0.70	0.17				
Tr2a	**0.92**	-0.02	-0.61			
Tr2b	-0.32	-0.33	**0.76**	-0.33		
Tr2c	-0.28	**0.68**	0.07	-0.12	-0.47	

Table 6.2: Cross-correlations of the parameter values that were optimized using two different training sets (Tr1 and Tr2), and three runs for each set (a, b, and c)

the set that the solutions were trained on. The second row shows the number of errors on the test set. The values in parentheses are the errors as a percentage of the total number of performance events in the data set.

The average number of annotation errors on the test set is about 32 on a total of 875 annotation elements in the test set, an error-percentage of 3.66%. This is only slightly higher than the error-percentages on the training sets: 2,60% for Tr1, and 2,37% for Tr2 (averaged over three runs), and substantially lower than the average error-percentage of random parameter settings on the test set, which is 13.70%.

Table 6.2 shows the pair-wise correlations of the parameter values found in each of the runs. As can be seen from the cross-correlations in the table, the parameter settings did not all converge to the same values. Nevertheless, there were some cases in which the parameters were highly correlated. In particular the solutions found in runs Tr1a, and Tr2a are highly similar (this can be easily verified by eye in figure 6.1). A rather strong correlation is also observed between the solutions found in Tr1c and Tr2b, and those in Tr1b, and Tr2c. It is interesting that the correlated solutions were obtained using non-overlapping sets of performances. This is evidence that the solutions found are approximations of a single parameter setting that is valid for the performances in both training sets. In the case of the solutions of Tr1a and Tr2a, the approximated parameter setting may also have a more general validity, since both solutions have a low error number of annotations on the test set as well (see table 6.1).

6.1.4 Conclusions

The two questions we wanted to answer through this experiment were whether it is possible to find a parameter setting that has a broader validity than just the set of performances

it was optimized for, and whether there is a single parameter setting that optimizes the annotations. All solutions from different trials on two non-overlapping sets of performances substantially improved the quality of annotation of a test set over random parameter settings. In particular, whereas random parameter settings have an annotation accuracy of around 87% on average, the average accuracy of optimized parameter settings is 96% on the test set. Affirming the first question, we conclude that it is indeed possible to improve the accuracy of the automatic performance annotation mechanism, and that the tuned cost functions also show improved accuracy on unseen performances.

Moreover, some cross-correlations were found between some parameter settings that were optimized for different training sets. This suggests that they are approximations of a parameter setting that works well for a larger group of performances. In general however, the solutions did not all converge to a single set of parameter values, implying that there are different, equally valid, parameter settings.

6.2 Melodic Similarity

This section is devoted to experiments on melodic similarity. It consists of two main parts. The first part is a mostly qualitative comparison of different melody representations for computing melodic similarity. In particular, we investigate how melodic similarity based on I-R representations relates to traditional melody representations such as pitch intervals and melody contour in terms of abstraction. In the second part we report how the I-R based melodic similarity measure was tuned to predict human similarity ratings on a set of queries from a database of melodic incipits, and discuss the results of this measure in the MIREX 2005 contest for symbolic melodic similarity.

6.2.1 Looking for a Good Abstraction Level

It has been argued that adequate comparison of melodies requires abstraction from the surface of the melodies (the melody as a sequence of notes). For example, in applications such as pattern discovery in musical sequences [Cope, 1991], [Hörnel and Menzel, 1998], or style recognition [Hörnel and Menzel, 1998], it has been established that melodic comparison should take into account not only the individual notes but also structural information of the melody based on music theory and music cognition [Rolland, 1999].

Common ways to view a melody in a more abstract way are by representing its pitch contour, either as a sequence of pitch intervals, or as a sequence of registral directions (taking only the sign of the pitch intervals). Pitch intervals eliminate the absolute pitch of the melody, so that for example the interval representation of a transposed variant of a melody will be identical to the original melody. Registral direction representations not only eliminate absolute pitch, but also the differentiation between pitch intervals.

The I-R analysis of a melody can also be regarded as an abstract representation of the melody. The I-R structures that correspond to particular melodic fragments convey both relations between subsequent pitch intervals, and the patterns of implication and realization of melodic expectations formed by those intervals. In addition, rhythmic and metric structure is reflected in the chaining and combining of I-R structures. As such, it is interesting to see

how the I-R analysis as a basis for assessing melodic similarity compares to the more common representations for melody similarity.

In the present experiment we compare a similarity measure based on the I-R representation against similarity measures based on interval and registral direction contours, and based on the note representation. We use the edit-distance as a method of comparison for all of the representations.

Comparison Criteria

The focus of the experiment is on the *discriminatory power* of the measures. We use this term informally to refer to the degree to which a similarity measure is informative about the inherent structure of a set of data. Even if the structure is unknown, a similarity measure whose pairwise similarity values on a set of melodies are diverse can be said to have a higher discriminatory power than a measure whose pairwise similarity values are all identical or near-identical, in the sense that the former is more 'in focus' for the data at hand. This illustrates that the term discriminatory power is relative to the data set under inspection, and more generally, to a desired range of similarity. A geometric example may clarify this point. To express the distances between points in a plane, we commonly compute the Euclidean distance between the points. Another distance measure is obtained by taking the square of the Euclidean distance. It will be clear that, compared to the unsquared Euclidean distance, this measure is not as discriminative for points that are close together as for points that are further apart, because small distance values are compressed, and large distance values are stretched further apart by the square function. Conversely, taking the logarithm of the Euclidean distance yields a measure that is more sensitive in the range of small values, and less in the range of large values. This shows that the discriminatory power of a measure may be localized in a certain range.

We will look at the discriminatory power of the measures both globally, and within the range of nearly identical phrases. We define the global discriminatory power of a measure as the diversity of the pairwise similarity values for a given data set. This can be measured as the entropy of the similarity value distribution: Let $p(x)$, $x \in [0, 1]$ be the normalized value distribution of a distance measure D on a set of phrases S, discretized into K bins, then the entropy of D on S is:

$$H(D) = -\sum_{k=0}^{K} p(k) \ln p(k)$$

where $p(k)$, is the probability that the distance between a pair of phrases is in bin k.

To measure the discriminatory power for near-identical phrases we employ the labeling that is available in the used data set (see the next subsection) that allows us to identify which phrases are very similar to each other. We define the discriminatory power as the difference between the similarity value distribution of pairwise comparisons between near-identical phrases to that of pairwise comparisons between non-related phrases. In other words, the discriminatory power according to this criterion is the ability of the measure to systematically assign lower distance values to near-identical phrases than to phrases that are not near-identical. The distributions are compared using the Kullback-Leibler (K-L) divergence, a measure of distance between two distributions. In particular, when computing

the distance measure D over a set of phrases S, let

$$P = \{\ \{s_1, s_2\} \mid s_1, s_2 \in S \wedge s_1 \neq s_2 \wedge nearIdentical(s_1, s_2)\ \}$$

be the set of pairs of near-identical phrases from S, and let

$$Q = \{(s_1, s_2) \mid s_1, s_2 \in S \wedge s_1 \neq s_2\} - P$$

be the set of remaining pairs from S. Furthermore, let $p(x)$, $x \in [0, 1]$ and $q(x)$, $x \in [0, 1]$ be the jointly normalized discretized value distributions of D over P and Q respectively. Then the K-L divergence is defined as:

$$KLD(P\ ||Q) = \sum_{k=0}^{K} p(k) \log \frac{p(k)}{q(k)}$$

Experiment Setup

The comparison of the different similarity measures was performed using 124 different musical phrases from 40 different jazz songs from the Real Book [The Real Book, 2004](see appendix C). The musical phrases have a mean duration of eight bars. Among them are jazz ballads like 'How High the Moon' with around 20 notes, many of them with long duration, and Bebop themes like 'Donna Lee' with around 55 notes of short duration. Jazz standards typically contain some phrases that are slight variations of each other (e.g. only different beginning or ending) and some that are more distinct. This is why the structure of the song is often denoted by a sequence of labels such as A1, A2 and B, where labels with the same letters denote that those phrases are similar. We will call such variations of the same phrase *phrase variants* henceforward.

The similarity measures were implemented as edit-distances, each one using specialized cost functions for the type of melody representation. The note distance operates on sequences of notes, having the attributes pitch, duration, and onset. The cost functions of the note distance are:

$$w(\mathbf{s}_i, \emptyset) = \mathcal{D}(s_i) \tag{6.1}$$

$$w(\emptyset, \mathbf{t}_j) = \mathcal{D}(t_j) \tag{6.2}$$

$$\tag{6.3}$$

$$w(\mathbf{s}_i, \mathbf{t}_j) = \begin{pmatrix} |\mathcal{P}(s_i) - \mathcal{P}(t_j)| + \\ |\mathcal{D}(s_i) - \mathcal{D}(t_j)| + \\ |\mathcal{O}(s_i) - \mathcal{O}(t_j)| \end{pmatrix} \tag{6.4}$$

$$w(\mathbf{s}_{i-K:i}, \mathbf{t}_j) = \begin{pmatrix} \sum_{k=0}^{K} |\mathcal{P}(s_{i-k}) - \mathcal{P}(t_j)| + \\ |\mathcal{D}(t_j) - \sum_{k=0}^{K} \mathcal{D}(s_{i-k})| + \\ |\mathcal{O}(s_{i-K}) - \mathcal{O}(t_j)| \end{pmatrix} \tag{6.5}$$

$$w(\mathbf{s}_i, \mathbf{t}_{j-L:j}) = \begin{pmatrix} \sum_{l=0}^{L} |\mathcal{P}(s_i) - \mathcal{P}(t_{j-l})| + \\ |\mathcal{D}(s_i) - \sum_{l=0}^{L} \mathcal{D}(t_{j-l})| + \\ |\mathcal{O}(s_i) - \mathcal{O}(t_{j-L})| \end{pmatrix} \tag{6.6}$$

In this experiment, we wish to test not only pure interval and direction contour representations, but also hybrid contour representations that contain both interval/direction and duration information. We form this representation as a list of pairs where each pair is a pitch interval and the corresponding interonset interval (IOI). The cost functions of the interval contour distance take into account both values, where the IOI value is weighted by a factor k:

$$
\begin{aligned}
w(\mathbf{s}_i, \emptyset) &= \mathcal{I}(s_i) + k\mathcal{IOI}(s_i) && (6.7)\\
w(\emptyset, \mathbf{t}_j) &= \mathcal{I}(t_j) + k\mathcal{IOI}(s_i) && (6.8)\\
w(\mathbf{s}_i, \mathbf{t}_j) &= \left(\begin{array}{l} |\mathcal{I}(s_i) - \mathcal{I}(t_j)| + \\ k\,|\mathcal{IOI}(s_i) - \mathcal{IOI}(t_j)| \end{array} \right) && (6.9)\\
& && (6.10)
\end{aligned}
$$

Obviously, when $k = 0$ the distance measures only differences in pitch interval contours. The direction contour representation is defined analogously, replacing the pitch interval values by the sign of the interval (that is, -1, 0, and +1). The cost functions of the direction contour distance are:

$$
\begin{aligned}
w(\mathbf{s}_i, \emptyset) &= \mathcal{C}(s_i) + k\mathcal{IOI}(s_i) && (6.11)\\
w(\emptyset, \mathbf{t}_j) &= \mathcal{C}(t_j) + k\mathcal{IOI}(s_i) && (6.12)\\
w(\mathbf{s}_i, \mathbf{t}_j) &= \left(\begin{array}{l} |\mathcal{R}(s_i) - \mathcal{R}(t_j)| + \\ k\,|\mathcal{IOI}(s_i) - \mathcal{IOI}(t_j)| \end{array} \right) && (6.13)
\end{aligned}
$$

The cost functions of the I-R distance defined similarly to the definitions given in section 5.2.2, but without the weighting parameters:

$$
\begin{aligned}
w(s_i, \emptyset) &= \mathcal{S}ize(s_i) && (6.14)\\
w(\emptyset, s_j) &= \mathcal{S}ize(s_j) && (6.15)\\
w(s_i, s_j) &= \left(\begin{array}{l} \mathit{LabelDiff}\,(s_i, s_j) + \\ |\,\mathit{Size}(s_i) - \mathit{Size}(s_j)\,| + \\ |\,\mathit{Dir}(s_i) - \mathit{Dir}(s_j)\,| + \\ |\,\mathit{Overlap}(s_i) - \mathit{Overlap}(s_j)\,| \end{array} \right) && (6.16)
\end{aligned}
$$

$$
\mathit{LabelDiff}\,(s_i, s_j) = \left\{ \begin{array}{ll} 0 & \mathit{Label}(s_i) = \mathit{Label}(s_j) \\ 0.5 & \mathit{Label}(s_i) = -\mathit{Label}(s_j) \\ 1 & \mathit{otherwise} \end{array} \right.
$$

X1:	note	note	note	int.	int.	dir.
X2:	int.	dir.	I-R	dir.	I-R	I-R
$E(X2 - X1)$	-0.19	-0.31	-0.26	-0.11	-0.06	0.05
$r_{X2,X1}$	0.15	0.19	0.25	0.77	0.74	0.84

Table 6.3: The distances compared pairwise. The distances are shown in the uppermost row. In the left column, $X1$ and $X2$ respectively refer to the two measures under comparison

Results

With the 124 jazz phrases we performed all the possible pair-wise comparisons (7626) using the four different distances. The resulting values were normalized per distance. Figure 6.2 shows the distribution of values for each distance. The results for the interval and direction distances were obtained by leaving IOI information out of the cost functions (i.e. setting the k parameter to 0).

Table 6.3 summarizes some statistical relations between the distances. The top row show the expected value of the difference between each pair of distances; The distances closest together on average are the direction and I-R distances, and furthest apart are the direction and note distances. The bottom row shows the correlation coefficients. There is a strong correlation between the interval, direction, and I-R distances, and only moderate correlation between the note distance and the other three distances.

Figure 6.2 reinforces the impression from table 6.3, that there is a clear difference in similarity assessments at the note-level on the one hand, and the interval, direction and I-R-levels on the other hand. Whereas the distance distributions of the last three distances are more spread across the spectrum with several peaks, the note distance has its values concentrated around one value. This means that roughly speaking, the note distance rates the majority of the phrases from the set equally similar, whereas the other distances rate some phrases more similar than others. This indicates a difference in overall discriminative power, as we described in the previous subsection. The entropy values for each measure are shown in figure 6.3(left). It can be seen that the discriminatory power is substantially higher for the interval, direction, and I-R measures than for the note measure.

Next to the main peak in the note distance distribution a very small peak can be observed in the distance range between 0.0 and 0.2, which comprises the comparisons between phrase variants. This peak is also present in the I-R distance, in the range $0.0 - .05$. In the interval and direction distance distributions this peak is not visible. In these two distributions, the phrase-variant comparisons are 'masked' by the main peak of the distribution, that comprises the non-phrase-variant comparisons (e.g. comparisons between phrases from different songs). This means that the note and I-R distances are better at identifying which phrases are very similar than the interval and direction measures, suggesting that the note and I-R distances have a relatively good discriminatory power at the low distance range.

The K-L divergence (KLD) between the distance distribution of phrase-variant comparisons and the distance distribution of non-phrase-variant comparisons supports this. The KLD is a measure for comparing distributions. High values indicate a low overlap between distributions and vice versa. Figure 6.3(right) shows the KLD values per distance. Note

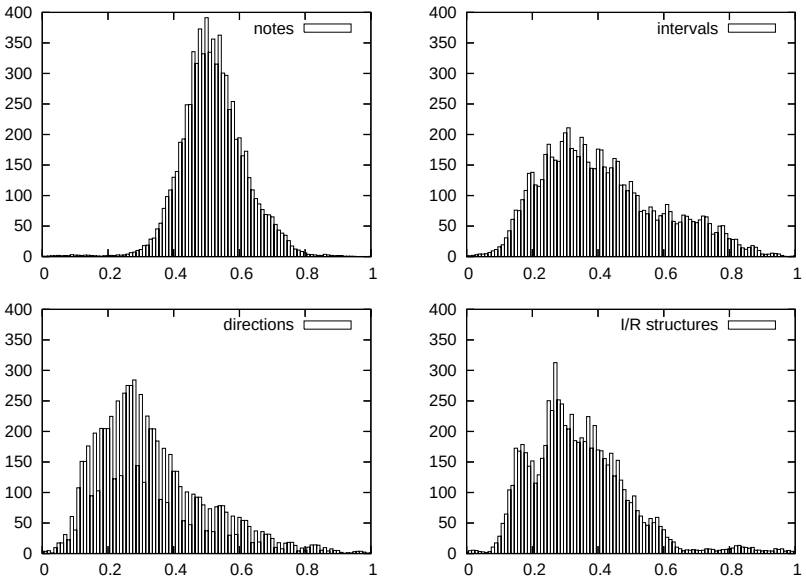

Figure 6.2: Distribution of distances for four melodic similarity measures. The x axis represents the normalized values for the distances between pairs of phrases. The y axis represents the number of pairs that have the distance shown on the x axis

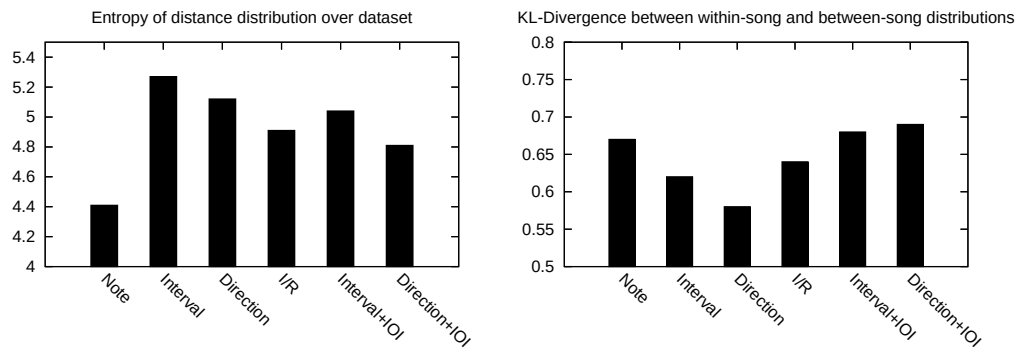

Figure 6.3: Left: Discriminatory power (measured as entropy); Right: KL-Divergence between within-song distance distribution and between-song distance distribution. The Interval+IOI and Direction+IOI measures were computed with $k = 2.0$

that the values for the interval and direction distances are slightly lower than those of the note and I-R measures.

The interval and direction measures do not include any kind of rhythmical/temporal information. Contour representations that ignore rhythmical information are sometimes regarded as too abstract, since this information may be regarded as an essential aspect of

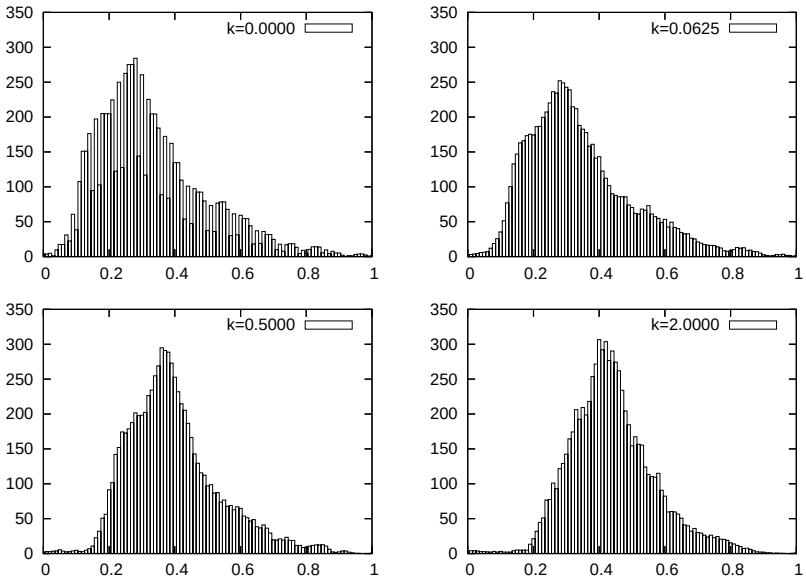

Figure 6.4: Distributions of distances of the direction measure for various weights of interonset intervals.

melody [Schlichte, 1990; Typke et al., 2003]. Therefore, we tested the effect of weighing the interonset time intervals (IOI) on the behavior of the interval and distance. Increasing the impact of IOI, through the parameter k in equations (6.7–6.13), improved the ability to separate phrase-variant comparisons from the non-phrase-variant comparisons. However, it decreased the discriminatory power of the measures (see figure 6.3). In figure 6.4, the distance distributions of the direction distance are shown for different weights of IOI. Note that, as the IOI weight increases, the form of the distribution smoothly transforms from a multi-peak form (like those of the interval, direction and I-R measures in figure 6.2), to a single-peak form (like the note-level measure in figure 6.2). That is, the direction level assessments with IOI tend to resemble the more concrete note level assessment.

Conclusions

In this experiment we have compared various edit-distance based melodic distances. We were especially interested in the distance that was based on the I-R analysis of the melody, since this is a novel way of computing melodic similarity. As a representation of melodic material, the I-R analysis provides an intermediate level of abstraction from the melodic surface, between a note representation as a less abstract representation, and the pitch contour (up/down patterns) representation as being more abstract. Although the labels given to the I-R structures merely represent pitch interval relations, the overlap and boundaries of the structures convey information about meter and rhythm, be it in an implicit way.

The measures we used for discriminatory power, the overall entropy and the K-L divergence, reflected the intermediate level abstraction of the I-R distance, in between the note

distance as the most concrete distance, and the interval and direction distances as the most abstract.

There appears to be a trade-off between discriminatory power on the short range of melodic similarity on the one hand, and discriminatory power on the long range of similarity on the other. Similarity measures based on more concrete melody representations tend to favor the former and those based on more abstract melody representations the latter. In terms of applications, concrete measures would be more suitable to find the single best match for a query (e.g. to implement Google's "I'm feeling lucky" functionality), whereas abstract measures would be more useful for multidimensional scaling of a set of melodies.

6.2.2 Ground Truth Prediction

In this subsection we report an experiment on melodic similarity ground-truth prediction, based on human similarity judgments. Part of the ground-truth data was made publicly available as training data for the symbolic melodic similarity contest, held as part of the 1st Annual Music Information Retrieval Evaluation eXchange (MIREX 2005). Using the training data we optimized our I-R distance, and submitted the algorithm for participation in the contest. Below we describe the contest setup, the optimization, and the results from the contest.

The *MIREX 2005* Contest for Symbolic Melodic Similarity

The *Music Information Retrieval Evaluation eXchange* is an annual event with the goal of comparing state-of-the-art algorithms and systems relevant for music information retrieval. Among the contest tasks of MIREX 2005 were for example audio tempo extraction, melody extraction, genre classification (for audio and symbolic data), and symbolic melodic similarity.

The symbolic melodic similarity contest aimed to evaluate melody retrieval algorithms, used to retrieve the most relevant melodies from a data base, given a query melody. The data used in the contest was taken from the RISM A/II database (a database containing about 476.600 bibliographic records of musical manuscripts written after 1600). Prior to the contest, Typke and co-workers [Typke et al., 2005b] carried out a survey in which subjects were asked to rank a set of melodic incipits (shown as in score notation). For eleven query incipits, the data base was filtered to obtain a set of around 50 candidate incipits per query, based on a rough filtering method that compared statistics like pitch range, interval histograms etc. In total 35 subjects participated; In majority they had either enjoyed musical education, or played an instrument, or both. The subjects ranked the 50 candidate incipits according to their similarity to the corresponding query, as judged by the subject. Candidate incipits that were judged entirely unrelated could be left unranked. The final ranking defined as ground-truth had the form of ranked groups of candidate incipits, where the between-group order of incipits was significant and the within-group order was not significant. The significance was based on the level of inter-subject agreement measured by a Wilcoxon rank sum test on the answers of all subjects.

The contest task consisted in ranking a subset of the database according to melodic similarity against eleven query incipits. The subset was the union of candidate incipits for all eleven queries (558 incipits in total). The ground-truth for the eleven queries of the

survey were available to participants as training data. The survey was repeated for another eleven queries that formed the test data. The melody incipits were available as MusicXML. MIDI versions derived from MusicXML were also available. Grace notes were removed in the MIDI versions, since including them without altering the durations of surrounding notes would break the time structure of the melody (the grace notes would incorrectly consume time).

The ranking computed by the participant algorithms were compared to rankings by human subjects. The rankings were evaluated using four different evaluation metrics:

average dynamic recall (*ADR*) The average of recall values after each retrieved incipit, where the set of relevant incipits is increased group-wise.

normalized recall at group boundaries (*NR*) A weighted average of the recall at group boundaries, where the recall value at each group boundary is weighted by the size of the preceding group.

average precision (*AP*) The average of the precision values at every retrieved relevant incipit.

precision at N incipits (*PN*) where N is the number of relevant incipits. The precision value after retrieving N incipits

The *ADR* [Typke et al., 2006] metric was defined to compare computed rankings to ground-truth rankings in such a way that the computed ranking is not penalized for changing the order of incipits whose order is not significant in the ground-truth ranking. That is, the order of incipits from the same group in the ground-truth ranking is not taken into account. Furthermore, the *ADR* penalizes deviations from the ground truth less towards the end of the candidate list.

Like *ADR*, the *NR* metric measures recall over an incremental list of relevant incipits. The main difference between the two metrics is that *NR* is sensitive to group size, whereas *ADR* is not.

The *AP* and *PN* metrics do not penalize the incorrect order of retrieved relevant documents, only the retrieval of any irrelevant documents before relevant documents.

The *ADR* was chosen as official evaluation metric for the contest (by a vote among the participants prior to the contest), because it best fitted the format of the ground-truth as partially ordered lists of relevant incipits.

Optimization of the I-R Measure on Training Data

The melodic similarity ground-truth for eleven queries was available as training data to optimize the participant algorithms. We used this training data to tune the I-R distance. The cost functions of the distance for insertion, deletion and replacement are identical to the ones used in TempoExpress, as given in definitions (5.1)–(5.3) on page 75.

The optimization approach we took is largely identical to the optimization of performance annotation, described in section 6.1: A genetic algorithm was applied to search the space of parameter settings. A random initial population of 30 members was evolved using an elitist approach, employing mutation and crossover. The fitness function for selection was based

parameter	operation/attribute	value
α_i	insertion	0.064
α_d	deletion	0.131
α_r	replacement	1.000
β	labels	0.587
γ	size	0.095
δ	direction	0.343
ϵ	overlap	0.112
ζ	retrospective counterparts	0.801

Table 6.4: Parameter values found by evolutionary optimization, using train data from RISM A/II database

on ground-truth training data. More concretely, a particular setting for the parameters α_d, α_r, β, γ, δ, ϵ, and ζ was evaluated by ranking the candidates for each query using the I-R distance with the given setting. The computed ranking was compared to the ground-truth ranking. The fitness was defined as the average dynamic recall value mentioned in the previous subsection, over all queries:

$$\mathit{fitness}(d) = \frac{1}{\mid Q \mid} \sum_{q \in Q} ADR(d, q) \tag{6.17}$$

where $Q = q_1, \cdots, q_{11}$ is the set of queries, and $ADR(d, q)$ is the average dynamic recall for a distance d on a query q.

A cross-validation setup on the 11 queries was chosen to prevent overfitting of the parameters to the training data. Sets of three queries were taken per run to optimize a population, and after 200 runs (which was generally enough for the fitness to stabilize) the best parameter setting from the population was selected and its ADR value was tested on a test set that included all the queries (both the three queries that were used for optimizing, and the eight remaining unseen queries). For each set of three queries (there were four sets) this procedure was executed twice, amounting to eight runs in total. The normalized parameter settings with the highest ADR value on the test set are shown in table 6.4.

Results

Seven different algorithms participated in the contest. The methods used are described in [Typke et al., 2005a], [Orio, 2005], [Suyoto and Uitdenbogerd, 2005], [Lemström et al., 2005], [Frieler and Müllensiefen, 2005], and [Grachten et al., 2005a], respectively. A rough characterization of each algorithm is given in table 6.5.

Table 6.6 shows the overall evaluation of each of the algorithms on the eleven test queries[1]. The official evaluation metric, the ADR, is shown, along with the additional three metrics mentioned in the previous subsection, and the runtimes needed to do the candidate ranking for the eleven queries. The entries with an asterisk (*) before the runtime were executed in

[1]Evaluation results per query are available here:
`http://www.music-ir.org/evaluation/mirex-results/sym-melody/`

Participant (Label)	Distance	Representation
Grachten, Arcos & Mántaras (GAM)	Edit-distance	I-R analysis
Orio (O)	N-grams matching	Pitch interval/IOI
Suyoto & Uitdenbogerd (SU)	N-grams matching	Pitch interval
Typke, Wiering & Veltkamp (TWV)	Earth Mover's Distance	Pitch/onset/duration
Lemström, Mikkilä, Mäkinen & Ukkonen (LMMU-P3)	Geometric Matching	Pitch/onset/duration
Lemström, Mikkilä, Mäkinen & Ukkonen (LMMU-DP)	Edit-distance	Pitch/pitch interval
Frieler & Müllensiefen (FM)	Hybrid	Multi-feature

Table 6.5: MIREX 2005 symbolic melodic similarity contest participants

the M2K evaluation environment[2] and thus include the time needed to evaluate the rankings. The evaluation time was approximately five seconds. Figure 6.5 shows the results in table 6.6 graphically (the left-most bar in each of the four metrics represents our I-R based distance).

The rating of algorithms is roughly similar for the two recall based metrics, *ADR* and *NR*. The same holds for the two precision based methods, *AP* and *PN*. With respect to recall, the three best scoring algorithms (GAM, O, and SU) performed substantially better than the remaining algorithms. These three algorithms also perform relatively good in terms of precision. Our I-R based distance even performs best in all four evaluation metrics. The good performance on the *ADR* and *NR* metrics indicate that it is good at ranking the relevant candidates in the right order. The high *AP* and *PN* values indicate that it is good at retrieving relevant candidates before retrieving irrelevant candidates (i.e. false positives, incipits that did not appear in the ground-truth ranking).

It may be tempting to interpret the good results of the I-R based distance as a corroboration of the I-R Model. However some reservations must be made, Firstly, one should bear in mind that the I-R analysis of a melody is hypothesized to express patterns of listening expectations (and their satisfaction/violation) that the melody generates. Evidence that perceptually similar melodies have similar I-R analyses is not necessarily evidence for this hypothesis. And secondly, the evaluation results are only partly determined by the choice of representation (in our case the I-R analysis), the actual distance metric may have a great impact as well. Nevertheless, the good performance of our algorithm indicates that the I-R analysis provides a relevant and useful representation of melody.

With respect to runtimes our I-R distance algorithm lags behind. This is hardly surprising, since our focus has not been on computational efficiency. In particular, the preprocessing step that performs the I-R analysis of the MIDI files is currently implemented as an interpreted Guile/Scheme script, which inevitably runs slower than compiled code. Furthermore, we used a C++ implementation of the edit distance that is very generic. It allows for an ar-

[2] `http://www.music-ir.org/evaluation/m2k/`

Rank	Participant	Average Dynamic Recall	Normalized Recall at group boundaries	Average Precision, non-interpolated	Precision at N incipits	Runtime (seconds)
1	Grachten, Arcos & Mántaras	65.98%	55.24%	51.72%	44.33%	∗80.17
2	Orio	64.96%	53.35%	42.96%	39.86%	24.61
3	Suyoto & Uitdenbogerd	64.18%	51.79%	40.42%	41.72%	48.13
4	Typke, Wiering & Veltkamp	57.09%	48.17%	35.64%	33.46%	51240
5	Lemström, Mikkilä, Mäkinen & Ukkonen (P3)	55.82%	46.56%	41.40%	39.18%	∗10.01
6	Lemström, Mikkilä, Mäkinen & Ukkonen (DP)	54.27%	47.26%	39.91%	36.20%	∗10.11
7	Frieler & Müllensiefen	51.81%	45.10%	33.93%	33.71%	54.59

Table 6.6: Results for the MIREX 2005 contest for symbolic melodic similarity, ranked according to Average Dynamic Recall

bitrary number of edit-operations, and supports context-aware edit-operations, thus trading speed for flexibility.

In an attempt to interpret the results better, we have characterized the participating algorithms in terms of matching method and the abstraction level of the melody representation. The matching method can either be global or local. Global matching methods force the query and the candidate to be compared entirely, whereas local matching methods perform (multiple) matches on parts of the query and candidate. Global matching can lead to an undesirably low matching result especially if the query and candidate are largely identical but have a very small portion that is very different. Also, in case the query and candidate have different lengths, the matching quality using global matching may be very low.

We define the level of abstraction of a melody representation scheme informally as proportional to the number of non-identical melodies (when represented as notes with pitch, onset, and duration) that lead to identical representations in that scheme. We classify the level of abstraction as low, medium, or high. By low abstraction we mean the representation of the melody as a sequence of primary note attributes, like absolute pitch, onset, and duration. Medium abstraction refers to for example pitch intervals, either with our without IOI information. High abstraction covers any representation that is more abstract than pitch intervals, e.g. melodic contour (direction), and other global contour representations like I-R structures.

Table 6.7 shows the characterization of each participating algorithm. With respect to matching method, note that the local matching algorithms generally performed better than the global matching algorithms. The exception to this rule is our I-R edit-distance. However, a look at the optimized parameter values of the edit-distance (table 6.4) reveals that the parameters α_i, α_d, and γ, that determine the cost of insertion and deletion are all very low. This indicates that the distance had to assign low distance values to query/candidate pairs

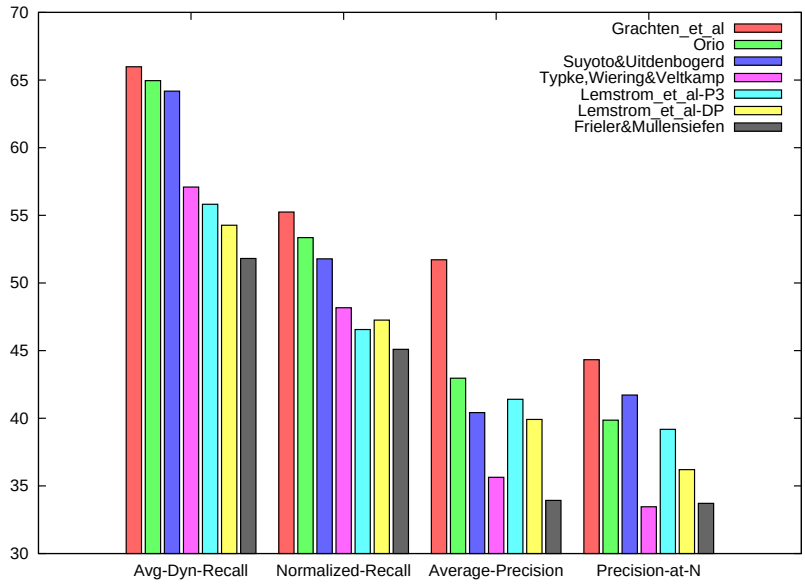

Figure 6.5: MIREX 2005 evaluation results, from table 6.6

that had rather different parts, or were different in length. Apparently, to a certain point the global matching approach can accommodate this, due to optimization of parameters. But it is obvious that the cost of insertion and deletion cannot be lowered without limits, because that would eliminate the influence of the actual melodic content on the distance computation. Local alignment is more appropriate in such cases.

With regard to the abstraction level of representation, apparently higher abstraction levels seems to work better than lower abstraction levels. This is evidence for the claim mentioned at the beginning of subsection 6.2.1, that measuring melodic similarity requires more than just note-to-note differences of melodies. It is surprising to see the relatively good results of Suyoto & Uitdenbogerd, as they apparently use only pitch interval representations, and discard duration information. This implies that either duration and other non-pitch information is irrelevant for melodic similarity (which we consider very unlikely), or that the N-grams counting method they used is very effective. Frieler and Müllensiefen's hybrid approach is hard to classify, as it involves both concrete and abstract melody representations. Moreover, the low ranking of their algorithm is probably due to the fact that they optimized their distance on melodies from another data set, rather than the training data for the MIREX 2005 contest [Frieler and Müllensiefen, 2005].

Conclusions

In the MIREX 2005 contest for symbolic melodic similarity, the I-R based melodic distance measure outperformed other state-of-the-art melody comparison methods, both in recall and precision, showing that the I-R representation of melody captures aspects of the melody that are relevant for similarity judgments.

Participant	Matching	Abstraction Level
GAM	global	high
O	local	medium
SU	local	medium
TWV	local	low
LMME-P3	local	low
LMME-DP	global	medium/low
FM	global	hybrid

Table 6.7: Characterization of participating algorithms in terms of matching approach and abstraction level of the melody representation

Analyzing the results of the contest, we conclude that the profile of a promising distance measure includes local matching, and a relatively high level of abstraction in its melody representation. Given that our I-R based measure employs a global matching edit-distance, an interesting question is whether combining I-R representations with other distance metrics can further improve the results. An obvious possibility is using a distance metric based on matching N-grams of I-R structures. Such a metric would accomplish local matching. This approach seems promising since n-grams based methods of Suyoto, and Uitdenbogerd, and Orio also give good results. Another possibility is to use a local alignment version of the edit-distance [Mongeau and Sankoff, 1990].

The computation speed of our distance measure can be improved in various ways. A major improvement can be made by implementing the I-R parser in a compiled language, rather than as an interpreted script. Also, loading and processing all MIDI files in a single run instead of invoking the program for each MIDI file individually will probably save runtime. Finally, using an edit-distance implementation that is more specialized (where edit-operations are not context-sensitive, and the set of edit-operations is hard-wired into the code) will reduce computation cost further. We believe that with this changes, our algorithm can be highly competitive in terms of runtimes as well, since the number of I-R structures in the I-R representation of a melody typically is only half the number of notes in the melody. With an average decrease of 50% of the sequence length of both query and database incipits, only 25% of the original number of comparisons are necessary, given that the complexity is of edit-distance sequence comparison is $\mathcal{O}(mn)$ in the length of the sequences m and n.

6.3 Large Scale Evaluation of Tempo Transformations

After describing tests of the annotation and retrieval components of TempoExpress in the previous sections, we report our evaluation of the TempoExpress system as a whole. The first question that needs to be addressed is how the tempo transformation results can be evaluated. Due to the subjective aspects involved, evaluation of results is not an obvious

issue (see subsection 2.4.6), as is the case with many applications that deal with musical or other artistic content such as generation, manipulation, or classification of musical content. Ground truth can not be established analytically in such domains. Instead, user surveys are often held to define the ground truth as the inter-subjective agreement between subjects. In the case of TempoExpress, the evaluation could take the form of a survey in which tempo transformed performances are presented to the subjects along with original (untransformed) performances at the target tempo. If subjects would not be able to distinguish the transformed performance from the original performance, that would prove that the expressivity in the transformed version is not perceived as unnatural.

We have however opted for an alternative evaluation approach, that measures the difference between the transformed performance and the original (target) performance. This method has some drawbacks, but also has important advantages. A drawback is that it evaluates a transformed performance using a single human performance as a reference. This may be overly restrictive, because the fact that the transformed performance does not sound like *that* (natural sounding) human performance does not mean that it does not sound natural. It is obvious that there may be different performances of the same phrase that all sound natural. Another drawback is that the results are not directly judged by human listeners, because the ultimate goal of TempoExpress as an application is that its transformation results sound natural to users in terms of expressivity.

A major advantage of the system evaluation based on difference to target performances is that it allows for a much more systematic evaluation of the results. Whereas a direct evaluation of the results by human listeners would only permit the evaluation of a small selection of results, with target-based evaluation is feasible to evaluate transformation results for every phrase and tempo for which a human performance is available. Another advantage of target-based evaluation is the possibility of experimentation. A feedback/experimentation cycle can be set up to test the effect of different settings of the system. This is not feasible with listener-based evaluation of results.

6.3.1 Evaluation Setup

As mentioned above, the quality of a tempo transformation is defined as the distance of the transformed performance of a phrase to a target performance. The target performance is a performance of the phrase played at the target tempo by a human player. In such an evaluation setup the distance measure is a crucial issue. For the distance measure to be of value, it must somehow reflect human perception of difference and similarity between performances. To achieve this, we have set up a web-survey to gather human similarity judgments of performances. An edit-distance measure was then modeled after the results of the survey in such a way that it predicts the human similarity judgments (described in subsection 6.3.3).

The distance measure compares the melodic descriptions of the performances, rather than the audio, for two reasons. Firstly, we are primarily interested in testing the *musical* quality of the tempo transformed performance (see subsections 1.2.1 and 1.2.2), whereas any kind of evaluation of the resynthesized audio would probably be strongly influenced by the *sound* quality. The melodic descriptions describe the aspects of the performance that the system has manipulated (ornamentations, timing and dynamics attributes etcetera), and therefore provide a more direct way to evaluate the changes the system has made. Secondly, the audio

resynthesis is currently done in a semi-automatic way (that is, timing and dynamics changes are translated to audio transformations automatically, but for note insertions and similar extensive changes, manual intervention is still necessary). This limitation would prevent a large-scale evaluation, if the evaluation was to be done using re-synthesized audio rather than the transformed melodic descriptions.

The musical unit TempoExpress handles as input data is the phrase. It processes the phrase by segmenting it, generating solutions for each segment and concatenating the solutions. Since no constraints have yet been defined that deal with inter-dependencies between segment-solutions, a solution to a phrase level tempo transformation problem can be regarded as a set of solutions to a set of segment level tempo transformation problems. An advantage of defining tempo transformation problems at the segment level rather than at the phrase level is that it increases both the number of possible tempo transformation problems to solve, and the amount of training data available, given the musical corpus. As a consequence, the statistical reliability of the system evaluation will be higher with segment level tempo transformations. Therefore, we defined the tempo transformation problems used to evaluate the system by segmenting the phrases in the proto case base into segments (having a typical length of about five notes), and using the performances at different tempos as problem descriptions and solutions respectively.

In this way, 6661 tempo transformation problems were defined. Each of the problems is solved and the tempo-transformed performance is compared to the target performance using the performance distance measure that was modeled after the similarity judgments gathered from the survey. For reference, the tempo transformation was also performed using uniform time stretching and the resulting performance is also compared to the target performance. This permits us to see to what extent TempoExpress improves the quality of the tempo transformations over uniform time stretching.

The distance measure for comparing expressive performances was modeled after human performance similarity judgments, in order to prevent the risk mentioned above, of measuring difference between performances that are not perceptually relevant (or conversely, failing to measure differences that *are* perceptually relevant). In subsection 6.3.2 we explain the setup of the survey by which we gathered human similarity judgments. In subsection 6.3.3 we show how the performance distance measure was derived from the survey results. Subsection 6.3.4 describes the results of tempo transformation evaluation.

6.3.2 Obtaining Ground Truth: a Web Survey on Perceived Performance Similarity

The human judgments were gathered using a web based survey[3]. Subjects were presented a target performance A (the nominal performance, without expressive deviations) of a short musical fragment, and two different performances B and C of the same score fragment. The task was to indicate which of the two alternative performances was perceived as most similar to the target performance. Thus, subjects were asked questions of the form:

$$\underline{A} \text{ is most similar to } \frac{B}{C}$$

[3]The survey is available on line at: `http://musje.iiia.csic.es/survey/introduction.html`

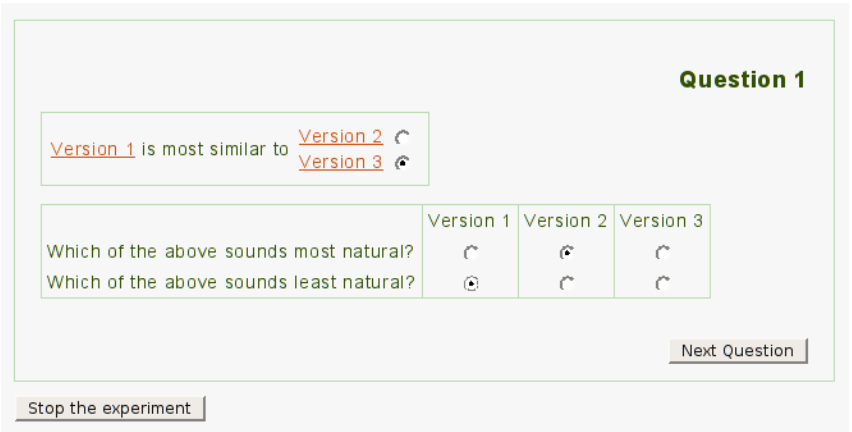

Figure 6.6: Screenshot of the web survey on performance similarity

The underlined items could be clicked to play the corresponding performance, and listeners were asked to mark their answer by selecting either B or C, through radio-buttons. Figure 6.6 shows a screenshot of the web-interface for presenting the questions to the subjects[4].

The two alternative performances were systematically varied in the expressive dimensions: fragmentation, consolidation, ornamentation, note onset, note duration, and note loudness. One category of questions tested the proportionality of the effect quantity to perceived performance distance. In this category, versions B and C contained variations in the same expressive dimension, but to a different degree. In the case of numerical parameters like duration and dynamics, this means that in version C the same notes were lengthened/shortened, loudened/softened as in version B, but to a lesser degree. In the case of discrete parameters such as ornamentation or consolidation, version C would have a smaller number of those events than version B. Another category measured the relative influence of the type of effect on the perceived performance distance. In this category version B contained deviations in a different dimension than version C (e.g. dynamics changes vs. ornamentation, or fragmentation vs. consolidation) [5].

Ten different score fragments were used for constructing the questions (i.e. triples of performances). The fragments were manually selected motifs (varying in length from six to nine notes) from eight different jazz standards (*All of Me*, *Body and Soul*, *Black Orpheus*, *Like Someone in Love*, *Once I Loved*, *How High the Moon*, *Sophisticated Lady*, and *Autumn Leaves*). More than one score fragment were used, because in initial tests, subjects reported losing their attention after answering several questions that employed the same score fragment. Therefore, care was taken to prevent the use of the same score fragment

[4]The second part of the question, regarding the naturalness of the fragments was added for a secondary aim of the survey, namely to study the possible relation between naturalness and the type and degree of expressive effects; this secondary study has not been included here, as it is not directly relevant for the current system evaluation experiment.

[5]In both types of questions, the performances of the labels B and C were interchanged occasionally, to prevent familiarization.

in two subsequent questions. The three performance variants A, B, and C were rendered into audio using a sampled saxophone, based on manually generated specifications of the expressive deviations from the score. The deviations were defined so as to comply to the question categories mentioned above.

A total of 92 subjects responded to the survey, answering on average 8.12 questions (listeners were asked to answer at least 12 questions, but were allowed to interrupt the survey). The results were filtered in several ways to discard artifacts and improve reliability. Firstly, answers to a question were discarded whenever the subject hadn't listened to each of the sound-links at least once[6]. Then, from the total set of questions (66), those questions were selected that were answered by at least ten subjects. This selection was again filtered to maintain only those questions for which there was significant agreement between the answers from different subjects (at least 70% of the answers should coincide). This yielded a set of 20 questions with answers, that is, triples of performances, together with dichotomous judgments, conveying which of the two alternative performances is closest to the target performance. The correct answer to a question was defined as the mode of all answers for that question (the filtering ensures that this is the answer on which at least 70% of the subjects were in agreement). This data formed the ground truth for modeling a performance distance measure.

6.3.3 Modeling the Ground Truth using a Performance Distance Measure

An edit-distance metric was chosen as the basis for modeling the ground truth, because the edit-distance is flexible enough to accommodate for comparison of sequences of different lengths (in case of e.g. consolidation/fragmentation) and it allows for easy customization to a particular use by adjusting parameter values of the edit-operation cost functions. In this subsection we will explain how the distance was fit to the human performance similarity judgments by optimizing parameter values.

The distance is intended to assess the similarity between different performances of the same score. Notice that this time, we are not interested in the optimal alignment, as in the performance annotation process, where a score and a performance were matched. Moreover, since one performance is not an interpretation or variation of the other (as in the case of performance vs. score), it is conceptually inappropriate to speak of the differences between the performances in terms of e.g. ornamentation, or consolidation. To avoid confusion, we will call the edit-operations in this context by their edit range, e.g. 1-0 or 1-N. To compute the distance, using equation (4.4), we defined the following cost functions for 1-0, 0-1, 1-1, N-1, and 1-N edit-operations, respectively:

[6]The number of times the subject clicked the sound-fragment links was included in the HTML form as hidden entries

$$w(\mathbf{s}_i, \emptyset) = \alpha_1 (\delta \mathcal{D}(s_i) + \epsilon \mathcal{E}(s_i)) + \beta_1 \quad (6.18)$$

$$w(\emptyset, \mathbf{t}_j) = \alpha_1 (\delta \mathcal{D}(t_j) + \epsilon \mathcal{E}(t_j)) + \beta_1 \quad (6.19)$$

$$w(\mathbf{s}_i, \mathbf{t}_j) = \alpha_2 \begin{pmatrix} \pi \, |\mathcal{P}(s_i) - \mathcal{P}(t_j)| + \\ \delta \, |\mathcal{D}(s_i) - \mathcal{D}(t_j)| + \\ o \, |\mathcal{O}(s_i) - \mathcal{O}(t_j)| + \\ \epsilon \, |\mathcal{E}(s_i) - \mathcal{E}(t_j)| \end{pmatrix} + \beta_2 \quad (6.20)$$

$$w(\mathbf{s}_{i-K:i}, \mathbf{t}_j) = \alpha_3 \begin{pmatrix} \pi \sum_{k=0}^{K} |\mathcal{P}(s_{i-k}) - \mathcal{P}(t_j)| + \\ \delta \, |\mathcal{D}(t_j) - \sum_{k=0}^{K} \mathcal{D}(s_{i-k})| + \\ o \, |\mathcal{O}(s_{i-K}) - \mathcal{O}(t_j)| + \\ \epsilon \sum_{k=0}^{K} |\mathcal{E}(s_{i-k}) - \mathcal{E}(t_j)| \end{pmatrix} + \beta_3 \quad (6.21)$$

$$w(\mathbf{s}_i, \mathbf{t}_{j-L:j}) = \alpha_3 \begin{pmatrix} \pi \sum_{l=0}^{L} |\mathcal{P}(s_i) - \mathcal{P}(t_{j-l})| + \\ \delta \, |\mathcal{D}(s_i) - \sum_{l=0}^{L} \mathcal{D}(t_{j-l})| + \\ o \, |\mathcal{O}(s_i) - \mathcal{O}(t_{j-L})| + \\ \epsilon \sum_{l=0}^{L} |\mathcal{E}(s_i) - \mathcal{E}(t_{j-l})| \end{pmatrix} + \beta_3 \quad (6.22)$$

Where $\mathbf{s}_i = \langle s_i \rangle$, and $\mathcal{P}(s_i)$, $\mathcal{D}(s_i)$, $\mathcal{O}(s_i)$, and $\mathcal{E}(s_i)$ respectively represent the pitch, duration, onset, and dynamics attributes of a note s_i. Each attribute has a corresponding parameter (π, δ, o, and ϵ, respectively), that controls the impact of that attribute on operation costs. The β parameters control the absolute cost of the operations. The α parameters control the partial cost of the operation due to (differences in) attribute values of the notes. Note that the same α and β parameters occur in the 1-0 and 0-1 cost functions, and also in the 1-N and N-1 cost functions. This ensures that the distance will be symmetric.

Fitting the edit-distance to the ground truth is a typical optimization problem, and an evolutionary optimization was used as a local search method to find good values for the ten parameters in equations (6.18) – (6.22).

The fitness function for evaluating parameter settings was defined to be the proportion of questions for which the correct answer was predicted by the edit-distance, using the parameter settings in question. A correct answer is predicted when the computed distance between the target performance and the most similar of the two alternative performances (according to the ground truth) is lower than the computed distance between the target and the remaining alternative performance. More precisely, let $Q = \{q_1, \cdots, q_n\}$ be the questions for which the ground truth is known, where q_i is a triple $\langle t_i, s_i, r_i \rangle$ containing the target performance t_i of question q_i, the alternative performance s_i that was selected by the subjects as being most similar to t_i, and the remaining (less similar) alternative performance r_i for that question. The fitness of a distance d is then defined as:

$$\mathit{fitness}(d) = \frac{|\{ \, q_i \in Q \mid d(t_i, s_i) < d(t_i, r_i) \, \}|}{n} \quad (6.23)$$

α_1	α_2	α_3	β_1	β_2	β_3	π	δ	o	ϵ
0.031	0.875	0.243	0.040	0.330	0.380	0.452	1.000	0.120	0.545

Table 6.8: Optimized values of edit-operation cost parameters

Using this fitness function a random population of parameter settings was evolved using an elitist method for selection. That is, the fittest portion of the population survives into the next population unaltered and is also used to breed the remaining part of the next population by crossover and mutation [Goldberg, 1989]. A fixed population size of 40 members was used. Several runs were performed and the fitness tended to stabilize after 300 to 400 generations. Typically the percentages of correctly predicted questions by the best parameter setting found were between 70% and 85%. The best parameter setting found (shown in table 6.8) was employed in the edit-distance that was subsequently used to evaluate the tempo transformed performances generated by TempoExpress.

6.3.4 Comparison of TempoExpress and Uniform Time Stretching

In this subsection we report the evaluation results of the TempoExpress system on the task of tempo transformation, and compare them to the results of uniformly time stretching the performance. As said before, the evaluation criterion for the tempo transformations was the computed distance of the transformed performance to an original performance at the target tempo, using the edit-distance optimized to mimic human similarity judgments on performances.

A leave-one-out setup was used to evaluate the CBR system where, in turn, each proto case is removed from the case base, and all tempo transformations that can be derived from that proto case are performed using the reduced case base. The constraint that restricted the generation of tempo transformation problems from the proto cases was that there must be an original human performance available at the source tempo (the performance to be transformed) and another performance of the same fragment at the target tempo of the tempo transformation (this performance serves as the target performance to evaluate the transformation result). Hence the set of tempo transformation problems for a given proto case is the pairwise combination of all tempos for which a human performance was available. Note that the pairs are ordered, since a transformation from say 100 BPM to 120 BPM is not the same problem as the transformation from 120 BPM to 100 BPM. Furthermore the tempo transformations were performed on a phrase segment basis, rather than on complete phrases, since focusing on phrase level transformations is likely to involve more complex higher level aspects of performance (e.g. interactions between the performances of repeated motifs), that have not yet been addressed in this research. Moreover, measuring the performance of the system on segments will give a finer grained evaluation than measuring on the phrase level.

Defining the set of tempo transformations for segments yields a considerable amount of data. Each of the 14 phrases in the case base consists of 3 to 6 motif-like segments, identified using Temperley's Melisma Grouper [Temperley, 2001], and has approximately 11 performances at different tempos (see subsection 3.2). In total there are 64 segments, and 6364 transformation problems were generated using all pairwise combinations of performances for each segment. For each transformation problem, the performance at the source tempo was

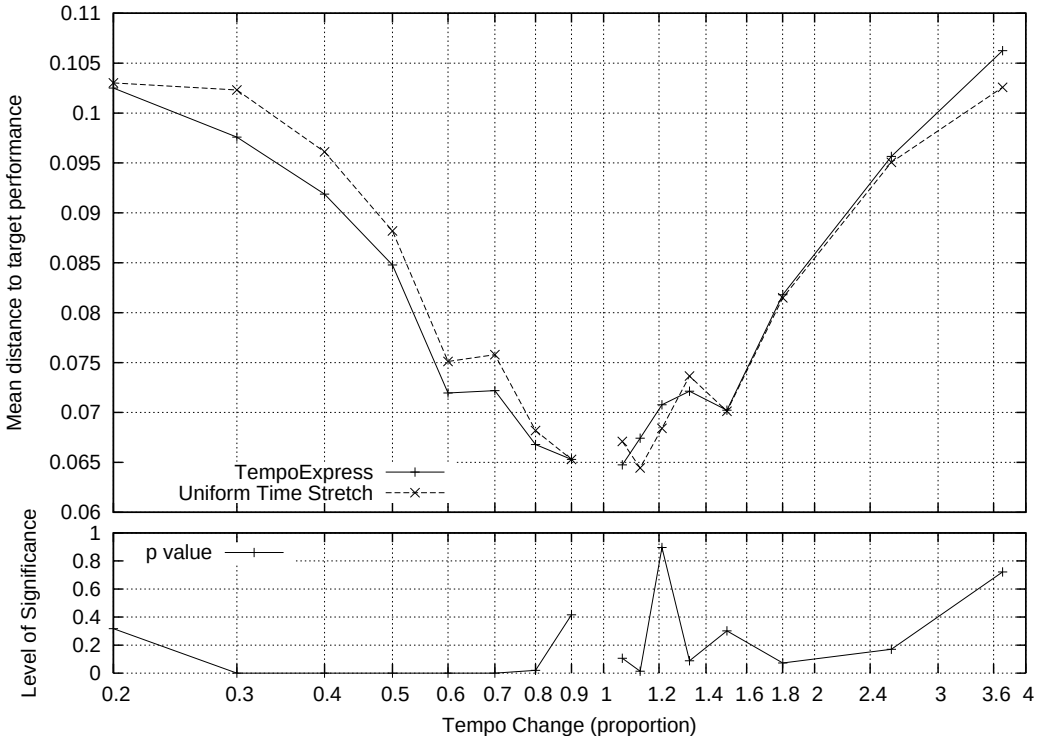

Figure 6.7: Performance of TempoExpress vs. uniform time stretching as a function of tempo change (measured as the ratio between target tempo and source tempo). The lower plot shows the probability of incorrectly rejecting H_0 (non-directional) for the Wilcoxon signed-rank tests

transformed to a performance at the target tempo by TempoExpress, as well as by uniform time stretching (UTS). Both of the resulting performances were compared to the human performance at the target tempo by computing the edit-distances. This resulted in a pair of distance values for every problem. Figure 6.7 shows the average distance to the target performance for both TempoExpress and UTS, as a function of the amount of tempo change (measured as the ratio between target tempo and source tempo). Note that lower distance values imply better results. The lower graph in the figure shows the probability of incorrectly rejecting the null hypothesis (H_0) that the mean of TempoExpress distance values is equal to the mean of UTS distance values, for particular amounts of tempo change. The significance was calculated using a non-directional Wilcoxon signed-rank test.

Firstly, observe that the plot in Figure 6.7 shows an increasing distance to the target performance with increasing tempo change (both for slowing down and for speeding up), for both types of transformations. This is evidence against the hypothesis of relational invariance, which implies that the UTS curve should be horizontal, since under relational variance, tempo transformations are supposed to be achieved through mere uniform time

	mean distance to target		Wilcoxon signed-rank test		
	TempoExpress	UTS	p <>	z	*df*
tempo ↑	0.0791	0.0785	0.046	1.992	3181
tempo ↓	0.0760	0.0786	0.000	9.628	3181

Table 6.9: Overall comparison between TempoExpress and uniform time stretching, for upwards and downwards tempo transformations, respectively

stretching.

Secondly, a remarkable effect can be observed in the behavior of TempoExpress with respect to UTS, which is that TempoExpress improves the result of tempo transformation specially when slowing performances down. When speeding up, the distance to the target performance stays around the same level as with UTS. In the case of slowing down, the improvement with respect to UTS is mostly significant, as can be observed from the lower part of the plot.

Finally, note that the p-values are rather high for tempo change ratios close to 1, meaning that for those tempo changes, the difference between TempoExpress and UTS is not significant. This is in accordance with the common sense that slight tempo changes do not require many changes, in other words, relational invariance approximately holds when the amount of tempo change is very small.

Another way of visualizing the system performance is by looking at the results as a function of absolute tempo change (that is, the difference between source and target tempo in beats per minute), as shown in figure 6.8. The overall forms of the absolute curves and the relative curves (figure 6.7) are quite similar. Both show that the improvements of TempoExpress are mainly manifest on tempo decrease problems.

Table 6.9 summarizes the results for both tempo increase and decrease. Columns 2 and 3 show the average distance to the target performance for TempoExpress and UTS, averaged over all tempo increase problems, and tempo decrease problems respectively. The remaining columns show data from the Wilcoxon signed-rank test. The p-values are the probability of incorrectly rejecting H_0 (that there is no difference between the TempoExpress and UTS results). This table also shows that for downward tempo transformations, the improvement of TempoExpress over UTS is small, but extremely significant ($p < .001$), whereas for upward tempo transformations UTS seems to be better, but the results are slightly less decisive ($p < .05$).

How can the different results for tempo increase and tempo decrease be explained? A practical reason can be found in the characteristics of the case base. Since the range of tempos at which the performances were played varies per song, it can occur that only one song is represented in some tempo range. For example, for *Up Jumped Spring* the tempos range from 90 BPM to 270 BPM, whereas the highest tempo at which performances of other songs are available is 220 BPM. That means that in the leave-one-out method, there are no precedents for tempo transformations to tempos in the range from 220 BPM to 270 BPM. This may explain the increasing gap in performance in favor of UTS, towards the end of the spectrum of upward tempo transformations.

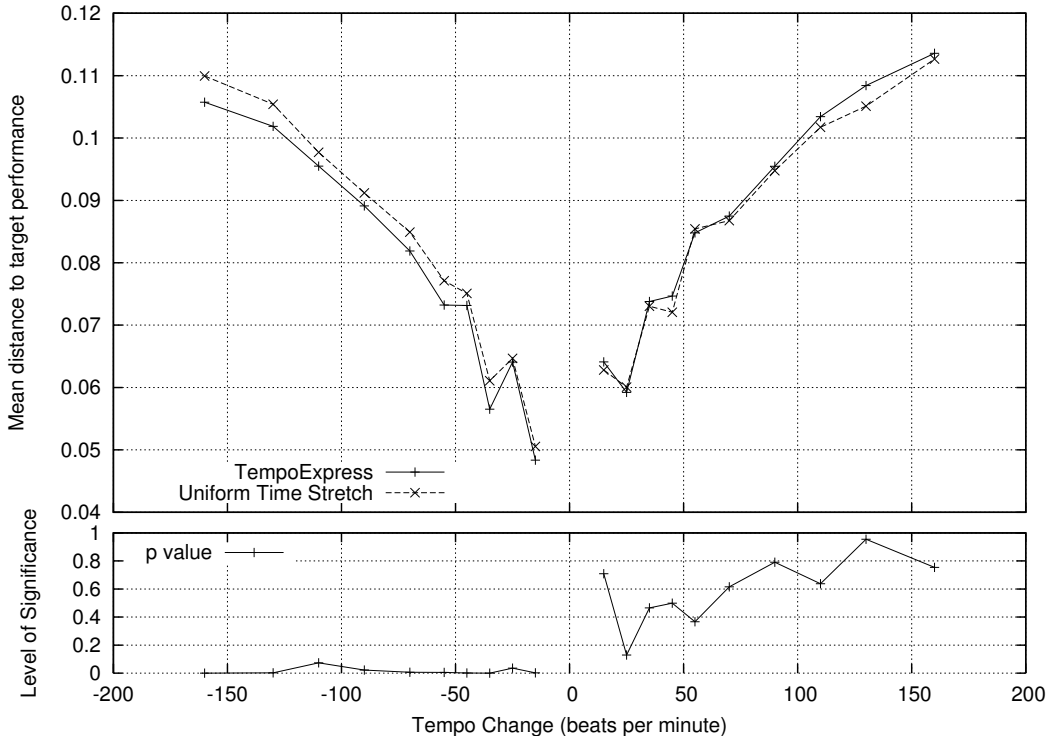

Figure 6.8: Performance of TempoExpress vs. UTS as a function of tempo change (measured in beats per minute). The lower plot shows the probability of incorrectly rejecting H_0 (non-directional) for the Wilcoxon signed-rank tests

6.3.5 Conclusions

With the experiment described in this section we have intended to give an extensive evaluation of the TempoExpress system. We have done so by testing the system on all tempo-transformation problem for which a target performance was available. The transformed performances are evaluated by measuring the distance to the target performance. This approach depends on a distance measure for (melodic descriptions of) performances that was modeled after human similarity judgments between performances. Note that rather than testing directly the *naturalness* of transformed performances in a survey, we have performed a survey on the *similarity* between excerpts. By using the survey to model human similarity judgments with a computational distance measure the gathered information has a broader applicability, because the model may be used to evaluate a much larger set of performance than is feasible through a survey. Moreover, the model may be used not only for judging naturalness of performances by computing the distance to naturalness ground truth (performances that are known to be natural), but for evaluating any descriptive label for performances (provided that ground truth performances are available for that label). This approach rests on the intuitively plausible assumption that if a performance P_1 is known to

have a perceptual quality Q and another performance P_2 is perceptually similar to P_1, then P_2 will also have perceptual quality Q.

The results of the evaluation showed that overall, there is a significant improvement of tempo transformation when realized by TempoExpress, over tempo transformations realized by uniform time stretching. Remarkably, the improvements are most significant for tempo transformations where the tempo of the original performance is decreased. For increasing tempo transformation, no significant improvements were observed.

Chapter 7

Conclusions

In this final chapter we first summarize the topic covered in this dissertation (section 7.1). Secondly, we list the contributions of our work to the field of melody retrieval, expressive music performance rendering and its evaluation, and content based audio processing (section 7.2). Finally, we elaborate on some future directions of work (section 7.3), which we believe will improve the quality and utility of our current work, as well as they will cover areas that have not yet been explored extensively.

7.1 Summary

In this dissertation we have explored a case based reasoning approach to deal with the problem of respecting expressivity in monophonic audio recordings under tempo-transformation. The problem that arises when changing the tempo of an audio recording is that the expressivity of timing does not scale uniformly with tempo [Honing, 2007]. Other expressive aspects like dynamics, and also more extensive effects like note ornamentations, do not remain constant under tempo change either. This causes the tempo transformed performance to sound unnatural in terms of expressive quality, when realized through uniform time stretching, the standard technique for tempo-transformation that is found in most state-of-the-art audio editing applications. Some existing techniques obtain high sound quality tempo-transformations employing knowledge of low-level audio-content, for example by applying the stretching only to stationary parts of the audio, and leaving the transients intact [Bonada, 2000]. However they do not address the musical, or expressive quality of the recording. We have proposed and implemented a case based reasoning method that employs knowledge of higher level audio content, like a description of the melody, to derive a suitable transformation of expressive features. It uses implicit knowledge in the form of human performed phrases. In the problem description and solving processes it employs general musical knowledge to segment phrases, to annotate performances, and to transfer expressive information from one phrase to another. A large scale evaluation of the system has shown that the quality of tempo transformations by the proposed system improves compared to uniform time stretching.

7.2 Contributions of this Dissertation

7.2.1 An I-R Parser for Melody

The Implication-Realization model provides a formalization of the cognitive processes of creation and fulfillment/violation of expectation in melody perception. It employs a number of principles based on Gestalt-laws that predict to what degree particular continuations of a melody are expected by human listeners. Computational models exist computing the degree to which each principle is fulfilled [Toiviainen and Eerola, 2002; Eerola and North, 2000]. However, the next step described in the model, the categorization and grouping of notes based on such expectancies, to our knowledge, has not been formalized into a computational model. We present such a model, in the form of an I-R parser that analyses symbolic monophonic melodies and returns the I-R analysis as a sequence of I-R structures that describe the melodic surface. The parsing steps are described in section 4.1.

7.2.2 An I-R based Distance Measure for Melody

The I-R analysis of a melody can be regarded as semi-abstract representation of the melodic surface, that reflects perception related concepts such as grouping and expectancy, as well as a rough melodic and rhythmic contour. Based on the cognitive theory of melodic similarity by Deliège [1997], and subsequent experiments of perceived melodic similarity [Lamont and Dibben, 2001], it is plausible that such a representation has a good abstraction level for assessing similarity between melodies. Hence we defined a set of weight-functions that enable comparison of sequences of I-R structures using the edit-distance (see subsection 5.2.2). In comparison to other state-of-the-art methods for melodic similarity assessment (the MIREX 2005 symbolic melodic similarity contest, see subsection 6.2.2), this melodic similarity measure was shown to be superior in its ability to predict human similarity judgments.

7.2.3 Comparison of Melodic Distance Measures

In addition to the comparison of distance measures with respect to melodic similarity ground truth as mentioned above, we have done a more qualitative comparison of four edit-distance based melodic similarity measures (subsection 6.2.1). Each of the measures compared melodies using a different representation. The melodies were represented respectively as sequences of notes, melodic intervals, melodic directions, and I-R structures. With each of the measures, the pairwise distances were calculated for a set of about 125 phrases from about 35 songs from the Real Book of Jazz. The main conclusions were that the note representation had a relatively low discriminative power (defined as the entropy of the distance distribution of the measure) for the whole data set, but was good at recognizing phrases of the same song that were close variants of each other. The interval, direction and I-R representations had similar discriminative power, but the I-R representations gave better results for the recognition of phrases from the same song. Since the interval and direction representations only contain information derived from pitch and do not capture any rhythmic information (which can be argued to be too abstract), we adapted the representations to include note durations. This decreased the discriminatory power, but increased the recognition of phrases from the same song. Using these two criteria, the interval representation

with durational information had slightly better results than the I-R representation.

These results are useful for the retrieval part of our CBR system, where melodic similarity is assessed as a criterion for retrieval. But the results may also be relevant for music retrieval in general, as for example in *query-by-humming* systems, where melodic similarity is frequently used for retrieval.

7.2.4 A Scheme for Performance Annotation

Due to the complexity of musical expressivity, computational models often are limited to only a few dimensions of expressivity, typically timing and dynamics curves. Such curves are constituted of the deviations of performed notes from their corresponding score complement. As such, they discard any relation between score and performance notes that is not a 1-to-1 correspondence, although such relations are clearly significant manifestations of expressivity [Arcos et al., 2003] (see sections 1.1 and 3.6). Moreover, interpolated timing and dynamics curves falsely give the impression of a sampled continuous timing/dynamics 'signal', whereas the performance is more properly regarded as a stream of events, where the timing and dynamics properties of these events depend on the musical context of the events Desain and Honing [1993].

To address these issues, we have presented an annotation scheme to represent expressive information from performances (see section 3.6). The annotation consists of a sequence of *performance events*, that describe how the performance relates to the score. These events capture in some sense the musical behavior of the musician while performing a melody. For example, if the musician ornamented a certain notes while playing, this is represented by an *Ornamentation Event*. Alternatively, changing the timing, dynamics, duration or other attributes of notes, is reflected by *Transformation Events*. The events are not just labels, but complex structures that have references to elements from the score and the performance, and can hold additional information such as the quantity of timing or duration deviations (in the case of *Transformation Events*). Other performance events are *Consolidation Events*, *Fragmentation Events*, *Insertion Events*, and *Deletion Events*.

This annotation scheme provides a more extensive description of expressivity in performances compared to timing/dynamics curves. Apart from timing/dynamics deviations, it accounts for various ubiquitous non-1-to-1 relations between score and performance notes as expressive gestures.

7.2.5 A Technique for Automated Performance Annotation

The performance annotations of the form described above can be constructed automatically given a score and a performance of that score. By mapping each of the performance event types that may occur in the performance annotation to an edit-operation, the edit-distance can be used to find the optimal sequence of edit-operations that maps the performance to the score (see subsection 4.3). The corresponding sequence of performance events forms the performance annotation.

Although many different possible performance annotations may account for the same performance, usually only one of them is correct, i.e. perceptually plausible. Therefore, the cost functions of the edit-operations must be well-designed and balanced in order for the edit-distance to derive the correct annotation. The design of the cost functions is an extension

of the cost functions proposed by Mongeau and Sankoff [1990] (see subsection 4.3.1). The standard definition of the edit-distance was extended to allow for context-sensitive cost functions (see subsection 4.2.1), to allow for the proper calculation of note ornamentations.

The terms that figure in the cost functions (like pitch/timing/duration of score and performance notes) are weighted by parameters, just as the relative cost of edit-operations with respect to each other. We observed that untuned cost functions lead to an average accuracy of about 76% on a test set of 40 performances. Manual parameter tuning can improve the accuracy when annotating individual performances, but is unfeasible for larger sets of performances. Using a cross-validation setup, a genetic algorithm was used to optimize performance annotations (using hand-corrected performance annotations as training data). Optimizing parameter settings in this way lead to improved accuracies on the test sets ranging from 92% to 97%. Although the optimal parameter settings found in different runs on different data sets did not all converge, there were some strongly correlated settings, amounting to the belief that the number of alternative valid parameter settings is small. Details of this experiment can be found in section 6.1.

In addition to its relevance to expressive performance research, this can be seen as a contribution to CBR research, as an example of automated case base acquisition. In knowledge-intensive domains, it is often difficult and cumbersome to gather and analyze data to form cases. In the domain of music processing, this method automates an important part of this task.

7.2.6 A System Architecture for Case Based Tempo-Transformation

We have developed and implemented the system architecture for expressivity aware musical tempo transformations based on case based reasoning, as described in chapter 3. The system loads a MIDI file containing a monophonic phrase, and an XML file containing a melodic description of some performance of that phrase (the tempo of that performance should be specified – it is not inferred from the performance). Furthermore a desired output specified. The input problem specification is constructed from those data automatically: an I-R analysis of the MIDI score is made, and the performance is annotated. Then, the CBR system is consulted to solve the problem. The result is a new performance annotation, which is used to convert the old melodic description into to a new one. The XML file containing the new melodic description can be used as the input to a sound synthesis module (such as the saxophone synthesizer *Salto* [Haas, 2001]).

7.2.7 Analogy Based Transfer of Expressivity across Performances

A problem we faced designing the reuse process of the CBR system is that often the most similar case in the case is a melodic fragment that is not very similar to the melodic fragment of the input problem. Moreover, even if the melodic fragments are similar, it is possible that the performance of the fragments differs substantially. In such situations it is not clear how the retrieved case can be reused. To overcome this problem we have designed a process that infers a tempo transformation for the input fragment based on an analogy with the tempo transformation of the retrieved fragment. The analogy is established by adaptation rules that compare performances of the different fragments. The crucial point here is that

the adaptation rules are defined using the concept of *perceptual similarity*. That is, they ignore the melodic scores and their interpretations, but only compare how similar the results sound. In this way, we take advance of situations where input problem and retrieved case are different both in melody and interpretation but where these differences cancel each other out. The result is a reuse method that is relatively robust to imperfect retrieval. This characteristic becomes more crucial as the input problem to the system is not similar to the material in the case base in terms of melody.

From the case based reasoning perspective, this contribution is an example of case reuse in a knowledge-intensive domain. In particular, it performs a synthetic configuration task, that deals with multi-layered sequential data. In contrast, most current CBR applications dealing with sequential data perform analytical tasks such as classification or regression.

7.2.8 An Evaluation Methodology for Expressive Performance Models

Finally, we have employed a novel evaluation methodology to test the tempo transformation system. This is a hybrid methodology, as it employs a computational distance measure to assess predicted results using target performances, but it also uses human similarity judgments on performances. By modeling the assessment distance measure after human judgments of similarity, we obtain a distance measure that can be used as a model of a human listener to evaluate how similar a predicted performance is to a target performance. In this way, we avoid the drawback of computational distance measures that they may fail to correspond to a perceptual notion of similarity. At the same time we avoid the drawback of evaluating model predictions directly on human judgments that large scale evaluations become infeasible.

This evaluation methodology is not limited to the evaluation of tempo transformations. It can be used in any evaluation context where target performances are available that are known to have desired expressive characteristics. This includes for instance expressive performance generation from the score. As an example of this, we have employed the distance measure fitted to human similarity judgments as a fitness function to evaluate expressive models that were constructed using genetic programming [Hazan et al., 2006a].

7.3 Future Directions

7.3.1 Finer Grained Performance-Annotations

A vital step for improving musical quality of the transformed performances further is to extend the performance annotations to include means of expressivity at the sub-note level, vibrato, timbre, attack-time, and articulation. Maestre and Gómez [2005] proposed an initial design of an algorithm and model for extracting and describing articulation and dynamics at the sub-note level. The model consists of a parametric description of the dynamics curve, where parameters characterize the principal stages of the note, like attack, sustain, and release. The degree of legato and staccato is modeled by another parameter. A first step in the direction of performance rendering using such features has been made by Ramirez et al. [2005].

To employ such extended performance annotations in TempoExpress, a more elaborate retrieval and reuse mechanism may be required. Based on the unconfirmed hypothesis that finer details of the performance are dependent on a smaller musical context, the results are likely to benefit from a further hierarchical decomposition of cases. Just as phrases were decomposed into segments in the current problem solving mechanism, deriving intra-note features could require a decomposition of the cases into note triples, duples, or even single notes.

7.3.2 Case Base Quality Evaluation

We have performed initial work for the evaluation of case base quality [Grachten et al., 2005b] (not covered in this dissertation). This work aims to visualize the problem solving capabilities (*competence*) of the case base so that different kinds of regions in the problem space can be identified. For example, it is possible that certain parts of the problem space may be covered with high competence by just a few cases, whereas other regions may need a dense population of diverse cases in order to cover the region sufficiently. It is also helpful for the maintenance of the case base over time. For example it allows one to study the effect of including individual cases on the competence of the case base. In a later stage of development, low competence regions may be analyzed automatically in order to guide case addition. For example, cases with specific characteristics may be proposed for addition to populate low density regions of the problem space, or regions where all available cases are of low quality.

7.3.3 User-Interaction

In conjunction with introspective competence-awareness of TempoExpress, a user interface is required to allow interaction with the user for revision and rentention of proposed solutions. In this dissertation we have focused on data preparation, and storage, retrieval, and reuse of cases. A prototype GUI was designed to control these subtasks of the system, as shown in figure 7.1. Future work on the interface includes non-linear control of the retrieve and reuse steps, where the user can investigate alternative precedent cases to solve partial problems and hear the effect of the reuse of different precedents on the final performance. This allows for pervasive learning. In the first place, allowing the user to correct solutions will lead to higher quality cases, that can be stored to improve future problem solving capabilities. Moreover, the user interference with the choice of precedents to be reused provides feedback on case retrieval. By adjusting the similarity measure to mimic the users preferences for certain precedents in certain problem contexts, the quality of the retrieval step can be improved over time.

7.3.4 Hybrid Problem Solving

An approach that is bound to improve the capabilities of TempoExpress is the combined application various problem solving techniques. In particular, a general model for performance prediction induced from the available data can be applied in situations where CBR provides no satisfying outcome.

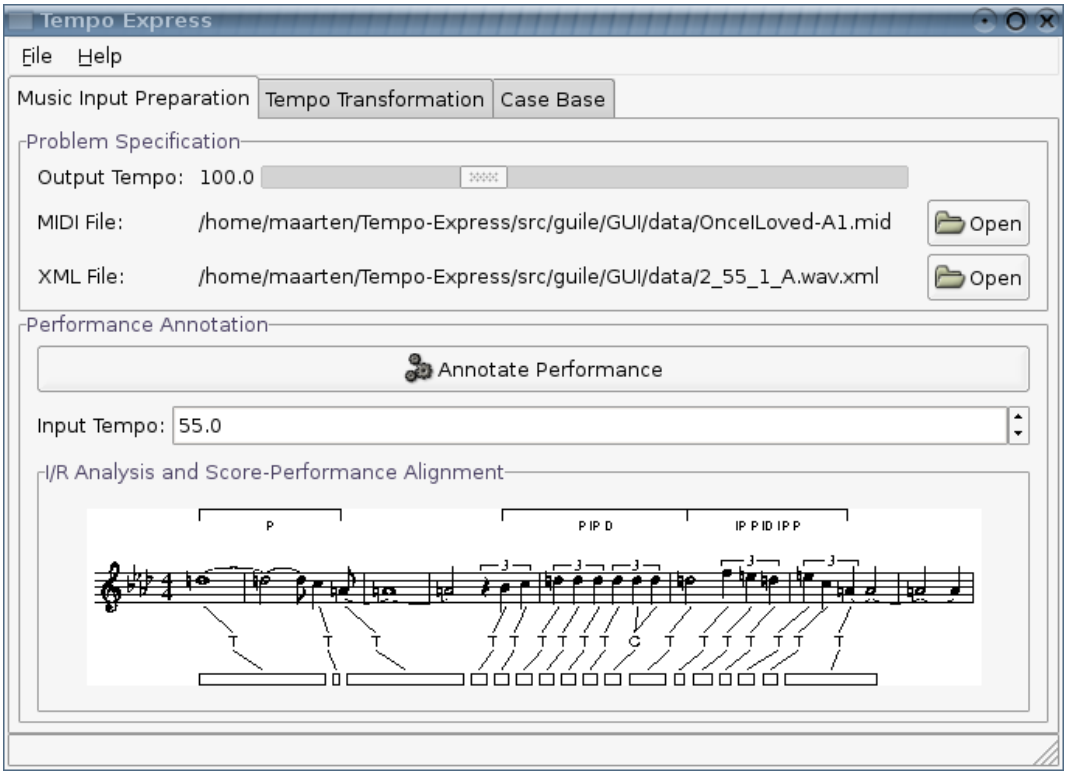

Figure 7.1: Screenshot of the TempoExpress GUI prototype

There are several ways to realize such hybridization in TempoExpress. The most straightforward and least intrusive way would be to learn a set of performance rules or decision trees such as those described by Ramirez and Hazan [2004], and Hazan et al. [2006a], and apply those rules whenever an unmatched note is encountered, or when no adaptation rule applies during the reuse of a retrieved segment (see subsection 5.4.3).

A more systematic way would be to experimentally establish the optimal division of labors between different problem solving approaches, that is, the one that maximizes the quality of the transformed performances. Alternatively, one could give priority to one of the problem solving strategies and tune the other strategy to fill the gaps of the primary strategy. For example, by determining the regions of the problem space where the competence of TempoExpress's CBR system is lowest (as described above), one could assemble a specific data set for an eager inductive learner that tries to derive a predictive model just for those problem space regions. Such a model is likely to have a higher accuracy on the target regions than a model that must cover the complete problem space.

Bibliography

Aamodt, A. (2004). Knowledge-intensive case-based reasoning in Creek. In Funk, P. and Calero, P. A. G., editors, *Advances in case-based reasoning, ECCBR 2004, Proceedings*, number 3155 in LNAI, Madrid, Spain. Springer.

Aamodt, A. and Plaza, E. (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches. *Artificial Intelligence Communications*, 7(1):39–59. Online: *http://www.iiia.csic.es/People/enric/AICom_ToC.html*.

Aigrain, P. (1999). Applications of content processing of music. *Journal of New Music Research*, 28(4):271–280.

Arcos, J. L., Grachten, M., and López de Mántaras, R. (2003). Extracting performer's behaviors to annotate cases in a CBR system for musical tempo transformations. In *Proceedings of the Fifth International Conference on Case-Based Reasoning (ICCBR-03)*, number 2689 in Lecture Notes in Artificial Intelligence, pages 20–34. Springer-Verlag.

Arcos, J. L., López de Mántaras, 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.

Ashley, K. D. (1990). *Modeling Legal Arguments: Reasoning with Cases and Hypotheticals*. MIT Press, Bradford Books, Cambridge, MA.

Ashley, R. (2002). Do[n't] change a hair for me: The art of jazz rubato. *Music Perception*, 19(3):311–332.

Bengtsson, I. and Gabrielsson, A. (1980). Methods for analyzing performance of musical rhythm. *Scandinavian Journal of Psychology*, 21:257–268.

Binet, A. and Courtier, J. (1896). Recherches graphiques sur la musique. L'année Psychologique (2), 201–222.

Bonada, J. (2000). Automatic technique in frequency domain for near-lossless time-scale modification of audio. In *Proceedings of the International Computer Music Conference*.

Bresin, R. and Friberg, A. (2000). Emotional coloring of computer-controlled music performances. *Computer Music Journal*, 24(4):44–63.

Cambouropoulos, E. (2001). The local boundary detection model (lbdm) and its application in the study of expressive timing. In *Proceedings of the International Computer Music Conference (ICMC'2001)*, Havana, Cuba.

Canazza, S., Poli, G. D., Rinaldin, S., and Vidolin, A. (1997a). Sonological analysis of clarinet expressivity. In Leman, M., editor, *Music, Gestalt, and Computing: studies in cognitive and systematic musicology*, number 1317 in Lecture Notes in Artificial Intelligence, pages 431–440. Springer.

Canazza, S., Poli, G. D., and Vidolin, A. (1997b). Perceptual analysis of musical expressive intention in a clarinet performance. In Leman, M., editor, *Music, Gestalt, and Computing: studies in cognitive and systematic musicology*, number 1317 in Lecture Notes in Artificial Intelligence, pages 441–450. Springer.

Cilibrasi, R., de Wolf, R., and Vitányi, P. (2003). Algorithmic clustering of music. Submitted. Online: `http://arxiv.org/archive/cs/0303025`.

Clarke, E. F. (1988). Generative principles in music. In Sloboda, J., editor, *Generative Processes in Music: The Psychology of Performance, Improvisation, and Composition*. Oxford University Press.

Clarke, E. F. (1991). Expression and communication in musical performance. In Sundberg, J., Nord, L., and Carlson, R., editors, *Music, Language, Speech and Brain*. MacMillan Academic and Professional Ltd.

Clynes, M. (1983). Expressive microstructure linked to living qualities. In Sundberg, J., editor, *Studies of Music Performance*, volume 39, pages 76–181. Royal Swedish Academy of Music.

Clynes, M. (1995). Microstructural musical linguistics: composers' pulses are liked most by the best musicians. *Cognition*, 55:269–310.

Cope, D. (1991). *Computers and Musical Style*. Oxford University Press.

Cuddy, L. L. and Lunney, C. A. (1995). Expectancies generated by melodic intervals: Perceptual judgments of melodic continuity. *Perception & Psychophysics*, 57:451–462.

Dannenberg, R. (1984). An on-line algorithm for real-time accompaniment. In *Proceedings of the 1984 International Computer Music Conference*. International Computer Music Association.

Dannenberg, R. and Hu, N. (2003). Polyphonic audio matching for score following and intelligent audio editors. In *Proceedings of the International Computer Music Conference*, pages 27–34.

De Poli, G., S., C., Rodà, A., Vidolin, A., and Zanon, P. (2001). Analysis and modeling of expressive intentions in music performance. In *Proceedings of the International Workshop on Human Supervision and Control in Engineering and Music*, Kassel, Germany.

Deliège, I. (1997). Similarity in processes of categorisation: Imprint formation as a prototype effect in music listening. In Ramscar, M., Hahn, U., Cambouropoulos, E., and Pain, H., editors, *Proceedings of the Interdisciplinary Workshop on Similarity and Categorisation*, pages 59–65, Edinburgh. University of Edinburgh.

Deliège, I. (2001). Introduction: Similarity perception ↔ categorization ↔ cue abstraction. *Music Perception*, 18(3):233–244.

Desain, P. and Honing, H. (1991). Towards a calculus for expressive timing in music. *Computers in Music Research*, 3:43–120.

Desain, P. and Honing, H. (1993). Tempo curves considered harmful. In "Time in contemporary musical thought" J. D. Kramer (ed.), Contemporary Music Review. 7(2).

Desain, P. and Honing, H. (1994). Does expressive timing in music performance scale proportionally with tempo? *Psychological Research*, 56:285–292.

Desain, P. and Honing, H. (1997). How to evaluate generative models of expression in music performance. isssues in AI and music evaluation and assessment. In *International Joint Conference on Artificial Intelligence*, pages 5–7, Nagoya, Japan.

Desain, P., Honing, H., and Heijink, H. (1997). Robust score-performance matching: Taking advantage of structural information. In *Proceedings of the 1997 International Computer Music Conference*, pages 337–340, San Francisco. International Computer Music Association.

Doraisamy, S. and Rüger, S. (2003). Robust polyphonic music retrieval with n-grams. *Journal of Intelligent Information Systems*, 21(1):53–70.

Drake, C. and Palmer, C. (1993). Accent structures in music performance. *Music Perception*, 10:343–378.

Eerola, T. and North, A. C. (2000). Expectancy-based model of melodic complexity. In *Proceedings of the Sixth International Conference on Music Perception and Cognition*, Keele, Staffordshire, UK. Department of Psychology.

Friberg, A. (1991). Generative rules for music performance: A formal description of a rule system. *Computer Music Journal*, 15 (2):56–71.

Friberg, A. and Battel, G. U. (2002). Structural communication. In Parncutt, R. and McPherson, G., editors, *The science and psychology of music performance*, chapter 13, pages 199–218. Oxford University Press.

Friberg, A. and Sundberg, J. (1999). Does music performance allude to locomotion? a model of final ritardandi derived from measurements of stopping runners. *Journal of the Acoustical Society of America*, 105(3):1469–1484.

Friberg, A. and Sundström, A. (2002). Swing ratios and ensemble timing in jazz performance: evidence for a common rhythmic pattern. *Music Perception*, 19(3):333–349.

Frieler, K. and Müllensiefen, D. (2005). The Simile algorithm for melodic similarity. MIREX `http://www.music-ir.org/evaluation/mirex-results/articles/similarity/frieler.pdf`.

Gabrielsson, A. (1987). Once again: The theme from Mozart's piano sonata in A major (K. 331). A comparison of five performances. In Gabrielsson, A., editor, *Action and perception in rhythm and music*, pages 81–103. Royal Swedish Academy of Music, Stockholm.

Gabrielsson, A. (1995). Expressive intention and performance. In Steinberg, R., editor, *Music and the Mind Machine*, pages 35–47. Springer-Verlag, Berlin.

Gabrielsson, A. (1999). The performance of music. In Deutsch, D., editor, *The Psychology of Music*, chapter 14, pages 501–602. Academic Press.

Gabrielsson, A. (2003). Music performance research at the millennium. *The Psychology of Music*, 31(3):221–272.

Goebl, W., Dixon, S., DePoli, G., Friberg, A., Bresin, R., and Widmer, G. (2005). 'sense' in expressive music performance: Data acquisition, computational studies, and models. S2S2 Summer School, Genova.

Goebl, W., Pampalk, E., and Widmer, G. (2004). Exploring expressive performance trajectories: Six famous pianists play six Chopin pieces. In *Proceedings of the 8th International Conference on Music Perception and Cognition, Evanston (ICMPC8)*.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Gómez, E., Grachten, M., Amatriain, X., and Arcos, J. L. (2003a). Melodic characterization of monophonic recordings for expressive tempo transformations. In *Proceedings of Stockholm Music Acoustics Conference 2003*.

Gómez, E., Klapuri, A., and Meudic, B. (2003b). Melody description and extraction in the context of music content processing. *Journal of New Music Research*, 32(1).

Goto, M. (2004). A real-time music scene description system: Predominant-F0 estimation for detecting melody and bass lines in real-world audio signals. *Speech Communication (ISCA Journal)*, 43(4):311–329.

Gouyon, F., Fabig, L., and Bonada, J. (2003). Rhythmic expressiveness transformations of audio recordings: swing modifications. In *Proceedings of the 6th International Conference on Digital Audio Effects*.

Grachten, M., Arcos, J. L., and López de Mántaras, R. (2004a). Evolutionary optimization of music performance annotation. In *Computer Music Modeling and Retrieval*, Lecture Notes in Computer Science. Springer.

Grachten, M., Arcos, J. L., and López de Mántaras, R. (2004b). Melodic similarity: Looking for a good abstraction level. In *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR 2004)*. Pompeu Fabra University.

Grachten, M., Arcos, J. L., and López de Mántaras, R. (2005a). Melody retrieval using the Implication/Realization model. MIREX `http://www.music-ir.org/evaluation/mirex-results/articles/similarity/grachten.pdf`.

Grachten, M., García, F. A., and Arcos, J. L. (2005b). Navigating through case base competence. In *Proceedings of the Sixth International Conference on Case-Based Reasoning (ICCBR-05)*, number 3620 in Lecture Notes in Computer Science, pages 282–295. Springer-Verlag.

Granot, R. and Donchin, E. (2002). Do re mi fa so la ti – constraints, congruity and musical training: An event-related brain potential study of musical expectancies. *Music Perception*, 19(4):487–528.

Haas, J. (2001). Salto - a spectral domain saxophone synthesizer. In *Proceedings of MOSART Workshop on Current Research Directions in Computer Music*, Barcelona.

Hazan, A., Grachten, M., and Ramirez, R. (2006a). Evolving performance models by performance similarity: Beyond note-to-note transformations. In *Proceedings of the Seventh International Conference on Music Information Retrieval (ISMIR)*.

Hazan, A., Ramirez, R., Maestre, E., Perez, A., and Pertusa, A. (2006b). Modelling expressive performance: A regression tree approach based on strongly typed genetic programming. In *Proceedings on the 4th European Workshop on Evolutionary Music and Art*, pages 676–687, Budapest, Hungary.

Henderson, M. T. (1937). Rhythmic organization in artistic piano performance. *University of Iowa studies in the psychology of music*, IV:281–305. Univ. Press Iowa.

Hinrichs, T. R. (1992). *Problem solving in open worlds : a case study in design*. L. Erlbaum Associates, Hillsdale, N.J.

Hiraga, R., Bresin, R., Hirata, K., and Katayose, H. (2004). Rencon 2004: Turing test for musical expression. In *Proceedings of the 2004 Conference on New Interfaces for Musical Expression (NIME)*.

Honing, H. (2003). The final ritard: on music, motion, and kinematic models. *Computer Music Journal*, 27(3):66–72.

Honing, H. (2007). Is expressive timing relational invariant under tempo transformation? *Psychology of Music*, 35(2):276–285.

Hörnel, D. and Menzel, W. (1998). Learning musical structure and style with neural networks. *Computer Music Journal*, 22 (4):44–62.

Huron, D. (1988). Alf gabrielsson (ed.): Action and perception in rhythm and music [review of]. *Psychology of Music*, 16(2):156–162.

Johnstone, J. A. (1913). *Phrasing in piano playing*. Witmark, New York.

Juslin, P. (2001). Communicating emotion in music performance: a review and a theoretical framework. In Juslin, P. and Sloboda, J., editors, *Music and emotion: theory and research*, pages 309–337. Oxford University Press, New York.

Klapuri, A. (1999). Sound onset detection by applying psychoacoustic knowledge. In *Proceedings of the 1999 International Conference on Acoustics, Speech, and Signal Processing (ICASSP'99)*.

Koffka, K. (1935). *Principles of Gestalt Psychology*. Routledge & Kegan Paul, London.

Köhler, W. (1947). *Gestalt psychology: An introduction to new concepts of modern psychology*. Liveright, New York.

Kolodner, J. L. (1983). Reconstructive memory, a computer model. *Cognitive Science*, 7:281–328.

Koton, P. (1988). Reasoning about evidence in causal explanations. In *Proceedings of AAAI-88*, Cambridge, MA. AAAI Press/MIT Press.

Lamont, A. and Dibben, N. (2001). Motivic structure and the perception of similarity. *Music Perception*, 18(3):245–274.

Langer, S. (1953). *Feeling and Form: a Theory of Art*. Scribner, New York.

Large, E. W. (1993). Dynamic programming for the analysis of serial behaviors. *Behavior Research Methods, Instruments & Computers*, 25(2):238–241.

Lemström, K., Mikkilä, N., Mäkinen, V., and Ukkonen, E. (2005). String matching and geometric algorithm for melodic similarity. MIREX `http://www.music-ir.org/evaluation/mirex-results/articles/similarity/mikkila.pdf`.

Lemström, K. and Ukkonen, E. (2000). Including interval encoding into edit distance based music comparison and retrieval. In *Proc. of the AISB'2000 Symposium on Creative & Cultural Aspects and Applications of AI & Cognitive Science*, pages 53–60. University of Birmingham.

Lerdahl, F. (1996). Calculating tonal tension. *Music Perception*, 13(3):319–363.

Lerdahl, F. and Jackendoff, R. (1983). *A Generative Theory of Tonal Music*. MIT Press.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10:707–710.

Lindström, E. (1992). 5 x "oh, my darling clementine". the influence of expressive intention on music performance. Department of Psychology, Uppsala University.

Maestre, E. and Gómez, E. (2005). Automatic characterization of dynamics and articulation of expressive monophonic recordings. In *Proceedings of the 118th Audio Engineering Society Convention*, Barcelona.

Melucci, M. and Orio, N. (2002). A comparison of manual and automatic melody segmentation. In *Proceedings of the 3rd International Conference on Music Information Retrieval*.

Meyer, L. (1956). *Emotion and meaning in Music*. University of Chicago Press, Chicago.

Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill.

Mongeau, M. and Sankoff, D. (1990). Comparison of musical sequences. *Computers and the Humanities*, 24:161–175.

Muggleton, S. (1991). Inductive logic programming. *New Generation Computing*, 8(4):295–318.

Narmour, E. (1990). *The analysis and cognition of basic melodic structures : the Implication-Realization model*. University of Chicago Press.

Narmour, E. (1992). *The analysis and cognition of melodic complexity: the Implication-Realization model*. University of Chicago Press.

Orio, N. (2005). Combining multilevel and multifeature representation to compute melodic similarity. MIREX `http://www.music-ir.org/evaluation/mirex-results/articles/similarity/orio.pdf`.

Orio, N. and Schwarz, D. (2001). Alignment of monophonic and polyphonic music to a score. In *Proceedings of the ICMC*, Havana, Cuba.

Palmer, C. (1988). *Timing in skilled music performance*. PhD thesis, Cornell University.

Palmer, C. (1989). Mapping musical thought to musical performance. *Journal of Experimental Psychology*, 15(12):331–346.

Palmer, C. (1996). Anatomy of a performance: Sources of musical expression. *Music Perception*, 13(3):433–453.

Palmer, C. (1997). Music performance. *Annual Review of Psychology*, 48:115–138.

Pardo, B. and Birmingham, W. (2002). Improved score following for acoustic performances. In *International Computer Music Conference (ICMC)*, Goteborg, Sweden. The International Computer Music Association.

Plaza, E. and Arcos, J. L. (2002). Constructive adaptation. In Craw, S. and Preece, A., editors, *Advances in Case-Based Reasoning*, number 2416 in Lecture Notes in Artificial Intelligence, pages 306–320. Springer-Verlag.

Puckette, M. and Lippe, A. C. (1992). Score following in practice. In *Proceedings, International Computer Music Conference*, pages 182–185, San Francisco. International Computer Music Association.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, 1:81–106.

Quinlan, J. R. (1990). Learning logical definitions from relations. *Machine Learning*, 5:239–266.

Ramirez, R. and Hazan, A. (2004). Rule induction for expressive music performance modeling. In *ECML Workshop Advances in Inductive Rule Learning*.

Ramirez, R., Hazan, A., and Maestre, E. (2005). Intra-note features prediction model for jazz saxophone performance. In *Proceedings of International Computer Music Conference 2005*, Barcelona.

Rasch, R. (1979). Synchronization in performed ensemble music. *Acustica*, 43:121–131.

Repp, B. H. (1990). Patterns of expressive timing in performances of a Beethoven minuet by nineteen famous pianists. *Journal of the Acoustical Society of America*, 88:622–641.

Repp, B. H. (1994). Relational invariance of expressive microstructure across global tempo changes in music performance: An exploratory study. *Psychological Research*, 56:285–292.

Repp, B. H. (1995a). Diversity and commonality in music performance - An analysis of timing microstructure in Schumann's "Träumerei". *Journal of the Acoustical Society of America*, 92(5):2546–2568.

Repp, B. H. (1995b). Expressive timing in Schumann's "Träumerei": An analysis of performances by graduate student pianists. *Journal of the Acoustical Society of America*, 98(5):2413–2427.

Repp, B. H. (1995c). Quantitative effects of global tempo on expressive timing in music performance: Some perceptual evidence. *Music Perception*, 13(1):39–58.

Rigg, M. (1964). The mood effects of music: a comparison of data from former investigations. *Journal of Psychology*, 58:427–438.

Ristad, E. S. and Yianilos, P. N. (1998). Learning string-edit distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(5):522–532.

Röbel, A. (2003). A new approach to transient processing in the phase vocoder. In *Proceedings of the 6th International Conference on Digital Audio Effects*.

Rolland, P. (1998). Flexpat: a novel algorithm for musical pattern discovery. In *Proceedings of the 12th Colloquium on Musical Infonnatics (XII CIM)*, pages 125–128, Italy.

Rolland, P. (1999). Discovering patterns in musical sequences. *Journal of New Music Research*, 28 (4):334–350.

Rubner, Y., Tomasi, C., and Guibas, L. J. (1998). A metric for distributions with applications to image databases. In *Proceedings of the 1998 IEEE International Conference on Computer Vision*, pages 59–66, Bombay, India.

Schank, R. (1982). *Dynamic Memory: a theory of reminding and learning in computers and people*. Cambridge University Press.

Scheirer, E. D. (1995). Extracting expressive performance information from recorded music. Master's thesis, MIT Media Laboratory.

Scheirer, E. D. (2000). *Music-Listening Systems*. PhD thesis, MIT Media Laboratory.

Schellenberg, E. G. (1996). Expectancy in melody: Tests of the implication-realization model. *Cognition*, 58:75–125.

Schlichte, J. (1990). Der automatische vergleich von 83.243 musikincipits aus der rism-datenbank: Ergebnisse - nutzen - perspektiven. *Fontes Artis Musicae*, 37:35–46.

Seashore, C. E. (1938). *Psychology of Music*. McGraw-Hill, New York. (Reprinted 1967 by Dover Publications New York).

Selfridge-Field, E., editor (1997). *Beyond MIDI: The Handbook of Musical Codes*. MIT Press, Cambridge, Massachusetts.

Serra, X. (1997). Musical sound modeling with sinusoids plus noise. In Roads, C., Pope, S. T., Picialli, A., and De Poli, G., editors, *Musical Signal Processing*, pages 91–122. Swets and Zeitlinger Publishers.

Skalle, P. and Aamodt, A. (2004). Knowledge-based decision support in oil well drilling. In *Proceedings of the ICIIP 2004, International Conference on Intelligent Information Systems*, Beijing, China.

Sloboda, J. A. (1983). The communication of musical metre in piano performance. *Quarterly Journal of Experimental Psychology*, 35A:377–396.

Smith, L. A., McNab, R. J., and Witten, I. H. (1998). Sequence-based melodic comparison: A dynamic programming approach. In Hewlett, W. B. and Selfridge-Field, E., editors, *Melodic Similarity. Concepts, Procedures, and Applications*, Computing in Musicology, pages 101–118. MIT Press.

Smyth, B. and Keane, M. (1998). Adaptation-guided retrieval: Questioning the similarity assumption in reasoning. *Artificial Intelligence*, 102.

Stahl, A. (2004). *Learning of Knowledge-Intensive Similarity-Measures in Case-Based Reasoning*. PhD thesis, Universität Kaiserslautern.

Sundberg, J., Friberg, A., and Frydén, L. (1991). Common secrets of musicians and listeners: an analysis-by-synthesis study of musical performance. In Howell, P., West, R., and Cross, I., editors, *Representing Musical Structure*, Cognitive Science series, chapter 5. Academic Press Ltd.

Suyoto, I. and Uitdenbogerd, A. (2005). Simple efficient n-gram indexing for effective melody retrieval. MIREX `http://www.music-ir.org/evaluation/mirex-results/articles/similarity/suyoto.pdf`.

Suzuki, T. (2003). The second phase development of case based performance rendering system "Kagurame". In *Working Notes of the IJCAI-03 Rencon Workshop*, pages 23–31.

Temperley, D. (2001). *The Cognition of Basic Musical Structures*. MIT Press, Cambridge, Mass.

The Real Book (2004). The Real Book: Sixth Edition. Hal Leonard Corporation.

Timmers, R. (2003). On the contextual appropriateness of expression. *Music Perception*, 20(3):225–240.

Timmers, R., Ashley, R., Desain, P., and Heijink, H. (2000). The influence of musical context on tempo rubato. *Journal of New Music Research*, 29(2):131–158.

Timmers, R. and Honing, H. (2002). On music performance, theories, measurement and diversity. In M.A. Belardinelli (ed.). Cognitive Processing (International Quarterly of Cognitive Sciences), 1-2, 1-19.

Timmers, R.and Ashley, R., Desain, P., Honing, H., and Windsor, L. (2002). Timing of ornaments in the theme of Beethoven's Paisiello Variations: Empirical data and a model. *Music Perception*, 20(1):3–33.

Tobudic, A. and Widmer, G. (2003). Playing Mozart phrase by phrase. In *Proceedings of the Fifth International Conference on Case-Based Reasoning (ICCBR-03)*, number 2689 in Lecture Notes in Artificial Intelligence, pages 552–566. Springer-Verlag.

Tobudic, A. and Widmer, G. (2004). Case-based relational learning of expressive phrasing in classical music. In *Proceedings of the 7th European Conference on Case-based Reasoning (ECCBR'04)*, Madrid.

Todd, N. (1989). A computational model of rubato. *Contemporary Music Review*, 3 (1).

Todd, N. (1992). The dynamics of dynamics: A model of musical expression. *Journal of the Acoustical Society of America*, 91:3540–3550.

Todd, N. (1995). The kinematics of musical expression. *Journal of the Acoustical Society of America*, 97 (3):1940–1949.

Todd, N. P. (1985). A model of expressive timing in tonal music. *Music Perception*, 3:33–58.

Toiviainen, P. and Eerola, T. (2002). A computational model of melodic similarity based on multiple representations and self-organizing maps. In *Proceedings of the 7th International Conference on Music Perception and Cognition*, Sydney. Causal Productions.

Typke, R., , Wiering, F., and Veltkamp, R. C. (2005a). Evaluating the earth mover's distance for measuring symbolic melodic similarity. MIREX `http://www.music-ir.org/evaluation/mirex-results/articles/similarity/typke.pdf`.

Typke, R., Giannopoulos, P., Veltkamp, R. C., Wiering, F., and van Oostrum, R. (2003). Using transportation distances for measuring melodic similarity. In *Proceedings of the Fourth International Conference on Music Information Retrieval (ISMIR)*.

Typke, R., Hoed, M., De Nooijer, J., Wiering, F., and Veltkamp, R. (2005b). A ground truth for half a million musical incipits. In *Proceedings of the Fifth Dutch-Belgian Information Retrieval Workshop*, pages 63–70.

Typke, R., Veltkamp, R. C., and Wiering, F. (2006). A measure for evaluating retrieval techniques based on partially ordered ground truth lists. In *Proceedings of ICME*.

Uitdenbogerd, A. L. and Zobel, J. (2002). Music ranking techniques evaluated. In Oudshoorn, M. J., editor, *Twenty-Fifth Australasian Computer Science Conference (ACSC2002)*, Melbourne, Australia. ACS.

Widmer, G. (1996). Learning expressive performance: The structure-level approach. *Journal of New Music Research*, 25 (2):179–205.

Widmer, G. (2000). Large-scale induction of expressive performance rules: First quantitative results. In *Proceedings of the International Computer Music Conference (ICMC2000)*, San Francisco, CA.

Widmer, G. (2002). Machine discoveries: A few simple, robust local expression principles. *Journal of New Music Research*, 31(1):37–50.

Widmer, G. (2003). Discovering simple rules in complex data: A meta-learning algorithm and some surprising musical discoveries. *Artificial Intelligence*, 146(2):129–148.

Windsor, W. L. and Clarke, E. F. (1997). Expressive timing and dynamics in real and artificial musical performances: using an algorithm as an analytical tool. *Music Perception*, 15(2):127–152.

Appendix A

Abbreviations and Notational Conventions

A.1 Abbreviations

MD Melodic Description
I-R Implication Realization
UTS Uniform Time Stretching
BPM Beats per Minute
PRD Principle of Registral Direction (I-R model)
PID Principle of Intervallic Difference (I-R model)
IOI Interonset Interval
OI Ornamentation Introduction (Adaptation)
II Insertion Introduction (Adaptation)
AT Analogy Transfer (Adaptation)

A.2 Names of I-R structures

P Process
VP Vector Process
IP Intervallic Process
R Reversal
IR Intervallic Reversal
VR Vector Reversal
ID Intervallic Duplication
D Duplication
[0-8] Dyadic Structure
(P) Retrospective Process
(VP) Retrospective Vector Process
(IP) Retrospective Intervallic Process
(R) Retrospective Reversal
(IR) Retrospective Intervallic Reversal
(VR) Retrospective Vector Reversal
(ID) Retrospective Intervallic Duplication
M Monadic Structure
8D Octave Dyad

A.3 Notation

$\mathbf{s}_{i:j}$	The subsequence $(s_i, \cdots, s_j)$ of a sequence $\mathbf{s}$; We adopt the convention that $\mathbf{s}_{i:j} = (s_i)$ whenever $i = j$ and $\mathbf{s}_{i:j} = \emptyset$ whenever $i > j$.
$\mathbf{s}_i$	The subsequence (s_i) of a sequence $\mathbf{s}$.
$\mathcal{P}(s_i)$	The pitch of a note s_i
$\mathcal{D}(s_i)$	The duration of a note s_i
$\mathcal{O}(s_i)$	The onset of a note s_i
$\mathcal{S}(s_i)$	The metrical strength of a note s_i
$\mathcal{E}(s_i)$	The energy of a note s_i
$\mathcal{I}(s_i)$	The size in semitones of an interval s_i
$\mathcal{IOI}(s_i)$	The interonset interval of interval s_i
$\mathcal{C}(s_i)$	The melodic direction (contour) of interval s_i
$\mathit{Size}(s_i)$	The number of notes spanned by an I-R structure s_i
$\mathit{Overlap}(s_i)$	The number of notes shared by an I-R structure s_i with its successor I-R structure
$\mathit{Direction}(s_i)$	The direction of an I-R structure s_i

Appendix B

I-R Annotated Phrases in Case Base

Scores with annotated I-R analyses of the phrases used in TempoExpress.

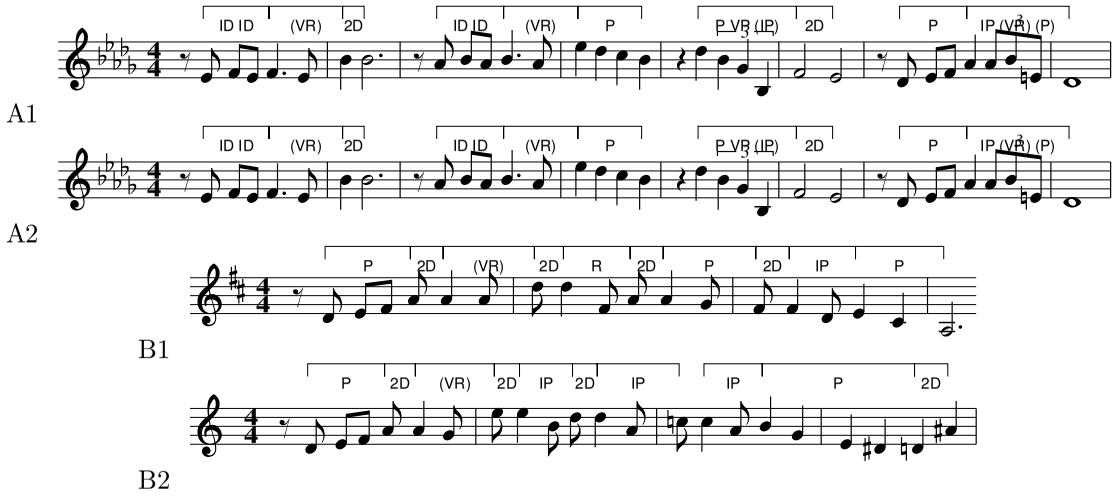

Figure B.1: Body And Soul

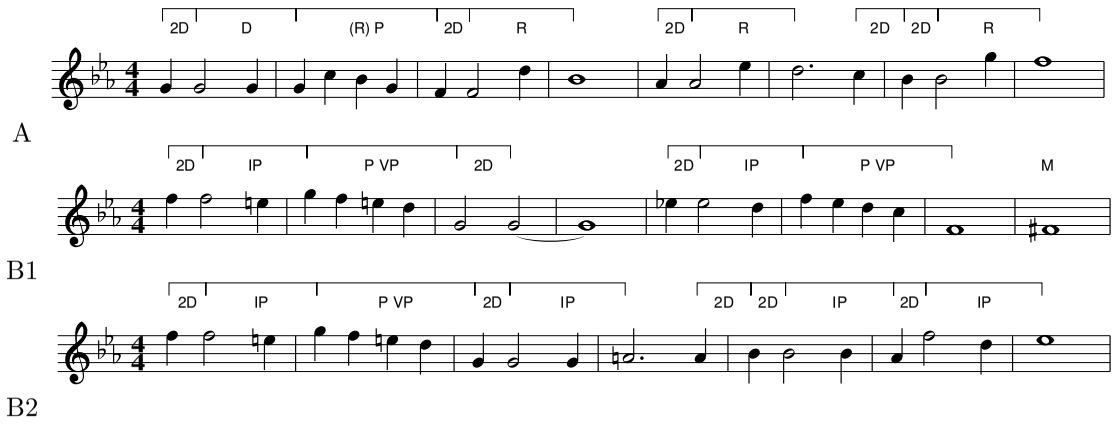

Figure B.2: Like Someone In Love

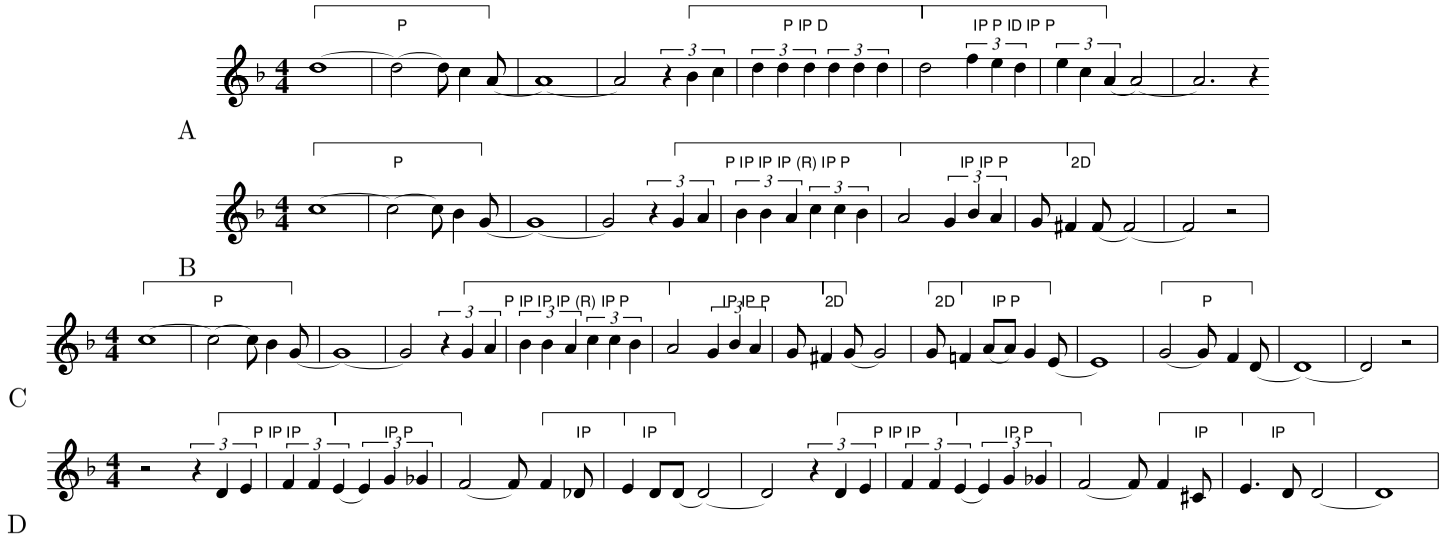

Figure B.3: Once I Loved

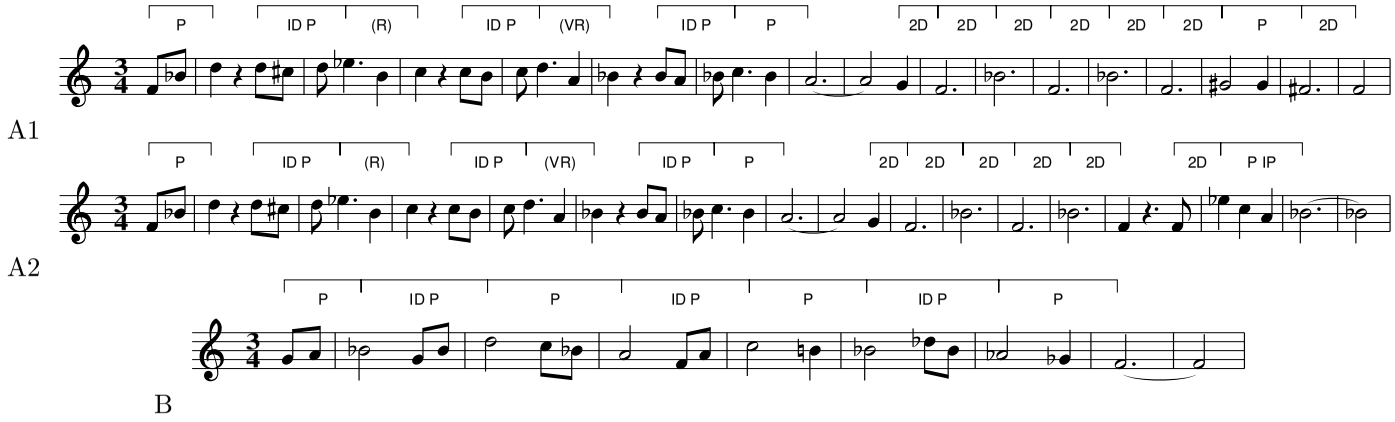

Figure B.4: Up Jumped Spring

Appendix C

Songs Used in Melodic Distance Comparison

Songs used for melodic distance comparison:

A Child Is Born
A Fine Romance
A Foggy Day
African Flower
Afro Blue
Afternoon In Paris
All Of Me
A Night In Tunisia
As Time Goes By
Autumn Leaves
Black Orpheus
Blue Bossa
Body And Soul
Confirmation
Dexterity
Donna Lee
How High The Moon
Impressions
In A Sentimental Mood
Inch Worm
Like Someone In Love
Misty
Molten Glass
My One And Only Love
Naima
Once I Loved
Ornithology
Round Midnight
Search For Peace
Sometime Ago
Song For My Father
Sophisticated Lady
Straight No Chaser
Sugar
The Song Is You
Up Jumped Spring
What Is This Thing Called Love
When I Fall In Love
When Sunny Gets Blue
Where Are You

Appendix D

List of Publications/Awards by the Author

Awards

- First prize in the Symbolic Melodic Similarity Contest of the Music Information Retrieval Exchange (MIREX) 2005.
- Best Paper Award, International Conference on Case Based Reasoning (ICCBR) 2003.

Refereed Journal Articles

- Grachten, M., Arcos, J. L., and López de Mántaras, R. (2006a). A case based approach to expressivity-aware tempo transformation. *Machine Learning*. In press. DOI: 0.1007/s10994-006-9025-9.

Refereed Conference Papers

- Hazan, A., Grachten, M., and Ramirez, R. (2006). Evolving performance models by performance similarity: Beyond note-to-note transformations. In *Proceedings of the 7th International Conference on Music Information Retrieval (ISMIR 2006)*. Victoria, Canada.
- Grachten, M., Arcos, J. L., and López de Mántaras, R. (2006). TempoExpress: An expressivity-preserving musical tempo transformation system. In *Proceedings of the 21st National Conference on Artificial Intelligence*. AAAI, Boston, MA.
- Grachten, M., Arcos, J. L., and López de Mántaras, R. (2005a). Melody retrieval using the Implication/Realization model. In *Online Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR 2005)*. Queen Mary University of London. `http://www.music-ir.org/evaluation/mirex-results/articles/similarity/grachten.pdf`.
- Grachten, M, García, F. A. , Arcos, J. Ll. (2005). Navigating through Case Base Competence In *Proceedings of the Sixth International Conference on Case Based Reasoning (ICCBR-05)*. LNCS 3620, Springer.

- Grachten, M., Arcos, J. L., and López de Mántaras, R. (2004c). Melodic similarity: Looking for a good abstraction level. In *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR 2004)*. Pompeu Fabra University.

- Grachten, M., Arcos, J. L., and López de Mántaras, R. (2004b). Evolutionary optimization of music performance annotation. In *Computer Music Modeling and Retrieval*, Lecture Notes in Computer Science. Springer.

- Grachten, M., Arcos, J. L., and López de Mántaras, R. (2004a). TempoExpress, a CBR approach to musical tempo transformations. In *Advances in Case Based Reasoning. Proceedings of ECCBR*. Springer.

- Grachten, M. and Arcos, J. L. (2004). Using the Implication/Realization model for measuring melodic similarity. In *Proceedings of the 16th European Conference on Artificial Intelligence, ECAI 2004*. IOS Press.

- Arcos, J. L., Grachten, M., and López de Mántaras, R. (2003b). Extracting performer's behaviors to annotate cases in a CBR system for musical tempo transformations. In *Proceedings of the Fifth International Conference on Case Based Reasoning (ICCBR-03)*, number 2689 in Lecture Notes in Artificial Intelligence, pages 20–34. Springer-Verlag.

- Gómez, E., Grachten, M., Amatriain, X., and Arcos, J. L. (2003a). Melodic characterization of monophonic recordings for expressive tempo transformations. In *Proceedings of Stockholm Music Acoustics Conference 2003*.

- Grachten, M., Arcos, J. L., and López de Mántaras, R. (2003). Tempo-express: tempo transformations preserving musical expression. In *Working notes of the IJCAI03 Rencon Workshop*. Online: `http://shouchan.ei.tuat.ac.jp/~rencon/IJCAI-03/IJCAI_Rencon2003_FINALLAST.pdf`.

- Grachten, M. (2001). JIG: Jazz Improvisation Generator. In *Proceedings of the MOSART Workshop on Current Research Directions in Computer Music*, Barcelona.

Theses/Reports

- Grachten, M. (2004). Global Musical Tempo Transformations using Case Based Reasoning. Unpublished doctoral pre-thesis work (DEA). Pompeu Fabra University, Barcelona, Spain.

- Grachten, M. (2002). Summary of the Music Performance Panel, MOSART Workshop 2001, Barcelona. In *Proceedings of the MOSART Midterm Meeting 2002, Barcelona*. Online: `http://www.diku.dk/publikationer/tekniske.rapporter/rapporter/03-12.pdf`.

- Grachten, M. (2001). JIG: An Approach to Computational Jazz Improvisation. Unpublished Master's Thesis. University of Groningen, The Netherlands.

Monografies de l'Institut d'Investigació en Intel·ligència Artificial

Num. 1 J. Puyol, *MILORD II: A Language for Knowledge–Based Systems*
Num. 2 J. Levy, *The Calculus of Refinements, a Formal Specification Model Based on Inclusions*
Num. 3 Ll. Vila, *On Temporal Representation and Reasoning in Knowledge–Based Systems*
Num. 4 M. Domingo, *An Expert System Architecture for Identification in Biology*
Num. 5 E. Armengol, *A Framework for Integrating Learning and Problem Solving*
Num. 6 J. Ll. Arcos, *The Noos Representation Language*
Num. 7 J. Larrosa, *Algorithms and Heuristics for Total and Partial Constraint Satisfaction*
Num. 8 P. Noriega, *Agent Mediated Auctions: The Fishmarket Metaphor*
Num. 9 F. Manyà, *Proof Procedures for Multiple-Valued Propositional Logics*
Num. 10 W. M. Schorlemmer, *On Specifying and Reasoning with Special Relations*
Num. 11 M. López-Sánchez, *Approaches to Map Generation by means of Collaborative Autonomous Robots*
Num. 12 D. Robertson, *Pragmatics in the Synthesis of Logic Programs*
Num. 13 P. Faratin, *Automated Service Negotiation between Autonomous Computational Agents*
Num. 14 J. A. Rodríguez, *On the Design and Construction of Agent-mediated Electronis Institutions*
Num. 15 T. Alsinet, *Logic Programming with Fuzzy Unification and Imprecise Constants: Possibilistic Semantics and Automated Deduction*
Num. 16 A. Zapico, *On Axiomatic Foundations for Qualitative Decision Theory - A Possibilistic Approach*
Num. 17 A. Valls, *ClusDM: A multiple criteria decision method for heterogeneous data sets*
Num. 18 D. Busquets, *A Multiagent Approach to Qualitative Navigation in Robotics*
Num. 19 M. Esteva, *Electronic Institutions: from specification to development*
Num. 20 J. Sabater, *Trust and Reputation for Agent Societies*
Num. 21 J. Cerquides, *Improving Algorithms for Learning Bayesian Network Classifiers*
Num. 22 M. Villaret, *On Some Variants of Second-Order Unification*
Num. 23 M. Gómez, *Open, Reusable and Configurable Multi-Agent Systems: A Knowledge Modelling Approach*
Num. 24 S. Ramchurn, *Multi-Agent Negotiation Using Trust and Persuasion*
Num. 25 S. Ontañon, *Ensemble Case-Based Learning for Multi-Agent Systems*
Num. 26 M. Sánchez, *Contributions to Search and Inference Algorithms for CSP and Weighted CSP*

Num. 27 C. Noguera, *Algebraic Study of Axiomatic Extensions of Triangular Norm Based Fuzzy Logics*

Num. 28 E. Marchioni, *Functional Definability Issues in Logics Based on Triangular Norms*

Num. 29 M. Grachten, *Expressivity-Aware Tempo Transformations of Music Performances Using Case Based Reasoning*

Num. 30 I. Brito, *Distributed Constraint Satisfaction*

Num. 31 E. Altamirano, *On Non-clausal Horn-like Satisfiability Problems*