Music Structure Analysis Using a Probabilistic Fitness Measure and a Greedy Search Algorithm

Jouni Paulus, *Student Member, IEEE*, and Anssi Klapuri, *Member, IEEE*

***Abstract*—This paper proposes a method for recovering the sectional form of a musical piece from an acoustic signal. The description of form consists of a segmentation of the piece into musical parts, grouping of the segments representing the same part, and assigning musically meaningful labels, such as "chorus" or "verse," to the groups. The method uses a fitness function for the descriptions to select the one with the highest match with the acoustic properties of the input piece. Different aspects of the input signal are described with three acoustic features: mel-frequency cepstral coefficients, chroma, and rhythmogram. The features are used to estimate the probability that two segments in the description are repeats of each other, and the probabilities are used to determine the total fitness of the description. Creating the candidate descriptions is a combinatorial problem and a novel greedy algorithm constructing descriptions gradually is proposed to solve it. The group labeling utilizes a musicological model consisting of N-grams. The proposed method is evaluated on three data sets of musical pieces with manually annotated ground truth. The evaluations show that the proposed method is able to recover the structural description more accurately than the state-of-the-art reference method.**

***Index Terms*—Acoustic signal analysis, algorithms, modeling, music, search methods.**

I. Introduction

HUMAN perception of music relies on the organization of individual sounds into more complex entities. These constructs occur at several time scales from individual notes forming melodic phrases to relatively long sections, often repeated with slight variations to strengthen the perception of musical organization. This paper describes a method for the automatic analysis of the musical structure from audio input, restricting the time scale to musical sections (or, parts), such as intro, verse, and chorus.

Information of the structure of a musical piece enables several novel applications, e.g., easier navigation within a piece in music players [1], piece restructuring (or mash-up of several pieces) [2], academic research of forms used in different musical styles, audio coding [3], searching for different versions of the same song [4], [5], or selecting a representative clip of the piece (i.e., music thumbnailing) [6]. A music structure analysis system provides relatively high-level information about the analyzed signal, on a level that is easily understood by an average music listener.

Manuscript received December 30, 2008; revised March 20, 2009. Current version published June 26, 2009. This work was supported by the Academy of Finland under Project 5213462 (Finnish Centre of Excellence Program 2006–2011). The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Yariv Ephraim.

The authors are with the Department of Signal Processing, Tampere University of Technology, Korkeakoulunkatu 1, FI-33720 Tampere, Finland (e-mail: jouni.paulus@tut.fi; anssi.klapuri@tut.fi).

Digital Object Identifier 10.1109/TASL.2009.2020533

A. Background

Several systems have been proposed for music structure analysis, ranging from attempts to find some repeating part to be used as a thumbnail, to systems producing a structural description covering the entire piece. The employed methods vary also. In the following, a brief overview of some of the earlier methods is provided.

To reduce the amount of data and to focus on the desired properties of the signal, features are extracted from it. The feature extraction is done in fixed-length frames or in frames synchronized to the musical beat. The main motivation for using beat-synchronized frames is that they provide a tempo-invariant time base for the rest of the analysis.

The employed features are often designed to mimic some aspects that have been found to be important for a human listener analyzing the musical structure, including changes in timbre or rhythm, indicating change of musical parts, and repetitions, especially melodic ones, as suggested in [7]. In the following, the feature vector in frame i, $i = 1, 2, \ldots, V$, is denoted by $\boldsymbol{v}_i$, and V is the number of frames in the signal.

A useful mid-level representation employed in many structure analysis methods is a $V \times V$ self-distance (or self-similarity) matrix $\boldsymbol{D}$. The element $D(i, j)$ of the matrix denotes the distance (or similarity) of the frames i and j. The self-distance matrix (SDM) is a generalization of the recurrence plot [8] in which the element values are binary (similar or different). In music structure analysis, the use of SDM was first proposed in [9] where it was used for music visualization. The patterns in the SDM are not only useful for visualization but also important in many analysis methods.

In [10], structure analysis methods are categorized into *state* and *sequence*-based systems. State-based methods consider the piece as a succession of states, while sequence-based methods assume that the piece contains repeated sequences of musical events. Fig. 1 presents an idealized view of the patterns formed in the SDM. The state representation methods basically aim to locate blocks of low distance on the main diagonal, while the sequence-based methods aim to locate off-diagonal stripes (a stripe representing low distance of two sequences). The blocks are formed when the used feature remains somewhat similar during an occurrence of a musical part, and the stripes are formed when there are sequences that are repeated later in the piece.

The locations of the block borders on the main diagonal can be searched from the SDM for segmentation [11]–[13], or

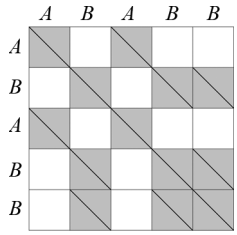

Fig. 1. Example of the structures formed in the self-distance matrix. Darker pixel value denotes lower distance. Time proceeds from left to right and from top to bottom. The example piece consists of five sections, where two parts, A and B, occur as indicated.

blocks themselves can be searched by dynamic programming [14], [15] for segmentation and recurrence analysis.

Some methods utilize the block-like information less explicitly by directly handling the feature vectors with agglomerative clustering [16], or by clustering them with hidden Markov models [17], [18]. The temporal fragmentation resulting from the use of the vector quantization models has been attempted to be reduced by pre-training the model [19], or by imposing duration modeling explicitly [20]–[22].

Because of the assumption of repetition, the sequence methods are not able to describe the entire song, but the parts that are not repeated remain undiscovered. This is not always a weakness, as some methods aim to find the chorus or a representative thumbnail of the piece utilizing the formed stripes. The stripes can be located from the SDM after enhancing them by filtering the matrix [23], [24], or by heuristic rules [25].

In addition to locating only one repeating part, some sequence methods attempt to provide a description of all repeated parts of the piece. By locating all of the repetitions, it is possible to provide a more extensive description of the structure of the piece [1], [26]. Finding a description of the whole piece can be obtained by combining shorter segments with agglomerative clustering [27], refining the segment iteratively [28], selecting repeated segments in a greedy manner [29], or by transitive deduction of segments found utilizing iterative search [30].

The authors of [31] propose to combine vector quantization of framewise features and string matching on the formed sequences to locate repeating parts. Aiming to find a path through the SDM so that the main diagonal is used as little as possible, thus utilizing the off-main diagonal stripes with *ad hoc* rules for piece structures has been attempted in [32]. Heuristic rules to force the piece structure to be one of the few stereotypical ones were presented in [33]. Formulating the properties of a typical or "good" musical piece structure mathematically, and utilizing this formulation to locate a description of the repeated parts has been attempted in [13], [34]. The method proposed in this paper can be seen as an extension of this kind of approach to provide a description of the structure of the whole piece.

B. Proposed Approach

The main novelty of the proposed method is that it relies on a probabilistic fitness measure in analyzing the structure of music pieces. A structure description consists of a segmentation of the piece to occurrences of musical parts, and of grouping of segments that are repeats of each other. The acoustic information of each pair of segments in the description is used to determine the probability that the two segments are repeats of each other. The probabilities are then used to calculate the total fitness of the description. A greedy algorithm is proposed for solving the resulting search problem of finding the structure that maximizes the fitness measured. Furthermore, the resulting description is labeled with musically meaningful part labels. To the authors' knowledge, this is the first time that the labeling can be for arbitrary music pieces.

The proposed method utilizes three acoustic features describing different aspects of the piece. Self-distance matrices are calculated from all the three features, and using the information embedded in the SDM, the system performs a search to create a segmentation and a segment clustering that maximize the fitness over the whole piece. The "blocks" and the "stripes" in multiple SDMs are used.

The rest of the paper is organized as follows. Section II details the proposed method. Then experimental results are described in Section III. Finally, Section IV concludes the paper. Parts of this work have been published earlier in [35]–[37].

II. Proposed Method

The proposed analysis method relies on a fitness function for descriptions of musical structures. This function can be used to compare different descriptions of the same piece and determine how plausible they are from the perspective of the acoustic signal. In addition to the fitness function, a search method for generating a maximally fit description is presented.

A. Fitness Measure

From the point of view of acoustic properties, a good description of musical structure has much in common with defining a good clustering of data points: the intra-cluster similarity should be maximized while minimizing the inter-cluster similarity. In terms of musical structure: the segments assigned to a group (forming the set of all occurrences of a musical part) should be similar to each other while the segments from different groups should be maximally dissimilar. Compared to basic clustering, individual frames of the musical piece cannot be handled as individual data points in clustering, because it would fragment the result temporally, as noted in [21]. Instead, the frames are forced to form sequences.

All the possible segments of a piece are denoted by set $\mathbb{S}$. A subset $\mathbb{S}' \subset \mathbb{S}$ of this consisting of S segments $s_i \in \mathbb{S}'$ that do not overlap and cover the whole piece defines one possible segmentation of the piece. The *group* of segment s_i is returned by a group assignment function $g(\cdot)$; if $g(s_i) = g(s_j)$, the segments belong to the same group and are occurrences of the same musical part. A description $\mathbb{E}$ of the structure of the piece is a combination of a segmentation and grouping of the segments $\mathbb{E} = (\mathbb{S}', g)$.

When a segmentation $\mathbb{S}'$ and the acoustic data is given, it is possible to compare all pairs of segments s_i and s_j, and to determine a probability $\hat{p}(g(s_i) = g(s_j))$ that the segments belong to the same group. Because the segments can be of different lengths, a weighting factor $W(s_i, s_j)$ is determined for each

segment pair in addition to the probability. The overall fitness of the description $\mathbb{E}$ is defined as

$$P(\mathbb{E}) = \sum_{s_i \in \mathbb{S}'} \sum_{s_j \in \mathbb{S}'} W(s_i, s_j) l(s_i, s_j, g) \tag{1}$$

where

$$l(s_i, s_j, g) = \begin{cases} \log(\hat{p}(g(s_i) = g(s_j))), & \text{if } g(s_i) = g(s_j) \\ \log(1 - \hat{p}(g(s_i) = g(s_i))), & \text{if } g(s_i) \neq g(s_j). \end{cases} \tag{2}$$

Here, the value of the weighting factor $W(s_i, s_j)$ is defined as

$$W(s_i, s_j) = L(s_i) L(s_j) \tag{3}$$

where $L(s_i)$ denotes the length of segment s_i in frames. This causes the sum of all weighting factors to equal the number of elements in the SDM.

Having defined the fitness measure, the structure analysis problem now becomes a task of finding the description $\mathbb{E}_{OPT}$ that maximizes the fitness function given the acoustic data

$$\mathbb{E}_{OPT} = \arg\max_{\mathbb{E}} \{P(\mathbb{E})\}. \tag{4}$$

Equation (1) defines the fitness of structural descriptions using relatively abstract terms. To apply the fitness measure, candidate descriptions should be constructed for evaluation and the probabilities in (1) and (2) should be calculated from the acoustic input. The rest of this paper describes how these tasks can be accomplished using a system whose block diagram is illustrated in Fig. 2. The system extracts acoustic features using beat-synchronized frame blocking. Separate SDMs are calculated for each feature, to be used as a mid-level representation. Using the information in the SDMs, a large amount of candidate segments is created and all non-overlapping segment pairs are compared. The comparison produces the pairwise probabilities $\hat{p}(g(s_i) = g(s_j))$ and the weights $W(s_i, s_j)$ that are used to evaluate the fitness measure (1). A greedy search algorithm is employed to create description candidates gradually and to evaluate their fitness. The resulting descriptions are labeled using musically meaningful labels, such as verse and chorus. The best description found is then returned. These steps are described in the rest of this section.

B. Feature Extraction

The use of three features is proposed, all of them with two different time scales to provide the necessary information for further analysis. The use of multiple features is motivated by the results of [7], which suggest that change in timbre and in rhythm are important cues for detecting structural boundaries. The use of multiple time scales has been proposed, e.g., in [4] and [38].

The feature extraction starts by estimating the locations of rhythmic beats in the audio using the method from [39]. It was noted that the system may do π-phase errors in the estimation. The effect of these errors is alleviated by inserting extraneous

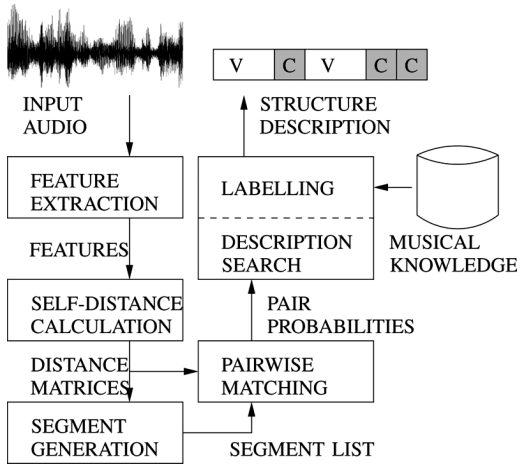

Fig. 2. Overview of the proposed method. See the text for description.

beats between each two beats, effectively halving the pulse period.

Like in several earlier publications, mel-frequency cepstral coefficients (MFCCs) are used to describe the timbral content of the signal. The rhythmic content is described with *rhythmogram* proposed in [14]. The third feature, chroma, describes the tonal content. The MFCCs and chroma are calculated in 92.9-ms frames with 50% frame overlap, while rhythmogram uses frames up to several seconds in length with the hop of 46.4 ms. After the calculation, each feature is averaged over the beat frames to produce a set of beat-synchronized features.

The MFCCs are calculated using 42-band filter bank, omitting the high-pass pre-emphasis filter sometimes used as a preprocessing. The log-energies of the bands are discrete cosine transformed (DCT) to reduce the correlation between bands and to perform energy compaction. After the DCT step, the lowest coefficient is discarded and 12 following coefficients are used as the feature vector.

The chroma is calculated using the method proposed in [40]. First, the saliences for different fundamental frequencies in the range 80–640 Hz are calculated. The linear frequency scale is transformed into a musical one by selecting the maximum salience value in each frequency range corresponding to a semitone. The semitone number for frequency f is given in MIDI note numbers by

$$F_{\text{MIDI}}(f) = F_{A4} + \left\lfloor 12 \log_2 \left(\frac{f}{f_{A4}} \right) \right\rceil \tag{5}$$

where $F_{A4} = 69$ is the MIDI note number for the reference frequency $f_{A4} = 440$, and $\lfloor \cdot \rceil$ denotes rounding to the nearest integer. Finally, the octave equivalence classes are summed over the whole pitch range to produce a 12-dimensional chroma vector. This method is used instead of directly mapping frequency bins after discrete Fourier transform (as done, e.g., in [1], [23]), because in the experiments the salience estimation front-end proved to focus more on the energy of tonal sounds and reduce some of the undesired noise caused by atonal sounds, such as drums.

For both MFCC and chroma, the feature sequences are temporally filtered with a Hanning window weighted median filter.

The purpose of the filtering is to focus the feature on the desired time-scale. The shorter filter length is used to smooth short-time deviations for enhancing the stripes on the SDM. The longer window length is intended to focus on longer time-scale similarities, enhancing the block formation on the SDMs.

The rhythmogram calculation utilizes the onset accentuation signal produced in the beat detection phase. The original method [14] used a perceptual spectral flux front-end to produce a signal sensitive to sound onsets. In the proposed method, this is replaced by summing the four accentuation signals to produce one onset accentuation signal. The rhythmogram is the autocorrelation function values of the accentuation signal calculated in successive windows after the global mean has been removed from it. The window length is determined by the target time-scale, and the autocorrelation values between the lags 0 and a maximum of 2 s are stored.

The time-scale focus parameters (the median filter window lengths for MFCCs and chroma, and the autocorrelation window length for rhythmogram) were selected with a method described in Section II-E. After the temporal filtering the features are normalized to zero mean and unity variance over the piece.

C. Self-Distance Matrix Calculation

From each feature and time-scale alternative, a self-distance matrix is calculated. Each element $D(i,j)$ of the matrix defines the distance between the corresponding frames i and j calculated with cosine distance measure

$$d(\boldsymbol{v}_i, \boldsymbol{v}_j) = 0.5\left(1 - \frac{\langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle}{||\boldsymbol{v}_i|| ||\boldsymbol{v}_j||}\right) \tag{6}$$

where $\boldsymbol{v}_i$ is the feature vector in frame i, $\langle \cdot, \cdot \rangle$ denotes vector dot product, and $|| \cdot ||$ is vector norm.

In many popular music pieces, musical modulation of the key in the last chorus section is used as an effect. This causes problems with the chroma feature as the energies shift to different pitch classes, effectively causing a circular rotation of the chroma vector.[1] To alleviate this problem, it has been proposed to apply chroma vector rotations and calculate several SDMs instead of only one testing all modulations and using the minimum distances [1], [41]. Modulation inversion both on frame and segment pairs were tested, but they did not have a significant effect on the overall performance and the presented results are calculated without them.

D. Segment Border Candidate Generation

Having the SDMs, the system generates a set of segment border candidates that are points in the piece on which a segment may start or end. If a segment is allowed to begin or end at any location, the number of possible segmentations and structural descriptions increases exponentially as a function of the border candidate locations. The combinatorial explosion is reduced by generating a smaller set of border candidates. Not all of the candidates have to be used in the final segmentation, but the points used in the segmentation have to be from this set.

[1]Naturally the modulation affects also MFCCs, but the effect is considerably smaller.

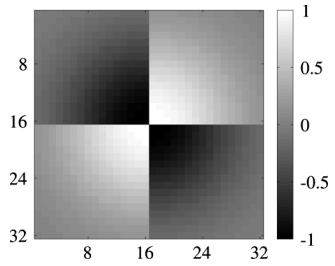

Fig. 3. Example of a Gaussian weighted detection kernel with $m = 32$ and $\sigma = 0.5$.

In the proposed method, the border candidates are generated using the novelty calculation proposed in [11]. A $m \times m$ detection kernel matrix $\boldsymbol{K}$ is correlated along the main diagonal of the SDM. The correlation values are collected to a novelty vector $\boldsymbol{n}$. Peaks in this vector, corresponding to corners in the SDM, are detected using median-based dynamic thresholding and used as the border candidates. The novelty vector is calculated from all six SDMs, three acoustic features and two time-scale parameters, and then summed. For one SDM the novelty is calculated as

$$n(k) = \sum_{i=-m/2}^{m/2-1} \sum_{j=-m/2}^{m/2-1} K\left(\frac{m}{2}+i, \frac{m}{2}+j\right) D(k+i, k+j). \tag{7}$$

The matrix $\boldsymbol{D}$ is padded with zeros in non-positive indices and indices larger than the size of the matrix.

The kernel matrix $\boldsymbol{K}$ has a 2×2 checkerboard-like structure

$$\boldsymbol{K} = \begin{pmatrix} \boldsymbol{Q}_{\mathrm{TL}} & \boldsymbol{Q}_{\mathrm{TR}} \\ \boldsymbol{Q}_{\mathrm{BL}} & \boldsymbol{Q}_{\mathrm{BR}} \end{pmatrix}$$

where the following symmetries hold:

$$\boldsymbol{Q}_{\mathrm{TL}} = -\boldsymbol{J}\boldsymbol{Q}_{\mathrm{BL}} = -\boldsymbol{Q}_{\mathrm{TR}}\boldsymbol{J} = \boldsymbol{J}\boldsymbol{Q}_{\mathrm{BR}}\boldsymbol{J}. \tag{8}$$

Matrix $\boldsymbol{J}$ is an $m/2 \times m/2$ matrix with ones on the main anti-diagonal and zeros elsewhere. It reverses the order of matrix columns when applied from right and the order of matrix rows when applied from left.

In the simplest approach, the values in $\boldsymbol{Q}_{\mathrm{TL}}$ are all -1, but as suggested in [11], the kernel matrix values are weighted by radial Gaussian function giving less weight to the values far from the center of the kernel

$$Q_{\mathrm{BR}}(x,y) = -\exp\left(-\frac{r^2}{2\sigma^2}\right) \tag{9}$$

where the radius r is defined by

$$r^2 = \frac{4}{m^2}\left((x-1)^2 + (y-1)^2\right) \tag{10}$$

and the width parameter value $\sigma = 0.5$ and kernel width $m = 32$ were noted to perform well in the evaluations. The resulting kernel is illustrated in Fig. 3. In the experiments, the 30 largest peaks in the novelty vector and the signal end points were used as the set of segment border candidates.

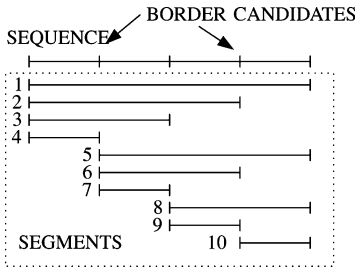

Fig. 4. Illustration of generating the segments.

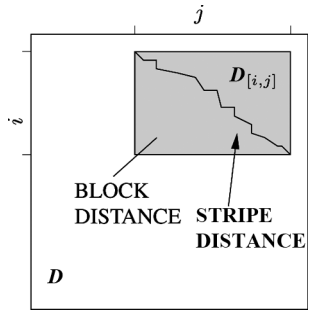

Fig. 5. Submatrix $\boldsymbol{D}_{[i,j]}$ of SDM $\boldsymbol{D}$ used in the calculation of the distances between the segments s_i and s_j.

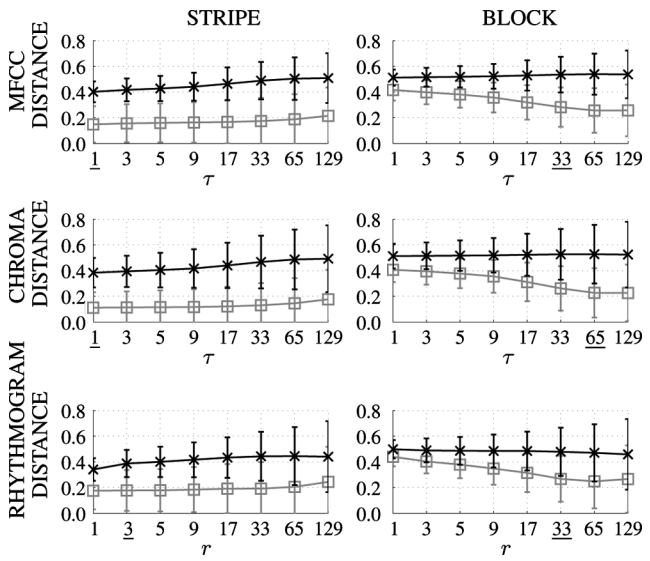

Fig. 6. Effect of the time-scale parameter on segment pair distances calculated over all pieces in the *TUTstructure07* data set. For MFCC and chroma feature the parameter τ is the median filtering window length. For rhythmogram the varied parameter r is the autocorrelation length. The lines denote the average distance values for segments from the same group (□) and from a different group (×). The error bars around the marker denote the standard deviation of the distances. The chosen parameter values are marked with underlining.

E. Segment Pair Distance Measures

After the set of border candidates has been generated, all segments between all pairs of border candidates are created. These segments form the set $\mathbb{S}$, from which the segmentation in the final description is a subset of. This is illustrated in Fig. 4, where ten possible segments are generated from five border candidates.

For each segment pair and feature, two distances are calculated: a *stripe distance* and a *block distance*. The stripe distance measures the dissimilarity of the feature sequences of the two segments, whereas the block distance measures the average dissimilarity of all frame pairs of the two segments. Two distance measures are used because it is assumed that they provide complementary information.

The main difference and motivation of using these two distance measures are illustrated in Fig. 1 which contains a stereotypical SDM of a simple piece with the structure "A, B, A, B, B." If only stripe distance was used, it would be difficult to locate the border between "A" and "B" without any additional logic, because "A" is always followed by "B." Similarly, if only block distance was used, the border between the second and third "B" would be missed without any addition logic.

The compared segments s_i and s_j define a submatrix $\boldsymbol{D}_{[i,j]}$ of distance matrix $\boldsymbol{D}$. The contents of this submatrix are used to determine the acoustic match of the segments. The submatrix and the distance measures are illustrated in Fig. 5.

The block distance $d_B(s_i, s_j)$ is calculated as the average of the distances in the submatrix

$$d_B(s_i, s_j) = \frac{1}{L(s_i)L(s_j)} \sum_{x=1}^{L(s_i)} \sum_{y=1}^{L(s_j)} D_{[i,j]}(x, y). \tag{11}$$

The stripe distance $d_S(s_i s_j)$ is calculated by finding the path with the minimum cost through the submatrix $\boldsymbol{D}_{[i,j]}$ and normalizing the value by the minimum possible path length

$$d_S(s_i s_j) = \frac{\tilde{D}_{[i,j]}(L(s_i), L(s_j))}{\max(L(s_i), L(s_j))} \tag{12}$$

where elements of the partial path cost matrix $\tilde{\boldsymbol{D}}_{[i,j]}$ are defined recursively by

$$\tilde{D}_{[i,j]}(x, y) = D_{[i,j]}(x, y) + \min \begin{cases} \tilde{D}_{[i,j]}(x-1, y-1) \\ \tilde{D}_{[i,j]}(x-1, y) \\ \tilde{D}_{[i,j]}(x, y-1) \end{cases} \tag{13}$$

with the initialization $\tilde{D}_{[i,j]}(0, 0) = 0$. Note that the path transitions do not have any associated cost.

The effect of the time-scale parameter on the resulting distance values was evaluated using a manually annotated data set of popular music pieces that will be described in Section III-A. The median filtering window length τ was varied with MFCC and chroma features, and the autocorrelation window length r was varied for rhythmogram. The values of distances for segments from the same groups and from different groups were calculated with both of the proposed distance measures. The effect of the time-scale parameter is illustrated in Fig. 6. The final parameter values used in the evaluations were determined from this data by assuming the distance values to be distributed as Gaussians and selecting the parameter value minimizing the overlapping mass of the distributions. The used parameter values are indicated in the figure.

F. Probability Mapping

Once the distance of two segments has been calculated based on the used features and distance measures, the obtained dis-

tance values are transformed to probabilities to enable evaluating the overall fitness measure (1). In the following, both block and stripe distance of a segment pair are denoted with $d(s_i, s_j)$ to simplify the notation and because the processing is similar to both. The probability that two segments s_i and s_j belong to the same group is determined from the distance $d(s_i, s_j)$ between the segments using a sigmoidal mapping function from distance to probability. The mapping function is given by

$$p(g(s_i) = g(s_i)) = (1 + \exp(z_1 d(s_i, s_j) + z_0))^{-1} \quad (14)$$

where $d(s_i, s_j)$ is the distance measured from the acoustic data. The sigmoid parameters z_1 and z_0 are determined using the Levenberg–Marquardt algorithm for two-class logistic regression [42]. The data for the fit is obtained from the manual ground truth annotations.

The probabilities obtained for all of the six distance values (three acoustic features and two distance measures) are combined with weighted geometric mean

$$\begin{aligned}\hat{p}(g(s_i) =& g(s_j)) \\ &= \left(\prod_b p_b(g(s_i) = g(s_j))^{w_b} \right)^{1/\sum_b w_b} \end{aligned} \quad (15)$$

where b is a variable distinguishing the six probability values, and w_b is the weight of the corresponding feature and distance combination. In the experiments, binary weights were tested and the presented results are obtained using all but rhythmogram stripe probability with equal weights. For more details on the feature combinations, see [36].

It is possible to impose heuristic restrictions on the segments by adjusting the pairwise probabilities manually after the different information sources have been combined. Here, a length restriction was applied prohibiting larger than 50% differences in segment lengths within a group.

G. Solution for the Optimization Problem

The optimization problem (4) is a combinatorial problem. It can be formulated as a path search in a directed acyclic graph (DAG) where each node represents a possible segment in the piece with a specific group assignment, and there is an arc between two nodes only if the segment of the target node is directly following the segment of the source node. This process is illustrated by the graph in Fig. 7 which is constructed from the segments in Fig. 4 after allowing the use of two groups.

The way the total fitness (1) is defined to evaluate all segment pairs in the description causes the arc costs to depend on the whole earlier path, i.e., the transition from a node to a following one has as many different costs as there are possible routes from the start to the source node. This prohibits the use of many efficient search algorithms as problem cannot be partitioned into smaller subproblems.

Considering the applications of the structure analysis system, it would be desirable that the search would be able to produce some solution relatively quickly, to improve it when given more time, and to return the globally optimal result at some point. If the search for the global optimum takes too long, it should be possible to stop the search and use the result found at that point. A novel algorithm named *Bubble token passing* (BTP) is proposed to fulfil these requirements. BTP is inspired by the token passing algorithm [43] often used in continuous speech recognition. In the algorithm, the search state is stored using *tokens* tracking the traveled path and recording the associated fitness. In the following, the term node is changed to *state* to better conform the token passing terminology.

A pseudocode description of the algorithm is given in Fig. 8. The search is initiated by augmenting the formed DAG with start and end states and inserting one token to the start state. After this, the algorithm main loop is executed until the solution converges, some maximum iteration limit is reached, or there are no more tokens in the system. At each iteration, each state selects the β best tokens and propagates them to the following states (loop on line 4 of Fig. 8). When a token is inserted to a state, the

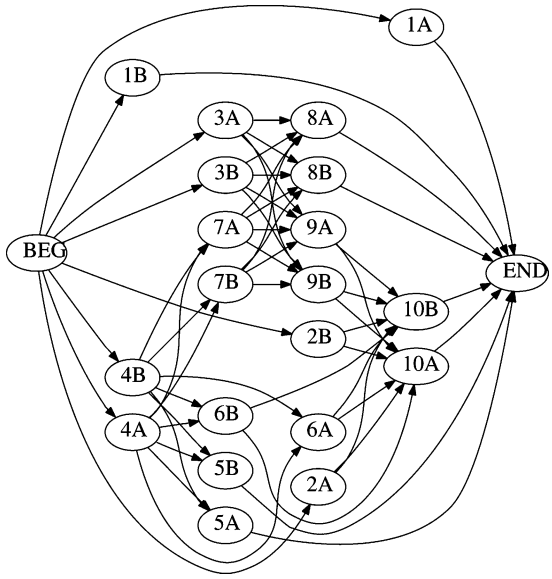

Fig. 7. Example DAG generated by the segments in Fig. 4 after allowing only two groups: A and B.

```
1:  while iter<maxItes & not converged do
2:    for sourceState in allStates do
3:      // propagate the β best tokens
4:      for n = 1 to β do
5:        // token is propagated to the following states
6:        for targetState in sourceState.followingStates do
7:          copyToken = sourceState.tokenList[n].copy()
8:          copyToken.updatePath(target)
9:          copyToken.updateCost(target)
10:         targetState.arrivingList.insert(copyToken)
11:       end for
12:     end for
13:     // after propagation, remove the token
14:     sourceState.tokenList.remove(1 ... β)
15:   end for
16:   for state in allStates do
17:     // merge arrived to storage and store only the best
18:     state.tokenList.mergeWith(arrivingList)
19:     state.tokenList.sort()
20:     state.tokenList[(α + 1) ... end].delete()
21:   end for
22: end while
```

Fig. 8. Pseudo-code description of the proposed bubble token passing search algorithm.

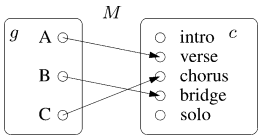

Fig. 9. Labeling process searches for an injective mapping M from a set of segment groups g to musically meaningful labels c.

state is added to the traveled path and the fitness value is updated with (16). After all states have finished the propagation, the arrived tokens are merged to a list of tokens, the list is sorted, and only α fittest are retained, the rest are removed (loop starting on line 16). After this the main iteration loop starts again.

The tokens arriving to the end state describe the found descriptions. The first solutions will be found relatively quickly, and as the iterations proceed, more tokens will "bubble" through the system to the final state. Since the tokens are propagated in best-first order and only some of the best tokens are stored to following iterations, the search is greedy, but the parameters β and α control the greediness and the scope of the search. The number of stored tokens α controls the overall greediness: the smaller the value, the fewer of the less fit partial paths are considered for continuation and more probable it will be to miss the global optimum. An exhaustive search can be accomplished by storing all tokens. The number of propagated tokens β controls the computational complexity of each main loop iteration: the more tokens are propagated from each state, the more rapidly the total number of tokens in the system increases and the more fitness updates have to be calculated at each iteration. The values used in the experiments ($\beta = 10,\ \alpha = 200$) proved to be a reasonable tradeoff between the exhaustivity and computational cost of the search, and the search converged often after 30–40 iterations.

When a token is inserted to a state corresponding to segment s_i with the group set to $g(s_i)$, the associated path fitness is updated with

$$\Delta P(s_i) = W(s_i, s_i) l(s_i, s_i, g) + 2 \sum_{s_j \in \mathbb{S}'_{i-1}} W(s_j, s_i) l(s_j, s_i, g) \tag{16}$$

where $\mathbb{S}'_i$ is a subset of $\mathbb{S}'$ after adding the ith segment to it, and starting from $\mathbb{S}_0 = \emptyset$.

The fitness of the whole description $\mathbb{E}$ can be obtained by summing these terms over the whole piece

$$P(\mathbb{E}) = \sum_{s_i \in \mathbb{S}'} \Delta P(s_i). \tag{17}$$

It is trivial to verify that this is equal to (1).

H. Musical Part Labelling

The description found by solving the optimization problem (4) consists of a segmentation of the piece and a grouping of the segments. Especially if the analysis result is presented for a human, the knowledge of musically meaningful labels on the segments would be appreciated, as suggested by a user study [44]. To date, none of the structure analysis systems, with the exception of the system proposed in [33], provides musically meaningful labels to the groups in the analysis result. The method in [33] utilized rigid forms where the analyzed piece was fitted to, and the forms contained also the part label information.

The method proposed here models sequences of musical parts with N-grams utilizing the $(N-1)$th order Markov assumption stating that the probability of label c_i given the preceding labels $c_{1:(i-1)}$ depends only on the history of length $N-1$

$$p(c_i | c_{1:(i-1)}) = p(c_i | c_{(i-N+1):(i-1)}). \tag{18}$$

The N-gram probabilities are trained using a set of musical part label sequences that are formed by inspecting the manually annotated structures of a large set of musical pieces. The parts are ordered based on their starting time, and the part labels are set in the corresponding order to produce a training sequence. The N-gram models are then used to find an injective mapping M from the groups g in the analysis result to the musical labels c

$$M : g \rightarrow c. \tag{19}$$

This process is illustrated also in Fig. 9.

When labeling the analysis result, the label assignment maximizing the resulting cumulative N-gram probability over the description

$$p(c_{1:S}) = \prod_{i=1}^{S} p(c_i | c_{(i-N-1):(i-1)}) \tag{20}$$

is searched. An algorithm for the *post-process labeling* a found structural description was presented and evaluated in [35].

Another way to perform the labeling is to integrate the labeling model to the overall fitness function. In this case, the fitness does not only assess the segmentation of the piece and the grouping of the segments, but also the labeling of the groups. The difference to (1) is that now the order in which the segments are evaluated matters, and the segment set $\mathbb{S}'$ needs to be ordered by the starting times of the segments $\mathbb{S}' = (s'_1, s'_2, \ldots, s'_S)$. The description $\mathbb{E}$ can be transformed into a label sequence by applying the mapping function by

$$M(g(s'_{1:i})) \equiv c_{1:i}. \tag{21}$$

The N-gram probabilities have to be evaluated already during the search which is accomplished by modifying the fitness measure (1) to

$$P(\mathbb{E}_{\mathrm{LM}}) = \sum_{i=1}^{S} \sum_{j=1}^{S} W(s'_i, s'_j) l(s'_i, s'_j, g) + \frac{w_l A}{S-1} \sum_{i=1}^{S} \log\left(p(M(g(s'_i)) M(g(s'_{1:(i-1)})))\right) \tag{22}$$

where w_l is the relative weight given for the labeling model, and

$$A = \sum_{i=1}^{S} \sum_{j=1}^{S} W\left(s'_i, s'_j\right). \tag{23}$$

The subscript in $\mathbb{E}_{\mathrm{LM}}$ is added to denote the integrated "labeling model." In effect, the additional term is the average part label transition log-likelihood multiplied by the weighting factors of

the segment pairs. The labeling model likelihoods are normalized with the number of transitions. This is done to ensure that explanations with different number of parts would give an equal weight for the labeling model.

Now the fitness function can be considered to be constructed of two terms: the acoustic information term on the top row of (22) and the musicological term on the bottom row. The optimization of this fitness function can be done using the same bubble token passing algorithm after modifying the token fitness update formula (16) to include the N-gram term. In fact, the same search algorithm can be used to perform the postprocess labeling, too. In that case, the acoustic matching terms have to be modified to enforce the grouping sequence.

III. Results

The proposed analysis system was evaluated with simulations using three manually annotated data sets of popular music pieces. Several different evaluation metrics were used to provide different points of view for the system performance.

A. Data

Three data sets were used in the evaluations *TUTstructure07*, *UPF Beatles*, and *RWC Pop*. The first consists of 557 pieces aimed to provide a representative sample of radio-play pieces. Approximately half of the pieces are from pop/rock genres and the rest sample other popular genres, such as hip hop, country, electronic, blues, jazz, and schlager.[2] The data set was compiled and annotated at Tampere University of Technology, and the annotation was done by two research assistants with some musical background. A notable characteristics of the data set is that it contains pieces from broad range of musical styles with differing timbral, melodic, and structural properties.

The second used data set consists of 174 songs by The Beatles. The original piece forms were analyzed and annotated by musicologist Alan W. Pollack [45], and the segmentation time stamps were added at Universitat Pompeu Fabra (UPF).[3] Some minor corrections to the data were made at Tampere University of Technology, and the corrected annotations along with a documentation of the modifications are available.[4] Major characteristic of this data set is that all the pieces are from the same band, with less variation in musical style and timbral characteristics than in the other data sets.

The audio data in the third data set consists of the 100 pieces of the Real World Computing Popular Music Database [46], [47]. All of the pieces were originally produced for the database; a majority of the pieces (80%) represent 1990's Japanese chart music, while the rest resemble the typical 1980s American chart hits.

All data sets contain the structure annotated for the whole piece. Each structural segment is described by its start and end times, and a label provided to it. Segments with the same label are considered to belong to the same group.

[2]A full list of pieces is available at http://www.cs.tut.fi/sgn/arg/paulus/TUT structure07_files.html

[3]http://www.iua.upf.edu/%7Eperfe/annotations/sections/license.html

[4]http://www.cs.tut.fi/sgn/arg/paulus/structure.html#beatles_data

B. Reference System

The performance of the proposed system is compared with a reference system [22] aimed for the same task. As the low-level feature it uses the MPEG-7 AudioSpectrumProjection [48] from 600 ms frames with 200-ms hop. The frames are clustered by training a 40-state hidden Markov model on them and then decoding with the same data. The resulting state sequence is transformed to another representation by calculating sliding state histograms from seven consecutive frames. The histograms are then clustered using temporal constraints. The used implementation was from the "QM Vamp Plugin" package version 1.5.[5] The implementation allows the user to select the feature used, the maximum number of different segment types, and minimum length of the segment. A grid search over the parameter space was done to optimize the parameters, and the presented results were obtained using the "hybrid" features, maximum of six segment types, and minimum segment length of 8 s. These parameter values provided the best general performance, and when tested with the same 30-song Beatles data set[6] as in the original publication they produced F-measure of 60.7% compared to the 60.4% reported in [22].

C. Experimental Setup

Because the proposed method needs training of some parameters, the evaluations were run using a tenfold cross-validation scheme with random fold assignment. At each cross-validation fold, 90% of the pieces are used to calculate the N-gram models for part label sequences and to train the distance-to-probability mapping functions, while the remaining 10% are used for testing. The presented results are averaged over all folds. As the reference method [22] does not need training, the evaluations were run for the whole data at once, and different parameter values were tested in a grid search manner.

To allow determining the possible bottlenecks of the proposed system, several evaluation schemes were employed:

- Full analysis. The system is given only the audio; it has to generate the candidate border locations, determine segmentation, grouping, and group labeling. Referred with *full* in the result tables.
- Segmentation and labeling, extraneous borders. The system generates border candidates by itself, but the border locations from the annotations are included in the candidate set by replacing the closest generated candidate with the one taken from annotations. Referred with *salted* in the results.
- Grouping and labeling. The system is given the correct segmentation, but it has to determine the grouping of the segments and labeling of the groups. Referred with *segs* in the tables.
- Labeling only. The correct segmentation and grouping is given to the system. It only has to assign each group with an appropriate musical label. This is referred with *labeling* in the result tables.

[5]http://www.elec.qmul.ac.uk/digitalmusic/downloads/index.html#qm-vamp-plugins

[6]http://www.elec.qmul.ac.uk/digitalmusic/downloads/#segment

TABLE I
EVALUATION RESULTS ON *TUTSTRUCTURE07* (%)

method	F	R_P	R_R	labeling	S_O	S_U
annotators	89.4	90.1	89.8	68.0	91.0	91.6
reference [22]	59.9	62.2	60.2	N/A	64.1	67.3
full w/LM	62.4	68.5	62.6	37.9	68.9	71.6
full post-LM	62.6	69.8	62.1	35.3	68.8	72.5
salted w/LM	67.5	73.8	67.7	38.2	74.5	79.0
salted post/LM	67.7	75.5	67.1	37.3	74.5	80.5
segs w/LM	84.4	91.4	81.4	48.1	87.8	94.5
segs post-LM	84.4	91.1	81.5	47.7	87.7	94.2
labeling	N/A	N/A	N/A	62.5	N/A	N/A

TABLE II
EVALUATION RESULTS ON *UPF BEATLES* (%)

method	F	R_P	R_R	labeling	S_O	S_U
reference [22]	58.4	68.3	53.3	N/A	55.2	68.3
full w/LM	59.9	72.9	54.6	35.2	60.4	71.7
full post-LM	58.6	73.7	52.6	26.6	59.3	72.5
labeling	N/A	N/A	N/A	71.8	N/A	N/A

TABLE III
EVALUATION RESULTS ON *RWC POP* (%)

method	F	R_P	R_R	labeling	S_O	S_U
reference [22]	58.1	49.3	73.8	N/A	77.0	60.5
full w/LM	63.7	60.3	72.1	34.4	79.3	72.0
full post-LM	63.3	59.9	72.1	34.3	78.9	71.3
labeling	N/A	N/A	N/A	72.8	N/A	N/A

Two different labeling schemes were tested. First, the labeling was done as a postprocessing step. This is denoted by *post-LM* in the result tables. As an alternative the labeling was integrated in the fitness function using (22). The results obtained with this are referred with *w/LM* in the result tables.

The label set used in all of the tasks is determined from the whole data set prior the cross-validation folds. All part occurrences of all the pieces were inspected and the labels covering 90% of all occurrences were used as the label set. The remaining labels were assigned an artificial "MISC" label.

The proposed system was implemented in Matlab with C++ routines for the feature extraction, the segment matching, and the search algorithm. When run on a 1.86-GHz Intel Core2-based PC, the average analysis time of a piece with the postprocessing labeling corresponds approximately to the duration of the piece.

D. Evaluation Metrics

Three different metrics are used in the evaluations: frame pairwise grouping F-measure (also precision and recall rates from which the F-measure is calculated are reported), conditional entropy based measure for over- and under-segmentation, and total portion of frames labeled correctly.

The first measure is also used in [22]. It considers all frame pairs both in the ground truth annotations and in the analysis result. If both frames in a pair have the same group assignment, the pair belongs to the set $\mathbb{F}_A$ in the case on ground truth and to $\mathbb{F}_\mathbb{E}$ in the case of analysis result. The pairwise precision rate is defined as

$$R_P = \frac{|\mathbb{F}_A \cap \mathbb{F}_\mathbb{E}|}{|\mathbb{F}_\mathbb{E}|} \tag{24}$$

the pairwise recall rate as

$$R_R = \frac{|\mathbb{F}_A \cap \mathbb{F}_\mathbb{E}|}{|\mathbb{F}_\mathbb{E}|} \tag{25}$$

and the pairwise F-measure as their harmonic mean

$$F = \frac{2R_P R_R}{R_P + R_R}. \tag{26}$$

In the equations above $|\cdot|$ denotes the cardinality of the set. The pairwise clustering measure is simple, yet effective and seems to provide values that agree quite well with the subjective performance.

The second evaluation measure considers the conditional entropy of the frame sequences labeled with the group information given the other sequence (ground truth versus result). The original entropy-based evaluation measure was proposed in [49], but it was further modified by adding normalization terms to allow more intuitive interpretation of the obtained numerical values in [50]. The resulting evaluation measures are over-segmentation score S_O and under-segmentation score S_U. Due to their complexity the formal definitions of S_O and S_U are omitted here, see [50] instead.

The third evaluation metric is the strictest: it evaluates the absolute analysis performance with musical labels. This is done by comparing the label assigned to each frame in the result and in the ground truth annotations. The evaluation measure is the proportion of correctly recovered frame labels.

E. Annotation Reliability Check

It has been noted in earlier studies, e.g., in [7], that the perception of structure in music varies from person to person; therefore, a small experiment was conducted to obtain an estimate of the theoretically achievable accuracy level. A subset of 30 pieces in the *TUTstructure07* data set was analyzed by both annotators independently. Then one set of annotations was considered as the ground truth while the other was evaluated against it. Despite the small size of the data set, this provides an approximation of the level of "human-like performance."

F. Evaluation Results

Tables I–III show the main evaluation results on the different data sets. When comparing the results of tasks with different segmentation levels, the results suggest that the segment border candidate generation is a crucial step for the overall performance. If there are too many extraneous candidate locations, as the case is in "salted" case, the performance drops. The difference between "salted" and "full" is surprisingly small, suggesting that the border candidate generation is able to recover the candidate locations relatively accurately.

The performance increase from the reference system is statistically significant ($p < 0.05$) in the data sets of *TUTstructure07* and *RWC Pop*, but not in *UPF Beatles*. The performance difference between postprocessing labeling and integrated labeling is not significant when evaluated with pairwise F-measure or with over- and under-segmentation measures. Based on the labeling measure, the improvement with integrated labeling in *TUTstructure07* and *UPF* Beatles data sets is statistically significant, whereas in *RWC Pop* it is not.

TABLE IV
SEGMENT BOUNDARY RETRIEVAL PERFORMANCE (%)

data	system	F	R_P	R_R
TUTstructure07	reference [22]	65.3	70.5	63.3
	full w/LM	55.9	50.7	65.9
UPF Beatles	reference [22]	61.2	60.0	64.6
	full w/LM	55.0	52.1	61.2
RWC Pop	reference [22]	64.5	77.3	56.6
	full w/LM	63.0	71.7	57.8

TABLE V
SEGMENTATION STATISTICS ON THE USED DATA SETS

data		segments	groups	seg. dur. (s)
TUTstructure07	annotation	12.1	6.01	19.6
	reference [22]	10.7	5.11	21.5
	full w/LM	12.1	6.98	20.0
UPF Beatles	annotation	8.57	4.59	19.2
	reference [22]	9.48	4.72	17.4
	full w/LM	10.3	6.21	16.8
RWC Pop	annotation	17.1	9.10	14.7
	reference [22]	12.2	5.06	20.6
	full w/LM	13.4	7.96	19.1

Table IV presents the segment boundary retrieval results for both systems on all data sets. A boundary in the result is judged as a hit if it is within 3 s from the annotated border as suggested in [22] and [28].

More direct analysis of the annotated structures and the obtained results is provided in Table V. The table provides the average number of segments in the pieces in the data sets, the average number of groups, and the average duration of a segment. The reference system groups the generated segments using fewer groups than was annotated, while the proposed system uses extraneous groups. Similar under-grouping behavior of the proposed system can be seen in the statistics for *UPF Beatles*. Both systems under-segment the result in *RWC Pop*. This may be partly because the structures in the data have more and shorter segments.

A detailed analysis on the labeling performance is given in Tables VI–VIII. The values describe for each ground truth label the average amount of its duration that was correctly recovered in the result, e.g., value 50% denotes that, on the average, half of the frames with that label were assigned the same label in the result. The tables present the result on all data sets in percents for the labeling only task and for the full analysis with integrated labeling model. The labels are ordered in descending order by their occurrences, the most frequently occurring on top.

G. Discussion

When comparing the results of different data sets, the differences in the material become visible. The performance of the proposed method measured with the F-measure quite similar in all data sets, but the recall and precision rates differ greatly: in *TUTstructure07* the two are close to each other, in *UPF Beatles* the method over-segments the result, and in *RWC Pop* the result is under-segmented. As the operational parameters were selected based on the *TUTstructure07* data, this suggests that some parameter selection should be done for differing material.

Some of the earlier methods tend to over-segment the result and the segment duration had to be assigned in the method

TABLE VI
PER LABEL RECOVERY ON *TUTSTRUCTURE*07 (%)

label	labeling	full w/LM
chorus	77.9	57.6
verse	64.3	44.2
MISC	26.5	10.1
bridge	48.6	17.9
intro	99.2	77.7
pre-verse	55.0	15.8
outro	99.0	60.2
c	47.0	25.9
solo	8.2	5.0
theme	2.2	0.5
chorus_a	7.1	1.8
a	24.4	11.1
chorus_b	6.8	5.1

TABLE VII
PER LABEL RECOVERY ON *UPF BEATLES* (%)

label	labeling	full w/LM
verse	83.9	38.4
refrain	54.8	26.6
bridge	81.4	34.6
intro	94.6	83.2
MISC	16.9	9.6
outro	99.3	62.5
verses	50.0	27.8
versea	9.1	20.5

TABLE VIII
PER LABEL RECOVERY ON *RWC POP* (%)

label	labeling	full w/LM
chorus a	75.0	41.0
verse a	76.0	45.7
MISC	52.3	22.9
verse b	73.9	25.1
chorus b	73.2	16.6
bridge a	61.5	24.2
intro	100.0	87.7
ending	100.0	61.5
pre-chorus	40.0	8.7
verse c	43.6	6.5

"manually," e.g., the reference method [22]. From this point of view it is encouraging to note how the proposed method is able to locate approximately correct length segments even though there is no explicit information given of the appropriate segment length. However, the segment length accuracy differences between the data sets suggest that some additional information should be utilized to assist determining the correct segment length.

It can be noted from Table I that the human baseline for the performance given by the annotator cross-evaluation is surprisingly low. Closer data analysis revealed that a majority of the differences between the annotators was due to hierarchical level differences. Some differences were also noted when a part occurrences contained variations: one annotator had used the same label for all of the occurrences, while the other had created a new group for the variations. It can be assumed that similar differences would be encountered also with larger population analyzing same pieces.

IV. CONCLUSION

A system for automatic analysis of the sectional form of popular music pieces has been presented. The method creates sev-

eral candidate descriptions of the structure and selects the best by evaluating a fitness function on each of them. The resulting optimization problem is solved with a novel controllably greedy search algorithm. Finally, the segments are assigned with musically meaningful labels.

An important advantage of the proposed fitness measure approach is that it distinguishes the definition of a good structure description from the actual search algorithm. In addition, the fitness function can be defined on a high abstraction level, without committing to specific acoustic features, for example. The system was evaluated on three large data sets with manual annotations and it outperformed a state-of-the-art reference method. Furthermore, assigning musically meaningful labels to the description is possible to some extent with a simple sequence model.

ACKNOWLEDGMENT

The authors would like to thank M. Levy for the assistance on his reference system.

REFERENCES

[1] M. Goto, "A chorus-section detecting method for musical audio signals," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Hong Kong, 2003, pp. 437–440.

[2] T. Jehan, "Creating music by listening," Ph.D. dissertation, Mass. Inst. of Technol., Cambridge, MA, 2005.

[3] V. M. Rao, "Audio compression using repetitive structures in music," M.S. thesis, Univ. of Miami, Miami, FL, 2004.

[4] M. Marolt, "A mid-level melody-based representation for calculating audio similarity," in *Proc. 7th Int. Conf. Music Inf. Retrieval*, Victoria, B.C, Canada, Oct. 2006, pp. 280–285.

[5] E. Gómez, B. Ong, and P. Herrera, "Automatic tonal analysis from music summaries for version identification," in *Proc. 12st Audio Eng. Soc. Conv.*, San Francisco, CA, Oct. 2006.

[6] T. Zhang and R. Samadani, "Automatic generation of music thumbnails," in *Proc. IEEE Int. Conf. Multimedia Expo*, Beijing, China, Jul. 2007, pp. 228–231.

[7] M. J. Bruderer, M. McKinney, and A. Kohlrausch, "Structural boundary perception in popular music," in *Proc. 7th Int. Conf. Music Inf. Retrieval*, Victoria, BC, Canada, Oct. 2006, pp. 198–201.

[8] J.-P. Eckmann, S. O. Kamphorst, and D. Ruelle, "Recurrence plots of dynamical systems," *Europhys. Lett.*, vol. 4, no. 9, pp. 973–977, Nov. 1987.

[9] J. Foote, "Visualizing music and audio using self-similarity," in *Proc. ACM Multimedia*, Orlando, Fl, 1999, pp. 77–80.

[10] G. Peeters, "Deriving musical structure from signal analysis for music audio summary generation: Sequence and state approach," in *Lecture Notes in Computer Science*. New York: Springer-Verlag, 2004, vol. 2771, pp. 143–166.

[11] J. Foote, "Automatic audio segmentation using a measure of audio novelty," in *Proc. IEEE Int. Conf. Multimedia Expo*, New York, Aug. 2000, pp. 452–455.

[12] M. Cooper and J. Foote, "Summarizing popular music via structural similarity analysis," in *Proc. 2003 IEEE Workshop Applicat. Signal Process. Audio Acoust.*, New Platz, NY, Oct. 2003, pp. 127–130.

[13] J. Paulus and A. Klapuri, "Music structure analysis by finding repeated parts," in *Proc. 1st ACM Audio Music Comput. Multimedia Workshop*, Santa Barbara, CA, Oct. 2006, pp. 59–68.

[14] K. Jensen, "Multiple scale music segmentation using rhythm, timbre, and harmony," *EURASIP J. Adv. Signal Process.*, 2007, article ID 73205.

[15] M. M. Goodwin and J. Laroche, "A dynamic programming approach to audio segmentation and music/speech discrimination," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Montreal, QC, Canada, May 2004, pp. 309–312.

[16] C. Xu, X. Shao, N. C. Maddage, M. S. Kankanhalli, and T. Qi, "Automatically summarize musical audio using adaptive clustering," in *Proc. IEEE Int. Conf. Multimedia Expo*, Taipei, Taiwan, Jun. 2004.

[17] B. Logan and S. Chu, "Music summarization using key phrases," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Istanbul, Turkey, Jun. 2000, pp. 749–752.

[18] J.-J. Aucouturier and M. Sandler, "Segmentation of musical signals using hidden Markov models," in *Proc. 110th Audio Eng. Soc. Conv.*, Amsterdam, The Netherlands, May 2001.

[19] S. Gao, N. C. Maddage, and C.-H. Lee, "A hidden Markov model based approach to music segmentation and identification," in *Proc. 4th Pacific Rim Conf. Multimedia*, Singapore, Dec. 2003, pp. 1576–1580.

[20] S. Abdallah, M. Sandler, C. Rhodes, and M. Casey, "Using duration models to reduce fragmentation in audio segmentation," *Mach. Lear.*, vol. 65, no. 2–3, pp. 485–515, Dec. 2006.

[21] M. Levy, M. Sandler, and M. Casey, "Extraction of high-level musical structure from audio data and its application to thumbnail generation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Toulouse, France, May 2006, pp. 13–16.

[22] M. Levy and M. Sandler, "Structural segmentation of musical audio by constrained clustering," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 16, no. 2, pp. 318–326, Feb. 2008.

[23] M. A. Bartsch and G. H. Wakefield, "Audio thumbnailing of popular music using chroma-based representations," *IEEE Trans. Multimedia*, vol. 7, pp. 96–104, Feb. 2005.

[24] A. Eronen, "Chorus detection with combined use of MFCC and chroma features and image processing filters," in *Proc. 10th Int. Conf. Digital Audio Effects*, Bordeaux, France, Sep. 2007, pp. 229–236.

[25] L. Lu and H.-J. Zhang, "Automated extraction of music snippets," in *Proc. ACM Multimedia*, Berkeley, CA, Nov. 2003, pp. 140–147.

[26] R. B. Dannenberg and N. Hu, "Pattern discovery techniques for music audio," in *Proc. 3rd Int. Conf. Music Inf. Retrieval*, Paris, France, Oct. 2002, pp. 63–70.

[27] W. Chai, "Automated analysis of musical structure," Ph.D. dissertation, Mass. Inst. of Technol., Cambridge, MA, 2005.

[28] B. S. Ong, "Structural analysis and segmentation of musical signals," Ph.D. dissertation, UPF, Barcelona, Spain, 2006.

[29] G. Peeters, "Sequence representation of music structure using higher-order similarity matrix and maximum-likelihood approach," in *Proc. 8th Int. Conf. Music Inf. Retrieval*, Vienna, Austria, Sep. 2007, pp. 35–40.

[30] M. Müller and F. Kurth, "Towards structural analysis of audio recordings in the presence of musical variations," *EURASIP J. Adv. Signal Process.*, 2007, article ID 89686.

[31] C. Rhodes and M. Casey, "Algorithms for determining and labeling approximate hierarchical self-similarity," in *Proc. 8th Int. Conf. Music Inf. Retrieval*, Vienna, Austria, Sep. 2007, pp. 41–46.

[32] Y. Shiu, H. Jeong, and C.-C. J. Kuo, "Similarity matrix processing for music structure analysis," in *Proc. 1st ACM Audio Music Comput. Multimedia Workshop*, Santa Barbara, CA, Oct. 2006, pp. 69–76.

[33] N. C. Maddage, "Automatic structure detection for popular music," *IEEE Multimedia*, vol. 13, no. 1, pp. 65–77, Jan. 2006.

[34] E. Peiszer, "Automatic Audio Segmentation: Segment Boundary and Structure Detection in Popular Music," M.S. thesis, Vienna Univ. of Technol., Vienna, Austria, 2007.

[35] J. Paulus and A. Klapuri, "Labelling the structural parts of a music piece with Markov models," in *Proc. Comput. in Music Modeling and Retrieval Conf.*, Copenhagen, Denmark, May 2008, pp. 137–147.

[36] J. Paulus and A. Klapuri, "Acoustic features for music piece structure analysis," in *Proc. 11th Int. Conf. Digital Audio Effects*, Espoo, Finland, Sep. 2008, pp. 309–312.

[37] J. Paulus and A. Klapuri, "Music structure analysis with probabilistically motivated cost function with integrated musicological model," in *Proc. 9th Int. Conf. Music Information Retrieval*, Philadelphia, PA, Sep. 2008, pp. 369–374.

[38] D. Turnbull, G. Lanckriet, E. Pampalk, and M. Goto, "A supervised approach for detecting boundaries in music using difference features and boosting," in *Proc. 8th Int. Conf. Music Inf. Retrieval*, Vienna, Austria, Sep. 2007, pp. 51–54.

[39] A. Klapuri, A. Eronen, and J. Astola, "Analysis of the meter of acoustic musical signals," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 14, no. 1, pp. 342–355, Jan. 2006.

[40] M. P. Ryynänen and A. P. Klapuri, "Automatic transcription of melody, bass line, and chords in polyphonic music," *Comput. Music J.*, vol. 32, no. 3, pp. 72–86, 2008.

[41] M. Müller and M. Clausen, "Transposition-invariant self-similarity matrices," in *Proc. 8th Int. Conf. Music Inf. Retrieval*, Vienna, Austria, Sep. 2007, pp. 47–50.

[42] J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," in *Advances in Large Margin Classifiers*, A. J. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, Eds. Cambridge, MA: MIT Press, 1999.

[43] S. J. Young, N. H. Russell, and J. H. S. Thornton, "Token passing: A simple conceptual model for connected speech recognition systems," Cambridge Univ. Eng. Dept., Cambridge, U.K., 1989, Tech. Rep. CUED/F-INFENG/TR38.

[44] G. Boutard, S. Goldszmidt, and G. Peeters, "Browsing inside a music track, the experimentation case study," in *Proc. 1st Workshop Learn. Semantics of Audio Signals*, Athens, Greece, Dec. 2006, pp. 87–94.

[45] A. W. Pollack, "Notes on… series," The Official rec.music.beatles Home Page, 1989–2001. [Online]. Available: http://www.recmusicbeatles.com

[46] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC music database: Popular, classical, and jazz music databases," in *Proc. 3rd Int. Conf. Music Inf. Retrieval*, Paris, France, Oct. 2002, pp. 287–288.

[47] M. Goto, "AIST annotation for the RWC music database," in *Proc. 7th Int. Conf. Music Inf. Retrieval*, Victoria, BC, Canada, Oct. 2006, pp. 359–360.

[48] M. Casey, "General sound classification and similarity," *MPEG-7, Organized Sound*, vol. 6, no. 2, pp. 153–164, 2001.

[49] S. Abdallah, K. Nolad, M. Sandler, M. Casey, and C. Rhodes, "Theory and evaluation of a Bayesian music structure extractor," in *Proc. 6th Int. Conf. Music Inf. Retrieval*, London, U.K., Sep. 2005.

[50] H. Lukashevich, "Towards quantitative measures of evaluating song segmentation," in *Proc. 9th Int. Conf. Music Inf. Retrieval*, Philadelphia, PA, Sep. 2008, pp. 375–380.

Jouni Paulus (S'06) received the M.Sc. degree from the Tampere University of Technology (TUT), Tampere, Finland, in 2002. He is currently pursuing a postgraduate degree at the Department of Signal Processing, TUT.

He has been as a Researcher at TUT since 2002. His research interests include signal processing methods and machine learning for music content analysis, especially automatic transcription of drums and music structure analysis.

Anssi Klapuri (M'06) received the M.Sc. and Ph.D. degrees from the Tampere University of Technology (TUT), Tampere, Finland, in 1998 and 2004, respectively.

In 2005, he spent six months at the Ecole Centrale de Lille, Lille, France, working on music signal processing. In 2006, he spent three months visiting the Signal Processing Laboratory, Cambridge University, Cambridge, U.K. He is currently a Professor at the Department of Signal Processing, TUT. His research interests include audio signal processing, auditory modeling, and machine learning.