Synthesizing Composite Hierarchical Structure from Symbolic Music Corpora

Ilana Shapiro[1] , **Ruanqianqian (Lisa) Huang**[1] , **Zachary Novack**[1] , **Cheng-i Wang**[2] , **Hao-Wen Dong**[1] , **Taylor Berg-Kirkpatrick**[1] , **Shlomo Dubnov**[1] and **Sorin Lerner**[1]

[1]University of California, San Diego, CA, USA
[2]AudioShake, Oakland, CA, USA
{ilshapiro, r6huang, znovack, h3dong, sdubnov, tberg, lerner}@ucsd.edu, {chw160}@audioshake.ai

Abstract

Western music is an innately hierarchical system of interacting levels of structure, from fine-grained melody to high-level form. In order to analyze music compositions holistically and at multiple granularities, we propose a unified, hierarchical meta-representation of musical structure called the *structural temporal graph* (STG). For a single piece, the STG is a data structure that defines a hierarchy of progressively finer structural musical features and the temporal relationships between them. We use the STG to enable a novel approach for deriving a representative structural summary of a music corpus, which we formalize as a nested NP-hard combinatorial optimization problem extending the Generalized Median Graph problem. Our approach first applies simulated annealing to develop a measure of *structural distance* between two music pieces rooted in graph isomorphism. Our approach then combines the formal guarantees of SMT solvers with nested simulated annealing over structural distances to produce a structurally sound, representative *centroid* STG for an entire corpus of STGs from individual pieces. To evaluate our approach, we conduct experiments verifying that structural distance accurately differentiates between music pieces, and that derived centroids accurately structurally characterize their corpora.

1 Introduction

A prevailing theory among Western music theorists and musicologists states that Western classical music exhibits an implicitly hierarchical structure [Simonetta *et al.*, 2018]. While several different theoretical systems have been proposed to formalize this structural hierarchy [Marsden *et al.*, 2013], a widely accepted modern interpretation of the hierarchy states that melodies form the bottom, followed by harmonic contour, rhythmic patterns, disjoint and possibly overlapping motifs, and finally contiguous sections [Nieto, 2015; Mount, 2020; Dai *et al.*, 2024]. Together, this composite hierarchy encapsulates the overall structure of a piece.

To analyze musical structure computationally, many automatic approaches have been developed for extracting structure at *single* levels of the structural hierarchy [Hsiao *et al.*, 2023; Levé *et al.*, 2011; Chen and Su, 2021; Salamon *et al.*, 2014], including methods to analyze sub-hierarchies within a single level [McFee *et al.*, 2017]. However, music perception researchers have shown that the levels are not perceptually independent: they relate to one another in both "vertical" (structural) and "horizontal" (temporal) directions [Narmour, 1983], interactions that have more recently been proven computationally [Dai *et al.*, 2024]. A comprehensive analysis of a piece thus must integrate these inter- and intra-level interactions into a unified model. Furthermore, while attempts have been made to *generate* structured music [Young *et al.*, 2022; Huang *et al.*, 2023; Wang *et al.*, 2024], to our knowledge, no existing research has examined generating music adhering to a complete, musically representative structural hierarchy, since no mechanism to computationally derive a complete structural specification from a desired music corpus currently exists. Such an encapsulation of music structure could also play a critical role in the generation of well-formed music by serving as a system of constraints on generative models.

Despite prior attempts at such an integrated, computational model of musical form, two challenges remain. First, prior approaches do not completely encapsulate the vertical and horizontal relationships of the structural hierarchy, cannot handle polyphonic music, or are not fully automatic [Hamanaka *et al.*, 2016; Simonetta *et al.*, 2018; Mokbel *et al.*, 2009; Carvalho and Bernardes, 2020]. Second, to our knowledge, existing methodologies focus only on *individual* pieces with no attempt to *summarize* the hierarchy over a music corpus to synthesize its overall structure and obtain a holistic representation of the entirety of the corpus.

To address these challenges, we introduce the *structural temporal graph* (STG) as a unified model of complete musical structure. The STG is a k-partite directed acyclic graph whose levels form the structural hierarchy, and edges encode temporal relationships between adjacent levels. We use simulated annealing to develop a measure of *structural distance* between two STGs based on graph isomorphism, and to obtain the overarching structure of a corpus of pieces, we develop an approach to derive a representative *centroid* graph from a corpus of STGs. We formalize centroid derivation as a nested NP-hard combinatorial optimization problem extending the Generalized Median Graph problem [Jiang *et al.*, 2001], and propose a solution combining nested simulated annealing with the formal guarantees of SMT solvers to produce a structurally sound result. Our experiments show that structural distance accurately differentiates pieces, with its performance reliant on the complete hierarchy, and that derived centroids accurately structurally characterize their corpora.[1]

In summary, the contributions of this paper are as follows:

1. We propose the *structural temporal graph*, a meta-representation of musical form unifying the entire structural hierarchy, and develop a *structural distance* measure between two STGs rooted in graph isomorphism.
2. We formalize the music summarization problem as a nested NP-hard combinatorial optimization problem, and contribute a novel solution using both stochastic and SMT-based techniques.
3. We conduct experiments verifying structural distance accurately differentiate pieces, and music corpora are accurately characterized by their derived centroids.

2 Related Work

Single-Level Analyses. Many existing algorithms extract structure at *single* levels of the music structural hierarchy. To extract segmentation, The Music Structure Analysis Framework (MSAF) toolkit [Nieto, 2015] features factorization-based techniques, including ordinal linear discriminant analysis [McFee and Ellis, 2014b], convex nonnegative matrix factorization [Nieto and Jehan,], checkerboard [Foote, 2000], spectral clustering [McFee and Ellis, 2014a], the Structural Features algorithm [Serrà *et al.*, 2014], 2D-Fourier Magnitude Coefficients [Nieto and Bello, 2014], and the Variable Markov Oracle [Wang and Mysore, 2016]. Motif discovery algorithms search for disjoint, repeating, and possibly overlapping patterns in a piece. String-based approaches [Wang *et al.*, 2015] represent music as a chromagram and detect patterns with sub-string matching, and geometry-based approaches [Hsiao *et al.*, 2023] represent music as multidimensional point sets, and translatable subsets identify patterns. Recent approaches in harmony identification are centered around neural networks, such as using transformers to incorporate chord segmentation into the recognition process [Chen and Su, 2019; Chen and Su, 2021]. Until very recently, the Melodia algorithm was the state of the art in melody extraction, but recent approaches have shifted to neural networks [Kosta *et al.*, 2022; Chou *et al.*, 2021].

Integrated Models of Structure. Music theorists have attempted to unify the structural hierarchy with frameworks such as Schenkerian theory [Marsden, 2010] and the Generative Theory of Tonal Music (GTTM) [Lerdahl and Jackendoff, 2020]. Schenkerian analysis applies a series of reductions that progressively simplify a musical piece by removing layers of structure. Attempts to automatically derive Scherkerian analyses are intractable for all but very short pieces, and have low accuracy [Marsden, 2010]. GTTM generates four different structural hierarchies (grouping structure, metrical structure, time-span tree, and prolongational tree) for a piece of music, to model human cognition [Hamanaka *et al.*, 2016]. Computational implementations GTTM (e.g. the Automatic Timespan Tree Analyser [Hamanaka *et al.*, 2016]) cannot handle polyphonic music, and are not fully automatic. Improved results with these theories are unlikely, as neither gives the precision required for complete computational implementation [Marsden *et al.*, 2013].

Such theoretical limitations led to a modern interpretation of the structural hierarchy: segmentation, motifs (disjoint/repeating patterns), rhythm, harmony, and melody [Nieto, 2015; Mount, 2020; Dai *et al.*, 2024]. Many approaches partially encode the hierarchy in graphs: topographic mappings for melodic progressions [Mokbel *et al.*, 2009], graphs for interactions between sections, melody, harmony and rhythm [Dai *et al.*, 2020], multi-edge graphs for bar-level relations [Bhandari and Colton, 2024], and undirected graphs for melodies and their reductions [Orio and Rodà, 2009]. The prototype graph [Young *et al.*, 2022] is a bipartite network relating prototype elements to the music they represent. Attempts to model the structural hierarchy with formal grammars [Sidorov *et al.*, 2014; Finkensiep *et al.*,] are limited to segmentation and motifs.

None of these approaches encapsulate the entire hierarchy, and to our knowledge, there have also been no attempts to synthesize representative structure from a music corpus.

3 Structural Temporal Graph

To address the lack of a fully automatic complete encapsulation of polyphonic musical structure, we introduce the *structural temporal graph* (STG), a unified meta-representation of musical structure that captures the levels of the music structural hierarchy and the temporal relationships between them. The STG is a k-partite directed acyclic graph (DAG), where each of the k layers encodes a level in the music structural hierarchy.[2] Following the modern music theoretic interpretation of the hierarchy [Nieto, 2015; Mount, 2020; Dai *et al.*, 2024], from top to bottom we denote the levels to be contiguous segmentation, motifs (both disjoint and overlapping), rhythmic contour, relative keys, functional harmonic chords, and melodic contour. The STGs we build include every level in the hierarchy except rhythmic contour, for which we were unable to access an analysis algorithm. We run individual analysis algorithms to generate each level of the hierarchy, which is elucidated in Section 6. Before formally defining the STG, we build intuition by walking through the derivation of an STG from an annotated piece.

Building the Graph. We walk through the derivation of an STG from Beethoven's Biamonti Sketch No. 461 that unifies contiguous segmentation; disjoint, overlapping motifs; relative keys; functional harmonic chords; and melodic contour. First, we manually analyze the piece by annotating its score with computer-generated hierarchical structure analyses in Figure 1. Each colored annotation corresponds to one level of the structural hierarchy. In purple, we see that this piece has one large contiguous segment, labeled 0. Next, disjoint motifs are in red. Motif 0 appears twice, at the beginning of bars 1 and 2. The gray filler bar indicates no more

[1] Paper code: https://github.com/ilanashapiro/stg_optimization

[2] Individual levels themselves can form sub-hierarchies of increasing granularity, which the STG supports

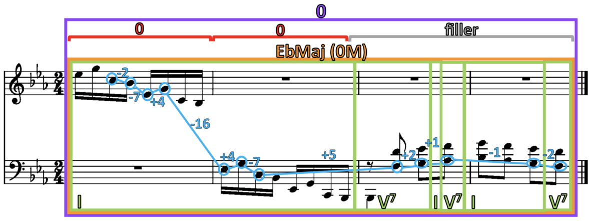

Figure 1: Computer generated analysis of Beethoven's Biamonti Sketch No. 461.

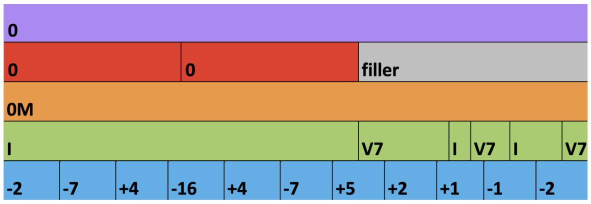

Figure 2: Spatial Visualization of Figure 1

motifs appear in the latter half. In orange, we see this piece is in a single key, Eb Major. We label this key symbolically as 0M to indicate this is relative key number 0 (i.e. the first key) in M for major. Subsequent keys would be numbered by their positive interval difference from the previous key within the 12-tone scale. We next see functional harmonic chords in green annotated with Roman numeral chord symbols, and finally melodic contour intervals in blue. Notably, such generated analyses may be slightly inaccurate (e.g. the third green I chord should correspond to the previous beat). Since the STG is a fully automatic *meta*-representation of musical structure, it is only as accurate as the analysis algorithms it uses.

We then equivalently represent the ground-truth annotations in Figure 1 as the stacked rectangles in Figure 2 to elucidate how each level of the structural hierarchy relates to the next. Finally, we transition to the STG in Figure 3. There is a surjection between Figures 1, 2, and 3. The edges and nodes of the STG, respectively, correspond to the vertical and horizontal alignments of the rectangles in Figure 2. All motif nodes, including the gray filler node indicating no motifs for that interval, fall into the time interval of purple segmentation node 0. The orange key node 0M *starts* in the first red motif node 0, and *ends* in the last gray motif filler node (i.e. the key spans the entire piece). All the green chord nodes fall in the orange key node's interval. Finally, we see how blue melodic contour nodes relate to green chord nodes. For instance, the first melody interval -2 begins and ends in the first chord node I, and the penultimate melody interval -1 begins in the fourth V7 chord node and ends in the fifth I chord node.

Formally, the nodes of an STG encode labeled musical sections along with their associated time intervals generated by the relevant analysis algorithm. Nodes are sorted within each level based on start time, and edges encode temporal relationships between nodes of adjacent levels. Specifically, for node n at level i, n must have either one or two parents in level $i-1$ directly above it: one if its associated time interval is a total subset of its parent's, and two if its time interval begins in one parent's, and ends in the other's.

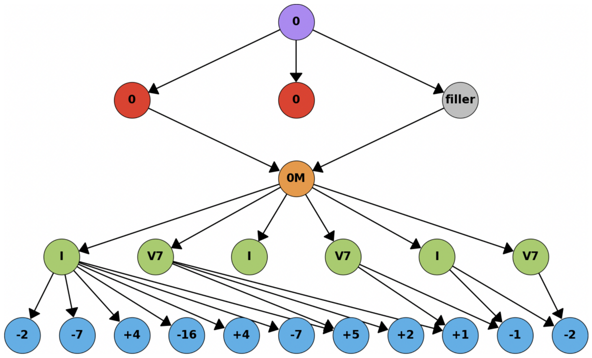

Figure 3: STG for Beethoven's Biamonti Sketch No. 461

4 Structural Distance

At a high level, the distance between two STGs is the minimum number of edit operations (deletion, insertion, and substitution of nodes/edges) required to transform one graph to the other, also known as graph edit distance (GED) [Serratosa, 2021]. However, GED measures *isomorphic* similarity between two graphs, i.e. it evaluates how closely graph structures match independent of labeling. We cannot currently leverage STG isomorphism because STGs are "compressed," with structure encoded in node ids. Specifically, structure is encoded in the defining features of each node id, and in the intra-level linear temporal orderings for each analysis (i.e. the horizontal order of each level in the graph, currently determined by node index). Thus, in order to reason about STGs isomorphically, we must augment them to encode all structural attributes directly within the graph's topology.

To encode element labels, recall that each node id encodes a defining feature set. All nodes can thus be alternatively encoded as *instances* of their feature *prototypes*. We create a *prototype node* for each feature and assign it as a parent of the corresponding instance node(s) with that feature. For instance, segmentation nodes encode a single feature: the section number they correspond to. Finally, to encode intra-level linear temporal relationships, we form a linear chain with edges between pairs of horizontally adjacent nodes. This results in a structurally complete STG we can reason about isomorphically. Figure 4 shows the first two levels of the STG from Figure 3, with yellow prototype nodes on the left for each instance feature (section number for segmentation nodes **S**, and pattern number and filler for motif/pattern nodes **P**), red edges connecting prototype features to instance nodes, and green edges for the pattern layer intra-level linear chain.

4.1 Graph Alignment Annealing

GED is a NP-hard combinatorial optimization problem [Serratosa, 2021], intractable for most STGs. Most GED approximation algorithms are slow and of dubious accuracy, and more generalized than we require [Abu-Aisheh *et al.*, 2015]. We thus introduce a new measure of *structural distance* computed with simulated annealing (SA), a stochastic optimization technique that estimates the global optimum of a discrete cost function. It comprises an objective "energy" function to minimize and a "move" function for generating a new solu-

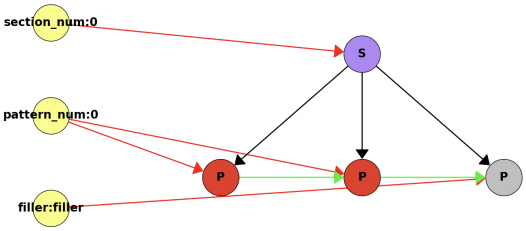

Figure 4: Augmenting the first two levels of the Beethoven STG. Yellow prototype nodes have the format feature_name:feature_value

Algorithm 1 Alignment Annealer Move Function

```
function MOVE
    Choose random index $i$ in $P$
    Choose random index $j$ from the same partition to which $i$ belongs
    Swap rows $i$ and $j$ in $P$
end function
```

tion from the current state. An annealer begins at a high *temperature* indicating the likelihood of accepting worse solutions to explore the solution space, and ends at a low temperature to refine near-optimal solutions [Guilmeau *et al.*, 2021].

To use SA, we convert the augmented STGs to adjacency matrices, and pad each matrix with zero-arity "dummy nodes" so they have identical dimensions. Given such matrices A_{G_1} and A_{G_2}, their distance is given in Equation 1, where $\|.\|_F$ denotes the Frobenius norm. When A_{G_1} and A_{G_2} are optimally aligned, $\text{DIST}(A_{G_1}, A_{G_2})$ is simply $\sqrt{\text{GED}}$.

$$\text{DIST}(A_{G_1}, A_{G_2}) = \|A_{G_1} - A_{G_2}\|_F \tag{1}$$

Finding permutation matrix P optimally aligning A_{G_2} to A_{G_1} to minimize Equation 1 is NP-hard, so we use SA. The alignment annealer's energy function is given in Equation 2. Given optimal P, Equation 2 computes the structural distance between A_{G_1} and A_{G_2}.

$$\text{ENERGY}(A_{G_1}, A_{G_2}, P) = \text{DIST}(A_{G_1}, P^T A_{G_2} P) \tag{2}$$

The move function for modifying P at each step of SA is given in Algorithm 1. A partition is either the set of instance nodes at a single level in the STG (e.g. the set of functional harmonic key nodes), or the set of prototype nodes for a given feature (e.g. chord quality). Permuting only within valid partitions leverages the STG's inherent structure to avoid invalid moves globally detrimental to Equation 2.

We set the alignment annealer's initial state to $P = I$, the identity matrix. By running the annealer for sufficiently many steps, we obtain optimal P.

5 Centroid Derivation

Centroid STG derivation is a constrained version of the Generalized Median Graph (GMG) problem, which, given a set of graphs, seeks to construct a prototype graph minimizing the distances over the input set graphs [Blumenthal *et al.*, 2021]. Formally, given a corpus of graphs C, the GMG g is:

$$g = \arg\min_{\hat{g}} \sum_{G \in C} d(\hat{g}, G) \tag{3}$$

1. No self-loops
2. No instance-prototype or prototype-prototype edges
3. No edges from a prototype to an instance whose feature set does not include the proto feature (e.g. melody interval sign proto-segmentation instance)
4. No edges from lower to higher level instance levels (must respect the hierarchy)
5. No edges between non-adjacent instance levels (must respect k-partite structure)

Table 1: Global Constraints

where d is the distance measure. GMGs have wide applications in representation-based learning, particularly in biological settings [Mukherjee *et al.*, 2009]. Prior attempts at the GMG use genetic search [Jiang *et al.*, 2001], linear programming[Mukherjee *et al.*, 2009], block coordinate updates [Blumenthal *et al.*, 2021], and median graph shift clustering [Jouili *et al.*, 2010]. To our knowledge, GMGs have never been applied to the music domain. Centroid STG derivation also operates in a significantly more constrained search space than the canonical GMG, since the centroid must be a well-formed STG. We thus propose a new approach combining nested simulated annealing with SMT solving.

5.1 Bi-Level Simulated Annealing

Given padded adjacency matrix A_g for an augmented centroid STG and its associated corpus of matrices $C = \{A_G\}$ that are optimally aligned to A_g, we seek to minimize LOSS in Equation 4, where DIST is as in Equation 1.

$$\text{LOSS}(A_g, C) = \frac{1}{|C|} \sum_{A_G \in C} \text{DIST}(A_g, A_G) \tag{4}$$

The centroid annealer's energy function is given in Equation 5, where C_{aligned} is the corpus aligned to the current centroid A_g, a process itself requiring SA as in Section 4.1 to obtain the optimal alignments. Finding the centroid A_g minimizing Equation 4 is thus a nested NP-hard problem (GED and minimizing over these distances) requiring nested SA.

$$\text{ENERGY}(A_{G_1}, A_{G_2}, P) = \text{LOSS}(A_g, C_{\text{aligned}}) \tag{5}$$

As the centroid annealer's temperature cools, the loss converges as the centroid is increasingly closely aligned to its corpus. Thus, as the *centroid* annealer's temperature cools, we scale down the number of steps and max temperature of the nested *alignment* annealer.[3]

The centroid annealer's move function for modifying the centroid A_g at each step of SA is given in Algorithm 2. To move strategically, we build the score matrix S revealing which edge(s) in A_g contribute most to the loss. We add or remove the edge at a highest score coordinate meeting the criterion in Algorithm 2. In particular, a globally structurally invalid move induces a terminally invalid structure in the centroid by violating one of the rules in Table 1. Some locally invalid moves, however, such as removing an edge in an intra-level linear chain, must be allowed as intermediate steps to a

[3] See Appendix A on arXiv (https://arxiv.org/abs/2502.15849) for precise cooling schedule

Algorithm 2 Centroid Annealer Move Function

```
function MOVE
    Calculate the list of absolute difference matrices $D_L = [|A_g - A_{G_i}|$ for $A_{G_i} \in C_{aligned}]$
    Calculate the element-wise sum-of-distances score matrix. Higher score at coord $(i, j)$ means that coord has a higher impact on the loss
    Flatten $X$ and sort in descending score order
    Partition $X_{\text{flat}}$ by unique score, and shuffle each partition randomly (increases variability of moves)
    Iterate through the indices $(i, j)$ of the sorted, partition-shuffled $X_{\text{flat}}$. Stop at the first (highest score) $(i, j)$ such that flipping the $(i, j)$ edge in $A_g$ is not:
        • a globally structurally invalid move
        • a move undoing the most recently accepted move (avoid oscillating states)
        • a move the annealer has already locally rejected since the last accept (avoid getting stuck)
    Execute move: $A_g[i, j] = 1 - A_g[i, j]$
end function
```

more optimal structurally valid state. Importantly, the STGs being compared must have the same number of levels; otherwise, edges spanning multiple levels must be allowed as they can be intermediate states towards the deletion of an entire level. Based on our experiments, this would be unacceptably detrimental to the performance of the annealer.

We set the centroid annealer's initial state A_g to the corpus STG in the corpus with the min loss over the rest of the corpus. Running the annealer for sufficient steps gives an approximate centroid that may contain locally invalid states.

5.2 Graph Repair with SMT Solving

In order to obtain a structurally sound centroid, we must "repair" the approximate centroid A_g by projecting it to the nearest valid STG. We achieve this by encoding the STG's structure as constraints in quantifier-free first-order logic formulae in the SMT (satisfiable modulo theory) solver Z3 [De Moura and Bjørner, 2008], which gives us formal guarantees on the soundness of the centroid. We use Z3's optimizer to minimize an objective over the constraints. Given approximate centroid A_g and valid centroid A'_g, our objective is Equation 6.

$$\text{OBJ}(A_g, A'_g) = \sum_i \sum_j |A_{g_{ij}} - A'_{g_{ij}}| \tag{6}$$

Our constraints include the global rules in Table 1, as well as additional constraints for instance nodes in Table 2, and finally prototype nodes in Table 3. We model relationships between nodes with uninterpreted functions.

1. Every instance node must have 1 or 2 instance parents in the level above
2. The instances nodes at level l must form a linear chain/total ordering via intra-level edges
3. The start and end nodes of the linear chain must have the previous level linear chain's start and end nodes, respectively, as parents.
4. In instance levels with non-overlapping nodes,[4] the *first* parent of a node at linear chain index $i > 0$, must not come before node $i - 1$'s *last* parent in the previous instance level's linear chain
5. The *first* parent of an instance node at linear chain index $i > 0$, must not come before node $i - 1$'s *first* parent in the previous level's linear chain

Table 2: Instance Constraints

1. Every instance node must have exactly one prototype parent per feature
2. For levels that require it,[5] no two linearly adjacent instance nodes can have identical prototype parent sets

Table 3: Prototype Constraints

Z3's optimizer supports integration with large neighborhood search (LNS) and can return intermediate semi-optimized solutions after a timeout. We run the optimizer with LNS, with initial soft constraints set to the approximate centroid A_g to guide the optimizer. Even so, naively running the optimizer on a full STG is generally intractable due to combinatorial explosion, so we partition A_g into subsets we can apply the constraints to incrementally.

We first partition A_g into pairs of consecutive instance levels without their prototypes (e.g. a segmentation/motif pair of instance levels), and optimize the instance constraints in Table 2 and relevant global constraints in Table 1 over each partition incrementally. We combine the results of each partition until we build a valid centroid subgraph of instance nodes. Then, we partition A_g into single levels, each containing the instance nodes of that level and all prototype nodes for each instance feature at that level (e.g. segmentation instance nodes + section number prototype nodes). The instance constraints are already optimized; we need only optimize the prototype constraints in Table 2 and relevant global constraints in Table 1 over the possible prototypes. This gives us the complete, structurally sound centroid A'_g we seek.[6]

6 Experiments

We conduct experiments to verify that: (1) structural distance accurately differentiates individual pieces, with its performance reliant on the complete hierarchy, and (2) the centroid encapsulates the overarching structure of its corpus.

To evaluate our approach, we create a dataset of polyphonic, symbolic MIDI piano music from the Kunstderfuge and Classical Piano MIDI datasets.Since some single-level analyses we use to generate STGs require the data to be in audio and CSV format, we convert MIDI to CSV (with a manual script) and to MP3 (with Fluidsynth).

We then generate an STG for each piece. For segmentation, we use the flat Structural Features algorithm [Serrà *et al.*, 2014] for segment boundaries and 2D-Fourier Magnitude Co-

[4] Segmentation, keys, chords, and melody, but not motifs

[5] Segmentation, keys, and chords only

[6] See Appendix B on arXiv for centroid derivation visualization

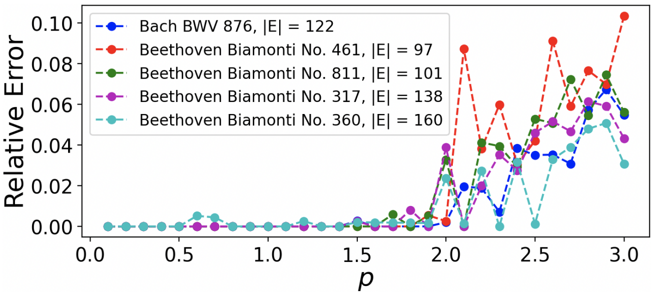

Figure 5: Relative Error: Computed vs Ground-Truth Structural Dist

efficients [Nieto and Bello, 2014] for segment labels, both of which are provided by the Music Structure Analysis Framework [Nieto, 2015]. For motifs, we use the BPS-motif discovery algorithm [Hsiao *et al.*, 2023], and for relative keys and functional harmonic chords we use the pretrained Harmony Transformer V2 [Chen and Su, 2021]. Finally, for melodic contour, we use the Melodia algorithm [Salamon *et al.*, 2014].

6.1 Structural Distance Evaluation

Mathematical Verification. Given two STGs G_1, G_2 recall that under optimal alignment, $\textsc{dist}(A_{G_1}, A_{G_2})$ is $\sqrt{\text{GED}}$. To evaluate our alignment annealer, we set G_1 with $|E_1|$ edges as the "base graph." From G_1, we generate a series of STGs G_2 by randomly adding $\lceil |E_1| \cdot p \rceil$ valid edits to G_1 (add/remove edge, verified with Section 5.2 Z3 solver), where $p \in \{0.1, 0.2, \dots, 3\}$ (i.e. p ranges from 10-300% edits in the size of $|E_1|$). We evaluate structural distance as a function of p for five base STGs G_1 by computing the relative error from experimental $\textsc{dist}(A_{G_1}, A_{G_2})$ to ground truth structural distance $\sqrt{\lceil |E_1| \cdot p \rceil}$ (Figure 5). Relative error is close to 0 (perfect alignment) for $p < 1.8$, and only deteriorates, at worst, to 10.33% at $p = 3$ for the Beethoven 461 base graph.

Musical Evaluation. To verify structural distance accurately differentiates pieces, we construct 210 STG sets from 32 pieces by J.S. Bach (21), Mozart (2), Beethoven (3), Schubert (2), and Chopin (4). Each set is a unique combination of 5 pieces, one from each composer, such that a piece's duration is within 7 seconds of any other piece in the set, since structural distance between disparate length pieces could be due to STG size differences rather than the local structural variations we aim to distinguish.[7] We compute pairwise structural distances between the STGs in each set with our Section 4 graph alignment annealer (2000 iterations, max and min temperature of 2 and 0.01) on 8 Nvidia RTX 2080 GPUs with 11GB RAM. This results in 210 structural distance matrices, which we average to a single mean distance matrix.

We evaluate structural distance against three baselines over the same 210 piece combinations. Baseline 1 is the mean distance matrix obtained by taking the cosine similarity between feature vectors extracted from each MIDI file using Music21. Baseline 2 is the mean distance matrix over pairwise Stent weighted audio similarities (SWAS) for each piece's paired MP3 file. SWAS is a composite audio similarity metric comprising zero-crossing rate, rhythm, chroma, spectral contrast, and perceptual similarity metrics, which we weigh equally. To demonstrate existing graph comparison metrics are insufficient, Baseline 3 is the Weisfeiler-Lehman (WL) Kernel applied pairwise to the STGs in each set, with five iterations and Neighborhood Hash (NH) as the base kernel. The WL Kernel iteratively refines node labels based on their neighbors, and uses a base kernel (in this case, NH, which efficiently captures local graph structure by distinguishing neighborhood configurations) to compare them [Shervashidze *et al.*, 2011]. Our ground-truth is the stylistic similarity indices between composers using human annotations and metadata from "The Classical Music Navigator" [Smith and Georges, 2014].

Metric	ρ_s	p-value
Ours: SD	**0.8207**	**0.0130**
Baseline 1: MIDI Features	0.4681	0.3150
Baseline 2: SWAS	0.5775	0.1690
Baseline 3: WL Kernel (+NH base)	-0.8389	0.0110

Table 4: Mantel Test with Spearman's rank correlation coefficient for normalized mean distance matrices

Metric	ρ_s	p-value
SD - complete STG (5 levels)	**0.8207**	**0.0130**
SD - 4 levels	0.7842	0.0390
SD - 3 levels	0.7173	0.0680
SD - 2 levels	0.6930	0.1150
SD - 1 level	-0.4377	0.2810

Table 5: Mantel Test with Spearman's rank correlation coefficient for normalized mean distance matrices with STG level ablations (first row is same as Table 4)

We normalize all matrices to [0, 1] range and apply the Mantel test with Spearman's rank correlation coefficient ρ_s to evaluate our results against the ground-truth similarity from the Classical Music Navigator (Table 4). ρ_s is highest (with $p < 0.05$) for structural distance (SD in Table 4), verifying that structural distance accurately differentiates between pieces and captures human conception of musical similarity.

Finally, to verify the importance of the full hierarchy, we repeat this experiment with bottom-up level ablations of the STG (Table 5). The structural distance algorithm's performance incrementally declines each time a layer of the STG is removed, confirming the necessity of the complete hierarchy.

6.2 Centroid Evaluation

Mathematical Verification. As a correctness check, we evaluate derived centroids against constructed ground truth centroids. Given base STG g with $|E|$ edges, we create a synthetic corpus C_k of k STGs by randomly adding $\lceil \frac{|E|}{2} \rceil$ valid edits to g, k times, as in Section 6.1. By construction, g must be a true centroid with min possible loss over C_k (Equation 4).[8] We derive candidate centroid g_d from C_k, and compare to g. We evaluate g_d against the naive centroid g_n, *i.e.,* the STG already in C_k with min loss over the rest of C_k.

[7] See Appendix C on arXiv for input pieces/sets details

[8] Proof in Appendix D on arXiv

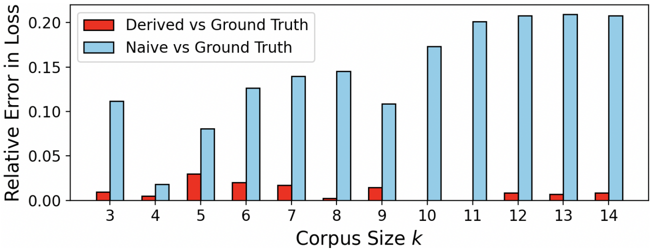

Figure 6: Relative Error in Loss ([0,1] range): Derived and Naive Centroids vs Ground-Truth Centroid

We choose our base STG g to be Beethoven's Biamonti Sketch No. 461, which, when augmented, has $|E| = 97$ edges. Thus, we add $\lceil \frac{97}{2} \rceil = 49$ edits to g, k times, to generate a synthetic corpus C_k of size k with g as its true centroid. 49 edits is structural distance of 7.0, since structural distance between optimally aligned graphs is $\sqrt{\text{GED}}$ (Equation 1).

We repeat this process for $k \in [3, 14]$. For each corpus C_k, we use the same GPU infrastructure as Section 6.1 to generate approximate centroids with the centroid annealer (1000 iterations, max and min temperature of 2.5 and 0.05). At each iteration, we run the nested graph alignment annealer, starting at 500 steps, max and min temperature of 1 and 0.01, and ending at 5 steps, max and min temperature of 0.05 and 0.01, as the outer centroid annealer's loss converges (see Section 5.1). We run our Z3 optimizer to generate each final, repaired centroid g_d, using 24 Intel Xeon cores and 32GB RAM.

There may be multiple non-isomorphic true centroids with the same minimal loss over C_k. Evaluating the relative error between optimal loss from g and experimental loss from g_d thus gives a more rigorous assessment of g_d than directly comparing g_d to g. Figure 6 shows the relative error in loss $E^g_{g_d}$ between g and g_d for $k \in [3, 14]$ in red, compared to the relative error in loss $E^g_{g_n}$ between g and g_n in blue. We observe $E^g_{g_d}$ is consistently small, with max $E^g_{g_d} = 2.99\%$ at $k = 5$, and min $E^g_{g_d} = 0$ (i.e. g_d is a true centroid) at $k = 10, 11$. g_d also greatly outperforms g_n: for nonzero $E^g_{g_d}$, $E^g_{g_n}$ is on average 17.23 times worse than $E^g_{g_d}$.

Musical Evaluation. This experiment requires pieces with more length (and STG size) variation than Section 6.1, as centroids derived from very similar input graphs may be trivial by construction and fail to generalize to more diverse corpora. We thus relax the Section 6.1 relative duration restriction to 80 seconds, and generate centroid STGs for four corpora with pieces by Alkan (11), Chopin (8), Haydn (12), and Mozart (14) with the same architecture as before.[9]

To verify each derived centroid g musically characterizes its corpus C, we use rustworkx [Treinish *et al.*, 2022] to enumerate the set of all 5-node subgraphs ($\{S_5\}$) common to every STG in C, thus extracting the most structurally salient musical relationships in C.[10] For each C, we evaluate the percentage of the graphs in $\{S_5\}$ that are also subgraphs of g; i.e. we evaluate how well g captures the most structurally salient musical relationships in C (Table 6). On average, 70.33% of each $\{S_5\}$ aligns *perfectly* with g, confirming each centroid captures the musically essential information of its corpus. To visualize these essential musical substructures, consider a 5-node subgraph common to the Mozart corpus and captured in its centroid (Figure 7). This reveals that a characterizing feature of the Mozart corpus is two consecutive major chords in the same major key, which in turn falls in a motif/pattern.

Composer	Size of $\{S_5\}$	% in Derived Centroid
Alkan	1804	76.67%
Chopin	2111	70.06%
Haydn	2504	63.90%
Mozart	1900	70.68%

Table 6: Analysis of Common 5-Node Subgraphs $\{S_5\}$

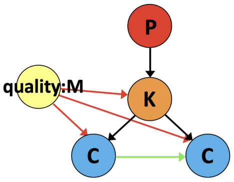

Figure 7: Example Common 5-Node Subgraph of Mozart Corpus

7 Conclusion and Future Work

We presented the *structural temporal graph* (STG) to encapsulate complete, hierarchical musical structure; a measure of *structural distance* between STGs; and an algorithm to derive a *centroid STG* structurally representing a music corpus. We showed structural distance and derived centroids both mathematically approximate ground truth; structural distance accurately differentiates music pieces; and derived centroids capture the essential structural relationships of their corpora.

The STG and derived centroids lay the groundwork for structured, controllable sequence data generation. For example, users could modify the STG of a generated music piece to update constraints on a generative model. Beyond music, the STG can model structural hierarchies for *any* sequence data given algorithms for analyses at each level. For instance, an STG could encode a poetry hierarchy—verses, stanzas, lines—with its centroid structurally summarizing the poetry corpus. Such applications enable human refinement of machine-generated data to meet desired structural specifications across fields.

Acknowledgments

This work is supported by National Science Foundation grants #1955457, #2220892, #2146151, and #2200333, and by the European Research Council under Europe's Horizon 2020 program, grant #883313 (ERC REACH).

References

[Abu-Aisheh *et al.*, 2015] Zeina Abu-Aisheh, Romain Raveaux, Jean-Yves Ramel, and Patrick Martineau. An

[9] See Appendix C on arXiv for input pieces details

[10] Larger subgraphs encode more robust relationships, but mining k-node subgraphs for $k > 5$ was intractable, and approximate subgraph mining tools were too imprecise for meaningful conclusions

exact graph edit distance algorithm for solving pattern recognition problems. In Maria De Marsico, Mário A. T. Figueiredo, and Ana L. N. Fred, editors, *ICPRAM 2015 - Proceedings of the International Conference on Pattern Recognition Applications and Methods, Volume 1, Lisbon, Portugal, 10-12 January, 2015*, pages 271–278. SciTePress, 2015.

[Bhandari and Colton, 2024] Keshav Bhandari and Simon Colton. Motifs, phrases, and beyond: The modelling of structure in symbolic music generation. In *Artificial Intelligence in Music, Sound, Art and Design*, pages 33–51, Cham, 2024. Springer Nature Switzerland.

[Blumenthal *et al.*, 2021] David B. Blumenthal, Nicolas Boria, Sébastien Bougleux, Luc Brun, Johann Gamper, and Benoit Gaüzère. Scalable generalized median graph estimation and its manifold use in bioinformatics, clustering, classification, and indexing. *Information Systems*, 100:101766, 2021.

[Carvalho and Bernardes, 2020] Nádia Carvalho and Gilberto Bernardes. A review of symbolic music representations and their hierarchical modeling. In *Proceedings of the Eleventh International Conference on Computational Creativity, Coimbra, Portugal*, pages 236–242. Association for Computational Creativity (ACC), 2020.

[Chen and Su, 2019] Tsung-Ping Chen and Li Su. Harmony transformer: Incorporating chord segmentation into harmony recognition. In *Proceedings of the 20th International Society for Music Information Retrieval Conference, Delft, The Netherlands*, pages 259–267, 2019.

[Chen and Su, 2021] Tsung-Ping Chen and Li Su. Attend to chords: Improving harmonic analysis of symbolic music using transformer-based models, Feb 2021.

[Chou *et al.*, 2021] Yi-Hui Chou, I-Chun Chen, Chin-Jui Chang, Joann Ching, and Yi-Hsuan Yang. Midibert-piano: Large-scale pre-training for symbolic music understanding, 2021.

[Dai *et al.*, 2020] Shuqi Dai, Huan Zhang, and Roger B. Dannenberg. Automatic analysis and influence of hierarchical structure on melody, rhythm and harmony in popular music. *CoRR*, abs/2010.07518, 2020.

[Dai *et al.*, 2024] Shuqi Dai, Huan Zhang, and Roger B. Dannenberg. The interconnections of music structure, harmony, melody, rhythm, and predictivity. *Music & Science*, 7:20592043241234758, 2024.

[De Moura and Bjørner, 2008] Leonardo De Moura and Nikolaj Bjørner. Z3: an efficient smt solver. In *Proceedings of the Theory and Practice of Software, 14th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, TACAS'08/ETAPS'08, page 337–340, Berlin, Heidelberg, 2008. Springer-Verlag.

[Finkensiep *et al.*,] Christoph Finkensiep, Matthieu Haeberle, Friedrich Eisenbrand, Markus Neuwirth, and Martin Rohrmeier. Repetition-structure inference with formal prototypes. In *Proceedings of the 24th International Society for Music Information Retrieval Conference, Milan, Italy*, pages 383–390.

[Foote, 2000] J. Foote. Automatic audio segmentation using a measure of audio novelty. In *2000 IEEE International Conference on Multimedia and Expo. ICME2000. Proceedings. Latest Advances in the Fast Changing World of Multimedia (Cat. No.00TH8532)*, volume 1, pages 452–455 vol.1, 2000.

[Guilmeau *et al.*, 2021] Thomas Guilmeau, Emilie Chouzenoux, and Víctor Elvira. Simulated annealing: a review and a new scheme. In *IEEE Statistical Signal Processing Workshop, SSP 2021, Rio de Janeiro, Brazil, July 11-14, 2021*, pages 101–105. IEEE, 2021.

[Hamanaka *et al.*, 2016] Masatoshi Hamanaka, Keiji Hirata, and Satoshi Tojo. *Implementing Methods for Analysing Music Based on Lerdahl and Jackendoff's Generative Theory of Tonal Music*, pages 221–249. Springer International Publishing, Cham, 2016.

[Hsiao *et al.*, 2023] Yo-Wei Hsiao, Tzu-Yun Hung, Tsung-Ping Chen, and Li Su. Bps-motif: A dataset for repeated pattern discovery of polyphonic symbolic music. In *Proceedings of the 24th International Society for Music Information Retrieval Conference, Milan, Italy*, pages 281–288, 2023.

[Huang *et al.*, 2023] Wenkai Huang, Yujia Yu, Haizhou Xu, Zhiwen Su, and Yu Wu. Hyperbolic music transformer for structured music generation. *IEEE Access*, 11:26893–26905, 2023.

[Jiang *et al.*, 2001] Xiaoyi Jiang, A. Munger, and H. Bunke. An median graphs: properties, algorithms, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(10):1144–1151, 2001.

[Jouili *et al.*, 2010] Salim Jouili, Salvatore Tabbone, and Vinciane Lacroix. Median graph shift: A new clustering algorithm for graph domain. In *2010 20th International Conference on Pattern Recognition*, pages 950–953, 2010.

[Kosta *et al.*, 2022] Katerina Kosta, Wei Tsung Lu, Gabriele Medeot, and Pierre Chanquion. A deep learning method for melody extraction from a polyphonic symbolic music representation. In *Proceedings of the 23rd International Society for Music Information Retrieval Conference, Bengaluru, India*, pages 757–763, 2022.

[Lerdahl and Jackendoff, 2020] Fred Lerdahl and Ray Jackendoff. *A Generative Theory of Tonal Music*. MIT Press, 2020.

[Levé *et al.*, 2011] Florence Levé, Richard Groult, Guillaume Arnaud, Cyril Séguin, Rémi Gaymay, and Mathieu Giraud. Rhythm extraction from polyphony symbolic music. In *Proceedings of the 12th International Society for Music Information Retrieval Conference, Miami, Florida, USA*, pages 375–380. University of Miami, 2011.

[Marsden *et al.*, 2013] Alan Marsden, Keiji Hirata, and Satoshi Tojo. Towards computable procedures for deriving tree structures in music : context dependency in gttm and schenkerian theory. 2013.

[Marsden, 2010] Alan Marsden. Schenkerian analysis by computer: A proof of concept. *Journal of New Music Research*, 39(3):269–289, 2010.

[McFee and Ellis, 2014a] Brian McFee and Dan Ellis. Analyzing song structure with spectral clustering. In *Proceedings of the 15th International Society for Music Information Retrieval Conference, Taipei, Taiwan*, pages 405–410, 2014.

[McFee and Ellis, 2014b] Brian McFee and Daniel P. W. Ellis. Learning to segment songs with ordinal linear discriminant analysis. In *International Conference on Acoustics, Speech and Signal Processing, ICASSP 2014, Florence, Italy, May 4-9, 2014*, pages 5197–5201. IEEE, 2014.

[McFee *et al.*, 2017] Brian McFee, Oriol Nieto, Morwaread M. Farbood, and Juan Pablo Bello. Evaluating hierarchical structure in music annotations. *Frontiers in Psychology*, 8, 2017.

[Mokbel *et al.*, 2009] Bassam Mokbel, Alexander Hasenfuss, and Barbara Hammer. Graph-based representation of symbolic musical data. In *Graph-Based Representations in Pattern Recognition*, pages 42–51. Springer Berlin Heidelberg, 2009.

[Mount, 2020] Andre Mount. *Fundamentals, Function, and Form*. Milne Open Textbooks, 2020.

[Mukherjee *et al.*, 2009] Lopamudra Mukherjee, Vikas Singh, Jiming Peng, Jinhui Xu, Michael J. Zeitz, and Ronald Berezney. Generalized median graphs and applications. *Journal of Combinatorial Optimization*, 17(1):21–44, 2009.

[Narmour, 1983] Eugene Narmour. Some major theoretical problems concerning the concept of hierarchy in the analysis of tonal music. *Music Perception: An Interdisciplinary Journal*, 1(2):129–199, Dec 1983.

[Nieto and Bello, 2014] Oriol Nieto and Juan Pablo Bello. Music segment similarity using 2d-fourier magnitude coefficients. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 664–668, 2014.

[Nieto and Jehan,] Oriol Nieto and Tristan Jehan. Convex non-negative matrix factorization for automatic music structure identification. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada*, pages 236–240.

[Nieto, 2015] Oriol Nieto. *Discovering Structure in Music: Automatic Approaches and Perceptual Evaluations*. PhD thesis, New York University, 2015.

[Orio and Rodà, 2009] Nicola Orio and Antonio Rodà. A measure of melodic similarity based on a graph representation of the music structure. In *Proceedings of the 10th International Society for Music Information Retrieval Conference, Kobe, Japan*, pages 543–548, 2009.

[Salamon *et al.*, 2014] Justin Salamon, Emilia Gomez, Daniel P. W. Ellis, and Gael Richard. Melody extraction from polyphonic music signals: Approaches, applications, and challenges. *IEEE Signal Processing Magazine*, 31(2):118–134, 2014.

[Serratosa, 2021] Francesc Serratosa. Redefining the graph edit distance. *SN Comput. Sci.*, 2(6):438, 2021.

[Serrà *et al.*, 2014] Joan Serrà, Meinard Müller, Peter Grosche, and Josep Ll. Arcos. Unsupervised music structure annotation by time series structure features and segment similarity. *IEEE Transactions on Multimedia*, 16(5):1229–1240, 2014.

[Shervashidze *et al.*, 2011] Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, and Karsten M. Borgwardt. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(77):2539–2561, 2011.

[Sidorov *et al.*, 2014] Kirill A. Sidorov, Andrew Jones, and A. David Marshall. Music analysis as a smallest grammar problem. In *Proceedings of the 15th International Society for Music Information Retrieval Conference, Taipei, Taiwan*, pages 301–306, 2014.

[Simonetta *et al.*, 2018] Federico Simonetta, Filippo Carnovalini, Nicola Orio, and Antonio Rodà. Symbolic music similarity through a graph-based representation. In *Proceedings of the Audio Mostly 2018 on Sound in Immersion and Emotion, Wrexham, United Kingdom, September 12-14, 2018*, pages 26:1–26:7. ACM, 2018.

[Smith and Georges, 2014] Charles H. Smith and Patrick Georges. Composer similarities through "the classical music navigator". *Empirical Studies of the Arts*, 32(2):205–229, Jul 2014.

[Treinish *et al.*, 2022] Matthew Treinish, Ivan Carvalho, Georgios Tsilimigkounakis, and Nahum Sá. rustworkx: A high-performance graph library for python. *J. Open Source Softw.*, 7(79):3968, 2022.

[Wang and Mysore, 2016] Cheng-i Wang and Gautham J. Mysore. Structural segmentation with the variable markov oracle and boundary adjustment. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 291–295, 2016.

[Wang *et al.*, 2015] Cheng-i Wang, Jennifer Hsu, and Shlomo Dubnov. Music pattern discovery with variable markov oracle. In *Proceedings of the 16th International Society for Music Information Retrieval Conference, Málaga, Spain*, pages 176–182, 2015.

[Wang *et al.*, 2024] Ziyu Wang, Lejun Min, and Gus Xia. Whole-song hierarchical generation of symbolic music using cascaded diffusion models. In *The Twelfth International Conference on Learning Representations*, 2024.

[Young *et al.*, 2022] Halley Young, Maxwell Du, and Osbert Bastani. Neurosymbolic deep generative models for sequence data with relational constraints. In *Advances in Neural Information Processing Systems*, volume 35, pages 37254–37266. Curran Associates, Inc., 2022.