Synthesizing Composite Hierarchical Structure from Symbolic Music Corpora

UCSD CSE Computer Science and Engineering

audioshake

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

[1] UC San Diego, Department of Computer Science and Engineering, La Jolla, CA, USA

[2]AudioShake, San Francisco, CA, USA

Motivation

- Music decomposes into a unified hierarchical system of interacting levels of structure
  - How do melodies relate to harmonies?
- Such structure isn't guaranteed in symbolic music generation (and other kinds of sequence data as well!)
  - Users have little ability to interact with such constraints

RQ: How can we synthesize interpretable structural constraints from corpora of symbolic music data?

Contributions

We propose a **graph data structure** to encapsulate **complete hierarchical, relational structure** from **symbolic music data.**

We use this data structure to **derive *representative* structure** from ***corpora* of symbolic music data** using **stochastic and SMT optimization techniques**.

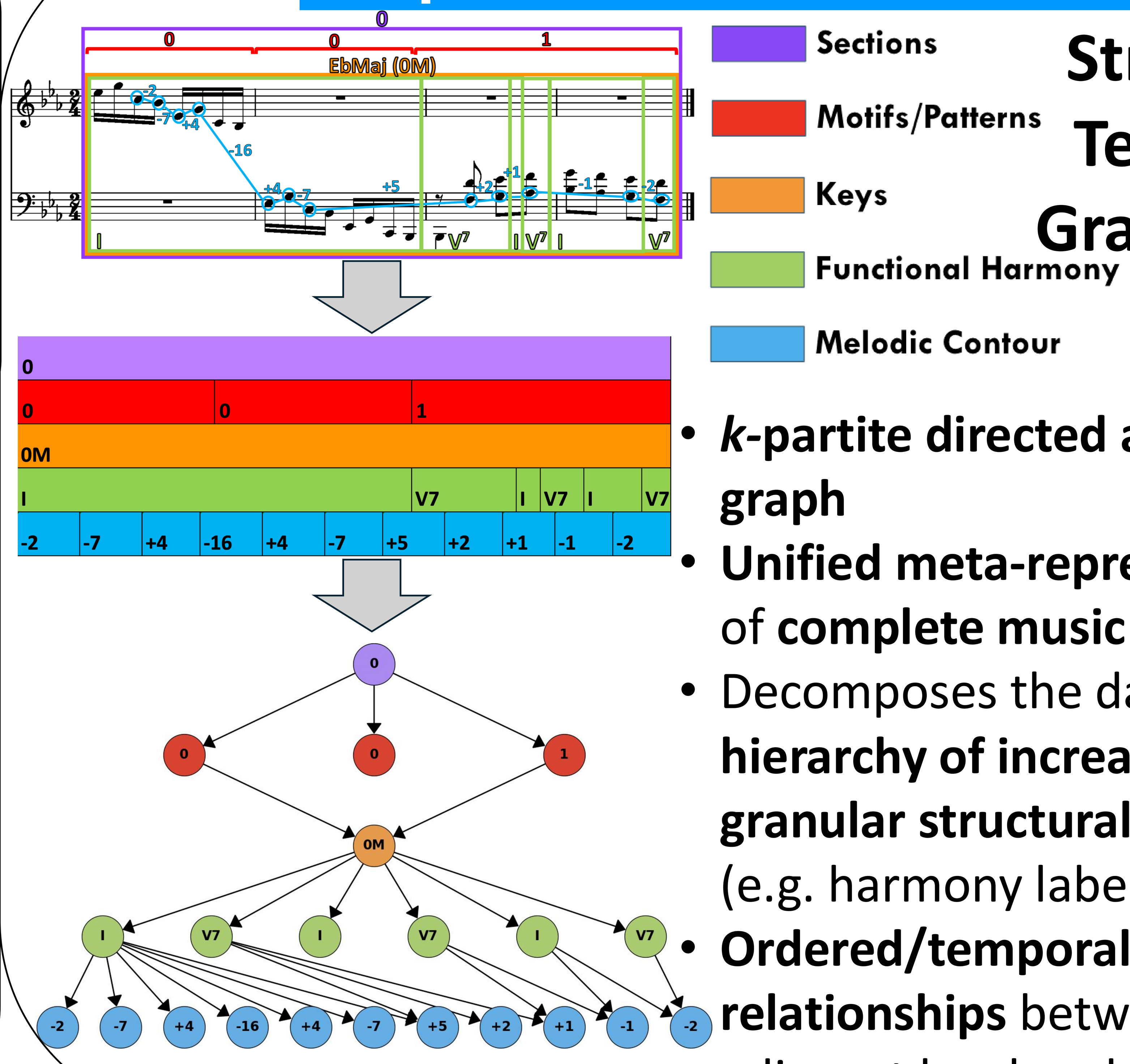
Overview of Method

Step 1: Develop an interpretable structural representation of individual items (i.e. music pieces)

Step 2: Define a distance metric between these representations

Step 3: Construct the "centroid" (i.e. median) of the set of representations under this distance metric

Step 1: STG Data Structure



Structural Temporal Graph (STG)

- ***k*-partite directed acyclic graph**
- **Unified meta-representation** of **complete music structure**
- Decomposes the data into a **hierarchy of increasingly granular structural features** (e.g. harmony labels) as nodes.
- **Ordered/temporal relationships** between adjacent level nodes (features) are edges

Step 2: Structural Distance

- Naively, distance between STGs is Graph Edit Distance (GED) -> but this is NP-Hard!
- Idea: take NP-Hard part of GED (the **optimal graph alignment problem**) and offshore to **simulated annealing**, a stochastic optimization process
- Once two STGs are near-optimally aligned, directly compute edit distance from their adjacency matrices
- We call this resulting metric **structural distance**

Step 3: Finding the Centroid STG

Step 3.1: Construct Approximate Centroid

- Goal: construct **centroid STG** structurally summarizing an STG corpus (i.e. find the **generalized median graph**)
- Two **nested NP-Hard combinatorial optimization problems**: structural distance, and minimization over these distances
- Offshore each NP-Hard part to simulated annealing (SA): solve with **nested SA over structural distances**
- But **result may not be a valid STG**

Step 3.2: Graph Repair with SMT

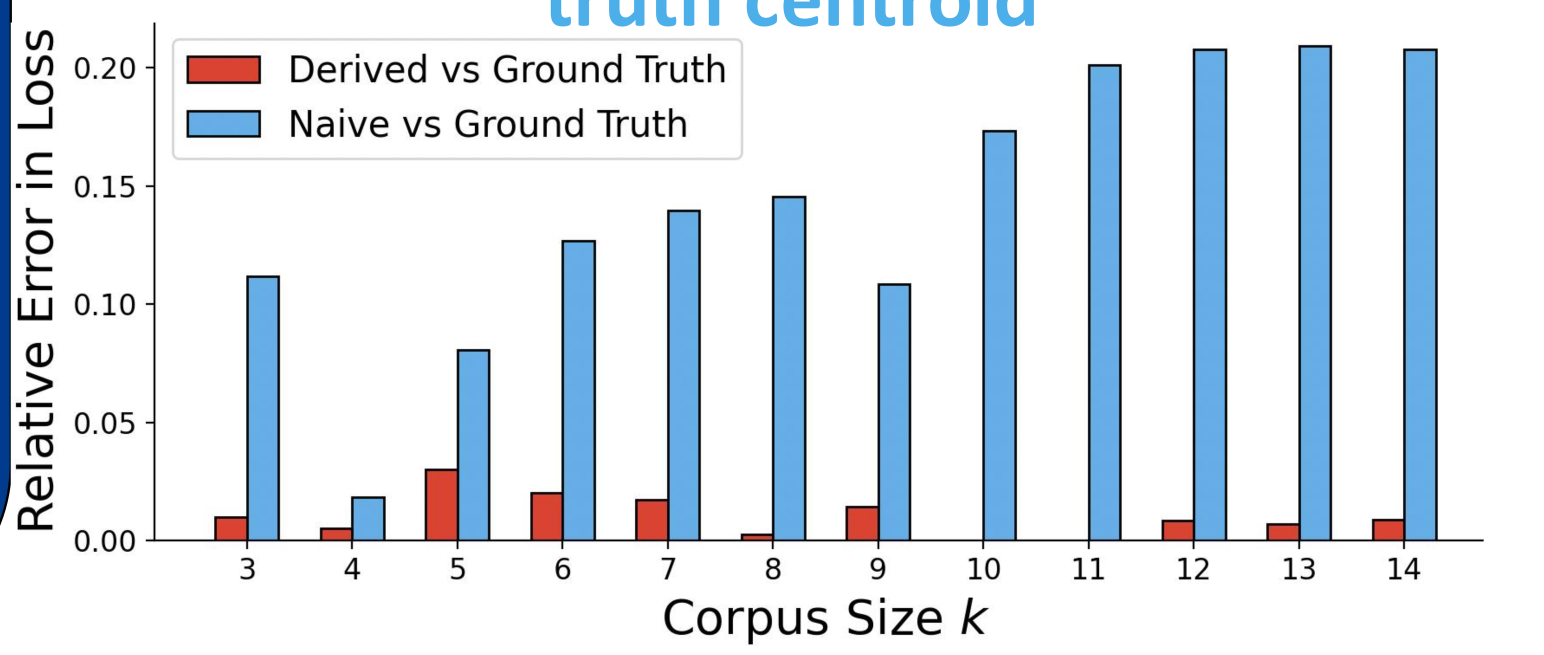
- Encode the STG's structure as rules in **quantifier-free first-order logic formulae**
- Use the **SMT solver Z3's optimizer** to **project the approximate centroid** graph onto the **nearest structurally valid STG** that satisfies all rules
  - i.e. search for the valid STG that's *as close as possible* to the approximate centroid
- Result: **final, valid centroid STG**

Conclusion

- Derived centroid STGs fulfill their objective of **minimizing structural distance over their respective corpora**
  - Centroid approaches true **generalized median STG**
  - Confirms the centroid serves as **representative structural constraints** for its corpus

Summary of Results

Relative error in loss: comparing derived vs ground-truth centroid to naive vs ground-truth centroid



- Start with base STG *G*
- Generate 11 corpora of synthetic STGs, all equidistant from *G* (ground-truth centroid by construction)
- Generate derived centroid for each corpus with our algorithm
- Compare with naïve centroid (best STG already in corpus)
- Derived centroids had **less than 3% error** from ground truth, **17.23x better than naïve**