Distributed Search for Large Collections of Symbolic Music

Alastair Porter

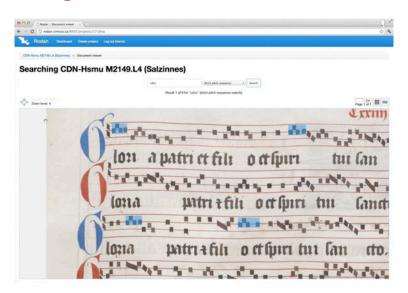
CIRMMT/McGill

28 July 2012

Outline

- Introduction
- Basic search
- Speeding up search
 - Distributed search
- 4 Further work

Searching in music



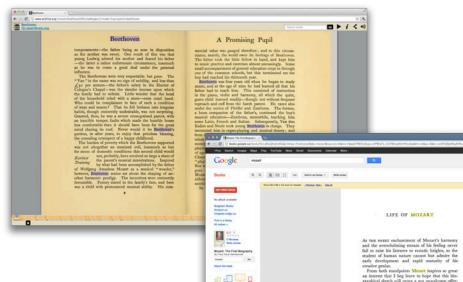
Searching in music



Current infrastructure

- Fast search for musical phrases
- Highly accurate transcriptions
- Search over monophonic pitches
 - Plus contour, neume names
 - (no rhythm, polyphony)
- Highlight results over top of original image source

Equivalents in text searching



Some terminology

- Corpus
 - A collection of documents to search over
- Document
 - A single source of content to look at (page of music, musical work, book)
- Query
 - What we are searching for

Linear search

Look at each document in the corpus and see if part of the document matches the query

Problems:

- As you get a larger corpus this takes longer and longer to complete
- Naive search has no optimisations or caching—if you perform the same query twice in a row, the search is performed twice (wasteful)

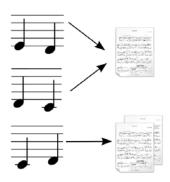
Speeding up search

- Process each document to extract musical information before making it available to search
- Split a document into sequential tokens of length n



Speeding up search

- Store as an inverted index
- Like a book index, look up all pages where a sequence of notes appears
- Can search thousands of pages of music in 10s of milliseconds, millions of pages of text in seconds (compared to hours)



Limitations of indexes in music search

- Text is relatively easy to index
- Music has many dimensions that need to be indexed
 - Pitch
 - Rhythm
 - Vertical intervals (Polyphony)
 - High-level relationships
- An n-gram search approach for music can work well for monophonic sequences of notes
- Difficult to make exploratory searches without doing a naive linear search

Searching with musical meaning

Our goal: To be able to perform musically complex queries on large corpora of music in a timely manner

- With indexed search the index needs to be computed ahead of time
- You can't predict all the types of search that might be made
- For example, search for instances of a cadence over a four-part score

Distributed search

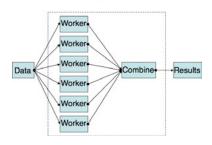
- Revisit linear search
- Use musically aware search software (music21/humdrum)
- Can we make it faster?
 - Get lots of computers to perform the search in parallel



http://hadoop.apache.org/

Hadoop

- Framework for parallel computing
- Split up the corpus evenly over a large number of computers
- Each computer processes a "mini corpus"
 - Could search a for a simple sequence of notes
 - or perform a more complex humdrum/music21 query
- Combine results once all computers have finished



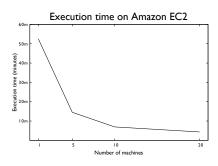
Our experiment

- Use two clusters
 - Amazon EC2
 - CLUMEQ (Consortium Laval, Université du Québec, McGill and Eastern Quebec)
- Input: 234,000 pages of music encoded in MEI format
- Distribute over 1, 5, 10, 20 computers
- **Expect** that using 10 machines takes $\frac{1}{10}$ th the time of 1
- Still using the same number of CPU hours

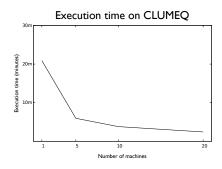




Results



Machines	Time	Speedup
I	3156s	-
5	87 I s	3.56×
10	417s	7.43×
20	261s	11.86×



Machines	Time	Speedup
I	1254s	-
5	358s	3.50×
10	229s	4.49×
20	146s	8.58×

What's next?

- We can create an index for common queries
 - e.g., key
- Perform an indexed query to limit the number of documents to search then perform an in-depth search
 - Get all works in the key of C
 - Analyse structure

Thanks







sciences humaines du Canada

