Parsing Jazz
Second Year Review

Mark Granroth-Wilding

12 Oct 2011

An article based on the presentation that I gave as a second-year review of my PhD.

Contents

1 Introduction

In this article, I am going to explain why parsing, of the sort we use in natural language processing (NLP), is useful for music processing. I will introduce the kinds of musical structure that we can interpret by parsing music and the parsing techniques that we have been using to do this.

Much of the article will focus on describing some early ideas for extending the models we've been using so far to make them more directly applicable to real music processing task.

1.1 Performances of a Song

A Song

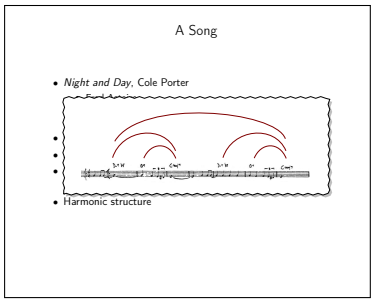
- *Night and Day*, Cole Porter
  - Fred Astaire
  - Ella Fitzgerald
  - Coleman Hawkins
- Obviously the same song: melody, rhythm, words, ...
- Still recognisable
- Different melody, different rhythm, no words, different harmony
- Harmonic structure

In the talk, I played three recordings of the same song, *Night and Day* by Cole Porter. The first two are sung by Fred Astaire and Ella Fitzgerald and, although they interpret the song quite differently, it is very easy to recognise that they are singing the same song. It is straightforward to identify some of the features that help us to spot this: the melodies are similar, at least in contour, though Ella plays around with it more; they contain recognisable rhythmic patterns, despite the fact that both singers are free with their timing; and, of course, the words are an obvious clue.

The third recording is played by Coleman Hawkins on the saxophone. Recognising the song is more difficult in this case, but nevertheless anyone who is familiar with *Night and Day* could probably confirm that this is a performance of it. But it's less obvious why it's possible to recognise this one. The melody and rhythm have almost completely changed (apart from a hint in the first bar).

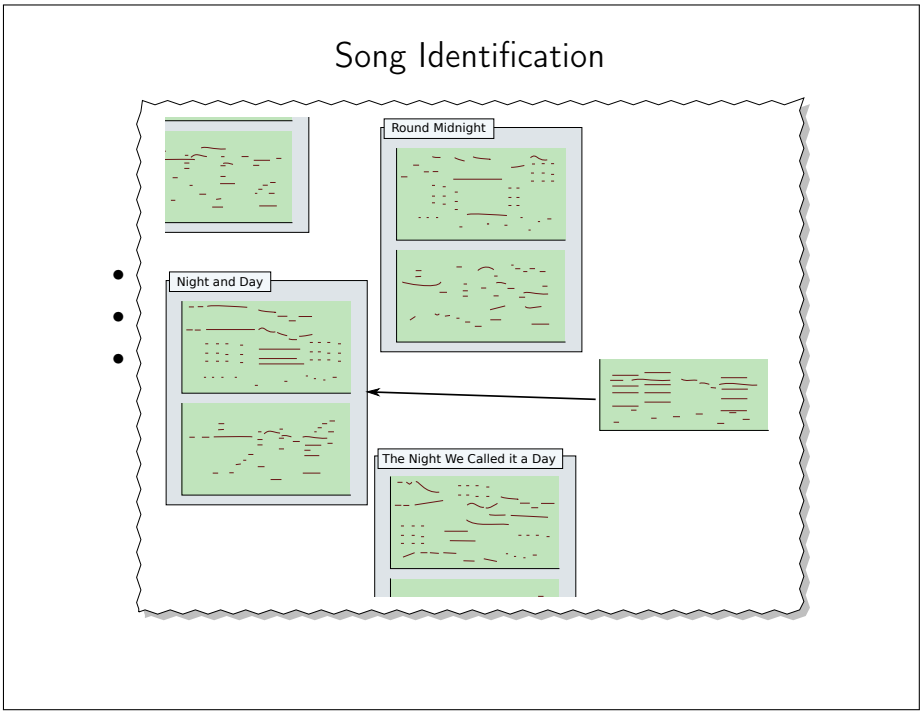
There are no words to help us out and the harmony of the accompaninent is even different.

But something must be the same! In fact, much of the similarity is in the harmony. Although the chords played are not identical, the structure of the harmony is very similar. The recognise that these are the same song, we must analyse a more abstract structure than just the notes played or the chord sequence. This sort of analysis actually underlies our understanding of harmony.

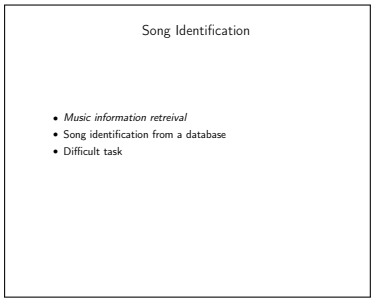


As it happens, we do a similar kind of structured analysis, somewhat orthogonally, of meter and rhythm, but I'm not going to consider that here.

1.2 Song Identification



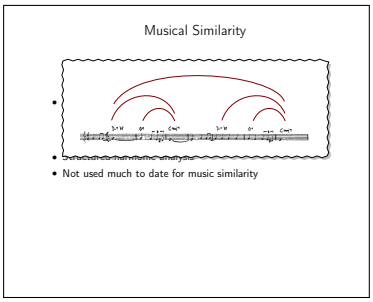
Music information retreival (MIR) is a field were metrics of musical similarity are required. Let us consider a task of song identification. Say we have a corpus of performances of songs, transcribed in some digital form, and we know what song each instance is a performance of. Now, given a new performance transcription, we want to be able to say what song it is. We could do this by comparing the new performance to those in our database and finding the closest match, which we assume to be the same song.



In jazz, where songs can be varied as freely in performance and the examples I played at the beginning of the talk, identifying two performances of the same song can be a very difficult task.

Musical Similarity

- Need metrics of musical similarity
  - Rhythmic patterns
  - Melodic similarity
  - Harmonic similarity
- Structured harmonic analysis
- Not used much to date for music similarity

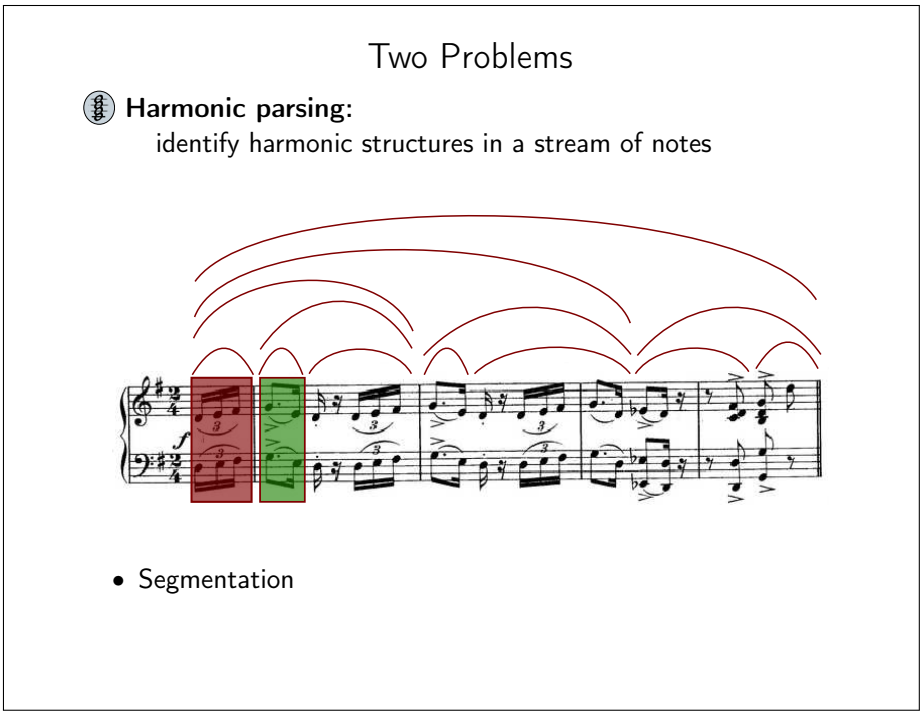


To do a task like this, we need metrics of musical similarity. Many aspects of music contribute to our perception of similarity. We recognise familiar rhythmic patterns and we hear similarity on melodic lines. Another aspect is harmony: to evaluate harmonic similarity we must analyse the structure of the harmony. This can be particularly important for song identification in jazz, as it was in the example performances I played.

Metrics of harmonic similarity have not been exploited much in these sorts of tasks up to now. We're going to use a process of harmonic analysis to reveal the structure of harmony. This structure could then be used as the basis for a metric of harmonic similarity.

2 Two Problems

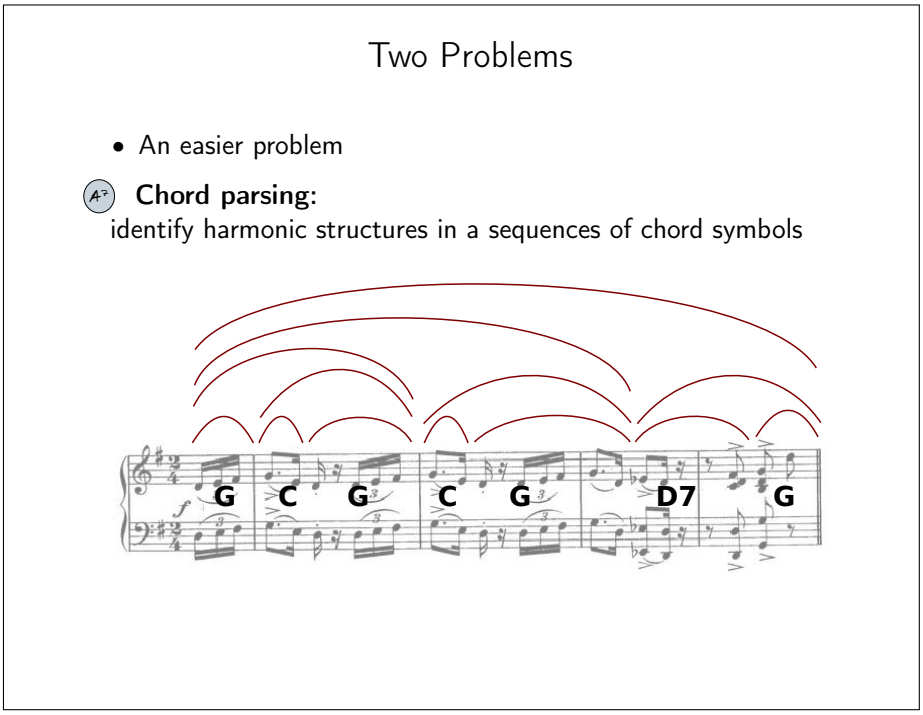
2.1 Harmonic Parsing



I'm going to introduce two problems. First of all, a difficult one: **harmonic parsing**. This is the problem of identifying the structures of harmony from a stream of notes. Just to be clear: I'm not talking about handling sound waves here, but rather some sort of discrete representation of the notes of a performance. Doing this involves identifying passages of similar harmony and then their relationship to one another.

This is a difficult problem. One reason why it's so difficult is that there's no minimal meaningful unit of harmony defined in the input – it could be anything from a pair of notes up. So, the first step in tackling this problem is always going to be decided where these units of harmonic structure – **chords** – are. This problem is referred to as **segmentation**.

2.2 Chord Parsing

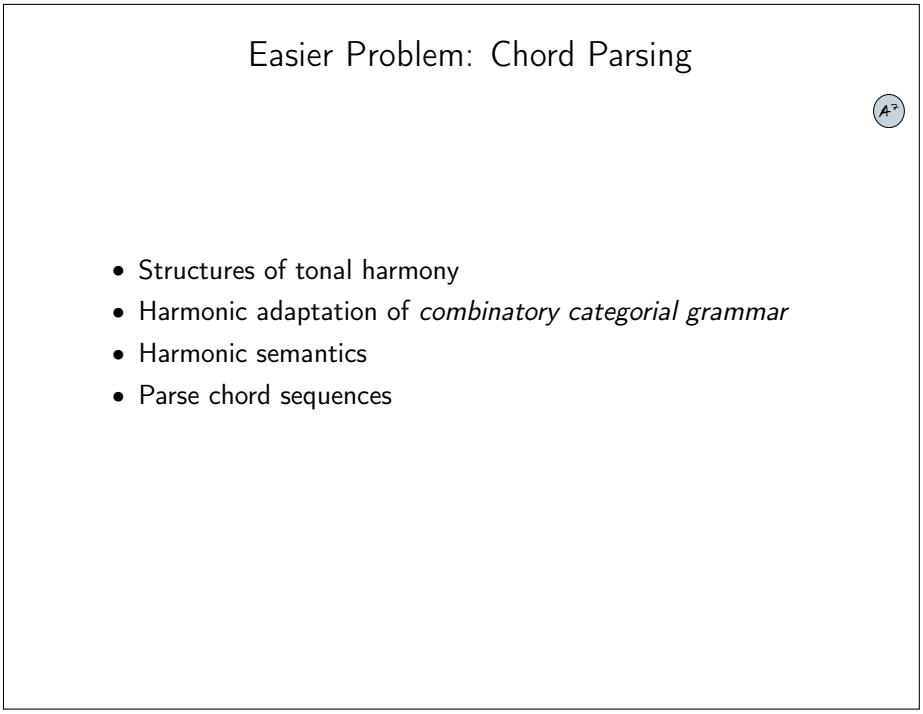


But let's consider an easier problem: that of **chord parsing**. Chord symbols, like those shown here, are used by musicians as a more abstract notation of the harmony of a piece of music than a score. They're used particularly in genres where it's common to improsive, like jazz or folk music.

We can identify the harmonic structures from these symbols. The problem of segmentation is already mostly done for us (there are one or two segmentation-like issues remaining, but it's at least a lot easier). A chord sequence is a simpler signal to process, but the higher level analysis, that that we reveal by parsing, is the same. This makes chord parsing a good place to start.

We've been tackling this problem over the last few years. I'm going to cover briefly how we've done that. Then I'll come back to the issue of how we could extend our models for chord parsing to the harder problem of full harmonic parsing.

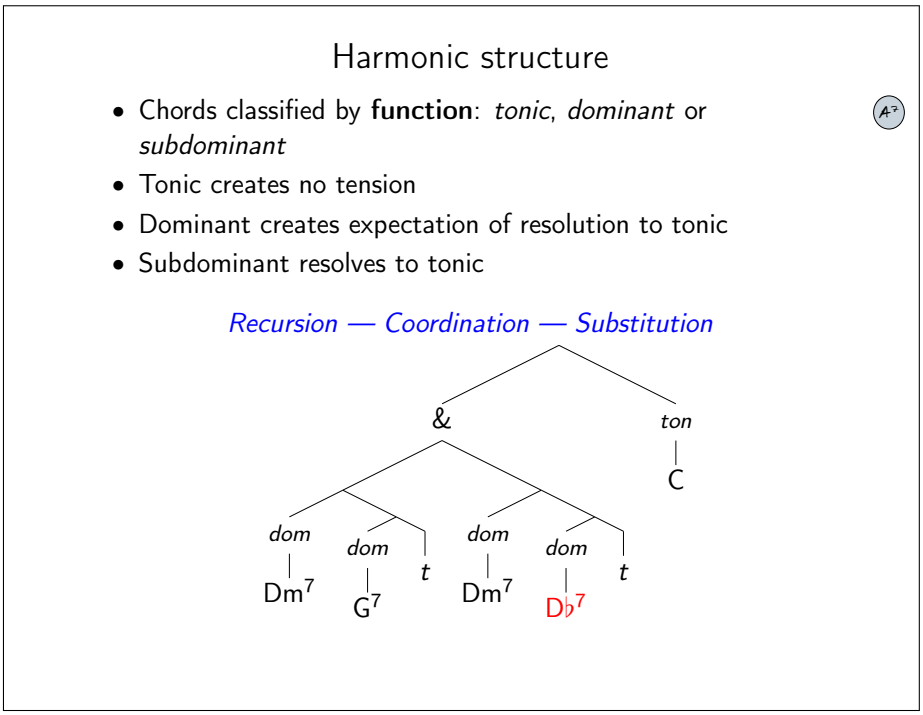
3 Parsing Chords



I'm going to introduce some of the structures of Western tonal harmony. We use a musical (harmonic) adaptation of combinatory categorial grammar (CCG) to handle these structures. CCG is a lexicalized grammar formalism. Don't worry if you're not familiar with it – you won't need to be for this discussion. The grammar has a semantics that represents the structured harmonic analysis that we want to get out. This grammar can be used to parse chord sequences more or less by treating chord symbols as words.

I'm not going to give details of the syntactic formalism or the harmonic semantics here, but rather just an overview of the kinds of structures they handle. Up to now, though, most of our work has been on just these details, as well as the parsing techniques to go with the grammar.

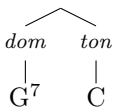
3.1 Harmonic Structure



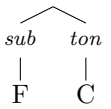
When performing harmonic analysis of music, chords are typically classified according to their **function**: tonic, dominant or subdominant. This indicates something about their relationship to the current key.

A **tonic** chord is the basic chord of a key. It creates no tension. Try playing a C major triad: that's a tonic chord for the key of C major.

A **dominant** chord creates a tension and an expectation of resolution to a particular other chord, namely the tonic. Try playing a C chord again (to remind yourself we're in C major) and then the chords G^7 C. This is a dominant chord and its resolution.

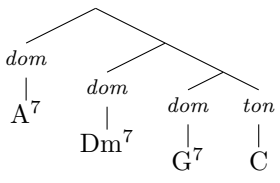


A **subdominant** chord also creates a tension. Its resolution is also to the tonic. Try playing F C this time.

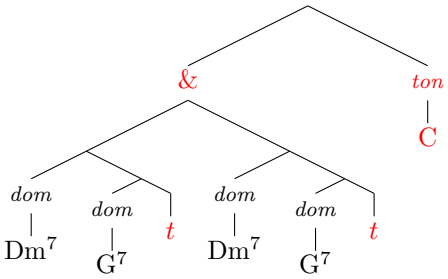


These two patterns – *cadences* – are the basic building blocks of structure in harmony.

They can be applied **recursively**. That is, the resolution of a dominant chord may itself be a dominant chord, requiring in turn its resolution. Try playing the extended cadence (right) with several dominants.



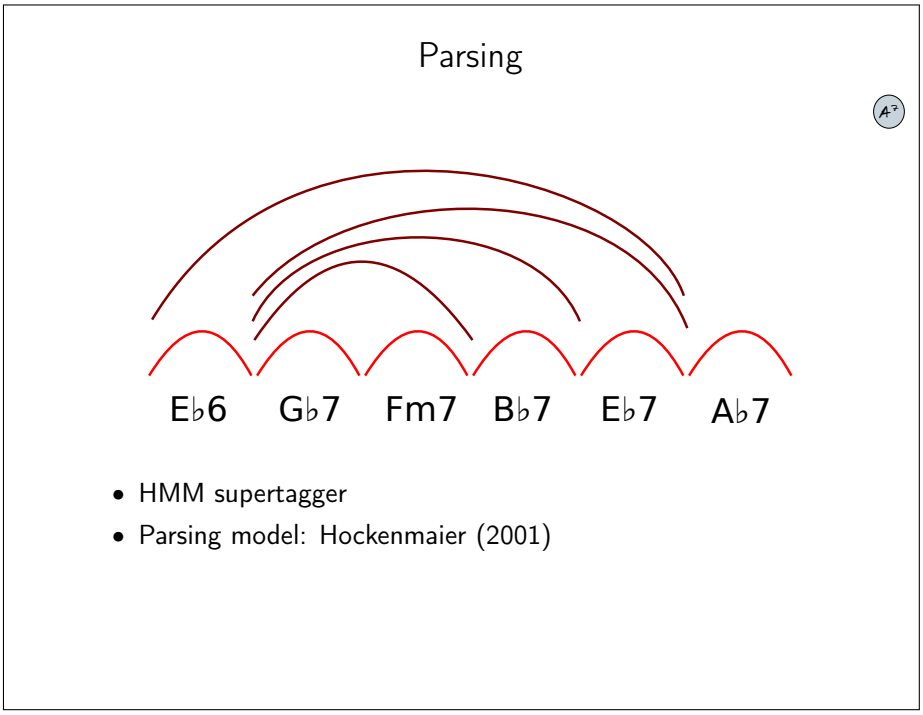
They can be **coordinated**. By this I mean that the resolution of a tension chord can be delayed and only provided later as the resolution to another, similar tension chord. Try the example below, in which t marks the shared, expected resolution.



Then chords may be **substituted**. In certain contexts, certain chords may substitute for others, preserving the harmonic structure. The example in the slide shows the above coordinated cadence with a substituted chord. This is an example of the *tritone substitution*, which is a common substitution in jazz.

In this way, complex hierarchical structures are built up. The categories of our CCG grammar allow us to assign interpretations to chords that reflect their roles in these structures. Then a full harmonic analysis is produced by parsing, using some of CCG's standard combinators and a couple of special combinators designed to build these musical structures.

3.2 Parsing



The process of parsing chord sequences involves two stages. First, we use a **supertagger**. A supertagger is a sequence model that suggests lexical categories to the parser. Recall that CCG is a lexicalized grammar formalism. This means that the lexical categories carry quite a bit of information that constrains the role the chord may play in harmonic structure. In other words, a large part of the process of structural interpretation is done by choosing the category to assign to it.

The supertagger we use is a hidden Markov model (HMM). We train the model on a small labeled corpus of chord sequences.

In the second stage, we use a parsing model to speed up and direct the process of parsing the categories suggested by the supertagger. We use a model like that of Hockenmaier (2001). This is a PCFG-style model that uses tree probabilities in the derivation tree. We train this model on the same labeled chord corpus.

Chord Parsing Results

Model	*P (%)*	*R (%)*	*F (%)*	*Cov. (%)*
HMM baseline	74.6	82.1	78.2	100
Supertagger/parsing model	**87.4**	85.7	86.5	95
Supertagger/parsing model, backoff	85.3	**91.0**	**88.1**	100

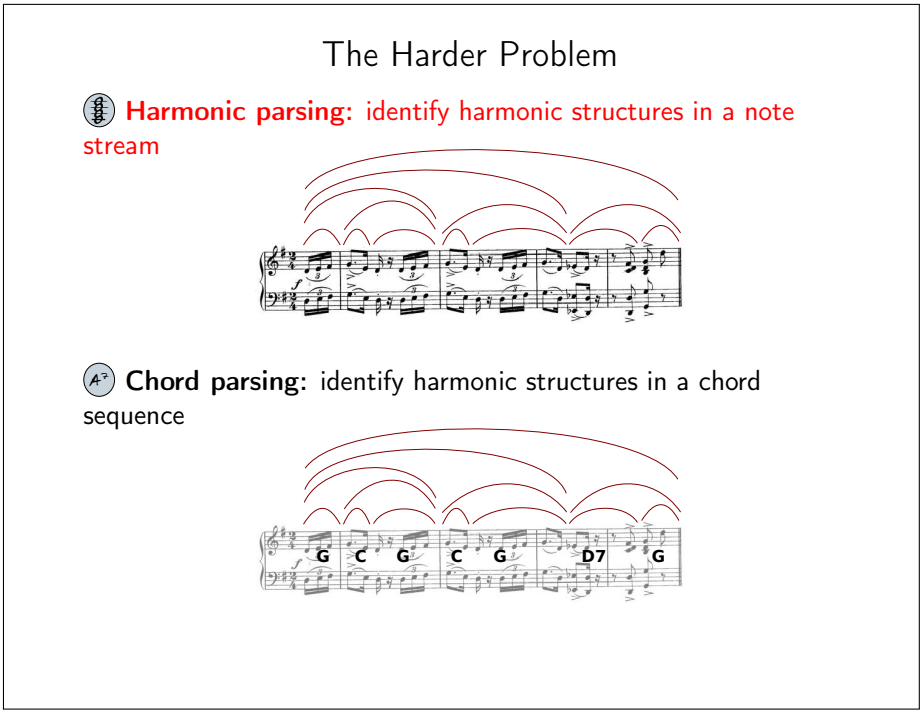
- Grammar model better than HMM baseline
- Using HMM baseline as backoff produces best results

For the purposes of this article, the details of this table are not important. These are the results of our evaluation of our parsing methods. They indicate the precision (P), recall (R) and f-score (F) of the harmonic structure recovered by each model.

I am displaying this table here in order to make two points. Firstly, our system that uses the parsing model and the harmonic grammar is able to do a better job at recovering the harmonic structure of the chord sequences than a pure HMM baseline model.

Secondly, this HMM baseline can be successfully used as a backoff model in the cases where the parser is unable to produce an interpretation at all, the combined system achieving our best results.

4 Musical Supertagging

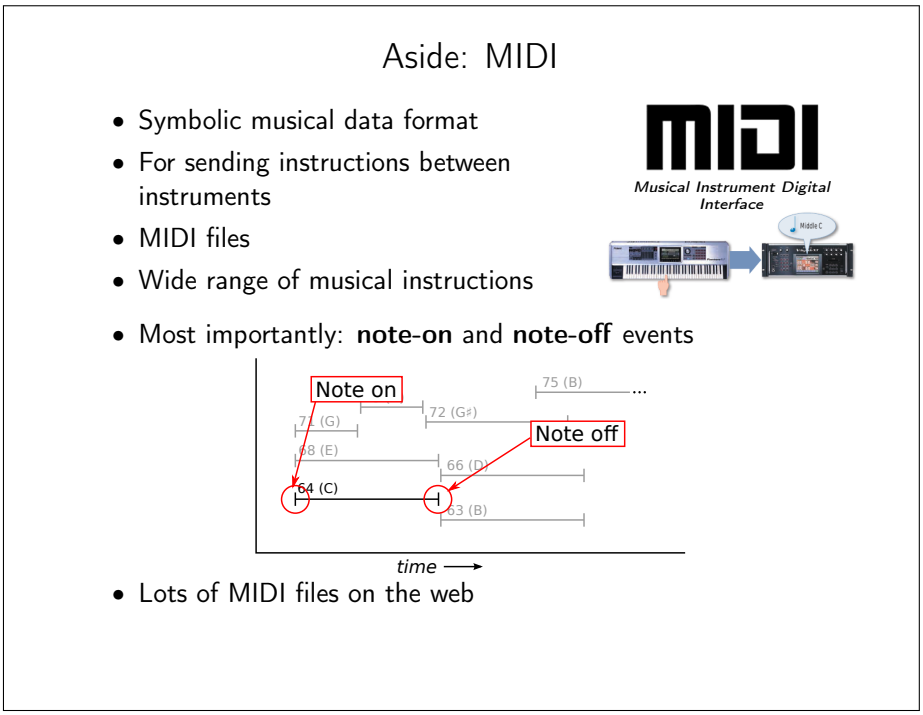


Recall the two tasks I introduced earlier: harmonic parsing and the simpler chord parsing. In the previous section, I outlined our approach to chord parsing. We have demonstrated that our methods tackling this task have a reasonable degree of success.

I'm now going to suggest an approach to the harder problem of harmonic parsing by building on the models we've been using for chord parsing.

But before that, a brief aside...

4.1 MIDI

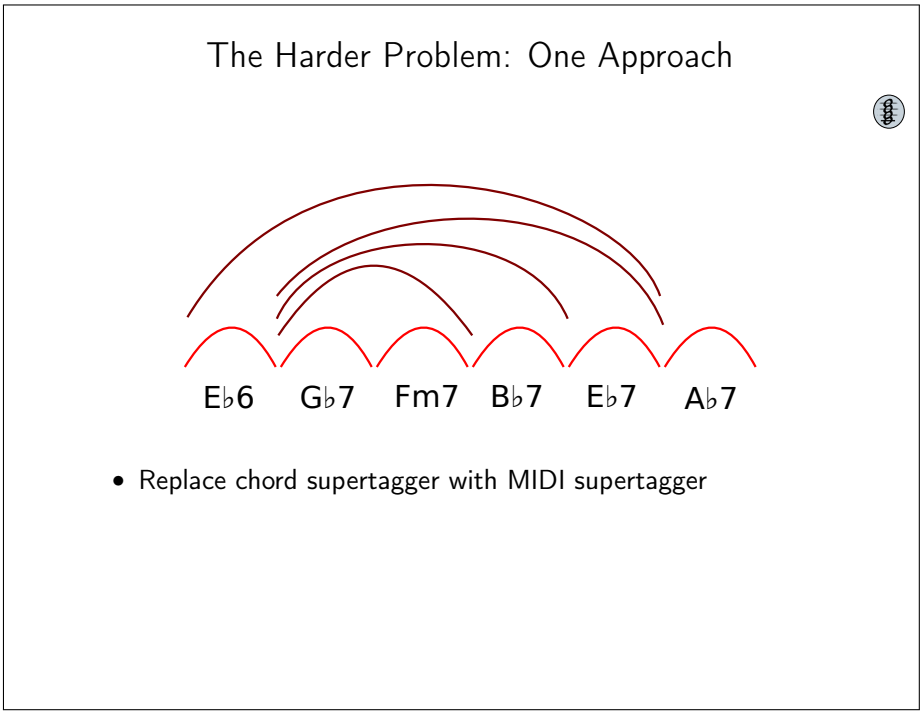


MIDI is a digital representation of the notes of a musical performance. It was designed for sending instructions between electronic musical instruments. These instructions (or *events*) can be stored in **MIDI files** – essentially a stream of events, each associated with a time. The data in these files is sometimes generated from an electronic representation of a score, or sometimes recorded directly from an instrument.

The instructions of a MIDI stream may include all sorts of information: the instrument type, changes in dynamics, even changes to the tuning of the instrument. But at its simplest, MIDI data is just a stream of **note onset** and **note offset** events.

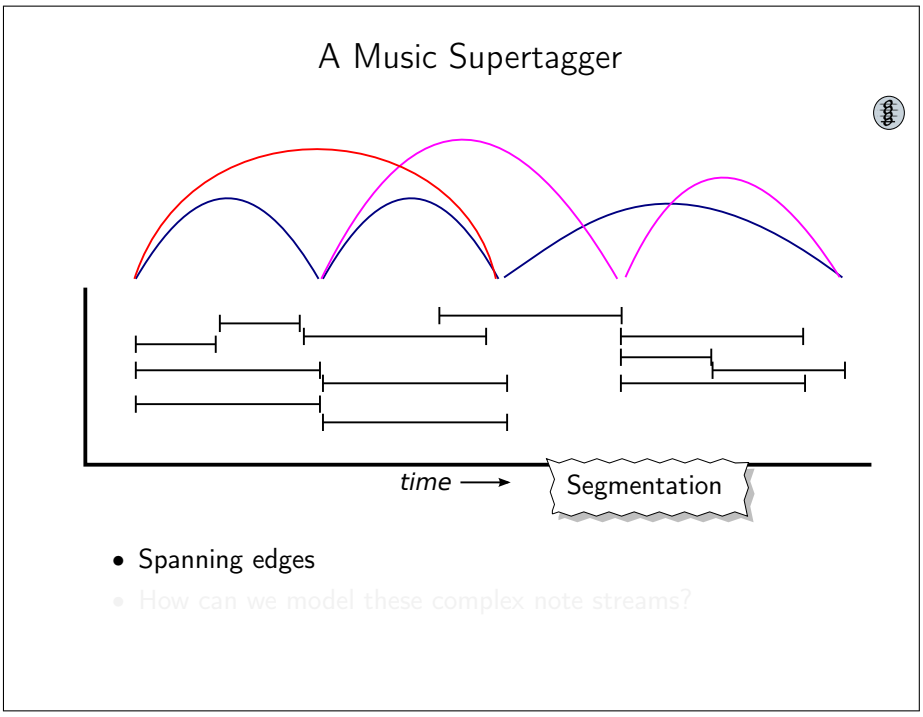
There's lots of this stuff around on the web in the form of MIDI files, of widely varying quality.

4.2 MIDI Supertagging



I'm now going to suggest an approach to the harder of the two problems: full harmonic parsing.

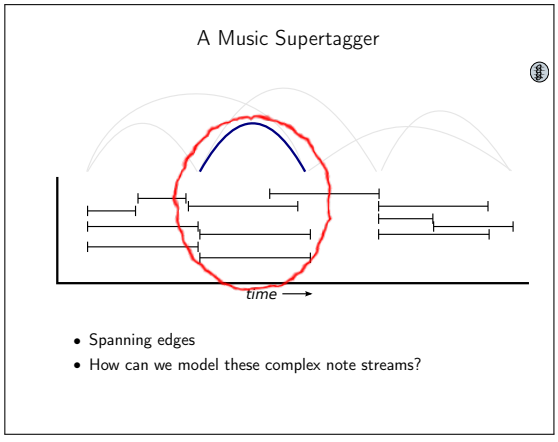
Recall that when we did chord parsing we began by using a supertagger to assign categories to chords, and then parsed them to produce the higher-level analysis. If we could replace the chord supertagger with a model that assigns categories directly to a stream of notes (such as a MIDI file), we could then use the same analysis and associated modeling techniques from there on up.



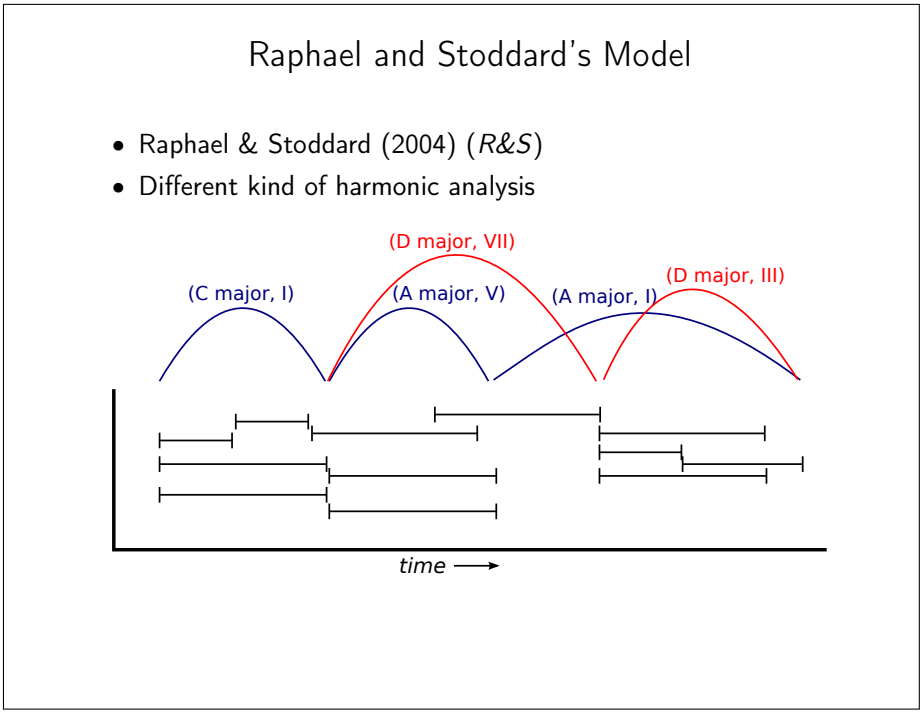
So, let's consider a musical supertagger. The input is represented as MIDI data. The job of the supertagger is to assign categories to spans covering the whole passage of the music. This involves the segmentation problem: in this case this comes down to deciding where these spans begin and end.

It may in fact wish to propose different analyses which segment the notes in different ways. The parser can deal with this, though. The supertagger just needs to propose these divisions as spanning edges in the parse chart – rather like allowing multi-word lexical items in natural language parsing.

So now the problem comes down to this: how can we construct a model of these complex note streams that segments them and proposes lexical categories for the segments. This is a difficult problem, so for inspiration I've taken a look at a related model, designed for a similar problem.



5 Raphael and Stoddard



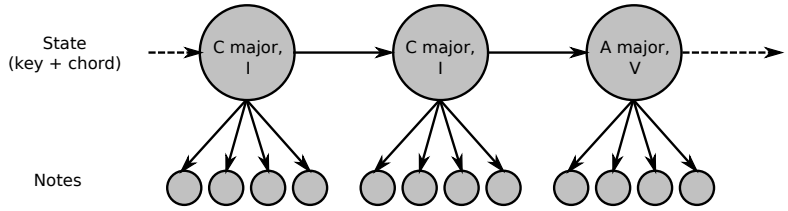
The model I've looked at is that proposed by Raphael & Stoddard (2004) (henceforth *R&S*). Their model produces a harmonic analysis, somewhat similar to ours, but shallower. It's one of the few approaches to this kind of problem that uses probabilistic modeling, so appears to be a good place to start.

The model takes its input in the form of MIDI files. It assigns a label to every chord consisting of two parts: a key ('C major' in this case) and a chord label ('I' here). This chord label is chosen from the seven triads (chords of three notes), I, II, III, ..., in the key. The labels may span different lengths.

Before going any further, it makes sense to ask how this form of analysis relates to that we do. Their chord labels I, IV and V relate closely to some of the categories of our lexicon, namely those that interpret **tonic**, **subdominant** and **dominant** chord functions (introduced on p. 9). The others are unclear in their meaning. The technique has no way of accounting for **substitutions**. It also cannot account explicitly for higher level structures including **coordination** (again, see p. 9), which consequently appear only as noise to the model.

The target domain Raphael and Stoddard and in mind was tonal classical music, such as that of Haydn.

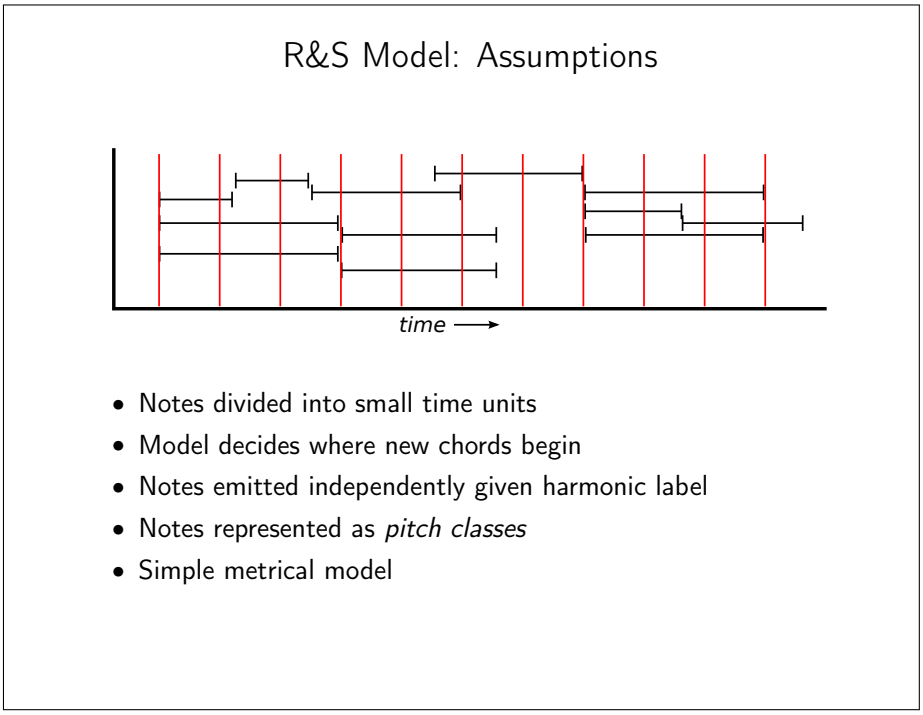
Raphael and Stoddard's Model



- HMM
- State labels → harmonic analysis
- Emissions → bag of notes

The model itself is a hidden Markov model (HMM). Its states are associated with the harmonic labels described above: each consists of a key and a chord label. The emissions are treated as a **bag of notes** (cf. a bag of words in NLP).

5.1 Assumptions



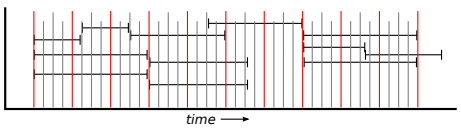
Certain assumptions are made in the construction of the model and I will run through a few of the key ones here.

Firstly, the model assumes that the input passage of notes has been divided into small time units beforehand. These are treated as if they correspond to bars in the music, though typically they are the length of half a bar. This is the smallest time unit to which a label may be assigned. This means that the **segmentation** problem is not completely free, but in fact rather constrained.

I have mentioned already that the emissions are treated as a **bag of notes**. This means that each note is emitted independently given the harmonic label (state).

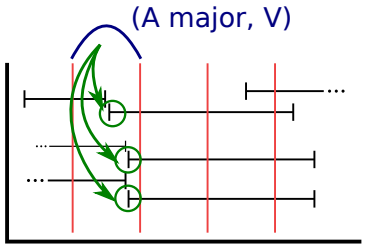
The notes themselves are represented as **pitch classes**. This means that the full range of equally tempered pitches, 12 per octave, is projected onto a single octave. As far as the model is concerned, there are only 12 possible notes (though each may be emitted more than once from a single state). Another consequence is that the ordering (in pitch) of the notes is ignored.

The model incorporates a simple model of **metrical prominence**. The emission probabilities are conditioned on the time at which the note occurs, in relation to divisions of the time units.



5.2 Emission Distribution

R&S Model: Emission Distribution

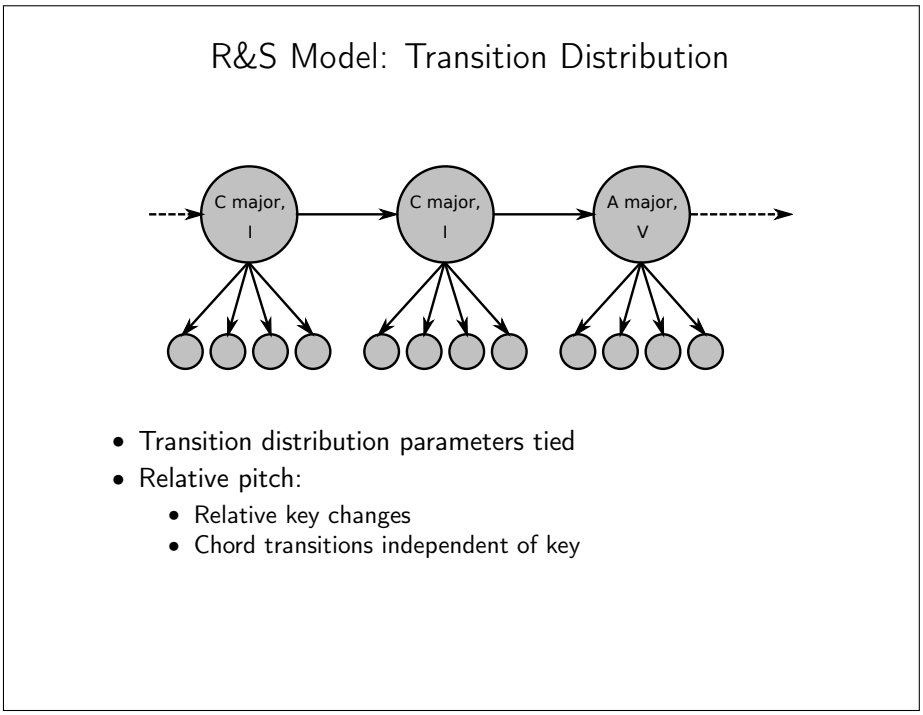


- Emission distribution shared across states
- Probabilities of notes relative to state:
  - triad
  - in-key notes
  - other notes
- Limited extension to tetrads

Raphael and Stoddard don't train a separate emission distribution for each state, but in fact share a single emission distribution across all states. They do this by learning probabilities of the notes relative to the state label. Distinct probabilities are learned for each of the notes of the triad (the chord of three notes represented by the chord label and key combined); the in-key notes (excluding those in the triad); and all other notes.

They also mention in passing a limited extension of the model to incorporate **tetrads** (chords of four notes). This is specifically to capture the **dominant seventh** chord – a tetrad including a flat seventh degree, typically indicative of dominant function. They do this by adding a single extra chord label representing the V^7 chord.

5.3 Transition Distribution



The transition distribution is simplified by tying parameters. This serves to reduce the parameter space and also to constrain the structure of the model. The most important of these modifications regards **relative pitch**. This is in fact a desirable feature of any musical model.

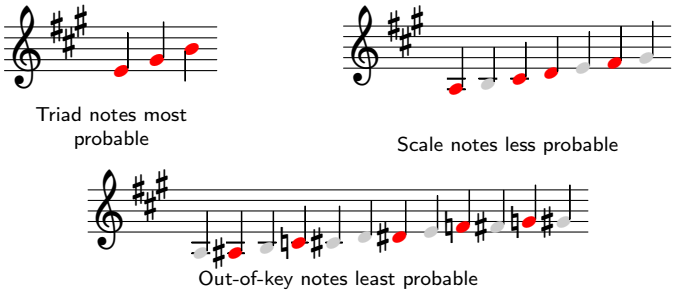
Firstly, the model only takes account of the relative pitch between key changes. For example, a transition from C major to D minor will have the same probability as a transition from G major to A minor.

Secondly, transitions between chord labels are considered independent of the key in which they take place. For example, a transition from V to I will have the same probability in C major as in G major.

5.4 Training

R&S Model: Training

- Unsupervised training by EM
- Initialize emission distribution



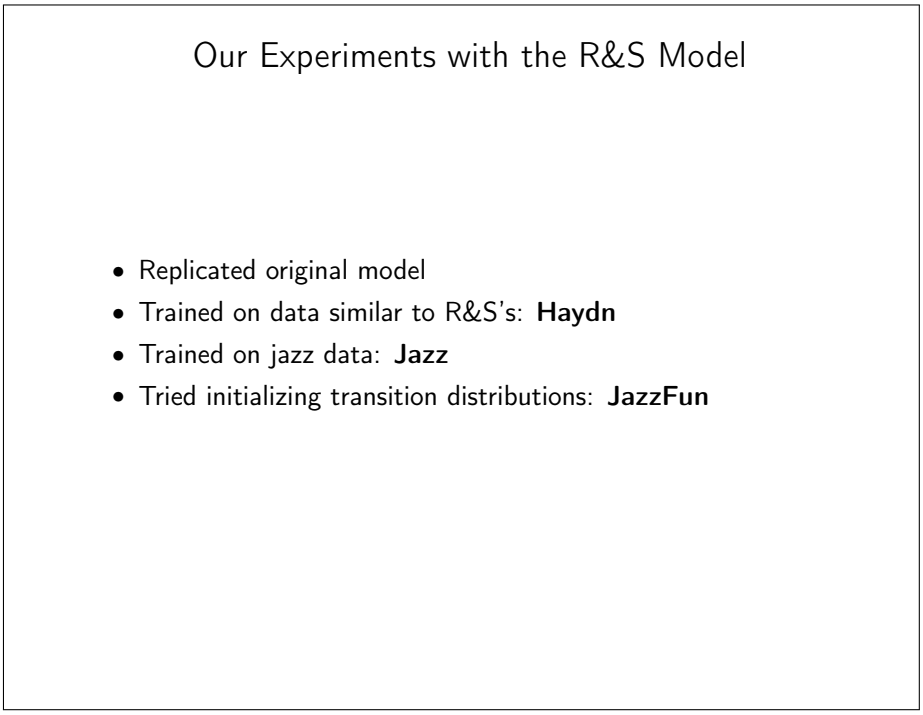
- Transition initialization makes no difference

Raphael and Stoddard do not use labeled MIDI data to train the model, since this is hard to come by. Instead they train it in an unsupervised manner using expectation maximization (EM). They make this work by initializing the emission distribution to bias the training towards a sensible outcome.

They initialize their emission distribution so that the notes of the triad of any state any most likely to be emitted, the in-key notes are next most probable, and the remaining, out-of-key notes are least probable.

They claim that initializing the transition distribution appears not to affect the outcome of training.

5.5 Experiments



I have implemented the R&S model, replicating as closely as possible the original model.

I trained a model on Haydn MIDI files, intended to be similar to those used by Raphael and Stoddard (the original files they used aren't available). I will refer to this model as **Haydn**. I also trained a model, identically structured, on jazz MIDI files. I'll call this **Jazz**.

I tried initializing the chord transition parameters to bias the model towards a chord functional interpretation. For example, we'd hope that a V chord is likely to be followed by its resolution, I. I then trained the model as the others. This is the **JazzFun** model.

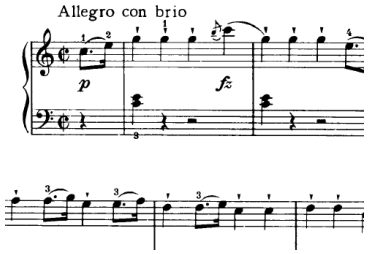
R&S Model: Evaluation

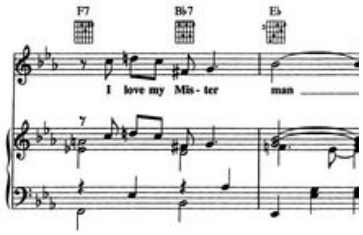
- R&S present no empirical evaluation
- No single correct analysis: many valid analyses
- Doesn't matter for now
- Parameters are easily interpretable

Raphael and Stoddard present no empirical evaluation of the model. Instead they offer a few examples of good analyses it produces. They argue that it's difficult to evaluate, since there's not one single correct harmonic interpretation, but many valid analyses, of any particular piece of music. This means it's not meaningful to evaluate accuracy against a gold standard.

For now, this needn't worry us. The aim of this exercise was to get an idea of what the model captures and how well it carries over to jazz data. Most of the parameters of the model are easy to interpret at a high level, so we can learn something just from looking at the parameters that the model learns.

R&S Model: Evaluation





- Informal conclusions from the models' output and distributions
- Tried jazz models on jazz MIDI files
- R&S hand-tweaked transition distribution: I haven't

I have drawn some informal conclusions about the model by looking at the output when it's applied to MIDI files and looking at the trained parameters. I'll present some of my key conclusions here. I've tried the jazz models on jazz MIDI files.

Just a note first, though. Raphael and Stoddard mention that they set some of the parameters of the transition distribution by hand, but they don't say which parameters or how they set them. I have not tried to guess what they did or find parameters that work best, but have just looked at the outcomes of training. It's worth bearing in mind that the original R&S model would have performed better, with some of the parameters set by hand.

R&S Model: Conclusions

- Transition distributions learned some aspects of chord function
- Inevitably noisy
- Transition initialization makes no difference
- Metrical model weaker for jazz, but still meaningful
- Tetrads (dominant seventh) helped in some cases, but not all
- Triad model was confused by use of tetrads in jazz
- Worth incorporating tetrads better

The transition parameters showed at least some signs of having learnt chord functions – that is, the resolution of dominant and subdominant chords. Since the model has no way of handling higher-level structures, this will inevitably be noisy. A V chord should have a high probability of resolving to a I, but often it doesn't resolve straight away and the model cannot account for why. However, this at least suggests that the model is able to pick up some notion of chord function, which is promising for our purposes.

The **JazzFun** model corroborated the claim about transition distribution initialization: its parameters were almost identical to **Jazz**.

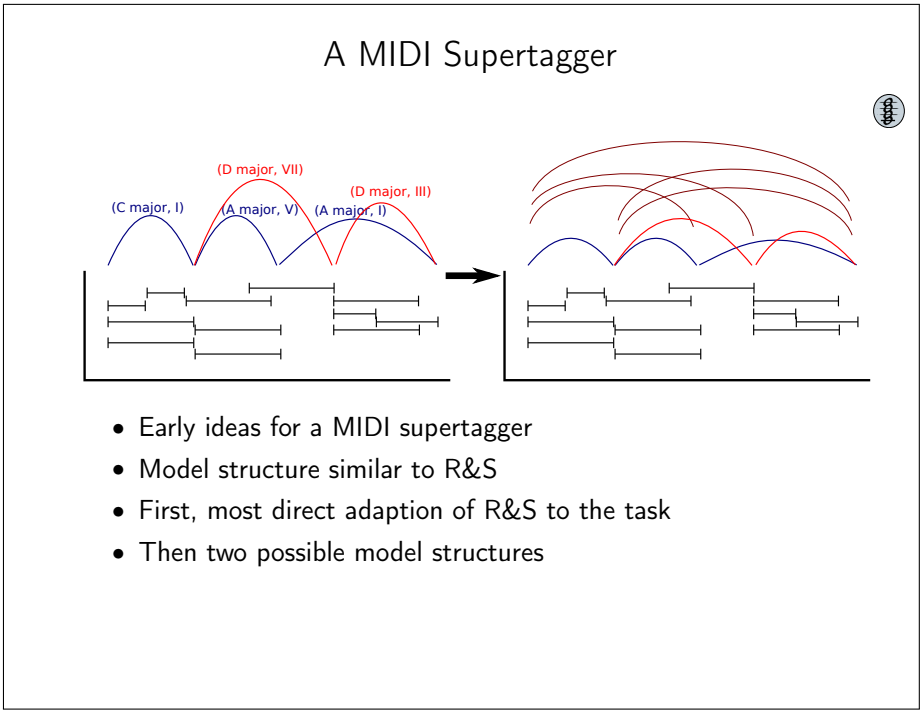
The metrical prominence model seemed to make a weaker distinction in the jazz models than **Haydn**, but did still seem somewhat pertinent.

The addition of a chord label to capture the dominant seventh chord seemed to help in some cases, but not others – this was rather inconclusive.

Interpretation of Haydn relies heavily on recognising triads. Tetrads (four-note chords) are much more heavily used in jazz than in Haydn and give important clues to chord interpretation. We can expect any model that looks only for triads to be confused by the heavy use of tetrads, since each tetrad contains at least two different triads. Indeed, the **Jazz** model did seem to be confused in just this way.

My conclusion from these last two points is that, if we wish to handle jazz music, it's going to be worth finding a way of incorporating tetrads better.

6 A MIDI Supertagger



The next step is to try using the R&S model as inspiration for a MIDI supertagger. These are just some early ideas – I've not tried implementing any of these yet.

The basic structure of the supertagger model could be similar to R&S. I'm therefore going to begin by presenting the most direct, naive adaption of R&S. I'll then go through two ideas for better models that may work for the task.

6.1 A Naive Adaptation

A Naive Adaptation of R&S

- States → lexical categories
- Lexicon of chord grammar:

Ton.	X	$:= I^T$
Dom.	X^7	$:= I^D / IV^{D\|T}$
Dom-tritone.	X^7	$:= \flat V^D / VII^{D\|T}$
	⋮	

Schema name *Chord class* *Syntactic category*

- State: *(lexical schema, root pitch)*
- E.g. (Dom, A♭) ⇒ category $A\flat^D / D\flat^{T|D}$

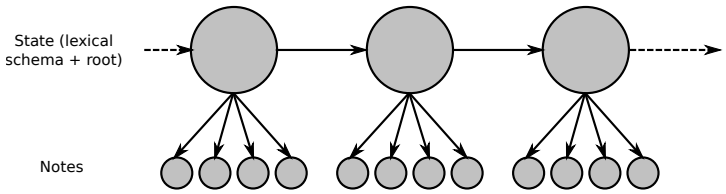
The simplest way to create a supertagger from the R&S model is to make the states represent lexical categories. I've skated over the details of the jazz grammar up to now, and I don't propose to go into them extensively here, but I need to show a bit of the lexicon for this to make sense.

Here are a few entries from the lexicon. Each entry is a schema. It has a name and is assigned to a **chord class**. These chord classes are groups of **chord types**. Then, after the :=, is the syntactic type – don't worry about what this means, there's no need to go into it here. The schemata generalize over **chord roots** (that is, the defining pitch of the chord). To produce a lexical category, a schema needs to be instantiated at a particular root.

Our states, therefore, need to consist of an identifier of a lexical schema (we can use the name) and a root pitch. So, for example, the pair (Dom, A♭) refers to the Dom schema. Together these values define a lexical category: $A\flat^D / D\flat^{T|D}$.

A Naive Adaptation of R&S

(Dom, A♭) $\Rightarrow$ $A\flat^{D}/D\flat^{7|D}$

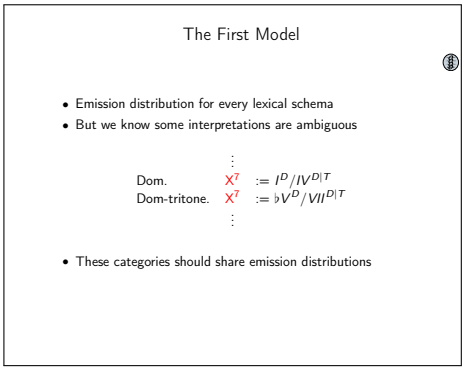


- Relative pitch $\rightarrow$ only change in chord root
- Emissions conditioned on lexical schema, not root
- Emission distribution over notes relative to state's root
- R&S metrical model less useful for jazz: try with and without

We can use the same approach as R&S to relative pitch. Only the difference between the consecutive chord roots matters, not their absolute pitch. Another aspect of relative pitch is that the emission distribution should be conditioned only on the lexical schema part of the state label, not the root. This can work if we learn probabilities of notes relative to the state's root. We then have an emission distribution that generalizes of the state's root.

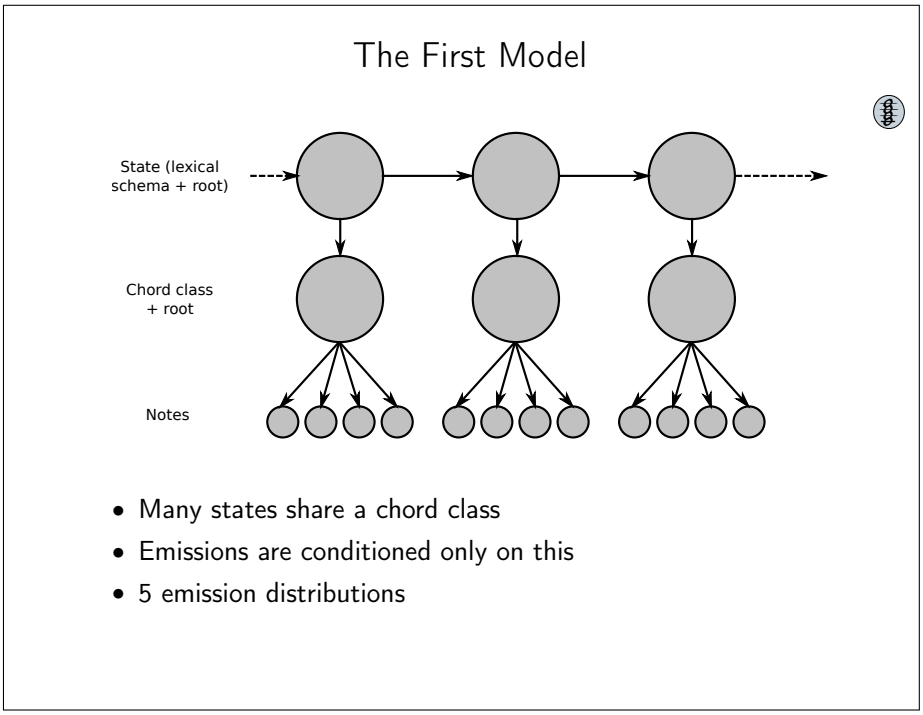
The R&S metrical model seemed to be less useful for jazz than Haydn. We could try a supertagger model with or without it. For simplicity, I'm going to assume for the rest of this discussion that we're not using it.

6.2 The First Model



The model on the previous page has a separate emission distribution for every lexical schema. But this shouldn't be necessary: we actually already know that some interpretations (represented by different categories) will always be ambiguous in the surface form.

Take a look at this fragment of the lexicon again. The two schemata, Dom and Dom-tritone, are assigned to the same chord class. When parsing chords they were always alternative interpretations of the same chords. The schemata should therefore share their emission distribution.



Here's the first model structure I'm proposing. A state is associated with a chord class and the emission distribution is conditioned just on this. Meanwhile, the state transitions are defined as before. The lexicon contains only five chord classes, so we now only have five emission distributions.

6.3 The Second Model

Let's look at those chord classes again. In the chord grammar, a chord class groups together chord types. Consider the class X^7, for chords rooted on C. This contains, among others, the two chords C^7 and C^{aug7}. These two chords contain different notes: C^7 includes a G, C^{aug7} a G♯. These two notes are unlikely to be generated at the same time, but the model can't capture this if it's generating them independently.

Chord Classes Are Too General

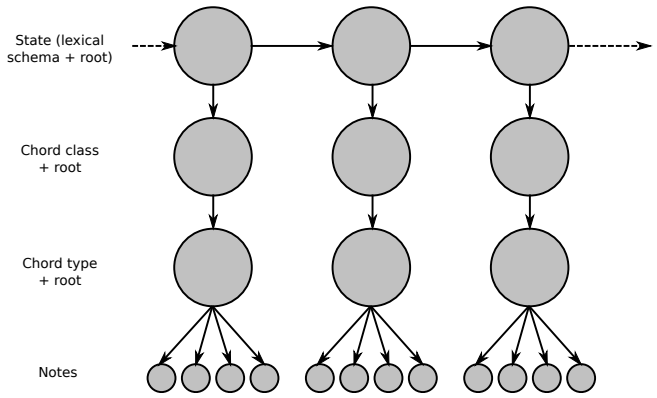
- In chord grammar, a chord class groups together chord types

  X^7 with root C $= \{\ldots, C^7, C^{aug7}, \ldots\}$

- C^7 includes a G
- C^{aug7} includes a G♯
- G and G♯ shouldn't be generated independently
- Chord classes are too general to condition emission distributions on
- We should condition on *chord type*

These chord classes appear to be too general to condition our emission distribution on. In fact, to make an independence assumption no stronger than that in R&S, we should condition on chord type.

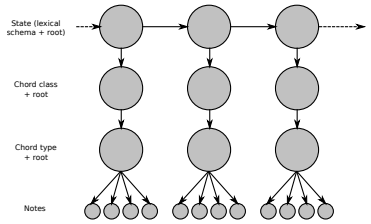
The Second Model



- Chord types shared by chord classes

We now have a further level in the model to represent chord type. The chord classes must now generate chord types according to some distribution. Note that a chord type may appear in multiple chord classes.

The Second Model

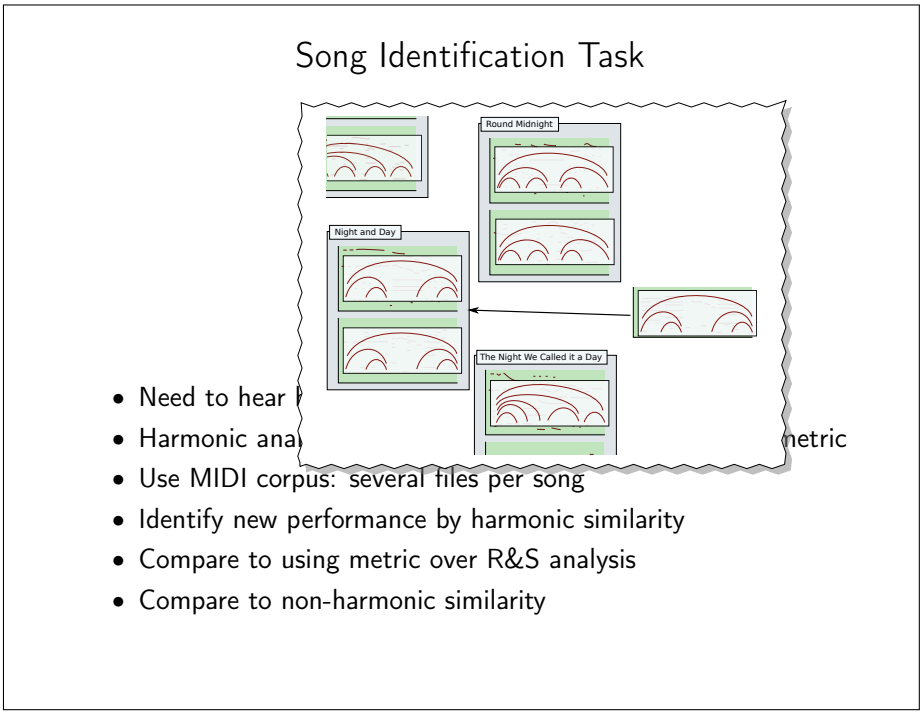


- Emission distribution for each tetrad
- Can initialize as R&S did
- Initialization of transition distributions will become important
- Use labelled corpus

It would make sense in this model to have a separate chord type state for each tetrad type. This gives us a good way to initialize the emission distributions, as in R&S – we can simply set the notes of the tetrad to have a high probability.

We can expect initialization of the transition distribution to become important in this model. But that's ok, since we already have a labeled corpus from which we can gather statistics about the transitions between categories.

6.4 Song Identification



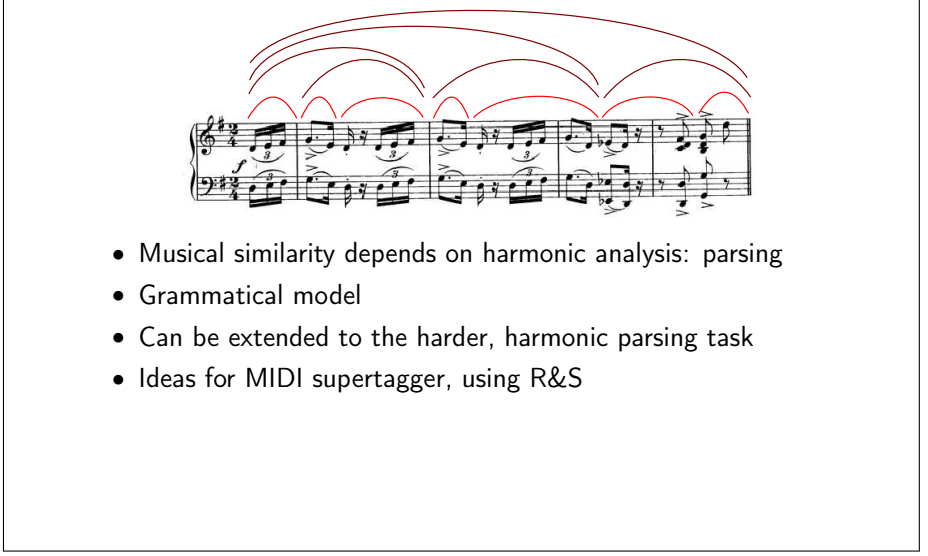
If one of these MIDI supertagging models does a reasonably good job at suggesting categories, we can combine it with the parsing model we already have. The resulting model can then be applied to a music querying task, like the song identification task I discussed on p. 4.

Recall from the beginning that one part of recognising a song is evaluating its harmonic similarity to performances we've heard before and that in some circumstances this may be the main cue. I've discussed an approach to performing automatic harmonic analysis to identify harmonic structure. The output of this process could be used as a signal over which we could define a harmonic similarity metric.

Consider again the song identification task. Let's say we have a corpus of transcriptions of musical performances (we can now assume these are in the form of MIDI files) and we know what songs they are performances of. Now given a new performance, we can identify what song it is by its harmonic similarity to the performances in our database. We can do this using the harmonic similarity metric.

If we do this, we should also compare to using a similar metric define over the analysis produced by R&S. We would also want to compare to other musical similarity metrics not based on harmony.

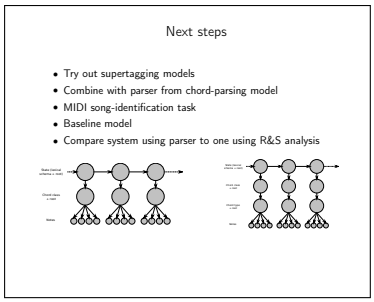
7 Conclusion



I began by discussing musical similarity and how it is, in part, dependent on an analysis of harmonic structure. We need to perform harmonic analysis to reveal this structure. I introduced the task of musical parsing as a way of approaching this analysis.

I considered a simpler problem as a stepping-stone to this difficult problem of harmonic parsing: chord parsing. We've tackled this problem over the last few years and I briefly summarized the techniques we've used to do so. We now have a model that can parse chord sequences to produce a structured harmonic analysis.

This model can be extended to the harder harmonic parsing problem by replacing the chord supertagger by a MIDI supertagger. I have investigated the related model of Raphael & Stoddard (2004) as a basis for a MIDI supertagger. Finally, I suggested some way of building a MIDI supertagger based on this model.



Next, I plan to try out some models along the lines of the MIDI supertaggers I've described here. If these can do a good enough job of suggesting categories, we can combine them with the parser to produce a system to tackle the full harmonic parsing task.

We will apply this model to the song identification task, or a similar MIR task. We will need to establish a baseline for this task, perhaps using non-harmonic metrics or simpler harmonic metrics, including a metric defined over the R&S analysis.

References

Hockenmaier, J. (2001). Statistical parsing for CCG with simple generative models. In *Association for Computational Linguistics 39th annual meeting and 10th conference of the European Chapter*, vol. 39, (pp. 7–12).

Raphael, C., & Stoddard, J. (2004). Functional harmonic analysis using probabilistic models. *Computer Music Journal*, *28*(3), 45–52.