A Formal Framework for Linguistic Annotation

Steven Bird and Mark Liberman

August 13, 1999

Abstract

'Linguistic annotation' covers any descriptive or analytic notations applied to raw language data. The basic data may be in the form of time functions – audio, video and/or physiological recordings – or it may be textual. The added notations may include transcriptions of all sorts (from phonetic features to discourse structures), part-of-speech and sense tagging, syntactic analysis, 'named entity' identification, co-reference annotation, and so on. While there are several ongoing efforts to provide formats and tools for such annotations and to publish annotated linguistic databases, the lack of widely accepted standards is becoming a critical problem. Proposed standards, to the extent they exist, have focused on file formats. This paper focuses instead on the logical structure of linguistic annotations. We survey a wide variety of existing annotation formats and demonstrate a common conceptual core, the *annotation graph*. This provides a formal framework for constructing, maintaining and searching linguistic annotations, while remaining consistent with many alternative data structures and file formats.

1 Introduction

In the simplest and commonest case, 'linguistic annotation' is an orthographic transcription of speech, time-aligned to an audio or video recording. Other central examples include morphological analysis, part-of-speech tagging and syntactic bracketing; phonetic segmentation and labeling; annotation of disfluencies, prosodic phrasing, intonation, gesture, and discourse structure; marking of co-reference, 'named entity' tagging, and sense tagging; and phrase-level or word-level translations. Linguistic annotations may describe texts or recorded signals. Our focus will be on the latter, broadly construed to include any kind of audio, video or physiological recording, or any combination of these, for which we will use the cover term 'linguistic signals'. However, our ideas also apply to the annotation of texts.

Linguistic annotations have seen increasingly broad use in the scientific study of language, in research and development of language-related technologies, and in language-related applications more broadly, for instance in the entertainment industry. Particular cases range from speech databases used in speech recognition or speech synthesis development, to annotated ethnographic materials, to cartoon sound tracks. There have been many independent efforts to provide tools for creating linguistic annotations, to provide general formats for expressing them, and to provide tools for creating, browsing and searching databases containing them – see [www.ldc.upenn.edu/annotation]. Within the area of speech and language technology development alone, hundreds of annotated linguistic databases have been published in the past fifteen years.

While the utility of existing tools, formats and databases is unquestionable, their sheer variety – and the lack of standards able to mediate among them – is becoming a critical problem. Particular bodies of data are created with

particular needs in mind, using formats and tools tailored to those needs, based on the resources and practices of the community involved. Once created, a linguistic database may subsequently be used for a variety of unforeseen purposes, both inside and outside the community that created it. Adapting existing software for creation, update, indexing, search and display of 'foreign' databases typically requires extensive re-engineering. Working across a set of databases requires repeated adaptations of this kind.

As we survey speech transcription and annotation across many existing 'communities of practice', we observe a rich diversity of concrete format. Various attempts to standardize practice have focused directly on these file formats and on the tags and attributes for describing content. However, we contend that file formats and content specifications are secondary. Instead, we focus on the logical structure of linguistic annotations, since it is here that we observe a striking commonality. We describe a simple formal framework having a practically useful formal structure. This opens up an interesting range of new possibilities for creation, maintenance and search. We claim that essentially all existing annotations can be expressed in this framework. Thus, the framework should provide a useful 'interlingua' for translation among the multiplicity of current annotation formats, and also should permit the development of new tools with broad applicability.

This distinction between data formats and logical structure can be brought into sharp focus by analogy with database systems. Consider the relationship between the abstract notion of a relational algebra, the features of a relational database system, and the characteristics of a particular database. For example, the definition of substantive notions like 'date' does not belong in the relational algebra, though there is good reason for a database system to have a special data type for dates. Moreover, a particular database may incorporate all manner of restrictions on dates and relations among them. The formalization presented here is targeted at the most abstract level: we want to get the annotation formalism right. We assume that system implementations will add all kinds of special-case data types (i.e. types of labels with specialized syntax and semantics). We further assume that particular databases will want to introduce additional specifications.

In the early days of database systems, data manipulation required explicit reference to physical storage in files, and application software had to be custom-built. In the late 1960s, with the development of the so-called "three-level architecture", database functionalities were divided into three levels: physical, logical and external. Here, we apply the same development to databases of annotated speech. Figure 1 depicts the speech annotation version of the three-level architecture.

This model permits users to create and manipulate annotation data in the way that conforms most closely to their own conception of the structure of the underlying data, to the contingencies of the task at hand, and to individual preference. Furthermore, it is possible to change an implementation at the physical level while leaving the higher levels intact – the *data independence principle*. By adopting this model, the volatile nature of formats and the open-ended issues associated with user interfaces no longer present barriers on the road towards standardization. In fact, a large number of tools will be able to comprehend a large number of formats, so tools can interoperate and formats are translatable. Therefore communities wedded to particular formats or tools are not left out in the cold.

Before we embark on our survey, a terminological aside is necessary. As far as we are aware, there is no existing cover term for the kinds of transcription, description and analysis that we address here. 'Transcription' may refer to the use of ordinary orthography, or a phonetic orthography; it can plausibly be extended to certain aspects of prosody ('intonational transcription'), but not to other kinds of analysis (morphological, syntactic, rhetorical or discourse structural, semantic, etc). One does not talk about a 'syntactic transcription', although this is at least as determinate a representation of the speech stream as is a phonetic transcription. 'Coding' has been used by social scientists to mean something like 'the assignment of events to stipulated symbolic categories,' as a generalization of the ordinary language meaning associated with translating words and phrases into references to a shared, secret code book. It would be idiosyncratic and confusing (though conceptually plausible) to refer to ordinary orthographic transcription in this way. The term 'markup' has come to have a specific technical meaning, involving the addition of typographical or structural information to a document.

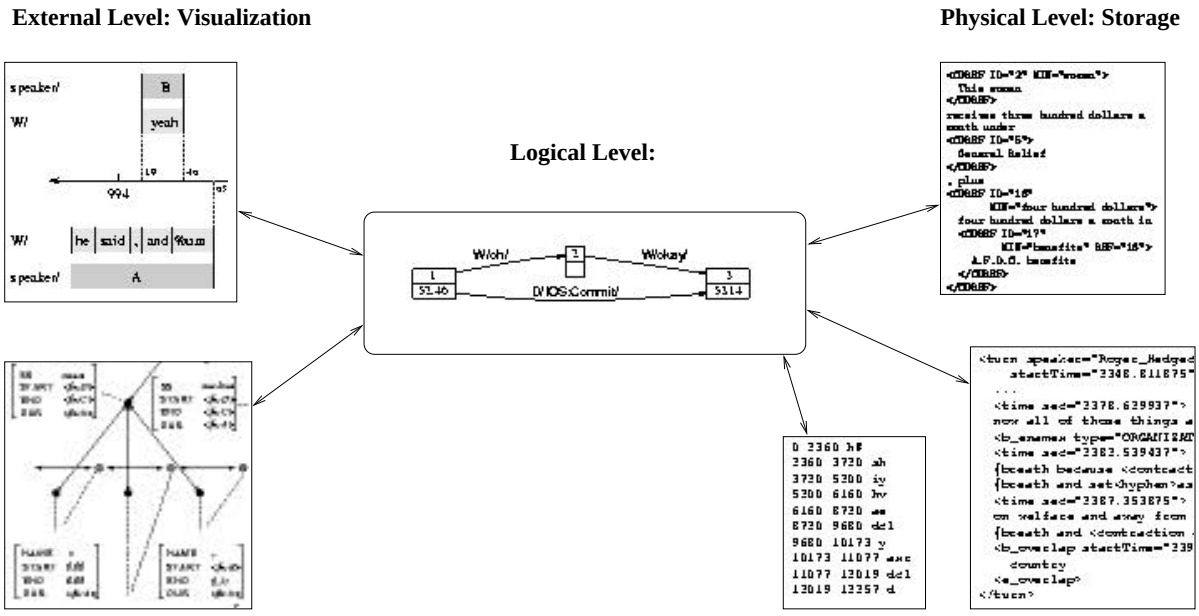

Figure 1: Three-Level Architecture for Speech Annotation

In ordinary language, 'annotation' means a sort of commentary or explanation (typically indexed to particular portions of a text), or the act of producing such a commentary. Like 'markup', this term's ordinary meaning plausibly covers the non-transcriptional kinds of linguistic analysis, such as the annotation of syntactic structure or of co-reference. Some speech and language engineers have begun to use 'annotation' in this way, but there is not yet a specific, widely-accepted technical meaning. We feel that it is reasonable to generalize this term to cover the case of transcribing speech, by thinking of 'annotation' as the provision of any symbolic description of particular portions of a pre-existing linguistic object. If the object is a speech recording, then an ordinary orthographic transcription is certainly a kind of annotation in this sense – though it is one in which the amount of critical judgment is small.

In sum, 'annotation' is a reasonable candidate for adoption as the needed cover term. The alternative would be to create a neologism ('scription'?). Extension of the existing term 'annotation' seems preferable to us.

2 Existing Annotation Systems

In order to justify our claim that essentially all existing linguistic annotations can be expressed in the framework that we propose, we need to discuss a representative set of such annotations. In addition, it will be easiest to understand our proposal if we motivate it, piece by piece, in terms of the logical structures underlying existing annotation practice.

This section reviews several bodies of annotation practice, with a concrete example of each. For each example, we show how to express its various structuring conventions in terms of our 'annotation graphs', which are networks consisting of nodes and arcs, decorated with time marks and labels. Following the review, we shall discuss some general architectural issues (§3), give a formal presentation of the 'annotation graph' concept (§4). The paper concludes in §5 with an evaluation of the proposed formalism and a discussion of future work.

The annotation models to be discussed in detail are TIMIT [17], Partitur [32], CHILDES [26], LACITO [28], LDC Telephone Speech, NIST UTF [30], Switchboard [19], MUC-7 Coreference [23]. Three general purpose models will also be discussed in brief: Emu [14], Festival [37], MATE [12]. These models are widely divergent in type and

```
train/dr1/fjsp0/sa1.wrd:
2360 5200 she
5200 9680 had
9680 11077 your
11077 16626 dark
16626 22179 suit
22179 24400 in
24400 30161 greasy
30161 36150 wash
36720 41839 water
41839 44680 all
44680 49066 year
```

```
train/dr1/fjsp0/sa1.phn:
0 2360 h#
2360 3720 sh
3720 5200 iy
5200 6160 hv
6160 8720 ae
8720 9680 dcl
9680 10173 y
10173 11077 axr
11077 12019 dcl
12019 12257 d
...
```

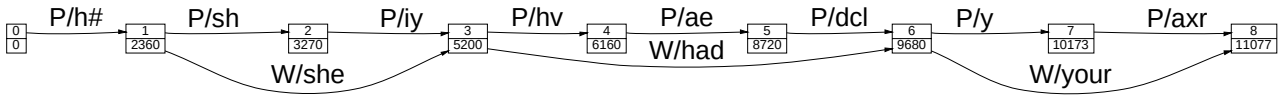

Figure 2: TIMIT Annotation Data and Graph Structure

purpose. Some, like TIMIT, are associated with a specific database, others, like UTF, are associated with a specific linguistic domain (here conversation), while still others, like Festival, are associated with a specific application domain (here, speech synthesis).

Several other systems and formats have been considered in developing our ideas, but will not be discussed in detail. These include Switchboard [19], HCRC MapTask [4], and TEI [39]. The Switchboard and MapTask formats are conversational transcription systems that encode a subset of the information in the LDC and NIST formats cited above. The TEI guidelines for 'Transcriptions of Speech' [39, p11] are also similar in content, though they offer access to a very broad range of representational techniques drawn from other aspects of the TEI specification. The TEI report sketches or alludes to a correspondingly wide range of possible issues in speech annotation. All of these seem to be encompassed within our proposed framework, but it does not seem appropriate to speculate at much greater length about this, given that this portion of the TEI guidelines does not seem to have been used in any published transcriptions to date. Still other models that we are aware of include [3, 22, 31].

Note that there are many kinds of linguistic database that are not linguistic annotations in our sense, although they may be connected with linguistic annotations in various ways. One example is a lexical database with pointers to speech recordings along with transcriptions of those recordings (e.g. HyperLex [7]). Another example would be collections of information that are not specific to any particular stretch of speech, such as demographic information about speakers. We return to such cases in §5.2.

2.1 TIMIT

The TIMIT corpus of read speech was designed to provide data for the acquisition of acoustic-phonetic knowledge and to support the development and evaluation of automatic speech recognition systems. TIMIT was the first annotated speech database to be published, and it has been widely used and also republished in several different forms. It is also especially simple and clear in structure. Here, we just give one example taken from the TIMIT database [17].

The wrd file in Figure 2 combines an ordinary string of orthographic words with information about the starting and ending time of each word, measured in audio samples at a sampling rate of 16 kHz. The path name `train/dr1/fjsp0/sa1.wrd` tells us that this is training data, from 'dialect region 1', from female speaker 'jsp0', containing words and audio sample numbers. The phn file contains a corresponding broad phonetic transcription.

```
KAN: 0 j'a:      ORT: 0 ja         TRL: 0 <A>           MAU: 4160 1119 0 j
KAN: 1 S'2:n@n   ORT: 1 sch"onen   TRL: 0 ja ,          MAU: 5280 2239 0 a:
KAN: 2 d'aNk     ORT: 2 Dank       TRL: 1 sch"onen      MAU: 7520 2399 1 S
KAN: 3 das+      ORT: 3 das        TRL: 1 <:<#Klopfen>  MAU: 9920 1599 1 2:
KAN: 4 vE:r@+    ORT: 4 w"are      TRL: 2 Dank:> ,      MAU: 11520 479 1 n
KAN: 5 z'e:6     ORT: 5 sehr       TRL: 3 das           MAU: 12000 479 1 n
KAN: 6 n'Et      ORT: 6 nett       TRL: 4 w"ar'         MAU: 12480 479 -1 <nib>
```

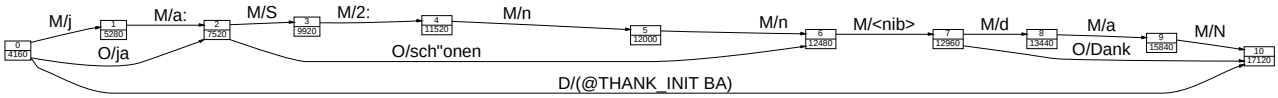

Figure 3: BAS Partitur Annotation Data and Graph Structure

We can interpret each line: <time1> <time2> <label> as an edge in a directed acyclic graph, where the two times are attributes of nodes and the label is a property of an edge connecting those nodes. The resulting annotation graph for the above fragment is shown in Figure 2. Observe that edge labels have the form<type>/<content> where the <type> here tells us what kind of label it is. We have used P for the (phonetic transcription) contents of the .phn file, and W for the (orthographic word) contents of the .wrd file. The top number for each node is an identifier, while the bottom number is the time reference.

2.2 Partitur

The Partitur format of the Bavarian Archive for Speech Signals [32] is founded on the collective experience of a broad range of German speech database efforts. The aim has been to create 'an open (that is extensible), robust format to represent results from many different research labs in a common source.' Partitur is valuable because it represents a careful attempt to present a common low-level core for all of those independent efforts, similar in spirit to our effort here. In essence, Partitur extends and reconceptualizes the TIMIT format to encompass a wide range of annotation types.

The Partitur format permits time-aligned, multi-tier description of speech signals, along with links between units on different tiers which are independent of the temporal structure. For ease of presentation, the example Partitur file will be broken into a number of chunks, and certain details (such as the header) will be ignored. The fragment under discussion is from one of the Verbmobil corpora at the Bavarian Archive of Speech Signals. The KAN tier provides the canonical transcription, and introduces a numerical identifier for each word to serve as an anchor for all other material. Tiers for orthography (ORT), transliteration (TRL), and phonetic segments (MAU) reference these anchors, using the second-last field in each case. The first seven lines of information for each tier are given in Figure 3.

The additional numbers for the MAU tier give offset and duration information. Higher level structure representing dialogue acts refers to extended intervals using contiguous sequences of anchors, as shown below:

```
DAS: 0,1,2 @(THANK_INIT BA)
DAS: 3,4,5,6 @(FEEDBACK_ACKNOWLEDGEMENT BA)
```

The content of the first few words of the ORT (orthography), DAS (dialog act) and MAU (phonetic segment) tiers can apparently be expressed as in Figure 3. Note that we abbreviate the types, using O/ for ORT, D/ for DAS, and M/ for MAU.

```
@Begin                                          *ROS:   yahoo.
@Filename:      boys73.cha                      %snd:   "boys73a.aiff" 7349 8338
@Participants:  ROS Ross Child, MAR Mark Child, *FAT:   you got a lot more to do # don't you?
                FAT Brian Father, MOT Mary Mother %snd: "boys73a.aiff" 8607 9999
@Date:  4-APR-1984                              *MAR:   yeah.
@Age of ROS:    6;3.11                          %snd:   "boys73a.aiff" 10482 10839
@Sex of ROS:    Male                            *MAR:   because I'm not ready to go to
@Birth of ROS:  25-DEC-1977                             <the bathroom> [>] +/.
@Age of MAR:    4;4.15                          %snd:   "boys73a.aiff" 11621 13784
@Birth of MAR:  19-NOV-1979                     ...
@Sex of MAR:    male
@Situation:     Room cleaning
```

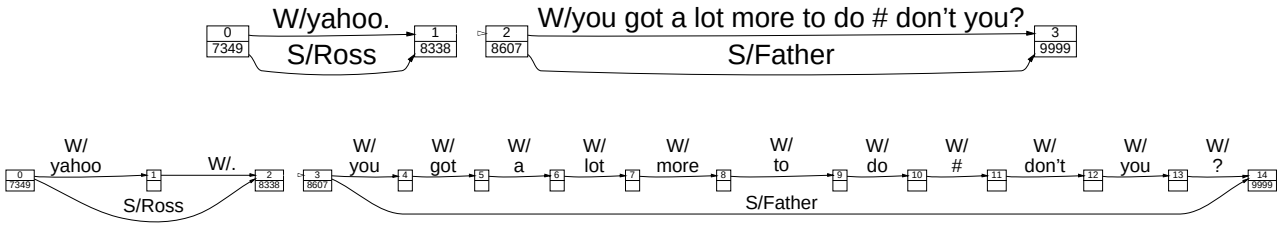

Figure 4: CHILDES Annotation Data and Graph Structure

2.3 CHILDES

With its extensive user base, tools and documentation, and its coverage of some two dozen languages, the Child Language Data Exchange System, or CHILDES, represents the largest scientific – as opposed to engineering – enterprise involved in our survey. The CHILDES database includes a vast amount of transcript data collected from children and adults who are learning languages [26]. All of the data are transcribed in the so-called 'CHAT' format; a typical instance is provided by the opening fragment of a CHAT transcription shown in Figure 4.

The %snd lines, by the conventions of this notation, provide times for the previous transcription lines, in milliseconds relative to the beginning of the referenced file. The first two lines of this transcript might then be represented as the first graph in Figure 4. However, this representation treats entire phrases as atomic arc labels, complicating indexing and search. We favor the representation in the second graph in Figure 4, where labels have uniform ontological status regardless of the presence vs. absence of time references. Observe that most of the nodes in the second version *could* have been given time references in the CHAT format but were not. Our approach maintains the same topology regardless of the sparseness of temporal information.

Some of the tokens of the transcript, i.e. the punctuation marks, do not reference stretches of time in the same way that orthographic words do. Accordingly, they may be given a different type, and/or assigned to an instant rather than a period (by giving the two nodes on either side of the punctuation mark the same time reference; see§3.1).

2.4 LACITO Linguistic Data Archiving Project

LACITO – Langues et Civilisations à Tradition Orale – is a CNRS organization concerned with research on unwritten languages. The LACITO Linguistic Data Archiving Project was founded to conserve and distribute the large quantity of recorded, transcribed speech data collected by LACITO members over the last three decades [28]. The annotation model uses XML, and different XSL stylesheets provide a variety of useful views on the base data.

In this section we discuss a transcription for an utterance in Hayu, a Tibeto-Burman language of Nepal. The gloss and free translation are in French. Consider the XML annotation data and the graph structure in Figure 5. Here we

```
<HEADER> <TITLE>Deux s?x0153;urs.</TITLE> <SOUNDFILE href="SOEURS.mp2"/> </HEADER>
<BODY lang="hayu">
  <S id="s1"> <AUDIO start="2.3656" end="7.9256"/>
    <TRANSCR> <W><FORM>nakpu</FORM><GLS>deux</GLS></W>
              <W><FORM>nonotso</FORM><GLS>s?x0153;urs</GLS></W>
              <W><FORM>si?x014b;</FORM><GLS>bois</GLS></W>
              <W><FORM>pa</FORM><GLS>faire</GLS></W>
              <W><FORM>la?x0294;natshem</FORM><GLS>allrent(D)</GLS></W>
              <W><FORM>are</FORM><GLS>dit.on</GLS></W>
              <PONCT>.</PONCT> </TRANSCR>
    <TRADUC lang="Francais">On raconte que deux soeurs allrent chercher du bois.</TRADUC>
    <TRADUC lang="Anglais">They say that two sisters went to get firewood.</TRADUC>
  </S>
...
```

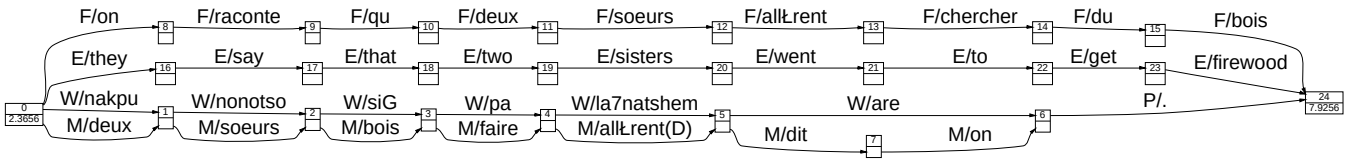

Figure 5: LACITO Annotation Data and Graph Structure

---

have three types of edge labels: `W/` for the wordforms of the Hayu story; `G/` for the gloss, and `F/`, `E/` for phrasal translations into French and English. In this example, the time references (which are in seconds) are again given only at the beginning and end of the phrase, as required by the LACITO format. Nevertheless, the individual Hayu words have temporal extent and one might want to indicate that in the annotation. Observe that there is no meaningful way of assigning time references to word boundaries in the phrasal translation, or for the boundary in the gloss for `dit.on`. Thus the omission of time references may happen because the times are simply unknown, as in the lower half of Figure 5, or are intrinsically un-knowable, as in the upper half of Figure 5.

2.5 LDC Telephone Speech Transcripts

The Linguistic Data Consortium (LDC) is an open consortium of universities, companies and government research laboratories, hosted by the University of Pennsylvania, that creates, collects and publishes speech and text databases, lexicons, and similar resources. Since its foundation in 1992, it has published some 150 digital databases, most of which contain material that falls under our definition of 'linguistic annotation.'

The LDC-published CALLHOME corpora include digital audio, transcripts and lexicons for telephone conversations in several languages [`www.ldc.upenn.edu /Catalog/LDC96S46.html`]. The corpora are designed to support research on speech recognition algorithms. The transcripts exhibit abundant overlap between speaker turns in two-way telephone conversations.

Figure 6 gives a typical fragment of an annotation. Each stretch of speech consists of a begin time, an end time, a speaker designation ('A' or 'B' in the example below), and the transcription for the cited stretch of time. Observe that speaker turns may be partially or totally overlapping.

Long turns (e.g. the period from 972.46 to 989.56 seconds) were broken up into shorter stretches for the convenience of the annotators. Thus this format is ambiguous as to whether adjacent stretches by the same speaker should be considered parts of the same unit, or parts of different units – in translating to an annotation graph representation, either choice could be made. However, the intent is clearly just to provide additional time references within long turns, so the most appropriate choice seems to be to merge abutting same-speaker structures while retaining the additional time-marks.

```
962.68 970.21 A: He was changing projects every couple of weeks and he
  said he couldn't keep on top of it. He couldn't learn the whole new area
968.71 969.00 B: %mm.
970.35 971.94 A: that fast each time.
971.23 971.42 B: %mm.
972.46 979.47 A: %um, and he says he went in and had some tests, and he
  was diagnosed as having attention deficit disorder. Which
980.18 989.56 A: you know, given how he's how far he's gotten, you know,
  he got his degree at &Tufts and all, I found that surprising that for
  the first time as an adult they're diagnosing this. %um
989.42 991.86 B: %mm. I wonder about it. But anyway.
991.75 994.65 A: yeah, but that's what he said. And %um
994.19 994.46 B: yeah.
995.21 996.59 A: He %um
996.51 997.61 B: Whatever's helpful.
997.40 1002.55 A: Right. So he found this new job as a financial
  consultant and seems to be happy with that.
1003.14 1003.45 B: Good.
```

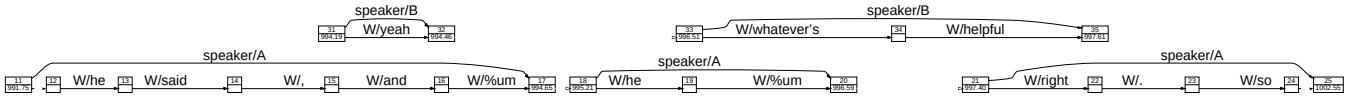

Figure 6: Graph Structure for LDC Telephone Speech Example

A section of this annotation including an example of total overlap is represented in annotation graph form in the lower half of Figure 6. The turns are attributed to speakers using the`spkr/` type. All of the words, punctuation and disfluencies are given the`W/` type, though we could easily opt for a more refined version in which these are assigned different types. Observe that the annotation graph representation preserves the non-explicitness of the original file format concerning which of speaker A's words overlap which of speaker B's words. Of course, additional time references could specify the overlap down to any desired level of detail (including to the level of phonetic segments or acoustic events if desired).

2.6 NIST Universal Transcription Format

The US National Institute of Standards and Technology (NIST) has developed a set of annotation conventions 'intended to provide an extensible universal format for transcription and annotation across many spoken language technology evaluation domains' [30]. This 'Universal Transcription Format' (UTF) was based on the LDC Broadcast News format. A key design goal for UTF was to provide an SGML-based format that would cover both the LDC broadcast transcriptions and also various LDC-published conversational transcriptions, while also providing for plausible extensions to other sorts of material.

A notable aspect of UTF is its treatment of overlapping speaker turns, which are marked with<`b_overlap`> (begin overlap) and <`e_overlap`> (end overlap) tags. Figure 7 contains a fragment of UTF, taken from the Hub-4 1997 evaluation set.

Observe that there are two speaker turns, where the first speaker's utterance of 'country' overlaps the second speaker's utterance of 'well I'. Note that the time attributes for overlap are not required to coincide, since they are aligned to 'the most inclusive word boundaries for each speaker turn involved in the overlap'. The coincidence of end times here is probably an artifact of the system used to create the annotations.

The structure of overlapping turns can be represented using annotation graphs as shown in Figure 7. Each speaker turn is a separate connected subgraph, disconnected from other speaker turns. The time courses of independent

```
<turn speaker="Roger_Hedgecock" spkrtype="male" dialect="native"
    startTime="2348.811875" endTime="2391.606000" mode="spontaneous" fidelity="high">
  ...
  <time sec="2378.629937">
  now all of those things are in doubt after forty years of democratic rule in
  <b_enamex type="ORGANIZATION">congress<e_enamex>
  <time sec="2382.539437">
  {breath because <contraction e_form="[you=>you][’ve=>have]">you’ve got quotas
  {breath and set<hyphen>asides and rigidities in this system that keep you
  <time sec="2387.353875">
  on welfare and away from real ownership
  {breath and <contraction e_form="[that=>that][’s=>is]">that’s a real problem in this
  <b_overlap startTime="2391.115375" endTime="2391.606000">country<e_overlap>
</turn>
<turn speaker="Gloria_Allred" spkrtype="female" dialect="native"
    startTime="2391.299625" endTime="2439.820312" mode="spontaneous" fidelity="high">
  <b_overlap startTime="2391.299625" endTime="2391.606000">well i<e_overlap>
  think the real problem is that %uh these kinds of republican attacks
  <time sec="2395.462500">
  i see as code words for discrimination
  ...
</turn>
```

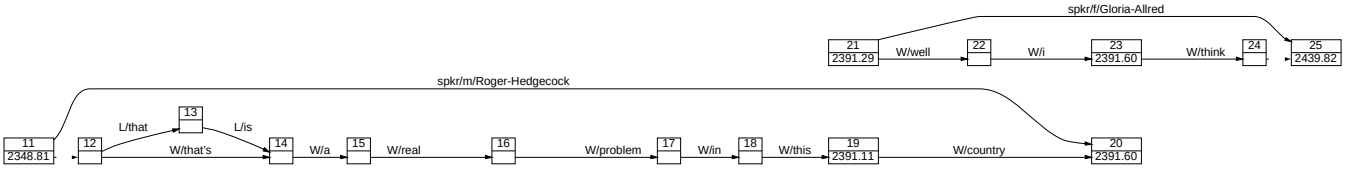

Figure 7: UTF Annotation Data and Graph Structure

utterances are logically asynchronous, and so we prefer not to convolve them into a single stream, as the SGML representation does. Observe that the information about overlap is now implicit in the time references. Partial word overlap can also be represented if necessary. This seems like the best choice in general, since there is no necessary logical structure to conversational overlaps – at base, they are just two different actions unfolding over the same time period. The cited annotation graph structure is thus less explicit about word overlaps than the UTF file[1].

Of course, the same word-boundary-based representation of overlapping turns could also be expressed in annotation graph form, by allowing different speakers' transcripts to share certain nodes (representing the word boundaries at which overlaps start or end). We do not suggest this, since it seems to us to be based on an inappropriate model of overlapping, which will surely cause trouble in the end.

Note the use of the `L/` 'lexical' type to include the full form of a contraction. The UTF format employed special syntax for expanding contractions. No additional ontology was needed in order to do this in the annotation graph. (A query to find instances of `W/that` or `L/that` would simply disjoin over the types.)

Note also that it would have been possible to replicate the type system, replacing `W/` with `W1/` for 'speaker 1' and `W2/` for 'speaker 2'. However, we have chosen instead to attribute material to speakers using the `spkr/` type on an arc spanning an entire turn. The disconnectedness of the graph structure means there can be no ambiguity about the attribution of each component arc to a speaker.

As we have argued, annotation graphs of the kind shown in Figure 7 are actually more general and flexible than the UTF files they model. The UTF format imposes a linear structure on the speaker turns and assumes that overlap only occurs at the periphery of a turn. In contrast, the annotation graph structure is well-behaved for partial word overlap, and it scales up naturally and gracefully to the situation where multiple speakers are talking simultaneously (e.g. for transcribing a radio talk-back show with a compere, a telephone interlocutor and a panel of discussants). It also works for arbitrary kinds of overlap (e.g. where one speaker turn is fully contained inside another), as discussed in the previous section.

2.7 Switchboard extensions

The Switchboard corpus of conversational speech [19] began with the three basic levels: conversation, speaker turn, and word. Various parts of it have since been annotated for syntactic structure [27], for breath groups and disfluencies [35], for speech act type [24, 25], and for phonetic segments [20]. These various annotations have been done as separate efforts, and presented in formats that are fairly easy to process one-by-one, but difficult to compare or combine. Figure 8 provides a fragment of a Switchboard conversation, annotated for words, part-of-speech, disfluency and syntactic structure. Observe that punctuation is attached to the preceding word in the case of word and disfluency annotation, while it is treated as a separate element in the part-of-speech and treebank annotation.

Figure 8 also shows the annotation graph for this Switchboard data, corresponding to the interval [21.86, 26.10]. In this graph, word arcs have type `W/`, Treebank arcs have `T/` and disfluency arcs have `DISF/` type. Types for the part-of-speech arcs have been omitted. The graph is represented in two pieces; the lower piece should be interpolated into the upper piece at the position of the dotted arc labeled `X`. Observe that the equivocation about the status of punctuation is preserved in the annotation graph.

[1] However, if a more explicit symbolic representation of overlaps is desired, specifying that such-and-such a stretch of one speaker turn is associated with such-and-such a stretch of another speaker turn, this can be represented in our framework using the inter-arc linkage method described in §3.3.

Aligned Word

```
B 19.44 0.16 Yeah,
B 19.60 0.10 no
B 19.70 0.10 one
B 19.80 0.24 seems
B 20.04 0.02 to
B 20.06 0.12 be
B 20.18 0.50 adopting
B 20.68 0.16 it.
B 21.86 0.26 Metric
B 22.12 0.26 system,
B 22.38 0.18 no
B 22.56 0.06 one's
B 22.86 0.32 very,
B 23.88 0.14 uh,
B 24.02 0.16 no
B 24.18 0.32 one
B 24.52 0.28 wants
B 24.80 0.06 it
B 24.86 0.12 at
B 24.98 0.22 all
B 25.66 0.22 seems
B 25.88 0.22 like.
A 28.44 0.28 Uh,
A 29.26 0.14 the,
A 29.48 0.14 the,
A 29.82 0.10 the
A 29.92 0.34 public
A 30.26 0.06 is
A 30.32 0.22 just
A 30.54 0.14 very
A 30.68 0.68 conservative
A 31.36 0.18 that
A 31.54 0.30 way
A 32.56 0.12 in
A 32.74 0.64 refusing
A 33.60 0.12 to
A 33.72 0.56 change
A 34.94 0.48 measurement
A 35.42 0.62 systems,
A 36.08 0.26 uh,
A 37.04 0.38 money,
A 37.62 0.30 dollar,
A 37.92 0.46 coins,
A 38.38 0.22 anything
A 38.60 0.18 like
A 38.78 0.30 that.
B 39.34 0.10 Yeah
B     *    * [laughter].
A 40.96 0.04 And,
A 41.32 0.04 and,
A 42.28 0.36 and
A 42.88 0.20 it
A     *    * [breathing],
A 43.08 0.16 it
A 43.48 0.46 obviously
A 43.94 0.22 makes
A 44.16 0.14 no
A 44.30 0.36 sense
A 44.66 0.06 that
A 44.72 0.12 we're
A 44.84 0.70 practically
A 46.52 0.32 alone
A 46.84 0.10 in
A 46.94 0.06 the
A 47.00 0.44 world
A 47.44 0.16 in,
A 48.52 0.04 in
A 48.56 0.26 using
A 48.82 0.08 the
A 48.90 0.22 old
A 49.12 0.40 system.
```

Part of Speech

```
====================
[ SpeakerB22/SYM ]
./.
====================

Yeah/UH ,/,
[ no/DT one/NN ]
seems/VBZ to/TO
be/VB adopting/VBG
[ it/PRP ] ./.

[ Metric/JJ system/NN ]
,/,
[ no/DT one/NN ]
's/BES very/RB ,/,
[ uh/UH ] ,/,
[ no/DT one/NN ]
wants/VBZ
[ it/PRP ]
at/IN
[ all/DT ]
seems/VBZ like/IN ./.

====================
[ SpeakerA23/SYM ]
./.
====================

[ Uh/UH ] ,/,
[ the/DT ] ,/,
[ the/DT ] ,/,
[ the/DT public/NN ]
is/VBZ just/RB very/RB
conservative/JJ that/DT
[ way/NN ]
in/IN refusing/VBG
to/TO change/VB
[ measurement/NN
  systems/NNS ]
,/,
[ uh/UH ] ,/,
[ money/NN ] ,/,
[ dollar/NN ] ,/,
[ coins/NNS ] ,/,
[ anything/NN ]
like/IN
[ that/DT ] ./.

====================
[ SpeakerB24/SYM ]
./.
====================

Yeah/UH ./.

====================
[ SpeakerA25/SYM ]
./.
====================

And/CC ,/, and/CC ,/,
and/CC
[ it/PRP ] ,/,
[ it/PRP ]
obviously/RB makes/VBZ
[ no/DT sense/NN ]
that/IN
[ we/PRP ]
're/VBP practically/RB
alone/RB in/IN
[ the/DT world/NN ]
in/IN ,/, in/IN
using/VBG
[ the/DT old/JJ
  system/NN ]
./.
```

Disfluency

```
B.22:   Yeah, / no one seems to be adopting it. /
  Metric system, [ no one's very, + F uh,  no one wants ]
  it at all seems like. /
A.23:   F Uh,  [ [ the, + the, ] + the ]
  public is just very conservative that way in
  refusing to change measurement systems,
  F uh,  money, dollar, coins, anything like that. /
B.24:   Yeah <laughter>. /
A.25:   [ [ C And,  +  C and,  ] + C and  ]
  [ it + <breathing>,  it ] obviously makes no sense
  that we're practically alone in the world [ in, + in ]
  using the old system. /
```

Treebank

```
((CODE SpeakerB22 .))
((INTJ Yeah , E_S))
((S (NP-SBJ-1 no one)
    (VP seems
        (S (NP-SBJ *-1)
           (VP to (VP be (VP adopting (NP it)))))) . E_S))
((S (NP-TPC Metric system) ,
    (S-TPC-1 (EDITED (RM [)
                     (S (NP-SBJ no one)
                        (VP 's (ADJP-PRD-UNF very))) ,
                     (IP +)) (INTJ uh) ,
             (NP-SBJ no one)
             (VP wants (RS ]) (NP it) (ADVP at all)))
    (NP-SBJ *)
    (VP seems (SBAR like (S *T*-1))) . E_S))
((CODE SpeakerA23 .))
((S (INTJ Uh) ,
    (EDITED (RM [)
            (EDITED (RM [) (NP-SBJ-UNF the) , (IP +))
            (NP-SBJ-UNF the) , (RS ]) (IP +))
    (NP-SBJ-1 the (RS ]) public)
    (VP is
        (ADVP just)
        (ADJP-PRD very conservative)
        (NP-MNR that way)
        (PP in
            (S-NOM (NP-SBJ-2 *-1)
                   (VP refusing
                       (S (NP-SBJ *-2)
                          (VP to
                              (VP change
                                  (NP (NP measurement systems) ,
                                      (INTJ uh) , (NP money) ,
                                      (NP dollar) , (NP coins) ,
                                      (NP (NP anything)
                                          (PP like
                                              (NP that))))))))))) . E_S))
((CODE SpeakerB24 .))
((INTJ Yeah . E_S))
((CODE SpeakerA25 .))
((S (EDITED (RM [)
            (EDITED (RM [) And , (IP +)) and , (RS ]) (IP +)) and (RS ])
    (EDITED (RM [) (NP-SBJ it) (IP +) ,)
    (NP-SBJ (NP it)
            (SBAR *EXP*-1))
    (RS ])
    (ADVP obviously)
    (VP makes
        (NP no sense)
        (SBAR-1 that
                (S (NP-SBJ-2 we)
                   (VP 're
                       (ADVP practically) (ADJP-PRD alone)
                       (PP-LOC in (NP the world))
                       (EDITED (RM [) (PP-UNF in) , (IP +))
                       (PP in (RS ])
                           (S-NOM (NP-SBJ *-2)
                                  (VP using
                                      (NP the old system)))))))) . E_S))
```

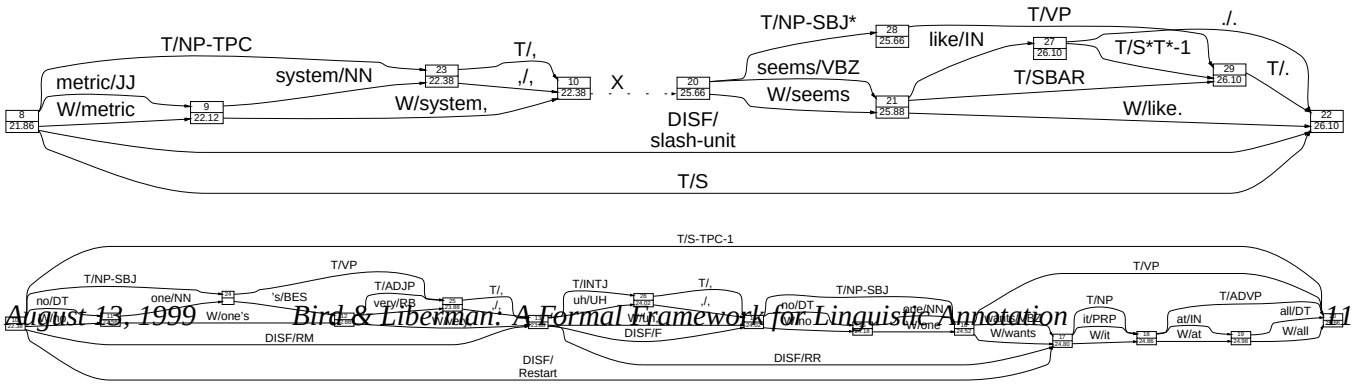

Figure 8: Multiple Annotations of the Switchboard Corpus, With Annotation Graph

```
<COREF ID="2" MIN="woman">This woman</COREF> receives three hundred dollars a month under
<COREF ID="5">General Relief</COREF>, plus <COREF ID="16" MIN="four hundred dollars"> four
hundred dollars a month in <COREF ID="17" MIN="benefits" REF="16">A.F.D.C.  benefits</COREF></COREF>
for <COREF ID="9" MIN="son"><COREF ID="3" REF="2">her</COREF>son</COREF>, who is
<COREF ID="10" MIN="citizen" REF="9">a U.S. citizen</COREF>.
<COREF ID="4" REF="2">She</COREF>'s among <COREF ID="18" MIN="aliens">an estimated five hundred
illegal aliens on <COREF ID="6" REF="5">General Relief</COREF> out of
<COREF ID="11" MIN="population"><COREF ID="13" MIN="state">the state</COREF>'s total illegal
immigrant population of <COREF ID="12" REF="11"> one hundred thousand </COREF></COREF></COREF>
<COREF ID="7" REF="5">General Relief</COREF> is for needy families and unemployable adults who
don't qualify for other public assistance.  Welfare Department spokeswoman Michael Reganburg says
<COREF ID="15" MIN="state" REF="13">the state</COREF> will save about one million dollars a year
if <COREF ID="20" MIN="aliens" REF="18">illegal aliens</COREF> are denied
<COREF ID="8" REF="5">General Relief</COREF>.
```

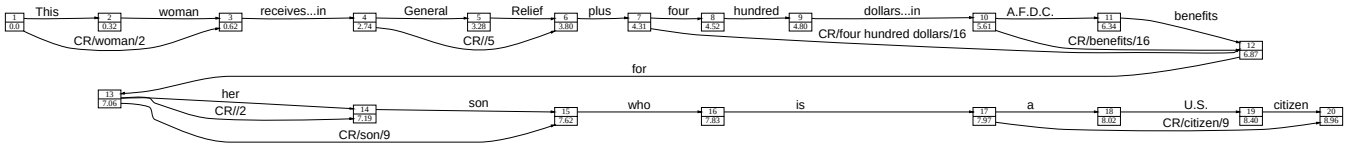

Figure 9: Annotation Graph for Coreference Example

2.8 MUC-7 Coreference Annotation

The MUC-7 Message Understanding Conference specified tasks for information extraction, named entity and coreference. Coreferring expressions are to be linked using SGML markup with `ID` and `REF` tags [23]. Figure 9 is a sample of text from the Boston University Radio Speech Corpus [`www.ldc.upenn.edu/Catalog/LDC96S36.html`], marked up with coreference tags.

Noun phrases participating in coreference are wrapped with `<coref>...</coref>` tags, which can bear the attributes `ID`, `REF`, `TYPE` and `MIN`. Each such phrase is given a unique identifier, which may be referenced by a `REF` attribute somewhere else. Our example contains the following references: $3 \rightarrow 2$, $4 \rightarrow 2$, $6 \rightarrow 5$, $7 \rightarrow 5$, $8 \rightarrow 5$, $12 \rightarrow 11$, $15 \rightarrow 13$, $17 \rightarrow 16$. The `TYPE` attribute encodes the relationship between the anaphor and the antecedent. Currently, only the identity relation is marked, and so coreferences form an equivalence class. Accordingly, our example contains the following equivalence classes: $\{2, 3, 4\}$, $\{5, 6, 7, 8\}$, $\{11, 12\}$, $\{13, 15\}$. $\{16, 17\}$. In our graph representation we choose the first number from each of these sets as the identifier for the equivalence class.

2.9 General Purpose Models

Several systems, constructed for a specific domain, are nevertheless sufficiently configurable that they can serve as general purpose models for linguistic annotation. The BAS Partitur model is a case in point (see §2.2). Here we consider three general systems, Emu, Festival and MATE.

The Emu speech database system [14] was designed to support speech scientists who work with large collections of speech data, such as the Australian National Database of Spoken Language [`andosl.anu.edu.au/andosl`]. Emu permits hierarchical annotations arrayed over any number of levels, where each level is a linear ordering. The levels and their relationships are fully customizable.

The Festival speech synthesis system [37, 38] uses a data structure called a 'heterogeneous relation graph', which is a collection of binary relations over attribute-value matrices (AVMs). Each matrix describes the local properties of some linguistic unit, such as a segment, a syllable, or a syntactic phrase. The value of an attribute could be atomic

(such as a binary feature or a real number), or another (nested) AVM, or a function. Functions have the ability to traverse one or more binary relations and incorporate values from other AVMs. For example, if duration was an attribute of a syllable, its value would be a function subtracting the start time of the first dominated segment from the end time of the last dominated segment.

MATE is a dialogue annotation workbench based on XML and XSL [12]. MATE's formal annotation model appears to be XML itself. Each layer of annotation is stored in a separate XML file, where a layer could be a sequence of words or nested tags representing a hierarchy. Pieces of annotation reference each other using hyperlinks; a tag can have a sequence of hyperlinks to represent a one-to-many relationship. MATE provides two ways to represent constituency – nested tags (within a layer) and hyperlinks (between layers). The structure of layers and their possible interrelationships is fully configurable.

While these three models have important differences, all emphasize the hierarchical constituent structure of annotations. Time offsets into the underlying data are stored but temporal information is very much in the background, and checking the temporal well-formedness of the annotation requires navigating a potentially complex network of multiple intersecting hierarchies. In all three cases, this checking task is simplified by storing the temporal information on one level only (and possibly propagating the information outwards from this level). However this approach is inflexible with respect to the typical mode of corpus reuse: for example, a corpus which has been time aligned at the sentence level (e.g. CHILDES, LACITO), may be subsequently annotated at the word or phoneme level, reusing the sentence level time offsets. Maintaining temporal consistency is also expensive and does not scale well as annotations get large (i.e. it has worse than linear time complexity; $O(n^2)$ for naive algorithms [2]). As far as we are aware, the three models in question have yet to be applied to large annotations.

A second similarity of the three models is that they are all able to express multiple intersecting hierarchies. For example, the same word string may be parsed into both a syntactic and a prosodic hierarchy. A given word in such a multi-hierarchy might have internal morphological and syllabic structure that are incommensurate. These independent morphological and syllabic constituents may nevertheless dominate a single stream of phonetic segments. Since nested bracketings are manifestly inadequate for such configurations, these multi-hierarchies must be represented using pointer structures. Herein lies another problem. In order for a constituent to be linked into the hierarchy, it must be addressable. Accordingly, every entity must carry its own unique identifier, just in case some other entity needs to reference it. This situation presents no problem for the native tools. However, people who want to write external programs for creating, manipulating, or querying such corpora are presented with the rather daunting challenge of ensuring that their programs are well-behaved with respect to the potentially intricate network of object identifiers. In this respect these models inherit some general problems of object oriented databases: restructuring of data can be extremely difficult, and query optimization technology is still in its infancy.

Another similarity of the models is their expressiveness. However, this means that it will probably be difficult to come up with implementations (of the unrestricted model) which remain efficient as the annotations and the corpora become large; it will be difficult to apply well-understood technologies to transform the annotations (e.g. FSTs); even simple checks, such as for avoiding cycles in the constituency relation, could become prohibitive to compute; other checks might also be hard, such as ensuring that the fringe of any non-terminal is a convex set of terminals after arbitrary edits on the constituency relationships between the non-terminal and its fringe.

We believe that these three problems underline the importance of having a simple formalism which foregrounds the temporal structure of annotations. The graph-based annotation model is not as rich as the above models, and so it lacks their problems. Yet it is sufficiently expressive to represent the diverse range of annotation practice described in §2.

3 Architectural Considerations

A wide range of annotation models have now been considered. Our provision of annotation graphs for each one already gives a foretaste of the formalism we present in§4. However, before presenting the formalism, we want to stand back from the details of the various models, and try to take in the big picture. In this section we describe a wide variety of architectural issues which we believe should be addressed by any general purpose model for annotating linguistic signals.

3.1 Various temporal and structural issues

Partial Information

In the discussion of CHILDES and the LACITO Archiving Project above, there were cases where our graph representation had nodes which bore no time reference. Perhaps times were not measured, as in typical annotations of extended recordings where time references might only be given at major phrase boundaries (c.f. CHILDES). Or perhaps time measurements were not applicable in principle, as for phrasal translations (c.f. the LACITO Archiving Project). Various other possibilities suggest themselves. We might create a segment-level annotation automatically from a word-level annotation by looking up each word in a pronouncing dictionary and adding an arc for each segment, prior to hand-checking the segment annotations and adding time references to the newly created nodes. The annotation should remain well-formed (and therefore usable) at each step in this enrichment process.

Just as the temporal information may be partial, so might the label information. For example, we might label indistinct speech with whatever information is available – 'so-and-so said something here that seems to be two syllables long and begins with a /t/'.

Beyond these two kinds of partiality, there is an even more obvious kind of partiality we should recognize. An annotated corpus might be annotated in a fragmentary manner. Perhaps only 1% of a recording bears on the research question at hand. It should be possible to have a well-formed annotation structure with arbitrary amounts of annotation detail at certain interesting loci, and limited or no detail elsewhere. This is a typical situation in phonetic or sociolinguistic research, where a large body of recordings may be annotated in detail with respect to a single, relatively infrequent phenomenon of interest.

Redundant information

An annotation framework (or its implementation) may also choose to incorporate arbitrary amounts of redundant encoding of structural information. It is often convenient to add redundant links explicitly – from children to parents, from parents to children, from one child to the next in order, and so on – so that a program can navigate the structure in a way that is clearer or more efficient. Although such redundant links can be specified in the basic annotation itself – cf. [37] – they might equally well be added automatically, as part of a compilation or indexing process. In our view, the addition of this often-useful but predictable structure should not be an intrinsic part of the definition of general-purpose annotation structures. We want to distinguish the annotation formalism itself from various enriched data structures with redundant encoding of hierarchical structure, just as we would distinguish it from various indices for convenient searching of labels and label sequences.

3.2 Multiple nodes at a time point

In addition to hierarchical and sequential structure, linguistic signals also exhibit parallel structure. Consider the gestural score notation used to describe the articulatory component of words and phrases (e.g. [10]). A gestural score

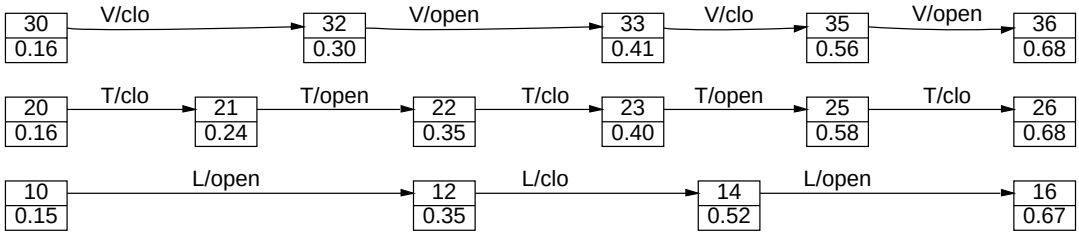

Figure 10: Gestural Score for the Phrase 'ten pin'

---

maps out the time course of the gestural events created by the articulators of the vocal tract. This representation expresses the fact that the articulators move independently and that the segments we observe are the result of particular timing relationships between the gestures. Figure 10 gives an annotation graph for a gestural score. The layers represent the the velum `V/`, the tongue tip `T/` and the lips `L/`.

Observe that nodes 12 and 22 have the same time reference. This alignment is a contingent fact about a particular utterance token. An edit operation which changed the start time of one gesture would usually carry no implication for the start time of some other gesture. Contrast this situation with a hierarchical structure, where, for example, the left boundary of a phrase lines up with the left boundary of its initial word. Changing the time of the phrase boundary should change the time of the word boundary, and vice versa. In the general case, an update of this sort must propagate both upwards and downwards in the hierarchy. In fact, we argue that these two pieces of annotation actually *share* the same boundary: their arcs emanate from a single node. Changing the time reference of that node does not need to propagate anywhere, since the information is already shared by the relevant arcs.

Instants

Even though a linguistic event might have duration, such as the attainment of a pitch target, the most perspicuous annotation may be tied to an instant rather than an interval. Some annotation formalisms (e.g. Emu, Festival, Partitur) provide a way to label instants. The alignment of these instants with respect to other instants or intervals can then be investigated or exploited.

We could extend our graph model to handle instants by introducing labels on the nodes, or by allowing nodes to have self-loops. However, we prefer to give all label information the same ontological status, and we are committed to the acyclic graph model. Therefore we adopt the following three approaches to instants, to be selected as the situation dictates: (i) instants can be treated as arcs between two nodes with the same time reference; or (ii) instants can be treated as short periods, where these are labeled arcs just like any other; or (iii) certain types of labels on periods could be interpreted as referring to the commencement or the culmination of that period.

Overlaps and gaps

As we have seen, annotations are often stratified, where each layer describes a different property of a signal. What are the possible temporal relationships within a given layer? Some possibilities are diagrammed in Figure 11, where a point is represented as a vertical bar, and an interval is represented as a horizontal line between two points.

In the first row of Figure 11, we see a layer which exhaustively partitions the time-flow into a sequence of non-overlapping intervals (or perhaps intervals which overlap just at their endpoints). In the second row we see a layer of discrete instants. The next two rows illustrate the notions of gaps and overlaps. Gaps might correspond to periods of silence, or to periods in between the salient events, or to periods which have yet to be annotated. Overlaps occur between speaker turns in discourse (see Figure 6) or even between adjacent words in a single speech stream (see

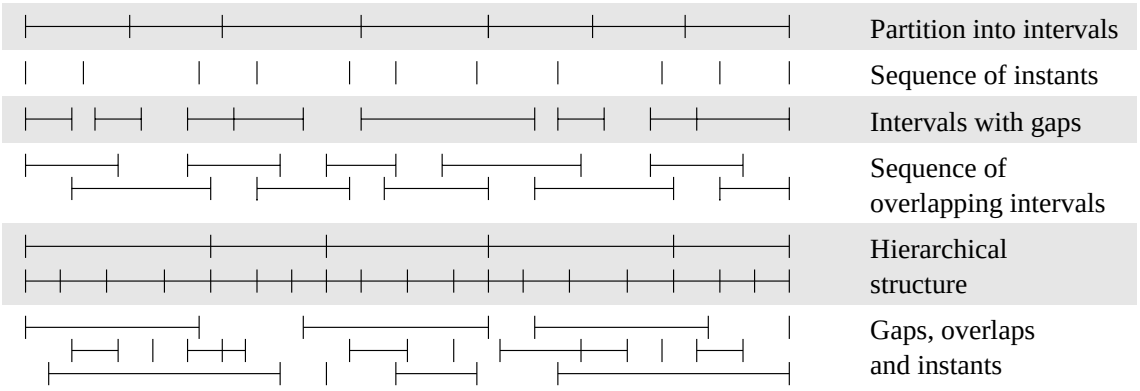

Figure 11: Possible Structures for a Single Layer

Figure 12a). The fifth row illustrates a hierarchical grouping of intervals within a layer. The final row contains an arbitrary set of intervals and instants. We adopt this last option as the most general case for the layer of an annotation. As we shall see, layers themselves will not be treated specially; a layer can be thought of simply as the collection of arcs sharing the same type information.

3.3 Equivalence classes

The arc data of an annotation graph is just a set. Computationally, we can think of it as an associative store – just as in the relational data model where "tuples are identified through a specification of their properties rather than by chasing pointers" [1, 35]. There are cases where this structure appears inadequate, and it seems necessary to enrich the ontology with inter-arc links. Now, these links are simply mappings defined on the arc set. In many cases, including those discussed in this section, the mappings are undirected (or the direction can be inferred) so we can treat them as symmetric relations. Transitivity seems harmless in these cases, and so each mapping can be treated as an equivalence relation.

We consider three cases here, and the solution picks up on the method which was used in §2.8. In §4.4 we show how to enrich the formalism with true identifiers and cross references.

Recall from §3.2 that an annotation graph can contain several independent streams of information, where no nodes are shared between the streams. The temporal extents of the gestures in the different streams are almost entirely asynchronous; any alignments are likely to be coincidences. However, these gestures may still have determinate abstract connections to elements of a phonological analysis. Thus a velar opening and closing gesture may be associated with a particular nasal feature, or with a set of nasal features, or with the sequence of changes from non-nasal to nasal and back again. But these associations cannot usually be established purely as a matter of temporal coincidence, since the phonological features involved are bundled together into other units (segments or syllables or whatever) containing other features that connect to other gestures whose temporal extents are all different. The rules of coordination for such gestures involve phase relations and physical spreading which are completely arbitrary from the perspective of the representational framework.

An example of the arbitrary relationship between the gestures comprising a word is illustrated in Figure 12a. We have the familiar annotation structure (taken from Figure 10), enriched with information about which words license which gestures. In the general case, the relationship between words and their gestures is not predictable from the temporal structure and the type structure alone.

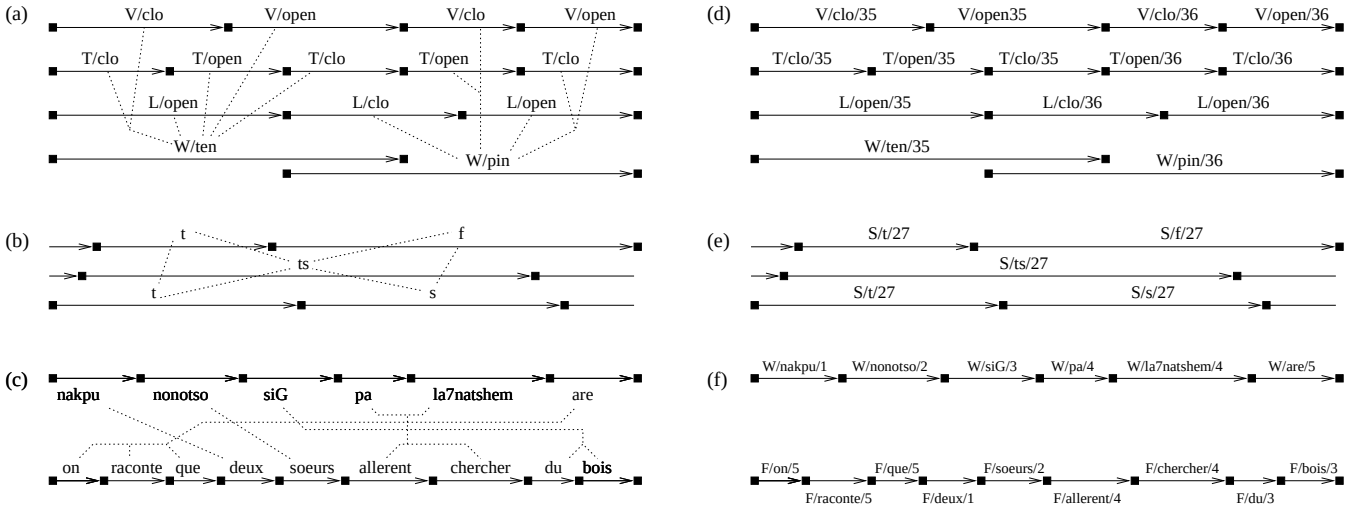

Figure 12: Inter-Arc Linkages Modeled Using Equivalence Classes

The example in Figure 12b shows a situation where we have multiple independent transcriptions of the same data. In this case, the purpose is to compare the performance of different transcribers on identical material. Although the intervals are not synchronized, it should be possible to navigate between corresponding labels.

The final example, Figure 12c, shows an annotation graph based on the Hayu example from Figure 5. We would like to be able to represent the relationship between words of a phrasal translation and the corresponding Hayu words. This would be useful, for example, to study the various ways in which a particular Hayu word is idiomatically translated. (The same linked multi-stream representation is employed in an actual machine translation system [11].) The temporal relationship between linked elements is more chaotic here, and there are examples of one-to-many and many-to-many mappings. In the general case, the words being mapped do not need to be contiguous subsequences.

As stated above, we can treat all of these cases using equivalence classes. Arcs are connected not by referencing one another, but by jointly referencing a particular equivalence class.

For the gestural score in Figure 12a, we assign each arc to an equivalence class, as in Figure 12b. The class names are arbitrary: in this case `35` and `36`. Now we can easily navigate around the set of gestures licensed by a word regardless of their temporal extent. We can use the type information on the existing labels in situations where we care about the directionality of the association. The same method works for the other cases, and the proposed representations are shown in Figure 12d,f. As a consequence of adopting this method, there are now no less than three ways for a pair of arcs to be 'associated': temporal overlap, hierarchy, and a more abstract, atemporal relationship (the equivalence-class linkages). This three-way possibility mirrors the three ways that "autosegmental association" is treated in the phonological literature [8, 9, 6].

3.4 Hierarchical structure

Existing annotated speech corpora always involve a hierarchy of several levels of annotation, even if they do not focus on very elaborate types of linguistic structure. TIMIT has sentences, words and phonetic segments; a broadcast news corpus may have designated levels for shows, stories, speaker turns, sentences and words. Some annotations may express much more elaborate hierarchies, with multiple hierarchies sometimes created for a single underlying body of speech data, such as Switchboard (see§2.7).

In the annotation graph model, annotations are akin to the arcs in so-called 'parse charts' [18, 179ff]. A parse chart is a particular kind of acyclic digraph, which starts with a string of words and then adds a set of arcs representing hypotheses about constituents dominating various substrings. In such a graph, if the substring spanned by arc a_i properly contains the substring spanned by arc a_j, then the constituent corresponding to a_i must dominate the constituent corresponding to a_j (though of course other structures may intervene). Hierarchical relationships are encoded in a parse chart only to the extent that they are implied by this graph-wise inclusion – thus two arcs spanning the same substring are unspecified as to their hierarchical relationship, and arcs ordered by temporal inclusion acquire a hierarchical relationship even when this is not appropriate given the types of those arcs (though a grammar, external to the parse chart for a particular sentence, may settle the matter).

The graph structures implicit in TIMIT's annotation files do not tell us, for the word spelled 'I' and pronounced /ay/, whether the word dominates the phoneme or vice versa; but the structural relationship is implicit in the general relationship between the two types of annotations.

We also need to mention that particular applications in the areas of creation, query and display of annotations may be most naturally organized in ways that motivate a user interface based on a different sort of data structure than the one we are proposing. For instance, it may sometimes be easier to create annotations in terms of tree-like dominance relations rather than chart-like constituent extents, for instance in doing syntactic tree-banking [27]. It may likewise be easier in some cases to define queries explicitly in terms of tree structures. And finally, it may sometimes be more helpful to display trees rather than equivalent annotation graphs – as done by some of the other general purpose annotation models discussed in §2.9. We believe that such user interface issues will vary from application to application, and may even depend on the tastes of individuals in some cases. In any case, decisions about such user interface issues are separable from decisions about the appropriate choice of basic database structures.

3.5 Discontinuous constituency

English lends itself to a description in terms of untangled tree-structures, leaving a few phenomena (adverbials, parentheticals, extraposed clauses, verb-associated particles, and so on) to be dealt with in a way that violates canonical constituency. In some languages, such as Latin, Czech, and Warlpiri, it is common for several constituents to be scrambled up together; the grammatical relations are encoded using case marking. Precisely for this reason, the surface syntax of such languages seems to be best described in terms of dependency relations, as opposed to constituent structures with no constraints on string-tangling. In the present context, the point at issue is the following. To what extent is it necessary for a treebanking representation system to conveniently encode discontinuous constituency?

To date, few corpora have encoded discontinuous constituency (see [34] for an example), and so it would be premature to propose a definitive answer to this question. However, annotation graphs permit two representational possibilities, both using the equivalence class construction. The first possibility amounts to a version of dependency grammar, while the second represents constituency in a manner that reduces to the chart construction in cases where there are no discontinuous constituents. We illustrate the two possibilities using a Latin sentence; see Figure 13.

In the first (dependency grammar) version, each word arc carries two additional fields. The first field identifies the set of dependents of the arc, while the second field identifies the head of the arc. In the second (constituency) version, the span of a non-terminal is the smallest contiguous word string which includes the words of its fringe. Here, the numbers can be used to recover the constituency relation.

3.6 Associations between annotations and files

An 'annotated corpus' is a set of annotation graphs and an associated body of time series data. The time series might comprise one or more audio tracks, one or more video streams, one or more streams of physiological data of

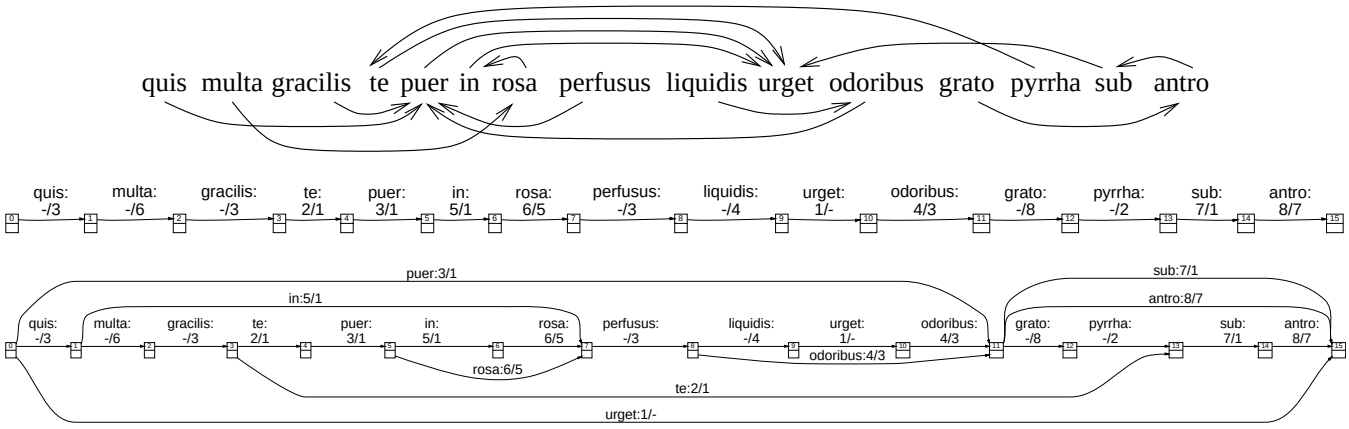

Figure 13: Sentence from Carmina 1.5 (Horace) Showing Dependency Structure, with Two Annotation Graphs

various types, and so forth. The data might be sampled at a fixed rate, or might consist of pairs of times and values, for irregularly spaced times. Different streams will typically have quite different sampling rates. Some streams might be defined only intermittently, as in the case of a continuous audio recording with intermittent physiological or imaging data. This is not an imagined list of conceptually possible types of data – we are familiar with corpora with all of the properties cited.

It is not appropriate for an annotation framework to try to encompass the syntax and semantics of all existing time series file formats. They are simply too diverse and too far from being stable. However, we do need to be able to specify what time series data we are annotating, and how our annotations align with it, in a way that is clear and flexible.

The time series data will be packaged into a set of one or more files. Depending on the application, these files may have some more or less complex internal structure, with headers or other associated information about type, layout and provenance of the data. These headers may correspond to some documented open standard, or they may be embedded in a proprietary system. The one thing that ties all of the time series data together is a shared time base. To use these arbitrarily diverse data streams, we need to be able to line them up time-wise. This shared time base is also the only pervasive and systematic connection such data is likely to have with annotations of the type we are discussing in this paper. We will call this shared time base the “timeline”, and ascribe it formal status in the model as part of the function assigning times to nodes. Arbitrary additional information could be contained in the internal structure of such time references, such as an offset relative to the file’s intrinsic time base (if any), or a specification selecting certain dimensions of vector-valued data.

These timeline names will permit an application to recover the time-series data that corresponds to a given piece of annotation – at least to the extent that the annotation is time-marked and any time-function files have been specified for the cited subgraph(s). Thus if time-marking is provided at the speaker-turn level (as is often the case for published conversational data), then a search for all the instances of a specified word string will enable us to recover usable references to all available time-series data for the turn that contains each of these word strings. The information will be provided in the form of timeline names, signal file names (and types where necessary), time references, and perhaps time offsets; it will be the responsibility of the application (or the user) to resolve these references. If time-marking has been done at the word level, then the same query will enable us to recover a more exact set of temporal references into the same set of files.

The formalization of timelines is presented in §4.2. Our preference is to allow the remaining details of how to define file references to fall outside the formalism. It should be clear that there are simple and natural ways to establish the

sorts of linkages that are explicit in existing types of annotated linguistic database. After some practical experience, it may make sense to try to provide a more formal account of references to external time-series data.

Spatial and image-plane references

We would also like to point out a wider problem for which we do not have any general solution. Although it is not our primary focus, we would like the annotation formalism to be extensible to spatially-specific annotations of video signals and similar data, perhaps by enriching the temporal anchors with spatial and/or image-plane information. Anthropologists, conversation analysts, and sign-language researchers are already producing annotations that are (at least conceptually) anchored not only to time spans but also to a particular spatial or image-plane trajectory through the corresponding series of video frames.

In the case of simple time-series annotations, we are tagging nodes with absolute time references, perhaps offset by a single constant for a given recorded signal. However, if we are annotating a video recording, the additional anchoring used for annotating video sequences will mostly not be about absolute space, even with some arbitrary shift of coordinate origin, but rather will be coordinates in the image plane. If there are multiple cameras, then image coordinates for each will differ, in a way that time marks for multiple simultaneous recordings do not.

In fact, there are some roughly similar cases in audio annotation, where an annotation might reference some specific two- or three-dimensional feature of (for instance) a time-series of short-time amplitude spectra (i.e. a spectrogram), in which case the quantitative details will depend on the analysis recipe. Our system allows such references (like any other information) to be encoded in arc labels, but does not provide any more specific support.

Relationship to multimedia standards

In this context we ought to raise the question of how annotation graphs relate to various multimedia standards like the Synchronized Multimedia Integration Language [`www.w3.org/TR/REC-smil/`] and MPEG-4 [`drogo.cselt.it /mpeg/standards/mpeg-4/mpeg-4.htm`]. Since these provide ways to specify both temporal and spatial relationships among strings, audio clips, still pictures, video sequences, and so on, one hopes that they will offer support for linguistic annotation. It is hard to offer a confident evaluation, since MPEG-4 is still in development, and SMIL's future as a standard is unclear.

With respect to MPEG-4, we reserve judgment until its characteristics become clearer. Our preliminary assessment is that SMIL is not useful for purposes of linguistic annotation, because it is mainly focused on presentational issues (fonts, colors, screen locations, fades and animations, etc.) and does not in fact offer any natural ways to encode the sorts of annotations that we surveyed in the previous section. Thus it is easy to specify that a certain audio file is to be played while a certain caption fades in, moves across the screen, and fades out. It is not (at least straightforwardly) possible to specify that a certain audio file consists of a certain sequence of conversational turns, temporally aligned in a certain way, which consist in turn of certain sequences of words, etc.

3.7 Node references versus byte offsets

The Tipster Architecture for linguistic annotation of text [21] is based on the concept of a fundamental, immutable textual foundation, with all annotations expressed in terms of byte offsets into this text. This is a reasonable solution for cases where the text is a published given, not subject to revision by annotators. However, it is not a good solution for speech transcriptions, which are typically volatile entities, constantly up for revision both by their original authors and by others.

In the case of speech transcriptions, it is more appropriate to treat the basic orthographic transcription as just another annotation, no more formally privileged than a discourse analysis or a translation. Then we are in a much

better position to deal with the common practical situation, in which an initial orthographic transcription of speech recordings is repeatedly corrected by independent users, who may also go on to add new types of annotation of their own, and sometimes also adopt new formatting conventions to suit their own display needs. Those who wish to reconcile these independent corrections, and also combine the independent additional annotations, face a daunting task. In this case, having annotations reference byte offsets into transcriptional texts is almost the worst imaginable solution.

Although nothing will make it trivial to untangle this situation, we believe our approach comes close. As we shall see in §4.3, our use of a flat, unordered file structure incorporating node identifiers and time references means that edits are as strictly local as they possibly can be, and connections among various types of annotation are as durable as they possibly can be. Some changes are almost completely transparent (e.g. changing the spelling of a name). Many other changes will turn out not to interact at all with other types of annotation. When there is an interaction, it is usually the absolute minimum that is necessary. Therefore, keeping track of what corresponds to what, across generations of distributed annotation and revision, is as simple as one can hope to make it.

Therefore we conclude that Tipster-style byte offsets are an inappropriate choice for use as references to audio transcriptions, except for cases where such transcriptions are immutable in principle.

In the other direction, there are several ways to translate Tipster-style annotations into our terms. The most direct way would be to treat Tipster byte offsets exactly as analogous to time references – since the only formal requirement on our time references is that they can be ordered. This method has the disadvantage that the underlying text could not be searched or displayed in the same way that a speech transcription normally could. A simple solution would be to add an arc for each of the lexical tokens in the original text, retaining the byte offsets on the corresponding nodes for translation back into Tipster-architecture terms.

3.8 What is time?

TIMIT and some other extant databases denominate signal time in sample numbers (relative to a designated signal file, with a known sampling rate). Other databases use floating-point numbers, representing time in seconds relative to some fixed offset, or other representations of time such as centiseconds or milliseconds. In our formalization of annotation graphs, the only thing that really matters about time references is that they define an ordering. However, for comparability across signal types, time references need to be intertranslatable.

We feel that time in seconds is generally preferable to sample or frame counts, simply because it is more general and easier to translate across signal representations. However, there may be circumstances in which exact identification of sample or frame numbers is crucial, and some users may prefer to specify these directly to avoid any possibility of confusion.

Technically, sampled data points (such as audio samples or video frames) may be said to denote time intervals rather than time points, and the translation between counts and times may therefore become ambiguous. For instance, suppose we have video data at 30 Hz. Should we take the 30th video frame (counting from one) to cover the time period from 29/30 to 1 second or from 29.5/30 to 30.5/30 second? In either case, how should the endpoints of the interval be assigned? Different choices may shift the correspondence between times and frame numbers slightly.

Also, when we have signals at very different sampling rates, a single sampling interval in one signal can correspond to a long sequence of intervals in another signal. With video at 30 Hz and audio at 44.1 kHz, each video frame corresponds to 1,470 audio samples. Suppose we have a time reference of .9833 seconds. A user might want to know whether this was created because some event was flagged in the 29th video frame, for which we take the mean time point to be 29.5/30 seconds, or because some event was flagged at the 43,365th audio sample, for which we take the central time point to be 43365.5/44100 seconds.

For reasons like these, some users might want the freedom to specify references explicitly in terms of sample or frame numbers, rather than relying on an implicit method of translation to and from time in seconds.

4 A Formal Framework

4.1 Background

All annotations of recorded linguistic signals require one unavoidable basic action: to associate a label, or an ordered sequence of labels, with a stretch of time in the recording(s). Such annotations also typically distinguish labels of different types, such as spoken words vs. non-speech noises. Different types of annotation often span different-sized stretches of recorded time, without necessarily forming a strict hierarchy: thus a conversation contains (perhaps overlapping) conversational turns, turns contain (perhaps interrupted) words, and words contain (perhaps shared) phonetic segments.

A minimal formalization of this basic set of practices is a directed graph with fielded records on the arcs and optional time references on the nodes. We call these 'annotation graphs' (AGs). We believe that this minimal formalization in fact has sufficient expressive capacity to encode, in a reasonably intuitive way, all of the kinds of linguistic annotations in use today. We also believe that this minimal formalization has good properties with respect to creation, maintenance and searching of annotations.

Our strategy is to see how far this simple conception can go, resisting where possible the temptation to enrich its ontology of formal devices, or to establish label types with special syntax or semantics as part of the formalism. It is important to recognize that translation into AGs does not magically create compatibility among systems whose semantics are different. For instance, there are many different approaches to transcribing filled pauses in English – each will translate easily into an AG framework, but their semantic incompatibility is not thereby erased.

4.2 Annotation graphs

We take an annotation label to be a fielded record. Depending on context, it is sometimes convenient to think of such labels as an n-tuple of values distinguished by position, or as an unordered list of attribute-value pairs, or as a set of functions from arcs to labels. Here we take the first option, and draw labels from the cross-product of a collection of label sets L_i.

The nodes N of an AG reference signal data by virtue of a function mapping nodes to time offsets. An annotation may reference more than one signal, and such signals may or may not share the same abstract flow of time (e.g. two signals originating from a stereo recording, versus two signals recorded independently). So we employ a collection of 'timelines' T, where each $T_i \in T$ is a totally ordered set. AGs are now defined as follows:

Definition 1 *An **annotation graph** G over a label set L and a set of timelines T is a 3-tuple $\langle N, A, \tau \rangle$ consisting of a node set N, a collection of arcs A labelled with elements of L, and a time function τ, which satisfies the following conditions:*

1. *$\langle N, A \rangle$ is an acyclic digraph labeled with elements of L, and containing no nodes of degree zero;*
2. *$\tau : N \rightharpoonup \biguplus T_i$, such that, for any path from node n_1 to n_2 in A, if $\tau(n_1)$ and $\tau(n_2)$ are defined, then $\tau(n_1) \leq \tau(n_2)$;*

Each node of an AG is linked to at least one other node. Note, however, that AGs may be disconnected (i.e. they may contain disjoint sub-parts). An AG may also be empty. If $a = \langle n_1, l, n_2 \rangle$ and $\tau(n_1) = \tau(n_2)$ then we call a

an instant. It follows from the second clause of this definition that any piece of*connected* annotation structure can refer to at most one timeline.

Note that the interpretation of labels as identifying substantive content, as conforming to a certain coding standard, as meta-commentary on the annotation, as signaling membership of some equivalence class, as referring to material elsewhere (inside or outside the annotation), as an anchor for an incoming cross-reference, as binary data, or as anything else, falls outside the formalism.

We now illustrate this definition for the TIMIT graph in Figure 2. Let L_1 be the types of transcript information (phoneme, word), and let L_2 be the phonetic alphabet and the orthographic words used by TIMIT. LetT_1 be the set of non-negative integers, the sample numbers.

$$
\begin{aligned}
N &= \{0, 1, 2, 3, 4, 5, 6, 7, 8\} \\
A &= \{\langle 0, \langle \text{P, h\#} \rangle, 1\rangle, \langle 1, \langle \text{P, sh} \rangle, 2\rangle, \langle 2, \langle \text{P, iy} \rangle, 3\rangle, \langle 1, \langle \text{W, she} \rangle, 3\rangle, \langle 3, \langle \text{P, hv} \rangle, 4\rangle, \langle 4, \langle \text{P, ae} \rangle, 5\rangle, \\
&\quad \langle 5, \langle \text{P, dcl} \rangle, 6\rangle, \langle 3, \langle \text{W, had} \rangle, 6\rangle, \langle 6, \langle \text{P, y} \rangle, 7\rangle, \langle 7, \langle \text{P, axr} \rangle, 8\rangle, \langle 6, \langle \text{W, your} \rangle, 8\rangle\} \\
\tau &= \{0 \to 0, 1 \to 2360, 2 \to 3270, 3 \to 5200, 4 \to 6160, 5 \to 8720, 6 \to 9680, 7 \to 10173, 8 \to 11077\}
\end{aligned}
$$

Definition 2 *An AG $\langle N', A', \tau' \rangle$ is a **subgraph** of an AG $\langle N, A, \tau \rangle$ iff $A' \subseteq A$; and N' and τ' are the restriction of N and τ to just those nodes used by A'. If G' is a subgraph of G we write $G' \subseteq G$.*

Observe that the process of moving from an AG to one of its subgraphs is fully determined by the selection of arcs. There is no freedom in the choice of the node set and the time function. Therefore, we think of the subgraph relation as more akin to a *subset* relation on the arc set. The reason for this move will become clear below.

Before proceeding further, we need to bring corpora into the scope of the model. A corpus is just a set of AGs along with a collection of signal files. However, the division of a corpus into its component annotations is somewhat arbitrary (cf. the division of a text corpus into paragraphs, lines, words or characters). For one operation we may want to view a speech corpus as a set of speaker turns, where each turn is its own separate annotation graph. For a different operation it may be more natural to treat the corpus as a set of broadcast programs, or a set of words, or whatever. Therefore we need to blur the distinction between a single annotation and a corpus of annotations. But this is simple; the following definition shows that a multi-annotation corpus counts as a single annotation itself.

Definition 3 *Let $G_1 = \langle N_1, A_1, \tau_1 \rangle$ and $G_2 = \langle N_2, A_2, \tau_2 \rangle$ be two AGs. Then the **disjoint union** of G_1 and G_2, written $G_1 \uplus G_2$, is the AG $\langle N_1 \uplus N_2, A_1 \cup A_2, \tau_1 \cup \tau_2 \rangle$.*

Observe that the arc sets A_i and the time functions τ_i are guaranteed to be non-overlapping, given that there can be no collision of elements of N_1 with N_2. In practice, nodes will simply be assigned unique identifiers, and these identifiers may be further qualified with a namespace. In this way, while the internal structure of the corpus into individual annotations might be reflected in file structure, it is formally represented in the patterning of node identifiers.

Returning once more to query, the result of a query against a corpus is some subgraph of the disjoint union of the elements of that corpus. Accordingly, the result of a query is itself an AG, which could be viewed as a derived corpus and queried further. Multiple independent queries on the same corpus, or (equivalently) multiple corpora derived from the same corpus, might then be combined by union, intersection or relative complement. The following definition is important for the desired closure properties. Let2^G be the powerset of the AG G, the set of subgraphs of G.

Definition 4 *The **algebra** $\mathcal{A}_G$ of an AG G is the boolean algebra $\langle 2^G, \cup, \cap, \bar{}, \emptyset, G \rangle$, where $\cup, \cap, \bar{}$ are set union, intersection and (relative) complement, respectively. Together with $\emptyset$ and G, these operations satisfy the following identities: $G_1 \cup \bar{G}_1 = G$, $G_1 \cap \bar{G}_1 = \emptyset$, where $G_1 \subseteq G$. Union and intersection also satisfy the usual distributive laws.*

Suppose we have a corpus containing a set of AGs G_i. Let C = $\biguplus G_i$. Then the space of all possible query results for C is 2^C. Now it is possible to endow a query language with a model-theoretic semantics in terms of $\mathcal{A}_C$.

4.3 Representation

Annotation graphs can be mapped to a variety of file formats, including some of the formats described in our survey. Here we describe an XML 'surface representation', which is maximally flat and which makes explicit our intuition that AGs are fundamentally a set of arcs. Here we give an XML representation for the above TIMIT example. The ordering of the arcs is not significant.

```
<annotation>
  <arc><source id="0" time="0"/><label att_1="P" att_2="h#"/><target id="1" time="2360"/></arc>
  <arc><source id="1" time="2360"/><label att_1="P" att_2="sh"/><target id="2" time="3270"/></arc>
  <arc><source id="2" time="3270"/><label att_1="P" att_2="iy"/><target id="3" time="5200"/></arc>
  <arc><source id="1" time="2360"/><label att_1="W" att_2="she"/><target id="3" time="5200"/></arc>
  <arc><source id="3" time="5200"/><label att_1="P" att_2="hv"/><target id="4" time="6160"/></arc>
  <arc><source id="4" time="6160"/><label att_1="P" att_2="ae"/><target id="5" time="8720"/></arc>
  <arc><source id="5" time="8720"/><label att_1="P" att_2="dcl"/><target id="6" time="9680"/></arc>
  <arc><source id="3" time="5200"/><label att_1="W" att_2="had"/><target id="6" time="9680"/></arc>
  <arc><source id="6" time="9680"/><label att_1="P" att_2="y"/><target id="7" time="10173"/></arc>
  <arc><source id="7" time="10173"/><label att_1="P" att_2="axr"/><target id="8" time="11077"/></arc>
  <arc><source id="6" time="9680"/><label att_1="W" att_2="your"/><target id="8" time="11077"/></arc>
</annotation>
```

In practice, the `time` and `id` attributes will be fully qualified names. Times will be qualified with timeline information to identify a collection of signal files sharing the same abstract timeline. The ids will be qualified with information about the annotation collection, sufficient to discriminate between multiple independent annotations of the same signal data. Under this scheme, the name tag `<source id="5" time="8720"/>` might become:

```
<source id="http://www.ldc.upenn.edu/~sb/timit-dr1-fjsp0#5"
        time="TIMIT86://train/dr1/fjsp0#8720"/>
```

The qualified node identifier now picks out the site, the annotator `sb`, a logical or physical name for the annotation, plus sufficient information (here `#5`) to pick out the node within that annotation. Multiple annotations of the same signal data will not overlap on these identifiers, and so they can be safely combined into a single annotation if necessary.

The qualified time now identifies the corpus (a name which may need to be resolved) and gives the path to the collection of signals sharing the same timeline. In the situation where multiple signals exist (as in the case of multichannel recordings), the label data will specify the appropriate signal(s). Now multiple annotations of different signal data can be safely combined into a single annotation if necessary.

As far as the annotation formalism is concerned, identifiers are just unanalyzed strings. Each timeline is a separate T_i, and we simply have to guarantee that any pair of times drawn from the same timeline can be compared using $\leq$. (The comparison of times from separate timelines is not defined in general.) The internal syntax for identifiers and timelines is outside the formalism, as is the rest of the above XML syntax (and any other syntax we may devise). The main point here is that any reordering of arcs, any selection of a subset of the arcs (via a query or some 'grep'-like process), and any concatenations of arc sets that came from the same corpus, are immediately well-formed as AG files.

4.4 References and cross-references

In some situations, an AG needs to support incoming references. The formalism is neutral on this subject. As with any set, we pick out a member of the arc set by naming it, that is, listing its two nodes and its label. Since sets, by definition, do not contain repeats, specifying the complete arc data individuates an arc. (The way tuples are identified in a relational database is entirely analogous.)

It is straightforward to add syntactic identifiers to arcs if this is required, and for these to be referenced from elsewhere. We can do this in an extended formalism as follows, though we shall see in a moment that this move is unnecessary.

Definition 5 *An **arc-individuated AG** is a four-tuple $\langle N, A, \tau, \iota \rangle$, such that $\langle N, A, \tau, \rangle$ is an AG, and ι is a function $\iota : A \rightarrow I$ where I is some set (the 'identifiers'). The function ι satisfies the following property: $\iota(a_1) = \iota(a_2) \rightarrow a_1 = a_2$ (i.e. ι is injective, or one-to-one).*

The ι function simply assigns a unique identifier to each arc. It is straightforward to define subgraph, disjoint union, and the algebra of arc-individuated AGs as before. However, these structures can be trivially mapped to the simpler AGs. Define a set of projection functions $\sigma_i : A \rightarrow L_i$, which map any arc $\langle n_1, \langle \ldots, l_i, \ldots \rangle, n_2 \rangle \in A$ to its ith label $l_i \in L_i$. So long as one of the σ_i is injective, it can take the role of ι.

Accordingly, the individuation of arcs to serve as anchors for incoming references amounts to a stipulation about one of the projection functions. (In the corresponding XML we can specify that a certain arc attribute has the type `ID`.)

In order to permit explicit inter-arc links, we can enrich the original AG definition with a set of binary relations on the arc set:

Definition 6 *A **meta-AG** is a four-tuple $\langle N, A, \tau, R \rangle$, such that $\langle N, A, \tau \rangle$ is an AG and R is a set of binary relations on A.*

For example, R_1 might be an equivalence relation for modeling coreference annotation;R_2 and R_3 might represent two different kinds of hierarchical structure (e.g. syntactic and prosodic);R_4 might be a relationship of autosegmental association; R_5 could point from each weak syllable of a given metrical foot to the strong syllable of that foot; and so on.

However, meta-AGs do not admit the natural boolean algebra (see definition 4). To see why this is the case, consider the following meta-AG, over the label set $\{\mathrm{a}, \mathrm{b}\}$: $N = \{1, 2, 3\}$, $A = \{\langle 1, \mathrm{a}, 2 \rangle, \langle 2, \mathrm{b}, 3 \rangle\}$ $R_1 = \{\langle \langle 2, \mathrm{b}, 3 \rangle, \langle 1, \mathrm{a}, 2 \rangle \rangle\}$. This meta-AG G has two arcs, and a cross-reference going from the second arc to the first arc. Consider the subgraph G_1 that consists only of the 'a' arc. The R_1 relation is forced to be empty for this subgraph. The same holds for a subgraph G_2 consisting of just the 'b' arc. But $\bar{G_1} = G_2$. Now, since the single tuple of R_1 is absent from both G_1 and $\bar{G_1}$, it must also be absent from $G_1 \cup \bar{G_1}$. Therefore, $G_1 \cup \bar{G_1} \neq G$, and so the subgraph relation on meta-AGs does not induce a boolean algebra. Thus, while meta-AGs can be defined, they lack a property that is important in the context of databases and query languages.

Note, however, that it is possible to *approximate* meta-AGs with AGs. This recovers the algebraic property, but it means that an inter-arc link is not guaranteed to point to anything. Suppose thatσ_i and σ_j are two projection functions, picking out the ith and jth positions of a label tuple respectively. Then we can create a relation between arcs by creating a relation on a pair of label sets$R_{ij} : L_i \rightarrow L_j$ and then lift it to the arcs thus: $\sigma_j^{-1} \circ R_{ij} \circ \sigma_i$.

4.5 Anchored annotation graphs

In addition to refining the labels and projection functions, there is more that can be said about the time function τ. For convenience, we refer to nodes which have a time reference – elements of the domain of τ – as *anchored nodes*. We now define an extension of AGs which constrains the positions in which unanchored nodes can appear.

Definition 7 *An **anchored AG** $G = \langle N, A, \tau \rangle$ is an AG satisfying two additional conditions:*

1. *If $n \in N$ is such that $\langle n, l, n' \rangle \notin G$ for any $l \in L$, $n' \in N$, then $n \in \mathrm{dom}(\tau)$;*
2. *If $n \in N$ is such that $\langle n', l, n \rangle \notin G$ for any $l \in L$, $n' \in N$, then $n \in \mathrm{dom}(\tau)$;*

Anchored AGs have no dangling arcs (or chains) leading to an indeterminate time point. It follows from this definition that, for any unanchored node, we can reach an anchored node by following a chain of arcs. In fact every path from an unanchored node will finally take us to an anchored node. Likewise, an unanchored node can be reached from an anchored node. A key property of anchored AGs is that we are guaranteed to have some information about the temporal locus of every node. All AGs in §2 are anchored.

Arbitrary subgraphs of anchored AGs may not be anchored, and so we cannot construct a the algebra of an anchored AG. In fact, this is not a serious problem. It is convenient for annotated speech corpora to be anchored, since this greatly facilitates speech playback and visual display. The result of querying an anchored AG will not generally be an anchored AG, but we do not have the same requirements for playback and display of query results. If ever we do want to playback or display the context of an unanchored arc, we simply look it up in the anchored AG, and proceed as before.

Note that there is a special case where anchored AGs regain the desired algebraic property:

Definition 8 *A **totally-anchored AG** $G = \langle N, A, \tau \rangle$ is an AG where τ is total.*

In totally-anchored AGs, every node carries a time reference. The AGs in Figures 2 and 3 are all totally-anchored.

4.6 Subsidiary relations on nodes and arcs

As a further step towards the development of a query language, we can define a variety of useful relations over nodes and arcs.

The first definition below allows us to talk about two kinds of precedence relation on nodes in the graph structure. The first kind respects the graph structure (ignoring the time references), and is called structural precedence, or simply *s-precedence*. The second kind respects the temporal structure (ignoring the graph structure), and is called temporal precedence, or simply *t-precedence*.

Definition 9 *A node n_1 **s-precedes** a node n_2, written $n_1 <_s n_2$, if there is a path from n_1 to n_2. A node n_1 **t-precedes** a node n_2, written $n_1 <_t n_2$, if $\tau(n_1) < \tau(n_2)$.*

Observe that both these relations are transitive. There is a more general notion of precedence which mixes both relations. For example, we can infer that node n_1 precedes node n_2 if we can use a mixture of structural and temporal information to get from n_1 to n_2. This idea is formalized in the next definition.

Definition 10 ***Precedence** is a binary relation on nodes, written $<$, which is the transitive closure of the union of the s-precedes and the t-precedes relations.*

We can now define some useful inclusion relations on arcs. The first kind of inclusion respects the graph structure, so it is called structural inclusion, or *s-inclusion*. The second kind, *t-inclusion*, respects the temporal structure.

Definition 11 *An arc $p = \langle n_1, n_4 \rangle$* ***s-includes*** *an arc $q = \langle n_2, n_3 \rangle$, written $p \supset_s q$, if $n_1 <_s n_2$ and $n_3 <_s n_4$. p* ***t-includes*** *q, written $p \supset_t q$, if $n_1 <_t n_2$ and $n_3 <_t n_4$.*

As with node precedence, we define a general notion of inclusion which generalizes over these two types:

Definition 12 ***Inclusion*** *is a binary relation on arcs, written $\supset$, which is the transitive closure of the union of the s-inclusion and the t-inclusion relations.*

Note that all three inclusion relations are transitive. We assume the existence of non-strict precedence and inclusion relations, defined in the obvious way.

The final definition concerns the *greatest lower bound (glb)* and the *least upper bound (lub)* of an arc.

Definition 13 *Let $a = \langle n_1, l, n_2 \rangle$ be an arc. $\mathrm{glb}(a)$ is the greatest time value t such that there is some node n with $\tau(n) = t$ and $n <_s n_1$. $\mathrm{lub}(a)$ is the least time value t such that there is some node n with $\tau(n) = t$ and $n_2 <_s n$.*

According to this definition, the *glb* of an arc is the time mark of the 'greatest' anchored node from which the arc is reachable. Similarly, the *lub* of an arc is the time mark of the 'least' anchored node reachable from that arc. The *glb* and *lub* are guaranteed to exist for anchored annotation graphs, but not for annotation graphs in general.

4.7 Visualization

It is convenient to have a variety of ways of visualizing AGs. Most of the systems we surveyed in §2 come with visualization components, whether tree-based, extent-based, or some combination of these. We would endorse the use of any descriptively adequate visual notation in concert with the AG formalism, so long as the notation can be endowed with an explicit formal semantics in terms of AGs. Note, however, that not all such visual notations can represent everything an AG contains, so we still need one or more general-purpose visualizations for AGs.

The primary visualization chosen for AGs in this paper uses networks of nodes and arcs to make the point that the mathematical objects we are dealing with are graphs. In most practical situations, this mode of visualization is cumbersome to the point of being useless. Visualization techniques should be optimized for each type of data and for each application, but there are some general techniques that can be cited.

Observe that the direction of time-flow can be inferred from the left-to-right layout of AGs, and so the arrow-heads are redundant. For simple connected sequences (e.g. of words) the linear structure of nodes and arcs is not especially informative; it is better to write the labels in sequence and omit the graph structure. The ubiquitous node identifiers should not be displayed unless there is accompanying text that refers to specific nodes. Label types can be effectively distinguished with colors, typefaces or vertical position. We will usually need to break an AG into chunks which can be presented line-by-line (much like interlinear text) in order to fit on a screen or a page.

The applicability of these techniques depends on the fact that AGs have a number of properties that do not follow automatically from a graphical notation. In other words, many directed acyclic graphs are not well-formed AGs.

Two properties are of particular interest here. First, as already noted, many of the AGs we have constructed as a result of our survey are actually anchored AGs. This means that every arc lies on a path of arcs that is bounded at both ends by time references. So, even when most nodes lack a time reference, we can still associate such paths with an interval of time. A second property, more contingent but equally convenient, is that AGs appear to be 'rightward planar', i.e. they can be drawn in such a way that no arcs cross and each arc is monotonically increasing

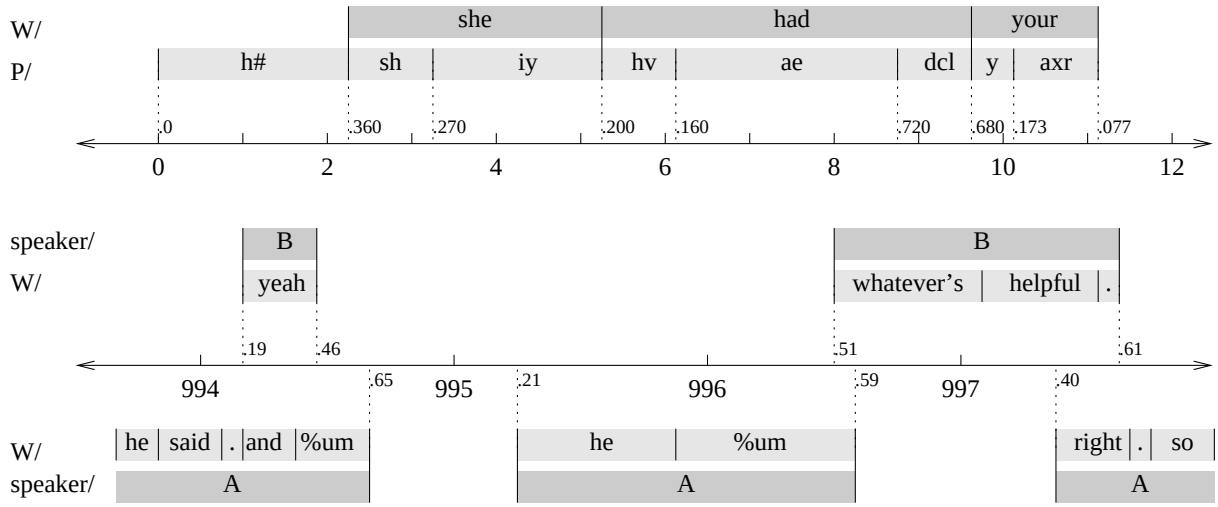

Figure 14: Visualizations for the TIMIT and LDC Telephone Speech Examples

in the rightwards direction (c.f. the definition of upward planarity in [15]). These properties are put to good use in Figure 14, which employs a score notation (c.f. [10, 14, 16, 29]).

The conventions employed by these diagrams are as follows. An arc is represented by a shaded rectangle, where the shading (or color, if available) represents the type information. Where possible, arcs having the same type are displayed on the same level. Arcs are labeled, but the type information is omitted. Inter-arc linkages (see §3.3) are represented using coindexing. The ends of arcs are represented using short vertical lines having the same width as the rectangles. These may be omitted if the tokenization of a string is predictable. If two arcs are incident on the same node but their corresponding rectangles appear on different levels of the diagram, then the relevant endpoints are connected by a solid line. For ease of external reference, these lines can be decorated with a node identifier. Anchored nodes are connected to the timeline with dotted lines. The point of intersection is labeled with a time reference. If necessary, multiple timelines may be used. Nodes sharing a time reference are connected with a dotted line. In order to fit on a page, these diagrams may be cut at any point, with any partial rectangles labeled on both parts.

Unlike some other conceivable visualizations (such as the tree diagrams and autosegmental diagrams used by Festival and Emu), this scheme emphasizes the fact that each component of an annotation has temporal extent. The scheme neatly handles the cases where temporal information is partial.

4.8 Multiple Annotations

Linguistic analysis is always multivocal, in two senses. First, there are many types of entities and relations, on many scales, from acoustic features spanning a hundredth of a second to narrative structures spanning tens of minutes. Second, there are many alternative representations or construals of a given kind of linguistic information.

Sometimes these alternatives are simply more or less convenient for a certain purpose. Thus a researcher who thinks theoretically of phonological features organized into moras, syllables and feet, will often find it convenient to use a phonemic string as a representational approximation. In other cases, however, different sorts of transcription or annotation reflect different theories about the ontology of linguistic structure or the functional categories of communication.

The AG representation offers a way to deal productively with both kinds of multivocality. It provides a framework for relating different categories of linguistic analysis, and at the same time to compare different approaches to a given type of analysis.

As an example, Figure 15 shows an AG-based visualization of eight different sorts of annotation of a phrase from the BU Radio Corpus, produced by Mari Ostendorf and others at Boston University, and published by the LDC [`www.ldc.upenn.edu /Catalog/LDC96S36.html`]. The basic material is from a recording of a local public radio news broadcast. The BU annotations include four types of information: orthographic transcripts, broad phonetic transcripts (including main word stress), and two kinds of prosodic annotation, all time-aligned to the digital audio files. The two kinds of prosodic annotation implement the system known as ToBI [`www.ling.ohio-state.edu/phonetics/E_ToBI/`]. ToBI is an acronym for "Tones and Break Indices", and correspondingly provides two types of information: *Tones*, which are taken from a fixed vocabulary of categories of (stress-linked) "pitch accents" and (juncture-linked) "boundary tones"; and *Break Indices*, which are integers characterizing the strength and nature of interword disjunctures.

We have added four additional annotations: coreference annotation and named entity annotation in the style of MUC-7 [`www.muc.saic.com/proceedings/muc_7_toc.html`] provided by Lynette Hirschman; syntactic structures in the style of the Penn TreeBank [27] provided by Ann Taylor; and an alternative annotation for the F_0 aspects of prosody, known as *Tilt* [36] and provided by its inventor, Paul Taylor. Taylor has done Tilt annotations for much of the BU corpus, and will soon be publishing them as a point of comparison with the ToBI tonal annotation. Tilt differs from ToBI in providing a quantitative rather than qualitative characterization of F_0 obtrusions: where ToBI might say "this is a L+H* pitch accent," Tilt would say "This is an F_0 obtrusion that starts at time t_0, lasts for duration d seconds, involves a Hz total F_0 change, and ends l Hz different in F_0 from where it started."

As usual, the various annotations come in a bewildering variety of file formats. These are not entirely trivial to put into registration, because (for instance) the TreeBank terminal string contains both more (e.g. traces) and fewer (e.g. breaths) tokens than the orthographic transcription does. One other slightly tricky point: the connection between the word string and the "break indices" (which are ToBI's characterizations of the nature of interword disjuncture) are mediated only by identity in the floating-point time values assigned to word boundaries and to break indices in separate files. Since these time values are expressed as ASCII strings, it is easy to lose the identity relationship without meaning to, simply by reading in and writing out the values to programs that may make different choices of internal variable type (e.g. float vs. double), or number of decimal digits to print out, etc.

Problems of this type are normal whenever multiple annotations need to be compared. Solving them is not rocket science, but does take careful work. When annotations with separate histories involve mutually inconsistent corrections, silent omissions of problematic material, or other typical developments, the problems are multiplied. In noting such difficulties, we are not criticizing the authors of the annotations, but rather observing the value of being able to put multiple annotations into a common framework.

Once this common framework is established, via translation of all eight "strands" into AG terms, we have the basis for posing queries that cut across the different types of annotation. For instance, we might look at the distribution of Tilt parameters as a function of ToBI accent type; or the distribution of Tilt and ToBI values for initial vs. non-initial members of coreference sets; or the relative size of Tilt F0-change measures for nouns vs. verbs.

In this section, we have indicated some of the ways in which the AG framework can facilitate the analysis of complex combinations linguistic annotations. These annotation sets are typically multivocal, both in the sense of covering multiple types of linguistic information, and also in the sense of providing multiple versions of particular types of analysis. Discourse studies are especially multivocal in both senses, and so we feel that this approach will be especially helpful to discourse researchers.

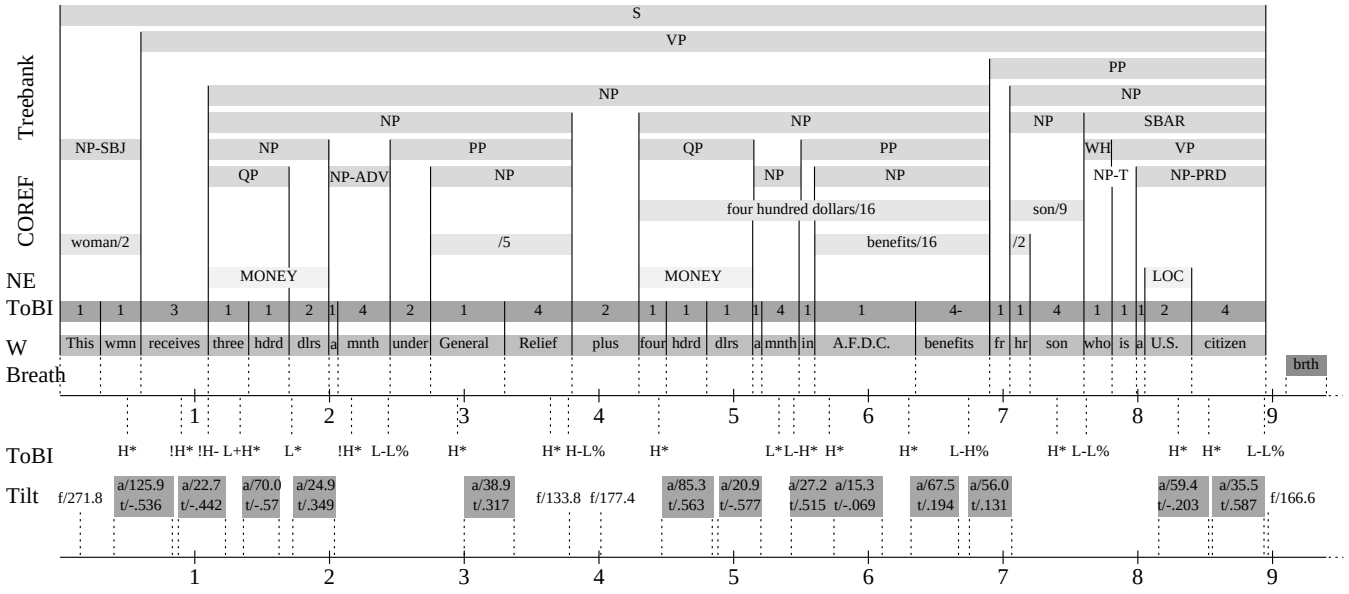

Figure 15: Visualization for BU Example

5 Conclusions and Future Work

5.1 Evaluation criteria

There are many existing approaches to linguistic annotation, and many options for future approaches. Any evaluation of proposed frameworks, including ours, depends on the costs and benefits incurred in a range of expected applications. Our explorations have presupposed a particular set of ideas about applications, and therefore a particular set of goals. We think that these ideas are widely shared, but it seems useful to make them explicit.

Here we are using 'framework' as a neutral term to encompass both the definition of the logical structure of annotations, as discussed in this paper, as well as various further specifications of e.g. annotation conventions and file formats.

Generality, specificity, simplicity

> Annotations should be publishable (and will often be published), and thus should be mutually intelligible across laboratories, disciplines, computer systems, and the passage of time.
>
> Therefore, an annotation framework should be sufficiently expressive to encompass all commonly used kinds of linguistic annotation, including sensible variants and extensions. It should be capable of managing a variety of (partial) information about labels, timing, and hierarchy.
>
> The framework should also be formally well-defined, and as simple as possible, so that researchers can easily build special-purpose tools for unforeseen applications as well as current ones, using future technology as well as current technology.

Searchability and browsability

> Automatic extraction of information from large annotation databases, both for scientific research and for technological development, is a key application.

Therefore, annotations should be conveniently and efficiently searchable, regardless of their size and content. It should be possible to search across annotations of different material produced by different groups at different times – if the content permits it – without having to write special programs. Partial annotations should be searchable in the same way as complete ones.

This implies that there should be an efficient algebraic query formalism, whereby complex queries can be composed out of well-defined combinations of simple ones, and that the result of querying a set of annotations should be just another set of annotations.

This also implies that (for simple queries) there should be efficient indexing schemes, providing near constant-time access into arbitrarily large annotation databases.

The framework should also support easy 'projection' of natural sub-parts or dimensions of annotations, both for searching and for display purposes. Thus a user might want to browse a complex multidimensional annotation database – or the results of a preliminary search on one – as if it contained only an orthographic transcription.

Maintainability and durability

Large-scale annotations are both expensive to produce and valuable to retain. However, there are always errors to be fixed, and the annotation process is in principle open-ended, as new properties can be annotated, or old ones re-done according to new principles. Experience suggests that maintenance of linguistic annotations, especially across distributed edits and additions, can be a vexing and expensive task. Therefore, any framework should facilitate maintenance of coherence in the face of distributed correction and development of annotations.

Different dimensions of annotation should therefore be orthogonal, in the sense that changes in one dimension (e.g. phonetic transcription) do not entail any change in others (e.g. discourse transcription), except insofar as the content necessarily overlaps. Annotations of temporally separated material should likewise be modular, so that revisions to one section of an annotation do not entail global modification. Queries on material that is not affected by corrections or additions should return the same thing before and after the updates.

In order to facilitate use in scientific discourse, it should be possible to define durable references which remain valid wherever possible, and produce the same results unless the referenced material itself has changed.

Note that it is easy enough to define an invertible sequence of editing operations for any way of representing linguistic annotations – e.g. by means of Unix 'diff' – but what we need in this case is also a way to specify the correspondence (wherever it remains defined) between arbitrary pieces of annotation before and after the edit. Furthermore, we do not want to impose any additional burden on human editors – ideally, the work minimally needed to implement a change should also provide any bookkeeping needed to maintain correspondences.

How well does our proposal satisfy these criteria?

We have tried to demonstrate generality, and to provide an adequate formal foundation, which is also ontologically parsimonious (if not positively miserly!).

Although we have not defined a query system, we have indicated the basis on which one can be constructed: (tuple sets constituting) AGs are closed under union, intersection and relative complementation; the set of subgraphs of an AG is simply the power set of its constituent tuples; simple pattern matching on an AG can be defined to produce a set of annotation subgraphs; etc. Obvious sorts of simple predicates on temporal relations, graphical relations, label types, and label contents will clearly fit into this framework.

The foundation for maintainability is present: fully orthogonal annotations (those involving different label types and time points) do not interact at all, while linked annotations (such as those that share time points) are linked only to

the point that their content requires. New layers of annotation can be added monotonically, without any modification whatsoever in the representation of existing layers. Corrections to existing annotations are as representationally local as they can be, given their content.

Although we have not provided a recipe for durable citations (or for maintenance of trees of invertible modifications), the properties just cited will make it easier to develop practical approaches. In particular, the relationship between any two stages in the development or correction of an annotation will always be easy to compute as a set of basic operations on the tuples that express an AG. This makes it easy to calculate just the aspects of a tree or graph of modifications that are relevant to resolving a particular citation.

5.2 Future work

Interactions with relational data

Linguistic databases typically include important bodies of information whose structure has nothing to do with the passage of time in any particular recording, nor with the sequence of characters in any particular text. For instance, the Switchboard corpus includes tables of information about callers (including date of birth, dialect area, educational level, and sex), conversations (including the speakers involved, the date, and the assigned topic), and so on. This side information is usually well expressed as a set of relational tables. There also may be bodies of relevant information concerning a language as a whole rather than any particular speech or text database: lexicons and grammars of various sorts are the most obvious examples. The relevant aspects of these kinds of information also often find natural expression in relational terms.

Users will commonly want to frame queries that combine information of these kinds with predicates defined on AGs: 'find me all the phrases flagged as questions produced by South Midland speakers under the age of 30'. The simplest way to permit this is simply to identify (some of the) items in a relational database with (some of the) labels in an annotation. This provides a limited, but useful, method for using the results of certain relational queries in posing an annotational query, or vice versa. More complex modes of interaction are also possible, as are connections to other sorts of databases; we regard this as a fruitful area for further research.

Generalizing time marks to an arbitrary ordering

We have focused on the case of audio or video recordings, where a time base is available as a natural way to anchor annotations. This role of time can obviously be reassigned to any other well-ordered single dimension. The most obvious case is that of character- or byte-offsets into an invariant text file. This is the principle used in the so-called Tipster Architecture [21], where all annotations are associated with stretches of an underlying text, identified via byte offsets into a fixed file. We do not think that this method is normally appropriate for indexing into audio transcriptions, because they are so often subject to revision.

Generalizing node identifiers and arc labels

As far as the AG formalism is concerned, node identifiers and arc labels are just sets. As a practical matter, members of each set would obviously be represented as strings. This opens the door to applications which encode arbitrary information in these strings. Indeed, the notion that arc labels encode 'external' information is fundamental to the whole enterprise. After all, the point of the annotations is to include strings interpreted as orthographic words, speaker names, phonetic segments, file references, or whatever. These interpretations are not built into the formalism, however, and this is an equally important trait, since it determines the simplicity and generality of the framework.

In the current formalization, arcs are decorated with fielded records. This structure already contains a certain amount of complexity, since the simplest kind of arc decoration would be purely atomic. In this case, we are convinced that the added value provided by multiple fields is well worth the cost: all the bodies of annotation practice that we surveyed had structure that was naturally expressed in terms of atomic label types, and therefore a framework in which arc decorations were just single uninterpreted strings – zeroth order labels – would not be expressively adequate. It is easy to imagine a wealth of other possible fields. Such fields could identify the original annotator and the creation date of the arc. They could represent the confidence level of some other field. They could encode a complete history of successive modifications. They could provide hyperlinks to supporting material (e.g. chapter and verse in the annotators' manual for a difficult decision). They could provide equivalence class identifiers (§3.3). And they could include an arbitrarily-long SGML-structured commentary.

In principle, we could go still further, and decorate arcs with arbitrarily nested attribute-value matrices (AVMs) endowed with a type system [13] – a second-order approach. These AVMs could contain references to other parts of the annotation, and multiple AVMs could contain shared substructures. Substructures could be disjoined as well as conjoined, and appropriate attributes could depend on the local type information. A DTD-like label grammar could specify available label types, their attributes and the type ordering. We believe that this is a bad idea: it negates the effort that we made to provide a simple formalism expressing the essential contents of linguistic annotations in a natural and consistent way. Typed feature structures are also very general and powerful devices, and entail corresponding costs in algorithmic and implementational complexity. Therefore, we wind up with a less useful representation that is much harder to compute with.

Consider some of the effort that we have put into establishing a simple and consistent ontology for annotation. In the CHILDES case (§2.3), we split a sentence-level annotation into a string of word-level annotations for the sake of simplifying word-level searches. In the Switchboard Treebank case (§2.7) we modeled hierarchical information using the syntactic chart construction. Because of these choices, CHILDES and Switchboard annotations become formally commensurate – they can be searched or displayed in exactly the same terms. With labels as typed feature structures, whole sentences, whole tree structures, and indeed whole databases could be packed into single labels. We could therefore have chosen to translate CHILDES and Switchboard formats directly into typed feature structures. If we had done this, however, the relationship between simple concepts shared by the two formats – such as lexical tokens and time references – would remain opaque.

Our preference is to extend the formalism cautiously, where it seems that many applications will want a particular capability, and to offer a simple mechanism to permit local or experimental extensions, or approximations that stay within the confines of the existing formalism.

5.3 Software

We have claimed that AGs can provide an interlingua for varied annotation databases, a formal foundation for queries on such databases, and a route to easier development and maintenance of such databases. Delivering on these promises will require software. For those readers who agree with us that this is an essential point, we will sketch our current perspective.

As our catalogue of examples indicated, it is fairly easy to translate between other speech database formats and AGs, and we have already built translators in several cases. We are also experimenting with simple software for creation, visualization, editing, validation, indexing, and search, and have specified an elementary API. Our first goal is an open collection of relatively simple tools that are easy to prototype and to modify, in preference to a monolithic 'annotation graph environment.' However, we are also committed to the idea that tools for creating and using linguistic annotations should be widely accessible to computationally unsophisticated users, which implies that eventually such tools need to be encapsulated in reliable and simple interactive form.

Other researchers have also begun to experiment with the annotation graph concept as a basis for their software tools, and a key index of the idea's merit will of course be the extent to which tools are provided by others.

Visualization, creation, editing

Existing open-source software such as Transcriber [5], Snack [33], and the ISIP transcriber tool [`www.isip.msstate.edu/resources/software`], whose user interfaces are all implemented in Tcl/tk, make it easy to create interactive tools for creation, visualization, and editing of AGs. For instance, Transcriber can be used without any changes to produce transcriptions in the LDC Broadcast News format, which can then be translated into AGs. Provision of simple input/output functions enables the program to read and write AGs directly. The architecture of the current tool is not capable of dealing with arbitrary AGs, but generalizations in that direction are planned.

Validation

An annotation may need to be submitted to a variety of validation checks, for basic syntax, content and larger-scale structure. First, we need to be able to tokenize and parse an annotation, without having to write new tokenizers and parsers for each new task. We also need to undertake some superficial syntax checking, to make sure that brackets and quotes balance, and so on. In the SGML realm, this need is partially met by DTDs. We propose to meet the same need by developing conversion and creation tools that read and write well-formed graphs, and by input/output modules that can be used in the further forms of validation cited below.

Second, various content checks need to be performed. For instance, are purported phonetic segment labels actually members of a designated class of phonetic symbols or strings? Are things marked as 'non-lexemic vocalizations' drawn from the officially approved list? Do regular words appear in the spell-check dictionary? Do capital letters occur in legal positions? These checks are not difficult to implement, e.g. as Perl scripts, especially given a module for handling basic operations correctly.

Finally, we need to check for correctness of hierarchies of arcs. Are phonetic segments all inside words, which are all inside phrases, which are all inside conversational turns, which are all inside conversations? Again, it is easy to define such checks in a software environment that has appropriately expressive primitives (e.g. a Perl AG module).

Indexing and Search

A variety of indexing strategies for AGs would permit efficient access to AG content centered on a temporal locus, or based on the label information, or based on the hierarchies implicit in the graph structure. Such indexing is well defined, algorithmically simple, and easy to implement in a general way. Construction of general query systems, however, is a matter that needs to be explored more fully in order to decide on the details of the query primitives and the methods for building complex queries, and also to try out different ways to express queries. Among the many questions to be explored are: how to express general graph- and time-relations; how to integrate regular expression matching over labels; how to integrate annotation-graph queries and relational queries; how to integrate lexicons and other external resources; and how to model sets of databases, each of which contains sets of AGs, signals and perhaps relational side-information.

It is easy to come up with answers to each of these questions, and it is also easy to try the answers out, for instance in the context of a collection of Perl modules providing the needed primitive operations. We regard it as an open research problem to find good answers that interact well, and also to find good ways to express queries in the system that those answers will define.

5.4 Envoi

Whether or not our ideas are accepted by the various research communities who create and use linguistic annotations, we hope to foster discussion and cooperation among members of these communities. A focal point of this effort is the Linguistic Annotation Page at [www.ldc.upenn.edu/annotation].

When we look at the numerous and diverse forms of linguistic annotation documented on that page, we see underlying similarities that have led us to imagine general methods for access and search, and shared tools for creation and maintenance. We hope that this discussion will move others in the same direction.

6 Acknowledgements

An earlier version of this paper was presented at ICSLP-98, and subsequently as CIS Technical Report 99-01 [xxx.lanl.gov /abs/cs.CL/9903003]. A paper applying the model to discourse [xxx.lanl.gov /abs/cs.CL/990700 was subsequently incorporated. We are grateful to the following people for discussions which have helped clarify our ideas about annotations, and for comments on earlier drafts: Sam Bayer, Peter Buneman, Jean Carletta, Steve Cassidy, Chris Cieri, Hamish Cunningham, David Day, George Doddington, David Graff, Lynette Hirschman, Ewan Klein, Brian MacWhinney, David McKelvie, Boyd Michailovsky, Michelle Minnick Fox, Dick Oehrle, Florian Schiel, Richard Sproat, Ann Taylor, Paul Taylor, Henry Thompson, Marilyn Walker, Peter Wittenburg, Jonathan Wright, members of the Edinburgh Language Technology Group, and participants of the COCOSDA Workshop at ICSLP-98 and the Discourse Tagging Workshop at ACL-99.

References

[1] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison Wesley, 1995.

[2] James F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26:832–43, 1983.

[3] T. Altosaar, M. Karjalainen, M. Vainio, and E. Meister. Finnish and Estonian speech applications developed on an object-oriented speech processing and database system. In *Proceedings of the First International Conference on Language Resources and Evaluation Workshop: Speech Database Development for Central and Eastern European Languages*, 1998. Granada, Spain, May 1998.

[4] A. Anderson, M. Bader, E. Bard, E. Boyle, G. M. Doherty, S. Garrod, S. Isard, J. Kowtko, J. McAllister, J. Miller, C. Sotillo, H. Thompson, and R. Weinert. The HCRC Map Task corpus. *Language and Speech*, 34:351–66, 1991.

[5] Claude Barras, Edouard Geoffrois, Zhibiao Wu, and Mark Liberman. Transcriber: a free tool for segmenting, labelling and transcribing speech. In *Proceedings of the First International Conference on Language Resources and Evaluation*, 1998.

[6] Steven Bird. *Computational Phonology: A Constraint-Based Approach*. Studies in Natural Language Processing. Cambridge University Press, 1995.

[7] Steven Bird. A lexical database tool for quantitative phonological research. In *Proceedings of the Third Meeting of the ACL Special Interest Group in Computational Phonology*. Association for Computational Linguistics, 1997.

[8] Steven Bird and Ewan Klein. Phonological events. *Journal of Linguistics*, 26:33–56, 1990.

[9] Steven Bird and D. Robert Ladd. Presenting autosegmental phonology. *Journal of Linguistics*, 27:193–210, 1991.

[10] Catherine Browman and Louis Goldstein. Articulatory gestures as phonological units. *Phonology*, 6:201–51, 1989.

[11] Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, Robert L. Mercer, and Paul S. Roossin. A statistical approach to machine translation. *Computational Linguistics*, 16:79–85, 1990.

[12] Jean Carletta and Amy Isard. The MATE annotation workbench: user requirements. In *Towards Standards and Tools for Discourse Tagging – Proceedings of the Workshop*, pages 11–17. Association for Computational Linguistics, 1999.

[13] Bob Carpenter. *The Logic of Typed Feature Structures*, volume 32 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1992.

[14] Steve Cassidy and Jonathan Harrington. Emu: An enhanced hierarchical speech data management system. In *Proceedings of the Sixth Australian International Conference on Speech Science and Technology*, 1996. [www.shlrc.mq.edu.au/emu/].

[15] Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis. Algorithms for drawing graphs: an annotated bibliography. [wilma.cs.brown/edu/pub/papers/compgeo/gdbiblio.ps.gz], 1994.

[16] Konrad Ehlich. HIAT – a transcription system for discourse data. In Jane A. Edwards and Martin D. Lampert, editors, *Talking Data: Transcription and Coding in Discourse Research*, pages 123–48. Hillsdale, NJ: Erlbaum, 1992.

[17] John S. Garofolo, Lori F. Lamel, William M. Fisher, Jonathon G. Fiscus, David S. Pallett, and Nancy L. Dahlgren. *The DARPA TIMIT Acoustic-Phonetic Continuous Speech Corpus CDROM*. NIST, 1986. [www.ldc.upenn.edu/lol/docs/TIMIT.html].

[18] Gerald Gazdar and Chris Mellish. *Natural Language Processing in Prolog: An Introduction to Computational Linguistics*. Addison-Wesley, 1989.

[19] J. J. Godfrey, E. C. Holliman, and J. McDaniel. Switchboard: A telephone speech corpus for research and develpment. In *Proceedings of the IEEE Conference on Acoustics, Speech and Signal Processing*, volume I, pages 517–20, 1992. [www.ldc.upenn.edu/Catalog/LDC93S7.html].

[20] S. Greenberg. The switchboard transcription project. LVCSR Summer Research Workshop, Johns Hopkins University, 1996.

[21] R. Grishman. TIPSTER Architecture Design Document Version 2.3. Technical report, DARPA, 1997. [www.nist.gov/itl/div894/894.02/related_projects/tipster/].

[22] Susan R. Hertz. The delta programming language: an integrated approach to nonlinear phonology, phonetics, and speech synthesis. In John Kingston and Mary E. Beckman, editors, *Papers in Laboratory Phonology I: Between the Grammar and Physics of Speech*, chapter 13, pages 215–57. Cambridge University Press, 1990.

[23] Lynette Hirschman and Nancy Chinchor. MUC-7 coreference task definition. In *Message Understanding Conference Proceedings*. Published online, 1997. [www.muc.saic.com/proceedings/muc_7_toc.html].

[24] Daniel Jurafsky, Rebecca Bates, Noah Coccaro, Rachel Martin, Marie Meteer, Klaus Ries, Elizabeth Shriberg, Andreas Stolcke, Paul Taylor, and Carol Van Ess-Dykema. Automatic detection of discourse structure for speech recognition and understanding. In *Proceedings of the 1997 IEEE Workshop on Speech Recognition and Understanding*, pages 88–95, Santa Barbara, 1997.

[25] Daniel Jurafsky, Elizabeth Shriberg, and Debra Biasca. Switchboard SWBD-DAMSL Labeling Project Coder's Manual, Draft 13. Technical Report 97-02, University of Colorado Institute of Cognitive Science, 1997. [stripe.colorado.edu/~jurafsky/manual.august1.html].

[26] Brian MacWhinney. *The CHILDES Project: Tools for Analyzing Talk*. Mahwah, NJ: Lawrence Erlbaum., second edition, 1995. [poppy.psy.cmu.edu/childes/].

[27] Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–30, 1993. www.cis.upenn.edu/~treebank/home.html.

[28] Boyd Michailovsky, John B. Lowe, and Michel Jacobson. Linguistic data archiving project. [lacito.vjf.cnrs.fr/ARCHIVAG/ENGLISH.htm].

[29] Carol Neidle and D. MacLaughlin. SignStream^{TM}: a tool for linguistic research on signed languages. *Sign Language and Linguistics*, 1:111–14, 1998. [web.bu.edu/asllrp/SignStream].

[30] NIST. A universal transcription format (UTF) annotation specification for evaluation of spoken language technology corpora. [www.nist.gov/speech/hub4_98/utf-1.0-v2.ps], 1998.

[31] Emanuel Schegloff. Reflections on studying prosody in talk-in-interaction. *Language and Speech*, 41:235–60, 1998. www.sscnet.ucla.edu/soc/faculty/schegloff/prosody/.

[32] Florian Schiel, Susanne Burger, Anja Geumann, and Karl Weilhammer. The Partitur format at BAS. In *Proceedings of the First International Conference on Language Resources and Evaluation*, 1998. [www.phonetik.uni-muenchen.de /Bas/BasFormatseng.html].

[33] Kåre Sjölander, Jonas Beskow, Joakim Gustafson, Erland Lewin, Rolf Carlson, and Björn Granström. Web-based educational tools for speech technology. In *ICSLP-98*, 1998.

[34] Wojciech Skut, Brigitte Krenn, Thorsten Brants, and Hans Uszkoreit. An annotation scheme for free word order languages. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, 1997.

[35] Ann Taylor. *Dysfluency Annotation Stylebook for the Switchboard Corpus*. University of Pennsylvania, Department of Computer and Information Science, 1995. [ftp.cis.upenn.edu/pub/treebank/swbd/doc/DFL-book.ps], original version by Marie Meteer et al.

[36] Paul A. Taylor. The tilt intonation model. In *Proceedings of the 5th International Conference on Spoken Language Processing*, 1998.

[37] Paul A. Taylor, Alan W. Black, and Richard J. Caley. The architecture of the Festival speech synthesis system. In *Third International Workshop on Speech Synthesis*, Sydney, Australia, November 1998.

[38] Paul A. Taylor, Alan W. Black, and Richard J. Caley. Heterogeneous relation graphs as a mechanism for representing linguistic information. [www.cstr.ed.ac.uk/publications/new/draft/Taylor_draft_a.ps], 1999.

[39] Text Encoding Initiative. *Guidelines for Electronic Text Encoding and Interchange (TEI P3)*. Oxford University Computing Services, 1994. [www.uic.edu/orgs/tei/].