Chapter 9

General Discussion

9.1 Introduction

The research objective of this study has been to design, build, and test a linguistically-based SLG system that generates speech with pitch accents and intonational boundaries that are most acceptable to experts (Section 1.3). Although the system generates speech output, our work focused on generating intonation at the phonological level, largely abstracting away from phonetic implementation. Similarly, little attention was paid to matters of tune choice. The scope of the work was further restricted by language (Dutch), genre (read speech), application domain (WFD and PTD), and a commitment to a particular theoretical framework (Chapter 2).

According to our research objective, the judgements of intonational experts were taken as a reference for system output. One reason is that acoustically-based approaches for assessing the intuitions of native speakers are faced with several methodological problems (Section 1.1.4). Therefore, we chose to use a symbolic assessement of the intuitions of speakers. This requires experts, because they are able to express their intonational intuitions directly in terms of a symbolic representation. Another reason is that experts have developed a greater awareness of the significance of intonation, and their judgements are taken to be superior to those of average native speakers. In an attempt to deal with the variability inherent to intonation, the intended output was narrowed down to intonation that is *most acceptable* according to the experts (Section 1.1.3). We noted that this goal is well suited to the research paradigm of NLG/SLG, where systems try to generate the *best* form of output (given the content and context of the message), rather than a form that is merely acceptable.

Designing and building an SLG system, generally conceived of as a combination of NLG and speech synthesis, led to a number of research questions regarding its architecture (Section 1.3):

1. What is the best way to combine NLG and speech synthesis?
2. What is the best place in the system to control the intonation?
3. What modifications to the 'standard' architecture of NLG and speech synthesis systems are required?
4. What additional information should be generated by the NLG system, and how can this be accomplished?

These questions were primarily addressed in Chapter 5, while the last was implicit in most of the other chapters. In the first part of the current chapter, we will review the answers to these questions provided by our work. In Section 9.2, we will recapitulate the results, compare them to other work, address unsolved issues, and speculate on further improvements.

Returning to our research objective, the intended SLG system is linguistically-based, because our motivation has been primarily linguistic. Our intention has been to use SLG as a testbed for linguistic theories that cover the relation between the intonation of an utterance and other aspects of its linguistic structure and context. The theoretical framework was presented in Chapter 2, and subsequently refined in the following chapters: phonological phrasing in Chapter 3, intonational phrasing in Chapter 4, and pitch accent placement in Chapter 7, while Chapter 8 reported on a final evaluation. This work has been driven by two research questions relating to the linguistic theory on which we based our efforts:

1. How adequate is it?
2. How can it be improved?

We will address these questions in the second part of this chapter (Section 9.3), in which we summarise our results, compare it to other work, and address remaining challenges. We will finish this chapter with a discussion of some of the implications of our work for applied speech synthesis systems in Section 9.4.

9.2 The architecture of an SLG system

9.2.1 Combining NLG and speech synthesis

Our first research question concerned the best way to combine NLG and speech synthesis in an SLG system. It was argued that using the text output of an NLG system as input to a TTS system does not make a good interface, because it fails to exploit the discourse, semantic, and syntactic information produced in the text generation process, for the generation of intonation (Section 5.2.3). Instead, the NLG system and the speech synthesiser should be integrated to a much higher degree, allowing detailed and reliable linguistic information to be used for generating adequate intonation, and prosody in general. This point has been made by many researchers, and forms the background for most work on SLG.

Our proposal for an SLG architecture resulted from several considerations. To begin with, it is an extension of the modular pipeline architecture that is currently regarded as the standard model for NLG systems. The model allows information to be concentrated and encapsulated in modules, facilitating design, implementation, documentation, maintenance, and modification of the system. In addition, it provides an opportunity to experiment with different versions of a particular module, or to reuse one or more modules in a new system. The same considerations argue in favour of using a pipeline architecture, in which the flow of information is strictly sequential, and no feedback is allowed. Although pipeline architectures have been criticised for being too restrictive (Inui, Tokunaga, and Tanaka 1992; Rubinoff 1992), we did not encounter any serious problems, suggesting that it is a viable choice for the purposes of SLG.

A second consideration affecting the architecture stems from a further research question: what is the best place in the system to control intonation? We argued that prosodic realization, i.e. generating a phonological description of the prosody (including the intonation), cannot be reduced to the standard NLG tasks of either microplanning or structure realization (Section 5.2.4). Nor should it be left to the speech synthesiser, which approximates the process by heuristic rules or purely stochastic models intended for a TTS environment. Instead, it should be recognised as a complex, inherently linguistic process, requiring a separate module. Therefore, we proposed a prosodic realization module that connects the syntactic realizer to the speech synthesiser.

A third consideration was related to reusability. For several languages, reusable surface realizers are available which translate syntactic specifications into well-formed expressions. These allow developers of NLG systems to ignore the low-level syntactic details of the target language, and instead to focus on issues of document structuring and microplanning. In contrast, reusable document planners are not yet available, primarily because planning is intrinsically bound to a certain domain. We conjectured that prosodic realization is fairly domain-independent as well. This opens up the possibility for a reusable 'content-to-speech' system that serves as the front-end for an SLG system, and allows developers of an SLG system to abstract away from the prosodic details of the target language as long as they make sure their planning components deliver the required input.

On the basis of these considerations, we proposed an SLG architecture that extends the standard modular pipeline NLG architecture with a separate module for prosodic realization, and that uses a domain-independent, reusable content-to-speech system; see Figure 9.1. Accordingly, our SLG systems, WFDGen and PTDGen, have two major parts:

1. a domain-specific text planner (PlanR) that generates the *content* in the form of a text specification containing a number of abstract phrase specifications

2. a domain-independent front-end called CONPAS ('CONtent to Prosodically Adequate Speech') that is responsible for the *form* of the spoken output.

The fact that CONPAS has been successfully used in building SLG systems for two different application domains shows the viability of a domain-independent, reusable content-to-speech system. As a result, we have a platform that allows us to build laboratory SLG systems for other domains with far less effort than it would take to build them from scratch.

Another of our research questions asked what modifications to the 'standard' NLG and speech synthesis architecture are required. As far as NLG is concerned, one of the most obvious modifications is that spoken output obviates the necessity of graphemic output by the surface realizer, and therefore also the need for a structure realization component that takes care of orthographic processing (punctuation) and visual lay-out.[1] Instead, the system needs a syntactic realizer that outputs syntactic structures, and transfers other useful discourse and semantic information from earlier stages of the generation process.

Speech synthesisers have traditionally been used in a TTS environment, where a stage of text preprocessing and linguistic analysis precedes actual synthesis. An SLG environment, however, has no need for expanding abbreviations, tagging of word classes, or syntactic parsing, because the resulting information is available from the NLG system. Preceded by a prosodic realization module generating prosody at the symbolic level, the tasks remaining for the speech synthesiser are the application of segmental rules, phonetic implementation of prosody in terms of fundamental frequency, duration, and intensity, followed by actual synthesis of the wave form. In our system, we chose to employ the existing lexica, and therefore left grapheme-to-phoneme conversion to the speech synthesiser. However, the syntactic realizer could equally generate phonemic forms instead of, or in addition to, the graphemic forms.

In the next two sections, the NLG modules (the text planner and the syntactic realizer) will be discussed in the light of the two remaining research questions: what additional information must be generated by the NLG system to facilitate the control of intonation, and how can this be accomplished?

9.2.2 The contribution of the text planner

Recall that the task of a text planner is to determine the content of the text, and to structure this content in the form of an abstract text specification (see Section 5.2.1 for a more precise description). It was argued that the distinction between document planning and microplanning, while fully compatible with the current architecture, is not essential for SLG as far as intonational phrasing and pitch accent placement are concerned. As our interest did not concern the planning process itself, but rather the question what information a

[1] With a view to possible NLG applications, our syntactic realizer is capable of generating text as well, although structure realization is still rather limited.

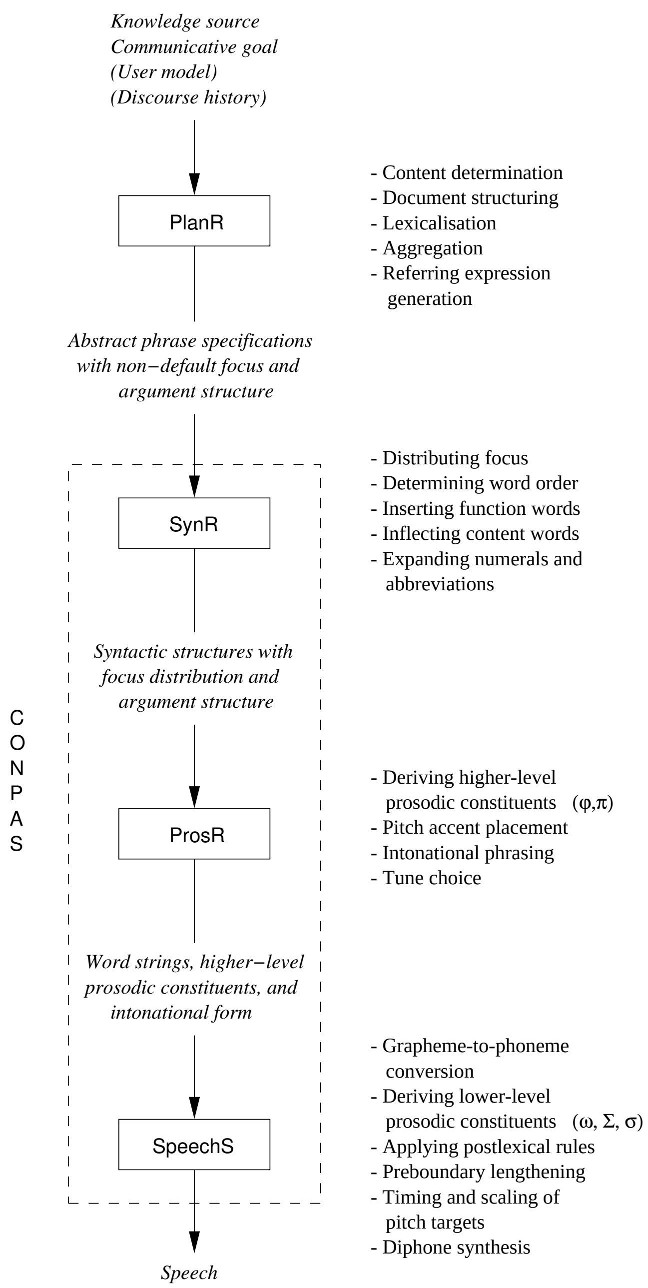

Figure 9.1: Overview of the input/output and tasks of each module in an SLG system based on CONPAS.

text planner should contribute for prosodic realization, we used relatively simple, domain-specific text planners (PTDPlanR and WFDPlanR) in which both planning processes were merged. Moreover, since our SLG systems were primarily intended as a test bed for linguistic theories, we aimed at generating maximal variation on the basis of limited input. This was accomplished using a straightforward approach to text planning, employing template-based generation techniques coded in a general-purpose programming language. This proved to be a concise, reliable and effective approach for our purposes.

The output of PlanR consists of a list of Abstract Phrase Specifications (APS's); we will consider the adequacy of this level of representation in the next section. Besides the abstract syntactic form, PlanR contributes two other types of information to an APS that are important for controlling the intonation. First, it specifies non-default focus, that is, it may explicitly focus or background elements of an APS. This affects the accentuation, as the focus distribution determines initial pitch accent placement. Rather than predicting focus distributions, which is a task for discourse theories, our work concerned their intonational *realization* (see also Section 2.5.1). Therefore, the non-default focus decisions are currently hard-wired in the APS templates used by PlanR. As the final evaluation revealed (Section 8.3.2), some accentuation errors could be traced back to in inappropriate distribution of focus; in particular, placenames were too often backgrounded by PTDPlanR. However, more advanced text planners will certainly be capable of dynamically generating a more reliable focus distribution by keeping track of the information status of entities in a discourse history, combined with pragmatic reasoning about cooperative behaviour, and other factors that may affect a speaker's focus decisions. As a matter of fact, focus has already been used in NLG work on text planning, e.g. McKeown (1985b), McCoy and Cheng (1991), Jokinen, Tanaka, and Yokoo (1998), Fraczak, Lapalme, and Zock (1998).[2] One approach at context modelling specifically for SLG is described in van Deemter and Odijk (1997).

The second type of information contributed by PlanR is the argument structure (predicates, arguments, modifiers). In our approach to accentuation, the argument structure is used by SAAR to decide whether the predicate needs a pitch accent. Like the focus distribution, the argument structure is currently explicitly encoded in the APS templates used by the text planner. It is nevertheless reasonable to assume that this information originates from the text planner, because knowing the number and type of arguments of a verb is essential during microplanning, in particular for lexicalisation and aggregation (Stede 1995). The argument structure must therefore either be listed in the lexicon or encoded in the microplanning algorithms (e.g. decision trees). In addition, we found that the argument structure is required during syntactic realization for determining VP-internal word order (Section 6.3.7.4).

Contrary to pitch accent placement, intonational phrasing appeared to require no additional information from the text planner (apart from abstract

[2] Although, rather confusingly, *focus* is defined in some NLG work as 'that which is being talked about'.

syntax). This suggests that whereas accentuation is closely related to the informational and semantic structure of an utterance, intonational phrasing is independent of it, and relies only on the syntactic structure. Some other approaches to SLG (Steedman 1991; Prevost and Steedman 1994; Hiyakumoto, Prevost, and Cassell 1997) assume that in addition to the focus distribution, the information structure of an utterance consists of a *theme* (what the participants have agreed to talk about) and a *rheme* (what the speaker has to say about the theme), and claim that theme and rheme must belong to separate intonational phrases, characterised by different tunes. However, their examples are limited to a rather specific type of contrast found in a certain type of dialogue. Actually, we have not encountered any examples of intonational phrasing or tune choice in our application domains that suggest the need to incorporate phrasal distinctions related to theme/rheme or topic/comment.[3]

In addition to abstract syntax, focus distribution and argument structure, a text planner may contribute several other types of information that can be used to control the intonation. One of these is the text specification. It is commonly assumed in NLG that text planning results in a hierarchically-structured text specification in the form of a tree whose the terminal nodes are phrase specifications and whose non-terminal nodes are textual units such as *paragraph* and *list*. In PTDGen, for instance, each description of a travel section (from A to B) forms a separate paragraph. In the case of text generation, the surface realizer maps the textual units onto appropriate punctuation symbols and mark-up codes for document presentation. For instance, a paragraph may be signalled by indentation or a blank line, whereas the items of a list may be preceded by numbers or bullets. In the case of SLG, as mentioned earlier, there is no use for either punctuation or visual lay-out. However, there is abundant evidence that the textual structure, often referred to as the global discourse structure (Grosz and Sidner 1986), is signalled by intonational and durational cues. For example, the end of discourse units is signalled by changes in pitch range, pitch register, and pitch contour (Swerts and Geluykens 1993). Likewise, paragraphs are usually closed with a pause that is longer than that between utterances within a paragraph (Lehiste 1975; Lehiste 1979), and intonational peaks are pushed rightwards at the start of a paragraph (Wichmann, House, and Rietveld 2000). Similar findings are reported in Grosz and Hirschberg (1992), Sluijter and Terken (1993), Swerts, Collier, and Terken (1994), Swerts (1994), van Donzel (1999), Noordman, Dassen, Swerts, and Terken (1999). It is obvious that the availability of a text specification in SLG provides an excellent opportunity to explore the relation between text structure and prosody.

In fact, our use of Downstep for simple coordinations (lists), and a %H boundary tone to mark the start of new paragraph can be considered as instances of this relation between text structure and prosody. However, as tune choice is beyond the scope of our research objective, we have not further explored this

[3]We did, however, insert intonational boundaries around appositions. Since an apposition may be interpreted in informational terms as a piece of backgrounded information, this may be considered a relation between informational structure and intonational phrasing.

topic. For the same reason, we did not go into the phonetic implementation of text structure. However, we think that this would require at least two things. First, the textual units (i.e. the non-terminal nodes in the text specification) have to be made available to the speech synthesiser, possibly in the form of prosodic annotations similar to those currently used for AD-boundaries and φ-boundaries. Second, the speech synthesiser needs additional rules for interpreting these textual units in terms of global intonational and durational phenomena (like gradual pitch lowering and pause insertion).

Besides signalling the hierarchical structure of the discourse, intonation can have other discourse functions as well, for instance, signalling the type of speech act. In fact, we made a crude distinction between questions and declaratives to guide the choice of the final pitch accent in an utterance (H^*L or H^*LH for questions versus the default H^*L). Of course, the relation between speech acts and tune choice deserves a more elaborate approach. SLG work by Teich, Haen, Grote, and Bateman (1997) explored the role of intonation in information exchange dialogues, relating tune choice to dialogue acts such as requesting, confirming, and updating information. In addition, tune may serve as a cue to turn-taking in dialogue, which is clearly relevant for any SLG system embedded in a spoken dialogue system.

A cross-fertilisation between intonational and SLG research could result in significant contributions here. On the one hand, SLG systems can be used as a test bed by building text planners that supply the prosodic realizer and speech synthesiser with the type of information that is required for the prosodic realization and phonetic implementation of the discourse structure. On the other hand, some practical formalisations of discourse theory and techniques of implementation have been developed in the field of NLG. In particular, Rhetorical Structure Theory (RST) (Mann and Thompson 1988) has stimulated a lot of work and has proven to be very useful for NLG (Hovy 1988; Hovy 1991). It would be interesting to see what RST has to offer intonational research with respect to discourse prosody.

9.2.3 The contribution of the syntactic realizer

In NLG, syntactic realization refers to translating phrase specifications to grammatical expressions. In our SLG systems, it is taken to include the distribution of (default) focus, and the expansion of abbreviations and numeral expressions. In addition to words, the output must contain the syntactic structures. Syntactic realization by simple string concatenation would fall short of supplying the necessary information, indicating that SLG requires a more advanced form of syntactic realization in order to fully exploit its advantages over TTS. As a matter of fact, in the ongoing discussion about the pros and cons of NLG techniques in comparison to straightforward template-based approaches (Reiter 1995), this is an additional argument in favour of NLG.

We implemented a syntactic realization module (SynR) that meets these requirements by mapping APS's onto syntactic trees.[4] An APS is a skeletal structure consisting of words, their hierarchical relations, and their informational, semantic, and syntactic features. We argued that an APS has the desired level of abstraction to function as input to a content-to-speech system (5.3.3). On the one hand, concrete specifications like canned text or string templates are not an option, because the corresponding surface realizer would fail to provide the syntactic structure required for prosodic realization. On the other hand, highly abstract specifications like lexicalised case frames appear to hold too much information for our purposes.[5]

SynR's output consists of syntactic structures in the form of labelled-bracket structures, enhanced with features for focus and argument structure. We proposed a syntactic analysis of Dutch that relies on minimal assumptions about the phrasal structure, and is compatible with most syntactic theories (Section 2.6). Nonetheless, it proved to provide sufficient input for the deaccentuation and phrasing rules, suggesting that syntactic details such as dominance relations within the verb cluster or among extraposed constituents are irrelevant for prosodic realization.

We will not repeat the discussion on SynR from an NLG point of view here (see Section 6.5), but rather point to three more issues in relation to prosodic realization. With respect to focus, SynR implements a default distribution based on the content/function word dichotomy. In addition, it applies focus projection rules to provide SAAR with the necessary information. Besides its use for pitch accent placement, the focus distribution is also used for determining the word order in a VP: within the constraints imposed by the grammar, non-focused constituents typically precede focused ones. This is just one example of the interaction between word order and accentuation mediated by focus. Thus, instead of deriving pitch accent locations from the syntactic structure, as in earlier treatments (Chomsky and Halle 1968), both syntax and pitch accent distribution are seen as dependent on the focus distribution. Earlier, we mentioned the fact that narrow focus allows a marked ordering of the adjectives in an NP. In addition, there appears to be a relation between the accentuation and the order of the verbs in Dutch verb clusters (Swerts 1999), and certain marked syntactic structures (e.g. clefts and pseudo-clefts) force particular focus patterns and corresponding accentuations (Rochemont and Cullicover 1990; Verhagen 1986). This an area that deserves further investigation, in which SLG can be instrumental.

With respect to numerals, we proposed a new syntactic analysis of numeral phrases in Dutch. It was shown that this analysis provides the required information to derive phonological phrases and apply rhythm rules, resulting in a precise account of the accentuation of numerals (see Section 7.3.5).

---

[4] We partly ignored punctuation and other matters of orthographic realization, as these are – strictly speaking – irrelevant in SLG. Nevertheless, with some additional effort in this direction, SynR can be used as a syntactic realization module for building Dutch NLG systems intended for research, which is a valuable spin-off of the current project.

[5] Section 6.5.1 discussed some additional advantages of APS's over string templates in relation to NLG in general.

Finally, it was found that SynR's output must contain some morphological structure as well. The structure of *phrasal compounds* like *groengeel* 'greenish-yellow', of which the place and number of accents is context-dependent, must be transparent to ProsR. This was accomplished by retrieving these forms from the syntactic lexicon. In addition, the LRR and URR deaccentuation rules rely on features marking exceptional words that also originate from the lexicon.

9.3 The linguistic theory

The prosodic realization module (ProsR) is based on linguistic theory about the relation between the intonation of an utterance and other aspects of its linguistic structure and context. ProsR produces a phonological description of the prosody in terms of two types of abstract representation. The first is the prosodic constituent structure of an utterance. In this respect, we followed prosodic phonology, and its central claim that every utterance has a strictly hierarchical structure consisting of a limited number of prosodic categories (Section 2.2). It was shown that this prosodic structure is instrumental in pitch accent placement (RR applies within φ's), application of rules of phrasal phonology (see Section 5.6.1), and phonetic implementation of the durational structure (preboundary lengthening and pause insertion, see Section 5.6.2). With respect to the theory of prosodic phonology, the main contribution of the current work is a detailed investigation of phonological phrasing in Dutch, which resulted in the proposal of a new algorithm.

The second type of representation generated by ProsR is the intonational structure. Here, we followed intonational phonology in adopting a two-level intonational model: phonological description and phonetic implementation. The former consists of the autosegmental description of Dutch intonation developed by Gussenhoven, which describes the intonation of an utterance in terms of pitch accents, boundary tones, AD's, tone morphemes, and their alignment to the segmental string (Section 2.3). The latter involves a set of phonetic rules that map a phonological description onto an actual pitch contour (Section 5.6.2).

Intonational phonology introduces a convenient layer of phonological abstraction, which allowed us to abstract away from phonetic details, and to focus on generating intonational descriptions at a symbolic level. The contribution of the current work is thus not so much to intonational phonology per se, but rather to the derivation of two aspects of the intonational descriptions: the locations of AD-boundaries, and the locations of pitch accents. With respect to pitch accent placement, we extended the theory developed by Gussenhoven, and subjected it to an evaluation. As far a intonational phrasing is concerned, we evaluated several existing intonational phrasing algorithms, and proposed a new algorithm specifically for Dutch, which was also subjected to a formal evaluation.

Below, we will further discuss these theoretical contributions, thereby addressing the research questions regarding the quality of our linguistic theory about pitch accent placement and intonational phrasing. We finish with some remarks on tune choice.

9.3.1 Phonological phrasing

Our algorithm for pitch accent placement relies in part on RR, which in turn requires that the φ-boundaries are correctly determined. In Chapter 3, therefore, we investigated phonological phrasing in Dutch. It was argued that the φ mapping rules proposed in Nespor and Vogel (1986) and Hayes (1986) are inadequate for Dutch. Therefore, we proposed a new φ-algorithm for Dutch based on the notion of adjacent modification. A careful examination of φ-phrasing in a substantial number of syntactic structures revealed that it was still not entirely adequate in VP's and S's. We therefore revised the definitions of φ-phrasing and adjacent modification.

For evaluation, we used a corpus of flower descriptions. The sentences were manually parsed, and pitch accents were assigned to them by a panel of experts. The algorithm was tested by deriving φ's, applying RR (in addition to SAAR), and checking if the accent deletions that took place were indeed correct. The outcome of this evaluation was that pitch accent placement using a combination of SAAR, φ-phrasing, and RR correctly accounts for 94.11% of all accent decisions in the reference corpus. Considering the length and complexity of the sentences, this appears to be a rather good result. Further analysis revealed more than three times as many undershoot errors as there were overshoot errors. In other words, the algorithm predicted too many rather than too few accents, suggesting that either the focus-based initial accentuation is too generous, or additional rules for deaccentuation are needed. Additional accent deletion rules were in fact introduced in Chapter 7.

As far as RR is concerned, it was found that it applied correctly in 39 cases and incorrectly in 5 cases. Two of these errors could be reduced to an incorrect initial accent assignment, whereas the remaining three may be real errors, requiring further research. Moreover, 19 out of the 42 undershoot errors may in theory be blamed on RR and/or φ-phrasing. It would appear that a postmodifying PP is not always preceded by a φ-boundary, as predicted by our φ-phrasing algorithm, although the exact conditions are unclear. It was suggested that some undershoot errors in compound adjectives can be explained by variability in their accent potential (they can carry one or two accents). Still, these undershoot errors do not provide compelling reasons to modify our φ-phrasing algorithm. We concluded that the proposed φ-algorithm for Dutch is fairly accurate. The later evaluation of SLG output in PTD and NUM showed that none of the errors in pitch accent placement were due to RR, confirming once more the correctness of the our φ-phrasing algorithm.

Although descriptively adequate, the algorithm is still somewhat unsatisfactory when viewed a a set of linguistic generalisations. In particular, we would like to get rid of the special conditions for adverbs and auxiliaries. Applying the algorithm in SLG also indicated that a rule for clitic movement is required to prevent clitics from ending up in the wrong φ. Perhaps an alternative syntactic analysis would allow a generalised statement of both the φ-algorithm and adjacent modification. Alternatively, other formalisations may be explored, for instance, Optimality Theory (Prince and Smolensky 1993), which may turn out to be more suitable; see Post (2000, ch. 4), Truckenbrodt (1999), and Selkirk (2000).[6]

The φ-algorithm is based exclusively on evidence from accent deletions by RR. Potentially, there are two other types of evidence. The first is formed by segmental rules that are bound to apply within a φ. However, it appears to be difficult to find compelling evidence of this type, because such rules tend to be rather sensitive to performance factors like speech rate and speaking style. Second, prosodic phonology claims that the prosodic structure of an utterance is reflected in its the durational structure. More specifically, the prediction here is that φ-boundaries trigger more preboundary lengthening, and are more likely to be accompanied by a pause than non-φ-boundaries. As a matter of fact, this regularity has been implemented in CONPAS. There is experimental work that supports this claim for the 'non-controversial' φ-boundaries, i.e. those predicted by most φ-algorithms (Gussenhoven and Rietveld 1992a; Hofhuis, Gussenhoven, and Rietveld 1995; Cambier-Langeveld 2000). It would be interesting to obtain this type of evidence for 'controversial' boundaries, i.e. those that are unique to our algorithm.

9.3.2 Pitch accent placement

We started with an account of pitch accent placement in the form of a rule-based, linguistic grammar (Section 2.5.4). Its input consisted of the argument structure, the focus distribution, the phonological phrases, and the surface string. The initial grammar had just two rules: SAAR and RR. In Chapter 7, its shortcomings were discussed, and some modifications and extensions were proposed.

9.3.2.1 SAAR

SAAR accounts for the basic focus-accent relation: if focused, every predicate, argument, and modifier must be accented, with the exception of a predicate that, discounting unfocused constituents, is adjacent to an argument. For current purposes, it turned out to be more convenient to reformulate SAAR in two steps: (1) initial accentuation, i.e. accenting all focused constituents, followed by (2) deaccentuation, i.e. deaccenting predicates meeting the condition imposed by SAAR.

With respect to initial accentuation, the focus-accent relation is simply one-to-one for normal words. For *phrasal compounds*, however, the location and number of accents depends on the context, so the focus-accent relation may (initially) be one-to-many. Although this is hardly a problem from a theoretical

[6] However, implementing OT for SLG may not be that easy. At any rate, blindly generating candidates and applying constraints until only one is left is an inefficient process from a computational point of view.

point of view, it requires special care in the implementation. Since ProsR can only accent terminal nodes, it must have access to the internal structure of phrasal compounds. This is accomplished by having SynR deliver the internal structure of phrasal compounds, while no internal structure is generated for normal words. Similarly, some additional processing in SynR was found to be necessary for the correct accentuation of a compound verb, since its verbal part and its particle (which receives the accent, if any) may occur at separate positions in an utterance.

SAAR is formulated in semantic rather than syntactic terms. The problem with this is that a focused argument can be a phrase consisting of many words and possessing a complex internal structure. In fact, it can contain other semantic constituents itself. This raises the question which parts of a phrase have to be focused in order to count the whole phrase as focused. For instance, if only a premodifier within an NP is focused, while the nominal head is not, the NP does not constitute a focused argument as far a SAAR is concerned.

We changed the perspective somewhat by assuming that focus is assigned to words rather than to semantic constituents. A set of focus projection rules was introduced to project focus upwards onto the syntactic phrases fulfilling the semantic roles. Most of these rules fit the pattern: a phrase is focused if its syntactic head is focused. As a result, ProsR can easily identify semantic constituents that are focused when executing the deaccentuation step. In addition to the known exceptions to SAAR described in Gussenhoven (1992b, Appendix) and Faber (1987), compelling examples were found that an AD boundary between a predicate and an extraposed argument prevents deaccentuation of the former. Also, the distinction between arguments and modifiers proved to be fuzzy in some contexts. We argued that a predicate's argument structure may depend on its contextual interpretation. This is an issue that needs further exploration and formalisation before it can be implemented.

In the final evaluation reported in the previous chapter, we encountered a number of cases in which the accent on a predicate was not deleted. Although most of these could be traced back to erroneously backgrounding of arguments, a few remain unexplained. In contrast, we did not find any SAAR overshoot errors.

9.3.2.2 Additional deaccentuation rules

As already mentioned in relation to phonological phrasing, RR showed only a few overshoot errors when applied to the corpus of flower descriptions. More undershoot errors were encountered, however, suggesting that additional deaccentuation rules were required. While applying RR to the domains of PTD and NUM during the final evaluation, we did not encounter any undershoot or overshoot errors. We consider this a good result, because it means that the complex combination of syntactic analysis, φ-derivation, and RR still produces the correct predictions when applied to new utterances in a different domain.

In addition to RR, we implemented the *Lexical Rhythm Rule* (LRR) for a very limited class of quantifiers and adverbs for which the location of the pitch accent depends on the right-side context.

During the development of the SLG system, we encountered accents that should be deleted, but were beyond the scope of SAAR, RR and LRR. One type of undershoot error is the lack of an accent on an NP that functions as a premodifier of another NP (e.g. *perron* in *perron vijf* 'platform five'). To account for this, we introduced the rule *NP Premodifier Deaccentuation* (NPPD).

We also explored the accentuation of numerals in considerable depth. It was shown that their accentuation can largely be accounted for with the independently motivated φ-algorithm and RR. However, this still failed to explain the lack of an accent on 'units' such as *honderd* 'hundred' and *duizend* 'thousand'. We therefore proposed the *Unit Rhythm Rule* (URR), and showed that its use is not limited to numeral expressions, but extends to other phrases containing units, such as expressions of time and price.

The final evaluation in PTD and NUM showed that the only substantial errors caused by rhythm rules are all due to URR. While half of these could be traced back to other errors, a few errors remain unexplained, in particular, four cases in which for some reason the unit *honderd* 'hundred' is not deaccented.

Finally, there appears to be interaction between pitch accent placement and intonational phrasing. With regard to SAAR, we found that an AD-boundary between a predicate and an extraposed argument blocks the deaccentuation of the predicate. Similarly, deaccentuation of a unit by URR is prevented by an intervening AD-boundary. The problem is that this contradicts the assumption that pitch accent placement *precedes* AD-derivation, leading to a theoretical rule-order paradox for which we have no solution. Interactions of this type remain a challenge to be addressed by future research.

9.3.3 Intonational phrasing

The intonational phrase is a prosodic constituent (π) placed between the utterance (U) and the phonological phrase (φ) in the prosodic hierarchy. It is defined on the basis of three criteria: it forms the domain of a coherent intonation contour, a pause may be introduced at its end, and it serves as the application domain of certain segmental rules. We summarised Gussenhoven's arguments against this view, showing these criteria can be contradictory, and therefore that the stretch of speech covered by an intonation contour cannot consistently be identified with a particular prosodic category (Section 2.3.7). Following Gussenhoven, we assumed a constituent corresponding to a coherent intonation contour that does not define a layer in the prosodic hierarchy: the Association Domain.

In Chapter 4, several algorithms for intonational phrasing were systematically compared by means of a partly automatic method of evaluation against an intonationally annotated corpus of flower descriptions. It was found that the algorithms proposed by Nespor& Vogel ($\mathcal{N}\&\mathcal{V}$), by Capers ($\mathcal{CAS}$) and by Gussenhoven ($\mathcal{GUS}$) could not generate intonational phrases that were as good as the reference phrasing produced by the human experts. After an extensive analysis of the errors, we concluded that what is needed is a syntax-driven algorithm for intonational phrasing, which is sensitive to particular syntactic configurations, the accent distribution, and the weight of AD's.

In Section 4.8.1, we proposed an algorithm ($\mathcal{NEW}$) along these lines consisting of eight intonational phrasing rules. Each rule describes a syntactic pattern and a place to insert the boundary. There are rules for inserting boundaries in appositions, sentences, verb phrases, noun phrases and coordinations. It was shown that $\mathcal{NEW}$ correctly accounts for 97.66% of the boundary decisions in the corpus of flower descriptions. Statistical analysis showed that $\mathcal{NEW}$ performs significantly better than the algorithms proposed by Nespor& Vogel ($\mathcal{N}\&\mathcal{V}$), and by Capers ($\mathcal{CAS}$).

In addition, $\mathcal{NEW}$ was implemented in ProsR and evaluated in the domain of travel descriptions and numerals. It was found that $\mathcal{NEW}$ agrees with four human experts in 96% of the boundary decisions (ignoring optional boundaries). Differences between ProsR and the human experts showed that ProsR tends to generate too few boundaries, but inserts virtually no erroneous boundaries. In NUM, an undershoot of the rule AD-in-Coordination was observed, which could be partly explained by interaction with accentuation errors. In PTD, the majority of the errors was due to undershoot of the rule AD-in-S-2, which failed to insert a boundary before an extraposed PP. Again, many of these could be reduced to the accentuation errors.

9.3.4 Tune Choice

Although we chose to abstract away from issues of tune choice, it is an essential part of any SLG system. We therefore implemented a provisional algorithm that chooses the type of pitch accents and boundary tones, and inserts tone morphemes like downstep. It delivers tunes that are acceptable, but lack the variation, expressiveness and liveliness of those produced by skilled human readers. They can be improved by varying the type of pitch accents, and employing tone rules/morphemes like *Tone Linking*, *Half Completion*, and *Narration* (Gussenhoven 1988). Part of the work of tune choice for TTS reported in (Terken 1993) may be reinterpreted in our intonational model. However, to use various tunes in a sensible way, we need to know how they relate to semantics and pragmatics of the utterance. While this is an area that is largely unexplored, some results for English (Pierrehumbert and Hirschberg 1990) and Dutch (Gussenhoven 1984; Keijsper 1984a; Keijsper 1984b; Caspers 1988; Caspers 1998; Grabe, Gussenhoven, Haan, Marsi, and Post 1998) are available. Of course, an SLG system offers an good opportunity for testing theories about meaning and intonation.

In a preliminary attempt to enhance the aforementioned tune choice, we proposed that downstep is preferable for simple coordinations (in which each non-final conjunct has at most one accent). Also, we found that a high initial boundary tone can be used to indicate the start of a paragraph.

We found that some tunes could not be realized within the current theoretical framework and its implementation. First, it was observed that apart from the normal type of downstep, there is a variant that applies only to the 'non-primary' accents of the conjuncts of a coordination, leaving the 'primary' accents unaffected. This variant may be incorporated into the phonological description as a new tone morpheme, with an appropriate phonetic implementation. Second, we found that appositions cannot be accounted for within the current system. These are typically accompanied by temporary lowering of the register: the overall pitch level drops at the start of the apposition, but returns to the previous level when it is finished. This type of tune may also be modelled by an extra tone morpheme for pitch reduction.

9.4 Practical implications

Anyone who is familiar with speech synthesis applications will probably agree that their intonation, and their prosody in general, needs to be improved significantly in order to match human performance. Not only does good prosody contribute to the perceived naturalness of synthetic speech, there is also a growing amount of experimental evidence that it facilitates human parsing of the (possibly ambiguous) syntactic structure of utterances (Ladd 1986; Price, Ostendorf, Shattuck-Hufnagel, and Fong 1991; Beach 1991; Kelly 1992; Grabe, Warren, and Nolan 1994; Sanderman 1996; Marsi, Coppen, Gussenhoven, and Rietveld 1997) interpreting its informational structure (Birch and Clifton 1995; Brown 1983; van Donzel 1999; Fowler and Housum 1987; Nooteboom and Terken 1982; Nooteboom and Kruyt 1987), determining the discourse structure (Lehiste 1975; Lehiste 1979; Swerts and Geluykens 1993; Sluijter and Terken 1993; Swerts, Collier, and Terken 1994; Swerts 1994; van Donzel 1999; Noordman, Dassen, Swerts, and Terken 1999), and inferring the attitudinal and emotional state of the speaker (Grabe, Gussenhoven, Haan, Marsi, and Post 1998; Caspers 1998).

Speech synthesis has traditionally been applied in a TTS environment. The first step in TTS is preprocessing and linguistic analysis of the input text, which typically includes part-of-speech tagging and syntactic parsing, as the syntactic information is used for grapheme-to-phoneme conversion and prosody assignment. It is common knowledge in NLP, however, that syntactic parsing of unrestricted text is still an unsolved problem. Notorious ambiguity problems such as PP-attachment and coordination can cause a parser to consume an excessive amount of time, and output a large number of alternative analyses. In addition, there are syntactic phenomena, such as apposition and gapping, which are difficult to account for. Moreover, the input may even be ill-formed, requiring a robust form of parsing. Therefore, syntactic parsing in TTS is usually limited to shallow parsing, leaving a considerable part of the syntactic structure underspecified.

Unfortunately, what is needed for controlling the intonation appears to be primarily the type of linguistic information that can *not* be automatically extracted from a text.[7] Our intonational phrasing algorithm, for instance,

[7] In fact, this may not be a coincidence. Given the assumption that there is some similarity

requires a correct syntactic analysis of PP-attachment, coordination, and appositional elements, whereas our accentuation algorithm relies, among other things, on a detailed analysis of NP-premodifiers.

Moreover, merely POS and syntactic information is not sufficient, as predicting intonation requires semantic and discourse information as well. Assuming the correctness of SAAR, accentuation of the predicate requires identification of its arguments. Accentuation requires focus/background information, or some other type of information about the discourse status of domain entities. Although attempts to keep track of the discourse have proved useful to a certain extent (Horne, Fillipson, Johansson, Ljungqvist, and Linström 1993; Hirschberg 1992; Hirschberg 1993; Hiyakumoto, Prevost, and Cassell 1997), reliable parsing of the discourse structure is still beyond our reach.

The lack of detailed linguistic information has its impact on the naturalness of synthetic speech as well. TTS systems are made to avoid disturbing, obvious errors. The safest intonation is the one that occurs most frequently. Consequently, these systems fail to employ intonational variation, which is probably one of the main reasons that they sound unnatural and tedious in comparison to competent human speakers.

To sum up so far, the current state of NLP technology does not allow us to build TTS systems that automatically extract all the required information from unrestricted text. Consequently, applied TTS systems have to rely on POS tags and shallow syntactic analyses, and resort to statistical regularities or handcrafted heuristic rules for prosodic realization. Although these approaches take us a long way, and are certainly of great value for building practical TTS systems, they appear to offer less perspective for improvement in the long run. The same view is expressed in the following quote from Taylor and Black, a comment on their statistical model for assigning phrase breaks from POS tags in TTS:

> "While we believe more investigation into tagset composition would help in reducing errors we also believe that there is only so far that superficial text analysis techniques like this can go. From examining the errors in the test set we believe that a more sophisticated analysis will be needed to correct some of the errors. Although we have argued in the introduction against using syntactic parsers for phrase break assignment, our reasons stem from the basic inaccuracy of these parsers, not because syntactic parses themselves are unhelpful."[8] (Taylor and Black 1998)

Clearly, SLG has an advantage over TTS as far as the availability of linguistic information is concerned. The aforementioned problems with respect to

between a computational parser and the human language processor, both will encounter the same type of problems with regard to parsing ambiguous utterances. Adding intonation, and prosodic structure in general, may in part be a strategy to deal with ambiguity problems.

[8]It should be noted that Taylor and Black do not necessarily advocate a rule-based approach to prosody assignment, but rather acknowledge the fact that more elaborate input is required.

automatic linguistic analysis do not exist in SLG, due to the fact that the NLG module is capable of producing not only text, but also the linguistic structure and context of this text. The same point has been made by a number of other researchers, e.g. Bateman (1997), Hess (1997), McKeown and Moore (1995), Sproat (1997), and McKeown and Pan (to appear); in fact, the idea goes back as far as Young and Fallside (1979). This potential of SLG for generating better prosody holds a promise for practical application of speech synthesis. This view is supported by the fact that the experimental evaluation showed that the SLG system outperforms the tested TTS system (most clearly with respect to accent placement).[9]

Yet, our SLG system is a laboratory project which is primarily intended as a testbed for linguistic theory about intonation. In its current state, the content-to-speech component CONPAS is not ready for practical application. Although the current system has so far been applied to two quite different application domains, which we consider representative of instructional types of text, other domains will undoubtedly reveal new challenges with respect to intonation. In addition, the type of output that is generated is essentially written text that is read aloud. From the viewpoint of ergonomics or human-computer interaction, an SLG application should rather produce the type of language found in speech communication, for instance, with shorter utterances and a more liberal syntax. Finally, little attention has been paid to performance issues, and the system is too slow for on-line tasks. Nonetheless, we believe experimental systems like ours provide the foundation for the future development of applied SLG systems.

Of course, SLG is no substitute for TTS, as there will always be a need for reading aloud existing text. However, building an SLG system that generates prosody satisfactorily will at least tell us what kind of linguistic information is required for prosody assignment, and therefore, what type of information should be automatically extracted from the input text during the linguistic analysis step in a TTS system. SLG may thus serve to set the goals for linguistic analysis in TTS.

[9] Given our focus on intonation, we tend to forget the advantages of SLG with respect to text preprocessing, all the more because they look trivial. For instance, homographs with different pronunciations (e.g. the numeral *een* 'one' versus the determiner *een* 'a') are easily resolved from the available syntactic structure. Likewise, there is no need to expand numerals or abbreviations, because the NLG systems provides them in their full lexical form. Also, end-of-sentence detection, which is sometimes difficult in TTS due to ambiguity of punctuation marks, is a non-problem in SLG.