

Chapter 9.

Automatic Phonetization.

In this chapter, we will assume that the Grapheme-To-Phoneme (GTP) module is provided with the syntactic information it requires (which almost corresponds to the output of the LIPSS text analyzer in its current state of development, except that accent groups are not yet defined), together with the text to transcribe phonetically. It is then possible to organize its task in two ways.

Dictionary based solutions consist of storing a maximum of phonological knowledge in a lexicon. In order to keep its size reasonably small, entries are generally restricted to morphemes, and the pronunciation of surface forms is accounted for by inflectional, derivational, and compounding morpho-phonological rules which modify the phonetic transcriptions of their morphemic constituents. This approach has been followed by the MITTALK system [Allen et al 87] from its very first day. A dictionary of up to 10,000 morphemes covered 95% of the input words. Others were transcribed by rule. In a word : dictionary is the rule, while rules are the exception.

A radically different strategy is adopted in *rule based* transcription systems, which transfer most of the phonological competence of dictionaries into a set of rules. Words which are pronounced in such a particular way that they constitute a rule on their own are stored in an exception dictionary. In accordance with the *reduced lexicon size* principle introduced in Chapter eight, it is the solution we have adopted.

Inferring rules from data in rule based systems can in turn be performed in at least three ways :

- *Expert systems* are the first one. They generally incorporate some knowledge extracted from human experts, or phonetic treatises in our case. Since *international* languages have been extensively studied by linguists far before the outgrowth of computers, this approach is the most often adopted one in related TTS systems. Up to now, it is also the one that provided the best results. Most existing grapheme-to-phoneme systems for French are based on this kind of approach. In the TOPH system [Aubergé 91], for instance, 1270 rules and 1650 exceptions account for 99.8% of the 60,000 words of the Petit Robert dictionary.
- Things are not the same when transcribing *nation-wide* languages or when developing multilingual TTS systems. *Trained rule-based systems* have thus emerged, for which human experts are no more required for rule development (some human-like clear-sightedness is still necessary for data preparation anyway). When provided with some human feedback, in the form of post-training corrections, they constitute an interesting alternative to the previous ones.
- *Neural Networks* have been recently proposed as an alternative to explicit rule based approaches. They constitute the most automatic but hidden way to proceed. They differ from the previous systems in the sense that rules implicitly appear in the form of a list of numbers (usually the synapses weights in a multi-layer perceptron), hardly understandable (neither corrigible) by humans. Given the complexity of the task to perform here, they cannot compete with rule-based systems, at least for the time being.

After a short introduction to these methods, in which their respective inherent limitations will be highlighted, we shall focus on expert systems and develop our own strategy in producing automatic phonetic transcriptions of French sentences, for we believe that ignoring the considerable amount of analysis work that has been done by linguists in that domain would simply be nonsense.

9.1. A survey.

The problem of developing a rule based system which produces automatic phonetization can be considered as a data reduction process that is assumed to have generalization properties. Starting from an important database of words and/or sentences and their transcriptions, the system is supposed to end up with a very limited set of fairly consistent rules, which are believed to be applicable on words and/or sentences that were not in the analysis database. The leading hypothesis in rule based systems (either trained or not) is that data reduction is on a par with generalization properties.

For simplicity, let us consider a training database composed of words only (the case with complete sentences is simply a generalization of this one, in which punctuations

and space characters (' ') are additionally considered), and let us exclude heterophonic homographs, for the transcription of which extra-lexical (syntactic in this case) information is required. As an example, let us imagine that our database contains items like :

'manger'=[mâZe], 'tondu'=[tôdy], 'panier'=[panje], 'taxi'=[taksi]

9.1.1. Classification criteria.

It is generally accepted that it is possible to account for all (*word, transcription*) couples with a limited set of regular translation rules [Aubergé 91], which implies that it can be achieved with simple finite state transducers, without the need to express input strings as hierarchical structures before being able to transcribe them¹. Deriving the expected set of regular rules then confines itself to making the following choices :

1. *In each word, which sequences of characters correspond to which sequences of phonemes ?*

One immediately notices that there is no one-to-one character-to-phoneme mapping for each word (i.e. each character cannot be associated with a single phoneme, and *vice versa*). Not all characters in 'manger', for instance, contribute to a distinct phoneme. Likewise, not all phonemes in [taksi] originate in a distinct character. As a result, aligning text strings on phonetic ones implies either that *void* characters or phonemes (' ') are accepted, like in :

'manger'=[mâ_Ze_], 'tondu'=[tô_dy], 'panier'=[panje_], 'tax_i'=[taksi]

or that groups of characters and phonemes are accepted as ultimate units (i.e. as ultimate *characters* in their respective languages), like in :

'm/an/g/er' = [m/â/Z/e], 't/on/d/u' = [t/ô/d/y], 'p/a/n/ier' = [p/a/n/j/e],
't/a/x/i' = [t/a/ks/i]²

¹This was not possible with the *Generative Phonology* formalism of [Chomsky & Halle 68], in which text was progressively transformed into phonemes by applying rewriting rules in a sequential way (i.e. the result of one rule was taken as the input of the next one).

²It is straightforward that both choices can be done in a variety of ways. One could *a priori* have chosen to write :

'manger'=[mâ_Ze_] or [m_âZe_] or [mâ_Ze_] or [m_âZ_e]

as well as :

'm/an/g/er' = [m/â/Z/e_] or 'm/an/ger' = [m/â/Ze] or 'ma/ng/er' = [m/âZ/e], ...

The corresponding groups of characters (possibly single ones) are often termed as *graphemes*. It should be clear that, if mono- and multi-character graphemes are simultaneously accepted, users (or the rule interpreter in this case) should be provided with a way to recognize them in a word. Why does 'manger', for example, correspond to the graphemic decomposition 'm/an/g/er' rather than 'm/a/n/g/er' ? In other words, one needs a *graphemic syntax*.

Whatever solution chosen, and whatever means adopted to fix the corresponding one-to-one mappings, conflicts will inevitably arise, in the form of identical graphemes being transcribed differently throughout the database.

2. *Assuming above correspondences have been fixed, what type of information will be used to resolve conflicts ?*

Answering this question implies to choose a set of phonemically discriminative features and therefore it conditions the linguistic insight one expects to acquire from the resulting rules.

A first approach consists of solving conflicts by inspecting the graphemic contexts in which they occur. It typically leads to rules of the form :

$$a \rightarrow [b] / l _ r \quad (9.1)$$

which express that grapheme *a* is transcribed into phoneme(s) [b] when surrounded by character strings *l* and *r*.

Alternatively, one could appreciate the fact that rules more directly highlight some linguistically pertinent properties of words, such as syllable break positions, part of speech categories, semantic features, or even etymological origins. This can be achieved by adding conditions to rules (9.1), so as to restrain their applicability :

$$a \rightarrow [b] / l _ r : \text{condition} \quad (9.2)$$

As we shall see, all the transcription systems developed up to now (with the exception of the generative phonology approach, which has never really been used in TTS synthesis) can be positioned with respect to one another by the peculiar answers they have brought to questions 1 and 2.

9.1.2. Pronunciation treatises.

A number of linguists have proposed, and more or less succeeded, to establish a set of transcription rules covering the whole French language. It is striking, however, to compare the resulting pronunciation treatises, since they account for the same facts in very different ways. What is more, they often assume an important prior knowledge from their potential readers.

All of them group characters into mono- and multi-character graphemes. Even though many graphemes have found a wide assent (think of 'an', for example), a lot of them (generally the less frequently encountered ones) remain questionable. In addition, no graphemic syntax is generally provided explicitly : graphemes are listed, but users are supposed to deduce a parsing strategy by themselves³. It can be assumed, for example, that if 'oi' and 'oin' are both accepted graphemes, then rules accounting for the transcription of 'oin' should first be tested when possible. In case no match is found, then rules for 'oi' are the next candidate. Better still, in case two graphemes are found to overlap in a word (as 'cc' and 'ce' in 'accepter'), it may be granted that, given a left-to-right transcribing direction, the application of a rule for 'cc' prevents 'ce' from being examined⁴. Clearly, graphemic syntax is given in these cases as a by-product of rules.

Finally, the rules proposed by expert linguists are mostly of the type (9.2), in which the additional conditions are subject to wide variations from one person to another, depending on the linguistic features chosen to solve transcription conflicts. [Bourciez 58], for instance, mainly develops an etymological description, the purpose of which is to establish what phonological transformations of Latin words have led to our current French vocabulary. The most automation oriented works we have found are [Nyrop 63] and [Leon 66]. The latter presents phonetization in the form of structured tables, driven by a rigorous list of a hundred graphemes or so. [Nyrop 63] proposes 232 contextual rules, augmented with exceptions lists originating in etymological or syntactic considerations when context does not suffice. We also found [Delattre 51] highly profitable, for the fairly precise account he gives of phonetic liaisons in French.

9.1.3. Expert systems.

Expert systems are based, as their name literally infers, on rules proposed by expert linguists, as the ones examined in the previous section. Their main contribution is the derivation of explicit phonetization procedures, which in turn implicitly define the graphemic syntax used. In most cases ([Divay & Guyomard 77] and [Aubergé 91] are examples), text is scanned from left to right and rule examination is ordered in two ways :

- bigger graphemes are tested first (cf. above for 'oin', 'oi');
- when several rules exist for the same grapheme in different contexts, rules are tested in the order in which they appear in the rule database. Rule ordering is carefully designed, so that pronunciations are scanned from the most particular to the most general (i.e. from the biggest context to the smallest).

³In that respect, pronunciation treatises are similar to traditional grammars : they address human readers, not machines.

⁴This is often referred to as *bleeding* in computational linguistics.

In some cases, syllabification is proposed as an explicit grapheme isolation method ([Goyer et al 79], [Catach 84]). Choosing between 'a/n' and 'an' in 'chantier' or 'canard', for example, is accounted for by syllabic decomposition : 'chan|tier', 'ca|nard'. Syllabification is itself described in terms of character-based rules (it does itself respect a regular syntax), and a set of exceptions stored in an appropriate lexicon.

Finally, some automatic phonetization programs are based on mono-character graphemes (like [Leroux & Miclet 79]). Graphemic syntax becomes obvious (one character = one grapheme). Rule ordering is maintained, and defined by an expert.

On the word level, the TOPH system of [Aubergé 91] has achieved the biggest success rate up to now (99.8%, see above). Its performance on complete sentences is significantly lower, for it does not include syntax-driven rules.

9.1.4. Trained rule based systems - Neural networks.

It is striking to see how, as a result of the heterogeneity of the linguistic literature on phonetic transcription, experts systems can be different from one another. This is expressed by :

- different sets of graphemes;
- different rule formalisms and rule application procedures;

Trained rule based systems lead to more uniformity. Most of them are based on a mono-character approach, since, as stated above, designing multi-character graphemes requires to simultaneously define a graphemic syntax. Text-to-phoneme alignment is generally performed manually (which is not trivial either). The resulting training database can then be straightforwardly described by a maximal canonical transducer, a transducer version of the maximal canonical grammar that was presented in Appendix II (Fig. A.II.3). The problem of trained rule based systems then confines itself to applying automatic rule inference procedures to derive a simpler finite state transducer from the trivial one.

In [Torkkola 93], for instance, this is achieved with a simple dynamic context extension strategy. The transcription of each character is considered separately. A list of all the possible phonemes (including '_') corresponding to it, is established and numbered. The couple (*character*, *phoneme*) which is most frequently encountered is then chosen as reference. All other couples have to be accounted for by expanding the context one letter to the right or to the left (the expansion strategy is known *a priori*). The same context expansion process is then applied for the remaining couples, until no more conflicts are encountered or a maximum context width is reached. The method is clearly sub-optimal, in the sense that the data reduction they allow is highly conditioned by an *a priori* context expansion order, which is furthermore fixed for all characters. Its swiftness, however, allows a lot of different strategies to be tested.

Other algorithms are presented in [Hochberg et al 91] and [Lucassen & Mercer 84], and [Klatt & Shipman 82].

The advantage of such automatic regular inference approaches is that they are somewhat language-independent. What is more, many of them allow a hierarchical description of rules. The transcription of a given character is then searched in a multi-level tree structure, in which levels correspond to increasingly wide contexts.

In parallel, self organized/connectionist architectures such as NETTALK [Sejnowski & Rosenberg 87], NETSPEAK [McCulloch et al 87], or the networks of [Matsumoto & Yamaguchi 90] and [Ainsworth & Pell 89] have been recently proposed as an alternative to rule based systems. As above, they make use of mono-character graphemes. Graphemic windows (typically seven characters wide) are presented, in a coded way, at the input of a multi-layer perceptron, which is trained to produce the phonetic transcription of the central character.

As for trained rule based systems, neural solutions are *a priori* interesting for their multi-lingual capability and their computational speed.

Drawbacks, however, are numerous (either for rule or neural approaches). Since each character is associated a single phoneme (including '_'), some of the graphemes traditionally proposed by phonetization treatises are distributed on several characters (think of 'eau', when transcribed into [_O_]). Consequently, data reduction is intrinsically limited. What is more, automatic handling of syntactic features, as needed to separate heterophonic homographs and to correctly apply phonetic liaisons, is difficult to incorporate. Finally, the number of errors per phoneme is increased (in the case of 'eau', three rules have to apply correctly for just one phoneme). Results are so bad that error rates are reported by letter or by phoneme : 92 % in the best case, which corresponds to a word error rate of approximately 50 %. This is clearly unacceptable in the context of high quality TTS synthesis.

9.2. Grapheme-To-Phoneme transcription with LIPSS.

As already said, we have chosen the rule based expert system approach for the LIPSS GTP module. Our main sources were [Nyrop 63], [Delattre 61], and [Léon 66], together with a phonetization program previously developed in the TCTS labs [Driemmel 86]. Our reference dictionaries were [Warnant 64] and [Robert 72]. The general principles we have followed throughout the development stage were very similar to the ones adopted for morphological and distributional analysis :

- **Readability** : As far as possible, transcription rules had to be expressed in a simple way. In that respect, a declarative formalism was welcome. As it will

appear below, most transcription rules have been written as Prolog facts, stored in a text database. They can be modified without the need to re-compile the whole program. We found it more convenient, however, to write some of them (the more complex ones) in the form of Prolog clauses.

- **Reduced memory size** : The data reduction process we have introduced above had to be maximized, by optimizing rule coverage and minimizing dictionary sizes.
- **High transcription speed** : In order to minimize processing times, exceptions lexis have again been stored in the form of *B+ trees*. Furthermore, the most complex transcription rules were partly expressed in Prolog, for the same reasons that had previously induced us to write distributional rules in the same way. The price to pay for such a hybrid text/Prolog rule implementation approach is some loss of generality, in the sense that switching from French to another language would inevitably require to rewrite some Prolog rules.

The resulting GTP module is a trade-off between these three constraints. Its peculiarity comes from the intensive use it makes of syntactic information, easily obtained by accessing the *part_of_speech* fields of words in the LIPSS multi-layer data structure. Given the partial results currently provided by the Text Analysis module, only the first part of speech category proposed for each word has been considered. The order in which part of speech hypotheses were stored in the related field had been previously arranged so that verbs were systematically proposed *before* nouns and adjectives when a verb/non_verb ambiguity could not be solved at the distributional analysis level.

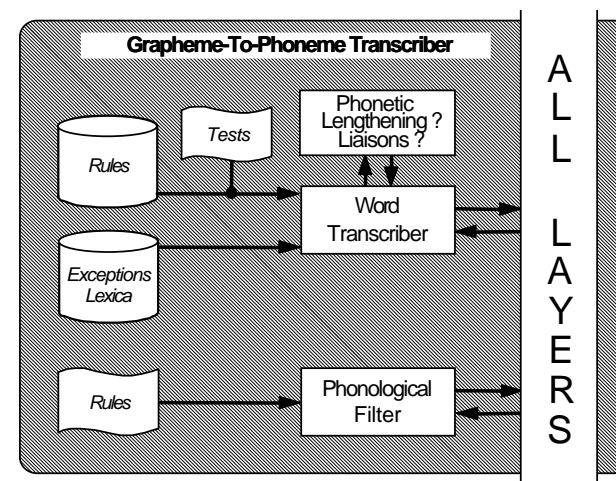


Fig. 9.1. The LIPSS Grapheme-To-Phoneme transcriber. Text and Prolog rules are indicated by cylinders and flags, respectively.

The GTP module is depicted as a two-level process in Fig. 9.1. The list of words is first accessed, on a word basis, by a Word Transcription (WT) module, which produces a first phonetic output for each of them and stores it in their *phonology* field. As it will appear below, we found it more convenient to express some rules as phonemic modification ones (i.e. as contextual phoneme-to-phoneme rewriting rules), and collected them in a Phonological Filter (PF) module which processes the results produced by the WT one and modifies them accordingly.

9.2.1. The Word Transcription module.

Before any other processing, each word is tested as a possible entry in four exceptions lexica (Fig. 9.1), which account for pronunciations that can only be applied on particular words or morphemes. The first one is an enlarged version of what is generally understood as an exceptions lexicon in other GTP systems. It accepts entries of the form :

list_of_previous_words, word, list_of_next_words, pronunciation, part_of_speech

in which *list_of_previous_words*, *list_of_next_words* and *part_of_speech* possibly restrict the left and right context of the word to be transcribed, as well as its part of speech category (possibly augmented with morphological features). This enables the system to test words in their context, in order to account for pronunciation exceptions that cannot be resolved at the word level ('*le cas échéant*', '*nuit et jour*', ...⁵). Furthermore, it allows to impose syntactic restrictions for some pronunciations ('*est*' = [E] if verb, '*os*' = [O] if plural).

However, in order to speed up exceptions spotting and to minimize our lexicon size, we have created three additional lexica, addressed by morphological roots, and corresponding to peculiar part of speech categories. They are : the function word lexicon, the noun/adjective lexicon, the verb lexicon. Each one contains items of the form :

morphological_root, Number_of_characters, pronunciation

which is understood as : "The *Number_of_characters* first characters of a word with *morphological_root* as canonical form are pronounced as *pronunciation*". As a result, complete families of words can be accounted for by a single lexical entry (all the forms

⁵see [Delattre 51] for a list of such contextual pronunciation.

of '*boycotter*', for instance, are simply deduced from *boycotter*, the seven first characters of which are pronounced [bOjkOt]).

The number of words currently included in the GTP exceptions lexica are : 51 for the full form dictionary, 31 function words, 456 nouns and/or adjectives, and 47 verbs. They are still being completed.

When no exception has been detected, rules come into action. Our 233 word transcription rules have the same format as given in (9.2) : graphemes are transcribed into particular phonemes, depending on their context, and provided an additional condition is verified. They are listed in Appendix V (read now the introduction of Appendix V, in which the format of the rules that will subsequently be quoted as examples is explained). A number of writing conventions have been adopted, some of which allow to keep the number of rules relatively small (so as to increase the readability of the whole set) :

- When no left or right context is encountered, any one is accepted.
ex : 195. **ss** S *assez, essai, ...*
- Word beginning and endings are indicated as ' _ '. Void pronunciations are denoted as ' ' '.
ex : 99. **g** _ *sang*
- When several left or right contexts have the same effect on the pronunciation of a grapheme, they all contribute to the same rule, separated by commas.
ex : 34. ar,or **ch** K *orchidée, archange*
- When a grapheme results in several phonemes, they are separated by commas.
ex : 71. _ **emm** AN,M *emmancher*
- Generic consonants and vowels are marked as upper case C and V in left and right contexts.
ex : 20. **ay** C,_ E2 *aymé*
- Possible liaisons are indicated by '|', followed by the phonemes pronounced if the liaison is accepted. In the particular case of un-nasalizations, nasalized forms are quoted first, and followed by '|,N'.
ex : 158. **on** _ ON,|,N *mon*

- Possibly mute 'e's are denoted as '(E)⁶. The final decision is left to the Phonological Filter module.

ex: 54. _r e ss (E) *ressembler, très ressemblant*

As we mentioned in Section 9.1, most automatic phonetization programs operate from left to right, even though the analysis direction is more used as part of an implicit graphemic syntax parser than really based on psycho-linguistic considerations. In LIPSS, transcription is performed from right to left, for reasons that will appear below. Each word is first added a word beginning and a word ending character (denoted as ' _ '), and a window is applied on it, starting from its end. Since graphemes have been kept smaller or equal to three characters, the window size is initially set to three. Rules (9.2) are then scanned with the current window as component *a*. If a match is found, [b] is written in the *phonology* field of the current word, and the analysis window is shifted *N* characters to the left, *N* being the number of characters in *a* (three in this case). If not, the window size is decremented (i.e. the first character is dropped), and rules are scanned again, and so on until a match is found (which always happens, as all single characters are graphemic entries in the rule database). Each time the analysis window is shifted, its size is reset to three.

Fifty five rules actually make use of the *condition* field, indicated either as a keyword or as a list of part of speech categories written in the internal LIPSS formalism. Each type of condition is now examined.

9.2.1.1. Final 's'.

In French, plural forms in 's' are not pronounced, whether in nouns, adjectives, or verbs, except in liaisons. Since many other rules actually test word endings (see above rule 99), we found it more efficient to systematically extract final 's' when encountered and ask the Word Transcriber to process the corresponding singular form (even though it happens that simply dropping 's' does not always restore a singular form, but this has no effect on pronunciation)⁷.

183. s _ |,Z *grands* final_s

As a result, rules which address word ends can always assume that they are handling singular forms. In contrast, words that have their final 's' pronounced ('os', 'as', 'jadis', 'hélas',..., and a lot of foreign words : 'cacatoès', 'Isis',...) are stored in the exception lexicon.

⁶We shall denote them as [(@)] in the text.

⁷This is the only case in which a *condition* in transcription rules actually results in an *action* which modifies the transcription process, rather than simply in an additional test.

9.2.1.2. Part of speech restrictions.

The ability to express syntactic restrictions to rule applicability is one of the most important features of the LIPSS GTP module. The built in *condition* interpreter simply checks that the word currently analyzed actually belongs to one of the part of speech categories passed in the *condition* field. Eleven rules make use of this facility. The most important ones are :

82.	ent	_	(E),,T	<i>parlent</i>	nat([verb(_,_,_)])
83.	er	_	E1,,R	<i>parler</i>	nat([infin(_)])
144.	o	tion_	O	<i>notions</i>	nat([verb(_,_,_person(prem,plur))])
197.	t	ie_	T	<i>aplatie</i>	nat([parté(_,ém,_)])
200.	t	ion_	T	<i>portions</i>	nat([verb(_,_,_)])
202.	t	ien	T	<i>retiens</i>	nat([verb(_,_,_)])

which directly address the problem of heterophonic homographs and provide a correct processing of sentences like '*les poules du couvent couvent*', '*le président et ses associés président*', '*nous portions des portions*', '*l'est est à l'est*', ... Let us nevertheless recall here that many words are understood by LIPSS as possible heterophonic homographs, given its typical lexical ignorance, even though they are not considered as such by human readers. Thus, even sentences like '*les dévotions favorisent la calvitie*' greatly benefit from syntactically restrained pronunciation rules.

9.2.1.3. Accented syllable restrictions.

It has been recalled in Appendix I that some possible vowel lengthenings, termed as *phonemic lengthenings* [Léon 66], are directly indicated in spelling by circumflex accents. They are, however, only produced if the accent affects the last pronounced vowel of a word, that is the accented one :

3.	â	A:	<i>pâte</i>	accented_syllable
4.	â	A	<i>pâtisserie</i>	

The *accented_syllable* keyword accounts for it, by checking that the current content of the *phonology* field of the current word (i.e. the transcription of all the characters on the right of the current grapheme, since analysis goes right to left) does not include vowel phonemes (semi-vowels and [(@)] not being considered as vowels).

9.2.1.4. open/closed syllable restrictions.

In addition to testing the position of the current grapheme with respect to the word accent, it is sometimes necessary to check that the current syllable is open or not. This typically happens when transcribing 'e' in French (both cases are encountered in '*mercredi*') :

57.	e	C	(E)	vendredi	open_syllable
58.	e	C	E2	verte	

The *open_syllable* keyword forces LIPSS to make the corresponding test, checking the content of the current *phonology* field. If it is empty or if it contains no consonant phoneme, the syllable is open. If not, the number *N* of neighbouring consonant phonemes is counted, up to the next vowel (from left to right). Then :

- if $N=0$, the syllable is open (as in 'table')
- if $N=1$, the already pronounced characters in the current word are tested. If they do not begin with {'ck', 'cq', 'sn', 'sc', 'sd'} or with geminate consonants, then the syllable is open (as in 'vendredi')
- if $N=2$, then the syllable is only open if the corresponding consonants do constitute a *strong consonant cluster* : {[v], [b], [k], [d], [f], [S], [g], [p], [t]} + {[l], [R]} (as in 'entreprise')
- if $N>2$, the syllable is always closed.

As clearly appears, open or closed syllables detection is mostly performed on a phoneme basis, rather than on a character basis, as in most other GTP programs. The advantage of the phonemic approach resides in its compactness (just think that [k] can appear as 'c', 'qu', 'cqu', 'k'...), which itself originates in the fact that syllabification is better defined at the phonemic level.

9.2.1.5. Mute 'h's.

There is no rule in French to decide whether an 'h' is mute or not. On the basis of [Warnant 64], we have established a list of word initials in which h is always mute : {'habi', 'haiti', 'halai', 'hallu', 'halo', 'halté', 'hameç', 'harmo', 'harpa', 'huil', 'huis', 'hui', 'hum', 'hurlu'} and a list in which h is never dropped : {'hâ', 'henn', 'hers', 'heurt', 'hérau', 'héris', 'héros', 'hib', 'hic', 'hide', 'hiéra', 'hiss', 'hob', 'hoc', 'hold', 'holl', 'homar', 'hong', 'hont', 'hoq', 'hord', 'hormi', 'hors', 'hott', 'hou'}. An initial 'h' is then accepted as mute if the related word begins by any of the elements of the first set. If no match is found, it is still considered as mute if the word does not begin with any element of the second set. We found it easier to implement this test with Prolog clauses rather than with text rules.

9.2.1.6. Last Phoneme test.

Some rules can only be applied when transcribing the last phoneme of a word. They are indicated by the *last_phoneme* keyword, which LIPSS understands as a request to check that the *phonology* field of the current word is still empty (again, liaisons and [(@)] are not taken into account). We found it more natural and convenient to express such a constraint on the phonemic level rather than on the basis of right contexts, since

it would have required to give a list of all consonants and consonant clusters that are not pronounced at the end of a word.

147.	o	AU	escroc	last_phoneme
------	---	----	--------	--------------

9.2.1.7. Semi-vowels.

In French, graphemes 'i', 'r', 'y', 'ou', 'u' are pronounced as vowels [i],[i],[i],[u],[y] or as semi-vowels [j],[j],[j],[w],[H], depending on their phonemic context. As a result, all the LIPSS rules that account for the production of semi-vowels are assigned keyword *semi_vowel* as their *condition* field. The application conditions we have applied can be summarized as :

- 'i', 'r', 'y', 'ou', 'u' are semi-vowels if they are followed by a vowel phoneme and not preceded by a strong consonant cluster : {'v', 'b', 'k', 'd', 'f', 'g', 't', 'c'} + {'l', 'r'} : 'chouette'⁸
- 'u' is always a semi-vowel before [i] : 'bruit';

9.2.1.8. The case of 'eu'.

We found it too complex to account for the transcription of 'eu' into [2] or [9] with the simple formalism given in (9.2). Rule 87 therefore merely is a 'memo' rule, which recalls the expert that 'eu' is actually handled by PROLOG clauses :

87.	eu	EU	peut	eu_EU
-----	----	----	------	-------

The test we perform is based on a simultaneous examination of the graphemic and phonemic right contexts, denoted as *x* and *X*, respectively. After [Nyrop 63], we have transcribed 'eu' into [2] :

- if the current syllable is open and not accented (see the related tests in Sections 9.2.1.3. and 9.2.1.4., in which *X* and *x* play a part) : 'deuxième';
- or if it is accented but $X=[]$ (i.e. [2] is the last phoneme of the word) or a possible liaison : 'deux';
- or if it is accented and the first phoneme in *X* is in {[d], [m], [Z], [t], [s], [z], [Z], [t], [s], [z], [g], [k], [l], [r]} : 'émeute';

9.2.1.10. Phonetic Liaisons - Phonetic Lengthening.

⁸Strong consonant groups are detected here on the basis of their spelling, since they have not yet been examined.

As introduced above, possible phonetic liaisons are directly introduced by the transcription rules :

84. ch,i,g er _ E1,|,R léger

In contrast, possible phonetic lengthening⁹ is not produced by default, as it only applies in a limited number of case. For both reasons, when the phonetic transcription of a word has been completed, its last syllable still needs some processing, indicated as a separate block in Fig. 9.1 since it is not directly related to pronunciation rules. We have implemented it in Prolog.

In LIPSS, phonemic lengthening is forced on the accented syllable¹⁰ (and indicated by [:]), provided it is open and one of the following conditions holds :

- the accented vowel is in {[O], [2], [ô], [â], [ê], [û]} : 'peintre';
- the accented vowel is [i] and it is followed by in [j] : 'fille';
- the accented vowel is in {[e], [E], [9], [u]} and it is followed by two consonants, the first being [v] : 'ouvre';
- the accented vowel is followed by a single consonant, included in {[z], [Z], [R]} : 'peur';
- the accented vowel is followed by a strong consonant group of the form {[b], [d], [g]} + {[l], [R]} : 'coupable';
- the accented vowel is followed by a single consonant, included in {[b], [d], [g], [v]}, itself followed by a possibly muted [@] : 'divague';

Deciding whether liaisons (and un-nasalizations) should be maintained or not is a little bit more complex (see [Delattre 51]). Apart from the case of words separated by a hyphen, which always accept possible liaisons ('prends-en'), we have preserved liaisons when the following two conditions are fulfilled :

- the next word begins by either a vowel or a mute 'h', and it is not included in {'un', 'onze', 'onzième', 'onzièmes', 'huit', 'huitième', 'huitième'};

⁹As opposed to phonemic lengthening, *phonetic lengthening* does not originate in graphemic peculiarities. It only affects accented vowels, when followed by given consonants. Therefore, one and the same word can be phonetically lengthened or not in different phonemic contexts, depending its position in the accent group (see Appendix I).

¹⁰Practically, phonetic lengthening should be restricted to the accented syllable of accent groups. In the current status of the LIPSS system, accented groups could not be accurately detected. We have temporarily assigned accent group limits to punctuation marks.

- the liaison occurs between words which are syntactically related :
 1. determiner + noun;
 2. determiner + adjective;
 3. adjective + noun;
 4. adjective + adjective;
 5. w word + w word;
 6. w word + verb;
 7. verb + w word;
 8. preposition + any part of speech category;
 9. non adjectival adverb + any part of speech category;
 10. noun(plural) + adjective(plural)¹¹;
 11. any part of speech category + past participle;
 12. verb('avoir') + any part of speech category;
 13. verb('être') + any part of speech category;

9.2.2. The Phonological Filter.

As such, the Word Transcription module does not easily allow to express phonemic constraints (cf. above : all the rules that access the *phonology* field are expressed in Prolog), as typically needed to account for vowel harmony. Neither is it able to account for pronunciation rules that affect a word on the basis of the pronunciation of the following one(s), as in the case of possibly mute [ə]'s¹². The Phonological Filter was precisely designed to remedy these impediments.

It basically examines all word transcriptions as a continuous stream and operates some modifications on phonemes as a function of their context (hence the term *filter*). It is also allowed to access the *spelling* and *part_of_speech* layers. It implements, in a Prolog style, rules of the form :

[a] → [b] / *left_phon_cont* _ *right_phon_context* : *condition* (9.3)

which rewrite [a] into [b] if it is surrounded by *left_phon_cont* and *right_phon_context*, and provided *condition* is verified. Conditions, as we shall see, can be used to restrict rule application to some particular word sequences.

9.2.2.1. Vowel Harmony rules.

¹¹The last four cases are considered as optional liaisons in [Delattre 51]

¹²Phonetic liaisons being a particular case, for which tests on subsequent words can be expressed in the form of graphemic conditions.

In many languages, including French, speakers tend to modify the aperture of vowels in function of their vocalic context. This even constitute a major source of concern for French pupils, desperately trying to write 'évènement', 'dussè-je', or 'cèdera', just as they hear it. These phonemic variations are termed as *vowel harmony* phenomena. They are much more numerous than one could think at first : most of the time, we are not aware of them. They are efficiently accounted in LIPSS for by the three following rules :

1. [e] → [E] / _ [C(@)CV]

in which C and V respectively stand for any consonant and vowel : 'évènement';

2. [E] → [e] / _ [C+V1]

in which V1 stands for [i], [y], or [e] : 'bêtise'

3. [E] → [e] / _ [CC+C1]

in which CC stands for a strong consonant cluster : {[v], [b], [k], [d], [f], [ʃ], [g], [p], [t]} + {[l], [R]} : 'maîtrise'

9.2.2.2. The question of the Mute [(@)].

As mentioned in Chapter 3, pronunciation is subject to wide variations, from region to region, from person to person, or even for a given speaker in different conditions. Possibly mute [(@)]'s are the best examples of such indetermination. Their pronunciation or elision does, however, respect some coherency, in the sense that they do not occur randomly in a reading.

From the many studies which have examined that question and tried to establish a general tendency (see [Dausès 73] and [Delattre 66], for instance), it appears that four cases should be distinguished : initial [(@)]'s, internal ones, final¹³ ones, and [(@)]'s in monosyllabic words :

- Internal [(@)]'s are governed by the very general rule :

'[(@)] is maintained before one or more consonants, provided it is preceded by two or more consonants'¹⁴

¹³Initial, internal, and final are understood here at accent group level, rather than at word level. Furthermore, initial and final refer to the position of [(@)] with respect to the other vowels of the group : consonants may appear before an initial [(@)] or after a final one.

¹⁴Consonants should be understood as *consonant phonemes*.

in which one notices that the number of preceding consonants is the key parameter. It is a particular case of the *three consonants rule*, which states that [(@)] is maintained if its removal produces a consonant cluster of width greater or equal to two. The general rule, however, suffers some exceptions, as in 'amènerions', 'enseveli', 'celui-ci',... We have summarized them as follows :

'[(@)] is also maintained when followed by a consonant cluster, the second consonant being either a semi-consonant ([j], [H], [W]) or a liquid ([l], [R])'

- Initial [(@)]'s are additionally often maintained in a formal way of reading, even when preceded by only one consonant ('nous venons', 'sa revue', 'un secret',...).
- Final [(@)]'s are like internal ones, except when the following word begins with an 'h' which is not dropped, in which case they are always maintained ('ce hameau').
- In monosyllabic words, [(@)] is always initial and final. It therefore conforms itself to the previous rule. The peculiarity of these words originates in the way they are handled when monosyllabic sequences are produced ('que je ne te le redise pas'). The reader immediately notices that, even though there obviously is not a single way of pronouncing them, many pronunciations cannot be accepted. Apart from the case of 'le' preceded by a hyphen, in which [(@)] is always maintained, the decision we have made in such cases is based on the simultaneous examination of the current and next words. It can be summarized by listing the sequences in which the first [(@)] is maintained :

'je' + 'ne', 'de' + 'ne', 'que' + 'de', 'que' + 'ne',
'ce' + 'que', 'ce' + 'n', 'je' + 'te', 'parce'+ 'que'

9.4. Conclusion.

The Grapheme-To-Phoneme transcriber implemented in the LIPSS system is a rule-based expert system which exhibits some interesting features, resulting from the options that have been taken from its very first day :

Readability - Reduced memory size - High transcription speed.

The resulting GTP module is a trade-off between these three constraints. Its peculiarity comes from :

- Its Prolog implementation, which we found to be a very convenient target language for expressing powerful rules in a compact way, whether as Prolog facts in a text database or as Prolog clauses compiled with the program source code.

- The intensive use it makes of syntactic information, whether for quickly accessing its exceptions lexica or for restraining the applicability of transcription rules to some part of speech categories, easily obtained by accessing the *part_of_speech* fields of words in the LIPSS multi-layer data structure.
- Its effective handling of accent group level problems, such as phonetic liaisons, possible elision of [@], and phonetic lengthening.

In the current state of development of LIPSS, however, one could hardly assess its performances with great precision. Indeed, phonetic liaisons, lengthening, and mute [@]'s obviously require some information about accent group limits, which can only be accurately given after text structuration has been performed. Nevertheless, the reader is invited to consult Appendix VI, in which informal results are presented, readily as produced by the *trace* option of LIPSS. They suffice to understand that high quality is being achieved.

Among the future features to be focused on in the LIPSS GTP module, the pronunciation of proper names takes first place. The problem is vast, as proper names only approximately obey classical transcription rules, given their various origins. A dictionary based approach is thus clearly desirable.

REFERENCES

- [Ainsworth & Pell 89] W.A. AINSWORTH, B. PELL, "Connectionist architectures for a text-to-speech system", Proc. Eurospeech 89, vol. 1, pp. 125-128.
- [Allen et al 87] J. ALLEN, S. HUNNICUT, D. KLATT, From Text To Speech, The MITTALK System, Cambridge University Press, 1987, 213 pp.
- [Aubergé 91] V. AUBERGE, La synthèse de la parole : des règles aux lexiques, Ph.D. dissertation, ICP Grenoble, 1991, 209 pp.
- [Bourciez 58] E. BOURCIEZ, Précis de phonétique française, 235 pp., Klincksieck, Paris, 1958.
- [Catach 84] N. CATACH, La phonétisation automatique du Français, éd. CNRS, Paris, 1984.
- [Chomsky & Halle 68] N. CHOMSKY, M. HALLE, The sound pattern of English, New York, Harper and Row, 1968.
- [Dausés 73] A. DAUSES, Etudes sur l'e instable dans le français familier, Max Niemeyer Verlag, Tübingen, 102 pp.
- [Delattre 51] P. DELATTRE, Principes de phonétique française à l'usage des étudiants anglo-américains, The College Store, Middlebury College, 5, Hillcrest road, Middlebury, Vermont, 1951, 68 pp.
- [Delattre 66] P. DELATTRE, Studies in French and compared phonetics, Mouton, The Hague, 1966, 286 pp.
- [Divay & Guyomard 77] M. DIVAY, M. GUYOMARD, "Grapheme-To-Phoneme transcription for French", Proc. ICASSP 77, pp. 575-578.
- [Driemmel 86] A. DRIEMMEL, Transcription phonétique d'un texte écrit en langue française en vue de sa synthèse vocale, Thesis, Faculté Polytechnique de Mons, 1986.

- [Goyer et al] P. GOYER, D. DEGRISE, B. GUERIN, "Trops, un système de transcription orthographique-phonétique et de synthèse en Français", 10èmes Journées d'Etudes sur la Parole (GALF), pp. 212-217.
- [Hochberg et al 91] J. HOCHBERG, S. MNISZEWSKI, T. CALLEJA, G. PAPCUN, "A default hierarchy for pronouncing English", IEEE Trans. on Pattern Analysis and Machine Intelligence, 13(9), september 91, pp. 957-964.
- [Klatt & Shipman 82] D. KLATT, D. SHIPMAN, "Letter to phoneme rules : a semiautomatic discovery procedure", J. Acous. Soc. Am., n° 72 (suppl 1), 1982, S.48.
- [Léon 66] P.R. LEON, Prononciation du français standard, éd. Didier, 183 pp., 1966.
- [Leroux & Miclet 79] J. LE ROUX, L. MICLET, "Transcription orthographique-phonétique et synthèse en temps réel de la parole par prédiction linéaire", 10èmes Journées d'Etudes sur la Parole (GALF), 1979, pp. 219-225.
- [Lucassen & Mercer 84] J.M. LUCASSEN, R.L. MERCER, "An information theoretic approach to the automatic determination of phoneme base forms", Proc. ICASSP 84, 52.5, pp. 42-45.
- [Matsumoto & Yamaguchi 90] T. MATSUMUTO & Y. YAMAGUCHI, "A multi-language text-to-speech system using neural networks", Proc. ESCA workshop on Speech Synthesis, Atrians, 1990, pp. 269-272.
- [McCulloch et al 87] N. MC CULLOCH, M. BEDWORTH, J. BRIDLE, "NETSPEAK - a re-implementation of NETTALK", Computer Speech and Language, 2, 1987, pp. 289-301.
- [Nyrop 63] Kr. NYROP, Manuel Phonétique du Français Parlé, éd. Gylendal, 242 pp., 1963.
- [Robert 71] P. ROBERT, Dictionnaire alphabétique et analogique de la langue française, Sté du Nouveau Littre, Paris, 1972, 1967pp.
- [Sejnowski & Rosenberg 87] T. SEJNOWSKI, C.R. ROSENBERG, Parallel networks that learn to pronounce English text", Complex Systems, n°1, 1987, pp. 145-168.
- [Torkkola 93] K. TORKKOLA, "An efficient way to learn English grapheme-to-phoneme rules automatically", Proc. ICASSP, 1993, pp. II.199-II.202.
- [Warnant 64] L. WARNANT, Dictionnaire de la Prononciation Française, Duculot, Gembloux, 1964, 414 pp.