From Text to Syllable in Castilian[1]

Harry Howard and Robert P. Goldman
Department of Spanish and Portuguese
Department of Computer Science
Tulane University, New Orleans, LA
howard@mailhost.tcs.tulane.edu and rpg@cs.tulane.edu

1. Introduction

This paper describes a Prolog program which translates Castilian orthography into a Castilian phonological representation and then syllabifies the representation according to the algorithm of Hualde (1991). It was tested on a list of more than 80,000 words, with the result that approximately 80 words could not be translated and another 120 could not be syllabified. Almost all of this residue consists either of foreign words which do not respect the orthographic or phonological conditions of Castilian or of words that were misspelled in the input word list. This experiment in implementing the system revealed shortcomings in Hualde's syllabification, algorithm which we have repaired. A second experiment on a sample randomly drawn from a Spanish dictionary shows that the system is robust.

Our system has a novel three-part architecture. It initially translates the input from ASCII codes into Castilian graphemes, then the graphemes are translated into phonemes, and finally the phonemes are parsed into syllables. The first component (the *tokenizer*) is a simple deterministic finite state machine. The second component, the *transducer*, which translates Spanish graphemes into phonemes, is a two-level transducer like those of Koskenniemi. Finally, a simple context-free parser brackets the phonemes into syllables.

This system has been implemented in Prolog. Prolog has the advantage of providing a simple execution engine capable of interpreting grammatical formalisms. This allows us to describe our syllabifier in declarative terms via two-level rules and (augmented) context-free grammars, making the system easy to understand and maintain.

The paper begins with an overview of the program and its division into components. It then treats each component of the program in turn, discussing both theory and implementation. It concludes with a discussion of the data used to test the program and some thoughts on further areas of investigation.

2. Overview of the program

[1] We would like to thank Debra Ellis Maxwell and John W. Benoit and their employer, the MITRE Corporation, for their generosity in giving us the word list. We would also like to thank Carlos Coello-Coello a graduate student in the Computer Science Department, for compiling the first version of the two-level rule set and Jennifer L. Jenkins for assistance in debugging. Partial support for Robert Goldman was provided by

which is not relevant to syllabification and so can be safely stripped off. However, the written accents on *i* and *u* do provide information which is crucial to the needs of syllabification: accented *i* and *u* cannot be read as glides. Accordingly, we represent accented *i* and *u* as distinct graphemes ('I' and 'U', respectively). This is tantamount to adding two new vowels to the surface alphabet for the two-level rules. Later we will see how the rules ensure that *I* and *U* will be translated to their correct phonemic content.

The final task of the tokenizer is the opposite of digraph formation. The transducer component is limited to translating grapheme sequences into phoneme sequences of equal lengths. This presents a problem when processing the Spanish grapheme 'x', which corresponds to the phonemic sequence /ks/. Our solution is for the tokenizer to translate each 'x' in the input into a pair of adjacent x-es, so that a word like 'extra' comes out of the tokenizer as 'exxtra'. The transducer then translates the first 'x' into /k/ and the second into /s/.

4. From graphemes to phonemes: the two-level component

In order to simplify the syllabification process and to capture linguistic generalities more perspicuously, we translate the Castilian graphemes into Castilian phonemes before syllabifying. It is the phonemes which are parsed into syllable trees. This corresponds to a generative model in which the syllable grammar first generates a string of phonemes, and the string of phonemes is then translated into a surface, graphemic, form. We utilize a two-level rule formalism based on the one developed by Koskenniemi (1983) to capture the relationship between phonemes and their realization as graphemes.

The two-level rule formalism was initially introduced as a way of specifying two-level finite state machines. The relationship between two-level rules and two-level automata was somewhat mysterious, since two-level rules are not of computational power equal to that of the automata. Previous two-level rule processors have either relied on users to translate the rules into automaton transition tables by hand or, more recently, taken rules and compiled them into two-tape finite automata automatically.[3] We have developed a Prolog meta-interpreter which interprets the two-tape rules directly. By dispensing with the intermediate automaton level, writing and debugging two-level rule sets is simplified. The two-level rule interpreter is described in more detail in Goldman (1993).

4.1. Strong correlation between phonemes and graphemes

One of the many advantages of Castilian for the investigation of syllable structure is the relatively straightforward relationship between phonemes and their written form. The standardization of Castilian orthography began in the 13th century, and Castilian pronunciation has changed little since then. This continuity provides for an extremely strong correlation between the phonemes of Castilian pronunciation and the graphemes of Castilian orthography. The strong correlation simplifies the recovery of phonemic sequences from written words and so allows for ready access to data sources with which to test a syllabification algorithm.

4.2. The two-level formalism

The two-level formalism was developed originally by Koskenniemi (1983), for the purpose of morphological analysis of natural languages. The two-level scheme allows one to represent relationships

---

[3] See Ritchie, et. al., (1992) for a discussion of a sophisticated recent implementation.

between a 'lexical' string and a 'surface' string in terms of constraints on lexical-surface symbol pairs. Koskenniemi used the two-level formalism in a form of finite-state morphology, to encode information about the realization of morphemes (encoded as sequences of letters on the lexical tape) in the written form of the language (represented on the surface tape). Morphotactic constraints were represented separately, by dividing the morphemes into separate 'continuation classes,' which were assembled into a finite-state grammar. Those are not relevant to our discussion here, and we will say no more about them.

We reserve our interest for the two-level rules themselves. In our discussion here, we adopt the notation used in a recent book describing morphological analysis of English, (Ritchie, Russell, Black & Pulman 1992). Two-level rules describe constraints on the alignment of two parallel tapes. Two-level rules are always of the form in (2a), exemplified in (2b, c):

2 a) pair operator left-context --- right-context

b) x:y => a:a --- b:b (Context-restriction rule)

c) i:y <= #:# --- #:# (Surface-coercion rule)

A context-restriction rule indicates that the symbol pair which heads the rule is permitted to appear in between the specified left- and right-contexts (i.e., in the position marked by the ---). For example, the rule in (2b) indicates that an 'x' on the lexical tape is permitted to align with a 'y' on the surface tape in any position surrounded by an 'a' aligned with an 'a' and a 'b' aligned with a 'b'. A surface-coercion rule such as (2c) indicates that the lexical symbol of the head pair *must* be realized as indicated in the pair in the specified context. (2c) captures the exceptional case of the /i/ phoneme being written as 'y' in the word *y* . If the lexical symbol (for us, the phoneme) 'i' appears between the two pairs #:# and #:# (the word-boundary markers), then it must be realized as 'y'.

A simple piece of syntactic sugar is to add composite rules, written using the operator <=>, which represent a pair of a context-restriction rule and a surface-coercion rule with the same head, right and left contexts. We follow Ritchie, et. al. in using a connective 'or' to allow rules with multiple left and right contexts. We also allow rules to have contexts specified in terms of sets of characters (whose names we give in italics) and the set-difference operator. The special set '=' indicates the set of all lexical or surface characters, whichever is appropriate. For example, the rule in (3) indicates that the phoneme /i/ may be realized as the grapheme 'i', either when surrounded by non-vowel phonemes (phonemes which are not in the set of vowels and whose graphemic realization is not constrained) or when preceded or followed by the /w/ glide phoneme (again, whose realization is not constrained by this rule).

3. i:i => not(v):= --- not(v):= or w:= --- =:= or =:= --- w:=.

5.4 A two-level implementation of Castilian orthography

This section reviews the two-level rule set we have developed for translating Spanish orthography into Castilian phonology. It begins with the vocoids and the sundry variations which they participate in, and then takes up the consonants. Due to limitations of space, the rules are stated with little discussion of their motivation. It is hoped that the reader is familiar enough with the conventions of Castilian orthography to recover them for him- or herself.

5.4.1. General comments on the vowel mappings

Given the fact mentioned in the previous section that the accented forms of 'a, e, o' do not provide information that is relevant for syllabification and so can be ignored, the two-level format automatically constructs three unrestricted pairs:

4. a:a, e:e, o:o

These are often referred to as "strong vowels", since they do not generally alternate with glides.

The pairings between the high vowel graphemes and their phonemes are much less transparent, because the "weak vowels" 'i, u' can be understood as the vocalic nuclei /i, u/ or the glides /j, w/, respectively. The rule of thumb is for 'i, u' to be interpreted as vowels unless they are adjacent to a vowel and unaccented, in which case they are interpreted as glides.[4] The two-level format captures this generalization by statements which describe exceptional cases, so that the general case results if none of the specific ones apply.

5.4.2. The front high vocoids *i, j, y*

The general mapping between 'i' and /i/ is as in (5a):

5. i:i =>

a)	not(v):= --- not(v):=	cinco	/θinko/
b)	or w:= --- =:=	lingüista, cuidado	/lingwista, kwidado/
c)	or =:= --- w:= .	(ciudad)	?/θiwdad/
d)	default pairs are (i:f).		

There are no incontrovertible instances of the off-glide or falling diphthong predicted by (5c), since 'iu' sequences can be interpreted both as /iw/ and the preferred /ju/, as in /θjudad/ for 'ciudad'.[5] For our concerns, it suffices to say that the rule set licenses both possibilities. The condition in (5d) states that 'i' is understood as /i/ is when it bears the orthographic accent 'f', which is encoded by adding "i:f" to the set of unrestricted pairs - that is, the set of pairs of the form ∂:∂ which lack an explicit restriction. This set will be extended as the discussion unfolds.

The conditions under which 'i' is understood as the glide /j/ are all others, except that it cannot be adjacent to its vocalic counterpart /i/.[6] The two-level rule which expresses this is (6):

---

[4] We refer to this generalization as a "rule of thumb" because there are many exceptions, such as 'cliente' /kli.en.te/, in which the high vowel remains a monopthong despite its contiguity to a strong vowel. See Hualde for discussion. These must be listed, a task which was not attempted in this initial version of the system.

[5] See among others, Bowen & Stockwell (1960:42) and Navarro Tomás (1977). The reader is referred to Carreira (1991:436) for some speculation on the cause of this ambiguity.

[6] There a handful of words which permit double 'i' sequences, e.g. 'friísimo' /fri.is.i.mo/. The second 'i' is locked into its /i/ correlate by the orthographic accent, while the first 'i' is prevented from translating to /i/ by its adjacency to the vowel 'f' rather than the glide /w/. It cannot be denuclearized because it would wind up next to its nuclear counterpart. The only solution which occurs to us is that the morpheme boundary between *fri-* and

6. j:i =>

a) not(i):= --- v – i:=[7]

b) or v – i:= --- not(i):= .

"Not(i)" stands for the set of vowel phonemes which is not /i, j/, while "v – i" stands for the set of vowel graphemes which is not 'i'. This is a general constraint on dipthongs which reappears in the discussion of 'u'.

There are also two special cases. One is the realization of the /j/ glide as word-final 'y', as in *buey* /bwej/ and a few other words:

7. j:y <=> v – i := --- #:#

Again, /j/ cannot occur next to its vocalic counterpart, which is the reason for the restriction in the left context.[8] The other special graphotactic process is that the conjunction /i/ is spelled with a 'y'. This exception is easily stated in the two-level format of (8), given that 'y' is an-independent word:

8. i as y rule: i:y <=> #:# --- #:#.

These rules now translate all of the relevant segments in the word list, with the exceptions mentioned.

5.4.3. The back high vocoids *u, w, ü*

The high back vocoids receive much the same treatment as the high front vocoids in Spanish orthography. The default correlation of vocalic *u:u* is blocked just in case 'u' is adjacent to a vowel and unaccented; in which case it is paired with the high front glide, *w:u* . In other words, the default correlation is "adjacent to a non-vowel", i.e. between non-vowels, as (9a), or after a /j/ glide, (9b), or before one, (9c):

-ísimo acts as a 'non-vowel' to rule in /i/. Unfortunately, questions of morphology go beyond the goals of this report, so 'frísimo' can only be treated as an exception here.

[7] This rule does not license 'ion' /i.on/ and 'iota' /i.o.ta/. Their initial 'i' does not correlate with /i/ because it is adjacent to the vowel /o/, and it does not correlate with /j/ because its left context # is not a 'non-i' element. If the definition of 'non-i' element were broadened to include #, the initial 'i' would be translated as the glide, incorrectly. Nevertheless, this outcome captures the historical evolution of Castilian, as seen in the conversion of 'i' into /y/ in many similar words, e.g. 'yodo'. Thus 'ion' and 'iota' can be treated as cases of exceptional blocking of the denulcearization of 'i', a process which was mentioned above.

[8] The statement of the left context in terms of 'non-i' vowels has the effect of blocking a translation for two members of the word list: 'my' and 'ny'. Neither are native Castilian words, so their rejection by the two-level rules would appear to be correct. Their presumed representation as /mi/ and /ni/, respectively, would have to be accounted for by an exception rule which is not pursued here. Note also that the usage of the biconditional means that /j/ in word-final position can only correlate to 'y', not 'i'. Taken in conjunction with the fact that the i rule in (5) does not permit the correlation of 'i' to /i/ before a vowel, (7) effectively prevents the two-level rules from translating five words which end in "vowel + i" sequences: 'agnusdei, cai, chai, paipai, samurai'. None of these are native to Castilian either, so it seems preferible to leave the rules as they stand rather than countenance the overgeneration which attempting to rule in these exceptions would create.

9. u:u =>

a) not(v):= --- not(v):= ángulo /angulo/

b) or j:= --- =:= triunfo /trjunfo/

c) or =:= --- j:= . (cuidado, muy) ?/kujdado, muj/)

As before, there are no incontrovertible instances of the off-glide predicted by (9c), since 'ui' and 'uy' sequences can be interpreted both as /uj/ in (9c) and as the preferred on-glide /wi/, e.g. /kwidado/. The reader is once again referred to Carreira's article for some ideas on the reasons for this state of affairs. The fourth context which requires the realization of vocalic /u/ is that of the accented 'ú'. In contrast to 'î', the *u:ú* pairing must be given with a restriction, due to the peculiar phonotactics of 'g' mentioned below:

10. u:ú => {non g}:= --- =:=. algún /algun/

The glide pairing results in all other cases, as long as it does not wind up next to its vocalic counterpart /u/:

11. w:u =>

a) not(u):= --- v – u:=

b) or v – u:= --- not(u):= .

As with j:i, the restriction of both contexts to 'non-u' vowels prevents the gliding rule from applying to word-initial and word-final 'u', of which the word list offers the examples 'ueste, bou, cauchau, guau, jau, llaullau, marramau, miau, mildeu, tau'. The standard spelling of 'ueste' is 'oeste', and though many of the others are adopted foreign words, there is no convention of using 'w' for /w/ word-finally in order to block gliding. The result is that the rules above translate every member of the word list to its correct phonemic representation.

5.4.4. The easy consonantal mappings

Thirteen of the consonantal pairings of Castilian involve unrestricted matching of the same symbol to itself and so can be generated automatically by the two-level system in (12a); the three others in (12b) are likewise unrestricted, but they do not involve matching a symbol to itself:[9]

12 a) b:b, ch:ch, d:d, f:f, l:l, m:m, n:n ,ñ:ñ, p:p, r:r, rr:rr, s:s, t:t

b) default pairs are {i:î, b:v, x:j, Ø:h}

The grapheme 'h' can for all practical purposes be considered silent in Castilian, so that a word spelled 'heno' is understood as /eno/. Nevertheless, it is retained in the standard orthography mainly as a kind of fossil of the etymological 'h' of Latin, e.g. *hora* < Lat. *hora* or the intermediate stage in the weakening process of Old Spanish /f/ to /h/ and then to silence, e.g. *hoja* < Lat. *folia* . A handful of words derived from Arabic do have an 'h' pronounced as /x/, e.g. 'haca', but they all have doublets with 'j', e.g. 'jaca', which indicate the need to pronounce the consonant. In view of the massive redundancy which would be introduced in allowing for the few cases of non-silent 'h', it is preferable to use the unrestricted default pairing in (12b), Ø:h, and treat the

---

[9] See Cardona (1987) and Martínez de Sousa (1991) for much fuller discussion.

exceptions as such.[10] There are several graphotactic constraints on these two sets of consonantal correlations which are made in the program but which we do not have space to include here. The other consonantal correlations are subject to a certain amount of contextual variation, so the full power of the two-level system must be invoked. The following subsection discusses the principal case.

5.4.5. Velar softening and silent 'u'

One of the few phonological changes which Castilian underwent after the standardization of the orthography was the fricativization of the velar stops /k, g/ under the influence of a following front vocoid, /Y, i, e/. In this context, /k/ became the voiceless interdental fricative /θ/, while /g/ became the voiceless velar fricative /x/. The rules we posit for this process are given in (13-18):

13. soft_g rule

    x:g => =:= --- fr_V:fr_V.

14. g:g => =:= --- not(fr_vowel):=.

15. silent_u rule

    Ø:u => velar:velar_let --- fr_V:fr_V.

16. soft_c rule

    q:c => =:= --- fr_V:fr_V.

17. hard_c rule

    k:c => =:= --- not(fr_V):not(fr_V).

18. k_as_qu rule

    (k:q => =:= --- Ø:u).

We apologize again for not being able to enter into a fuller discussion.

5. From phonemes to syllables: the syllabifier

An advantage of Castilian for syllabification studies is the availability of a considerable amount of detailed investigation and analysis in works such as Saporta & Contreras (1962), Hooper (1976:§10-12), Harris (1983, 1989a, 1989b), Holt (1984), Manas (1987), D'Introno, Ortiz & Sosa (1989), Hualde (1989, 1991), Roca (1991), and Carreira (1991). In particular, Hualde (1991) provides a convenient summary of previous results, in the form of a six-part algorithm which lends itself naturally to a more rigorous formalization as a context-free grammar, which can be interpreted as a parser by Prolog.

5.3. A syllable grammar

5.3.1. Preliminary definitions

---

[10] An anonymous reviewer objects that 'h' deletion collapses the distinction in syllabification between 'ahumo' /a.u.mo/ and 'ahumado' /aw.ma.do/, cf. 'aula' /aw.la/, and suggests that some allowance must be made for 'h' to mark stress assignment. The problem with this proposal are counterexamples to the 'ahumado' type which do not require dipthongization across 'h': 'ahijado' [a.i.xa.do], 'ahidalgar' [a.i.dal.gar], 'prohibir' [pro.i.bir].

To get the system going, a word must be defined as a sequence of one or more syllables, and the phonemic segments on which these syllables are based must be defined. Hualde takes these definitions for granted, but Prolog is not quite as accomodating. The definition of the segments entails nothing more complicated than defining each letter as a vowel, glide or consonant.[11] A first approximation to the recursive division of words into syllables is given by the Prolog program in (19):

(19) a) word([S]) --> syllable(S).

b) word([S1|Rest]) --> syllable(S1), word(Rest).

These rules correspond to the two grammar rules *word → syllable* and *word → syllable, word*. The arguments to the non-terminals are used by Prolog to assemble a parse tree as a side effect of the parsing (or generation) process. The square brackets indicate a list, so that the second rule in (19) states that a word will correspond to a list of parse trees, the first element of which is a syllable parse tree and the rest of which is recursively defined.

5.3.2. Onset projection

The initial step in Hualde's algorithm is **node projection**, which marks vowels as the heads of syllables by projecting N, N' and N" nodes from them. This step encodes directly the fundamental linguistic fact that all syllables in Spanish contain a vocalic head, but it does so in a way that is counterintuitive for the null hypothesis of left-to-right parsing: it skips over all of the initial material until it finds the vowel, and then backtracks to the beginning to incorporate the rest. Our alternative is more top-down. Since we present a grammar rather than an algorithm, we are not committed to any particular sequence of parsing operations. Thus, we are free to state in (20a) that the first segment of a syllable be an onset, followed by a rime:

(20) a) syllable(s(O,Rime)) --> onset(O),rime(Rime).

An onset can be a consonant, or null:

b) onset(o(C)) --> consonant(C).

c) onset(o(epsilon)) --> [].

(20) produces the illustrative bracketings in (21a-c), or the tree in (21d):

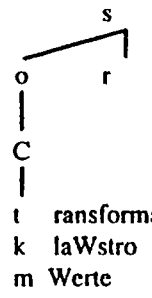
(21) a) transformar: s(o(t, r(...ransformar

b) klaWstro: s(o(k, r(...laWstro

c) mWerte: s(o(m, r(...Werte

---

[11] Given limitations of space, we cannot include our implementation of the phonemic inventory of Castilian here, but it is the standard five-vowel, two-glide, and twenty-consonant system, using IPA notation wherever typographically possible.



d)

5.3.3. Complex onset creation

Having formed the onset, we can go straight to incorporating any following consonant into it, via (22):

22. onset(o(cluster(I,L))) --> initiator(I),liquid(L),{l_constraint(I,L)}.

The curly brackets indicate that l_constraint is not a constituent, but rather names a constraint which must be satisfied. The "initiator" is a stop or /f/:

23 a) initiator(C) --> stop(C).

b) initiator(f) --> [f]. (The square brackets indicate a terminal.)

The *l_constraint* rules out the forbidden onset clusters *dl* and *tl* . It is captured by the Prolog predicate in (24):

24 a) l_constraint(d,l) :- !,fail.

b) l_constraint(t,l) :- !,fail.

c) l_constraint(_,_).

Note that the cut in this context is benign - it is just easier to list the two exceptions instead of the twelve acceptable clusters. These statements produce the bracketings and diagram in (25):

25 a) transformar: s(o(cluster(t,r), r(...ansformar

b) klaWstro: s(o(cluster(k,l), r(...aWstro

c) mWerte: s(o(m, r(...Werte

d)

There can be but two consonants per onset, so the next step is to create the rime.

5.3.4. Rime formation

The rime was introduced in Onset formation without any internal structure. In Castilian, it can consist of a nucleus with an optional coda:

26 a) rime(r(N,K)) --> n(N),c(K).

b) rime(r(N)) --> n(N).

The nucleus itself consists of a vowel with an optional on-glide:

27 c) nucleus(n(G,V)) --> glide(G),vowel(V).

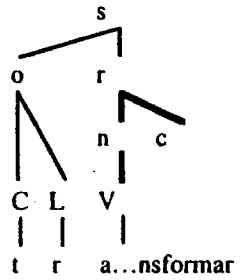
d) nucleus(n(V)) --> vowel(V).

These statements produce the bracketings and diagram in (28):

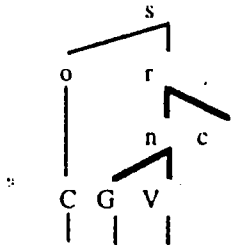
28 a) transformar: s(o(cluster(t,r), r(n(a),c(...nsformar

b) klaWstro: s(o(cluster(k,l), r(n(a),c(...Wstro

c) mWerte: s(o(m, r(n(W,e),c(...rte



d)



e)

The nucleus can contain no more material, so we turn to the creation of the coda.

5.3.5. Coda formation

A coda consists of a single consonant, or glide, or both, or neither:[12]

29 a) coda(c(C)) --> consonant(C).

b) coda(c(G)) --> glide(G).

c) coda(c(G,C)) --> glide(G),consonant(C).

d) This case is captured by rule (26b).

The statements in (29) produce the bracketings and diagrams in (30):

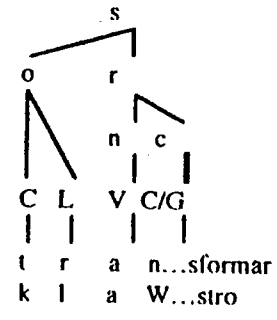
30 a) transformar: s(o(cluster(t,r), r(n(a),c(n)...sformar

b) klaWstro: s(o(cluster(k,l), r(n(a),c(W)...stro

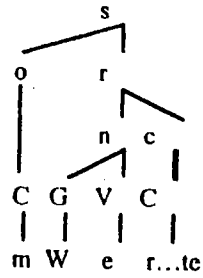
c) mWerte: s(o(m, r(n(W,e),c(r)...te

---

[12] An anonymous reviewer objects that this formalization is incorrect for word-final position in two ways. One is that it overgenerates word-final codas, such as *'casall' and *'carrañ'. We agree and intend to fix this, but this problem does not diminish the success of the program with real-life Castilian text. The other objection is that common borrowings like 'parking' and 'film' cannot be syllabified. We agree and note that such nativized words do diminish the success of the program with real-life Castilian text, though they were not found in our word list. Fortunately, the two-level rules can be augmented to correlate the borrowed orthographies to their nativized phonology.



d)



e)

The coda of *mWerte* is full, which ends the syllabification of the first syllable, but *transformar* and *klaWstro* can undergo a further step.[13]

5.3.6. s-Adjunction

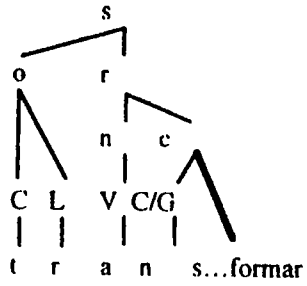
/s/ as the only consonant to follow a consonant or glide within a coda, a fact encoded in (31) and illustrated in (32):

31 a) coda(c(C,s)) --> consonant(C),[s].

b) coda(c(G,s)) --> glide(G),[s].

32 a) transformar: s(o(cluster(t,r), r(n(a),c(n,s)))...formar

b) klaWstro: s(o(cluster(k,l), r(n(a),c(W,s)))...tro



c)

5.3.7. Onset maximalization

The final step is to join one syllable to the next. This can not be done in just any fashion, since Spanish requires that material which could either be syllabified into the current coda or the following onset go as much as possible into the onset. Such a requirement to maximalize onsets is responsible for the common (CV)* syllable structure of Spanish exemplified in (33):

---

[13] It should be noted that any tautosyllabic sequence of "liquid or nasal + C" in which C is not /s/ will fail Hualde's algorithm, such as in 'esprint', 'coulomb', 'golf', 'volt', or 'árctico' /ark.ti.ko/. The reason has to do with the syllabification of C. It cannot be put into the second syllable by Onset Maximization, since "C + resonant" is not a valid onset cluster. On the other hand, C cannot terminate the coda, because the only consonant which can follow a coda consonant is /s/, via /s/ Adjunction. Lacking either derivation the syllabification crashes. Of course, most of these examples are non-native words which undergo epenthesis in Castilian. The exact nature of this epenthesis process is the subject of ongoing research.

a) be.be, pa.ta.ta, Se.vi.lla

b) *beb.e, *pat.at.a, *Sev.ill.a

This requirement is best stated by elaborating the recursivity of syllabification set forth in (19b) to include an onset maximalization predicate, as in (34):

a) word([S]) --> syllable(S).

b) word([S1,S2|Rest]) --> syllable(S1), word([S2|Rest]), {onset_max(S1,S2)}.

At our present level of understanding, we can do little more than stipulate the exclusion of the two cases which onset maximalization covers in Spanish.

The first case to be excluded involves an empty onset following a syllable whose coda is a consonant:

a) onset_max(Syllable, Follower) :-
  coda_consonant(Syllable, _Cons),
  onset_of(Follower, o(epsilon)),
  !, fail.

b) onset_max(_,_).

c) coda_consonant(s(_,r(_,c(C))), C) :- icons(C).

c') coda_consonant(s(_,r(_,c(_,C))), C) :- icons(C).

This program rules out the starred examples in (33b), as instances of the bracketings in (36):

a) *beb.e: s(o(b),r(n(e),c(b))), s(o(epsilon)...e

b) be.be: s(o(b),r(n(e),c(epsilon))), s(o(b),r(...e

Prolog will backtrack to generate an empty coda and so place the second *b* in the second onset, as in (36b).

The second case to be excluded concerns a coda containing a consonant which could be followed by a liquid preceding an onset with a liquid, illustrated in (37a):

a) *ab.re, *cop.la, *af.ri.ca, *mic.ro.bio

b) a.bre, co.pla, a.fri.ca, mi.cro.bio

(37b) gives the correct syllabification, in which the coda consonant must be moved into the following onset. The program in (38) ensures that an (37a)-type syllabification does not appear:[14]

onset_max(Syllable, Follower) :-
  coda_consonant(Syllable, Cons),
  initiator(Cons),
  onset_of(Follower, o(L)),
  iliq(L),

---

[14] An anonymous reviewer points out that onset maximalization fails systematically for words like 'subrayado', which is phonologically /sub.ra.ya.do/ and not */su.bra.ya.do/, cf. 'sublime' /su.bli.me/. There is no way for our system to express such morphological exceptions, though we intend to take this up in the next version.

!, fail.

An illustration of the excluded and included bracketings is (39):

39 a) *ab.re: s(o(epsilon),r(n(a),c(b))), s(o(r), r(...e

b) a.bre: s(o(epsilon),r(n(a),c(epsilon))), s(o(b),r(...e

We have implemented the onset maximization constraint in terms of exclusion/negation, using the meta-logical "cut" (!) operator of Prolog. In general, the cut is to be avoided as corrupting the declarative reading of a Prolog program. However, in this context the cut is not harmful, as it simply provides a shorthand alternative to enumerating the various possible syllable-syllable sequences. Since there are only a finite number of syllable trees, the onset_max predicates above could be translated into a somewhat bulkier representation in terms of allowable sequences of tree shapes. This translation could be done automatically by partial evaluation if so desired. This completes the major points of the grammatical translation of Hualde's algorithm.

6. Testing

The discussion has already touched on the testing procedure several times, as it was an integral part of the refinement of the program. This section discusses the two tests which were performed more formally.

6.1. Testing the syllabification model against a word list

The entire 86,016-item word list was fed through the program many times, giving a list of syllabic bracketings, plus a list of untranslated and unsyllabified items. The errors which forced the translation and syllabification components to fail were in general obvious from visual inspection. As mentioned in the Introduction, the final residue amounted to about 200 words which do not conform to the graphotactic or phonotactic requirements of Castilian because of their non-native origins. Among this residue there were also found a score of words which were misspelled. There are also a few instances of words which the program should analyze successfully but does not, such as the realization of word-final /w/ as 'u'. These cases will be incorporated in future work.

6.2. Testing the syllabification output against a dictionary

A much more daunting task is the verification of the correctness of some 80,000 syllabified phonemic representations. We know of no way of doing this automatically, since we know of no machine-readable source of syllabified phonemic representations against which to check our own. The only way is to check our output against written sources, such as the *University of Chicago Bilingual Spanish Dictionary* . In a first attempt to do so, we took a random sample of 300 of the successfully syllabified words, to compare them against the dictionary syllabifications. Disappointingly, only 58 of the chosen 300 words appeared in the dictionary! In retrospect, this is not surprisingly, since a large number of the 300 words were derived or inflected forms and not represented in the dictionary. However, our system did syllabify all 58 words in agreement with the dictionary.

In order to bolster our confidence in the system, we randomly chose 100 words from the dictionary and used our system to syllabify them. Of this sample, ninety-eight were correct, and one, 'constituido', contained a 'ui' sequence for which the program reported the /uj/ alternative rather than the preferred /wi/ representation. However, the system would accept the /wi/; it just happened to find it second. Since the resolution of unstressed

ak-vowel sequences is a poorly understood area, this result does not seem to reflect poorly on the program. We will investigate this further in future work.

The one word that failed the test, 'cacahuate', did so because the deletion of the 'h' made the 'u' contiguous to 'a', so it would be correlated to the /w/ glide and syllabify with the preceding 'a' rather than the following one. That is, the outcome was /ka.kaw.te/, rather than the correct /ka.ka.wa.te/. The existence of such failures was to be expected, since we do not understand as yet the conditions under which 'h' is used to break up syllables. This too must be left for future research.[15]

The tests reported in this subsection were done after the system was completed, and no changes were made in the system in the course of this test. It is encouraging to find a 98-99% accuracy rate in this test, especially since the few errors are localized in a clearly-defined, if poorly-understood, domain.

7. Conclusions

The conclusions, apart from the apparent success of the program at doing what it was designed to do as well as the need for more detailed analysis in certain areas, are fairly general. The usefulness of Prolog for natural-language processing is further vindicated, especially in view of the ease with which it can be used to build a meta-interpreter which avoids the intermediate step of the two-level automata. Prolog also allowed us to describe the syllabification process declaratively, rather than procedurally, so that the model as a whole may be used for generation as well as recognition. Our ability to experiment with a large word list allowed us to discover and repair shortcomings of earlier work.

The computational architecture presented in this paper opens up several lines of further research, which can be described in terms of breadth and depth. In view of the successful processing of Castilian text, a natural next step would be to modify the rule set to cover other dialects of Spanish and other Romance languages. Catalan, Portuguese and Italian are all appealing candidates, because of their relatively simple orthography. French is a much more challenging case, and we do not expect to take it up in the near future.

As far as depth, the automatic generation of syllabified phonemic representations for Castilian opens the door to further phonological processing. The first step would be to generate the allophonic and stress assignment variants of the phonemic sequences. The next step would be to group phonological words into phonological phrases and state the allophonic processes that apply across phonemic words. A slightly different line of inquiry is to break the phonological words into their morphemic constituents. This is an especially exciting task, given the observation made above that morphological junctures affect syllable division in certain words. A logical application of all this work would be to use the resulting phonological representations to drive speech synthesis.

8. References

Bowen, J. D. & R. Stockwell. 1960. *Patterns of Spanish Pronunciation*. Chicago, Illinois: University of Chicago Press.

---

[15] An anonymous reviewer points out that the correct Castilian word is 'cacahuete' /ka.ka.we.te/, which would still fail the test.

Cardona, Angeles. 1987. *A la ortografía por la gramática*. Barcelona: Promociones y Publicaciones Universitarias.

Carreira, Maria. 1991. The alternating dipthongs in Spanish. In *Current Studies in Spanish Linguistics*, ed. Héctor Campos & Fernando Martínez-Gil, 407-445. Washington, D.C.: Georgetown University Press.

D'Introno, Francesco, Judith Ortiz & Juan Sosa. 1989. On Resyllabification in Spanish. In *Studies in Romance Linguistics*, ed. Carl Kirschner & Janet DeCesaris, 97-114. Amsterdam: John Benjamins.

Goldman, Robert. 1993. *A Logic-programming interpreter for Two-level Morphology*. Tulane Computer Science Department Technical Report TUTR 93-106. New Orleans: Tulane University.

Harris, James. 1983. *Syllable Structure and Stress in Spanish: A Nonlinear Analysis*. Cambridge, Mass.: MIT Press.

Harris, James. 1989a. Our Present Understanding of Spanish Syllable Structure. In *American Spanish Pronunciation: Theoretical and Applied Perspectives*, ed. Peter Bjarkman & Robert Hammond, 151-169. Washington, DC: Georgetown University Press.

Harris, James. 1989b. Sonority and Syllabification in Spanish. In *Studies in Romance Linguistics*, ed. Carl Kirschner & Janet DeCesaris, 129-153. Amsterdam: John Benjamins.

Holt, Katherine. 1984. An autosegemental approach to syllabification in Spanish. In *Papers from the XIIth Linguistic Symposium on Romance Languages*, ed. Philip Baldi, 169-193. Amsterdam: John Benjamins.

Hooper, Joan. 1976. *An Introduction to Natural Generative Phonology*. New York, NY: Academic Press.

Hualde, José Ignacio. 1989. Silabeo y estructura morfémica en español. *Hispania* 72:821-831.

Hualde, José Ignacio. 1991. On Spanish Syllabification. In *Current Studies in Spanish Linguistics*, ed. Héctor Campos & Fernando Martínez-Gil, 475-494. Washington, D.C.: Georgetown University Press.

Kaye, Jonathan. 1989. *Phonology: A Cognitive View*. Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Mañas, José. 1987. *Word Division in Spanish*. Communications of the ACM 30:612-616.

Martínez de Sousa, José. 1991. *Reforma de la ortografía española*. Madrid: Visor Libros.

Navarro Tomás, T. 1977. *Manual de pronunciación española*. Madrid: Consejo Superior de Investigación Científica.

Ritchie, Graeme, Graham Russell, Alan Black & Stephen Pulman. 1992. *Computational morphology: Practical mechanisms for the English lexicon*. Cambridge, Massachusetts: The MIT Press.

Roca, Iggy. 1991. Stress and syllables in Spanish. In *Current Studies in Spanish Linguistics*, ed. Héctor Campos & Fernando Martínez-Gil, 599-635. Washington, D.C.: Georgetown University Press.

Saporta, Sol & Heles Contreras. 1962. *A Phonological Grammar of Spanish*. Seattle, Washington: University of Washington Press.