GRAPHEME-TO-PHONE USING FINITE-STATE TRANSDUCERS

D. Caseiro, I. Trancoso, L. Oliveira

INESC-ID/IST
Rua Alves Redol 9, 1000-029 Lisbon, Portugal

C. Viana

CLUL
Av. Prof. Gama Pinto 2, Lisbon, Portugal

ABSTRACT

Several approaches have been adopted over the years for grapheme-to-phone conversion for European Portuguese: hand-derived rules, neural networks, classification and regression trees, etc. This paper describes different approaches implemented as Weighted Finite State Transducers (*WFSTs*), motivated by their flexibility in integrating multiples sources of information and other interesting properties such as inversion. We describe and compare rule-based, data-driven and hybrid approaches. Best results were obtained with the rule-based approach, but one should take into account the fact that the data-driven one was trained with automatically transcribed material.

1. INTRODUCTION

This paper describes the development of a grapheme-to-phone conversion module based on *WFSTs* for European Portuguese. We investigated both the use of knowledge-based and data-driven approaches.

The objective of a grapheme-to-phone module implemented as *WFSTs* is justified by their flexibility in the efficient and elegant integration of multiple sources of information, such as the information provided by other "text-analysis" modules. The flexibility of *WFSTs* also allows the easy integration of knowledge-based with data-driven methods.

Our first approach to grapheme-to-phone (GtoP) conversion for European Portuguese was a rule-based system (DIXI), with about 200 rules[1]. All the code was programed in C, directly in the case of the stress assignment rules, and using the SCYLA ("Speech Compiler for Your Language") [2] rule compiler, developed by CSELT, for the remaining rules. The multi-level structure of this compiler allowed each procedure to simultaneously access the data resulting from all the previous procedures, so the rules could simultaneously refer to several levels (such as the grapheme level, phone level, sandhi level, etc.)

Later, this rule-based approach was compared with a neural net approach [3] that, in spite of the fairly good results, was never integrated in our synthesizer. Instead, we integrated an approach based on *CARTs* (Classification and Regression Trees) combined with a large coverage lexicon, as part of the porting of our TTS system (now designated as DIXI+) [4] to the Festival framework [5].

The present work is part of Diamantino Caseiro's PhD thesis, initially sponsored by a FCT scholarship (PRAXIS XXI/BD/15836/98). It has been conducted within the DIXI+ project, supported by the Portuguese Foundation for Science and Technology (FCT). INESC-ID Lisboa had support from the POSI program of the "Quadro Comunitario de Apoio III".

Some of the most common approaches to grapheme-to-phone convertion can be compiled to *WFSTs*, among which are *CARTs* [6], and most rule systems, such as two-level [7] and rewriting rules [8].

In this work, we first show how we compiled the rules of the DIXI system to *WFSTs* (Section 2), we then present data-driven approaches to the problem (Section 3), and finally we combine the knowledge-based with the data-driven approaches (Section 4).

In order to assess the performance of the different methods, we used a pronunciation lexicon built on the PF ("Português Fundamental") corpus. The lexicon contains around 26000 forms. 25% of the corpus was randomly selected for evaluation. The remaining portion of the corpus was used for training or debugging.

The size of the training material for the data-driven approaches was increased with a subset of the BD-Público [9] text corpus. This corpus includes a collection of texts from the on-line edition of the Público newspaper. We used all the words occurring in the first 1,000,000 paragraphs of this corpus, and obtained their transcription by rule using DIXI. The 205k words not in PF were added to the training set.

2. KNOWLEDGE-BASED SYSTEM

Our first goal was to convert DIXI's rules to a set of *WFSTs*. SCYLA rules are of the usual form $\phi \rightarrow \psi / \lambda \_\_ \rho$ where ϕ, ψ, λ and ρ can be regular expressions that refer to one or multiple levels. The meaning of the rules is that when ϕ is found in the context with λ on the left and ρ on the right, ψ will be applied, replacing it or filling a different level of ψ.

The application of standard generative rewriting rules [10] to a sequence of graphemes, poses some well known problems, as it leads to unnecessary rule dependencies due to the replacement of graphemes by phones: the first rule has only graphemes on its context, while the last ones have mainly phones. That happens to a small-extend in [11], for example.

In DIXI's case, some of these problems may be avoided, as most of the grapheme-to-phone rules were written such that ϕ, λ and ρ only refer to the grapheme level (with stress marks already placed on it) and ψ only to the phone level, represented in a different tier of the multi-level data-structure. There are no intermediate stages of representation and no rule creates or destroys the necessary context for the application of another rule. In order to prevent some common errors, a small set of 6 rules was neverthless added which refer grapheme-phone correspondances on either context λ or ρ. Note, however, that although some similarities may be found between DIXI's and a Two-Level Phonology approach ([12], [13]), DIXI's rules are not two-level rules: contexts are not fully specified as strings of two-level correspondances and within the set of rules for each grapheme, a specific order of application is required.

Default rules need to be the last and in some cases in which the contexts of different rules overlapp partially, the most specific rule needs to be applied first.

In order to preserve the semantic of DIXI's rules we opted to use rewriting rules, but in the following way:

First, the grapheme sequence $g_1, g_2, ..., g_n$, is transduced into $g_1, \_, g_2, \_, ..., \_, g_n$, where _ is an *empty* symbol, used as a placeholder for phones. Each rule will replace _ with the phone corresponding to the previous grapheme, keeping it. The context of the rules can now freely refer to the graphemes. The few DIXI rules whose context referred to phones can also be straightforwardly implemented. The very last rule removes all graphemes, leaving a sequence of phones. The input and output language of the rule transducers is thus a subset of $(grapheme\ phone)^*$. The set of graphemes and the set of phones do not overlap.

2.1. Rule specification language

The rules are specified using a rule specification language, whose syntax resembles the BNF (Backus Naur Form) notation, allowing the definition of non-terminal symbols (e.g. `$Vowel`). Regular expressions are also allowed in the definition of non-terminals. Transductions can be specified by using the *simple transduction* operator $a \rightarrow b$, where a and b are terminal symbols. This work motivated us to extend the language with two commands.

The first one is:

`OB_RULE` $n,\ \phi \rightarrow \psi / \lambda \_\_ \rho$

where n is the rule name and $\phi, \psi, \lambda, \rho$ are regular expressions. `OB_RULE` specifies a context dependent *obligatory rule*, and is compiled using Mohri and Sproat's algorithm[14].

The second command is:

`CD_TRANS` $n,\ \tau \Rightarrow \lambda \_\_ \rho$

where τ is a transducer (an expression that might include the $\rightarrow$ operator). `CD_TRANS` (Context-Dependent Transduction) is a generalization where the replacing expression depends on what was matched. It is compiled using a variation of Mohri and Sproat's algorithm, that uses $\pi_1(\tau)$ instead of ϕ, and τ instead of the cross product $\phi \times \psi$. Its main advantage is that it can succinctly represent a set of rules that apply to the same context. We use it mainly in the stress-marking phase of the grapheme-to-phone conversion.

2.2. Grapheme-to-Phone Phases

The rules of the grapheme-to-phone system are organized in various phases, each represented by transducers that can be composed to build the full system. Figure 1 shows how the various phases are composed.

Each phase has the following function:

- the `stress` phase consists of 27 rules that mark the stressed vowel of the word.
- `introduce-phones` is the simple rule that inserts the _ *empty phone* placeholder after each grapheme. (`$Letter` (`NULL` $\rightarrow$ `EMPTY`)) $\Rightarrow$ `__`).
- `prefix-lexicon` consists of pronunciation rules for compound words, namely with roots of Greek or Latin origin such as "tele" or "aero". It includes 92 rules.
- `gr2ph` is the bulk of the system, and consists of 340 rules, that convert the 45 graphemes (including graphically stressed versions of vowels) to phones.
- `sandhi` implements word co-articulation rules across word boundaries. (This rule set was not tested here, given the fact that the test set consists of isolated words.)
- `remove-graphemes` removes the graphemes in order to produce a sequence of phones.
  (`$Letter` $\rightarrow$ `NULL` / `__`).

```
      stress       o
introduce-phones   o
 prefix-lexicon    o
      gr2ph        o
      sandhi       o
remove-graphemes
```

Fig. 1. Phases of the knowledge based system.

The following example illustrates the specification of 2 `gr2ph` rules for deriving the pronunciation of grapheme g: either as /Z/ (e.g. *agenda, gisela*) when followed either by e or i, or as /g/ otherwise (SAMPA symbols used).

```
OB_RULE 0200, g EMPTY -> g _Z \
    / NULL ___ ($AllE | $AllI)

OB_RULE 0201, g EMPTY -> g _g \
    / NULL ___ NULL
```

2.3. From rules to transducers

The compilation of the rules results in a very large number of *WFSTs* (almost 500) that need to be composed in order to build a single grapheme-to-phone transducer. We did not build a single *WFST* but selectively composed the *WFSTs* and obtained a small set of 10 *WFSTs* that are composed with the grapheme *WFST* in runtime to obtain the phone *WFST*.

The most problematic phase was `gr2ph`. We started by composing each of the other phases in a single *WFST*. `gr2ph` was first converted to a *WFST* for each grapheme. Some graphemes, such as e, lead to large transducers, while others, lead to very small ones. Due to the way we specified the rules, the order of composition of these *WFSTs* was irrelevant. Thus we had much flexibility in grouping them and managed to obtain 8 transducers with an average size of 410k. Finally, `introduce-phones` and `remove-graphemes` were composed with other *WFSTs* and we obtained the final set of 10 *WFSTs*.

In runtime, we can either compose the grapheme *WFST* in sequence with each *WFST*, removing dead-end paths at each step, or we can perform a lazy simultaneous composition of all *WFSTs*. This last method is slightly faster than the DIXI system.

2.4. Evaluation

We evaluated the *WFST*-based rule approach, and compared its performance with the one of our previous rule-based DIXI system. As can be seen in table 1, the *WFST* achieved almost the error rate of the DIXI system it is emulating, both at a word level and

at a segmental level. The two rightmost columns show the error rates obtained without taking stress mark errors into account. The difference between the performance of the current and previous approaches is due to the *exception lexicon* included in DIXI that we did not yet implement. Such a lexicon can be easily implemented as a *WFST* applied after `prefix-lexicon`. We plan to integrate this lexicon and balance its size with the rule system, in order to simplify it by replacing rules that apply to just a few words with lexicon entries.

System	*% Error*		*% Error w/o stress*	
	word	*segm.*	*word*	*segm.*
WFST	3.56	0.54	3.13	0.47
DIXI	3.25	0.50	2.99	0.45

Table 1. Comparison of the current and previous rule-based approaches.

3. DATA-DRIVEN APPROACH

3.1. Grapheme-Phone Alignment

The first step in preparing the corpus for the data driven techniques consisted of aligning each grapheme with the corresponding phone.

We performed the alignment by minimizing the string-edit distance between corresponding grapheme and phone strings. We encoded the distance between any grapheme and any phone, as well as the insertion costs of phones and the deletion costs of graphemes as a transducer s. The alignment between a grapheme sequence g and a phone sequence p was obtained as $bestpath(g \circ s \circ p)$.

When creating s we opted to capitalize on the knowledge obtained from the rule system, although automatic techniques exist that can learn such a transducer automatically [15].

Besides the usual matching of 1 grapheme to 1 phone, we also allowed the direct matching of some sequences. The cost of matching a grapheme *sequence* with a phone *sequence* was set to zero if there is a rule that assigns the phone sequence to the grapheme sequence (completely ignoring the context of the rule). In most cases, the matching was of 1 grapheme to 1 phone, but we modeled some cases of 2 graphemes to 1 phone (such as nh → /J/, lh → /L/, rr → /R/) and some cases of 1 grapheme to 2 or 4 phones (such as ê → /6∼j∼6∼j∼/ in *têm*). In order to score all possible alignments, we allowed the alignment of a grapheme with any phone, the deletion of graphemes and the insertion of phones, also at a cost. The costs were set empirically but are similar to the costs used to determine the word error rate in speech recognition (3 for insertion and deletion, 4 for substitution and 0 for matching).

3.2. N-gram approach

The alignment obtained in section 3.1 is a sequence of pairs (grapheme, phone), where the grapheme or the phone can also be ϵ. Our first data-driven approach consisted of modeling that sequence using an n-gram model, as proposed by [16].

This model is based on the probability of a grapheme matching a particular phone given the history up to the previous $n-1$ pairs $(P((g_i, p_i)|(g_{i-n-1}, p_{i-n-1})...(g_{i-1}, p_{i-1})))$.

The language model is first converted to a finite-state acceptor (*WFSA*) over pairs of symbols, and then to a finite-state transducer t, by transforming each pair of symbols into an input and an output label. t is ambiguous because epsilons are used to model backoff transitions during the conversion from n-gram to *WFSA*, and hence, even is there is an explicit n-gram in the model, the *WFSA* will still allow alternative paths that use the backoff.

Due to this ambiguity, in order to use the *WFST* to convert a grapheme sequence *WFST* g to phones, we need to compute $bestpath(\pi_2(g \circ t))$.

We trained various n-gram backoff language models using history lengths $n-1$ ranging from 2 to 7. Table 2 shows the size of the various models, and table 3 shows the error rate on the test set (second and third columns).

n	*n-grams*	*states*	*edges*	*bytes*
8	1,392,426	820,778	1,983,113	42M
7	981,565	592,184	1,459,738	30M
6	657,107	361,944	980,123	20M
5	401,855	159,425	549,398	11M
4	173,307	37,869	208,668	4M
3	42,451	3,618	46,018	0.8M

Table 2. Pair n-gram *WFSTs*.

n	*% Error*		*% Error w/o stress*	
	word	*graph.*	*word*	*graph.*
8	9.04	1.37	6.11	0.90
7	9.02	1.37	6.12	0.90
6	9.16	1.37	6.13	0.90
5	9.86	1.46	6.38	0.93
4	15.34	2.25	9.23	1.32
3	31.62	4.62	18.42	2.67

Table 3. Performance of the n-gram approach.

4. COMBINING DATA-DRIVEN AND KNOWLEDGE-BASED APPROACHES

One of the greatest advantages of the *WFST* representation is the flexible way in which different methods may be combined. In this section we show some examples of the combination of data-driven with knowledge-based methods.

In [16], as an example of the integration of knowledge-based with data-driven methods, some improvements were obtained by composing the n-gram *WFST* with a *WFST* that restricts the primary stress to exactly one per word. This type of restriction had also been implemented in our neural network method as a post-processing filter.

We opted for a different approach: as we have the stress marking *WFST* `stress`, we decided to perform the grapheme-phone alignment of the training data not with the original words, but with the output of the `stress` *WFST*. The alignments thus obtained were used to build n-gram *WFSTs*, as described in section 3.2. To convert a sequence of graphemes g to phones, we now use $bestpath(\pi_2(g \circ \texttt{stress} \circ t))$. Table 4 shows the results obtained with this variation with several n-gram models. We observe a reduction of the word error rate to less than half. The result is even more impressive when we remember that around 90% of the training set was converted by rule with a system that has around 3% errors. The size of the n-gram *WFSTs* was similar.

n	*% Error*		*% Error w/o stress*	
	word	*graph.*	*word*	*graph.*
8	4.01	0.61	3.65	0.54
7	3.94	0.59	3.58	0.53
6	4.02	0.61	3.66	0.55
5	4.04	0.61	3.68	0.55
4	4.48	0.67	4.13	0.60
3	6.40	0.96	6.15	0.91

Table 4. Performance of the n-gram approach, when trained and used after the stress marking *WFST*.

5. CONCLUDING REMARKS

This paper compared several *WFST*-based approaches to GtoP for European Portuguese: rule-based, data-driven and hybrid. Best results were obtained with the rule-based approach, but one should take into account the fact that the data-driven one was trained with automatically transcribed material.

The comparison between the different approaches should consider as well the size of the resulting transducers and other properties which may also be quite relevant, such as the fact that the rule-based approach generates dead ends, whereas the n-gram approach does not, but requires a best-path search.

In the future, we plan to improve our rule-based approach by obtaining a better balance between number of rules and lexicon size, as explained earlier. We also plan to convert our *CART*-based approach to the *WFST* framework. This will give us much flexibility in combining the various methods, for example, a *WFST* resulting from the conversion of the tree of a particular grapheme could replace the respective grapheme rules in the *WFST* rule-based system.

Another type of approach we plan to explore is transformation-based learning (TBL). This approach was first proposed as an effective way to learn part-of-speech (POS) rules [17], and it has since been used in other tasks such as spelling correction, dialogue act tagging, etc. In [11] it was used as a way to improve a Dutch rule-based grapheme-to-phoneme system. TBL is a very attractive technique because the learned rules can be very readable and interpretable. Furthermore, the learned rule-set can be converted to an efficient deterministic *WFST* [18].

The inversion property of transducers opens the possibility of using GtoP techniques in tasks such as reconstructing out of vocabulary words [19] in large vocabulary speech recognition systems. This is an area which we also plan to explore in the near future.

6. REFERENCES

[1] L. Oliveira, M. Viana, and I. Trancoso, "A Rule-Based Text-to-Speech System for Portuguese," in *Proc. ICASSP '1992*, San Francisco, USA, March 1992.

[2] S. Lazzaretto and L. Nebbia, "Scyla: Speech Compiler for your Language," in *Proc. of the European Conf. on Speech Technology*, Edimburgh, UK, September 1987, vol. 2.

[3] I. Trancoso, M. Viana, F. Silva, G. Marques, and L. Oliveira, "Rule-Based vs. Neural Network Based Approaches to Letter-to-Phone Conversion for Portuguese Common and Proper Names," in *Proc. ICSLP '94*, Yokohama, Japan, Sept. 1994.

[4] L. Oliveira, M. C. Viana, A. I. Mata, and I. Trancoso, "Progress Report of Project DIXI+: A Portuguese Text-to-Speech Synthesizer For Alternative and Augmentative Communication," Tech. Rep., FCT, Jan. 2001.

[5] A. Black and K. Lenzo, "Building Voices in the Festival Speech Synthesis System," in *Festival version 1.4.1, CMU*, 2000.

[6] R. Sproat and M. Riley, "Compilation of Weighted Finite-State Transducers from Decision Trees," in *34th Annual Meeting of the Association for Computational Linguistics*, Santa Cruz, USA, 1996.

[7] L. Karttunen, K. Koskenniemi, and R. Kaplan, "A Compiler For Two-Level Phonological Rules," Tech. Rep. CSLI-87-108, Center for the Study of Language and Information, Stanford University, 1987.

[8] R. Kaplan and M. Kay, "Regular Model of Phonological Rule Systems," *Computational Linguistics*, vol. 3, no. 20, pp. 331–378, 1994.

[9] J. Neto, C. Martins, H. Meinedo, and L. Almeida, "The Design of a Large Vocabulary Speech Corpus for Portuguese," in *Proc. Eurospeech '97*, Rhodes, Greece, Sept. 1997.

[10] N. Chomsky and M. Halle, *Sound Pattern of English*, Harper and Row, 1968.

[11] G. Bouma, "A Finite-State and Data-Oriented Method for Grapheme to Phoneme Conversion," in *1st Meeting of the North American Chapter of the Association for Computational Linguistics*, Seattle, USA, 2000.

[12] K. Koskenniemi, *Two-Level morphology: A general Computational Model for Word-Form Recognition and Production.*, Ph.D. thesis, University of Helsinki, 1983.

[13] E. L. Antworth, "PC-KIMMO: A Two-Level Processor for Morphological Analysis," Tech. Rep., Occasional Publications in Academic Computing No 16. Dallas, TX: Summer Institute of Linguistics, 1990.

[14] M. Mohri and R. Sproat, "An Efficient Compiler for Weighted Rewrite Rules," in *34th Annual Meeting of the Association for Computational Linguistics*, Santa Cruz, USA, 1996.

[15] W. Daelemans and A. van den Bosch, "Language-Independent Data-Oriented Grapheme-to-Phoneme Conversion," in *Progress in Speech Synthesis*, J. van Saten, R. Sproat, J. Olive, and J. Hirschberg, Eds. Springer, New York, USA, 1997.

[16] R. Sproat, "Corpus-Based Methods and Hand-Build Methods," in *Proc. ICSLP '2000*, Beijing, China, October 2000.

[17] E. Brill, "Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part-of-Speech Tagging," *Computational Linguistics*, vol. 21, pp. 543–566, 1995.

[18] E. Roche and Y. Schabes, "Deterministic Part-of-Speech Tagging with Finite-State Transducers," *Computational Linguistics*, vol. 21, pp. 227–253, June 1995.

[19] B. Decadt, J. Duchateau, W. Daelemans, and P. Wambacq, "Transcription of Out-of-Vocabulary Words in Large Vocabulary Speech Recognition Based on Phoneme-to-Grapheme Conversion," in *Proc. ICASSP'2002*, Orlando, Florida, May 2002.