SPOKEN LANGUAGE PROCESSING USING WEIGHTED FINITE STATE TRANSDUCERS

Isabel Trancoso, Diamantino Caseiro

L^2F Spoken Language Systems Lab.
INESC-ID/IST
Rua Alves Redol 9, 1000-029 Lisbon, Portugal
{Isabel.Trancoso, dcaseiro}@l2f.inesc-id.pt

ABSTRACT

The main goal of this paper is to illustrate the advantages of weighted finite state transducers (*WFSTs*) for spoken language processing, namely in terms of their capacity to efficiently integrate different types of knowledge sources. We shall illustrate their applicability in several areas: large vocabulary continuous speech recognition, automatic alignment using pronunciation modeling rules, grapheme-to-phone conversion, and speech-to-speech translation. The impact of the use of *WFSTs* in spoken language processing for European Portuguese was particularly noticeable in the area of broadcast news recognition, in which we used a specialized composition algorithm for composing the lexicon with the language model. Among other properties, this algorithm allows the on-the-fly generation of the composition *WFST*, being thus amenable to be embedded in a dynamic recognition system. The *WFST* approach achieved a 6 times reduction of the decoding time relative to a previous decoder not based on *WFSTs*.

1. INTRODUCTION

This paper tries to illustrate the advantages of using weighted finite state transducers (*WFSTs*) for spoken language processing, namely in terms of providing an elegant framework in which different types of knowledge sources can be efficiently integrated. There are many possible applications of *WFSTs* to spoken language processing. The ones that we have chosen as topics for this paper attempt to emphasize different aspects of transducers.

- Our first example is large vocabulary continuous speech recognition (*LVCSR*). Our focus will be on how to do the composition of knowledge sources dynamically, with limited memory requirements, and being able to preserve the original sources.
- Our second example is a related topic - alignment, in which our emphasis will be on the flexible way in which additional knowledge sources such as alternative pronunciation rules may be integrated.
- Our third example is a particular module of text-to-speech synthesis system, which is also closely related to the above pronunciation rules - grapheme-to-phone (*GtoP*) conversion. Here, we hope to illustrate the benefits of combining both knowledge-based and data-driven approaches via transducers.
- Our final example is speech-to-speech machine translation (*S2SMT*), where our emphasis will be on how to integrate recognition with translation.

2. TRANSDUCERS IN LVCSR

The main advantages of the transducer-based approach relative to traditional systems are the elegant and uniform formalism that allows very flexible ways of integrating multiple knowledge sources, and the superior search performance obtained when the search network is optimized using automata determinization and minimization. The main disadvantages are related to the search space optimization, both in terms of memory requirements and dynamic adaptation. In fact, the *WFST* determinization algorithm, based on subset construction, and normally used during optimization, requires large amounts of memory relative to the size of the resulting *WFST*. Moreover, although the optimized search network is not much larger than the language model (typically only 2 to 2.5 times larger), it can still be very large and requires large amounts of memory in runtime. The fact that the optimization of the search network is performed offline means that the original knowledge sources are not available at runtime. Thus, it may be troublesome to preserve the optimality of the network, when dynamically adjusting the knowledge sources. For example, when adapting the language model probabilities or when adding new words or pronunciations to the vocabulary.

This section describes how we addressed these problems. In particular, we shall focus on our lexicon and language model "on-the-fly" composition algorithm which, together with a memory efficient representation for *WFSTs* and other language model optimizations [1], allowed us to extend the *WFST* approach to larger systems, such as the ones used for broadcast news recognition.

2.1. Lexicon and Language Model Composition

The determinization of the composition of the lexicon L with the language model G is probably the most resource intensive subtask when optimizing the search network. The reason lies on the large size of the language model, and on the fact that, when applying the subset-construction determinization algorithm, every state of the resulting *WFST* ($det(L \circ G)$) corresponds to a *set* of states of the non-deterministic $L \circ G$ transducer.

In [2] we presented a memory-efficient specialized algorithm for the composition of the lexicon with the language model. Our algorithm is based on Mohri's theorem [3] that states that the composition of sequential transducers is also sequential. This important result means that if we determinize both the lexicon and the language model, then we only need to compose them to obtain the deterministic composition. In practice, we cannot just apply the usual composition algorithm [4], because of ϵ labels on the output tape of the lexicon, which generate too many non-coaccessible[1] paths.

Our specialized composition algorithm works as follows: in a preprocessing step, the set of reachable non-ϵ output labels is associated with each ϵ-output edge of the lexicon. That set is used during composition to avoid the generation of non-coaccessible paths by only following ϵ-output edges on the lexicon that will lead to a non-ϵ label compatible with labels in the language model state. This specialized algorithm was later extended to allow output-label and weight pushing. In [5] we showed how to extend the algorithm to approximate “on-the-fly” minimization.

The specialized algorithm imposes some limitations on the structure of the lexicon. We feel that the most restrictive one is imposing that the lexicon loops through the initial state, since it limits cross word lexicon modelling. Yet we believe that this restriction can be easily overcome if word transitions are somehow marked (for example, by using special ”end-of-word” transitions) to be identified by the composition algorithm.

When used with a caching scheme, the overhead of performing both the $LG = L \circ G$ specialized composition and the $H \circ LG$ composition in runtime, is only 20% of the search effort[2].

2.2. Application to Broadcast News Recognition

All the recognition experiments described in this section were based on the broadcast news corpus collected in the scope of the ALERT European project[6] for European Portuguese.

The acoustic models are based on the combination of the output of various neural networks [7]. We extracted 3 different sets of features from the speech signal: 12 plp coefficients + log energy + deltas; 12 log-rasta coefficients + log energy + deltas; and 28 modulation spectrogram features.

We used 3 separate multilayer perceptrons (*MLP*), one for each set of features. The input of each *MLP* was a window of 7 vectors centered on the vector being analyzed. The *MLPs* had a 3-layer architecture with 1000-4000 units in the hidden layer, and the output consisted of 40 softmax units corresponding to 38 context independent phones plus silence and inspiration noise. The output of the 3 *MLPs* was combined using the average of the logarithm of the probability estimated for each phone.

The acoustic model topology consisted of a sequence of states with no self-loop to enforce the minimal duration of the model, and one final state with a self-loop. The acoustic models were encoded in a single acoustic model *WFST*.

We used an European Portuguese lexicon with 57k words and 4-gram backoff language models, trained from more than 384 million words from newspaper texts and interpolated with models obtained from broadcast news transcriptions.

Table 1 shows the results obtained with a development test set of 6h, in successive alignment passes (first column) as we changed several parameters in the system: number of units in the hidden layer (second column), number of hours of training data (third column), and type of decoder (fourth column). The word error rates are shown both for F0 and all conditions (fifth and sixth columns) together with the corresponding speed (rightmost column).

The most impressive result is the change in real-time performance observed when we moved from our previous stack decoder to the *WFST* based implementation, obtained without degradation in terms of WER.

Align	MLP	Data	Decoder	F0	All	xRT
5	1000	22h	Stack	18.3	33.6	30.0
“	“	“	WFST beam 5.5	18.7	32.0	6.4
6	“	46h	“	18.7	31.6	4.1
7	“	“	“	18.8	31.6	4.8
“	2000	“	“min det L	18.0	30.7	4.3
“	4000	“	“	16.9	29.1	3.7

Table 1. Results with BN recognition in successive alignment passes.

3. TRANSDUCERS IN ALIGNMENT

An automatic alignment module can be used for different purposes. In our past research, we have use such modules as a step in the bootstrap process of training better models for ASR and also as a step to segment better units for concatenative speech synthesis. This section starts with the description of how we modified our *WFST*-based decoder to be used as an aligner, and proceeds with the description of how we tried to cope with pronunciation variation by adding a new knowledge source to our search space - alternative pronunciation rules.

3.1. Decoder modifications for alignment purposes

Our aligner is based on *WFSTs* in the sense that its search space is defined by a distribution-to-word (or distribution-to-phone) transducer that is built outside the decoder. For the alignment task, that search space is usually build as $H \circ L \circ W$, where H is the phone topology, L is the lexicon and W is the sequence of words that constitutes the orthographic transcription of the utterance. As no restrictions are placed on the construction of the search space, it can easily integrate other sources of knowledge, and can be optimized and replaced by an optimal equivalent one.

In order to cope with possible de-synchronizations between the input and output labels of the *WFST*, the decoder was extended to deal with special input labels that are internally treated as epsilon labels (similar to skip arcs in Hidden Markov Models), but are used to mark time transitions or boundaries. Whenever such end-of-segment labels are crossed, the time is recorded in the current hypothesis. The user may choose to place those labels at the end of each phone *WFST* or at the end of each word *WFST*, depending on choosing either phone-level alignment or word-level alignment, respectively.

3.2. Alternative pronunciation rules

The way we dealt with pronunciation variation has some similarities with the one described in [8]. The variations that depend on word-level features of lexical items (such as part-of-speech) and those that are particular to specific lexical entries (such as many acronyms in Portuguese, for instance) are included in the lexicon (denoted by *Lex0* in our experiments).

The remaining variants that depend on the local immediate segmental context are modeled through rules [9]. Rather than specifying rules which would mainly affect function words and

[1] A non-coaccessible path is a “dead-end” paths that does not reach a final state.

[2] The time spent evaluating the distributions (neural network or Gaussian mixtures) is not included in this percentage.

forms of the verb *estar* (to be), we included in the lexicon (denoted as *Lex1*) multiple pronunciations for 40 such words in which we could observe so much deviation from the canonical pronunciation.

Some of the rules concern variations that depend on the stress and syllable position. The lexicon may use different labels for representing segments in particular positions. For instance, label I denotes a frequent alternation between [i], [E] and [e] in the beginning of some words starting by "e". When no rules are applied, the default pronunciation is [i].

The main phonological aspects that the remaining rules are intended to cover are: vowel devoicing, deletion and coalescence, voicing assimilation, and simplification of consonantal clusters, both within words and across word boundaries. Some common contractions are also accounted for, with both partial or full syllable truncation and vowel coalescence. Vowel reduction, including quality change, devoicing and deletion, affects mostly European Portuguese, being one of the features that distinguishes it from Brazilian Portuguese and that makes it more difficult to learn for a foreign speaker. As a result of vowel deletion, rather complex consonant clusters can be formed across word boundaries.

When aligning using alternative pronunciation rules, the search space becomes $H \circ P^{-1} \circ L \circ W$, where P^{-1} is the inverse of the rule transducer. The rules are specified using a finite-state grammar whose syntax is similar to the Backus-Naur-form augmented with regular expressions. We added the operator →, simple transduction, to the usual set of operators, such that $(a \rightarrow b)$ means that the terminal symbol a is transformed into the terminal symbol b. The language allows the definition of non-terminal symbols (e.g. *$Vowel*). All rules are optional by default, and are compiled into *FSTs*. In our case, we did not have enough manually labeled material to train weights. We do not apply the rules one by one on a cascade of compositions, but, because they are optional, we rather build their union in order to avoid the exaggerated growth of the resulting transducer, which can be exponential with the length of the composition cascade. The rule transducer P is thus build as $P = \bigcup_i \Sigma^*(P_i\Sigma^*)^*$ where P_i is the transducer corresponding to a particular rule specification expression.

We also allow the specification of negative constraints, or *forbidden* rules, that disallow the occurrence of expressions or sequences. Such an expression P is compiled to $\Sigma^* \cap \overline{\Sigma^* P \Sigma^*}$.

Figure 3.2 shows an example of a sandhi rule specification, together with the forbidden counterpart. The rule set allows for /S/ not to be changed into /z/ only when there is a silence before the next word starting by a vowel. If there is no silence, then the path /S/ end-of-word (EOW) vowel is forbidden. (SAMPA symbols used.)

A different type of rules, involving contractions and multiword reductions can also be implemented through a reduction transducer (Rd) that encodes rules that map such reductions to their canonical form [8] (e.g. *gonna* → *going to*). The aligner search space becomes then $H \circ P^{-1} \circ L \circ Rd \circ W$.

```
$V = ($Vowel|$NasalVow|$Glide|$NasalGli)
$WB = (EOW (sil→NULL) (EOW→NULL))
DEF_RULE S_z, ($V (S → z) $WB $V)
FORBIDDEN_RULE No_S_z, (S EOW $V)
```

Fig. 1. Example of rule specification.

3.3. Application to the alignment of dialogs

The efficiency of the alternative pronunciation rule module was tested in the alignment of Coral - a map task dialog corpus with 64 dialogs, which was orthographically transcribed following the same transliteration conventions using SGML format of other map task corpora[3]. Because of the large amount of cross-talk observed (namely in overlapping turns), we adopted an adaptive noise canceling scheme [10], as a pre-processing stage.

Our alignment experiments were done with the acoustic models described above. The results shown in table 2 refer to the small subset of the Coral corpus that has been manually annotated at the phone level. In order to evaluate the distinct versions of our aligner, we used as a metric the phone level error rate and two additional measures: the percentage of matching phone labels for which the absolute error is less than 10ms and the average absolute error in 90% of the cases. A dynamic programming algorithm was developed to match the manually annotated labels with the automatically derived ones, minimizing their string-edit distance. The algorithm penalizes substitutions, insertions and deletions (costs 10, 7 and 7 respectively), but favors very common ones (cost 3).

Our first experiment was made with *Lex0* and no alternative pronunciation rules. An analysis of the largest errors shows they are due to the fact that we did not try to align laughs, annoying grunts, and filled pauses, which causes severe misalignments in the neighboring words (up to 5 ms). In fact, our acoustic models could not yet cope with such phenomena. The performance in overlapping turns is on the same level as the one in non overlapping turns. The second experiment was made with *Lex1* and still no rules. The values obtained with this new lexicon were good enough to make all further tests using this lexicon. We then followed an exhaustive process of testing the efficiency of several types of alternative pronunciation rules. The results obtained with 32 rules are shown in the last line of the table. We expected greater improvements but cannot dismiss the generalization capabilities of our hybrid acoustic models and also the fact that they cannot adequately model laughs and other voice quality changes that seriously affect some portions of the dialogs. A last experiment was made to test the efficiency of reduction rules, but such examples were not so frequent in our test corpus.

Lex	Rules	%ACC	≤ 10ms	Percentil 90%
Lex0	no	70.04	46.54	0.0122
Lex1	no	71.50	47.29	0.0118
Lex1	yes	78.19	48.57	0.0115

Table 2. Alignment results.

The application of our transducer-based aligner to spoken books is also worth mentioning in this context. The memory limitations of our previous alignment tool, based on the stack decoder, imposed a maximum of 3-minute audio segments. This demanded a very tedious partition of the audio into smaller segments with their associated text. A major advantage of our transducer-based approach is that it allowed us to align the full audio version of a book in a single step. The word segmentation of a 2-hour book took 197.5 seconds in a 600MHz Pentium III computer (0.024 xRT), and required 200MB of RAM.

[3] http://www.hcrc.ed.ac.uk/dialogue/maptask.html

4. TRANSDUCERS IN GTOP CONVERSION

The objective of a GtoP 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 GtoP conversion for European Portuguese was a rule-based system (DIXI), with about 200 rules [11]. All the code was programed in C, directly in the case of the stress assignment rules, and using the SCYLA [12] 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.).

This section starts with a description of how we compiled the rules of the DIXI system to *WFSTs*. It proceeds with the presentation of a data-driven approach and finally with the combination of the knowledge-based and the data-driven approach.

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 a newspaper text corpus. The resulting 205k words not in PF were automatically transcribed using DIXI and added to the training set.

4.1. 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.

In order to preserve the semantic of DIXI's rules we opted to use rewriting rules, but, to avoid unnecessary rule dependencies due to the replacement of graphemes by phones, we used them 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 sets of graphemes and phones do not overlap.

The rules are specified using the rule specification language described in section 3.2. This work motivated us to extend it with commands such as:

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[13].

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 4.1 shows how the various phases are composed. Each phase has the following function:

- introduce-phones is the simple rule that inserts the _ *empty phone* placeholder after each grapheme. (($Letter (NULL → EMPTY)) ⇒ ___).
- the exceptions-lexicon phase consists of a list of 364 entries which correspond to exceptions to the common lexica rules. A significant percentage of these entries is devoted to monosyllabic unstressed function words which constitute exceptions to the stress rules.
- the stress phase consists of 27 rules that mark the stressed vowel of the word.
- prefix-lexicon consists of pronunciation rules for compound words, namely with roots of Greek or Latin origin such as "tele" or "aero". It includes 92 rules.
- gr2ph is the bulk of the system, and consists of 340 rules, that convert the 45 graphemes (including graphically stressed versions of vowels) to phones.
- sandhi implements word co-articulation rules across word boundaries. (This rule set was not tested here, given the fact that the test set consists of isolated words.)
- remove-graphemes removes the graphemes in order to produce a sequence of phones.
  ($Letter → NULL / ___).

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

Fig. 2. Phases of the knowledge based system.

Figure 4.1 illustrates the specification of 4 gr2ph rules for deriving the pronunciation of grapheme r: either as /R/ (e.g. in the beginning of words, following certain consonants, or in double rr) or as /r/ otherwise. The default rule is applied last.

OB_RULE r1, r EMPTY → r _R / WordBoundary NULL ___ NULL /* rato */
OB_RULE r2, r EMPTY → r _R / (l | n | s | r) NULL ___ NULL /* bilro */
OB_RULE r3, r EMPTY → r DEL / NULL ___ r /* carro */
OB_RULE r4, r EMPTY → r _r / NULL ___ NULL /* caro */

Fig. 3. Example of GtoP rule specification.

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. 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 a final set of 10 *WFSTs* that are sequentially composed with the grapheme *WFST* in runtime to obtain the phone *WFST*. We can also perform a lazy simultaneous composition of all *WFSTs*. This last method is slightly faster than the DIXI system.

We evaluated the *WFST*-based rule approach, and compared its performance with the one of our previous rule-based DIXI system, which shared the same exceptions lexicon. As expected, the *WFST* achieved the same error rate of the DIXI system it is emulating, both at a word level (3.25%) and at a segmental level (0.54%).

4.2. Data-Driven Approach

The first step in preparing the training 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, obtaining a sequence of pairs (grapheme, phone), where the grapheme or the phone can both be ϵ. Our first data-driven approach consisted of modeling that sequence using an n-gram model, as proposed by [14].

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 3 shows the size of the various models, and table 4 shows the error rate on the test set (second and third columns).

n	*n-grams*	*states*	*edges*	*bytes*
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

Table 3. Pair n-gram *WFSTs*.

n	*% Error*		*% Error w/o stress*	
	word	*graph.*	*word*	*graph.*
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

Table 4. Performance of the n-gram approach.

4.3. 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 [14], 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 a neural network based approach that we developed [15] as a post-processing filter.

Here, 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 4.2. To convert a sequence of graphemes g to phones, we now use $bestpath(\pi_2(g \circ \texttt{stress} \circ t))$. Table 5 shows the results obtained with this variation using 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.*
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

Table 5. Performance of the combined approach.

5. SPEECH-TO-SPEECH TRANSLATION

This section describes very preliminary ongoing work on speech-to-speech translation between two very close languages: Portuguese and Spanish. *WFSTs* provide a natural framework for the integration of the components of a *S2SMT*system. Such a system could be build as $R \circ T \circ S$, where R, T and S are, respectively the recognition, translation and synthesis systems. We are still far from the construction of such a system and are starting working on $R \circ T$, the integration of recognition with translation. Our focus has been on integrating a translation transducer T built using grammar inference techniques [16, 17], by using it as a “language model” in a *WFST*-based speech recognizer. The search space used is $N = H \circ L \circ T$, N is a transducer which maps directly from source speech distributions to target language text. By using this search space instead of the usual $H \circ L \circ G$, the speech recognizer can search directly for the best translation for the input speech. This integration technique is interesting particularly when translating between linguistically close languages, with similar word order. The use of the specialized composition described in section 2.1 to optimize N is advantageous relative to the use of explicit determinization, since it does not require the use of a deterministic translation transducer, thus allowing more flexibility in the construction of the translation module.

6. CONCLUDING REMARKS

This paper attempted to illustrate the potential of *WFSTs* for spoken language processing. This potential is leading us to apply transducers to yet more areas such as unit selection [18] and text normalization in concatenative speech synthesis. At the same time as pursuing these ambitious goals, we have not forgotten many open issues in the topics we have investigated so far.

In terms of LVCSR, we are currently working on extending our system to include larger lexica and language models, and also on how to incorporate more sophisticated knowledge sources.

In terms of alignment, the whole investigation of alternative pronunciation rules is still dependent on the existence of manually labeled spontaneous speech data, so that we can test the effectiveness of different types of rule. This manual labeling process is now currently being done using as a starting point the automated labels achieved so far. In spite of this open issue, the robustness of the *WFST*-based aligner was fully demonstrated both by its application to spoken books and to channel-separated dialogs.

In terms of GtoP, the number of open issues is also too large. 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.

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. Another goal is the development of GtoP modules for other varieties of Portuguese.

This long wish list could not be complete without mentioning all the work that remains to be done in terms of the application of *WFSTs*to speech-to-speech translation, an area in which our group has barely scratched the surface.

7. ACKNOWLEDGEMENTS

The authors would like to thank FLAD for their support. This work was patially done in the scope of project POSI/PLP/47175/2002, funded by FSE/FEDER. INESC-ID Lisboa had support from the POSI program of the "Quadro Comunitario de Apoio III".

8. REFERENCES

[1] D. Caseiro and I. Trancoso, "Using Dynamic WFST Composition for Recognizing Broadcast News," in *Proc. ICSLP '2002*, Denver, USA, Sept. 2002.

[2] D. Caseiro and I. Trancoso, "On Integrating the Lexicon with the Language Model," in *Proc. Eurospeech '2001*, Aalborg, Denmark, Sept. 2001.

[3] M. Mohri, "Finite-State Transducers in Language and Speech Processing," *Computational Linguistics*, vol. 2, no. 23, 1997.

[4] M. Mohri, F. Pereira, and M. Riley, "Weighted Automata in Text and Speech Processing," in *ECAI 96 Workshop*. Budapest, Hungary, Aug. 1996.

[5] D. Caseiro and I. Trancoso, "Transducer Composition for "On-the-Fly" Lexicon and Language Model Integration," in *Proc. ASRU '2001 Workshop*, Madonna di Campiglio, Trento, Italy, Dec. 2001.

[6] H. Meinedo, N. Souto, and J. Neto, "Speech Recognition of Broadcast News for the European Portuguese Language," in *Proc. ASRU '2001 Workshop*, Madonna di Campiglio, Trento, Italy, Dec. 2001.

[7] H. Meinedo and J. Neto, "Combination of Acoustic Models in Continuous Speech Recognition Hybrid Systems," in *Proc. ICSLP '2000*, Beijing, China, Oct. 2000.

[8] T. Hazen, I. Hetherington, H. Shu, and K. Livescu, "Pronunciation Modeling Using a Finite-State Transducer Representation," in *Proc. ISCA Tutorial and Research Workshop on Pronunciation Modeling and Lexicon Adaptation*, Aspen, CO, USA, Sept. 2002.

[9] I. Trancoso, D. Caseiro, C. Viana, F. Silva, and I. Mascarenhas, "Pronunciation modeling using finite state transducers," in *Proc. 15th International Congress of Phonetic Sciences (ICPhS'2003)*, Barcelona, Spain, Aug. 2003.

[10] B. Widrow, J. Glover, J. McCool, C. Williams, R. Hearn, J. Zeidler, Dong E., and R. Goodlin, "Adaptive Noice Canceling: Principles and Applications," *Proceedings of the IEEE*, vol. 63, pp. 1692–1716, Dec. 1975.

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

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

[13] 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.

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

[15] 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.

[16] A. Castellanos, I. Galiano, and E. Vidal, "Application of OSTIA to Machine Translation Tasks," in *Grammatical Inference and Applications (ICGI-94)*, Berlin, Sept. 1994.

[17] E. Cubel, J. Gonzalez, A. Lagarda, F. Casacuberta, A. Juan, and E. Vidal, "Adapting Finite-State Translation to the TransType2 Project," in *Proc. EAMT/CLAW 2003*, Dublin, Ireland, May 2003.

[18] P. Carvalho, I. Trancoso, and L. Oliveira, "WFST based Unit Selection for Concatenative Speech Synthesis in European Portuguese," in *Proc. 15th International Congress of Phonetic Sciences (ICPhS'2003)*, Barcelona, Spain, Aug. 2003.

[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.