Asymmetric Term Alignment with Selective Contiguity Constraints by Multi-Tape Automata

Mădălina Barbaiani[1,2], Nicola Cancedda[1], Chris Dance[1], Szilárd Fazekas[1,2], Tamás Gaál[1], and Éric Gaussier[1,3]

[1] Xerox Research Centre Europe – Grenoble Laboratory, 6 chemin de Maupertuis, 38240 Meylan, France
[2] Department of Computer Science, Rovira i Virgili University, Tarragona, Spain
[3] Joseph Fourier University, Grenoble, France

Abstract. This article describes a HMM-based word-alignment method that can selectively enforce a *contiguity constraint*. This method has a direct application in the extraction of a bilingual terminological lexicon from a parallel corpus, but can also be used as a preliminary step for the extraction of phrase pairs in a Phrase-Based Statistical Machine Translation system. Contiguous *source* words composing terms are aligned to contiguous *target* language words. The HMM is transformed into a Weighted Finite State Transducer (WFST) and contiguity constraints are enforced by specific multi-tape WFSTs. The proposed method is especially suited when basic linguistic resources (morphological analyzer, part-of-speech taggers and term extractors) are available for the source language only.

1 Introduction

Specialized bilingual terminologies are essential to technical translators for ensuring correctness and consistency in large translation projects. This is attested by the presence, in professional translation environments, of tools to collect and navigate terminologies. Several methods for extracting multilingual terminologies from parallel document collections have been proposed [1–6]. Unlike these methods, the method described here does not require the availability of a morphological analyzer and a POS tagger for both languages.

Besides lexicon and terminology extraction, word alignment is also an essential step in most Statistical Machine Translation approaches, as well as in the projection of linguistic resources across parallel corpora. Like existing methods for performing word alignment based on Hidden Markov Models (HMMs) [7], ours builds an HMM with one state for each words in a source language emitting words in the target language, and associates alignments with paths through the HMM. Our method allows the enforcement of the constraint that target words aligned to a same source term should be contiguous (*contiguity constraint*), thus restricting the alignment search space according to a generally acknowledged linguistic principle, and leading to improved lexica. This is done without imposing that alignments be *monotonic*, i.e. that word order is fully preserved in

the two languages. The method uses an implementation of weighted multi-tape finite state automata [8–10].

This paper focuses on the task of word alignment as it is critical in performing automatic terminology extraction. Existing methods to align documents at the sentence level [11–14], to identify candidate terms [15] and to extract a terminological dictionary from the word-aligned corpus can be used in conjunction with what we present here to complete the overall task.

2 Word Alignment, Sentence-Pair HMM

Consider a sentence pair $({e_1}^I, {f_1}^J)$, ${e_1}^I$ being the English sentence and ${f_1}^J$ its foreign translation consisting of I and J words, respectively[4]. Elements in ${e_1}^I$ can be either (possibly multi-word) terms or individual words occurring outside term boundaries. For reasons that will become clear soon, we will call such elements *states*. Let ${a_1}^J$ be the sequence of alignment variables, with $a_j = i$ if and only if the foreign word f_j is aligned to the English word/term e_i. We will restrict our attention to alignments in which a foreign word must be assigned to one and only one English state. Our objective is to determine ${a_1}^{J*}$ such that:

$$ {a_1}^{J*} = \text{argmax}_{{a_1}^J}\{P({a_1}^J|{f_1}^J, {e_1}^I)\} \tag{1} $$

Applying Bayes' rule and making some conditional independence assumptions, this becomes:

$$ {a_1}^{J*} = \text{argmax}_{{a_1}^J}\{\prod_{j=1}^{J} p(f_j|e_{a_j})p(a_j|a_{j-1})\} \tag{2} $$

We can model our conditional probability distribution by means of a Hidden Markov Model (HMM) with states $e_i, i = 1 \ldots I$ and emitting in the foreign alphabet Σ_F.[5]

The method needs estimates of word translation probabilities in the form $p(f|e)$, that is, for each English word e, the probability that it will be translated with word f. These can be achieved, for example, by the use of an Expectation-Maximization algorithm [16] using a translation model like in [17]. Further preprocessing may include lowercasing, lemmatisation, stemming, and the decomposition of compound words.

Emission probabilities can be estimated from the word translation probabilities. For transition probabilities, we would prefer to favor monotonic alignments but allow other arrangements, too: a two-component discrete Gaussian mixture is appropriate to model this. The states of the HMM are either contiguous multi-word terms or out-of-term words. Figure 1 shows an example.

[4] Following a convention similar to the standard one in the MT community, in the following we will assume that English is the source language, for which resources are available, and a *foreign* language is what we previously referred to as the target language. Needless to say, the method is, in its elements described here, independent of the actual language pair.

[5] We use throughout the term *alphabet* in the language theory sense of "set of symbols". In our case symbols will be words, not individual characters.

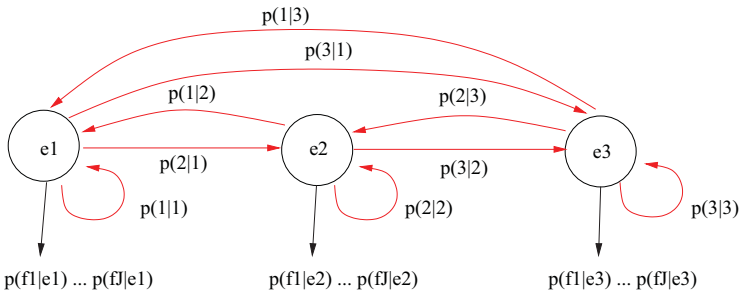

Fig. 1. HMM for three English terms; there is a single state for each term and the transition probabilities are modeled by two-component discrete Gaussian mixtures. I (3 here) and J are source- and target-language term indices, respectively.

The transformation of the HMM into a WFST of tropical semiring (W_{HMM}) is as follows:

- For every state of the HMM we create a state in the WFST
- Between any two states e_i and $e_{i'}$ (including $i = i'$) we add J transitions labeled by the foreign words on the input tape and i' on the output tape with weight $-\log p(i'|i) - \log p(f_k|e_{i'})$
- We create an initial state from which there will be a transition to every state e_i labeled f_1 with weight $-\log p(f_1|e_i)$
- Every state except the initial one is final.

A part of such an automaton is illustrated in Figure 2.

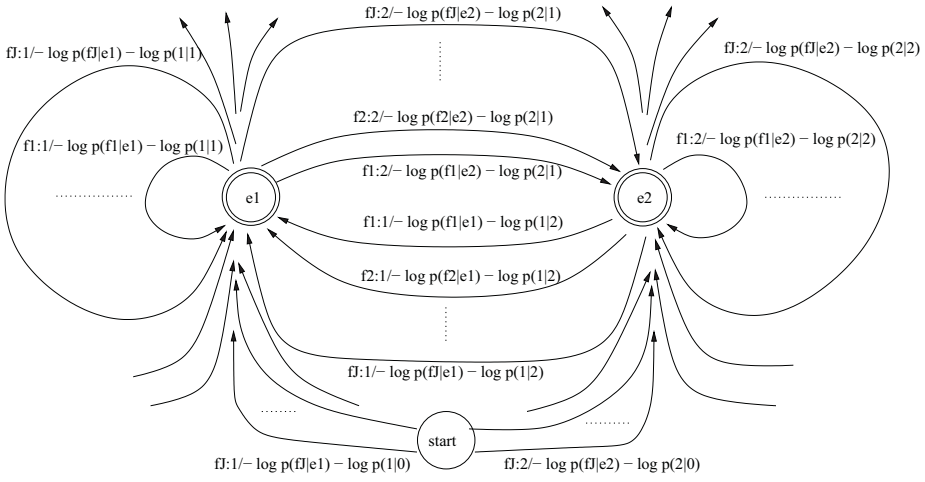

Fig. 2. Two states of the WFST built from the HMM, and their connection to the initial state.

3 Enforcing Contiguity

After turning our HMM into a weighted transducer, we can obtain the most probable alignment by applying Dijkstra's or Viterbi's shortest path algorithm. If we do so directly, with the weighted transducer, however, we are not guaranteed to obtain alignments respecting the contiguity constraint for terms. Formally:

An alignment $a_1{}^J$ *is contiguous if and only if there do not exist integers* i, j, k*, with* $1 \leq i < j < k \leq J$*, such that* $a_i = a_k = t$ *and* $a_j \neq t$*, for any* t *such that* e_t *is a term.*

The method we propose to solve this problem is inspired by [18]. In [18] the authors considered the problem of modeling a probability distribution over all alternative permutations of the chunks in a sequence: in our case, however, the foreign and the English sequence do not necessary have the same length, and the constraint we need to enforce is somewhat different.

Given a WFST encoding an HMM with I states, we build an automaton PE with one state for each pair of a subset of states (term states visited before the last) and a state (the last visited state if it is a term state, a wildcard symbol otherwise). The path-set in this automaton will be such that only sequences over $\{1 \ldots I\}$ that do not contain discontinuous repetitions of term identifiers will be accepted:

$$PE = (Q_{PE}, \Sigma_{PE}, E_{PE}, I_{PE}, F_{PE})$$

Q_{PE}: $\{\text{start}\} \cup \{\langle A, \sigma \rangle \in 2^{\Sigma_t} \times \Sigma | \sigma_{PE} \notin A\}$

Σ_{PE}: $\{1, \ldots, I\}$

$$\Sigma_{PE} = \Sigma_t \cup \Sigma_n,\ \Sigma_t \cap \Sigma_n = \emptyset$$

E_{PE}: $(\text{start}, \sigma, \langle \emptyset, \sigma \rangle)$

$$\forall \sigma \in \Sigma_{PE}$$

$$(\langle A, \sigma \rangle, \sigma', \langle A \cup \{\sigma\}, \sigma' \rangle)$$

$$\forall A \subseteq \Sigma_t, \forall \sigma \in \Sigma_t, \sigma \notin A, \forall \sigma' \in \Sigma_{PE}, \sigma' \notin A \cup \{\sigma\}$$

$$(\langle A, \sigma \rangle, \sigma, \langle A, \sigma \rangle)$$

$$\forall A \subseteq \Sigma_t, \forall \sigma \in \Sigma_{PE}, \sigma \notin A$$

$$(\langle A, \sigma \rangle, \sigma', \langle A, \sigma' \rangle)$$

$$\forall A \subseteq \Sigma_t, \forall \sigma \in \Sigma_n, \forall \sigma' \in \Sigma_{PE}, \sigma' \notin A \cup \{\sigma\}$$

I_{PE}: $\{\text{start}\}$

F_{PE}: $Q_{PE} \setminus \{\text{start}\}$

Edges in the first set (first line for E_{PE}) are used to match the first symbol in the input sentence. Edges in the second set are followed whenever a new symbol is aligned to a new term or out-of-term word, and the previous was aligned to a term. The third set covers the case when a symbol is aligned to the same term or out-of-term word as the symbol just before it. Finally, the fourth set covers transitions such that the current symbol is aligned to a new term or out-of-term word, and the previous was aligned to an out-of-term word.

By making every non-start state final and by introducing a special initial state, we obtain the automaton that accepts all and only the alignments satis-

fying the contiguity constraint. In the example of Fig. 3, the alignment 1122143 is accepted, while 12324 is not, since 2 is repeated twice separated by a 3. Notice that constructing the contiguity enforcing automaton can be done once, in advance, for all possible combinations of $|\Sigma_t|$ and $|\Sigma_n|$ present in the corpus, renaming states for specific source sentences.

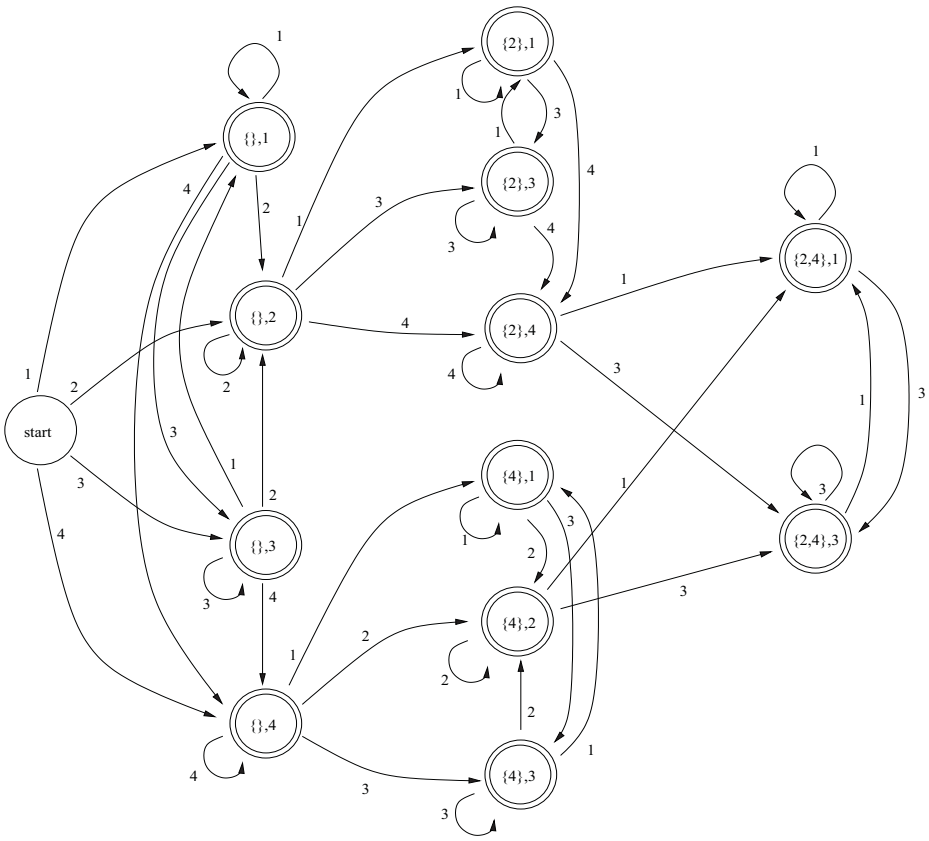

Fig. 3. Contiguity enforcing automaton for an English sentence with four states, two of which (e_2 and e_4) correspond to multi-word terms and two of which (e_1 and e_3) correspond to words occurring outside terms.

The composition of the contiguity enforcing automaton PE and the original WFST built from the HMM can be constructed directly by expanding every transition on symbol $\sigma \in \Sigma_{PE}$ in the contiguity enforcing automaton with $|\Sigma_F|$ transitions, one for each possible foreign word and assigning weights according to the appropriate transition and emission log-probabilities:

$$(\text{start}, \sigma, \langle \emptyset, \sigma \rangle)$$
$$\rightarrow (\text{start}, f : \sigma / -logP(f|e_\sigma) - logP(\sigma|0), \langle \emptyset, \sigma \rangle)$$
$$\forall f \in \Sigma_F, \forall \sigma \in \Sigma_{PE}$$

$$(\langle A, \sigma \rangle, \sigma', \langle A \cup \{\sigma\}, \sigma' \rangle)$$
$$\rightarrow (\langle A, \sigma \rangle, f : \sigma' / -logP(f|e_{\sigma'}) - logP(\sigma'|\sigma), \langle A \cup \{\sigma\}, \sigma' \rangle)$$
$$\forall A \subseteq \Sigma_t, \forall f \in \Sigma_F, \forall \sigma \in \Sigma_t, \sigma \notin A, \forall \sigma' \in \Sigma_{PE}, \sigma' \notin A \cup \{\sigma\}$$

$$(\langle A, \sigma \rangle, \sigma, \langle A, \sigma \rangle)$$
$$\rightarrow (\langle A, \sigma \rangle, f : \sigma / -logP(f|e_\sigma) - logP(\sigma|\sigma), \langle A, \sigma \rangle)$$
$$\forall A \subseteq \Sigma_t, \forall f \in \Sigma_F, \forall \sigma \in \Sigma_{PE}, \sigma \notin A$$

$$(\langle A, \sigma \rangle, \sigma', \langle A, \sigma' \rangle)$$
$$\rightarrow (\langle A, \sigma \rangle, f : \sigma' / -logP(f|e_{\sigma'}) - logP(\sigma'|\sigma), \langle A, \sigma' \rangle)$$
$$\forall A \subseteq \Sigma_t, \forall f \in \Sigma_F, \forall \sigma \in \Sigma_n, \forall \sigma' \in \Sigma_{PE}, \sigma' \notin A \cup \{\sigma\}$$

where Σ_F denotes the set of distinct words in the foreign sentence to be aligned.

Overall computational complexity is $O(I^2 2^{|\Sigma_t|} J)$. The result, again, is the best path of the directly constructed $PE' = W_{HMM} \diamond PE$.

3.1 Lazy contiguity enforcing

In the previous section, the whole contiguity enforcing automaton is composed *a priori* with the HMM-derived WFST before the best-path algorithm is run. While this time-consuming operation ensures that the selected alignment will respect the contiguity constraint, it is actually pointless in the cases where the best path in the original HMM-derived WFST does not violate the constraint. A variation of the previous method thus consists in first running a best-path algorithm on the initial WFST, checking whether the constraint is violated for any term and, if it is, compose the WFST with a reduced contiguity-enforcing automaton limited to the terms for which contiguity was violated, and iterate (Algorithm 1). It is easily verified that the automaton in Fig.4 enforces the contiguity constraint for the term $\sigma \in \Sigma_t$

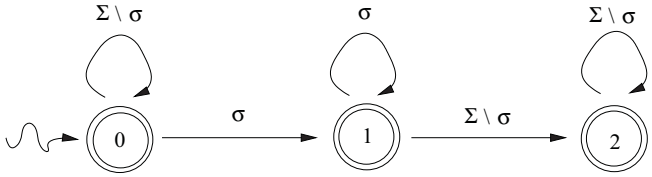

Fig. 4: The three-state automaton enforcing the constraint that whenever the state σ in the HMM-derived WFST is left, it is never entered again. For efficiency [10], every single arc with label $\Sigma \setminus \sigma$ represents a set of arcs, one for each $\sigma' \neq \sigma$.

Algorithm 1 The iterative algorithm that only composes the HMM-derived WFST with single-term contiguity checker for terms for which contiguity was violated.

$W \leftarrow W_{HMM}$
repeat
  $\pi \leftarrow$ Viterbi(F, W)
  $Z \leftarrow$ violate-contiguity(π, W)
  for all z in Z **do**
    $W \leftarrow W \diamond A_z$
until $Z = \emptyset$
return π

Each composition has the effect of doubling the number of states and edges. In the worst case, the algorithm is forced to compose again and again for enforcing contiguity one term at a time, and the finally resulting automaton is the same as what would be obtained by composing W_{HMM} directly with P. In this case there is no asymptotic penalty in the repeated composition itself, but we executed in vain $|\Sigma_t| - 1$ times the best-path algorithm, on WFSTs of exponentially increasing size. Notice, though, that the global asymptotic complexity for all repetitions of the best-path algorithm is the same as the complexity of the last iteration, because $\sum_{i=0}{}^{|\Sigma_t|} 2^i IJ = IJ(2^{|\Sigma_t|+1} - 1)$, and $\mathrm{O}(IJ(2^{|\Sigma_t|+1} - 1)) = \mathrm{O}(IJ2^{|\Sigma_t|})$. In other words, by performing composition lazily, we pay at most a constant factor in time, and nothing in space.

3.2 Local Reordering

While the permutation automaton is, in many cases, an appropriate solution, it has the drawback of growing exponentially in size with the number of source terms. The present section describes a method for enforcing the contiguity constraint which is not exponential in size in the length of the input and consists in a single best-search path on a multi-tape weighted finite-state transducer. The reduction in complexity is obtained at the price of allowing only *local reorderings* between the source and the target sequence. A permutation π is a local reordering with *jump length* m_j and *window size* m_w if and only if every element is at most m_j steps away from its position in the identity permutation (i.e. $|\pi_i - i| \leq m_j$) and every deviation from the identity permutation occurs within a subsequence of size at most m_w (i.e. the corresponding permutation matrix is block-diagonal with blocks of size at most m_w). For example, for maximal jump length $m_j = 2$ and window size $m_w = 3$ an acceptable permutation would be $(3, 2, 1, 4)$ but not $(3, 4, 1, 2)$. It is possible to write automatically automata to recognise sequences of jumps (i.e. $\pi_1 - 1, \pi_2 - 2, \ldots, \pi_I - I$) corresponding to local reorderings for arbitrary values of m_j and m_w. We will first describe how such a local reordering automata can be automatically generated, and then describe a method for compiling the original HMM into a new (multi-tape) WFST [8, 10] that represents the same probability distribution but outputs a sequence

of jumps as well as a sequence of visited states, and that can thus be joined with the local reordering automaton.

Algorithm 2 builds a local reordering automaton given the parameters m_j and m_w. Although the construction takes $O(m_w!m_w\log(m_w!m_w))$ time, where

Algorithm 2 *ReorderingAutomata*$(m_j, m_w)(R)$

```
Create state s_0, which will be the initial state and the only final state
Add a loop edge to s_0 labeled with 0
Generate the list Perm of all possible permutations of m_w many elements
Create empty list JumpSeq
for all π in Perm do
    tmp=""
    for all j in {1, 2, .., m_w} do
        if |π_j − j| ≤ m_j then
            Append π_j − j to tmp
    if length(tmp) = m_w then
        Add tmp to JumpSeq
for all i in JumpSeq do
    Strip leading and trailing 0s from i
    Create state s_{i,1}
    Add an arc going from s_0 to s_{i,1} labeled by i[1]
    for all j in {2, .., m_w − 1} do
        Create state s_{i,j}
        Add an arc going from s_{i,j−1} to s_{i,j} labeled by i[j]
    Add an arc going from s_{i,m_w−1} to s_0 labeled by i[m_w]
Minimize the automaton
```

complexity is dominated by the minimization operation, it needs be done once only for the desired (max. jump, window size) pair and does not need be repeated for each sentence. This algorithm was implemented and used in the preliminary experiments reported in Section 4. Figure 5 shows such an automaton. We note here that we have a more direct way of building the local reordering automaton, too.

In order to be able to use this automaton, we need to generate suitable input: the WFST built from HMM should output jumps between term positions instead of plain word/term identifiers like with the permutation automaton. Moreover, the reordering automaton only accepts (appropriately constrained) *permutations* of the state identifiers. There are two issues then that must be solved in order to use an approach based on the reordering automaton. The first is that we want to allow repeated visits to a same term, provided they are contiguous. The jump sequence will thus need to be generated from a term sequence from which contiguous repetitions are replaced by a single occurrence of a special term identifier. A second issue is that we might or might not be willing to accept the additional constraint that all English terms must be aligned to some foreign word. We will first present a solution for the case in which we assume that all

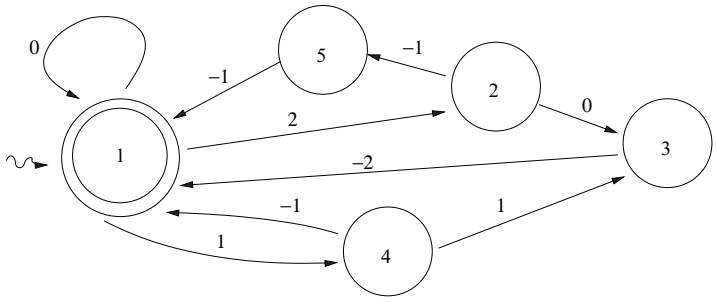

Fig. 5. The automaton produced by Algorithm 2 with parameters $m_j = 2$ and $m_w = 3$.

occurrences of terms on the English side have a counterpart on the foreign side, and then propose a modification that relaxes this assumption at the cost of introducing some complexity in the process.

Cascaded Automata with All English Terms Aligned In order to produce appropriate input for the reordering automaton, we add to the WFST derived from the original HMM an additional tape, besides the first reading the input sequence of foreign words and the second writing the sequence of English word/term identifiers corresponding to the alignment. A special term identifier ranging from 1 to the number of terms in the sentence (1 for e_2 and 2 for e_4 in the example) is written on the third tape every time a term-state is entered from a different state (Fig. 6). Formally:

$$H = (Q_H, \Sigma_H, E_H, I_H, F_H, A)$$

Q_H: $\{\text{start}\} \cup \{q_\sigma | \sigma \in \Sigma_H\}$
Σ_H : $\Sigma_{H1} \times \Sigma_{H2} \times \Sigma_{H3}$
Σ_{H1}: Σ_F
Σ_{H2}: $\{1, \ldots, I\}$ $\Sigma_{H2} = \Sigma_t \cup \Sigma_n$, $\Sigma_t \cap \Sigma_n = \emptyset$
Σ_{H3}: $\{1, \ldots, |\Sigma_t|\}$
E_H: $(\text{start}, f : \sigma : \epsilon, q_\sigma)/ - logP(f|e_\sigma) - logP(\sigma|0)$
$\forall f \in \Sigma_F, \forall \sigma \in \Sigma_n$
$(\text{start}, f : \sigma : t_\sigma, q_\sigma)/ - logP(f|e_\sigma) - logP(\sigma|0)$
$\forall f \in \Sigma_F, \forall \sigma \in \Sigma_t$
$(q_\sigma, f : \sigma' : \epsilon, q_{\sigma'})/ - logP(f|e_{\sigma'}) - logP(\sigma'|\sigma)$
$\forall f \in \Sigma_F, \forall \sigma \in \Sigma_{H2}, \forall \sigma' \in \Sigma_n$
$(q_\sigma, f : \sigma' : t_{\sigma'}, q_{\sigma'})/ - logP(f|e_{\sigma'}) - logP(\sigma'|\sigma)$
$\forall f \in \Sigma_F, \forall \sigma \in \Sigma_{H2}, \forall \sigma' \in \Sigma_t, \sigma \neq \sigma'$
$(q_\sigma, f : \sigma : \epsilon, q_\sigma)/ - logP(f|e_\sigma) - logP(\sigma|\sigma)$
$\forall f \in \Sigma_F, \forall \sigma \in \Sigma_t$
I_H: $\{\text{start}\}$
F_H: $Q_H \setminus \{\text{start}\}$

where $t_\sigma, \sigma \in \Sigma_t$ is a special term identifier in $\{1, \ldots, |\Sigma_t|\}$ and $A = \langle \mathbb{R} \cup \{+\infty\}, min, +, 0, +\infty \rangle$ is the tropical semiring.

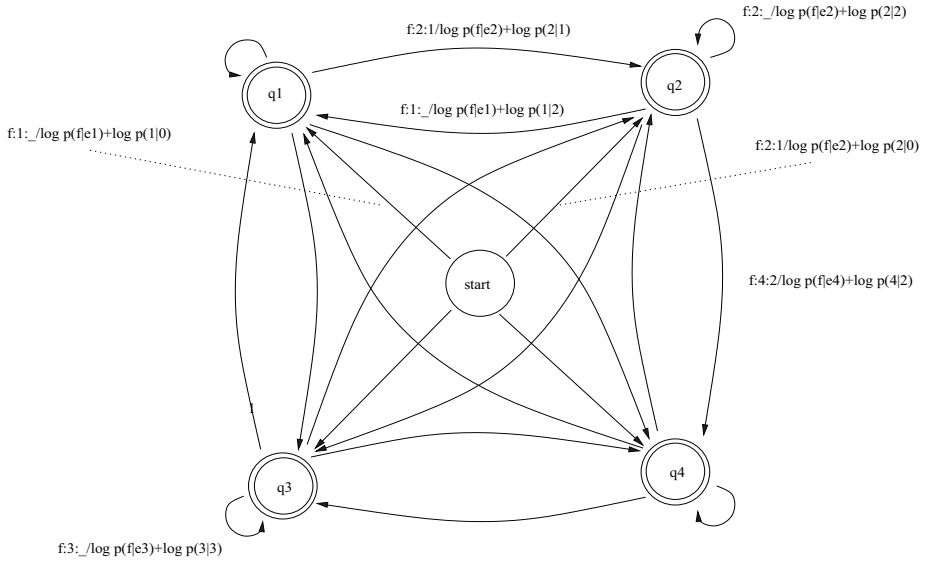

Fig. 6. An example of a 3-tape WFST encoding an HMM. e_2 and e_4 are terms, e_1 and e_3 are out-of-term words. Only one transition for every state pair is shown, while in the actual automaton there is one for every word f in the foreign sentence. Also, only some labels/weights are shown.

We introduced two simple auxiliary finite-state machines: the *identical permutation* automaton I and the *difference* automaton D: the former just encodes an increasing integer sequence $1, 2, \ldots, |\Sigma_t|$, while the latter has three tapes and puts on the third the difference of the first two.

By appropriately joining[6] the transducers defined so far we obtain a transducer accepting only alignments corresponding to local reorderings of the English terms,

$$G = ((H \bowtie_{\{3=1\}} D) \bowtie_{\{4=1\}} I) \bowtie_{\{5=1\}} R, \tag{3}$$

where the result G has five tapes: $H_1 : H_2 : H_3 = D_1 : D_2 = I_1 : D_3 = R_1$. The best alignment satisfying the contiguity constraint and representing a local reordering (with parameters m_j and m_w) is then simply obtained through a best-path search on G, on the tape G_2.

Cascaded Automata with Empty Alignments for Some English Terms The method described above imposes that all English terms are aligned. If we do

[6] Algorithms for joining and composing automata are described in [9].

not want to impose this additional constraint, we can still enforce the contiguity constraint with local reorderings, but at the price of making the method more complicated.

The reordering automaton recognises the language of all jump sequences corresponding to local reordering permutations. We generated the jump sequence by computing a symbol-wise difference between the sequence of term identifiers in the order they were entered and their identical permutations. We could do that simply through the D transducer because these two sequences have the same length. If now we want to relax the constraint that all English terms are aligned we need a way to "visit" all states of the HMM other than by aligning a foreign word to it. We can do this by introducing special transitions going into term states that do not "use" any input symbol. When assessing the probability of an alignment, this will alter the contribution of the transition probabilities. To account for this effect, we will thus create two separate WFSTs from the original HMM: one (E) will account for the emission probabilities only, and one will account for the transition probabilities (T). The former will accept the input sequence and will output on one of its tapes a sequence (with no contiguous repetitions) of identifiers of those visited states which are aligned to real foreign words only:

$$E = (Q_E, \Sigma_E, E_E, I_E, F_E, A)$$

Q_E: $\{\text{start}\} \cup \{q_\sigma | \sigma \in \Sigma_E\}$
Σ_E : $\Sigma_{E1} \times \Sigma_{E2} \times \Sigma_{E3}$
Σ_{E1}: Σ_F
Σ_{E2}: $\{1, \ldots, I\}$
$\Sigma_{E2} = \Sigma_t \cup \Sigma_n$, $\Sigma_t \cap \Sigma_n = \emptyset$
Σ_{E3}: $\{1, \ldots, |\Sigma_t|\}$
E_E: $(\text{start}, f : \sigma : \epsilon, q_\sigma)/ - logP(f|e_\sigma)$
$\forall f \in \Sigma_F, \forall \sigma \in \Sigma_n$
$(\text{start}, f : \sigma : t_\sigma, q_\sigma)/ - logP(f|e_\sigma)$
$\forall f \in \Sigma_F, \forall \sigma \in \Sigma_t$
$(\text{start}, \epsilon : \epsilon : t_\sigma, q_\sigma)/0$
$\forall \sigma \in \Sigma_t$
$(q_\sigma, f : \sigma' : \epsilon, q_{\sigma'})/ - logP(f|e_{\sigma'})$
$\forall f \in \Sigma_F, \forall \sigma \in \Sigma_{E2}, \forall \sigma' \in \Sigma_n$
$(q_\sigma, f : \sigma' : t_{\sigma'}, q_{\sigma'})/ - logP(f|e_{\sigma'})$
$\forall f \in \Sigma_F, \forall \sigma \in \Sigma_{E2}, \forall \sigma' \in \Sigma_t, \sigma \neq \sigma'$
$(q_\sigma, f : \sigma : \epsilon, q_\sigma)/ - logP(f|e_\sigma)$
$\forall f \in \Sigma_F, \forall \sigma \in \Sigma_t$
$(q_\sigma, \epsilon : \epsilon : t_\sigma, q_\sigma)/0$
$\forall \sigma \in \Sigma_t$
I_E: $\{\text{start}\}$
F_E: $Q_E \setminus \{\text{start}\}$

where $t_\sigma, \sigma \in \Sigma_t$ is a special term identifier in $\{1, \ldots, |\Sigma_t|\}$ and $A = \langle \mathbb{R} \cup \{+\infty\}, min, +, 0, +\infty \rangle$ is the tropical semiring.

The latter will compute the contribution of transition probabilities based on this state sequence:

$$T = (Q_T, \Sigma_T, E_T, I_T, F_T, A)$$

Q_T: $\{\text{start}\} \cup \{q_\sigma | \sigma \in \Sigma_T\}$
Σ_T : $\{1, \ldots, I\}$
E_T: $(\text{start}, \sigma, q_\sigma)/ - logP(\sigma|0)\ \forall \sigma \in \Sigma_T$
$(q_\sigma, \sigma', q_{\sigma'})/ - logP(\sigma'|\sigma)\ \forall \sigma, \sigma' \in \Sigma_T$
I_T: $\{\text{start}\}$
F_T: $Q_T \setminus \{\text{start}\}$

The result of joining these transducers with the D, I and R transducers (see before) is a transducer accepting only alignments corresponding to local reorderings of the English terms where some English terms can remain unaligned:

$$G' = (((E \bowtie_{\{2=1\}} T) \bowtie_{\{3=1\}} D) \bowtie_{\{4=1\}} I) \bowtie_{\{5=1\}} R \tag{4}$$

where the result G' has five tapes: $E_1 : E_2 = T1 : E_3 = D_1 : D_2 = I_1 : D_3 = R_1$. The best local reordering alignment (with m_j and m_w) satisfying the contiguity constraint is then obtained through a best-path search on G', on the tape G'_2.

4 Experiments

Some experiments applying the described word-alignment methods to multilingual terminology extraction were performed. 576 sentence pairs coming from Xerox manuals in English-German were annotated with 897 term boundaries and alignments by a native speaker of German. As a base for our projection, we first took manually identified English terms, then NPs extracted using patterns of parts-of-speech. For both cases, we aligned German words to English terms and out-of-term words according to some early variants of the methods described above:

- SELECT: lazy enforcement of the contiguity constraint to terms only;
- REORD: local reordering method with $m_j = 2$ and $m_w = 3$
- SYMMETRIC: the standard symmetric method by which term candidates are first identified in German using POS patterns and are then aligned, but without a heuristic that greedily extends term boundaries to improve boundary detection.

For each method we measured the number of exact matches, together with precision, recall and F-score at the word level according to the following definitions:

$$P = \frac{\sum_{i=1}^{n} p_i}{n}; \quad p_i = \frac{a_i}{c_i}; \quad R = \frac{\sum_{i=1}^{n} r_i}{n}; \quad r_i = \frac{a_i}{w_i}; \quad F = \frac{2PR}{P+R}$$

where c_i is the number of words identified as part of terms in sentence i, w_i is the number of words in terms in the reference for sentence i, and a_i is the number of correctly aligned in-term tokens in sentence i.

Test results are in table 1, where terms are all German gold standard terms for which a translation was provided or not in the candidate sentences.

method	**manual annotation**			
	exact matches	**P**	**R**	**F**
SELECT	291 (32.44%)	86.39%	64.51%	73.87%
REORD	258 (28.76%)	83.65%	67.83%	74.91%
SYMMETRIC	282 (31.44%)	97.41%	98.12%	97.76%
method	**automated annotation**			
	exact matches	**P**	**R**	**F**
SELECT	275 (30.66%)	86.69%	67.07%	75.63%
REORD	235 (26.20%)	83.38%	70.53%	76.42%
SYMMETRIC	205 (22.85%)	89.93%	96.04%	92.88%

Table 1. Preliminary experiment results.Total number of terms: 897, total number of sentences: 576.

From such preliminary experiments, it would seem that in the automated case, performance is comparable with those obtained with the symmetric method, recall is smaller, more resource-intensive, although this is somewhat underestimated due to the fact that an effective recall-oriented heuristic (although possibly precision-degrading) was not used.

We aligned 576 sentences in 6 hours when enforcing local reordering contiguity constraint and in 10 minutes for the selective lazy one. The standard symmetric alignment is much faster.

After these experiments, a Norwegian-English terminology containing 4211 candidate term pairs was extracted from a parallel corpus of 38487 sentence pairs from Xerox manuals using an early version of the lazy contiguity enforcement method.

5 Summary

We have shown that our method can be used to extract bilingual terminologies in cases when neither a morphological analyzer nor a POS tagger are available for the target language. This new result is based on a powerful probabilistic model that explicitly models distortion in alignments and ensures that the produced

alignment is optimal respecting the contiguity constraint according to the probabilistic model. The statistical model is advantageously exploited by the use of weighted multi-tape calculus. Test results confirm the claimed advantages.

Extensive previous literature on the problem of word alignment, from [17] then [7, 19] and [20, 21, 18] to [22] either does not cover the contiguity issue at all or can not enforce it as a constraint.

As far as bilingual terminology from parallel corpora is concerned, most proposed methods [1–6] rely on target-source matching sequences of Part-of-Speech requiring reliable POS taggers for both. We need only one, on the source side. [23] is asymmetric but can not guarantee optimality with respect to the underlying model [7].

Acknowledgments

This work was supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778. This publication only reflects the authors' views.

References

1. van der Eijk, P.: Automating the acquisition of bilingual terminology. In: Proceedings of the sixth conference on European chapter of the Association for Computational Linguistics, Morristown, NJ, USA, Association for Computational Linguistics (1993) 113–119
2. Dagan, I., Church, K.: Termight: identifying and translating technical terminology. In: Proceedings of the fourth conference on Applied natural language processing, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc. (1994) 34–40
3. Gaussier, É., Hull, D., Ait-Mokhtar, S.: Term alignment in use: Machine-aided human translation. In Veronis, J., ed.: Parallel text processing: Alignment and use of translation corpora. Kluwer Academic Publishers (2000) 253–274
4. Déjean, H., Gaussier, E., Sadat, F.: An approach based on multilingual thesauri and model combination for bilingual lexicon extraction. In: Proceedings of the 19th international conference on Computational linguistics, Taipei, Taiwan (2002)
5. Hull, D.: (2005) US patent US6885985.
6. Hull, D.: Automating the construction of bilingual terminology lexicons. Terminology (1997)
7. Vogel, S., Ney, H., Tillmann, C.: HMM-based word alignment in statistical translation. In: COLING 96, Copenhagen (1996)
8. Kempe, A., Baeijs, C., Gaál, T., Guingne, F., Nicart, F.: WFSC – a new weighted finite state compiler. In Dang, Z., Ibarra, O.H., eds.: Proc. CIAA 2003. Volume 2759 of LNCS., Heidelberg, Springer-Verlag (2003) 108–119
9. Kempe, A., Champarnaud, J.M., Guigne, F., Nicart, F.: Wfsm auto-intersection and join algorithms. In: FSMNLP 2005, Helsinki, Finland (2005)
10. Nicart, F., Champarnaud, J.M., Csáki, T., Gaál, T., Kempe, A.: Multitape automata with symbol classes. In Ibarra, O.H., Yen, H.C., eds.: Proc. CIAA 2006. Volume 4094 of LNCS., Berlin, Heidelberg, Springer-Verlag (2006) 126–136

[7] The method in [23] is protected by IBM patent US6236958.

11. Gale, W.A., Church, K.W., Yarowsky, D.: Using bilingual materials to develop word sense disambiguation methods. In: Fourth International Conference on Theoretical and Methodological Issues in Machine Translation, Montreal (1992)
12. Simard, M., Plamondon, P.: Bilingual sentence alignment: Balancing robustness and accuracy. Machine Translation **13**(11) (March 1998)
13. Aswani, N., Gaizauskas, R.: A hybrid approach to align sentences and words in English-Hindi parallel corpora. In: ACL workshop on Building and Using Parallel Text. (2005)
14. Kumar Singh, A., Husain, S.: Comparison, selection and use of sentence alignment algorithms for new language pairs. In: ACL workshop on Building and Using Parallel Text. (2005)
15. Jacquemin, C., Bourigault, D.: Term extraction and automatic indexing. In Mitkov, R., ed.: Handbook of Computational Linguistics. Oxford University Press (2000)
16. Och, F.J., Ney, H.: A systematic comparison of various statistical alignment models. Computational Linguistics **29**(1) (2003) 19–51
17. Brown, P.F., Pietra, V.J.D., Pietra, S.A.D., Mercer, R.L.: The mathematics of statistical machine translation: parameter estimation. Comput. Linguist. **19**(2) (1993) 263–311
18. Kumar, S., Byrne, W.: A weighted finite state transducer implementation of the alignment template model for statistical machine translation. In: HLT-NAACL 2003, Edmonton, Canada (2003)
19. Moore, R.: A discriminative framework for bilingual word alignment. In: Proc. of HLT/EMNLP 2005, Vancouver, BC, Canada (2005)
20. Och, F., Tillmann, C., Ney, H.: Improved alignment models for statistical machine translation. In: Proc. of EMNLP/VLC 1999, University of Maryland, College Park, MD, USA (1999)
21. Koehn, P., Och, F., Marcu, D.: Statistical phrase-based translation. In: Proc. of HLT-NAACL 2003, Edmonton, Canada (2003)
22. Goutte, C., Yamada, K., Gaussier, E.: Aligning words using matrix factorisation. In: Proc. of ACL'04., Barcelona, Spain (2004)
23. Gaussier, E.: Flow network models for word alignment and terminology extraction from bilingual corpora. In Boitet, C., Whitelock, P., eds.: Proceedings of the Thirty-Sixth Annual Meeting of the Association for Computational Linguistics and Seventeenth International Conference on Computational Linguistics, San Francisco, California, Morgan Kaufmann Publishers (1998) 444–450