Disambiguation Strategies for Data-Oriented Translation

Mary Hearne and Andy Way*

National Centre for Language Technology, School of Computing, DCU, Dublin, Ireland

{mhearne | away}@computing.dcu.ie

Abstract

The Data-Oriented Translation (DOT) model – originally proposed in (Poutsma, 1998, 2003) and based on Data-Oriented Parsing (DOP) (e.g. (Bod, Scha, & Sima'an, 2003)) – is best described as a hybrid model of translation as it combines examples, linguistic information and a statistical translation model. Although theoretically interesting, it inherits the computational complexity associated with DOP. In this paper, we focus on one computational challenge for this model: efficiently selecting the 'best' translation to output. We present four different disambiguation strategies in terms of how they are implemented in our DOT system, along with experiments which investigate how they compare in terms of accuracy and efficiency.

1 Introduction

The merits of combining the positive elements of the rule-based and data-driven approaches to MT are clear: a combined model has the potential to be highly accurate, robust, cost-effective to build and adaptable to different domains. Nevertheless, how best to combine these techniques into a model which retains the positive characteristics of each approach, while inheriting as few of the disadvantages as possible, remains a challenging problem. One possible solution is the Data-Oriented Translation (DOT) model originally proposed in (Poutsma, 1998, 2003), which is based on Data-Oriented Parsing (DOP) (e.g. (Bod et al., 2003)) and combines examples, linguistic information and a statistical translation model.

Although DOT embodies many positive characteristics on a theoretical level, it also inherits the computational complexity associated with DOP. In this paper, we focus on one of the computational challenges: efficiently selecting the 'best' translation to output. The DOT model calls for ranking of the output translations according to translation probability over a DOT grammar. However, this is analogous to the problem of finding the most probable parse over a DOP grammar, which has been shown to be an NP-hard problem (Sima'an, 1996). As the exact solution to this problem cannot be found in an efficient way, we must either find a way of approximating the search for the most probable translation such that we do not perform an exhaustive search of the space of possible derivations, or we must choose a different criterion to maximise.

In this paper, we consider both of these possibilities. We use random sampling to compute the most probable translation (MPT) for each input string without having to look at every derivation, and thus output the MPT as the best translation. However, algorithms which approximate an NP-hard search problem are generally not adopted if a deterministic alternative can be found which does not introduce an unacceptable degradation in performance. Thus, we also disambiguate by selecting for output the translation yielded by the most probable representation (or bilingual parse) (MPP),

*This work was supported by Science Foundation Ireland grant 05/IN/1732.

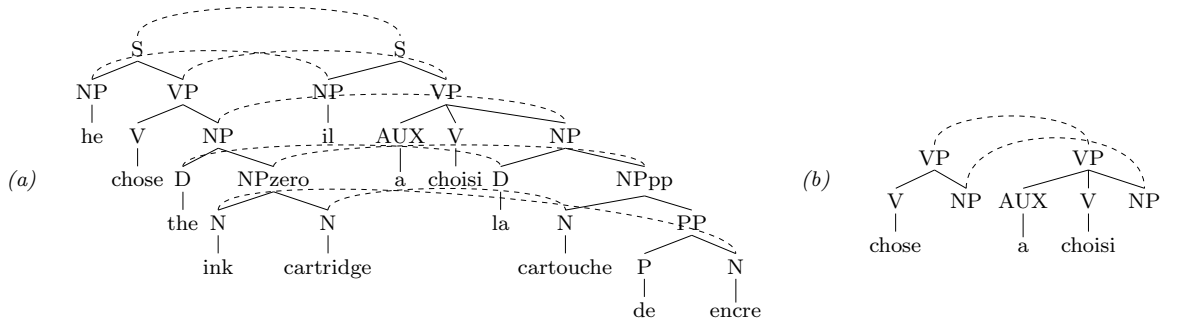

Figure 1: *(a)* an example DOT representation and *(b)* an example DOT fragment extracted from *(a)*.

the most probable derivation (MPD) and the shortest derivation (SDER), i.e. the derivation using the fewest fragments. We present details of how these four disambiguation strategies are implemented in our DOT system, along with experiments which look at how they compare in terms of both translation accuracy and efficiency.

The paper is structured as follows. In section 2 we present the DOT model which assumes context-free phrase structure tree representations. In section 3 we present the four disambiguation strategies under investigation, focussing on the algorithms used to implement them. In sections 4 and 5 we describe our experiments, and present and discuss the results achieved. Finally, section 6 gives some avenues for future work.

2 The DOT Model

Providing a specification of the DOT model means specifying four elements: the type of representation we expect to find in the example base, how fragments are to be extracted from those representations, how extracted fragments are to be recombined when analysing and translating new input strings, and how the resulting translations are to be ranked. The model described here follows (Poutsma, 2003).

Representations Many different linguistic formalisms can be used to annotate the example base which underpins any DOT model; here, we assume context-free phrase structure tree representations. Representations for this model comprise *pairs* of trees, i.e. we assume a bilingual aligned treebank such that each tree pair constitutes an example translation pair. We also assume that links denoting translational equivalence are present in each example: node A_x in the source tree of the example and node B_y in the corresponding target tree are linked if the substrings they dominate can be considered translations of each other. An example representation is given in Figure 1*(a)*. Note that the source node V is unlinked despite the fact that *chose* corresponds to *a choisi*; in this case there is no single node dominating *a choisi* to which V can be linked. Note also that the target node P is unlinked; this is because *de* has no overt realisation in the source string. Thus, a minimally-linked tree pair will be linked *only* at sentence level – this is the case for sentence-idioms.

Fragmentation The fragmentation process involves extracting pairs of linked generalised subtrees from the linked tree pairs contained in the example base via the *root* and *frontier* operations, which for Tree-DOT are defined as follows:

- given a copy of tree pair $<S,T>$ called $<S_c,T_c>$, select a **linked** node pair $<S_N,T_N>$ in $<S_c,T_c>$ to be *root* nodes and delete all except these nodes, the subtrees they dominate and the links between them, and
- select a set of **linked** node pairs in $<S_c,T_c>$ to be *frontier* nodes and delete the subtrees they dominate.

Thus, every fragment $<f_s,f_t>$ is extracted such that the root nodes of f_s and f_t are linked, and every non-terminal frontier node in f_s is linked to exactly one non-terminal frontier node in f_t and vice versa. The fragment in Figure 1*(b)* was extracted from the tree in Figure 1*(a)* as follows: the node pair $<VP,VP>$ was selected by the root operation and the set of linked nodes $\{<NP,NP>\}$ was selected by the frontier

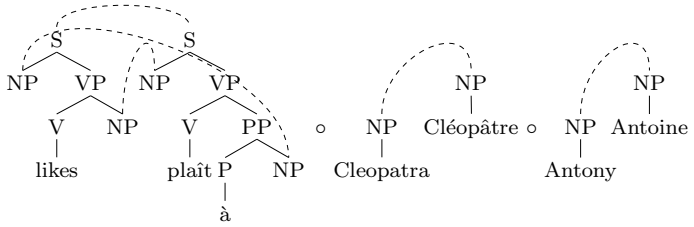

Figure 2: An example DOT composition sequence.

operation.

Composition The Tree-DOT composition operation (∘) is a leftmost substitution operation: where a fragment has more than one open substitution site, composition must take place at the leftmost site on the source subtree of the fragment. Furthermore, the synchronous target substitution must take place at the site *linked to* the leftmost open source substitution site. This ensures (i) that each derivation is unique and (ii) that each translation built adheres to the translational equivalences encoded in the example base. We can illustrate both these issues using the composition sequence given in Figure 2.

If it were not the case that composition must take place at the leftmost site on the source subtree of the fragment, the composition sequence in Figure 2 could yield either *Cleopatra likes Antony* or *Antony likes Cleopatra*. However, the leftmost substitution restriction means that it can yield only *Cleopatra likes Antony* – in order to get *Antony likes Cleopatra*, we must use a different composition sequence (where the order of the last two fragments is swapped).

If it we allowed composition to take place at the leftmost site on the target tree also, then the target language string corresponding to the composition sequence in Figure 2 would be *Cléopâtre plaît à Antoine*, which is a semantically incorrect translation for *Cleopatra likes Antony*.[1] However, because we specify that target substitution must take place at the site linked to the leftmost open source substitution site, the correct translation – where the source subject *Cleopatra* translates as the target prepositional object *Cléopâtre*, and the source object as the target subject – is generated.

Tree-DOT derivations are built by simultaneously building source and target representations using the composition operation. Once an initial fragment is chosen to start the derivation, further fragments are successively substituted at the leftmost source open substitution site and its linked target counterpart until no open substitution sites remain. The output translation associated with each derivation is extracted by simply concatenating the frontier nodes of the target tree.

Computation of Probabilities The probability of a fragment is its relative frequency in the set of fragments.[2] The relative frequency of a fragment is computed by dividing the frequency of the fragment by the sum of the frequencies of all fragments with the same source and target root nodes $(r(s), r(t))$ as it, as in equation (1).

$$P(\langle s_x, t_x \rangle) = \frac{n(\langle s_x, t_x \rangle)}{\sum_{r(s)=r(s_x) \wedge r(t)=r(t_x)} n(\langle s, t \rangle)} \quad (1)$$

The probability of each derivation is the product of the probabilities of the fragments used to build that derivation as given in equation (2).

$$P(D_x) = \prod_{\langle s_x, t_x \rangle \in D_x} P(\langle s_x, t_x \rangle) \quad (2)$$

The probability of a representation (i.e. a pair of source and target trees) is the sum of the probabilities of the derivations which

[1] This is a relation-changing case, and *Cléopâtre plaît à Antoine* actually means *Antony likes Cleopatra*.

[2] Estimating fragment probabilities according to their relative frequencies is known to be undesirable for DOP. (Hearne, 2005) discusses the ramifications of using this method for DOT: the negative impact on accuracy is less than for DOP, but improved estimation methods (e.g. (Sima'an & Buratto, 2003)) are likely to improve translation quality once adapted to the bilingual case.

yield that representation as given in equation (3).

$$P(\langle S_x, T_x \rangle) = \sum_{D_x \ yields \ \langle S_x, T_x \rangle} P(D_x) \quad (3)$$

Finally, the probability that the source string s translates as the target string t is the sum of the probabilities of the representations which yield both s and t, as given in equation (4).

$$P(s,t) = \sum_{\langle S_x, T_x \rangle \ yields \ s,t} P(\langle S_x, T_x \rangle) \quad (4)$$

3 Disambiguation Strategies

The four disambiguation strategies we investigate are summarised as follows:

MPT: the most probable sequence of target terminals given the input string;

MPP: the sequence of target terminals read from the most probable bilingual representation for the input string;

MPD: the sequence of target terminals read from the most probable derivation of a bilingual representation for the input string;

SDER: the sequence of target terminals read from the shortest derivation of a bilingual representation for the input string.

Finding the MPT and MPP requires the use of random sampling, whereas the MPD and SDER can be found using the Viterbi algorithm. In this section, we present details of how these four disambiguation strategies are implemented in our DOT system. First, however, we must describe how the set of fragments relevant to a given input string is retrieved from the grammar.

3.1 Computing the Translation Space

(Sima'an, 1995) presents a two-phase analysis approach to building the parse space for a given input string and DOP grammar. Firstly, the context-free grammar underlying the fragment set is used to approximate the parse space of the input string. Correspondences between these CFG rules and the fragments in which they occur facilitate the transition from this CFG parse space to the required DOP parse space for the input. This algorithm can also be applied to the computation of the DOT translation space.

Each DOT fragment is associated with a unique identifier. The CFG underlying the source side of the fragment set is extracted such that each rule in the CFG is associated with the set of fragment identifiers in which it occurs. The first phase of the algorithm generates a monolingual parse space comprising the CFG rules which can be used to parse the input string. The second phase then generates from this, bottom-up, the set of (bilingual) fragments which can be used to build representations (and, therefore, translations) for the input string.

Figure 3 gives the DOT translation space for the string *anthony likes cleopatra* over some DOT grammar containing at least those fragments which appear in the translation space. The notion of translation space corresponds directly to the notion of parse space (or parse chart or parse forest) from chart-parsing. The translation space is a two-dimensional chart of size N^2 where N is the length of the input string. Each token in the input string is assigned a number i such that $1 \leq i \leq N$. These numbers appear along the horizontal axis; the numbers which appear on the vertical axis (generally represented by j) indicate the number of input tokens spanned. Thus, fragment f_x appearing in chart position $[i][j]$ signifies that derivations of one or more (bilingual) trees representing the portion of the input string which starts with token i and spans j consecutive tokens can be started with fragment f_x. Each fragment's frontiers can consist of non-terminal and/or terminal symbols. Each non-terminal frontier node of any fragment present on the chart points to the chart position from which fragments which can be combined with it must be selected.[3]

[3] In Figure 3, pointers $[i][j]$ are shown on the source nodes only; each target frontier pointer corresponds to the pointer of the source frontier non-terminal to which it is linked.

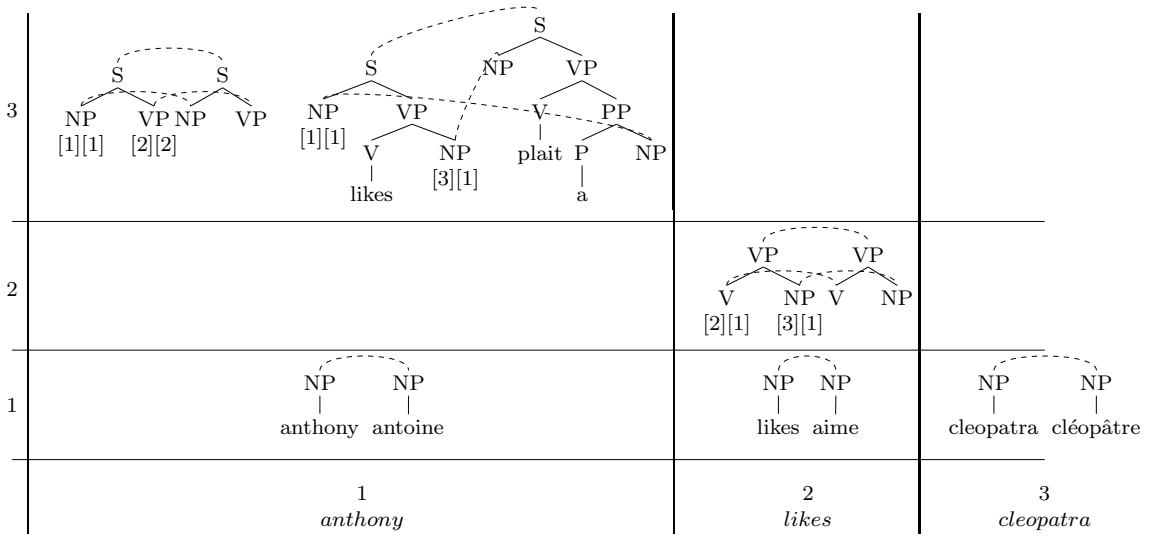

Figure 3: The DOT translation space for the sentence *Anthony likes Cleopatra*.

3.2 Finding the MPT and MPP using Monte Carlo

Monte Carlo sampling can be used to estimate the MPT by ranking the possible translations according to how often each one occurs in a reduced random sample of the possible derivations. This approach to disambiguation was introduced for DOP in (Bod, 1998) and further expanded on and refined in (Chappelier & Rajman, 2003); application of the algorithms proposed by Chappelier and Rajman (*op. cit.*) to translation was presented in (Hearne, 2005). The sampling methodology itself is very simple: in order to sample a derivation, we select and compose fragments at random from the translation space in a top-down left-to-right manner until no open substitution sites remain. However, we must select fragments at random such that if the DOP probability of fragment f_x is n times that of f_y, then f_x is n times more likely to be chosen during random selection than f_y. The main issue, therefore, is to correctly define the sampling probability of each fragment at each chart position $SP(f_{ij})$ such that the distribution of translations in the sample set converges to the true P_{DOT}.

The sampling probability used when selecting fragments can be defined in advance very easily (Bod, 1998). If we do so, however, then we cannot be certain that the distribution of the sample set will converge to give the DOT probability for each translation. The correct values must instead be obtained by *rescoring* the relative frequencies of the translations in the sample set when sampling is complete (Hoogweg, 2000; Chappelier & Rajman, 2003). Here, we apply *exact* sampling (Chappelier & Rajman, 2003), the purpose of which is to ensure that the sampling probability of each translation is directly equal (without rescoring) to the conditional DOT probability of that translation given the input string. Thus, the sampling probability of fragment f_{ij} is equal to its DOT probability multiplied by the total sampling probability mass available at each of its substitution sites, and divided by the total sampling probability mass available at position $[i][j]$.

The other issue to be addressed when using random sampling is to decide when enough samples have been seen: in order to statistically control the size of the sample set, we must determine the minimum number of samples needed to be certain that the most frequent translation in the sample set corresponds to the most probable translation according to the DOT model. The solution we apply is Bechhofer-Kiefer-Sobel (BKS) sampling, adapted for translation from (Chappelier & Rajman, 2003) in (Hearne, 2005). BKS is a sequential sampling method, meaning that we continue to sample derivations at random until we fulfil a stopping condition which is predefined but recalculated each time a sample is taken. The decision to stop sampling is based on three factors: (i) how closely matched, in

terms of frequency of occurrence, the translations in the sample set are, (ii) how many of the possible translations for the given input string are present in the set of sampled translations, and (iii) how certain we wish to be that the most frequent translation in the sample set is in fact the most probable translation according to the DOT model. BKS relies on the following: for any input sentence S with k translations ($< t_{[1]}...t_{[k]} >$) such that $n_{[1]} \geq \theta n_{[2]}$ with $\theta > 1$, the probability that the most frequent translation in the sample is also the most probable one is always greater than $\frac{1}{1+Z}$ where $Z = \sum_{i=2}^{k} (\frac{1}{\theta})^{(n_{[1]} - n_{[i]})}$ and where $n_{[i]}$ is the number of occurrences of the translation in ith position on the ordered list (decreasing order) of translations seen. The BKS method is then:

- choose values for $\theta = \frac{n_{[1]}}{n_{[i]}}$ and the error probability P_{err},
- sample (updating the ordered list of translations and their frequencies, and Z) until $\frac{1}{1+Z} \geq P_{err}$,
- output the most frequent translation in the sample as the most probable one.

We can also use random sampling to disambiguate by selecting for output the translation yielded by the most probable representation (MPP). In order to find this representation, we sample derivations according to their DOT sampling probabilities as just described. However, in this case our sample set comprises representations – i.e. linked source and target tree pairs – rather than translations (as for DOT) or parses (as for DOP) and our stopping conditions are altered to reflect this.

3.3 Finding the MPD and SDER using Viterbi

There exists an efficient algorithm, the Viterbi algorithm, to compute the MPD over a PCFG. This algorithm can also be used to compute the MPD over a DOT grammar. It involves pruning sub-derivations with low probabilities from the translation space in a bottom-up manner. Two different sub-derivations which have the same root node pair and span the same portion of the input string are used in building derivations for the entire input string in exactly the same way. This means that derivations containing the more probable of these sub-derivations will always have greater probability than those derivations containing the less probable sub-derivation. Consequently, the less probable sub-derivation will never be used to build the most probable derivation and can be removed from the parse space. This algorithm is integrated into the second phase of the procedure used to build the translation space described in section 3.1: as the sets of fragments which are relevant to the input string at each position $[i][j]$ in the translation space are computed, only the fragment starting the highest-scoring sub-derivation is retained.

Although the search for SDER does not involve actually estimating probabilities, the Viterbi algorithm can nevertheless be used (Bod, 2000). Derivation lengths are computed by assigning each fragment equal probability, meaning that the shortest derivation can be computed as the most probable one using Viterbi: if each fragment has probability p, then the probability of a derivation which uses n fragments is p^n and, since $0 < p < 1$, the smallest n must have the largest probability.

4 Experiments

We present bidirectional DOT experiments on the English-French section of the Home-Centre corpus, which contains 810 parsed, sentence-aligned translation pairs. This corpus comprises a Xerox printer manual, which was was translated by professional translators and sentence-aligned and annotated at Xerox Parc. As one would expect, the translations it contains are of extremely high quality. As observed in (Frank, 1999), the corpus provides a rich source of both linguistic and translational complexity. While English and French are syntactically quite similar, they often differ significantly in the surface styles used to express the same concept, and translational divergences which generally prove challenging for MT models (e.g. nominalisation, head-switching, lexical divergence, stylistic divergence, etc.) are

very much in evidence in this dataset.

We preprocessed the parses by removing unary-branching structures. We combined groups of sentence representations forming a single translation unit into a single phrase-structure tree by simply inserting a root node PAIR such that each tree is a child of that pair. We manually inserted the translational links between paired trees; each English-French tree pair was linked only at the root node but DOT also requires links indicating translational equivalences at sub-structural level.[4] Finally, our dataset was split randomly into 12 training/test splits such that all test words also appeared in the training set. Each of these splits comprises 80 test sentences and 730 training tree pairs; 6 of the splits have English as the source language and French as the target language and the other 6 splits have French as source and English as target.

We translate each test sentence using the four ranking strategies – MPT, MPP,[5] MPD and SDER) – as described in section 3. We prune the fragment base extracted from each training set with respect to link depth (Hearne & Way, 2003), namely the greatest number of steps taken *which depart from a linked node* to get from the root node to any frontier node.[6] This yields fragment bases comprising fragments of link depth 1, link depth 2 or less, link depth 3 or less and link depth 4 or less – the corresponding grammar sizes are given in Table 1. In the interests of robustness, we handle input sentences not covered fully by the grammar by assigning to them the best sequence of partial analyses according to the relevant ranking strategy, leaving untranslated words in the output string where necessary. We evaluate using three different automatic translation evaluation metrics: exact match, BLEU and F-score.[7]

[4]An algorithm to accomplish this task automatically, which gives encouraging preliminary results, is described in (Groves, Hearne, & Way, 2004).

[5]When computing MPT and MPP, we set the sampling thresholds P_{err} and θ given in section 3.2 to 0.01 and 2 respectively (determined empirically), and the maximum number of samples to 10,000.

[6]For example, the link depths of the representations in Figure 1*(a)* and *(b)* and are 5 and 1 respectively.

[7]We used version 11a of the BLEU evaluation software to calculate BLEU scores; we downloaded this software from http://www.nist.gov/speech/tests/mt/resources/scoring.htm. We calculated f-scores using GTM v1.2 downloaded from http://nlp.cs.nyu.edu/GTM/.

	depth=1	depth≤2	depth≤3	depth≤4
EN-to-FR:	6,140	29,081	148,165	1,956,786
FR-to-EN:	6,197	29,355	150,460	2,012,632

Table 1: Grammar sizes for each training set.

5 Results and Discussion

5.1 English-to-French Translation Accuracy

Table 2 shows, for each evaluation metric, how the different ranking strategies compare in terms of translation accuracy at each link depth. (For examples of the types of translations generated, see Table 3.) At link depth 1, we see that the BLEU and F-score metrics show that best performance is achieved using MPD ranking whereas the exact match metric ranks MPP translations slightly ahead. At link depth 2, the F-score metric also shows that MPD ranking performs best but the BLEU and exact match scores favour SDER ranking. At link depth 3, BLEU and exact match both attribute best performance to SDER ranking but again the F-score measure places MPD ranking slightly ahead on accuracy. At link depth 4, there is little to choose between MPD and SDER ranking according to BLEU and F-score but the exact match measure puts SDER ahead by 1.25%. Interestingly, ranking according to absolute translation probability does not achieve highest accuracy at any link depth according to *any* of the three evaluation measures. Focussing on link depth 4 – the link depth at which all rankings give their best performance – we see that MPT output is consistently ranked in third place (behind MPD and SDER output) according to the BLEU and F-score metrics and takes second place over MPD ranking on the exact match metric by only 0.21%. Overall, these results show that the highest quality translations are generated using all fragments up to and including link depth 4 and using either MPD or SDER ranking. Finally, for the sake of com-

parison, we note that previous experiments (Hearne, 2005) with a word-based SMT system[8] trained and tested on the same data give a BLEU score of 0.2686, less than half the score our DOT system achieves.

		=1	≤2	≤3	≤4
BLEU	MPT	0.4479	0.5034	0.5277	0.5343
	MPP	0.4507	0.4946	0.5192	0.5216
	MPD	**0.4572**	0.5069	0.5269	**0.5386**
	SDER	0.4168	**0.5080**	**0.5314**	**0.5386**
F-score	MPT	0.6712	0.7035	0.7179	0.7222
	MPP	0.6733	0.6990	0.7135	0.7149
	MPD	**0.6793**	**0.7083**	**0.7213**	**0.7257**
	SDER	0.6513	0.7074	0.7204	0.7254
Exact match	MPT	30.21	37.92	40.00	41.25
	MPP	**30.62**	37.50	38.96	40.00
	MPD	30.42	37.08	39.17	41.04
	SDER	25.62	**38.12**	**41.46**	**42.29**

Table 2: Results for English-to-French DOT translation experiments which compare ranking strategies over each link depth for each metric.

Source	setting printer options
Reference	configuration de les options de impression
DOT	configuration de les options de impression
Source	checking the status of your pending print jobs
Reference	vérification de l' état de les travaux en file d'attente de impression
DOT	vérification de l' état de les travaux de impression en attente

Table 3: Examples of English-to-French translations produced by DOT (MPT, link depth=4).

5.2 French-to-English Translation Accuracy

Table 4 shows, for each evaluation metric, how the different ranking strategies compare in terms of translation accuracy at each link depth. (For examples of the types of translations generated, see Table 5.) As expected, absolute translation scores are higher when translating into English rather than into French because boundary friction problems are less prevalent. The BLEU and F-score measures indicate – with the exception of BLEU at link depth 3 – that the best performance at all link depths is achieved by searching for the MPT. The exact match scores do not follow the same trends: MPD performs best at link depth 1, MPP at link depth 2 and SDER at link depths 3 and 4; the MPT is ranked third at link depths 1 and 2 and last at link depths 3 and 4. The evidence presented here does not allow us to conclude which combination of link depth and ranking method gives the best result. According to the BLEU scores, best performance is at link depth 2 using MPT ranking. According to the F-scores, however, equally high accuracy is achieved using MPT ranking at link depths 2 and 4. Finally, according to the exact match scores, overall best performance is obtained using SDER ranking at fragment link depth 4. Again for the sake of comparison, we note that previous experiments (Hearne, 2005) with a word-based SMT system (see footnote 8) trained and tested on the same data give a BLEU score of 0.3076, which is 45% worse in real terms than the score achieved by our DOT system.

		=1	≤2	≤3	≤4
BLEU	MPT	**0.4990**	**0.5513**	0.5447	**0.5494**
	MPP	0.4915	0.5406	0.5454	0.5449
	MPD	0.4946	0.5396	0.5436	0.5434
	SDER	0.4316	0.5318	**0.5465**	0.5488
F-score	MPT	**0.7177**	**0.7463**	**0.7443**	**0.7463**
	MPP	0.7098	0.7407	0.7423	0.7427
	MPD	0.7119	0.7376	0.7386	0.7396
	SDER	0.6832	0.7343	0.7401	0.7421
Exact match	MPT	43.75	49.17	48.75	49.38
	MPP	44.38	**50.00**	49.38	50.21
	MPD	**44.79**	49.38	49.79	50.21
	SDER	36.46	48.54	**50.00**	**50.42**

Table 4: Results for French-to-English DOT translation experiments which compare ranking strategies over each link depth for each metric.

5.3 Efficiency

Table 6 gives the average number of seconds required to translate each sentence at each link depth and using each of the four ranking strategies.[9]

Not surprisingly, the time taken to translate each sentence increases as fragment link depth increases, with a large increase from link depth 3 to link depth 4. The extra time taken for each sentence at greater

[8] Training was carried out using Giza++ (Och & Ney, 2003) downloaded from http://www.fjoch.com/GIZA++.html. Translations were generated using the ISI ReWrite Decoder (Germann, Jahr, Knight, Marcu, & Yamada, 2001) downloaded from http://www.isi.edu/licensed-sw/rewrite-decoder/ and the CMU-Cambridge Statistical Language Modeling toolkit (Clarkson & Rosenfeld, 1997) downloaded from http://mi.eng.cam.ac.uk/~prc14/toolkit.html.

[9] All experiments were carried out on a Pentium 4 with 2.39GHz CPU and 2Gb RAM.

Source	modification de les options de impression enregistrées dans un fichier de préréglages
Reference	editing the printer options defined in a preset file
DOT	editing the printer options defined in a preset file
Source	débranchez le cordon d'alimentation de la prise murale .
Reference	unplug the power cord from the wall outlet .
DOT	disconnect the power cord from the wall outlet .

Table 5: Examples of French-to-English translations produced by DOT (MPT, link depth=4).

	ENGLISH-TO-FRENCH					FRENCH-TO-ENGLISH				
	CPU seconds/sentence					CPU seconds/sentence				
	MPT	MPP	MPD	SDER		MPT	MPP	MPD	SDER	
1	1.39	1.33	0.29	0.30	1	0.72	3.73	3.12	3.13	
2	2.06	1.55	0.57	0.58	2	1.16	3.85	3.53	3.58	
3	3.05	2.28	1.40	1.41	3	2.32	4.96	4.62	4.64	
4	12.8	11.9	11.3	11.1	4	18.9	21.5	21.1	20.8	

Table 6: Average time taken to translate each sentence for all link depths and ranking strategies, and both translation directions.

link depths is spent building the translation space (which contains increasing numbers of fragments) rather than ranking the output translations. Absolute translation times are greater when translating from French than from English because the average French sentence length is longer than the average English sentence length (10.1 words/sentence vs. 8.8 words/sentence) and, consequently, larger translation spaces must be built for French.

Looking at the different ranking algorithms, we observe that for English to French translation at each link depth, MPT ranking takes longest, followed by MPP ranking and MPD, and SDER rankings are fastest but the difference between the fastest and slowest at link depth 4 is just 1.7 seconds. The opposite, however, holds for French to English translation: MPP ranking is slowest, followed by MPD and SDER, and MPT ranking is consistently fastest. (Again, the difference in time taken between fastest and slowest at link depth 4 is small.) We conclude that the ranking methods which require random sampling do not take significantly longer to translate each sentence than our ranking strategies based on the Viterbi algorithm.

5.4 Conclusion

We looked in sections 5.1 and 5.2 at the accuracy of each system configuration for each translation direction. In fact, if we ignore the direction issue and evaluate each configuration over *all* splits, we see that highest accuracy is obtained over all three evaluation metrics by searching for the shortest derivation and using all fragments of link depth 4 or less (BLEU=0.5433; F-score=0.7254; Exact Match=46.35%). Having also considered the efficiency of each configuration and observed that for the configurations which give the best accuracy (MPT and SDER at link depth 4), there is little difference in efficiency – MPT takes, on average, 1.7 seconds per sentence longer than SDER when translating from English to French but SDER takes 1.9 seconds per sentence longer when translating from French to English. Thus, for the DOT model over this particular dataset, we conclude that there is no need to sacrifice accuracy for efficiency as the most accurate model – SDER at link depth 4 – is as efficient as its closest competitor.

6 Future Work

We would like to apply the improved parameter estimation methods developed for DOP (e.g. (Sima'an & Buratto, 2003)) to the DOT model. Better estimation of the fragment probabilities should lead to further improvements in accuracy as the fragment set increases in size. Furthermore, searching for the most probable translation may yield higher translation accuracy than searching for the shortest derivation if parameter estimation is improved. We also intend to experiment with combining probabilities with SDER ranking, as proposed for DOP in (Bod, 2003). We are currently carrying out empirical evaluation of DOT on much larger datasets than heretofore, and for different language pairs.

Finally, DOT models can also be defined for representations corresponding to more sophisticated linguistic formalisms. We intend to carry out an empirical evaluation of the LFG-DOT model (Way, 1999; Hearne, 2005), which uses LFG f-structure informa-

tion in addition to the phrase-structure trees used in the DOT model described here.

References

Bod, R. (1998). *Beyond Grammar: An Experience-Based Theory of Language.* Stanford CA: CSLI Publications.

Bod, R. (2000). Parsing with the Shortest Derivation. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING'00)* (pp. 69–75). Saarbrücken, Germany.

Bod, R. (2003). An Efficient Implementation of a New DOP Model. In *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL'03)* (pp. 19–26). Budapest, Hungary.

Bod, R., Scha, R., & Sima'an, K. (Eds.). (2003). *Data-Oriented Parsing.* Stanford CA: CSLI Publications.

Chappelier, J.-C., & Rajman, M. (2003). Parsing DOP with Monte-Carlo Techniques. In R. Bod, R. Scha, & K. Sima'an (Eds.), *Data-Oriented Parsing* (pp. 63–81). Stanford CA: CSLI Publications.

Clarkson, P., & Rosenfeld, R. (1997). Statistical Language Modeling Using the CMU–Cambridge Toolkit. In *Proceedings of the 5th biennial European Conference on Speech Communication and Technology (EUROSPEECH'97)* (pp. 2707–2710). Rhodes, Greece.

Frank, A. (1999). LFG-based syntactic transfer from English to French with the Xerox Translation Environment. In *Proceedings of the ESSLLI'99 Summer School.* Utrecht, The Netherlands.

Germann, U., Jahr, M., Knight, K., Marcu, D., & Yamada, K. (2001). Fast Decoding and Optimal Decoding for Machine Translation. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL'01)* (p. 228-235). Toulouse, France.

Groves, D., Hearne, M., & Way, A. (2004). Robust Sub-Sentential Alignment of Phrase-Structure Trees. In *Proceedings of The 20th International Conference on Computational Linguistics (COLING'04)* (pp. 1072–1078). Geneva, Switzerland.

Hearne, M. (2005). *Data-Oriented Models of Parsing and Translation.* Unpublished doctoral dissertation, Dublin City University, Dublin, Ireland.

Hearne, M., & Way, A. (2003). Seeing the Wood for the Trees: Data-Oriented Translation. In *Proceedings of the Ninth Machine Translation Summit* (pp. 165–172). New Orleans, USA.

Hoogweg, L. (2000). *Extending DOP1 with the insertion operation.* Unpublished master's thesis, University of Amsterdam, The Netherlands.

Och, F. J., & Ney, H. (2003). A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, *29*(1), 19–51.

Poutsma, A. (1998). Data-Oriented Translation. In *Ninth Conference of Computational Linguistics in the Netherlands.* Leuven, Belgium.

Poutsma, A. (2003). Machine Translation with Tree-DOP. In R. Bod, R. Scha, & K. Sima'an (Eds.), *Data-Oriented Parsing* (pp. 63–81). Stanford CA: CSLI Publications.

Sima'an, K. (1995). An optimized algorithm for Data Oriented Parsing. In *Proceedings of International Conference on Recent Advances in Natural Language Processing.* Tzigov Chark, Bulgaria.

Sima'an, K. (1996). Computational Complexity of Probabilistic Disambiguation by means of Tree-Grammars. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING'96)* (pp. 1175–1180). Copenhagen, Denmark.

Sima'an, K., & Buratto, L. (2003). Backoff Parameter Estimation for the DOP Model. In *Proceedings of the 14th European Conference on Machine Learning (ECML'03)* (pp. 373–384). Cavtat-Dubrovnik, Croatia.

Way, A. (1999). A Hybrid Architecture for Robust MT using LFG-DOP. *Journal of Experimental and Theoretical Artificial Intelligence*, **11**, 441–471.