

Generating English-Persian Parallel Corpus Using an Automatic Anchor Finding Sentence Aligner

Meisam VOSOUGHPOUR YAZDCHI
School of Electrical and Computer Engineering,
College of Engineering, University of Tehran
Email: vosoughp@ualberta.ca

Heshaam FAILI
School of Electrical and Computer Engineering,
College of Engineering, University of Tehran
Email: hfaili@ut.ac.ir

Abstract—The more we can enlarge a parallel bilingual corpus, the more we have made it effective and powerful. Providing such corpora demands special efforts both in seeking for as much already translated texts as possible and also in designing appropriate sentence alignment algorithms with as less time complexity as possible. In this paper, we propose algorithms for sentence aligning of two Persian-English texts in linear time complexity and with a surprisingly high accuracy. This linear time-complexity is achieved through our new language-independent anchor finding algorithm which enables us to align as a big parallel text as a whole book in a single attempt and with a high accuracy. As far as we know, this project is the first automatic construction of an English-Persian parallel sentence-level corpus.

I. INTRODUCTION

Sentence-level aligned parallel corpora have several applications of high importance, especially in NLP, like cross-language information retrieval [1], machine translation [2], lexicography [3], and language learning [4]. During the last decade, numerous projects are defined and fulfilled aimed to build up as larger parallel bilingual corpora as possible. Usually, one side of such projects is English language. Chinese-English [5], French-English [6], Hungarian-English [7], Swedish-English [1] are most frequent examples. Such projects encounter two major challenges: gathering bilingual texts to be aligned and designing sentence-alignment algorithms appropriate for the specified languages. Bilingual books are usually rare. Therefore, gathering bilingual resources from the World Wide Web could be a good choice. Resnik has had valuable attempts in this. STRAND (Structural Translation Recognition for Acquiring Natural Data) is an automatic language independent bilingual text finder from the Web presented by Resnik [8]. Latter, he extends STRAND by adding a language identification facility [9]. Chen in [10] with PTMiner and Ma in [11] with BITS have also had similar successful attempts. In a latter work, enumerating some drawbacks of STRAND, PTMiner, and BITS, a new approach is presented for gathering bilingual texts through RSS news feeds which has tried to cover those drawbacks and create a corpus simpler and faster [12]. However, Web mining does not necessarily work for corpora of any languages; as bilingual texts for some languages are rarely found in Web. Towards

this, Callison-Burch, not interested in extracting new data from the Web, try to recreate new data out of existing resources in the corresponding language and also from resources in other languages [13].

While mentioned projects concentrate on methods of gathering sufficient resources, there are also valuable efforts on methods of automatic sentence alignment. In [14] and [15] two algorithms are proposed for this problem. Both algorithms suppose a short sentence is more likely the translation of a short sentence in the other language and the same condition holds for sentences of any lengths. Both algorithms are also based on a dynamic programming (DP) approach with time complexity quadratic in the length of parallel texts [6]. Again both of them need the text to be manually divided into acceptably small chunks (due to non-linear time complexity). In 1993, a new method based on the probabilities of word-translations were proposed, [6], which focused on the accuracy of alignment especially in conditions that input texts are too noisy or are partially skipped. But, this algorithm was much slower than Brown and Gale algorithms.

Latter methods, mostly try to automatically find anchors by different means; then the divided text is sentence aligned using a combination of classic Gale or Chen methods. During the alignment of a Hungarian-English corpus, a hybrid algorithm for sentence alignment is proposed in which the automatic anchor finding is accomplished using Named Entity recognition [7]. In another work, again a new anchor finding method by mapping cognates in a French-English corpus is focused to have an accurate and robust sentence alignment [16]. In [7] named entities are used as anchors and in [16] anchors are cognates in a bilingual corpus. In this paper, an automatic anchor finding method is presented which is implemented for gathering an aligned English-Persian bilingual corpus. In this project, neither of the mentioned methods of anchor finding can be successfully applied; since, Persian and English alphabets and written letters are absolutely distinct and irrelevant. Instead, in our work anchors are sentence beads [14] which are automatically found by a smart algorithm described in section IV-B. This algorithm is a part of our three pass sentence alignment method in each of which a certain characteristic of sentences are taken into account, such as length or probabilistic word to word translations. By this, we have proposed a linear-time algorithm that can align texts

of any length (like a whole translated book in one attempt) with surprisingly high accuracy.

As far as we know, no other parallel corpus for English-Persian language pairs is reported except for Mosavi’s work [4] in which a manually aligned parallel corpus containing about five million words is introduced. In this paper, after defining the problem formally, two length-based alignment algorithms are presented. The first one is a greedy linear-time approach which is not optimal, while the other is a new DP algorithm with high accuracy. Also, a human-interactive method as well as a fully automatic algorithm for anchor finding are introduced. Finally, the experimental evaluations for gathering English-Persian aligned corpus are reported.

II. FORMAL PROBLEM FORMULATION

Let’s have the following primitive definitions in an English-Persian bilingual text:

$$\begin{aligned} E &= \{e_i : \text{an English sentence} | 0 \leq i < n_e\} \\ P &= \{p_i : \text{a Persian sentence} | 0 \leq i < n_p\} \end{aligned} \quad (1)$$

where E is the set of all English and P is the set of all Persian sentences; e_i represents the i th English sentence in order from the beginning of the text and p_i has the same definition correspondingly.

The ordered English and Persian texts (denoted by E_{ordered} and P_{ordered} respectively) and also an English and a Persian part (denoted by E_p and P_p respectively) are formally defined as follows:

Definition 1. E_{ordered} : The sequence of all E members sorted in order of their indices.

P_{ordered} : The sequence of all P members sorted in order of their indices.

E_p : A subsequence of E_{ordered} of length $l_e : 0 \leq l_e \leq n_e$.

P_p : A subsequence of P_{ordered} of length $l_p : 0 \leq l_p \leq n_p$.

A sentence bead [14] is a pair of two parts: the English and Persian parts which is formulated as follows:

$$b = (E_p, P_p). \quad (2)$$

Also, elen and plen parameters are defined as the length of sentences in term of the number of characters:

Definition 2. $\text{elen}[0 \leq i < n_e]$: The length of i th English sentence.

$\text{plen}[0 \leq i < n_p]$: The length of i th Persian sentence.

It should be considered that different languages tend to have different sentence lengths of the same meaning. In order to align this difference between two texts, we calculate a

balancing ratio as follows:

$$\begin{aligned} \text{Total}_{\text{English}} &= \sum_{i=0}^{n_e-1} \text{elen}[i], \\ \text{Total}_{\text{Persian}} &= \sum_{i=0}^{n_p-1} \text{plen}[i], \\ \text{balancing} &= \frac{\text{Total}_{\text{English}}}{\text{Total}_{\text{Persian}}}. \end{aligned} \quad (3)$$

We then multiply all plen elements by this balancing ratio to have two texts with equal total lengths. After this, everywhere we say plen we mean the new plen after the application of the balancing ratio; now, we’ve got:

$$\text{Total}_{\text{English}} = \text{Total}_{\text{Persian}}. \quad (4)$$

An alignment is a sequence of sentence beads, say $b_i (0 \leq i < k_b)$, with the following properties:

$$\begin{aligned} \text{Concatenation}(b_0.E_p, b_1.E_p, \dots, b_{k-1}.E_p) &= \text{English-text} \\ \text{Concatenation}(b_0.P_p, b_1.P_p, \dots, b_{k-1}.P_p) &= \text{Persian-text} \end{aligned} \quad (5)$$

We still need some other definitions to elaborate our algorithm’s core. Definition 3 illustrates a formal definition for an accurate sentence bead:

Definition 3. The bead b_i is said to be accurate iff $b_i.E_p$ is the translation of $b_i.P_p$.

Definition 4. The bead b_i is empty if $(b_i.E_p = \emptyset \oplus b_i.P_p = \emptyset)$.

Definition 5. The empty bead b_i is truly empty if: $\nexists j$ in the range: b_j is accurate $\wedge (b_j.E_p \cap b_i.E_p \neq \emptyset \vee b_j.P_p \cap b_i.P_p \neq \emptyset)$.

Definition 6. Alignment g is said to be conditionally golden iff: $\forall 0 \leq i < k_g : (g_i$ is accurate $\vee g_i$ is truly empty).

Definition 7. Alignment g is golden if: g is conditionally golden $\wedge \nexists h : h$ is golden and $k_h < k_g$.

In fact, a conditionally golden alignment is the one in which each bead is either accurate or truly skips a part of text in one language that is not translated in the other; it is done by means of a truly empty bead that is defined in definition (5). Furthermore, a golden alignment is a kind of maximal conditionally golden alignment that is more clarified in the following example: Suppose an alignment consists of only one bead whose English and Persian parts contains the entire English and Persian texts respectively. Such an alignment is conditionally golden, though obviously a trivial one. Actually, in spite of its appearance, there are a lot of conditionally golden alignments for two sets of English and Persian sentences; but, finding a golden alignment is usually difficult and they are really few and most of the times there exists only one of them. In fact, an alignment cannot be said to be golden unless it is a maximal alignment in the sense that it is no longer possible to break an accurate bead into

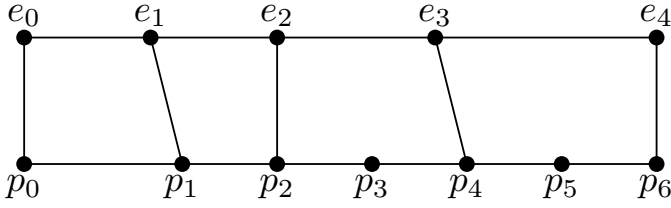


Fig. 1. A visual representation of a sentence alignment

two new accurate beads, towards enhancing the resolution and granularity of the alignment.

Now, let us define the accuracy of an alignment:

- An alignment's accuracy is a real number between 0 and 1.
- A golden alignment has a full accuracy of 1.
- The accuracy for a non-golden alignment is higher if it is more similar to a golden alignment.

In order to accomplish the goal of creating an alignment with as highest accuracy as possible, two algorithms are elaborated in continue. Let's define a graphical version of sentence alignment problem. In this representation, every sentence is represented by a segment with length proportional to the corresponding sentence length, forming the sequence of sentences from left to right. Between any two consecutive segments (sentences) there is a node named with little letters (e_0 to $e_{(n_e)}$ and p_0 to $p_{(n_p)}$). An edge can be drawn between a Persian and an English node on the condition that no edge intersection is allowed. We use the following notation:

$$\text{edge}(e_i, p_j) = \begin{cases} 0 & \text{if there is no edge} \\ & \text{between } e_i \text{ and } p_j, \\ 1 & \text{if there is an edge} \\ & \text{between } e_i \text{ and } p_j. \end{cases} \quad (6)$$

An alignment in this representation is defined as follows:

Assuming that $\text{edge}(e_0, p_0) = 1$ and $\text{edge}(e_{(n_e)}, p_{(n_p)}) = 1$ indicating that two texts have the same beginnings and endings, any possible state of edges represents a distinct sentence alignment. It is easy to understand that the space between any two consecutive edges represents a bead. Figure 1, shows a graphical representation of a sample alignment. This graphical representation will be frequently referred to in continue.

Let b_i be a sentence bead; then the function $\text{fitness}(b_i)$ tries to estimate the accuracy of this bead through different heuristics and can be clarified as follows:

$\text{fitness}(b_i)$: the more this amount is, the more it is likely that b_i is accurate.

In the same way, let b be an alignment, then $\text{fitness}(b)$ is an estimate for accuracy of the entire alignment. It is very important to note that these functions are only estimates of corresponding accuracies, computed through different heuristics most of which are even regardless of the text and based on simple appearances of sentences (like sentence length).

Usually, the fitness of an alignment is a function of fitness of its sentence beads. For instance, one way to compute the

fitness of an alignment out of its members (sentence beads) is as following formula in which b is the alignment:

$$\text{fitness}(b) = \sum_{i=0}^{i=k_b-1} \text{fitness}(b_i). \quad (7)$$

The above formula is only an example; in fact, neither do all fitness functions are computed as above nor is this formula necessarily efficient.

A sentence alignment algorithm's goal is to find an alignment with maximum fitness. For this purpose, two different methods exist. First, by searching in the space of possible alignments, (i.e. searching in all possible states of non-intersecting edges in the mentioned visual model); using a brute-force search, the efficient alignment can be generated. In this case, we just need to iterate huge number of states which is exponential to number of sentences. In section III-B, a smart DP approach to find the efficient alignment (with maximum fitness) with time complexity of $O(n^2)$ is proposed. The second method is to design a heuristic to greedy search the answer in linear time with not necessarily to be optimal. Again, a method based on this approach is proposed on section III-A.

III. LENGTH-BASED ALIGNMENT ALGORITHMS

In order to build a highly accurate English-Persian aligned corpus, two length-based methods are proposed. The first one is a DP approach while the second is a greedy linear-time aligning method. Based on the logic that two English and Persian sentences being each other's translations tend to have approximately the same lengths, we propose the following fitness functions:

Let b be an alignment, then:

$$\text{fitness}(b_i) = -|\text{len}(b_i.E_p) - \text{len}(b_i.P_p)|, \quad (8)$$

$$\text{fitness}(b) = \sum_{i=0}^{i=k_b-1} \text{fitness}(b_i) + k_b \times \text{Bonus}. \quad (9)$$

The formula for computing the fitness of a single sentence bead simply says that the more the difference in lengths of two parts is, the less the bead's fitness will be. But, the fitness of an alignment is computed through a more complicated formula which has two parts, the second which is $k_b \times \text{Bonus}$ and needs some more elaborations. If we did not have the second part, we would have the same problem as we mentioned during the explanation of the difference between *conditionally golden* and *golden* alignments; that is, the first part alone does not tend to increase the number of beads and enhance the alignment's resolution. Why does it happen? It is very simple: breaking an accurate sentence bead into two accurate sentence beads decreases the term $\sum_{i=0}^{i=k_b-1} \text{fitness}(b_i)$ and if the fitness function consists only this term, then the whole fitness of the alignment will also decrease. This claim is mathematically proven which is not discussed here due to lack of space.

The second term adds the constant of Bonus per bead, encouraging searching algorithms to add beads. In fact, this second term makes up the decline of fitness caused by breaking an existing bead and adding to the number of beads.

A. Greedy Approach

Let's consider the graphical representation of sentence alignment a sample of which is shown in figure 1. The greedy algorithm is presented by the following pseudo-code that is self-explanatory. After this pseudo-code, extra explanations will better clarify it:

```

for  $i = 1$  to  $n_e - 1$  do
   $p \leftarrow$  closest node in set of Persian nodes to  $e_i$ 
  if horizontal distance between  $e_i$  and  $p$  is less than  $d$ 
  then
    connect  $e_i$  to  $p$ 
  end if
end for

```

where d is the upper bound for the horizontal distance of two nodes to be connected. This algorithm is rational in the sense that when we draw an edge that is approximately vertical, we are in fact dividing two texts into two parts of equal lengths; this policy is towards the main principle of length-based algorithms. The explained greedy algorithm has linear time complexity. But, since it is based on greedy decisions and does not consider all possible states, the results do not have acceptable accuracies.

B. Dynamic Programming Approach

In this section, we will draw a scheme by which we can find the best alignment with maximum fitness in $O(n^2)$ time complexity. First, we should have the following definitions:

Definition 8. $f(i, j)$: Suppose $E' = \{E_i, E_{i+1}, \dots, E_{n_e-1}\}$ and $P' = \{P_j, P_{j+1}, \dots, P_{n_p-1}\}$, then $f(i, j)$ is the fitness of the best alignment for this pair of texts, E' and P' . It is obvious that $f(n_e, n_p) = 0$.

Definition 9. $\text{fitness}(i, j, k, l)$ stands for $\text{fitness}(b(E_p, P_p))$ such that $E_p = \text{Concat}(E_i, E_{i+1}, \dots, E_{k-1})$, $P_p = \text{Concat}(P_j, P_{j+1}, \dots, P_{l-1})$.

Definition 10. $F(i, j, k, l) = \text{fitness}(i, j, k, l) + f(k, l)$.

Definition 11. The set $\text{FS}_{ij} = \{F(i, j, k, l) | i \leq k \leq n_e - 1 \wedge j \leq l \leq n_p - 1 \wedge (i \neq k \vee j \neq l)\}$.

Having above definitions, the recursive relation of our DP approach is:

$$f(i, j) = \max\{FS_{ij}\}. \quad (10)$$

Finally, we can simply conclude that an alignment's fitness can be computed as follows:

$$\text{fitness}(b) = f(0, 0). \quad (11)$$

$|FS_{ij}|$ is of $O(n^2)$, therefore the entire algorithm has the time complexity of $O(n^4)$. But, after pruning the trivial states by setting an upper bound for maximum allowed length for any of English or Persian parts of a bead, $|FS_{ij}|$ will be of $O(1)$ and the algorithm's time complexity is enhanced to $O(n^2)$.

IV. ANCHOR FINDING

Our experiments have shown that our DP algorithm aligns about five-six standard pages of English text and its translation in Persian in less than a second. In order to align a book of 200 pages, we have to divide it into at least 35 chunks of less than six pages and then align these 35 chunks separately. We can find the dividing points of these chunks, anchors, manually. But, in this section we propose a method to automatically find anchors accurately in two parallel texts; in this way we can align as much text in linear time. In this section, we first describe a semi-automatic anchor finding and in other words a *human interactive* one; then we will elaborate our two-pass fully-automatic anchor finding algorithm by which we can surprisingly align as much data in linear time.

A. Human-Interactive Anchor Finding

A good human-interactive process demands a user-friendly GUI. We have designed and implemented such GUI; you can see a sample snapshot of this user-friendly software in figure 2. This method simply uses the idea behind the greedy method in section III-A. First, looking at the whole size of the text, it simply calculates how many chunks are needed; then it divides the total length of English text by this number resulting in each chunk's approximate length. Finally, the user-friendly GUI suggests the best sentences in both texts which can best divide two texts in parallel into calculated number of chunks. Based on our experiments, for a pair of relatively clean texts, this suggested anchor sentence is most of the time true, though these points are only suggestions and the interacting human can simply change the dividing points by moving this point to the next or previous sentences in a user-friendly manner.

B. Three-pass Linear-time Alignment by an Automatic Anchor Finding

This three-pass method is completely automatic with no human interaction. In the first pass, our greedy sentence aligner is applied to have an initial alignment in linear time. In section V we will see that each bead created by this greedy algorithm is accurate with probability of more than 0.6. As previously mentioned, our DP approach can align approximately 5–6 pages (also called a *chunk*) in less than a second; it is equal to about 200 sentences and 150 beads. We also define a *half-chunk* with half of the size of a chunk. Therefore, with probability p there is at least one accurate bead in a given sequence of 75 beads (a half-chunk) which are created by the greedy aligner; this probability can be computed as follows:

$$p = 1 - (1 - 0.6)^{75} \approx 1 \quad (12)$$

Therefore, we can be sure that almost always there is at least one accurate bead in any half-chunk. Now, we need to have the following definitions:

Definition 12. *Lexicon-based fitness function(lff)*: A lexicon-based fitness function, is the one which heavily considers words of English and Persian parts of a bead to estimate its accuracy.

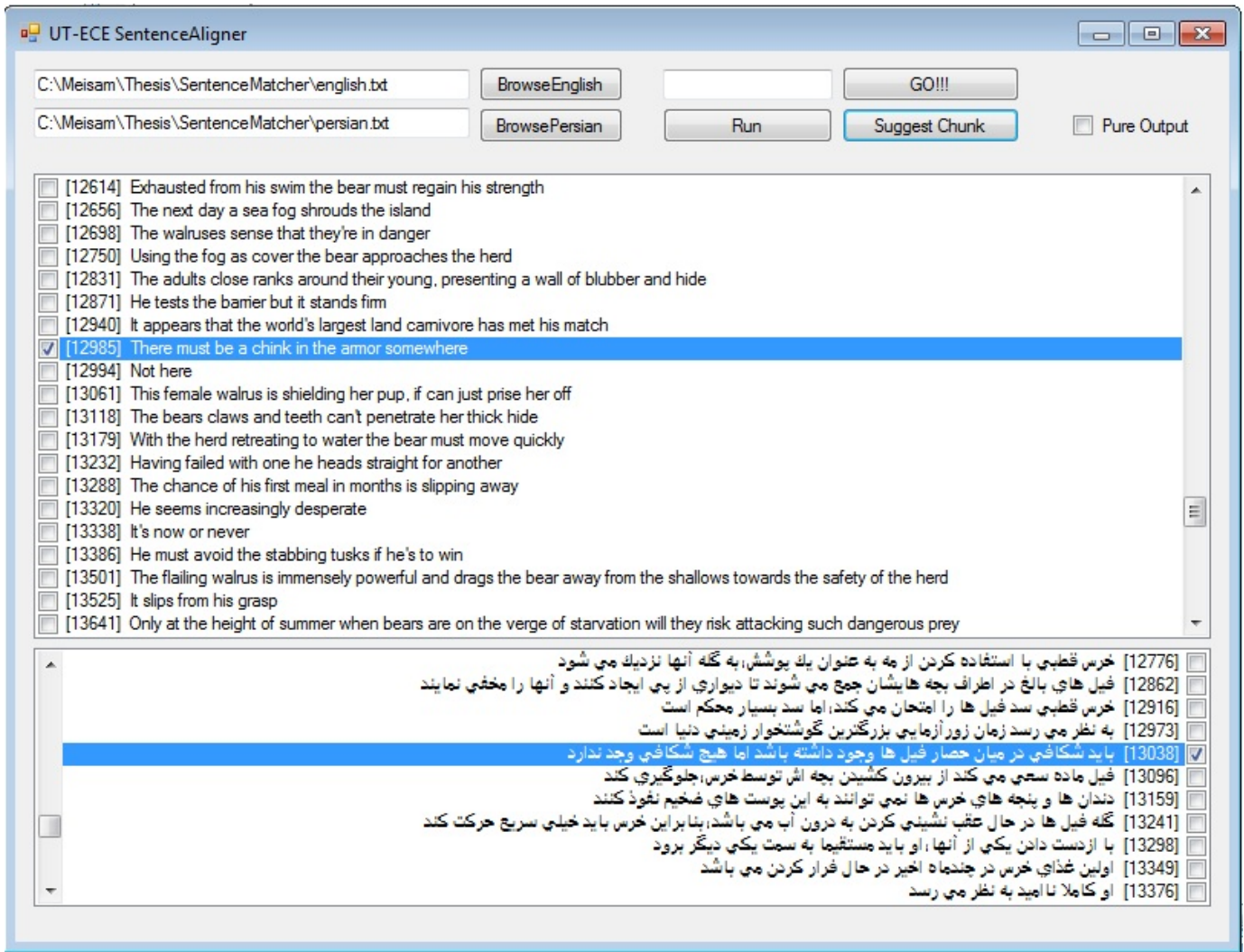


Fig. 2. A linear time sentence aligner with a semi-automatic anchor finder.

For alignment b , $lff(b_i) : 0 \leq i < k_b$ is defined as a post-alignment lexicon-based fitness for each bead.

In the first pass, an alignment is obtained using our greedy algorithm; we call this alignment b . In the second pass, we should divide our Persian and English texts into necessary number of half-chunks that is simply calculated by: $N_c = \left\lceil \frac{\max(n_e, n_p)}{100} \right\rceil$. It is done in the following procedure:

We first apply our lff on all beads in b . Then, we search in each half-chunk for a bead which has the maximum lff in that half-chunk; finally we will have found N_c beads with highest lff in their half-chunks which statically can be proven to be accurate beads with probability of very close to 1. We use these beads as our anchors by which we can divide the whole texts into $N_c + 1$ chunks with sizes between 100 to 200 sentences. We used half-chunks instead of chunks to ensure that the distance between two consecutive anchors does not exceed the size of a chunk so as to be alignable in less than a second by the DP algorithm.

In the third pass, the DP algorithm is applied on each of $N_c + 1$ parallel chunks. Assuming that the DP approach aligns each chunk in constant time of T (e.g. $T = 1$ sec.), we can align any parallel texts of any size in time: $\text{Time}_{\text{Total}} = T \times (N_c + 1)$, where T is constant and N_c grows linearly. It means that we have proposed an alignment algorithm with the same high accuracy of our DP approach and with linear time complexity.

V. RESULTS AND CONCLUSIONS

We introduced a greedy and also a DP algorithm for sentence alignment. Then we combined these two approaches and added another stage to form a three-pass sentence alignment which also contains a smart *anchor finder* to divide bilingual texts of any length into acceptably smaller chunks each of which can be independently sentence aligned. Up to 60 pages of English text with its Persian translation in three different genres are focused as the test case; the genres include novels, computer science academic books, and a BBC report

TABLE I

ACCURACY OF THREE DIFFERENT SENTENCE-ALIGNMENT ALGORITHMS.

Alignment Method	Accuracy
Greedy	0.674
DP with Manual Anchor Placement	0.995
DP with Automatic Anchor Finding	0.994

about ecosystems. We have applied three different sentence alignment approaches to this 60-page bilingual text:

- 1) The greedy algorithm in section III-A.
- 2) The DP algorithm in section III-B with anchors manually inserted.
- 3) The DP algorithm in section III-B with our automatic anchor finding algorithm in section IV-B.

The results obtained in this experiment are shown in table I. These accuracies are measured by comparing the computed alignment with a manually created golden alignment which results in a number in $[0, 1]$ that indicates the degree of similarity between the given alignment and the golden one. This comparison is done in the following procedure: Let G be the golden alignment and B be the obtained alignment. By definition, G and B are two sets of sentence beads and their degree of similarity is obtained by calculating their intersection set. It is formulated as follows:

$$C = G \cap B, \quad \text{accuracy} = \frac{|C|}{|G|}, \quad (13)$$

where *accuracy* simply computes the percent of accurate beads in the computed alignment.

In table I, the number in the second row (0.995) indicates that in alignment of a bilingual chunk of acceptably small size (less than 5 pages) the DP approach has the average error of approximately 0.5%. It may be expected that the accuracy rate of DP with manual anchor placement must be meaningfully higher than that of DP with automatic anchor finding; but, approximately the same accuracy rate is obtained. In fact, it suggests that the error of automatic anchor finding algorithm is very low is such a way that for a text of 60 pages it can find necessary sentence anchors with almost no error. However, based on what happens during the algorithm and the computed probability in formula 12, this strength is not a matter of much surprise.

REFERENCES

- [1] T. Talvensaaari, J. Laurikkala, K. Järvelin, M. Juhola, and H. Keskustalo, "Creating and exploiting a comparable corpus in cross-language information retrieval," *ACM Trans. Inf. Syst.*, vol. 25, no. 1, p. 4, 2007.
- [2] M. Guidère, "Toward corpus-based machine translation for standard arabic," *Translation Journal*, vol. 6, no. 1, 2002.
- [3] R. Krishnamurthy, "Corpus-driven lexicography," *International Journal of Lexicography*, vol. 21, no. 3, pp. 231–242, July 2008.
- [4] T. Mosavi Miangah, "Constructing a large-scale english-persian parallel corpus," *Meta: journal des raducteurs / Meta: Translators' Journal*, vol. 54, no. 1, pp. 181–188, January 2009.
- [5] L. Sun, S. Xue, W. Qu, X. Wang, and Y. Sun, "Constructing of a large-scale chinese-english parallel corpus," in *COLING '02: Proceedings of the 3rd workshop on Asian language resources and international standardization*. Morristown, NJ, USA: Association for Computational Linguistics, 2002, pp. 1–8.
- [6] S. F. Chen, "Aligning sentences in bilingual corpora using lexical information," in *Proceedings of the 31st annual meeting on Association for Computational Linguistics*. Morristown, NJ, USA: Association for Computational Linguistics, 1993, pp. 9–16.
- [7] K. Tóth, R. Farkas, and A. Kocsor, "Sentence alignment of hungarian-english parallel corpora using a hybrid algorithm," *Acta Cybern.*, vol. 18, no. 3, pp. 463–478, 2008.
- [8] P. Resnik, "Parallel strands: A preliminary investigation into mining the web for bilingual text," in *AMTA '98: Proceedings of the Third Conference of the Association for Machine Translation in the Americas on Machine Translation and the Information Soup*. London, UK: Springer-Verlag, 1998, pp. 72–82.
- [9] —, "Mining the web for bilingual text," in *In Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, 1999, pp. 527–534.
- [10] J. Chen and J.-Y. Nie, "Automatic construction of parallel english-chinese corpus for cross-language information retrieval," in *Proceedings of the sixth conference on Applied natural language processing*. Morristown, NJ, USA: Association for Computational Linguistics, 2000, pp. 21–28.
- [11] X. Ma and M. Y. Liberman, "Bits: a method for bilingual text search over the web," in *Machine Translation Summit VII*, September 1999, pp. 538–542.
- [12] J. Fry, "Assembling a parallel corpus from rss news feeds," in *Proceedings of the Workshop on Example-Based Machine Translation, MT Summit X*, Phuket, Thailand, September 2005.
- [13] C. Callison-Burch and M. Osborne, "Bootstrapping parallel corpora," in *Proceedings of the HLT-NAACL 2003 Workshop on Building and using parallel texts*. Morristown, NJ, USA: Association for Computational Linguistics, 2003, pp. 44–49.
- [14] P. F. Brown, J. C. Lai, and R. L. Mercer, "Aligning sentences in parallel corpora," in *Proceedings of the 29th annual meeting on Association for Computational Linguistics*. Morristown, NJ, USA: Association for Computational Linguistics, 1991, pp. 169–176.
- [15] W. A. Gale and K. W. Church, "A program for aligning sentences in bilingual corpora," in *Proceedings of the 29th annual meeting on Association for Computational Linguistics*. Morristown, NJ, USA: Association for Computational Linguistics, 1991, pp. 177–184.
- [16] M. Simard and P. Plamondon, "Bilingual sentence alignment: Balancing robustness and accuracy," in *In proceedings of the second conference of the association for machine translation in the Americas (AMTA)*, 1996, pp. 59–80.