The applications of Statistical Machine Translation tool – PGIZA++

Arkadiusz Szał (Adam Mickiewicz University, Faculty of Mathematics and Informatics)

Giza [1] is a tool designed for the alignment of bilingual corpora at the word level. The program takes a pair of raw texts as input and returns a set of word pairs (one word in each of the languages) labelled with the probability of the words being mutual equivalents. Word alignment is the crucial phase in Statistical Machine Translation ([2], [3], [4]). It may also be used for creating dynamically expanded lexicons – particularly those oriented for a specific domain.

Giza is reputed to be a powerful tool. Its main drawback however is computational complexity (squared against the word volume of the corpora). Several extensions to Giza have been implemented, aiming to decrease the computation time.

The paper compares three implementations of Giza, namely: Giza++, MGiza++, PGiza++([5], [6]). Giza++ is a single-threated implementation. MGiza++ and PGiza++ use multiple threats to create output dictionary. MGiza++ is run on one machine with more than one cores and/or processors. PGiza++ is run on several machines (called also nodes), connected with each other to form a cluster. On each computer a separate threat is run to carry out computations. Figure 1) compares the three algorithms (the pictures come from [5]).

In our experiment we used an IBM Blade cluster with the Intel(R) Xeon(TM) processor. Each node worked on Debian GNU. We compared speed of computation of those three implementations with different input data and different work environment (32-bit and 64-bit machine).

Our experiments show significant advantage of MGiza++ over Giza++. Comparison of Giza++ and MGiza++ speed shows the benefit of using multi threads. MGiza+ is several times faster on a multiple- core processor. Last but not least: MGiza++ is easier to install and manage.

However, we have found out that the current implementations of PGiza++ do not meet all expectations. The tool is ineffective, as far as both time and memory consumption are concerned. There is no evidence that running PGiza++ on a computer cluster may be of any practical use.

We conclude that a new method should be developed that would work on multiple nodes using multiple cores on each node. The method would take benefits from the fact that MGiza++ weakens the negative impact of the normalisation process in PGiza++.

The idea that we have in mind is alignment symmetrisation with multithread implementation. Alignment symmetrisation consists in building two-directional lexicons (e.g. French-to-English and English-to-French) in the alignment process. We will implement symmetrisation after each step of model iteration. We believe that this improvement should increase the quality of resulting dictionary in fewer iteration steps. Computations will be executed in two threads – one for each direction.

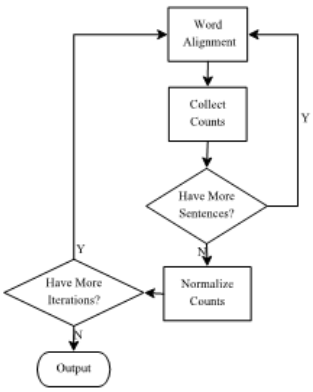

Figure 1a Giza++ algorithm

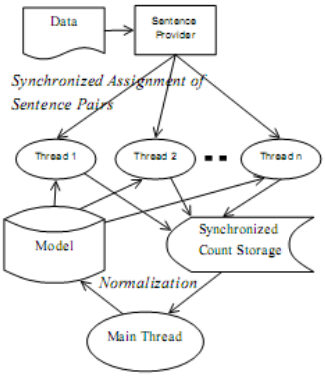

Figure 1b MGiza++ algorithm

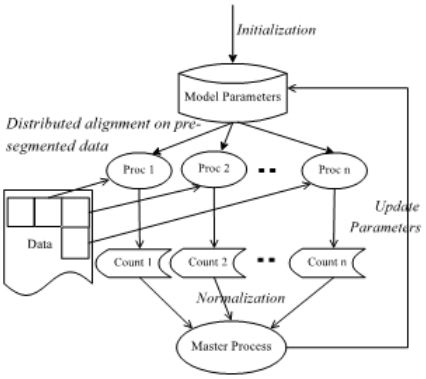

Figure 1c PGiza++ algorithm

[1] Och, J. Franz, Giza++: Training of statistical translation models, 2000

[2] Peter F. Brown, Stephan A. Della Pietra, Vincent J. Della Pietra, Robert L. Mercer, The Mathematics of Statistical Machine Translation: Parameter Estimation. Computational Linguistics, 1993

[3] Franz Josef Och, Herman Ney, A Systematic Comparison of Various Statistical Alignment Models, 2003

[4] Stephan Vogel, Herman Ney, Christoph Tillmann, HMM-based Word Alignment in Statistical Translation. In COLING '96: Yhe 16th International Conference on Computational Linguistics, Copenhagen, Denmark, 1996

[5] Qin Gao, Stephan Vogel, Parallel Implementation of Word Alignment Tool, 2008

[6] Qin Gao, Parallelizing the Training of Statistical Phrase-based Machine Translation

443 words