TR-S-0007

A Series of Experiments with Recursive Analysis by Analogy

Eddy Taillefer Yves Lepage

October 2000

Abstract

This report describes a set of experiments with analysis by analogy. The influence of three factors has been inspected: the use of restriction, a constraint, and a threshold of recursion. The restriction does not really influence the results, although it slightly increases the accuracy. The constraint does not increase the number of exact answers, but contributes to decrease the number of bad answers. Thus, the space searched can be efficiently reduced, yielding an acceleration of the method. The recursion threshold does not increase the number of exact answers, as was expected, and it was shown to have no influence after a maximum value.

Keywords

Analysis by analogy, analogy, experiments, measures.

Contents

Chapter 1

Introduction

Analysis by analogy is a technique which is based on a possible computational definition of analogy. An evaluation of direct analysis by analogy has already been done in [Lepage 99].

However, some questions remained to be answered. Firstly, would it be possible to accelerate the method by reducing the searched space? Secondly, could the use of a "cascade" method increase the coverage of analysis by analogy, *i.e.*, the number of sentences for which a parse can be obtained?

This report gives some answers to these questions based on results obtained by a series of experiments.

Chapter 2

Analysis by analogy

2.1 Principle

Analogy is the operation, noted $A : B = C : D$, by which a fourth object D is obtained from three given objects A, B, and C, drawing from their relative proportions. For instance: *to talk* : *I have talked* = *to work* : *I have worked.*

Anlalogy can work on sentences, and can be seen as contributing to the generation of new sentences. For instance:

How are you ? : *"How are you ?" she asked* = *Where are you ?* : x
$\Rightarrow$ x = *"Where are you ?" she asked*

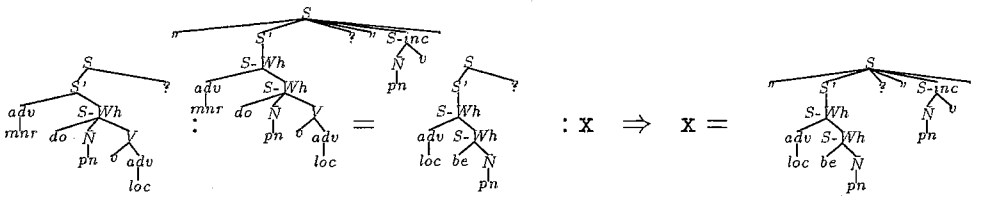
Analysis by analogy relies on the idea that, if there is analogy on some low representational level like that of syntactic classes:

mnr do pn v loc ? : *"mnr do pn v loc ?" pn v* = *loc be pn ?* : x
$\Rightarrow$ x = *"loc be pn ?" pn v*

then, there should be analogy on a higher representational level like that of structural representations (see Figure 2.1).

2.2 Direct analysis by analogy

In this method, as usual in natural language processing, the syntax of a sentence is represented by a tree. To compute the analysis by analogy of a new sentence, a tree-bank of sentences is needed. If the new sentence is



$$\begin{array}{l} mnr\ do \\ pn\ v\ loc? \end{array} : \begin{array}{l} "mnr\ do \\ pn\ v \\ loc?"\ pn \\ v. \end{array} = loc\ be\ pn? \quad : \mathtt{x} \Rightarrow \mathtt{x} = \begin{array}{l} "loc\ be\ pn?" \\ pn\ v. \end{array}$$

Figure 2.1: Principle of analysis by analogy.

inside the tree bank, then the corresponding tree (the linguistic structure) of the sentence is automatically known. But, if the sentence is outside the tree-bank, the tree-bank is used to search for three[1] sentences able to verify the analogy relationship with the input sentence.

If such sentences exist, then the corresponding three linguistic structures of the sentences are taken to form an analogy equation, which is solved to possibly deliver a linguistic structure (a tree). This tree is claimed to be a parse of the input sentence.

To verify and solve analogies, we use the algorithm proposed in [Lepage 96] or [Lepage & Iida 98]. This algorithm covers phenomena like prefixing, suffixing, and also parallel infixing. Analogies on linguistic structures are simply solved by applying the same algorithm to the parenthesised representation of the tree structures.

Figure 2.2 shows the direct method of analusis by analogy. In this figure, the set of pairs on the left, called the set of *models*, is the Cartesian product of the set of sentences. The set of pairs on the right is, in a similar way, the Cartesian product of the set of tree structures. Both sets, being in one-to-one correspondence, have the same cardinality, N^2, for a set of sentences of cardinality N. In our experiments, as $N = 5000$, this number will be twenty five million! Obviously, this has to be reduced by some means. We

[1] *A priori*, for a tree-bank of N elements there are $N \times (N-1) \times (N-2)$ possibilities of triples of sentences. This cubic search can be reduced by using some properties of analogies ($A : B = C : D \Leftrightarrow A : C = B : D$ or $A : B = C : D \Leftrightarrow D : C = B : A$). But this is not enough.

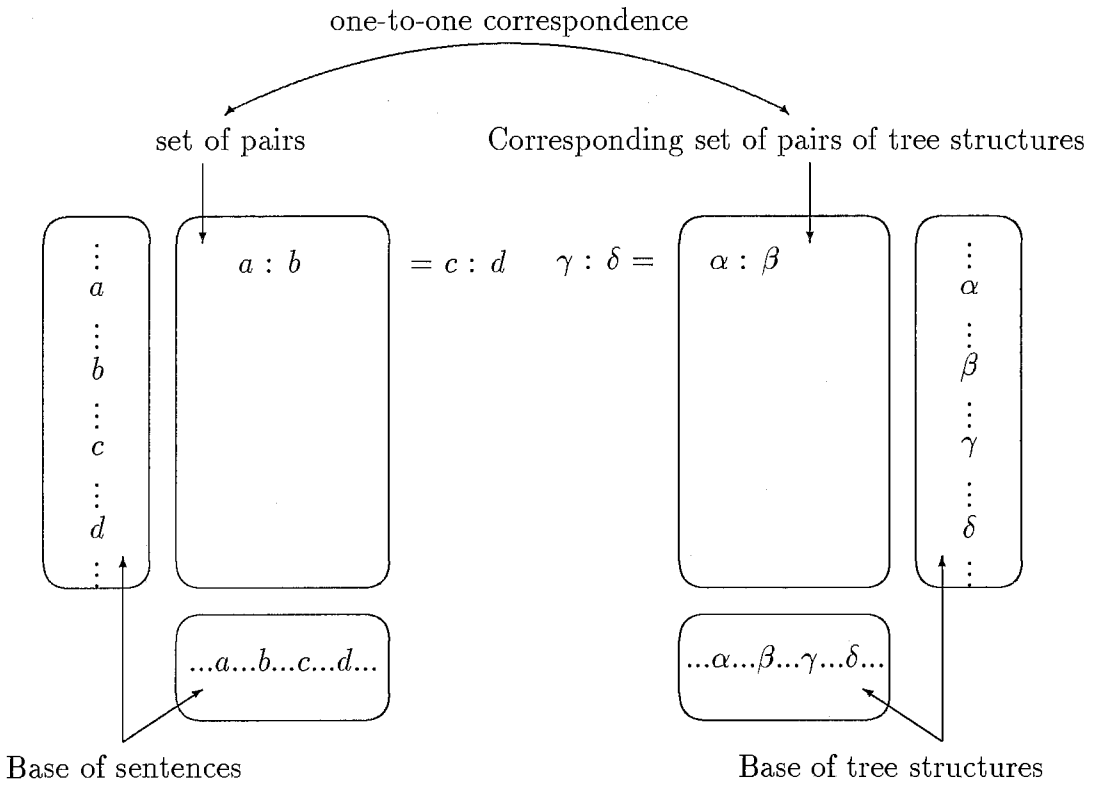


Figure 2.2: Analysis by analogy: the direct method.

will inspect this possibility by imposing a restriction and some constraint.

2.3 "Cascades" of analogies

The method described in the previous section consisted of:

- Firstly looking for two sentences in the tree-bank
- Then solving the analogy to get a third sentence. When the third sentence belongs to the tree-bank, the method applies.

In the case where the third sentence does not belong to the tree-bank, a recursive application of the method is possible so as to obtain a parse for this sentence and, one step back, a parse for the input sentence [Yvon 94]. This method, pictured in Figure 2.3, is called "cascades" of analogy. The elements in the pair (a, b) belongs to the sentence part of the tree-bank. In the first step, c is the given sentence, and d_1 the sentence delivered by solving the analogy.

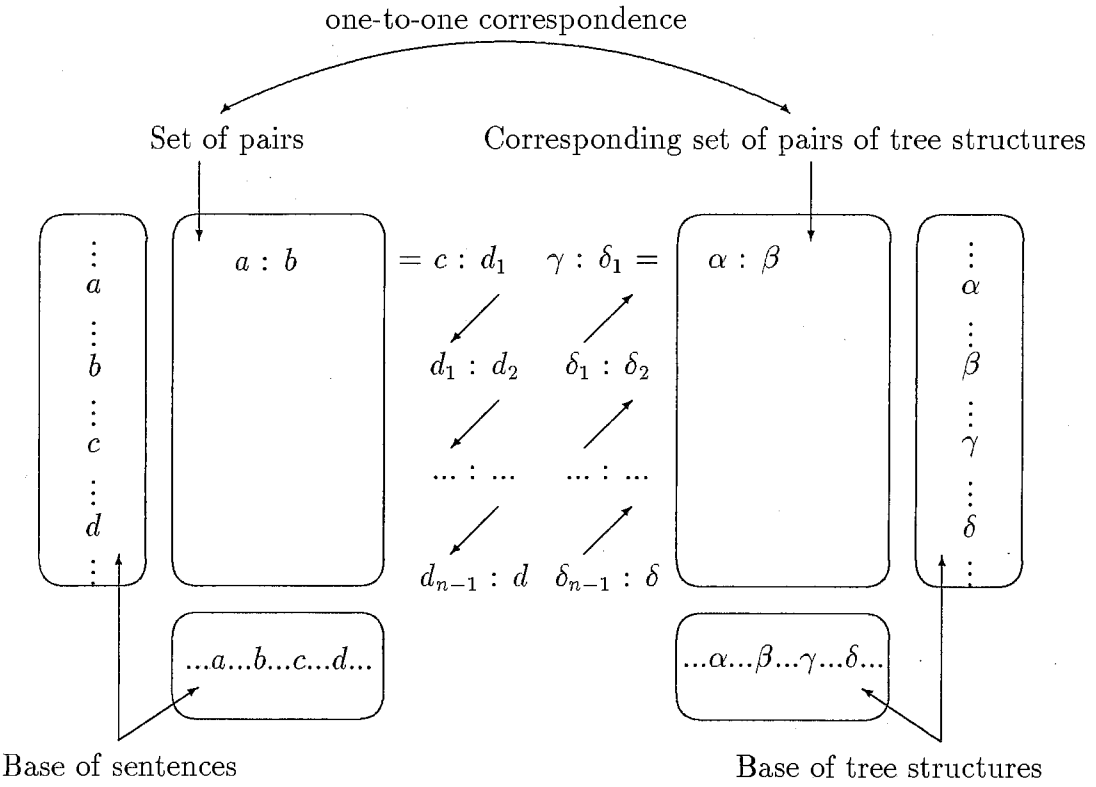


Figure 2.3: Analysis by analogy: the "cascade" method.

The number of recursive application of the method can be limited by a given threshold m. In the case of direct analysis by analogy, this threshold is 1. Intuitively, the use of a threshold greater than 1 should increase the number of sentences for which a parse can be obtained. The experiments

aim to confirm the increase in coverage of analysis by analogy, when using a recursive method. We will see that this is unfortunately not the case.

Although there is formally no reason that the length of $d_1, \ldots, d_{n-1}$ would decrease, this may have some advantage by ensuring the convergence of the recursive application of the method, because as a matter of fact, eventually d_n would tend to be the empty string ε. Following from a property of analogy ([Lepage 00]), we can get this decrease by imposing that the length of a would be greater than the length of b for all pairs (a, b) in the set of pairs. Our experiments will inspect this possibility, and we will look after a possible change in the results with and without this restriction.

Chapter 3

Experiments

3.1 Protocol

As we have just seen, three methods will be tested:

- analogy with recursion;
- analogy; with the restriction $\mid a \mid < \mid b \mid$ (where $\mid a \mid$ is the length of a)
- and analogy with a constraint.

To measure the coverage of these different settings, *i.e.* the number of sentences for which we get a parse, we used a tree-bank of 5000 sentences to parse 1553 other sentences in Japanese [Lepage 96]. The average tree size is about 11 nodes, with a standard deviation of about 6. This tree-bank uses dependency representations. In fact, because the tree-bank has 6553 sentences, we know the exact tree corresponding to every sentence we try to parse. This allows us to assess the quality of the results obtained by counting the number of sentences for which an exact parse is obtained, and also to compare the wroong parses with the exact parse by computing their distance.

3.1.1 Analogy with a restriction ($\mid a \mid > \mid b \mid$)

In accordance with the analogy properties, the restriction $\mid a \mid > \mid b \mid$ will be imposed to each pair of sentences in the set of models. This restriction aims to shrink the size of the set of models used, in order to reduce the runtime of the method.

3.1.2 Analogy with a constraint (parameter k)

A parametric constraint may also be introduced for selecting models. The introduction of a constraint aims to reduce the number of models. As a matter of fact, some general properties of analogy allow us to choose the pairs of sentences which are likely to best solve the analogy. These pairs only shall constitute the set of models.

This will reduce the space searched and the time spent, maybe at the expense of the quality of the results. To inspect the influence on the quality of the results, this constraint will be parameterised.

The constraint will correspond to a desired feature of the model (a, b) used in the analogy $a : b = c : d$ where: c is the given sentence, (a, b) the pair taken from the set of models, and d the solution of the analogy.

Several constraints may be proposed:

1. $\mid a \mid = k \times \mid b \mid$ with $0 < k < 1$.
2. $\mid a \mid = k \times (sim(a, b) + sim(a, c))$ and where sim is the similitude[1] This constraint comes from the following property of an analogy: $a : b = c : d \Rightarrow \mid a \mid \leq sim(a, b) + sim(a, c)$
3. $2 \times sim(a, b) / \mid a \mid + \mid b \mid \geq k$ with $0 < k < 1$. This constraint comes from the same property above.

In this report, we shall inspect only the third constraint, which seems to be the most promising one. However, in the long-term, the user should be able to choose among a set of constraints depending on his or her desired type of search.

3.1.3 Analogy with cascades (recursion threshold m)

We will also compute the quality in the function of the threshold m. We expect to have an increase in the quality of the results when m increases in the recursive method, until it reaches the quality of the direct method. We shall see, however, that in the current settings, this hope will be deceived.

[1]The similitude between two words (a and b) is the length of the longest sub-sequence that they have in common.

3.2 Evaluation

3.2.1 Measures

The quality of the results will be inspected by counting the number of parses obtained per sentence, and in particular, the number of times exact parses are obtained.

We also used a much finer evaluation, by computing the edit distance [Selkow 77] between each parse obtained and the exact parse. With this measure, not only labels are compared, but also any structural difference is counted. This measure allows us to characterise the quality of the parses and to present their distribution from the exact parses: the quality of a parse obtained is the inverse of its distance to the exact parse (a value of 0 means that the parse is equal to the exact parse).

A relative measure of the quality of the gparses obtained will also be done in the form of graphs. The abscissae scale will be the relative error, *i.e.* the distance to the exact parse divided by the size of the exact parse. For instance, an error of 0.5 means that a tree-banker would have to correct half of the parse obtained to get the right parse. The ordinate will be the number of parses obtained with this relative error. For instance, a value of 15 for a relative error of 0.5 means that the method delivered 15 parses containing as many errors as half the size of the correct parse. corresponding to the smallest one.

3.2.2 Implementation

A specific command (`statistic.t`) has been implemented in C in order to present all necessary statistical information, and other useful information. A list of the most important information which can be delivered by this command is listed hereafter.

- Number of sentences
- Number of parsed sentences
- Number of sentences not parsed
- Number of sentences correctly parsed
- Total number of parses obtained

- Number of exact parses obtained
- Mean number of parses obtained per sentence
- Mean number of pases obtained per sentence analysed
- Mean number of exact parses obtained per sentence correctly parsed
- Distance to exact parse for each parse obtained
- Distance to exact parse for each parse obtained divided by the size of the exact parse

In addition, this command gives some tables showing the distribution and the frequency of errors. It also generates the gnuplot files, which are necessary for drawing the different graphs given in the sequel of this report.

3.2.3 Application

To evaluate the quality of the constraint, the number of models will be computed for different values of the constraint parameter k.

To measure the effect of the "cascades of analogies" implementation (*cf.* Figure 2.3), a set of experiments will be done with different values of the threshold m. A representation of the number of exact parse functions to the threshold, will allow a comparison between the direct method and the "cascades" method.

Finally, it will be interesting to analyse the conjunct effects of the threshold and the constraint on the quality of the results.

3.3 Results

3.3.1 Size of the set of models

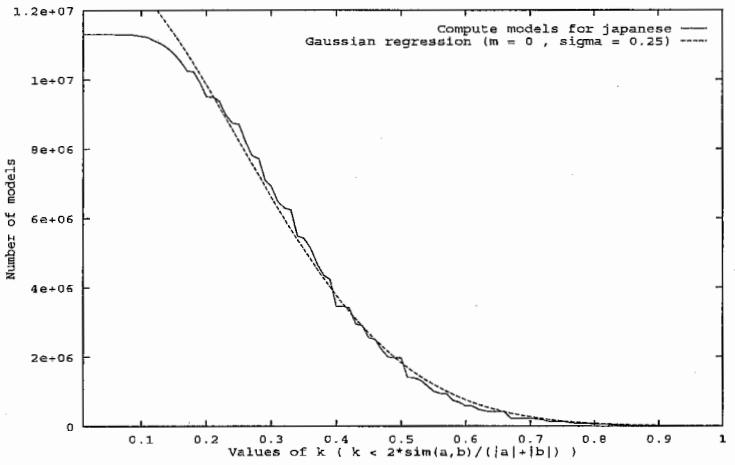
Influence of the restriction

The use of the restriction simply divides the size of set of models by 2. This alone is not negligeable. We shall see later that this considerable reduction in the isze of the set of models does not affect the quality of the result in a noticeable way.

Influence of the constraint (parameter k)

Firstly, we inspect the influence of the constraint k on the number of models.

The following graph gives the size of the models set, with the constraint $\frac{2 \times sim(a,b)}{|a|+|b|} \geq k$ for a hundred of values of k ($0 < k < 1$).



For the maximum number of models there are about eleven million. for k equal to a quarter about nine million models; for k equal to an half, two million models; and for three quarters, a hundred and forty thousand models. The maximum number of models which can be obtained is twelve and a half million. This value is reached when we compute the set of models without constraint and without counting the repetitive pairs.

We notice that the curve varies as a Gaussian curve. Hence, it follows the following equation:

$$f(x) = \frac{1}{\sqrt{2 \cdot \pi \cdot \sigma^2}} \cdot \exp\left(-\frac{1}{2} \cdot \frac{(x-\mu)^2}{\sigma^2}\right)$$

where, μ is the mean, σ the standard deviation. In our case the curve is centered, which means that $\mu = 0$. Parameters to obtain the Gaussian regression have been calculated, and this regression is dotted on the graph.

As a result, we can say that:

- the curve decreases when k increases, (this was the expected effect).
- the curve looks like a Gaussian
  - this means that the constraint k does not change the data distribution.
  - this confirms that the size of sentences has a random distribution.

We will now inspect the influence of the three factors on the results of analysis by analogy (*i.e.* the on the parses obtained) one after another.

3.3.2 Quality of the results

Influence of the restriction ($\mid a \mid > \mid b \mid$)

All the graphs in the sequel will look like the following graph. The abscissae are the relative error to the exact parse: the closer the bars to zero, the better the parses obtained. The ordinates give the number of parses obtained. Hence, the bar located on zero represents the number of exact parses obtained.

To inspect the influence of the restriction, we have drawn the graphs for several values of k, in two cases:

- with the restriction (Graph 3.1).
- without restriction (Graph 3.2).

It seems that there is no differences between the two cases. A fine analysis of the results shows that there are less bad answers with the restriction than without restriction, but in an insignificant number.

Thus, we can say that:

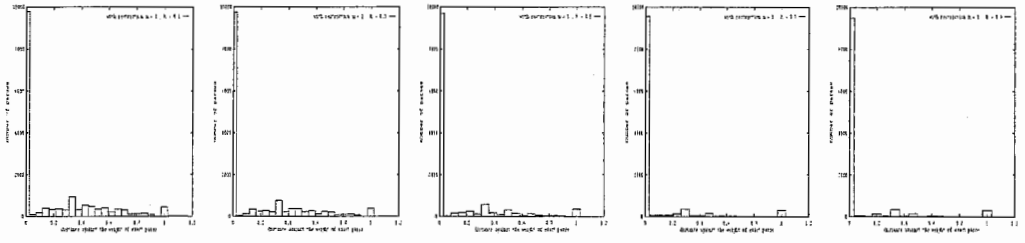


Figure 3.1: Number of models without restriction; abs.: distance to exact parse/size of exact parse; ord.: number of parses

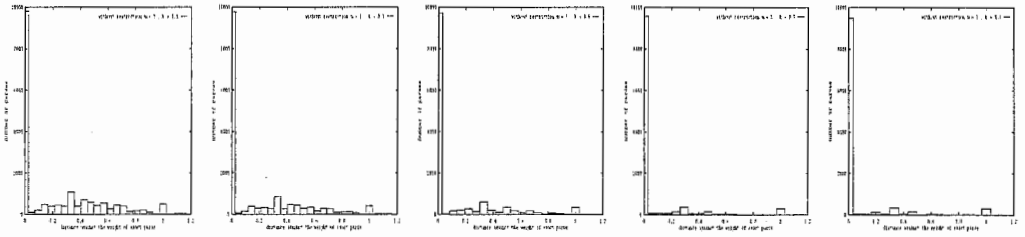


Figure 3.2: Number of models with restriction; abs.: distance to exact parse/size of exact parse; ord.: number of parses

- there are more exact parses than bad parses in both cases.
- the restriction seems good:
  - it does not modify the shape of the results.
  - it slightly reduces the number of bad answers.
  - and thus slightly increases the accuracy (number of exact parses / total number of parses).

The constraint (parameter k)

Now we will inspect the influence of the constraint k.

Quality for k=0.1, 0.3, 0.5, 0.7, 0.9. Below are the graphs obtained for different values of k.

We can notice that, as we expected, the number of bad answers decreases with the increase of k. Also there is an insignificant decrease of exact answers.

We can explain as follows:

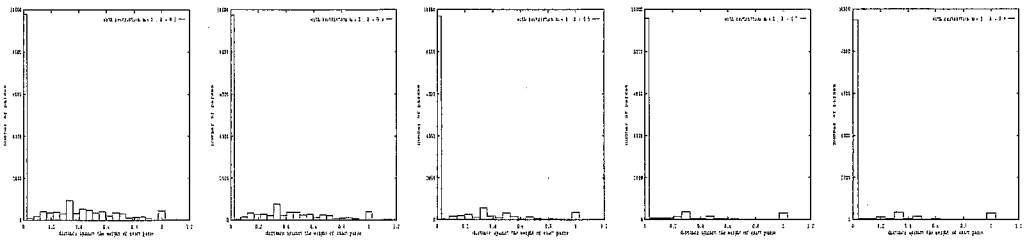


Figure 3.3: Influence of constraint; abs.: distance to exact parse/size of exact parse; ord.: number of parses

- generally the closer k becomes to 0 (resp. to 1), the more (resp. less) important the linguistic transformations.
- $\Rightarrow$ the closer k to 0, the more the bad answers, supposedly because the transformations become too loose.

Accuracy for k=0.1, 0.3, 0.5, 0.7, 0.9. It is also interesting to inspect the accuracy of the method. We have computed different measures on the previous results:

1. absolute ratio of sentences parsed and exactly parsed: an indicator of the recall of the method.
2. ratio of sentences exactly parsed to sentences parsed: an indicator of the precision per sentence.
3. ratio of exact parses to total number of parses: an indicator of the accuracy of the method.

In the first graph of Figure 3.4, we have the ratio of the sentences parsed, and the sentences exactly parsed. It seems that the ratio of sentences parsed converge with k on the ratio of sentences exactly parsed.

We can say that:

- when k increases:
  - parses tend to be exact.
  - absolute accuracy and precision per sentence increase.
  - however, the recall decreases.

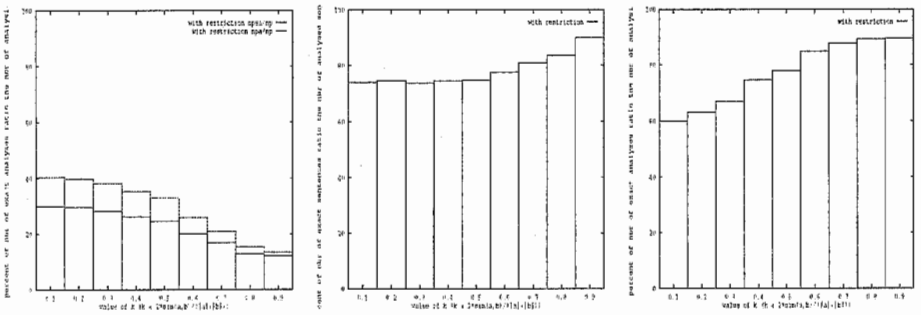


Figure 3.4: Accuracy of the method; abs.: k=0.1, 0.3, 0.5, 0.7, 0.9 ; ord.: number of parses

Influence of recursion (threshold m)

Here we expect the influence of the recursivity

In these experiments, we have fixed a value for k: 0.9. As in previous graphs, the abscissae give distance to exact parse / size of exact parse and the ordinates the number of parses.

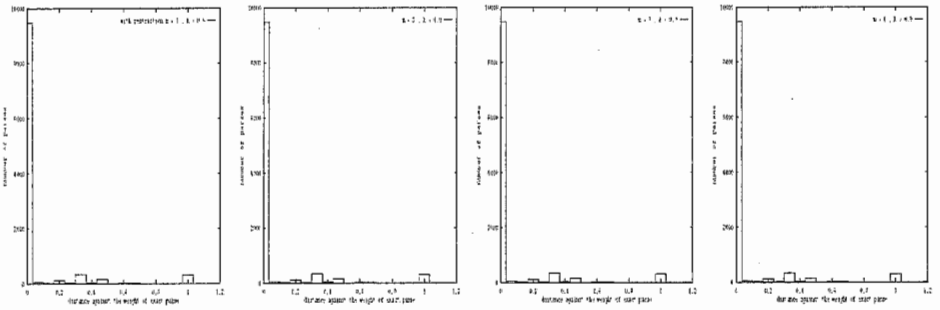


Figure 3.5: Recursivity; abs.: distance to exact parse / size of exact parse; ord.: number of parses

An analysis of theses results is as follows:

- no obvious change is noticed when m increases.
- for k=0.9, the same results for m=2, 3, 4, ... are obtained
- for k=0.8, the same results for m=3, 4, ... are obtained
- hence, it seems that the optimal threshold of recursion is reached almost immediately.

This general impression remains to be conformed by further experiments with different values of k.

Chapter 4

Conclusion

To summarise, my work in ATR was to automate the assessment of results of experiments with analysis by analogy. For that, I wrote aprogram which delivers a number of measures concerning the number of results obtained and their quality. I performed a range of exeperiments for which I measured the influence of three factors:

- a restriction
- a constraint
- a threshold of recursion

A summary of the analysis of teh results is as follows. There is a negative point:

- **recursion:** although an increase of exact answers was expected with the increase of m, no influence was observed after a relatively low value of m.

but there are two positive points:

- the **constraint** does not increase the number of exact answers, but contributes to decrease the number of bad answers, by eliminating them. Hence, the space search is efficiently reduced.
- as for the **restriction**, there is no obvious influence on the results, but it slightly increases the accuracy of the method.

We can thus conclude that the restriction can be applied without problem, and that a trade off between the constraint parameter k and the recursion threshold m will have to be found by future experiments.

Bibliography

[Lepage 96] Yves Lepage
Tesnière's structural syntax: notations for tree-banking using BoardEdit
ATR report TR-IT-0176, Kyoto, July 1996.

[Lepage 98] Yves Lepage
Solving Analogies on Words: an Algorithm
Proceedings of COLING-ACL'98, vol I, Montréal, August 1998, pp. 728–735.

[Lepage & Iida 98] Yves Lepage & 飯田仁
言語に依存しない早期終了型類推解決手法
言語処理学会第 4 回年次大会, 九州大学, 1998 年 3 月, pp. 266-269.

[Lepage 99] Yves Lepage
Open Set Experiments with Direct Analysis by Analogy
Proceedings of NLPRS-99, Beijing, November 1999, pp 363-368.

[Lepage 00] Yves Lepage
Languages of Analogical Strings
Proceedings of COLING 2000, vol 1, Saarbrücken, July-August 2000, pp. 488–494.

[Selkow 77] Stanley M. Selkow
The Tree-to-Tree Editing Problem
Information Processing Letters, Vol. 6, No. 6, December 1977, pp. 184-186.

[Yvon 94] François Yvon
Paradigmatic Cascades: a Linguistically Sound Model of Pronunci-

ation by Analogy
Proceedings of ACL-EACL-97, Madrid, 1994, pp 428-435.