Prolog models of classical approaches to MT

Harold SOMERS

Centre for Computational Linguistics
UMIST, PO Box 88
Manchester M60 1QD, England
Harold.Somers@umist.ac.uk

Abstract

This paper describes a number of "toy" MT systems written in Prolog, designed as programming exercises and illustrations of various approaches to MT. The systems include a dumb word-for-word system, DCG-based "transfer" system, an interlingua-based system with an LFG-like interface structure, a first-generation-like Russian-English system, an interactive system, and an implementation based on early example-based MT.

1. Introduction

There is something of a consensus in the recent literature on "MT in the Classroom" that distinguishes the teaching of MT to different target students, notably Computer Science (CS) and Computational Linguistics (CL) students vs. trainee translators (cf. Kenny & Way, 2001; Somers, 2001) among others. The present paper discusses the development of "toy" systems (cf. v. Hahn & Vertan, 2002), specifically written in Prolog, developed with the aim of providing relatively complex natural-language programs for students of Prolog and, secondarily, to provide demos of various approaches to MT. A number of different of languages are illustrated, usually for no other reason than variety. Some of the systems described here were developed with the help of students at UMIST.

An obvious and reasonable initial question might be why working on model implementations in Prolog might be useful, and to what sort of student? Way (2002) describes how his students undergo three years of training in CL including programming in various languages. he states that these students "may find themselves in the position of implementing changes to current systems, or indeed developing new ones" (p. 54). He describes exercises which involve Prolog programming within a framework based on Eurotra's E-framework (Bech & Nygaard, 1988), and it might be an interesting point of discussion for both Way and the current author whether Prolog is the most appropriate vehicle for this activity, given its lack of status as a programming language of choice for a real industrial application. We will leave this discussion for another time!

2. Similar approaches

The (relatively restricted) literature describing teaching of MT aimed at CL and CS students can be divided into those that use standard programming languages, and those that have developed more task-specific tools.

We have described Way's (2002) approach above. The students of v. Hahn & Vertan have arguably an even more difficult task, as they are left to devise on their own the basic architecture. From their discussion it appears that the only tools provided beforehand for the students is a corpus of test sentences, and a full-form lexicon in an XML-like formalism. No mention is made of the choice of programming language.

Amores (2002) describes Xepisteme, a tool for developing LFG-based MT systems. Amores stresses that the transparency of the system means that the user requires no programming skills theough he admits that an appropriate specification language must be learned, entailing the "usual difficulty" (p. 65), and indeed the examples of rules and so on that are shown suggest that this specification language is effectively a high-level task-specific programming language. Nevertheless, the tool also has a graphical interface that displays c- and f-structures, and the illustration is reminiscent to this author of the Metalshop tool that developers of the Metal system had available in the 1980s, as illustrated in Hutchins & Somers (1992: 264ff),and still available to users of the commercial *T1* system.

Sheremetyeva (2002) illustrates the use of the "developer tools" part of the APTrans system. It si not always clear from the paper how all the tools are used, the impression is that the ideal user will be a highly trained computational linguist. But no programming skills as such seem to be implied.

3. Systems that students can develop

3.1 A really dumb system

The simplest of MT models, if it can be called that, is one which translates word-for-word. This is very easy to implement in Prolog, which has good list-handling facilities. In fact, it is basically a one-predicate program (1) accompanied by a lexicon.

```
translate([],[]).
translate([S1|SRest],[T1|Trest]) :-
        tx(S1,T1),
translate(Srest,Trest).
tx(cat,chat).
tx(dog,chien).
tx(the,le).
tx(chased,chassa). % etc.
tx(X,X).
```

(1) *Word-for-word replacement model*

Obviously, this program will result in crude translations, but can serve as a starting point to bring home to students how inadequate this naïve approach is, in a way similar to the method described by Pérez-Ortiz & Forcada (2001), and can be used to invite students to identify what needs to be added. Even with a program of this simplicity, students can surprise you. One student who was brave enough to undertake translation between two foreign languages (French and German) needed much convincing that his rules mapping *le* onto *der* and *la* onto *die* left something to be desired.

3.2 A Prolog-like model: DCG-based transfer

Since we are programming in Prolog, an obvious model is one which takes best advantage of the most appropriate features of Prolog. That an MT system implemented in Prolog does not necessarily do this is well illustrated by McCord's (1989) LMT which includes code like that shown in (2), which is evidently a sequence of actions that could just as easily be programmed in any procedural language.

Prolog provides the DCG formalism (Pereira & Warren, 1980) which of course originally developed out of work on MT (cf. Colmerauer & Roussel, 1993). Although a very flexible formalism, it is usually associated with phrase-structure grammars, and this is the approach for our next model, also found in Gal et al. (1991:181ff). All other things being equal, DCGs can be run as parsers or generators, and so a system which links two DCGs will be reversible. This model consists of linking the source- (SL) and target-language (TL) DCGs with a transfer component which maps the SL phrase-structure trees onto those appropriate to be input to the TL grammar. The system is reversible if "Prolog escapes" in the DCG are used judiciously, for example for agreement and subcategorization. (3a) shows the top-level predicate, (3b) some example rules. (4) shows the transfer "formalism". Note that the DCG rules must be distinguished (e.g. `snp` vs `tnp`), but the structure is shared. The `tnp` rule shown here illustrates one way to do agreement.

```
nt(F,G,H,I,J,syn(Lab,B1,Mods),U,Z) :-
  copylabel(nt(F,G),nt(F1,G1)),
  bbrackets(B,U,V),
  preconj(PC,Mods,Mods1,V,W),
  ntbase(F1,G1,H,I,J,B,SynO,W,X),
  ntconj(F1,G1,F,G,H,I,J,PC,SynO,
            syn(Lab,B1,Mods1),X,Y),
  ebrackets(B1,Y,Z).
```

(2) *Example from LMT (McCord, 1989:37)*[1]

```
%(a)
translate(SText,TText) :-
                ss(Sstruc,Stext,[]),
                transfer(SStruc,TStruc),
                ts(TStruc,Ttext,[]).
%(b)
ss(s(NP,VP)) --> snp(NP), svp(VP).
tnp(np(Det,N,Adj)) -->
            tdet(Det), tn(N),tadj(Adj),
            {agree(Det,N,Adj)}.
```

(3) *DCG-based transfer system.*

The transfer component works top-down on the tree-structure, applying the most specific rules first, with a default of structure preservation. The examples show complex structural transfer for head-switching or "idioms" , and TL word selection based on category.

3.3 An LFG-like interlingua model

Another model that we present, again based on DCGs but this time without the explicit transfer phase, builds predicate–argument structures not

[1] The Prolog syntax has been harmonized with the syntax used elsewhere in this paper.

```
%(a)
transfer(vp(v(swim,SVF),
            pp(p(across),NPS)),
         vp(v(cruzar,TVF),
            np(NPT),adv(natando))) :-
         transfer(SVF,TVF),
         transfer(NPS,NPT).
%(b)
transfer(vp(v(get_up,SVF),adv(early)),
         vp(v(madrugar,TVF))) :-
         transfer(SVF,TVF).
%(c)
transfer(n(book),n(libro)).
transfer(v(book,SVF),n(reservar,TVF)) :-
                      transfer(SVF,TVF).
```

(4) *Transfer rules*

The transfer rules illustrate (4a) *swim across* → *cruzar natando*,(4b) *get up early* → *madrugar*, and (4c) *book* translated as *libro* (noun) or *reservar* (verb).The structures assume that verb features(`SVF`, `TVF` etc.) have been percolated, and are also subject to transfer.

unlike LFG f-structures, taking advantage of Prolog's straightforward unification (Gal et al., 1991:180). (5a) shows an NP rule that builds a feature structure, and (5b) the kind of structure that can be built. This is then transformed into a TL structure via rules (5c) which map SL attribute–value pairs which may be lexical or otherwise.

4. Return to Georgetown

The remaining systems to be described are more for demo purposes. The first[2] was developed more for fun and illustration than as a model of how to do MT in Prolog. I wanted to replicate the "first generation" design as described in sources such as Hutchins & Somers (1992:72), in which translation consists of morphological analysis, bilingual dictionary look-up and then "local reordering" which is a not particularly systematic attempt at handling structural and other divergences between the two languages. True to the spirit of the first generation, I decided to handle Russian–English, and, not knowing much Russian decided to work through that old stand-by *Teach Yourself Russian* (Fourman, 1943). The system covers the first 15 lessons, and has a vocabulary of 820 words.

The translation process begins with morphological analysis of the Russian input which, like in the original, is in transcription. This in turn introduces problems of ambiguous letter sequences which would evaporate if input

```
%(a)
% NP ->     det     n
%          ↑.num=↓ ↑=↓
%          ↑.det=↓ ↑=↓
np(np(Det,N),Fnp) --> det(Det,Fdet),

{unify([],[num:Num,det:Fdet],F1),
           extract(Fdet,num:Num)},
   n(N,Fn), {unify(F1,Fn,Fnp)}.
%(b)
[pred:eat(subj,opt(obj),
 tense:past,
 num:plur,
 subj:[pred:boy,
       num:sing,
       det:[num:sing]],
 obj: [pred:cake,
       num:plur,
       det:[num:plur]]
]
%(c)
xlex([pred:boy],[pred:ragazzo,gen:masc])
.
xlex(det:[num:X],det:[num:X,gen:Y]).
xlex(eat,mangiare).
```

(5) *LFG-like interlingua system.*

The rule in (5a) assumes a predicate `unify/3` *which succeeds if its third argument is a unification of its first two, and* `extract/2` *which finds the given attribute–value pair from a feature structure. (5b) shows the structure for* The boy ate the cakes*, and (5c) some rules for Italian, in which a feature for gender is introduced. The TL grammar will unify the NP structure and pick up the appropriate gender for the determiner. Notice that* `xlex/2` *rules can specify feature structures, attribute–value pairs or atoms.*

was in Cyrillic.[3] Russian morphology is quite rich, and the analysis creates a "word record" which indicates the position in the sentence, the word and underlying stem, and any grammatical information picked up from either the dictionary or the morphology.

The word is then looked up in the dictionary, which generally pairs the Russian and English words one-to-one. Sometimes however the dictionary lists alternatives, in which case an explicit lexical disambiguation procedure is triggered. Otherwise, ambiguities can have been introduced by the morphological component. This procedure, very much (it is hoped) in the spirit of the first-generation systems, can look at the details associated with any word to the right or left of the word to be disambiguated, or within a specific window. The system does not build any structure as such, so we have to rely on rules like the ones exemplified in (6).

[2] Nicknamed the "Georgetwon system", due to my inability to type that word correctly.

[3] I did develop a version of the system that allowed input in Cyrillic.

(6) a. *говорить* → *speak* if any word (later) in the sentence begins with a capital letter; else *say*
b. *клуб* → *cloud* if there is a preposition up to three words before it, and the sentence contains the verb *fly*; else *club*.
c. *моеи* → genitive of *моя* 'my' if followed by a genitive noun; else imperative of *мыть* 'wash'
d. the adjective endings *-им* and *-ым* can indicate instrumental singular masculine or neuter, or dative plural any gender: look for a preceding preposition which governs one or other case, or look for a following noun of the appropriate gender and case.

The "restructuring" phase is similarly fairly *ad hoc* in nature, and includes rules such as those illustrated in (7). Some are quite general, others specifically mention lexical items.

(7) a. The neuter nominative singular short form of an adjective is an adverb if the verb is not *be*
b. If there is no verb, translate *y* + N1(acc) N2(nom) as N1(nom) + *have* + N2(acc)
c. If there is no verb, insert be after the first pronoun, or before an adverb, or as the 2nd word.
d. Insert indefinite article on singular subject, unless verb is inverted.
e. *искать* + acc → *look for*
f. insert *with* if instrumental is not preceded by preposition

A final pass handles some trivial aspects of English morphology.

The system works well in the sense that it accurately reproduces the kind of translation quality that you might expect with this approach – see (8) for some examples (English output only). Many sentences are translated quite well. Many more are understandable but slightly odd, and there is a pervading Russian-ness about the output (read the examples with a cod Russian accent). And some are just gibberish. I am sorry to say I did not try it with *Душа готова, но плоть слаба*.[4]

a coast is not big.
I don't know to speak Russian.
I love Russian popular songs.
to our street be always many avtomobilej.
I didn't have the letter from my friend.
their conversation lasted whole hour.
we sent own baggage with the fast train.
hand over me, please, newspaper which lies by you.
why you don't lay their to the shelf.
this knigam is here not a place.
I want to the tea.
we saw some picture young artist, about who you read in the newspaper.
chji of the picture you saw yesterday to the exhibition.
my mother ask me to bring her didn't big carpet.
nashi friends came to a harbour to see off nas.
want cigarette?
in the evening after the work I rest.
give to the ill woman of the milk.
he didn't listen to the teacher's explanation.
my son doesn't understand this simple rule because he not listened to the teacher's explanation.
does your sister speak Russian ? no, she doesn't say, but she understands all, what she reads.
a brother's English newspaper is here.
from the coast of the sea to our house is not remote.
at-home is your father ?
do you wish the plate of the meat ?
a left sleeve of the new woollen dress too(much) is long and narrow.

(8) *Example translations from the Georgetwon* [sic] *system.*

5. Interactive system

Another area of interest was interactive MT, which "enables direct involvement on-line during the course of translation" (Hutchins & Somers, 1992:77). Obviously it would be a major undertaking to try to emulate the sophisticated interactive interfaces available with commercial systems, so the aim with this system was more to explore some issues which had become apparent at the time when I was associated with the Ntran project (Whitelock et al., 1986), and in my own even earlier work (Somers & Johnson, 1979). These issues are that the system must know not only when to interact, but also how. If the system interacts whenever it meets a problem, from the user's point of view the interactions may seem disjointed and illogical, since they will be following the system's "flight-path". So for example, it may do lexical disambiguation for various parts of the SL text, then "come back" to do some syntactic disambiguation, and then return for a third time to the "same" problem if there is a question of TL lexical choice. On top of this, there may be a tension between the system's

[4] 'The spirit is willing but the flesh is weak.'

need to translate the current text, and its "learning" function whereby it tries to update its dictionaries (and grammars?): the tension arises because of the potential conflict between the answer to the immediate question, and the answer to a more generic question.

Although the architecture of the system[5] is not the primary interest, some care was taken over it. The system has as its basic data structure a classical chart (Kaplan, 1973) with fairly simple feature bundles on the arcs. The system was developed for French–English translation with, as a reference corpus, a small set of sentences from a corpus of Swiss avalanche warning bulletins.[6] The user is assumed to have some knowledge of both languages though in fact since only interactions during analysis were developed, one could claim it to model a user who knows only the source language.[7] The system interacts with the user at three points: after morphological analysis, during syntactic analysis to resolve category ambiguities, and again later to resolve attachment ambiguities; interaction during lexical transfer was planned but unfortunately never developed.

The morphological analysis looks only at suffixes, and operates in a fairly standard rule-based manner. Since our focus is how an interactive system should work, unknown words are of particular interest: we want our model to make whatever inferences it can from morphological analysis. In the case of an unknown word, all segmentations are tried and then these must be presented to the user. (An alternative would be to keep the alternatives on the chart and wait to see if syntactic analysis ruled out the wrong solutions.) French morphology is quite rich, particularly in adjective and verb paradigms: this richness involves not so much a large range of inflections, but a large variety of interactions between stems and endings. Supposing, for example, that the word *basses* was not in the lexicon. According to our rules, there are 21 different interpretations of this string, assuming it is inflected (9).

[5] The development of the morphological interaction part was carried out by Gillian Chamberlain for her 1994 MSc dissertation.

[6] The corpus was obtained from colleagues at ISSCO, Geneva – cf. Bouillon & Boesefeldt (1991).

[7] The fact that the source language is French but the interactions are presented in English is a superficial anomaly.

stem	cat	gen	nmbr	para
basse	n	masc	pl	
basse	n	fem	pl	
basses	adj	masc	sing	1
basses	adj	masc	sing	2
basses	adj	masc	sing	6
basses	adj	masc	sing	11
basses	adj	masc	sing	15
basses	adj	masc	sing	18
basse	adj	masc	pl	1
basses	adj	masc	pl	2
basse	adj	masc	pl	3
basse	adj	masc	pl	6
basses	adj	masc	pl	18
basses	adj	fem	sing	18
bass	adj	fem	pl	1
bass	adj	fem	pl	2
basse	adj	fem	pl	3
basx	adj	fem	pl	4
bas	adj	fem	pl	6
bas	adj	fem	pl	15
basses	adj	fem	pl	18

(9) *21 possible interpretations of* basses.

From a user-interaction point of view, it would clearly be impractical to present all these alternatives in one menu, so the program works out how best to reduce the list by interacting feature by feature: Is it a noun or adjective? Is it masculine or feminine? Is it singular or plural? The correct answers (adjective, feminine, plural) in this case would reduce the 21 possible solutions to seven, at which point an interaction such as shown in (10) can take place.

```
basses
a. stem: bass, paradigm: 1
b. stem: bass, paradigm: 2
c. stem: basse, paradigm:3
d. stem: basx, paradigm: 4
e. stem: bas, paradigm: 6
f. stem: bas, paradigm: 15
g. stem: basses, paradigm: 18

enter letter corresponding to
choice
enter ? for help
enter * to display context
```

(10) *Interaction to disambiguate* basses.

The three last options are presented in all interactions. The "context" option shows the whole sentence, since the interpretation might depend on this. The "help" option in this case explains the paradigm codes, as in (11).

The first syntactic pass sometimes identifies sequences of words which are ambiguous, for example *sont tombés*, which could be copular + adjective 'are fallen' or perfect tense of *tomber*

```
************** HELP MENU ***************
KEY TO OPTIONS: INFLECTION OF ADJECTIVES

1. inflects like "GRAND":      -s -e -es
2. inflects like "GRIS":       -  -e -es
3. inflects like "ROUGE":      -s -  -s
4. inflects like "COURAGEU|X": -x -se -ses
6. inflects like "BO|N":       -s -ne -nes
15. inflects like "GROS":      -  -se -ses
18. invariable, e.g. compass points
****************************************
```

(11) *Help text for adjectives.*
This is canned text but note that only the paradigms mentioned in (10) are explained.

'have fallen'. It is very difficult to know how to ask a user to disambiguate this kind of thing, and we have taken the somewhat unsatisfactory step of presenting the two grammatical analyses more or less "raw" (12).

```
sont tombés
a.
vg([v(tomber,pl,perf(pres),masc,pos)])
b. vg([v(être(tombé),pl,pres,masc,pos)])

enter letter corresponding to choice
enter ? for help
enter * to display context
```

(12) *Interaction to disambiguate* sont tombés.

Of more interest is the second interaction after syntactic analysis, which deals with PP-attachment and coordination. The difficulty here, as we discovered with Ntran (see above) is to frame a question that makes sense even if the answer is "No". Whitelock et al. (1986) illustrate the difficulties in using "natural metalanguage" or "disambiguating paraphrases" to frame interactions (13).

Our solution is to take the lexical heads of the phrases to which the PP might attach and simply present them in a list. When the context is shown close by, this may be an effective method, at least for a user with some "feel" for linguistics. Note that apart from the word *attach*, no linguistic jargon is used (14).

The user can always elect to pass over any interaction. Because the system does not allow criss-crossing attachments, some attachments can be resolved automatically, as a result of other resolutions. For example, in the case in (14), if *des Alpes* is attached to *sont tombés*('fallen from the Alps'), then only options *a* and *f* would be valid choices.

Of all the toy systems, the interactive one is perhaps the least satisfactory as a model, though it does illustrate some difficult questions.

```
"I saw a man in the park"
  1. The action [saw] takes place [in
       the park]
  2. [a man] is [in the park]
"This module provides the interface to
the system"
  1. The action [provides] takes place
     [to the system]
  2. [the interface] is [to the system]

  1. The interface to the system is
     provided by this module.
  2. This module provides the system
     with an interface.
```

(13) *Examples of interaction*
From Whitelock et al. (1986:333). While the first (from Tomita, 1985) works well, with another text the result can be simply absurd. The problem with the third case is finding a set of rules that can correctly generate the paraphrases.

```
"60 à 80 cm de neige sont tombés de
samedi à mardi matin sur le versant
nord et la crête des Alpes au dessus de
2000 m."
Does the PP "au dessus de 2000 m"
attach to:
        a. tomber
        b. samedi
        c. matin
        d. versant
        e. crête
        f. Alpes
```

(14) *Example of PP attachment disambiguation*. [8]

6. Example-based MT

Our final model is a re-implementation of one of the first EBMT systems, the original ATR system (Sumita et al., 1990) handling Japanese adnominal noun phrases of the form A *no* B.[9] These structures are notoriously difficult to translate, the default rendering B *of* A is appropriate less than 40% of the time. Some examples are shown in (15) (from Sumita et al., 1990:207).

Like the original, our model handles translations in the domain of conference registration. It has a database of previously translated examples of A *no* B structures, in fact a subset of ATR's own corpus. The system has a

[8] The text reads '60 to 80 cm of snow fell from Saturday to Tuesday morning on the north face and the crest of the Alps above 2000m.'
[9] This system was programmed by Rachel Patterson for her 1994 MSc dissertation.

(15) a. *yooka no gogo* the afternoon of the 8th
b. *kaigi no sankairyoo* the application fee for the conference
c. *kyooto deno kaigi* the conference in Kyoto
d. *isshukan no yasumi* a week's holiday
e. *hoteru no goyoyaku* the hotel reservation
f. *mitsu no hoteru* three hotels

vocabulary of just over 300 words. The database contains 250 examples, stored as triples consisting of the Japanese text (romanized), the translation, and a coded indication of the target structure, e.g. `BofA`, `AB`, `BtoA`, and so on.

As in the ATR original, the input is matched against the database of examples, and the best match is used as a model for the translation. The matching procedure involves a distance measure based on proximity in a thesaurus. The thesaurus is a shallow (four-level) hierarchy of about 120 domain-specific primitives. A small part of it is shown in (16). All vocabulary items are identified with one thesaurus term, as illustrated in (17) (the first argument is the semantic marker).

```
root
..actions
   ....travelling
   ......travel,sightseeing,sport,activities
   ....booking
......application,registration,reservation,ca
       ncellation,reception,arrangement
   ......attendance
   ....study
   ......research
   ..objects
%etc.
```

(16) *Excerpt from semantic hierarchy*

```
x(sport, tenisu,  tennis).
x(reception, uerukamu, welcome).
x(arrangement, enjo, support).
x(arrangement, 'un-ei', steering).
x(study, kenkyuu, research).
x(research, kagaku, science).
```

(17) *Examples of dictionary entries*

The distance measure is defined as in (18),

$$(18)\quad d(I,E) = \sum \left(sd(I_i, E_i) \times w(I_i, E_i) \right)$$

where *sd* is the semantic distance between I_i and E_i, the corresponding words in the input and example, and *w* is a weighting which reflects the frequency with which the same pattern is used when I_i is translated as E_i. *sd* is given as the level of the most specific common abstraction of the two terms, divided by the depth of the thesaurus, always 4 in our case. *w* is defined as in (19).

$$(19)\quad w = \sum \left(\frac{|patt(I,E,j)|}{N(I,E)} \right)^2$$

where *patt*(*I*,*E*,*j*) is the number of examples where *I* is translated as *E* with pattern *j*, and *N*(*I*,*E*) the total number of examples where *I* is translated as *E*. The idea behind this weight is that if, when *I* is translated as *E*, a variety of patterns is used, then *I* should be less influential in the choice of translation pattern.[10] These weights can be precompiled.

Let us work through an actual example. Suppose we want to translate *rondon no ofisu* (lit. 'London ADN office'). The semantic feature associated with *rondon* is `city`, while *ofisu* is `branch`. Among the examples we look at are (20a–c). The calculations of the distance scores shown in (20).

```
(a) tookyoo no hoteru 'my hotel in Tokyo'      BinA
(c) oosaka no kaigi 'the Osaka conference'     AB
(b) nihon no daigaku 'a Japanese university'   ^AB
   sd(rondon,tookyoo)                  = 0.00
   sd(ofisu,hoteru)                    = 0.25
   w(tookyoo,Tokyo)                    = 1.00
   w(hoteru,hotel)                     = 0.28
   d = (0.00x1.00) + (0.25x0.28)       = 0.07
   sd(rondon,oosaka)                   = 0.00
   sd(ofisu,kaigi)                     = 1.00
   w(oosaka,Osaka)                     = 1.00
   w(kaigi,conference)                 = 0.15
   d = (0.00x1.00) + (1.00x0.15)       = 0.15
   sd(rondon,nihon)                    = 0.25
   sd(ofisu,daigaku)                   = 0.25
   w(nihon,Japanese)                   = 1.00
   w(daigaku,univ.)                    = 1.00
   d = (0.25x1.00) + (0.25x1.00)       = 0.50
```

(20) *Calculation of distance scores for* rondon no ofisu *and three examples*.

Based on the distance scores, (20a) is chosen as the best fit. The English side of the example is taken as the template, and the words *Tokyo* and *hotel* replaced by *London* and *office* (as given by the dictionary) to give *my office in London*. The system can also show the next best options, which in this case would be *the London office* and *a London office*.

There are a number of points that make the process slightly more complex however. The first is that for many words there are multiple translations: Japanese does not distinguish singular and plural, cf. *hoteru* in (15e,f). In the

[10] Strangely, in the original article, Sumita et al. multiply *sd* by *w*, which has the effect of *decreasing* the distance measure when there are varied patterns.

pattern `^AB`, the ^ symbol indicates adjectival form, so for example *nihon* may be *Japan* or *Japanese*. Thus, for the TL generation the translation patterns should really carry some grammatical information to guide the choice of surface form.

Another complication is that the form of the adnominal is also variable: in some examples the *no* particle is attached to a postposition, as in *kaiba madeno shatorubasu* 'shuttle bus to the conference site'. This is fairly easily handled by extending the distance calculation in (18) to include the adnominal.

The last example shows a case where the single Japanese word is rendered in English as a compound. Actually, this is not a big problem as long as the compound can be treated as a unit. More significantly, in the A *no* B construction in general the A and B can be noun *phrases*, not just simple nouns, as the examples in (21) illustrate.

(21) a. *50 nin hodo no guruupu* 'a group of about 50 members'
b. *sono kimpen no bijinesu hoteru* 'a business hotel in this neighbourhood'
c. *kaidan kikanchuu no sukejuuru* 'your schedule during the conference'

In this case, the distance measure has to be adapted to be able to compare single- and multi-word constructions. For example, the system should recognize that *kimpen no hoteru* is usefully similar to (21b).

A final (and realistic) difficulty is in handling inconsistency in the example set. In even our small example set, dates are translated in a number of different ways (e.g. *the 5th of August*, *March 8th*) and there are even examples of the same phrase with two different translations.

All of these are interesting problems which do give the student an insight into some of the pros and cons of EBMT.

7. Afterword

We have presented here a number of Prolog implementations of model systems. In fact, the author (and his students) have worked on one or two more, not reported here, including a French–English system of somewhat similar design to the Russian–English system described above, and (ironically, considering the history of Prolog – see above) a re-implementation of Météo. Students also had (at the time) limited access to Eurotra which, while not perhaps (intended as) a "toy" system certainly was written in Prolog!

References

Amores, J. Gabriel (2002) 'Teaching MT with Xepisteme', in *6th EAMT Workshop Teaching Machine Translation*, Manchester, pages 63–68.

Bech, Annelise and Anders Nygaard (1988) 'The E-Framework: A Formalism for Natural Language Processing', in *Coling Budapest: Proceedings of the 12th International Conference on Computational Linguistics*, Budapest, pages 36–39.

Bouillon, Pierrette and Katharina Boesefeldt (1991) 'Applying an experiemtnal MT system to a realistic problem', in *Machine Translation Summit III Proceedings*, Washington, DC, pages 45–49.

Chamberlain, Gillian T. (1994) *A model of an interactive machine translation system*. MSc Dissertation, Department of Language and Linguistics, UMIST.

Colmerauer, Alain and Philippe Roussel (1993) 'The Birth of Prolog', *SIGPLAN Notices* **28**.3, 37–52.

Fourman, Maximilian (1943) *Russian*, London: Teach Yourself Books.

Gal, Annie, Guy Lapalme, Patrick Saint-Dizier and Harold Somers (1991) *Prolog for Natural LanguageProcessing*, Chichester: John Wiley.

Hutchins, W. John and Harold L. Somers (1992) *An Introduction to Machine Translation*, London: Academic Press.

Kaplan, R.M. (1973) 'A general syntactic processor', in R. Rustin (ed.) *Natural Language* Processing, New York: Algorithmic Press, pages 193–241.

Kenny, Dorothy and Andy Way (2001) 'Teaching Machine Translation & Translation Technology: A Contrastive Study', in *MT Summit VIII Workshop on Teaching Machine Translation*, Santiago de Compostela, pages 13–17.

McCord, Michael C. (1989) 'Design of LMT: A Prolog-Based Machine Translation System', *Computational Linguistics* **15**, 33–52.

Patterson, Rachel M. (1994) *A Prolog simulation of the example-based machine translation experiment of A.T.R. Laboratories*. MSc Dissertation, Department of Language and Linguistics, UMIST.

Pereira, Fernando C.N. and David H.D. Warren (1980) 'Definite Clause Grammars for natural language analysis – a survey of the formalism and a comparison with Augmented Transition Networks', *Artificial Intelligence* **13**, 231–278.

Sheremetyeva, Svetlana (2002) 'An MT Learning Environment for Computational linguistics Students', in *6th EAMT Workshop Teaching Machine Translation*, Manchester, pages 79–87.

Somers, Harold (2001) ‘Three Perspectives on MT in the Classroom’, in *MT Summit VIII Workshop on Teaching Machine Translation*, Santiago de Compostela, pages 25–29.

Somers, H.L. and R.L. Johnson (1979) ‘PTOSYS: an interactive system for “understanding” texts using a dynamic strategy for creating and updating dictionary entries, in M. MacCafferty and K. Gray (eds) *The analysis of meaning: Informatics 5*, London: Aslib, pages 85–103.

Sumita, Eichiro, Hitoshi Iida and Hideo Kohyama (1990) ‘Translating with Examples: A New Approach to Machine Translation’, *The Third International Conference on Theoretical and Methodological Issues in Machine Translation of Natural Language*, Austin, Texas, pages 203–212.

Tomita, Masaru (1985) *An efficient context-free parsing algorithm for natural languages and its application*, PhD thesis, Carnegie Mellon University, Pittsburgh.

v. Hahn, Walther and Cristina Vertan (2002) ‘Architectures of “toy” systems for teaching Machine Translation’, in *6th EAMT Workshop Teaching Machine Translation*, Manchester, pages 69–77.

Way, Andy (2002) ‘Testing Students’ Understanding of Complex Transfer’, in *6th EAMT Workshop Teaching Machine Translation*, Manchester, pages 53–61.

Whitelock, P.J., M. McGee Wood, B.J. Chandler, N. Holden and H.J. Horsfall (1986) ‘Strategies for Interactive Machine Translation: the experience and implications of the UMIST Japanese project’, in *11th International Conference on Computational Linguistics: Proceedings of Coling ’86*, Bonn, pages 329–334.