Aspects of an empirical approach to language processing

Terence Lewis

Hook & Hatton Ltd, UK

This paper discusses practical MT systems, particularly the Dutch-English system developed by the author's own company. For the purposes of this paper, a practical MT system is one that is designed to meet specific translation requirements in a defined environment and has a built-in modifiability. In this context "built-in modifiability" basically refers to a possibility to add, modify and even remove translation rules and routines. Needless to say, the type of system under discussion is a rule-based one.

The essential strategy of the author's Dutch-English system is to retrieve and manipulate linguistic information contained in a database, or dictionary, in response to the string in the input file. The retrieved information is manipulated in accordance with a large number of grammatical and semantic rules. As the application of the rules is dependent upon the initial information delivered by the dictionary, it can only commence once the entire look-up phase is complete. Of necessity, the system under discussion involves a multi-pass approach.

The origin of this Dutch-English system was entirely practical. As translators of technical documents for the chemical industry, the author and his colleagues were struck by the amount of repetition and relatively small vocabulary in the documents passing through their hands. Indeed, many documents of a hundred pages of more were found to contain between one and two hundred words used recurrently. The initial objective was simply to find a way of automating the repetitive part of the translation process. However, things have moved on from there, and our company is now supplying MT output on a commercial basis to industrial customers who want quick, low-cost translations.

Given the marked differences in word order between Dutch and English and the inherent ambiguities of such everyday syntactic items as "zijn", "of" and "dat", it was soon realised that automatic look-up alone would do little to speed up and facilitate the translation process. A decision was made to develop an expandable system of rules which would use the information in the database to generate grammatically and semantically correct English sentences from the string contained in the Dutch input file. This undertaking was always seen as a "computer problem" to be solved with our programming skills and our knowledge of the Dutch and English languages. Whilst we obviously read the basic "MT literature" to gain an idea of what others were doing in what was to us a "new" field, we built up our application without any communication with the MT community. One of the disadvantages of this approach was that we fell into some of the pitfalls already visited by those legendary MT pioneers of the 1950's and 1960's. On the other hand, the adoption of a practical or empirical approach presented us with shifting perspectives. We have never become wedded to any particular theory or solution strategy; our sole, constant aim has been to generate English output for which our industrial customers are happy to pay. "Unified theory" is not a term likely to be discussed during our coffee breaks. If this approach has involved employing a mixed bag of programming techniques, no customer has yet complained about that. However, most of our customers do expect us to have their specialised terminology in our database.

Text A

This paper reports on the performance and crystallisation morphology of polyphenyl quinoxalines and Nylon-6 molecular composite material. The molecular complex formed by polyphenyl quinoxalines (PPQ) and Nylon-6 can bring about great changes in the wear resistance, hardness, heat resistance and the crystallisation morphology of the Nylon-6 substrate. The results of research in morphology and crystallisation dynamics show that PPQ plays a crystallisation-inducing role in the Nylon-6 substrate. Our results have not only provided a practical method for improving the properties of cast Nylon-6, but have also provided a possible new use for molecular combination.

Text B

In the known method the reaction is carried out in a homogenous solution of benzaldehyde in ethanol and a Raney nickel catalyst is used. By varying the benzaldehyde-ammonia ratio used, a maximum of 81 mol-% dibenzylamine is obtained, calculated with respect to the amount of converted benzaldehyde. The disadvantage of a Raney nickel catalyst is that even with relatively large amounts of catalyst relatively long reaction times are required to achieve satisfactory conversion. Furthermore, this catalyst is toxic and pyrophoric, so that it is less suitable for large-scale application.

Figure 9 - 1

Figure 9 - 1 shows the kind of output successfully generated by our system. Text A was written by a human translator; Text B is an unedited computer translation. Our pricing is based on the assumption that an average of 10 - 15 minutes post-editing is required to bring the computer translation up to publication quality. This should still yield our customers around 50% savings in translation costs. On the other hand, several large multinational companies are quite happy with the MT output for specific purposes and do no editing once they are satisfied that the translation conveys the basic information contained in the original. We recently used our program to deliver the translation of a 100-page military telecommunications specification within 48 hours; we know that our direct customer did no post-editing of the computer translation before forwarding it to the end user within his organisation. Costs and pricing have been mentioned at this stage of the paper because, in a real way, they are one of the sets of parameters within which the system has been developed.

The basic elements of our system are shown in Figure 9 - 2.

ELEMENTS OF TRANSLATION SYSTEM

- Automatic pre-editing module
- Extensive phrase dictionary
- Pattern recognition module
- Specialised dictionaries
- Common compound dictionary
- General dictionary
- Rule application module
- Problem-solving routines

Figure 9 - 2

The first stage in our translation process is to extract a list of the words occurring in a document, together with an indication of their frequency. This is currently done using low-cost third-party software, which is run prior to the translation program proper. The word list is then run through the translation program so that unknown words are identified at this stage. If there is time, the dictionaries are updated.

The automatic pre-editing module is designed to solve some basic problems before the main processes begin. For instance, a sub-module identifies and codes any Dutch surnames and Dutch and foreign place names in the document, so that the program will not, for example, attempt to translate "de Heer van den Berg" as "Mr of the Mountain" or the Norwegian town of "Bergen" as "Mountains". The tagging generated at this pre-translation stage is, in the case of place names, utilised in the disambiguation of the Dutch word "te". A sub-module termed "SPELSTAN" tries to bring some orthographic consistency to the Dutch text. This module is both useful and necessary in view of even educated people's erratic use of the numerous alternative spellings in the Dutch language. To give a practical example, it is not unknown to find such alternative spellings as "consequentie" and "konsekwentie" in consecutive sentences. This module also identifies and codes sentences and tags numbers, converting decimal points to commas where necessary. We have found that chances of successful parsing are increased by the provision of this information in advance of the main modules.

In our system, the next stage of the process involves the establishment of possible matches between sentences or parts of sentences in the input text and the contents of an extensive phrase or idiom dictionary. This part of the database contains idiomatic phrases, standard expressions which cannot be translated literally, and even complete sentences which have a standard or non-literal translation. After a successful match, the English translation of the phrase is coded as a noun or verbal phrase, written to the output file and passes through the rest of the program in a "sealed capsule". If the source text contains nothing but simple sentences or lists of words (e.g. parts lists), we can simply switch the phrase module off; the program will then run considerably faster.

Historically, our pattern matching module is a development of the phrase dictionary. One of its tasks is to bring together the separate components of an idiomatic phrase which may be separated in a sentence by intervening words. For example, in the sentence "hij stelt informatie ter beschikking" (literally translated as "he places information at the disposal"), the program identifies the relationship between "stelt" and "ter beschikking" so that the sentence is translated idiomatically as "he makes information available". The pattern matching module also takes care of the translation of idiomatic phrases that have a variable component. This module, which can bring about a significant improvement in the quality of the computer translation, is in fact implemented by a very simple "high-school level" programming technique.

The system was initially conceived to facilitate the translation of chemical engineering texts, so the capability to deal successfully with specialist terms is a key feature of our program. The specialised dictionaries are searched before the general dictionaries, and the order in which they are searched can be altered. For instance, if the source text is known to be a study of cardiovascular diseases, the biomedical dictionary goes at the top of the list. Or, if it is a telecommunications specification, the search routine may begin with the electrical engineering dictionary followed by the computer science module. Modularity is a key concept here. To provide the best results the system still needs a helping hand from the human user who actually determines the dictionary search sequence. Hopefully, future implementations will include some automatic decision-making on subject-matter leading to "soft dictionary selection", but we have not got there yet.

After searching the specialist dictionaries, the program performs look-up routines in the general dictionary. All the dictionary modules, specialist and general, contain syntactic and a varying amount of semantic information. Words that are ambiguous are identified as such in the dictionary and marked up for disambiguation in the rule application routine. The minimum information for

each entry is part-of-speech identification. Infrequently used words generally carry only this minimum level of information. On the other hand, terms frequently encountered in technical literature are accompanied by semantic categorisations.

The rule application module is the stage of the process where some degree of "intelligence" is brought into play. Each rule is applied sentence-by-sentence in a single pass through the text, although that single pass may involve left-to-right and right-to-left movement within the sentence. The rule searches for its corresponding tag or marker. If the sentence does not contain one, it moves onto the next sentence. At the end of the application of the particular rule, a rewind function sets the file position to the beginning of the file and the next rule is applied. Together with the pattern matching module, the rules are the brains of the program; they are what sets it apart from a limited but useful look-up program. There are rules for grammatical operations, word order resolution, syntactic disambiguation and semantic disambiguation.

The use of individual rules, each performing a single function, has enabled us to develop a permanently modifiable system. Every translation produced by the program is analysed for failures; every failure is recorded and becomes the subject of a new rule or the improvement of an existing rule. Let us illustrate this process with an example. As the program was initially developed to translate technical texts, we simply - indeed stupidly - failed to take into account the fact that the Dutch formal second person pronoun "U" (you) may have a third person singular verb in Dutch but is always translated into English with a second person verb , e.g. "U heeft" becomes "you have". Once we realised our oversight it took just a couple of hours' work to write a suitable conversion rule and slot it into place in the rule list.

The rule application module does not solve every problem. In particular, semantic disambiguation may require reference to "real world knowledge". The last module, which is only in its initial stage of development, contains a series of problem-solving routines, which attempt to resolve some of the questions left outstanding by the other modules. To give an example, the establishment of the appropriate meaning of the verb "uitvoeren", the possible translations of which include "export" and "implement", is handled in this module.

Although the program now appears to have the look of a well-designed system, it was developed by trial and error. We moved from look-up to boiler-plating, and from there to basic rules, adding first the phrase dictionary, next the pattern matching module and lastly the problem-solving routines. Leaving aside the individuals involved, the way the program has evolved owes a great deal to manufacturing and pricing developments in the computer industry. Without the advent of cheap RAM and faster disk access times, the phrase dictionary module would have involved search times so long as to make the program unusable. Even with a 33 MHz processor, the program now flies through hundreds of rules in a matter of minutes. This availability of low-cost computer power has significant implications for the development of language processing applications. Some of these are shown in Figure 9 - 3.

IMPLICATIONS OF LOW-COST COMPUTER POWER FOR MT

- Fewer memory constraints
- Greater permanent storage & faster access to it
- Lightning processor speeds!
- Possibility of "cottage MT development"

Figure 9 - 3

Nowadays many business users have as much computer power on their desks as their main frames delivered 15 years ago. The availability of cheap RAM releases developers from memory constraints. This provides the potential for sophisticated cross-referencing to large knowledge bases, which are stored on low-cost hard disks with capacities and access times that were the stuff of science fiction ten years ago. Even under MS-DOS, it is now possible to create a 32 MB virtual disk, with all the implications this has for the acceleration of search times. One practical outcome of this trend is that developers working in academic institutions and research establishments no longer have to beg time on the main frame to try out their programs; these can be designed for and run on the PC or workstation.

We are not the only company to take advantage of this revolutionary change in the computing environment. All over the world, in university departments and companies of varying sizes, people are working away on the development of both small-scale and large-scale language processing applications. The application may be a "letter writing assistant" or a program for translating nothing but avalanche bulletins, but the likelihood is that it was written on a PC or workstation for use on a PC or workstation. Coupled with low-cost computing power, there is a whole new generation of graduate systems analysts and programmers who've been brought up on high-level languages such as C, and have the mental agility to tackle anything from designing intensive care monitoring systems to ATM's that understand natural language. The removal of frontiers all over Europe will result in a growing demand for quick translation at regional or local level (e.g. in courts, police stations, customs posts, etc.). It is likely that the organisations concerned will turn to their own computer people to deliver practical MT systems to meet this need, and these programmers will probably be working in C rather than LISP.

The fact that the proprietors of some of the "historical" MT systems have rewritten their applications in C would suggest that C and C++ will be the languages of choice for the development of practical MT applications. The "struct" (data structure) and the possibility of having arrays of different types as members of the "struct" and of nesting a "struct" within another structure template open up exciting opportunities for references to knowledge bases and the real world. The flexibility of C also allows a program originally written to handle a tiny domain to be expanded into a more general-purpose application as and when the need arises.

The legendary TAUM METEO and the ISSCO program for translating avalanche bulletins are two of the better known examples of practical, restricted-domain systems. The RUMP system, which translates from Russian to Ukrainian, is another typical example of an application developed to meet a practical requirement that arose from a political decision, namely the Law on Languages which makes Ukrainian the official language throughout the predominantly Russian-speaking Ukraine. It can run on a IBM/286 PC and is routinely used to translate court documents and police information. While the raw output may require substantial post-editing, the program ensures compliance with the law at a much lower cost than that of employing a large number of human translators. Open borders and the increasing movement of people and goods will require easy-to-operate, practical MT applications. In a few years time, traffic policemen will probably be faxing foreign driving licences and vehicle documents back to headquarters for automatic translation. Or they will use handheld translation systems combined with portable scanners for on-the-spot translation.

It is well-known that the best examples of MT output are provided by systems designed for use within restricted domains. The domain does not, however, need to be that restricted. The example of output generated by our system shown in Figure 9 - 1 is a chemical text, the type of text for which the system was originally designed. Figure 9 - 4 shows a computer translation in the field of artificial intelligence, also produced by our system.

TAUMES is a prototype second generation expert system that operates in the domain of a Metro network. It assists the human expert in the search for and the evaluation of route plans for the maintenance of the rail traffic in a disturbed environment. TAUMES has both the rules of thumb that are used by the expert in those situations and incidental topological and functional representations of the Metro network. After each solved problem a learning mechanism is activated which ensures that the problem-solving method shifts gradually to a simple selection and in this way increases the performance of the system.

Figure 9 - 4

The unedited output shown in Figure 9 - 4 has generally been judged acceptable for information purposes. The switch from one domain to another has not impaired the quality of the translation. However, if the new domain were that of popular journalism, the program would probably generate barely intelligible gibberish. This is because we have so far only written rules to deal with the types of grammatical structures commonly encountered in technical documents. In theory, it would be possible to write all the rules, pattern-matching and problem-solving routines needed to translate an Amsterdam comedian's patter or Curaçao patois, but our customers are hardly likely to want us to do that and we are only concerned with meeting their practical requirements.

The main conclusion which we have drawn from our MT experiences is that acceptable results can be achieved when the developer designs the program as a solution to a practical problem. Indeed, many of the developments in computer technology are spin-offs from concrete military and space applications, so this approach has proven merits. The advantage of the empirical or pragmatic way of "doing MT" is that the designer is not hampered by theory. Our use of the phrase dictionary at an early stage of the process with the encapsulation of "ready-made" translations probably runs counter to all the theory of the last twenty years, but it works. Our multi-pass strategy may be pedestrian but it enables quite complicated Dutch sentences, say with two subordinate clauses, to be parsed and translated correctly. As we are neither propounding a theory nor even selling software, none of this matters. Our customers can upload a file containing a 30-page document at three in the afternoon and download a usable translation at nine o'clock the following morning at 30% of the cost of a human translation. That's all that matters. Furthermore, by going for a modular solution and with the flexibility afforded by a high-level programming language, we have developed a system that can be readily adapted to meet specific requirements. We would suggest that academic institutions and research organisations pursuing MT development might have something to learn from our approach.