[From: *Overcoming the language barrier*, 3-6 May 1977, vol.1 (München: Verlag Dokumentation, 1977)]

# SLUNT (SPOKEN LANGUAGES UNIVERSAL NUMERIC TRANSLATION)

**W. Goshawke**

**Pearl Assurance Company Limited**

## Abstract

Results of computer translation research have hitherto been disappointing. Most researchers use free-input systems, whereby texts are fed into computers as authors wrote them, with ambiguities and idioms unaltered. Better results are obtainable by restricted-input systems, whereby such parts of texts which are ambiguous or complex are rewritten (retaining their original meaning) in forms which programs can accurately translate, before being input. SLUNT is such a system, designed for writers wanting their work translated into several foreign languages.

SLUNT translations take place through Number Language, an intermediate language consisting of numerals. Number Language statements are unambiguous and, although designed for computers, are decipherable without using a computer by anyone knowing the rules of Number Language and having a Number Language dictionary for a language which he speaks. All messages are sent in Number Language. Number Language numbers will be accurately defined in all spoken languages.

SLUNT programming is easy. Programmers need only know their mother tongue and the rules of Number Language, which are simple. Progressive and easily distinguished grades of SLUNT are planned, starting with simple sentences and limited vocabularies, and moving to complex sentences and larger or specialised vocabularies. SLUNT is valid for all spoken languages.

## 1. A new approach to automatic translation

Millions of dollars and hundreds of man years have been spent in recent decades on research into computer translation of spoken languages. So little has been achieved by this that many responsible people sincerely feel that the task is an impossible one. Consequently, any new attempt to solve the problem is apt to be met with scepticism. This scepticism is only justified if the new researcher is tackling the problem in the old way. This is definitely not the case with Slunt. Slunt has many characteristics not present in the methods of other researchers and is intended for the writer who wants his work translated into a number of different spoken languages, and who is willing to co-operate by rewriting any sentences not readily translatable by computer.

## 2. Pre-editing

2.1. One important feature of Slunt is that it is a restricted-input system, whereas earlier researchers have usually preferred a free-input system. With a free-input system the text is fed into the computer just as the author wrote it, without any modification whatever. This causes difficulties of various kinds, three of which, (a) ambiguities, (b) idioms and (c) unusual or complex sentence structures, are discussed in the following paragraphs.

2.2. (a) There are many ambiguities in spoken languages, where a sentence can have more than one different meaning. The human reader can usually resolve the ambiguity by examining the context. Attempts have been made to examine contexts by computer, but this is a very formidable task. It increases enormously not only the complexity and length of programs but also the complexity and size of the dictionary necessary to be retained in store. Moreover, the results cannot be wholly relied on. Some ambiguities can only be resolved by reference to the author himself. Slunt programs do not examine contexts.

2.3. (b) Writers frequently use idioms which cannot be translated literally. For example, English business men sometimes say they have "caught a cold" when they have taken a risk which has turned out to their disadvantage. It is possible to program a computer to treat this as "made an unwise decision". Some idioms are so common that it is necessary to include them in programs in this way. To include all idioms is impossible, as new idioms are constantly being created. The method suggested for Slunt is that acceptable idioms are listed and made known to users of the programs. All idioms in the list are treated as idioms when they occur, except when otherwise stated. Thus "catching a cold", if on the list of idioms, would never be taken as meaning having an illness, unless this were specifically stated.

2.4.(c) Sentences can be long or short, simple or complex. With a free-input system the programmer tries to make his program capable of dealing with every type of sentence. If the writer of the material to be translated uses very unusual or complex sentence forms, the program may fail. With Slunt, only acceptable sentence forms are input.

2.5.The above are the most common causes of failure of the free-input method. With Slunt they are avoided. All texts are edited before being input. When they are ambiguous, the ambiguity is resolved, if necessary by reference to the author. Each idiom which the programs cannot handle is rewritten in a form which has the same meaning but which is acceptable to the computer. Unusual and complex sentences which cannot be handled are rewritten in a simpler and acceptable form, but retain their original meaning.

## 3. Universality and accuracy

3.1. Unlike other methods, Slunt is applicable to all spoken languages. Most researchers have confined their work to translation of one specific language into another specific language, such as Russian into French, or Chinese into English. Such projects, even when successful, have only a limited usefulness. Slunt uses an intermediate language, and the intention is that all messages shall be sent and received in this. Moreover, all messages in the intermediate language are unambiguous, and the intermediate language version of the message, once despatched, becomes the official version. This system reduces enormously the amount of programming to be done and minimises the educational qualifications needed by the programmer.

3.2. The intermediate language used in Slunt is called Number Language and consists entirely of numerals. Messages sent in Number Language can be handled by any digital computer in the world, whatever alphabet it uses. English words such as man, book, office and so on are represented in Number Language by Number Language numbers or NLnumbers. Bearing in mind that the Number Language version of the message is the official one, it is intended that all NLnumbers shall be defined accurately in all spoken languages which use Slunt. It is well known that an English word may not have an exact equivalent in French or German and vice versa. This means that an NLnumber may be equivalent to a single word in one spoken language, but may need several words to define it in other languages. There is also another feature of spoken languages to allow for. A word in English or any other language may have three or four different meanings. If so, there must be three or four different NL numbers to express these meanings. Accurate definitions in the dictionary are fundamental to the system. It follows from this that the Number Language message can be made to express the precise meaning of the writer, and that the message can be translated accurately into any other language by computer, provided efficient programs are available. In the event of any doubt, the Number Language message can be deciphered without the aid of a computer by anyone with a knowledge of the rules of Number Language and a Slunt dictionary for a language which he speaks.

## 4. Stages of vocabulary

Because of its importance in the system and of the paramount need for accuracy, the creation of the dictionary will be a very lengthy task. It is proposed to undertake it in stages, starting with the simplest vocabulary and moving to less common words, bearing in mind the needs of users. Each stage will be clearly defined and numbered. About a hundred words are needed for testing programs. For the next stage, a very important one, when programmers exchange elementary messages with programmers in other countries, a vocabulary of about 500 words is needed. For business correspondence a vocabulary of about 2000 words is proposed. At a later stage there will be special vocabularies for different branches of science and technology and for other purposes. The limitation of the vocabulary for a particular purpose reduces the scope for ambiguities, and is very valuable. It also has the advantage that if users can identify which stage of vocabulary they and their correspondents are using, they can limit the storage necessary for the dictionary and reduce the dictionary search time. This does not mean that rigid restriction of vocabulary must be practised at all times. If on occasion additional words are necessary and are used, a list of these can be appended to the message.

## 5. Grades of difficulty

5.1. The writing of programs is also a lengthy task and is also graded, in this case according to difficulty. At all levels accurate and grammatical translations are obtained. The lowest grade has only simple sentences and later grades have progressively more complex ones. This feature is particularly valuable for researchers. It enables them to know what grade they have reached compared with other users and researchers. Thus at a very early period they can exchange simple messages in simple language with correspondents in other countries through computer translation. Not only does this give realism to the research, but it enables them to test programs of a simple nature in real life conditions before embarking on more complex ones. The grading of Slunt is made possible by one of the most powerful features of Number Language, the Statement Code.

5.2. Every statement in Number Language contains a Statement Code, and this shows clearly the nature of the statement. There are many types of statement. Some are very simple, consisting of a subject and a verb. Others are more complex. Each different type of statement has a different Statement Code. When a Number Language text is input for translation into a spoken language, the program tests each statement for its Statement Code to ascertain whether the program can translate it. Thus no attempt is made to translate a passage if the program is not versatile enough to handle it. Similarly, when a spoken language text is input for translation, a Statement Code has to be input immediately preceding each sentence for the same reason. At present, the Statement Code for each sentence in the spoken language has to be determined by inspection. At a later stage it is proposed to write programs which will allot Statement Codes by computer. It is expected that these will be successful in the great majority of cases. Where a Statement Code cannot be allotted automatically, this will usually mean that editing of the text is necessary.

## *6.* Slunt programming

6.1. Slunt programming is easy. The English programmer has to write programs for translating Number Language into English and for translating English into Number Language. The French programmer writes programs for translating French into Number Language and Number Language into French; the German programmer for German into Number Language and Number Language into German; and so on. The programmer has to know the grammar of his own language and the rules of Number Language and needs to know nothing of other spoken languages.

6.2. The rules of Number Language have a similar function to that of the grammar of spoken languages, but are not identical with the grammar of any spoken language. Spoken languages abound with irregularities and idiomatic features. The aim in devising the rules of Number Language has been to be as logical and consistent as possible. The English programmer will find that many everyday expressions in English cannot be translated literally into Number Language because they are idiomatic. This is especially the case with sentences involving prepositions. The same will also apply to German and French and to all other spoken languages. Nevertheless the programming is simple. It is just a question of applying the rules.

## 7. The Slunt package

7.1. A package has been produced and is available which translates simple English sentences into German by Slunt. It includes four translation programs and a miscellaneous file which contains dictionaries and material suitable for translation. Two of the programs (E002/3) translate English into Number Language, another (E001) translates Number Language into English, and the fourth (G001) Number Language into German. The package only handles sentences with Statement Code 0001, the simplest of all the Statement Codes, which indicates that the statement includes a subject and a verb.

7.2. The programs not only carry out the translations, but also provide detailed explanations of all the programming methods, and it is intended that they shall be used as prototypes which can be modified so as to produce similar programs for other spoken languages, or to produce programs for translating English or German sentences which are more complicated.

7.3. The programming language used in the package is Slunt Cobol, which is a very elementary subset of Cobol. This means that it can be easily adapted for use on any computer which uses Cobol or PL/1. The package is stored in the Archives of the University of Essex, Colchester, Essex, England, and is readily available to users all over the world. Copies of the package have already been sent to a considerable number of people in various countries.

7.4. The fact that Cobol is the programming language used in the package must not be taken to imply the opinion that Cobol is the most suitable programming language for Slunt programming. Many other programming languages may be just as good or better.

## 8. Natural Language Translation Specialist Group

8.1. It is important that the maximum possible co-operation be established between users and researchers of Slunt in all countries. Programs and dictionaries created by one researcher can be used by others working on the same spoken language. Workers in different countries who are aware of each others' existence can begin exchanging messages in Number Language at an early stage, and can so test their methods in practical conditions. By this means standard practices can be established, duplication of effort can be avoided and greater rapidity of progress can *be* achieved.

8.2. With these objects in view, a Natural Language Translation Specialist Group of the British Computer Society has been formed. Anyone interested in computer translation is eligible to join the Group, whether involved in Slunt or not, whether resident in Britain or not and whether a member of the British Computer Society or not. A number of Slunt enthusiasts from various countries are members, and replies to a recent survey show that Slunt research is being undertaken or contemplated in seven different spoken languages.

## 9. Who can help with Slunt?

9.1. Many universities are already engaged in a variety of kinds of literary and linguistic research involving computers. Many have created dictionaries. Such universities are well equipped to engage in Slunt research, and it is hoped that many will do so. Valuable work can also be done in schools and colleges, in commercial undertakings and by individuals working alone. A number of wellwishers have expressed their willingness to donate free computer time for Slunt research.

*9.2.* It is particularly desirable that Slunt research should begin in as many countries as possible, so that Slunt can be shown to be valid for all spoken languages. Every new spoken language for which Slunt programs are written will benefit users in every country.

## 10. Where to write for more details of Slunt

A much more thorough description of Slunt is to be found in an unpublished paper entitled "Spoken Languages Universal Numeric Translation (Slunt)", which is obtainable by writing to the author, Walter Goshawke at 68 Harrington Road, Bexleyheath, Kent DA7 4UW, England.

## APPENDIX

11. A summary of certain of the more elementary rules of Number Language and an example of a sentence coded in Number Language are given in this appendix. An example of the output of the package mentioned above is also provided.

### RULES

### 12. NLstatements, NLunits and NLnumbers

12.1. The characters of Number Language are the ten digits 0, 1, 2, 3, 4, 5, 6, 7, 8 and 9 and always appear in 10-digit units called Number Language units or NLunits. Any of the ten digits may appear more than once in an NLunit, Number Language statements or NLstatements are always made up of NLunits.

12.2. There are three kinds of NLunit, and all three kinds appear in every Nlstatement. Type 1 Units give general information about the structure of the NLstatement. Every NLstatement begins with a Type 1 Unit which includes as its four final digits the Statement Code of the NLstatement. Type 2 Units give precise details of the structure of the NLstatement, identifying the subject, verb, direct object and so on. Type 3 Units usually have *a* dictionary meaning but sometimes have other functions such as indicating tense. When a Type 3 Unit has a dictionary meaning this occupies the first eight digits of the NLunit and is called a Number Language number or NLnumber.

12.3. The seventh and eighth digits of the eight-digit NLnumber indicate which part of speech the word corresponding to the NLnumber is. Thus 01 is a verb, 05 is a noun, and so on. In Number Language the NLnumber for a noun is unchanged whether the noun is singular or plural or whether it is the subject, or the direct or indirect object of the sentence. Similarly, the NLnumber for a verb is unchanged whatever its person, tense or mood. The differences which occur in many spoken languages in nouns, verbs and other parts of speech are indicated in Number Language by NLunits specially designed for the purpose.

## 13. The tenses of the verb in Number Language

Immediately following a Type 3 Unit containing a verb, there is always a Type 3 Unit indicating the tense of that verb. The digits of the latter are as follows.

| | English examples |
|---|---|
| Digits 1-2, always 88 | |
| Digit 3 active-passive | |
| 0 active | I ask |
| 1 passive | I am asked |
| Digit 4 tense-type | |
| 0 complete action | I wrote |
| 1 incomplete action | I was writing |
| 2 repeated complete action | I usually wrote |
| 3 repeated incomplete action | I was usually writing |

| Digit 5 tense-code | English examples |
|---|---|
| 3 past-past | I had eaten |
| 4 past | I have eaten |
| 5 present | I eat |
| *6* future | I shall eat |
| 7 future-future | I shall have eaten |

Digit 6 affirmative-negative

0 affirmative
1 negative

Digits 7-10, zeroes

It will be seen that there are 20 tenses in the active indicative mood, as each of the tense-types in digit 4 can be associated with each of the tense-codes in digit 5.

## 14. Examples of simple Statement codes

The final four digits of the first Type 1 Unit of an NLstatement are the Statement Code and indicate the type of statement which follows. Examples of three simple Statement Codes are given below.

| Statement Code | Nature of Statement | English example |
|---|---|---|
| 0001 | Subject and verb | The man writes |
| 0003 | Subject, verb and direct object | He will be sending my letters |
| 0005 | Subject, verb, direct object and indirect object | The mother gave the books to the father |

## 15. An example of coding

15.1. The man comes.

A simple example of coding in Number Language is now given. The sentence "The man comes" appears in Number Language as follows.

| Type 1 Unit | Type 2 Units | | Type 3 Units | | | |
|---|---|---|---|---|---|---|
| | Noun<br>The man | Verb<br>comes | man | the<br>(singular) | to come | (present<br>tense) |
| 9903070001 | 5004050000 | 5006070000 | 0311000500 | 0537101700 | 0111000100 | 8800500000 |
| 01 | 02 | 03 | 04 | 05 | 06 | 07 |

For the sake of clarity the 10-digit units are numbered (1) to (07) in the above illustration, but this numbering, although used, is not part of the Number Language statement.

15.2. The Type 1 Unit indicates (a) by its first 2 digits (99) that a new statement has commenced, (b) by the Statement Code shown in its final 4 digits (0001) that the statement contains one noun and one verb and that there are no further Type 1 Units, (c) by its third and fourth digits (03) that the final Type 2 Unit is the third and (d) by its fifth and sixth digits (07) that the final unit of the statement is the seventh.

15.3. The Type 2 Units indicate which Type 3 Units are associated with the noun and the verb respectively. In unit (2) the digits 500405 indicate that the units (04) and (05) are associated with the noun. In unit (03) the digits 500607 indicate that units (06) and (07) are associated with the verb.

15.4. All the Type 3 Units in the statement are dictionary units except unit (07) which indicates the tense of the verb. In order to alter the tense of the statement it is only necessary to alter unit (07), the rest of the statement remaining unchanged. The NLnumber for the verb "to come" is the first 8 digits of unit (06), (01110001). This NLnumber for the verb never changes whatever the tense, person or number.

15.5. Similarly the NLnumber for the noun "man" is the first 8 digits of unit (04), (03110005). This NLnumber never alters whether the noun is singular or plural or whether the noun is the subject, the object or the indirect object of the statement. Unit (05) in this statement includes the NLnumber for "the" in the singular (05371017). The NLnumber for "the" in the plural is 05371027.

## 16. Output from the Slunt package

16.1. Figures 1 and 2 illustrate output from the Slunt package. Figure 1 shows printed output from Program E002, which translates English sentences into Number Language statements. The punched card output, produced at the same time, was used as input for Program G001, which translates Number Language statements into German sentences.

16.2. Figure 2 shows printed output from Program G001, which has translated the Number Language statements shown in Figure 1 into German. Figures 1 and 2 taken together illustrate an example of English sentences being translated into German sentences by means of Slunt.

```
01 YOU(INFORMAL-SINGULAR) DRESSED TUESDAY.
903070001 5004040000 5005070000 8511211800 0157000100 8800400000
559009400
01 HE DID NOT DRESS TUESDAY.
903070001 5004040000 5005070000 8501311800 0157000100 8800410000
559009400
01 SHE DRESSES.
903060001 5004040000 5005060000 8502311800 0157000100 8800500000
01 WE SHALL DRESS TUESDAY.
903070001 5004040000 5005070000 8503121800 0157000100 8800600000
559009400
01 YOU(FORMAL-PLURAL) WILL NOT DRESS.
903060001 5004040000 5005060000 8503221800 0157000100 8800610000
01 YOU(INFORMAL-PLURAL) WILL HAVE DRESSED.
903060001 5004040000 5005060000 8513221800 0157000100 8800700000
01 THEY WILL HAVE NOT DRESSED.
903060001 5004040000 5005060000 8503321800 0157000100 8800710000
01 COLD WINE COMES.
903080001 5004060000 5007080000 0598500500 0109000300 0500011700
111000100 8800500000
01 SOME COLD WATER COMES.
903080001 5004060000 5007080000 0581000500 0109000300 0500001700
111000100 8800500000
01 SOME COLD DRINKS COME.
903080001 5004060000 5007080000 0158000500 0109000300 0500002700
111000100 8800500000
01 MY YOUNG WOMAN COMES QUICKLY.
903090001 5004060000 5007090000 0605000500 0619000300 8501111700
111000100 8800500000 0428000400
01 THE YOUNG WOMEN WRITE QUICKLY.
903090001 5004060000 5007090000 0605000500 0619000300 0537102700
612000100 8800500000 0428000400
01 THE YOUNG MAN IS QUICK.
903080001 5004060000 5007080000 0311000500 0619000300 0537101700
428000100 8800500000
01 THE YOUNG CHILD ARRIVED TUESDAY.
903090001 5004060000 5007090000 0103000500 0619000300 0537101700
037500100 8800400000 0559009400
01 THE YOUNG WOMAN HAD USUALLY BEEN WRITING.
903080001 5004060000 5007080000 0605000500 0619000300 0537101700
612000100 8803300000
01 MY YOUNG CHILDREN WASH QUICKLY.
903090001 5004060000 5007090000 0103000500 0619000300 8501112700
578500100 8800500000 0428000400
01 MY YOUNG CHILD WRITES QUICKLY.
903090001 5004060000 5007090000 0103000500 0619000300 8501111700
612000100 8800500000 0428000400
01 A NEW ANSWER WILL ARRIVE.
903080001 5004060000 5007080000 0033000500 0355000300 0000001700
037500100 8800600000
01 A MAN WILL HAVE NOT WASHED.
903070001 5004050000 5006070000 0311000500 0000001700 0578500100
800710000
```

```
 9903070001 5004040000 5005070000 8511211800 0157000100 8800400000
 0559009400
0001 AM DIENSTAG HAST DU DICH ANGEKLEIDET.
 9903070001 5004040000 5005070000 8501311800 0157000100 8800410000
 0559009400
0001 AM DIENSTAG HAT ER SICH NICHT ANGEKLEIDET.
 9903060001 5004040000 5005060000 8502311800 0157000100 8800500000
0001 SIE KLEIDET SICH AN.
 9903070001 5004040000 5005070000 8503121800 0157000100 8800600000
 0559009400
0001 AM DIENSTAG WERDEN WIR UNS ANKLEIDEN.
 9903060001 5004040000 5005060000 8503221800 0157000100 8800610000
0001 SIE WERDEN SICH NICHT ANKLEIDEN.
 9903060001 5004040000 5005060000 8513221800 0157000100 8800700000
0001 IHR WERDET EUCH ANGEKLEIDET HABEN.
 9903060001 5004040000 5005060000 8503321800 0157000100 8800710000
0001 SIE WERDEN SICH NICHT ANGEKLEIDET HABEN.
 9903080001 5004060000 5007080000 0598500500 0109000300 0500011700
 0111000100 8800500000
0001 KALTER WEIN KOMMT.
 9903080001 5004060000 5007080000 0581000500 0109000300 0500001700
 0111000100 8800500000
0001 ETWAS KALTES WASSER KOMMT.
 9903080001 5004060000 5007080000 0158000500 0109000300 0500002700
 0111000100 8800500000
0001 EINIGE KALTE GETRAENKE KOMMEN.
 9903090001 5004060000 5007090000 0605000500 0619000300 8501111700
 0111000100 8800500000 0428000400
0001 MEINE JUNGE FRAU KOMMT SCHNELL.
 9903090001 5004060000 5007090000 0605000500 0619000300 0537102700
 0612000100 8800500000 0428000400
0001 DIE JUNGEN FRAUEN SCHREIBEN SCHNELL.
 9903080001 5004060000 5007080000 0311000500 0619000300 0537101700
 0428000100 8800500000
0001 DER JUNGE MANN IST SCHNELL.
 9903090001 5004060000 5007090000 0103000500 0619000300 0537101700
 0037500100 8800400000 0559009400
0001 AM DIENSTAG IST DAS JUNGE KIND ANGEKOMMEN.
 9903080001 5004060000 5007080000 0605000500 0619000300 0537101700
 0612000100 8803300000
0001 DIE JUNGE FRAU WAR GEWOEHNLICH BEIM SCHREIBEN GEWESEN.
 9903090001 5004060000 5007090000 0103000500 0619000300 8501112700
 0578500100 8800500000 0428000400
0001 MEINE JUNGEN KINDER WASCHEN SICH SCHNELL.
 9903090001 5004060000 5007090000 0103000500 0619000300 8501111700
 0612000100 8800500000 0428000400
0001 MEIN JUNGES KIND SCHREIBT SCHNELL.
 9903060001 5004060000 5007080000 0033000500 0355000300 0000001700
 0037500100 8800600000
0001 EINE NEUE ANTWORT WIRD ANKOMMEN.
 9903070001 5004050000 5006070000 0311000500 0000001700 0578500100
 8800710000
0001 EIN MANN WIRD SICH NICHT GEWASCHEN HABEN.
```