[*Proceedings of a Workshop on Machine Translation*, July 1990, UMIST]

INTERACTIVE TRANSLATION USING QUASI LOGICAL FORMS

Hiyan Alshawi and David Carter
SRI International Cambridge Research Centre
23 Millers Yard, Mill Lane
Cambridge CB2 1RQ, U.K.

1 Introduction

This paper describes work in progress on employing a general purpose natural language processing system, the Core Language Engine (CLE), to an interactive translation application. The CLE is a system for deriving Logical Form (LF) representations of natural language which are capable of supporting reasoning. The initial development of the CLE included substantial domain independent coverage of English syntax and semantics. The output of the linguistic analysis stages of CLE processing consists of structures at the level of Quasi Logical Form (QLF), a superset of LF having additional constructs for representing contextually determined aspects of interpretation.

For use in knowledge base systems, QLFs undergo quantifier scoping and reference resolution to produce one or more fully resolved LFs. However, an experimental project under way at SRI Cambridge Research Centre and the Swedish Institute of Computer Science uses QLFs directly. In this project, the CLE is being adapted to perform as a prototype Bilingual Conversation Interpreter (BCI) which would allow communication through typed text between two humans using different languages. The choice of languages for the prototype system is English and Swedish. Input sentences are analysed as far as the QLF level, and then, instead of further ambiguity resolution, undergo transfer into another QLF having constants and predicates corresponding to word senses in another language. The transfer rules used in this process correspond to a certain kind of meaning postulate. An output sentence is then generated from the target language QLF, using exactly the same linguistic data as is used for analysis of that language. When it is necessary, for correct translation, to resolve an ambiguity present at QLF level, the system will interact with either the source or the target language user to make the necessary decision.

QLFs were selected as the appropriate level for transfer because they are far enough removed from surface linguistic form to provide the flexibility required by cross-linguistic differences. On the other hand, the linguistic, unification-based processing involved in creating them can be carried out efficiently and without the need to reason about the domain or context. The purpose of the prototype BCI project is to attempt to demonstrate that QLF-level transfer is the way to take maximum advantage both of the linguistic knowledge processed by the CLE and of the reasoning capabilities and monolingual knowledge of the users of the system.

2 Levels of Representation in the CLE

As mentioned above, our approach to translation is based on transfer at the QLF level of representation. In this section we explain how QLF fits into the overall architecture of the CLE and in the following section we discuss the reasons for choosing it for interactive dialogue translation.

When the CLE is being used as an interface to a computerized information system (e.g. a database system), its purpose is to derive a Logical Form (LF) representation giving the truth conditions of an utterance input by a user. Logical Form (LF) is based on first order predicate logic with generalized quantifiers and some higher order extensions (Alshawi and van Eijck, 1989). For example, a possible LF for *She met a friend of John* is:

```
[dcl,
 quant(exists,
      X,
      [and,[friend,X],[friend_of,john1,X]],
      quant(exists.E,[event,E],
           [past,[meet,E,mary1,X]]))]
```

In this notation quantified formulae consist of a generalized quantifier, a variable, a restriction and a scope; square brackets are used for the application of predicates and operators to their arguments. To arrive at such LF representations, a number of intermediate levels of representation are produced by successive modular components.

2.1 CLE Processing Phases

A coarse view of the CLE architecture is that it consists of a linguistic analysis phase followed by a contextual interpretation phase. The output of the first phase is a set of QLF analyses of a sentence, while the output of the second is an RQLF (resolved QLF) representation of the interpretation of an utterance:

Sentence —LINGUISTIC ANALYSIS→ *QLFs* —CONTEXTUAL INTERPRETATION→ *RQLF.*

Deriving the LF from the RQLF is then a simple formal mapping which removes the information in the RQLF that is not concerned with truth conditions.

Linguistic analysis and contextual interpretation each consist of several subphases. For analysis these are:

- segmentation/orthography
- morphological analysis
- syntactic analysis (parsing)
- (compositional) semantic analysis

Apart from the first, these analysis subphases are based on the unification grammar paradigm using declarative bidirectional rules (REFERENCES). For example, syntactic analysis uses phrase structure production rules with categories consisting of feature specifications, and results in syntactic analysis trees with such categories at the nodes.

QLF representations resulting from the semantic analysis subphase differ in the predicates (senses) they select for words, and in the attachment decisions they correspond to. Thus several QLFs may potentially be produced for one sentence. However, they are neutral with respect to the choice of referents for pronouns and definite descriptions, and relations implied by compound nouns and ellipsis. They are also neutral with respect to other ambiguities corresponding to alternative scoping of quantifiers and operators and to the collective/distributive distinction and the referential/attributive distinction. The QLF is thus the level of representation encoding the results of compositional linguistic analysis independently of contextually sensitive aspects of understanding. These aspects are addressed by the contextual interpretation phase which has the following subphases:

- quantifier scoping
- reference resolution
- plausibility constraint filtering

A quantifier scoping mechanism for the CLE has been described by Moran (1988) and a later version by Moran and Pereira (to appear). Reference resolution and the application of plausibility constraints is discussed in Alshawi (1990).

2.2 The QLF Language

The QLF language is a superset of the LF language containing additional expressions corresponding, for example, to unresolved anaphors. More specifically, there are two additional term constructs (anaphoric terms and quantified terms), and one additional formula construct (anaphoric formulae):

`a_term(`*Category,Entity Var, Restriction*`)`.
`q_term(` *Category, Entity Var, Restriction*`)`.
`a_form(`*Category,Predicate Var,Restriction*`)`.

These QLF constructs contain linguistic information in the category and logical (truth-conditional) information in the restriction, itself a QLF formula binding the variable. Details of the use of these expressions in representing a wide range of natural language constructions are given in Alshawi (1990). A QLF from which the LF displayed earlier could have been derived is:

```
[meet,
   q_term(<t=quant,n=sing,l=ex>,E,[event,E]),
   a_term(<t=ref ,p=pro, l=she ,n=sing, a=<>>,
          Y,
```

```
        [female,Y]),
  q_term(<t=quant,n=sing,l=a>,
          X,
          a_form(<t =pred,p=genit >,
             R,
             [and,[friend.X],[R,john,X]])])])).
```

in which categories are shown as a list of feature-value specifications.

The RQLF representation of an utterance includes all the information from the QLF, together with the resolutions of QLF constructs made during the contextual interpretation phase. For example, the referent of an `a_term` is unified with the `a_term` variable.

Some constraints on plausibility can be applied at the QLF level before a full interpretation has been derived. This is because most of the predicate-argument structure of an utterance has been determined at that point, allowing, in particular, the application of sortal constraints expected by predicates of their arguments. Sortal constraints cut down on structural (e.g. attachment) ambiguity, and on word sense ambiguity, the latter being particularly important for the translation application in the context of large vocabularies. CLE sortal constraints are defined in terms of an extensible sort hierarchy. The constraints are applied and combined by unification, using a compile-time encoding of sorts (Mellish 1988). The sort checking mechanism can take into account non-local constraints on sorts arising from grammatical control relations expressed through shared variables at QLF since the constraints on all occurrences of a variable are checked for consistency.

2.3 Generation

Generation of linguistic expressions in the CLE always proceeds via the QLF level. Since the rules used during the analysis phase are declarative and bidirectional, these are also used for generation. (To achieve computationally efficient analysis and generation, the rules are pre-compiled in different ways for application in the two directions.) Generation uses the semantic-head driven algorithm (Shieber *et al,* 1989) for which the grammar and lexicon are pre-processed to produce the transitive closure of the semantic-head relation. Syntactic rules are composed with semantic analysis rules, as are morphological rules with their semantic counterparts.

Generation is also possible from the RQLF level, first by mapping the RQLF to a suitable QLF. If the mapping applied replaces referential anaphors (referential definite descriptions, and pronouns that are not of the bound variable type) with their referents, then this results in paraphrases showing aspects of reference resolution. This paraphrasing of the results of full interpretation can also show the effects of ellipsis resolution (resolved `a_forms`). There are, however, other aspects of resolution, such as quantifier scoping and distributivity, which are difficult or awkward to exhibit in paraphrases except in special cases.

3 Which Level for Transfer?

The representational structures on which transfer operates must contain information corresponding to several linguistic levels; for example, predicate-argument structures alone cannot guarantee that old/new or topic/comment structuring is preserved, and purely syntactic structures alone cannot encode the semantic distinctions needed for correct translation of ambiguous lexical items. This is a constraint on the *content* of the transferred structures.

However, there are also constraints on the *form* of the structures used. Each type of information must be stored in a clearly defined place, and in particular, one type must be selected as the “organizing” level. This is the level that determines the outer structure of the representation used. For example, McCord’s (1989) LMT system transfers structures organized around surface syntax but annotated with deep grammatical and some semantic information, while in the BCI’s QLF structures are organized around logical predicate-argument relations while containing some syntactic information at well-defined places. If the transfer rules operate recursively, and if maximising compositionality is viewed as important, then the choice of organizing level is crucial.

Thus both form and content need to be considered in designing a representation for transfer. Syntactic phrase structure trees are inappropriate because they are too closely related to the surface form of a source language: the transformations required for mapping between differing syntax trees result in complex transfer rules. For example, the following all mean "I like to eat sweets" in Swedish:

```
Jag äter gärna godis.
Jag äter godis gärna.
Gärna äter jag godis.
Godis äter jag gärna.
```

and transfer rules will proliferate if we need to match all possible orderings of subject (jag, “I”), object (godis, “sweets”) and adverb (gärna, “gladly”). In addition, the transformation from the Swedish equivalent of “eat gladly” to the syntactically quite different English “like to eat” is also more easily handled at a logical level. The predicate-argument structure required for the application of sortal restrictions is also absent from syntactic analyses; this is a matter of both form and content.

Sortal restrictions can be applied at the LF level, but at this level the form of noun phrase descriptions used and also information on topicalization is no longer present. Vagueness present in specifier phrases will also have been removed by an explicit commitment to a particular quantifier. It is also well-known that producing completely resolved interpretations can require arbitrary knowledge of the domain of discourse, knowledge which is usually not available to an automatic translation system.

This leaves us with the QLF and RQLF levels. Both these levels are deep enough to allow the application of sortal restrictions required for word sense disambiguation during translation. Both representations also contain noun phrase descriptions and syntactic information in the categories of QLF constructs.

However, not all the information appearing in the RQLF about how QLF constructs have been resolved is necessary for translation. For example, while pronoun resolution is sometimes required for translation between language pairs with differing pronoun systems (especially with regard to gender), definite descriptions are often best translated into target definite descriptions rather than referents, since otherwise the view of the referent in the source is lost during translation. As mentioned earlier, scoping and collective/distributive distinctions do not normally manifest themselves in paraphrases of RQLF interpretations, so ambiguities corresponding to these distinctions are often preserved during translation. Another case is that translation from resolved ellipsis can result in unwieldy target sentences.

It would thus appear that, for many constructions, there is little advantage to be gained for the purpose of translation from the process of interpreting unresolved QLF constructs. For practical systems, given the current state of the art, there might even be something to be lost by doing so: the lack of contextual knowledge and appropriate means for applying it mean that the interpretation process is error prone. Contextual knowledge is available to humans in the machine-aided translation setting, so we are concentrating at present on systems in which humans can provide contextual resolution for the cases where this is required. The BCI application is well suited to this approach, and issues concerning interaction with users are discussed in section 5.

The explicit representation of unresolved expressions in QLF, and the capability for generation from RQLF, allows the possibility of translation with limited resolution. For example, the application of reference resolution rules could be limited to those that resolve pronouns, if this is important, perhaps only in one direction, for translation between a particular language pair. The cases where resolution is required could even be specified explicitly in the relevant transfer rules. In this way, our framework for translation is potentially flexible with respect to the depth of interpretation as required by the needs of the translation task and as constrained by the available capabilities for accurate contextual resolution.

In arguing for QLF-level transfer, we are asserting that predicate-argument relations of the type used in QLF are the appropriate organizing level for compositional transfer, while not denying the need for syntactic information to ensure that, for example, topichood or the given/new distinction is preserved.

Kaplan *et al* (1989) argue for a translation methodology in which cross-linguistic correspondences are stated between various different levels. They motivate this by observing, as we have done here, that no single level can convey all the content involved. However, in their model, there is less clear a separation between monolingual and contrastive knowledge. It is also not clear that which of their levels provides the recursive organization on which compositionality can be based: mother-daughter relationships at one level may not hold between the corresponding objects at another level. This could be the root of the problems in this approach noted by Arnold and Sadler (1990).

McCord's (1988, 1989) organizing level appears to be that of surface syntax, with additional deep syntactic and semantic content attached to nodes. As we have argued, this level is not optimal, which may be related to the fact that McCord's system is explicitly not symmetrical: different grammars are used for the analysis and synthesis of the same language, which are viewed as quite different tasks.

Isabelle and Macklovitch (1986) argue against such asymmetry between analysis and synthesis on the grounds that, although it is tempting as a short-cut to building a structure sufficiently well-specified for synthesis to take place, asymmetry means that the transfer component must contain a lot of knowledge about the target language, with dire consequences for the modularity of the system and the reusability of different parts of it. In the BCI, however, the transfer rules contain only cross-linguistic knowledge, allowing the analysis and generation to make use of exactly the same data.

Nagao and Tsujii (1986) present a Japanese-English system with a transfer representation organized around semantic case relations which may be a little more superficial than QLFs. The system is uni-directional, and as noted above, this leads to a less clear modularisation of linguistic knowledge. Their main transfer process is recursive and compositional, but they argue for additional pre- and post-transfer phases, involving rules with global scope, to deal with certain phenomena which, they claim, cannot easily be handled by local rules. It may be that these phases would be unnecessary were a representation with an organizing level more like that of QLF to be used; however, such a conclusion would certainly not be warranted on the grounds of the work done so far in the BCI project. Nagao and Tsujii's description is of a substantial project on a pair of languages that differ far more than English and Swedish do.

4 QLF Transfer

QLF transfer involves taking the QLF analysis of a source sentence, say QLF1, and deriving from it another expression, QLF2, from which it is possible to generate a sentence in the target language. QLF2 will either be identical to a possible analysis of a sentence in the target language, or a less specific version of such an analysis in that features in categories in QLF2 could be uninstantiated.

Transfer rules specify a pair of QLF patterns, the LHS matches QLF expressions for one language and the RHS matches QLF expressions for the other language:

```
trans(<QLF1  subexpression pattern>
      <Operator>
      <QLF2 subexpression pattern>).
```

The QLF patterns in these rules can be QLF (sub)expressions, or such expressions with transfer variables showing the correspondence between the two sides, as explained later on. If the operator is **<=>** this states that the rule is bidirectional, otherwise a single direction of applicability is indicated by use of one of the operators => or <=.

Leaving aside unresolved referential expressions, the main difference between QLF1 and QLF2 is that they will contain constants, particularly predicate constants, that originate in word sense entries from the lexicons of the respective languages. The most common transfer rules are simple bidirectional ones like the following English-Swedish examples (the vertical bars and other non-alphabetic characters in the Swedish words represent accented letters):

```
trans(carl  <=> bil1).
trans(buy1  <=> k|p1).
```

giving the correspondence between predicates for a pair of noun and verb senses respectively. A slightly more complex rule might be the following one relating the English phrase “small farm” with the single Swedish word “hemman”: adjectives:

```
trans([and,[small1,X],[farm1.X]]   <=>   [hemman1,X]).
```

The above transfer rules correspond to the following meaning postulates (Carnap):

```
Ax(carl(x)   <--> bill(x))
AeAsAo(buy1(e,s,o)   <--> k|p1(e,s,o))
Ax(hemman1(x)  <--> smalll(x)/\farm 1(x))
```

where the postulates are expressed in a logical language for which the set of predicate constants is the union of the set of such constants in QLF expressions for both languages.

Viewed in this way, it is natural also to have transfer rules in which both sides contain constructs corresponding to word sense predicates of the same language. Such monolingual rules can be employed to produce sentence paraphrases with differing content words, without the need for additional processing components. As will be discussed later, such rules can play an important role in interactive disambiguation for translation.

Transfer rules are applied recursively, this process following the recursive structure of the expression tree for the source QLF. In order to allow transfer between structurally different QLFs, rules with ‘transfer variables’ need to be used. These variables show how subexpressions in the source QLF correspond to subexpressions translating them in the target QLF.

Transfer variables may appear more than once on either side of a transfer rule. A simple example of this is the following rule for translating the English intransitive verb *practise* into the Swedish transitive verb *övar* (thus “I practise” translates as “Jag övar mig”, I train myself):

```
trans([practise_2p,tr(event),tr(agent)]   <=>
      ['|var_3p',tr(event),tr(agent),
        a_term(<p=reflexive>,X,[person,X])]).
```

Transfer rules are not restricted to the logical elements of QLF, but are also used to indicate the correspondence between unresolved QLF expressions containing linguistic categories. For example, the `a_term` for a pronoun may be transferred into an `a_term` with a category having linguistic features that are specific to the target language, as in the following uni-directional transfer rule for pronouns (the Swedish pronoun “ni” always translates to English “you”, but not necessarily vice versa):

```
trans(a_term(<...l=you,...>,X,...)  <=
      a_term(<...l=ni...>,X,...))
```

Transfer between `q_terms` for descriptive noun phrases requires both the translation of the categories and the restrictions:

```
trans(q_term(tr(cat),X,tr(rest))  <=>
      q_term(tr(cat),X,tr(rest))).
```

Other transfer rules for categories would then indicate the correspondence between definiteness, number and gender in the two languages, such as the following rule for demonstrative articles:

```
trans(<1=_,t=ref,p=dem,n=N>) <=>
      <1=_,t=ref,p=dem,n=N,g=_>)
```

Formally, we can regard an `a_term` for a pronoun, or a `q_term` for a referential definite description, as a function from (linguistic and non-linguistic) contexts to referents. The linguistic featural information in these terms is an important part of the specification of such a function. A bidirectional transfer rule between two referential terms can thus be regarded as a meaning postulate stating equality of two functions, f1 and f2:

```
AcAx(f1(c)=x <--> f2(c)=x)
```

where `'c'` ranges over contexts and `'x'` over referents. Similarly, a uni-directional transfer rule, such as the one given earlier for `a_terms`, corresponds to an implication in one direction:

```
AcAx(f1(c)=x --> f2(c)=x)
```

so it is not possible to infer from the fact that the unresolved expression corresponding to `f2` refers to `'x'` in a given context, that the one corresponding to `f1` will also refer to `'x'` in that context.

5 Disambiguation and Interaction

The linguistic information available to the BCI defines a mapping of source language sentences onto source language QLFs, another of source language QLFs onto target language ones, and a third of target language QLFs onto target language sentences. In general, a given input will map onto several values, and it will be necessary for the system to choose which value is appropriate at each stage.

There are two observations to be made here. Firstly, there will be sentences for which the composition of the three mappings yields no values, i.e. which are untranslatable; so some means of recovery is desirable. This is discussed in section 5.4 below. Secondly, a particular choice between values may be spurious in that each choice leads to the same, or a very similar, meaning being conveyed to the target language user. This occurs in different ways in all of analysis, transfer and generation.

Thus when faced with a choice at any stage, the system can take any of the following actions, which are discussed in turn.

- Choose one option arbitrarily, on the grounds that the choice makes little or no difference to the meaning of the end result.

- Decide that one of the options is intrinsically better, for example because it conforms more closely to sortal restrictions, or because (if a QLF) it contains more salient or more frequent word senses. Reasoning, if performed, also comes into this class of actions.
- Present one of the users with (paraphrases of) the choices, and perhaps ask for a decision.

5.1 Recognizing a choice as spurious

It turns out to be easier to recognize a choice as spurious or unnecessary the closer we are to generating the output.

The simplest case is generation. In generating from target language QLFs, spurious choices are the rule rather the exception, because sentences corresponding to the same QLF will normally just be syntactic variants of one another. The BCI therefore presents the first sentence it generates from a QLF, only considering alternatives if the user asks for them.

In transfer, spurious choices can arise because QLFs are not canonical: different QLFs can have identical, or very similar, meanings; if the Swedish sentence “Jag äter gärna godis.” transferred to English QLFs for, say, “I like to eat sweets” and “I enjoy eating sweets”, the difference would not be important. Such choices can be much reduced by taking advantage of the directionality of transfer rules; when one term in language A corresponds to two alternatives in language B, but one of those alternatives will always in fact be acceptable, the transfer rules involving the others can be made unidirectional. Thus we might have the following rules, where the English “marvellous” and “terrific” can both map onto Swedish “storartad”, but in the other direction “storartad” is always translated as “marvellous”:

```
trans(marvellous1  <=>  storartad1).
trans(terrific1     =>  storartad1).
```

The same technique can be applied to structural alternatives too.

In analysis, a choice (this time between distinct meanings) can be spurious when source and target languages share an ambiguity; for example, the English “John drove the car without any insurance” and its Swedish translation “John körde bilen utan försäkring” are both ambiguous with respect to whether the car itself, or John’s driving of it, is uninsured.

There are two ways in which one might recognize a spurious choice between analyses. One is by “look ahead”: for example, if two source language QLFs differ only in the sense selected for a given word, the difference is unimportant if the two senses transfer to senses of the same target language word. In the English CLE lexicon, the word “bank” is defined with senses corresponding to (among other things) banks as buildings and banks as organizations. This distinction can be important in other language processing applications, but the same Swedish word is used for both.

In general, one might attempt to establish whether any given choice of analyses is spurious by following each option through all the way to generating multiple target language sentences, and see whether any generated sentence occurred in both sets. However, this is likely to be too time consuming to be practically useful. It may also have other practical difficulties in an architecture where the BCI components for the two languages run on two machines connected over a wide-area network.

A choice could also be recognized as spurious if two source QLFs were transferred to the same target. However, such situations would only arise fortuitously, for example when a semantic distinction had been made in the lexicon for one language but not in that for the other; it is not obvious that there would be any motivated reason for that to be the case.

The other way of recognizing a spurious choice would be to deprefer certain kinds of analysis. For example, QLFs in which (certain) prepositional phrase meanings attach to noun meanings could be depreferred. If the corresponding meaning, resulting from verb phrase attachment, is valid, it will be preferred, thus avoiding the need for any reasoning or user interaction. If it is not valid, the first QLF will be selected anyway. This technique effectively involves disguising a spurious choice as a real one that is easy to make automatically.

5.2 Selecting one option automatically

Non-spurious choices occur principally in analysis, but also to some extent in transfer, when the competing QLFs represent different meanings which cannot be expressed by the same sentence in the target language. In analysis, this is the familiar problem of linguistic ambiguity. In transfer, it will normally involve a lexical distinction made in the target language but not in the source language; the source language word cannot truly be called ambiguous, but nevertheless, in any given context, only one of the target words is correct. An example would be translating the English “river” to the Swedish “flod” (large river) or “å” (small river).

The fundamental choice of strategies here is between making the choice automatically and querying one of the users, although in practice a mixed approach will be adopted, the system doing what it can and consulting a user when it cannot decide on its own.

Automatic decision-making can be carried out:

- on context-independent, sentence-internal grounds, such as sortal restrictions on predicates;
- using general linguistic or domain-specific tendencies, such as frequencies of word senses or grammatical constructions;
- using reasoning about the relation between the analyses under consideration and the specific or general context, as in the CLE processing levels that (in other systems) follow on from QLF construction.

In the BCI, the first type of decision is currently made by the system during the analysis phase, and will in due course be applied to target language QLFs too. The second could in principle be made, while the third, in general, is dependent upon advances in the state of the art, or at least on the availability of domain-specific back-end systems.

The CLE's procedure for checking sortal restrictions has been adapted in the BCI to work on a preferential basis: a QLF that disobeys sortal constraints is only rejected if there is another QLF that disobeys fewer. This is because, especially in a relatively unconstrained domain, unpredictable uses such as metaphor may be quite common. A literal translation is better than nothing, but should be received with caution. Thus if a translation is produced from a sortally imperfect QLF, the fact can be indicated by attaching a query to the output, for example:

Input: `I want a car that won't die when it rains hard.`
Output: `(?) Jag vil ha en bil som inte dor då det hällregnar.`

It would be quite feasible to attach weights, possibly domain-specific, to particular word senses and grammar rules, so that in the absence of convincing sortal differences, the reading that resulted from the most frequent senses and rules would be preferred. The potential already exists to do this in the CLE, to some extent, by ordering rules and lexical entries. The CLE's representational levels also allow the results of reasoning to be incorporated if suitable intelligent systems become available.

5.3 Querying one of the users

When the BCI is unable to decide by itself on a QLF in analysis or transfer, it needs either to query one of the users, or to proceed with all the options, producing alternative outputs. The strategy that will be adopted in the BCI project is as follows:

- The system interacts with the source user to decide on a single QLF to be transferred, i.e. to resolve any source language ambiguities. This is done by generating paraphrases of each source QLF in the source language, and focusing on the points where they differ.
- One or more target language QLFs, ordered by preference information, may be produced from the single approved source QLF. A sentence is generated from each of these, and the target user decides (but need not indicate) which one is preferable.
- Sentences generated from the same QLF will be fairly close paraphrases of each other. For clarification, the target user may request a second paraphrase of any of the sentences presented to him.
- The target user may query the source user (via the BCI, of course) about any remaining ambiguities.

Thus a (purely illustrative!) exchange involving all four of these stages might be as follows:

User 1: `The poor man was probably swept away by the river.`

System to user 1: `Choose one of the following for "poor":`
`(1) unfortunate.`
`(2) penniless.`

User 1: `2`

System to user 2: `Den fattige mannen spolades nog bort av(ån/ floden)`

(The penniless man was swept probably away by the (small river / large river).)

User 2: `Ar det en a eller en flod?`

System to user 1: `Is that a small river or a large river?`

The reasons for this division of labour between source user, target user and system are as follows.

It is more appropriate for the source user to choose between source language QLFs, firstly because only the source user can reliably be expected to know which meaning he intended, and secondly because users may feel happier if they know that the system will not generate anything for the other user that might derive from an unintended meaning. For the system to be able to generate paraphrases from QLFs requires only a set of mono-lingual transfer rules, which will in any case be useful in the main transfer sequence when direct source-to-target transfer postulates are not defined for the predicates in question.

Correspondingly, the target user is better placed to choose between target language QLFs (implicitly, by inspection of the resulting generated sentences) for the practical reason that to ask the source user to do this would mean having to transfer structures back into *different* source language QLFs, which would not always be possible, and in any case would be beset by almost all the linguistic problems facing the whole translation process. The strategy of presenting together sentences generated from multiple QLFs should not swamp the target user with information because the transfer rules will have been written so as not to produce large numbers of results (as in the case of “storartad” above).

This overall interaction strategy would also allow a variant of the BCI’s interactive use in which it would be used to encode e-mail messages in QLF format, eliminating source language ambiguities, and to decode them on reaching the other user.

In the case of source language ambiguity, the CLE’s use of packing of local ambiguities allows multiple ambiguities in a sentence to be factored out, with the user's attention being focused on just the relevant parts of the sentence. Suppose a sentence contains three twofold ambiguities, giving a total of eight source language QLFs. Then rather than asking the user to choose one out of eight complete sentences containing considerable redundancy, the system could present paraphrases of each ambiguous phrase in turn.

Even when the choice is between target language QLFs, and no packed substructures are available, a similar effect could be achieved by generating complete sentences and then matching word by word to isolate the points where they diverge. This was assumed in the “ån / floden” example above.

5.4 Recovery from Translation Failure

The strategy for the case when no complete translation can be generated also makes use of packed structures. The procedure adopted here is to find a minimal set of strings which together span the input sentence and for which QLFs do exist. If these QLFs can be translated, an output consisting of a set of phrase translations can be produced. This translation will clearly be sub-optimal (in the limit, it will just be word for word), but, especially for language pairs with similar word orders, should often produce useful results. Another strategy here would be to ask the source user to rephrase the sentence, and to help him in doing this by displaying the phrases that had received analyses. An example might be:

```
Input: Can you see your way clear to giving me a discount?

System: Analysis failed. Phrases successfully analysed included:
          "Can you see your way"
          "giving me a discount"
          Please try again.

Input: Can you give me a discount?
```

References

Alshawi, H. and J. van Eijck (1989) “Logical Forms in the Core Language Engine”. *27th Annual Meeting of the Association for Computational Linguistics,* Vancouver, British Columbia, pp. 25-32.

Alshawi, H. (1990, to appear) “Resolving Quasi Logical Forms”. *Computational Linguistics.*

Kaplan, R.M., K. Netter, J. Wedekind and A. Zaenen (1989) “Translation by Structural Correspondences”, *Fourth Conference of the European Chapter of the Association for Computational Linguistics,* Manchester, pp. 272-281.

Isabelle, P. and E. Macklovitch (1986) “Transfer and MT Modularity”, *Eleventh International Conference on Computational Linguistics* (COLING-86), Bonn, pp. 115-117.

McCord, M. (1988) “A Multi-Target Machine Translation System”, *Proceedings of the International Conference on Fifth Generation Computer Systems,* Tokyo, pp. 1141-1149.

McCord, M. (1989) “Design of LMT: a Prolog-based Machine Translation System”, *Computational Linguistics,* Vol. 15, pp. 33-52.

Mellish, C.S. 1988. “Implementing Systemic Classification by Unification”, *Computational Linguistics,* Vol. 14, pp. 40-51.

Moran, D.B. 1988. “Quantifier Scoping in the SRI Core Language Engine”, *26th Annual Meeting of the Association for Computational Linguistics,* State University of New York at Buffalo, New York, pp. 33-40.

Nagao, M., and J. Tsujii (1986) “The Transfer Phase of the Mu Machine Translation System”, *Eleventh International Conference on Computational Linguistics* (COLING-86), Bonn, pp. 97-103.

Shieber, S.M., G. van Noord, R.C. Moore and F.C.N. Pereira 1989. “A Semantic-Head-Driven Generation Algorithm for Unification-Based Formalisms”. *27th Annual Meeting of the Association for Computational Linguistics,* Vancouver, British Columbia, pp. 7-17.