Anusaraka or Language Accessor:

Anusaraka tries to take **advantage** of the **relative strengths** of the **computer** and the **human reader**, where the **computer takes** the language load and leaves the **world knowledge** load on the reader. It is particularly effective when the languages are close, as is the case with Indian languages.

Keeping in line with the anusaraka philosophy, it bridges the gap between languages by chosing the **most appropriate or nearest construction available** in the target language together with suitable additional notation. It is argued in this section that there are only three major differences in the south Indian languages and Hindi.

All these can be **bridged by simple additional notation in Hindi.** The resulting language can be viewed as a southern dialect of Hindi. **Anusaraka uses this dialect to make source text in southern languages accessible to Hindi readers.**

5.2.1 Background :

The **problem** being addressed here is how to **overcome the language barrier in India**. Fully-automatic general-purpose **high-quality** machine translation systems (FGH-MT for short) are **extremely difficult** to build. There are **no existing translation systems** for any pair of languages in the world that qualify to be called **FGH-MT**. The **difficulty arises** from the following reasons:

1. In any **natural language text**, only part of the information to be conveyed is explicitly expressed and it is the **human mind** which fills up the details by using the **world knowledge.**

   The **basic reason** for this state of **affairs** is that the concepts and shades which a natural language purports to describe form a continuum and the number of lexical items available are finite, so necessarily they have to be overloaded and it is only because of the total context and shared background understanding that we are able to disambiguate them and are able to communicate.

2. **Different natural languages** adopt different conventions about the **type and amount** of information to be used.
   The **reasons** for this may be: **the history of language development** (differing tastes, arbitrary choices made in the long history of language, presence of other languages and mixing with them, etc.) and the **envisaged primary function of the language** etc.

The net result of this is that unless we provide **machines with knowledge and inferring capability comparable** to human beings FGH-MT will not be feasible. It will not be an exaggeration to say that inspite of tremendous progress in computer technology (mainly in terms of speed, memory and programming environments) FGH-MT remains a distant dream.

5.2.2 Cutting the Gordian Knot

In spite of the difficulty of FGH-MT, it can be claimed that with the help of machines, the language barrier can be overcome today.

If this sounds paradoxical let us consider an analogy. Scientists even today are struggling to build a machine that can walk like a human, avoiding obstacles to reach a destination.

At the same time, we have been successfully using rail transport for more than a century. The distance barrier has been overcome even without the machines learning to walk.

True, for this we have had to lay railway tracks all over the country; build bridges across rivers; dig tunnels through mountains and build a huge infra-structure to make the whole thing feasible. Even then it delivers the goods and passengers at only the railway stations; we need separate arrangements to get the things and persons at home! But all said and done, it does enable us to overcome the distance barrier.

If FGH-MT is like the walking machine, what is the counterpart to the railway locomotive? The answer lies in separating language-based analysis of text, from knowledge and inference-based analysis. The former task is left to the machine, and the latter task to the human reader because of their proficiency for the respective tasks. We also relax the requirement that the output be grammatical. We do require, however, that the output be comprehensible. Apart from the effort to build appropriate multilingual databases from a computational viewpoint, creative ideas are needed to establish language bridges. Though this is challenging, it is definitely feasible. With the appropriate infra-structure, the language barrier can be overcome with today's technology at least among the Indian languages.

5.2.3 The Problem

The practical aspect of this problem is to divide the load between man and machine in such a way that the aspects which are difficult for the human being are handled by the machine, and aspects which are easy for the human being are left to him. The aim is to minimize the effort of the human being.

The approach, however, can not be a heavy-handed method because the problem is very complex. A brute force solution will either blow up as a result of the large amount of resources it will need, or the large amount of time it will take. Another desirable feature of the approach would be the ease with which the system can be extended to other related languages. The system is likely to possess efficiency, extensibility, etc. only if it is based on sound principles and theory.

Theoretical issues of interest relate to information and how it is coded in language. How is the information extracted? A related question is why the same amount of surface information in one language is clearly understood while in another language, it is not clear to the reader. What are the sources of knowledge that are used in information extraction? How should the knowledge be organized?

Although the concept of anusaraka is general, we will discuss it in the context of Indian languages: with respect to Kannada-Hindi anusaraka in particular.

5.2.4 Structure of Anusaraka System

Structure of the anusaraka is shown in Figure 7.1. A source language sentence is first processed by morphological analyzer (morph). The morph considers a word at a time, and for each word it checks whether the word is

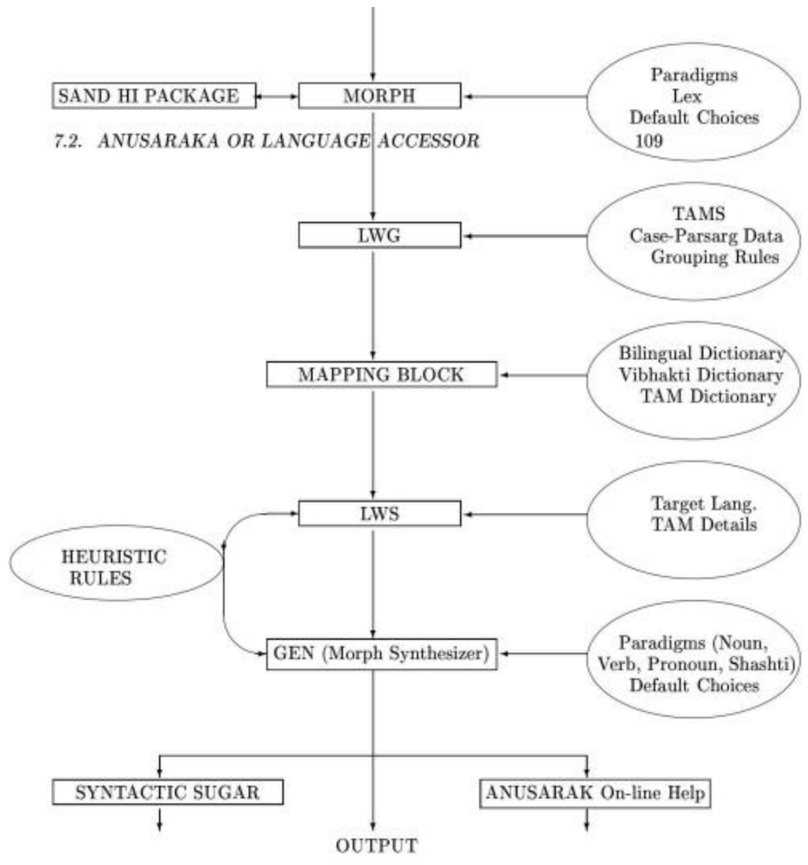

Figure 7.1: : Block Schematic of Anusaraka

in the dictionary of indeclinable words. If found, it returns its grammatical features. It also uses the word paradigms to see whether the input word can be derived from a root and its paradigm. If the derivation is possible, it returns the grammatical features associated with the word form (obtained from the root and the paradigm). In case, the input word cannot be derived, it is possibly a compound word and is given to the sandhi package to split it into two or more words, which are then again analyzed by morph.

The output of morph is given as input to the local word grouper. Its main task is to group function words with the content words based on local information such as postposition markers that follow a noun, or auxiliary verbs following a main verb. This grouping (or case endings in case of inflectional languages), identifies vibhakti of nouns and verbs. The vibhakti of verbs is also called TAM (tense-aspect-modality) label.

After the above stage, sentential analysis can be done. Current anusaraka does not do this analysis because it requires a large amount of linguistic data to be prepared. Also, since the Indian languages are close, the 80- 20 rule applies to vibhakti. Use of vibhakti produces 80% “correct” output with only 20% effort.? Sentential parser can be incorporated when large lexical databases are ready.

The next stage of processing is that of the mapping block. This stage uses a noun vibhakti dictionary, a TAM dictionary, and a bilingual dictionary. For each word group, the system finds a suitable root and vibhakti in the target language. Thus, it generates a local word group in the target language.

The local word groups in the target language are passed on to a local word splitter (LWS) followed by a morphological synthesizer (GEN). LWS splits the local word groups into elements consisting of root and features. Finally, GEN generates the words from a root and the associated grammatical features.

5.2.5 User Interface

Anusaraka output is usually not the target language, but close to it. Thus, the Kannada-Hindi anusaraka produces a dialect of Hindi, that does not have agreement etc. It can be called a sort of Dakshini (southern) Hindi. Some additional notation may also be used in the output. Certain amount of training is needed for a user to get used to the anusaraka output language.

The role of the anusaraka interface (or on-line help in Figure 7.2) is to facilitate the reading of output by the reader. It should keep track of what concepts have been introduced to the user, and also provide on-line help when the user faces a problem.

2It is our estimate that the sentential parser will improve the performance only marginally for translating from south Indian languages to Hindi

7.2. ANUSARAKA OR LANGUAGE ACCESSOR 111

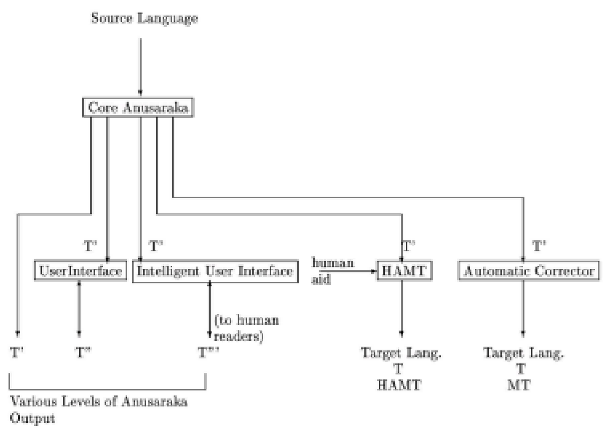

Figure 7.2: : Different Interfaces for Anusaraka

Depending on the nature of interface, one can have an ordinary user interface, intelligent user interface, ggHAMT (human aided machine translation), and machine translation .

5.2.6 Linguistic Area

The reason the above approach works for Indian languages even without a full fledged parser is that the source and target languages are similar. At mapping level, it is possible to find roots, TAMs, and noun vibhakti in Hindi from an equivalent Kannada text without too many clashes.

Based on our experience, we can say that apart from agreement, there are only 3 major differences in the south Indian languages with Hindi. All these can in fact be bridged. They are described below for Kannada-Hindi anusaraka. But first let us look at agreement.

5.2.7 Giving up Agreement in Anusaraka Output

Hindi has an agreement rule which can be stated as follows:

Gender number person (gnp) of the verb agrees with the gnp of the karta if it has ¢ vibhakti*®, otherwise the gnp of the verb agrees with the karma if it has ¢@ vibhakti; otherwise the verb takes masculine, singular, third person form.

When the input Kannada sentence has an ambiguity in identifying karta (or karma whichever is relevant for agreement) and the gnp of the two candidate kartas (or karmas) is different, an ambiguity will appear in the gnp of the verb.

Similarly, possessive modifiers of nouns need information about gnp of the related nouns. There are a number of cases where it is not easy to compute. Sometimes, the relevant information may not even be available.

Kannada has three genders (masculine, feminine and neuter), whereas Hindi has only two (masculine and feminine). Consequently, some information loss is bound to occur in going from Kannada to Hindi. To avoid this loss, it will be necessary to provide additional notation to mark the neuter gender. Here is an example to give its significance:

In Hindi, for karma-kartr prayoga, typically separate verb forms are available; but in Kannada, frequently a single verb stem is employed. Thus, on the surface, it seems that it does not distinguish between “daravaajaa khulaa” and “daravaajaa kholaa”. However, in practice one can distinguish between the two usages by paying attention to the gender marking on the verb; in the first case it will be neuter gender while in the second, typically, it will be a non-neuter gender® (where K specifies Kannada sentence, @H specifies Hindi produced by anusaraka).

K: baagilu tereyitu.

@H: daravaajaa khulaa [kholaa].

(The door opened.)

K: baagilu teredanu.

@H: daravaajaa khulaa [kholaa].

((He) opened the door.)

3Karta having ¢ vibhakti means that it is not followed by a postposition marker (a function word). 4There is a temptation to provide correct agreement according to standard Hindi grammar, wherever it is possible without much effort and not bother about agreement when it is not possible to do so easily; but such a policy will be potentially confusing to the user. From the utility point of view, it always helps to keep the working of the system simple and to provide a faithful picture of the working of the system to the user.

5“daravaajaa kholaa” (opened the door.) is an incomplete sentence in isolation but it can occur in discourse in response to the question “raama ne kyaa kiyaa?” (What did Ram do?) Loss of agreement is not expected to be too jarring to Hindi speakers in view of their exposure to various varieties of non-native Hindi heard everyday as propagated by television, radio and films.

5.2.8 Language Bridges

Apart from agreement there are only three major syntactic differences between Hindi and Kannada. Surprisingly all of these can be taken care of by enriching Hindi with a few additional functional particles or suffixes as shown below. Thus, they can be viewed as lexical gaps or function word gaps. language bridges “ki” construction

In case of embedded sentences in Hindi, these are put after the main verb unlike in Kannada. For example (where K, H, and E specify the language of the sentence as Kannada, Hindi and

English, respectively; '!H' stands for gloss in Hindi, '!E' for gloss in English, @H for anusaraka output in Hindi, etc.):

(7.1) H: raama ne kahaa ki mohana kala aayegaa.

'E: Rama said that mohana tomorrow come-fut

E: (Rama said that mohana will come tomorrow.)

K:*raama heLidanu eneMdare naanu manege hoguttene.

'H: raam kahaa ki meiM ghara ko jaauuMgaa. (Ram said that he will go home.)

Note, that the above Kannada sentence sounds odd to a Kannada person, because of the order of the constituents. However by using "eisaa" instead of "ki" we can easily avoid this problem, as illustrated below:

(7.2) K: mohana naaLe baruvanu eMdu raama heLidanu.

'H: mohana kala aayegaa eisaa raama ne kahaa.

'E: Mohana tomorrow come-fut that Rama _ said.

'eisaa' construction is a proper construction in Hindi; only it is used less frequently. In the dialect of Hindi produced by anusaraka, however, this will be the normal construction used. "jo" construction Kannada has a large number of adjectival participial phrases or clauses which convey information about tense, aspect etc. but they do not have information about karaka relations. In sentences (7.3) and (7.4), Kannada codes information about tense in eating and making.

(7.3) K: raama tiMda camacavannu toLe.

@H: raam khaayaa thaa cammaca dho Daalo.

(7.4) K: raama tayaarisida camacavannu toLe.

@H: raama banaayaa thaa cammaca dho Daalo.

It does not contain information about the relation eat has to spoon or make has to spoon. It is the background knowledge that provides the relation between them. For example, the general world knowledge may be used to say that spoon is the instrument of eat (and is not eaten) in (7.3). In the next sentence (7.4) spoon could be the instrument or theme of make.

Hindi, on the other hand, has only two participial phrases viz. yaa_huaa and taa_huaa which code perfective and continuous aspects only, e.g.,

(7.5) H: khaayaa huaa phala eaten fruit

(7.6) H: khaataa huaa laDakaa eating boy

Thus, anusaraka would be able to use these constructions in Hindi only when the tense information in Kannada is appropriate. But what about the other tense, aspect and modality? There is a syntactic hole in Hindi!.

There is another problem, however. The two participial phrases in Hindi have coding for karaka relations which is absent in Kannada. yaa_huaa codes karma', while taa_huaa codes karta. Participial phrase (7.5) says that fruit that was eaten, while (7.6) says the boy who is eating. Thus, Hindi is poorer than Kannada in coding tense, aspect, modality information, while richer in coding karaka information in case of adjectival participles. But this compounds the problem for anusaraka. Using these constructions in Hindi would mean putting in something that is not contained in the source language sentence.

The answer lies in identifying another construction in Hindi and creating a correspondence between it and the constructions under consideration in Kannada.

Hindi has another construction-the ‘jo’ construction-which allows both tense and karaka information to be specified. For example, to say ‘wash the spoon with which Ram has eaten’ we can write any of the following:

(7.7a) raama ne jisa cammaca se khaayaa thaa usako dho Daalo.

SMost native speakers are surprised to learn of this fact. The information dynamic view, therefore, raises questions very different than those raised and studied in conventional linguistics.

7More correctly, yaa_huaa codes karma in case of sakarmaka or transitive verbs, and karta in case of intransitive verbs