4.1 Introduction

The parser for a language needs the part of speech tag set for various words in the sentence, the groups of co-occurring words known as word chunks, the structure of sentences in a language and the hierarchical dependencies of chunks in sentences. A detailed description of various classes of sentences in a language is given with specific examples from Malayalam language. The need for syntactic structure transfer between two languages is also explained in this chapter. Finally, the different computational models for parsers are also discussed.

4.2 Syntactic structure of languages

In order to arrive at a computational grammar for the language the set of word classes (Part Of Speech tagset), chunk tagset and the hierarchical dependencies among the chunks are needed. This requires a careful analysis of the different classes of sentences in the language and the hierarchical dependencies among the word groups in a sentence. A study of the syntactic difference between the source and target language is also required for applications like machine translation to produce the target sentence in the correct structure.

4.2.1 Part of Speech

Words are divided into classes called parts of speech (POS) according to their use in a sentence. The significance of POS tags of words in the present day NLP is widely known. POS tagging is an important activity for investigators of
natural language processing, speech recognition and other related areas. It proves to be a basic building block for constructing statistical models and it needs an annotated corpora for automatic processing of natural languages [68,69]. Annotation of corpora can be done at various levels viz, part of speech, phrase/clause level, dependency level, etc. Part of speech tagging forms the basic step in any natural language processing application. Chunking can form the next level of tagging [70,71].

A 'word' in a text carries the linguistic knowledge such as a) grammatical category and b) grammatical features. The POS tag should be based on the 'category' of the word and the features such as gender, number, person etc. can be acquired from the morphological analyser.

POS tags are very important for words in language translation task. Same word can have different meanings in different contexts. Word sense disambiguation is required to find the correct meaning of the word in the context. POS tags of words can be used as a means for word sense disambiguation.

Malayalam : രമു കരയുന്ന് (Ramu is crying)

Transliteration: raamu karayunnu

Morphemes: രമു / Ramu (Noun)+ കര / kara (Noun/verb) + ഉന്ന / unnu (Verbal suffix)

The word കര / kara can have two tags: noun and verb root. Depending on the tags the word can be translated as either land or cry. Parser uses these tags for word sense disambiguation. When it tries to build the parse tree for the sentence using the lexicalized grammar it will be able to identify the word കര / kara as a verb root and finally the correct meaning “cry” can be found out.
4.2.2 Selection criteria for POS tagset

The tagset is arrived at considering the following factors:

a) it should be comprehensive and complete
b) it should be simple.
c) the tags are to be used for word sense disambiguation for getting the correct translation by the parser,

Major issues that are encountered in arriving at a POS tagset are:

1. Fineness v/s Coarseness in linguistic analysis
2. New tags v/s tags from a standard tagger

i) Fineness v/s Coarseness

One of the issues while deciding a tagset for annotating the input text is the 'fineness' or 'coarseness' in linguistic analysis. A decision had to be taken whether the tags will account for finer distinctions of the various features of the parts of speech. It has to be decided if plurality, gender and such other information will be marked distinctly or only the lexical category will be marked.

It is preferred to come up with a tagset which avoids fine distinctions. The motivation behind this is to have less number of tags since less number of tags leads to simplicity. Accuracy of manual tagging is higher when we have lesser number of tags.

However, a matter of greater concern concern is regarding the grammatical and other relevant linguistic knowledge which is encoded in a word, particularly in agglutinating languages (which several Indian languages are). If tags are too coarse, some crucial information for further processing might be missed out.
Too coarse an analysis is also not of much use. A balance between fineness and coarseness is required.

ii) New tags v/s tags from a standard tagger

While deciding the tags for a tagger we can either come up with a totally new tag set or take any other standard tagger as a reference and make modifications in it according to the objective of the new tagger. The Penn tag set is a standard tag set used for English. Many tag sets designed after this have been a variant of this tag set (e.g. Lancaster tag set). While deciding the tags for our work, the IIIT tag set has been used as a benchmark. We have used our own tag names for convenience. New tags have been introduced wherever Penn tags have been found inadequate for our translation problem.

4.2.3 Clauses

A group of words that forms part of a sentence and has a subject and a predicate of its own is called a clause. Clauses are classified into:

1. Adverb clause
2. Adjective clause
3. Noun clause

The clause markers in Malayalam are given Table 4.1.

4.2.3.1 Adverb clause

An adverb clause is a group of words which contains a subject and a predicate of its own and does the work of an adverb. They are further classified into:
a) Adverb clause of time

Adverb clauses of time are introduced by the subordinating conjunctions / appOzhokke, / appOL, / kazhinjinju, / athinumunpu, / athuthottu, / athumuthal etc.

1. (It rained when you came)
   nee vannappoL mazha peythu
2. (It had been raining since you came)
   nee vannathu muthal mazhayaanu

b) Adverb clause of place

Adverb clause of place are introduced by the subordinating conjunctions / evite and the verb ending / um, / evite and / o suffix with the verb.

(Ramu’s dog will be with him wherever he goes)

raamu evitepoyalum avante patti kootekkaaNum

c) Adverb clause of purpose

Adverb clause of purpose are introduced by the subordinating conjunction / aakaanvEnti.

(Ramakrishna’s brave son will follow)
raajavakaanvEnti veeran makane konnu.)
d) Adverb clause of reason or consequence

Adverb clause of reason are introduced by the subordinating conjunction / athukondu, athukaraNam.

(Ram to become the king)

Ramu paThikkathathu kaaraNam pareekshayil thOttu.

(Ramu failed in the exam because he didn’t study)

e) Adverb clause of condition

Adverb clauses of condition are introduced by the subordinating conjuctions aal, aalum.

(If you study you can win)

f) Adverb clause of comparison

Adverb clause of comparison is two types:

i) Adverb clause of comparison of degree : In this the subordinating conjunction athinEkkaL, athupOle.

(If you study you can win)

ii) Adverb clause of comparison of manner : In this the subordinating conjunction used is athupOle
g) **Adverb clause of supposition or concession**

Adverb clause of supposition or concession are introduced by the subordinating conjunctions അല്ല / aalum, ക്കാലം / enkilum.

>(All happened as I planned)

*Raman nErathe pOyenkilum dOctoRe kaaNaan pattiyilla*

> (Even though Raman went early he couldn’t see the doctor)

### 4.2.3.2 **Adjective clauses**

An adjective clause in a complex sentence is a subordinate clause which does the work of an adjective and so qualifies some noun or pronoun in the main clause.

1. An adjective clause is introduced by a relative pronoun or by a relative adverb. The adjective clause is marked by a relative participle like ചിത്ര / cheytha, പാരാജനി / paRanjna etc.

> (It is this boy who broke the window)

> janaala pottichcha kutti ivanaaNz.

> (It is this boy who broke the window)
2. An infinitive with a suffix അണ്‌ aan is also often used as the equivalent of an adjective clause.

ഇതിനാണ്‌ പഠിക്കാൻ പുസ്തകം വെച്ചാം.
enikk paThikkaan pusthakam vENam.
(I want a book to study)

4.2.3.3 Noun clause

Noun clause is a subordinate clause which does the work of a noun in a complex sentence. It is further classified into:

a) The subject of a verb: In this use the clause ends with a verb with the suffix അതി/ athu.

ഒന്നിച്ച് പറഞ്ഞതിന്‌ സത്യമാണ്‌
njaan paRanjathu sathyamaaNu.
(what he told is true)

b) The object of a transitive verb: in this use of the noun clause it ends with a word എന്ന്/ ennu.

മോഹൻ വരുമെന്ന് സത്യമല്ല
mOhan varumennu seethe paRanju.
(Seetha told that Mohan will come)

c) The object of a preposition
There is no meaning in what you say

d) In apposition to a noun or pronoun

(It is saddening that you were on leave.)

e) The complement of a verb of incomplete predication.

(It is my belief that he will not come)

4.2.4 Classification of Sentences

Sentences in a language are classified as

1. simple sentences
2. complex sentence
3. compound sentences

4.2.4.1 Simple sentence

A simple sentence (choornika) is one which has only one subject and one predicate. In another way, a simple sentence is one which has only one finite verb.
### Table 4.1 Clauses and Clause markers in Malayalam

<table>
<thead>
<tr>
<th>Type of Clause</th>
<th>Clause function</th>
<th>Clause marker/ suffix to verb</th>
</tr>
</thead>
<tbody>
<tr>
<td>adjective</td>
<td>Adjective</td>
<td>Relative participle /paRanja</td>
</tr>
<tr>
<td></td>
<td>Adjective</td>
<td>Infinitive before noun</td>
</tr>
<tr>
<td>Noun</td>
<td>Subject of a verb</td>
<td>/athu</td>
</tr>
<tr>
<td></td>
<td>Object of a transitive verb</td>
<td>/ennu</td>
</tr>
<tr>
<td></td>
<td>Object of a preposition</td>
<td>/athu+preposition</td>
</tr>
<tr>
<td></td>
<td>In apposition to a noun or pronoun</td>
<td>/ennathu</td>
</tr>
<tr>
<td></td>
<td>Complement of a verb of incomplete predication</td>
<td>/ennathaaNu</td>
</tr>
<tr>
<td>Adverb</td>
<td>Time</td>
<td>/appOL, /kazhinju, /athinuSEsham, /athinumunpu, /athuthottu, /Athumathal</td>
</tr>
<tr>
<td></td>
<td>Place purpose</td>
<td>verb prefixed by evite and suffixed by /o</td>
</tr>
<tr>
<td></td>
<td>Reason/ consequence</td>
<td>/athukontu, /athukaaraNam</td>
</tr>
<tr>
<td></td>
<td>condition</td>
<td>/aal</td>
</tr>
<tr>
<td></td>
<td>Comparison of degree /manner</td>
<td>/AthinekkaL, /athupOle</td>
</tr>
</tbody>
</table>

1. Sooryan kizhakkuudikkunnu
(Sun rises in the east)

2. മോഹൻ മിട്ടിക്കണം

   *mOhan mitukkan aanz*
   
   (Mohan is intelligent)

4.2.4.2 Compound sentence

When two or more independent simple sentences are combined by a coordinating conjunction like അഡ്ഡം we get a compound sentence (mahaavakyam). In a compound sentence each of the sentences is called a principal clause or main clause.

   രാമൻ പറഞ്ഞ് മോഹൻ മാത്രാക്ഷം

   *Raaman patukayum mohan aatukayum cheythu.*
   
   (Raman is singing and Mohan is dancing.)

4.2.4.3 Complex sentence

A complex sentence (sangeernakam) consists of a principal clause and one or more subordinate clauses.

   രാജൻ ഏതാഴന്റെ കുട്ടി കരാൻ

   *raajan atichchappOL kutti karannjju.*
   
   (The child cried when Rajan hit.)

In the above sentence, കുട്ടി കരാൻ / *kutti karanjju* is the principal clause, which can stand by itself and രാജൻ ഏതാഴന്റെ / *raajan atichchappOL* cannot
stand by itself and make good sense. This clause is dependent on the clause കുത്തി കരാൻ / kutti karanjnu. Subordinate clauses are also called dependent clauses.

പഠിക്കാത്ത താഴെ / paThikkaththathukondu raju pareekhayil thOttu
(Raju failed in the exam because he didn’t study)

പഠിക്കാത്ത / paThikkaththathukondu is the subordinate clause and
ബാഹ്യ സൂചി / pareekshayil thOttu is the main clause.

മോഹൻ വന്നപ്പള / Mohan vannappOL seethayekandillennu raaju paRanjnu
(Raju told that when Mohan came he didn’t see Seetha)

This sentence has two subordinate clauses and one main clause. The two subordinate clauses: a noun clause, മോഹൻ വന്നപ്പള / mohan vannappOL seethayekandillennu and an adverbal clause മോഹൻ / Mohan vannappOL which is part of the above subordinate clause.

This shows the hierarchical dependencies of the clauses in a sentence. റജു / raaju paranjnu is the main clause.

4.2.5 The Hierarchical structure

Clauses in a sentence can be nested one inside the other, resulting in a hierarchical or tree like structure. This aspect of structure is called the hierarchical structure [72,73,40,41]. Clauses in a sentence are not completely independent of one another but there are inter-clause dependencies. For example, a noun phrase being modified by a relative clause has two roles to play, one in the relative clause and the other in the outer clause.
According to Universal clause structure grammar (UCSG) all inter-clause dependencies systematically flow down the clause structure tree from the root towards the leaves [74]. Also, the constituents of a clause do not cross clause boundaries in scrambling. Violations of this principle can be viewed as exceptions to this general rule rather than as evidence for the invalidity of this principle. Verb groups and sentinels contain all the required information for recognizing clauses, for determining the nested or hierarchical structure of clauses and for determining the clause boundaries, although only partially. It is seen that every clause in a sentence except for the main clause has a sentinel which marks one of the boundaries of that clause. The sentinel marks either the beginning or the end of the clause depending upon the language in use. Also, by definition, every clause must have exactly one verb group. Thus verb groups and sentinels behave like brackets and impose very strong constraints – the brackets must match properly. Thus the total number of verb groups in a sentence must be exactly one more than the total number of sentinels.

Constraints on clause structure imposed by verb groups and sentinels are thus very strong yet very easy to apply. These constraints also help us in reducing lexical ambiguities to some extent, especially the more critical ambiguities such as noun/verb and sentinel/non-sentinel ambiguities. The hierarchical structure for sentences is used in language understanding and also in machine translation [75,76,77,78]. The job of syntactic analyzer is to accept a sentence in natural language and to produce a description of its internal structure in the light of the given grammar - a formal specification of all the valid structures in that language [79,80]. There are several aspects of structure that a syntactic theory is expected to deal with, including assignment of functional roles to the various constituents, analyzing the modifier-modified relationships, resolution of
anaphoric and other kinds of references, attachment of prepositional phrases and subordinate clauses, and analysis of emphasis, focus, topic etc.

4.2.6 Syntactic structure difference between languages

One of the crucial tasks in machine translation is the syntactic structural transfer, which is the conversion from a syntactic analysis structure of the source language to the structure of the target language. Structural information can be in the form of constituent transfer: for example, how is a noun phrase or a sentence constructed in a language, and how does the ordering of words and groups of words change when translated into another language, etc. [81,82]. Languages like Malayalam and English belong to two language families and their sentence structure differs a lot. So a simple morpheme to morpheme mapping will not give the correct translation of the input sentence. The parser has to perform the syntactic structure transfer also to get the target sentence in the correct structure.

Malayalam sentence: സീത രാമാന്ത് കുട മോഹൻ കൊടുത്തുനു രാധ പരാജൻജ്

*seetha raamante kuta mOhanu kotuththennu raadha paRanju*

Sequence of morphemes: സീ റ, േ seetha, റ റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ, റ,ംപാറാജന്ന്

English translation: Radha told that Seetha gave Raman’s umbrella to Mohan
For the example given above the reordering required at the morpheme level is shown below.

From the above example it is clear that a lot of reordering of words is required to get the target sentence in the correct form for applications like machine translation. It is also found that the reordering occurs at the word group level. The reordering required is specific to the language pair under consideration.

4.3 Parsing

The process of generating the sentence through derivation using a set of grammar rules is called parsing and the generated hierarchical structure is called the parse tree of the sentence. Although a lot of work has gone into developing full syntactic parsers, high performance wide coverage syntactic parsing remained a difficult challenge. In recent times there has been an interest in shallow parsing or partial parsing. Shallow parsing is restricted to finding phrases in sentences. These phrases are called chunks. With partial parsing it is
possible to disambiguate words based on part of speech tags like verb/noun/suffix conflict. But for handling sentences like I turned off highway I-90 and I turned off my radio we need more world knowledge to distinguish between the two meanings of the phrase ‘turned off’. More knowledge regarding the type of the object, like whether it is the name of a road or it is a device, is needed to find the correct meaning of ‘turned off’. This kind of disambiguation can be done only by a global analysis, including syntactic analysis, discourse analysis and even world knowledge. It is possible that many different phrase structure trees derive the same sequence of words. If a sentence has multiple parses then the grammar is called ambiguous grammar.

4.3.1 Computational models for parsing

Finding the right syntactic structure is a search process. The search finds all possible parses for a sentence. Parsing with sub grammars reduces the problem of rapid growth in parsing table sizes as number of rules increase [83,84,85]. The following constraints guides the search process:

1. The first constraint comes from the words in the input sentence. A valid parse is one that covers all the words in a sentence. So these words will constitute the leaves of the final parse tree.

2. The second constraint comes from the grammar. The root of the final parse tree must be the start symbol of the grammar.

Using the two constraints two widely used search strategies are used:

a) Top down search or (goal directed search)

b) Bottom up search or (data directed search)
4.3.1 Top down parsing

As the name suggests, top down parsing starts its search from the root node S and works downwards towards the leaves. The parser needs the set of grammar rules represented in the context free form. The parser has to find all subtrees which can start with S. To generate the subtrees of the second level search, the root node is expanded using all the grammar rules with S on the left hand side. Each non terminal in the resultant sub trees is expanded next using the grammar rules having a matching nonterminal symbol on the left hand side. The right hand side of the grammar rules provides the nodes to be generated which are then expanded recursively. As the expansion continues the tree grows downward and eventually reaches a state where the bottom of the tree consist only of terminal symbols. The subtrees whose leaves do not match words in the input sentence are rejected and other options for its parent are tried.

4.3.1.2 Bottom-up parsing

A bottom up parser starts with the words in the input sentence and attempts to construct a parse tree in an upward direction towards the root. At each step the parser looks for rules in the grammar where the right hand side matches some of the portions in the parse tree constructed so far and reduces it using the left hand side of the production. The parse is considered successful if the parser reduces the tree to the start symbol of the grammar.

Both the methods have advantages and disadvantages. The top down search generates trees with the start symbol of the grammar. So it never wastes time exploring a tree leading to a different root. But it wastes time exploring trees that produce words that are inconsistent with the input. A bottom up parser
never explores a tree that does not match the input. It wastes time generating
trees that have no chance of leading to the start state.

The top down parsing needs backtracking as a word can have more than one tag
and the tree can be traversed in more than one way. So when the first choice is
found to be false at a later stage it has to go back and try the other tag for the
same word. The second problem with such algorithms is that the trees built
may be discarded during backtracking. At a later time these may have to be
rebuilt during subsequent steps in the parse.

4.3.2 Previous works

Both the above problems of top down depth first approach are solved by
dynamic programming algorithms. The CYK algorithm, Graham Harrison
Ruzzo (GHR) algorithm and the Earley algorithm are the three parsers which
use dynamic programming. Earley parser is a top down search using dynamic
programming. The CYK parser is a parser based on bottom up approach using
dynamic programming.

Rajiv Sangal et. al. has proposed a karaka based approach to parsing Indian
languages [86]. Their parser does karaka role assignment for verbs and word
sense disambiguation for verbs and nouns in Hindi. For karaka role assignment
they use integer programming techniques. For word sense disambiguation they
use a merged lakshan chart for each verb which contains the different
conditions to be tested to assign a particular meaning to a word.

The verb “jotha” has different meanings in different contexts. The
disambiguation is done using a merged karaka chart.
to find the meaning of ‘jotha as harnessed, hitched or ploughed, context dependent rules like the following are used. “if category of the karta is human and category of karma is animal then meaning is hitched”.

“if the karma is the word khet or jamin then the meaning is ploughed”.

Partial parsing via finite state cascades has been described by Abney [87,88]. A finite state cascade consists of a sequence of levels. Phrases at one level are built on phrases at the previous level and there is no recursion. Phrases never contain same level or higher level phrases. Deterministic parsers specified by finite state cascades are fast and reliable. They can be extended to construct parse tree with finite feature structures.

Miles Osborne has proposed that shallow parsing can be used for part of speech tagging [89]. Zhou et al. has proposed an HMM based chunk tagger with context dependent lexicon. Rob Koeling applied maximum entropy models for chunking [90].

Purely linguistic as well as purely machine learning approaches have proved to be impractical. Hand crafting rules in the linguistic approach can be very laborious and time consuming. Parsers tend to produce a large number of possible parses and in the absence of suitable ranking and rating mechanism selecting the right parse can be difficult. In addition to these, morpheme based parsing increases number of possible parses [91].

Murthy has proposed a method which combines a finite state chunker and a statistical chunker using HMM [17,92]. Here, instead of looking for a grammar that can capture all and only valid structures, a finite state chunker which captures all valid word groups without necessarily restricting to only those
word groups which are appropriate in the context of a given sentence. A separate statistical component is used to rank the word groups so produced. The system only rates the word groups produced. The system rates all the parses produced. The system is more than a shallow parser as it can do disambiguation in context to some extent.

Win Win Thant et. al. has proposed a Context Free Grammar Based Top-Down Parsing of Myanmar Sentences [93]. In this they describe a method to parse simple and complex sentences in Myanmar. The accuracy reported is 90% for simple sentence and 91% for complex sentences.

Mark A Jones et. al. has reported on the application of a probabilistic parser applied to software test documents [94,95]. They train a statistical parser from a bracketed corpus and its use in a software testing application that translates English specifications into automated testing language. No grammar rules are explicitly specified. The rules and contextual probabilities are generated from the corpus. The parser is successful in identifying the correct parse and deterministic in the number of parses it produces. With 211 sentences for training and 147 for testing, parses were found for 77% of test sentences. Of these the top ranked parse was correct 90% of the time and 99.3% of bracketing decisions were correct.

Brian Roark describes the functioning of a broad-coverage probabilistic top-down parser, and its application to the problem of language modeling for speech recognition [96].

Richard A. Frost describes a new top-down parsing algorithm to accommodate ambiguity and left recursion in polynomial time [97]. He discusses on how exponential complexity can be avoided by memorization using a table, and that
left-recursive productions can be accommodated through a technique like the length of input to stop recursive descent or calling nonterminals with how many terminals should be selected etc.

Gabor Proszeky has discussed a morphological analyzer as a syntactic parser [49]. The system was called Humor ESK (High speed unification morphology enhanced with syntactic knowledge). It consists of numerous meta lexicons. Each of them has the name of the syntactic category it describes. The categories like S, NP, VP etc. are stored in separate lexicons. Meta lexicons form a hierarchy. Parsing on a level can be realized as a lexicon lookup. No backtracking, look ahead, or the time consuming parser steps are needed for the analysis of a sentence.

Stuart M. Schieber describes Sentence disambiguation by a shift reduce parsing technique and a uniform architecture for parsing and language generation [98,99].

4.4 Conclusion

This chapter discussed the parsing methods, the syntactic structure of sentences in a language and also the syntactic differences between sentences of two languages. The different classes of sentences and the hierarchical structure in sentences were explained in this chapter. The need for part of speech tagging and the criteria for tagset selection were also discussed in the chapter. The next chapter discusses the design and development of the prototype translator.