**CHAPTER 2**

**LITERATURE SURVEY**

There are various approaches to develop the Shallow Parser. In this chapter some of these approaches are described with their working methodologies and advantages and limitations of each approach. Section 2.1 gives a brief introduction to the various terms used in parsing. Section 2.2 contains the detailed description of the various approaches of the Shallow parser. Section 2.3 gives an overview to the morphology of the “Hindi” language.

**2.1 Parsing**

In linguistics and computer science, **parsing**, or, more formally, **syntactic analysis**, is the process of analyzing a text, made of a sequence of tokens (for example, words), to determine its grammatical structure with respect to a given (more or less) formal grammar. Parsing can also be used as a linguistic term, for instance when discussing how phrases are divided up in garden path sentences.

Parsing is also an earlier term for the diagramming of sentences of natural languages, and is still used for the diagramming of inflected languages, such as the Romance languages or Latin. The term parsing comes from Latin *pars* (*ōrātiōnis*), meaning part (of speech).

Parsing is a common term used in psycholinguistics when describing language comprehension. In this context, parsing refers to the way that human beings, rather than computers, analyze a sentence or phrase (in spoken language or text) "in terms of grammatical constituents, identifying the parts of speech, syntactic relations, etc." This term is especially common when discussing what linguistic cues help speakers to parse garden-path sentences.

Parsing in natural languages can be classified at two abstraction levels namely, deep parsing and Shallow parsing.

**Shallow Parsing**

In this technique, we get hierarchical and grammatical information while preserving robustness and efficiency of the processing. In this perspective, we make use of a grammar represented in the Property Grammar formalism.

**Deep Parsing**

Deep analysis is directly based on property grammars. It consists, for a given sentence, in building all the possible subsets of juxtaposed elements that can describe a syntactic category. A subset is positively characterized if it satisfies the constraints of a grammar. These subsets are called edges; they describe a segment of the sentence between two positions.

**Word**: Word is defined as a smallest thought unit vocally expressible composed of one or more sounds combined in one or more syllables. A word is a minimum free form consisting of one or more morphemes. There are many ways to combine morphemes to create words. Four of these methods are common and play important roles in speech and language processing: Inflection, Derivation, Compounding and Cliticization.

**Inflection:** Inflection is the combination of a word stem with a grammatical morpheme, usually resulting in a word of the same class as the original stem, and usually filling some syntactic function, e.g. plural of nouns.

Table(singular)

Table+s(plural)

The meaning of the resulting word is easily predictable. Inflectional morphemes modify a word’s tense, number, aspect and so on.

**Derivation:** Derivation is the combination of a word stem with a grammatical morpheme, usually resulting in a word of a different class, often with a meaning hard to predict exactly.

**Compounding:** Compounding is the joining of two or more base forms to form a new word. For instance, two nouns “car” and “driver” can be fused to create “car-driver’. Such frequent root-root fusions are very common in written Hindi. Semantic interpretation of compound words is even more difficult than with derivates. Almost any syntactic relationship may hold between the components of a compound.

**Phrase:** In everyday speech, a **phrase** may refer to any group of words. In linguistics, a phrase is a group of words (or sometimes a single word) that form a constituent and so function as a single unit in the syntax of a sentence. A phrase is lower on the grammatical hierarchy than a clause.

Most phrases have an important word defining the type and linguistic features of the phrase. This word is the head of the phrase and gives its name to the phrase category. The heads in the following phrases are in bold:

too **slowly** - Adverb phrase (AdvP)

very **happy** - Adjective phrase (AP)

the massive **dinosaur** - Noun phrase (NP)

**at** lunch - [Preposition phrase](http://en.wikipedia.org/wiki/Preposition_phrase) (PP)

**watch** TV - [Verb phrase](http://en.wikipedia.org/wiki/Verb_phrase) (VP)

The head can be distinguished from its *dependents* (the rest of the phrase other than the head) because the head of the phrase determines many of the grammatical features of the phrase as a whole. The examples just given show the five most commonly acknowledged types of phrases. Further phrase types can be assumed, although doing so is not common. For instance one might acknowledge subordinator phrases:

**before** that happened - Subordinator phrase (SP)

This "phrase" is more commonly classified as a full subordinate [clause](http://en.wikipedia.org/wiki/Clause) and therefore many grammars would not label it as a phrase. If one follows the reasoning of heads and dependents, however, then subordinate clauses should indeed qualify as phrases. Most theories of syntax see most if not all phrases as having a head. Sometimes, however, non-headed phrases are acknowledged. If a phrase lacks a head, it is known as [exocentric](http://en.wikipedia.org/wiki/Exocentric), whereas phrases with heads are [endocentric](http://en.wikipedia.org/wiki/Endocentric).

**Head:** In linguistics, the head of a [phrase](http://en.wikipedia.org/wiki/Phrase) is the word that determines the [syntactic](http://en.wikipedia.org/wiki/Syntax) type of that phrase or analogously the [stem](http://en.wikipedia.org/wiki/Word_stem) that determines the semantic category of a [compound](http://en.wikipedia.org/wiki/Compound_%28linguistics%29) of which it is a part. The other elements modify the head and are therefore the head's *dependents*. Headed phrases and compounds are [endocentric](http://en.wikipedia.org/wiki/Endocentric), whereas [exocentric](http://en.wikipedia.org/wiki/Exocentric) ("headless") phrases and compounds (if they exist) lack a clear head. Heads are crucial to establishing the direction of [branching](http://en.wikipedia.org/wiki/Branching_%28linguistics%29). Head-initial phrases are right-branching, head-final phrases are left-branching, and head-medial phrases combine left- and right-branching.

Examine the following expressions:

big red **dog**

The word dog is the **head** of big red dog, since it determines that the phrase is a [noun phrase](http://en.wikipedia.org/wiki/Noun_phrase), not an adjective phrase. Because the adjectives big and red modify this head noun, they are its *dependents*.

**2.2 Approaches of Shallow Parser**

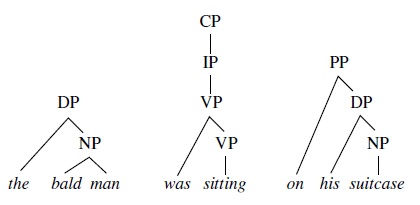
Various NLP research groups have developed different approaches and algorithm for shallow parsing. Some of the algorithms are language dependent and some of them are language independent. A brief survey of various approaches involved in Shallow parsing includes the following:

1. Parsing By Chunks
2. Machine Learning Approach for Shallow Parser
3. Memory based Shallow Parsing
4. HMM based Shallow Parsing
5. Text chunking based on generalized version of the Winnow Algorithm
6. Shallow Parsing with Conditional Random Fields

**2.2.1 Parsing By Chunks**

This approach was given by Steven P. Abney who used intuitive reading as the basis of parsing. He realized the fact that we read a sentence one chunk at a time. These chunks correspond in some way to prosodic patterns. A simple context-free grammar is quite adequate to describe the structure of chunks. The approach has its roots in the research done by Gee and Grosjean. However, Gee and Grosjean did not assign syntactic structure to chunks. To remedy these, Abney assumed that chunk has syntactic structure and defined them in terms of major heads. Major heads are all content words except those that appear between a function word f and the content word that f selects.

The parse tree segments associated with some sample chunks are illustrated in (2).



Chunker implemented is a non-deterministic version of an LR parser employing a best-first search. Grammar used in the published implementation can be found in the references section.

Since shallow parsers have to deal with the entire Natural Language, they need thousands of rules. This makes building shallow parsers a labor-intensive task. This was a big limitation of Abney’s approach which used hand-crafted cascaded FST (Finite State Transducers).

**2.2.2 Machine Learning Approach**

This approach was given by Ramshaw and Marcus in their paper published in 1995. It formulates NP-chunking as a tagging task. The approach targets higher level of chunk structure using Brill's transformation-based learning mechanism, in which a sequence of transformational rules is learned from a corpus; this sequence iteratively improves upon a baseline model for some interpretive feature of the text.

In this study, training and test sets marked with two different types of chunk structure were derived algorithmically from the parsed data in the Penn Treebank corpus of Wall Street Journal text (Marcus et al., 1994). The source texts were then run through Brill's part-of-speech tagger (Brill, 1993c), and, as a baseline heuristic, chunk structure tags were assigned to each word based on its part-of-speech tag. Rules were then automatically learned that updated these chunk structure tags based on neighbouring words and their part-of-speech and chunk tags.

The automatic derivation of training and testing data from the Treebank analyses allowed for fully automatic scoring, though the scores are naturally subject to any remaining systematic errors in the data derivation process as well as to *bona fide* parsing errors in the Treebank source. By representing text chunking as a kind of tagging problem, it became possible to easily apply transformation-based learning. The approach is able to automatically induce a chunking model from supervised training that achieves recall and precision of 92% for baseNP chunks and 88% for partitioning N and V chunks.

The approach suffered from disadvantages inherent in the Machine Learning approach itself. The main disadvantage with ML approach is that the labelled training material is frequently noisy and exists in small quantity. Moreover, real world sentences tend to be long. Learners which do not operate in (near) linear time are simply unfit for the task.

**2.2.3 Memory Based Shallow Parsing**

Memory-Based Learning (MBL) is a classification based supervised learning approach: a memory-based learning algorithm constructs a classifier for a task by storing a set of examples. Each example associates a feature vector (the problem description) with one of a finite number of classes (the solution). Given a new feature vector, the classifier extrapolates its class from those of the most similar feature vectors in memory. The metric defining similarity can be automatically adapted to the task at hand.

Algorithms generally employed (example IBI-IG and IGTrtEE) are simple and efficient supervised learning algorithms. When compared with new memory-based learning algorithm, memory-based sequence learning (MBSL, [Argamon *et al.,* 1998] MBL is a computationally simpler algorithm, is able to reach similar precision and recall when restricted to the MBSL definition of the NP chunking, subject detection and object detection tasks. More importantly, MBL is more flexible in the definition of the shallow parsing tasks: it allows nested relations to be detected; it allows the addition and integration into the task of various additional sources of information apart from POS tags; it can segment a tagged sentence into different types of constituent chunks in one pass; it can scan a chunked sentence for different relation types in one pass (though separating subject-verb detection from object-verb detection is surely an option that must be investigated).

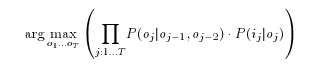
Clear advantages of MBL are its efficiency (especially when using IGTREE), the ease with which information apart from POS tags can be added to the input (e.g. word information, morphological information, wordnet tags. chunk information for subject and object detection), and the fact that NP and VP chunking and different types of relation tagging can be achieved in one classification pass.

A major disadvantage of the approach is that it is unclear how MBSL could be extended to incorporate other sources of information apart from POS tags, and what the effect would be on performance. More limitations of MBSL are that it cannot find nested sequences, which nevertheless occur frequently in tasks such as subject identification, and that it does not mark heads.

**2.2.4 HMM Based Shallow Parsing**

This is a unified technique to solve diﬀerent shallow parsing tasks as a tagging problem using a Hidden Markov Model-based approach (HMM). This technique consists of the incorporation of the relevant information for each task into the models. To do this, the training corpus is transformed to take into account this information. In this way, no change is necessary for either the training or tagging process, so it allows for the use of a standard HMM approach.

Initially, shallow parsing is treated as HMM-based tagging. We consider shallow parsing to be a tagging problem. From the statistical point of view, tagging can be solved as a maximization problem. Due to the fact that this maximization process is independent of the input sequence, and taking into account the Markov assumptions, the problem is reduced to solving the following equation (for a second–order HMM):



The parameters of the equation can be represented as a second-order HMM whose states correspond to a tag pair. Contextual probabilities, P (oj |oj−1, oj−2 ), represent the transition probabilities between states and P (ij |oj ) represents the output probabilities.

HMM based solution allows us to tackle diﬀerent natural language disambiguation tasks as tagging problems. Using this technique, the relevant information for both shallow parsing and clause identification can be determined. Thus, a speciﬁc task can be performed using a standard HMM-based tagger without modifying the learning and testing processes. The results reported show that the HMM approach performs in line with other approaches that use more sophisticated learning methods when an appropriate deﬁnition of the input and output vocabularies is provided. Moreover, this approach maintains the eﬃciency of the system throughout both the learning and the testing phases.

**2.2.5 Text chunking based on generalized version of the Winnow Algorithm**

Text chunking based on generalized version of the Winnow Algorithm was proposed by Tong Zang, Fred Damerau and David Johnson. The approach proposes a general statistical model for text chunking which is then converted into a classiﬁcation problem. The Winnow family of algorithms is particularly suitable for solving classiﬁcation problems arising from NLP applications, due to their robustness to irrelevant features. However in theory, Winnow may not converge for linearly non-separable data. To remedy this problem, a generalization of the original Winnow method is employed. An additional advantage of the new algorithm is that it provides reliable conﬁdence estimates for its classiﬁcation predictions. This property is required in statistical modelling approach. The system achieves state of the art performance in text chunking with less computational cost then previous systems.

The advantage of the new method compared with the original Winnow is its ability to handle linearly non-separable data and its ability to provide reliable confidence estimates. Such confidence estimates are required in the statistical sequential modelling approach to the text chunking problem.

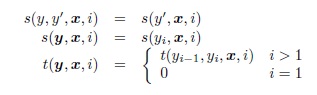
The approach achieved the best result for a non-ensemble classifier in the CoNLL-2000 shared task.

**2.2.6 Shallow Parsing with Conditional Random Fields**

Conditional random fields for sequence labelling offer advantages over both generative models like HMMs and classifiers applied at each sequence position. The generative approach provides well-understood training and decoding algorithms for HMMs and more general graphical models. However, effective generative models require stringent conditional independence assumptions. For instance, it is not practical to make the label at a given position depend on a window on the input sequence as well as the surrounding labels, since the inference problem for the corresponding graphical model would be intractable. The sequential classification approach can handle many correlated features, as demonstrated in work on maximum-entropy (McCallum et al., 2000; Ratnaparkhi,1996) and a variety of other linear classifiers, including winnow (Punyakanok and Roth, 2001), AdaBoost (Abney et al., 1999), and support-vector machines (Kudo and Matsumoto, 2001). Furthermore, they are trained to minimize some function related to labelling error, leading to smaller error in practice if enough training data are available.

Conditional random fields (CRFs) bring together the best of generative and classification models. Like classification models, they can accommodate many statistically correlated features of the inputs, and they are trained discriminatively. But like generative models, they can trade off decisions at different sequence positions to obtain a globally optimal labeling. Lafferty et al. (2001) showed that CRFs beat related classification models as well as HMMs on synthetic data and on a part-of-speech tagging task.

A CRF on (X; Y ) is specified by a vector f of *local features* and a corresponding *weight vector λ*. Each local feature is either a *state feature* s(y; x; i) or a *transition feature* t(y; y0; x; i), where y; y0 are labels, x an input sequence, and i an input position. To make the notation more uniform, we also write

****

for any state feature s and transition feature t.

The chunking CRFs have a second-order Markov dependency between chunk tags. This is easily encoded by making the CRF labels pairs of consecutive chunk tags.

(Log)-linear sequence labelling models trained discriminatively with general-purpose optimization methods are a simple, competitive solution to learning shallow parsers. These models combine the best features of generative finite-state models and discriminative (log)-linear classifiers, and do NP chunking as well as or better than ‘ad hoc’ classifier combinations, which were the most accurate approach until now.

However, full discriminative parser training faces significant algorithmic challenges in the relationship between parsing alternatives and feature values (Geman and Johnson, 2002) and in computing feature expectations.

**2.3 Morphology of Hindi Language**

This section deals with the “Hindi” language morphological structure of different word classes, describing their inflectional and derivational forms of Hindi language. Word classes described include nouns, pronouns, adjectives, verbs, adverbs, particles, connectives, and interjections.

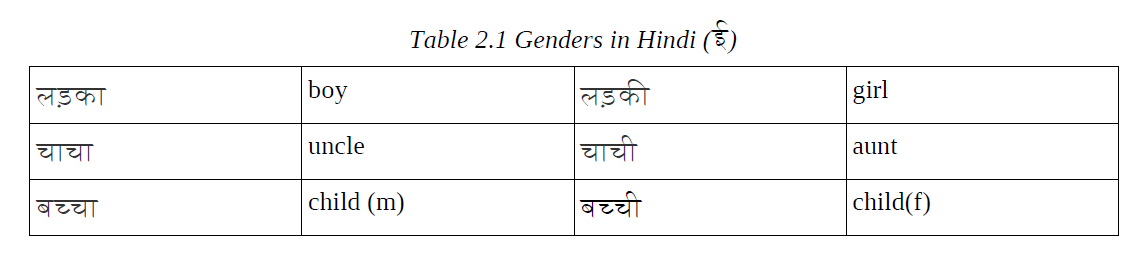
**2.3.1 Nouns**

In the following section there is brief introduction to the Noun-Inflection, Noun-Derivations, and Noun-Compounds according to the grammatical structure of the Noun words.

**Noun Inflection:** Nouns in Hindi are inflected for gender, number, and case. There are three declensions of nouns; Declension I includes आ /a:/ ending masculine nouns; Declension II includes all other masculine nouns; and Declension III includes all feminine nouns.

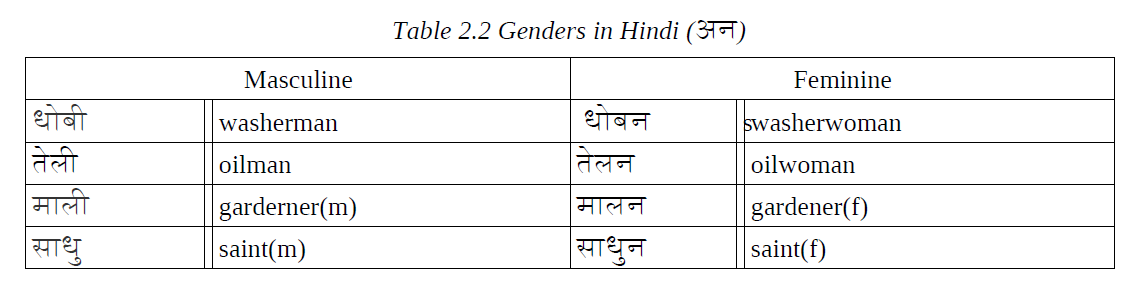
• **Gender**

There are two genders in Hindi: masculine and feminine. Besides the natural gender of animate nouns, every inanimate noun is assigned a gender. Though the gender of a large number of inanimate nouns can be predicted by their endings, there are no hard and fast rules for assigning the genders. Masculine forms are traditionally taken as basic. The gender formation involves (a) suffixation, (b) phonological changes, and (c) suppletion. We can make some general observations as follows. (i) Most of the आ /a:/ ending masculine nouns have their feminine forms ending in ई /i:/.

****

In the above examples, the final - आ/-a:/ in the masculine nouns is replaced by -ई /-i:/ in their feminine forms .

(ii) Most of the - ई /-i:/ ending animate masculine nouns have their feminine forms ending in –अन /-an/.

****

(iii) Some nouns ending in - आ /-a:/ form their feminine (diminutive) by replacing - आ /-a:/ with –इया /-iya:/.

e.g.

डबबा box

िडिवया a small box

(iv) Most of the - आ /-a:/ ending inanimate nouns are masculine and - ई /-i:/ ending inanimate nouns are feminine.

e.g.

पंखा fan

पंखी a small fan

In the above examples, the final - आ /a:/ in the masculine forms is replaced by the suffix – ई/i:/.

(v) The suffix -नी /-ni:/ is added to the masculine nouns to form the feminine.

e.g.

शेर (lion)

शेरनी (lioness)

(vi) The suffix -ई/-i:/ is added to the masculine nouns to form the feminine.

e.g.

पुत son

पुती daughter

• **Number**

There are two numbers: **singular** and **plural**.

(i) The - आ/-a:/ ending masculine nouns (including pronouns and adjectives), with a few exceptions change into -ए/-e/ ending forms in the plural.

e.g.

लड़का लड़के

घोड़ा घोड़े

(ii) All other consonant and/or other vowel-ending nouns do not change in their plural forms.

e.g.

मोर कोट गांव

(iii) The feminine plurals are formed by adding the suffix -एं /ẽ/ to the consonant-ending singular forms.

e.g.

िकताब िकताबे

मेज मेजे

गाय गाये

(iv) The plural suffix -इयाँ -iya: is added to the -ई -i: ending feminine nouns.

e.g.

लड़की लड़िकयाँ

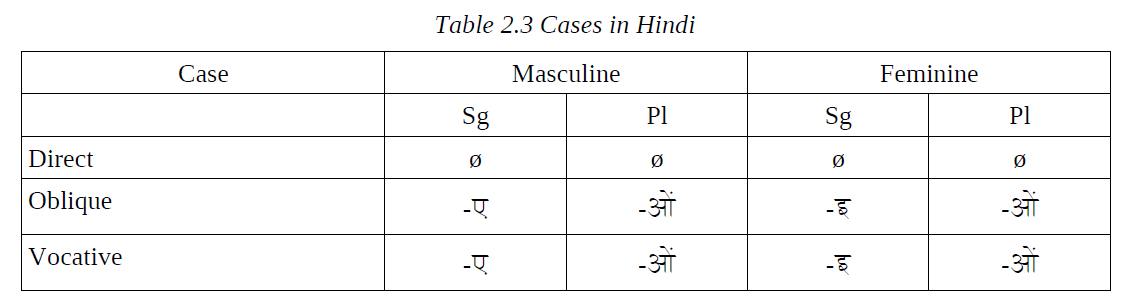
कुसी कुिसरयाँ

कहानी कहािनयाँ

Notice that when the suffix is added the final vowel of the stem is deleted.

• **Case**

The syntactic and semantic functions of noun phrases are expressed by case-suffixes, postpositions and derivational processes. There are two cases: direct and oblique. Case-suffixes and postpositions are used to express syntactic and semantic functions. Case suffixes are defined as bound suffixes, which do not occur independently as words and are added only to the noun phrases. Case suffixes added to the oblique forms of nouns agreeing in number and gender.

****

The vocative address forms may be preceded by the vocative morphemes ओं o/ हे he/ अरे are.

• **Postpositions**

Postpositions have specific semantic functions. They express the semantic dimensions of a noun such as benefaction, manner, or location. The main postpositions are: ने ne ‘ergative marker’; को ko ‘to’; के िलये ke liye ‘for’; पर par ‘on’; मे mẽ ‘in’; से se ‘from’; से se ‘with’; का /को /की ka/ke/ki: ‘of’.’

The postpositions are written as separate words with nouns (अिमत ने amit ne,उमा को uma: ko), but they are tagged to pronouns ( मैने m ~́ ne उसको usko, िकसका kiska:).

**Noun Derivatives:** A large number of nouns in Hindi are derived from nouns, adjectives, and verbs by using prefixes and suffixes. In this process certain morphophonemic changes take place.

• **Nouns from Noun**

Mostly Persian and Sanskrit prefixes and suffixes are used with the nouns of Persian and Sanskrit origin respectively. Some of these are used with native words. The most common prefixes are: वे be-, बद bad-, बर bar-, ना na:- अप ap-, कु ku-, दरु ्dur-, and िनर nir-.

e.g.

ईमान बेईमान

तमीज बदतमीज

पसंद नापसंद

• **Nouns from adjectives**

The most productive suffixes used for deriving abstract nouns from adjectives are -इ -i:, -ता -ta:, -pan, -आइ -a:i:, - इयत -iyat, - आस -a:s.

e.g.

कमजोर कमजोरी

मूखर मूखरता

कमीना कमीनापन

• **Noun from Verbs**

The suffix -ना -na: is used to derive gerundive nouns from verb stems. The suffixes -अस -as, -अन -an, -ई: -ie, -वट -vat, and -2 are also used to derive abstract nouns from verb stems.

e.g.

आ आना

पढ पढना

धडक धडकन

जोड जोडी

**Noun Compounds:** Compounds belonging to the noun category are headed by a noun, which is a final

member of the group. The first member may be a noun, an adjective, or a participle and may be

declined for number, gender and case. A postposition is attached to the final member of the compound.

• **Noun Noun Compounds**

Noun-noun compounds can be divided into several subgroups based on semantic criteria: copulative

compounds, partial duplicated compounds, superordinate compounds, complex compounds, hybrid

compounds, genitive-noun compound, and participial compounds.

• **Copulative Compounds**

Copulative compounds, also known as co-compounds, are composed of semantically-related nouns.

Each noun behaves as an independent constituent in the sense that each may be separately inflected for

gender and number, though not for a postposition. Members of some compounds occur in a fixed order.

e.g.

माता िपता

भाई बहन