1.Discuss about morphological models?

There are many possible approaches to design and implement morphological models

The most prominent types of computational approaches to morphology are.

1. **Dictionary lookup:** This approach involves maintaining a dictionary or lexicon containing mappings between word forms and their morphological analyses. When a word needs to be analyzed or generated, the system simply looks up the word in the dictionary to retrieve its morphological properties.

   It is simple and efficient for languages with relatively simple morphology, this approach may struggle with languages having extensive inflectional or derivational systems.

   Dictionaries can be implemented, for instance, as lists, binary search trees, tries, hash tables, and so on.

2. **Finite state morphology:** The specifications written by human programmers are directly compiled into finite-state transducers. Finite-state transducers are computational devices extending the power of finite-state automata. They consist of a finite set of nodes connected by directed edges labeled with pairs of input and output symbols. In such a network or graph, nodes are also called states, while edges are called arcs. Traversing the network from the set of initial states to the set of final states along the arcs is equivalent to reading the sequences of encountered input symbols and writing the sequences of corresponding output symbols.

   The set of possible sequences accepted by the transducer defines the input language; the set of possible sequences emitted by the transducer defines the output language.

   Finite state morphological analyzers can perform tasks such as morphological analysis, generation, and tokenization efficiently. They are widely used due to their simplicity, scalability, and ability to handle complex morphological phenomena.

3. **Unification based morphology**: The concepts and methods of these formalisms are often closely connected to those of logic programming. In this approach, linguistic rules and constraints are represented as feature structures, and morphological analysis involves unifying these structures to generate or analyze word forms. Feature structures can be viewed as directed acyclic graphs. Unification is the key operation by which feature structures can be merged into a more informative feature structure.

   Morphological models of this kind are typically formulated as logic programs, and unification is used to solve the system of constraints imposed by the model.

   Advantages of this approach include better abstraction possibilities for developing a morphological grammar as well as elimination of redundant information from it.

Unification-based systems are often used for languages with rich morphological systems and complex linguistic phenomena such as Russian, Czech, Slovene, Persian, Hebrew, Arabic, and other languages.

.

4. **Functional morphology:** Functional morphology focuses on the functional aspects of morphological processes, emphasizing the relationships between form and meaning. It treats morphological operations and processes as pure mathematical functions and organizes the linguistic as well as abstract elements of a model into distinct types of values and type classes.

   Linguistic notions like paradigms, rules and exceptions, grammatical categories and parameters, lexemes, morphemes, and morphs can be represented intuitively (without conscious reasoning; instinctively) and succinctly (in a brief and clearly expressed manner) in this approach.

2. Describe words and components in NLP?

There are two components of NLP, Natural Language Understanding (NLU) and Natural Language Generation (NLG).

- **Natural Language Understanding (NLU)** which involves transforming human language into a machine-readable format.

It helps the machine to understand and analyse human language by extracting the text from large data such as keywords, emotions, relations, and semantics.

- **Natural Language Generation (NLG)** acts as a translator that converts the computerized data into natural language representation.It mainly involves Text planning, Sentence planning, and Text realization. The NLU is harder than NLG.

In NLP, Words are the smallest linguistic units that can form complete utterances in most languages. They are composed of several integral parts.

The three important terms which are integral parts of a word are:

**Phonemes –** the distinctive units of sound in spoken language. For example, the words "pat" and "bat" differ only in their initial phoneme (/p/ and /b/).

**Graphemes –** Graphemes are the written symbols used to represent phonemes in a writing system. They are the smallest units of a written language which corresponds to a phoneme.

**Morphemes -** the minimal part of a word that delivers aspects of meaning to the word.

Some of the key concepts related to words are

**Tokens:** The process of breaking text into smaller units, each unit is often referred to as a token. Word tokens are the individual words extracted from a text after tokenization.

For example, in the sentence "The cat is sleeping," the word tokens are "The," "cat," "is," and "sleeping."

**Lexemes**: Lexemes are sets of alternative forms that a word can take and form the lexicon of a language. They are categorized by their grammatical categories like verbs, nouns, adjectives.

The citation form of a lexeme by which it is identified is called lemma.
When we convert a word into its other forms, such as turning the singular mouse into the plural mice or mouses, we say we **inflect** the lexeme.

When we transform a lexeme into another one that is morphologically related, regardless of its lexical category, we say we **derive** the lexeme:

for instance, the nouns receiver and reception are derived from the verb to receive.

Example: Did you see him? I didn't see him. I didn't see anyone.
Example presents the problem of tokenization of didn't and the investigation of the internal structure of anyone.

**Morphology:** It is the study of the structure and form of words in a language. Words are built up of minimal meaningful elements called morphemes:
Ex: played = play-ed
cats = cat-s
unfriendly = un-friend-ly

Morphological analysis – exploring the structure of words

**Typology:** Morphological typology divides languages into groups by characterizing the prevalent morphological phenomena in those languages. It can consider various criteria, and during the history of linguistics, different classifications have been proposed. Typology that is based on quantitative relations between words, their morphemes, and their features

- Isolating, or analytic, languages include no or relatively few words that would comprise more than one morpheme (typical members are Chinese, Vietnamese, and Thai; analytic tendencies are also found in English).
- Synthetic languages can combine more morphemes in one word and are further divided into agglutinative and fusional languages.
- Agglutinative languages have morphemes associated with only a single function at a time (as in Korean, Japanese, Finnish, and Tamil, etc.)
- Fusional languages are defined by their feature-per-morpheme ratio higher than one (as in Arabic, Czech, Latin, Sanskrit, German, etc.).

3. Briefly describe the issues and challenges in finding the structure of words?

**Irregularity:** Many words in a language do not follow standard morphological rules.
Ex: English verbs like "go" (past tense "went") and nouns like "mouse" (plural "mice").
Standard algorithms and models that rely on predictable patterns struggle with these exceptions. Irregular forms need to be explicitly learned and stored, complicating the processing and increasing the computational resources required.

**Ambiguity:** A single word form can have multiple meanings or grammatical functions depending on its context.

Ex: The word "lead" can mean to guide (verb) or a type of metal (noun). The word "bark" can mean the sound a dog makes or the outer covering of a tree.
Disambiguating these words requires understanding the surrounding text. Simple morphological analysis isn't enough; systems must incorporate syntactic and semantic analysis to correctly interpret the intended meaning.

**Productivity:** The ability of a language to create new words and word forms.

Ex: New technological terms like "googling" or new compound words like "work-from-home".

The lexicon of a language is continually expanding. NLP systems need to be flexible and adaptive to recognize and process newly coined words or forms that they haven't encountered before. This requires dynamic learning capabilities and continual updates to the language models.

4. Define parsing. Consider the grammar

N -> N and N
N -> N or N
N -> a | b | c

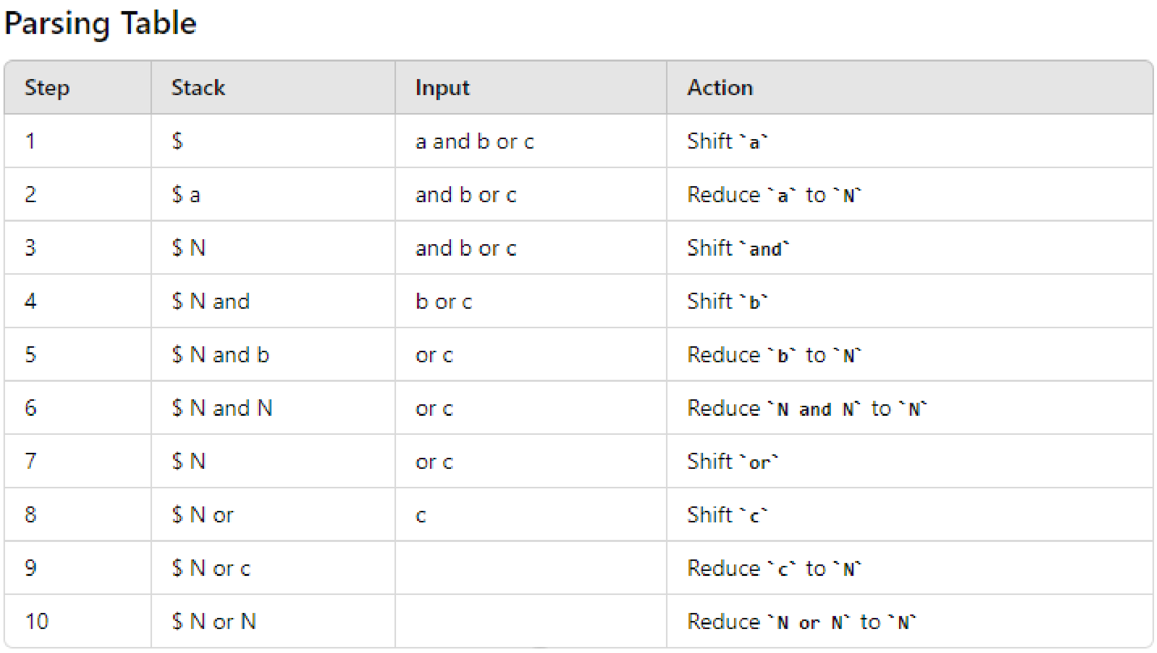
Perform Shift Reduce parsing for input string a and b or c.

This parser requires some data structures i.e.

- An input buffer for storing the input string.
- A stack for storing and accessing the production rules.

**Basic Operations –**

- Shift: This involves moving symbols from the input buffer onto the stack.
- Reduce: If the handle appears on top of the stack then, its reduction by using appropriate production rule is done i.e. RHS of a production rule is popped out of a stack and LHS of a production rule is pushed onto the stack.
- Accept: If only the start symbol is present in the stack and the input buffer is empty then, the parsing action is called accept. When accepted action is obtained, it is means successful parsing is done.
- Error: This is the situation in which the parser can neither perform shift action nor reduce action and not even accept action.



Parsing Table

| Step | Stack | Input | Action |
|---|---|---|---|
| 1 | $ | a and b or c | Shift `a` |
| 2 | $ a | and b or c | Reduce `a` to `N` |
| 3 | $ N | and b or c | Shift `and` |
| 4 | $ N and | b or c | Shift `b` |
| 5 | $ N and b | or c | Reduce `b` to `N` |
| 6 | $ N and N | or c | Reduce `N and N` to `N` |
| 7 | $ N | or c | Shift `or` |
| 8 | $ N or | c | Shift `c` |
| 9 | $ N or c |  | Reduce `c` to `N` |
| 10 | $ N or N |  | Reduce `N or N` to `N` |

Final state: $N

The input string "a and b or c" is successfully parsed according to the given grammar, resulting in the stack containing the start symbol N with initial stack state$.

5. Explain different approaches used to find the structure of documents?

In human language, words and sentences do not appear randomly but usually have a structure. For example, combinations of words form sentences—meaningful grammatical units, such as statements, requests, and commands. Likewise, in written text, sentences form paragraphs—self-contained units of discourse about a particular point or idea. Sentences may also be related to each other by explicit discourse connectives such as *therefore*.

Automatic extraction of structure of documents helps subsequent natural language processing (NLP) tasks; for example, parsing, machine translation, and semantic role labeling use sentences.

Two types of segmentation

**Sentence boundary detection:** The task of deciding where sentences start and end given a sequence of characters.
**Topic segmentation:** The task of determining when a topic starts and ends in a sequence of sentences.

**Approaches for sentence and topic segmentation:**

Let’s assume our task is to determine the language of a text document. How do we solve this task with the help of machine learning?

We can learn each language and then determine the language. This is how generative models work.

Alternatively, we can learn just the linguistic differences and common patterns of languages without actually learning the language. This is the discriminative approach. In this case, we don’t speak any language.

Given a boundary candidate( between two word tokens for sentence segmentation and between two sentences for topic segmentation), the goal is to predict whether or not the candidate is an actual boundary (sentence or topic boundary).

let xƐX be the vector of features (the observation) associated with a candidate and y ƐY be the label predicted for that candidate. The label y can be b for boundary and B¯ for non-boundary.

Classification problem: given a set of training examples$(x,y)_{train}$, find a function that will assign the most accurate possible label y of unseen examples $x_{unseen.}$

The classification can be done at each potential boundary *i* (local modelling); then, the aim is to estimate the most probable boundary type $\hat{y}_i$for each candidate $x_i$

$$\hat{y} = \underset{y_i\, in\, Y}{argmax} P(y_i|x_i)$$

- **<u>Generative sequence classification method:</u>**
  Most commonly used generative sequence classification method: Hidden Markov Model. The above probability can be rewritten as

$$\hat{Y} = \underset{Y}{\operatorname{argmax}} P(Y|X) = \underset{Y}{\operatorname{argmax}} \frac{P(X|Y)P(Y)}{P(X)} = \underset{Y}{\operatorname{argmax}} P(X|Y)P(Y)$$

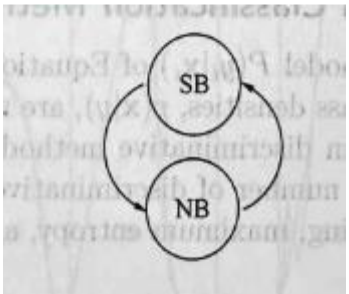
$P(X)$ in the denominator is dropped because it is fixed for different $Y$ and hence does not change the argument of *max*. $P(X|Y)$, and $P(Y)$ can be estimated as

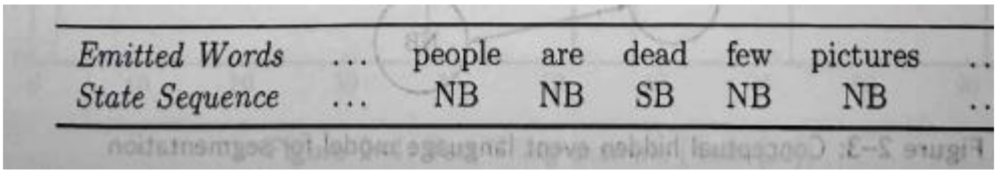
$$P(X|Y) = \prod_{i=1}^{n} P(\mathrm{x}_i|y_1, \ldots, y_i) \qquad P(Y) = \prod_{i=1}^{n} P(y_i|y_1, \ldots y_{i-1})$$

Which can be further simplified to

$$P(\mathrm{x}_i|y_1, \ldots, y_i) \approx P(\mathrm{x}_i|y_i) \qquad P(y_i|y_1, \ldots, y_{i-1}) \approx P(y_i|y_{i-1})$$

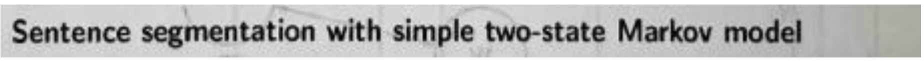
Ex: HMM with 2 states; NB Non boundary, SB-Sentence boundary





| *Emitted Words* | ... | people | are | dead | few | pictures | .. |
|---|---|---|---|---|---|---|---|
| *State Sequence* | ... | NB | NB | SB | NB | NB | .. |

Fig:



**Sentence segmentation with simple two-state Markov model**

- **<u>Discriminative local classification method:</u> Discriminative algorithms focus on modeling a direct solution.** For example, the logistic regression algorithm models a decision boundary. Then it decides on the outcome of an observation based on where it stands relative to the decision boundary.
  Discriminative classifiers aim to model P(yi |xi) directly.

  Some popular discriminative algorithms are:
  k-nearest neighbors (k-NN), Logistic regression, Support Vector Machines (SVMs), Decision Trees, Random Forest, Artificial Neural Networks (ANNs)

The most Text tiling method Hearst for topic segmentation uses a lexical cohesion metric in a word vector space as an indicator of topic similarity.

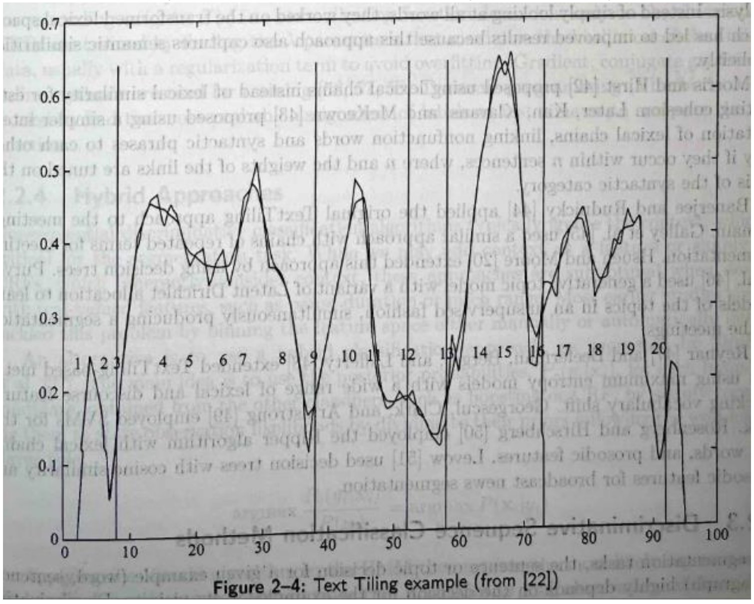


Figure 2–4: Text Tiling example (from [22])

Fig shows the typical graph of similarity with respect to consecutive segmentationn units. The document is chopped when the similarity is below threshold.

- **<u>Discriminative sequence classification method:</u>** It is an extension of discriminative local classification method that finds the best assignment of labels by looking at the neighboring decisions to label an example.

  Conditional random fields(CRFs) are extension of maximum entropy, SVM struct is an extension of SVM, and maximum margin Markov networks(M3N) are extensions of HMM. CRFs are a class of log-linear models for labelling structures.

- **Hybrid approaches:** Non sequential discriminative classification models typically ignore the context, which is critical for segmentation task. We may add context as a feature, but they are sub optimal when dealing with real valued features such as pause duration or pitch range.

  The main idea in hybrid approach is to use posterior probabilities $P_c(y_i|\mathbf{x}_i)$, for each boundary candidate, obtained from other classifiers, such as boosting or CRF, by simply converting them to state observation likelihoods by dividing them to their priors following the bayes rule.

$$\operatorname*{argmax}_{y_i} \frac{P_c(y_i|\mathbf{x}_i)}{P(y_i)} = \operatorname*{argmax}_{y_i} P(\mathbf{x}_i|y_i)$$

6. Discuss the challenges in multilingual texts?
Multilingual texts pose several challenges for natural language processing (NLP) systems due to the diversity of languages, writing systems, and linguistic features.

- Identifying the language(s) present in a multilingual text is a fundamental task but can be challenging, especially when languages are mixed within the same document or sentence.
- Texts may contain code-switching, where multiple languages are used interchangeably within the same discourse, making it difficult to determine the predominant language.
- Different languages may use distinct writing systems (e.g., Latin script, Cyrillic script, Arabic script), each with its own set of characters, symbols, and diacritics.
- Languages vary in morphological complexity, with some having rich inflectional systems (e.g., Turkish) while others are more isolating (e.g., Mandarin Chinese).

7. Explain how CKY parsing constructs the parsing chart for a given sentence and context-free grammar.
CYK means cocke-Kasami-Younger
It is a parsing algorithm for context free grammar.
Inorder to apply CYK algorithm to a grammar, it must be in chomsky normal form. It uses dynamic programming algorithm to tell whether a string is in the language of a grammar.

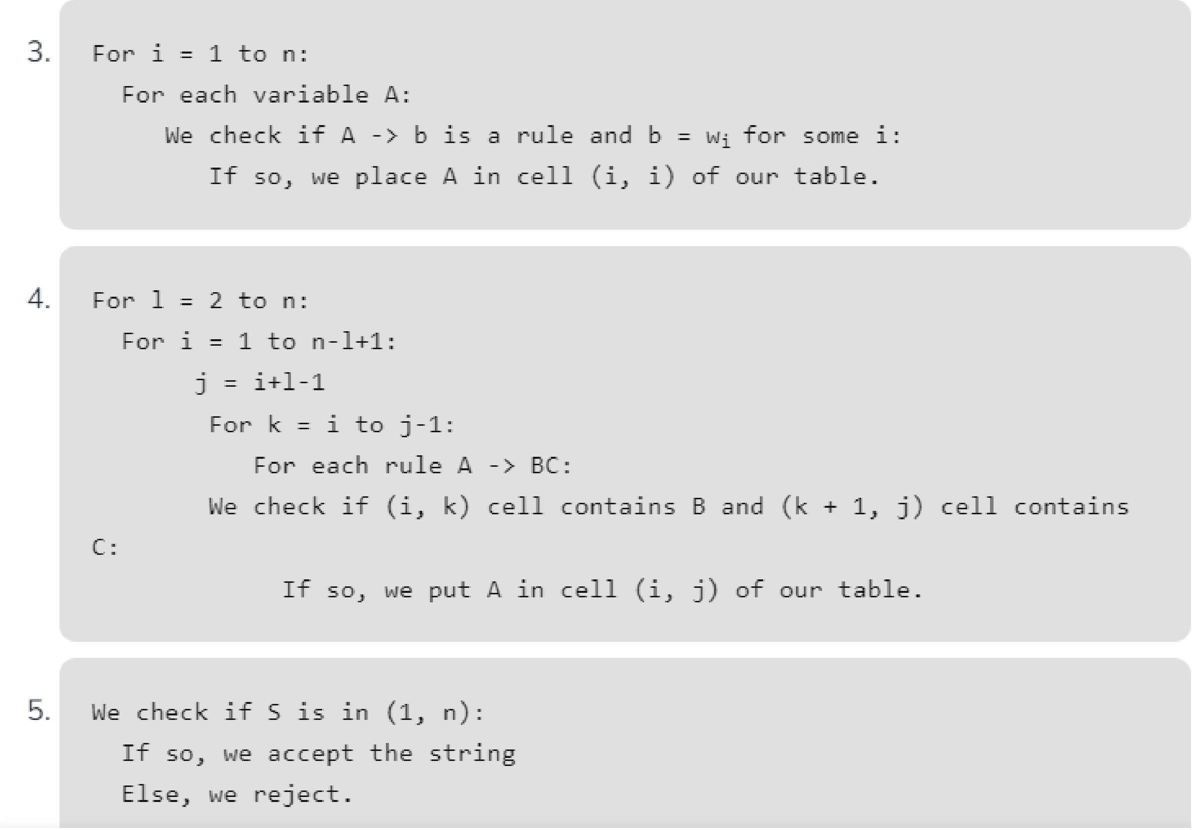
**Chomsky Normal Form:**
A Context Free Grammar G is in Chomsky Normal Form (CNF) if each rule if each rule of *G* is of the form:

- *A* –> *BC*, [ with at most two non-terminal symbols on the RHS ]
- *A* –> *a*, or [ one terminal symbol on the RHS ]
- *S* –> nullstring, [ null string ]

**Algorithm** :
Let w be the n length string to be parsed. And G represent the set of rules in our grammar with start state S.

1. Construct a table DP for size n × n.

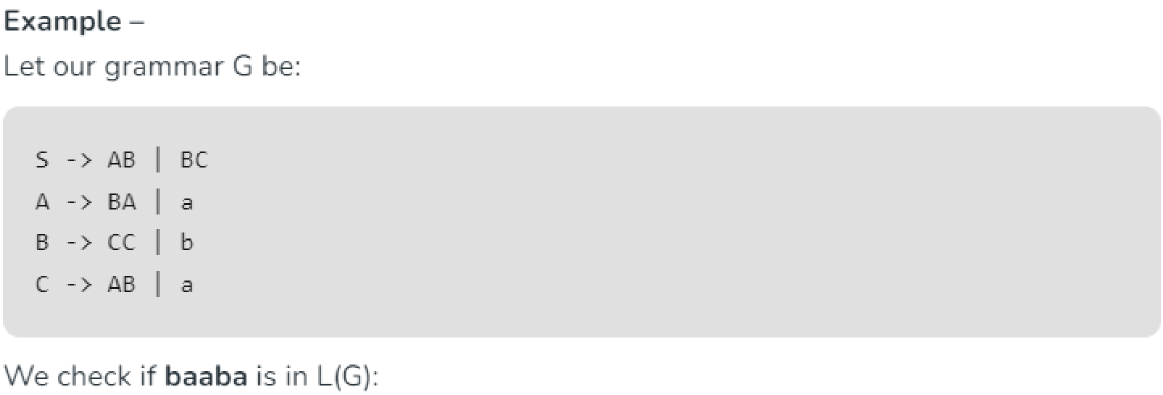
2. If w = e (empty string) and S -> e is a rule in G then we accept the string else we reject.



3.
```
For i = 1 to n:
  For each variable A:
     We check if A -> b is a rule and b = w_i for some i:
        If so, we place A in cell (i, i) of our table.
```

4.
```
For l = 2 to n:
  For i = 1 to n-l+1:
       j = i+l-1
        For k = i to j-1:
           For each rule A -> BC:
        We check if (i, k) cell contains B and (k + 1, j) cell contains
C:
             If so, we put A in cell (i, j) of our table.
```

5.
```
We check if S is in (1, n):
  If so, we accept the string
  Else, we reject.
```

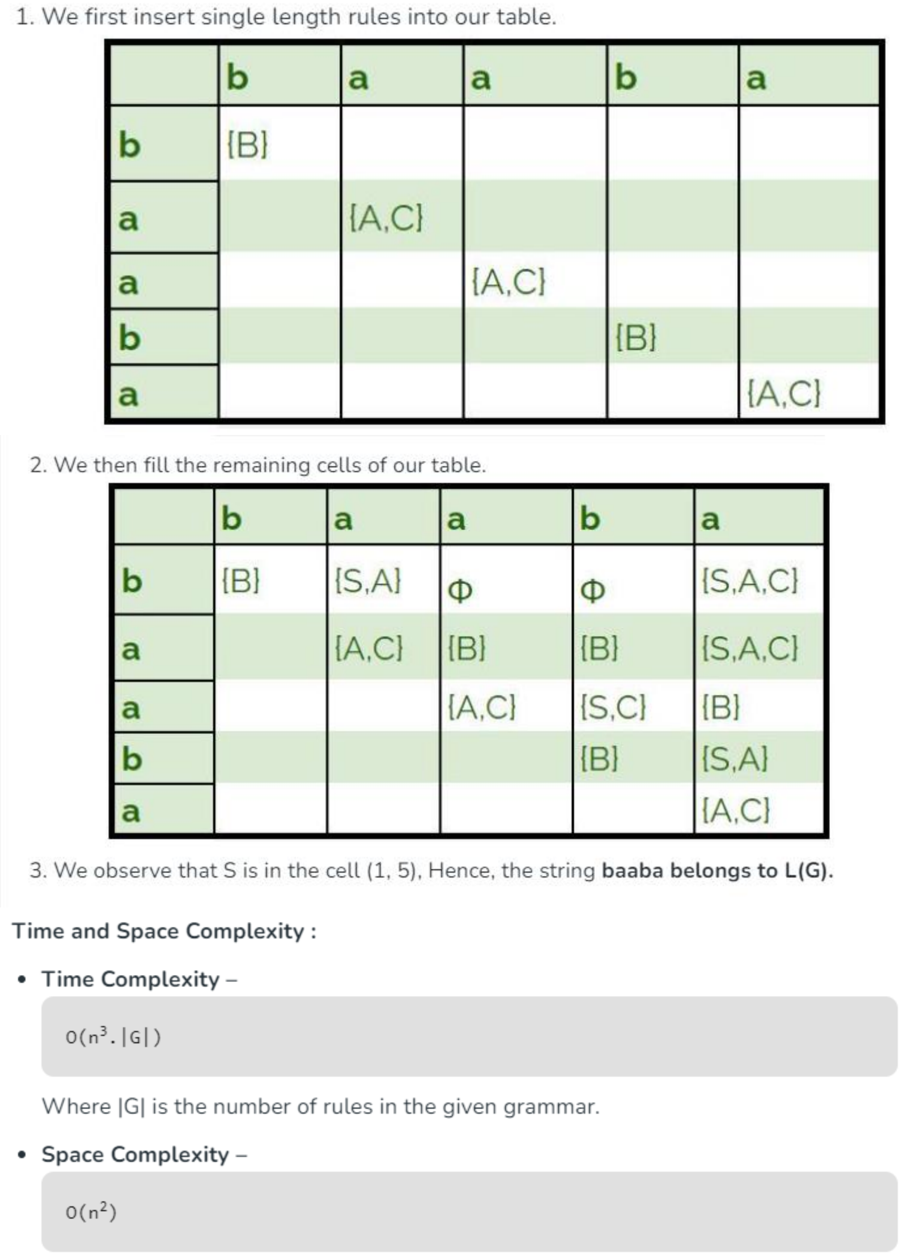


**Example** –

Let our grammar G be:

```
S -> AB | BC
A -> BA | a
B -> CC | b
C -> AB | a
```

We check if **baaba** is in L(G):



1. We first insert single length rules into our table.

| | b | a | a | b | a |
|---|---|---|---|---|---|
| b | {B} | | | | |
| a | | {A,C} | | | |
| a | | | {A,C} | | |
| b | | | | {B} | |
| a | | | | | {A,C} |

2. We then fill the remaining cells of our table.

| | b | a | a | b | a |
|---|---|---|---|---|---|
| b | {B} | {S,A} | Φ | Φ | {S,A,C} |
| a | | {A,C} | {B} | {B} | {S,A,C} |
| a | | | {A,C} | {S,C} | {B} |
| b | | | | {B} | {S,A} |
| a | | | | | {A,C} |

3. We observe that S is in the cell (1, 5), Hence, the string **baaba belongs to L(G).**

**Time and Space Complexity :**

- **Time Complexity –**

  $O(n^3.|G|)$

  Where |G| is the number of rules in the given grammar.

- **Space Complexity –**

  $O(n^2)$

Advantages of CYK: It is used to solve the membership problem using a dynamic programming approach. The algorithm is based on the principle that the solution to problem *[i, j]* can constructed from solution to subproblem *[i, k]* and solution to sub problem *[k, j]*.

The **Membership problem** is defined as: Grammar G generates a language L(G). Is the given string a member of L(G)?