Assignment

Chapter 4

Part A:

1. Perform the pairwise dis-jointness test for the following grammar rules (State whether each pass or fails).

a. A → aB | b | cBB

first(aB) = a

first(b) = b

first(cBB)= c

[PASS]

b. B → aB | bA | aBb

first(aB) = ab

first(bA) = b

first(aBb) = a

[FAIL]

c. C → aaA | b | caB

first(aaA) = a

first(b) = b

first(caB) = c

[PASS]

2. Give the three reasons why lexical analysis is separated from syntax analysis.

1. The lexical analysis is simpler than the syntax analysis, so the lexical analysis process can be easier by separated.

2. By separation, the lexical analyzer can be optimized without thinking about the affection of syntax analyzer, because to optimize the syntax analyzer sometimes is not necessary.

3. The lexical analyzer reads input program files and includes buffering in that input, so it is platform dependent, but syntax analyzer can be platform independent.

3. What are the primary tasks of a lexical analyzer?

The main tasks include to skip comments and white space outside the lexemes, because they have no meaning to the program. The other task is to inserts lexemes for user-defined names to the symbol table, so the names can be used later. Lexical analyzer also detect syntactic error for the program.

4. What are the two distinct goals of syntax analysis?

The first goal for the syntax analyzer is to check any input syntax and make sure they are correct. If it finds error, then it must output a diagnostic message and go back to its normal state and continuing check the program.

The other goal is to produce complete parse tree, and it will be used as the basis for translation for checking the correct input.

5. Describe the differences between top-down and bottom up parsers.

Top-down parsing, also known as “Left-most derivation”, as the parsing mimics doing a leftmost derivation of a sentence. The parse tree is constructed top to down. It considers “which production” for every point of generating.

Bottom-up parsing, also known as “Right-most derivation”, as the parsing mimics doing a rightmost derivation of a sentence. The parse tree is constructed bottom to up. It considers “When to reduce” for every point of generating.

6. Describe the parsing problems for a top-down parser.

Assume A-rules are A->bB, A->cBb, and A->a, the top-down parser has to choose among these three rules to get the next sentential form, which can be xbB, xcBb, or xa. This is the parsing decision problem for top-down parsers.

7. Describe the parsing problem for a bottom-up parser.

The bottom-up parser only identify and process the low level details of the structure of the text before going to the secondary level, and left the entire structure of the highest level.

8. Briefly describe what a phrase, simple phrase and a handle is of a sentential form.

Phrase: The string of terminals collected from the leaves of the partial parse tree, rooted at one specific internal node in the original parse tree.

Simple phrase: A phrase that takes a single derivation step from its root nonterminal node.

Handle: The handle of a sentential form defined as the right-hand-side form that should be rewritten to yield the next sentential form in the right most derivation.

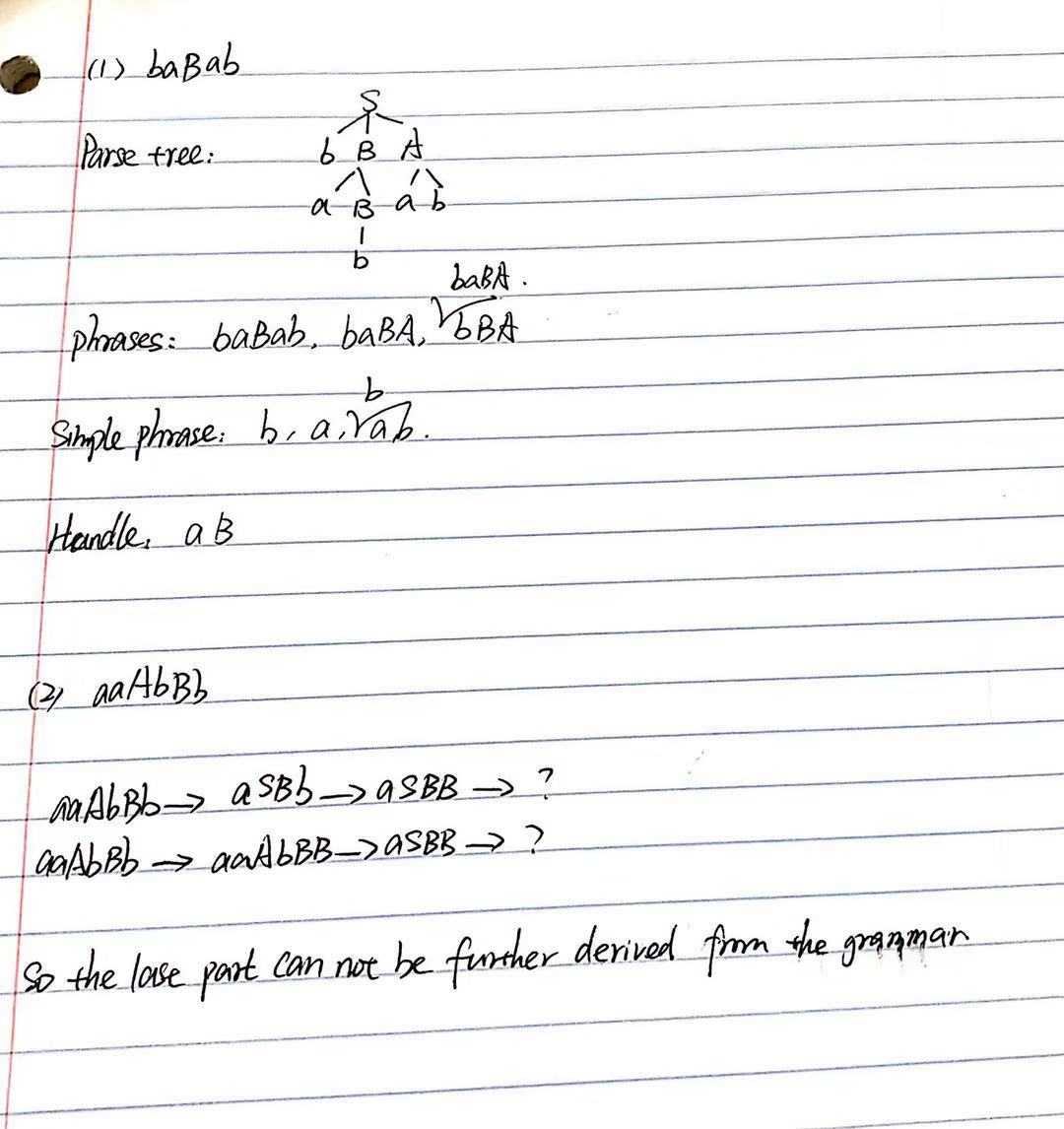
9. Given the following grammar and the right sentential form draw a parse tree and show the phrases, simple phrases, and the handle.

Grammar: S → aAb | bBA

A → ab | aAB

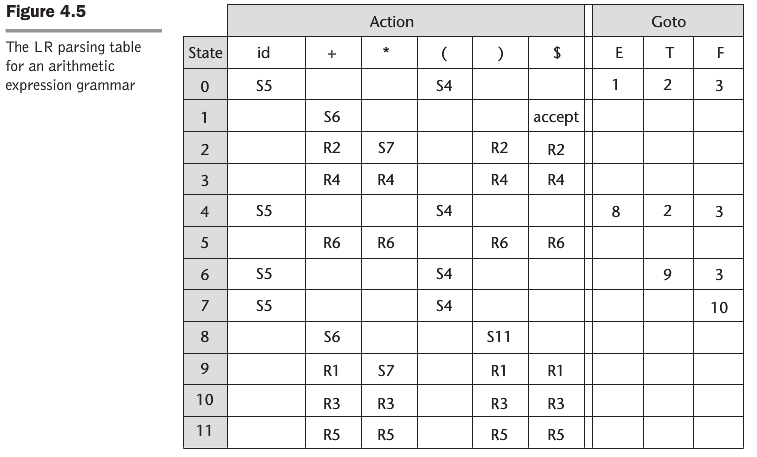
B → aB | b

1. baBab
2. aaAbBb



10. Show a complete parse, including the parse Stack contents, input string, and action for the string

id \* (id + id) , using the following grammar and parse table. Your results can look like the tables shown in the lecture slides.



Grammar: 1. E → E + T

2. E → T

3. T → T \* F

4. T → F

5. F → (E)

6. F → id

