TD4 - Tuesday, October 5

1 Syntax Analysis

1.1 Buttom Up Parsing : LR(0), SLR(1), LR(1), LALR(1)

Exercise 1

Consider the following grammar. Note that id, (, and) are terminals.

$$S \to TS|T$$

$$T \to (TF)|F$$

$$F \to (id)$$

Construct the SLR parsing table for the grammar.
Parse the string (((id)(id))(id))(id). Show the stack, the input, and the actions taken.

Exercise 2

Consider the following grammar :

$$P \to F$$

$$F \to (T$$

$$T \to);$$

$$T \to ;$$

$$T \to (F$$

Construct an LR(1) automaton and an SLR(1) automaton for the grammar.

Exercise 3 Consider the following grammar :

$$S \to Aa|bAc|Bc|bBa$$

$$A \to d$$

$$B \to d$$

- Construct the LR(1) automaton and the LR(1) parsing table.
- Explain why this grammar is LR(1) but not LARL(1).

Exercise 4

We want to built an interpreter for a language of musical scores. A score is formed by one or more notations of the following types :

- Indications of speed (tempo) : letter T followed by a number from 0 to 6, with the following interpretation : 0 : Largo ; 1 : Larguetto ; 2 : Adagio ; 3 : Andante ; 4 : Moderato ; 5 : Allegro ; 6 : Presto

– Indications of the octave : the letter O followed by an integer from 0 to 6
– Indications of pause : the letter P followed by an integer from 0 to 6
– Note specification : one of the letters from A to G, followed by :
  – an optional alteration : + (for Sharp) and - (for Flat)
  – an obligatory integer between 0 and 6 indicating the length of the note.
– The note is interpreted as follows : A : La ; B : Si ; C : Do ; D : Re ; E : Mi ; F : Fa ; G : Sol Each score must contain end with a note. Example of a musical score : T3O3C2E2G2B-2 plays the notes Do, Mi, Sol and Si flat of the 3rd octave, with duration of 2 and tempo andante.

The following grammar represents the possible scores :

SCORE → ELEMLIST
ELEMLIST → NOTE ELEMLIST| CNTRL ELEMLIST | NOTE
CNTRL → CLET NUM
NOTE → NLET ALTER NUM
ALTER → + | - | ε

Assume that during lexical analysis CLET, NUM and NLET are recognized as terminals :

$$CLET \rightarrow T|O|P$$

$$NUM \rightarrow 0|1|2|3|4|5|6$$

$$NLET \rightarrow A|B|C|D|E|F|G$$

Answer the following :
1. Construct the LR(1) automaton for the above grammar.
2. How would the LALR(1) automaton differ ?
3. How would the SLR(1) automaton
4. How would the LR(0) automaton differ ?
5. Construct the parse table for the LR(1) automaton.

1.2 Error recovery using the error symbol

Exercise 5 Add special grammar rules to the following grammar, for error recovery, such that the parser is allowed to resume :

$$E \rightarrow id$$

$$E \rightarrow \{E\}$$

$$E \rightarrow E; E$$

What are the synchronization tokens ?

What is the effect of a rule : $E \rightarrow error$ in look-ahead parsers (SLR and LALR) ?

1.3 Attribute Grammars

Exercise 6 Code generation for array references
When addressing array elements A, if an array element has width w, then the ith element of array A begins at

address `base + ( i - low ) * w` where `base` is the address of the first element of A and `low` is lower bound on subscript. We can rewrite the expression as `i * w + ( base - low * w )`. The first term depends on i (a program variable). The second term can be precomputed at compile time. This generalizes to n dimensions.

Give a simple attribute grammar (that may contain global attributes also) for handling working with array references. Examples :
expressions of type : `x:=A[m,n,p,q]` or `A[m,n,p,q]:=x` or `A[m,n,p,q]:=A[m,p,n,q]`
where A is an array of 4 dimensions in this case, x,m,n,p,q are variables.
You can use :
– a `lookup` function to search the symbol table to find corresponding id entries.
– a `newtemp` function that returns a new temporary variable name. The name consists of a leading character t and a number. For the i-th call, the number is i. In other words, the temporary variables generated by a sequence of call to newtemp function are : t1, t2, t3, ...