CS3470 Assignment 3 2018/19

This assignment should be submitted in PDF format via Moodle by **2pm** on **Friday 23rd November 2018**. Remember solutions must be your own work.

1.[**23 marks**] Using the method described in the course notes, construct the LR(0) NFA for the (augmented) grammar whose terminals are `* a b ) (`
The rules are numbered as shown.

```
0.  S' ::= S          1.  S ::= E b a       2.  S ::= ϵ
3.  E  ::= E * T      4.  E ::= T           5.  T ::= b
6.  T  ::= ( S )      7.  T ::= T b
```

Use the subset construction to construct the LR(0) DFA from your NFA and then write out the SLR(1) parse table for the grammar.

2.[**24 marks**] (i) Using the direct DFA construction method, construct the LR(0) DFA and the SLR(1) parse table for the following (already augmented) grammar, whose start symbol is `S'` and terminals are `a` and `b`.

```
0. S'::= S          1. S ::= A S        2. S ::= b
3. A ::= a          4. A ::= S a        5. A ::= ϵ
```

(ii) Use your table and the longest match strategy to parse the string `aab` showing the stack, remaining input and next action at each step in the process. (iii) Is the grammar ambiguous? Justify your answer.

3.[**30 marks**] Below are the first lines of a GLL parser for the grammar

```
 S::= d S b b  |  T d b          T ::= b | ϵ
```

whose start symbol is `S` and whose terminals are `b` and `d`.

```
    create GSS nodes $u_1 = (L_0, 0)$, $u_0 = (\$, 0)$ and an edge from $u_1$ to $u_0$
    $i := 0$; $s := u_1$; $\mathcal{U} := \{(L_S, u_1, 0)\}$; $\mathcal{R} := \emptyset$; $\mathcal{P} := \emptyset$
    goto $L_S$
$L_0$: if ($\mathcal{R} \neq \emptyset$) { remove $(L, s_1, j)$ from $\mathcal{R}$; $s := s_1$; $i := j$;  goto $L$ }
    else {  if ($(L_0, u_0, m) \in \mathcal{U}$) report success  else report failure }
```

(i) Using the templates discussed in lectures, complete this GLL parser. (ii) Use your parser to parse the string *db*, showing the GSS, the set $\mathcal{P}$, and all the descriptors created.

(You need to compute the SELECT sets for each alternate. You may assume the existence of the following functions, as described in lectures: *add*$((L, s, i), \mathcal{R})$, which adds a descriptor (L, s, i) to $\mathcal{R}$ if it has not already been added, *pop*$(s, i, \mathcal{R})$, which creates descriptors for each child of s, and *create*(L, s, i), which creates a GSS node labelled (L, i) with child s and applies any contingent pop actions.)

4.[**23 marks**]*[modified CS3470 Exam 2014]* (i) Use the GLL algorithm on the next page to parse the input string *ab*, showing the GSS, the set $\mathcal{P}$, and all the descriptors created. (ii) Write out the grammar for which the algorithm is a GLL parser.

```
        $u_0 := (\$, 0)$;  $u_1 := (L_o, 0)$;  $i := 0$;  $s := u_1$;  $\mathcal{U} := \{(L_S, u_1, 0)\}$;  $\mathcal{R} := \emptyset$
        goto $L_S$
$L_0$:  if $(\mathcal{R} \neq \emptyset)$ { remove $(L, s_1, j)$ from $\mathcal{R}$;  $s := s_1$;  $i := j$;   goto $L$ }
         else {  if $((L_0, u_0, m) \in \mathcal{U})$ report success  else report failure }

$L_S$:   if $(I[i] \in \{b\})$  $add((L_{S_1}, s, i), \mathcal{R})$
         if $(I[i] \in \{a, b\})$  $add((L_{S_2}, s, i), \mathcal{R})$
         if $(I[i] \in \{a, \$\})$  $add((L_{S_3}, s, i), \mathcal{R})$
         goto $L_0$
$L_{S_1}$:   $s := create(R_1, s, i)$;  goto $L_A$
 $R_1$:   $pop(s, i, \mathcal{R})$;   goto $L_0$
$L_{S_2}$:   $s := create(R_2, s, i)$;  goto $L_S$
  $R_2$:  if $(I[i] == a)$ { $i := i + 1$ }  else  goto $L_0$
          $s := create(R_3, s, i)$;  goto $L_A$
 $R_3$:   $pop(s, i, \mathcal{R})$;   goto $L_0$
$L_{S_3}$:  $pop(s, i, \mathcal{R})$;   goto $L_0$
$L_A$:   if $(I[i] \in \{b\})$  $add((L_{A_1}, s, i), \mathcal{R})$
         goto $L_0$
$L_{A_1}$:   if $(I[i] == b)$ { $i := i + 1$ }  else  goto $L_0$
          $pop(s, i, \mathcal{R})$;   goto $L_0$
```

PTO/..

Assessment Details And Criteria

This assignment is worth 5% of the final course mark. The questions on the assignment are not all of the same weight. The marks for each question are shown on the sheet.

For all questions, you should show your working where appropriate. Some marks will be given for 'correct working or approach' even when the final answer is not correct. You should use exactly the methods/algorithms specified in the questions. Marks will not be given for answers obtained by other means.

You should try all the questions and hand in what you have done, even if you are not able to complete some of the questions. Solutions to the assignment will be discussed in a lecture slot after the assignments have been marked.

Feedback arrangements

• Solutions must be submitted in PDF format via the links on the course Moodle page. The submissions must be PDFs not jpeg, Word documents or any other format. You may hand write and then scan your solutions to obtain a PDF file.
Write your name on every page and make sure the printed version is legible.
The mark will be set to 0 if the submission is in the wrong format or not legible.

• In the event of central IT issues at the time of the deadline, inform the course lecturer.

• The assignments will be marked and a feedback grade given. Note, the feedback grades are provisional and subject to change by our External Examiners during the examination moderation process.

• General feedback for this assignment will also be posted on the course webpage on Friday 7th December.

Learning outcomes assessed

The primary purpose of this assignment is formative. It is designed to

- provide a focus for reading and understanding the material on LR automata, and SLR(1) and GLL parsers,
- test your ability to construct and use SLR(1) parse tables
- test your ability to construct and use GLL parsers
- provide you with a way of assessing for yourselves the degree to which you have understood the material

If you do not know how to do a question, begin by reading the corresponding text in the notes and working through related examples discussed in lectures. Please feel free to make an appointment to talk to me about things you don't understand, and remember all solutions must be your own work.