CS4120/4121/5120/5121—Spring 2018
Homework 2
Syntactic Analysis
Due: Monday, February 12, 11:59PM

0 Updates

- 2/5: Due date extended to February 12.

1 Instructions

1.1 Partners

You may work alone or with *one* partner on this assignment. But remember that the course staff is happy to help with problems you run into. Use Piazza for questions, attend office hours, or set up meetings with any course staff member for help.

1.2 Homework structure

There are two parts of the homework. The first part is required of all students. The second part is required of students taking CS5120, but those enrolled in CS4120 are welcome to try it for good KARMA.

1.3 Tips

You may find the Dot and Graphviz packages helpful for drawing graphs. You can get these packages for multiple OSes from the Graphviz download page.

2 Problems

1. Context-free grammars

Consider a simple tag-based markup language. Terminal symbols are <, >, /, =, `name`, and `value`. A tag begins with < and ends with >. A tag may be an open or a close tag. In an open tag, the first token after < is `name` representing the tag's name, followed by an optional list of attributes which are `name` and `value` pairs joined together by =. In a close tag, the first token after < is a /, followed by `name`, but no attributes. Every open tag must be paired with a close tag. Any number of `name`s or tag pairs may appear between an open and close tags.

A sentence of this language is an arbitrary length sequence of matching tags. For example, here is a valid string in this markup language:

```
<name></name> <name name=value name=value>name name name <name></name></name>
```

(a) Write a context-free grammar for this language.

(b) Find the nullable predicate, and FIRST and FOLLOW sets for your grammar in part (a).

(c) Construct the LL(1) parse table and clearly identify any conflicts in the table. Make sure the columns of the table are in this order: <, >, /, =, `name`, `value`, and `$`.

(d) Fundamentally, what makes this *language* (not the grammar you wrote down) not LL(1)? Is there a k for which this language is LL(k)?

2. Ambiguous grammars

Consider the following grammar:

$$\begin{aligned} E &\rightarrow E\ Op\ E \mid \texttt{(}E\texttt{)} \mid \texttt{<num>} \\ Op &\rightarrow \texttt{+} \mid \texttt{*} \mid \texttt{\^{}} \end{aligned}$$

This grammar is ambiguous. We would like a grammar to derive parse trees in which exponentiation (^) has higher precedence than multiplication (*), and both have higher precedence than addition (+). Exponentiation should be right-associative, while multiplication and addition should be left-associative.

(a) Give an example to show that the grammar is ambiguous.

(b) Write an LL(1) grammar that accepts the same string as this grammar and respects the desired operator precedence. Associativity need not be enforced, however.

(c) Show the derivation of the expression `2^3*4+5` using your grammar in part (b).

(d) Write an LR(1) grammar that accepts the same language, but enforces both the precedence and associativity of the operators. You need not show the parsing tables.

3. LR grammars

Show that the following grammar is LR(1) but not LALR(1):

$$\begin{aligned} S &\rightarrow A\texttt{a} \mid \texttt{b}A\texttt{c} \mid B\texttt{c} \mid \texttt{b}B\texttt{a} \\ A &\rightarrow \texttt{d} \\ B &\rightarrow \texttt{d} \end{aligned}$$

Hint: construct the LR(1) parser state diagram, and look at what states the LALR(1) parser state machine would merge. To get you started, we provided a partial state diagram for the LR(1) parser in Figure 1.

3 Problem for CS5120

4. Predictive parsing

Give an unambiguous grammar that is not LL(1), but for which every nonterminal has a unique applicable production given the first token, i.e., for each nonterminal, the FIRST sets of its productions are disjoint.

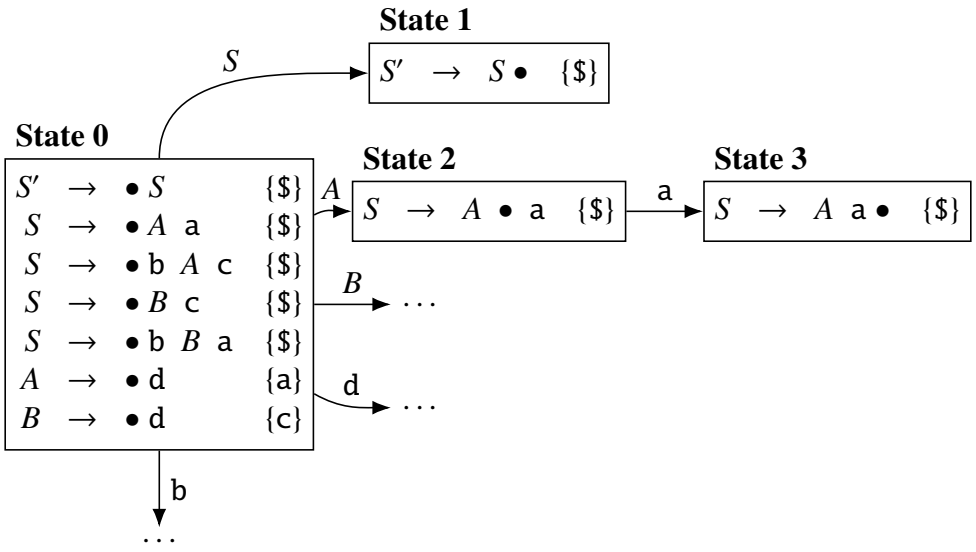

Figure 1: Partial LR(1) parser state diagram for Problem 3

4 Submission

Submit your solution as a PDF file on CMS. This file should contain your name, your NetID, all known issues you have with your solution, and the names of anyone you have discussed the homework with.