

Summer 2008 Midterm

Your Name: _____

SUNet ID: _____

Instructions:

- **Honor.** You must complete this exam alone and in the time specified. You must follow the Stanford Honor Code.
- **Time Limit.** You have from 7:00pmPDT to 8:30pmPDT to complete this exam.
- **Show and explain your work.** Do this whenever possible. It allows us to better characterize your understanding of the material - and better grade your hard work.
- **Notes.** You may refer to any books, notes, or other printed material which you bring to the exam. You must *not* use any electronic materials or equipment.
- **DFA transitions.** In the interest of time, you may omit transitions for any DFA; such missing transitions will be assumed to go to an implicit reject state.
- **Submission for Remote Students.** Submit your work electronically via e-mail to dgu@stanford.edu. You may alternatively submit it via fax. You must submit your work immediately after the 8:30pmPDT deadline.
- **Questions During the Exam.** I will be outside the exam room in case you have a question. Remote students may call 304-541-2399 to ask a question.

| Problem # | Points Earned | Points Possible |
|--------------|---------------|-----------------|
| 1 | | 6 |
| 2 | | 16 |
| 3 | | 20 |
| 4 | | 12 |
| 5 | | 6 |
| 6 | | 15 |
| Total | | 75 |

Problem 1 File Extensions Considered Harmful

Background. In UNIX, the `sed` utility can be used for transforming text which matches a regular expression. Specifically, `sed` can replace an all occurrences of a regular expression `RE` with some new text `TXT`. This is the format of the `sed` command:

```
sed -e 's#RE#TXT#g' INPUT_FILE
```

For example, to use `sed` to get rid of the ugly carriage returns (`\r`) from a DOS file, you could do the following to replace them with the empty string:

```
sed -e 's#\r##g' INPUT_FILE
```

Problem. Assume you have an input file which contains a large list of file names with one file name per line. **Write a compact regular expression to match file extensions** which you could use as the `RE` in the above command to remove all file extensions. *Use `lex` regular expression syntax* to specify your expression. Make the following assumptions:

- A file extension is a period followed by zero or more symbols except periods and forward slashes.
- A file extension only occurs at the end of a string.

Example Input

```
p1.tex  
/home/dgu/ChessMeister.cpp  
/usr/class/PP-1.tar.gz  
/dev/nf2  
/dev/null/pp3/.gitignore
```

Example Output

```
p1  
/home/dgu/ChessMeister  
/usr/class/PP-1.tar  
/dev/nf2  
/dev/null/pp3/
```

Hint: You are only writing a regular expression to match the *file extension* - it should not match any other part of the file name.

Problem 2 Nothing Ever Changes

- a. Remove any λ -cycles from the N DFA shown in Figure 1.

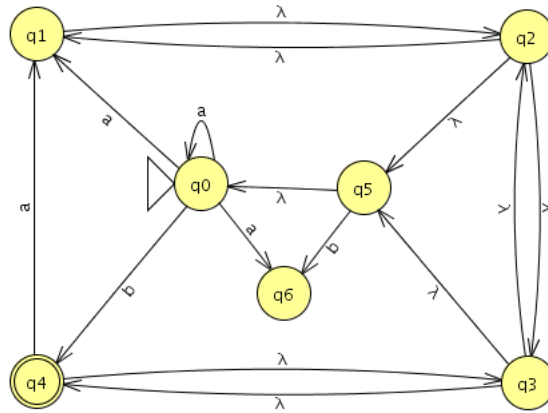


Figure 1: The NFA to remove λ cycles and transitions from.

-
- b. With the NFA resulting from the previous step, remove any remaining λ transitions. Remove one λ transition at a time, and show *each* resulting NFA.

- c. Algorithmically remove any non-determinism from the NFA shown in Figure 2. Show your work with a table, and do not try to simplify the NFA. You do *not* need to draw the resulting DFA.

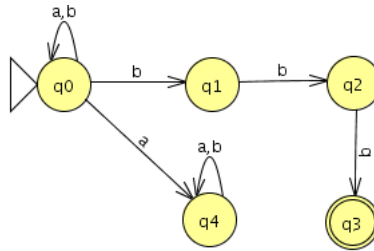


Figure 2: The NFA to remove non-determinism from in order to create an equivalent DFA.

- d. Algorithmically minimize the DFA shown in Figure 3. Show your work with table(s). Draw the resulting DFA.

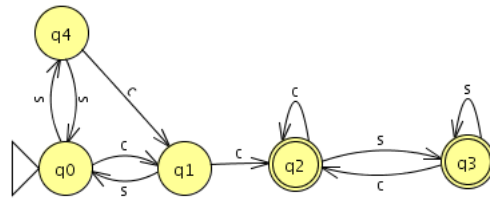


Figure 3: The DFA to minimize.

| | wood | could | chuck | a | howmuch | if | \$ |
|---------|------|-------|-------|---|---------|----|----|
| S | | | | | | | |
| OptCond | | | | | | | |
| Stmnt | | | | | | | |
| V | | | | | | | |
| OptObj | | | | | | | |
| N | | | | | | | |
| Q | | | | | | | |

Problem 4 Broken Calculator

Assume each production in the following grammar is associated with the correct semantic action that computes its value in bison. *Please write your answers in the right margin.*

$$E \rightarrow E O E \mid T_IntConst$$

$$O \rightarrow '+' \mid '-' \mid '*' \mid '/'$$

- a. If no precedence is specified, what value will Bison compute for the string $5*3+2$? Draw the parse tree.

- b. Draw the parse tree and specify the value it computes for the string $5*3+2$ if we provide only the following precedence directive:

```
%left * /
```

- c. Draw the parse tree and specify the value it computes for the string $5*3+2$ if we provide only the following precedence directive:

```
%left * /
%left + -
```

- d. If we specify that division is left-associative, what is the value of $9/3/3$? Draw the parse tree.

- e. If we specify that division is right-associative, what is the value of $9/3/3$? Draw the parse tree.

- f. Draw the parse tree and specify the value it computes for $9/3/3$ if we give the following precedence directive:

```
%nonassoc /
```

Problem 5 The Infamous Alderson Loop

Rewrite the following grammar to be LL(1):

$$\mathbf{A} \rightarrow \mathbf{A}bb \mid \mathbf{A}bc \mid cc$$

Problem 6 Simple Lists

Draw the LR(0) configurating sets for the following grammar. If there are any problems in the grammar, identify where they occur when constructing the configurating sets.

$$\mathbf{NT} = \{ \mathbf{D}, \mathbf{P}, \mathbf{S}, \mathbf{T} \}$$

$$\mathbf{T} = \{ (,), \text{id}, \text{int}, \text{void} \}$$

$$\mathbf{S} \rightarrow (\mathbf{P})$$

$$\mathbf{P} \rightarrow \mathbf{D} \mathbf{P}$$

$$\mathbf{P} \rightarrow \mathbf{D}$$

$$\mathbf{D} \rightarrow \mathbf{T} \text{ id}$$

$$\mathbf{T} \rightarrow \text{void}$$

$$\mathbf{T} \rightarrow \text{int}$$

This is extra space in case you need it.