VICTORIA UNIVERSITY OF WELLINGTON

Te Whare Wananga o te Upoko o te Ika a Maui

EXAMINATIONS — 2003

END-YEAR

COMP 202

Formal Methods of Computer Science

Time Allowed: 3 Hours

Instructions: There are seven (7) questions, worth fourteen (14) marks each, making ninety-eight (98) marks in total.
Answer all the questions.

You may use printed foreign language dictionaries.
You may not use calculators or electronic dictionaries.

The answers are in boxes.

Question 1.

The following program takes as input two strings, s and t, which are assumed to be the same length (i.e., $|s| = |t|$). Strings are indexed starting from 0. It is possible to determine from the final value of i whether s and t are in fact the same string.

```
begin
  i := |s| − 1;
  while i ≥ 0 and s[i] = t[i] do
    i := i − 1
  od
end
```

(a) Give a specification (signature, precondition, and postcondition) for the problem that this program satisfies. [4 marks]

(b) Explain how a **loop invariant** may be used in the verification of a while-program. [5 marks]

(c) State a loop invariant that may be used to verify the above program. **You do not need to complete the proof.** [5 marks]

Question 2.

$M_1 = (Q, \Sigma, \delta, q_0, F)$, where:

- $Q = \{S_0, S_1, S_2, S_3\}$
- $\Sigma = \{a, b\}$
- $q_0 = S_0$
- $F = \{S_3\}$

and δ is given by the table:

δ	a	b
S_0	$\{S_1, S_2\}$	$\{\}$
S_1	$\{S_1\}$	$\{S_0, S_3\}$
S_2	$\{S_2\}$	$\{S_0\}$
S_3	$\{S_3\}$	$\{S_1\}$

(a) Draw M_1. [4 marks]

(b) Find an FA which accepts the same language as M_1. [10 marks]

We need to construct an FA M_2 whose states are sets of states of M_1. The alphabet of M_2 is just that of M_1. The start state of M_2 is the singleton set containing the start state of M_1. The transition function for M_2 on some symbols σ takes us from a set of states of M_1 to the set of states of M_1 reachable on σ. The set of states of M_2 is the set of states reachable on sequences of symbols from the start state of M_2. The accepting states of M_2 are states of M_2 containing an accepting state of M_1.
$M_2 = (Q', \Sigma', \delta', q_0', F')$, where:

- $Q' = \{\{S_0\}, \{S_1, S_2\}, \{\}, \{S_0, S_3\}, \{S_1, S_2, S_3\}, \{S_1\}, \{S_0, S_1, S_3\}\}$
- $\Sigma' = \Sigma$
- $q_0' = \{S_0\}$
- $F' = \{\{S_0, S_3\}, \{S_1, S_2, S_3\}, \{S_0, S_1, S_3\}\}$

and δ' is given by the table:

δ'	a	b
$\{S_0\}$	$\{S_1, S_2\}$	$\{\}$
$\{S_1, S_2\}$	$\{S_1, S_2\}$	$\{S_0, S_3\}$
$\{\}$	$\{\}$	$\{\}$
$\{S_0, S_3\}$	$\{S_1, S_2, S_3\}$	$\{S_1\}$
$\{S_1, S_2, S_3\}$	$\{S_1, S_2, S_3\}$	$\{S_0, S_1, S_3\}$
$\{S_1\}$	$\{S_1\}$	$\{S_0, S_3\}$
$\{S_0, S_1, S_3\}$	$\{S_1, S_2, S_3\}$	$\{S_0, S_1, S_3\}$

The smallest FA (with a total transition function) which accepts the same language as M_1 is:
$M_3 = (Q'', \Sigma', \delta'', q_0'', F'')$, where:

- $Q'' = \{T_1, T_2, T_3, T_4, T_5\}$
- $\Sigma'' = \Sigma$
- $q_0'' = \{T_1\}$
- $F'' = \{T_4, T_5\}$

and δ'' is given by the table:

δ'	a	b
T_1	T_2	T_3
T_2	T_2	T_4
T_3	T_3	T_3
T_4	T_5	T_2
T_5	T_5	T_5

You were not expected to produce the smallest FA, of course.

Question 3.

(a) Let $\mathbf{r}$ and $\mathbf{s}$ be regular expressions, and let $L_{\mathbf{r}} = \text{Language}(\mathbf{r})$ and $L_{\mathbf{s}} = \text{Language}(\mathbf{s})$.

Give regular expressions which describe the following languages:

1. $L_{\mathbf{r}} \cup L_{\mathbf{s}}$ $\boxed{\mathbf{r} + \mathbf{s}}$
2. $L_{\mathbf{r}}^*$ $\boxed{\mathbf{r}^*}$
3. $L_{\mathbf{r}} \cap \overline{L_{\mathbf{r}}}$ $\boxed{\text{This is the empty language: } \emptyset}$ [3 marks]

(b) Let: $\Sigma = \{a, b\}$
$L_1 = \text{Language}(\mathbf{a}(\mathbf{a} + \mathbf{b})^*)$
$L_2 = \text{Language}((\mathbf{ba})^*)$
$L_3 = \text{Language}(\mathbf{a}^* + \mathbf{b}^*)$

L_1 is every string starting with an **a** i.e. $\{\mathbf{a}, \mathbf{aa}, \mathbf{ab}, \mathbf{aaa}, \mathbf{aab}, \mathbf{aba}, \mathbf{abb}, \ldots\}$.
L_2 is sequences of **ba**s i.e. $\{\Lambda, \mathbf{ba}, \mathbf{baba}, \mathbf{bababa}, \mathbf{babababa}, \ldots\}$
L_2 is sequences of **b**s and sequences of **a**s i.e. $\{\Lambda, \mathbf{a}, \mathbf{b}, \mathbf{aa}, \mathbf{bb}, \mathbf{aaa}, \mathbf{bbb}, \ldots\}$

Give a string which:

1. is in L_1 but not in L_2 or L_3 $\boxed{\mathbf{ab}}$
2. is in L_2 but not in L_1 or L_3 $\boxed{\mathbf{ba}}$
3. is in L_3 but not in L_1 or L_2 $\boxed{\mathbf{bb}}$ [3 marks]

(c) State Kleene's theorem. [4 marks]

Theorem 1 (Kleene)
A language is regular iff it is accepted by a FA.
A language is regular iff it is accepted by an NFA.
A language is regular iff it is generated by a regular grammar.

(d) *Either* give an example of a finite language which is not regular *or* show that every finite language is regular. [4 marks]

Every finite language is regular. We prove this by induction on the size of the language. A finite language is a finite set of words. The base case is that we must show that the empty language is regular. The induction step allows us to assume that the language with some word removed from it is a regular language, and asks us show that the language is regular.
Recall that a language is regular iff it can be described by a regular expression (Kleene). Base case: trivial the empty language is described by the regular expression $\emptyset$.
Induction step. Let L be a regular language, and w be a word (not occurring in L). As L is regular there is a regular expression $\mathbf{e}$ such that L is $\mathrm{Language}(\mathbf{e})$. If we can find a regular expression $\mathbf{q}$ such that $\{w\}$ is $\mathrm{Language}(\mathbf{q})$, then $L \cup \{w\}$ will be $\mathrm{Language}(\mathbf{e} + \mathbf{q})$. Hence $L \cup \{w\}$ will be regular. Hence the induction step will be proved. Hence we will have shown every finite language is regular.
We show that every singleton language of one string can be described by a regular expression by induction on the length of the string. The base case is where the string is of length 0, and the induction step is that we show the property holds for strings of length $n+1$ if it holds for strings of length n.
Base case $\{\Lambda\}$ is $\mathrm{Language}(\Lambda)$
Induction step we show that if σ is a symbol from some alphabet Σ and if $\{\tau\}$ is regular then so is $\{\sigma\tau\}$. If $\{\tau\}$ is regular then there is a regular expression $\mathbf{t}$ such that $\{\tau\}$ is $\mathrm{Language}(\mathbf{t})$. So $\{\sigma\tau\}$ is $\mathrm{Language}(\sigma\mathbf{t})$. So the induction step is proved. So every language of just one word is regular, and we have completed the proof that every finite language is regular.
You were not expected to produce this proof in the exam. An informal argument that if L is $\{w_1, \ldots, w_n\}$, then it L is $\mathrm{Language}(\mathbf{w_1} + \ldots + \mathbf{w_n})$ was sufficient.

Question 4.

Consider the context free grammar

$$\begin{array}{ll} (1,2) & S \to aS \mid T \\ (3,4,5) & T \to aSbS \mid U \mid \Lambda \\ (6) & U \to b \end{array}$$

(a) List the **nullable nonterminals**. [1 mark]

(b) List the **unit productions**. [1 mark]

(c) Find an equivalent grammar with no unit productions. [4 marks]

(d) Give two different **leftmost derivations** of the string $aabb$. [2 marks]

(e) Find an equivalent **unambiguous grammar**. [4 marks]

(f) Using your grammar from part **(e)**, draw a parse tree for the string $aabb$. [2 marks]

Question 5.

The following is a context-free grammar for a fragment of **HTML**: the markup language used for web documents. Nonterminal symbols are written in *Italics*; terminal symbols are enclosed in "quotation marks".

$$
\begin{array}{rll}
(1,2) & \mathit{Doc} & \rightarrow \mathit{Element}\ \mathit{Doc} \mid \mathit{Element} \\
(3) & \mathit{Element} & \rightarrow \text{“<OL>”}\ \mathit{List}\ \text{“</OL>”} \\
(4) & \mathit{Element} & \rightarrow \text{“<UL>”}\ \mathit{List}\ \text{“</UL>”} \\
(5,6,7) & \mathit{Element} & \rightarrow \text{“a”} \mid \text{“b”} \mid \text{“c”} \\
(8,9) & \mathit{List} & \rightarrow \text{“<LI>”}\ \mathit{Element}\ \mathit{List} \mid \Lambda
\end{array}
$$

Thus, the nonterminals of the grammar are $\{\mathit{Doc}, \mathit{Element}, \mathit{List}\}$, and the terminals are {“<OL>”, “</OL>”, “<UL>”, “</UL>”, “<LI>”, “a”, “b”, “c”}.

(a) Explain what it means for a grammar to be in LL(1) form. [3 marks]

(b) Show that the above grammar is **not** in LL(1) form. [2 marks]

(c) Rewrite the grammar so that it is in LL(1) form. [4 marks]

(d) Complete the $\mathit{ParseList}$ procedure whose heading appears below, for a recursive-descent parser to recognize the List nonterminal. You may assume that procedures $\mathit{ParseDoc}$ and $\mathit{ParseElement}$ for recognizing the other nonterminals are already written. The parameter ss, providing the input for the parser, is a sequence of terminal symbols. You do not need to return a parse tree.

procedure $\mathit{ParseList}$ (**in out** ss)
begin
  ...
end

[5 marks]

Question 6.

(a) Define a pushdown automaton that accepts each of the following languages:

(i) The language described by the regular expression $\mathbf{a}^*\mathbf{b}^+$ [2 marks]

(ii) The language of even-length palindromes, $\{ww^R \mid w \in \{a,b\}^*\}$ [2 marks]

(iii) The language consisting of strings over the alphabet $\{a,b\}$ with the same number of as and bs, but occurring in any order [3 marks]

(iv) The language $\{a^m b^n c^{m+n} \mid m,n \geq 1\}$ [3 marks]

(b) Define a pushdown automaton that accepts the language generated by the following grammar, using a **bottom-up** (shift-reduce) strategy.

$$\begin{aligned} S &\to TU \mid \Lambda \\ T &\to aS \mid Sb \\ U &\to aU \mid a \end{aligned}$$

[4 marks]

Question 7.

(a) Let L_1 and L_2 be regular languages. State whether it is possible to write a program which decides whether L_1 and L_2 are the same language. Justify your answer.

[4 marks]

Yes. If two languages are equal then the set of strings in one but not the other is empty. In other words $(L_1 \cap \overline{L_2}) \cup (L_2 \cap \overline{L_1}) = \emptyset$ If L_1 and L_2 are regular then so is $(L_1 \cap \overline{L_2}) \cup (L_2 \cap \overline{L_1})$. It is decidable whether a regular language is empty.

(b) Give an example of a language which is:

(i) context-free but not regular $\{a^n b^n | n \geq 0\}$ [1 mark]

(ii) computable but not context-free $\{ww | w \in \{a,b\}^*\}$ [2 marks]

(iii) computably enumerable but not computable

$A_{\mathrm{TM}} = \{(T,w) | w \in \mathrm{accept}(T)\}$ (see lecture notes) [3 marks]

(iv) not computably enumerable $\overline{A_{\mathrm{TM}}}$ (see lecture notes) [4 marks]

****