TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI
VICTORIA
UNIVERSITY OF WELLINGTON

EXAMINATIONS — 2006

END-YEAR

COMP 202
Formal Methods of Computer Science

Time Allowed: 3 Hours

Instructions: There are six (6) questions, worth thirty (30) marks each, making one hundred and eighty (180) marks in total.

Answer all the questions.

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

Question 1.

(a) Let L_1 be the language accepted by the nondeterministic finite automaton with Λ transitions (NFAΛ) defined by

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

where:

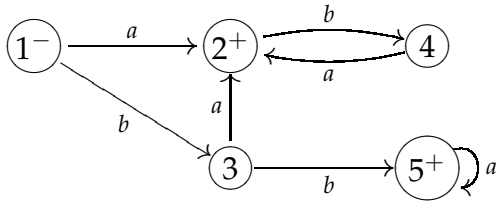
- $Q = \{S_1, S_2, S_3, S_4\}$
- $\Sigma = \{a, b\}$
- $q_0 = S_1$
- $F = \{S_4\}$

δ	a	b	Λ
S_1	$\{S_2, S_3\}$	$\{\}$	$\{\}$
S_2	$\{\}$	$\{S_2\}$	$\{S_4\}$
S_3	$\{S_3\}$	$\{S_4\}$	$\{\}$
S_4	$\{S_2, S_4\}$	$\{\}$	$\{S_3\}$

(i) Draw M_1.

(ii) Find a determinstic finite automaton (FA) that accepts L_1. [10 marks]

(b) Let L_2 be the language accepted by the FA M_2 with pictorial representation



(i) State Q, Σ, δ, q_0, and F for M_2.

(ii) Give an iterative scanner that recognizes L_2.

(iii) Give a regular grammar that generates L_2. [15 marks]

(c) Let L_3 be the language described by the regular expression $a^*((a+b)(a+b))^*b$.

Give a NFAΛ that accepts L_3. [5 marks]

Question 2.

(a) State Kleene's Theorem. [6 marks]

(b) Explain how to construct a recursive scanner that recognizes the language accepted by a given regular grammar. [6 marks]

(c) Explain how to construct a regular expression that describes the language accepted by a given regular grammar. [6 marks]

(d) Describe a language that is *not* regular, and explain *informally* why it cannot be regular. [6 marks]

(e) Define *ambiguity* for a context free grammar, and describe one technique for avoiding ambiguity. [6 marks]

Question 3.

(a) Describe two techniques that may sometimes be used to find an LL(1) grammar that generates the same language as a given context free grammar (CFG). [6 marks]

(b) Explain how to construct a recursive descent parser for a given LL(1) grammar. [6 marks]

(c) Consider the CFG defined by

$$G = (\Sigma, N, S, P)$$

with productions

$$\begin{array}{ll} (1,2,3) & A \to aBc \mid ac \mid b \\ (4,5) & B \to bAA \mid b \end{array}$$

(i) State Σ, N, S, and P for G.

(ii) Give a leftmost derivation in G for the string *abacbc*.

(iii) Give a derivation in G that is *not* leftmost for the string *abacbc*.

(iv) Draw a parse tree for the string *abacbc*.

(v) Show that G is *not* LL(1).

(vi) Find an LL(1) grammar that generates the same language as G, and show that your grammar is LL(1). [18 marks]

Question 4.

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

(i) The language $\{a^n b^{2n} \mid n > 0\}$.

(ii) The language consisting of strings over the alphabet $\{a, b\}$ with the same number of as and bs, but occurring in any order. [8 marks]

(b) Consider the CFG $G = (N, \Sigma, S, P)$ with productions:

$$S \to STc \mid T$$
$$T \to aT \mid U$$
$$U \to aS \mid \Lambda$$

(i) Define a pushdown automaton that accepts the language generated by G, using a **top-down** (expand-match) strategy.

(ii) Define a pushdown automaton that accepts the language generated by G, using a **bottom-up** (shift-reduce) strategy. [10 marks]

(c) "For every pushdown automaton, there exists a deterministic pushdown automaton that accepts the same language".

(i) State whether the above statement is true or false.

(ii) IF the statement is true, outline a proof.
IF the statement is false, give a counterexample. [6 marks]

(d) Briefly outline a proof that any language accepted by a pushdown automaton may be generated by a context free grammar. [6 marks]

Question 5.

(a) A Turing machine may be defined by a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, F)$.

Explain each of the following:

(i) Σ

(ii) Γ

(iii) δ [6 marks]

(b) Let T be a Turing machine with input alphabet Σ.

(i) Define $accept(T)$.

(ii) Define $reject(T)$.

(iii) Define $loop(T)$.

(iv) Define "computably enumerable language".

(v) Explain the relevance of the existence of universal Turing machines in proving that there are undecidable problems. [12 marks]

(Question 5 continued on next page)

(Question 5 continued)

(c) A **two-player game** is a game in which two players (A and B) alternate making moves according to some predetermined set of rules. Thus, player A makes the first and all subsequent odd-numbered moves; player B makes the second and all subsequent even-numbered moves.

A **finite** two-player game is one which we can guarantee will terminate after a finite number of moves, either because one player has won, or because no move is available for the player whose turn it is to move. For example, noughts-and-crosses (tic-tac-toe) is a finite game, because we can be sure that after a finite number of moves (the number of squares on the board) the game must terminate.

An **infinite** two-player game is a two player game that is not finite (that is, one in which there is at least one play sequence that will never terminate). For example, chess played *without* the championship rule that forbids endless sequences of repeated moves is an infinite game, because the players can continue indefinitely without declaring a draw.

Clearly, every two-player game is either finite or infinite.

The rules of **hypergame** are as follows:

- There are two players, A and B.
- The first move is for A to define the rules of any finite game G.
- The second move is for B to play the first move of G.
- The third move is for A to make the second move of G.
- ...
- Hypergame terminates (at its $(n+1)$th move) when G terminates (at its nth move).

(i) Is hypergame a two-player game, according to the definition above?

(ii) Is hypergame a finite game, according to the definition above?

(iii) Is hypergame an infinite game, according to the definition above?

(iv) Explain the relevance of your answers to computability theory. [12 marks]

Question 6.

(a) The following program takes two integers, a and b, and returns integers x and y such that x is the smaller of a and b, and y is the larger of a and b.

```
        begin
{A1}      if a < b then
{A2}        x := a;
{A3}        y := b
{A4}      else
{A5}        x := b;
{A6}        y := a;
{A7}        skip
{A8}    end
```

(i) Write a formal specification (signature, precondition, and postcondition) for this problem.

(ii) Give assertions $A_1, \ldots, A_8$, and use them to prove this program correct. [18 marks]

(b) The following is (part of) a definition of a semantic function $\mathcal{V}$ for evaluating Boolean expressions:

$$\mathcal{V}(p \textbf{ and } q) = \begin{cases} \mathcal{V}(q), & \text{if } \mathcal{V}(p) = 1 \\ 0, & \text{otherwise} \end{cases}$$

$$\mathcal{V}(\textbf{not } p) = \begin{cases} 1, & \text{if } \mathcal{V}(p) = 0 \\ 0, & \text{otherwise} \end{cases}$$

$$\mathcal{V}(\textbf{true}) = 1$$

$$\mathcal{V}(\textbf{false}) = 0$$

Extend the definition to handle **Boolean variables**, v, whose values are determined by a store. [12 marks]

****