VICTORIA UNIVERSITY OF WELLINGTON

Te Whare Wananga o te Upoko o te Ika a Maui

EXAMINATIONS — 2002

TERM TEST

COMP 202

Formal Methods of Computer Science
WITH ANSWERS

Time Allowed: 50 minutes

Instructions: There are **four** (4) questions.
Each question is worth **twenty-five** (25) marks.
Answer **all** the questions.
Show **all** your working.

Question 1.

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}      fi
{A8}    end
```

(a) Write a formal specification (signature, precondition, and postcondition) for this problem. [6 marks]

Input: integers a and b
Output: integers x and y
Precondition: $true$
Postcondition: $\{x, y\} = \{a, b\} \wedge x \leq y$

(b) Give assertions $A_1, \ldots, A_8$ that may be used to prove this program correct. **You do not need to complete the proof!** [12 marks]

A_1	$true$	A_2	$a \leq b$
A_3	$a \leq b \wedge x = a$	A_4	$a \leq b \wedge x = a \wedge y = b$
A_5	$a \not\leq b$	A_6	$a \not\leq b \wedge x = b$
A_7	$a \not\leq b \wedge x = b \wedge y = a$	A_8	$A_4 \vee A_7$

(c) Explain how a **loop invariant** may be used in the verification of a while-program. [7 marks]

Part 1 notes, Sec 5.2.3

Question 2.

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

$$\begin{aligned} Q &= \{S_0, S_1, S_2, S_3\} \\ \Sigma &= \{0, 1\} \\ q_0 &= S_0 \\ F &= \{S_1\} \end{aligned}$$

δ	0	1	Λ
S_0	$\{\}$	$\{S_2\}$	$\{S_1\}$
S_1	$\{S_1\}$	$\{S_3\}$	$\{\}$
S_2	$\{\}$	$\{S_2\}$	$\{S_3\}$
S_3	$\{S_1\}$	$\{\}$	$\{\}$

(a) For each of the following strings state whether it is accepted by M_1:

1. 00 $\boxed{\text{accepted}}$
2. 10 $\boxed{\text{accepted}}$
3. 01 $\boxed{\text{not accepted}}$
4. 011 $\boxed{\text{not accepted}}$
5. 11011 $\boxed{\text{not accepted}}$

[5 marks]

(b) *Outline* the method for constructing an NFA from an NFA with Λ transitions, such that both machines accept the same language. [10 marks]

Given NFA with Λ transitions $M_4 = (Q, \Sigma, \delta, q_0, F)$ we create an NFA $M_5 = (Q', \Sigma', \delta', q_0', F')$.
We keep the same states and the same alphabet, so $Q' = Q$ and $\Sigma = \Sigma'$. We construct a new transition function, so that if before we could get from two states on a combination of Λ moves and a single σ move we can now get between the two states on a single σ move.
All the accepting states of M_4 are accepting states of M_5. If we can get from the **start** state to a final state on only Λ moves then M_4 accepts Λ, and so q_0 is also an accepting state of M_5
So, M_5 is an NFA, and for any path from start state to an accepting state of M_4 labelled by the symbols in a word w (in sequence of course) and possibly Λ, there is a path from start state to an accepting state of M_5 labelled by the symbols in a word w (in sequence of course). Hence M_5 accepts the same language as M_4.

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

$M_2 = (Q', \Sigma', \delta', q_0', F')$ where:

- $Q' = Q$
- $\Sigma' = \Sigma$
- $F' = F \cup \{q_0\}$ since M_1 accepts Λ
- $q_0' = q_0$

and

δ'	0	1
S_0	$\{S_1\}$	$\{S_2, S_1\}$
S_1	$\{S_1\}$	$\{S_3\}$
S_2	$\{S_1\}$	$\{S_2, S_3\}$
S_3	$\{S_1\}$	$\{\}$

Most of the mistakes made in this question involved forgetting to include q_0 in F', and getting $\delta'(S_2, 1)$ wrong.

Question 3.

The following machines M_2 and M_3 accept $\text{Language}(\mathbf{1})$ and $\text{Language}(\mathbf{0})$ respectively.

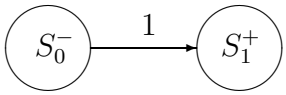


Figure 1: M_2

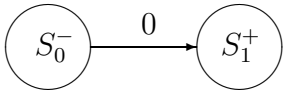
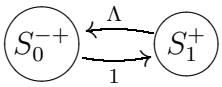


Figure 2: M_3

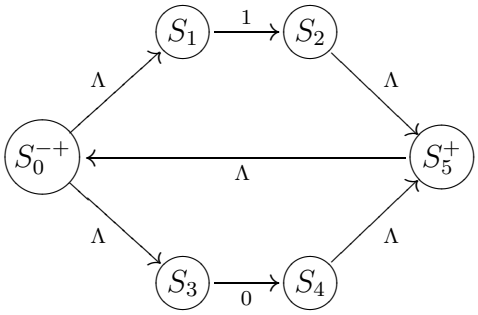
Draw NFA with Λ transitions which accept the following languages.

The following are possible (correct) solutions; there are numerous others.

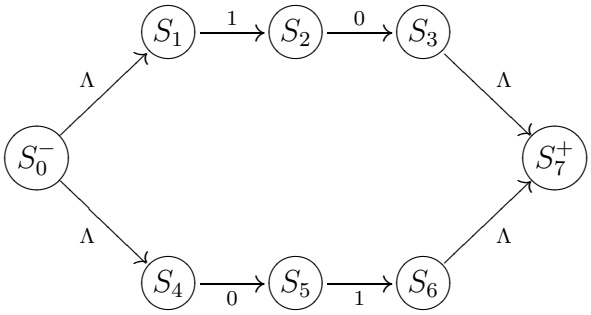
(a) $\text{Language}(\mathbf{1}^*)$ [5 marks]



(b) $\text{Language}((\mathbf{1}+\mathbf{0})^*)$ [5 marks]



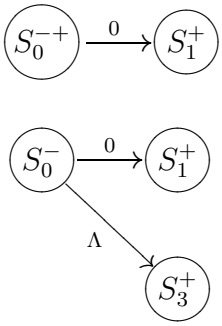
(c) Language($\mathbf{10} + \mathbf{01}$) [5 marks]



(d) Language($\mathbf{1} + \emptyset$) [5 marks]

Language($\mathbf{1} + \emptyset$) = Language($\mathbf{1}$), which is shown in Fig. 1

(e) Language($\mathbf{0} + \mathbf{\Lambda}$) [5 marks]



Question 4.

(a) *State* Kleene's Theorem. [5 marks]

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.

(b) The pumping lemma tells us that if L is a regular language then there is a number p, such that $(\forall s \in L)(|s| \geq p \Rightarrow s = xyz)$, where:

1. $(\forall i \geq 0)xy^iz \in L$
2. $|y| > 0$
3. $|xy| \leq p$

Outline the proof of the pumping lemma. [10 marks]

The explanation for lectures appears as the next box. The key points that I was looking for were really to say that if the language is regular then there is an FA which accepts it, to mention the pigeon-hole principle, and hence, for sufficiently long strings in the language they can be split and pumped as the PL requires.
Generally this question was not answered well.

We have three things to prove corresponding to conditions 1, 2 and 3 in the pumping lemma. Let:

- $M = (Q, \Sigma, \delta, q_0, F)$ be a FA which accepts L,
- p be the number of states in M (i.e. $p = 2^Q$)
- $s = s_1 s_2 \ldots s_{n-1} s_n$ be a string in L such that $n \geq p$
- $r_1 = q_0$
- $r_{i+1} = \delta(r_i, s_i)$, $1 \leq i \leq n$

Then the sequence $r_1 r_2 \ldots r_n r_{n+1}$ is the sequence of states that the machine goes through to accept s. The last state r_{n+1} is an accepting state.
This sequence has length $n + 1$, which is greater than p. The *pigeonhole principle* tells us that in the first $p + 1$ items in $r_1 r_2 \ldots r_n r_{n+1}$ one state must occur twice.
We suppose it occurs *second* as r_l and *first* as r_j.
Notice: $l \neq j$, and $l \leq p + 1$.
Now let:

- $x = s_1 \ldots s_{j-1}$
- $y = s_j \ldots s_{l-1}$
- $z = s_l \ldots s_n$

So:

- x takes M from r_1 to r_j
- y takes M from r_j to r_j
- z takes M from r_j to r_{n+1}

Hence M accepts $xy^i z, i \geq 0$. Thus we have shown that condition 1 of the pumping lemma holds.
Because $l \neq j$ we know that $|y| \neq 0$. Thus we have shown that condition 2 of the pumping lemma holds.
Because $l \leq p + 1$ we know that $|xy| \leq p$. Thus we have shown that condition 3 of the pumping lemma holds.
Hence the pumping lemma holds.

(c) Using the pumping lemma, or otherwise, show that the language $\{1^n 0^n | n \geq 0\}$ is not regular. [10 marks]

We begin the proof by assuming that this is a regular language, so there is some machine N which accepts it.
Hence, by the pumping lemma, there must be some integer k, such that the string $0^k 1^k$ can be pumped to give a string which is also accepted by N.
We let $xyz = 0^k 1^k$, and show that $xyyz$ is not in $\{0^n 1^n | n \geq 0\}$
There are three cases to consider:

1. y is a sequence of 0s
2. y is a sequence of 0s followed by a sequence of 1s
3. y is a sequence of 1s

In case 1 $xyyz$ will have more 0s than 1s, and so $xyyz \notin L$.
In case 3 $xyyz$ will have more 1s than 0s, and so $xyyz \notin L$.
In case 2 $xyyz$ will have two occurrences of the substring 01, and so $xyyz \notin L$.
So in each case the assumption that $\{0^n 1^n | n \geq 0\}$ is regular leads to a contradiction.
So $\{0^n 1^n | n \geq 0\}$ is not regular.

****