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

EXAMINATIONS — 2005

TERM TEST

COMP 202
Formal Methods of Computer Science
WITH ANSWERS

Time Allowed: 90 minutes

Instructions: There are **five** (5) questions.
Each question is worth **twenty** (20) marks.
Answer **all** the questions.
Show **all** your working.

Question 1.

The following while-program takes as input an integer n and a sequence $a[0..n-1]$ of integers. It will return an integer m that is the value of the minimum element in the sequence.

procedure *Min* (**in** n, a; **out** m);
begin
$\{I_0\}$ $i := 0$;
$\{I_1\}$ $j := 0$;
$\{I_2\}$ **while** $i < n$ **do**
$\{I_3\}$ **if** $a[j] < a[i]$ **then**
$\{I_4\}$ $j := i$
$\{I_5\}$ **else**
$\{I_6\}$ **skip**;
$\{I_7\}$ $i := i + 1$;
$\{I_8\}$ $m := a[j]$
$\{I_9\}$ **end**

The following loop invariant may be used to prove the program correct:

$$I_2 \triangleq n > 0 \quad \wedge \quad 0 \leq i \leq n \quad \wedge \quad ((\forall k \in 0..i-1)a[j] \leq a[k])$$

(a) Write a formal specification (signature, precondition, and postcondition) that this program satisfies. [4 marks]

Input: integer n, integer sequence a
Output: integer m
Precondition: $I_0 \triangleq n > 0$
Postcondition: $I_9 \triangleq ((\forall k \in 0..n-1)m \leq a[k]) \wedge ((\exists k \in 0..n-1)m = a[k])$

(b) Here are proof laws for assignment (:=) and sequence (;):

$$\{Q[e/x]\}x := e\{Q\} \qquad \frac{\{P\}S_1\{Q\} \quad \{Q\}S_2\{R\}}{\{P\}S_1; S_2\{R\}}$$

i. State what needs to be proved to show that I_2 holds upon entry to the loop.
$\{I_0\}i := 0; j := 0\{I_2\}$

ii. Use the laws given above to show that I_2 holds upon entry to the loop. [7 marks]

$I_1 \triangleq n > 0 \wedge i = 0$.
By the assignment law, $\{I_0\}i := 0\{I_1\}$, since $I_1[0/i]$ is equivalent to I_0.
By the assignment law, $\{I_1\}j := 0\{I_2\}$, since $I_2[0/j]$ is equivalent to I_1.
Thus, by the sequence law, $\{I_0\}i := 0; j := 0\{I_2\}$.

(c) Assume that the loop invariant is correctly maintained by the loop body, so that I_2 still holds after the last iteration of the loop.

i. State an assertion I_8 that may be used to verify the program.
Hint: What property of j is required to show $\{I_8\}m := a[j]\{I_9\}$?

$I_8 \triangleq ((\forall k \in 0..n-1)a[j] \leq a[k]) \wedge ((\exists k \in 0..n-1)a[j] = a[k])$

ii. Hence, show that the program is correct. [9 marks]

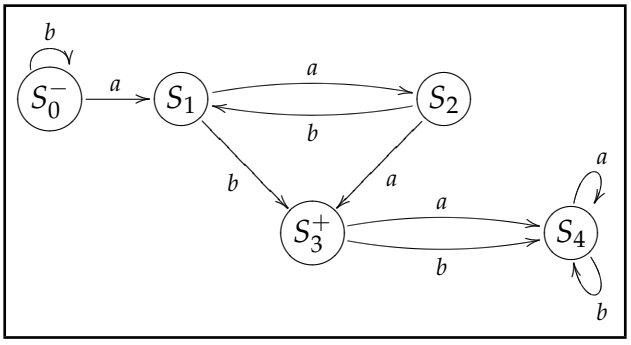
The loop invariant holds before the loop by (b) above.
The loop invariant is maintained by the body by assumption.
Hence, upon exit from the loop, $I_2 \wedge i \geq n$ holds. But this implies $i = n$ and hence I_8.

Question 2.

Let L_1 be the language defined by the FA $M_1 = (Q, \Sigma, \delta, q_0, F)$, where $Q = \{S_0, S_1, S_2, S_3, S_4\}$, $\Sigma = \{a, b\}$, $q_0 = S_0$, $F = \{S_3\}$, and

δ	a	b
S_0	S_1	S_0
S_1	S_2	S_3
S_2	S_3	S_1
S_3	S_4	S_4
S_4	S_4	S_4

(a) Draw M_1. [4 marks]



(b) For each of the following strings, state if it is *accepted* or *rejected* by M_1. (Remember: showing your working may gain you partial credit for incorrect answers.)

i. *baabb*

ii. *babaa*

iii. *aabba*

iv. *aababaa*

v. *aaaa* [10 marks]

i. *accept*: the sequence of transitions is $S_0, b, S_0, a, S_1, a, S_2, b, S_1, b, S_3$ and $S_3 \in F$.

ii. *reject*: $S_0, b, S_0, a, S_1, b, S_3, a, S_4, a, S_4$ and $S_4 \notin F$.

iii. *reject*: $S_0, a, S_1, a, S_2, b, S_1, b, S_3, a, S_4$ and $S_4 \notin F$.

iv. *accept*: $S_0, a, S_1, a, S_2, b, S_1, b, a, S_2, b, S_1, a, S_2, a, S_3$ and $S_3 \in F$.

v. *reject*: $S_0, a, S_1, a, S_2, a, S_3, a, S_4$ and $S_4 \notin F$.

(c) Describe the language L_1. [6 marks]

$L_1 = \text{Language}(\mathbf{b}^*\mathbf{a}(\mathbf{ab})^*(\mathbf{b} + \mathbf{aa}))$.

Question 3.

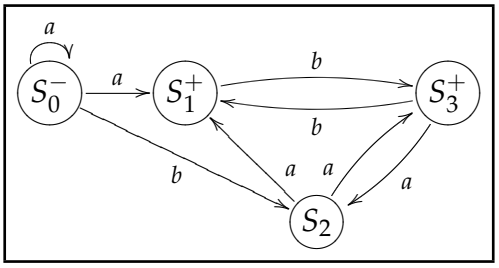
Let L_2 be the language defined by the NFA $M_2 = (Q, \Sigma, \delta, q_0, F)$, where
$Q = \{S_0, S_1, S_2, S_3\}$, $\Sigma = \{a, b\}$, $q_0 = S_0$, $F = \{S_1, S_3\}$, and

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

(a) State Kleene's Theorem. [6 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 described by a regular expression.
A language is regular *iff* it is generated by a regular grammar.

(b) Draw M_2. [4 marks]



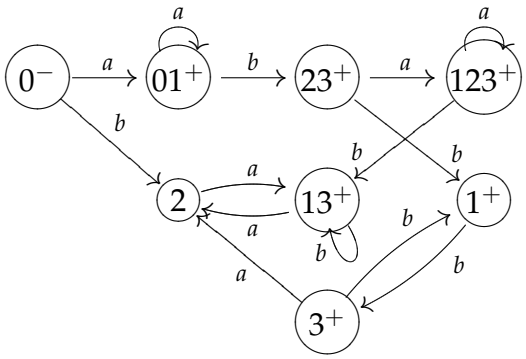
(c) Find an FA which accepts L_2. [10 marks]

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

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

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

Note: $\{\}$ may be omitted.



Question 4.

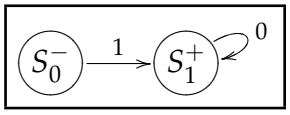
(a) Outline the method for constructing a NFA from a NFA with Λ transitions, such that both machines accept the same language. [8 marks]

See Lecture notes.
Main points:

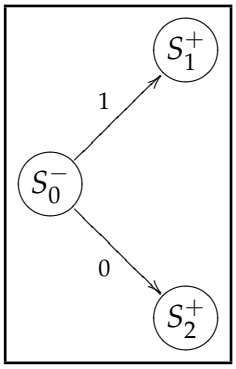
- same states
- same alphabet
- same initial state
- same non-Λ transitions
- for every transition $B \xrightarrow{x} C$, add transitions $A \xrightarrow{x} D$ if B is reachable from A on Λ-transitions alone, and D is reachable from C on Λ-transitions alone. (That is, if D is reachanble from A by a sequence of Λ-transitions, followed by an x-transition, followed by another sequence of Λ-transitions.)
- same final states, with the addition that the initial state is final if there is a path to any final state on Λ-transitions alone.

(b) Given an alphabet $\Sigma = \{a, b\}$, draw NFA with Λ transitions which accept the following languages:

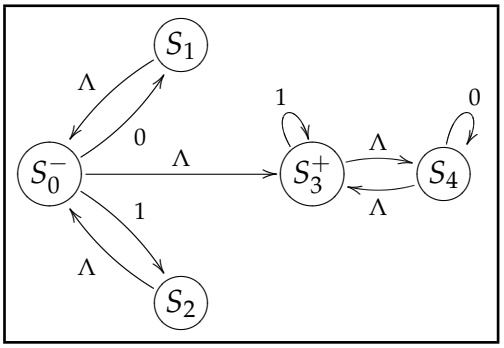
i. Language($\mathbf{ba}^*$)



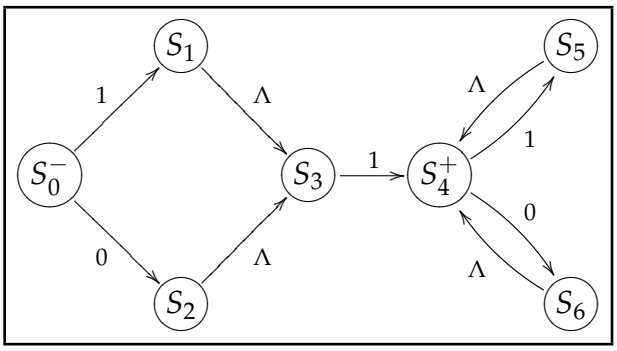
ii. Language($\mathbf{b} + \mathbf{a}$)



iii. Language($(\mathbf{a} + \mathbf{b})^*(\mathbf{b}^*\mathbf{a}^*)^*$)



iv. Language$((\mathbf{b}+\mathbf{a})\mathbf{b}(\mathbf{b}+\mathbf{a})^*)$



[12 marks]

Question 5.

(a) Let L_3 and L_4 be regular languages. Show:

i. $L_3 \cup L_4$ is regular.

By Kleene's theorem, there exist regular expressions $\mathbf{r_3}$ and $\mathbf{r_4}$ for L_3 and L_4.
Let $\mathbf{r} = \mathbf{r_3} + \mathbf{r_4}$.
This regular expression describes the language $L_3 \cup L_4$, and so by Kleene's theorem the language is regular.

ii. $\overline{L_3}$ is regular.

By Kleene's theorem, there exists a total finite automaton M_3 that accepts L_3. Let M_3' be a FA whose states Q, alphabet Σ, and transition function δ are the same as M_3's, but whose final states are given by $F' = Q - F$. Now M_3' accepts $\overline{L_3}$, and so by Kleene's theorem the language is regular.

iii. $(L_3L_4)^*$ is regular. [6 marks]

By Kleene's theorem, there exist regular expressions $\mathbf{r_3}$ and $\mathbf{r_4}$ for L_3 and L_4.
Let $\mathbf{r} = (\mathbf{r_3}\mathbf{r_4})^*$.
This regular expression describes the language $(L_3L_4)^*$, and so by Kleene's theorem the language is regular.

(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. [7 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.

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. $p = 2^Q$)
- $s = s_1 s_2 \dots s_{n-1} s_n$ 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 $r_1 r_2 \dots r_n r_{n+1}$ 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 $r_1 r_2 \dots r_n r_{n+1}$ 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:

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

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) Give an example of a non-regular language, and explain why it is non-regular.

[7 marks]

Example: $\{b^n a^n | n \geq 0\}$
Proof: 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 $a^k b^k$ can be pumped to give a string which is also accepted by N.
We let $xyz = a^k b^k$, and show that $xyyz$ is not in $\{a^n b^n | n \geq 0\}$
There are three cases to consider:

1. y is a sequence of as
2. y is a sequence of as followed by a sequence of bs
3. y is a sequence of bs

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

****