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

EXAMINATIONS — 2006

MID-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.

Consider the following informal (and ambiguous) description of a problem:

> Given two strings, determine which comes first in alphabetical order.

(a) Give a formal specification (signature, precondition, and postcondition) for the problem. Clearly state any assumptions you make about the problem, and give clear descriptions (in English or formal) of any notation you introduce. [7 marks]

The first decision is how to return the answer. I have assumed that we will return -1 if the first string should come first, and 1 if the second string comes first.
The next decision is what to do if both strings are the same. A possible solution is to make the precondition say that is not allowed. I have chosen to return 0 in that case.
I also introduce some notation: for strings s and t, I write $s \prec t$ if s precedes t in alphabetical order. That is true when s and t are identical up to some index position, but either (a) they differ in the next position, with the character of s at that position alphabetically preceding the character of t at that position; or (b) that position is the last of s but there is more of t.
A possible formal definition for this is:

$$\begin{aligned} s \prec t \quad \triangleq \quad & (\exists j \in 0..|s|-1 . s[0..j] = t[0..j] \text{ and } s[j+1] < t[j+1]) \\ & \text{or } (|s| < |t| \text{ and } (\forall j \in 0..|s|-1 . s[j] = t[j])) \end{aligned}$$

Now we are ready for the specification:

input:	strings s and t
output:	integer i
precondition:	*true*
postcondition:	$(s \prec t \text{ and } i = -1) \text{ or } (s = t \text{ and } i = 0) \text{ or } (t \prec s \text{ and } i = 1)$

(b) Define an iterative program in the imperative language which satisfies your specification. Use $s[i]$ to access the ith element of a string s (counting from 0), and $|s|$ to determine the length of s. [7 marks]

```
j ← 0; i ← 0;
while j ≤ |s| and j ≤ |t| and i = 0 do
    if s[j] < t[j] then i ← −1
    elsif t[j] < s[j] then i ← 1
    else j ← j + 1;
if i = 0 and j < |s| then i ← 1
elsif i = 0 and j < |t| then i ← −1
```

(c) Define a recursive program in the applicative language which satisfies your specification. Use *head* and *tail* to access the parts of a string, and *empty* to determine if a string is empty. [6 marks]

$findfirst(s, t) \triangleq$
    if $empty(s)$ and $empty(t)$ **then** 0
    elsif $empty(s)$ **then** -1
    elsif $empty(t)$ **then** 1
    elsif $head(s) < head(t)$ **then** -1
    elsif $head(t) < head(s)$ **then** 1
    else $findfirst(tail(s), tail(t))$
Isn't that a lot easier to understand than the imperative version?

Question 2.

(a) Give a proof of the theorem:

> Any language which is definable by a (deterministic) finite automaton is definable by a nondeterministic finite automaton.

[3 marks]

Let M be a FA defining the language. Define a NFA M' with the same states, alphabet, initial state, and final states as M, whose transition function δ' has $\delta'(q, \sigma,) = \{\delta(q, \sigma)\}$ whenever M has a transition $\delta(q, \sigma)$, and $\delta'(q, \sigma) = \{\}$ whenever $\delta(q, \sigma)$ is undefined. Then M' accepts the same language as M.

(b) Outline a method for constructing a (deterministic) finite automaton from a nondeterministic finite automaton with Λ transitions, such that both machines accept the same language. [6 marks]

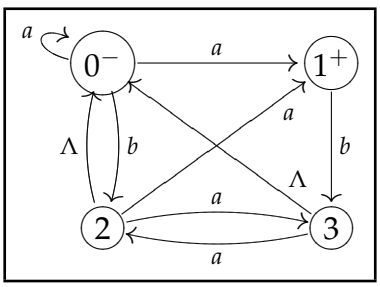
See Lecture notes.

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

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

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

(i) Draw M_1. [3 marks]



(ii) Give a (deterministic) finite automaton which accepts the same language as M_1. You may present your answer as a diagram or in a table. [8 marks]

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

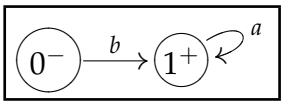
$$
\begin{aligned}
Q' &= \{\{S_0\}, \{S_0, S_1\}, \{S_2\}, \{S_0, S_2\}, \{S_0, S_1, S_2, S_3\}, \{S_0, S_1, S_3\}, \{S_0, S_1, S_2\}\} \\
\Sigma' &= \Sigma \\
q_0' &= \{S_0\} \\
F' &= \{\{S_0, S_1\}, \{S_0, S_1, S_2, S_3\}, \{S_0, S_1, S_2\}, \{S_0, S_1, S_3\}\}
\end{aligned}
$$

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

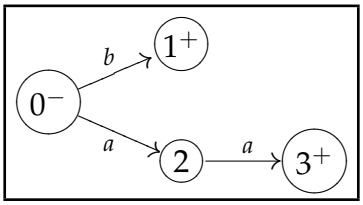
Question 3.

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

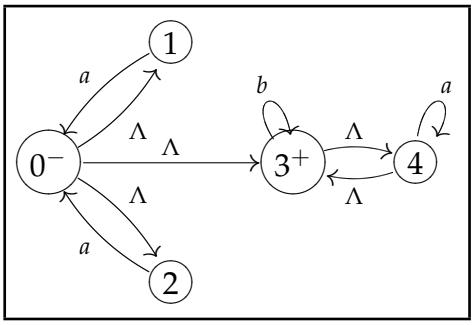
(i) *Language*$(\mathbf{ba}^*)$ [3 marks]



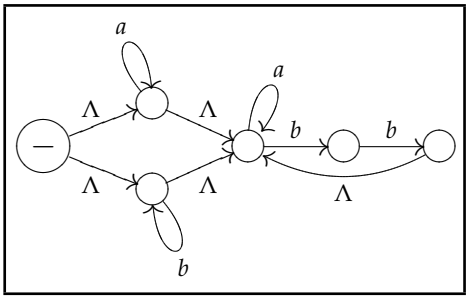
(ii) *Language*$(\mathbf{b} + \mathbf{aa})$ [3 marks]



(iii) *Language*$((\mathbf{a} + \mathbf{b})^*(\mathbf{b}^*\mathbf{a}^*)^*)$ [4 marks]



(iv) *Language*$((\mathbf{a}^* + \mathbf{b}^*)(\mathbf{a}^*\mathbf{bb})^*)$ [4 marks]



(b) Explain the steps involved in constructing a program which, given a regular expression **r** and a string s, will decide whether $s \in$ *Language*$(\mathbf{r})$. [6 marks]

First construct a NFAΛ corresponding to the regular expression.
Next construct a FA corresponding to the NFAΛ.
Finally, construct a scanner which simulates the effect of the FA: the initial state is q_0; while there is more string to consume and there is a transition from the current state and the next symbol of the string, change the current state to the target of that transition and repeat; if the string is all consumed and the current state is final, accept; otherwise reject.

Question 4.

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

$$\begin{aligned} Q &= \{A, B, C\} \\ \Sigma &= \{a, b\} \\ q_0 &= A \\ F &= \{B\} \end{aligned}$$

δ	a	b
A	B	C
B	$-$	B
C	A	B

(a) Give a regular grammar that generates the language accepted by M_2. [5 marks]

$G = (N, \Sigma, S, P)$ where $N = \{A, B, C\}$; $\Sigma = \{a, b\}$; $S = A$; and P is given by

$$\begin{array}{ll} A \to aB & A \to bC \\ B \to \Lambda & B \to bB \\ C \to aA & C \to bB \end{array}$$

(b) Define a recursive scanner that recognizes the language accepted by M_2. [7 marks]

$A(s) \triangleq$ **if** $empty(s)$ **then reject**
elsif $head(s) = a$ **then** $B(tail(s))$
elsif $head(s) = b$ **then** $C(tail(s))$
else reject

$B(s) \triangleq$ **if** $empty(s)$ **then accept**
elsif $head(s) = b$ **then** $B(tail(s))$
else reject

$C(s) \triangleq$ **if** $empty(s)$ **then reject**
elsif $head(s) = a$ **then** $A(tail(s))$
elsif $head(s) = b$ **then** $B(tail(s))$
else reject

(c) Give a regular expression that describes the language accepted by M_2. [8 marks]

The equations are $\mathbf{A} = \mathbf{aB} + \mathbf{bC}$; $\mathbf{B} = \mathbf{\Lambda} + \mathbf{bB}$; $\mathbf{C} = \mathbf{aA} + \mathbf{bB}$.
$\mathbf{B}$ is already in tail-recursive form, so $\mathbf{B} = \mathbf{b}^*$.
Substituting $\mathbf{B}$ and $\mathbf{C}$ in $\mathbf{A}$ gives $\mathbf{A} = \mathbf{ab}^* + \mathbf{baA} + \mathbf{bbb}^*$.
Now $\mathbf{A} = (\mathbf{ab}^* + \mathbf{bbb}^*) + \mathbf{baA}$, which is tail recursive, so the regular expression is $(\mathbf{ba})^*(\mathbf{ab}^* + \mathbf{bbb}^*)$

Question 5.

(a) 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. $|xy| \leq p$

(i) Explain the relevance of *Kleene's theorem* and the *pigeonhole principle* in the proof of the pumping lemma. [6 marks]

See Lecture notes.

(ii) Give an example of a non-regular language, and show that it is non-regular. [8 marks]

See Lecture notes.

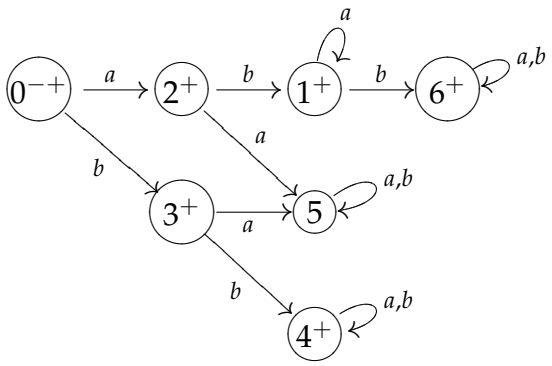
(b) Let $M_3 = (Q, \Sigma, \delta, q_0, F)$, where

$$\begin{aligned} Q &= \{S_0, S_1, S_2, S_3, S_4, S_5\} \\ \Sigma &= \{a, b\} \\ q_0 &= S_0 \\ F &= \{S_5\} \end{aligned}$$

δ	a	b
S_0	S_2	S_3
S_1	S_1	$-$
S_2	S_5	S_1
S_3	S_5	S_4
S_4	S_4	S_4
S_5	S_5	S_5

Give a finite automaton which accepts exactly those strings which are *not* accepted by M_3. [6 marks]

First, make the FA total, by adding a new state S_6 with a transition from S_1 on b to S_6, and transitions on a and b from S_6 to S_6.
Then set $F = \overline{F}$.
The resulting FA looks like:



****