Two hours - online

The exam is hybrid and will be taken on line and answered on paper.

UNIVERSITY OF MANCHESTER
SCHOOL OF COMPUTER SCIENCE

Fundamentals of Computation

Date: Tuesday 5th June 2018
Time: 09:45 - 11:45

This is a hybrid examination with sections to be answered online and questions to be answered on paper.

Please answer All Questions

Use a SEPARATE answerbook for each paper based SECTION

This is a CLOSED book examination

The use of electronic calculators is permitted provided they are not programmable and do not store text

[PTO]
Section A and Section B contain multiple choice questions (MCQs) and are restricted for publication.
20. Consider the language $L$ defined by the regular expression $(0|1)^*(01|10)$. An NFA for the language is given below:

![NFA Diagram](attachment://nfa.png)

Proposition: Any DFA that accepts precisely the language $L$ must have at least two distinct accepting states.

Provide a justification for the above proposition OR demonstrate that it is false.

(5 marks)

21. Consider the following DFA over the alphabet $\{a, b, c\}$. Give a regular expression that defines the same language as the DFA.

![DFA Diagram](attachment://dfa.png)

(6 marks)
22. For each of the languages below which are both over the alphabet \( \{a, b\} \), either

(a) give a DFA that recognises the language;

OR

(b) provide an argument as to why a DFA that recognises the language cannot be produced

For a given word \( w \), \( |w_a| \) is the number of occurrences of the character \( a \) in the word, i.e. if \( w = abaab \), then \( |w_a| = 3 \) and \( |w_b| = 2 \).

a) All words \( w \) s.t. \( |w_a| > |w_b| \). (2 marks)

b) All words where the following equality holds:

\[
|w_a| \mod 2 = |w_b| \mod 2
\]

(2 marks)
23. This question considers the following program that takes two inputs in \( x \) and \( y \) and returns its output in \( z \).

\[
\begin{align*}
z &:= 1 \\
\text{while } 1 \leq x \text{ do } (z := z \times 2; x := x - 1); \\
   z &:= (z \times ((2 \times y) + 1)) - 1
\end{align*}
\]

This program computes the bijective coding function \( \phi(x, y) = 2^x(2y + 1) - 1 \) introduced in the course.

a) Explain how this program can be used to restrict our attention to functions of the form \( \mathbb{N} \to \mathbb{N} \) when discussing the computational power of the while language. (1 mark)

b) The following gives the partial correctness specification for the above program (represented here by \( S \)).

\[
\{ a = x \land x \geq 0 \land y \geq 0 \} \ S \{ z = 2^a(2y + 1) - 1 \}
\]

Why do we need to introduce the new variable \( a \)? (1 mark)

c) Give a loop invariant \( I \) that satisfies the following specification

\[
\{ I \} \text{ while } 1 \leq x \text{ do } (z := z \times 2; x := x - 1) \{ I \land \neg(1 \leq x) \}
\]

and can be used in the proof for the partial correctness statement given in (b) e.g.

- It is true in the state before the loop where \( z = 1 \) and \( x = a \), and
- \( (I \land \neg(1 \leq x)) \rightarrow (z(2y + 1) - 1 = 2^a(2y + 1) - 1) \)

(2 marks)

d) Use the axiomatic rules at the end of this paper to prove that the above program satisfies the partial correctness specification given in (c) (3 marks)
24. Consider the following while program that takes an input in \( n \).

\[
y := 0; \quad r := 0; \quad \text{while } y < 100 \text{ do (}
\begin{align*}
  &z := 0; \\
  &\text{while } z < n \text{ do (}
  &\quad z := z + 1; \\
  &\quad r := r + y; \\
  &\quad \text{)}; \\
  &y := y + 1
\end{align*}
\text{)}
\]

State with justification whether this program belongs to the following classes:

- \( O(1) \)
- \( O(n) \)
- \( O(n^2) \)
- \( \Theta(n^2) \)

(4 marks)

25. Through the use of proof sketches and/or examples, argue for the existence of undecidable predicates on natural numbers (e.g. boolean functions of the form \( f : \mathbb{N} \rightarrow \mathbb{B} \)).

(4 marks)
### Semantics of the while language

<table>
<thead>
<tr>
<th>Rule</th>
<th>Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ass]</td>
<td>( &lt; x := a, s &gt; \Rightarrow s[x \mapsto A[a]] )</td>
</tr>
<tr>
<td>[skip]</td>
<td>( &lt; \text{skip}, s &gt; \Rightarrow s )</td>
</tr>
<tr>
<td>[comp']</td>
<td>( \frac{&lt; S_1, s &gt; \Rightarrow &lt; S'_1, s'&gt;}{&lt; S_1 ; S_2, s &gt; \Rightarrow &lt; S'_1 ; S_2, s'&gt;} )</td>
</tr>
<tr>
<td>[comp?]</td>
<td>( \frac{&lt; S_1, s &gt; \Rightarrow s'}{&lt; S_1 ; S_2, s &gt; \Rightarrow &lt; S_2, s'&gt;} )</td>
</tr>
<tr>
<td>[if]</td>
<td>( &lt; \text{if } b \text{ then } S_1 \text{ else } S_2, s &gt; \Rightarrow &lt; S_1, s &gt; ) if ( B[b] s = \text{tt} )</td>
</tr>
<tr>
<td>[if]</td>
<td>( &lt; \text{if } b \text{ then } S_1 \text{ else } S_2, s &gt; \Rightarrow &lt; S_2, s &gt; ) if ( B[b] s = \text{ff} )</td>
</tr>
<tr>
<td>[while]</td>
<td>( &lt; \text{while } b \text{ do } S, s &gt; \Rightarrow &lt; \text{if } b \text{ then } (S; \text{while } b \text{ do } S) \text{ else skip}, s &gt; )</td>
</tr>
</tbody>
</table>

### Axiomatic Rules for Partial Correctness of while

<table>
<thead>
<tr>
<th>Rule</th>
<th>Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>ass_p</td>
<td>( { P[x \mapsto A[a]] } x := a { P } )</td>
</tr>
<tr>
<td>skip_p</td>
<td>( { P } \text{skip} { P } )</td>
</tr>
<tr>
<td>comp_p</td>
<td>( \frac{{ P } S_1 { Q }, { Q } S_2 { R }}{{ P } S_1 ; S_2 { R }} )</td>
</tr>
<tr>
<td>if_p</td>
<td>( \frac{{ P \land B[b] } S_1 { Q }, { P \land \neg B[b] } S_2 { Q }}{{ P } \text{if } b \text{ then } S_1 \text{ else } S_2 { Q }} )</td>
</tr>
<tr>
<td>while_p</td>
<td>( \frac{{ P \land B[b] } S { P }}{{ P } \text{while } b \text{ do } S { P \land \neg B[b] }} )</td>
</tr>
<tr>
<td>cons_p</td>
<td>( \frac{{ P' } S { Q' }}{{ P } S { Q }} ) if ( P \Rightarrow P' ) and ( Q' \Rightarrow Q )</td>
</tr>
</tbody>
</table>