PMA/1050 Discrete Foundations January 2005 — Solutions

1. (i) [9 marks: unseen problem]

(a)
$$(A \cap B) \cup C = \{3,6\} \cup \{1,4,6\} = \{1,3,4,6\} \qquad \textbf{[3 marks]}$$

(b)
$$(A \cup B) \setminus C = \{1,2,3,5,6\} \setminus \{1,4,6\} = \{2,3,5\} \qquad \textbf{[3 marks]}$$

(c)
$$(A \setminus B) \cap C = \{1,5\} \cap \{1,4,6\} = \{1\} \qquad \textbf{[3 marks]}$$

1. (ii)[8 marks: unseen problem]

1. (iii) [6 marks: unseen problem]

(a)
$$A \times B = \{(a,p),(b,p),(c,p),(a,q),(b,q),(c,q)\}. \qquad \textbf{[2 marks]}$$

(b) We have $P(a,p) = a = P(a,q)$, so P is not injective. **[2 marks]** Every $x \in A$ can be written as $x = P(x,p)$ and is therefore in the range of P, so P is surjective. **[2 marks]**

1. (iv) [11 marks: unseen problem]

The function f is both injective **[1 mark]** and surjective **[1 mark]** because it is invertible: it is its own inverse **[1 mark]** .

The function g is not surjective **[1 mark]** since no odd length string is in the image **[1 mark]** , and it is not injective **[1 mark]** since, for example, $g(ab) = ab = g(abc)$. **[1 mark]**

The function h is not injective, **[1 mark]** since, for example $h(ab) = a = h(ac)$ (or, even simpler, $h(a) = \varepsilon = h(\varepsilon)$), **[1 mark]** but it is surjective **[1 mark]** since any string x may be written as $x = h(xa)$. **[1 mark]**

2. (i) [11 marks: unseen problem]

Induction base. For $n = 1$, the equation reduces to

$$1^3 = \frac{1^2 \times 2^2}{4}$$

so the result holds for $n = 1$.**[2 marks]**

Induction step. We now assume that the theorem is true for a given arbitrary value of n, and show that it is true for $n + 1$. **[1 mark]** Thus we assume:

Induction hypothesis:

$$\sum_{k=1}^{n} k^3 = \frac{n^2(n+1)^2}{4}. \qquad \textbf{[1 mark]}$$

Then

$$\begin{aligned}
\sum_{k=1}^{n+1} k^3 &= \frac{n^2(n+1)^2}{4} + (n+1)^3, \text{ by the induction hypothesis,} \qquad \textbf{[2 marks]} \\
&= \frac{n^2(n+1)^2 + 4(n+1)(n+1)^2}{4} \\
&= \frac{(n+1)^2(n^2+4n+4)}{4} \\
&= \frac{(n+1)^2(n+2)^2}{4} \\
&= \frac{(n+1)^2((n+1)+1)^2}{4} \qquad \textbf{[3 marks]}
\end{aligned}$$

which is the original formula with $(n + 1)$ in place of n.**[1 mark]** This completes the induction step.

Conclusion: By mathematical induction, the result holds for all $n \geqslant 1$.**[1 mark]**

2. (ii) [13 marks: unseen problem]

$$\begin{aligned}
a_1 &= 1, \\
a_2 &= 1 + \frac{1}{2} = \frac{3}{2}, \\
a_3 &= \frac{3}{2} + \frac{1}{6} = \frac{5}{3}, \\
a_4 &= \frac{5}{3} + \frac{1}{12} = \frac{20}{12} + \frac{1}{12} = \frac{21}{12} = \frac{7}{4}. \qquad \textbf{[3 marks]}
\end{aligned}$$

We prove $a_n = \dfrac{2n-1}{n}$.

Induction base. The formula holds for $n = 1$ since $\dfrac{2.1-1}{1} = 1 = a_1$.**[2 marks]**

Induction step. We now assume that the theorem is true for a given arbitrary value of n, and show that it is true for $n + 1$. **[1 mark]** Thus we assume:

Induction hypothesis:

$$a_n = \frac{2n-1}{n}. \qquad \textbf{[1 mark]}$$

Then

$$a_{n+1} = a_n + \frac{1}{n(n+1)}, \text{ by definition,} \qquad \textbf{[1 mark]}$$

$$
\begin{aligned}
&= \frac{2n-1}{n} + \frac{1}{n(n+1)}, \text{ by the induction hypothesis,} \qquad \textbf{[2 marks]} \\
&= \frac{(2n-1)(n+1)+1}{n(n+1)} \\
&= \frac{2n^2+n-1+1}{n(n+1)} \\
&= \frac{2n+1}{(n+1)} \\
&= \frac{2(n+1)-1}{(n+1)} \qquad \textbf{[1 mark]}
\end{aligned}
$$

which is the original formula with $(n+1)$ in place of n.[**1 mark**] This completes the induction step.

Conclusion: By mathematical induction, the result holds for all $n \geqslant 1$.[**1 mark**]

2. (iii) [10 marks: unseen problem]

(a)

P	Q	$P \vee Q$	$P \wedge Q$	$P \Rightarrow Q$	$(P \vee Q) \Rightarrow (P \wedge Q)$	expression
T	T	T	T	T	T	T
T	F	T	F	F	F	T
F	T	T	F	T	F	T
F	F	F	F	T	T	T

This is a tautology.[**5 marks**]

(b)

P	Q	$\neg Q$	$P \Rightarrow Q$	$(P \Rightarrow Q) \Rightarrow \neg Q$	$Q \wedge ((P \Rightarrow Q) \Rightarrow \neg Q)$
T	T	F	T	F	F
T	F	T	F	T	F
F	T	F	T	F	F
F	F	T	T	T	F

This is a contradiction.[**5 marks**]

3. (i) [8 marks: bookwork]

- Q is a **finite** set; the **set of states** — the machine is always in one of these states; **[1 mark]**
- Σ is a finite set of characters; the **input alphabet** — input strings are made up from these;**[1 mark]**
- $\delta : Q \times \Sigma \to Q$ is the **transition function** — if the machine is in state q and receives an input character a, it changes to state $\delta(q, a)$ and awaits the next input;**[2 marks]**
- $q_0 \in Q$; the **initial state** — the state in which the machine starts;**[1 mark]**
- $F \subseteq Q$; the set of **final** or acceptance states.**[1 mark]**

The language $L(M)$ is the set of strings of characters in the input alphabet which when input cause the machine to proceed to one of the final states.**[2 marks]**

3. (ii) [8 marks: unseen problem]

[3 marks]

The action of the FA is to move to state q_2 on receipt of either 22 or 21 **[2 marks]** having reached q_2, it stays there and accepts the string. **[1 mark]** Thus $L(M)$ is the set of all strings containing either 22 or 21. **[2 marks]**

3. (iii) [8 marks: unseen problem]

We design a FA with $Q = \{q_0, q_1, q_2, q_3\}$, **[1 mark]** initial state q_0 and set of final states $\{q_3\}$ **[1 mark]** such that The state $\hat{\delta}(q_0, w)$ reached by the FA on input $w \in \Sigma^*$ is:
q_0 if w has an even number of 0s and an even number of 1s;
q_1 if w has an odd number of 0s and an even number of 1s;
q_2 if w has an even number of 0s and an odd number of 1s;
q_3 if w has an odd number of 0s and an odd number of 1s. **[2 marks]**

[4 marks]

3. (iv) [10 marks: unseen problem]

A simple FA accepting the set of all strings containing precisely two occurrence of the character b would be the following.

Either from considering this FA, **or** directly, we see that a regular expression corresponding to the set of all strings containing precisely two occurrence of the character b is $(\boldsymbol{a} \cup \boldsymbol{c})^* \boldsymbol{b} (\boldsymbol{a} \cup \boldsymbol{c})^* \boldsymbol{b} (\boldsymbol{a} \cup \boldsymbol{c})^*$ **[4 marks]**

A simple FA accepting the set of all strings not containing the substring *cab* would be the following.

Hence, a regular expression corresponding to the set of all strings not containing the substring *cab* is

$$(\boldsymbol{a} \cup \boldsymbol{b} \cup \boldsymbol{c}\boldsymbol{c}^* \boldsymbol{b} \cup \boldsymbol{c}\boldsymbol{c}^* \boldsymbol{a} (\boldsymbol{a} \cup \boldsymbol{c}))^* \, (\varepsilon \cup \boldsymbol{c}\boldsymbol{c}^* (\varepsilon \cup \boldsymbol{a}). \qquad \textbf{[6 marks]}$$

[The reasoning here is that $\boldsymbol{a} \cup \boldsymbol{b} \cup \boldsymbol{c}\boldsymbol{c}^* \boldsymbol{b} \cup \boldsymbol{c}\boldsymbol{c}^* \boldsymbol{a} (\boldsymbol{a} \cup \boldsymbol{c})$ correspond to the input strings that cause the machine to move from q_0 to itself with no intermediate visit to q_0: in fact $\boldsymbol{a}$ and $\boldsymbol{b}$ cause a direct move from q_0 to itself; $\boldsymbol{c}\boldsymbol{c}^* \boldsymbol{b}$ causes movement from q_0 to q_1 and thence, possibly after some moves from q_1 to itself, back to q_0. The string $\boldsymbol{c}\boldsymbol{c}^* \boldsymbol{a} (\boldsymbol{a} \cup \boldsymbol{c})$ causes movement from q_0 to q_1 and thence, possibly after some moves from q_1 to itself, on to q_2 and then back to q_0.

Therefore

$$(\boldsymbol{a} \cup \boldsymbol{b} \cup \boldsymbol{c}\boldsymbol{c}^* \boldsymbol{b} \cup \boldsymbol{c}\boldsymbol{c}^* \boldsymbol{a} (\boldsymbol{a} \cup \boldsymbol{c}))^*$$

are the strings which are accepted by ending at q_0,

$$(\boldsymbol{a} \cup \boldsymbol{b} \cup \boldsymbol{c}\boldsymbol{c}^* \boldsymbol{b} \cup \boldsymbol{c}\boldsymbol{c}^* \boldsymbol{a} (\boldsymbol{a} \cup \boldsymbol{c}))^* \, \boldsymbol{c}\boldsymbol{c}^*$$

are the strings which are accepted by ending at q_1, and

$$(\boldsymbol{a} \cup \boldsymbol{b} \cup \boldsymbol{c}\boldsymbol{c}^* \boldsymbol{b} \cup \boldsymbol{c}\boldsymbol{c}^* \boldsymbol{a} (\boldsymbol{a} \cup \boldsymbol{c}))^* \, \boldsymbol{c}\boldsymbol{c}^* \boldsymbol{a}$$

are the strings which are accepted by ending at q_2. The desired answer is the union of these.]

4. (i) [8 marks: bookwork]
A Mealy machine consists of a 6-tuple $M = (Q, \Sigma, \Gamma, q_0, \delta, \omega)$ where:

- Q is a finite set of states;**[1 mark]**
- Σ is a finite set of input symbols;**[1 mark]**
- Γ is a finite set of output symbols;**[1 mark]**
- q_0 is the initial state;**[1 mark]**
- $\delta : Q \times \Sigma \to Q$ is the state transition function;**[1 mark]**
- $\omega : Q \times \Sigma \to \Gamma$ is the output function.**[1 mark]**

If M is in state $q \in Q$ with an incoming input $a \in \Sigma$, the state changes to $\delta(q, a) \in Q$ **[1 mark]** and outputs the character $\omega(q, a) \in \Gamma$.**[1 mark]**

4. (ii) [8 marks: unseen problem]The machine has two states q_0, q_1, corresponding to the last input digit being 0,1 respectively.**[2 marks]**
State diagram:

[4 marks] The initial state is q_0 and the transition/output table

	input	q_0	q_1
δ	0	q_0	q_0
δ	1	q_1	q_1
ω	0	0	1
ω	1	1	2

[2 marks]

4. (iii) [10 marks: unseen problem]
The states r, s are unreachable, so may be removed. **[2 marks]**
The remaining states fall into the following 1-equivalence classes:

$$\{p, q, u, v\}, \quad \{t\}. \qquad \textbf{[2 marks]}$$

The 2-equivalence classes are $\{p, v\}, \quad \{q, u\}, \quad \{t\}.$ **[2 marks]**
The 3-equivalence classes are $\{p, v\}, \quad \{q, u\}, \quad \{t\}.$ **[2 marks]**
The situation has stabilized, so we have a minimal machine with states w, x, corresponding to $\{p, v\}, \{q, u\}$, respectively, and t. Its transition/output table is

		w	x	t
δ	0	t	x	x
δ	1	x	w	w
ω	0	0	0	1
ω	1	1	1	0

[2 marks]

4. (iv) [8 marks: unseen problem]
This grammar is context-free, since every production has just one non-terminal variable on the left-hand side. **[1 mark]**

The grammar is not regular, since the productions $S \to XaSbX$ is neither left- nor right-linear. **[1 mark]**

Generation in this grammar necessarily takes the form

$$S \stackrel{*}{\Rightarrow} (Xa)^n S(bX)^n \Rightarrow (Xa)^n S(bX)^n \stackrel{*}{\Rightarrow} w_1 c w_2,$$

where w_1, w_2 strings of as and bs of equal and even length (possibly zero) and such that every even character of w_1 is an a and every odd character of w_2 is a b; the language generated is the set of all such strings. **[6 marks]**