

UG3 COMPUTABILITY AND INTRACTABILITY (2009-2010)  
EXERCISE SHEET 1: SAMPLE SOLUTIONS

*Submission date:* Thursday 15 October.

1. (a) The function  $f$  is computable. Given  $n$  we can construct  $P_n$  and run it on  $n$ . If we get a result we just add 1 and return that as the answer. [3 marks]

(b) Suppose that  $g$  extends  $f$ , is computable and total. Then there is a program  $P_m$  that computes  $g$ . But now we have a contradiction since by definition the value of  $g(m)$  is  $P_m(m) + 1$ , i.e.,  $g(m) + 1$ . Thus  $g$  cannot be computable *and* total. [4 marks]

(c) Just define

$$g(n) = \begin{cases} f(n), & \text{if } f \text{ is defined on } n, \\ 0, & \text{otherwise.} \end{cases}$$

This is clearly a total function that extends  $f$ . It cannot be computable by the preceding part. [4 marks]

2. (a) Here is one possible machine:

# Given an input string  $\$B@$  where  $B$  is a binary string the machine  
# halts with  $\$B@B$  on the tape (and accepts the string).

Q = {left, right, zero, one, halt}  
I = right  
F = halt  
S = {0, 1, \$, @}  
G = {0, 1, A, B, \$, @, b}  
D = {(right, 0, zero, A, R), (right, 1, one, B, R),  
(right, @, halt, @, L), (right, \$, right, \$, R),  
(zero, b, left, 0, L), (zero, ?, zero, ?, R),  
(one, b, left, 1, L), (one, ?, one, ?, R),  
(left, A, right, 0, R), (left, B, right, 1, R),  
(left, ?, left, ?, L)}

[6 marks]

(b) [2 marks]

(c) [3 marks]

3. (a) We use a three tape machine. Our first action is to copy the input string on the second tape. On the third tape we list the strings in  $\Sigma^*$  one at a time. After each string is generated we copy the saved input string on the second tape to the first tape and then append the string on the third tape (we do the usual sensible thing about marking the first square of the tapes so that we don't fall off). After

this we return to the first (non marked) square on the first tape and simulate the machine that accepts  $L$ . This will of course halt. If it accepts we accept otherwise we start the cycle all over. [7 marks]

(b) The result is still true. We modify the construction above by, e.g., keeping a counter  $n$ . At each phase we try the computation of the preceding part for all strings of length at most  $n$  and for each attempt we compute for at most  $n$  steps (i.e., a version of dovetailing). [4 marks]

Don Sannella, September 2009