CSE 102
Spring 2024
Midterm Exam 2 **Solutions**

1. (25 Points) The recursive algorithm below determines whether an array is sorted. Variables B_1, B_2 and B_3 are Boolean, and $\wedge$ represents the Logical And operator.

```
Sorted(A, p, r)   precondition: r ≥ p
1. if r = p
2.     return TRUE
3. else
4.     q = ⌊(p + r)/2⌋
5.     B1 = Sorted(A, p, q)
6.     B2 = Sorted(A, q + 1, r)
7.     B3 = (A[q] ≤ A[q + 1])
8.     return (B1 ∧ B2 ∧ B3)
```

a. (15 Points) Use induction on $m = \text{length}(A[p \cdots r])$ to prove the correctness of the above algorithm, i.e. prove that Sorted(A, p, r) returns TRUE if and only if $A[p \cdots r]$ is sorted in increasing order.

Proof:

I. Let $m = 1$. Then $\text{length}(A[p \cdots r]) = r - p + 1 = 1 \Rightarrow r = p$, and TRUE is returned on line 2 of the algorithm. Indeed, an array of length 1 is always sorted, so the algorithm returns a correct value. The base case is therefore established.

II. Let $m > 1$ and assume Sorted() returns a correct value on all sub-arrays of length less than m. We must show that Sorted() returns a correct value when run on any sub-array of length m. Since $m > 1$, we have $m = r - p + 1 > 1 \Rightarrow r > p$, so line 2 is skipped and lines 4-8 are executed.

Also

$$\begin{aligned} p < r &\Rightarrow p + r < 2r \Rightarrow \lfloor (p+r)/2 \rfloor < r \Rightarrow q < r \\ &\Rightarrow q - p + 1 < r - p + 1 \\ &\Rightarrow \text{length}(A[p \cdots q]) < m \end{aligned}$$

and

$$\begin{aligned} p < r &\Rightarrow 2p < p + r \Rightarrow p < \frac{p+r}{2} \Rightarrow p \le \lfloor (p+r)/2 \rfloor \\ &\Rightarrow p < \lfloor (p+r)/2 \rfloor + 1 \Rightarrow p < q + 1 \\ &\Rightarrow r - q < r - p + 1 \\ &\Rightarrow \text{length}(A[q+1 \cdots r]) < m \end{aligned}$$

The induction hypothesis guarantees that lines (5) and (6) return correct values for sub-arrays $A[p \cdots q]$ and $A[q+1 \cdots r]$. Observe $A[p \cdots r]$ is sorted in increasing order if and only if: $A[p \cdots q]$ is sorted, $A[q+1 \cdots r]$ is sorted and $A[q] \le A[q+1]$. Thus $A[p \cdots r]$ is sorted if and only if the value of the Boolean expression $B_1 \wedge B_2 \wedge B_3$ returned on line (8) is TRUE. Therefore, Sorted(A, p, r) returns TRUE if and only if $A[p \cdots r]$ is sorted in increasing order, as required. ■

b. (10 Points) Let $T(n)$ denote the number of array comparisons performed by Sorted() on an array of length n. Write a recurrence relation for $T(n)$. Determine a tight asymptotic bound for $T(n)$.

Solution:
If $p = 1$, $r = n$, and $q = \lfloor (n + 1)/2 \rfloor$ then length$(A[1 \cdots q]) = q = \lfloor (n + 1)/2 \rfloor = \lceil n/2 \rceil$, and length$(A[q + 1 \cdots n]) = n - q = n - \lceil n/2 \rceil = \lfloor n/2 \rfloor$. Therefore $T(n)$ must satisfy the recurrence

$$T(n) = \begin{cases} 0 & n = 1 \\ T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + 1 & n \geq 2 \end{cases}$$

First first simplify the recurrence to $T(n) = 2T(n/2) + 1$. We compare $1 = n^0$ to $n^{\log_2(2)} = n^1$. Let $\epsilon = 1 - 0 = 1$. Then $\epsilon > 0$ and $1 = O(n^0) = O\left(n^{\log_2(2) - \epsilon}\right)$, and by case (1) we have $T(n) = \Theta(n)$.

■

Alternative Solution:
One can show directly that $T(n) = n - 1$ is an exact solution to this recurrence. First note that when $n = 1$, $T(1) = 0$. If $n \geq 1$ then

$$\begin{aligned} \text{RHS} &= T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + 1 \\ &= (\lfloor n/2 \rfloor - 1) + (\lceil n/2 \rceil - 1) + 1 \\ &= (\lfloor n/2 \rfloor + \lceil n/2 \rceil) - 1 \\ &= n - 1 \\ &= T(n) \\ &= \text{LHS} \end{aligned}$$

so $T(n) = n - 1$ solves the recurrence, and $T(n) = \Theta(n)$. ■

2. (25 Points) Suppose we are given an unlimited number of coins in each of the denominations $d = (1, 2, 5, 7, 9)$. We wish to pay $N = 14$ monetary units using the least number of coins. Let $C[i, j]$ denote the minimum number of coins needed to pay j units using only coins in the denominations $(d_1, \dots, d_i)$, where $1 \le i \le 5$ and $0 \le j \le 14$.

   a. (5 Points) Write a recursive formula for $C[i, j]$. Carefully define boundary values and out-of-bounds values in such a way that $C[i, j]$ is defined for all i and j.

   Solution:

$$C[i,j] = \begin{cases} 0 & i \ge 1 \text{ and } j = 0 \\ \min(C[i-1, j], 1 + C[i, j - d_i]) & i \ge 1 \text{ and } j > 0 \\ \infty & i \le 0 \text{ or } j < 0 \end{cases}$$

   b. (10 Points) Fill in the following table containing the values of $C[i, j]$.

									j							
i	d	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	1	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
2	2	**0**	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**	**5**	**5**	**6**	**6**	**7**	**7**
3	5	**0**	**1**	**1**	**2**	**2**	**1**	**2**	**2**	**3**	**3**	**2**	**3**	**3**	**4**	**4**
4	7	**0**	**1**	**1**	**2**	**2**	**1**	**2**	**1**	**2**	**2**	**2**	**3**	**2**	**3**	**2**
5	9	**0**	**1**	**1**	**2**	**2**	**1**	**2**	**1**	**2**	**1**	**2**	**2**	**2**	**3**	**2**

   c. (10 Points) Use this table to determine *two* optimal solutions to this problem, i.e. two different ways to pay 14 monetary units using the least possible number coins. Express your solutions by giving a vector $x = (x_1, x_2, x_3, x_4, x_5)$ for which $\sum_{i=1}^{5} x_i d_i = 14$. (It is not necessary to show your work on this problem.)

   Optimal Solution 1: $\boldsymbol{x} = (\mathbf{0}, \mathbf{0}, \mathbf{1}, \mathbf{0}, \mathbf{1})$**, one 5 unit coin, and one 9 unit coin.**

   Optimal Solution 2: $\boldsymbol{x} = (\mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{2}, \mathbf{0})$**, two 7 unit coins.**

3. (25 Points) A thief wishes to steal objects $\{1, 2, 3, 4, 5, 6\}$, having values $v[1 \cdots 6] = (5, 5, 9, 4, 4, 12)$ and weights $w[1 \cdots 6] = (1, 4, 3, 4, 1, 6)$, where it is permissible to steal a fraction of an object. His goal is to maximize the total value of the goods stolen $\sum_{i=1}^{6} x_i v_i$, where x_i denotes the fraction of object i to be stolen ($0 \le x_i \le 1$ for $1 \le i \le 6$). The total weight of the stolen goods $\sum_{i=1}^{6} x_i w_i$ must not exceed the capacity of his knapsack: $W = 9$. Determine an optimal solution to this problem using a greedy strategy, with selection function $f(i) = v_i / w_i$, i.e. order the objects by decreasing value-to-weight ratios, then steal as much of each object as is possible, in that order, never exceeding the capacity of the knapsack. Express your solution as the vector $x = (x_1, x_2, x_3, x_4, x_5, x_6)$, and give the value of this optimal solution.

Solution:
The value to weight ratios are: $(5, 1.25, 3, 1, 4, 2)$. Thus the thief should steal, in order

- All of object 1 (value 5 and weight 1)
- All of object 5 (value 4 and weight 1)
- All of object 3 (value 9 and weight 3)
- $2/3$ of object 6 (value 8 and weight 4)

The solution vector is therefore $x = (1, 0, 1, 0, 1, 2/3)$, with total weight 9 and total value 26. ■

4. (25 Points) Consider the coin changing problem again, where we have an unlimited number of coins in each of the denominations $d = (1, 5, 10)$, and we apply the *greedy strategy* to pay N monetary units using the fewest number of coins. In other words, start with sum $= 0$. Then, from amongst all coins whose addition to sum would not cause sum to exceed N, choose the largest, and add it to sum. Stop when sum $= N$. Thus we choose as many dimes (10 units) as possible, then choose as many nickles (5 units) as possible, then choose as many pennies (1 unit) as necessary to achieve a sum of N units. Prove that this strategy yields an optimal solution (i.e. fewest number of coins) for any $N \geq 0$. (Hint: let $x = (x_1, x_2, x_3)$ be an optimal solution, and let $g = (g_1, g_2, g_3)$ be the solution produced by the greedy strategy, then prove $x = g$, whence g is optimal.)

Proof:
Let $N \geq 0$, let $x = (x_1, x_2, x_3)$ be an optimal solution, and let $g = (g_1, g_2, g_3)$ be the solution produced by the greedy strategy. It is sufficient to show that $g = x$, for then g is optimal. Since both solutions pay N units, we have

(1)
$$x_1 + 5x_2 + 10x_3 = N = g_1 + 5g_2 + 10g_3$$

Reduce equation (1) modulo 5 to obtain the congruence $x_1 \equiv g_1 (\text{mod } 5)$. Observe that

- $0 \leq x_1 < 5$, since if $x_1 \geq 5$, we could trade 5 pennies for one nickel. This is impossible since x is optimal.
- $0 \leq g_1 < 5$, since the greedy strategy chooses the maximum possible number of nickles before any pennies are chosen.

These facts, together with $x_1 \equiv g_1 (\text{mod } 5)$, imply that $x_1 = g_1$. Cancel this common value from both sides of (1) to obtain $5x_2 + 10x_3 = 5g_2 + 10g_3$, then divide through by 5 to get

(2)
$$x_2 + 2x_3 = g_2 + 2g_3$$

Reduce equation (2) modulo 2 to obtain the congruence $x_2 \equiv g_2 (\text{mod } 2)$. Now observe

- $0 \leq x_2 < 2$, since if $x_2 \geq 2$, we could trade 2 nickles for one dime, again impossible since x is optimal.
- $0 \leq g_2 < 2$, since the greedy strategy chooses the maximum number of dimes before any nickles are chosen.

These facts, together with $x_2 \equiv g_2 (\text{mod } 2)$, imply that $x_2 = g_2$. Cancel this value from both sides of (2) to obtain $2x_3 = 2g_3$, and hence $x_3 = g_3$. Therefore $g = x$, and g is optimal as claimed. ■

Remark: This proof actually shows a little more, namely that in this problem an optimal solution is unique.