INTRO TO ALGORITHMS FINAL SOLUTION

1 True or False, and ***Justify***.

a) The problem of computing Fibonacci numbers is polynomial-time solvable. (6%)

b) The LONGEST-SIMPLE-PATH problem can be solved by dynamic programming. (4%)

Solution:

a) True

LEMMA

$$\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}^n = \begin{bmatrix} F_{n-1} & F_n \\ F_n & F_{n+1} \end{bmatrix} \quad n \geq 1, \text{ where } F_n \text{ is the } n\text{th Fibonacci number}$$

According to this lemma, $\boldsymbol{F_n}$ may be computed by the fast exponentiation algorithm, i.e.

$M^1 = M$

$M^n = M \cdot M^{n-1}$ if $n > 1$ is odd

$= (M^2)^{\frac{n}{2}}$ if $n > 1$ is even

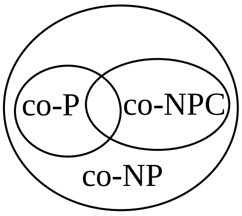
The computation takes a time in $O(\lg n)$, which is polynomial in terms of the input size $\lg n$.

b) False

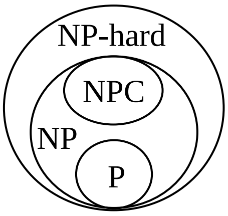
See Chap15, p55.

2 Each diagram below contradicts the current state of our knowledge about the complexity classes. Explain and draw a correct diagram. (12%)

a)



b)



Solution:

a) **Lemma** If $L \in$ co-NPC and $L \in$ co-P, then co-P $=$ co-NP

Proof

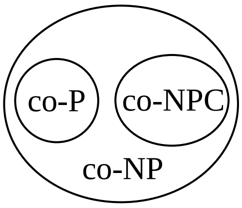
Need only show that co-NP $\subseteq$ co-P

Let $L' \in$ co-NP

Then, L is co-NPC $\Rightarrow L' \leq_p L$

Thus, $L' \leq_p L$ and $L \in$ co-P $(=$ P$)$ $\Rightarrow L' \in$ co-P $(=$ P$)$ ■

Corrected diagram



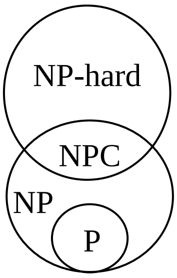
b) **Lemma** If $L \in$ NP and $L \in$ NP-hard, then $L \in$ NPC

Proof

$L \in$ NP-hard $\Rightarrow \forall L' \in$ NP, $L' \leq_p L$

Since $L \in$ NP, it follows that $L \in$ NPC ■

Corrected diagram



3 Consider the MATRIX-CHAIN MULTIPLICATION problem

The dynamic programming recurrence for $m[i.j]$, i.e. the minimal number of multiplications needed to compute the matrix $A_i A_{i+1} \dots A_j$ is defined by

$$m[i,j] = 0, \qquad \text{if } i = j$$
$$= \min_{i \le k < j} \{m[i,k] + m[k+1,j] + p_{i-1}p_k p_j\}, \qquad \text{if } i < j$$

a) Write pseudocode for the *memorized* recursive algorithm that solves the recurrence. (6%)

b) Show that the *memorized* recursive algorithm of part a) runs in $\Theta(n^3)$ time (6%)

Solution:

a) See Chap15, p26

b) See Chap15, pp27~28

4 Consider the knapsack optimization problem

$$\text{Maximize} \sum_{i=1}^{n} v_i x_i \quad \text{subject to} \sum_{i=1}^{n} w_i x_i \le W, \qquad \text{where } x_i = 0 \text{ or } 1$$

Let $c[i,w]$ be the value of the solution for items $1,2 \dots, i$ and knapsack weight w

a) Write down the recurrence for $c[i,w]$. (4%)

b) Show how to compute $c[n,W]$ by bottom-up tabulation. (DON'T write any pseudocode, just explain how.) (6%)
What is the time complexity of your method?

Solution:

a) See Exercise 16.2-2. *The solution is posted publicly*.

b) Same

5 Prove that the problem of determining an optimal prefix code satisfies the greedy-choice property stated below.

Let x and y be two characters having the lowest frequencies. Then, there is an optimal prefix code tree in which x and y are siblings.

(Note: You may use the fact that an optimal prefix code tree is a full binary tree i.e. every internal node has two children.) (8%)

Solution:

See Chap16, pp36~38

6 Consider a sequence of insertion or deletion operations on a dynamic table T discussed in class.

Let $\alpha_i = num_i / size_i =$ load factor after the i^{th} operation

a) Use the accounting method to determine the amortized cost of each operation below. (6%)

1. Insertion, $\alpha_{i-1} \geq 1/2$
2. Deletion, $\alpha_i \geq 1/2$

b) Use the potential method to determine the amortized cost of each operation below. (6%)

1 Insertion, $\alpha_{i-1} < 1/2$

2 Deletion, $\alpha_{i-1} \leq 1/2$

Solution:

a) 1 See Chap17, pp27,37

2 See Chap17, p37

Charge \$ -1 per deletion of x

– Cost of deleting x is paid by credit.

– Withdraw \$1 credit of some other item

b) 1 See Chap17, p42

2 See Chap 17, pp43~44

7 Let KANPSACK-OPT be the knapsack optimization problem and KANPSACK be the language of the corresponding knapsack decision problem.

a) Define the language KANPSACK. (4%)

b) Prove that if KANPSACK $\in$ P, then KANPSACK-OPT can be solved in polynomial time. (8%)

Solution:

a) $\text{KNAPSACK} = \left\{ \langle W, v_i, w_i, B \rangle \,\middle|\, \begin{array}{l} \text{There exists } x_i = 0 \text{ or } 1 \text{ such that} \\ \sum_{i=1}^{n} w_i x_i \leq W \text{ and } \sum_{i=1}^{n} v_i x_i \geq B \end{array} \right\}$

b) See Chap34, pp.98~102

8 Let HAM-PATH $= \{\langle G \rangle \mid G \text{ has a Hamiltonian path}\}$

Show that HAM-PATH is NPC. (8%)

Solution: See Ex. 34.5-6

9 Prove the following theorem.

THEOREM

If P $\neq$ NP, then for any constant $\rho \geq 1$, $\nexists$ polynomial-time ρ-approximation algorithm for the general TSP. (8%)

Solution: Chap35, pp13~16

10 Consider the following approximation algorithm for the set covering problem.

GREEDY-SET-COVER$(X, \mathcal{F})$ // $X = \bigcup_{s \in F} S$

$U = X$

$\mathcal{C} = \emptyset$

while $U \neq \emptyset$ **do**

  select an $S \in \mathcal{F}$ that maximizes $|S \cap U|$

  $U = U - S$

  $\mathcal{C} = \mathcal{C} \cup \{S\}$

return $\mathcal{C}$

Show that this is a polynomial-time $\rho(n)$-approximation algorithm, where $\rho(n) = H\left(\max\{|S| : S \in \mathcal{F}\}\right)$. (8%)

Hint: You may use the fact that for any $S \in \mathcal{F}$, $\sum_{x \in S} c_x \leq H(|S|)$.

N.B. $H(d) = \sum_{i=1}^{d} 1/i$ is the d^{th} harmonic number.

Solution: Chap35, pp21~23