
Assignment THREE

Problem 1

Consider a 1 by n chessboard. Suppose that we color each square of the chessboard in either blue or red. Let $T(n)$ represent the number of possible colored chessboards in which no two squares that are colored red are adjacent.

- a) Find a recurrence relation for $T(n)$.
- b) Find the closed form of $T(n)$.

Problem 2

Let P_1, P_2, \dots, P_n be n programs to be stored on a disk. Program P_i requires s_i kilobytes of storage, and the capacity of the disk is D kilobytes, where $D < \sum_{i=1}^n s_i$.

- a) We want to maximize the number of programs held on the disk. Prove or give a counter-example: we can use a greedy algorithm that selects programs in order of nondecreasing s_i .
- b) We want to use as much of the capacity of the disk as possible. Prove or give a counter-example: we can use a greedy algorithm that selects programs in order of nonincreasing s_i .

Problem 3

Let $T(h, k)$ be the maximum number of leaves of a tree of height h , where each node has outdegree (number of children) k or less.

- a) Find a recurrence relation for $T(h, k)$.
- b) Solve the recurrence relation.

Problem 4

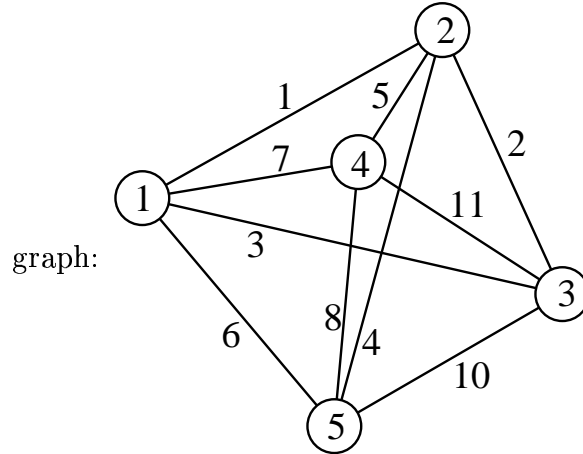
Solve the recurrence relation

$$nT(n) + nT(n-1) - T(n-1) = 2^n$$
$$T(0) = 273$$

Problem 5

Use the dynamic programming algorithm we studied to solve the following instance of TSP. Recall that the algorithm computes values $P(S, k)$, for $S \subseteq \{2, 3, \dots, n\}$ and $k \in S$, that represent the cost of a shortest path that starts at 1, visits all vertices in $S \setminus \{k\}$, and ends at k . For $|S| > 1$, the value of $P(S, k)$ is calculated as $\min\{P(S \setminus \{k\}, m) + w(m, k) \mid m \in S \setminus \{k\}\}$.

$$\text{cost matrix: } \begin{pmatrix} \infty & 1 & 3 & 7 & 6 \\ 1 & \infty & 2 & 5 & 4 \\ 3 & 2 & \infty & 11 & 10 \\ 7 & 5 & 11 & \infty & 8 \\ 6 & 4 & 10 & 8 & \infty \end{pmatrix}$$



Also, define a $Pred(S, k)$, for each $P(S, k)$, to be the value of “ m ” which minimizes the expression in calculating $P(S, k)$. Use these values $Pred(S, k)$ to find the optimal Hamiltonian circuit.

Problem 6

We describe a dynamic programming algorithm for the minimum tardy task weight problem. Suppose we are given lengths ℓ_i , weights w_i , and deadlines d_i , for $1 \leq i \leq n$. We assume that $d_1 \leq \dots \leq d_n$, and $\ell_i \leq d_i$, for $1 \leq i \leq n$.

To apply dynamic programming, we need to generalize the problem slightly, by including an overall deadline d , a time by which all tasks must be finished. This has the effect of lowering any deadline $d_i > d$ to d . For $1 \leq j \leq n$, and $0 \leq t \leq d_n$, define $W(j, t)$ to be the minimum tardy task weight obtainable when considering tasks $1, \dots, j$, with overall deadline t . Given this definition, the solution to the given problem instance is $W(n, d_n)$.

- (a) Explain why W satisfies the following recurrence:

$$W(j, t) = \min\{W(j-1, d_j - \ell_j), w_j + W(j-1, t)\}, \text{ if } d_j < t \text{ and } j > 1$$

$$W(j, t) = \min\{W(j-1, t - \ell_j), w_j + W(j-1, t)\}, \text{ if } l_j \leq t \leq d_j \text{ and } j > 1$$

$$W(j, t) = w_j + W(j-1, t), \text{ if } t < l_j \text{ and } j > 1$$

The starting conditions are:

$$W(1, t) = w_1, \text{ if } l_1 > t$$

$$W(1, t) = 0, \text{ if } l_1 \leq t$$

- (b) Using dynamic programming, solve the following instance of this problem:

lengths 1, 5, 3, 2

deadlines 5, 10, 5, 2 (resp.)

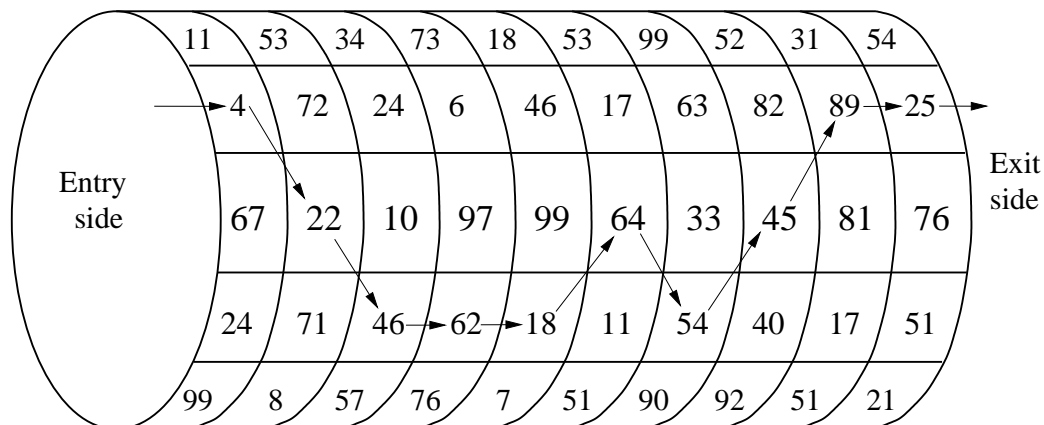
weights 3, 2, 5, 4 (resp.)

Note: Calculate the table of $W(j, t)$'s one row at a time.

- (c) From the table of $W(j, t)$'s, show how to feasibly schedule the optimal subset.

Problem 7

Consider an $n \times n$ array of positive integers (a_{ij}) , $0 \leq i, j < n$, rolled into a cylinder, as illustrated below.



A path is to be threaded from the entry side of the cylinder to the exit side, subject to the restriction that from a given square it is possible to go only to one of the three positions in the next column adjacent to the current position. The path may begin at any position on the entry side and may end at any position on the exit side. The cost of such a path is the sum of the integers

in the squares through which it passes. Thus the cost of the sample path shown above is 429.

- (a) How many distinct paths are there?
- (b) Show how we can use the dynamic programming technique to find the optimal path (cheapest path).
- (c) Find the growth rate of the running time function of your solution to (b).