Final Review Solutions

1. Show that the Hamiltonian Cycle problem for directed graphs is in NP-Complete.

Solution

1) Show you can verify a solution in Polynomial time

2) Show that Ham-Cycle (undirected) which is in NP-Complete can be reduced to Ham-Cycle-Directed.

Transform an undirected graph G into a directed graph H by replacing each edge vw of G by edges in each direction, vw and wv.

Then if there is a Hamiltonian cycle in G, there is one in H. For each edge in the cycle in G, the cycle in H uses the edge from a vw and wv pair that corresponds to the direction the edge was traversed in G.

If there is a Hamiltonian cycle in H, then there is one in G; simply replace either edge vw or wv in the cycle in H by the corresponding undirected edge in G.

Final Review Solutions

2. Macrosoft has a 24-hour-a-day, 7-days-a-week toll free hotline that is being set up to answer questions regarding a new product. The following table summarizes the number of full-time equivalent employees (FTEs) that must be on duty in each time block. Macrosoft may hire both full-time and part-time employees.

- Full-timers work 8-hour shifts with hourly wages of $15.20
- Part-timers work 4-hour shifts with hourly wages of $12.95.
- Employees may start work only at the beginning of 1 of the 6 intervals.
- Part-time employees can only answer 5 calls in the time a full-time employee can answer 6 calls. (i.e., a part-time employee is only 5/6 of a full-time employee.)
- At least two-thirds of the employees working at any one time must be full-time employees.

Formulate an LP to determine how to staff the hotline at minimum cost.

Interval	Time	FTEs
1	0-4	15
2	4-8	10
3	8-12	40
4	12-16	70
5	16-20	40
6	20-0	35

Decision Variables

x_t = # of full-time employees that begin the day at the start of interval t and work for 8 hours

y_t = # of part-time employees that are assigned interval t

$$\min \quad \underbrace{121.6(x_1 + \cdots + x_6)}_{(8 \times 15.20)} + \underbrace{51.8(y_1 + \cdots + y_6)}_{(4 \times 12.95)}$$

s.t.

$$\begin{aligned}
x_1 + x_6 + \tfrac{5}{6}y_1 &\geq 15 \\
x_1 + x_2 + \tfrac{5}{6}y_2 &\geq 10 \\
x_2 + x_3 + \tfrac{5}{6}y_3 &\geq 40 \\
x_3 + x_4 + \tfrac{5}{6}y_4 &\geq 70 \\
x_4 + x_5 + \tfrac{5}{6}y_5 &\geq 40 \\
x_5 + x_6 + \tfrac{5}{6}y_6 &\geq 35
\end{aligned}$$

All shifts must be covered

PT employee is 5/6 FT employee

$$\begin{aligned}
x_1 + x_6 &\geq \tfrac{2}{3}(x_6 + x_1 + y_1) \\
x_1 + x_2 &\geq \tfrac{2}{3}(x_1 + x_2 + y_2) \\
&\vdots \\
x_5 + x_6 &\geq \tfrac{2}{3}(x_5 + x_6 + y_6)
\end{aligned}$$

At least 2/3 workers must be full time

$$x_t \geq 0, \; y_t \geq 0 \quad t = 1,2,\ldots,6$$

Nonnegativity

Final Review Solutions

3. Suppose you have four production plants for making cars. Each works a little differently in terms of labor needed, materials, and energy used per car:

Location	labor hrs	materials	energy
plant1	2	3	15
plant2	3	4	10
plant3	4	5	9
plant4	5	6	7

Suppose we need to produce at least 400 cars at plant 3 according to a labor agreement. We have 3300 hours of labor and 4000 units of material available. We are allowed to use at most 12000 units of energy, and we want to maximize the number of cars produced. Formulate this as a linear programming problem: (1) what are the variables, (2) what is the objective in terms of these variables, and (3) what are the constraints.

Solution:

1. What are the variables? x1,x2,x3,x4, where xi denotes the number of cars at plant i.

2. What is our objective? maximize $x1 + x2 + x3 + x4$.

3. What are the constraints?

$xi \geq 0$ (for all i)

$x3 \geq 400$

$2x1 + 3x2 + 4x3 + 5x4 \leq 3300$

$3x1 + 4x2 + 5x3 + 6x4 \leq 4000$

$15x1 + 10x2 + 9x3 + 7x4 \leq 12000$

Final Review Solutions

4. Prove that 4-SAT is NP-complete.

Instance: A collection of clauses C where each clause contains exactly 4 literals.

Question: Is there a truth assignment to x such that each clasue is satisfied?

Example $\Phi = (x_1 \vee \neg x_1 \vee \neg x_2 \vee \neg x_3) \wedge (x_1 \vee x_3 \vee x_2 \vee x_4)$

Ans: To show that 4-SAT is NP-complete, we prove that 4-SAT is in NP and NP-hard.

First, 4-SAT is in NP, we can write a nondeterministic polynomial-time algorithm which takes a 4-SAT instance and a proposed truth assignment as input. This algorithm evaluates the 4-SAT instance with the truth assignment. If the 4-SAT instance evaluates to true, the algorithm outputs *yes*; otherwise, the algorithm outputs *no*. This runs in polynomial time.

To prove that 4-SAT is NP-hard, we reduce 3-SAT to 4-SAT as follows. Let ϕ denote an instance of 3-SAT. We convert ϕ to a 4-SAT instance ϕ' by turning each clause $(x \vee y \vee z)$ in ϕ to $(x \vee y \vee z \vee h) \wedge (x \vee y \vee z \vee \neg h)$, where h is a new variable. Clearly this is polynomial-time doable.

$\Rightarrow$ If a given clause $(x \vee y \vee z)$ is satisfied by a truth assignment, then $(x \vee y \vee z \vee h) \wedge (x \vee y \vee z \vee \neg h)$ is satisfied by the same truth assignment with h arbitrarily set. Thus if ϕ is satisfiable, ϕ' is satisfiable.

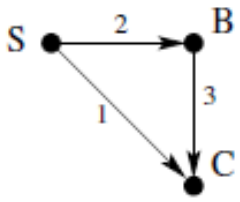
$\Leftarrow$ Suppose ϕ' is satisfied by a truth assignment T. Then $(x \vee y \vee z \vee h) \wedge (x \vee y \vee z \vee \neg h)$ must be true under T. As h and $\neg h$ assume different truth values, $x \vee y \vee z$ must be true under T as well. Thus ϕ is satisfiable.

Final Review Solutions

5. Let G = (V, E) be a DAG, where each edge is annotated with some positive length. Let s be a source vertex in G. Suppose we run Dijkstra's algorithm to compute the distance from s to each vertex v ∈ V, and then order the vertices in increasing order of their distance from s. Are we guaranteed that this is a valid topological sort of G?

No .Justify your answer as follows

Answer:



Comment: Sorting by increasing distance gives S, C, B; but B must precede C in any valid topological sort.

Final Review Solutions

6. Consider a connected weighted directed graph G = (V, E, w). Define the *fatness* of a path P to be the maximum weight of any edge in P. Give an efficient algorithm that, given such a graph and two vertices u,v ∈V, finds the minimum possible fatness of a path from u to v in G.

We can see that that fatness must be the weight of one of the edges, so we sort all edge weights and perform a binary search. To test whether or not a path with a fatness no more than x exists, we perform a breadth-first search that only walks edges with weight less than or equal to x. If we reach v, such a path exists.

If such a path exists, we recurse on the lower part of the range we are searching, to see if a tighter fatness bound also applies. If such a path does not exist, we recurse on the upper part of the range we are searching, to relax that bound. When we find two neighboring values, one of which works and one of which doesn't, we have our answer. This takes $O((V + E) \lg E)$ time.

Final Review Solutions

7. Let T be a complete binary tree with n vertices. Finding a path from the root of T to a given vertex v in T using breadth-first search takes

a. O(n)
b. O(lgn)
c. O(nlgn)
d. O(logn)
e. $O(n^2)$

8. A Hamiltonian path in a graph G=(V,E) is a simple that includes every vertex in V. Design an algorithm to determine if a directed acyclic graph (DAG) G has a Hamiltonian path. Your algorithm should run in O(V+E). Provide a written description of your algorithm including why it works, pseudocode and an explanation of the running time.

Solution:

Compute a topological sort and check if there is an edge between each consecutive pair of vertices in the topological order. If each consecutive pair of vertices are connected, then every vertex in the DAG is connected, which indicates a Hamiltonian path exists. Running time for the topological sort is O(V+E), running time for the next step is O(V) so total running time is O(V+E).

PSEUDOCODE:

```
HAS_HAM_PATH(G)
    1. Call DFS(G) to computer finishing time v.f for each vertex v.
    2. As each vertex is finished, insert it into the front of the list
    3. Iterate each through the lists of vertices in the list
    4. If any pairs of consecutive vertices are not connected RETURN FALSE
    5. After all pairs of vertices are examined RETURN TRUE
```

Final Review Solutions

9. **Variable Definitions**

xt : Number of drivers scheduled at time *t*, *t* = 0, 4, 8, 12, 16, 20

We assume that this problem continues over an indefinite number of days with the same bus requirements and that *xt* is the number used in every day at time *t.*

The objective is to

minimize x0 + x4 + x8 + x12 + x16 + x20

Constraints

We need constraints that assure that the drivers scheduled at the times that cover the requirements of a particular interval sum to the number required. For the interval from time 0 to 4, drivers starting at time 20 of the previous day and time 0 of the current day cover the needs from time 0 to time 4. Then we write

x20 + x0 >= 4

The remaining constraints are:

x0 + x4 >= 8

x4 + x8 >=10

x8 + x12 >= 7

x12 + x16 >= 12

x16 + x20 >= 4

xi >= 0, xi an integer

10. Consider a variant of BIN-PACKING Problem where the bin size is b (not necessarily an integer) and each item has size less than b/3. This version is still NP-complete, though we won't prove this here. Your problem is to give an algorithm that *approximates* the optimal packing into bins, and prove a bound on the quality of your approximation. In particular, prove that if your algorithm uses 3a+1 bins for some integer a, then the optimal algorithm uses at least 2a+1 bins.

You can use the Next-Fit algorithm where we keep one bin open at a time, look at each item in turn, and put it into the current bin if and only if it fits. If it doesn't fit, we open a new bin. Every bin except the last one we use must be filled to more than 2b/3, since we could only close it if a new item, of size less than b/3, failed to fit.

Thus if our algorithm uses 3a+1 bins, the first 3a bins contain a total size of more than 3a(2b/3) = 2ab. If we include the last partially filed las bin, then the 3a+1 bins have a size > 2ab. The optimal algorithm could not possibly fit this much size into 2a bins of size b, so it must use at least 2a+1 bins as desired.

Final Review Solutions