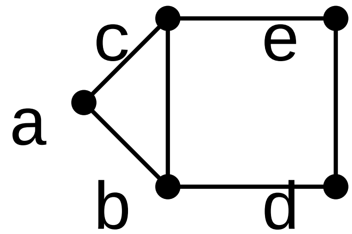
cmput 304 2023 study questions 8 (with hints)



Below is an integer program IP1 for finding a maximum indendent set in this graph.

```
maximize x1 + x2 + x3 + x4 + x5       why?
such that
         x1 + x2 + x3           <= 1  why?
              x2 +      x4      <= 1  why?
                   x3 +      x5 <= 1  why?
                        x4 + x5 <= 1  why?

each of x1,x2,x3,x4,x5 in {0, 1}      why?
```

1. In IP1, what do variables $x1, x2, x3, x4, x5$ represent? In your answer, mention graph nodes a, b, c, d, e.
2. Justify/explain each line of IP1 (answer each "why?").
3. Formulate the dual of IP1.
4. Rephrase the dual of IP1 as a graph problem.
5. IP2 is obtained from IP1 by replacing the constraint `x1+x2+x3` ≤ 1 with the three constraints `x1+x2` ≤ 1, `x1+x3` ≤ 1, `x2+x3` ≤ 1. Repeat questions 1,2,3,4 for IP2.
6. Is $x =$ (1, 0, 0, 1, 0) an optimal solution to IP1? Is x an optimal solution to IP2? Is $x =$ (1, 0, 0, 1, 0) an optimal solution to the LP-relaxation of IP1? If yes, give a dual feasible solution with the same value. Is $x =$ (1/2, 1/2, 1/2, 1/2, 1/2) an optimal solution to the LP-relaxation of IP2? If yes, give a dual feasible solution with the same value.
7. For the above graph, if we want to find a maximum independent set by solving the LP-relaxation of an integer program, should we use IP1 or IP2? Explain briefly.

8. Let G be the weighted graph from the tsp-approx handout:

   `http://webdocs.cs.ualberta.ca/~hayward/304/asn/tsp.pdf`

   Recall that we call an MST *Kruskal* if it is found by some execution of Kruskal's algorithm. Recall that every MST is Kruskal. Give the number of MSTs of G. Explain briefly.

9. a) Give the Hamiltonian cycle found by the 2-approx method, assuming that the starting Eulerian tour is ACEGEFHIHFJFEDBDECA.

   b) Give the Hamiltonian cycle found by the 1.5-approx method.

10. Let G be a weighted complete graph with at least 4 nodes and positive edge weights. Let v be a node in G and let H be the graph obtained by removing v and all edges incident with v. Let h be the length of a min-weight Hamiltonian cycle in H. Let g be the length of a min-weight Hamiltonian cycle in G. Prove/disprove: $h \leq g$.

11. Recall: we say that a weighted graph G satisfies the *triangle inequality* if for each triangle in G with edges a, b, c, $\text{wt}(a) \leq \text{wt}(b) + \text{wt}(c)$. E.g. the triangle with edge weights 1,2,3 satisifies the triangle inequality. E.g. the triangle with edge weights 1,1,3 does not satisfy the triangle inequality. Repeat the previous question, assuming that G satisfies the triangle inequality.

12. Let G be a weighted complete graph with at least 4 nodes that has positive edge weights and satisfies the triangle inequality. Let T be an MST of G. Let Z be the odd-degree nodes in g. Let H be the complete subgraph of G induced by Z. Let g be the min weight of a Hamiltonian cycle of G. Let h be the min weight of a Hamiltonian cycle of H. Prove/disprove: $h \leq g$.

13. For the 2-approx example, prove that each shortcutting step yields a Hamiltonian cycle with weight less than or equal to the original Hamiltonian tour. I have started the answer: you finish it.

```
starting Hamiltonian tour:
A C E D B D E G E F J F H I H F E C A

tour after 1st part of 1st shortcut:
A C E D B   E G E F J F H I H F E C A
by the triangle inequality, this tour's length is not longer
than the length of the preceding tour, because:
  from triangle BDE, edges BD DE have been replaced with edge BE

tour after 2nd part of 1st shortcut:
A C E D B     G E F J F H I H F E C A
by the triangle inequality, this tour's length is not longer
than the length of the preceding tour, because:
...

tour after 2nd shortcut:
...

tour after 3rd shortcut:
...

tour after 4th shortcut (this is the Hamiltonian cycle):
...
```

14. Prove that the greedy set cover algorithm finds a min cover if the number of elements is at most 3.

15. a) Trace the greedy set cover algorithm on the instance below. In case of a tie, always pick the set with smaller index. E.g. if there is a tie between picking S9 and S13, pick S9.

    b) From the text, give an upper bound on the size of the set cover that will be found in a).

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
S 0	*	*	-	*	-	-	-	-	-	-	-	-	-	-	*	-	-	-	*	*
S 1	-	*	-	-	*	-	-	-	*	-	-	-	-	-	-	*	-	-	-	-
S 2	-	-	-	-	-	*	-	-	-	*	*	*	-	*	-	-	-	-	-	*
S 3	-	-	*	-	*	-	*	-	-	-	-	*	*	*	-	-	-	-	-	-
S 4	-	-	*	-	*	-	-	-	-	-	-	-	-	*	-	-	-	-	-	-
S 5	-	-	-	*	-	-	*	-	-	-	-	-	-	-	-	-	*	-	-	*
S 6	-	-	-	-	*	*	-	*	-	-	-	-	-	-	*	-	-	-	-	-
S 7	-	-	-	*	-	-	-	-	-	-	-	-	*	*	-	*	-	-	*	-
S 8	-	*	-	-	-	-	-	-	*	-	-	-	-	-	-	-	*	*	-	-
S 9	-	-	-	-	-	-	*	-	-	-	-	-	-	-	-	-	-	-	-	*
S10	-	-	-	-	-	-	-	*	-	-	-	-	-	-	-	-	*	-	-	*
S11	-	-	-	*	-	*	-	-	-	-	*	-	-	-	-	*	*	-	-	-
S12	-	-	-	*	-	-	-	-	*	-	*	-	-	-	-	-	-	-	*	-
S13	*	-	-	-	-	-	-	-	-	-	-	*	-	-	-	-	-	*	-	-
S14	-	-	-	-	-	*	-	-	-	-	-	-	*	*	-	*	-	-	-	-

16. Consider the greedy set cover algorithm and analysis from §5.4 of Dasgupta et al.. Assume that $n = 100$ and $k = 10$. In a), b), c) do not use the claim in the text: instead, use the reasoning from the proof of the claim.

    a) Prove disprove: $n_1 \leq 90$.

    b) Prove disprove: $n_2 \leq 81$.

    c) Prove disprove: $n_3 \leq 72$.

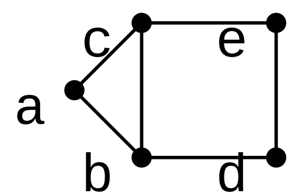
    d) By the claim in the text, give an upper bound on the size of the set cover that will be found for this instance.

hints

1. $x1$ is the weight of node a in the set S that we are looking for.

2. the first line is the objective function $c^t x$ (the sum of all weights), so here c is [1 1 1 1 1]. the next four lines are the constraints that each maximal clique hits at most one node in S, e.g. $x_1 + x_2 + x_3$ is the constraint that clique $\{a, b, c\}$ hits at most one node of S. the last line says that we seek an integer solution, so each xj is either 0 (the corresponding node is not in S) or 1 (the corresponding node is in S).

3.
```
minimize y1 + y2 + y3 + y4
such that
         y1                  >= 1
         y1 + y2             >= 1
         y1      + y3        >= 1
              y2      + y4   >= 1
                   y3 + y4   >= 1
y1, y2, y3, y4 >= 0
```



4. find a min-weight weighted set of maximal cliques that hits each node with total weight at least one

5. 1) $y1$ is weight of maximal clique $\{a, b, c\}$ in the weighted set of cliques T we are looking for.

   2) same answer as for IP1, except replace *maximal clique* with *edge*

```
primal
max x1 + x2 + x3 + x4 + x5 s.t.
    x1 + x2                <= 1
    x1      + x3           <= 1
         x2 + x3           <= 1
         x2      + x4      <= 1
              x3      + x5 <= 1
                   x4 + x5 <= 1
x1, x2, x3, x4, x5 in {0, 1}
```

```
dual
min y1 + y2 + y3 + y4 + y5 + y6 s.t.
    y1 + y2                     >= 1
    y1      + y3 + y4           >= 1
         y2 + y3      + y5      >= 1
                   y4      + y6 >= 1
                        y5 + y6 >= 1

    y1, y2, y3, y4, y5, y6 >= 0
```

   4) find a min-weight weighted set of edges that hits each node with total weight at least one

6. yes, {a,d} is a max size independent set.

   yes.

   yes: (1, 0, 0, 1, 0) is primal feasible, (1, 0, 0, 1) is dual feasible, and both have objective value 2, so they are both optimal.

   yes: (1/2, 1/2, 0, 1/2, 1/2, 1/2) is dual feasible and has objective value 2.5.

7. Use IP1: as we see from the previous question, if we use IP2, the objective value is fractional (2.5) and so does not have an all-integer solution and so will not solve the independent set problem.

   (As mentioned in the lectures, if we use the maximal cliques formulation and the graph is in the class called perfect, which this graph is, then a theorem tells us that every extreme point of the feasible polytope will have all-integer co-ordinates, so if we use the simplex method to solve the LP-relaxation, we are guaranteed to find a solution to the independent set problem.

   Wait a minute, you say: doesn't this imply we can solve independent set in poly time? No, because the number of maximal cliques in a graph with n nodes can be superpoly in n.)

8. Every MST is Kruskal, so to answer this question it suffices to consider every Kruskal MST.

   The min weight of an edge is 2. There are 4 edges with weight 2, and they form no cycles among themselves, so they will all be in each Kruskal MST.

   There are 4 edges with weight 3. Adding them all creates no cycles in the forest-so-far, so they are all in each Kruskal MST.

   At this point our forest has one component and includes all nodes except G, There are 13 edges with weight 4. Of these, only DG, EG, FG, GJ can be added without creating a cycle, and only one can be added, so there are exactly 4 Kruskal MSTs. So there are exactly 4 MSTs.

9. (a) `ACEGEFHIHFJFEDBDECA`

   `ACEG.FHI..J..DB...A`

   (b) ABFGIJ are the odd degree nodes. A min weight matching on these 6 nodes is {AB, FG, IJ}, weight 13.

	B	F	G	I	J
A	5	6	7	9	9
B		6	5	11	8
F			4	4	3
G				7	4
I					4

   Your Hamiltonian cycle will depend which Eulerian tour of MST plus matching you start with. If you start with ACEFHIJFGEDBA then your cycle will be as shown below.

   `ACEFHIJFGEDBA`

   `ACEFHIJ.G.DBA`

10. Let n be the number of nodes in H (so $n+1$ is the number of nodes in G). Assume all edges in H have weight 3 and all edges incident with v have weight 1. Then $h = 3n$ but $g = 3n - 3 + 1 + 1 = 3n - 1$.

11. Let C be a minimum weight Hamiltonian cycle in G. Let x, y be the neighbors of v in C. Then $C' = C - v + (x, y)$ is a Hamiltonian cycle in H, and $\mathrm{wt}(C') \leq \mathrm{wt}(C)$ (why? because we have replaced edges (v, x) and (v, y) with the shortcut edge (x, y), and the graph satisfies the triangle inequality). Let h, g be the respective minimum weights of Hamiltonian cycles in H, G respectively. We have $h \leq \mathrm{wt}(C') \leq \mathrm{wt}(C) = g, \quad$ so $h \leq g$.

12. this was mentioned in the lectures

13. left for you

14. We assume that S covers the universe and that all sets in S are non-empty.

    a) Assume $n = 1$, say $U = \{0\}$. Then S has only one set, namely $S_0 = \{0\}$, the greedy algorithm picks S_0, which covers the universe.

    b) Assume $n = 2$, say $U = \{0, 1\}$. Then the greedy algorithm will take the set $\{0, 1\}$ if this set is in S; if not, then $\{0\}$ and $\{1\}$ must both be in S, and the greedy will pick each. In each case we have a min cover.

    c) Assume $n = 3$, say $U = \{0, 1, 2\}$. This case is left for you :)

15. a) left for you `:)`

    b) in order to answer this question, we need to find the size of a min cover. you can do this by hand, or use program `hard/setcover.py` from the class github repo. $S_0, S_1, S_2, S_3, S_6, S_8$ is a min cover, so the size k of a min cover is 6. Greedy will always find a cover of size at most $k \ln(n) = 6 \ln(15) = 16.2\ldots$, which is not a useful bound: there are only 15 sets, so we already know that greedy will return a cover with size at most 15.

16. a) The average number of elements in a min cover is $n/k = 100/10 = 10$, so there are ten sets with average number of elements 10, so there is at least one set with number of elements at least the average, so there is at least one set with number of elements at least 10. So the first set picked by greedy has at least 10 elements, so the $n_1 \leq (100 - 10) = 90$.

    b) The uncovered n_1 elements are covered by the optimal 10 sets, so some set has at least $n_1/10$ of them, so $n_2 \leq (1 - .1)n_1 \leq (1 - .1)90 = 81$.

    c) $n_3 \leq (1 - .1)n_2 \leq (1 - .1)81 = 72.9$, but n_3 is an integer, so $n_3 \leq 72$.

    d) $10 \times \ln(100) = 46.05\ldots$, so greedy will find a cover with at most 46 sets.