ualberta cmput 304, fall 2014, section a1/ea1 (hayward) final exam 3 hours december 10

## last name

$\qquad$ (no student id\# this page)

## first name

$\qquad$

- print your name above
- do not detach any page from the staple this exam has 7 physical pages
- 6 exam pages, 8 marks/page, 48 marks total
- write legibly: use available space (and back of pages if needed) for your solutions
- to receive part marks for an incorrect answer, you must show your work
- closed book: no notes, no electronic devices: pen/pencil only


## for instructor use only: do not write below

| total marks | question | your marks |
| :--- | :--- | :--- |
| 8 | 1 |  |
| 8 | 2 |  |
| 8 | 3 |  |
| 8 | 5 |  |
| 8 | 6 |  |
| 8 |  |  |
| 48 |  |  |

rough work for problem on next page

```
G = {'S': [['A',1],['C',2]],
    'A': [['S',1],['B',2],['D',5]],
    'B': [['A',2],['D',1],['T',4]],
    'C': [['S',2],['D',3]],
    'D': [['S',5],['A',5],['B',1],['C',3],['T',1]],
    'T': [['B',4],['D', 1]]}
```

    \(\begin{array}{llllll}999 & 999 & 999 & 999 & 0 & 999\end{array}\)
    source $S$ at most 1 edges
A via $S$ now 1

1. The program below finds, for each integer $t$, for each vertex $v$, the (i) Below are the first three lines of
length of a shortest path from source to $v$ with at most $t$ edges.
```
def infinity(): return 999
def sssrp(G,source): #shortest reliable paths
    n,D = len(G), {}
    for v in G: D[v] = infinity()
    D[source] = 0
    show (D)
    for t in range(n):
            print 'source',source,' at most', t+1, 'edges'
            newD = {}
            for v in G: newD[v] = D[v]
            for v in G:
            for wd in G[v]:
                if wd[0]!=source:
                    dfromv = D[v] + wd[1]
                    if dfromv < newD[wd[0]]:
                    newD[wd[0]] = dfromv
                    print ' ',wd[0],'via',v,'now',dfromv
        for v in G: D[v] = newD[v]
        show(D)
def show(D):
    for v in sorted(G): print '%4d' % D[v],
    print ',
G = {'S': [['A',1],['C',2]],
            'A': [['S',1],['B',2],['D',5]],
            'B': [['A',2],['D',1],['T',4]],
            'C': [['S',2],['D',3]],
            'D': [['A',5],['B',1],['C',3],['T',1]],
            'T': [['B',4],['D',1]]}
sssrp(G,'S')
```

output. Show the rest of the output.
$999 \quad 999 \quad 999 \quad 999 \quad 0 \quad 999$ source $S$ at most 1 edges

A via $S$ now 1
(ii) Give the worst-case runtime, as a function of number of vertices $n$ and edges $m$. Justify briefly. Assume that all numbers are small enough to fit in memory.
(iii) Give a loop invariant useful for correctness that holds at the beginning of the loop for $t$ in range ( $n$ ): .
(iv) Briefly, justify correctness.
rough work for problem on next page
2. (i) Sketch the feasible region for this LP. Circle the solution point.

$$
\begin{aligned}
& \max 5 \mathrm{x}+2 \mathrm{y} \text { s.t. } \\
& \mathrm{x}+\mathrm{y}<=3 / 2 \\
& 2 \mathrm{x}+\mathrm{y}<=9 / 4 \\
& \mathrm{x} \quad<=1 \\
& \mathrm{y}<=1 \\
& \mathrm{x}, \mathrm{y}>=0
\end{aligned}
$$

(ii) Give the dual of the above LP.
(iii) Give 4 reasons (at most 10 words each) that linear programming is mentioned in the text.
rough work for problem on next page

$\qquad$
3. Find a max flow from A to P in the network. Each arc is labelled with its capacity. For each arc, the flow can go in either direction (but not both).

(i) Show your flow: on each edge, label the flow (an integer) and direction (an arrow).

(ii) Prove that your flow is maximum.
rough work for problem on next page

| 1 | 2 | -3 |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: |
| 1 | 2 |  |  |  | -6 |
| 1 | -2 |  |  |  | 6 |
| 1 |  | -3 | 4 |  |  |
| 1 |  | -3 |  | 5 |  |
| 1 |  |  | 4 |  | -6 |
| 1 |  |  |  | 5 | 6 |
| -1 | 2 |  |  | 5 |  |
| -1 | -2 |  | 4 |  |  |
| -1 | -2 |  |  | 5 |  |
| -1 |  | -3 | -4 |  |  |
| -1 |  |  | -4 | -5 |  |
| -1 |  |  |  | -5 | 6 |
|  | 2 | 3 | 4 |  |  |
|  | 2 | 3 |  | -5 |  |
|  | 2 | -3 | 4 |  |  |
|  | 2 |  | -4 | -5 |  |
|  | 2 |  | -4 |  | 6 |
|  | 2 |  |  | 5 | 6 |
|  | 2 |  |  | 5 | -6 |
|  | -2 | -3 |  | -5 |  |
|  | -2 |  | 4 | 5 |  |
|  | -2 |  |  | 5 | 6 |
|  | -2 |  |  | -5 | -6 |
|  |  | 3 | -4 | -5 |  |
|  |  | 3 | -4 |  | 6 |
|  |  | -3 |  | 5 | 6 |
|  |  | -3 |  | -5 | -6 |

4. (i) The array below represents a boolean formula in CNF. Each line represents one clause, i.e.
$\left(x_{1} \vee x_{2} \vee \sim x_{3}\right) \wedge\left(x_{1} \vee x_{2} \vee \sim x_{6}\right) \wedge \ldots \wedge\left(\sim x_{3} \vee \sim x_{5} \vee \sim x_{6}\right)$. Trace the start of the DPLL algorithm on the formula: show all steps up to the point at which the first backtrack occurs. Explain each step.

(ii) Is the formula satisfiable? If yes, give a satisfying assignment. If no, justify briefly.
(iii) Give the worst case runtime of DPLL, for a formula with $n$ variables and $m$ clauses. Assume all numbers are small enough to fit into memory. Justify briefly.
rough work for problem on next page
$\qquad$
5. A weighted graph is polar if there is a vertex partition into two parts $P_{1}, P_{2}$, called poles, such that the weight of every cross edge (i.e., with ends in different poles) is greater than the weight of every polar edge (i.e., with ends in the same pole). Such a partition is called polar. E.g., a weighted graph with all weights the same is not polar. E.g., the triangle with edge weights $1,2,2$ is polar. Polar TSP is the following problem.

Instance: a weighted graph $G$ with a polar partition, and an integer $k$.
Query: does $G$ have a Rudratan cycle with weight at most $k$ ?
Either (i) give a polytime algorithm to solve Polar TSP, or (ii) prove that Polar TSP is NP-complete.
rough work for problem on next page

$\qquad$
6. (i) Starting from A, trace Edmond's algorithm on the graph and matching below. As you follow alternating paths, for each unmatched edge that you follow, draw an arrow on the edge showing traversal direction. If a blossom is encountered, circle the nodes, shrink the blossom to form a new graph, and continue. For each graph created by the shrinking of a blossom, list the augmenting path you find in that graph.

(iii) Below, mark the edges of a maximum matching of G . Also, prove that your matching is maximum.


End of exam.

