Amaiya Singhal 2021CS50598

COL754 Assignment 2

5.10 Uniform Labeling Problem: We are given a graph $G = (V, E)$, costs $c_e \geq 0$ for all $e \in E$, and a set of labels L that can be assigned to the vertices of V. There is a nonnegative cost $c_v^i \geq 0$ for assigning label $i \in L$ to vertex $v \in V$, and an edge $e = (u, v)$ incurs cost c_e if u and v are assigned different labels. The goal of the problem is to assign each vertex in V a label to minimize the total cost.

5.10 a)

We show the given integer programming formulation models the uniform labeling problem.

$$\text{minimize } \frac{1}{2}\sum_{e\in E} c_e \sum_{i\in L} z_e^i + \sum_{v\in V, i\in L} c_v^i x_v^i$$

$$\begin{aligned}
\text{subject to} \quad & \sum_{i\in L} x_v^i = 1 && \forall v \in V \\
& z_e^i \geq x_u^i - x_v^i && \forall (u,v) \in E, \forall i \in L \\
& z_e^i \geq x_v^i - x_u^i && \forall (u,v) \in E, \forall i \in L \\
& z_e^i \in \{0,1\} && \forall e \in E, \forall i \in L \\
& x_v^i \in \{0,1\} && \forall v \in V, \forall i \in L
\end{aligned}$$

Suppose there is an instance of the Uniform Labeling Problem with optimal value OPT. We show that there is a feasible solution to the given integer program that has the same cost as OPT. Set

$$z_e^i = \begin{cases} 1 & \text{if exactly one of } u \text{ and } v \text{ is assigned label } i \\ 0 & \text{otherwise} \end{cases}$$

$$x_v^i = \begin{cases} 1 & \text{if vertex } v \text{ is assigned label } i \\ 0 & \text{otherwise} \end{cases}$$

We check that this solution satisfies all the constraints. The last 2 constraints $x_v^i \in \{0,1\}$ and $z_e^i \in \{0,1\}$ are satisfied by above definition of x_v^i and z_e^i. Also $\sum_{i\in L} x_v^i = 1$ because every vertex is assigned exactly 1 label. For the remaining 2 constraints we consider the following cases. For an edge $(u, v) \in E$ and $i \in L$:

Case 1: Both u and v are assigned label i i.e. $x_u^i = 1$ and $x_v^i = 1$
The constraints become $z_e^i \geq 1 - 1 = 0$ which is satisfied by the definition of z_e^i

Case 2: Neither u or v is assigned label i i.e. $x_u^i = 0$ and $x_v^i = 0$
The constraints become $z_e^i \geq 0 - 0 = 0$ which is satisfied by the definition of z_e^i

Case 3: Exactly one of u and v is assigned the label i. Consider $x_u^i = 1$ and $x_v^i = 0$ w.l.o.g. The constraints become $z_e^i \geq 1 - 0 = 1$ and $z_e^i \geq 0 - 1 = -1$ which is equivalent to $z_e^i \geq 1$. Since $z_e^i = 1$ if exactly one of u and v is assigned the label i, the constraint is satisfied.

Hence we have shown that this is a feasible solution to the linear program. Now we show that the cost of this solution is the same as OPT.

$$\begin{aligned}
\text{Cost } &= \frac{1}{2}\sum_{e\in E} c_e \sum_{i\in L} z_e^i + \sum_{v\in V, i\in L} c_v^i x_v^i \\
&= \frac{1}{2}\sum_{e\in E} c_e \Big(\sum_{i\in L, z_e^i=1} 1 + \sum_{i\in L, z_e^i=0} 0 \Big) + \sum_{x_v^{i'}=1, v\in V} c_v^{i'}.1 + \sum_{x_v^i=0, v\in V, i\neq i'} c_v^i.0 \\
&= \frac{1}{2}\sum_{e\in E} c_e \Big(\sum_{i\in L, z_e^i=1} 1 \Big) + \sum_{x_v^{i'}=1, v\in V} c_v^{i'}.1 \\
&= \frac{1}{2} \sum_{e=(u,v)\in E, u,v \text{ assigned different labels}} c_e\big(2\big) + \sum_{x_v^i=1, v\in V, i\in L} c_v^i \\
&= \sum_{e=(u,v)\in E, u,v \text{ assigned different labels}} c_e + \sum_{x_v^i=1, v\in V, i\in L} c_v^i \\
&= \text{Value of the Integer Program}
\end{aligned}$$

Similarly we need to show that given a feasible solution to the given integer program, we can construct a valid solution to the uniform labeling problem with the same cost. We do this by assigning each vertex v the label i such that $x_v = 1$. Since $\sum_{i\in L} x_v^i = 1 \;\; \forall v \in V$, there is exactly 1 such i for each vertex.

$$\begin{aligned}
\text{Cost } &= \sum_{e=(u,v)\in E, u,v \text{ assigned different labels}} c_e + \sum_{x_v^i=1, v\in V, i\in L} c_v^i \\
&= \frac{1}{2} \sum_{e=(u,v)\in E, u,v \text{ assigned different labels}} c_e\big(2\big) + \sum_{x_v^i=1, v\in V, i\in L} c_v^i(1+0) \\
&= \frac{1}{2} \sum_{e=(u,v)\in E, x_u^{i'}=1, x_v^{i''}=1, i'\neq i''} c_e\big(1+1\big) + \sum_{x_v^i=1, v\in V, i\in L} c_v^i x_v^i + \sum_{x_v^i=0, v\in V, i\in L} c_v^i x_v^i \\
&= \frac{1}{2} \sum_{e=(u,v)\in E, x_u^{i'}=1, x_v''^{i}=1, i'\neq i''} c_e\big(z_e^{i'} + z_e^{i''}\big) + \sum_{v\in V, i\in L} c_v^i x_v^i \\
&= \frac{1}{2}\sum_{e\in E} c_e \sum_{i\in L} z_e^i + \sum_{v\in V, i\in L} c_v^i x_v^i \\
&= \text{cost of the Integer Program}
\end{aligned}$$

Hence we have shown that for every optimal solution to an instance of the problem, there is an integer program that is feasible and has the same cost. Similarly every feasible solution to the linear program can be converted into a solution of the uniform labeling problem with the same cost. Therefore the integer program does model the original problem.

5.10 b)

The relaxed linear program is as follows:

$$\text{minimize } \frac{1}{2}\sum_{e\in E} c_e \sum_{i\in L} z_e^i + \sum_{v\in V, i\in L} c_v^i x_v^i$$

$$\begin{aligned}
\text{subject to} \qquad & \sum_{i\in L} x_v^i = 1 && \forall v \in V\\
& z_e^i \geq x_u^i - x_v^i && \forall (u,v)\in E, \forall i \in L\\
& z_e^i \geq x_v^i - x_u^i && \forall (u,v)\in E, \forall i \in L\\
& 0 \leq z_e^i \leq 1 && \forall e \in E, \forall i \in L\\
& 0 \leq x_v^i \leq 1 && \forall v \in V, \forall i \in L
\end{aligned}$$

Coming to the problem,

$$\begin{aligned}
\text{P}(v\in V \text{ gets label } i \in L \text{ in the next phase}) &= \text{P}(\text{Label } i \text{ is chosen and } \alpha \leq x_v^i)\\
&= \text{P}(\text{Label } i \text{ is chosen}).\ \text{P}(\alpha \leq x_v^i \mid i \text{ is chosen})\\
&= \frac{1}{|L|}.\frac{x_v^i}{1}\\
&= \frac{x_v^i}{|L|}
\end{aligned}$$

where P(Label i is chosen) $= \frac{1}{|L|}$ because label i is chosen uniformly at random
Similarly $\text{P}(\alpha \leq x_v^i \mid i \text{ is chosen}) = \frac{x_v^i}{1}$ because α is chosen uniformly at random $\in [0,1]$ and $0 \leq x_v^i \leq 1$

$$\begin{aligned}
\text{P}(v\in V \text{ gets any label } i \in L \text{ in next phase}) &= \sum_{i\in L}\text{P}(\text{Label } i \text{ is chosen and } \alpha \leq x_v^i)\\
&= \sum_{i\in L}\text{P}(\text{Label } i \text{ is chosen}).\ \text{P}(\alpha \leq x_v^i \mid i \text{ is chosen})\\
&= \sum_{i\in L}\frac{1}{|L|}.\frac{x_v^i}{1}\\
&= \frac{1}{|L|}\sum_{i\in L} x_v^i = \frac{1}{|L|}.1 = \frac{1}{L}
\end{aligned}$$

where the last result ($\sum_{i\in L} x_v^i = 1$) follows from the fact the fact that x_v is a solution to the linear program with constraint $\sum_{i\in L} x_v^i = 1$

Let us assume that v is assigned label i in the j^{th} phase then the probability that v is assigned label i by the algorithm is the sum over all j such that v gets label i in phase j and does not get assigned any label before phase j

$$\begin{aligned}
\mathrm{P}(v \text{ gets label } i) &= \sum_{j=1}^{\inf} \mathrm{P}(v \text{ gets label } i \text{ in the } j^{th} \text{ phase}) \\
&= \sum_{j=1}^{\inf} \mathrm{P}(v \text{ is not assigned a label in the first } j-1 \text{ phases}). \\
&\qquad \mathrm{P}(v \text{ gets label } i \text{ in } j^{th} \text{ phase} \mid v \text{ is not assigned in the first } j-1 \text{ phases}) \\
&= \sum_{j=1}^{\inf} \left(1 - \frac{1}{|L|}\right)^{j-1} . \frac{x_v^i}{|L|} \\
&= \frac{x_v^i}{|L|} . \frac{1}{1-\left(1-\frac{1}{|L|}\right)} = \frac{x_v^i}{|L|} . \frac{1}{\frac{1}{|L|}} = x_v^i
\end{aligned}$$

5.10 c)

$$\begin{aligned}
\mathrm{P}(e \in E \text{ is separated by a phase}) &= \mathrm{P}(\text{Only 1 endpoint is assigned a label in this phase}) \\
&= \sum_{i \in L} \frac{1}{L} . \frac{|x_v^i - x_u^i|}{1} \\
&\leq \sum_{i \in L} \frac{1}{L} . z_e^i && [\text{From the constraint } z_e^i \geq |x_u^i - x_v^i|] \\
&\leq \frac{1}{L} \sum_{i \in L} z_e^i \\
&= \frac{1}{L} \sum_{i \in L} z_e^i && [\text{Inequality is tight for optimal solution}]
\end{aligned}$$

The last inequality is tight since if $z_e^i > |x_u^i - x_v^i|$ then we can set $z_e^i = |x_u^i - x_v^i|$ to still get a feasible solution but with lower cost. This contradicts the fact that the solution was optimal. Therefore the inequality must be tight.

5.10 d)

Let P' denote the probability that the endpoints of an edge e receive different labels.

$$\begin{aligned}
\text{P'} &= \text{P'.P(Neither } u, v \text{ get a label)} + \text{P(}u\ (v) \text{ gets a label, } v\ (u) \text{ gets a different label later)}\\
&= \text{P'.P(Neither } u, v \text{ get a label)} + \text{P(}u\ (v) \text{ gets a label).P(}v\ (u) \text{ gets a different label later)}\\
&\leq \text{P'.P(Neither } u, v \text{ get a label)} + \text{P(}u\ (v) \text{ gets a label)}\\
&= \text{P'.P(Neither } u, v \text{ get a label)} + \frac{1}{|L|}\sum_{i\in L} z_e^i\\
&\leq \text{P'.P(}u \text{ does not get assigned a label)} + \frac{1}{|L|}\sum_{i\in L} z_e^i\\
&= \text{P'}\Big(1 - \frac{1}{|L|}\Big) + \frac{1}{|L|}\sum_{i\in L} z_e^i
\end{aligned}$$

From this we get

$$\begin{aligned}
\text{P'} - \text{P'}\Big(1 - \frac{1}{|L|}\Big) &\leq \frac{1}{|L|}\sum_{i\in L} z_e^i\\
\text{P'}.\frac{1}{|L|} &\leq \frac{1}{|L|}\sum_{i\in L} z_e^i\\
\text{P'} &\leq \sum_{i\in L} z_e^i
\end{aligned}$$

Hence we get that the probability that the endpoints of edge e receive different labels is at most $\sum_{i\in L} z_e^i$

5.10 e)

Let OPT_{LP} denote the optimal solution to the relaxed linear program and OPT denote the optimal value of the given Uniform Labeling instance.

Let $X_e\ \ \forall e \in E$ and $Y_v^i\ \ \forall v \in V,\ \forall i \in L$ be random variables defined as

$$X_e = \begin{cases} 1 & \text{if the endpoints of } e \text{ are assigned different labels in the algorithm}\\ 0 & \text{otherwise}\end{cases}$$

$$Y_v^i = \begin{cases} 1 & \text{if vertex } v \text{ is assigned label } i\\ 0 & \text{otherwise}\end{cases}$$

Using these random variables, the total cost of the solution becomes From this we get

$$\begin{aligned}
\text{Cost} &= \sum_{e\in E} c_e.X_e + \sum_{v\in V, i\in L} c_v^i.Y_v^i \\
E[\text{Cost}] &= E\Big[\sum_{e\in E} c_e.X_e + \sum_{v\in V, i\in L} c_v^i.Y_v^i\Big] \\
E[\text{Cost}] &= \sum_{e\in E} c_e.E\big[X_e\big] + \sum_{v\in V, i\in L} c_v^i.E\big[Y_v^i\big] \\
&\le \sum_{e\in E} c_e.\sum_{i\in L} z_e^i + \sum_{v\in V, i\in L} c_v^i.x_v^i \\
&\le \sum_{e\in E} c_e.\sum_{i\in L} z_e^i + 2.\sum_{v\in V, i\in L} c_v^i.x_v^i \\
&= 2.\left(\frac{1}{2}.\sum_{e\in E} c_e.\sum_{i\in L} z_e^i + \sum_{v\in V, i\in L} c_v^i.x_v^i\right) \\
&= 2.OPT_{LP} \le 2.OPT
\end{aligned}$$

Therefore this algorithm is a 2-approximation for the Uniform Labeling Problem.

6.2 Max 2-SAT Problem: In the maximum-2-satisfiability problem (MAX-2-SAT), the input is a formula in conjunctive normal form with two literals per clause, and the task is to determine the maximum number of clauses that can be simultaneously satisfied by an assignment.

$$(x_1 \vee x_2) \wedge (x_3 \vee x_2) \wedge \cdots$$

6.2 a)

We can express the MAX 2SAT problem as a "integer quadratic program" as follows:

$y_i \in \{-1, 1\}$ for $i = 0, \ldots, n$ (where n is the number of variables)

$y_0 = y_i$ if and only if x_i is true.

For the objective, we want each clause C to have a value $v(C)$ that is 1 if and only if the clause is satisfied, and 0 otherwise:

$$\text{For clauses with 1 variable} : v(x_i) = \frac{1 + y_i y_0}{2}$$

$$\text{For clauses with 2 variables} : v(x_i \vee x_j) = 1 - v(x_i)v(x_j)$$

$$\begin{aligned}
v(x_i \vee x_j) = 1 - v(x_i)v(x_j) &= 1 - \left(\frac{1 - y_i y_0}{2}\right) \cdot \left(\frac{1 - y_j y_0}{2}\right) \\
&= 1 - \frac{1 - y_i y_0 - y_j y_0 + y_i y_j y_0^2}{4} = \frac{4 - (1 - y_i y_0 - y_j y_0 + y_i y_j)}{4} \\
&= \frac{3 + y_i y_0 + y_j y_0 - y_i y_j}{4} = \frac{1 + y_i y_0}{4} + \frac{1 + y_j y_0}{4} + \frac{1 - y_i y_j}{4}
\end{aligned}$$

Similarly for other types of clauses with 2 variables,

$$\begin{aligned}
v(\neg x_i \vee x_j) &= \frac{1+y_iy_0}{4} + \frac{1-y_jy_0}{4} + \frac{1+y_iy_j}{4} \\
v(x_i \vee \neg x_j) &= \frac{1-y_iy_0}{4} + \frac{1+y_jy_0}{4} + \frac{1+y_iy_j}{4} \\
v(\neg x_i \vee \neg x_j) &= \frac{1+y_iy_0}{4} + \frac{1+y_jy_0}{4} + \frac{1-y_iy_j}{4}
\end{aligned}$$

The objective for this problem is

$$\sum_{C \equiv x_i} v(x_i) + \sum_{C \equiv x_i \vee x_j} v(x_i \vee x_j) + \sum_{C \equiv x_i \vee \neg x_j} v(x_i \vee \neg x_j) + \sum_{C \equiv x_i \vee \neg x_j} v(\neg x_i \vee x_j) + \sum_{C \equiv \neg x_i \vee \neg x_j} v(\neg x_i \vee \neg x_j)$$

Using the terms calculated above we can simplify the objective and write the integer quadratic program as

$$\max \sum_{0 \le i,j \le n} \beta_{ij}(1+y_iy_j) + \gamma_{ij}(1-y_iy_j) \quad \text{where } \beta_{i,j} \text{ and } \gamma_{i,j} \text{ are constants}$$

$$\text{subject to} \quad y_i \in \{1,-1\}, \quad i = 0,1,2\dots n$$

6.2 b)

To obtain a .878 approximation-algorithm for the MAX 2SAT Problem, we proceed similar to what we did for the MAX CUT Problem in the class. We construct a vector programming relaxation to the earlier integer quadratic program as follows:

$$\max \sum_{0 \le i,j \le n} \beta_{ij}(1+v_i.v_j) + \gamma_{ij}(1-v_i.v_j) \quad \text{where } \beta_{i,j} \text{ and } \gamma_{i,j} \text{ are constants}$$

$$\text{subject to} \quad v_i.v_i = 1 \text{ and } \mathbb{R}^{n+1}, \ i = 0,1,2\dots n$$

Algorithm 1 Vector Programming Rounding for MAX 2SAT

```
$v_0, v_1 \dots v_n \leftarrow$ Solution to the Vector Program
$r \leftarrow$ Randomly chosen hyperplane
$y_i \leftarrow 1$ if $v_i.r \ge 0$, $y_i \leftarrow 0$ otherwise
```

Claim 1: Algorithm 1 is a polynomial time algorithm.

Proof. There are only as many constraints and variables as the number of unique variables which is polynomial in number. Hence, this vector program can be solved in polynomial time. Choosing a random hyperplane and assigning values to y_i can also be done trivially in polynomial time. □

Claim 2: Algorithm 1 is a .878-approximation algorithm for the MAX 2SAT problem.

Proof. Let X_{ij} be a random variable defined as

$$X_{ij} = \begin{cases} 1 & \text{if } y_i = y_j \\ -1 & \text{otherwise} \end{cases}$$

Let W be the random variable representing the total cost of the solution

$$\begin{aligned}
E[W] &= \sum_{0 \le i,j \le n} \beta_{ij}(1+1).\mathrm{P}(y_i = y_j) + \gamma_{ij}(1-(-1)).\mathrm{P}(y_i \neq y_j) \\
&= 2\left[\sum_{0 \le i,j \le n} \beta_{ij}.\left(1 - \frac{\theta_{ij}}{\pi}\right) + \gamma_{ij}.\frac{\theta_{ij}}{\pi}\right] = 2\left[\sum_{0 \le i,j \le n} \beta_{ij}.\left(\frac{\pi - \theta_{ij}}{\pi}\right) + \gamma_{ij}.\frac{\theta_{ij}}{\pi}\right] \\
&\ge 2\left[\sum_{0 \le i,j \le n} \beta_{ij}.\alpha\left(\frac{1-\cos(\pi - \theta_{ij})}{2}\right) + \gamma_{ij}.\alpha\left(\frac{1-\cos(\theta_{ij})}{2}\right)\right] \text{ (similar to MAX CUT)} \\
&= \sum_{0 \le i,j \le n} \beta_{ij}.\alpha.\big(1-\cos(\pi-\theta_{ij})\big) + \gamma_{ij}.\alpha.\big(1-\cos(\theta_{ij})\big) \\
&= \sum_{0 \le i,j \le n} \beta_{ij}.\alpha.\big(1+\cos(\theta_{ij})\big) + \gamma_{ij}.\alpha.\big(1-\cos(\theta_{ij})\big) \\
&= \sum_{0 \le i,j \le n} \beta_{ij}.\alpha.\big(1+v_i.v_j\big) + \gamma_{ij}.\alpha.\big(1-v_i.v_j\big) \\
&= \alpha.\left(\sum_{0 \le i,j \le n} \beta_{ij}.\big(1+v_i.v_j\big) + \gamma_{ij}.\big(1-v_i.v_j\big)\right) = \alpha.OPT_{VP}
\end{aligned}$$

Hence, since the value of α is at least .878, we get

$$E[W] \ge .878\ OPT_{VP} \ge .878\ OPT$$

which proves our claim.

□

7.5 Minimum-Cost Branching Problem: We are given a directed graph $G = (V, A)$, a root vertex $r \in V$, and weights $w_{ij} \ge 0$ for all $(i,j) \in A$. The goal of the problem is to find a minimum-cost set of arcs $F \subseteq A$ such that for every $v \in V$, there is exactly 1 directed path in F from r to v.

We can formulate the problem as an integer program as follows:

$$\text{minimize} \sum_{(i,j) \in A} w_{ij} x_{ij}$$

$$\text{subject to} \quad \sum_{(i,j)\in\delta^-(S)} x_{ij} \geq 1 \qquad \forall S \subseteq V - \{r\}$$

$$\sum_{(i,j)\in\delta^-(v)} x_{ij} = 1 \qquad \forall v \in V - \{r\}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in A$$

where $\delta^-(S)$ is the set of incoming edges to the set S. Formally it can be defined as $\delta^-(S) = \{(i,j) \in A \mid i \notin S, j \in S\}$. The first constraint ensures that there is atleast 1 path from r to any vertex $v \in V$ and the second constraint ensures that there is only 1 incoming edge to each vertex v which ensures there can be exactly 1 path from r to v. Hence, this integer program models the Minimum Cost Branching Problem.

We relax the integer program into a linear program as follows:

$$\text{minimize} \sum_{(i,j)\in A} w_{ij}x_{ij}$$

$$\text{subject to} \quad \sum_{(i,j)\in\delta^-(S)} x_{ij} \geq 1 \qquad \forall S \subseteq V - \{r\}$$

$$x_{ij} \geq 0 \qquad \forall (i,j) \in A$$

The above linear program is a relaxation of the integer program because any solution to the integer program still satisfies the relaxed constraints (the second constraint has been removed and the third constraint $x_{ij} \in \{0,1\}$ is satisfied by $x_{ij} \geq 0$).

The dual of the linear program is

$$\text{maximize} \sum_{S\subseteq V-r} y_S$$

$$\text{subject to} \quad \sum_{S:(i,j)\in\delta^-(S)} y_S \leq w_{ij} \qquad \forall (i,j) \in A$$

$$y_S \geq 0 \qquad \forall S \subseteq V - \{r\}$$

Claim: We can find a set S which satisfies the 2 conditions in every iteration of the algorithm.

Proof. Such a set S exists because we can contract all strongly connected components (in F) and the resulting DAG would have such a super vertex which has no incoming edges (because it is a

Algorithm 2 Primal Dual Algorithm for Minimum Cost Branching

```
$F \leftarrow \phi$
$\forall S \subseteq V - \{r\}, y_S \leftarrow 0$
while $\exists v \in V$ : There is no path from $r$ to $v$ in $F$ do
    Choose S such that
    (i) $F \cap G(S)$ is strongly connected
    (ii) $F \cap \delta^-(S) = \phi$
    do
        $y_S \leftarrow y_S + \Delta$
    while $\nexists\ (i,j) \in A,\ \sum_{S:(i,j)\in\delta^-(S)} y_S = w_{ij}$
    $F \leftarrow F \cup \{(i,j)\}$
end while
$F' \leftarrow$ Reverse Delete($F$)
return $F'$
```

DAG). Then the set of vertices corresponding to this super vertex forms S, since it has no incoming edges (in F). Note that there would always be a super vertex S' which has no incoming edges and $r \notin S'$ because if the only super vertex with no incoming edges had r in it then the algorithm would have stopped earlier. This is because in that case r would already have a directed path to every vertex $v \in V$. □

Claim: F' is a directed spanning tree of G

Proof. We claim that F' has exactly $|V| - 1$ edges with exactly 1 edge incoming on every vertex $v \in V - \{r\}$. Suppose not, then there must be a vertex v with 2 incoming edges (i,j) and (i',j') where (i',j') is added to the solution in a later step than (i,j). Then we must have removed (i',j') in the Reverse Delete phase because any vertex that is reached from r through (i',j') can be reached through (i,j) as well. □

Claim: F' is the optimal solution.

Proof. We prove the optimality of F' using standard primal-dual analysis. For every edge $(i,j) \in F'$ we have $w_{ij} = \sum_{S:(i,j)\in\delta^-(S)} y_S$, and we know that

$$\sum_{(i,j)\in F'} w_{ij} = \sum_{(i,j)\in F'} \sum_{S:(i,j)\in\delta^-(S)} y_S = \sum_{S\subseteq V-\{r\}} |F' \cap \delta^-(S)| y_S$$

If we can now show that whenever we have $y_S > 0$, we have $|F' \cap \delta^-(S)| = 1$, then we will show that

$$\sum_{(i,j)\in F'} w_{ij} = \sum_{S\subseteq V-\{r\}} y_S \leq OPT$$

□

by weak duality. But of course since F' is Directed Spanning Tree of cost no less than OPT, it must have cost exactly OPT.

We now show that if $y_S > 0$, then $|F' \cap \delta^-(S)| = 1$. Suppose otherwise, and $|F' \cap \delta^-(S)| > 1$.

When S was chosen, there were no edges in $\delta^{-}(S)$ prior to adding the edge say (i', j') (which is why S was chosen). This means all the other edges in $|F' \cap \delta^{-}(S)|$ must have been added after (i', j'). Now, we claim that these edges (say (i'', j'')) should have been deleted in the reverse delete procedure. We know F was strongly connected within S, so we can take the edge (i', j') and traverse to the vertex j''. We can do this since F is strongly connected within S and then we can continue with the path that we were taking from j''. Hence we have shown that $|F' \cap \delta^{-}(S)| = 1$.

7.7 K-Path Partition Problem We are given a complete, undirected graph $G = (V, E)$ with edge costs $c_e \geq 0$ that obey the triangle inequality (that is $c_{(u,v)} \leq c_{(u,w)} + c_{(w,v)}$ for all $u, v, w \in V$), and a parameter k such that $|V|$ is a multiple of k. The goal is to find a minimum-cost collection of paths of k vertices each such that each vertex is on exactly one path.

7.7 a)

Consider an instance $G = (V, E)$ of the K-Path Partition Problem. We apply the α approximation algorithm for the second problem to get a set of trees T (each of size 0(mod k)) such that the cost of the solution is $\leq \alpha.OPT_2$ where OPT_2 is the optimal solution to the 2nd problem.

Suppose OPT_1 is the optimal solution of the K-path Partition Problem on the graph G. Since any solution to this problem is also a valid solution to the 2nd problem (and not vice-versa), we have $OPT_2 \leq OPT_1$.

For each tree (say T_i) obtained from the α-approximation algorithm of the 2nd problem, duplicate each edge to get a graph G_i of cost $= 2.\text{cost}(T_i)$. Similar to the approach used for Steiner Tree in class, we construct an Eulerian Tour E_i in G_i which has cost $= 2.\text{cost}(T_i)$ as well. Since c_e is metric, we can construct a cycle among these vertices with cost $\leq \text{cost}(E_i) = 2.\text{cost}(T_i)$.

We now need to convert these cycles of 0(mod k) vertices into k-paths. There are exactly k unique ways to do this (by removing every kth edge starting from some arbitrary vertex on the cycle). We choose the set of edges which has the maximum combined cost of edges. Since we choose the maximum cost set of such edges, say the cost of these edges is C then we can say $C \geq \frac{2.\text{cost}(T_i)}{k}$. Hence the total cost of the solution becomes $\sum_i (2.\text{cost}(T_i) - \frac{2.\text{cost}(T_i)}{K}) = 2(1 - \frac{1}{k}) \sum_i \text{cost}(T_i)$. From this we get

$$\begin{aligned}
\text{cost} &= 2\left(1 - \frac{1}{k}\right) \sum_i \text{cost}(T_i) \\
\text{cost} &\leq 2\left(1 - \frac{1}{k}\right) \alpha.OPT_2 \\
&= 2\alpha.\left(1 - \frac{1}{k}\right) OPT_2 \\
&\leq 2\alpha.\left(1 - \frac{1}{k}\right) OPT_1
\end{aligned}$$

Hence this is an $2.\alpha.(1 - \frac{1}{k})$-approximation algorithm for the K-Path Partition Problem.

7.7 b)

We construct a requirement function $r : 2^V \to \{0,1\}$ as follows:

$$\forall S \subseteq V,\ r(S) = \begin{cases} 1 & \text{if } |S| \neq 0(\text{mod k}) \\ 0 & \text{otherwise} \end{cases}$$

Using this function we write a linear program for the problem as:

$$\text{minimize} \sum_{e \in E} x_e c_e$$

$$\text{subject to} \qquad \sum_{e \in \delta(S)} x_e \geq r(S) \qquad \forall S \subseteq V$$

$$x_e \in \{0,1\} \qquad \forall e \in E$$

We show that this integer program models the given problem. Firstly we show that we can convert the optimal solution O to the problem into a feasible integer program with the same value. Let

$$\forall x_e = \begin{cases} 1 & e \in \text{O} \\ 0 & \text{otherwise} \end{cases}$$

This formulation satisfies the constraints. Suppose $\exists S \subseteq V$ such that $\sum_{e \in \delta(S)} x_e < r(S)$ where $r(S) = 1$. Then this means there is a subset of vertices of size $\neq 0$ (mod k) which has no outgoing edges. Then there must be a subset of these vertices which is connected and has size = 0 (mod k). But this violates the fact that O was a solution to this problem. Hence this is a feasible solution. Also the cost of the integer program solution is trivially the same as that of O because $x_e = 1$ iff $e \in O$.

Now we show that given any feasible solution to the integer program, we can construct a solution to the given instance of the problem with the same cost. Let the set of edges in the solution be $\{e \in E \mid x_e = 1\}$. The cost of this subgraph is the same as that of the integer program solution by construction of x_e. We also claim that this is a valid partition of the graph G into trees of size 0 (mod k). Suppose there is a maximal connected component of size $\neq 0$ (mod k). Consider the set (say S) containing the vertices in the component. Since this set is a maximal connected component, there are no edges coming out of it, hence $\sum_{e \in \delta(S)} x_e = 0$ which violates the constraint that $\sum_{e \in \delta(S)} x_e \geq 1$ since $r(S) = 1$. This violates the fact that this was a feasible solution to the linear program.

Hence, we have shown that this integer program models the given problem.

We show that this function r is proper

1. $r(V) = 0$ because $|V| = 0$ (mod k)
2. $r(V) = r(\bar{V})$ because $|V| = 0$ (mod k) iff $|\bar{V}| = 0$ (mod k)

3. If A and B are 2 disjoint subsets of V and $r(A \cup B) = 1$, then $r(A) = 1$ or $r(B) = 1$. This is also true because $r(A \cup B) = 1$ iff $|A \cup B| \neq 0$ (mod k), this means that atleast one of A or B must be of size 0 (mod k) otherwise $A \cup B$ must have been of size 0 (mod k).

Similar to what was done in class for the Steiner Tree problem, since r is proper, this algorithm is a 2-approximation algorithm for the problem.

7.7 c)

Similar to what was done in 7.7 a), we show a $4.\alpha.(1-\frac{1}{k})$-approximation algorithm for the k-cycles partition problem. From what we had done earlier, we had constructed a solution to the k-path partition problem such that the cost $\leq 2.\alpha.(1-\frac{1}{k})OPT_{\text{Path}}$. We add an edge to each of these k-edge paths to complete the cycle. This increases the cost of each path by atmost a factor of 2 (because we have a metric). Hence the cost of this solution becomes $\leq 4.\alpha.(1-\frac{1}{k}).OPT_{\text{Path}}$. Since $OPT_{\text{Path}} \leq OPT_{\text{Cycle}}$, we directly get that the cost of the solution obtained is atmost $4.\alpha.(1-\frac{1}{k}).OPT_{\text{Cycle}}$

8.11 Capacitated Dial-a-ride Problem We are given a metric (V, d), a vehicle of capacity C, a starting point $r \in V$, and k source-sink pairs $s_i - t_i$ for $i = 1, \ldots k$, where $s_i, t_i \in V$. At each source s_i there is an item that must be delivered to the sink t_i by the vehicle. The vehicle can carry at most C items at a time. The goal is to find the shortest possible tour for the vehicle that starts at r, delivers each item from its source to its destination without exceeding the vehicle capacity, then returns to r; note that such a tour may visit a node of V multiple times. We assume that the vehicle is allowed to temporarily leave item at any node in V.

8.11 a)

Since we are considering the case of (V, d) being a tree metric (V, T), each item i has a unique path $P(i)$ from its source to destination. For an edge $e \in E$, define $flow_u(e)$ to be the total number of paths $P(i)$ that pass upwards through the edge e i.e. they go through the edge e when going from the source to the destination. Similarly $flow_d(e)$ is defined as the total number of paths that pass through it from the destination to the source.

Since we can carry atmost C items, each edge e will be traversed at least $2.\max\big(\lceil\frac{flow_u(e)}{C}\rceil, \lceil\frac{flow_d(e)}{C}\rceil\big)$ times.

Algorithm for the capacitated Dial-a-ride problem: We perform 2 depth-first traversals of the tree T in the following way. In the first dfs run, after the subtree rooted at an edge e is explored, we recursively ensure that all $flow_u(e)$ items that need to move up the edge e are collected at the lower end of point e. We then carry these items (at most C at once) across the edge e. Since we can drop off items at places, we are fine in doing this. We traverse each edge $2.\max(1, \lceil\frac{flow_u(e)}{C}\rceil)$ times.

Similarly in the second dfs run, we repeat the exact same procedure but in the opposite direction. We ensure that all $flow_d(e)$ items that need to move down the edge e are collected at the upper end of point e. In this run, we traverse each edge $2.\max(1, \lceil\frac{flow_d(e)}{C}\rceil)$ times.

Hence the total number of times an edge e is traversed in the algorithm is $2.\left(\lceil\frac{flow_d(e)}{C}\rceil + \lceil\frac{flow_u(e)}{C}\rceil\right)$ times which is $\leq 4.\left(\lceil\frac{flow_d(e)}{C}\rceil, \lceil\frac{flow_u(e)}{C}\rceil\right)$ times.

$$\begin{aligned}\text{Number of Passes} &\leq 4.\max\left(\lceil\frac{flow_d(e)}{C}\rceil, \lceil\frac{flow_u(e)}{C}\rceil\right)\\ &= 2.2.\max\left(\lceil\frac{flow_d(e)}{C}\rceil, \lceil\frac{flow_u(e)}{C}\rceil\right)\\ &\leq 2.(\text{Number of Passes in optimal})\end{aligned}$$

Hence, this is a 2-approximation for the Capacitated Dial a Ride Problem.

8.11 b)

To give a randomized $O(\log|V|)$-approximation algorithm for the capacitated dial-a-ride problem in the general case we convert the graph $G = (V, d)$ into a tree metric T using the probabilistic approach studied in class.

The algorithm described in part a) gives a 2-approximation algorithm for the case when (V, d) is a tree metric which gives us that the cost of our solution is $\leq 2.OPT_{\text{Tree}}$. Now applying the result proved in class that by converting a metric into a tree metric we increase the costs at most by a factor of $\log(|V|)$, we can write

$$\begin{aligned}\text{cost} &\leq 2.OPT_{\text{Tree}}\\ &\leq 2.\log(|V|).OPT\end{aligned}$$

Hence we have shown a randomized $O(\log|V|)$-approximation algorithm for the capacitated dial-a-ride problem in the general case.