We are given a graph $G = (V, E)$ a set of facilities $F \subseteq V$ and a set of demand points $D \subseteq V$. We are also given a set of resource types R and each demand point d_i requires a resource of type $r_{d_i} \in R$. We wish to open some of the facilities and connect the demands to open facilities via Steiner Trees that minimizes the total cost, where we have to pay a fixed cost of f_i for opening the i-th facility and a linear cost of $n_e c_e$ for the edge e if it ships n_e different resource types. We call c_e the length of the edge e.

We represent this problem as the following integer program. We use the variables y_i to represent whether the facility i is open, and $x_{e,r}$ to denote whether the edge e ships the resource type r or no. We use $\delta(S)$ to denote the set of edges whose one end is in S and the other in $V - S$.

$$minimize \sum_i f_i y_i + \sum_{e,r} c_e x_{e,r} \tag{1}$$

$$\sum_{e \in \delta(S)} x_{e,r} + \sum_{i \in S} y_i \geq 1 \quad \forall r, S \quad s.t. \quad \exists d_j \in D, r_{d_j} = r \tag{2}$$

$$y_i, x_{e,r} \in \{0, 1\}. \tag{3}$$

The first constraint in the above IP states that for all sets S that contain a demand point requiring a resource type r, either there is an open facility in S or there exists an edge crossing S which ships the resource type r.

We obtain a linear relaxation to the integer program by replacing the last constraint by

$$y_i, x_{e,r} \geq 0. \tag{4}$$

For the dual of this linear program, we create a variable $\gamma_{S,r}$ for each set S and resource type r such that again there exists a demand point in S requesting r. Then the dual becomes:

$$maximize \sum_{S,r} \gamma_{S,r} \tag{5}$$

$$\sum_{S: e \in \delta(S)} \gamma_{S,r} \leq c_e \quad \forall e, r \tag{6}$$

$$\sum_{S,r: i \in S} \gamma_{S,r} \leq f_i \quad \forall i \tag{7}$$

$$\gamma_{S,r} \geq 0. \tag{8}$$

These algorithms will follow the primal-dual scheme with dual accent and will consist of an assignment phase, an edge addition and/or cleanup phase and a conflict resolution phase.

1 First Solution

The Assignment Phase

The algorithm will in many ways resemble the standard Steiner Forest algorithm run for many resources independently. Throughout the algorithm we will say an edge e is *tight* for a resource type r if the dual constraint (6) is tight for the pair (e, r). We will say a set S is *active for* r if it contains a spanning tree (which we call the *derived tree* of S) of tight edges supplying r. We will interchangeably refer to active sets as *components*. For each resource type r we start uniformly at unit rate raising the dual variables $\gamma_{S,r}$ for all active sets, which initially contain singleton demand points requesting r. Whenever the dual constraint (6) becomes tight for an edge e and a resource type r, we say that the edge e was *bought* for shipping r. We also stop raising all the dual variables $\gamma_{S_i,r}$ that contributed to making the inequality tight and instead start raising $\gamma_{S,r}$ at the same uniform rate, where $S = \bigcup S_i$ (notice that S is now active since the addition of the edge e created a spanning tree amongst all the nodes that are now in S). Whenever the dual constraint (7) becomes tight for a facility i we declare the facility *temporarily open* and freeze all the dual variables that contributed to the left side of the newly tight inequality. We also declare the facility i to be the *connection witness* for all the demand points of component S requesting the resource r corresponding to the newly frozen dual variables $\gamma_{S,r}$.

We continue this process until all the dual variables are frozen and end the assignment phase.

The Edge Cleanup Phase

This phase completes the pruning of spanning trees. If an edge e was shipping a resource r but retracting that shipment would still allow all the demands to reach their connection witnesses, then we prune r from the set of resources shipped by e in the solution.

The Conflict Resolution Phase

We define two facilities to be at *conflict at time* t if they are within an active set S at time t whose corresponding dual variable is not frozen and the length of the path in the derived tree of S (which we will call the *conflict path*) is no more than t. We define two facilities to be at *conflict* if there exists a time t where they are in conflict.

We wish to open only one of several conflicting facilities. Thus we define a graph G_F on facilities F where there is an edge between two facilities f_i and f_j if they are in conflict. We find a maximal independent set F' in G_F greedily (facilities are considered in the order they were declared temporarily open) and declare these facilities *open*. For a facility i that was not opened, we consider the components (S_k, r_k) for whom that facility was a connection witness. From the conflict resolution step it follows that there exists an open facility that is in conflict with i via a conflicting path π, in which case we purchase the path π for all the resource types r_i. We refer to this last step as *rerouting*.

1.1 Analysis

We break up the analysis into many steps, which are described below. We note that the cost of the solution is the sum of the costs of opening the facilities and the cost of buying the edges, i.e. $\mathcal{C} = \mathcal{F} + \mathcal{E}$.

- Feasibility of the Solution: We need first to show that each demand node is connected to an open facility via a path supplying the resource type r that this node demands. For each demand point j, if the connection witness i for j is open then we already have a path purchased from j to i, because j was in some active component S that included i and each such set contains a spanning tree. On the other hand if the connection witness of j was not opened then we purchased a path supplying r from i to an open facility i'. Since j was connected with i then there is a path from j to i' shipping resource r. Finally the cleanup step maintains feasibility of the solution.

- Edge Costs ($\mathcal{E}$): We break down the edge cost into two parts - the spanning tree cost $\mathcal{T}$ and the rerouting cost $\mathcal{R}$.

  At the end of the edge cleanup phase we observe that thus far we have simply run the well-known Steiner Forest algorithm for each resource separately, therefore the cost of the edges bought is upperbounded by twice the value of the dual solution: $\mathcal{T} \leq 2\sum_{S,r}\gamma_{S,r}$

  On the other hand, let as assume that due to the closing of facility i we have to purchase a path to a facility i' for a resource type r with a corresponding active set S^*. From our greedy construction of the independent set we know that facility i' was declared temporarily open at an earlier time than facility i ($t_{i'} \leq t_i$). Let π be the conflict path between i and i' and let t be the earliest time when the conflict was observed.

  Notice that $t \leq t_i$, since otherwise at time earlier than t both of the facilities i and i' would have been declared temporarily open, therefore freezing the growth of the active set that included them both. In the latter case either the conflict would be observed earlier (at most at time t_i) or would never be observed.

Let π be the conflict path that we had to purchase for resource type r. From our definition of conflict it follows that $|\pi| \leq t$.

Finally at time t_i

$$\sum_{S,r:S\subseteq S^*} \gamma_{S,r} \geq t_i \geq t \tag{9}$$

since at least one of the dual variable in the left hand side of the above inequality was increasing at the uniform rate any time before t_i. Therefore the dual accumulated in S^* and its subsets could pay for the purchase of the path π for resource type r.

Applying this analysis for all the rerouted resources we obtain that the total rerouting cost is no more than the value of the dual solution: $\mathcal{R} \leq \sum_{S,r} \gamma_{S,r}$.

Therefore, $\mathcal{E} \leq 3 \sum_{S,r} \gamma_{S,r}$.

- Facility Costs ($\mathcal{F}$): We first observe that each facility was totally paid for when it was declared temporarily open. Therefore all the facilities that were opened will be totally paid for. Of course our conflict resolution allows us to have the same dual variable pay for multiple facilities.

  We claim that the total contributions that the dual variables made towards the facilities is bounded by five times the value of the dual. The claim follows from the following observation.

  Let a dual variable $\gamma_{S^*,r}$ contribute towards open facilities $f_1, \ldots, f_k$ where S^* is the (maximal) component of r. We assume k to be even for now. Since S^* was an active set there exists a spanning tree among the facilities consisting of tight edges. Thus there exists a bipartite matching between the facilities $f_1, \ldots, f_k$ such that the tree paths in S^* connecting the matched facilities are disjoint edgewise.

  Let t be the time when the set S^* was frozen. Since there was no conflict among the open facilities, we derive that all the paths connecting matched open facilities must be of length at least t. Therefore the tree cost is at least $\frac{k}{2}t$. But the total contribution that S^* and its subsets could make towards $f_1, \ldots, f_k$ is at most kt. We also know that the tree cost is at most twice the dual accumulated by S^* and its subsets. Thus the total contribution that S^* and its subsets could make towards the open facilities is bounded by $4 \sum_{S,r:S\subseteq S^*} \gamma_{S,r}$.

  The five mentioned above is obtained if k is odd. Then we have to add one more for the contribution that S^* and its subsets could make towards the last facility. Thus, $\mathcal{F} \leq 5 \sum_{S,r} \gamma_{S,r}$.

Putting all the pieces together we obtain that the total cost is within a factor 8 of the dual value, thus obtaining an 8-approximation.

2 Second Solution

For this algorithm we will assume that the edge lengths of the graph are sufficiently large integers.

The Assignment Phase

This algorithm will run in stages. Each stage i will last for time 2^i. We will also have two different types of sets and we will refer to them as *real sets* and *virtual sets*, or *real components* and *virtual components*. Intuitively virtual components will all grow at a uniform rate and are intended to construct a Steiner Tree. The real components on the other hand will grow at a slower rate, depending on how many real components are there in a given virtual component. This is used in the analysis of the conflict resolution step.

Initially the virtual sets for a resource r are identical to the real sets for that resource and are the singleton sets containing the demand nodes requesting that resource. During a given stage, the virtual components for all resources will grow at a uniform rate. Whenever for a given resource r and an edge e the sum of the growths of the virtual components of r bordering e equals the edge cost c_e we say that edge e is *prebought* at stage i. We also stop growing the virtual components bordering e and start growing either the new merged virtual component (if e was purchased from both sides) or the new virtual component that includes e (if e was purchased only from one side). In other words during each stage we are running the traditional Steiner Tree algorithm on virtual components and ignoring the partial edge purchases after the end of a given stage.

The significant difference is that after the stage i if two virtual components are within $\gamma 2^i$ of each other (that is their shortest path is of length no more than $\gamma 2^i$) we merge them into a new virtual component. Here γ is a constant parameter whose value will be set later. Finally, virtual components do not directly correspond to the variables of the dual program.

The real components on the other hand directly correspond to the variables $\gamma_{S,r}$ of the dual program. The real components of a given virtual component grow at a slower rate, more specifically at a rate $1/(k-1)$ where k is the number of real components in that virtual components (if there is only one real component, then the growth rate is 1). They merge when the condition (6) is violated for an edge in which case we say that the edge was *bought* to ship the resource under consideration. Notice that whereas at the end of a stage the virtual components can merge even when not entirely touching one another, the real components need to purchase the edge fully in order to merge.

When the real components of many resource types sufficiently contribute towards a facility i so as to make the constraint (7) tight, we declare i *temporarily open* and freeze (1) the aforementioned real components, (2) the virtual components that include these real components, and (3) all other real components within the newly frozen virtual components. We say that this facility is the *connection witness* for all the virtual and real components that became frozen.

We continue this process until all virtual components are frozen and end the assignment phase.

The Edge Addition and Cleanup Phase

Add edges and cleanup so that the solution becomes primal feasible had we opened all the temporarily open facilities.

*** Explained in the other writeup ***

The Conflict Resolution Phase

We define two facilities to be at a *conflict* if there is a real component that contributed to both of the facilities. Again we construct a graph G_F on facilities where there is an edge between two facilities if they are in conflict and find greedily as before (first come first serve) a maximal independent set F' on this graph. We declare those facilities *open*.

For a given demand node requesting a resource type r, if the connection witness facility i of the component in which it lay was opened then we already have a path to an open facility supplying the desired resource type. If on the other hand the connection witness is closed, then there is another open facility i' as well as a real component S^* that contains both i and i'. We denote with π the path connecting i and i' in S^*. In this case we purchase the path π to supply resource type r.

2.1 Analysis

Feasibility of the solution follows from our construction as before. We need to show that the cost of the solution is bounded by the value accumulated in the dual variables corresponding to the real components. Again we note that the total cost $\mathcal{C}$ can be split into facility costs $\mathcal{F}$ and edge costs $\mathcal{E}$.

- Facility Costs ($\mathcal{F}$):

  We know that when the facilities were declared temporarily open they were fully paid for by the dual variables. As no dual variable can contribute to two open facilities (otherwise there would be conflict) the total facility cost $\mathcal{F}$ is bounded by the total dual value $\sum_{S,r} \gamma_{S,r}$.

- Edge Costs ($\mathcal{E}$):

  We divide the edge costs again into the tree cost and the rerouting cost. The tree cost is bounded as shown in the other writeup.

  In order to bound the rerouting cost we notice that it is sufficient to show the following claim: if a real component has a path of length L with facilities at both ends, then at least $L/(2\beta)$ time must have passed for this path to connect (β here is a constant which will be set later).

  If we succeed showing this then by an argument analogous to the rerouting argument in the first solution we can claim that if a resource r needs to be rerouted from a facility i to a facility i' which are connected by a path π of length L in some real component, then 2β times the dual accumulated at the real components of the frozen virtual component of r at i would be able to pay for the path L. Thus the rerouting cost would be bounded by 2β times the total dual value. Therefore we will focus our attention on proving that a path of length L takes time $L/(2\beta)$ to connect.

  We prove a series of three lemmas after which the desired result follows. The lemmas all will assume that the system under observation consists of a single path and only one resource type.

Lemma 1 *Let us assume we are given a path π consisting of demand nodes $d_1, d_2, \ldots, d_k$ requesting a certain resource type which are separated by edges of lengths $e_1, e_2, \ldots, e_{k-1}$. Let us also assume that all these demand points are in a single virtual components. We claim that the time it takes for the nodes $d_1, \ldots, d_k$ to get connected is exactly equal to than $\frac{\sum e_i}{2}$.*

Proof: Let $t_0 = 0$ be the current time $t_1 \leq t_2 \leq \ldots \leq t_{k-1}$ be the times when an edge was bought. Then from time t_i to time t_{i+1} the total contribution towards a given edge was $\frac{2}{n-i-1}$, since the nodes grow at a speed of $\frac{1}{n-i-1}$ (there are $n-i$ components) and each edge is purchased from both ends. Since there are $n-i-1$ such edges, the total contributions at time $[t_i, t_{i+1})$ is $(t_{i+1} - t_i)\frac{2(n-i-1)}{(n-i-1)} = 2(t_{i+1} - t_i)$. Given that at time t_{k-1} the whole path was purchased, we can conclude that

$$\sum_{i=1}^{k-1} e_i = 2\sum_{i=0}^{k-2}(t_{i+1} - t_i) = 2(t_{k-1} - t_0) = 2t_{k-1}. \quad \square \tag{10}$$

Lemma 2 *Let us be again given a path π of length L consisting of demand points $d_1, d_2, \ldots, d_k$ requesting a certain type of resource. Unlike the previous lemma, here we do not assume that the demand points lie in a single virtual component. We claim that for some constant β the time required for this path to become connected (that is contained in a real component) is at least L/β.*

Proof: We prove this by induction on k. For $k = 2$ if the edge between the two ends is of length l, then after time $l/2$ the real component will contain the entire path. Thus as long as $\beta \geq 2$ the claim will be satisfied.

The inductive step is a bit more involved. Let j be the stage during which the giant virtual component including all the path forms. Then at beginning of stage j there are the virtual components $C_1, C_2, \ldots, C_l$ and edges between them of lengths $e_1, e_2, \ldots, e_{l-1}$. Let us assume that the virtual component C_i is of length p_i. Therefore,

$$L = \sum_{i=1}^{l} p_i + \sum_{i=1}^{l-1} e_i. \tag{11}$$

Observe that $e_i \geq \gamma 2^{j-1}$ otherwise the two virtual components separated by e_i would merge after stage $j - 1$.

Denote by $p_{i_1}, \ldots p_{i_{l_i}}$ the lengths of the real components inside the virtual component C_i after stage j including the edges that are parially paid for (by setting γ large and thus having a large enough values for edges e_i we can ensure that no real component of virtual component C'_i would join with any real component of a neightboring virtual component during stage j). Let $\epsilon_i = p_i - \sum_{\alpha=1}^{l_i} p_{i_\alpha}$. Intuitively ϵ_i is the portion of path in p_i that has not yet been purchased (by real components) after stage j.

We want to give an upper and a lower bound on the time required for each component C_i to connect were it observed in complete isolation. On one hand, by inductive hypothesis, since the length of C_i is p_i, at least time p_i/β must pass for C_i to connect. On the other hand if the component C_i was observed in solitude, then by Lemma 1 after $\epsilon_i/2$ more time the component C_i would get connected (notice that after stage j the whole C_i would be in a single virtual component, which allows us to invoke the previous lemma). Finally, at the end of stage j the total time passed is no more than 2^{j+1}, and thus at time $2^{j+1} + \epsilon_i/2$ the component C_i would connect were it observed in solitude.

Therefore,

$$\begin{aligned}
\frac{p_i}{\beta} &\leq \frac{\epsilon_i}{2} + 2^{j+1} \\
\sum_i \frac{p_i}{\beta} &\leq \sum_i \frac{\epsilon_i}{2} + k2^{j+1} \\
\sum_i \frac{p_i}{\beta} + \sum_i \frac{e_i}{\beta} &\leq \sum_i \frac{\epsilon_i}{2} + k2^{j+1} + \sum_i \frac{e_i}{\beta} \\
L/\beta &\leq \sum_i \frac{\epsilon_i}{2} + k2^{j+1} + \sum_i \frac{e_i}{\beta} \\
2L/\beta &\leq \sum_i \epsilon_i + k2^{j+2} + 2\sum_i \frac{e_i}{\beta}.
\end{aligned}$$

On the other hand, we observe that after stage j all of path L is in a single virtual component. Therefore we shall invoke Lemma 1 again. As of the end of stage j if we can claim that the unpurchased edge length is at least $2L/\beta$, then by invoking the above lemma we obtain the desired result. Notice that the unpurchased edge length is going to be at least $\sum_i \epsilon_i + \sum_i (e_i - 2^{j+2})$, because (1) we need to purchase ϵ_i by the definition and (2) given that the time passed at the end of stage j is no more than 2^{j+1} the total contribution by real components towards edges e_i could not have been more than $e_i - 2^{j+2}$ (an edge can be contributed from both ends), and thus the rest of these edges need to be purchased, too.

If we succeed showing that

$$\sum_i \epsilon_i + \sum_i (e_i - 2^{j+2}) \geq \sum_i \epsilon_i + k2^{j+2} + 2\sum_i \frac{e_i}{\beta},$$

then combining this inequality with the last inequality in the previous set we get the desired result. This latter can be accomplished by setting the values of $\gamma = 16$ and $\beta = 4$ $\square$

Lemma 3 *Let us again be given a path π of length L connecting two facilities via some demand points. Let us assume that the edge connecting the first facility to the first demand point is of length e_1 and the edge connecting the last demand point to the second facility is of length e_2. We claim that at least $L/(2\beta)$ time has to pass in order for the two facilities to be connected via the path π.*

Proof: Let $l = L - e_1 - e_2$ be the path length between the first and last demands. We know that it takes at least time l/β for the path to connect because of Lemma 2. So if $e_1 + e_2 \leq l$ we obtain the desired result.

If $e_1 + e_2 > l$, then we know that the time it takes for the demand points to connect to the two facilities is at least $max\{e_1, e_2\}$, which is at least $L/4$, thus the claim holds again as long as $\beta \geq 2$.

□

We succeeding in showing that if there is a path of length L connecting two facilities and containing demand nodes of a certain resource type, then in the absence of other demand nodes it will take at least time $L/(2\beta)$ for this path to connect. Notice that the addition of more demand nodes could only increase the time, as the real components would grow no faster. Thus if we need to rerout a resource type r via path π corresponding to a component S, then (2β) times the dual accumulated in the real components of the frozen virtual component that includes S would be sufficient to buy the path π.

Combining the edge cost part and the facility cost part of the analysis we obtain a constant approximation to the problem.