

## 2.11

This problem can be thought of much as the set cover problem we discussed in class, where  $t(v)$  represents the ‘price’ of vertex  $v$  and  $c(e)$  represents the amount that edge  $e$  is paying. When  $t(v) = 0$ , we can say that  $v$  is effectively ‘paid for.’ The benefit of this intuition is that it allows us to see that an edge will never be paying too much for any given vertex. The algorithm will pick an edge, and then pay a price for its endpoints. An edge will never overpay because the price it picks is the minimum, and similarly no vertex will ever receive more than its price because we stop paying for it once  $t(v)$  becomes 0 (also because we are picking the minimum price). In other words, we know that for every vertex  $v$ ,  $\sum_{e \text{ incident to } v} c(e) \leq w(v)$ .

Now, following the hint, I’ll prove that the total amount charged to all the edges is less than or equal to the weight of the optimum solution (or actually, any solution).

**Lemma 1.** For a vertex cover  $S$ ,  $\sum_{e \in E} c(e) \leq w(S)$ .

*Proof.* Because the edges never overpay for the vertices, we know that  $\sum_{e=(u,v)} c(e) \leq w(u)$  for all  $u$  in the cover  $S$ . Summing over all these  $u$ , we get

$$\sum_{u \in S} \sum_{e=(u,v)} c(e) \leq \sum_{u \in S} w(u) = w(S). \quad (1)$$

Because  $S$  is a vertex cover, we know that every edge in  $E$  must contribute at least once to the left-hand side of (1), as every edge in the graph is covered and therefore paying some price. This means that  $\sum_{e \in E} c(e) \leq \text{LHS of (1)}$ , which means that  $\sum_{e \in E} c(e) \leq w(S)$ .  $\square$

I continue with the hint to show that the output  $C$  of the algorithm has weight at most twice the total amount charged to edges.

**Lemma 2.**  $w(C) \leq 2 \sum_{e \in E} c(e)$ .

*Proof.* We know that, for every vertex  $v$  in  $C$ , we have  $t(v) = 0$  (by the last line in Step 2). This means that they must have been completely paid for by their surrounding edges, so  $\sum_{e=(u,v)} c(e) = w(u)$  for all  $u \in C$ . Using the same methods as before, we see that

$$w(C) = \sum_{u \in C} w(u) = \sum_{u \in C} \sum_{e=(u,v)} c(e). \quad (2)$$

Now we need to look more closely at the right-hand side. Every edge in  $E$  must contribute at least once because every edge is covered, but it is also possible for an edge to contribute twice if both  $u$  and  $v$  are in the cover  $C$ . This means that

$$\sum_{u \in C} \sum_{e=(u,v)} c(e) \leq 2 \cdot \sum_{e \in E} c(e)$$

which means that  $w(C) \leq 2 \cdot \sum_{e \in E} c(e)$ .  $\square$

Now we are basically done. It is clear that  $C$  must be a vertex cover, or else our while loop would not have terminated. Putting our lemmata together, we see that Lemma 1 tells us that  $\sum_{e \in E} c(e) \leq w(OPT)$  and that Lemma 2 tells us that  $2 \cdot \sum_{e \in E} c(e) \geq w(C)$ . Putting these two equations together, we see that  $w(C) \leq 2 \cdot w(OPT)$ , which tells us that this is a factor 2 approximation algorithm and we are done.