Approximation Algorithms Part II: week 2 assignment

Anonymous

September 4, 2019

Primal-dual algorithm for Shortest Path problem

Q1.

$$\begin{array}{lrl} \text{maximize} & \sum_{S\in\mathcal{S}} y_S & \\ \text{s.t.} & \sum_{S\in\mathcal{S}:e\in S} y_S \leq w(e) & \forall e \in E \\ & y_S \geq 0 & \forall S \in \mathcal{S} \end{array}$$

Q2. At most $|V|-1$ iterations.

Q3. After each iteration, an edge is added and the size of C increases by 1. After $|V|-1$ iterations, C contains all V. So there is at least one path from s to t when the algorithm terminates.

Proof of Lemma 1 We want to show after i iterations, $|F_i| = i$. Initially, $|F_0| = 0$ because C contains only s. After the 1st iteration, $|F_1| = 1$ because step 3 adds one edge. Assuming after iteration $i-1$, $|F_{t-1}| = i-1$. Then after iteration i, $|F_t| = |F_{t-1}|+1 = t$ because of step 3. And this concludes our proof.

Q4. $\text{val}(y^*) \leq \text{val}(P^*)$ due to weak complementary slackness.

Q5. At any iteration, suppose an edge e is added to F. And nods in $e^{'}$ are included into C. Obviously, e is not crossing C and $V \setminus C$. In the next step, we only increase y_C. Therefore, the constraint in dual related to $w(e^{'})$ is never broken. Therefore, the solution y is feasible for the dual.

Q6. We have $\text{val}(y) \leq \text{val}(P^*)$.

Q7. $w(e) = \sum_{S\in\mathcal{S}:e\in\delta(S)} y_S$.

Q8. $\sum_{e\in P} w(e) = \sum_{e\in P} \sum_{S\in\mathcal{S}:e\in\delta(S)} y_S$

Q9. $\sum_{e\in P} \sum_{S\in\mathcal{S}:e\in\delta(S)} y_S = \sum_{S\in\mathcal{S}} y_S \sum_{e\in P\cap\delta(S)} = \sum_{S\in\mathcal{S}} y_S |P \cap \delta(S)|$

Q10. If $|P \cap \delta(S)| >= 2$, it means there is a cycle in some $S^{'}$ after S. This contradicts with the fact that $S^{'}$ is a tree (Lemma 1).

Q11. The primal-dual algorithm is optimal.

Q12. Intuitively, pruning is needed to remove unnecessary edges. It's used in question 10.

Q13. The forward direction: if (s, i) is the first edge added to F, it means (s, i) has the minimum $w(s, j)$ among all neighbors j of s. In Dijkstra algorithm, i is not in D and minimizes $d(i)$. So it's selected.

The reverse direction, if i is selected, it means $d(i)$ is minimal among all neighbors of s. And $d(i)$ is initialized to be $w(s, i)$. Therefore, (s, i) is chosen first by the primal-dual algorithm.

Q14. The vertex j with minimum $l(j)$ will be added to C_0.

Q15. $\{i \in \text{neighbor}(j) : i \notin S^{'}\}$, i.e. j's neighbors that are not in $S^{'}$

Q16. For $k \notin S^{'}, l(k) = l(j) + w(k, j)$.

Q17. The same order!

Q18. $O(n^2)$ if you consider a complete graph.