Optimization and Approximation ENS Lyon

Lecture 9

Lecturer: Alantha Newman *November 22, 2016*

1 Primal-Dual Algorithms

In this lecture, we study combinatorial algorithms that use both the primal and the dual linear programs.

1.1 Complementary Slackness

Recall our canonical linear programs, where $\mathbf{x} \in \mathbb{Q}^n,\ \mathbf{y} \in \mathbb{Q}^m,\ \mathbf{A} \in \mathbb{Q}^{m \times n},\ \mathbf{b} \in \mathbb{Q}^m,\ \mathbf{c} \in \mathbb{Q}^n$.

$$\begin{aligned} \min\ & \mathbf{c} \cdot \mathbf{x} \\ \mathbf{A}\mathbf{x} &\geq \mathbf{b} \\ \mathbf{x} &\geq 0 \end{aligned} \tag{P}$$

$$\begin{aligned} \max\ & \mathbf{b}^\intercal \mathbf{y} \\ \mathbf{A}^\intercal \mathbf{y} &\leq \mathbf{c} \\ \mathbf{y} &\geq 0 \end{aligned} \tag{D}$$

Also recall that *strong duality* ensures that if both (P) and (D) have finite optima, they are equal. Assume this is the case. Then, for the optima $\mathbf{x}^*$ and $\mathbf{y}^*$ of the primal and dual programs respectively, we have:

$$\mathbf{c} \cdot \mathbf{x}^* = \sum_{i=1}^{n} c_i x_i^* \geq \sum_{i=1}^{n} (\sum_{j=1}^{m} A_{ji} y_j^*) x_i^* = \sum_{j=1}^{m} (\sum_{i=1}^{n} A_{ji} x_i^*) y_j^* \geq \sum_{j=1}^{m} b_j y_j^* = \mathbf{b}^\intercal \mathbf{y}.$$

Since $\mathbf{c} \cdot \mathbf{x}^* = \mathbf{b}^\intercal \mathbf{y}^*$, the first and second inequalities should hold with equality. This implies that:

$$\text{Primal complementary slackness:} \quad \text{Either } \sum_{j=1}^{m} A_{ji} y_j^* = c_i \text{ or } x_i^* = 0.$$

$$\text{Dual complementary slackness:} \quad \text{Either } \sum_{i=1}^{n} A_{ji} x_i^* = b_j \text{ or } y_j^* = 0.$$

These conditions are together terms as the (full) *complementary slackness* conditions. They state that either $x_i^* = 0$ or the corresponding dual constraint is tight (or both). Similiarly, either $y_j^* = 0$ or the corresponding primal constraint is tight (or both). In other words, if $x_i^* > 0$, then the corresponding dual constraint is tight. Similarly, if $y^* > 0$, then the corresponding primal constraint is tight. If a dual constraint is *not* tight, then the corresponding primal variable x_i^* must be zero. The same holds for a primal constraint and the respective dual variable.

Consider the following approach for finding optimal primal and dual solutions. Assume that $\mathbf{c} \geq 0$ and that both (P) and (D) have finite optimal values. Furthermore, suppose that we have an initial feasible solution $\mathbf{y}$ for (D). (For example, $\mathbf{y} = 0$ is a feasible dual solution.) Our goal is:

1. Either find a primal feasible solution $\mathbf{x}$ such that complementary slackness conditions are satisfied for $\mathbf{x}$ and $\mathbf{y}$, thus certifying the optimality of both $\mathbf{x}$ and $\mathbf{y}$.

2. Or find a feasible dual solution with a greater objective value, i.e. an improved dual solution, to yield a larger (better) lower bound.

We now apply this idea to the hitting set problem.

1.2 Hitting Sets

Recall the definition of the *hitting set problem*. The input is a universe of n elements and a family of m subsets of U. Each element is associated with a cost $c : U \to \mathbb{R}^+$. The goal is to find a subset of U of minimum cost that contains at least one element from each T_j in $\mathcal{T}$.

Input: $U = \{e_1, e_2, \ldots, e_n\}, \mathcal{T} = \{T_1, T_2, \ldots, T_m\}$, where $T_j \subseteq U$.

Output: $A \subseteq U$ with minimum cost $c(A)$ such that $T_j \cap A \neq \emptyset$ for all $j = 1, 2, \ldots, m$.

Let us consider the primal and dual linear programming relaxations for the hitting set problem:

$$\begin{aligned} \min & \sum_{e \in U} c_e x_e \\ \text{subject to:} & \sum_{e \in T_j} x_e \geq 1, \quad \text{for all } j = 1, 2, \ldots, m, \\ & x_e \geq 0. \end{aligned} \tag{P_{HS}}$$

$$\begin{aligned} \max & \sum_{j=1}^{m} y_j \\ \text{subject to:} & \sum_{j : e \in T_j} y_j \leq c_e, \quad \text{for all } e \in U, \\ & y_j \geq 0. \end{aligned} \tag{D_{HS}}$$

If $\mathbf{x}$ is the incidence vector of a solution set $A \subseteq U$, and $\mathbf{y}$ is a dual feasible solution, then the primal complementary slackness conditions are:

$$e \in A \;\Rightarrow\; \sum_{j : e \in T_j} y_j = c_e,$$

and the dual complementary slackness conditions are:

$$y_j > 0 \;\Rightarrow\; |A \cap T_j| = 1.$$

We now develop a basic primal-dual algorithm for the hitting set problem that is based on enforcing the primal complementary slackness conditions and relaxing the dual conditions.

2 Primal-Dual Algorithm 1.0

Begin with a feasible dual solution for (D_{HS}) (e.g. $\mathbf{y} = 0$). At each iteration of the algorithm, we have a feasible dual solution $\mathbf{y}$.

1. We construct a (possibly infeasible) primal solution (determined by complementary slackness):

$$A = \quad \{e \mid \textstyle\sum_{j : e \in T_j} y_j = c_e\}, \quad \text{and} \quad x_e = 1 \text{ iff } e \in A.$$

2. If $\mathbf{x}$ is a feasible solution for (P_{HS}), we stop, or
3. If $\mathbf{x}$ is infeasible, we find a new improved dual feasible solution $\mathbf{y}$ and go to Step 1.

With respect to a feasible dual solution $\mathbf{y}$, we define an *improved* solution $\mathbf{y}'$ as one in which:

(i) $\mathbf{y}' \geq \mathbf{y}$, and

(ii) $\mathbf{y}'$ has increased objective value for (D_{HS}), i.e. $\sum_{j=1}^{m} y'_j > \sum_{j=1}^{m} y_j$.

How do we find such an improved dual solution $\mathbf{y}'$? Since $\mathbf{x}$ is infeasible for (P_{HS}), then A is an infeasible solution for the hitting set problem. Thus, there must be some set T_k such that $A \cap T_k = \emptyset$. In this case, we say that the constraint in (P_{HS}) corresponding to the set T_k is *violated*. Then we can increase the dual variable y_k by the following amount:

$$\Delta \quad = \quad \min_{e \in T_k} \{c_e - \sum_{j: e \in T_j} y_j\}.$$

Observe that $\Delta > 0$, since no element e in T_k currently belongs to A. Setting $y'_k = y_k + \Delta$ and $y'_j = y_j$ for all $j \neq k$ yields the improved feasible dual solution $\mathbf{y}'$. For some edge $e \in T_k$, the corresponding dual constraint becomes tight, and this new edge e is added to the solution set A in the next iteration. Thus, the size of the solution set is strictly increased. We now summarize this basic framework.

PRIMAL-DUAL ALGORITHM 1.0

$\mathbf{y} := 0; A := \emptyset$
While A is infeasible (i.e. $\exists k : A \cap T_k = \emptyset$)

Find $k : A \cap T_k = \emptyset$

Increase y_k until $\exists e \in T_k : \sum_{j: e \in T_j} y_j = c_e$

$A \leftarrow A \cup \{e\}$

Output A and $\mathbf{y}$

How good is the solution produced by this algorithm? Suppose that A and $\mathbf{y}$ are output by the algorithm. Recall that a feasible dual solution provides a lower bound on the value of an optimal hitting set:

$$\sum_{j=1}^{m} y_j \quad \leq \quad OPT_{HS}$$

Since each edge e in A is added when its corresponding dual constraint is tight, we see that the cost of the solution set A is:

$$c(A) \quad = \quad \sum_{e \in A} c_e \quad = \quad \sum_{e \in A} \sum_{j: e \in T_j} y_j \quad = \quad \sum_{j=1}^{m} |T_j \cap A| \cdot y_j.$$

Thus, if we can show that $|A \cap T_j| \leq \alpha$ for all sets T_j, then

$$c(A) \ \leq \ \alpha \sum_{j=1}^{m} y_j \ \leq \ \alpha \cdot OPT_{HS},$$

and our primal-dual algorithm would have an approximation guarantee of α. This immediately implies that if all sets have a bounded size, e.g. $|T_j| \leq \alpha$ for all sets T_j, then we have an α-approximation algorithm.

In terms of efficiency, the algorithm requires at most $|U|$ iterations. Although there may be exponentially many sets T_j, the number of nonzero dual variables (i.e. $y_j > 0$) does not exceed the number of iterations. Thus, as long as we can find a violated set efficiently, the algorithm is efficient.

2.1 Shortest Path

Given a graph $G = (V, E)$ with nonnegative costs on the edges and two specified vertices $s, t \in V$, the *shortest path* problem is to find the cheapest path from s to t. We now apply the primal-dual algorithm from the previous section to this problem. It will be necessary to modify the algorithm slightly.

Let the subset $S \subset V$ belong to $\mathcal{T}$ if $s \in S$ and $t \notin S$. Let $A \subset E$ be a candidate solution for the shortest path problem and let $\mathbf{x}$ be the incidence vector of A. We say that a set $T_k \in \mathcal{T}$ is *violated* if $x(\delta(T_k)) = 0$. We say that T_k is *minimally violated* if it is violated and it does not contain any violated sets. An algorithm for the minimum s-t cut can be used to find a minimially violated set if one exists.

PRIMAL-DUAL SHORTEST PATH

$\mathbf{y} := 0; A := \emptyset$
While A is infeasible

Find a minimally violated $T_k \in \mathcal{T}$.

Increase y_k until $\exists e \in \delta(T_k) : \sum_{j:e \in T_j} y_j = c_e$

$A \leftarrow A \cup \{e\}$

Output A and $\mathbf{y}$

Let A and $\mathbf{y}$ denote the final output of the primal-dual algorithm. The set A will be a (possibly partial) shortest path tree, implying that A does not contain any cycles.[1] However, we are simply looking for a shortest path from s to t and not a shortest path tree. Thus, we modify the algorithm by adding an additional *pruning* step: For each edge $e \in A$, we test to see if $A \setminus e$ still contains an s-t path. (In other words, if we remove edge e, is there a set $T_k \in \mathcal{T}$ that is now violated?) If so, we remove edge e from A. Call this pruned set P, since it is in fact a path. If we can show that $|P \cap \delta(T_k)| = 1$ for all sets $T_k \in \mathcal{T}$ such that $y_k > 0$, then this will prove that the primal-dual algorithm is an optimal algorithm for the shortest path problem.

We now show that if $y_k > 0$, then $|P \cap \delta(T_k)| = 1$. Suppose that this is not the case and that there is some T_k such that $|P \cap \delta(T_k)| > 1$. Then there is some subpath $P' \subset P$ whose start vertex s' and end vertex t' are both in T_k. Consider the iteration in the algorithm in which T_k was chosen as the violated set and dual variable y_k was increased. There are two possibilities:

1. Either the set T_k contains a tree containing a path from s to s' and s to t'. Then (eventually) adding edges in the path P' to the solution set A will result in A containing a cycle.
2. Or the set T_k does not contain a path from s to s' and from s to t'. In this case, T_k is not a minimally violated set.

Since A does not contain cycles and each chosen violated set is minimally violated, both cases lead to contradictions. We conclude that $|P \cap \delta(T_k)| = 1$ for any violated set T_k chosen during iteration of the algorithm.

3 Primal-Dual Algorithm 2.0

In the primal-dual algorithm from Section 2, exactly *one* dual variable is increased per iteration. Sometimes this does not result in a good solution; the size of $|A \cap T_j|$ may be too large. We now consider a modification of the algorithm in which we increase *multiple* dual variables at each iteration. Specifically, suppose that at some iteration of the algorithm, an oracle returns a set of violated sets instead of just one

[1] Exercise 4 in TD8.

violated set. Call these violated sets $\mathcal{V} = \{T_1, T_2, \ldots, T_\ell\}$. We increase the dual variables $y_1, y_2, \ldots, y_\ell$ simultaneously at the same rate until a new dual contraint becomes tight.

PRIMAL-DUAL ALGORITHM 2.0

$\mathbf{y} := 0; A := \emptyset; \ell := 0$
While A is infeasible

Find a set $\mathcal{V}$ of violated constraints

For all $T_k \in \mathcal{V}$, increase y_k uniformly until $\exists e \notin A$:

- $e \in T_k$ for some $T_k \in \mathcal{V}$, and
- $\sum_{j:e \in T_j} y_j = c_e$

$A \leftarrow A \cup \{e\}$

Output A and $\mathbf{y}$

Now let us analyze the cost of a solution produced by this algorithm. Let A and $\mathbf{y}$ denote the output of the algorithm. Let $\mathcal{V}_i$ denote the collection of violated sets returned by the oracle in the i^{th} iteration of the algorithm and let ℓ denote the number of iterations. Let ϵ_i denote the increase in the dual variables corresponding to the sets in this collection. Then, we have:

$$y_j \quad = \quad \sum_{i:T_j \in \mathcal{V}_i} \epsilon_i.$$

Thus, the dual lower bound on the value of an optimal hitting set can be written as:

$$\sum_{j=1}^{m} y_j \quad = \quad \sum_{j=1}^{m} \sum_{i:T_j \in \mathcal{V}_i} \epsilon_i \quad = \quad \sum_{i=1}^{\ell} |\mathcal{V}_i| \epsilon_i.$$

The cost of the solution set is:

$$c(A) \quad = \quad \sum_{j=1}^{m} |T_j \cap A| \cdot y_j \quad = \quad \sum_{j=1}^{m} |T_j \cap A| \sum_{i:T_j \in \mathcal{V}_i} \epsilon_i \quad = \quad \sum_{i=1}^{\ell} \left(\sum_{T_j \in \mathcal{V}_i} |A \cap T_j| \right) \epsilon_i.$$

So if the following holds for all $i = 1, 2, \ldots, \ell$ and for some $\alpha \geq 1$,

$$\sum_{T_j \in \mathcal{V}_i} |A \cap T_j| \quad \leq \quad \alpha \cdot |\mathcal{V}_i|,$$

then this method yields an α-approximation algorithm.

3.1 Steiner Forest

We will now apply the primal-dual method from the previous section to the *steiner forest* problem. Given a graph $G = (V, E)$ with nonnegative costs on the edges and k pairs (s_i, t_i) of vertices in V, the goal of the steiner forest problem is to find a minimum cost subset of edges $F \subseteq E$ such that there is a path from s_i to t_i in F for each of the k pairs.

Let $\mathcal{S}_i$ denote the subsets of vertices separating s_i and t_i and let $\mathcal{S} = \cup_{i=1}^{k} \mathcal{S}_i$.

$$\mathcal{S}_i \quad = \quad \{S \subset V : \ |S \cap \{s_i, t_i\}| = 1\}.$$

Then we have the following primal and dual linear programs for the steiner forest problem.

$$\begin{aligned} \min & \sum_{e \in E} c_e x_e \\ \text{subject to:} & \sum_{e \in \delta(S)} x_e \geq 1, \quad \text{for all } S \subset V : S \in \mathcal{S}_i \text{ for } i = 1, 2, \ldots, k, \\ & x_e \geq 0. \end{aligned} \tag{P_{SF}}$$

$$\begin{aligned} \max & \sum_{\substack{S \subset V: \\ \exists i, S \in \mathcal{S}_i}} y_S \\ \text{subject to:} & \sum_{S : e \in \delta(S)} y_S \leq c_e, \quad \text{for all } e \in E, \\ & y_S \geq 0. \end{aligned} \tag{D_{SF}}$$

A solution to the linear program (D_{SF}) can be viewed as a packing of cuts.

Now we describe an algorithm based on the primal-dual method from Section 3. Right before an iteration of the algorithm, let C be a set of vertices such that its induced edge set $E(C) \in A$ is a maximal connected component. Then $C \in \mathcal{C}$ if $|C \cap \{s_i, t_i\}| = 1$ for some $i = 1, 2, \ldots, k$. Note that $\mathcal{C}$ is a collection of sets violated with respect to the current (infeasible) solution A.

PRIMAL-DUAL STEINER FOREST

$\mathbf{y} := \emptyset$; $A := \emptyset$; $F := \emptyset$
While A is infeasible

Find all (violated) sets in $\mathcal{C}$

For all $C \in \mathcal{C}$, increase y_C uniformly (if possible) until $\exists e \notin A$:

- $e \in \delta(C)$ for some $C \in \mathcal{C}$, and
- $\sum_{C : e \in \delta(C)} y_C = c_e$

$A \leftarrow A \cup \{e\}$

$F := A$
For each $e \in F$, test if $F \setminus e$ is feasible (if so, delete e)
Output F and $\mathbf{y}$

During some iterations, the dual variables may not actually increase, because edges crossing some cut $C \in \mathcal{C}$ may already be tight from increases during a previous iteration. In this case, we add a single tight edge and move to the next iteration (with a recomputed set $\mathcal{C}$). Adding only one edge to the solution per iteration avoids creating cycles in A.

We observe that each time an edge e is added to the solution set A, the dual variables y_C in the dual constraint corresponding to e are frozen in the sense that they will never belong to a violated set at a future iteration. In other words, if $e \in \delta(C)$, then y_C will never be increased in the future since the edge set $E(C)$ is no longer a maximal connected component in A. The "freezing" of these dual variables implies that $\mathbf{y}$ is a dual feasible solution at termination.

At termination, we also claim that F is primal feasible and F is a forest. The former follows from the fact that the algorithm would not terminate if there were remaining violated constraints. The latter follows from the fact that no edge belonging to a connected component $E(C)$ is ever added to the solution set A, so A is in fact a forest implying that $F \subseteq A$ is also a forest.

Now we want to prove the following lemma.

Lemma 1. PRIMAL-DUAL STEINER FOREST *is a 2-approximation algorithm.*

Proof. From the tools in Section 3, it suffices to prove:

$$\sum_{C \in \mathcal{C}} |F \cap \delta(C)| \quad \leq \quad 2 \cdot |\mathcal{C}|,$$

where $\mathcal{C}$ denotes the collection of violated sets found at some iteration of the algorithm. Let B denote the set of edges from F that are in $\delta(C)$ for some $C \in \mathcal{C}$. If for each edge $e \in B$, both endpoints of e belong to (different) sets in $\mathcal{C}$, then the desired inequality holds, since in this case B forms a forest on the components of $\mathcal{C}$. However, there may be edges in B that have only one endpoint in some $C \in \mathcal{C}$.

So consider the subset of edges $F' \subseteq F$ such that each edge in F' is on a (unique) path from C' to C'' for some $C', C'' \in \mathcal{C}$. Edges inside a component $C \in \mathcal{C}$ are not included in F'. The set F' can be decomposed into at most $|\mathcal{C}| - 1$ paths such that the first or last edge on each path (or both) belongs to B. Every edge in B is the endpoint of one of these paths, otherwise it would have been deleted. (Suppose such an edge e is on some path between $C \in \mathcal{C}$ and some component $C' \notin \mathcal{C}$. If e is on the unique path in F connecting s_i in C to t_i in C', then C' should belong to $\mathcal{C}$, which is a contradiction.) So we have the following inequality,

$$\sum_{C \in \mathcal{C}} |F \cap \delta(C)| \quad = \quad |B| \ \leq \ 2|\mathcal{C}| - 2,$$

since in the worst case, each of the paths uses two edges from B. □

4 Dual-Fitting

Another approach to using duality in the design of approximation algorithms is the method of *dual fitting*. In this method, we find an *integer* solution $\mathbf{x}$ for the primal linear program and show that:

$$\mathbf{c} \cdot \mathbf{x} \leq \alpha \cdot \mathbf{b}^\intercal \mathbf{y}, \tag{1}$$

for some dual-feasible solution $\mathbf{y}$, where $\mathbf{y}$ is not necessarily integer. Note that by weak duality, $\mathbf{b}^\intercal \mathbf{y}$ is a lower bound on the value of an optimal solution. Thus, such a solution $\mathbf{x}$ would result in an α-approximation algorithm. Suppose we can obtain $\mathbf{x}$ and $\mathbf{y}$ without solving a linear program, and we can show that $\mathbf{x}$ and $\mathbf{y}$ are feasible for the primal and dual linear programs, respectively. If we can show that (1) holds, then we have a combinatorial algorithm. We now give two examples that illustrate the dual fitting method.

4.1 Vertex Cover

Recall the vertex cover problem. Given a graph $G = (V, E)$, the goal is to find the minimum subset of vertices such that every edge is covered. The linear program for this problem and its dual are formulated below:

Primal (Vertex Cover) LP:

$$\begin{array}{lrcll} \min & \sum\limits_{v \in V} x_v & & & \\ \text{subject to} & x_u + x_v & \geq & 1 & \forall\, (u,v) \in E \\ & x_v & \geq & 0 & \end{array}$$

Dual (Matching) LP:

$$\begin{array}{lll} \max & \sum_{e \in E} y_e & \\ \text{subject to} & \sum_{e: v \in e} y_e \leq 1 & \forall\, v \in V \\ & y_e \geq 0 & \end{array}$$

Let M be a maximal matching of the graph G, and let y'_{ij} be defined as follows:

$$y'_{ij} = \begin{cases} 1 & \text{if } e = (i,j) \in M, \\ 0 & \text{o.w.} \end{cases}$$

Now we construct a vertex cover as follows. For all $(i,j) \in M$, we add vertices i, j to the vertex cover. This set covers all edges of the graph, since M was chosen to be maximal. Hence, the cost of vertex cover is:

$$\text{cost}(VC) = 2 \cdot \sum_{i,j \in E} y'_{ij} = 2 \cdot |M|.$$

Since M is a maximal matching, $\{y'_{ij}\}$ is dual-feasible. Hence, by weak duality, $|M|$ is a lower bound for the vertex cover linear program. Thus, we obtain a 2-approximation algorithm for the vertex cover problem.

4.2 Feedack Arc Set on Tournaments

Let $G = (V, A)$ be a tournament, which is a complete graph in which each edge is oriented in one direction. The goal of the feedback arc set problem is to find a minimum cardinality subset of arcs $S \subset A$ such that $G' = (V, A \setminus S)$ is acyclic. The primal linear program and its dual are shown below. Note that $\mathcal{C}$ is the set of directed cycles in G.

Primal (Covering) LP:

$$\begin{array}{lll} \min & \sum_{(i,j) \in A} x_{ij} & \\ \text{subject to:} & \sum_{(i,j) \in c} x_{ij} \geq 1, & \forall\, c \in \mathcal{C} \\ & x_{ij} \geq 0 & \end{array}$$

Dual (Packing) LP:

$$\begin{array}{lll} \max & \sum_{c \in \mathcal{C}} y_c & \\ \text{subject to:} & \sum_{c: (i,j) \in c} y_c \leq 1, & \forall\, (i,j) \in A \\ & y_c \geq 0 & \end{array}$$

We consider the following algorithm for the feedback arc set problem on tournaments. Note that this problem is equivalent to finding an ordering of the vertices that minimizes the number of backward edges, i.e. the set S comprises the backward edges in the ordering.

ALGORITHM: RANDOMFAS
Input: $G = (V, A)$.
Output: Ordering of vertices in V.

- Choose $i \in V$ at random.
- if $(j, i) \in A$, $\Rightarrow$ $j \to L$
- if $(i, j) \in A$, $\Rightarrow$ $j \to R$
- **return** (RANDOMFAS(L, A_L), RANDOMFAS(R, A_R))

We now prove that RANDOMFAS is a 3-approximation algorithm for the problem of feedback arc set on tournaments.

Let T be the set of directed triangles: $\{t \in T : i \to j \to k \to i\}$. Note that $T \subset \mathcal{C}$. Let A_t be the event that one vertex of $t = \{i, j, k\}$ is chosen before triangle t is broken by the algorithm, i.e. when $\{i, j, k\}$ occur in same recursive call. Let p_t be the probability of event A_t. Then, we have:

$$\mathbb{E}[\text{cost of solution}] = \mathbb{E}[\text{number of backward edges}] = \sum_{t \in T} p_t.$$

In the next lemma, we will show that setting $y'_c = 0$ for all $c \in \mathcal{C}$, where c is not a triangle (i.e. $c \notin T$), and $y'_t = \frac{p_t}{3}$ for all $t \in T$ yields the dual feasible solution $\{y'_c\}$.

Lemma 2. *Setting $y'_c = y'_t = \frac{p_t}{3}$, if $c = t \in T$ and 0 otherwise is dual feasible.*

Proof. Let B_e be the event that edge e is backwards in the output ordering. Let $B_e \wedge A_t$ be the event that edge e is backwards due to A_t. For example, suppose vertices i, j, k form triangle t which is yet unbroken when vertex k is chosen as the pivot. Then we say that edge $e = (i, j)$ is backwards due to event A_t. Since, given event A_t, each edge in t is equally likely to be a backwards edge, we have:

$$\begin{aligned} Pr(B_e \wedge A_t) &= Pr(B_e | A_t) Pr(A_t) \\ &= \frac{1}{3} \times p_t \\ &= \frac{p_t}{3}. \end{aligned}$$

Note that for any $t \neq t' \in T$ such that $e \in t$ and $e \in t'$, $B_e \wedge A_t$ and $B_e \wedge A_{t'}$ are disjoint events. Hence, $\sum_{t: e \in t} \Pr(B_e \wedge A_t) \leq 1$. This implies that, for all $e \in A$:

$$\sum_{c: e \in c} y'_c = \sum_{t: e \in t} \frac{p_t}{3} \leq 1.$$

We can therefore conclude that $\{y'_c\}$ is a dual-feasible solution. □

Thus, we obtain a 3-approximation algorithm for the problem of feedback arc set on tournaments.

Theorem 3. *The approximation guarantee of* RANDOMFAS *is* 3.

Proof.

$$\mathbb{E}[\text{cost of solution}] = \sum_{t \in T} p_t = \sum_{c \in C} 3y_c = 3 \cdot \sum_{c \in \mathcal{C}} y_c \leq 3 \cdot OPT.$$

Since we have shown that the values $\{y'_c\} = \{p_t/3\}$ are dual-feasible, it follows from weak duality that $\sum_{t \in T} p_t/3 = \sum_{c \in \mathcal{C}} y'_c$ is a lower bound on the size of minimum feedback arc set. □

References

[ACN08] Nir Ailon, Moses Charikar, and Alantha Newman. Aggregating inconsistent information: Ranking and clustering. *Journal of the ACM*, 55(5):23, 2008.

[AKR95] Ajit Agrawal, Philip Klein, and R. Ravi. When trees collide: An approximation algorithm for the generalized steiner problem on networks. *SIAM Journal on Computing*, 24(3):440–456, 1995.

[GW95] Michel X. Goemans and David P. Williamson. A general approximation technique for constrained forest problems. *SIAM Journal on Computing*, 24(2):296–317, 1995.

[GW97] Michel X. Goemans and David P. Williamson. The primal-dual method for approximation algorithms and its application to network design problems. *Approximation algorithms for NP-hard problems*, pages 144–191, 1997.

[WS11] David P. Williamson and David B. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, 2011.

The original 2-approximation algorithm for Steiner forest is due to Agrawal, Klein and Ravi [AKR95]. Subsequently, Goemans and Williamson gave a primal-dual interpretation and applied the primal-dual method to many other network design problems [GW95, GW97]. The algorithm for the feedback arc set problem on tournaments can be found in [ACN08]

These lecture notes are based in part on Chapter 7 of [WS11], a survey on primal-dual algorithms [GW97], lecture notes by Stéphan Thomassé from previous versions of the same course and lecture notes from a course on approximation algorithms at EPFL (`http://theory.epfl.ch/osven/courses/Approx13`).