Module 3: Duality Through Examples

CO 250: Introduction to Optimization

David Duan, 2019 Spring

Contents

Lecture 17. Duality Through Examples

17.1 The Shortest Path Problem

17.2 Finding an Intuitive Lower Bound: Cardinality Case

17.3 Finding an Intuitive Lower Bound: General Case

Lecture 18. Weak Duality

18.1 Dual of Primal LP

18.2 Lower-bounding the Length of s,t-Paths

Lecture 19. Shortest Path Algorithm

19.1 Slack

19.2 Shortest Path Algorithm

Lecture 20. Correctness

20.1 Shortest Path Characterization

20.2 Proof for Correctness

Lecture 17. Duality Through Examples

> 1. *A width assignment $y_U \geq 0$ for all s,t-cuts $\delta(U)$ is feasible if for all $e \in E$, $\sum(y_U : e \in \delta(U)) \leq c_e$.*
> 2. *If y is a feasible width assignment and P is an s,t-path, then $c(P) \geq \sum y_U$.*
> 3. *Intuitively, a width assignment counts the number of cuts of the same type in the corresponding cardinality graph.*

17.1 The Shortest Path Problem

Given a graph $G = (V, E)$, a non-negative length c_e for each edge $e \in E$ and a pair of vertices $s, t \in V$, our goal is to compute an s,t-path P of smallest total length.

An s,t-path is a sequence $P := u_1u_2, u_2u_3, \ldots, u_{k-1}u_k$ where $u_iu_{i+1} \in E$ for all i and $u_1 = s$, $u_k = t$, and $u_i \neq u_j$ for all $i \neq j$. The length of P is given by $c(P) = c_{u_1u_2} + c_{u_2u_3} + \cdots + c_{u_{k-1}u_k}$.

We focus on two questions:

1. Given a shortest-path instance and a candidate shortest s,t-path P, is there a short proof of its optimality?
2. How can we find a shortest s,t-path?

17.2 Finding an Intuitive Lower Bound: Cardinality Case

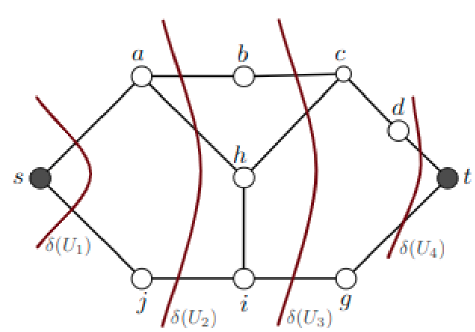
We will first consider the cardinality special case of the shortest path problem. That is, each edge $e \in E$ has length 1, so we are therefore looking for an s,t-path with the smallest number of edges.

Definition 17.2.1 For a subset of vertices $U \subseteq V$, we define $\delta(U) = \{uv \in E : u \in U, v \notin U\}$ and call it an **s,t-cut**, if $s \in U$ and $t \notin U$.

Remark 17.2.2 Recall the following:

1. If P is an s,t-path and $\delta(U)$ is an s,t-cut, then P contains an edge from $\delta(U)$.
2. If $S \subseteq E$ contains an edge from *every* s,t-cut, then S contains an s,t-path.

Example 17.2.3 The following example shows 4 s,t-cuts, $\delta(U_i)$ for $i = 1, 2, 3, 4$:



$$\delta(U_1) = \{sa, aj\}$$
$$\delta(U_2) = \{ab, ah, ji\}$$
$$\delta(U_3) = \{bc, hc, ig\}$$
$$\delta(U_4) = \{dt, gt\}$$

We have two important notes:

1. $\delta(U_i) \cap \delta(U_j) = \varnothing$ for $i \neq j$, and
2. An s,t-path must contain an edge from $\delta(U_i)$ for all i.

By (2), we know that every s,t-path must have at least 4 edges. Thus, sj, ji, ig, gt is the shortest s,t-path! Notice if an edge is not in any of the cuts (e.g., hi), then it must not be on any of the shortest s,t-path.

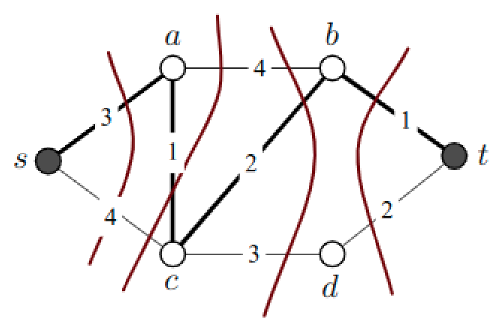
17.3 Finding an Intuitive Lower Bound: General Case

We now consider the general case, where we assign a non-negative width y_U to every s,t-cut $\delta(U)$.

Definition 17.3.1 A **width assignment** $\{y_U : \delta(U)\ s,t\text{-cut}\}$ is *feasible* if, for every edge $e \in E$, the *total width* of all cuts containing e is no more than c_e. Using math, y is feasible if for all e,

$$\sum (y_U : e \in \delta(U)\ s,t\text{-cut}) \leq c_e.$$

Example 17.3.2 Consider the following example with 4 s,t-cuts:

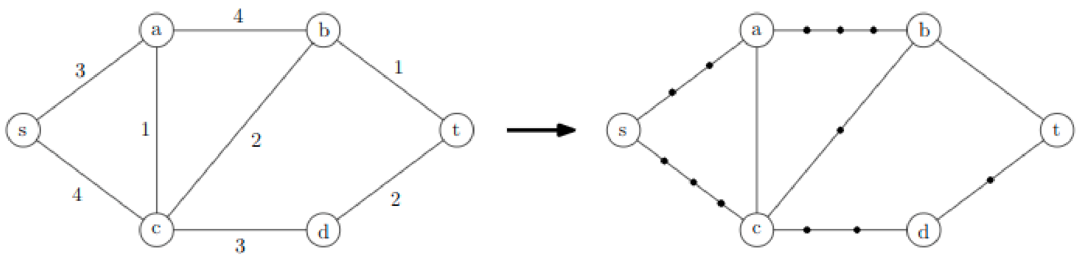


$$\begin{aligned} U_1 &= \{s\} & \delta(U_1) &= \{sa, sc\} \\ U_2 &= \{s, a\} & \delta(U_2) &= \{ab, sc, ac\} \\ U_3 &= \{s, a, c\} & \delta(U_3) &= \{ab, cb, cd\} \\ U_4 &= \{s, a, b, c, d\} & \delta(U_4) &= \{bt, dt\} \end{aligned}$$

Consider the width assignment $y_{U_1} = 3, y_{U_2} = 1, y_{U_3} = 2, y_{U_4} = 1$. It is easy to check this is feasible.

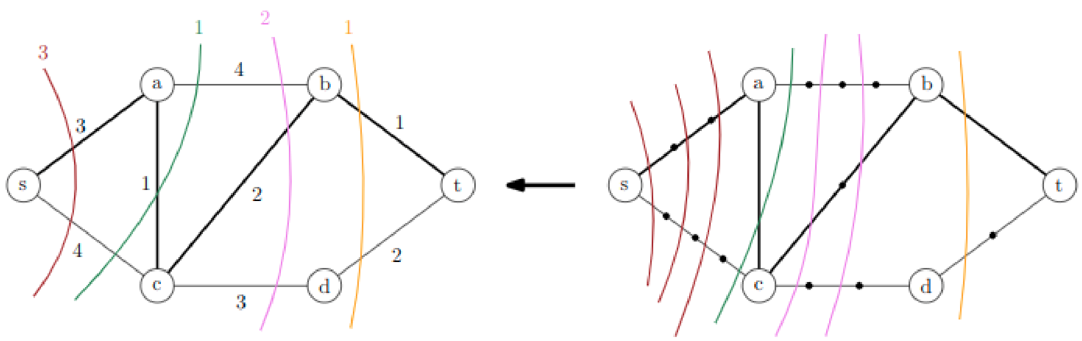
- sa is contained in $\delta(U_1)$ only, and we see that $y_{U_1} = 3 \leq 3 = c_{sa}$.
- sc is contained in $\delta(U_1)$ and $\delta(U_2)$, and we see that $y_{U_1} + y_{U_2} = 4 \leq 4 = c_{sc}$.
- The rest of edges can be checked using an identical argument.

Remark 17.3.3 We now provide the intuition behind width assignments. For simplicity, assume all lengths are positive integers. We take each edge with length k and split the edge into k edges.



Let G be the original graph (left) and H be the transformed (right). It should be easy to see that any s,t-path from G of length ℓ corresponds to an s,t-path from H that uses ℓ edges, and vice versa.

Suppose now we have some disjoint s,t-cuts in H that proves our s,t-path is the shortest.



We set the restriction such that each edge is only allowed to be used in at most one such cut. **Then, we combine cuts of the same type (i.e., they cross the same corresponding edges in G) into one, and give the number of such cuts as the width assignment of the corresponding cut in G.** For example, we see that three cuts in H go through sa and sc, which corresponds to $\delta\{U_1\}$ in G. Thus, we set $y_{U_1} = 3$. Similarly, we see that only one cut in H go through ab, ac and sc, which is U_2, so we set $y_{U_2} = 1$. We set $y_{U_3} = 2$ and $y_{U_4} = 1$ the same way.

Note that the constraints on the width assignments (i.e., $\sum(y_U : e \in \delta(U)\ s,t\text{-cut}) \leq c_e$) are satisfied, since each edge in G of length k is part of at most k cuts in the corresponding k edges in H. For example, edge ab has length 4 in G, and that corresponds to 4 edges in H. Since each edge in H is used at most once, this means that at most 4 cuts in H use these edges. By combining them to get cuts in G (there are two of them, $\delta(U_2)$ and $\delta(U_3)$), we do not exceed a total width assignment of 4 for edge ab.

Again, the key intuition is that we combined the cuts of the same type into one and gave the number of such cuts as the width assignment of the corresponding cut in G.

Proposition 17.3.4 If y is a feasible width assignment, then any s,t-path must have length at least

$$c(P) \geq \sum(y_U : U\ s,t\text{-cut}).$$

Proof. Consider an s,t-path P. It follows that

$$\begin{aligned} c(P) &= \sum(c_e : e \in P) \\ &\geq \sum\Big(\sum(y_U : e \in \delta(U)) : e \in P\Big) \\ &\geq \sum(y_U : \delta(U)\ s,t\text{-cut}) \end{aligned}$$

The first inequality follows from the definition of a feasible width assignment. Next, if $\delta(U)$ is an s,t-cut, then P contains at least one edge from $\delta(U)$, thus variable y_U appears *at least* once on the LHS. □

Remark 17.3.5 This result should be quite intuitive as it's merely a combination of 17.3.2 (2) and 17.3.3. From 17.3.3, we know that the width assignment of a general graph G corresponds to the number of cuts in its corresponding cardinality graph H. By 17.3.2 (2), an s,t-path must contain one edge from each cut. By the constraint on the width assignments, each edge in G of length k is part of *at most* k cuts in the corresponding k edges in H, i.e., each $e \in E(G)$ has the property that c_e is no less than the total number of cuts going through its corresponding edge(s) in $E(H)$. Translating this from cardinality graph to general graph, the s,t-path then must have a length at least the sum of all width assignments.

Remark 17.3.6 Consider the following two questions:

1. In an instance with a shortest path, can we *always* find feasible widths to prove optimality?
2. If so, *how* do we find a path and these widths?

In future lectures, we will answer (A) affirmatively, and provide an efficient algorithm for (B).

Lecture 18. Weak Duality

> 1. *The dual LP of* $\min\{c^T x : Ax \geq b, x \geq \mathbf{0}\}$ *is given by* $\max\{b^T y : A^T y \leq c, y \geq \mathbf{0}\}$.
> 2. *Weak Duality: If* $\boldsymbol{x}$ *is feasible for (P) and* $\boldsymbol{y}$ *is feasible for (D), then* $\mathbf{b}^T \boldsymbol{y} \leq \mathbf{c}^T \boldsymbol{x}$.
> 3. *The LP relaxation of an IP is obtained by dropping the integrality restriction.*
> 4. *The dual of the shortest path LP is given by*
>
> $$\begin{array}{ll} \max & \sum(y_U : \delta(U)\ s,t\text{-cut}\} \\ s.t. & \sum(y_U : e \in \delta(U)) \leq c_e \quad (e \in E) \\ & y \geq \mathbf{0} \end{array}$$

18.1 Dual of Primal LP

Example 18.1.1 The LP below is feasible (e.g., $x = (5, 13)^T$)):

$$\begin{array}{ll} \min & (2,3)x \\ s.t. & \\ & \begin{pmatrix} 2 & 1 \\ 1 & 1 \\ -1 & 1 \end{pmatrix} x \geq \begin{pmatrix} 20 \\ 18 \\ 8 \end{pmatrix} \\ & x \geq \mathbf{0} \end{array}$$

Can we find a good lower-bound on the objective value of the LP? Suppose that x is feasible for the given LP. Then it satisfies

$$\begin{pmatrix} 2 & 1 \\ 1 & 1 \\ -1 & 1 \end{pmatrix} x \geq \begin{pmatrix} 20 \\ 18 \\ 8 \end{pmatrix}.$$

Adding up LHS and RHS, x must also satisfy

$$\begin{array}{crl} & (2,1)x & \geq 20 \\ + & (1,1)x & \geq 18 \\ + & (-1,1)x & \geq 8 \\ \hline = & (2,3)x & \geq 46 \end{array}$$

Additionally, for any $y_1, y_2, y_3 \geq 0$, it satisfies

$$\begin{array}{crl} & y_1 \cdot (2,1)x & \geq y_1 \cdot 20 \\ + & y_2 \cdot (1,1)x & \geq y_2 \cdot 18 \\ + & y_3 \cdot (-1,1)x & \geq y_3 \cdot 8 \\ \hline = & (2y_1 + y_2 - y_3, y_1 + y_2 + y_3)x & \geq 20y_1 + 18y_2 + 8y_3 \end{array}$$

In other words, for any $y_1, y_2, y_3 \geq 0$, a feasible x for the LP satisfies

$$(y_1, y_2, y_3)\begin{pmatrix} 2 & 1 \\ 1 & 1 \\ -1 & 1 \end{pmatrix} x \geq (y_1, y_2, y_3)\begin{pmatrix} 20 \\ 18 \\ 8 \end{pmatrix}.$$

Let $y = (0, 2, 1)^T$, we obtain $(1, 3)x \geq 44$, or $0 \geq 44 - (1, 3)x$. Thus,

$$z(x) = (2, 3)x \geq (2, 3)x + 44 - (1, 3)x = 44 + (1, 0)x.$$

Given $x \geq \mathbf{0}$, it follows that $z(x) \geq 44$ for every feasible solution x. The solution $\bar{x} = (5, 13)^T$ yields a value of 49, so the optimal value of the LP is in the closed interval $[44, 49]$. Can we find a better lower bound on $z(x)$ for a feasible x?

We know that a feasible x satisfies

$$0 \geq (y_1, y_2, y_3)\begin{pmatrix} 20 \\ 18 \\ 8 \end{pmatrix} - (y_1, y_2, y_3)\begin{pmatrix} 2 & 1 \\ 1 & 1 \\ -1 & 1 \end{pmatrix} x$$

for any $y_1, y_2, y_3 \geq 0$. Therefore,

$$z(x) \geq (y_1, y_2, y_3)\begin{pmatrix} 20 \\ 18 \\ 8 \end{pmatrix} + \left((2, 3) - (y_1, y_2, y_3)\begin{pmatrix} 2 & 1 \\ 1 & 1 \\ -1 & 1 \end{pmatrix}\right) x.$$

We want the second term to be non-negative (because we are trying to find a lower bound for $z(x)$; making the second term negative would defeat the purpose). Since $x \geq \mathbf{0}$, this amounts to choosing y such that

$$(y_1, y_2, y_3)\begin{pmatrix} 2 & 1 \\ 1 & 1 \\ -1 & 1 \end{pmatrix} \leq (2, 3). \quad (\spadesuit)$$

Now with the same y, we also have

$$z(x) \geq (y_1, y_2, y_3)\begin{pmatrix} 20 \\ 18 \\ 8 \end{pmatrix}. \quad (\diamondsuit)$$

In other words, to find the best possible lower-bound on z, we need to find $y \geq \mathbf{0}$ such that $\spadesuit$ holds and the RHS of $\diamondsuit$ is maximized! We arrive at the following LP:

$$\begin{aligned} \max \quad & (20, 18, 8)x \\ s.t. \quad & \\ & \begin{pmatrix} 2 & 1 \\ 1 & 1 \\ -1 & 1 \end{pmatrix} y \leq (2, 3) \\ & y \geq \mathbf{0} \end{aligned}$$

Solving this gives $(\bar{y}_1, \bar{y}_2, \bar{y}_3) = (0, 5/2, 1/2)$ and the objective value is 49. Thus, there is no feasible solution x to the original LP that has an objective value smaller than 49. Since $x = (5, 13)^T$ is a feasible with value 49, it must be optimal! We now present a general argument.

Proposition 18.1.2 Suppose we are given an LP

$$\begin{array}{ll} \min & c^T x \\ s.t. & Ax \geq b \qquad (P) \\ & x \geq \mathbf{0} \end{array}$$

Then any feasible solution x must satisfy $y^T Ax \geq y^T b$ for $y \geq \mathbf{0}$ and hence $0 \geq y^T b - y^T Ax$. Therefore,

$$z(x) = c^T x \geq c^T x + y^T b - y^T Ax = y^T b + (c^T - y^T A)x.$$

If we also know that $A^T y \leq c$, then $x \geq \mathbf{0}$ implies that $z(x) \geq y^T b$. The best lower bound on $z(x)$ can thus be found by solving the following LP:

$$\begin{array}{ll} \max & b^T y \\ s.t. & A^T y \leq c \qquad (D) \\ & y \geq \mathbf{0} \end{array}$$

The LP (D) is called the **dual** of **primal** LP (P).

Theorem 18.1.3 [Weak Duality] If $\bar{x}$ is feasible for (P) and $\bar{y}$ is feasible for (D), then $b^T \bar{y} \leq c^T \bar{x}$.

Proof.

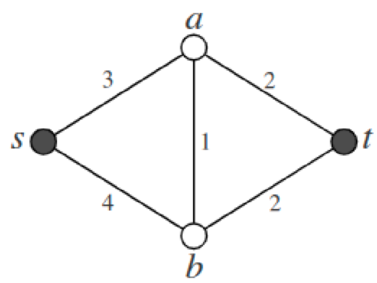
$$\begin{aligned} b^T \bar{y} &= \bar{y}^T b \\ &\leq \bar{y}^T (A\bar{x}) && \bar{y} \geq \mathbf{0} \wedge b \leq A\bar{x} \\ &= (A^T \bar{y})^T \bar{x} \\ &\leq c^T \bar{x} && \bar{x} \geq \mathbf{0} \wedge A^T y \leq c. \quad \square \end{aligned}$$

18.2 Lower-bounding the Length of s,t-Paths

Example 18.2.1 Recall that given a shortest path instance $G = (V, E)$, $s, t, \in V$, $c_E \geq 0$ for all $e \in E$, the shortest-path LP is

$$\begin{array}{ll} \min & \sum(c_e x_e : e \in E) \\ s.t. & \sum(x_e : e \in \delta(U)) \geq 1 \quad (U \subseteq V, s \in U, t \notin U) \\ & x \geq \mathbf{0}, x \in \mathbb{Z} \end{array}$$

Consider the following example:



$$\min \quad (3,4,1,2,2)x$$

$$\text{s.t.} \quad \begin{array}{c} \\ \{s\} \\ \{s,a\} \\ \{s,b\} \\ \{s,a,b\} \end{array} \begin{array}{c} \begin{array}{ccccc} sa & sb & ab & at & bt \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix} \end{array} x \geq \mathbb{1}$$

$$x \geq \mathbb{0}, x \text{ integer}$$

If P is an s,t-path, then letting $\bar{x}_e = 1$ iff e is an edge of P and 0 otherwise for all $e \in E$ would yield a feasible IP solution with an objective value $c(P)$. For example, path $P = sa, ab, bt \implies x = (1,0,1,0,1)^T$ with an objective value 6. As a remark, the optimal value of the shortest path IP is, at most, the length of a shortest s,t-path.

Now, if we drop the condition $x \in \mathbb{Z}$, the objective value could go down but not up (as the objective function would seek to minimize the value; taking decimal values could achieve this). The resulting LP is called the **linear programming relaxation** of the IP. Straight from Weak Duality theorem, we have that *the dual of (P) has optimal value no larger than of (P).*

The dual of (P) is given by

$$\max \quad \mathbb{1}^\top y$$

$$\text{s.t.} \quad \begin{array}{c} \\ sa \\ sb \\ ab \\ at \\ bt \end{array} \begin{array}{c} \begin{array}{cccc} \{s\} & \{s,a\} & \{s,b\} & \{s,a,b\} \end{array} \\ \begin{pmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix} \end{array} y \leq \begin{pmatrix} 3 \\ 4 \\ 1 \\ 2 \\ 2 \end{pmatrix}$$

$$y \geq \mathbb{0}$$

Note that the dual solutions assign the value $y_U \geq 0$ for every s,t-cut $\delta(U)$! Focus on the constraint for edge ab, i.e., $y_{\{s,a\}} + y_{\{s,b\}} \leq 1$: LHS is precisely the y-value assigned to the two s,t -cuts which contain ab! In other words, it is saying that adding the width assignments to s,t-cut containing ab should be no more than the length of edge ab! This is true for all inequalities here, which should remind you of the width assignment constraint $\sum(y_U : e \in \delta(U)) \leq c_e$.

In conclusion, y is feasible for the above LP iff it is a feasible width assignment for the s,t-cuts in the given shortest path instance.

Proposition 18.2.2 Here is the general argument for general shortest path instances. Given $G = (V,E)$, $s,t, \in V$, $c_E \geq 0$ for all $e \in E$, the LP is of the form

$$\begin{array}{lll} \min & c^T x & \\ s.t. & Ax \geq \mathbf{1} & \qquad (P) \\ & x \geq \mathbf{0} & \end{array}$$

- A has a column for every edge and a row for every s,t-cut $\delta(U)$,

- $A[U,e] = 1$ if $e \in \delta(U)$ and $\mathbf{0}$ otherwise.

Its dual is of the form

$$\begin{array}{lll} \max & \mathbf{1}^T y & \\ s.t. & A^T y \leq c & \qquad (D) \\ & y \geq \mathbf{0} & \end{array}$$

Note that the dual has a constraint for every edge $e \in E$; each constraint precisely correspond to what we have seen before: $\sum(y_U : e \in \delta(U)) \leq c_e$.

Remark 18.2.3 Feasible solutions to (D) correspond precisely to feasible width assignments. Weak duality implies that $\sum y_U$ is, at most, the length of a shortest s,t-path. (Again, think about the general graph vs cardinality graph example! It explains the inequality in $\sum(y_U : e \in \delta(U)) \leq c_e$.)

Lecture 19. Shortest Path Algorithm

> 1. *An arc is an ordered pair of vertices; a directed path (dipath) is a sequence of arcs.*
> 2. *The **slack** of an edge $e \in E$ is defined as* $\text{slack}_y(e) = c_e - \sum(y_U : e \in \delta(U)\ s, t\text{-cut})$.
> 3. *For the shortest path algorithm, start with $y_U = 0$ for all s, t-cut $\delta(U)$ and set $U = \{s\}$. In each iteration, we find the minimal slack, set y_U to that value, and add the new vertex to U. When the algorithm returns, we get a directed s, t-path and a feasible width assignment.*

19.1 Slack

Definition 19.1.1 An **arc** is an ordered pair of vertices. We denote an arc from u to v as $\overrightarrow{uv}$, and draw it as an arrow from u to v.

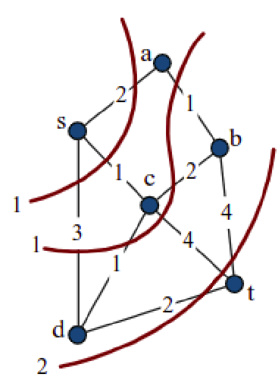
Definition 19.1.2 A **directed path** (**dipath**) is a **sequence of arcs** $\overrightarrow{v_1v_2}, \overrightarrow{v_2v_3}, \ldots, \overrightarrow{v_{k-1}v_k}$, where $\overrightarrow{v_iv_{i+1}}$ is an arc in the given graph and $v_i \neq v_j$ for all $i \neq j$. For example, $\overrightarrow{uv}, \overrightarrow{vw}, \overrightarrow{wx}$ is a u, x-dipath.

Definition 19.1.3 Let y be a feasible dual solution. The **slack** of an edge $e \in E$ is defined as

$$\text{slack}_y(e) = c_e - \sum(y_U : e \in \delta(U)\ s, t\text{-cut}).$$

Remark 19.1.4 Recall the width assignment constraint $\sum(y_U : e \in \delta(U)\ s, t\text{-cut}) \leq c_e$, i.e., the sum of widths of all cuts containing an edge e is no greater than c_e. If we think of c_e as some sort of resource and the widths as users, then the slack is like the portion of resource not allocated by the width assignment.

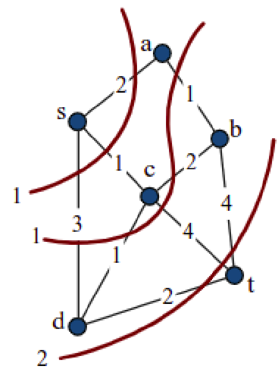
Example 19.1.5 For the dual y given below,



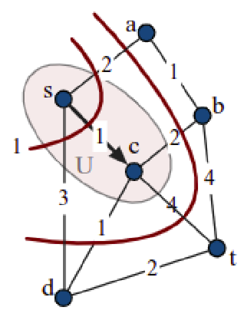
$$\begin{aligned} \text{slack}_y(sa) &= 2 - 1 = 1, \\ \text{slack}_y(sd) &= 3 - 1 - 1 = 1, \\ \text{slack}_y(ct) &= 4 - 1 - 2 = 1. \end{aligned}$$

19.2 Shortest Path Algorithm

Example 19.2.1 We present a step-by-step example finding the shortest s, t-path for the given graph.



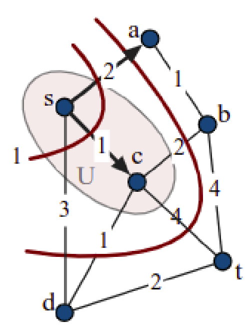
We start with the trivial dual $y = \mathbf{0}$. The simplest s,t-cut is $\delta(\{s\})$. The key strategy is to increase $y_{\{s\}}$ as much as we can while still maintaining feasibility (recall the constraint $\sum(y_U : e \in \delta(U)) \le c_e$). Let $y_{\{s\}} = 1$. This decreases the slack of sc to 0, so we replace sc by $\overrightarrow{sc}$.



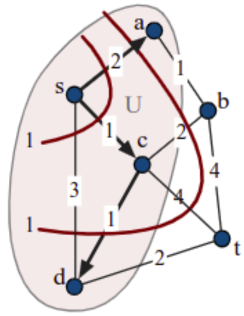
Next, we look at all vertices that are reachable from s via directed paths ($U = \{s, c\}$) and try to increase y_U. By how much we can increase it? The maximum increase possible for $y_{\{s,c\}}$ is determined by the (minimum of) slack of edges in $\delta(\{s,c\})$:

$$\begin{aligned}
\text{slack}_y(sa) &= 2 - 1 = 1 \\
\text{slack}_y(cb) &= 2 \\
\text{slack}_y(ct) &= 4 \\
\text{slack}_y(cd) &= 1 \\
\text{slack}_y(sd) &= 3 - 1 = 2
\end{aligned}$$

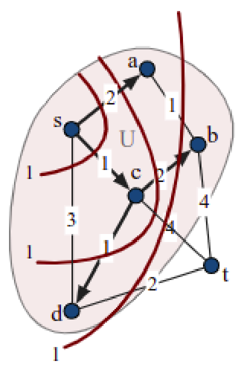
Note that edges cd and sa minimize slack. We pick one arbitrary, say sa, setting $y_U = \text{slack}_y(sa) = 1$ and converting sa into arc $\overrightarrow{sa}$. (The picture should add vertex a to the shaded region.)



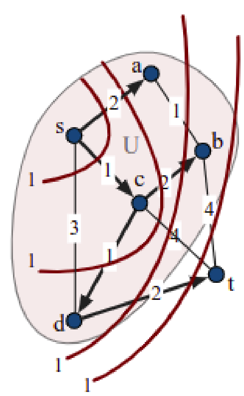
Now which vertices are reachable from s via directed paths? We get $U = \{s, a, c\}$. Again, we increase $y_{\{s,a,c\}}$ by as much as possible. Since the slack of cd is 0, we have $y_{\{s,a,c\}} = 0$. We then change cd to $\overrightarrow{cd}$ and let $U = \{s, a, c, d\}$ be the set of reachable vertices from s.



Again, we compute the slack of edges in $\delta(U)$. We find ab and cb both have slack 1, so we let $y_{\{s,a,c,d\}} = 1$, and add the equality arc $\overrightarrow{cb}$ and update $U = \{s, a, b, c, d\}$.



Finally, we compute the slack of edges in $\delta(U)$ and see edge dt has the minimum slack of 1. We let $y_{\{s,a,b,c,d\}} = 1$ and add the equality arc $\overrightarrow{dt}$.



We now have a directed s,t-path in our graph: $P = \overrightarrow{sc}, \overrightarrow{cd}, \overrightarrow{dt}$. It has a length of 4. We also have a feasible dual solution: $y_{\{s\}} = y_{\{s,c\}} = y_{\{s,a,c,d\}} = y_{\{s,a,b,c,d\}} = 1$ and $y_U = 0$ otherwise. Therefore, we know that path P is a shortest path!

Algorithm 19.2.2 The following algorithm helps us find a shortest s,t-path in a graph.

Algorithm: Shortest Path

Input: Graph $G = (V, E)$, costs $c_e \geq 0$ for all $e \in E$, $s, v \in V$ where $s \neq t$.

Output: A shortest s, t-path P.

$y_W := 0$ for all s, t-cuts $\delta(W)$. Set $U := \{s\}$
while $t \notin U$ do
    Let ab be an edge in $\delta(U)$ of smallest slack for y where $a \in U$, $b \notin U$
    $y_U := \text{slack}_y(ab)$
    $U := U \cup \{b\}$
    Change edge ab into an arc $\overrightarrow{a}b$
end while
return A directed s, t-path P

Lecture 20. Correctness

> 1. *An edge $uv \in E$ is called an equality edge if it has zero slack.*
> 2. *A cut $\delta(U)$ is said to be active for a dual solution y if $y_U = 0$.*
> 3. *Given an s,t-path P and a feasible dual solution y, P is shortest if all edges are equality edges and every active cut $\delta(U)$ has exactly one edge in P.*
> 4. *The correctness can be proved using five invariants: y feasible; all equality arcs; no entering arc for any active cut; existence of directed s,u-path $\forall u \in U$; all arcs have both ends in U.*

20.1 Shortest Path Characterization

Remark 20.1.1 In this lecture, we prove that both of the following yield an answer of true:

1. Will the algorithm *always* terminate?
2. Will it *always* find an s,t-path P whose length is equal to the value of a feasible dual solution?

Definition 20.1.2 Recall the *slack* of an edge $uv \in E$ for a feasible dual solution y is $c_{uv} - \sum(y_U : e \in \delta(U))$. We call an edge $uv \in E$ an **equality edge** if its *slack is zero.*

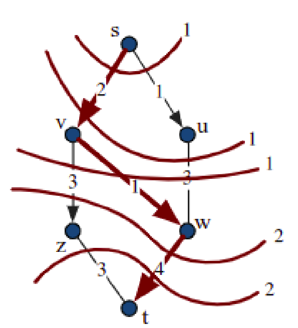
Definition 20.1.3 We call a cut $\delta(U)$ **active** for a dual solution y if $y_U > 0$.

Proposition 20.1.4 Let y be a feasible dual solution and P an s,t-path. Then P is a shortest path if

1. All edges on P are equality edges, i.e. $c_e = \sum(y_U : e \in \delta(U))$,
2. Every active cut $\delta(U)$ has *exactly* one edge of P, i.e., $|P \cap \delta(U)| = 1$ for all $\delta(U)$ if $y_U > 0$.

Proof. Suppose P and y satisfy both conditions. Then the length of the path satisfies the equality $\sum_{e\in P} c_e = \sum_{e\in P}(\sum(y_U : e \in \delta(U)))$ because every edge on P is an equality edge by (1). Consider RHS: how often does y_U for an active cut $\delta(U)$ appear on the RHS? Exactly the number of edges in P that is contained in $\delta(U)$. Thus, we can rewrite it as $\sum(y_U \cdot |P \cap \delta(U)| : \delta(U))$. But, by (2), $y_U > 0$ only if $|P \cap \delta(U)| = 1$. Hence, $\sum_{e\in P} c_e = \sum_U y_U$. $\square$

Example 20.1.5 Consider the following graph:



- vz is an equality edge $(1+2=3)$; zt is not $(2<3)$.
- If we have $U=\{s\}\to\{s,u\}\to\{s,v,u\}\to\{s,v,u,w\}\to\{s,v,u,w,z\}$, then $\delta(\{s,v,u\})$ is active and $\delta(\{s,v\})$ is not, as those cuts have $y_U>0$.
- Both conditions for proposition 20.1.4 are satisfied, thus $\overrightarrow{sv},\overrightarrow{vw},\overrightarrow{wt}$ is a shortest s,t-path.

20.2 Proof for Correctness

Remark 20.2.1 First, observe the algorithm always terminates since one vertex is added to U in every step and V is finite. We now prove the correctness of the algorithm.

Proposition 20.2.2 The Shortest Path Algorithm maintains throughout its execution if:

1. y is a feasible dual,
2. Arcs are equality arcs (i.e., always have 0 slack),
3. No active cut $\delta(U)$ has an *entering arc*, i.e., an arc wu with $w\notin U$ and $u\in U$,
4. For every $u\in U$ there is a directed s,u-path,
5. Arcs have both ends in U.

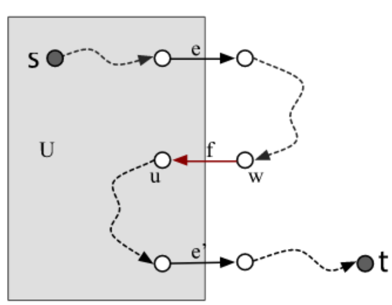
Remark 20.2.3 Before we start the proof, we need to recognize the implication of the proposition above. Suppose the invariants hold when the algorithm terminates. Then:

1. $t\in U$ and (4) implies there is a directed s,t-path P,
2. y is feasible by (1),
3. Arcs on P are equality arcs by (2).

Now, to show that P is the shortest path, we are left to show that every active cut contains exactly one arc from P. Once we have this, we have proved the correctness of the algorithm by the 20.1.4.

Lemma 20.2.4 If all 5 conditions for proposition 20.2.2 are satisfied, then every active cut $\delta(U)$ contains exactly one edge from path P.

Proof. For a contradiction, suppose $\delta(U)$ is an active cut containing more than one edge in P. Let e and e' be the first two edges on P that *leaves* $\delta(U)$.



Since there are two edges leaving U, there must also be an edge between e and e' that enters U; we call it f. But this contradicts (3)! The result follows. $\square$

Proof for proposition. For convenience, we provide a copy of the algorithm here.

Algorithm: Shortest Path
Input: Graph $G = (V, E)$, costs $c_e \geq 0$ for all $e \in E$, $s, v \in V$ where $s \neq t$.
Output: A shortest s, t-path P.
1. $y_W := 0$ for all s, t-cuts $\delta(W)$. Set $U := \{s\}$
2. while $t \notin U$ do
3. Let ab be an edge in $\delta(U)$ of smallest slack for y where $a \in U$, $b \notin U$
4. $y_U := \text{slack}_y(ab)$
5. $U := U \cup \{b\}$
6. Change edge ab into an arc $\overrightarrow{a\,b}$
7. end while
8. return A directed s, t-path P

First, it is trivial that all five conditions hold after line 1:

- y is indeed feasible because $c_e > 0$ for all e and $y = \mathbf{0}$.
- There are no arcs found yet.
- The only active arc is $\delta(U) = \delta(\{s\})$, which obviously has no entering arc.
- U only contains s, and there is a trivial s, s-path.
- There are no arcs found yet.

Now, suppose they hold before line 3 (before a new iteration). We want to show that they still hold after line 6 (after a new iteration).

Note that, during line 3 to 6, the only change to the dual solution is that y_U for the current U changes (line 4). Which of the dual constraints can be impacted by this change in the dual variable? Recall the constraints for the dual LP: $\sum(y_S : e \in \delta(S)) \leq c_e \quad (e \in E)$; y_U appears only on the LHS of edges e in the dual if $e \in \delta(U)$. Thus, the only constraint in the dual that might be affected by this change in dual variable are those constraints corresponding to edges in $\delta(U)$. Since we choose the smallest slack of any of these constraints to update y_U, no constraints corresponding to $\delta(U)$ gets violated by increasing y_U. Thus, y remains feasible after line 6; (1) holds.

Next, the constraint of the newly created arc holds with equality after the increase, thus, (2) continues to hold as the constraints for arcs have slack 0.

From inductive hypothesis (5), all old arcs have both ends in U. The new arc $\overrightarrow{ab}$ has tail in U and head outside U, so it is not an entering arc and thus (3) holds.

Suppose the new arc is $\overrightarrow{ab}$ where $a \in U$ and $b \notin U$. By inductive hypothesis (4), there exists a directed path P from s to a in U. By inductive hypothesis (5), any arc different from $\overrightarrow{ab}$ has both ends in U. Since $b \notin U$, it cannot be on P, and thus P together with ab is a directed s, b-path; (4) holds.

Finally, the new arc added is ab. As b is added to U, (5) holds.

The proof is complete. $\square$

Remark 20.2.5 In this and previous lecture, we saw that the shortest path algorithm always produces an s, t-path P and a feasible dual solution y. Moreover, the length of P always equals the objective value of y and hence, P must be a shortest s, t-path. Implicitly, we therefore showed that the shortest path LP always has an integer solution!