

Module 3: Duality through examples (Correctness Shortest Path Algorithm)

Recap: Shortest Path Algorithm

Previous lecture: we showed an algorithm for the shortest path problem that computes

- An s, t -path P

Shortest path LP:

$$\begin{aligned} \min \quad & \sum (c_e x_e : e \in E) \\ \text{s.t.} \quad & \sum (x_e : e \in \delta(S)) \geq 1 \\ & (\delta(S) \text{ } s, t\text{-cut}) \\ & x \geq 0 \end{aligned}$$

Shortest path dual:

$$\begin{aligned} \max \quad & \sum (y_S : \delta(S) \text{ } s, t\text{-cut}) \\ \text{s.t.} \quad & \sum (y_S : e \in \delta(S)) \leq c_e \\ & (e \in E) \\ & y \geq 0 \end{aligned}$$

Recap: Shortest Path Algorithm

Previous lecture: we showed an algorithm for the shortest path problem that computes

- An s, t -path P whose **characteristic vector**, x^P , is feasible for the shortest path LP, and

Shortest path LP:

$$\begin{aligned} \min \quad & \sum (c_e x_e : e \in E) \\ \text{s.t.} \quad & \sum (x_e : e \in \delta(S)) \geq 1 \\ & (\delta(S) \text{ } s, t\text{-cut}) \\ & x \geq 0 \end{aligned}$$

Shortest path dual:

$$\begin{aligned} \max \quad & \sum (y_S : \delta(S) \text{ } s, t\text{-cut}) \\ \text{s.t.} \quad & \sum (y_S : e \in \delta(S)) \leq c_e \\ & (e \in E) \\ & y \geq 0 \end{aligned}$$

Recap: Shortest Path Algorithm

Previous lecture: we showed an algorithm for the shortest path problem that computes

- An s, t -path P whose **characteristic vector**, x^P , is feasible for the shortest path LP, and
- a feasible solution, y , for the dual of the shortest path LP.

Shortest path LP:

$$\begin{aligned} \min \quad & \sum (c_e x_e : e \in E) \\ \text{s.t.} \quad & \sum (x_e : e \in \delta(S)) \geq 1 \\ & (\delta(S) \text{ } s, t\text{-cut}) \\ & x \geq 0 \end{aligned}$$

Shortest path dual:

$$\begin{aligned} \max \quad & \sum (y_S : \delta(S) \text{ } s, t\text{-cut}) \\ \text{s.t.} \quad & \sum (y_S : e \in \delta(S)) \leq c_e \\ & (e \in E) \\ & y \geq 0 \end{aligned}$$

Recap: Shortest Path Algorithm

Previous lecture: we showed an algorithm for the shortest path problem that computes

- An s, t -path P whose **characteristic vector**, x^P , is feasible for the shortest path LP, and
- a feasible solution, y , for the dual of the shortest path LP.

Important: $c^T x = \mathbb{1}^T y$

Shortest path LP:

$$\begin{aligned} \min \quad & \sum (c_e x_e : e \in E) \\ \text{s.t.} \quad & \sum (x_e : e \in \delta(S)) \geq 1 \\ & (\delta(S) \text{ } s, t\text{-cut}) \\ & x \geq 0 \end{aligned}$$

Shortest path dual:

$$\begin{aligned} \max \quad & \sum (y_S : \delta(S) \text{ } s, t\text{-cut}) \\ \text{s.t.} \quad & \sum (y_S : e \in \delta(S)) \leq c_e \\ & (e \in E) \\ & y \geq 0 \end{aligned}$$

Recap: Shortest Path Algorithm

Previous lecture: we showed an algorithm for the shortest path problem that computes

- An s, t -path P whose **characteristic vector**, x^P , is feasible for the shortest path LP, and
- a feasible solution, y , for the dual of the shortest path LP.

Important: $c^T x = \mathbb{1}^T y \rightarrow P$ is a shortest path!

Shortest path LP:

$$\begin{aligned} \min \quad & \sum (c_e x_e : e \in E) \\ \text{s.t.} \quad & \sum (x_e : e \in \delta(S)) \geq 1 \\ & (\delta(S) \text{ } s, t\text{-cut}) \\ & x \geq 0 \end{aligned}$$

Shortest path dual:

$$\begin{aligned} \max \quad & \sum (y_S : \delta(S) \text{ } s, t\text{-cut}) \\ \text{s.t.} \quad & \sum (y_S : e \in \delta(S)) \leq c_e \\ & (e \in E) \\ & y \geq 0 \end{aligned}$$

Recap: Shortest Path Algorithm

Previous lecture: we showed an algorithm for the shortest path problem that computes

- An s, t -path P whose **characteristic vector**, x^P , is feasible for the shortest path LP, and
- a feasible solution, y , for the dual of the shortest path LP.

Important: $c^T x = \mathbb{1}^T y \rightarrow P$ is a shortest path!

We will start this lecture with another example!

Shortest path LP:

$$\begin{aligned} \min \quad & \sum (c_e x_e : e \in E) \\ \text{s.t.} \quad & \sum (x_e : e \in \delta(S)) \geq 1 \\ & (\delta(S) \text{ } s, t\text{-cut}) \\ & x \geq 0 \end{aligned}$$

Shortest path dual:

$$\begin{aligned} \max \quad & \sum (y_S : \delta(S) \text{ } s, t\text{-cut}) \\ \text{s.t.} \quad & \sum (y_S : e \in \delta(S)) \leq c_e \\ & (e \in E) \\ & y \geq 0 \end{aligned}$$

Recall the algorithm we developed previously:

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

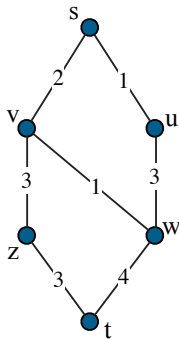
Recall the algorithm we developed previously:

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_w := 0$ for all st -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 \vec{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-



→ Run this on the example instance on the right.

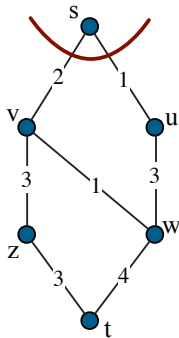
Recall the algorithm we developed previously:

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_w := 0$ for all st -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{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

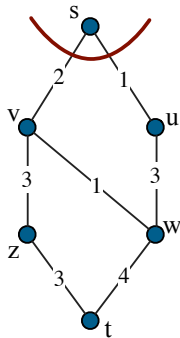


→ Run this on the example instance on the right.

Initially: $y = \mathbb{0}$ and $U = \{s\}$

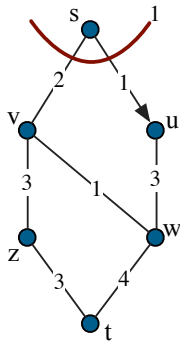
Initially: $y = \emptyset$ and $U = \{s\}$

Step 1 su edge with smallest slack in $\delta(U)$



Initially: $y = \emptyset$ and $U = \{s\}$

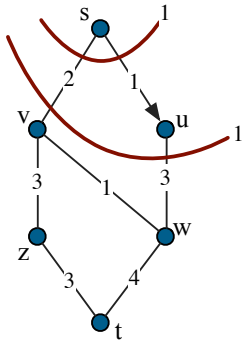
Step 1 su edge with smallest slack in $\delta(U)$
 \rightarrow increase y_U by 1



Initially: $y = \emptyset$ and $U = \{s\}$

Step 1 su edge with smallest slack in $\delta(U)$
 \rightarrow increase y_U by 1

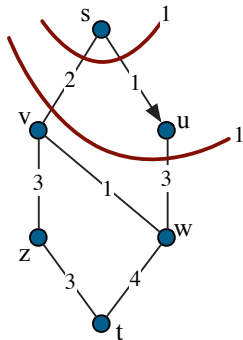
Step 2 **Now:** $U = \{s, u\}$
Slack-minimal edge is sv



Initially: $y = \emptyset$ and $U = \{s\}$

Step 1 su edge with smallest slack in $\delta(U)$
→ increase y_U by 1

Step 2 **Now:** $U = \{s, u\}$
Slack-minimal edge is sv
→ increase y_U by 1

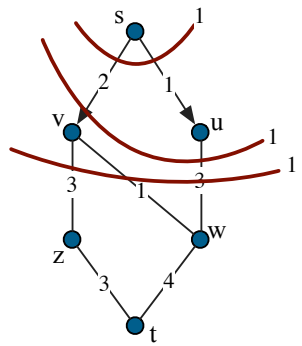


Initially: $y = \emptyset$ and $U = \{s\}$

Step 1 su edge with smallest slack in $\delta(U)$
→ increase y_U by 1

Step 2 **Now:** $U = \{s, u\}$
Slack-minimal edge is sv
→ increase y_U by 1

Step 3 $U = \{s, v, u\}$
Slack minimizer is vw

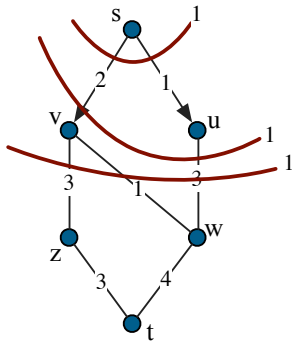


Initially: $y = 0$ and $U = \{s\}$

Step 1 su edge with smallest slack in $\delta(U)$
→ increase y_U by 1

Step 2 **Now:** $U = \{s, u\}$
Slack-minimal edge is sv
→ increase y_U by 1

Step 3 $U = \{s, v, u\}$
Slack minimizer is vw
→ increase y_U by 1



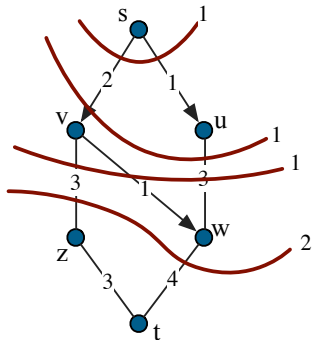
Initially: $y = \emptyset$ and $U = \{s\}$

Step 1 su edge with smallest slack in $\delta(U)$
 \rightarrow increase y_U by 1

Step 2 **Now:** $U = \{s, u\}$
Slack-minimal edge is sv
 \rightarrow increase y_U by 1

Step 3 $U = \{s, v, u\}$
Slack minimizer is vw
 \rightarrow increase y_U by 1

Step 4 $U = \{s, v, u, w\}$
Slack minimizer is vz



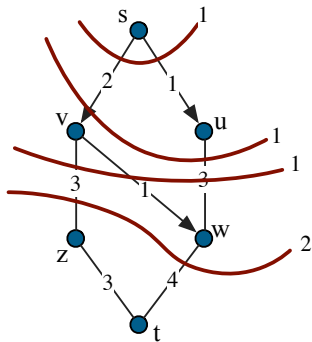
Initially: $y = \mathbb{0}$ and $U = \{s\}$

Step 1 su edge with smallest slack in $\delta(U)$
→ increase y_U by 1

Step 2 **Now:** $U = \{s, u\}$
Slack-minimal edge is sv
→ increase y_U by 1

Step 3 $U = \{s, v, u\}$
Slack minimizer is vw
→ increase y_U by 1

Step 4 $U = \{s, v, u, w\}$
Slack minimizer is vz
→ increase y_U by 2



Initially: $y = 0$ and $U = \{s\}$

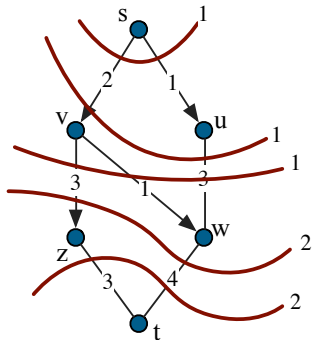
Step 1 su edge with smallest slack in $\delta(U)$
→ increase y_U by 1

Step 2 **Now:** $U = \{s, u\}$
Slack-minimal edge is sv
→ increase y_U by 1

Step 3 $U = \{s, v, u\}$
Slack minimizer is vw
→ increase y_U by 1

Step 4 $U = \{s, v, u, w\}$
Slack minimizer is vw
→ increase y_U by 2

Step 5 $U = \{s, v, u, w, z\}$
Slack minimizer is wt



Initially: $y = 0$ and $U = \{s\}$

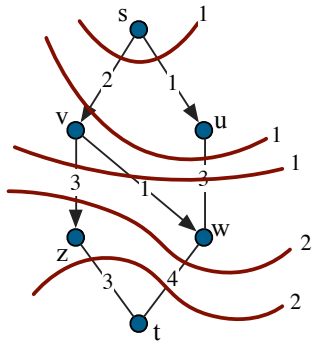
Step 1 su edge with smallest slack in $\delta(U)$
→ increase y_U by 1

Step 2 **Now:** $U = \{s, u\}$
Slack-minimal edge is sv
→ increase y_U by 1

Step 3 $U = \{s, v, u\}$
Slack minimizer is vw
→ increase y_U by 1

Step 4 $U = \{s, v, u, w\}$
Slack minimizer is vw
→ increase y_U by 2

Step 5 $U = \{s, v, u, w, z\}$
Slack minimizer is wt
→ increase y_U by 2



Initially: $y = 0$ and $U = \{s\}$

Step 1 su edge with smallest slack in $\delta(U)$

→ increase y_U by 1

Step 2 **Now:** $U = \{s, u\}$

Slack-minimal edge is sv

→ increase y_U by 1

Step 3 $U = \{s, v, u\}$

Slack minimizer is vw

→ increase y_U by 1

Step 4 $U = \{s, v, u, w\}$

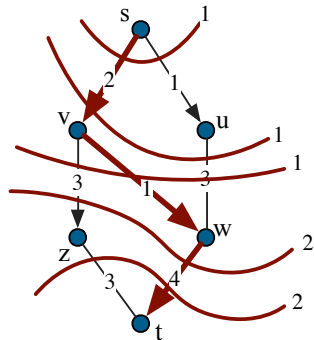
Slack minimizer is vz

→ increase y_U by 2

Step 5 $U = \{s, v, u, w, z\}$

Slack minimizer is wt

→ increase y_U by 2



Now: We have a directed s, t -path P of length 7,

Initially: $y = 0$ and $U = \{s\}$

Step 1 su edge with smallest slack in $\delta(U)$

→ increase y_U by 1

Step 2 **Now:** $U = \{s, u\}$

Slack-minimal edge is sv

→ increase y_U by 1

Step 3 $U = \{s, v, u\}$

Slack minimizer is vw

→ increase y_U by 1

Step 4 $U = \{s, v, u, w\}$

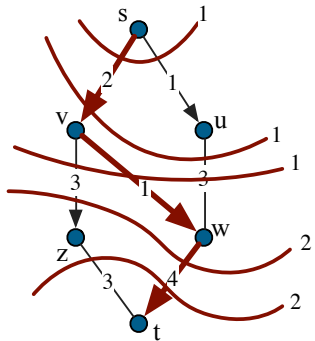
Slack minimizer is vz

→ increase y_U by 2

Step 5 $U = \{s, v, u, w, z\}$

Slack minimizer is wt

→ increase y_U by 2



Now: We have a directed s, t -path P of length 7, and a **dual feasible** solution of the same value!

Initially: $y = 0$ and $U = \{s\}$

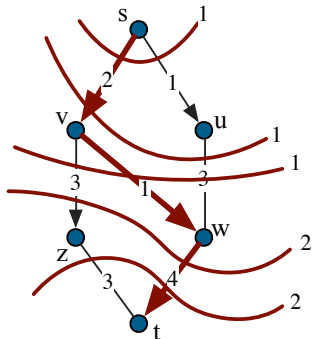
Step 1 su edge with smallest slack in $\delta(U)$
→ increase y_U by 1

Step 2 **Now:** $U = \{s, u\}$
Slack-minimal edge is sv
→ increase y_U by 1

Step 3 $U = \{s, v, u\}$
Slack minimizer is vw
→ increase y_U by 1

Step 4 $U = \{s, v, u, w\}$
Slack minimizer is vz
→ increase y_U by 2

Step 5 $U = \{s, v, u, w, z\}$
Slack minimizer is wt
→ increase y_U by 2

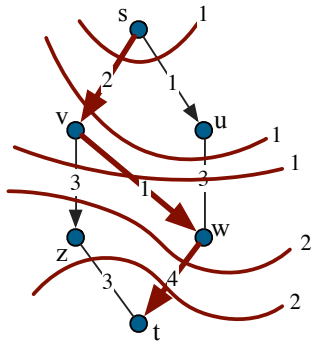


Now: We have a directed s, t -path P of length 7, and a **dual feasible** solution of the same value!

→ P is a **shortest** path!

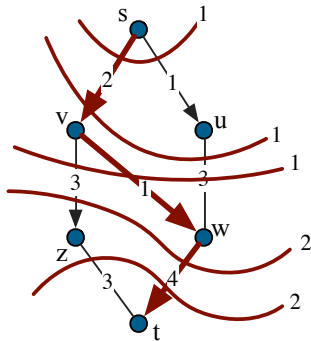
Question

Will the algorithm *always* terminate?



Question

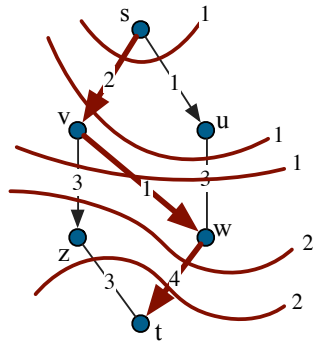
Will the algorithm **always** terminate? Will it **always** find an s, t -path P whose length is equal to the value of a feasible dual solution?



Question

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

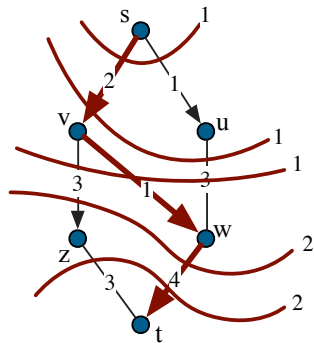
This lecture: We will show the answers to the above are yes & yes!



Revisited: Shortest Path Optimality Conditions

Recall: the **slack** of an edge $uv \in E$ for a feasible dual solution y is

$$c_{uv} - \sum (y_U : e \in \delta(U))$$

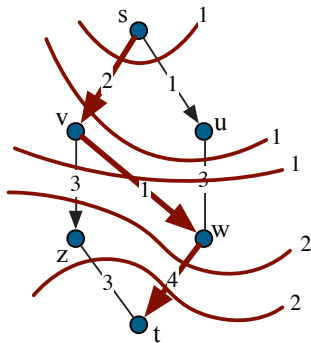


Revisited: Shortest Path Optimality Conditions

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 0**.



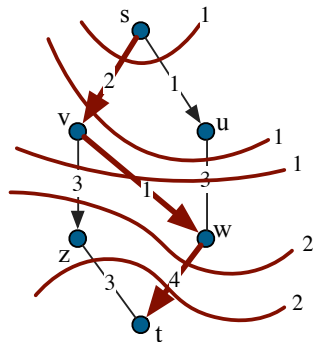
Revisited: Shortest Path Optimality Conditions

Recall: the **slack** of an edge $uv \in E$ for a feasible dual solution y is

$$c_{uv} - \sum (y_U : e \in \delta(U))$$

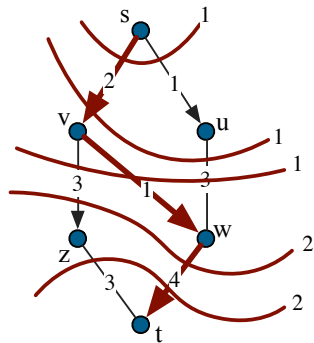
We call an edge $uv \in E$ an **equality edge** if its **slack is 0**.

Example: edge vz is an equality edge, and zt is not!



Revisited: Shortest Path Optimality Conditions

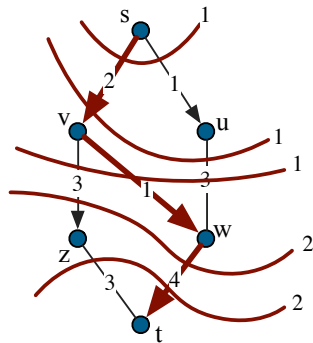
We will also call a cut $\delta(U)$ **active** for a dual solution y if $y_U > 0$.



Revisited: Shortest Path Optimality Conditions

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

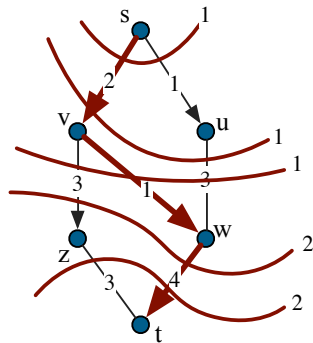
Example: $\delta(\{s, v, u\})$ is active, while $\delta(\{s, v\})$ is not!



Revisited: Shortest Path Optimality Conditions

Proposition

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

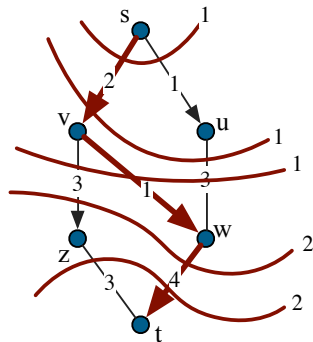


Revisited: Shortest Path Optimality Conditions

Proposition

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

- (i) all edges on P are equality edges, and

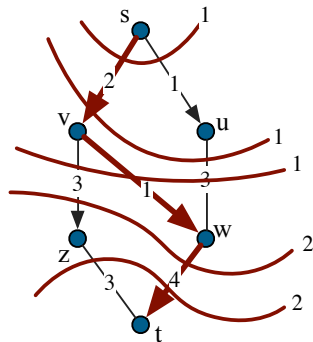


Revisited: Shortest Path Optimality Conditions

Proposition

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

- (i) all edges on P are equality edges, and
- (ii) every active cut $\delta(U)$ has exactly one edge of P .



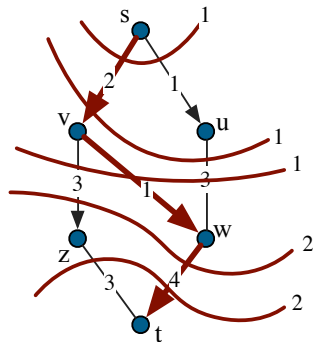
Revisited: Shortest Path Optimality Conditions

Proposition

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

- (i) all edges on P are equality edges, and
- (ii) every active cut $\delta(U)$ has exactly one edge of P .

Note: Both conditions are satisfied in the example on the right.



Revisited: Shortest Path Optimality Conditions

Proposition

Let y be a feasible dual solution, and P an s, t -path.

P is a shortest path if

- (i) all edges on P are equality edges, and
- (ii) every active cut $\delta(U)$ has exactly one edge of P .

Proof: Let's suppose that P and y satisfy (i) and (ii) of the proposition.

Revisited: Shortest Path Optimality Conditions

Proposition

Let y be a feasible dual solution, and P an s, t -path.

P is a shortest path if

- (i) all edges on P are equality edges, and
- (ii) every active cut $\delta(U)$ has exactly one edge of P .

Proof: Let's suppose that P and y satisfy (i) and (ii) of the proposition. Then,

$$\sum_{e \in P} c_e = \sum_{e \in P} \left(\sum_{U : e \in \delta(U)} y_U \right)$$

because every edge on P is an equality edge by (i).

Revisited: Shortest Path Optimality Conditions

Proposition

Let y be a feasible dual solution, and P and s, t -path.

P is a shortest path if

- (i) all edges on P are equality edges, and
- (ii) every active cut $\delta(U)$ has exactly one edge of P .

Proof: Let's suppose that P and y satisfy (i) and (ii) of the proposition. Then,

$$\sum_{e \in P} c_e = \sum_{e \in P} \left(\sum (y_U : e \in \delta(U)) \right)$$

because every edge on P is an equality edge by (i). The right-hand side equals

$$\sum (y_U \cdot |P \cap \delta(U)| : \delta(U))$$

Revisited: Shortest Path Optimality Conditions

Proposition

Let y be a feasible dual solution, and P and s, t -path.

P is a shortest path if

- (i) all edges on P are equality edges, and
- (ii) every active cut $\delta(U)$ has exactly one edge of P .

Note: Both conditions are satisfied in the example on the right!

Proof: Let's suppose that P and y satisfy (i) and (ii) of the proposition. Then,

$$\sum_{e \in P} c_e = \sum_{e \in P} \left(\sum_{U : e \in \delta(U)} y_U \right)$$

because every edge on P is an equality edge by (i). The right-hand side equals

$$\sum_U (y_U \cdot |P \cap \delta(U)| : \delta(U))$$

But, by (ii), $y_U > 0$ only if $|P \cap \delta(U)| = 1$. Hence:

$$\sum_{e \in P} c_e = \sum_U y_U$$

□

Correctness of the Shortest Path Algorithm

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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 \vec{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

Note: The algorithm terminates since one vertex is added to U in every step and V is finite.

Correctness of the Shortest Path Algorithm

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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 \vec{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

Note: The algorithm terminates since one vertex is added to U in every step and V is finite.

It suffices to show:

Proposition

The Shortest Path Algorithm maintains throughout its execution that:

(I1) y is a feasible dual,

Correctness of the Shortest Path Algorithm

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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 \vec{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

Note: The algorithm terminates since one vertex is added to U in every step and V is finite.

It suffices to show:

Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),

Correctness of the Shortest Path Algorithm

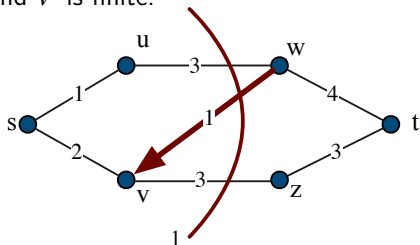
Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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 \vec{ab}
- 7: **end while**
- 8: **return** A directed st -path P .

Note: The algorithm terminates since one vertex is added to U in every step and V is finite.



It suffices to show:

Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),
- (I3) no active cut $\delta(U)$ has an **entering arc**: an arc wu with $w \notin U$, and $u \in U$,

Correctness of the Shortest Path Algorithm

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

1: $y_W := 0$ for all st -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: $y_U := \text{slack}_y(ab)$

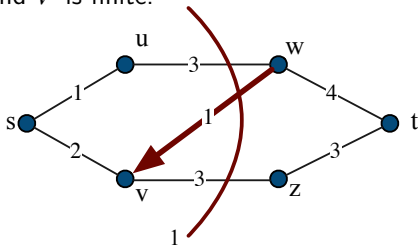
5: $U := U \cup \{b\}$

6: change edge ab into an arc \vec{ab}

7: **end while**

8: **return** A directed st -path P .

Note: The algorithm terminates since one vertex is added to U in every step and V is finite.



It suffices to show:

Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),
- (I3) no active cut $\delta(U)$ has an **entering arc**: an arc wu with $w \notin U$, and $u \in U$,
- (I4) for every $u \in U$ there is a directed s, u -path, and

Correctness of the Shortest Path Algorithm

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

1: $y_W := 0$ for all st -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: $y_U := \text{slack}_y(ab)$

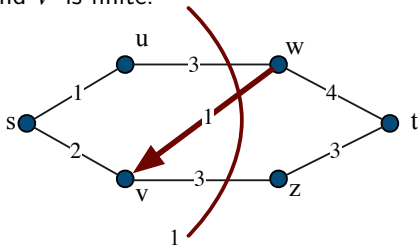
5: $U := U \cup \{b\}$

6: change edge ab into an arc \vec{ab}

7: **end while**

8: **return** A directed st -path P .

Note: The algorithm terminates since one vertex is added to U in every step and V is finite.



It suffices to show:

Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),
- (I3) no active cut $\delta(U)$ has an **entering arc**: an arc wu with $w \notin U$, and $u \in U$,
- (I4) for every $u \in U$ there is a directed s, u -path, and
- (I5) arcs have both ends in U .

Correctness of the Shortest Path Algorithm

Suppose the invariants hold when the algorithm terminates. Then:

Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),
- (I3) no active cut $\delta(U)$ has an **entering arc**: an arc wu with $w \notin U$, and $u \in U$,
- (I4) for every $u \in U$ there is a directed s, u -path, and
- (I5) arcs have both ends in U .

Correctness of the Shortest Path Algorithm

Suppose the invariants hold when the algorithm terminates. Then:

- $t \in U$ and (I4) implies that there is a directed s, t -path P ,

Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),
- (I3) no active cut $\delta(U)$ has an entering arc: an arc wu with $w \notin U$, and $u \in U$,
- (I4) for every $u \in U$ there is a directed s, u -path, and
- (I5) arcs have both ends in U .

Correctness of the Shortest Path Algorithm

Suppose the invariants hold when the algorithm terminates. Then:

- $t \in U$ and (I4) implies that there is a directed s, t -path P ,
- y is feasible by (I1), and

Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),
- (I3) no active cut $\delta(U)$ has an entering arc: an arc wu with $w \notin U$, and $u \in U$,
- (I4) for every $u \in U$ there is a directed s, u -path, and
- (I5) arcs have both ends in U .

Correctness of the Shortest Path Algorithm

Suppose the invariants hold when the algorithm terminates. Then:

- $t \in U$ and (I4) implies that there is a directed s, t -path P ,
- y is feasible by (I1), and
- arcs on P are equality arcs by (I2)

Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),
- (I3) no active cut $\delta(U)$ has an entering arc: an arc wu with $w \notin U$, and $u \in U$,
- (I4) for every $u \in U$ there is a directed s, u -path, and
- (I5) arcs have both ends in U .

Correctness of the Shortest Path Algorithm

Suppose the invariants hold when the algorithm terminates. Then:

- $t \in U$ and (I4) implies that there is a directed s, t -path P ,
- y is feasible by (I1), and
- arcs on P are equality arcs by (I2)

To show: $\delta(U)$ active $\rightarrow P$ has exactly one edge in $\delta(U)$.

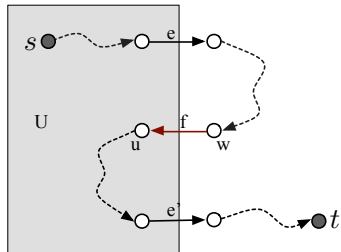
Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),
- (I3) no active cut $\delta(U)$ has an entering arc: an arc wu with $w \notin U$, and $u \in U$,
- (I4) for every $u \in U$ there is a directed s, u -path, and
- (I5) arcs have both ends in U .

Correctness of the Shortest Path Algorithm

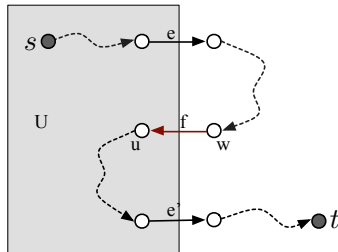
For a contradiction suppose $\delta(U)$ active
and P has more than one edge in $\delta(U)$



Correctness of the Shortest Path Algorithm

For a contradiction suppose $\delta(U)$ active and P has more than one edge in $\delta(U)$

Let e and e' be the first two edges on P that leave $\delta(U)$.

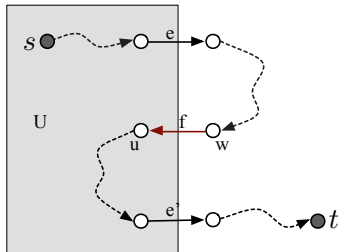


Correctness of the Shortest Path Algorithm

For a contradiction suppose $\delta(U)$ active and P has more than one edge in $\delta(U)$

Let e and e' be the first two edges on P that leave $\delta(U)$.

Then, there must also be an arc f on P that enters U ,



Correctness of the Shortest Path Algorithm

For a contradiction suppose $\delta(U)$ active and P has more than one edge in $\delta(U)$

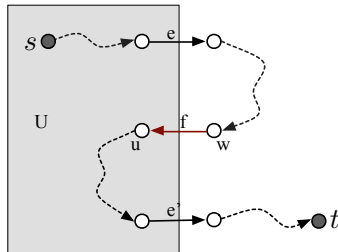
Let e and e' be the first two edges on P that leave $\delta(U)$.

Then, there must also be an arc f on P that enters U , but this contradicts (I3)!

Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I3) no active cut $\delta(U)$ has an entering arc: an arc wu with $w \notin U$, and $u \in U$



Correctness of the Shortest Path Algorithm

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

Let's now prove the proposition!

Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),
- (I3) no active cut $\delta(U)$ has an **entering, arc**: an arc wu with $w \notin U$, and $u \in U$,
- (I4) for every $u \in U$ there is a directed s, u -path, and
- (I5) arcs have both ends in U .

Correctness of the Shortest Path Algorithm

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

Let's now prove the proposition!

Trivial: (I1) – (I5) hold after Step 1.

Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),
- (I3) no active cut $\delta(U)$ has an **entering, arc**: an arc wu with $w \notin U$, and $u \in U$,
- (I4) for every $u \in U$ there is a directed s, u -path, and
- (I5) arcs have both ends in U .

Correctness of the Shortest Path Algorithm

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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 \vec{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

Let's now prove the proposition!

Trivial: (I1) – (I5) hold after Step 1.

Suppose (I1) – (I5) hold before Step 3.

Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),
- (I3) no active cut $\delta(U)$ has an **entering, arc**: an arc wu with $w \notin U$, and $u \in U$,
- (I4) for every $u \in U$ there is a directed s, u -path, and
- (I5) arcs have both ends in U .

Correctness of the Shortest Path Algorithm

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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 \vec{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

Let's now prove the proposition!

Trivial: (I1) – (I5) hold after Step 1.

Suppose (I1) – (I5) hold before Step 3.

We will show that they also hold after Step 6.

Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),
- (I3) no active cut $\delta(U)$ has an **entering, arc**: an arc wu with $w \notin U$, and $u \in U$,
- (I4) for every $u \in U$ there is a directed s, u -path, and
- (I5) arcs have both ends in U .

(I1) y is Dual Feasible

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

1: $y_W := 0$ for all st -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 \vec{ab}

7: **end while**

8: **return** A directed st -path P .

Shortest path dual:

$$\begin{aligned} \max \quad & \sum (y_S : \delta(S) \text{ } s, t\text{-cut}) \\ \text{s.t.} \quad & \sum (y_S : e \in \delta(S)) \leq c_e \\ & \quad \quad \quad (e \in E) \\ & y \geq 0 \end{aligned}$$

(I1) y is Dual Feasible

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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 \vec{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

Shortest path dual:

$$\begin{aligned} \max \quad & \sum (y_S : \delta(S) \text{ } s, t\text{-cut}) \\ \text{s.t.} \quad & \sum (y_S : e \in \delta(S)) \leq c_e \\ & \qquad \qquad \qquad (e \in E) \\ & y \geq 0 \end{aligned}$$

Note: In Step 3-6, only y_U for the current U changes.

(I1) y is Dual Feasible

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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 \vec{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

Shortest path dual:

$$\begin{aligned} \max \quad & \sum (y_S : \delta(S) \text{ } s, t\text{-cut}) \\ \text{s.t.} \quad & \sum (y_S : e \in \delta(S)) \leq c_e \\ & \qquad \qquad \qquad (e \in E) \\ & y \geq 0 \end{aligned}$$

Note: In Step 3-6, only y_U for the current U changes.

y_U appears **only** on the left-hand sides of constraints for edges in $\delta(U)$.

(I1) y is Dual Feasible

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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 \vec{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

Shortest path dual:

$$\begin{aligned} \max \quad & \sum (y_S : \delta(S) \text{ s, t-cut}) \\ \text{s.t.} \quad & \sum (y_S : e \in \delta(S)) \leq c_e \\ & \qquad \qquad \qquad (e \in E) \\ & y \geq 0 \end{aligned}$$

Note: In Step 3-6, only y_U for the current U changes.

y_U appears **only** on the left-hand sides of constraints for edges in $\delta(U)$.

The smallest **slack** of any of these constraints is precisely the increase in y_U .

(I1) y is Dual Feasible

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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 \vec{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

Shortest path dual:

$$\begin{aligned} \max \quad & \sum (y_S : \delta(S) \text{ } s, t\text{-cut}) \\ \text{s.t.} \quad & \sum (y_S : e \in \delta(S)) \leq c_e \\ & \qquad \qquad \qquad (e \in E) \\ & y \geq 0 \end{aligned}$$

Note: In Step 3-6, only y_U for the current U changes.

y_U appears **only** on the left-hand sides of constraints for edges in $\delta(U)$.

The smallest **slack** of any of these constraints is precisely the increase in y_U .

\longrightarrow y remains feasible!

(I1) y is Dual Feasible

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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 \vec{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

Shortest path dual:

$$\begin{aligned} \max \quad & \sum (y_S : \delta(S) \text{ } s, t\text{-cut}) \\ \text{s.t.} \quad & \sum (y_S : e \in \delta(S)) \leq c_e \\ & \qquad \qquad \qquad (e \in E) \\ & y \geq 0 \end{aligned}$$

Note: In Step 3-6, only y_U for the current U changes.

y_U appears **only** on the left-hand sides of constraints for edges in $\delta(U)$.

The smallest **slack** of any of these constraints is precisely the increase in y_U .

\longrightarrow y remains feasible!

Also: The constraint of the newly created arc holds with equality after the increase

(I1) y is Dual Feasible

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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 \vec{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

Shortest path dual:

$$\begin{aligned} \max \quad & \sum (y_S : \delta(S) \text{ } s, t\text{-cut}) \\ \text{s.t.} \quad & \sum (y_S : e \in \delta(S)) \leq c_e \\ & \qquad \qquad \qquad (e \in E) \\ & y \geq 0 \end{aligned}$$

Note: In Step 3-6, only y_U for the current U changes.

y_U appears **only** on the left-hand sides of constraints for edges in $\delta(U)$.

The smallest **slack** of any of these constraints is precisely the increase in y_U .

→ y remains feasible!

Also: The constraint of the newly created arc holds with equality after the increase

→ (I2) continues to hold and constraints for arcs have slack 0.

Correctness of the Shortest Path Algorithm

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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 \vec{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

- the only new active cut created is $\delta(U)$

Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),
- (I3) no active cut $\delta(U)$ has an **entering arc**: an arc wu with $w \notin U$, and $u \in U$,
- (I4) for every $u \in U$ there is a directed s, u -path, and
- (I5) arcs have both ends in U .

Correctness of the Shortest Path Algorithm

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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 \vec{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

- the only new active cut created is $\delta(U)$
- (I5) \longrightarrow all old arcs have both ends in U

Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),
- (I3) no active cut $\delta(U)$ has an **entering arc**: an arc wu with $w \notin U$, and $u \in U$,
- (I4) for every $u \in U$ there is a directed s, u -path, and
- (I5) arcs have both ends in U .

Correctness of the Shortest Path Algorithm

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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 \vec{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

- the only new active cut created is $\delta(U)$
- (I5) \longrightarrow all old arcs have both ends in U
- one new arc has tail in U , and head outside U

Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),
- (I3) no active cut $\delta(U)$ has an **entering arc**: an arc wu with $w \notin U$, and $u \in U$,
- (I4) for every $u \in U$ there is a directed s, u -path, and
- (I5) arcs have both ends in U .

Correctness of the Shortest Path Algorithm

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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 \vec{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

- the only new active cut created is $\delta(U)$
- (I5) \longrightarrow all old arcs have both ends in U
- one new arc has tail in U , and head outside U

\longrightarrow (I3) holds after Step 6

Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),
- (I3) no active cut $\delta(U)$ has an **entering arc**: an arc wu with $w \notin U$, and $u \in U$,
- (I4) for every $u \in U$ there is a directed s, u -path, and
- (I5) arcs have both ends in U .

Correctness of the Shortest Path Algorithm

Note: Algorithm adds arc ab in current step, and (I4) implies that there is a directed s, a -path P .

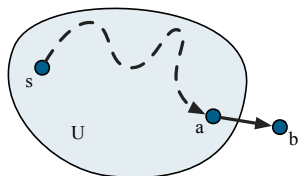
Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),
- (I3) no active cut $\delta(U)$ has an **entering arc**: an arc wu with $w \notin U$, and $u \in U$,
- (I4) for every $u \in U$ there is a directed s, u -path, and
- (I5) arcs have both ends in U .

Correctness of the Shortest Path Algorithm

Note: Algorithm adds arc ab in current step, and (I4) implies that there is a directed s, a -path P .



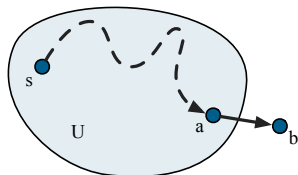
Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),
- (I3) no active cut $\delta(U)$ has an **entering arc**: an arc wu with $w \notin U$, and $u \in U$,
- (I4) for every $u \in U$ there is a directed s, u -path, and
- (I5) arcs have both ends in U .

Correctness of the Shortest Path Algorithm

Note: Algorithm adds arc ab in current step, and (I4) implies that there is a directed s, a -path P .



(I5) \longrightarrow arcs different from ab have both ends in U

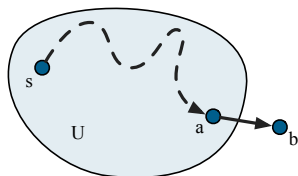
Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),
- (I3) no active cut $\delta(U)$ has an **entering arc**: an arc wu with $w \notin U$, and $u \in U$,
- (I4) for every $u \in U$ there is a directed s, u -path, and
- (I5) arcs have both ends in U .

Correctness of the Shortest Path Algorithm

Note: Algorithm adds arc ab in current step, and (I4) implies that there is a directed s, a -path P .



- (I5) \longrightarrow arcs different from ab have both ends in U
 \longrightarrow since b is outside U , it cannot be on P , and thus, P together with ab is a directed s, b -path

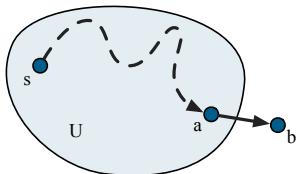
Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),
- (I3) no active cut $\delta(U)$ has an **entering arc**: an arc wu with $w \notin U$, and $u \in U$,
- (I4) for every $u \in U$ there is a directed s, u -path, and
- (I5) arcs have both ends in U .

Correctness of the Shortest Path Algorithm

Note: Algorithm adds arc ab in current step, and (I4) implies that there is a directed s, a -path P .



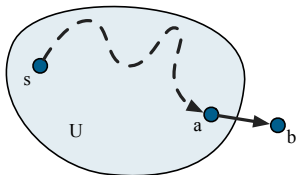
- (I5) \rightarrow arcs different from ab have both ends in U
 \rightarrow since b is outside U , it cannot be on P , and thus, P together with ab is a directed s, b -path
 \rightarrow (I4) holds at the end of loop

Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),
- (I3) no active cut $\delta(U)$ has an **entering arc**: an arc wu with $w \notin U$, and $u \in U$,
- (I4) for every $u \in U$ there is a directed s, u -path, and
- (I5) arcs have both ends in U .

Correctness of the Shortest Path Algorithm



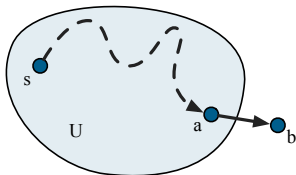
Finally, the only new arc added is ab . As b is added to U , (I5) continues to hold.

Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),
- (I3) no active cut $\delta(U)$ has an **entering arc**: an arc wu with $w \notin U$, and $u \in U$,
- (I4) for every $u \in U$ there is a directed s, u -path, and
- (I5) arcs have both ends in U .

Correctness of the Shortest Path Algorithm



Finally, the only new arc added is ab . As b is added to U , (I5) continues to hold.

We are now done!

Proposition

The Shortest Path Algorithm maintains throughout its execution that:

- (I1) y is a feasible dual,
- (I2) arcs are equality arcs (i.e., have 0 slack),
- (I3) no active cut $\delta(U)$ has an **entering arc**: an arc wu with $w \notin U$, and $u \in U$,
- (I4) for every $u \in U$ there is a directed s, u -path, and
- (I5) arcs have both ends in U .

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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 \vec{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

Recap

- We saw that the shortest path algorithm

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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 \vec{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

Recap

- We saw that the shortest path algorithm
 - (i) always produces an s, t -path P , and

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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 \vec{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

Recap

- We saw that the shortest path algorithm
 - (i) always produces an s, t -path P , and
 - (ii) a feasible dual solution y .

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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 \vec{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

Recap

- We saw that the shortest path algorithm
 - (i) always produces an s, t -path P , and
 - (ii) a feasible dual solution y .
- Moreover, the length of P **equals** the objective value of y , and hence, P must be a shortest s, t -path.

Algorithm 3.2 Shortest path.

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

Output: A shortest st -path P

- 1: $y_W := 0$ for all st -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 \vec{ab}
 - 7: **end while**
 - 8: **return** A directed st -path P .
-

Recap

- We saw that the shortest path algorithm
 - (i) always produces an s, t -path P , and
 - (ii) a feasible dual solution y .
- Moreover, the length of P 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 optimal integer solution!