graph algorithms 2

minimum spanning tree and shortest paths

def: *weighted graph* $G = (V, E, w)$

- (V, E) undirected graph
- $w: E \rightarrow R^{+}$ nonnegative edge weights

def: *weighted graph* $G = (V, E, w)$

- (V, E) undirected graph
- $w: E \to R^+$ nonnegative edge weights

(V, T) spanning tree

$w(T) = \Sigma_{e \in T} w(e)$ *weight of T*

minimum spanning tree

def: *weighted graph* $G = (V, E, w)$

- (V, E) undirected graph
- $w: E \to R^+$ nonnegative edge weights

(V, T) spanning tree

$w(T) = \Sigma_{e \in T} w(e)$ *weight of T*

algorithm

- input: weighted connected graph
- output: edges T of a spanning tree with minimal weight

def: *weighted graph* $G = (V, E, w)$

- (V, E) undirected graph
- $w: E \rightarrow R^+$ nonnegative edge weights

(V, T) spanning tree

$w(T) = \Sigma_{e \in T} w(e)$ *weight of T*

def: *cut* $C = (V', V'')$ with

- $V' \cup V'' = V$
- $V' \cap V'' = \emptyset$

cuts

algorithm

- input: weighted connected graph
- output: edges T of a spanning tree with minimal weight

cuts

def: *weighted graph* $G = (V, E, w)$

- (V, E) undirected graph
- $w : E \rightarrow R^+$ nonnegative edge weights

(V, T) spanning tree

$w(T) = \Sigma_{e \in T} w(e)$ *weight of T*

def: *cut* $C = (V', V'')$ with

- $V' \cup V'' = V$
- $V' \cap V'' = \emptyset$

$T' \subset E$ *respects* cut C if

no edge in T' crosses the cut

$\{u, v\} \in T' \rightarrow u, v \in V' \vee u, v \in V''$

algorithm

- input: weighted connected graph
- output: edges T of a spanning tree with minimal weight

def: *weighted graph* $G = (V, E, w)$

- (V, E) undirected graph
- $w: E \rightarrow R^{+}$ nonnegative edge weights

(V, T) spanning tree

$w(T) = \Sigma_{e \in T} w(e)$ *weight of T*

def: *cut* $C = (V', V'')$ with

- $V' \cup V'' = V$
- $V' \cap V'' = \emptyset$

$T' \subset E$ *respects* cut C if

no edge in T' crosses the cut

$\{u, v\} \in T' \rightarrow u, v \in V' \vee u, v \in V''$

Lemma 3. *Let $T' \subseteq E$ be a set of edges contained in the edges of a minimum spanning tree (V,T) of graph $G = (V,E,w)$.*

$$T' \subseteq T$$

Assume T' respects cut C and e is an edge of minimum weight cossing cut C. Then there is a minimum spanning tree (V,R) of G where

$$T' \cup \{e\} \subseteq R$$

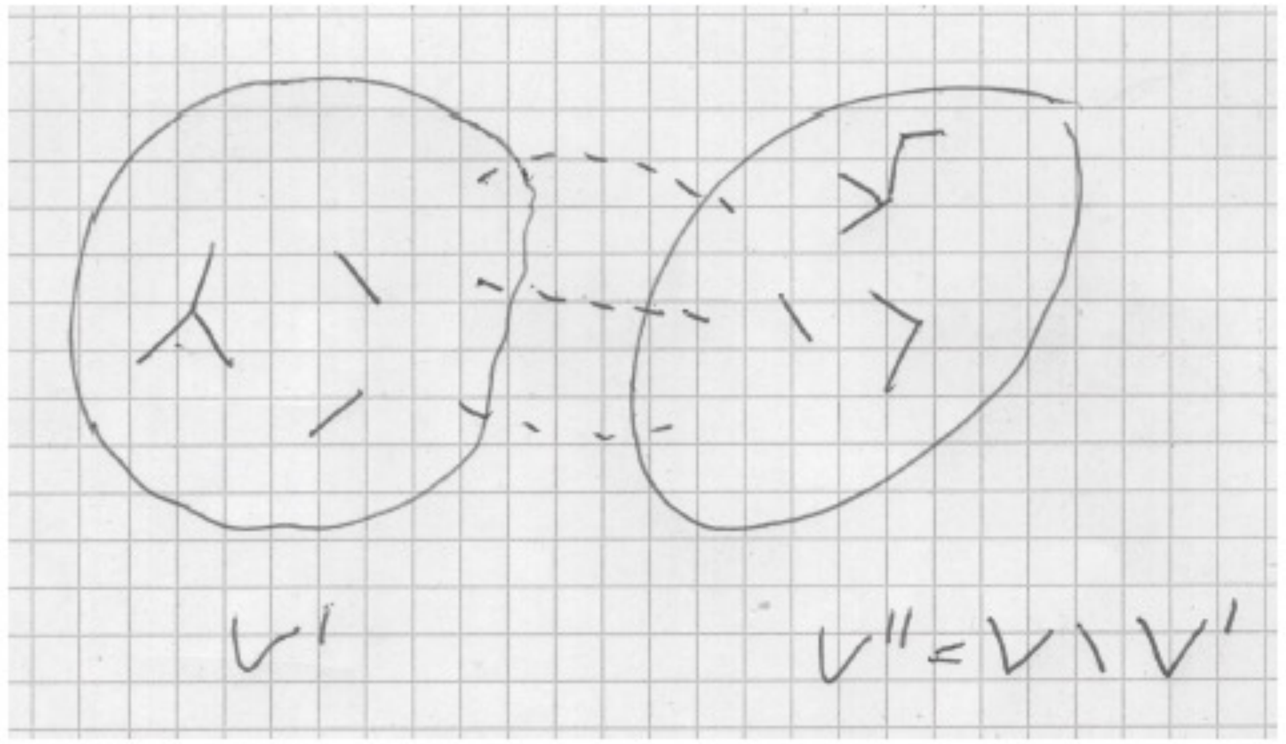


Figure 6: Edges is T' are drawn solid. Dotted edges e cross the cut $C = (V', V'')$. Such edges are not in T'.

def: *weighted graph* $G = (V, E, w)$

- (V, E) undirected graph
- $w: E \rightarrow R^+$ nonnegative edge weights

(V, T) spanning tree

$w(T) = \Sigma_{e \in T} w(e)$ *weight of T*

def: *cut* $C = (V', V'')$ with

- $V' \cup V'' = V$
- $V' \cap V'' = \emptyset$

$T' \subset E$ *respects* cut C if

no edge in T' crosses the cut

$\{u, v\} \in T' \rightarrow u, v \in V' \vee u, v \in V''$

Lemma 3. *Let $T' \subseteq E$ be a set of edges contained in the edges of a minimum spanning tree (V, T) of graph $G = (V, E, w)$.*

$$T' \subseteq T$$

Assume T' respects cut C and e is an edge of minimum weight cossing cut C. Then there is a minimum spanning tree (V, R) of G where r

$$T' \cup \{e\} \subseteq R$$

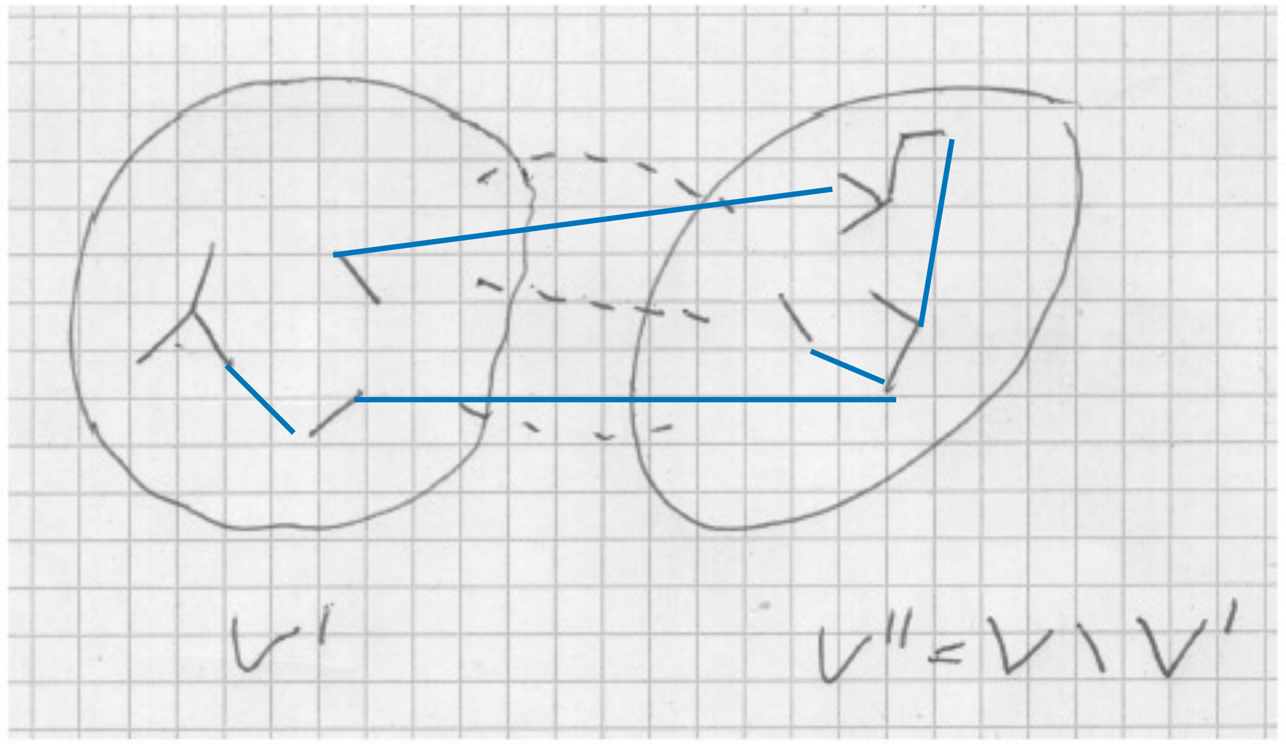


Figure 6: Edges is T' are drawn solid. Dotted edges e cross the cut $C = (V', V'')$. Such edges are not in T'.

def: *weighted graph* $G = (V, E, w)$

- (V, E) undirected graph
- $w: E \rightarrow R^+$ nonnegative edge weights

(V, T) spanning tree

$w(T) = \Sigma_{e \in T} w(e)$ *weight of T*

def: *cut* $C = (V', V'')$ with

- $V' \cup V'' = V$
- $V' \cap V'' = \emptyset$

$T' \subset E$ *respects* cut C if

no edge in T' crosses the cut

$\{u, v\} \in T' \rightarrow u, v \in V' \vee u, v \in V''$

Lemma 3. *Let $T' \subseteq E$ be a set of edges contained in the edges of a minimum spanning tree (V, T) of graph $G = (V, E, w)$.*

$$T' \subseteq T$$

Assume T' respects cut C and e is an edge of minimum weight cossing cut C. Then there is a minimum spanning tree (V, R) of G where r

$$T' \cup \{e\} \subseteq R$$

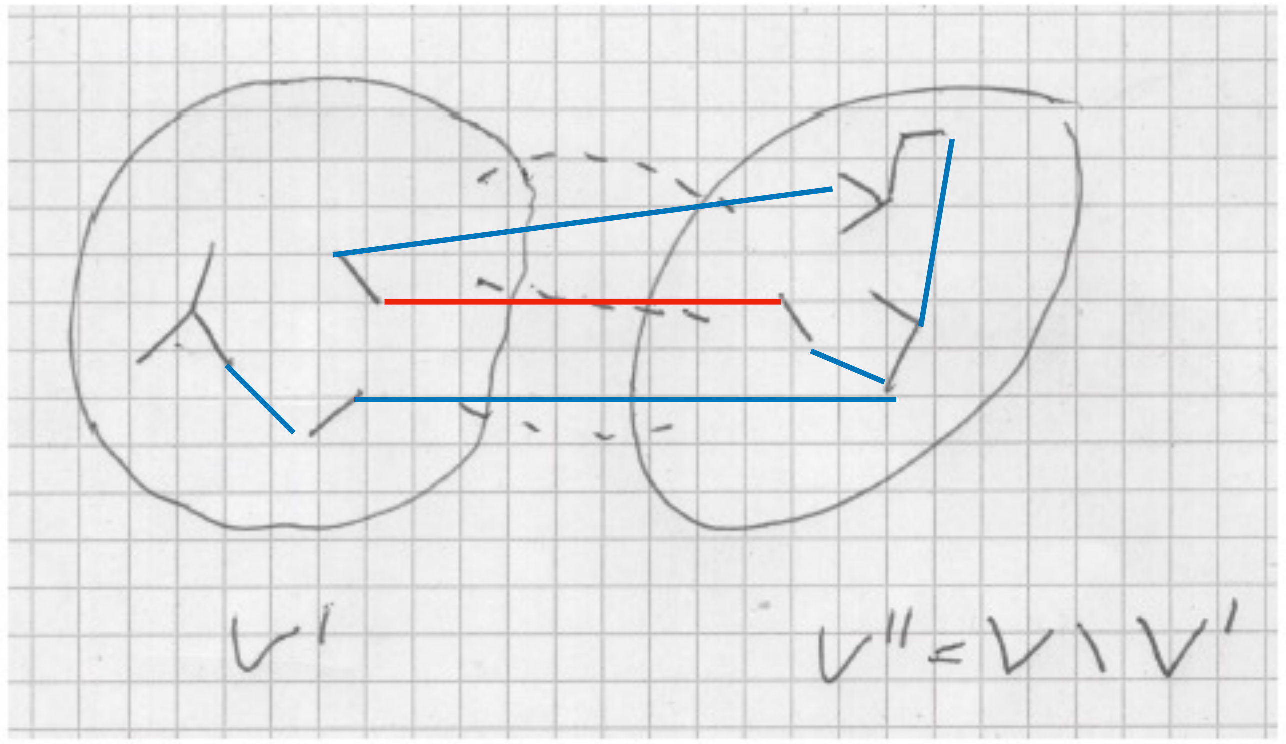


Figure 6: Edges is T' are drawn solid. Dotted edges e cross the cut $C = (V', V'')$. Such edges are not in T'.

- if $e \in T$: done

Lemma 3. *Let $T' \subseteq E$ be a set of edges contained in the edges of a minimum spanning tree (V,T) of graph $G=(V,E,w)$.*

$$T' \subseteq T$$

Assume T' respects cut C and e is an edge of minimum weight cossing cut C. Then there is a minimum spanning tree (V,R) of G where r

$$T' \cup \{e\} \subseteq R$$

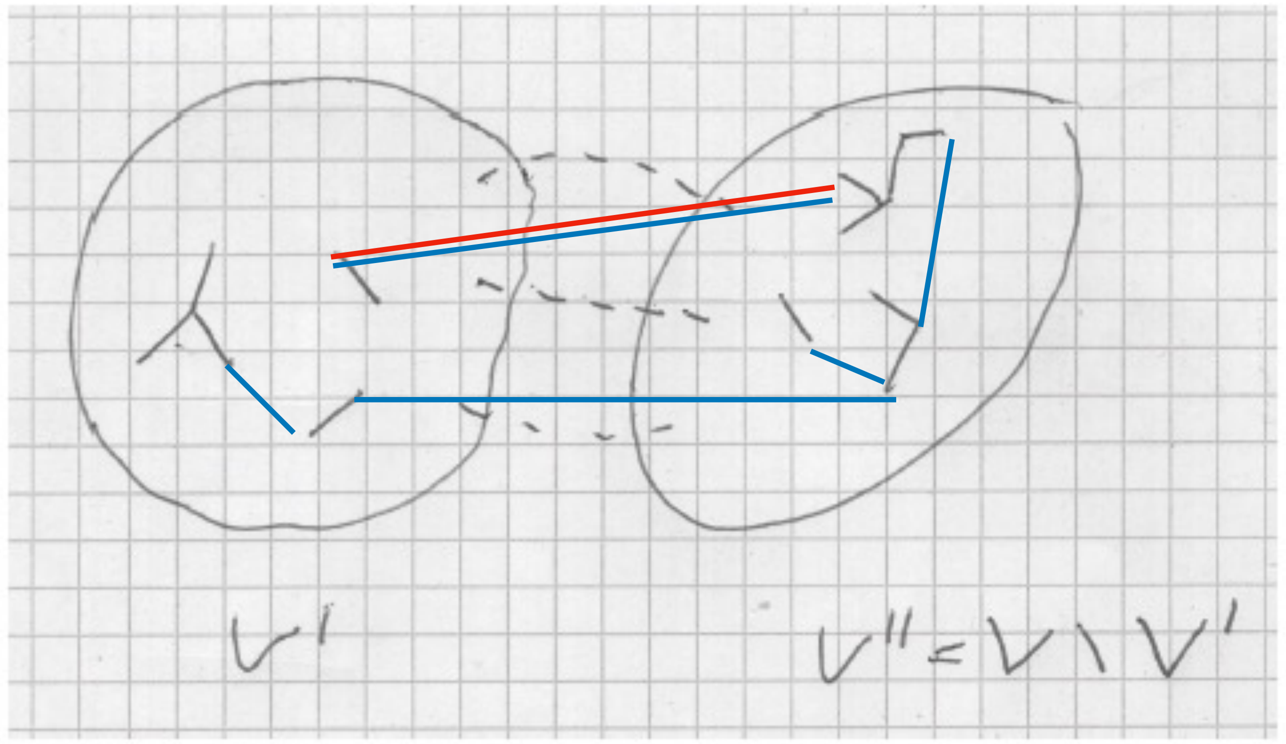


Figure 6: Edges is T' are drawn solid. Dotted edges e cross the cut $C=(V',V'')$. Such edges are not in T'.

- if $e \in T$: done

- otherwise add e to (V,T). This closes a single cycle.

Lemma 3. *Let $T' \subseteq E$ be a set of edges contained in the edges of a minimum spanning tree (V,T) of graph $G = (V,E,w)$.*

$$T' \subseteq T$$

Assume T' respects cut C and e is an edge of minimum weight cossing cut C. Then there is a minimum spanning tree (V,R) of G where r

$$T' \cup \{e\} \subseteq R$$

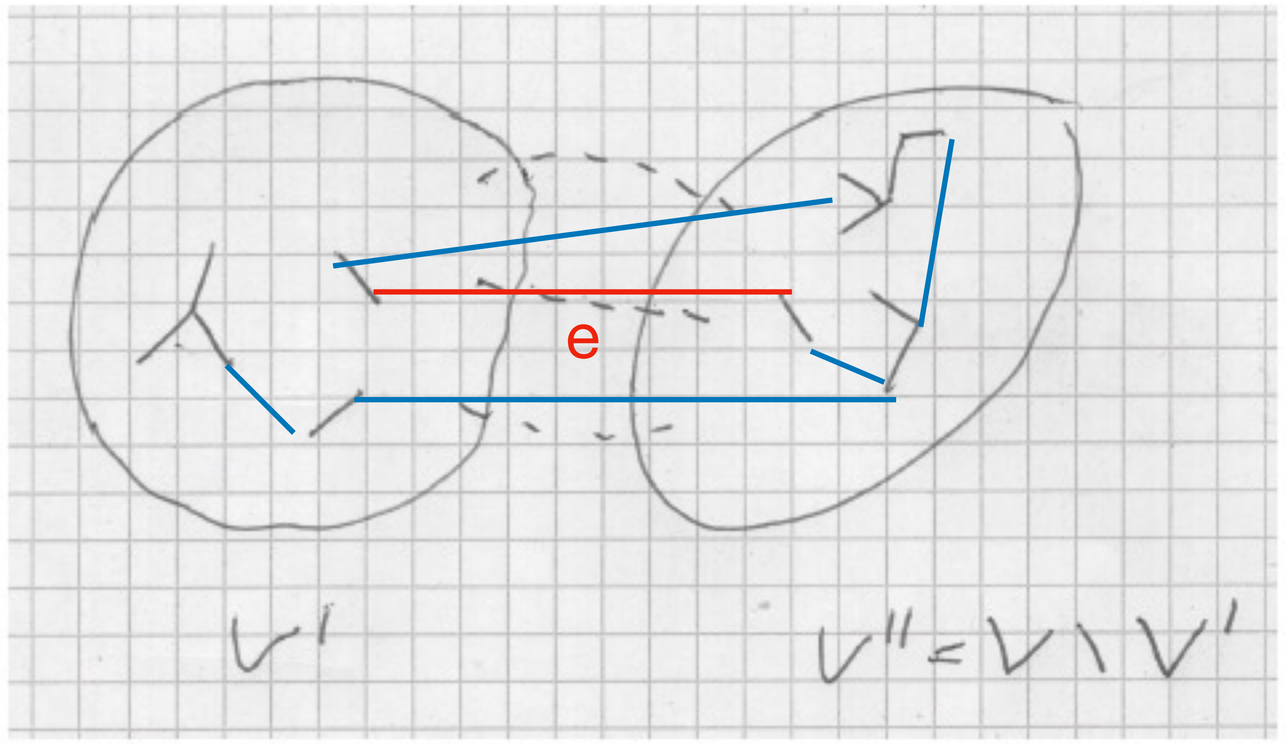


Figure 6: Edges is T' are drawn solid. Dotted edges e cross the cut $C = (V',V'')$. Such edges are not in T'.

- if $e \in T$: done

- otherwise add e to (V,T). This closes a single cycle.

- the cycle must contain an edge e' crossing the cut. Remove e'.

$$R = T \setminus \{e'\} \cup \{e\}$$

$$w(R) = w(T) - w(e') + w(e) \leq w(T)$$

Lemma 3. *Let $T' \subseteq E$ be a set of edges contained in the edges of a minimum spanning tree (V,T) of graph $G = (V,E,w)$.*

$$T' \subseteq T$$

Assume T' respects cut C and e is an edge of minimum weight cossing cut C. Then there is a minimum spanning tree (V,R) of G where r

$$T' \cup \{e\} \subseteq R$$

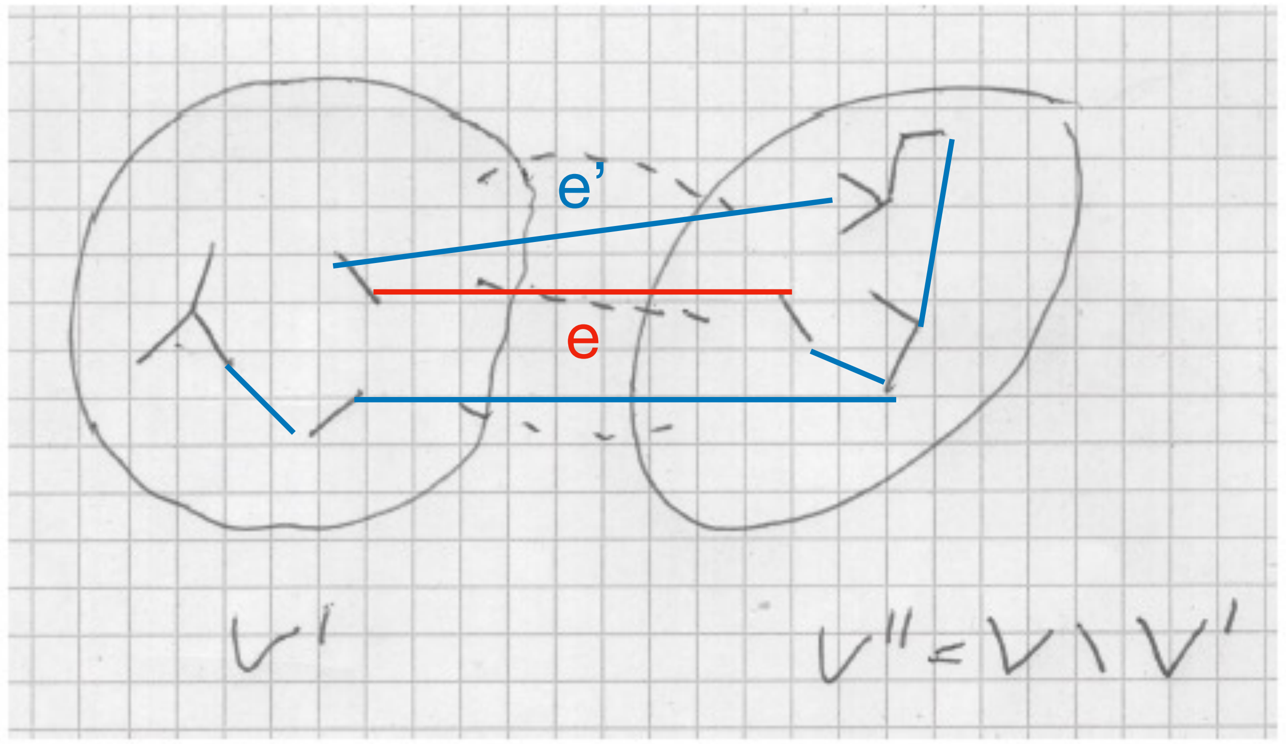


Figure 6: Edges is T' are drawn solid. Dotted edges e cross the cut $C = (V', V'')$. Such edges are not in T'.

- if $e \in T$: done

- otherwise add e to (V,T). This closes a single cycle.

- the cycle must contain an edge e' crossing the cut. Remove e'.

$$R = T \setminus \{e'\} \cup \{e\}$$

$$w(R) = w(T) - w(e') + w(e) \leq w(T)$$

Lemma 3. *Let $T' \subseteq E$ be a set of edges contained in the edges of a minimum spanning tree (V,T) of graph $G = (V,E,w)$.*

$$T' \subseteq T$$

Assume T' respects cut C and e is an edge of minimum weight cossing cut C. Then there is a minimum spanning tree (V,R) of G where r

$$T' \cup \{e\} \subseteq R$$

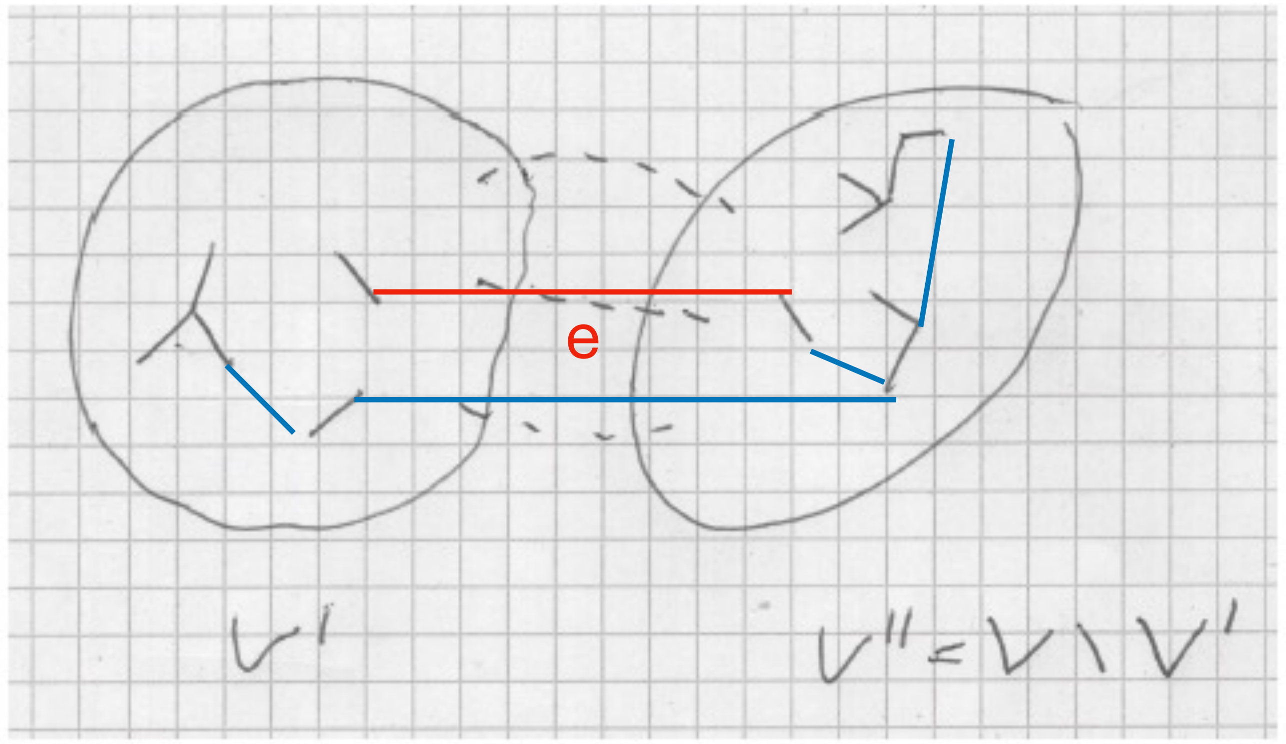


Figure 6: Edges is T' are drawn solid. Dotted edges e cross the cut $C = (V', V'')$. Such edges are not in T'.

Kruskal’s Algorithm

Algorithm 50 Kruskal-MST

Input: connected weighted undirected graph $G = (V, E, w)$
Output: a minimum spanning tree T of G

```
Sort the edges by increasing weight
for each vertex v ∈ V do
    Make-set(v)
E_T := ∅
for each edge e = {u, v} ∈ E, in order by increasing weight do
    if Find(u) ≠ Find(v) then
        E_T := E_T ∪ {e}
        Union(u, v)
return (V, E_T)
```

correctness:

- T contained in edges of minimum spanning tree
- induction using lemma 3

Lemma 3. *Let $T' \subseteq E$ be a set of edges contained in the edges of a minimum spanning tree (V,T) of graph $G = (V,E,w)$.*

$$T' \subseteq T$$

Assume T' respects cut C and e is an edge of minimum weight cossing cut C. Then there is a minimum spanning tree (V,R) of G where r

$$T' \cup \{e\} \subseteq R$$

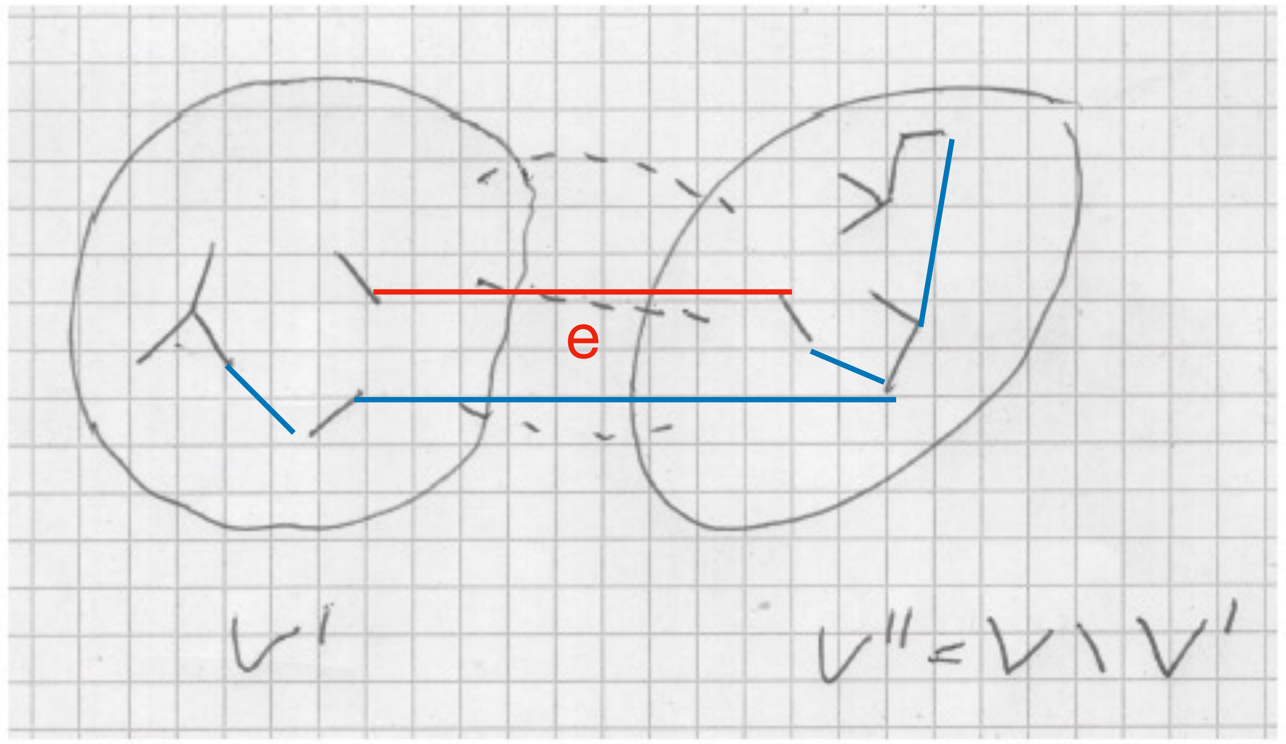


Figure 6: Edges is T' are drawn solid. Dotted edges e cross the cut $C = (V', V'')$. Such edges are not in T'.

Kruskal's Algorithm

Algorithm 50 Kruskal-MST

Input: connected weighted undirected graph $G = (V, E, w)$
Output: a minimum spanning tree T of G

Sort the edges by increasing weight
for each vertex $v \in V$ **do**
    Make-set(v)
$E_T := \emptyset$
for each edge $e = \{u, v\} \in E$, in order by increasing weight **do**
    if Find(u) $\neq$ Find(v) **then**
        $E_T := E_T \cup \{e\}$
        Union(u, v)
return (V, E_T)

---

greedy

correctness:

- T contained in edges of minimum spanning tree
- induction using lemma 3

greedy algorithms:

- go locally for optimum
- hope globally for the best

Lemma 3. *Let $T' \subseteq E$ be a set of edges contained in the edges of a minimum spanning tree (V, T) of graph $G = (V, E, w)$.*

$$T' \subseteq T$$

Assume T' respects cut C and e is an edge of minimum weight cossing cut C. Then there is a minimum spanning tree (V, R) of G where r

$$T' \cup \{e\} \subseteq R$$

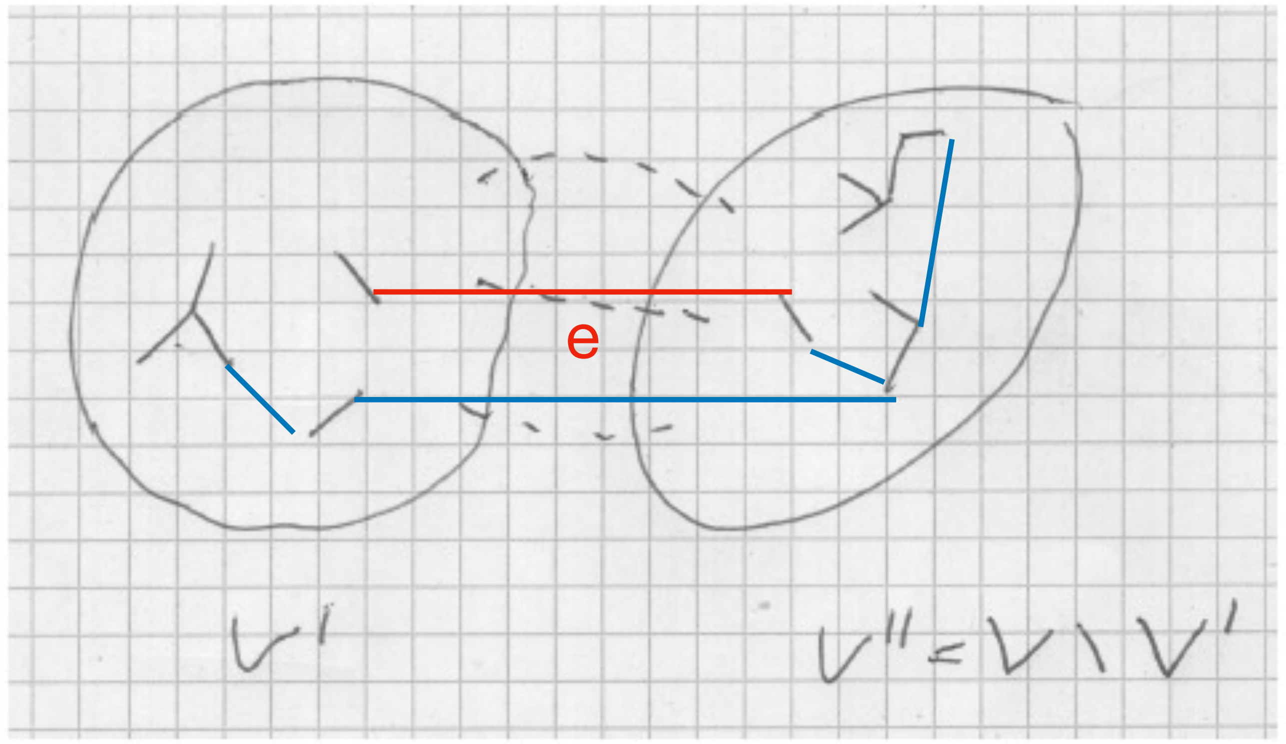


Figure 6: Edges is T' are drawn solid. Dotted edges e cross the cut $C = (V', V'')$. Such edges are not in T'.

Kruskal's Algorithm

Algorithm 50 Kruskal-MST

Input: connected weighted undirected graph $G = (V, E, w)$
Output: a minimum spanning tree T of G

```
Sort the edges by increasing weight
for each vertex v ∈ V do
    Make-set(v)
E_T := ∅
for each edge e = {u, v} ∈ E, in order by increasing weight do
    if Find(u) ≠ Find(v) then
        E_T := E_T ∪ {e}
        Union(u, v)
return (V, E_T)
```

correctness:

- T contained in edges of minimum spanning tree
- induction using lemma 3

run time:

- sorting: $O(|E|\log|E|)$
- union find: $O(|E| \cdot \alpha(|E|))$

Lemma 3. *Let $T' \subseteq E$ be a set of edges contained in the edges of a minimum spanning tree (V, T) of graph $G = (V, E, w)$.*

$$T' \subseteq T$$

Assume T' respects cut C and e is an edge of minimum weight cossing cut C. Then there is a minimum spanning tree (V, R) of G where r

$$T' \cup \{e\} \subseteq R$$

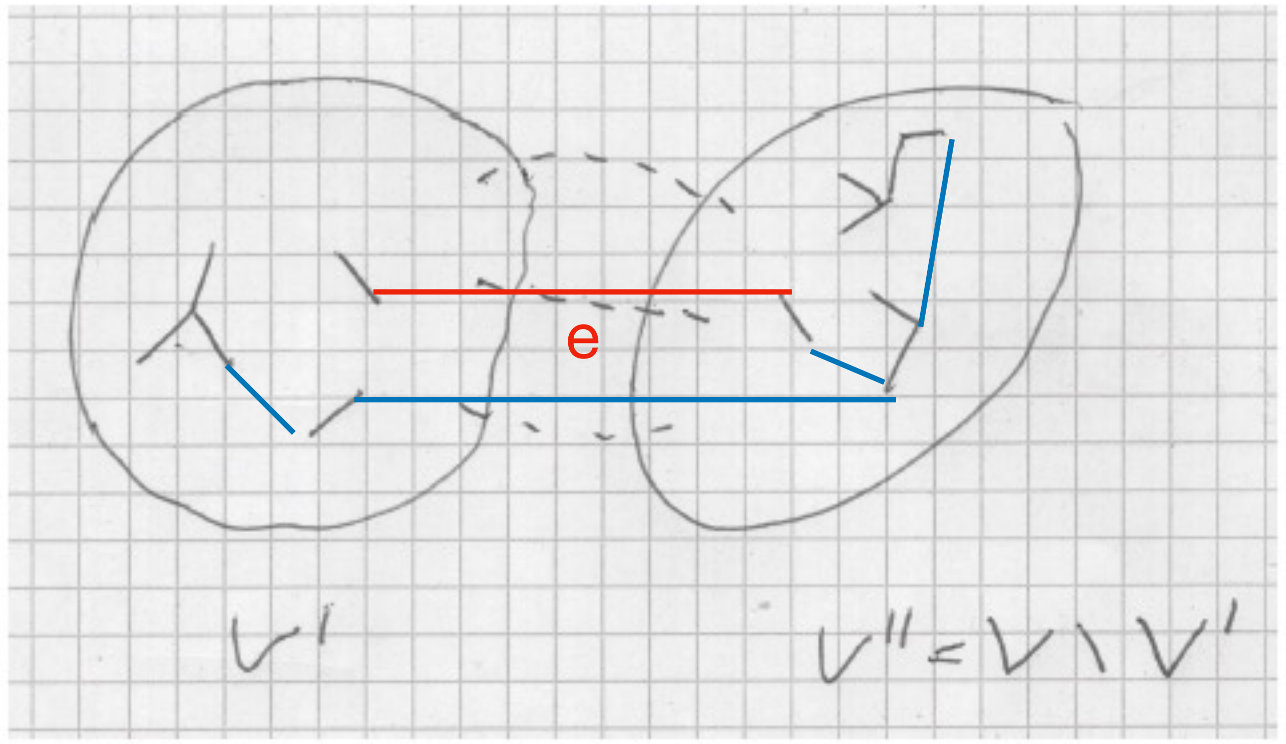


Figure 6: Edges is T' are drawn solid. Dotted edges e cross the cut $C = (V', V'')$. Such edges are not in T'.

Kruskal’s Algorithm

Algorithm 50 Kruskal-MST

Input: connected weighted undirected graph $G = (V, E, w)$

Output: a minimum spanning tree T of G

```
Sort the edges by increasing weight
for each vertex v ∈ V do
    Make-set(v)
E_T := ∅
for each edge e = {u, v} ∈ E, in order by increasing weight do
    if Find(u) ≠ Find(v) then
        E_T := E_T ∪ {e}
        Union(u, v)
return (V, E_T)
```

correctness:

- T contained in edges of minimum spanning tree
- induction using lemma 3

run time:

- sorting: $O(|E|\log|E|)$
- union find: $O(|E| \cdot \alpha(|E|))$

Lemma 3. *Let $T' \subseteq E$ be a set of edges contained in the edges of a minimum spanning tree (V,T) of graph $G = (V,E,w)$.*

$$T' \subseteq T$$

Assume T' respects cut C and e is an edge of minimum weight cossing cut C. Then there is a minimum spanning tree (V,R) of G where r

$$T' \cup \{e\} \subseteq R$$

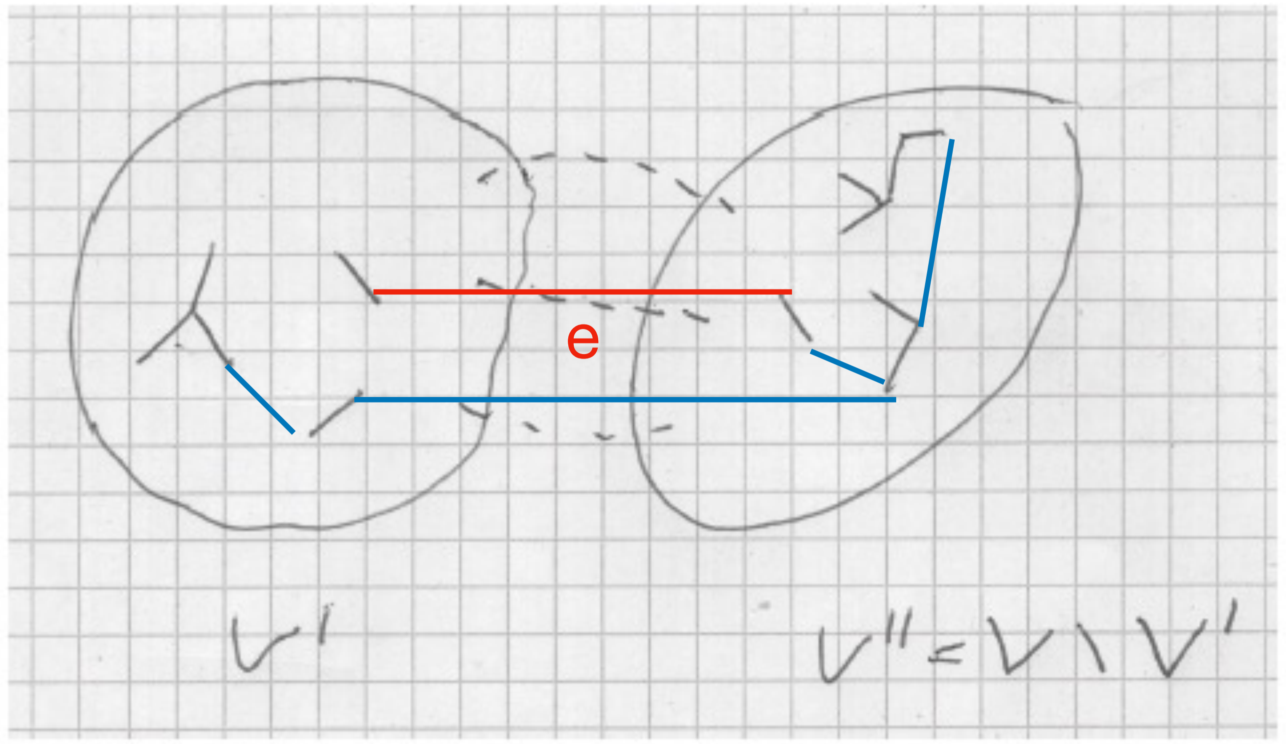


Figure 6: Edges is T' are drawn solid. Dotted edges e cross the cut $C = (V', V'')$. Such edges are not in T'.

length of paths

def: *weighted graph* $G = (V, E, w)$

- (V, E) directed graph
- $w: E \to R^+$ nonnegative edge weights

$p = (p_0, \dots, p_\ell)$ path

$\Sigma_{i=1}^{\ell} w(p_{i-1}, p_i)$ length *of path*

sum of edge weights

single source shortest path

def: *weighted graph* $G = (V, E, w)$

- (V, E) directed graph
- $w: E \rightarrow R^+$ nonnegative edge weights

$p = (p_0, \ldots, p_\ell)$ path

$\Sigma_{i=1}^{\ell} w(p_{i-1}, p_i)$ length *of path*

sum of edge weights

algorithm

- input: weighted connected graph, start node $s \in V$
- output: for each node $\delta[v]$ =
  length of a shortest path from s to v

Dijkstra's Algorithm

idea:

- grow set of nodes S
- obtain upper bounds $d[x] \geq \delta(x)$ on length of shortest path
- $d[x]$ length of shortest path found so far, i.e in S or in S plus one further edge.

Dijkstra’s Algorithm

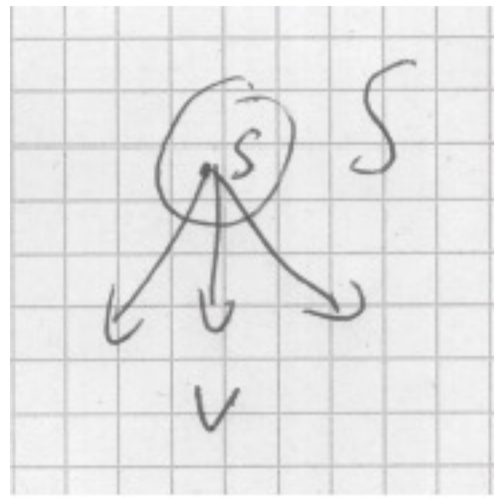
idea:

- grow set of nodes S
- obtain upper bounds $d[x] \geq \delta(x)$ on length of shortest path
- $d[x]$ length of shortest path found so far, i.e in S or in S plus one further edge.

initialization

$S = \{s\}$; $d[s] = 0$;
for all $v \neq s$

$$d[v] = \begin{cases} w(s,v) & (s,v) \in E \\ \infty & \text{otherwise} \end{cases}$$



Dijkstra’s Algorithm

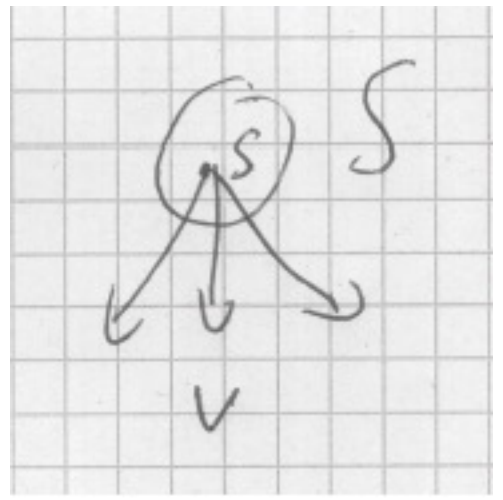
idea:

- grow set of nodes S
- obtain upper bounds $d[x] \geq \delta(x)$ on length of shortest path
- $d[x]$ length of shortest path found so far, i.e in S or in S plus one further edge.

initialization

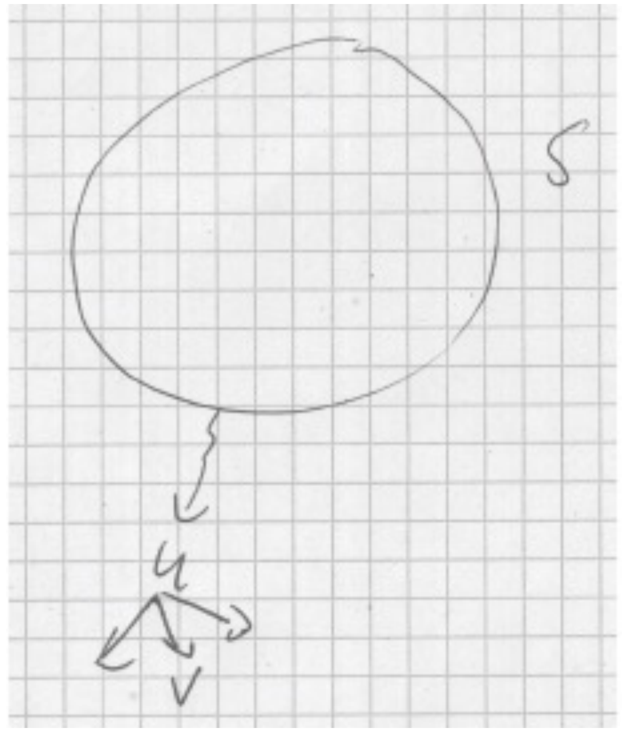
$S = \{s\}$; $d[s] = 0$;
for all $v \neq s$

$$d[v] = \begin{cases} w(s,v) & (s,v) \in E \\ \infty & \text{otherwise} \end{cases}$$



iteration:

while $S \neq V$
{
choose $u \in V \setminus S$ with minimal $d[u]$;
$S = S \cup \{u\}$;
for all v with $(u,v) \in E$, $v \notin S$
$\{d[v] = \min\{d[v], d[u] + w(u,v)\}\}$
}



Dijkstra's Algorithm

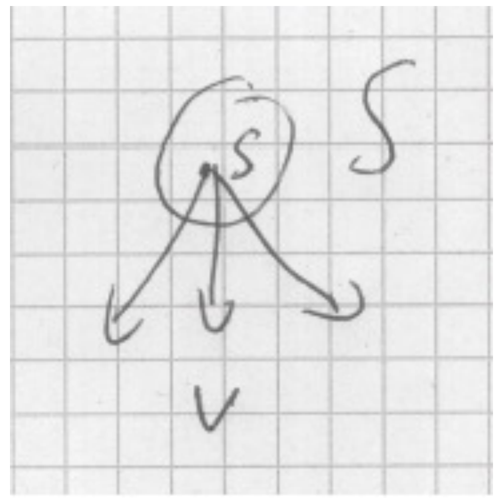
idea:

- grow set of nodes S
- obtain upper bounds $d[x] \geq \delta(x)$ on length of shortest path
- $d[x]$ length of shortest path found so far, i.e in S or in S plus one further edge.

initialization

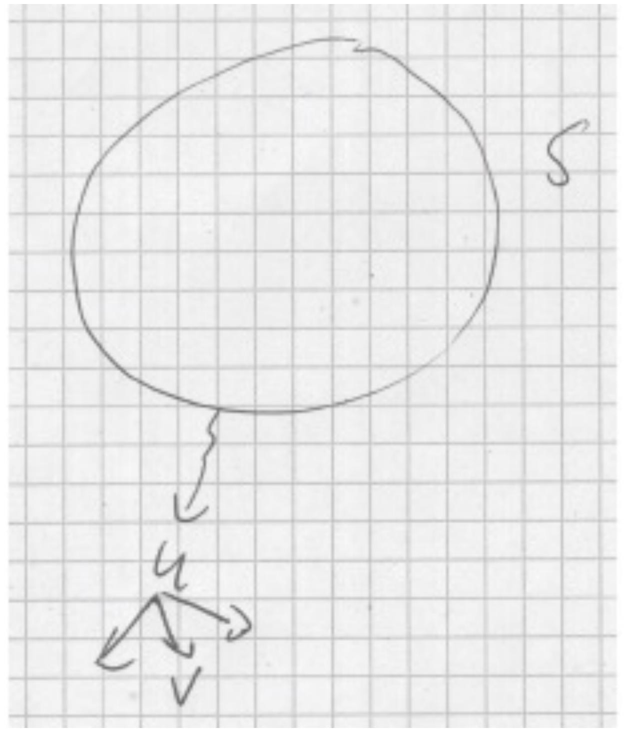
$S = \{s\}$; $d[s] = 0$;
for all $v \neq s$

$$d[v] = \begin{cases} w(s,v) & (s,v) \in E \\ \infty & \text{otherwise} \end{cases}$$



iteration:

while $S \neq V$
{
choose $u \in V \setminus S$ with minimal $d[u]$;
$S = S \cup \{u\}$;
for all v with $(u,v) \in E$
$\{d[v] = \min\{d[v], d[u] + w(u,v)\}\}$
}



primed notation

- S^t and d^t: set S and distance bound d before iteration t of the while loop.
- in statements which hold for all t: drop superscript t
- in statement involving only t and $t+1$ (e.g. induction steps): replace X^t by X and X^{t+1} by X'; here for $X \in \{S, d\}$

Dijkstra’s Algorithm

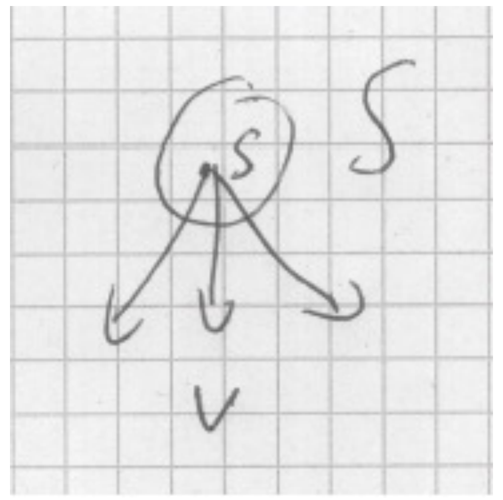
idea:

- grow set of nodes S
- obtain upper bounds $d[x] \geq \delta(x)$ on length of shortest path
- $d[x]$ length of shortest path found so far, i.e in S or in S plus one further edge.

initialization

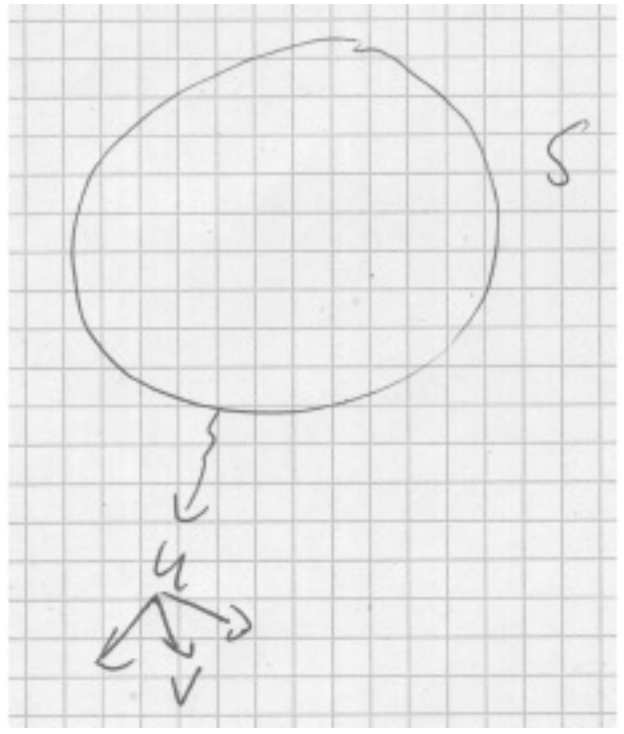
$S = \{s\}$; $d[s] = 0$;
for all $v \neq s$

$$d[v] = \begin{cases} w(s,v) & (s,v) \in E \\ \infty & \text{otherwise} \end{cases}$$



iteration:

while $S \neq V$
{
choose $u \in V \setminus S$ with minimal $d[u]$;
$S = S \cup \{u\}$;
for all v with $(u,v) \in E$
$\{d[v] = \min\{d[v], d[u] + w(u,v)\}\}$
}



primed notation

- S^t and d^t: set S and distance bound d before iteration t of the while loop.
- in statements which hold for all t: drop superscript t
- in statement involving only t and $t+1$ (e.g. induction steps): replace X^t by X and X^{t+1} by X'; here for $X \in \{S, d\}$

properties

1. shortest discovered distances nonincreasing; stable for nodes $u \in S$

$$d'[u] \leq d[u] \ , \ u \in S \rightarrow d'[u] = d[u]$$

2. $d[u]$ is upper bound on length of real shortest paths

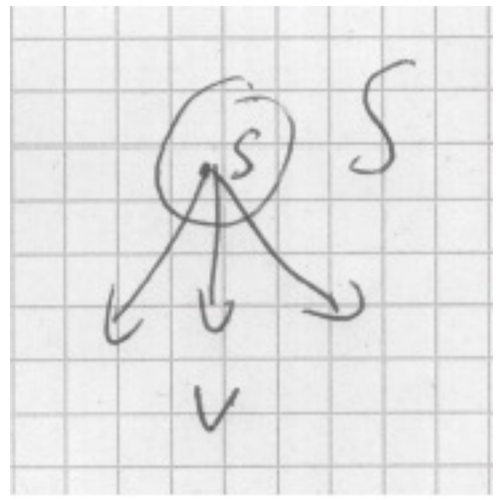
$$\delta[u] \leq d[u]$$

correctness

initialization

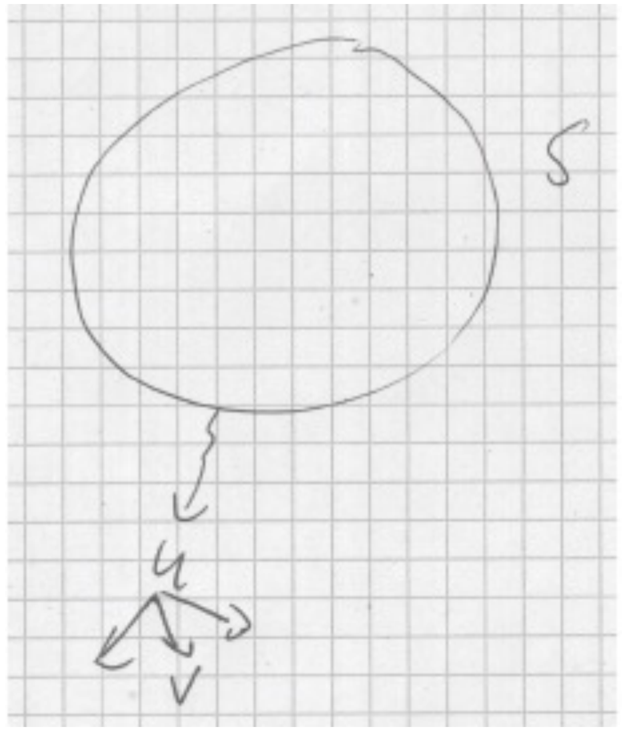
$S = \{s\}$; $d[s] = 0$;
for all $v \neq s$

$$d[v] = \begin{cases} w(s,v) & (s,v) \in E \\ \infty & \text{otherwise} \end{cases}$$



iteration:

while $S \neq V$
{
choose $u \in V \setminus S$ with minimal $d[u]$;
$S = S \cup \{u\}$;
for all v with $(u,v) \in E$
$\{d[v] = \min\{d[v], d[u] + w(u,v)\}\}$
}



properties

1. shortest discovered distances nonincreasing; stable for nodes $u \in S$

$$d'[u] \leq d[u]\ ,\ u \in S \rightarrow d'[u] = d[u]$$

2. $d[u]$ is upper bound on length of real shortest paths

$$\delta[u] \leq d[u]$$

Lemma 4.

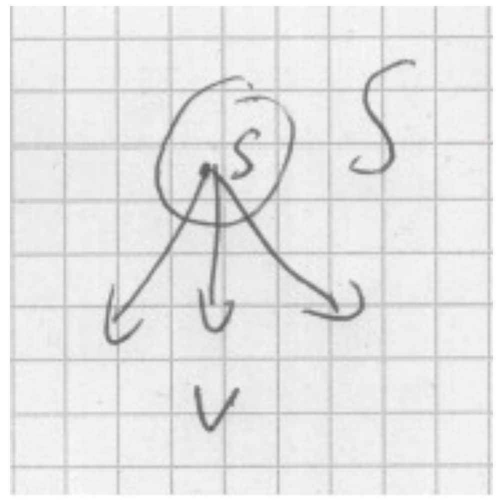
$$u = s \ \vee \ u \in S' \setminus S \ \rightarrow \ d[u] = \delta[u]$$

correctness

initialization

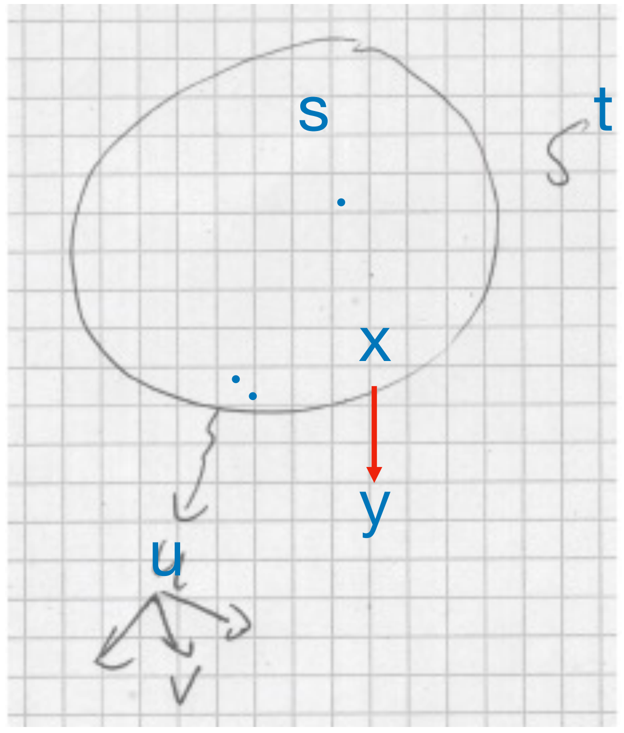
$S = \{s\}$; $d[s] = 0$;
for all $v \neq s$

$$d[v] = \begin{cases} w(s,v) & (s,v) \in E \\ \infty & \text{otherwise} \end{cases}$$



iteration:

while $S \neq V$
{
choose $u \in V \setminus S$ with minimal $d[u]$;
$S = S \cup \{u\}$;
for all v with $(u,v) \in E$
$\{d[v] = \min\{d[v], d[u] + w(u,v)\}\}$
}



properties

1. shortest discovered distances nonincreasing; stable for nodes $u \in S$

$$d'[u] \leq d[u]\ ,\ u \in S \rightarrow d'[u] = d[u]$$

2. $d[u]$ is upper bound on length of real shortest paths

$$\delta[u] \leq d[u]$$

Lemma 4.

$$u = s \ \vee \ u \in S' \setminus S \ \rightarrow \ d[u] = \delta[u]$$

$u = s$ trivial. For $u \neq s$ assume in some iteration of while loop:

$$u \in S^{t+1} \setminus S^t\ ,\ \delta[u] < d^t[u]$$

Consider *first* such t and u. Let P be shortest path from s to u.

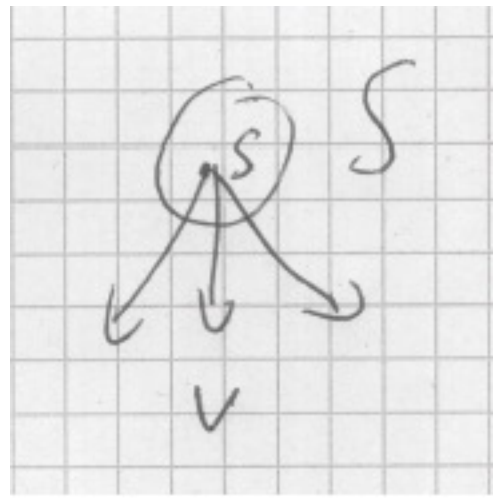
$$s \in S^t \wedge u \notin S^t \rightarrow \exists \text{ edge } (x,y) \text{ on } P.\ x \in S^t \wedge y \notin S^t$$

correctness

initialization

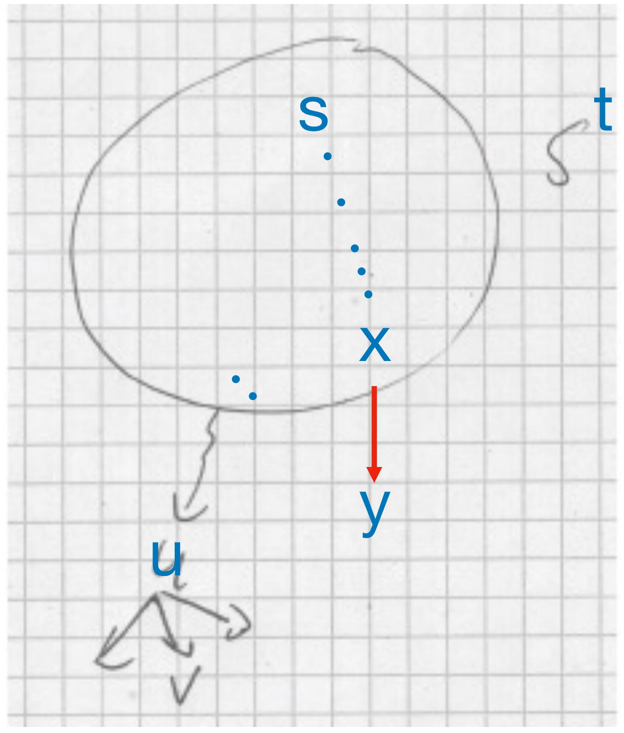
$S = \{s\};\ d[s] = 0;$
for all $v \neq s$

$$d[v] = \begin{cases} w(s,v) & (s,v) \in E \\ \infty & \text{otherwise} \end{cases}$$



iteration:

while $S \neq V$
{
choose $u \in V \setminus S$ with minimal $d[u]$;
$S = S \cup \{u\}$;
for all v with $(u,v) \in E$
$\{d[v] = \min\{d[v], d[u] + w(u,v)\}\}$
}



properties

1. shortest discovered distances nonincreasing; stable for nodes $u \in S$

$$d'[u] \leq d[u]\ ,\ u \in S \rightarrow d'[u] = d[u]$$

2. $d[u]$ is upper bound on length of real shortest paths

$$\delta[u] \leq d[u]$$

Lemma 4.

$$u = s\ \vee\ u \in S' \setminus S\ \rightarrow\ d[u] = \delta[u]$$

$u = s$ trivial. For $u \neq s$ assume in some iteration of while loop:

$$u \in S^{t+1} \setminus S^t\ ,\ \delta[u] < d^t[u]$$

Consider *first* such t and u. Let P be shortest path from s to u.

$$s \in S^t \wedge u \notin S^t \rightarrow \exists \text{ edge } (x,y) \text{ on } P.\ x \in S^t \wedge y \notin S^t$$

x entered S before u:

$$x \in S^{r+1} \setminus S^r\ ,\ r < t$$

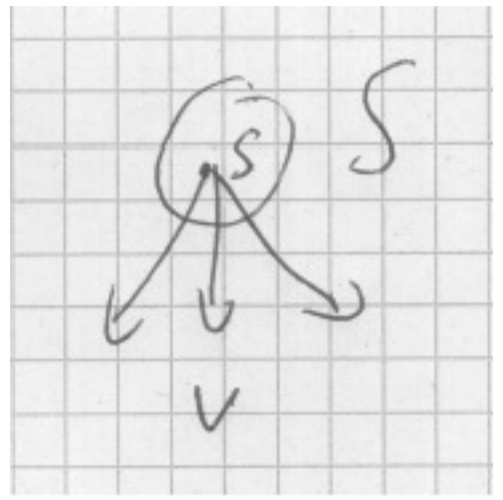
$$\begin{aligned} \delta[x] &= d^r[x] && \text{(minimality of } t\text{)} \\ &= d^t[x] && \text{(property 1)} \end{aligned}$$

correctness

initialization

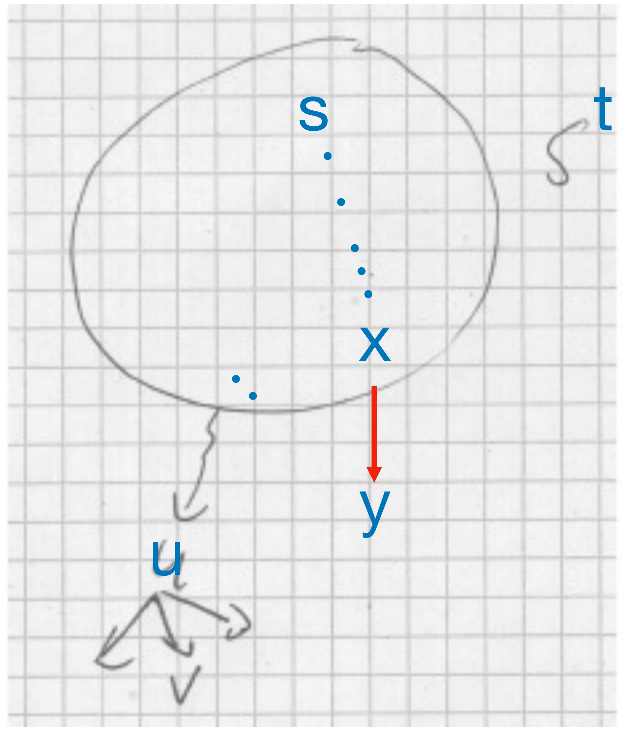
$S = \{s\};\ d[s] = 0;$
for all $v \neq s$

$$d[v] = \begin{cases} w(s,v) & (s,v) \in E \\ \infty & \text{otherwise} \end{cases}$$



iteration:

while $S \neq V$
{
choose $u \in V \setminus S$ with minimal $d[u]$;
$S = S \cup \{u\}$;
for all v with $(u,v) \in E$
$\{d[v] = \min\{d[v], d[u] + w(u,v)\}\}$
}



properties

1. shortest discovered distances nonincreasing; stable for nodes $u \in S$

$$d'[u] \leq d[u]\ ,\ u \in S \to d'[u] = d[u]$$

2. $d[u]$ is upper bound on length of real shortest paths

$$\delta[u] \leq d[u]$$

Lemma 4.

$$u = s\ \vee\ u \in S' \setminus S\ \to\ d[u] = \delta[u]$$

$u = s$ trivial. For $u \neq s$ assume in some iteration of while loop:

$$u \in S^{t+1} \setminus S^t\ ,\ \delta[u] < d^t[u]$$

Consider *first* such t and u. Let P be shortest path from s to u.

$$s \in S^t \wedge u \notin S^t \to \exists \text{ edge } (x,y) \text{ on } P.\ x \in S^t \wedge y \notin S^t$$

x entered S before u:

$$x \in S^{r+1} \setminus S^r\ ,\ r < t$$

$$\begin{aligned} \delta[x] &= d^r[x] && (\text{minimality of } t) \\ &= d^t[x] && (\text{property 1}) \end{aligned}$$

$$\begin{aligned} d^t[y] &\leq d^r[y] && (\text{property 1}) \\ &\leq d^r[x] + w(x,y) && (\text{algorithm}) \\ &= \delta[x] + w(x,y) && (\text{above}) \\ &= \delta[y] && ((x,y) \text{ on shortest path } P) \\ &\leq d^t[y] && (\text{property 2}) \end{aligned}$$

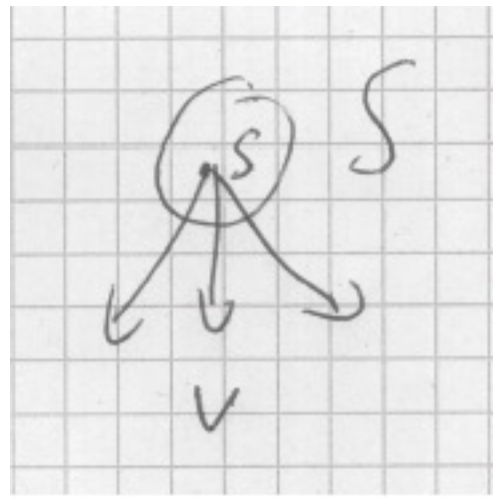
$$d^t[y] = \delta[y]$$

correctness

initialization

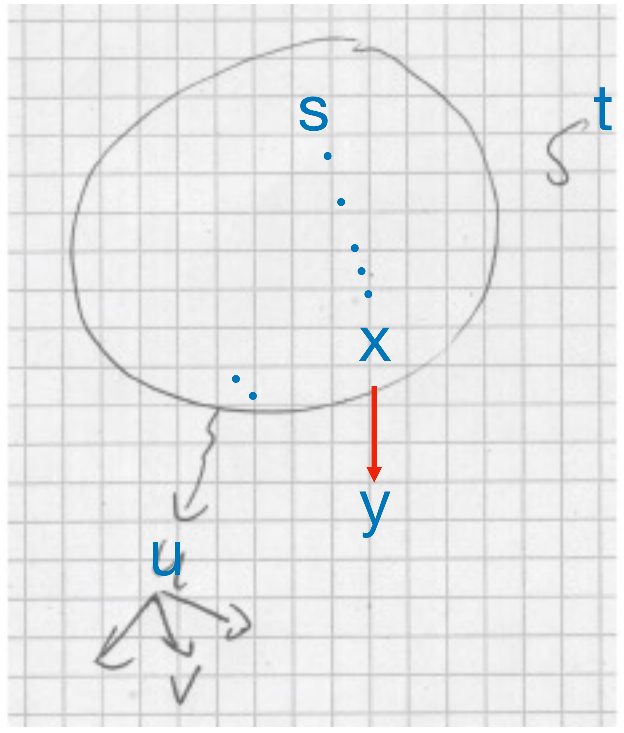
$S = \{s\}$; $d[s] = 0$;
for all $v \neq s$

$$d[v] = \begin{cases} w(s,v) & (s,v) \in E \\ \infty & \text{otherwise} \end{cases}$$



iteration:

while $S \neq V$
{
choose $u \in V \setminus S$ with minimal $d[u]$;
$S = S \cup \{u\}$;
for all v with $(u,v) \in E$
$\{d[v] = \min\{d[v], d[u] + w(u,v)\}\}$
}



properties

1. shortest discovered distances nonincreasing; stable for nodes $u \in S$

$$d'[u] \leq d[u] \ , \ u \in S \rightarrow d'[u] = d[u]$$

2. $d[u]$ is upper bound on length of real shortest paths

$$\delta[u] \leq d[u]$$

Lemma 4.

$$u = s \ \vee \ u \in S' \setminus S \ \rightarrow \ d[u] = \delta[u]$$

$u = s$ trivial. For $u \neq s$ assume in some iteration of while loop:

$$u \in S^{t+1} \setminus S^t \ , \ \delta[u] < d^t[u]$$

Consider *first* such t and u. Let P be shortest path from s to u.

$$s \in S^t \wedge u \notin S^t \rightarrow \exists \text{ edge } (x,y) \text{ on } P.\ x \in S^t \wedge y \notin S^t$$

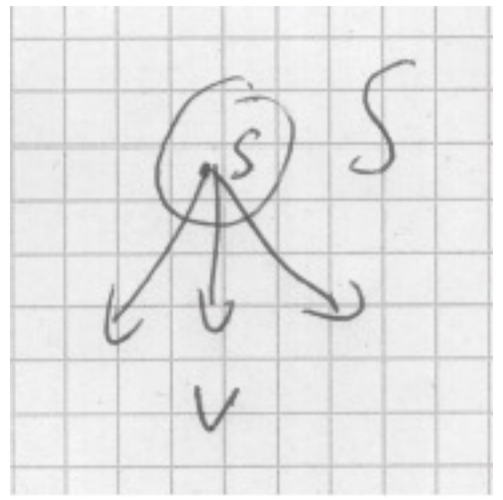
$$d^t[y] = \delta[y]$$

correctness

initialization

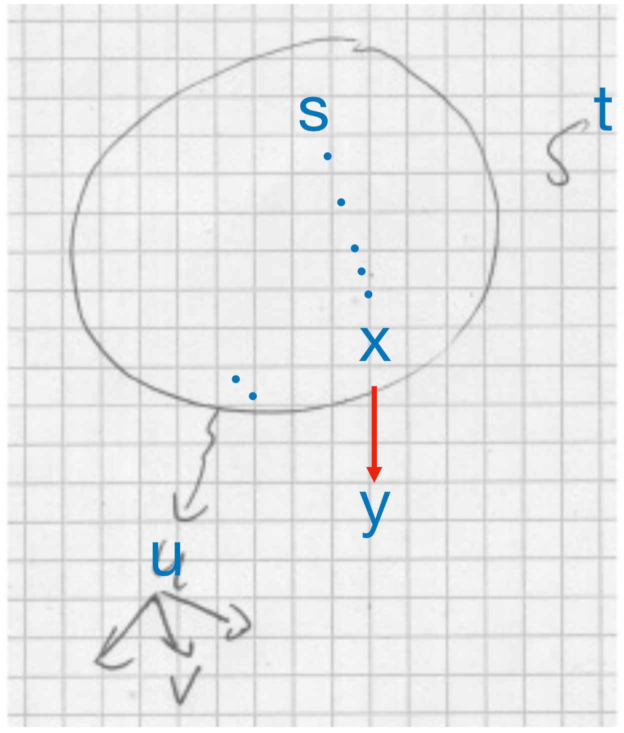
$S = \{s\};\ d[s] = 0;$
for all $v \neq s$

$$d[v] = \begin{cases} w(s,v) & (s,v) \in E \\ \infty & \text{otherwise} \end{cases}$$



iteration:

while $S \neq V$
{
choose $u \in V \setminus S$ with minimal $d[u]$;
$S = S \cup \{u\}$;
for all v with $(u,v) \in E$
$\{d[v] = \min\{d[v], d[u] + w(u,v)\}\}$
}



properties

1. shortest discovered distances nonincreasing; stable for nodes $u \in S$

$$d'[u] \leq d[u]\ ,\ u \in S \rightarrow d'[u] = d[u]$$

2. $d[u]$ is upper bound on length of real shortest paths

$$\delta[u] \leq d[u]$$

Lemma 4.

$$u = s \vee u \in S' \setminus S \rightarrow d[u] = \delta[u]$$

$u = s$ trivial. For $u \neq s$ assume in some iteration of while loop:

$$u \in S^{t+1} \setminus S^t\ ,\ \delta[u] < d^t[u]$$

Consider *first* such t and u. Let P be shortest path from s to u.

$$s \in S^t \wedge u \notin S^t \rightarrow \exists \text{ edge } (x,y) \text{ on } P.\ x \in S^t \wedge y \notin S^t$$

$$d^t[y] = \delta[y]$$

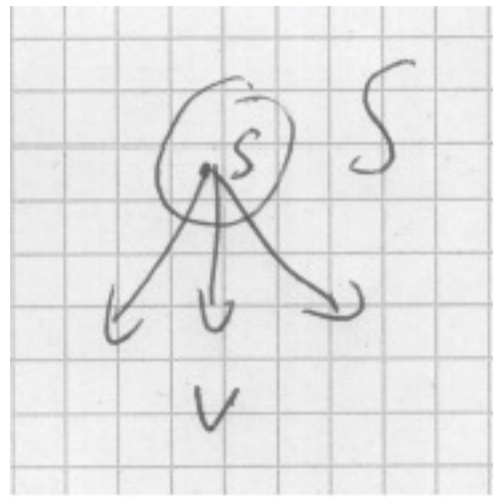
$$\begin{aligned} d^t[y] &= \delta[y] && \text{(above)} \\ &\leq \delta[u] && \text{(edge weights nonnegative)} \\ &< d^t[u] && \text{(assumption)} \end{aligned}$$

correctness

initialization

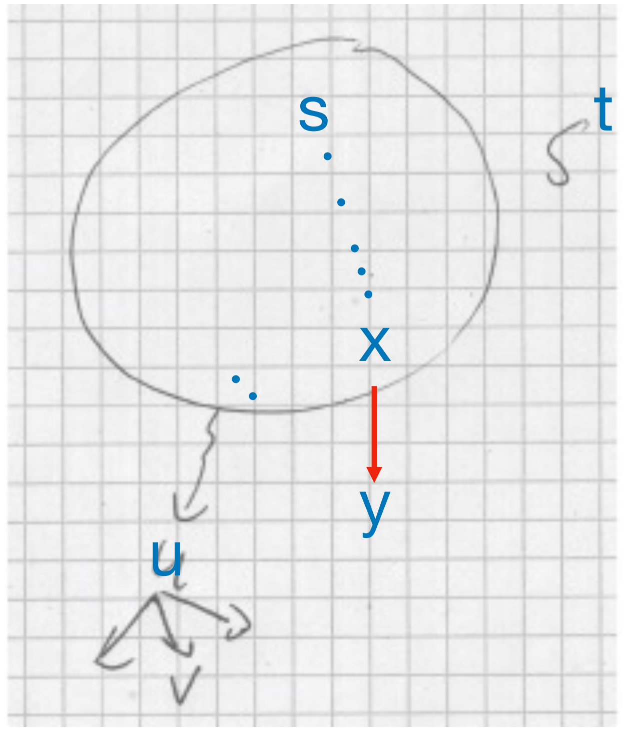
$S = \{s\};\ d[s] = 0;$
for all $v \neq s$

$$d[v] = \begin{cases} w(s,v) & (s,v) \in E \\ \infty & \text{otherwise} \end{cases}$$



iteration:

while $S \neq V$
{
choose $u \in V \setminus S$ with minimal $d[u]$;
$S = S \cup \{u\}$;
for all v with $(u,v) \in E$
$\{d[v] = \min\{d[v], d[u] + w(u,v)\}\}$
}



properties

1. shortest discovered distances nonincreasing; stable for nodes $u \in S$

$$d'[u] \leq d[u]\ ,\ u \in S \rightarrow d'[u] = d[u]$$

2. $d[u]$ is upper bound on length of real shortest paths

$$\delta[u] \leq d[u]$$

Lemma 4.

$$u = s \ \vee\ u \in S' \setminus S \ \rightarrow\ d[u] = \delta[u]$$

$u = s$ trivial. For $u \neq s$ assume in some iteration of while loop:

$$u \in S^{t+1} \setminus S^t\ ,\ \delta[u] < d^t[u]$$

Consider *first* such t and u. Let P be shortest path from s to u.

$$s \in S^t \wedge u \notin S^t \rightarrow \exists \text{ edge } (x,y) \text{ on } P.\ x \in S^t \wedge y \notin S^t$$

$$d^t[y] = \delta[y]$$

$$\begin{aligned} d^t[y] &= \delta[y] && \text{(above)} \\ &\leq \delta[u] && \text{(edge weights nonnegative)} \\ &< d^t[u] && \text{(assumption)} \end{aligned}$$

As $u \notin S^t$ is chosen in pass t with minimal $d[u]$

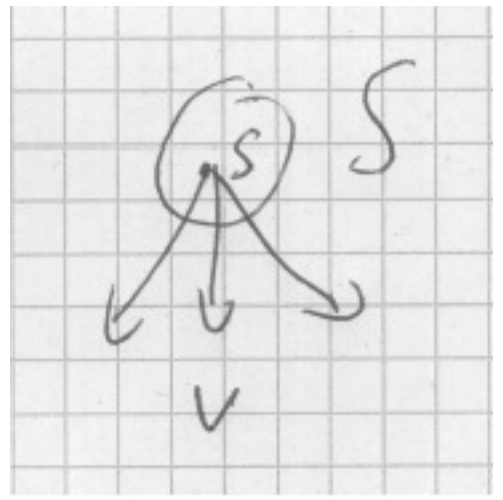
$$d^t[u] \leq d^t[y] \quad \text{contradiction}$$

run time

initialization

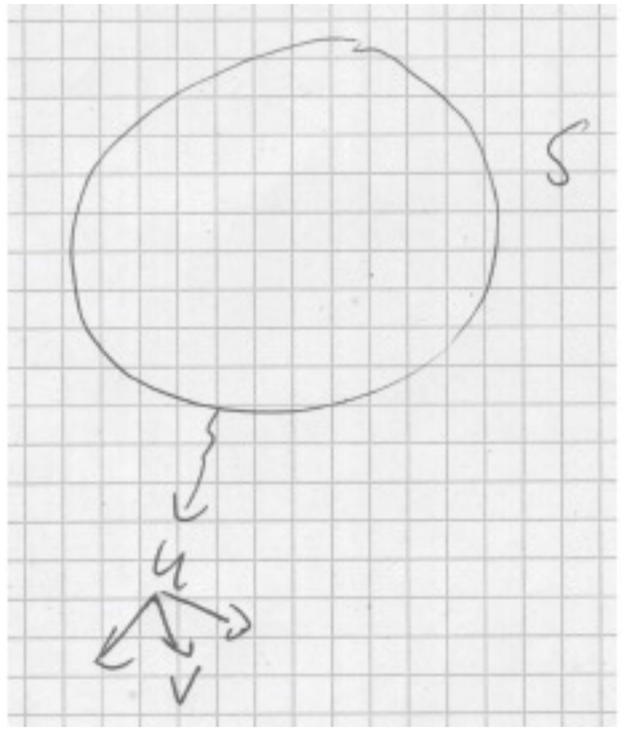
$S = \{s\};\ d[s] = 0;$
for all $v \neq s$

$$d[v] = \begin{cases} w(s,v) & (s,v) \in E \\ \infty & \text{otherwise} \end{cases}$$



iteration:

while $S \neq V$
{
choose $u \in V \setminus S$ with minimal $d[u]$;
$S = S \cup \{u\}$;
for all v with $(u,v) \in E$
$\{d[v] = \min\{d[v], d[u] + w(u,v)\}\}$
}



properties

1. shortest discovered distances nonincreasing; stable for nodes $u \in S$

$$d'[u] \leq d[u]\ ,\ u \in S \to d'[u] = d[u]$$

2. $d[u]$ is upper bound on length of real shortest paths

$$\delta[u] \leq d[u]$$

data structure for $V \setminus S$

balanced search tree: AVL or 2/3

records

u	d[u]	

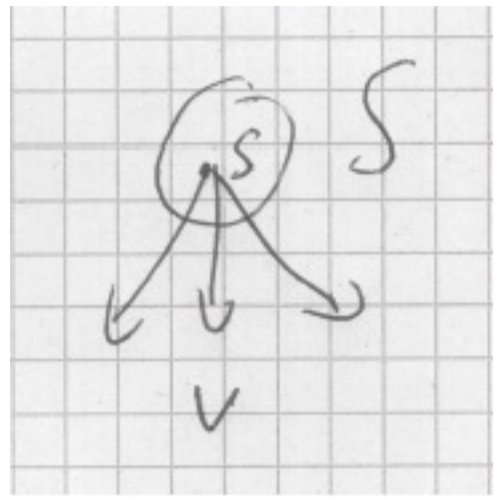
use d[u] as key;
pointers to these records in array elements a[u]

run time

initialization

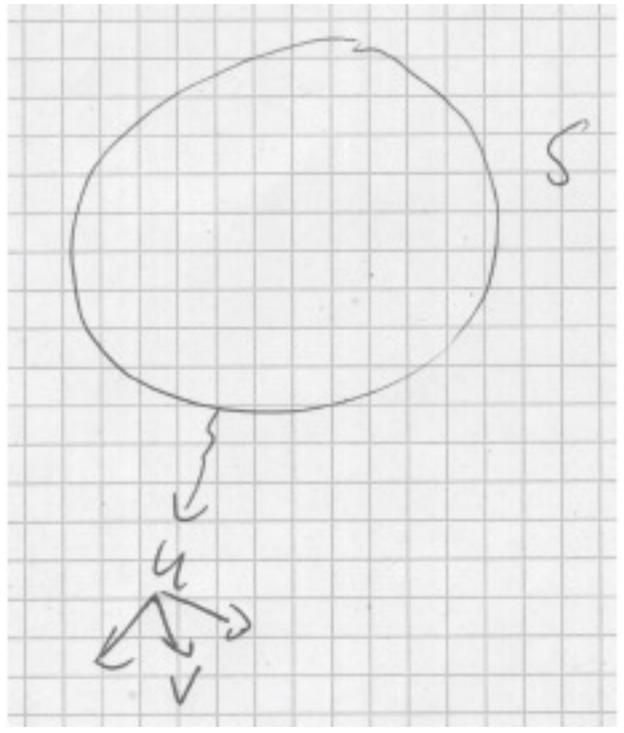
$S = \{s\}$; $d[s] = 0$;
for all $v \neq s$

$$d[v] = \begin{cases} w(s,v) & (s,v) \in E \\ \infty & \text{otherwise} \end{cases}$$



iteration:

while $S \neq V$
{
choose $u \in V \setminus S$ with minimal $d[u]$;
$S = S \cup \{u\}$;
for all v with $(u,v) \in E$
$\{d[v] = \min\{d[v], d[u] + w(u,v)\}\}$
}



properties

1. shortest discovered distances nonincreasing; stable for nodes $u \in S$

$$d'[u] \leq d[u] \ , \ u \in S \rightarrow d'[u] = d[u]$$

2. $d[u]$ is upper bound on length of real shortest paths

$$\delta[u] \leq d[u]$$

data structure for $V \setminus S$

balanced search tree: AVL or 2/3

records

u	d[u]	

use d[u] as key;
pointers to these records in array elements a[u]

initialization

at most |V| insert operations

iterations

at most |E| operations

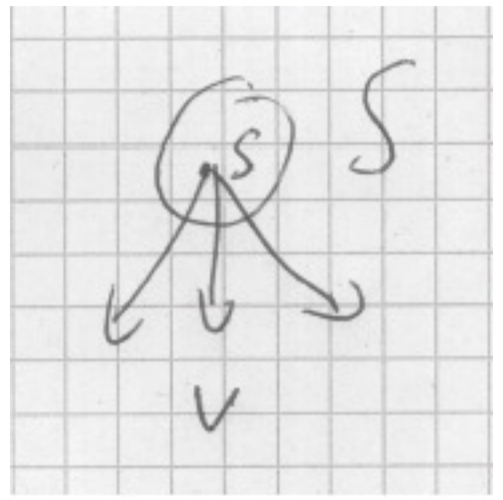
find
- ~~delete~~ min (go over lef edges or sth.)
- delete ~~v~~ u
- ~~insert v with~~ d'[v]
update

run time

initialization

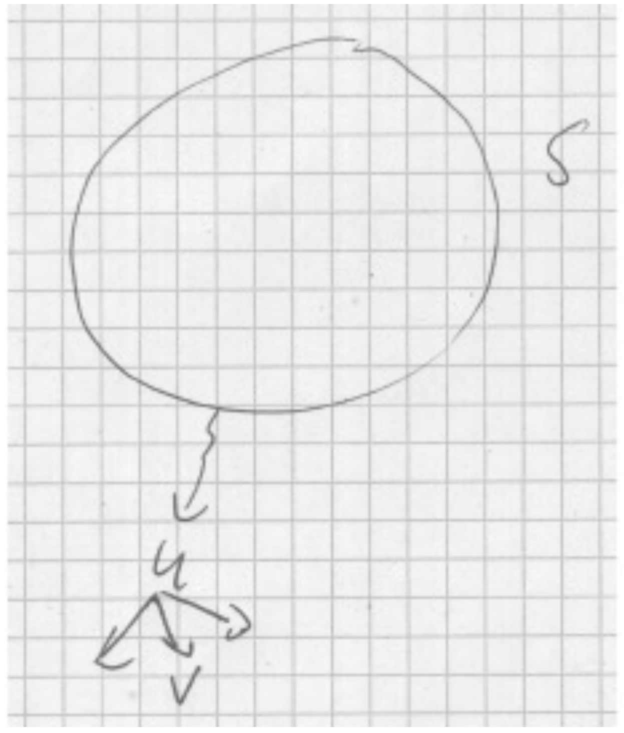
$S = \{s\}$; $d[s] = 0$;
for all $v \neq s$

$$d[v] = \begin{cases} w(s,v) & (s,v) \in E \\ \infty & \text{otherwise} \end{cases}$$



iteration:

while $S \neq V$
{
choose $u \in V \setminus S$ with minimal $d[u]$;
$S = S \cup \{u\}$;
for all v with $(u,v) \in E$
$\{d[v] = \min\{d[v], d[u] + w(u,v)\}\}$
}



properties

1. shortest discovered distances nonincreasing; stable for nodes $u \in S$

$$d'[u] \leq d[u] \text{ , } u \in S \rightarrow d'[u] = d[u]$$

2. $d[u]$ is upper bound on length of real shortest paths

$$\delta[u] \leq d[u]$$

data structure for $V \setminus S$

balanced search tree: AVL or 2/3

records

u	d[u]	

use d[u] as key;
pointers to these records in array elements a[u]

initialization

at most |V| insert operations

This log V time for E operat.

iterations

at most |E| operations
- delete min
- delete v
- insert v with d'[v]

nodes of tree maintaining V\S at most V.

$\boxed{\text{run time = } O(|E|\log|V|)}$