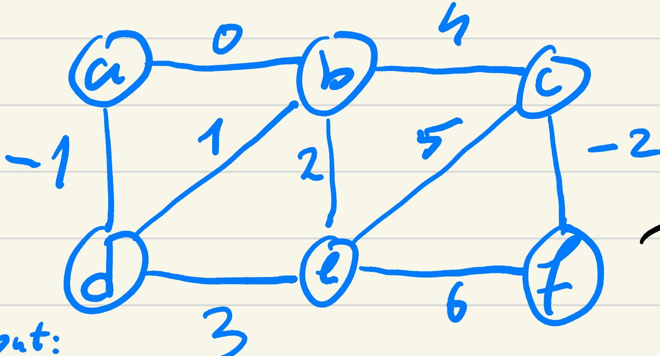
10.17

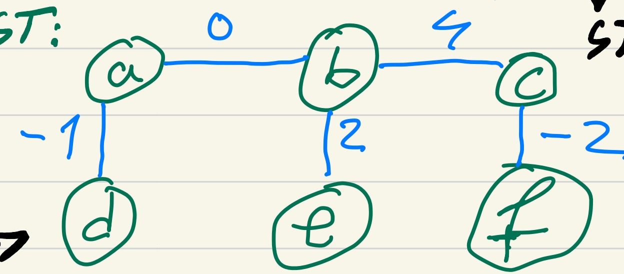
Élsúlyozott gráfok ábrázolásai:

ld. ad2jegyzetGrafok2.pdf (Canvas)

I. Minimális feszítőfa (Minimum $\underbrace{\text{Spanning Tree}}_{ST}$: MST)



MST:



Input:
összefüggő, élsúlyozott, ir. tlan gráf

(ST)
Feszítőfa: { lefedi az összes csúcsot & öf. körmentes részgráf

MST: minimális $\underbrace{\text{össz-élsúlyú}}$ feszítőfa [w(MST) min.]

$\text{GenMST}(G : \mathcal{G}_w \; ; \; A : \mathcal{E}\{\})$

$A := \{\} \; ; \; k := \lvert G.V \rvert - 1$
// k edges must be added to A
$k > 0$
find an edge (u, v) that is safe for A
$A := A \cup \{(u, v)\} \; ; \; k - -$

2.5. Definíció. *Tegyük fel, hogy $G = (V, E)$ élsúlyozott, irányítatlan, összefüggő gráf, és $A \subseteq$ a G valamelyik minimális feszítőfája élhalmazának! Ekkor az $(u, v) \in E$ él* biztonságosan hozzávehető *az A élhalmazhoz* (safe for *A*), *ha $(u, v) \notin A$ és $A \cup \{(u, v)\} \subseteq$ a G valamelyik (az előzővel nem okvetlenül egyező) minimális feszítőfája élhalmazának.*

2.6. Következmény. *Tegyük fel, hogy $G = (V, E)$ élsúlyozott, irányítatlan, összefüggő gráf! Ha egy kezdetben üres A élhalmazt újabb és újabb biztonságosan hozzávehető éllel bővítünk, akkor $\lvert G.V \rvert - 1$ bővítés után éppen a G egyik minimális feszítőfáját kapjuk meg.*

A kérdés most az, hogyan tudunk mindig biztonságos élet választani az A élhalmazhoz. Ehhez lesz szükségünk az alábbi fogalmakra és tételre.

Az MST „A" élhalmazát kell meghatározni.

Pl. a fenti gráfra $A := \{\}$, majd sorban $c \overset{2}{—} f$, $a \overset{1}{—} d$, $a \overset{0}{—} b$, $b \overset{2}{—} e$, $b \overset{2}{—} c$ biztonságosan hozzávehető.

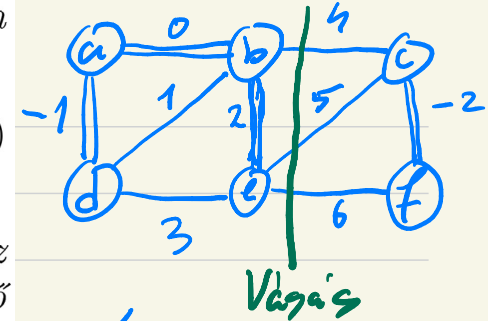
2.7. Definíció. *Ha* $G = (V, E)$ *gráf és* $\{\} \subsetneq S \subsetneq V$, *akkor a* G *gráfon* $(S, V \setminus S)$ *egy* vágás.

2.8. Definíció. $G = (V, E)$ *gráfon az* $(u, v) \in E$ *él* keresztezi *az* $(S, V \setminus S)$ vágást, *ha* $(u \in S \wedge v \in V \setminus S) \vee (u \in V \setminus S \wedge v \in S)$.

2.9. Definíció. $G = (V, E)$ *élsúlyozott gráfon az* $(u, v) \in E$ könnyű él *az* $(S, V \setminus S)$ vágásban, *ha* (u, v) *keresztezi a vágást, és* $\forall (p, q)$, *a vágást keresztező élre* $w(u, v) \le w(p, q)$.

2.10. Definíció. *A* $G = (V, E)$ *gráfban az* $A \subseteq E$ *élhalmazt* elkerüli *az* $(S, V \setminus S)$ vágás, *ha az A egyetlen éle sem keresztezi a vágást.*

2.11. Tétel. *Ha a* $G = (V, E)$ *irányítatlan, összefüggő, élsúlyozott gráfon*
(1) A részhalmaza a G valamelyik minimális feszítőfája élhalmazának,
(2) az $(S, V \setminus S)$ *vágás elkerüli az A élhalmazt, és*
(3) az $(u, v) \in E$ könnyű él *az* $(S, V \setminus S)$ vágásban,
$\Longrightarrow$ *az* (u, v) *él biztonságosan hozzávehető az A élhalmazhoz.*



$b \xrightarrow{4} c$, $c \xrightarrow{5} e$, $e \xrightarrow{6} f$ keresztezi a vágást.

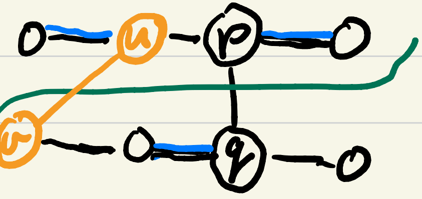
$b \xrightarrow{4} c$ könnyű él a vágásban.

A vágás elkerüli az $\{(a,b), (a,d), (b,e), (c,f)\}$ élhalmazt: MST része

$\Rightarrow (b,c)$ biztonságosan hozzávehető.

Biz. Legyen $A \subseteq T$: MST-nek! Feltehető: $(u,v) \notin T$

T:



vágás: $\exists (p,q)$: keresztezi

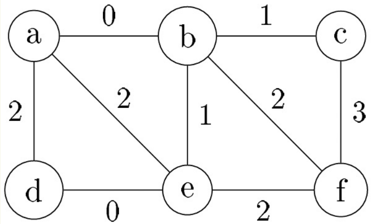
$T' := T \setminus (p,q) \cup (u,v) \Rightarrow T'$: ST $\Rightarrow w(T') \geq w(T)$

$w(T') = w(T) - w(p,q) + w(u,v) \leq w(T)$

$w(u,v) \leq w(p,q)$

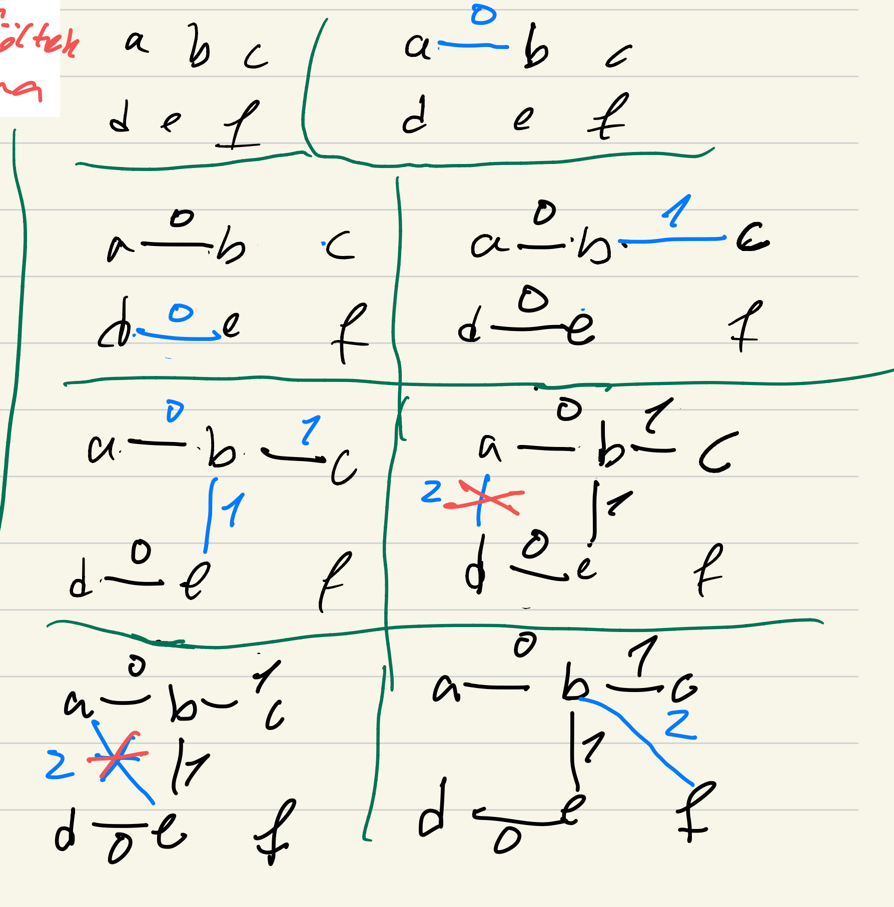
$\Rightarrow w(T') = w(T) \Rightarrow T'$: MST $\Rightarrow \checkmark$

1. Kruskal algoritmus



a – b, 0 ; d, 2 ; e, 2.
b – c, 1 ; e, 1 ; f, 2.
c – f, 3. ← nem kerültek sorra
d – e, 0.
e – f, 2. ←

Komponensek	$u \overset{w}{—} v$	bizt?
a, b, c, d, e, f	$a \overset{0}{—} b$	+
ab, c, d, e, f	$d \overset{0}{—} e$	+
ab, c, de, f	$b \overset{1}{—} c$	+
abc, de, f	$b \overset{1}{—} e$	+
abcde, f	$a \overset{2}{—} d$	–
~	$a \overset{2}{—} e$	–
~	$b \overset{2}{—} f$	+
abcdef		



$MT(n,m)$? $m \geq n-1$ (G öf.)

Kruskal($G : \mathcal{G}_w$; $A : \mathcal{E}\{\}$) : $\mathbb{N}$

$\forall v \in G.V$ — $\Theta(n)$
makeSet(v) // a spanning forest of single vertices is formed
$A := \{\}$; $k := \lvert G.V \rvert$ — $\Theta(1)$
// k is the number of components of the spanning forest
// let Q be a minimum priority queue of $G.E$ by weight $G.w$:
Q : minPrQ($G.E, G.w$) — $\Theta(m)$
$k > 1 \wedge \neg Q$.isEmpty() — $\leq m$-szer it.
$e : \mathcal{E} := Q$.remMin() — $O(\log m) \leq O(\log n)$
$x :=$ findSet($e.u$) ; $y :=$ findSet($e.v$) — $O(\log n)$
$x \neq y$ — $\Theta(1)$
$\underbrace{A := A \cup \{e\}}_{\Theta(1)}$; $\underbrace{\text{union}(x,y)}_{\Theta(1)}$; $k--$ | SKIP
return k — $O(1)$

$\Theta(m) \leq O(m)$

Loop body: $O(\log n)$; loop: $O(m \log n)$

Total: $O(m \log n)$

This algorithm checks whether G is connected. If so, it returns $k = 1$. Otherwise $k > 1$.

$MT(n,m) \in O(m * \lg n)$

$m < n^2 \Rightarrow \log m < 2 \cdot \log n$

$MT_{findSet(u)} \in O(\log n)$: ehhez elég: $MT_{findSet(u)} \in O(\log(\lvert set(u) \rvert))$

2.2.1 The set operations of the Kruskal algorithm

makeSet(v : $\mathcal{V}$)

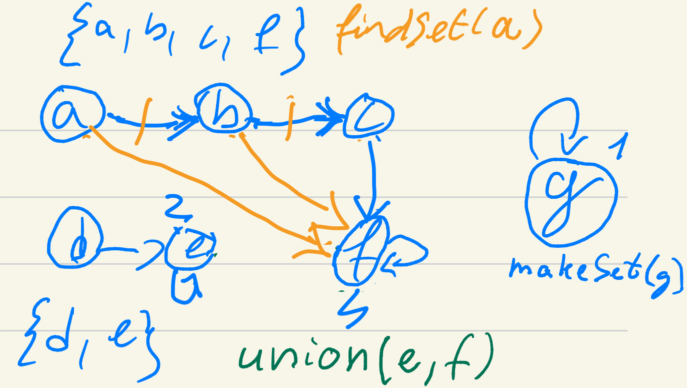
$\pi(v) := v$
$s(v) := 1$

findSet(v : $\mathcal{V}$) : $\mathcal{V}$

$\pi(v) \neq v$	
$\pi(v) :=$ findSet($\pi(v)$)	SKIP
return $\pi(v)$	

union($x, y : \mathcal{V}$)

$s(x) < s(y)$	
$\pi(x) := y$	$\pi(y) := x$
$s(y) += s(x)$	$s(x) += s(y)$



Áll. $MT_{findSet(u)}(k) \in O(\log k)$, ahol $k = |\underbrace{set(u)}|$ az u csúcsot tart. csúcshalmaz mérete. (Ezt a gyökere felé irányított fával reprezentálták.)

Megoldás: Jelölje „h" az „u" csúcshoz tart. irott fa magasságát!

Biz.: $h \le \log k$: ez kezdetben igaz, és union tartja

① makeSet(u) után $h=0 \wedge k=1$: $h = 0 = \log 1 = \log k \Rightarrow h \le \log k$ ✓

$h(x) \le \log(s(x))$ $\quad$ $h(y) \le \log(s(y))$

② union(x,y): előtte $h(x)$:= „x gyökerű fa magassága"; $s(x)$:= „a mérete". Ugyanígy „y"-ra.

Biz.: $h(r) \le \log s(r)$

Feltehető: $h(x) \le \log(s(x)) \wedge h(y) \le \log(s(y)) \wedge s(x) \ge s(y)$. Jel. „r" a fa gyökerét a union után! $\Rightarrow h(r) = \max(h(x), h(y)+1)$

2.a) $h(x) > h(y) \Rightarrow h(r) = h(x) \le \log s(x) \le \log(s(x)+s(y)) = \log s(r)$

2.b) $h(x) \le h(y) \Rightarrow h(r) = h(y)+1 \le \log s(y) + \log 2 = \log(2s(y)) \le \log(s(x)+s(y)) = \log s(r)$ ✓

2.2.1 The set operations of the Kruskal algorithm

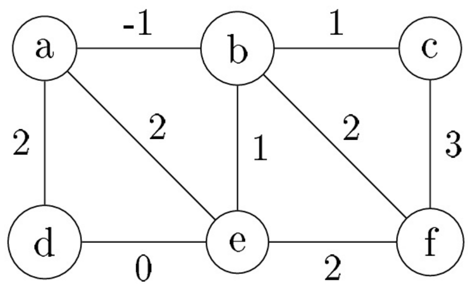
makeSet($v : \mathcal{V}$)
- $\pi(v) := v$
- $s(v) := 1$

findSet($v : \mathcal{V}$) : $\mathcal{V}$
- if $\pi(v) \neq v$: $\pi(v) :=$ findSet($\pi(v)$) | else: SKIP
- **return** $\pi(v)$

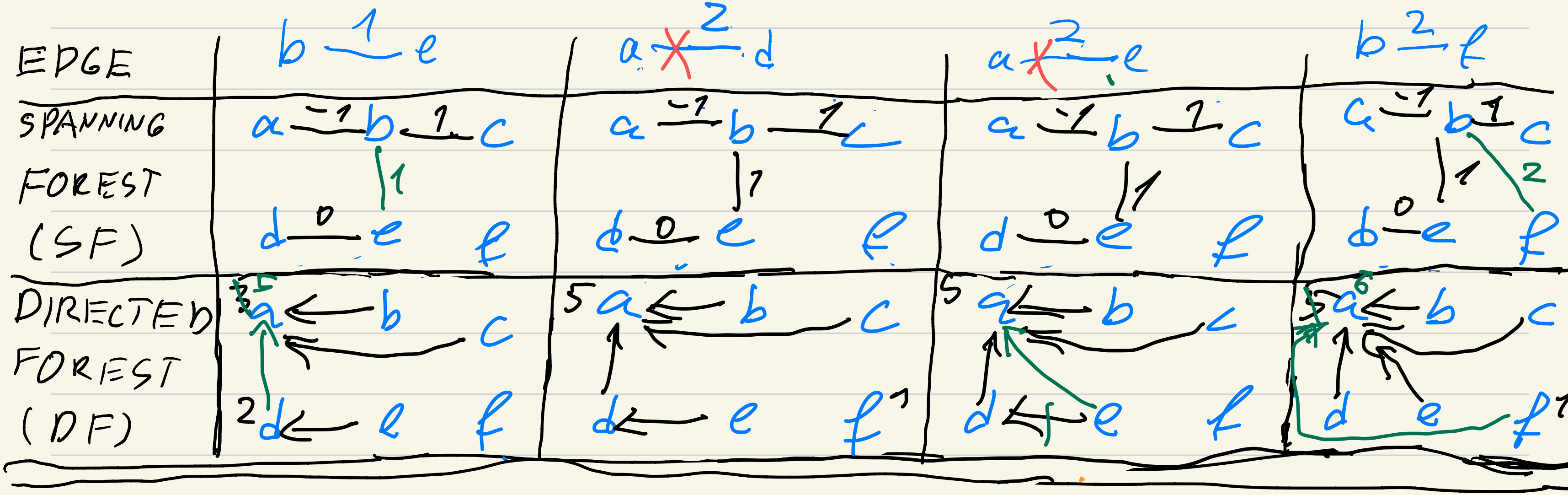
union($x, y : \mathcal{V}$)
- if $s(x) < s(y)$: $\pi(x) := y$; $s(y) += s(x)$
- else: $\pi(y) := x$; $s(x) += s(y)$

makeSet() and union() ensure that for each directed tree, $h \leq \log s$ where h is the height of the tree and s is its size. Thus findSet() runs in $O(\log n)$ time.

EDGE		$a \xrightarrow{-1} b$	$d \xrightarrow{0} e$	$b \xrightarrow{1} c$
SPANNING FOREST (SF)	$a \quad b \quad c$ $d \quad e \quad f$	$a \overset{-1}{—} b \quad c$ $d \quad e \quad f$	$a \overset{-1}{—} b \quad c$ $d \overset{0}{—} e \quad f$	$a \overset{-1}{—} b \overset{1}{—} c$ $d \overset{0}{—} e \quad f$
DIRECTED FOREST (DF)	$^1a \quad ^1b \quad ^1c$ $^1d \quad ^1e \quad ^1f$	$^2a \leftarrow b \quad c^1$ $d^1 \quad e^1 \quad f^1$	$^2a \leftarrow b \quad c^1$ $^2d \leftarrow e^1 \quad f^1$	$^3a \leftarrow b$, $a \leftarrow c$ $^2d \leftarrow e \quad f^1$



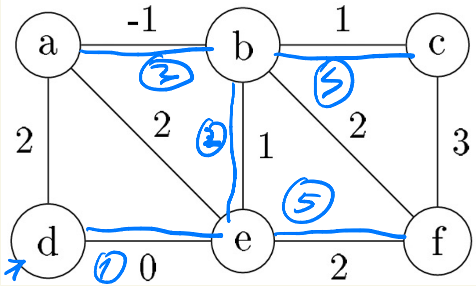
a – b, -1 ; d, 2 ; e, 2.
b – c, 1 ; e, 1 ; f, 2.
c – f, 3.
d – e, 0.
e – f, 2.



② PRIM algoritmus: Tetsz. vál. r csúcsból indul:

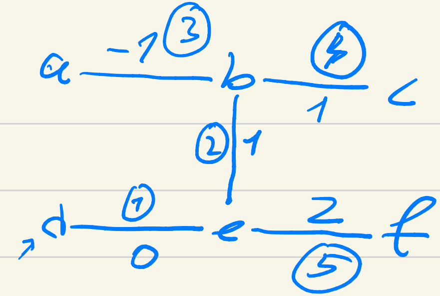
részleges MST: élenként/csúcsonként bővíti: $\begin{cases} v \text{ csúcs} \\ (v, p(v)) \text{ él}: c(v) = w(v, p(v)) \end{cases}$

$\boxed{11.07}$



r = d

a — b, -1 ; d, 2 ; e, 2.
b — c, 1 ; e, 1 ; f, 2.
c — f, 3.
d — e, 0.
e — f, 2.



MST -be	c & p címkék változásai					
	a	b	c	d	e	f
	∞ ∅	∞ ∅	∞ ∅	0 ∅	∞ ∅	∞ ∅
d	2 d				0 d	
1 e $\overset{0}{—}$ d		1 e				2 e
2 b $\overset{1}{—}$ e	-1 b		1 b			
3 a $\overset{-1}{—}$ b						
4 c $\overset{1}{—}$ b						
5 f $\overset{2}{—}$ e						

Prim($G : \mathcal{G}_w$; $r : \mathcal{V}$)

$\forall v \in G.V$
$c(v) := \infty$; $p(v) := \emptyset$ // costs and parents still undefined
// edge $(p(v), v)$ will be in the MST where $c(v) = G.w(p(v), v)$
$c(r) := 0$ // r is the root of the MST where $p(r)$ remaines undefined
// let Q be a minimum priority queue of $G.V \setminus \{r\}$ by label values $c(v)$:
Q : minPrQ($G.V \setminus \{r\}, c$) // $c(v)$ = cost of light edge to (partial) MST
$u := r$ // vertex $u = r$ has become the first node of the (partial) MST
$\neg Q$.isEmpty()
// neighbors of u may have come closer to the partial MST
$\forall v \in G.A(u) : v \in Q \wedge c(v) > G.w(u, v)$
$p(v) := u$; $c(v) := G.w(u, v)$; Q.adjust(v)
$u := Q$.remMin() // $(p(u), u)$ is a new edge of the MST

$\Theta(n)$

$\Theta(1)$

$\Theta(n)$

$\Theta(1)$

$\Theta(n)$ $\cap$ $O(n)$ $MT(n,m)$

$O((n+m)\log n)$

Δ

$(n(n-1)$ it.$)$

$O(\log n)$

$\forall$ végrehajtásra $O(m \log n)$

$O(\log n)$

$O(n \log n)$

$(n-1) * \Theta(n)$

Szomsz.list.ábr.+ Q min.kupac: G.öf.: $m \geq n-1$

$mT(n,m), MT(n,m) \in O((n+m)\log n) = O(m \log n)$

Csúcsmtx.+ rlen.tömb$_\Delta$a$_\Delta$Q: $MT(n)$, $mT(n) \in \Theta(n^2)$ Δ (Q.adjust(v): SKIP)