$A \Rightarrow B$
$\neg B \Rightarrow \neg A$

$r \approx \frac{1}{2} \log n$

$= O(\frac{n}{r} \log n)$ pts of S.

$\Rightarrow$ in each iter,

$|R| = O(r + 3 \cdot \frac{n}{r} \log n)$

Set $r = \sqrt{n}$ $\Rightarrow$ $O(\sqrt{n} \log n)$

$\Rightarrow$ $T(n) \le 4\, T(\sqrt{n} \log n) + O(n)$

ignore

$\Rightarrow$ $O(n + 4\sqrt{n} + 16 n^{1/4} + \dots)$

$= \boxed{O(n)}$

Rmk: generalizes to min enclos. ball in d dims
or LP in d vars

$T(n) \le (d+2)\, T(d\sqrt{n}) + O(d^2 n)$

base case: $T(d^2) = O((d^2)^{O(d)})$

$\Rightarrow$ $T(n) = O(d^2 n + d^{O(d)})$.

linear for any const d.

(improves to $O(d^2 n + 2^{\tilde{O}(\sqrt{d})})$ rand.
current record for "combinatorial algms" ...)

deterministic? Megiddo $2^{2^d} n$ time
Chazelle-Matoušek '93/ $d^{O(d)} n$ time
C. '16

idea - derandomize Clarkson
ε-nets can be constructed in $\sim n^d$ time
by greedy hitting set algm

new idea. divide into small groups
of size $d^{O(1)}$

new idea: divide into small ... of size $d^{O(1)}$
compute ε-net of each group
take union of ε-nets $\quad \varepsilon \sim \frac{1}{2d^2}$

$$T(n) \approx (d+2)\, T\left(\frac{n}{2d}\right) + O(d^2 n)$$

$$\Rightarrow O(\square\, n) \ldots$$

Min Spanning Tree (MST)

$n-1 \leq m \lesssim n^2$

Given weighted undir. connected graph $G=(V,E)$, $\quad n=|V|$, $m=|E|$

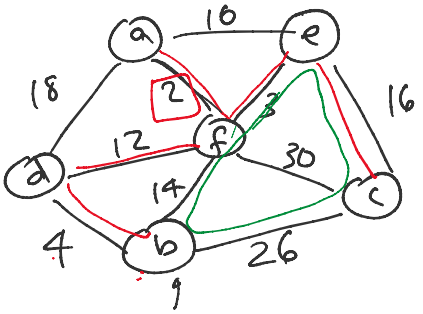
find min-weight spanning tree $MST(G)$

uses all vertices

connected & acyclic

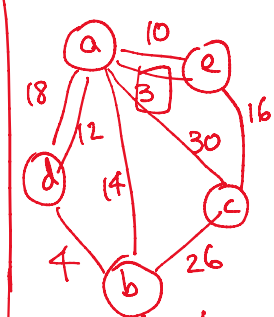
(if wts > 0, min spanning connected $\Rightarrow$ acyclic)

$2+3+\cancel{14}^{12}+4+16 = \cancel{49}\ 47$

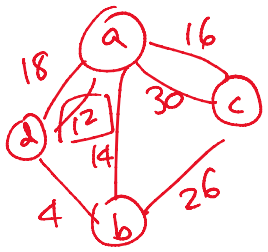


(assume wts are distinct) $\Rightarrow$ MST unique)

Prim: $s=a$
contract f to a



contract e to a



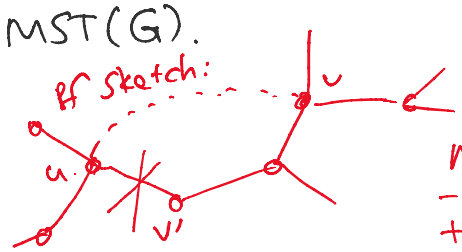
⋮

Inclusion Rule

If u's nearest neighbor is v,
then $uv \in MST(G)$.

So, can output uv
& contract u to v
in G.

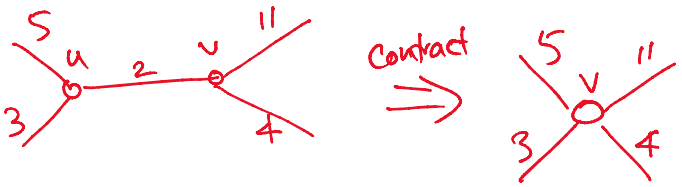
Pf sketch:



MST
$-uv'$
$+uv$

$\square$

e.g.



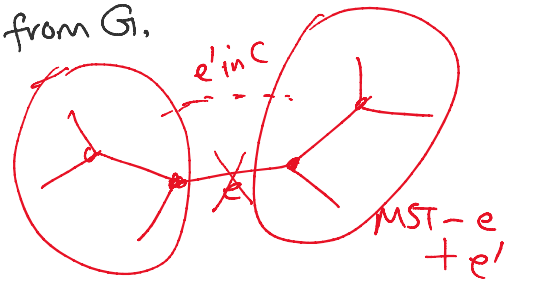
Exclusion Rule

If a cycle C's heaviest edge is e,

then $e \notin MST(G)$,

then $e \notin MST(G)$,
So, can delete e from G.

Pf Sketch:



□

Rmk: all classical MST alg'ms use only these two rules.

eg. Kruskal's Alg'm ('56):

```
repeat {
    take lightest edge uv      ← by inclusion rule
    output uv
    contract u, v
}
```

implementation: sorting + union-find data structure (inverse Ack. $m\,\alpha(n)$)
$\Rightarrow O(m \log n)$ time

Prim's Alg'm ('57):

```
fix s ∈ V
repeat {
    find s's nearest neighbor v
    contract v to s
}
```

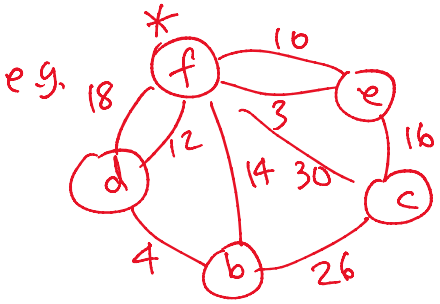
implementation: heap $\Rightarrow O(m \log n)$ time
"Fibonacci heap" $\Rightarrow O(n \log n + m)$ time
(decrease-key; linear if not too sparse)

Borůvka's Alg'm ('26):

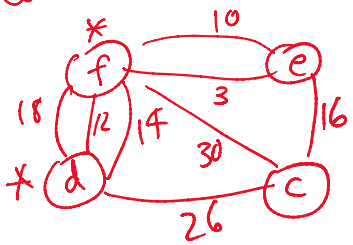
```
repeat {
    BoruvkaStep()
}
BoruvkaStep():
    1. unmark all vertices
```

1. unmark all vertices
2. for each $u \in V$
3. if u unmarked {
4. find u's nearest neighbor v
5. output uv
6. contract u to v
7. mark v
}

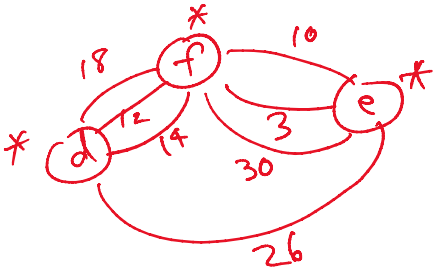
e.g.



Contract b to d



Contract c to e



end of one Boruvka step

Implementation: no data structures (just linked lists)

line 4 $O(\deg(u))$ time

line 6 $O(\deg(u))$ time

$\Rightarrow$ time for one Boruvka step

$$= O\left(\sum_{u \in V} \deg(u)\right)$$

$$= O(m)$$

after one Boruvka step.

vertices $\leq n/2$

because each marked vertex is a contraction of ≥ 2 vertices

$\Rightarrow T(m,n) \leq T(m, \frac{n}{2}) + O(m)$

$\Rightarrow$ $O(m \log n)$ time

A "hybrid" alg'm:

for $i = 1$ to r do BoruvkaStep()

Prim()

$\Rightarrow$ time $O(rm + (\frac{n}{2^r} \log n + m))$

$O(n)$

Set $r = \log\log n$

$= O(m \log\log n)$ time!!

⋮