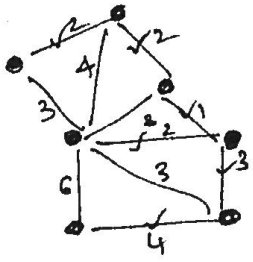
Shortest tour

→ luckily, has a constant factor approximation.

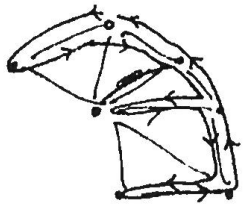
the idea: use spanning tree
- → cover all vertices
- → have no cycles
- → connected subgraph

Minimum Spanning Tree?
- ↳ has minimum weight



how can we use a minimum spanning tree to find shortest tour?

A walk: traverse exactly each edge twice and covers all the vertices



Approx. algorithm
- ⇒ find minimum spanning tree $O(n^2)$
- → The walk of this tree is the soln. $O(n^2)$

Weight of m.s.t. < length of shortest tour

⇒ shortest tour is longer than walk on any spanning tree (since shortest tour has a cycle)

∴ weight of m.s.t < len. of shortest tour.

length of walk on m.s.t.

$$w(m.s.t) < len(S.To.)$$

$$len(walk(m.s.t)) = 2 \cdot w(m.s.t)$$

$$\Rightarrow \quad len(walk(m.s.t.)) < 2 \cdot len(S.To.)$$

↳ upper bound of approx. algor.

We already know that

$$len(walk(m.s.t.)) \geq len(S.To.)$$

(equal in the case of a tree)

We have just shown that the algo. is 2-factor approximate

$$len(walk(m.s.t)) \geq len(S.To.) > \frac{1}{2} len(walk(m.s.t))$$

2. $len(S.To.) > len(walk(m.st.)) \geq len(S.To.)$

2-factor bound

What about clique & independent set?

Can we use reductions & approximations together?

Can we have a factor-2 approximation for them?

↳ Reduction can mess with the approximation factor.

n vertices

V.C.

Size of min. V.C $\Rightarrow k$

$\Rightarrow$

I.S

Size of max. ind. set $\Rightarrow n-k$

approx $\Rightarrow \leq 2k$

redn. $\Rightarrow \geq n-2k$

sum should be n

factor $= \frac{2k}{2} = 2$ (min.)

factor $= \frac{n-k}{n-2k} = 1 + \frac{k}{n-2k}$

Is $1 + \frac{k}{n-2k}$ a useful approximation factor ?

NOPE :(

→ it depends on the min. vertex cover
→ it can be calculated only in hindsight

For clique & i.s. there is no constant factor approx. algo. (unless P = NP)

Knowing part of soln can make finding optimum soln. much faster.