Approximation Problems

Goal:

- Discuss Optimization
- Introduce Approximation
- Problems/Algorithms

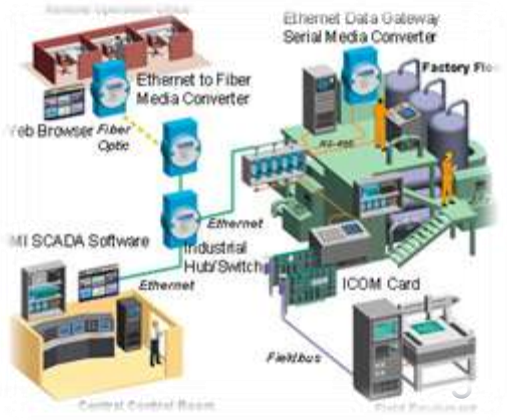
Plan:

- VERTEX-COVER, SET-COVER
- Greedy Algorithm
- TSP

Network Power

Assume a network **with some links** between **components**. A link requires power supply; need to supply power

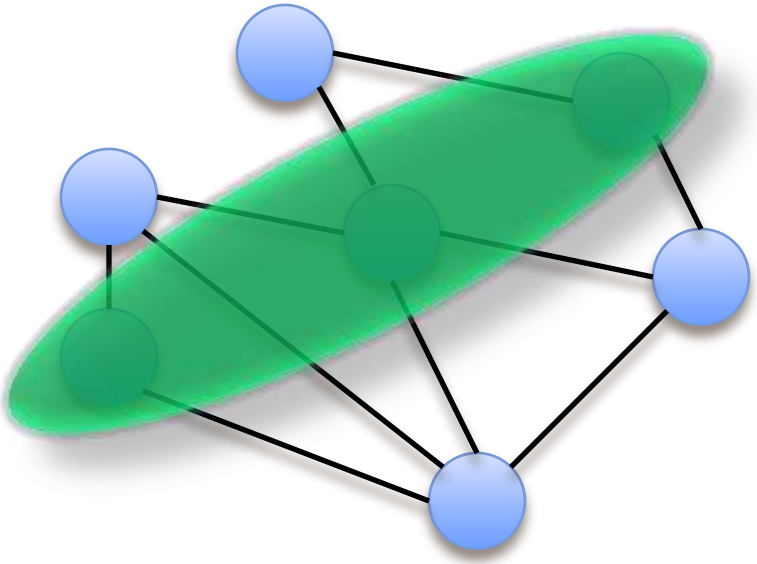
to some nodes so as to cover all links. Obviously, you'd like to power the smallest number of nodes. Can you find such a set?



VERTEX-COVER

$C \subseteq V$ is a *cover* of $G=(V, E)$

- If $\forall (u,v) \in E$, $u \in C$ or $v \in C$



Instance:

- An undirected $G=(V,E)$

Minimization Problem:

- find a *minimal* cover C

Instance:

- An undirected $G=(V,E)$, k

Decision Problem:

- Is there a cover C, $|C|=k$?

Theorem:

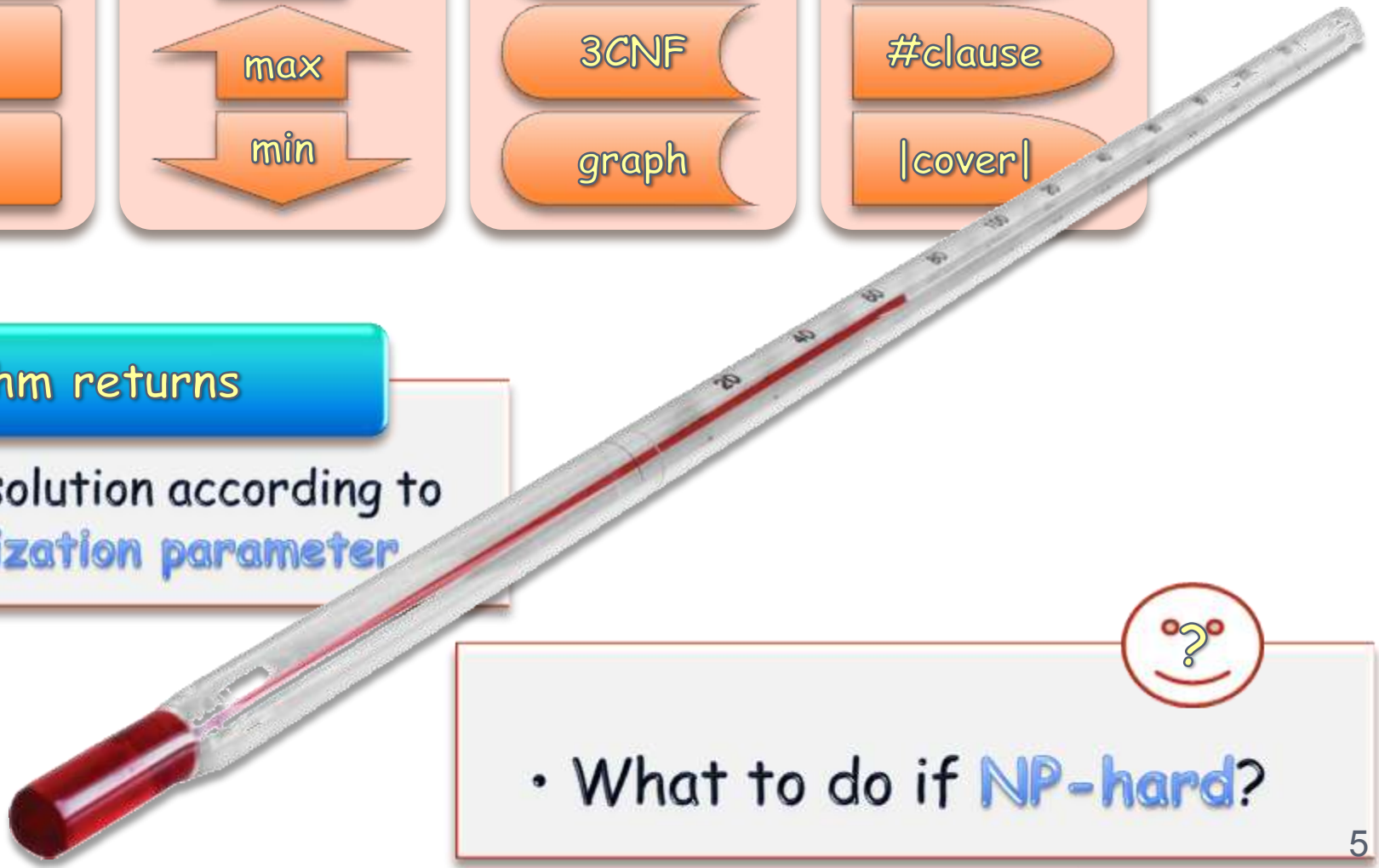
- Min-V.C. is NP-hard

Proof:

- For a cover C, $V \setminus C$ is an independent-set ■

Optimization

problem	Min/Max	Instance	Parameter
CLIQUE	max	graph	|clique|
3SAT	max	3CNF	#clause
V.C.	min	graph	|cover|



Algorithm returns

- Best solution according to optimization parameter

- What to do if NP-hard?

Approximation

Definition:

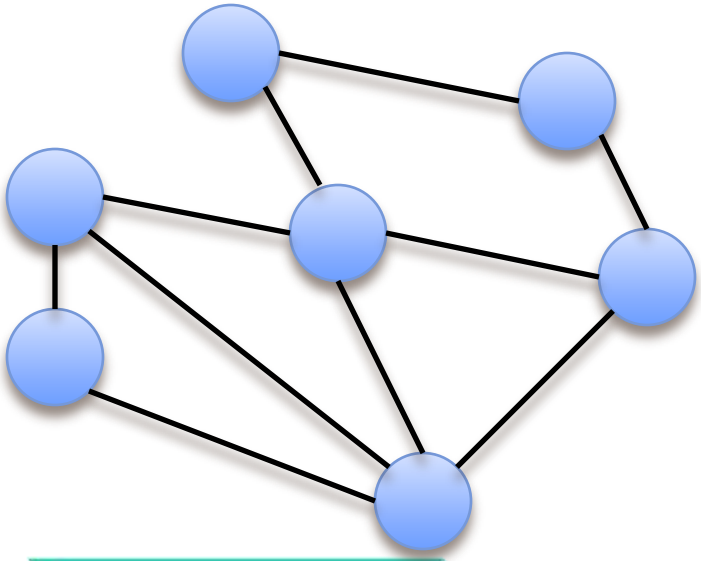
- An α-approximation algorithm for a maximization/minimization problem with an optimum solution O, returns a solution that is $\geq \alpha O / \leq \alpha O$

Note

- α and O may depend on the size of the input

VC - Approximation Algorithm

1. $C \leftarrow \phi$
2. begin while C not a cover
3. Pick $(u,v) \in E$ s.t. $u,v \notin C$
4. $C \leftarrow C \cup \{u,v\}$
5. end while
7. return C



note: algorithm runs in poly-time

Theorem:

$|C| \leq 2$ optimal

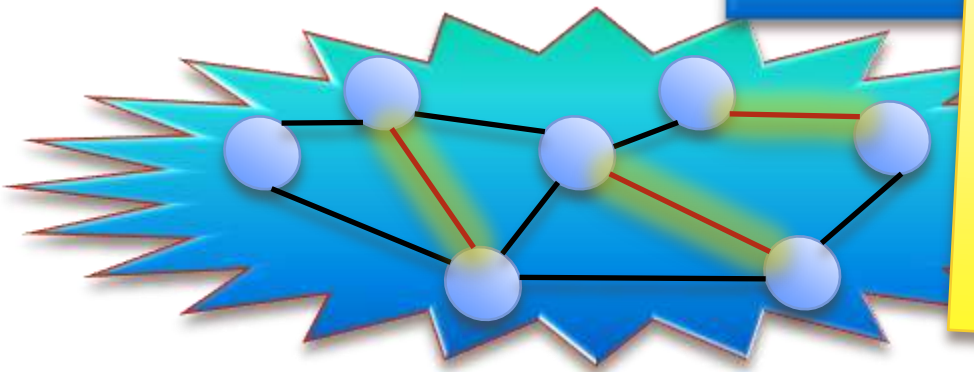
note: C is a V.C.

Proof:

Edges algorithm picks:

Have no common vertexes

Optimal V.C. must contain ≥1 end



Mass Mailing

You'd like to send some message to a large list of recipients (e.g. all campus)

Some mailing-lists are available, however, each list charges $1 for each message sent

You'd like to find the smallest set of lists that covers all recipients

"We must be on a mailing list."

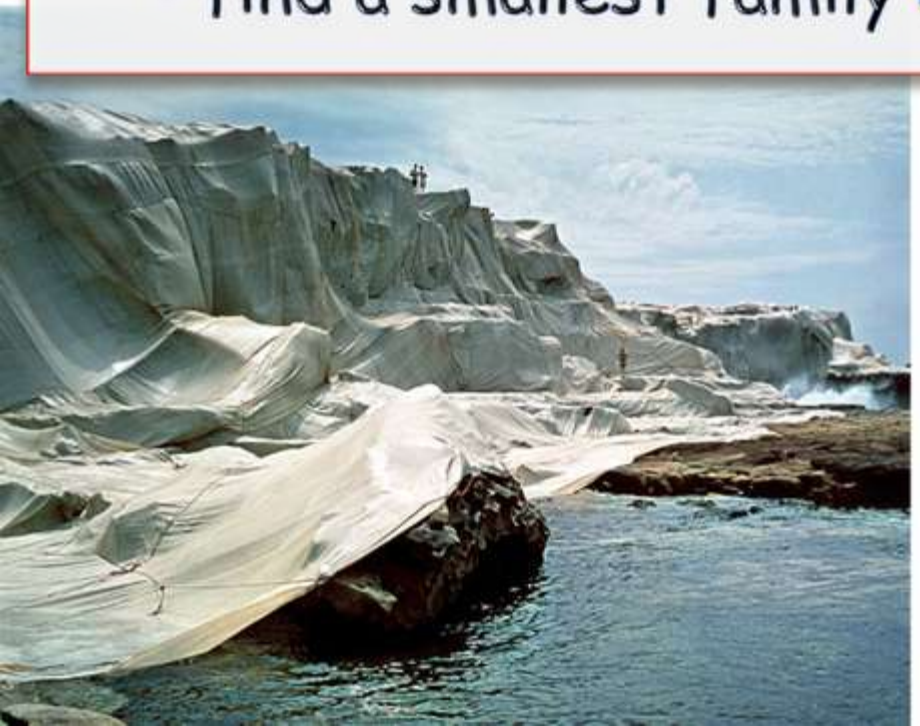
SET-COVER

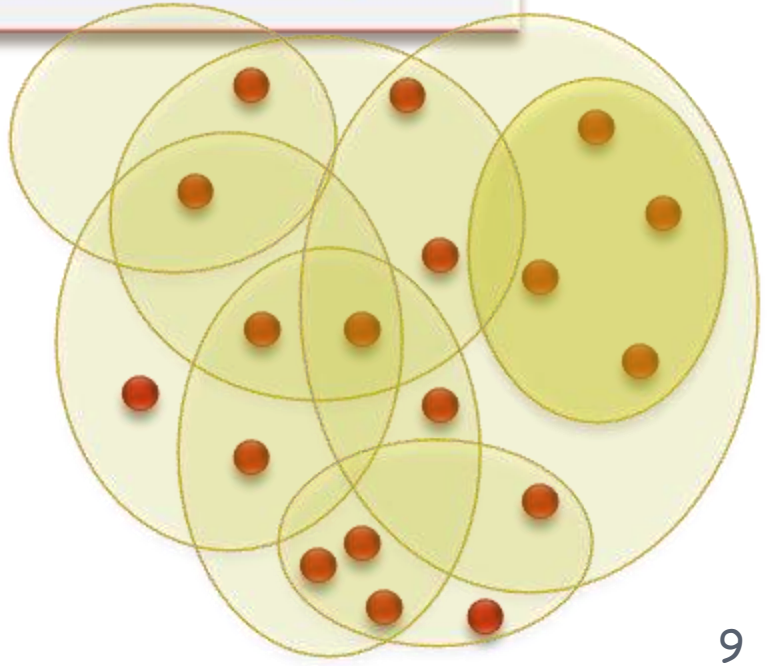
Instance:

- a finite set U and a family F of subsets of U, which covers U

Minimization Problem:

- find a smallest family $C \subseteq F$ that *covers* U



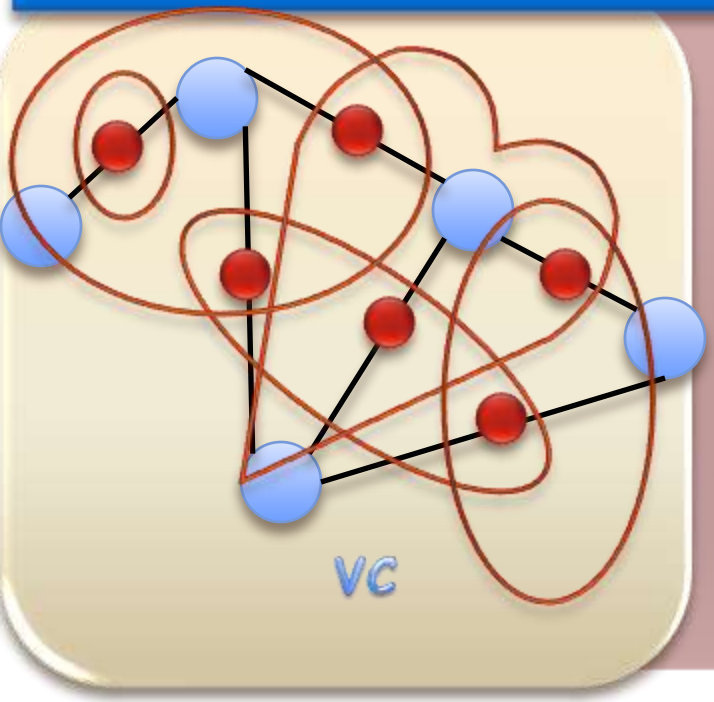


Theorem:

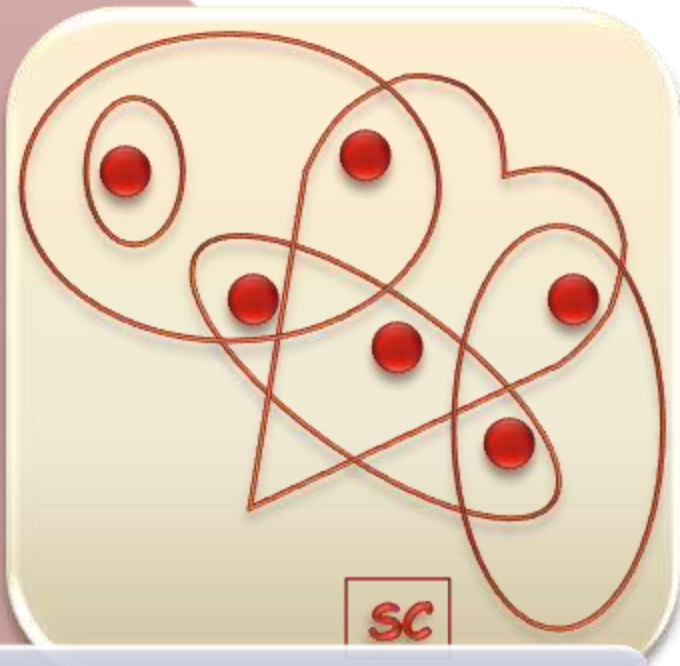
ET-COVER is NP-Hard

- SET-COVER is NP-Hard

Proof:



V.C.
$\leq_L$
SET-COVER



U

- 1 element for each edge

F

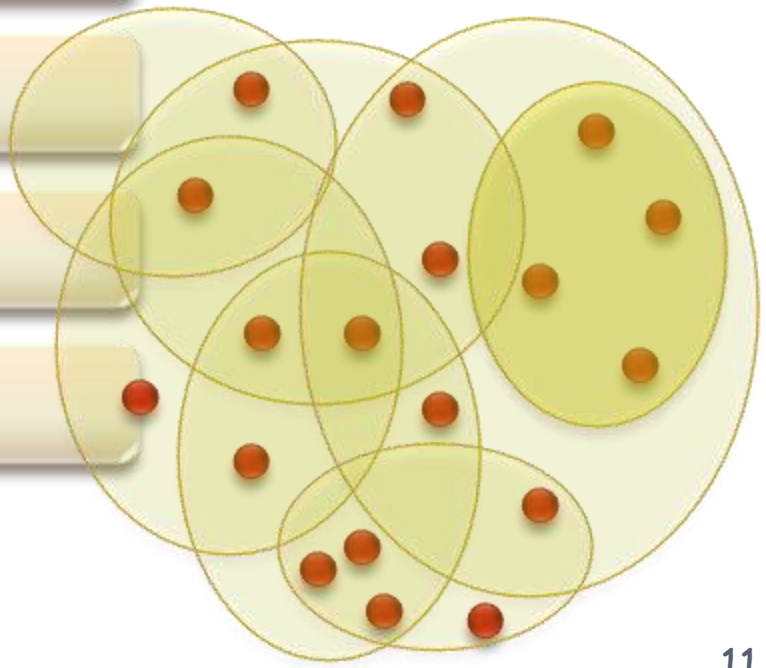
- 1 set for each vertex, comprising adjacent edges

The Greedy Algorithm

1. $C \leftarrow \phi$, $U \leftarrow X$
2. **begin** while $U \neq \phi$
3. Pick $S \in F$ that maximizes $|S \cap U|$
4. $C \leftarrow C \cup \{S\}$
5. $U \leftarrow U - S$
6. **end** while
7. return C

note: algorithm runs in poly-time

note: C is a S.C.



Just:

- Described the greedy algorithm for SET-COVER

Next:

- Analyze its approximation ratio in 3 distinct ways: $\lg_2 n$, $\ln n$, even better

How to prove an Approximation Ratio?

Need to:

- compare the size of the cover returned by the greedy algorithm to optimal

However

- The optimal is unknown

Observation:

- $\exists$k-cover $\Rightarrow$ any part of the universe has a k -cover!

Corollary:

- Each step of the greedy algorithm removes 1/k of elements

Observe:

- After k(=size of optimal S.C.) stages the algorithm covers at least $\frac{1}{2}$ of U

Proof:

- By way of contradiction, assume $<\frac{1}{2}$ are covered
  The uncovered part of U intersects with a set in F in >n/2k elements
  Hence, all previous k stages have covered >n/2k elements
  And must have covered >kn/2k=n/2

It can be covered by k sets

Let $S_1, \ldots, S_t$ be the sequence of sets picked by the algorithm

Let U_i be the set of elements not yet covered after i stages

Note $|U_{i+1}| = |U_i - S_{i+1}| \leq |U_i|(1-1/k)$

U_i can be covered by k sets

Hence $|U_{ik}| \leq |U_0|(1-1/k)^{ik} \leq |U|e^{-i}$

and $t \leq k \ln(n) + 1$

Best Ratio-Bound

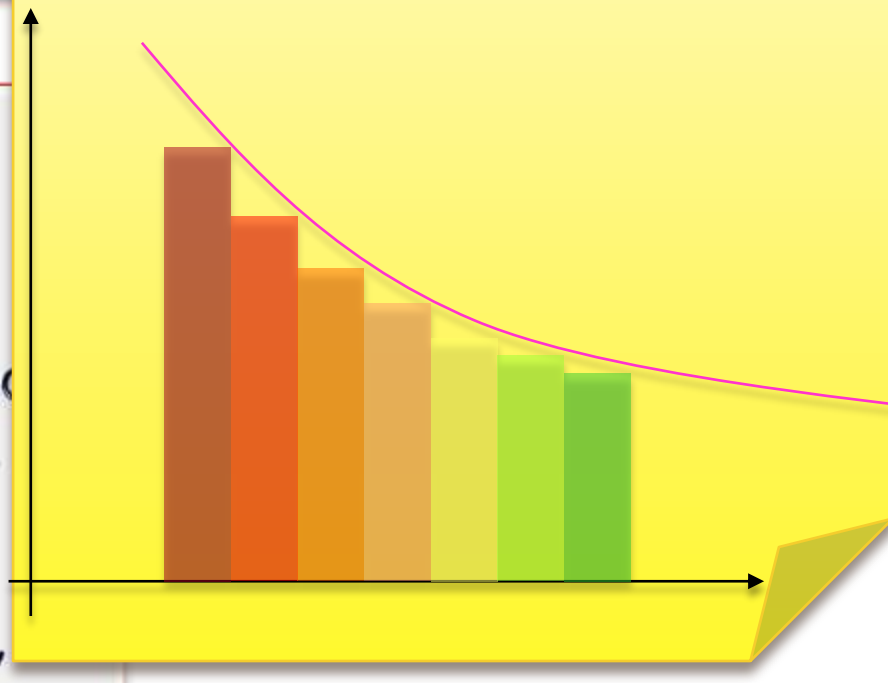
Lemma:

- Greedy algorithm approximates the optimal set-cover to within a factor $H(\max\{ |S|: S \in F \})$

Proof:

- Split the "cost of \$1", for set S_i picked i^{th} by the greedy algorithm, among newly covered
- Now, bound the sum paid, over any $S \in F$, by $H(|S|)$
- "Imagine" the optimal solution, and bound the total paid

$$H(n) = \sum_{k=1}^{n} \frac{1}{k} = \sum_{k=2}^{n} \frac{1}{k} + 1 \leq \int_{1}^{n} \frac{1}{x} dx + 1 = \ln n + 1$$



Generalized Tour Problem

- Each segment of the tour problem now has a cost
- find a **least-costly** tour

Approximate the optimal tour?

i.e. - find a tour that costs, say, no more than twice as much as the least costly.

Next:

- NP-hard optimization problems
- Approximate to within a certain factor
- NP-hard or in P?

Plan:

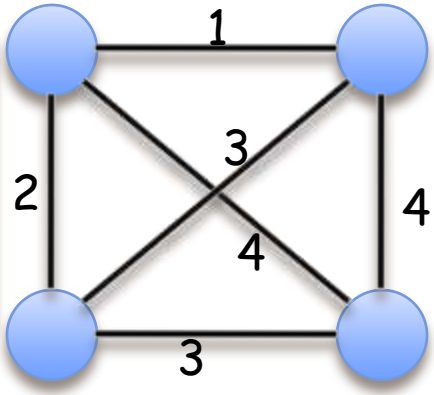
- Traveling Salesman Problem (TSP)
- TSP is NP-hard
- Approximation algorithm for special cases
- Inapproximability result

"What's my favorite book?
How about 'Death of a Salesman'?"

Traveling Salesperson Problem

Instance:

- A complete weighted undirected $G=(V,E)$ (non-negative weights)

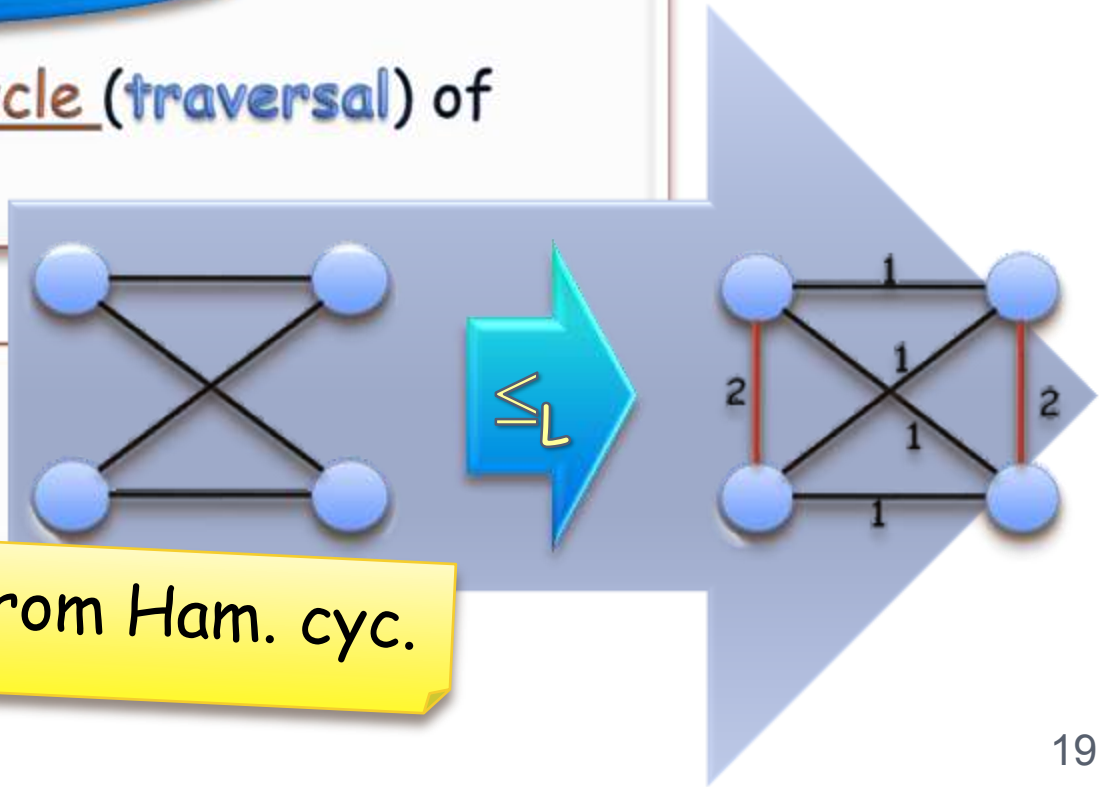


Minimization Problem:

- find a Hamiltonian cycle (traversal) of minimal cost

Theorem:

- TSP is NP-hard



By a simple **reduction** from Ham. cyc.

Next: show a 2-Approximation algorithm for TSP, in case the cost function satisfies the triangle inequality

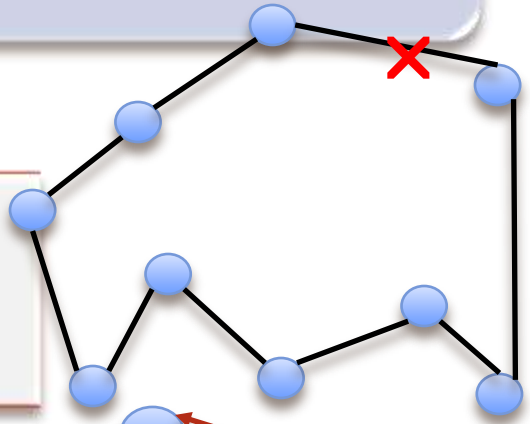
$\forall u,v,w \in V:$
$c(u,v)+c(v,w) \geq c(u,w)$

via

Min Spanning Tree

Approximation Algorithm

1. Find a Minimum Spanning Tree (MST) T for G
2. Traverse along DFS of T --- jump over visited

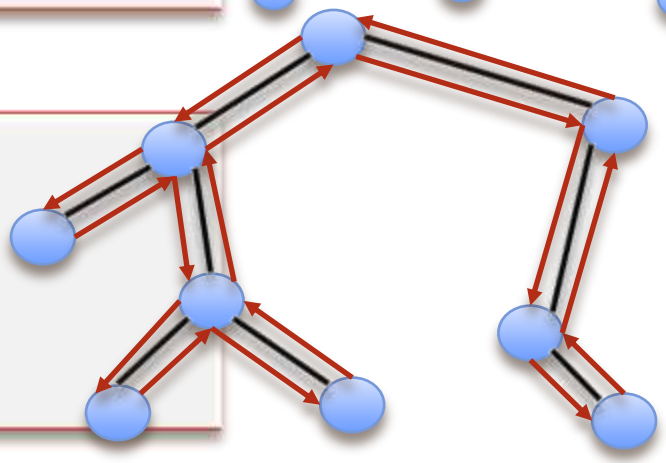


Observation

- MST weight ≤ Cheapest Traversal

Observation

- Algorithm's traversal costs 2 weight MST



Next:

- Show it is NP-hard to approximate TSP -the general case- to within any factor $h \geq 1$

How?

- Introduce its *gap version*
- Show it is NP-hard

gap-TSP

Instance:

- a complete weighted undirected graph $G=(V,E)$

GAP Problem: $[|V|, h|V|]$

- distinguish between the following cases:

Yes

- $\exists$Hamiltonian cycle of cost $\leq |V|$

No

- The cost of $\forall$Hamiltonian cycle $\geq h|V|$

gap-TSP

Algorithm returns

- Depending on the **least costly traversal**

GAP

- Whatever

Observation:

- Efficient **h**-approximation for **TSP** resolves **gap-TSP[C, hC]**

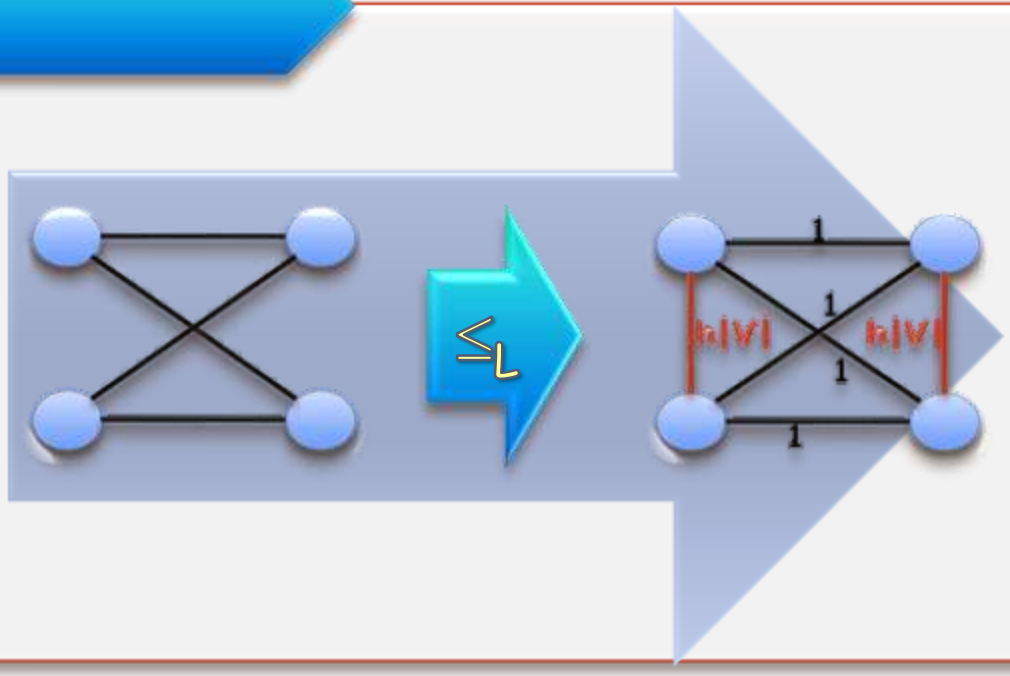
gap-TSP is NP-hard

Theorem:

- For any $h \geq 1$, HAM-CYCLE $\leq_L$ gap-TSP[|V|, h|V|]

Proof:

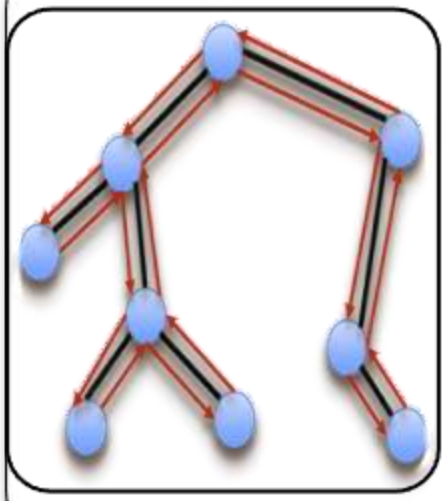
-



Corollary:

- Approximating TSP to within any factor is NP-hard

Synopsis



Some NP-hard optimization problems can be efficiently approximated:

- VERTEX-COVER (2)
- SET-COVER (ln n)
- TSP[triangle ineq.] (2)

For some factors it may still be NP-hard

We've introduced GAP problems for that purpose

WWindex