Approximation Algorithms (Cormen 35)

Now we focus on optimization problems e.g.

- Given $G=(V,E)$
  Find a vertex cover U of G with $|U|$ minimum

- Given $G=(V,E)$
  Find a longest cycle

- Given $G=(V,E)$
  Find a maximum clique

- Given a 3-SAT formula φ over variables $x_1, \dots x_n$
  Find a truth assignment which satisfies the maximum # of clauses of φ (MAX-3-SAT)

- Given a complete graph K_n with a cost function on its edges
  Find a TSP tour of minimum cost

- Given a set $S = \{x_1, x_2, \dots, x_n\}$ of integers and $t \in \mathbb{Z}$

  Find a subset $S' \subseteq S$ which minimizes

$$t - \sum_{x_i \in S'} x_i \quad \text{while} \quad \sum_{x_i \in S'} x_i \le t$$

Our goal: Find, in polynomial time, a solution C such that C is 'close' to C^* when C^* is the value of an optimal solution

We think of C as both a solution and its value

An algorithm A for a problem P has approximation ratio $\rho(n)$ if $\boxed{\max\left\{\frac{C}{C^*}, \frac{C^*}{C}\right\} \leq \rho(n)}$ for input of size n

For a minimization problem we have $C \geq C^*$ so we want $\frac{C}{C^*} \leq \rho(n)$

For a maximization problem we have $C \leq C^*$ so we want $\frac{C^*}{C} \leq \rho(n)$

Example 1 Vertex cover

We will show a very simple 2-approximation algorithm, that is $\frac{C}{C^*} \leq 2$ independent of n

Basic observation: For every vertex cover X of G and

(*) For each edge $uv \in E$ $|X \cap \{u,v\}| \geq 1$

Algorithm $\mathcal{A}$:

$X \leftarrow \emptyset$, $E' \leftarrow E$

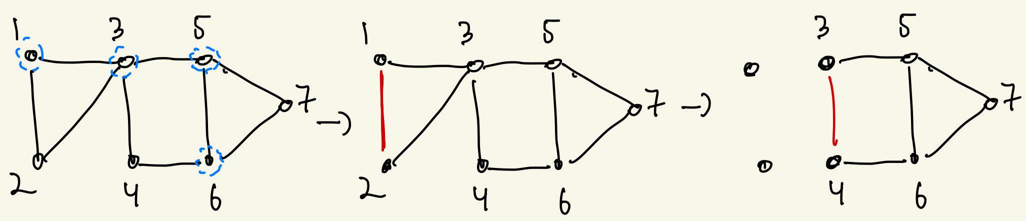
while $E' \neq \emptyset$

pick $uv \in E'$ arbitrarily

$X \leftarrow X \cup \{u,v\}$

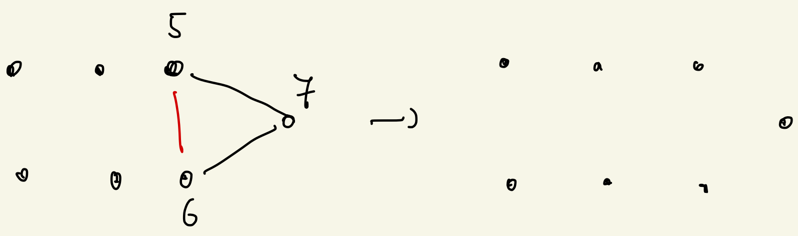
delete all edges incident with u or v from E'

Return X



$X = \emptyset$

$X = \{1,2\}$

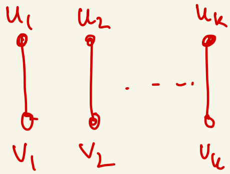


$X = \{1,2,3,4\}$

$X = \{1,2,3,4,5,6\}$

why is it a 2-approximation algorithm?

- look at the special edges u_iv_i that we extract from E'
- They form a matching: $\{u_i, v_i\} \cap \{u_j, v_j\} = \emptyset$ when $i \neq j$



$|X| = 2k$

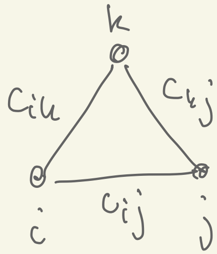
- By (*) $|X^*| \geq k$ when X^* is a minimum vertex cover
- Hence $\dfrac{|X|}{|X^*|} \leq 2$

The algorithm $\mathcal{A}$ runs in polynomial time in $n = |V|$, $m = |E|$

Hence $\mathcal{A}$ is a 2-approximation alg for vertex cover

TSP with Δ-inequality

A TSP instance (K_n, c) satisfies the Δ-inequality if $c_{ij} \le c_{ik} + c_{kj}$ $\forall i,j,k \in V(K_n)$



Idea: Find a minimum spanning tree T^* of (K_n, c) and use this to construct a hamiltonian cycle H s.t $\frac{c(H)}{c(H^*)} \le 2$

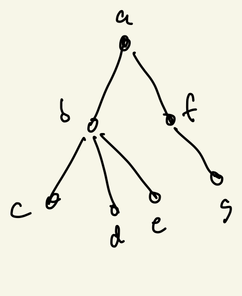
when H^* is an optimal (min cost) TSP tour

Observation 1 $c(T^*) \le c(H^*)$

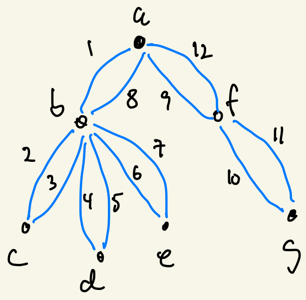
- recall that $c_{ij} \ge 0$ $\forall ij$
- If we delete an edge of H^* we get a spanning tree T' of K_n so $c(T') \ge c(T^*)$

- Hence $c(H^*) \ge c(T') \ge c(T^*)$ $\square$.

Double walk of a spanning tree:



T



W

$$c(W) = 2 \cdot c(T)$$

$W = abcbdbebafgfa$

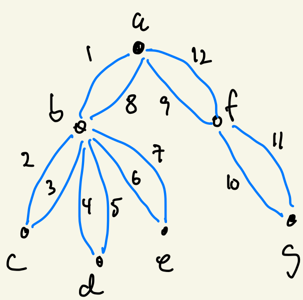
Shortcutting a double walk:

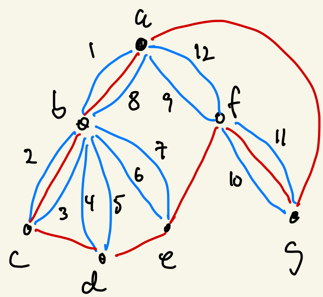
Keep only first occurrence of an internal vertex on walk

$$W = abcbdbebafgfa \longrightarrow abc\cancel{b}d\cancel{b}e\cancel{b}\cancel{a}fg\cancel{f}a = abcdefga$$

$\parallel$

H

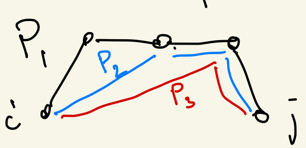




H is a hamilton cycle and

Observation 2 $c(H) \le c(W)$

This follows from the Δ-inequality



$$c(P_1) \ge c(P_2) \ge c(P_3) \ge c_{ij}$$

A': input (K_n, c)
output H with $c(H) \leq 2c(H^*)$

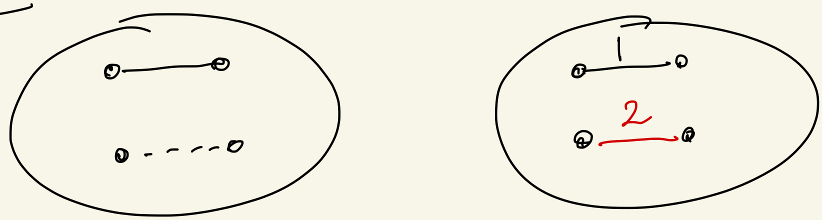
1. Construct a minimum spanning tree T^*
2. Form the double walk W of T^*
3. Shortcut W to a hamiltonian cycle H
4. Return H

$$c(H) \underset{\text{observation 2}}{\leq} c(W) = 2c(T^*) \underset{\text{observation 1}}{\leq} 2c(H^*)$$

so $\dfrac{c(H)}{c(H^*)} \leq 2$

<u>Theorem</u> TSP is NPC even if input costs satisfy the Δ-inequality

<u>proof</u>



G $\qquad\qquad$ (K_n, c) $\quad n = |V|$

- $c_{ij} \le c_{ik} + c_{kj}$ $\forall i,j,k$ so Δ-inequality holds

- (K_n, c) has a hamilton cycle of cost n

$\Updownarrow$

G has a hamiltonian cycle

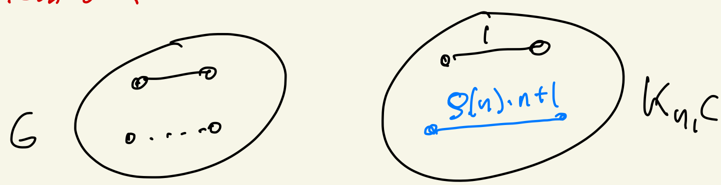
so Hamilton cycle $\le_p$ TSP with Δ-ineq

Did we really need the Δ-inequality?

Theorem Unless $P = NP$ there is no $\varrho(n)$-approximation algorithm for general TSP (no assumptions on costs) for any function $\varrho(n)$

Proof: Suppose $\mathcal{C}$ is a polynomial algorithm which given (K_n, C) with $c_{ij} \geq 0 \ \forall i,j$ finds a hamilton cycle H s.t $\frac{c(H)}{c(H^*)} \leq \varrho(n)$.

Let G be an instance of Hamilton cycle and define (K_n, C) with $n = |V(G)|$:



- If G has a hamilton cycle then $c(H^*) = n$
- Every hamilton cycle of K_n which uses at least one blue edge has cost at least $(n-1) + \varrho(n)\cdot n + 1 = (\varrho(n)+1)\cdot n > \varrho(n)\cdot n$
- $\mathcal{C}$ is an $\varrho(n)$-apx alg so $c(H) = n$ must hold if G has a hamilton cycle as $\frac{c(H)}{c(H^*)} \leq \varrho(n)$

- Thus $\mathcal{C}$ solves Hamcycle in polynomial time $\Rightarrow P = NP$