Approximations : Sep 22, 2015

2-approximation for metric TSP

- Find a minimum spanning tree
- Duplicate all the edges
- Find an Euler path
- Shortcut by skipping previously visited nodes

MST $\leq$ OPT

$\times 2$

✓

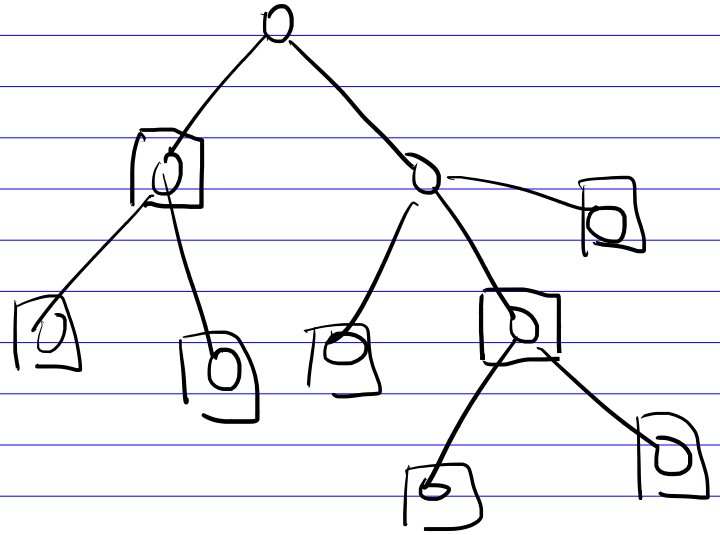
"reduced the cost" $\therefore$ Δ-inequality

Cost of path $\leq 2\,OPT$

In order to improve the "2",

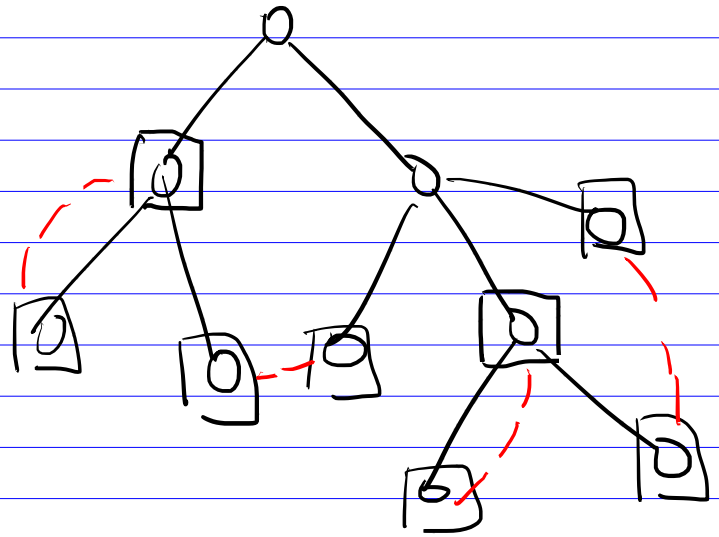
we need a better way of making MST Eulerian

Problem: vertices of odd degree



odd degree nodes is even

Pair up all odd degree nodes



All the edges added form a matching - M

M^* = minimum cost matching in G.

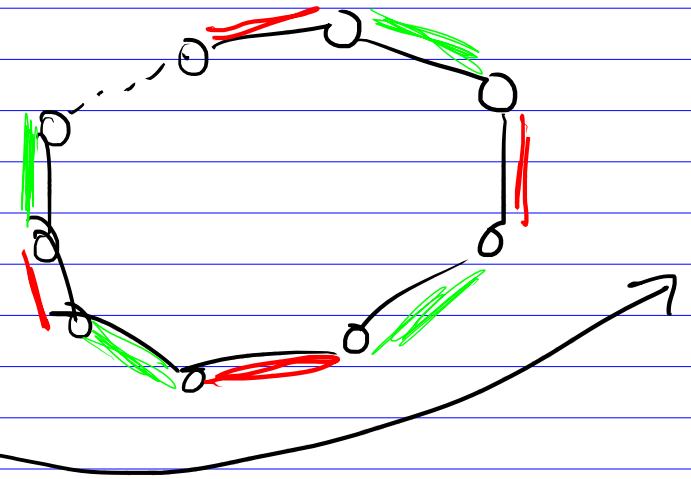
(cost of edge = distance between endpts)

M was min-cost matching on odd-degree vertices

$c(M) \leq c(M^*)$

$c(M^*) \leq \frac{OPT}{2}$

OPT TSP tour

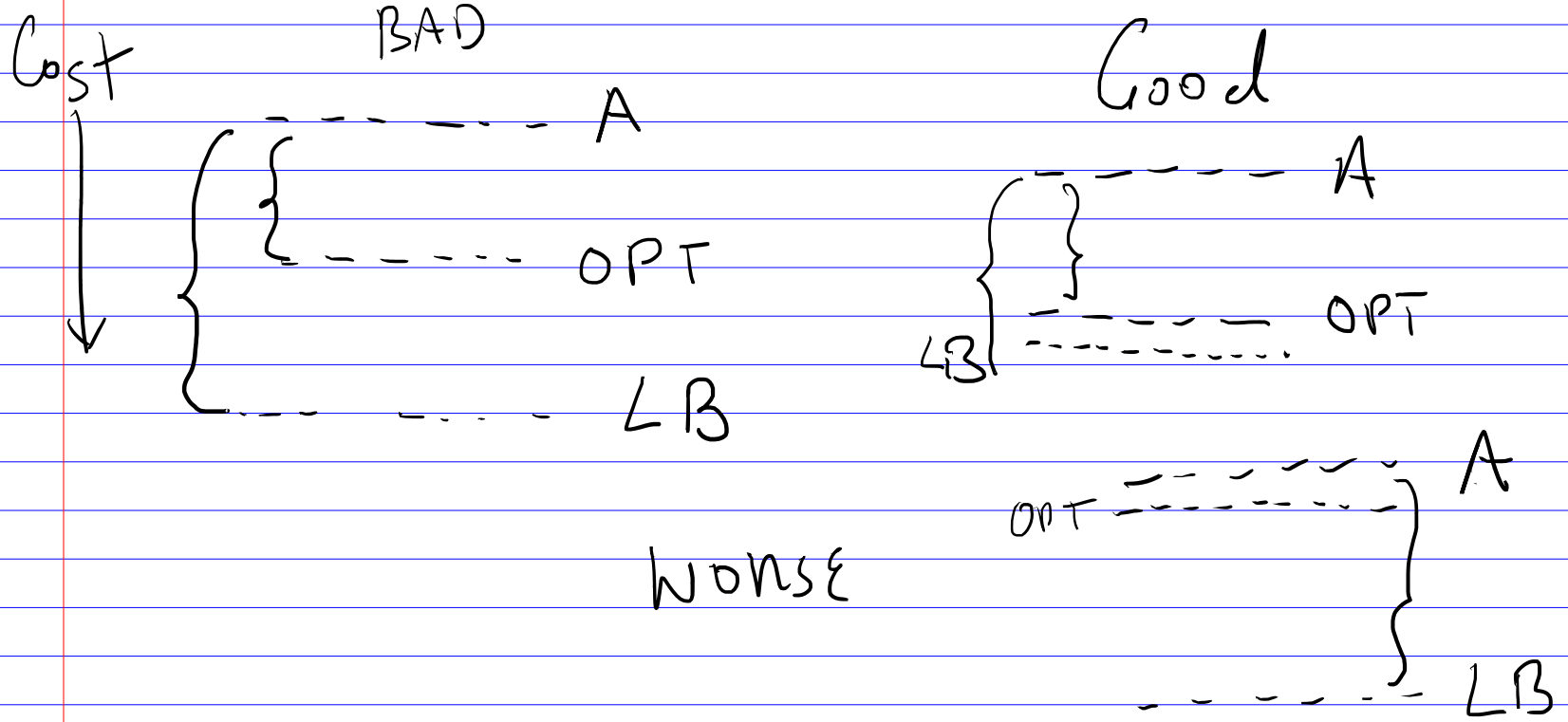


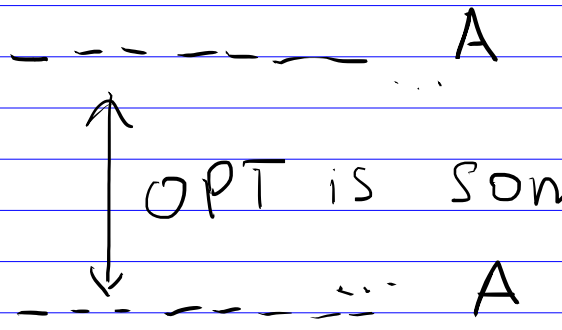
$c(M) + c(M) \leq OPT$

$$c(\text{tour}) = \text{MST} + \text{matching}$$

$$\leq OPT \qquad \leq \frac{OPT}{2}$$

$$c(\text{tour}) \leq \frac{3}{2} OPT \leftarrow \text{Christofides heuristic}$$





- VERTEX COVER - 2
- SET COVER - $\log n$ - approximation
- General TSP - ∞
- metric TSP - 1.5

- Poly time Approximation Schemes. $(1+\epsilon)$

Poly Time Approx Scheme

Want $(1+\epsilon)$ approximation, $\epsilon > 0$

↳ 1.1 approx $(\epsilon = 0.1)$

↳ 1.001 approx $(\epsilon = 0.001)$

Running time: poly(n), but arbitrary fn of $\frac{1}{\epsilon}$

e.g. • $n^2 2^{1/\epsilon}$

• $n^{2^{1/\epsilon 50}}$

• $n^{10} 2^{2^2}$

• $n^{2^{1/\epsilon^2}}$

• n^2

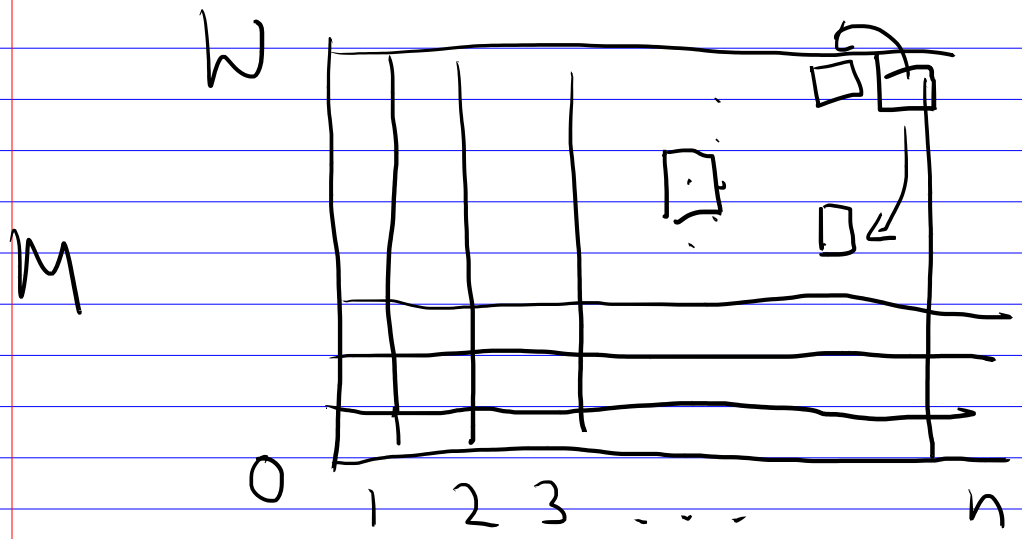
Knapsack

n objects, each with value v_i
and weight w_i

knapsack carries weight W

find subset of items of total weight $\leq W$
of maximum value

Dynamic Program for knapsack



$M(W, i)$

= best value you get with weight $\leq W$ using items $1 \ldots i$

$OPT = M[W, n]$

$n \in OPT$. then $OPT = v_n + OPT(W - w_n, n-1)$

$n \notin OPT$ then $OPT = OPT(W, n-1)$

$$M[W,n] = \max\{v_n + M[W-w_n, n-1], M[W, n-1]\}$$

Dynamic Program running time

$= O(nW)$

Constant time to update each entry of matrix.

W is exponential in $\log W$

but $\log W$ = # bits needed to express W

nW is exponential in input size

Let $v^* = \max v_i$

$\exists$ an algorithm for KNAPSACK running in time $O(n^2 v^*)$

Fix number b

$i \quad v_i$

Replace v_i by $\tilde{v}_i = \left\lceil \frac{v_i}{b} \right\rceil \cdot b$

$v_i \leq \tilde{v}_i \leq v_i + b$

$$\hat{v}_i = \frac{\tilde{v}_i}{b} = \left\lceil \frac{v_i}{b} \right\rceil$$

$$\widetilde{OPT}(\tilde{v}_i) = b \cdot \widehat{OPT}(\hat{v}_i)$$

1. Replace v_i by $\tilde{v}_i$
2. Replace $\tilde{v}_i$ by $\hat{v}_i$
3. Run $O(n^2 v^*)$ algorithm on instance with $\hat{v}_i$
4. Multiply answer by b

Set $b = \frac{\epsilon}{2n} \max v_i = \frac{\epsilon v^*}{2n}$

What is $\hat{v}^*$?

$$\hat{v}_i = \left\lceil \frac{v_i}{b} \right\rceil = \left\lceil \frac{2n v_i}{\epsilon v^*} \right\rceil \leq \left\lceil \frac{2n}{\epsilon} \right\rceil \leq \frac{2n}{\epsilon} + 1$$

Run time $= O(n^2 \hat{v}^*)$

$$= O\left(n^2 \left(\frac{2n}{\epsilon} + 1\right)\right)$$

$$= O\left(\frac{n^3}{\epsilon}\right)$$