Travelling Salesman Problem ([illegible]

Given a graph $G=(V,E)$, a <u>Hamiltonian tour</u> is a cycle that contains all of the nodes. If each edge has a ~~weight~~ distance d_e, the <u>travelling salesman problem</u> is to find the tour with least ~~weight~~ distance.

In polyhedral terms:

Let $x_e = \begin{cases} 1 & \text{if use edge } e \text{ in tour } T. \\ 0 & \text{o/w} \end{cases}$

The TSP is:

$$\min \sum d_e x_e$$

$$T \text{ is a tour.}$$

Christofides heuristic

i) Find a minimum ~~weight~~ distance spanning tree τ on G.

ii) Consider all vertices which have odd degree in τ.

Find a minimum ~~weight~~ distance perfect matching M on these vertices.

iii) Now every vertex has even degree in $M \cup \tau$ (some edges may be used twice)

Find an Eulerian ~~tour~~ walk W on $M \cup \tau$ (ie a ~~tour~~ closed walk on which node appears at least once and each edge appears exactly once)

iv) Short circuit W to get a tour T.

Best known bound of any heuristic for ΔTSP.

Theorem ~~The Chr~~

Provided the edge weights satisfy the triangle inequality $d_{ij} + d_{jk} \geq d_{ik}$, the Christofides heuristic returns a tour which is no more than 50% from optimality.

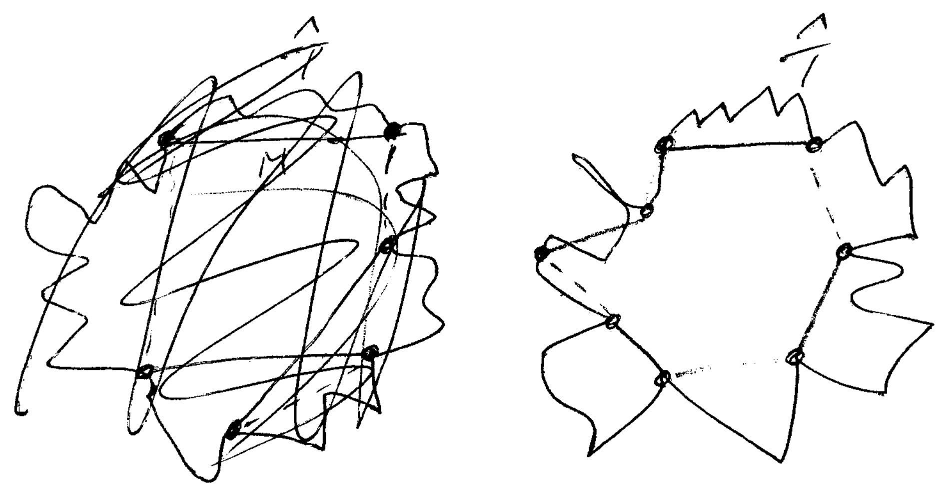
Proof

$$d(T) \leq d(W)$$ since short circuit W to get T (need Δ-ineq here)

$$= d(\tau) + d(M)$$

Let $\hat{T}$ be optimal tour.

$\hat{T}$ is connected, so it contains a spanning tree $\therefore d(\hat{T}) \geq d(\tau)$.



Consider odd nodes. Match them up in order, in two different ways

By Δ-ineq, $d(\hat{T}) \geq d(M_1) + d(M_2)$

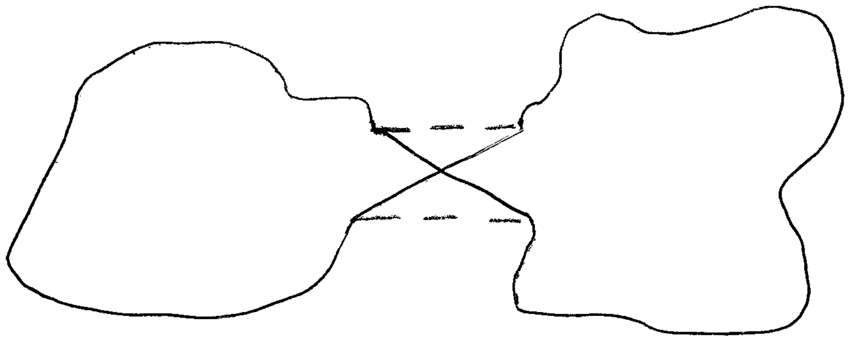
$$\therefore \tfrac{1}{2} d(\hat{T}) \geq \min\{d(M_1), d(M_2)\}$$

$$\geq d(M).$$

$$\therefore d(T) \leq d(\tau) + d(M) \leq \tfrac{3}{2} d(\hat{T})$$

//.

2-change heuristic for improving tours



Have two nonconsecutive edges in a tour, and the corresponding four vertices. Consider other ways of using two edges between these vertices. (Careful not to disconnect graph.)

3-change, k-change:

Generalize 2-change to more edges.

Lin showed 3-change very efficient: if start 3-change from a lot of different tours, ~~very~~ likely to get optimal tour.

How do you know when you are optimal?

How far are we from optimal?

Cutting plane algorithm

Grotschel
Padberg
Rinaldi

(POR, 84)

What constraints describe tours?

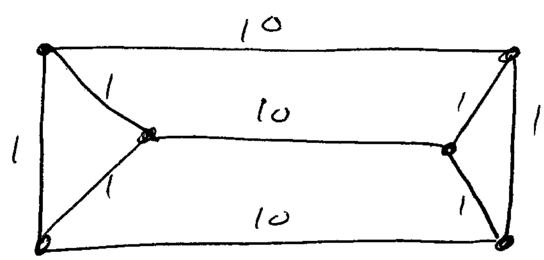
x binary.

One edge entering and one edge leaving each vertex

ie $\sum_{e \in \delta(v)} x_e = 2$.

Are these enough?

No: Eg

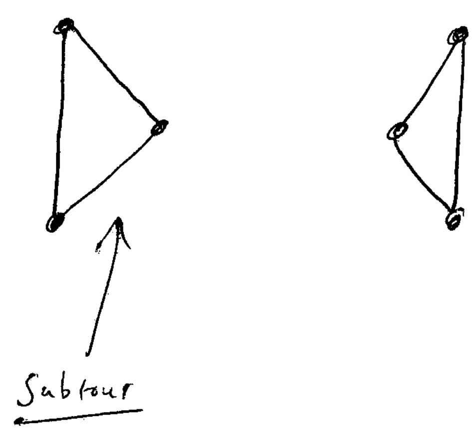


Optimal soln to

$\min \sum d_e x_e$

$\sum_{e \in \delta(v)} x_e = 2$

x binary



Need to eliminate subtours.

Every subset $U \subseteq V$, $U \neq V, \phi$, must have at least two edges connecting it to the rest of the graph in any tour.

Cut constraints:

$$\sum_{e \in \delta(U)} x_e \geq 2 \qquad \forall\, U \subseteq V,\ U \neq V, \phi.$$

$\delta(U)$ = set of edges from U to $V \setminus U$.

Now optimal soln to

$$\min \sum d_e x_e$$

$$\sum_{e \in \delta(v)} x_e = 2$$

$$\sum_{e \in \delta(U)} x_e \geq 2$$

x binary

Large number of these constraints add them as cutting planes.

is optimal tour.

What about LP-relaxation?

$$\min \sum d_e x_e$$

$$\sum_{e \in \delta(v)} x_e = 2$$

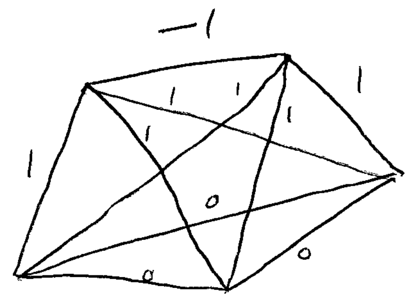
$$\sum_{e \in \delta(U)} x_e \geq 2 \qquad U \subseteq V,\ 3 \leq |U| \leq \frac{|V|+1}{2}.$$

$$0 \leq x_e \leq 1.$$

151

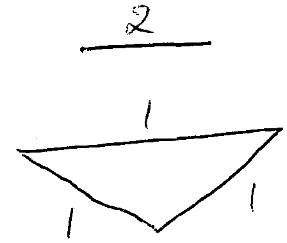
$x_{ij} \geq 0$ is necessary.

Eg:



$x_{ij} \leq 1$ is necessary:

Eg:



Constraints $\sum_{e \in \delta(v)} x_e = 2$ are lin indep:

$$
\begin{array}{l|cccccccccccccc}
 & 12 & \cdots & 1n & 23 & \cdots & 2n & 34 & \cdots & 3n & \cdots & n-2,n-1 & n-2,n & n-1,n \\
\hline
 & 1 & \cdots & 1 & 0 & & & & & & & & & 0 \\
 & 1 & & 0 & 1 & \cdots & 1 & 0 & \cdots & & & & & 0 \\
\text{these rows clearly lin indep} & & 1 & & 1 & & 0 & 1 & \cdots & 1 & 0 & & & 0 \\
 & & & & & & & & & 0 & 1 \cdots 1 & & & \\
 & & & & & & & & & & & 1 \cdots 1 & & \\
 & 0 & \cdots & 1 & 0 & & 0 & & & 0 & & & & 1 \\
\hline
 & 0 & \cdots & 0 & 1 & & 1 & & & 1 & & & 1 & 1
\end{array}
$$

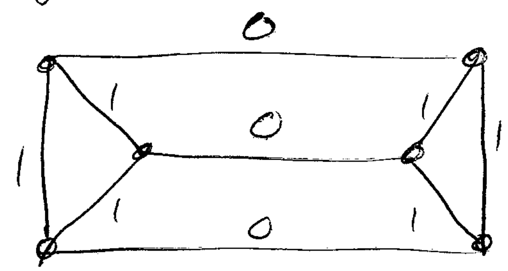
If last row was combo of first $n-1$, would need to add first $n-1$. But this would give 2's, instead of zeroes, in all ij positions with i, j both $\neq n$.

$\sum_{e \in \delta(U)} x_e \geq 2$ facet defining, for $3 \leq |U| \leq |V|/2$:

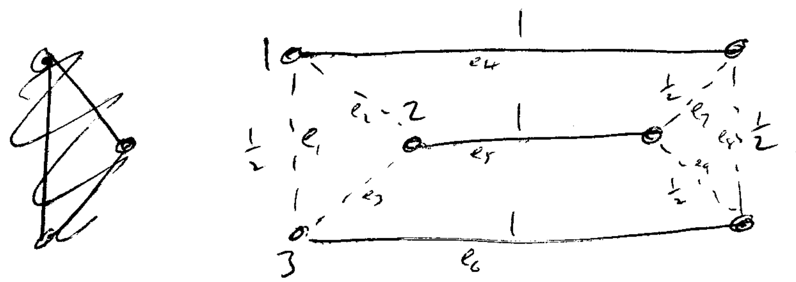
If we drop one of these we can satisfy all the remaining ones but not this one, by taking one subtour through U and one through $V \setminus U$.

Solution may not be integral:

d:



Opt soln to LP relaxation:



Too much ~~distance~~ ~~weight~~ x coming out of $\{1, 2, 3\}$.

C-G cut:

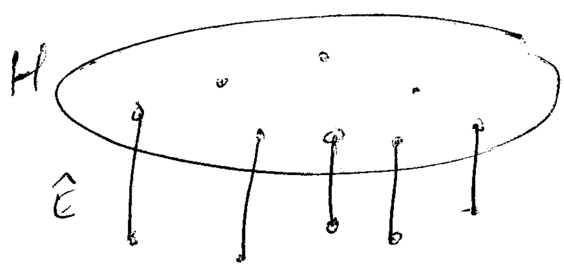
Combine adjacency for 1, 2, 3 with weight $\frac{1}{2}$ $\quad x_1 + x_2 + x_4 = 2$, $x_1 + x_3 + x_6 = 2$, $x_2 + x_3 + x_5 = 2$

Combine upper bounds on e_4, e_5, e_6 with weight $\frac{1}{2}$: $x_4 \le 1$, $x_5 \le 1$, $x_6 \le 1$

$$\cancel{e_1 + e_2}$$
$$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 \le \lfloor 4\tfrac{1}{2} \rfloor$$
$$= 4.$$

This is an example of a 2-matching inequality:

(These suffice for $|V| \le 6$. More ineqs are needed for larger $|V|$.)

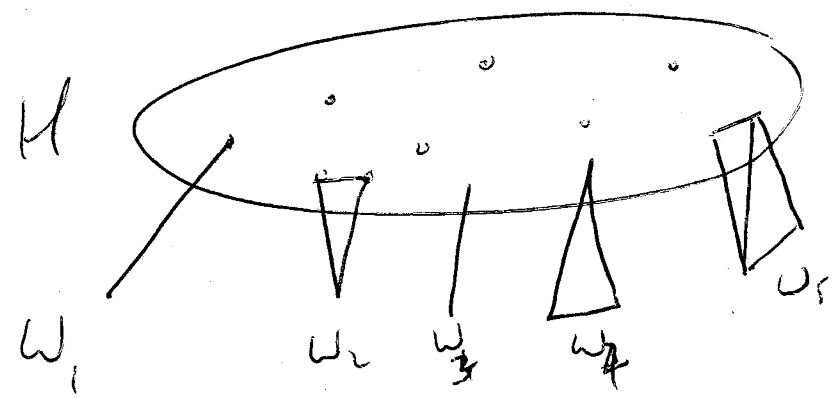


$$\sum_{e \in E(H)} x_e + \sum_{e \in \hat{E}} x_e \le |H| + \left\lfloor \frac{|\hat{E}|}{2} \right\rfloor.$$

(Eg, if use $|H| - 1$ edges in H, can only use 2 edges in $\hat{E} \subseteq \delta(H)$.)

Argue why this is valid.

Can generalize to comb inequalities:



$|H \cap W_i| \geq 1$

$|W_i \setminus H| \geq 1$

$W_i \cap W_j = \phi$

Number of teeth W_i is odd.

Valid ineq. $\displaystyle\sum_{e \in E(H)} x_e + \sum_{i=1}^{k} \sum_{e \in E(W_i)} x_e \leq |H| + \sum_{i=1}^{k} (|W_i| - 1) - \frac{k+1}{2}.$

Facet defining for a complete graph.

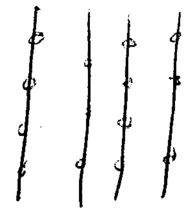
Can be generalized further – see Nemhauser & Wolsey for details.

Well-solvable cases of the TSP (Burkard et al, SIAM Review 40 (1998), 496-546.)

Unfortunately (?!), even the planar TSP is NP-complete.

① k-line TSP

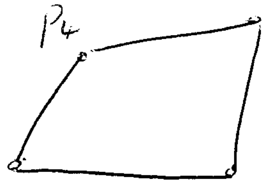
Used in, eg, VLSI design:
Cities are on k parallel lines:



Can be solved in $O(n^k)$ operations,
using a dynamic programming approach

Result uses the fact that the optimal planar TSP tour does not cross itself, because of:

For a rectangle, have:



$$d(p_1,p_2) + d(p_3,p_4) \leq d(p_1,p_3) + d(p_2,p_4)$$

② A cost matrix is a MONGE MATRIX if

$$d_{ij} + d_{rs} \leq d_{ir} + d_{js} \quad \text{if } 1 \leq i < r \leq n, \; 1 \leq j < s \leq n$$

For such a cost matrix, there exists an optimal tour that is PYRAMIDAL,

ie $1 \to i_1 \to i_2 \to \dots i_r \to n \to j_1 \to \dots \to j_s \to 1$

with $1 < i_1 < i_2 < \dots < i_r < n$ and $n > j_1 > \dots > j_s > 1$.

The best pyramidal tour can be found in polynomial time $(O(n^2))$, even though there are exponentially many such tours

Eg: points on boundary of a convex set: you need the boundary for the optimal tour

Can make it slightly nonconvex

Another pyramidal example

Not dense

Distances sum(?) ...

Vehicle Routing Problem

Eg: School bus scheduling, ~~UPS deliveries, etc.~~

Have several salesmen visiting the cities.
Each city is visited by exactly one of the salesmen.

So: Need to (1) ~~Choose which~~ Assign customers to salesmen
(2) Find best route through ~~their~~ its customers for each salesman. (TSP).

Best algorithms will do (1) and (2) simultaneously.

In practice, far harder than TSP, with duality gaps of 20% common, even with only 100 customers.
(For TSP, ~~prob~~ usually easy to get within 2%, even for problems with thousands of cities.)

Heuristic: Optimal partitioning: (~~Good~~ Applicable if all vehicles have same capacity, all customers have same demand.)
Find one tour through customers and depot.
Optimally partition the tour.

Can be twice as bad as true optimal value.

Complications: Customers with varying demands (package sizes)
Vehicles of varying sizes
Time windows for deliveries
⋮