Heuristics for TSP (Best for Dense graphs!)

Nearest Neighbor

Avg Cost: 1.26 X.

If triangle inequality: $c_{u,v} + c_{v,w} \geq c_{u,w}$

holds, then approx ratio is $\leq \underbrace{\frac{1}{2}\lceil \log_2 n \rceil + \frac{1}{2}}_{\text{sharpish}}$ time opt.

Insertion Methods:

Start w/ a small cycle, "grow" by inserting a point between two vertices.

Options:

Farthest Insertion: Start between two nodes far apart.
Then, for each node not in the cycle, compute cheapest insertion.
Insert the vertex w/ worst cheap insertion!
(Idea: This gets rough shape figured out first.) $\leq 1.16X$

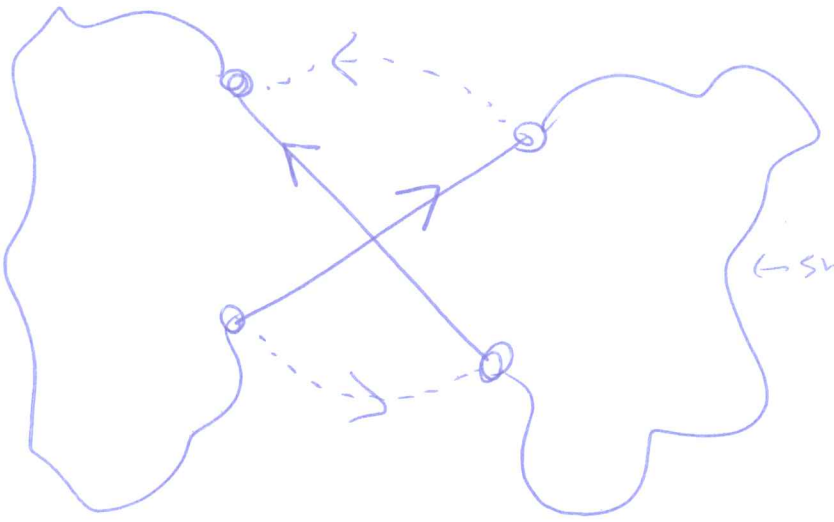
$\leq \lceil \log_2 n \rceil + 1$ X
All insertions

Nearest Insertion: Closest to any node in cycle. $\leq 2X$

Cheapest Insertion: Closest to a pair of adjacent in cycle $\leq 2X$

No insertion method worse than 4x known!

k-switches



A k-switch on C removes k edges, and inserts k new edges, such that what remains is a spanning cycle.

Prop: An optimal TSP tour in the plane has no crossings.

2.3. Optimization and Trees

"Best spanning tree"?

Def: weighted graph (non-negative)

length of a path: sum of edge weights.

minimum-weight spanning tree. (MST)

Kruskal's Alg:

Input: Weighted connected graph

~~Output~~:

Idea: Maintain acyclic subgraph H, added edges of low weight without creating a cycle.

Order $E(G) = \{e_1, \dots, e_m\}$ s.t. $w(e_i) \le w(e_{i+1})$.

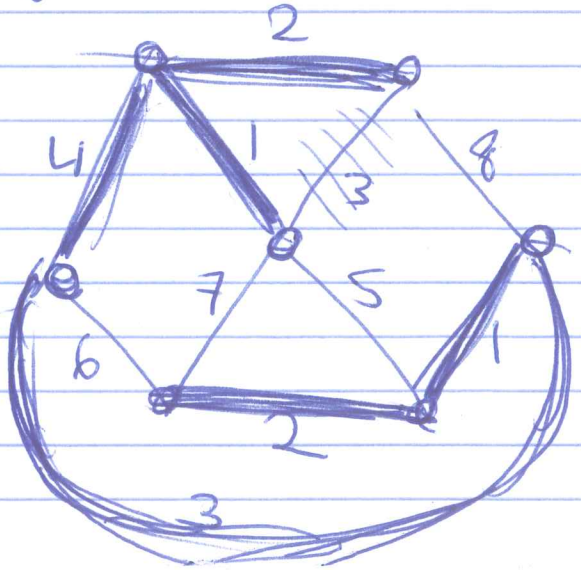
Initialization: $E(H) = \emptyset$.

Iteration: If the next least-weight edge connects two components of H, add it to H.

Otherwise, discard it.

greedy algorithm

Ex:



Thm (Kruskal 1956): An a connected weighted graph G, Kruskal's Algorithm constructs a min-weight spanning tree.

Pf: ① The algorithm forms a tree.

Due to component-joining constraint, we'll never make a cycle.
By connectivity, always some edge to connect.

② Let T be the tree and T^* an MST.
Extremal Choice: $\max |E(T) \setminus E(T^*)|$

If $T = T^*$, we are done.

If $T \neq T^*$, let e be the first edge chosen for T that is not in T^*
($E(T) \setminus E(T^*) \neq \emptyset$, pick left-weight...)

$T^* + e$ has a unique cycle C since T has no cycles, there exists $e' \in E(C) \setminus E(T)$.

Consider spanning tree $T^* + e - e'$.

Since T^* (all edges of T chosen before e, both e & e' Construction and
are available when algorithm chooses e. So $w(e) \leq w(e')$

Therefore $w(T^* + e - e') \leq w(T^*)$ and equality by MST.

But now $|E(T) \cap E(T^* + e - e')| > |E(T) \cap E(T^*)|$ ↯

$\therefore T$ is MST. □

Christofides' Heuristic

T a min-wt. spanning tree.

M a min-wt p.m. between odd verts of T.

$J = E(T) \cup M$. → even and connected!

Iteratively remove edges from vertices of deg > 2, while remaining connected.:  (Δ-ineq!)

Thm: The cycle C given by Christofides' Heuristic is at most $\frac{3}{2} \times$ the opt (given nonneg edges & Δ-ineq.)

Pf: Let H^* be an optimal tour.

Removing any edge from H^* reveals a spanning tree, so the cost $c(T)$ of a MST T is at most $c(H^*)$. We can also form a circuit C on the odd nodes W of T by going them in the order of H^*.

Observe $|W|$ is even, and the edge set of C partitions into two perfect matchings.

Thus, $c(M) \leq \frac{1}{2} c(C) \leq \frac{1}{2} c(H^*)$.

We therefore have $c(T \cup M) \leq \frac{3}{2} c(H^*)$, and by the triangle inequality, we only improve the tour by reducing degrees.

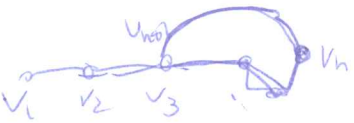
□

More Local Search: Lin-Kernighan

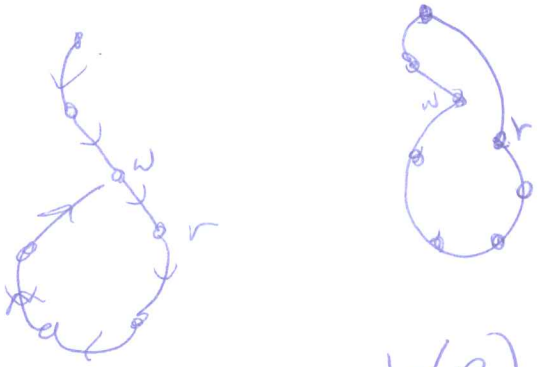
Idea: Perform a special kind of k-opt.

- Value of k can vary
- If an improvement is found, then we should recc. use it right away.

Def: A S-path is a walk of n edges and $n+1$ nodes where all nodes are distinct except the last:



If P is a S-path that is not a tour, let w be the repeated node and r the one following the first instance.
Remove wr and add r to first node:



P $\quad$ $+(P)$

This creates a specific tour $T(P)$

Lower Bounds

Instead of using Branch-and-Bound, we must construct "proofs" that optimal TSPs have certain costs.

Consider an MST T. Recall $c(T) \leq c(C)$ for any spanning cycle C. In fact, T has $n-1$ edges, so it is a little bit weaker than that!

In C, every vertex has degree 2, so what if we first choose "best" two edges for a vertex v, then find an MST for $T-v$?

1-tree Lower Bound:

Select $v \in V(G)$, let e_1, e_2 be edges in $\delta(v)$ with minimum cost, and let T be an MST of $G-v$. Then $c(T) + c(e_1) + c(e_2) = c(T^*)$ (where $T^* = T + v + e_1 + e_2$) is a lower bound on any spanning cycle.

Held-Kap Lower Bound

For each $v \in V$, let y_v be a real number.

$\bar{c}_{uv} := c_{uv} - y_u - y_v$, ~~and find a MST T of G w.r.t. costs $\bar{c}$~~

~~$\bar{c}(T)$ is a bound~~ let C be the 1-tree bound for G over $\bar{c}$.

Then $2\sum_{v \in V(G)} y_v + C$ is a lower bound for TSP in G over c.

How to find a good Held-Karp Lower Bound.

ITERATIVE SCHEME:

Let $\bar{c}_{uv} = c_{uv} - y_u - y_v$ be adjusted costs.

Compute the max 1-Tree T.

$d_T(v)$ = degree of v in T.

Want: $d_T(v) = 2$ everywhere. (If so, then DONE! LB is a tour!)

So: Increase costs on high degree vertices
& Decrease costs on low degree vertices ($d_T(v) = 1$)

So: $$y_v \leftarrow y_v + t(2 - d_T(v))$$

where t is the step size.

Iteratively improve this, while making a choice of t that is "decreasing" and helps approach a target.

Initial Variables:

$y_v \leftarrow 0 \quad \forall v \in V$

$H^* \leftarrow -\infty$ (best LB so far)

TSMALL $\leftarrow 0.001$

$\alpha \leftarrow 2$

$\beta \leftarrow 0.5$ (decay factor)

$U \leftarrow$ Best UB so far

MAXCHANGES $\leftarrow$ 100?

NUMITERATIONS $\leftarrow$ ITERATIONFACTOR $\cdot |V|$

~~ITERATIONFACTOR~~ $\leftarrow$ 0.015?

ALL OF THESE ARE ADJUSTABLE

For $\alpha \in \{1, \ldots, \text{MAXCHANGE}\}$

  For $k \in \{1, \ldots, \text{NUMITERATIONS}\}$

    Let T be optimum 1-tree w/r/t $\bar{c}$.

    $H \leftarrow \bar{c}(T)$

    If $H > H^*$ then:

      $H^* \leftarrow H$

    If T is a tour then:

      STOP

    $t^{(k)} \leftarrow \alpha \cdot (U - H) / \sum_{v \in V} [(2 - d_T(v))^2]$

    If $t^{(k)} < \text{TSMALL}$ then:

      STOP

    For $v \in V$:

      $y_v \leftarrow y_v + t^{(k)}(2 - d_T(v))$

  $\alpha \leftarrow \beta \cdot \alpha$

Linear Programming Bound:

$$\min \quad \sum c_{u,v} x_{u,v}$$

$$\text{s.t.} \quad \sum_{u \in V} x_{u,v} = 2 \qquad \forall v \in V$$

$$\left(\sum_{\substack{u \in S \\ v \notin S}} x_{u,v} \geq 2 \qquad \forall \emptyset \subsetneq S \subsetneq V \right) \leftarrow \text{Exponential \# of constraints!!!}$$

$$0 \leq x_{u,v} \leq 1 \qquad \forall u, v \in V$$

(Dantzig, Fulkerson, & Johnson Relaxation)

Thm: The optimal value of the DFL Relaxation is equal to the optimal Held-Karp Lower Bound!

Pf: Fix $v_1 \in V$ and only consider $\emptyset \subsetneq S \subsetneq V$ where $v_1 \notin S$, since constraints for S & $V-S$ are identical.

Since $x(\delta(v)) = 2 \quad \forall v \in V$, we can add constraints:

$$\sum_{u,v \in S} x_{u,v} = x(\gamma(S)) \leq |S| - 1$$

and the equation

$$\sum_{u,v \neq v_1} x_{u,v} = x(\gamma(V - \{v_1\})) = |V| - 2$$

Now, if we remove the equation $x(\delta(v)) = 2$, we have:

P_1:

$$\min \ c^T x$$

$$\text{s.t.} \quad x(\gamma(S)) \leq |S| - 1 \qquad \emptyset \neq S \neq V, \ S \not\ni v_1$$

$$x(\gamma(V - \{v_1\})) = |V| - 2$$

$$0 \leq x_e \leq 1 \qquad \forall e$$

} This encodes MST for $V - v_1$

Consider the dual of P_0 w/ $x(\delta(v)) = 2$ constraints.

y_v is dual variable for $x(\delta(v)) = 2$ constraint.

Let y_v^* be optimal for the dual.

Then since $x(\delta(v)) = 2$, we have

$$\sum (c_{u,v} - y_u^* - y_v^*) x_{uv} = \sum c_{u,v} x_{uv} - 2 \sum y_v^*$$