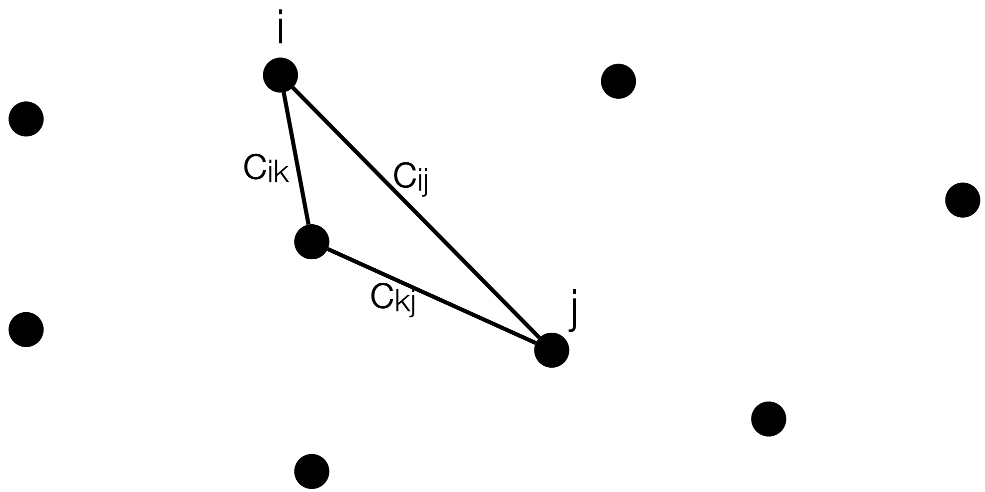
Traveling salesman problem

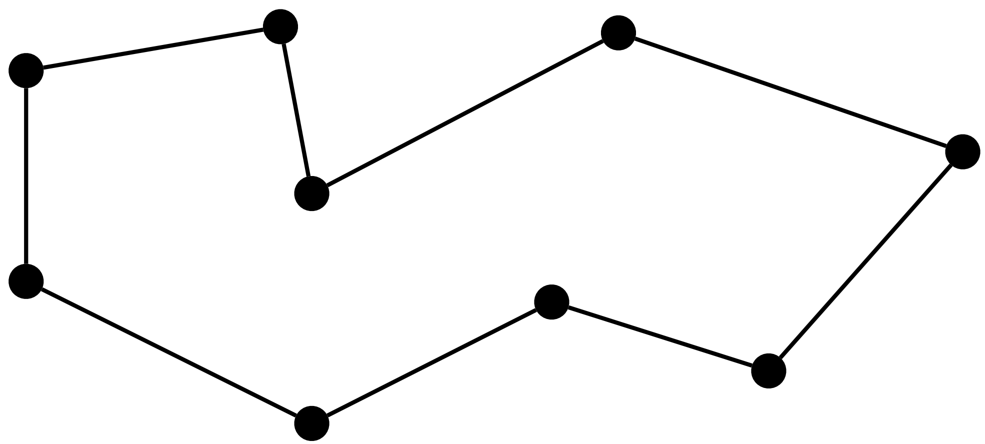
Inge Li Gørtz

Traveling Salesman Problem (TSP)



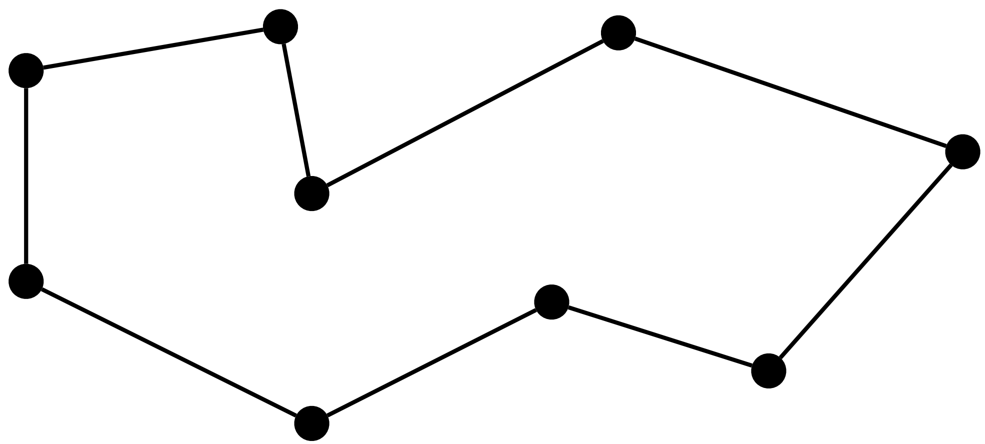
- Set of cities {1,…,n}
- $c_{ij} \geq 0$: cost of traveling from i to j.
- c_{ij} a metric:
  - $c_{ii} = 0$
  - $c_{ij} = c_{ji}$
  - $c_{ij} \leq c_{ik} + c_{kj}$ (triangle inequality)
- Goal: Find a *tour of minimum cost visiting every city exactly once*.

Traveling Salesman Problem (TSP)



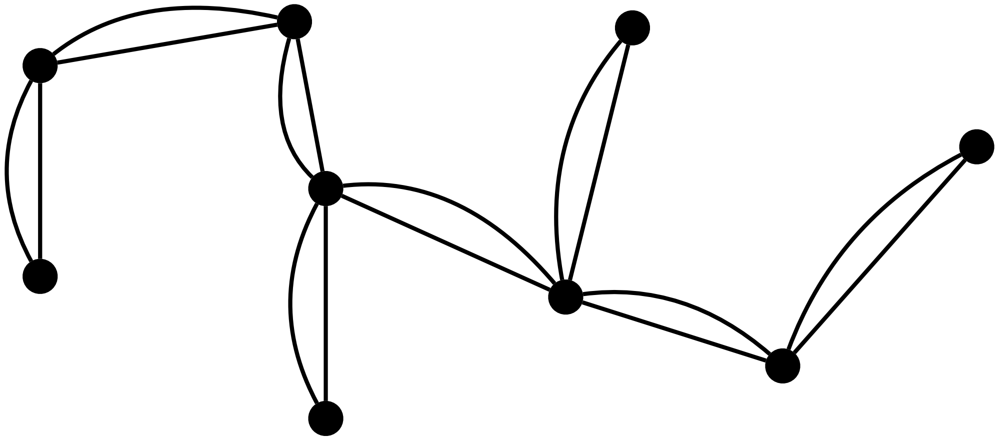
- Set of cities $\{1,\ldots,n\}$
- $c_{ij} \geq 0$: cost of traveling from i to j.
- c_{ij} a metric:
  - $c_{ii} = 0$
  - $c_{ij} = c_{ji}$
  - $c_{ij} \leq c_{ik} + c_{kj}$
- Goal: Find a *tour of minimum cost visiting every city exactly once*.

Double tree algorithm



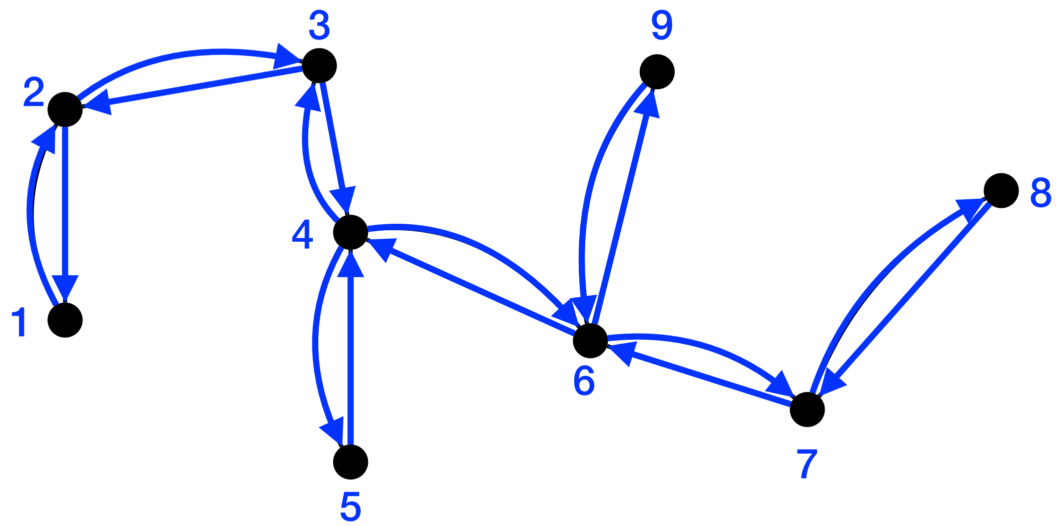
- MST is a lower bound on TSP.
  - Deleting an edge e from OPT gives a spanning tree.
  - OPT ≥ OPT - c_e ≥ MST.

Double tree algorithm



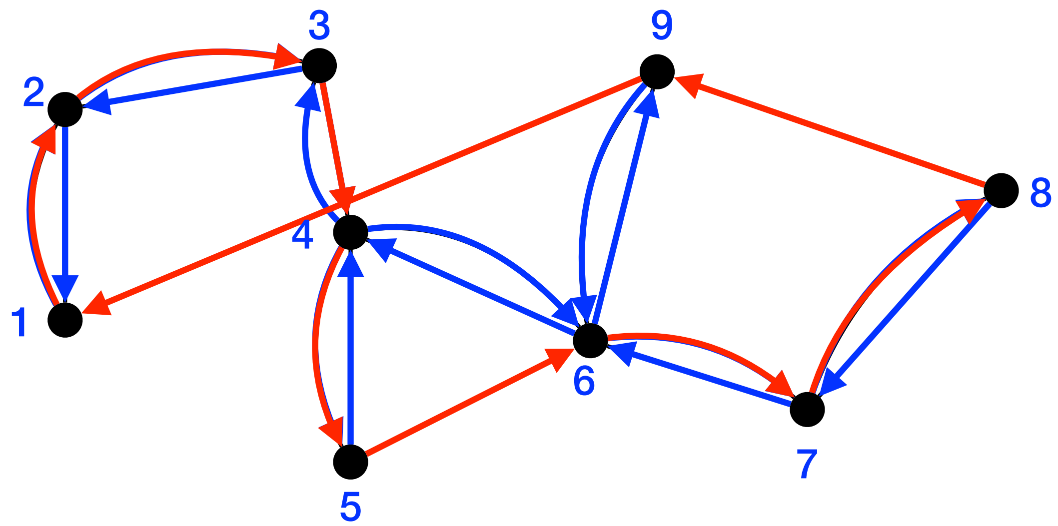
- Double tree algorithm
    - Compute MST T.
    - Double edges of T
    - Construct Euler tour **τ** (a tour visiting every edge exactly once).

Double tree algorithm



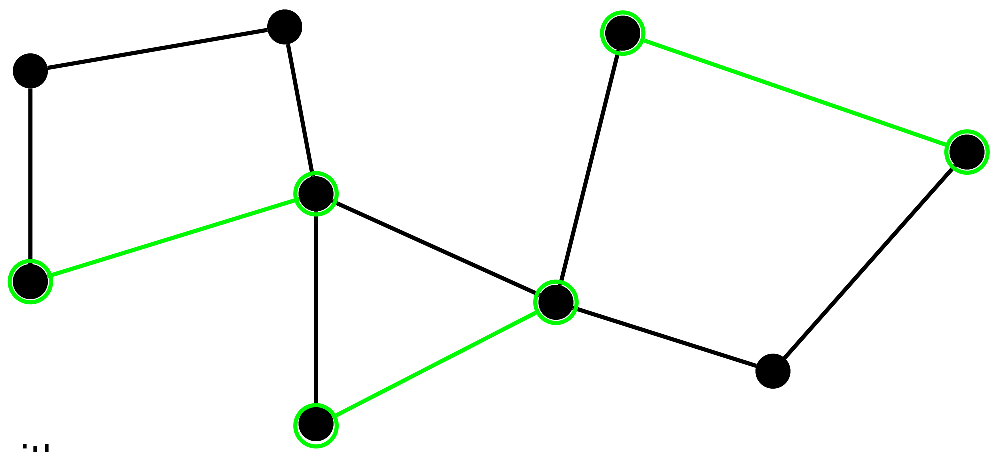
- Double tree algorithm
  - Compute MST T.
  - Double edges of T
  - Construct Euler tour **τ** (a tour visiting every edge exactly once).

Double tree algorithm



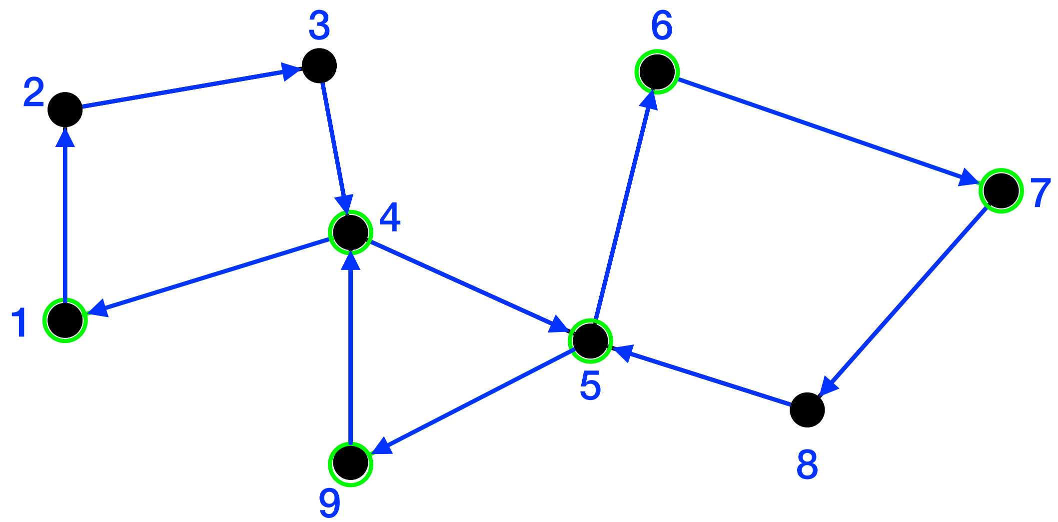
- Double tree algorithm
  - Compute MST T.
  - Double edges of T
  - Construct Euler tour **τ** (a tour visiting every edge exactly once).
  - Shortcut **τ** such that each vertex only visited once (**τ'**)
- length(**τ'**) ≤ length(**τ**) = 2 cost(T) ≤ 2 OPT.
- The double tree algorithm is a 2-approximation algorithm for TSP.

Christofides' algorithm



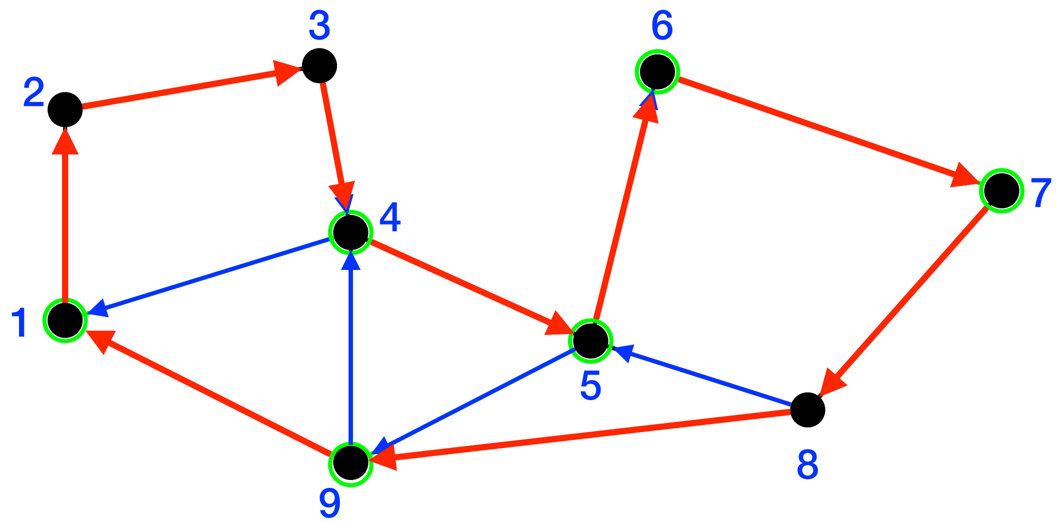
- Christofides' algorithm
    - Compute MST T.
    - No need to double all edges:
        - Enough to turn it into an Eulerian graph: *A graph Eulerian if there is a traversal of all edges visiting every edge exactly once.*
            - G Eulerian iff G connected and all nodes have even degree.
        - Consider set O of all odd degree vertices in T.
        - Find minimum cost perfect matching M on O.
            - Matching: no edges share an endpoint.
            - Perfect: all vertices matched.
            - Perfect matching on O exists: Number of odd vertices in a graph is even.
        - T + M is Eulerian (all vertices have even degree).

Christofides' algorithm



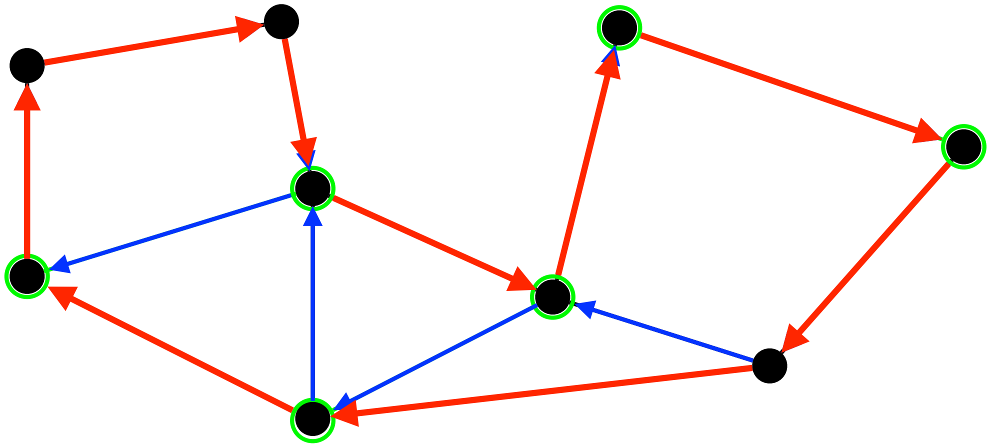
- Christofides' algorithm
  - Compute MST T.
  - O = {odd degree vertices in T}.
  - Compute minimum cost perfect matching M on O.
  - Construct Euler tour **τ**
  - Shortcut such that each vertex only visited once (**τ'**)

Christofides' algorithm



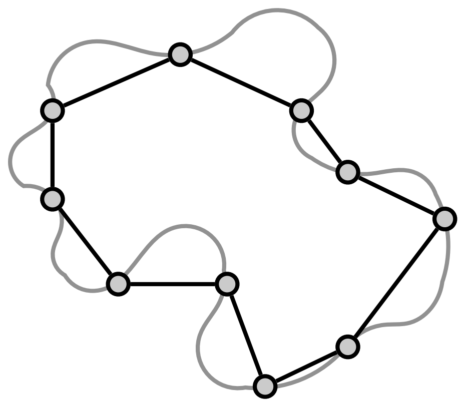
- Christofides' algorithm
  - Compute MST T.
  - O = {odd degree vertices in T}.
  - Compute minimum cost perfect matching M on O.
  - Construct Euler tour τ
  - Shortcut such that each vertex only visited once (τ')

Christofides' algorithm



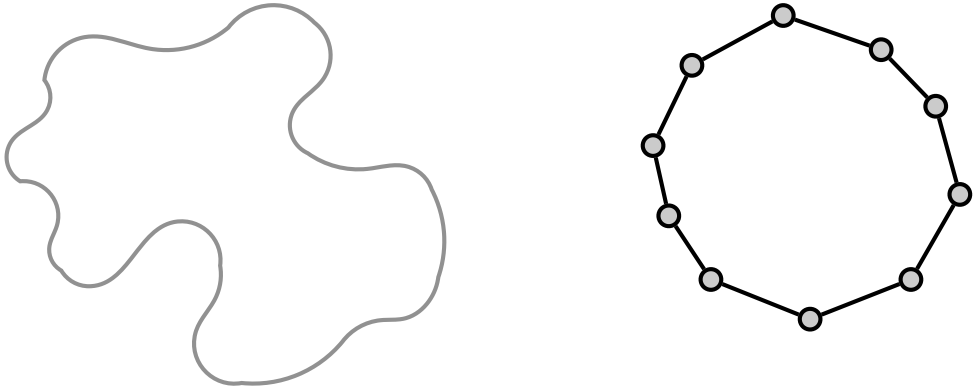
- Christofides' algorithm
    - Compute MST T.
    - O = {odd degree vertices in T}.
    - Compute minimum cost perfect matching **M** on O.
    - Construct Euler tour **τ**
    - Shortcut such that each vertex only visited once (**τ'**)
- length(**τ'**) ≤ length(**τ**) = cost(T) + cost(**M**) ≤ OPT + cost(**M**).

Analysis of Christofides' algorithm



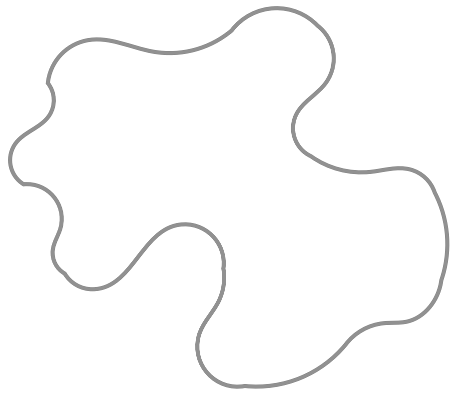
- cost(M) ≤ OPT/2.
    - OPT_O = OPT restricted to O.
    - OPT_O ≤ OPT.

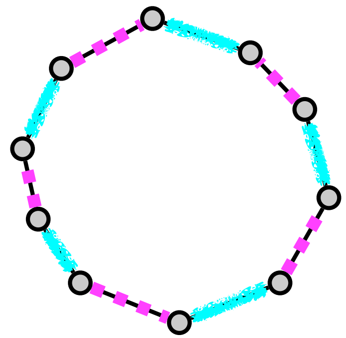
Analysis of Christofides' algorithm



- cost(M) ≤ OPT/2.
  - OPT_O = OPT restricted to O.
  - $OPT_O \leq$ OPT.

Analysis of Christofides' algorithm





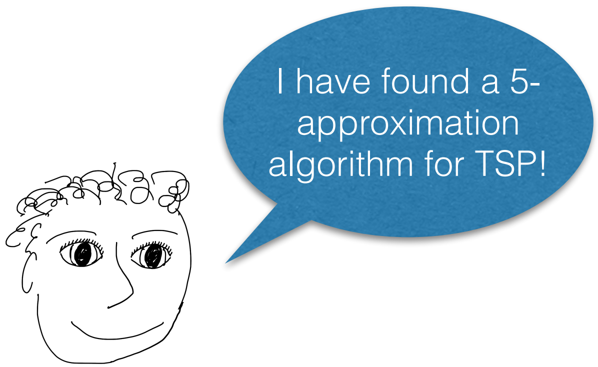
- cost(M) ≤ OPT/2:
  - OPT_O = OPT restricted to O.
  - OPT_O ≤ OPT.
  - can partition OPT_O into two perfect matchings O_1 and O_2.
  - cost(M) ≤ min(cost(O_1), cost(O_2)) ≤ OPT/2.
- length(τ') ≤ length(τ) = cost(T) + cost(M) ≤ OPT + OPT/2 = 3/2 OPT.
- Christofides' algorithm is a 3/2-approximation algorithm for TSP.

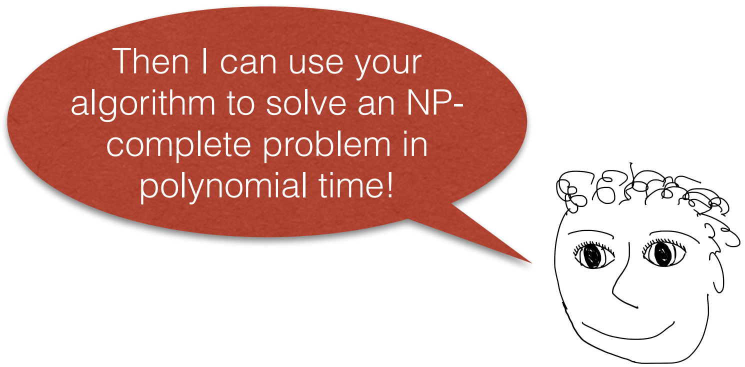
Hardness of Approximation

Inge Li Gørtz

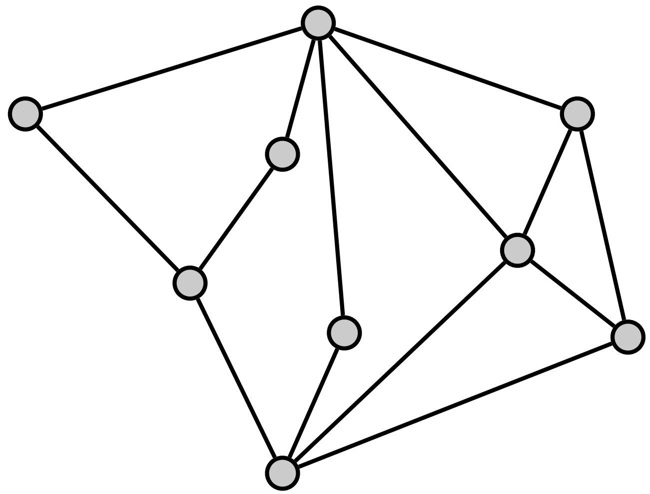
TSP: Inapproximability

- There is no α-approximation algorithm for the non-metric TSP for unless P=NP.

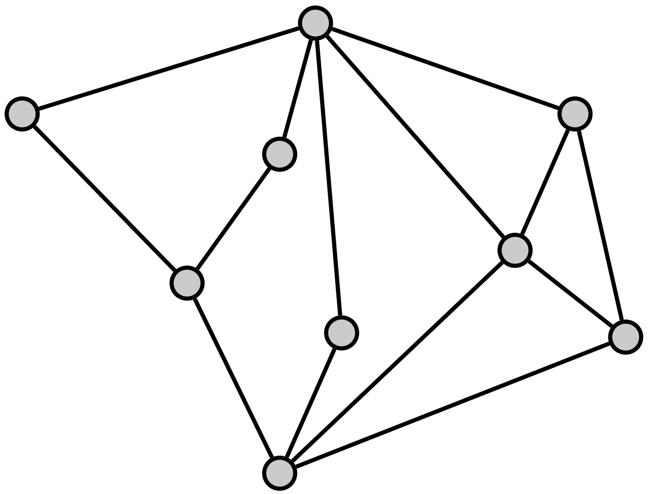




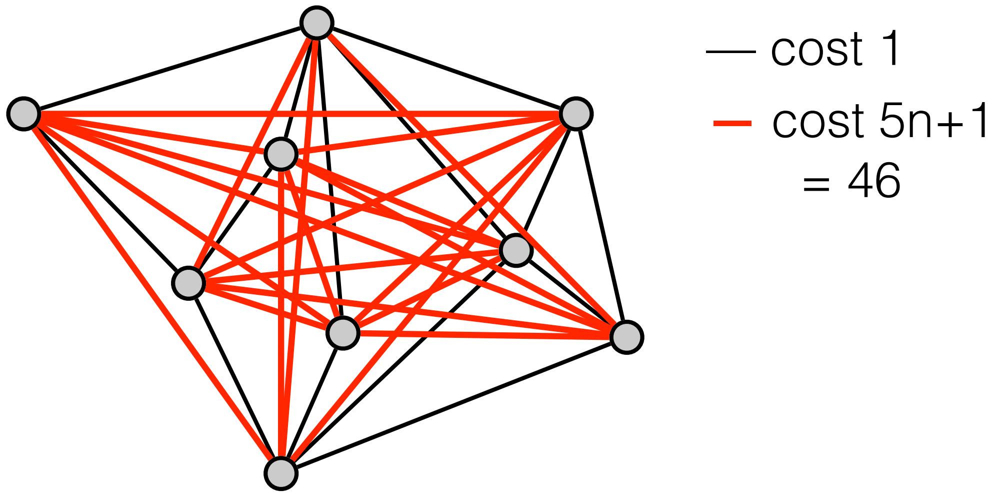
- *Hamiltonian cycle.* Given G=(V,E). Is there a cycle visiting every vertex exactly once?



TSP: Inapproximability



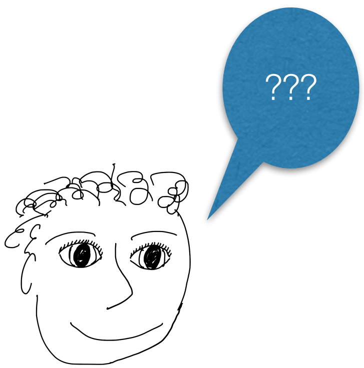
G

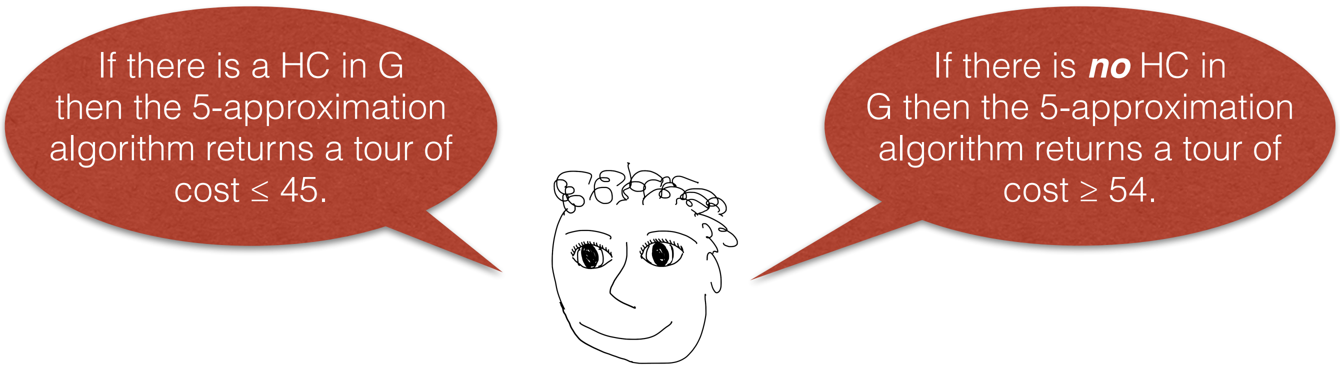


G'

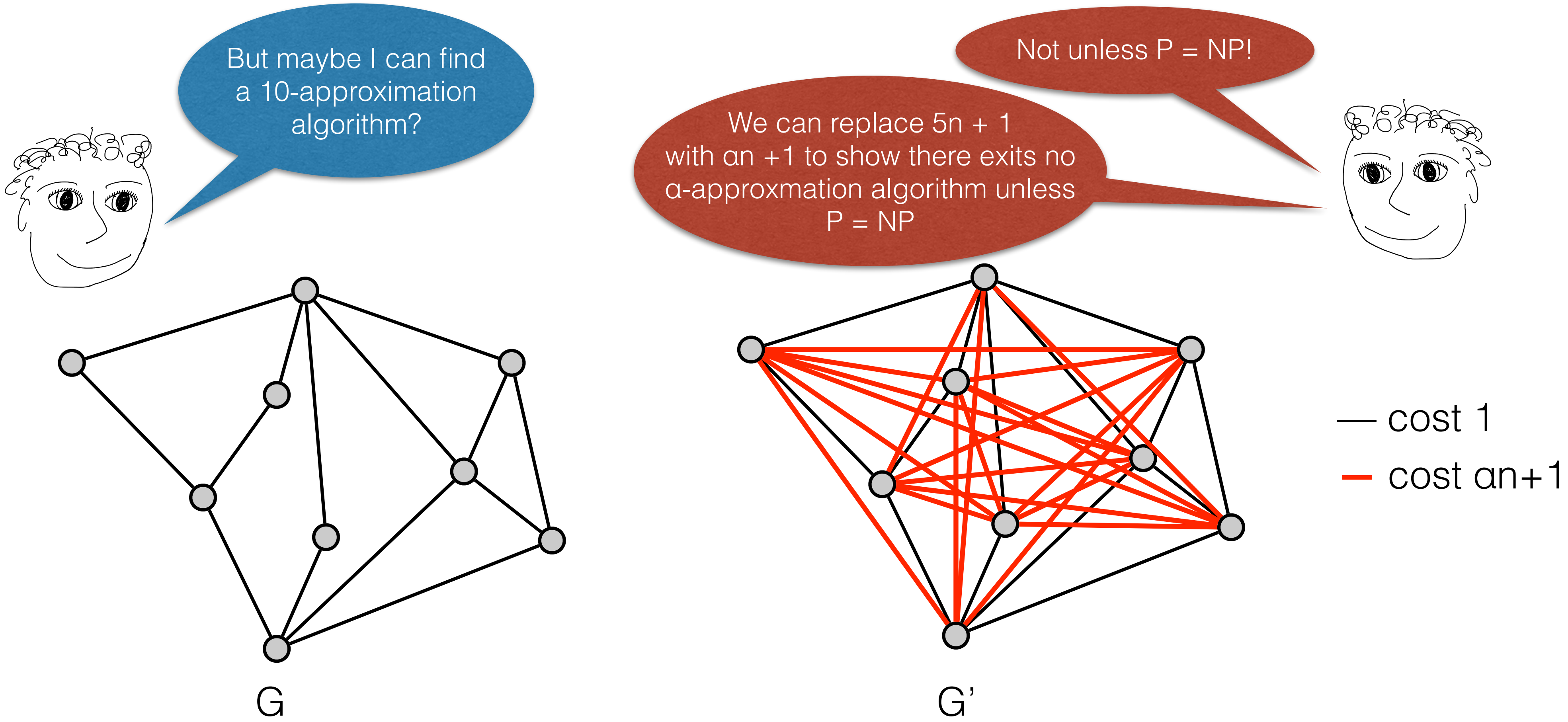
- *G has a Hamiltonian cycle* ⇔ *optimal cost of TSP in G' is $n = 9$.*
- *G has no Hamiltonian cycle* ⇔ *optimal cost of TSP in G' is at least $n - 1 + 5n + 1$*

$= 6n = 8 + 46 = 54$





TSP: Inapproximability



- *G has a Hamiltonian cycle* ⇔ *optimal cost of TSP in G' is n.*
- *G has no Hamiltonian cycle* ⇔ *optimal cost of TSP in G'* $\geq n - 1 + (\alpha n + 1)$

$= (\alpha+1)n$

k-center: Inapproximability

- There is no α-approximation algorithm for the k-center problem for α < 2 unless P=NP.
- **Proof.** Reduction from dominating set.
- *Dominating set.* Given G=(V,E) and k. Is there a (dominating) set S ⊆ V of size k, such that each vertex is either in S or adjacent to a vertex in S?
- Given instance of the dominating set problem construct instance of k-center problem:
  - Complete graph G' on V.
  - All edges from E has weight 1, all new edges have weight 2.
  - Radius in k-center instance 1 or 2.
  - G has an dominating set of size k <=> opt solution to the k-center problem has radius 1.
  - Use α-approximation algorithm A:
    - opt = 1 => A returns solution with radius at most α < 2.
    - opt = 2 => A returns solution with radius at least 2.
    - Can use A to distinguish between the 2 cases.