1. Consider the following algorithm for the MAX-CUT problem.

   A. Start with an arbitrary cut (S,T) of V.

   B. So long as possible, repeat:

      (i) If there exists $u \in S$ such that the cut $(S-u, T+u)$ has more cross edges than (S,T), delete u from S, and add u to T.
      (ii) If there exists $v \in T$ such that the cut $(S+v, T-v)$ has more cross edges than (S,T), add v to S, and delete v from T.

   C. Return (S,T).

(a) Prove that this algorithm terminates in polynomial time.

Each vertex switch increases the cut size by at least one. So the number of vertex switches is bounded by the number of edges in G.

(b) Prove that the approximation ratio of this algorithm is 1/2.

Take any vertex u of G. Consider the following quantities associated with u.

$c_u$ = the number of cross edges incident upon u
$b_u$ = the number of non-cross edges incident upon u
$d_u$ = the degree of u

We have $d_u = b_u + c_u$.

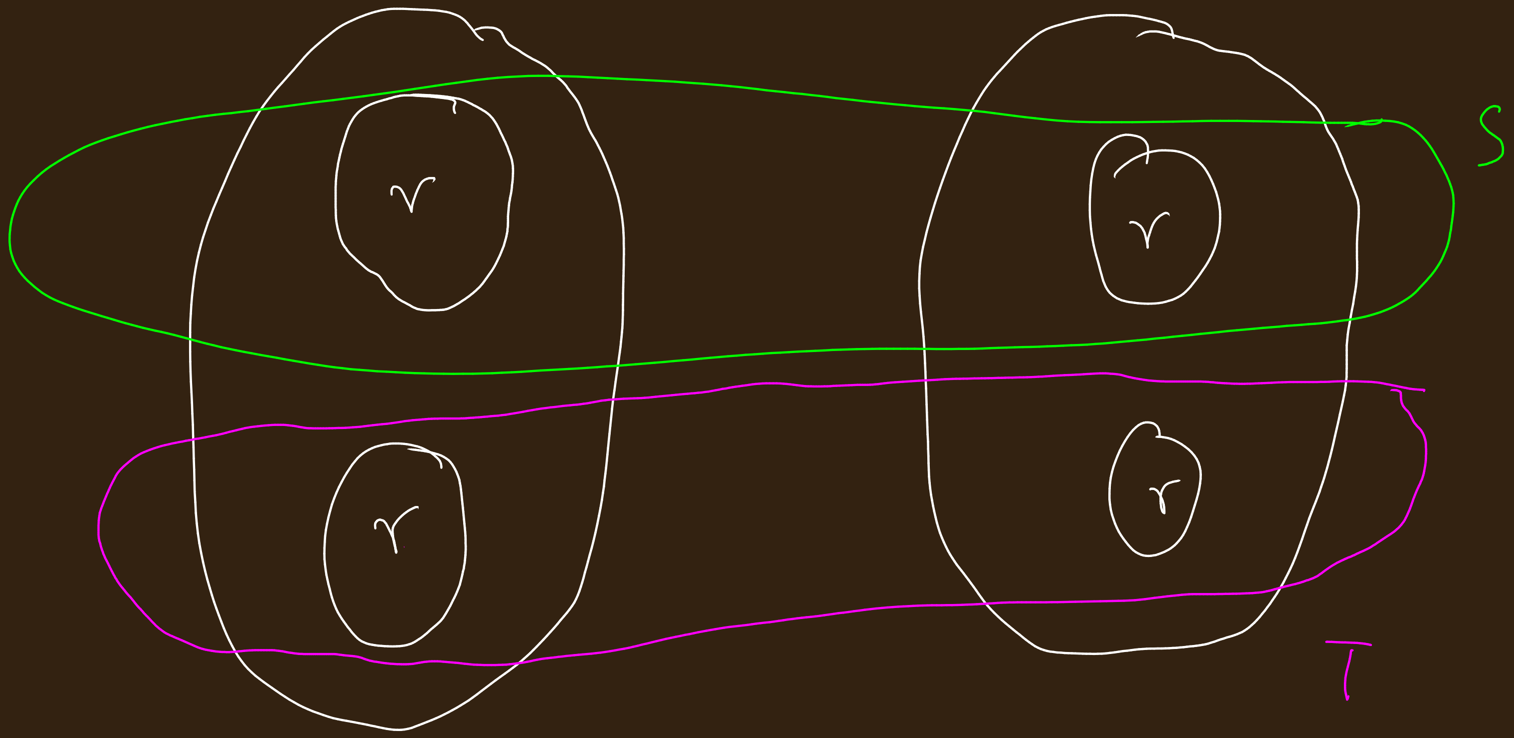
The algorithm terminates when $c_u \geqslant b_u$ for all vertices u. Let m be the number of edges of G. By the degree-sum formula, we have

$$2m = \Sigma_u\, d_u = \Sigma_u\, (b_u + c_u) \leq \Sigma_u\, 2c_u = |cut(S,T)|.$$

Also OPT $\leq$ m.

(c) Prove that this approximation ratio is tight.

$K_{2r,2r}$ complete bipartite graph




$$OPT = 4r^2$$

$$|cut(S,T)| = 2r^2$$

$$\rho = 1/2$$

2. Consider the following algorithm for EUCLIDEAN-TSP.

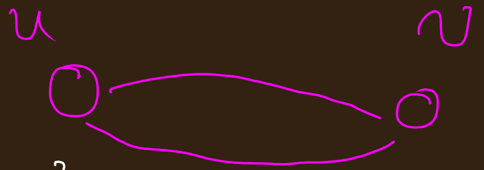
a) Select the pair (u,v) such that d(u,v) is smallest among all pairs. Start with the tour C = (u,v).

b) Repeat until C is a Hamiltonian cycle: $n-2$ iterations

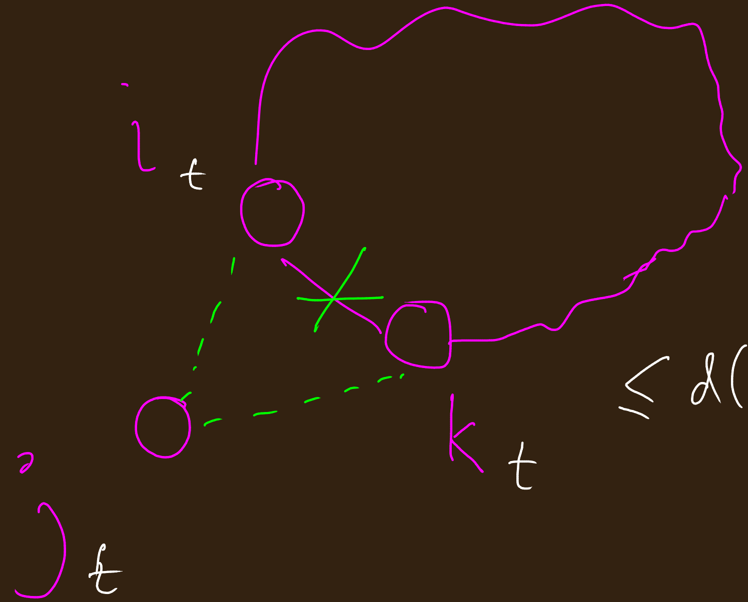
Find i ∈ C and j ∉ C such that d(i,j) is the minimum. Let k be the city next to i in C.
Replace i,k by i,j,k in C.

Prove that this is a 2-approximation algorithm.

Prim's algo chooses precisely these edges




$$|MST| = d(u,v) + \sum_{t=1}^{n-2} d(i_t, j_t)$$




Initial cost $= 2d(u,v)$

In the $t^{th}$ iteration cost of the tour increases by $d(i_t, j_t) + \boxed{d(j_t, k_t) - d(i_t, k_t)}$

$\leq 2d(i_t, j_t)$

$\leq d(i_t, j_t)$

final cost of the tour $\leqslant 2\,|MST|$

$OPT \geqslant OPT - cost(e) \geqslant MST$

The algorithm runs for $n-2$ iterations. Let the $t$-th iteration replace $i_t, k_t$ by $i_t, j_t, k_t$. The initial edge $(u,v)$ and the edges $(i_t, j_t)$ for $t = 1, 2, \ldots, n-2$ construct a minimum spanning tree of the complete weighted graph on $n$ vertices (the cities), because Prim's algorithm chooses precisely these edges.

At the beginning, the 2-city tour has cost $d(u,v) + d(v,u) = 2d(u,v)$. The $t$-th iteration increases the cost of the tour by the amount $\delta_t = d(i_t, j_t) + d(j_t, k_t) - d(i_t, k_t)$. By the triangle inequality, we have $d(j_t, k_t) \leqslant d(j_t, i_t) + d(i_t, k_t)$, that is, $d(j_t, k_t) \leqslant d(i_t, j_t) + d(i_t, k_t)$, that is, $d(j_t, k_t) - d(i_t, k_t) \leqslant d(i_t, j_t)$. Therefore $\delta_t \leqslant 2d(i_t, j_t)$. The cost of the final tour produced by the algorithm is

$$c = 2d(u,v) + \sum_{t=1}^{n-2} \delta_t \leqslant 2\left[d(u,v) + \sum_{t=1}^{n-2} d(i_t, j_t)\right],$$

where the sum within square brackets is the cost of the MST.

Let OPT be the cost of the optimal tour. If we delete any edge $e$ from the optimal tour, we get a spanning tree of cost $\text{OPT} - d(e)$. We clearly have

$$\text{OPT} \geqslant \text{OPT} - d(e) \geqslant \text{cost(MST)},$$

that is,

$$c \leqslant 2 \times \text{cost(MST)} \leqslant 2 \times \text{OPT}.$$