问题求解（四）习题

助教：宋锦华

第七周作业

- **Exercise 4.2.1.4.** Change the greedy strategy of the algorithm GMS to an arbitrary choice, i.e., without sorting $p_1, p_2, \ldots, p_n$ (removing Step 1 of GMS), assign the jobs in the on-line manner as described in Step 2 and 3 of GMS. Prove that this simple algorithm, called GRAHAM'S ALGORITHM, is a 2-approximation algorithm for MS, too.

  Proof.

  (1) Similarly we can get that the following inequality still holds

  $$cost\big(\mathrm{GMS}(I)\big) - Opt_{\mathrm{MS}}(I) \leq p_k$$

  (2) We observe $p_k \leq Opt_{\mathrm{MS}}(I)$ then

  $$\frac{cost\big(\mathrm{GMS}(I)\big) - Opt_{\mathrm{MS}}(I)}{Opt_{\mathrm{MS}}(I)} \leq \frac{p_k}{Opt_{\mathrm{MS}}(I)} \leq 1$$

- **Exercise 4.2.1.5.** Find, for every integer $m \geq 2$, an input instance I_m of MS such that $R_{\mathrm{GMS}}(I) = \frac{cost(\mathrm{GMS}(I))}{Opt_{\mathrm{MS}}(I)}$ is as large as possible.

Solution:

(1) Optimal MS must have a machine which is assigned only small jobs

(2) GMS has a additional small job

(3) The small job should have as much time as possible

$I_m = \{2m-1, 2m-1, 2m-2, 2m-2, \ldots, m+1, m+1, m, m, m, m\}$

Where $(p_1, p_2, \ldots, p_{2m+1}) = \{2m-1, 2m-1, 2m-2, 2m-2, \ldots, m+1, m+1, m, m, m\}$

$$\frac{cost(\mathrm{GMS}(I))}{Opt_{\mathrm{MS}}(I)} = \frac{4m-1}{3m} < \frac{4}{3}$$

- **Exercise 4.2.3.3**. Prove that the functions $dist$, $dist_k$, and $distance$ (defined in Example 4.2.3.2) are distance functions for TSP according to L_Δ.

Proof:

(i) Since c satisfying the triangle inequality, for every $x \in L_\Delta$, $\frac{c(\{u,v\})}{c(\{u,p\})+c(\{p,v\})} < 1$.

Through induction, we can get $\frac{c(\{u,v\})}{\sum_{i=1}^{m} c(\{p_i,p_{i+1}\})} < 1$.

So $dist(x) = 0, dist_k = 0$ and $distance(x) = 0$

(ii) the functions are *polynomial-time computable*

For complete graph, every k vertices form a path.

So $dist$ needs $C_{|V|}^3 = O(|V|^3)$ and $dist_k$ needs $C_{|V|}^3 + C_{|V|}^4 + \cdots + C_{|V|}^{k+1} = O(|V|^{k+2})$

For exhaustion search, $distance(x)$ needs $C_{|V|}^{3} + C_{|V|}^{4} + \cdots + C_{|V|}^{|V|} = O(2^{|V|})$. It is not polynomial.

Observe the equation:

$$dist_k(G, c) = \max\left\{0, \max\left\{\frac{c(\{u, v\})}{\sum_{i=1}^{m} c(\{p_i, p_{i+1}\})} - 1 \mid m + 1 \leq k\right\}\right\}$$

For each $\{u, v\} \in G$, maximizing $\frac{c(\{u,v\})}{\sum_{i=1}^{m} c(\{p_i,p_{i+1}\})}$ is equal to minimizing $\sum_{i=1}^{m} c(\{p_i, p_{i+1}\})$ which can be obtained by searching for the shortest path.

Then many polynomial-time algorithms can be used.

- **Exercise 4.2.3.4**. Let $U = (\Sigma_I, \Sigma_O, L, L_I, M, cost, goal)$ and $\overline{U} = (\Sigma_I, \Sigma_O, L, L, M, cost, goal) \in NPO$. Let h_{index} be defined as follows:

  (i) $h_{index}(w) = 0$ for every $w \in L_I$, and

  (ii) $h_{index}(u)$ is equal to the order of u according to the canonical order of words in Σ_I^*.

  Prove that

  a) h_{index} is a distance function of $\overline{U}$ according to L_I.

  b) For every δ-approximation algorithm A for U, if A is consistent for $\overline{U}$, then A is stable according to h_{index}.

Proof:

a) (i) is satisfied. (ii) computing the order is polynomial.

b) The length of $Ball_{r,h}(L_I)$ is fixed for a certain r. So there exists an upper bound δ of $R_A(x)$. Then for every $0 < r \leq p$, $\delta_{r,\delta}$ is the largest bound. Thus for any p and every real $0 < r \leq p$, there exists a $\delta_{r,\delta}$-approximation algorithm for U_r

(Some notions

Stable: A is p-stable according to h_{index} for every $p \in R^+$.

***p*-stable**: For a p and every real $0 < r \leq p$, there exists a $\delta_{r,\epsilon}$-approximation algorithm for $U_r = (\Sigma_I, \Sigma_O, L, Ball_{r,h}(L_I), M, cost, goal)$ where $Ball_{r,h}(L_I) = \{w \in L_I | h(w) \leq r\}$.

***δ*-approximation**: for every $x \in Ball_{r,h}(L_I), R_A(x) \leq \delta_{r,\epsilon}$)

- **Exercise 4.2.3.5**. Define two optimization problems $U = (\Sigma_I, \Sigma_O, L, L_I, M, cost, goal)$ and $\overline{U} = (\Sigma_I, \Sigma_O, L, L, M, cost, goal)$ from NPO with infinite $|L| - |L_I|$, and a distance function h for $\overline{U}$ according to L_I such that:

  (i) h has the property of infinite jumps, and

  (ii) for every δ-approximation algorithm A for U, if A is consistent for $\overline{U}$, then A is stable according to h.

  Solution:

  We can provide a very very simple optimization problem since $P \subseteq NP$.

  Note that a) $|L| - |L_I|$ is infinite

  b) h should satisfy $h(w) = 0$ for $w \in L_I$

  c) A is stable

Example:

Given a graph $G = (V, E)$, find the longest edge.

L_I contains all graphs whose weight are smaller than 1.

and $h = w_{max} - 1$.

Every A is stable.

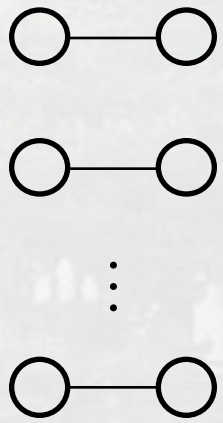
The size of the graph is infinite, but the weights are bounded.

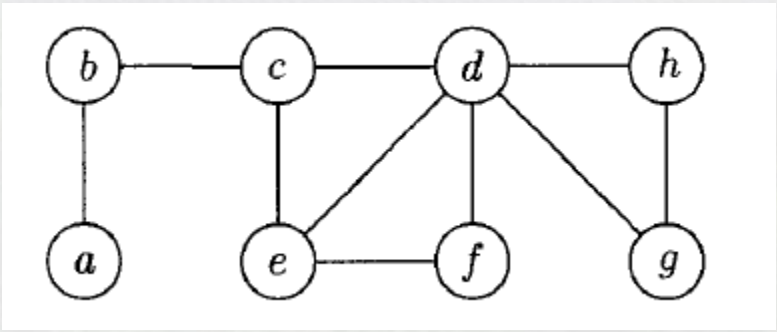
第八周作业

- **Exercise 4.3.2.3.** (a) Find a subgraph G' of the graph G in Fig. 4.3 such that Algorithm 4.3.2.1 can output a vertex cover whose cardinality is 6 while the cardinality of an optimal vertex cover for G' is 3.

  (b) Find, for every positive integer n, a Graph G_n with a vertex cover of the cardinality n, but where Algorithm 4.3.2.1 can compute a vertex cover of the size $2n$.

Solution:
(a) remove h
(b) a even cycle (or right)





- **Exercise 4.3.2.6.** Construct an infinite family of graphs for which Algorithm 4.3.2.1 always (independent of the choice of edges from *E)* computes an optimal vertex cover.

Solution:

Complete graph with $2n + 1$ $(n > 0)$ vertices (i.e., K_{2n+1}). The optimal vertex cover has $2n$ vertices while Algorithm 4.3.2.1 output a maximal matching with n edges.

It can be proved by induction.

(1) It is trivial when $n = 3$.

(2) If it is established for $n = k$.

(3) When $n = k + 1$, the graph is K_{2n+3}. We delete one edges (u, v) and all edges incident to u and v. Then the graph become K_{2k+1}, and Algorithm 4.3.2.1 output the optimal C_k, $|C_k| = 2k$. For K_{2n+3}, $C_{k+1} = C_k + \{u, v\}$, $|C_{k+1}| = 2k + 2$ is optimal.

- **Exercise 4.3.2.9.** (a) Design a polynomial-time algorithm for MIN-VCP when the set of input graphs is restricted to the trees.

  (b) Design a polynomial-time d-approximation algorithm for MIN-VCP with $d < 2$ when the input graphs have their degree bounded by 3.

Solution:

(a) (1) Select a node, whose sub-nodes are all leaves, to C and delete all connected edges (and connected leaves). Then we get a new tree. (2) For the new tree call (1). If we do not select the node, we should select all the leaves whose number is larger than 1….(try to prove it)

(b) VCP->SCP. According to **Lemma 4.3.2.12,** $d \leq Har(3) = 1 + 1/2 + 1/3 = 11/6$

Q&A