# CS300 Homework #3

TA: Seungchan Jeong

winner_sc@kaist.ac.kr

Total 100 points  Due : 2020-10-16 18:00:00 KST

- Write in English or Korean
- Make sure your writing is readable

## 1. Black-box (25 pts)

**Algorithm 1** Linear Sorting

**Require:** $M$: machine, $S$: Set of elements with order
**Ensure:** $L$: Sorted List
  $m, S \leftarrow M(s)$
  $Top, Bot \leftarrow m$
  **while** $S \neq \varnothing$ **do**
    $m, S \leftarrow M(s)$
    **if** $m \leq Bot$ **then**
      $L.PushFront(m)$
      $Bot \leftarrow m$
    **else**
      $L.PushBack(m)$
      $Top \leftarrow m$
    **end if**
  **end while**
  **return** $L$ =0

This algorithm is correct. It's enough to prove that the median $m_i$ in i'th iteration satisfies $Top_i \leq m_i$ or $m_i \leq Bot_i$. I'll prove it by mathematical induction. At first, $m_0$ = Top = Bottom. Suppose that $Bot_{k-1} \leq m_{k-1}$ or $Top_{k-1} \geq m_{k-1}$. Assume $Bot_k < m_k < Top_k$. You can check that there is no element $m \in S$ s.t. $m_{k-1} < m < m_{k-2}$ or $m_{k-2} < m < m_{k-1}$. Because if such m exists, $m_{k-1}$ must be m. However, according to algorithm, $Bot_k$ and $Top_k$ are either $m_{k-1}$ or $m_{k-2}$. So, it is impossible that $Bot_k < m_k < Top_k$.

## 2. Graph Coloring (25 pts)

Create a new graph whose vertices represent the edges of the given graph and connect two vertices in the new graph by an edge if and only if these vertices represent two edges with a common endpoint in the original graph. A solution of the vertex-coloring problem for the new graph solves the edge-coloring problem for the original graph.

# 3. Bridge and biconnected components (25 pts)

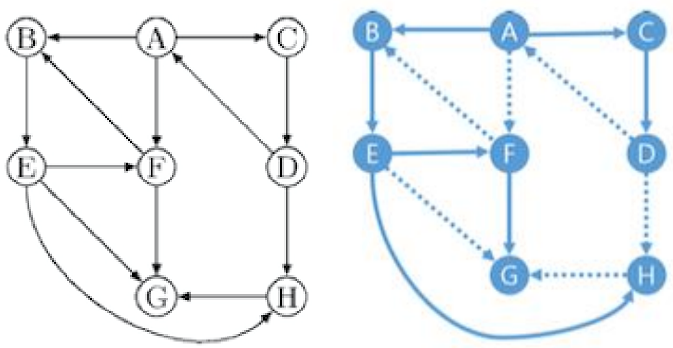
a. (→) Assume that bridge b = (i, j) is in simple cycle. Then there is another path which connects i and j. So, if we delete b, by changing b to such path, there exists path for every vertexes. It's contradiction that b is bridge. So b is not in any simple cycle.

(←) If b = (i,j) is not in any simple cycle, there is no path from i to j without edge b. So, removal of b disconnects i and j, and G. So b is bridge.

b. Define $e_1 \equiv e_2$ iff $e_1$ and $e_2$ are in same simple cycle. If ≡ is equivalence relation on set of every edge which is not bridge, it proves original claim. (∵ Equivalence class forms partition of set.)

- – If e is not a bridge, e is in some simple cycle. It means $e \equiv e$.
- – It's trivial that $e_1 \equiv e_2$ implies $e_2 \equiv e_1$.
- – Assume $e_1 \equiv e_2$ and $e_2 \equiv e_3$. $C_1$,$C_2$ be simple cycle which includes $e_1$ and $e_2$, $e_2$ and $e_3$ resp. Then $C_1 \cup C_2 - \{e_2\}$ is also simple cycle which contains $e_1$ and $e_3$. So $e_1 \equiv e_3$.

# 4. DFS (25 pts)




The order of reaching: A-B-E-F-G-H-C-D

The order of dead-ends: G-F-H-E-B-D-C-A