SC2001/CX2101 Algorithm Design and Analysis

Tutorial 6: Introduction to NP

College of Computing and Data Science *Nanyang Technological University*

This tutorial helps you develop skills in the learning outcome of the course: "Able to classify some decision problems into P or NP problems and apply greedy heuristic approach to solve NP-complete problems".

1. Problem: Given a network of cities G and a positive integer k. Are the shortest paths between all pairs of cities not longer than k? Is this problem in the class of P or NP? Justify your answers.

2. Clique problem: Given a graph G = (V, E), and a positive integer $k \leq |V|$. Does G contain a k-clique? In other words, is there a subset $V' \subseteq V$ such that $|V'| \geq k$ and every two vertices in V' are joined by an edge in E? A clique with k vertices is called k-clique. Show that the clique problem is in NP.

3. 3-CNF-SAT problem: Let $U = \{u_1, u_2, \ldots, u_n\}$ and $C = \{c_1, c_2, \ldots, c_m\}$ where each u_i is a variable and each c_j is a disjunction of 3 variables. The 3-CNF-SAT problem asks if there is a satisfying truth assignment to variables that simultaneous satisfies all the clauses in C. Show that the 3-CNF-SAT problem is in NP.

   An example of a 3-CNF-SAT problem:
   $U = \{u_1, u_2, u_3, u_4\}$ and $C = \{\{u_1, \neg u_2, u_3\}, \{\neg u_1, u_2, u_4\}\}$. Is there a truth assignment that makes
   $(u_1 \vee \neg u_2 \vee u_3) \wedge (\neg u_1 \vee u_2 \vee u_4)$ true?

4. Implement the shortestLinkTSP() algorithmshown in next page (slide 29 of lecture notes) to find a TSP tour in graph G. You may consider using a minimizing heap, a union-find data structure and other data structures in your implementation of the algorithm.

```
shortestLinkTSP(V, E, W)
{   R = E;
    C = empty;   // C is a forest

    while (no. of edges in C < |V| - 1)  {
                  remove the lightest edge vw from R;
                  if (vw does not form a cycle in C and
                       vw would not be the third edge in C incident on v or w)
                    add edge vw to C;   }

   add edge connecting the end points to C;
   return C;
}
```