



JOINT INSTITUTE
交大密西根学院

UM-SJTU Joint Institute
VE477 Intro to Algorithms

Homework 4

Wang, Tianze
515370910202

Question 1 Time vs. Space

In this problem we suppose one clock cycle is enough for executing an operation.

(1) NUDT

For 2^{64}

$$\frac{2^{64}}{33.86 \times 10^{15}} = 544.8s$$

For 2^{80}

$$\frac{2^{80}}{33.86 \times 10^{15}} = 3.6 \cdot 10^7 s$$

(2) PC

For each computer, one day it can calculate

$$24 \cdot 60 \cdot 60 \cdot 4 \text{ (cores)} \cdot 3.8 \cdot 10^9 = 1.31 \cdot 10^{15}$$

The total number of computers needed is

$$\frac{2^{64}}{1.31 \cdot 10^{15}} = 14047$$

For 2^{80} operations,

$$\frac{2^{80}}{30 \cdot 1.31 \cdot 10^{15}} = 3.07 \cdot 10^7$$

(3) Storage

To store 2^{64} with the hard drive (16TB),

$$\frac{2^{64}}{8 \cdot 2^{12} \cdot 16} = 3.5 \cdot 10^{13}$$

for 2^{80}

$$\frac{2^{80}}{8 \cdot 2^{12} \cdot 16} = 2.3 \cdot 10^{18}$$

Question 2 Critical Thinking

The critical part in this problem is that every element must be visited. So in one pass, S' could be obtained as

1. First fill in an array $A[k]$ of size k with the first k element in S . Obviously these k elements are visited.
2. For the rest elements, each time get a random number $t = \text{rand}(1, \text{element index})$, and if the rand number t is smaller or equal to k , replace $A[t]$ with the current element.

This method is effective in that each time when visiting a new element, it has $1/k \cdot k/\text{index} = 1/\text{index}$ of chance to be selected. And the former elements are derived similarly, so each element has same probability of being selected.

Question 3 Algorithm and Complexity

(1) Algorithm

The algorithm is given as

```
Input : i
Output : Sum over the  $i$ -th line
1 Function sum(i):
2    $A[0 : i][0 : i] \leftarrow \text{an } (i + 1) \times (i + 1) \text{ matrix};$ 
3    $A[1 : i][1] \leftarrow 1;$ 
4    $A[1][1 : i] \leftarrow 1;$ 
5    $A[0][1 : i] = A[0 : i][0] = 0;$ 
6   for  $0 < k \leq i$  and  $0 < j \leq i$  do
7     if  $k < j$  then
8        $A[k][j] = A[k - 1][j - 1] + A[k - 2][j - 1] + A[k - 3][j - 1]$ 
9     else if  $k > j$  then
10       $A[k][j] = A[k - 1][j - 1] + A[k - 1][j - 2] + A[k - 1][j - 3]$ 
11    else
12       $A[k][j] = A[k - 1][j - 1] + A[k - 2][j - 1] + A[k - 1][j - 2]$ 
13    end if
14  end for
15  return  $\sum A[i][1 : i] + A[1 : i][i] - A[i][i]$ 
16 end
```

(2) Complexity

The complexity of this algorithm is $O(i^2)$ since we need to calculate each element in the matrix.

This method is correct in that it exactly follows the definition of the triangle. Each layer i is defined as

$$A[i][1 : i] \text{ and } A[1 : i][i]$$

Question 4 SAT to 3-SAT

Not done yet.

Question 5 Clique Problem

(1) Definition

A clique problem is to find the maximum clique(Complete graph) in a given graph.

(2) \mathcal{NP}

It is easy to verify that, given a simple certificate, namely given the clique, we are able to check the clique is in the graph and whether it is a clique in polynomial time. ($\mathcal{O}(n^2)$)

(3) 3-SAT to Clique

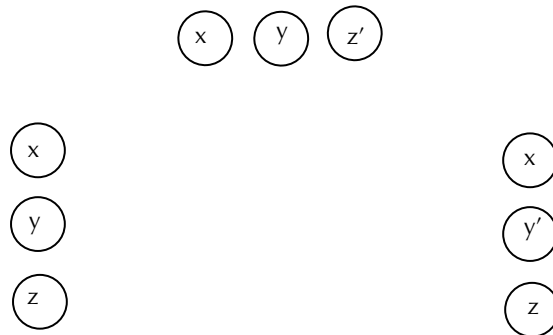
A 3-SAT problem can be transferred to a Clique problem by: setting the 3 literals as from the same group, and connect each literal to all literals satisfying:

1. In other groups
2. Different literal from this literal

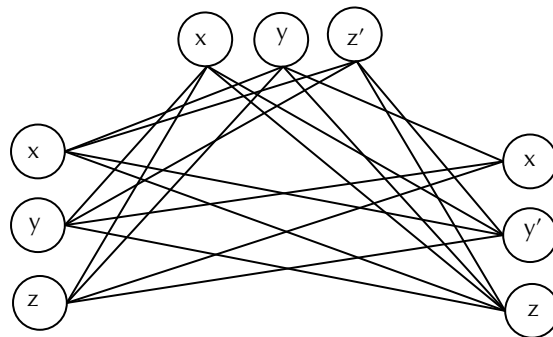
Do this step for all groups. And the clique in this graph evaluates true will lead to a true for the 3-SAT problem. Since this operation can be done in polynomial time, we can conclude the 3-SAT problem can be reduced to a clique problem.

An example is given as

$$(x \vee y \vee z) \wedge (x \vee y \vee z') \wedge (x \vee y' \vee z)$$



The clique is then



(4) Complexity Class

We know that 3-SAT problem is NP-Complete. And since it can reduce to a clique problem in polynomial time, the clique problem is also \mathcal{NP} -complete.

Question 6 IND-SET problem

(1) Definition

A set of vertices from a graph that no two are adjacent.

(2) Definition II

Given a graph G and an integer k , does there exist a set of vertex whose size is larger or equal to k s.t. all elements inside are non-adjacent.

(3) \mathcal{NP}

To prove it is in \mathcal{NP} , we let the clue to be the set of vertex. Through linear search, we can check whether all vertices in the set are non-adjacent, this will lead to $\mathcal{O}(n^2)$ operations, which is polynomial time.

(4) Graph construction

The Graph is constructed as:

Given the graph G with (V, E) , let G_1 be the complete graph with V , and denote all edges from G_1 as E_1 , the G' is then defined as

$$(V, E_1 \setminus E)$$

(5) Complexity Class

Since we can reduce a 'k-clique' problem into an 'independent set problem' within polynomial operations, the independent set problem is also in \mathcal{NP} -complete complexity class.