Algorithm Problem Solving
17ECSE309

Hamiltonian cycle

USN:01FE15BEC138
Name : Rahul Jain

- A **Hamiltonian cycle**, also called a **Hamiltonian circuit**, **Hamilton cycle**, or **Hamilton circuit** is a graph **cycle** (i.e., closed loop) through a graph that visits each node exactly once .A graph possessing a **Hamiltonian cycle** is said to be a **Hamiltonian** graph.

Ways of checking whether a graph contains a Hamiltonian Path or not.

1.Using all permutation of the vertices

- **Suedo code**

```
function check_all_permutations(adj[][], n)
for i = 0 to n
p[i]=i
while next permutation is possible
valid = true
for i = 0 to n-1
      if adj[p[i]][p[i+1]] == false
   valid = false
break
if valid == true return true
p = get_next_permutation(p)
return false
```

- Time complexity is O(N * N!)

- Using dynamic programming
- **Suedo code**

```
function check_using_dp(adj[][], n)
for i = 0 to 2^n
for j = 0 to n
dp[j][i] = false
for i = 0 to n
dp[i][2^i] = true
For i = 0 to 2^n
for j = 0 to n
if j^th bit is set in i
    for k = 0 to n
        If j != k and k^th bit is set in i and adj[k][j] == true
            If dp[k][ i XOR 2^j ] == true
                dp[j][i]=true
                break
for i = 0 to n
    if dp[i][2^n-1] == true
        return true
return false
```

- Time complexity of the above algorithm is $O(2^n n^2)$.

- Using Depth first search and backtracking
- Suedo code

```
#define NOT_IN_STACK 0
#define IN_STACK 1
bool dfs(int v, bool adj[][MAXN], int label[MAXN], int instack_count, int n)
{ if(instack_count == n)
return true;
for(int i=0; i<n; i++)
if(adj[v][i] && label[i] == NOT_IN_STACK)
{
label[i]=IN_STACK;
if(dfs(i, adj, label, instack_count+1, n))
return true;
 label[i]=NOT_IN_STACK;
}
 return false;
 }
bool check_using_dfs(bool adj[][MAXN], int n)
{ int label[MAXN];
for(int i=0; i<n; i++)
label[i]=NOT_IN_STACK;
for(int i=0; i<n; i++){
 label[i]=IN_STACK;
if(dfs(i, adj, label, 1, n))
 return true;
label[i]=NOT_IN_STACK;
 }
 return false;
 }
```

- Worst case complexity of using DFS and backtracking is O(N!).

Application

- Multi-threshold CMOS ,which requires a Hamiltonian-cycle routing to serially connect all the power switches.
- *RoundTrip* game (a.k.a. GrandTour) you must find an Hamiltonian circuit in a grid of points in which some of the edges are given.

References

- www.hackerearth.com/practice/algorithms/graphs/hamiltonian-path/
- http://mathworld.wolfram.com/HamiltonianCycle.html