

The background of the entire image is a dark blue field filled with a pattern of red dots. These dots are arranged in a way that they form a large, faint, stylized circular shape, reminiscent of a DNA helix or a molecular structure, with the density of the dots varying to create a sense of depth and movement.

HUST

ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

ONE LOVE. ONE FUTURE.



ĐẠI HỌC
BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY
OF SCIENCE AND TECHNOLOGY

Applied Algorithm Lab

Bus inter-city

ONE LOVE. ONE FUTURE.

- There are n city: $1, 2, \dots, n$.
 - Between 2 cities, there may exist a **bidirectional road** connecting them.
 - Each city i has a **bus route i** with:
 - **$C[i]$** : The fare required each time a passenger boards the bus.
 - **$D[i]$** : The maximum number of cities the bus can travel through in a single journey using the connected roads.
- **Objective**: Find a path from city 1 to city n with lowest cost.
- **Input**: n, m (#roads), $\{C(1), D(1)\}, \dots, \{C(n), D(n)\}$, the set of roads
- **Output**: The minimum cost of the path

- Example

Input

6 6
10 2
30 1
50 1
20 3
30 1
20 1
1 2
1 3
1 5
2 4
2 5
4 6

Output

30

Explain: the path found from city 1 to city 6 with minimum cost is

On bus from city 1 -> city 4 cost \$10

On bus from city 4 -> city 6 cost \$20

Total cost: $\$10 + \$20 = \$30$

- Idea to solve: Use Dijkstra algorithm on a new cost graph
 - Build **cost graph**: A graph with same node set but edge (u,v) means we can go on bus from u to go to v , cost $c(u,v) = C[u]$, $\text{dist}[u][v] \leq D[u]$
 - Use BFS(u) with a depth limitation
 - Run Dijkstra from node 1 to find min cost path to city n

Bus inter-city - Implementation

```
1  #include <bits/stdc++.h>
2  using namespace std;
3  #define maxN 5001
4  #define int long long
5  int m, n, C[maxN], D[maxN], a, b;
6  vector<int> A[maxN];
7  int dis[maxN][maxN], disfrom1[maxN];
8  bool visited[maxN];
9  void input(){
10     cin >> n >> m;
11     for (int i = 1; i <= n; i++){
12         cin >> C[i] >> D[i];
13     }
14     for (int i = 1; i <= m; i++){
15         cin >> a >> b;
16         A[a].push_back(b);
17         A[b].push_back(a);
18     }
19     for (int i = 1; i <= n; i++){
20         for (int j = 1; j <= n; j++){
21             dis[i][j] = INT_MAX;
22         }
23     }
24 }
```

Bus inter-city - Implementation

```
25 void buildGraph(int start){
26     queue<pair<int, int>> qe;
27     qe.push(make_pair(start, 0));
28     dis[start][start] = 0;
29     while (!qe.empty()){
30         auto u = qe.front();
31         qe.pop();
32         if (u.second < D[start]){
33             for (auto v : A[u.first]){
34                 if (dis[start][v] > C[start]){
35                     dis[start][v] = min(C[start], dis[start][v]);
36                     qe.push(make_pair(v, u.second + 1));
37                 }
38             }
39         }
40     }
41 }
```

Bus inter-city - Implementation

```
70  signed main(){
71      input();
72      for (int i = 1; i <= n; i++){
73          buildGraph(i);
74      }
75      Dijkstra();
76      return 0;
77  }
```


A large graphic on the left side of the slide. It features a dark blue background with a circular pattern of red dots of varying sizes, creating a sense of depth and movement. The word "HUST" is centered within this pattern in a white, bold, sans-serif font.

HUST

THANK YOU !