HUST

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

ONE LOVE. ONE FUTURE.

DATA STRUCTURES AND ALGORITHMS

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

DATA STRUCTURES AND ALGORITHMS

WEEK 4: BRANCH AND BOUND

ONE LOVE. ONE FUTURE.

CONTENT

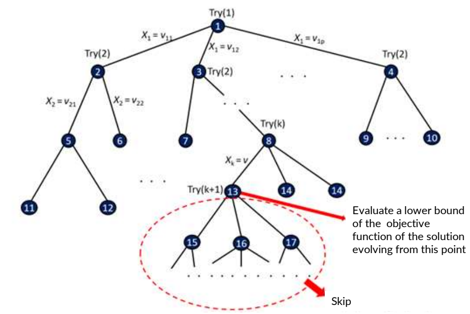
- General diagram of branch and bound
- The traveling salesman problem

GENERAL DIAGRAM

- Branch and Bound: One of the methods to solve combinatorial optimization problems
  - Use the backtracking technique to list all options, thereby retaining the best option
  - Use bound evaluation (upper bound for the problem of finding max and lower bound for the problem of finding min) to cut down the search space during the listing process.

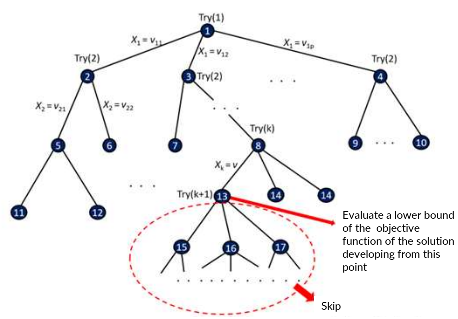
GENERAL DIAGRAM

- Consider the problem of finding the minimum of the objective function in which the solution is represented by a set of variables X = (X_1, X_2, . . ., X_n).
- The Try(k) function is used to try the value for variable X_k during the listing process
- Symbol f*: objective function, the value of the best solution found



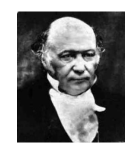
GENERAL DIAGRAM

- After assigning the value v to X_k, we evaluate the lower bound g of the objective function of the development options continuing from point 13.
- If g is greater than or equal to f*, do not continue developing the solution from point 13



THE TRAVELING SALESMAN PROBLEM

- Problem statement:
  - A tourist wants to visit n cities 1, 2, . . ., n
  - A itinerary (journey) is a way to start from city 1, go through all the remaining cities, each city exactly once, and then return to starting city 1.
  - Know that c(i, j) is the cost of traveling from city i to city j (i, j = 1, 2,..., n)
  - Find the itinerary with the smallest total cost.


Sir William Rowan Hamilton [1]
(1805 – 1865)

Some comments:

- The number of traveler's itineraries is (n-1)!
- We have a one-to-one correspondence between a tourist's itinerary:
- 1 ->x[2] -> x[3] ->...-> x[n]-> 1 with a permutation x = (x[2], x[3],..., x[n]) of n -1 natural number 2, 3,..., n.
- Journey cost: f(x) = c(1, x[2]) + c(x[2], x[3]) +... + c(x[n-1],x[n]) + c(x[n] ,1)

THE TRAVELING SALESMAN PROBLEM

- Solve using searching in the whole space:
- Itinerary : x = (1, x[2], x[3],..., x[n], 1)

```
try(k){//try values assignable to x[k]
  for v in candidates(k) do {
     if (check(v,k)) then {
        x[k] = v;
        [Update the data structure D]
        if (k == n) then solution();
        else try(k+1);
        [Recover the data structure D]
     }
  }
}
```

Determine:
1) candidates(k)
2) check(v, k)

```
Init:
•  f* = +∞; f = 0; x[1] = 1;
•  for (int v = 2; v<=n; v++)  visited[v]=0;

void Try(int k) {
    for (int v = 2; v<=n; v++) {
       if (!visited[v]) {
           x[k] = v;
           visited[v] = 1;
           f = f + c(x[k-1],x[k]);
           if (k == n) { //Update record
              ftemp = f + c(x[n],x[1]);
              if (ftemp < f*) f* = ftemp;
           }
           else Try(k + 1);
           f = f – c(x[k-1],x[k]);
           visited[v] = 0;
        }
    }
}
```


THE TRAVELING SALESMAN PROBLEM

Solve using branch and bound

Calculate bound:

- Let c_{min} = min { $c(i, j)$, i, j = 1, 2, ..., n, $i \neq j$ } be the minimum traveling cost between cites
- Need to estimate the itinerary cost for the current branch corresponding to the part (1, u_2, . . ., u_k) passing through: $1 \rightarrow u_2 \rightarrow . . . \rightarrow u_{k-1} \rightarrow u_k$
- If the lower bound $g(1, u_2, . . ., u_k) \geq f^*$ then skip (1, u_2, . . ., u_k)

THE TRAVELING SALESMAN PROBLEM

- Need to estimate the itinerary cost for the current branch corresponding to the part (1, u_2, . . ., u_k) passing through: $1 \rightarrow u_2 \rightarrow . . . \rightarrow u_{k-1} \rightarrow u_k$
  - The cost for the partial itinerary (1, u_2, ..., u_k) is:

    $$\sigma = c(1,u_2) + c(u_2, u_3) + ... + c(u_{k-1}, u_k)$$

  - For a full itinerary:

    $$1 \rightarrow u_2 \rightarrow . . . \rightarrow u_{k-1} \rightarrow u_k \rightarrow u_{k+1} \rightarrow u_{k+2} \rightarrow ..... \rightarrow u_n \rightarrow 1$$

- We need n-k+1 more stages, each stage has a cost at least c_{min}, hence the minimum cost for the remaining itinerary : $(n\text{-}k+1)\ c_{min}$
- If the partial itinerary is (1, u_2, ..., u_k) then the full itinerary has the cost at least $g(1, u_2, ..., u_k) = \sigma + (n\text{-}k+1)\ c_{min}$

THE TRAVELING SALESMAN PROBLEM

- Function Try(k) finds the optimal solution to the tourist problem using branch and bound techniques

```
Main(){
  //Init:
  f* = +∞; f = 0; x[1] = 1;
  for v = 2 to n do  visited[v] = false;
  Try(2);
  print(f*);
}
```

```
Try(k) {
    for v = 2 to n do {
       if not visited[v] {
           x[k] = v;
           visited[v] = true;
           f = f + c(x[k-1],x[k]);
           if k = n then { //Update record
              ftemp = f + c(x[n],x[1]);
              if (ftemp < f*) f* = ftemp;
           }
           else {
              g = f + (n-k+1)*cmin;
              if g < f*  then Try(k+1);
           }
           f = f - c(x[k-1],x[k]);
           visited[v] = false;
        }
    }
}
```

HUST

hust.edu.vn fb.com/dhbkhn

THANK YOU !