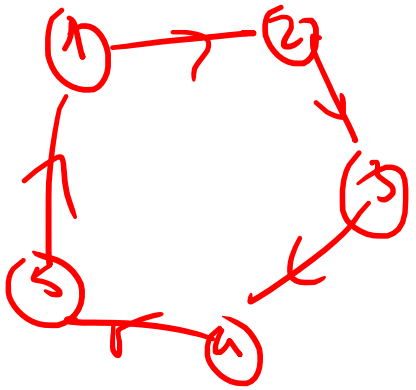
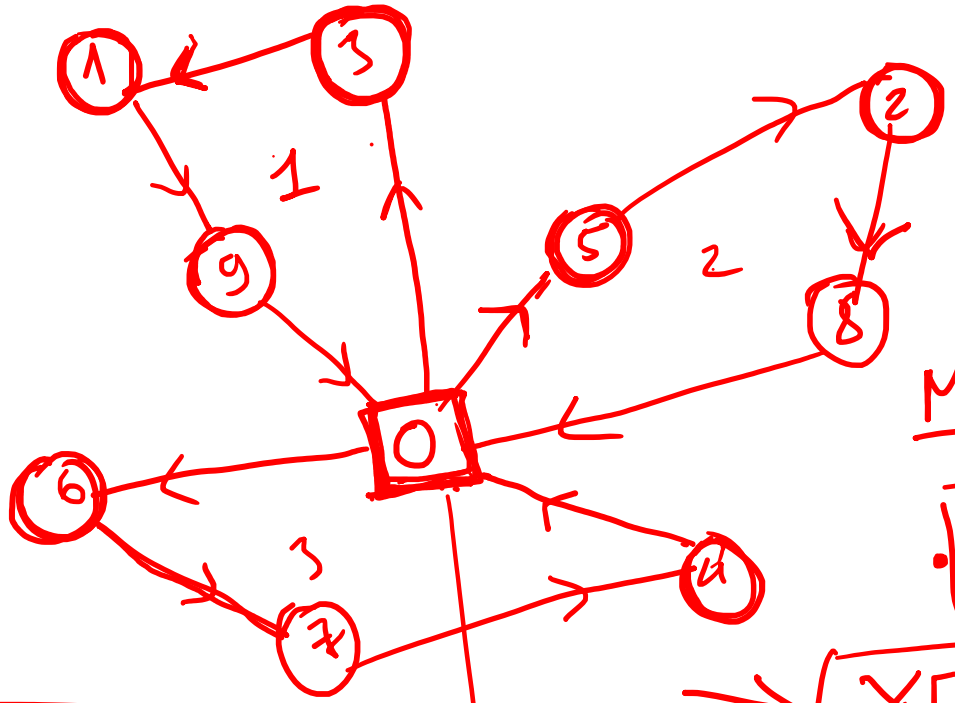

1
2
3
4
5



$k = 3$ xe, tải trọng Q.

Backtracking with Branch-and-Bound.

- TSP → có 1 route visit all points
- CVRP → có k routes visit all points.

TSP → $x_1, x_2, x_3 \ldots x_n, x_1$.

Modelling: $X[1..n]$, $X[i]$ là điểm tiếp theo của điểm KH i

$Y[1..k]$: $Y[k]$ là điểm tiếp theo của điểm 0 trên route của xe thứ k

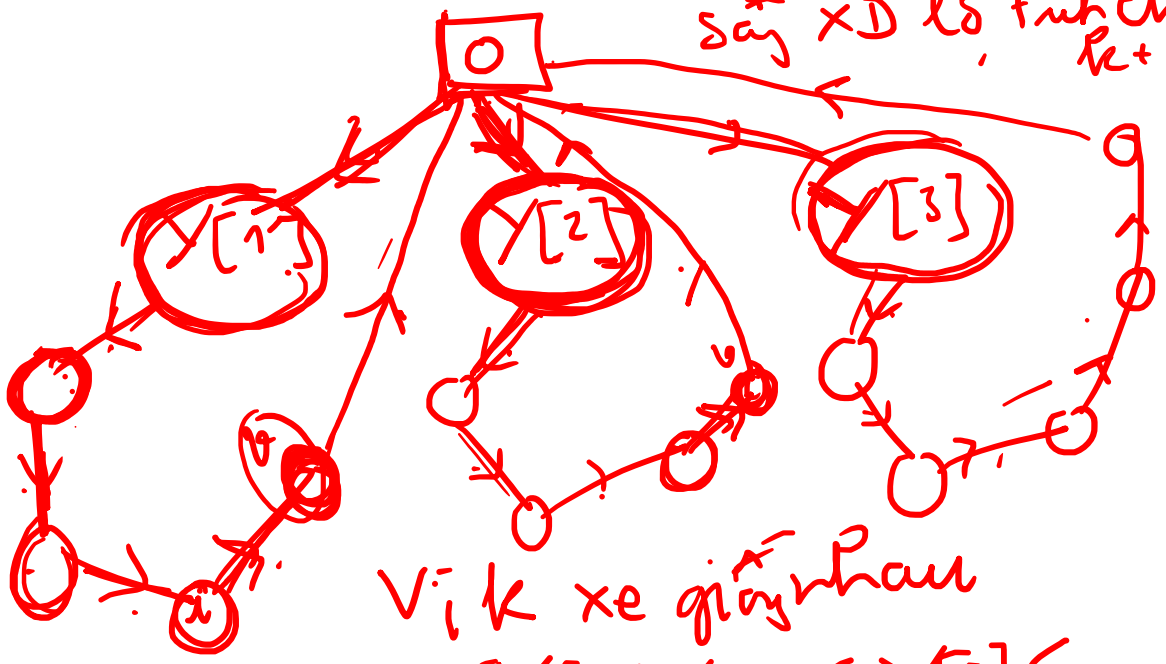
$\Rightarrow$ $Y[1] = 3$, $Y[2] = 5$, $Y[3] = 6$.
$X[1] = 9$, $X[2] = 8$, $X[3] = 1$, $X[4] = 0$, $X[5] = 2$, $X[6] = 7$
$X[7] = 4$, $X[8] = 0$, $X[9] = 0$.

```
Try(k)
{
  for v ...
    x_k = v
    if k = n solution()
    else Try(k+1)
}
```

kho · các BT trước $X[1..n]$ → Try(k) thử giá trị cho $X[k]$

· CVRP
- TRYX(k): thử giá trị cho $X[k]$
- TryY(k): thử giá trị cho $Y[k]$.

Thứ tự duyệt: Duyệt giá trị cho Y[1..K], với mỗi bộ g/trị của Y[1..K], ta duyệt các g/trị cho X[1..N] để XD lộ trình cho các xe. XD xong lộ trình cho xe k sẽ chuyển sang XD lộ trình cho xe k+1



Với k xe giống nhau

g/s $Y[1] < Y[2] < \dots < Y[k]$

→ Total segments = N + K

```
TRyY(k)//thử g/trị cho Y[k]
for v ∈ {...} do
    if checkY(v,k) then
        Y[k] = v;
        // update
        if k = K then
            TryX(X[1],1)
```

```
TryX(i,k)// XD điểm tiếp theo cho điểm i trên route[k]
for v ∈ {...}
    if checkX(v,i,k) then
        X[i] = v;
        // update
        if v = 0 then // hoàn tất route[k]
            if k = K then solution()
            else TryY(Y[k+1], k+1);
        else TryX(v,k);
```

- load[k]: lượng hàng trên xe k.
- nbSegments: số segments đã visited
- visited[v] = true : v đã được visited.
- fmin : độ dài của best solution
- f : tổng quãng đường đã đi qua.

TryY(k) $Y[1] < Y[2] < .. < Y[K]$

```
for v = Y[k-1]+1 → n-K+k do
  if checkY(v,k) then
    Y[k] = v; visited[v] = true;
    f = f + C[nbSegment, Y[k]]; load[k] += d[v];
    if k = K then TryX(Y[1], 1)
    else
        TryY(k+1)
    visited[v] = false
    f = f - C[v, Y[k]]; load[k] -= d[v];
    nbSegments = nbSegments - 1;
```

TryX(i, k) // tìm điểm tiếp theo X[i] của điểm i trên route[k].

```
for v = 0 → N do
if checkX(v, i, k) then
  X[i] = v; visited[v] = True;
  f = f + C[i, X[i]]; load[k] += d[v].
  nbSegment = nbSegment + 1;
  if v = 0 then // quay về kho
    if k = K then solution()
    else TryX(Y[k+1], k+1).
  else TryX(v, k);
  visited[v] = false
  f = f - C[i, X[i]]; load[k] = load[k] - d[v];
  nbSegments = nbSegment - 1;
```