

Assignment 3: The Rectilinear Traveling Salesperson Problem

Pseudocode: Exhaustive Optimization Algorithm

```
void print_cycle(int n, point2D *P, int *seq)
```

```
    For i to n
```

```
        cout<<"(" << P[seq[i]].x << ", " << P[seq[i]].y << ")";
```

```
    End For
```

```
    cout<<"(" << P[seq[0]].x << ", " << P[seq[0]].y << ") "<<endl;
```

$$\sum_{i=0}^n 1 = \left(\frac{n-0}{1} + 1\right)1 = n + 1$$

T.C : $n + 1$

Big-Oh: $O(n)$

```
void print_perm(int n, int *A, int sizeA, point2D *P, int *&bestSet, float &bestDist)
```

```
    (1) int i;
```

```
    (1) float dist;
```

```
    if (n == 1) {
```

```
        (1) dist = 0;
```

```
        For i to sizeA - 1
```

```
            dist += sqrt((pow((P[A[i]].y - P[A[i + 1]].y), 2)) + (pow((P[A[i]].x - P[A[i + 1]].x), 2)));
```

```
        End For
```

$$\sum_{i=0}^{n-1} 8 = \left(\frac{n-1-0}{1} + 1\right)8 = 8n$$

```
    (8) dist += sqrt((pow((P[A[sizeA - 1]].y - P[A[0]].y), 2)) + (pow((P[A[sizeA - 1]].x - P[A[0]].x), 2)));
```

```
    if (dist < bestDist)
```

```
        bestDist = dist;
```

```
        for i to sizeA - 1
```

```
            bestSet[i] = A[i];
```

```
        End For
```

```
    End If
```

$$\sum_{i=0}^{n-1} 1 = \left(\frac{n-1-0}{1} + 1\right)1 = n \text{ -----} > n+1$$

```

else {
    for i to n - 1
        print_perm(n - 1, A, sizeA, P, bestSet, bestDist);
        if (n % 2 == 0) {
            int temp = A[i];
            A[i] = A[n - 1];
            A[n - 1] = temp;
        }
        End If
    else
    {
        int temp = A[0];
        A[0] = A[n - 1];
        A[n - 1] = temp;
    }
    End Else
    End For
    End Else
    print_perm(n - 1, A, sizeA, P, bestSet, bestDist);
    End If

```

$$1 + \max(n+1, n!) = 1 + n!$$

T.C: $(8n + 10) * (n!)$

Big-Oh: $O(n * n!)$

```

C:\Users\zz-jmarvel\Documents\Visual Studio 2015\Projects\335_assignment3\Debug\335_assignment3.exe
CPSC 335-x - Programming Assignment #3
Rectilinear traveling salesperson problem: exhaustive optimization algorithm
Enter the number of vertices (>2)
4
Enter the points; make sure that they are distinct
x=2
y=0
x=1
y=1
x=3
y=1
x=0.1
y=0
The Hamiltonian cycle of the minimum length
(2,0)(3,1)(1,1)(0.1,0)(2,0)
Minimum length is 6.65958
elapsed time: 2.8e-05 seconds
Press any key to continue . . .

```

Pseudocode: Improved Nearest Neighbor Algorithm

int farthest_point(int n, point2D *P)

(2) int i, j = 0, index = 0;

(2) float dist = 0, max_dist = 0;

for i to n-1

dist = 0;

for (j = 0; j < n - 1; j++)

dist += (P[i].x - P[j].x)*(P[i].x - P[j].x)+(P[i].y - P[j].y)*(P[i].y - P[j].y); (9)

if (max_dist < dist) 1 + max(3,0) = 4

max_dist = dist;

index = i;

dist = 0;

End If

End For

End For

(1) return index;

$$\sum_{i=0}^{n-1} \sum_{j=0}^{n-1} 13 = \left(\frac{n-1-0}{1} + 1\right)13 = 13n = \sum_{i=0}^{n-1} 13n = \left(\frac{n-1-0}{1} + 1\right)13n = 13n(n)$$

T.C: $13n(n) + 4 = 13n^2 + 4$

Big-Oh: $O(n^2)$

int nearest(int n, point2D *P, int A, bool *Visited)

(2) int i, index = 0;

(2) float dist = 0, min_dist = 0;

for i to n

dist += (P[A].x - P[i].x)*(P[A].x - P[i].x)+(P[A].y - P[i].y)*(P[A].y - P[i].y);

if (!Visited[i])

1 + (1 + max(3,0)) = 1 + (1+3) = 5

if (dist < min_dist)

min_dist = dist;

index = i;

dist = 0;

End If

End For

(1) return index;

$$\sum_{i=0}^n 14 = \left(\frac{n-0}{1} + 1\right)14 = 14n + 14$$

T.C: $14n + 14$

Big-Oh: O(n)

```
void print_cycle(int n, point2D *P, int *seq)
```

```
    For i to n
```

```
        cout << "(" << P[seq[i]].x << "," << P[seq[i]].y << ")";
```

```
    End For
```

$$\sum_{i=0}^n 1 = \left(\frac{n-0}{1} + 1\right)1 = n + 1$$

```
    cout << "(" << P[seq[0]].x << "," << P[seq[0]].y << ") " << endl;
```

$$(n+1)+1 = n + 2$$

T.C: n + 2

Big-Oh: O(n)

```
c:\users\zz-jmarvel\documents\visual studio 2015\Projects\335_assignment3part2\Debug\335_assignment3part2.exe

CPSC 335-x - Programming Assignment #3
Rectilinear traveling salesperson problem: INNI algorithm
Enter the number of vertices (>2)
8
Enter the points; make sure that they are distinct
x=0
y=4
x=2
y=1
x=1
y=6
x=2
y=7
x=3
y=5
x=3
y=2
x=5
y=2
x=6
y=5
The Hamiltonian cycle of a relative minimum length
(5,2)(6,5)(3,2)(3,5)(2,7)(1,6)(2,1)(0,4)(5,2)
The relative minimum length is 36
elapsed time: 9e-06 seconds
Press any key to continue . . .
```