Abraham Ludlam

CPSC 335-02

October 22nd, 2016

Project #3: Euclidean Traveling Salesperson Problem

Input: a positive integer n and a list P of n distinct points representing vertices of a Euclidean graph

Output: a list of n points from P representing a Hamiltonian cycle of minimum total weight for the graph.

**Pseudocode- Exhaustive Optimization:**

Int smallest = -1; \Time = 1

Nodes BHamil[n+1];

Generate permutations function(n); \Time = n!

{

int i;

if (n == 1)

{

float sum=0; \time =1

for(i=0;i<n; i++) \time =n

{

float xv= P[A[i]].x-P[A[i+1]].x \Time= 3

float yv= P[A[i]].y-P[A[i+1]].y \Time = 3

sum+= squareroot((xv\*xv)+(yv\*yv)) \Time = 5

}\Time = 11n

if(sum<smallest or smallest = -1)

{

for(i=0;i<n; i++) \Time = n

{

BHamil[i]= P[A[i]] \Time= 1

} \Time = n

BHamil[n-1]= P[A[0]]; \Time= 2

}\Time = n+4

} \Time = 12n+5

else

{

for(i = 0 ; i< n-1; i++) \Time = n-1

{

generate\_perm(n - 1, A, sizeA);

if (n%2 == 0) \Time = 2

{

int temp = A[i]; \Time = 1

A[i] = A[n-1]; \Time = 2

A[n-1]=temp; \Time = 2

} \Time= 7 for upper if

else

{

int temp = A[0]; \Time =1

A[0] = A[n-1]; \Time= 2

A[n-1]=temp; \Time = 2

} \Time = Max(7,7) = 7

} \Time = 7n - 7 for else statement, Max of(12n+5,7n-7)= 12n+5

generate\_perm(n - 1, A, sizeA);

}

}

return Bhamil Time= 1

Total Time Complexity= **(n!)(12n+5)+2**

**Proving that (n!)(12n+5)+2 is O(n\*n!)**

**C:\Users\Steve\Desktop\P1 335\p41.png**

12 is a non-negative constant, therefore the algorithm is O(n\*n!)

**Pseudocode- Improved Nearest Neighbor Algorithm**

float max=0; \Time =1

int position;

node start;

node \*unseen = P; \Time= 1

float hold;

float min;

int usize =n ; \Time = 1

node answer[n+1]

node \*current;

int apos = 1 \Time =1

for(int x=0; x<n-1; x++) \Time= n

{ for(i=x+1; i<n; i++) \Time =

{

float xv= P[x].x-P[i].x \Time= 2

float yv= P[x].y-P[i].y \Time =2

hold= squareroot((xv\*xv)+(yv\*yv)) \Time= 5

if(hold>max) \Time= 1

{

max=hold; \Time=1

start=P[x]; \Time=1

position = i; \Time=1

} \Time = 4

} \Time = 13

} \Time = C:\Users\Steve\Desktop\P1 335\here.png

min = max;

for(int i = position; i<n-1; i++) \Time= Max of n-1

{

unseen[i]=unseen[i+1] \Time=2

} \Time = 2n-2

answer[0]=start; \Time =1

usize--; \Time=1

current = start; \Time =1

while(usize) \Time = n-1

{

for(i=0; i<usize; i++) \Time = Max n

{

float xv= current.x-unseen[i].x \Time= 3

float yv= current.y-unseen[i].y \Time =3

hold= squareroot((xv\*xv)+(yv\*yv)) \Time= 5

if(hold<min) \Time= 1

{

min=hold; \Time=1

position = i; \Time=1

} \Time = 3

} \Time= 14n

min = max;

current = unseen[position] \Time =1

answer[apos++]= current \Time =2

for(int i = position; i<usize-1; i++) \Time= Max of n-1

{

unseen[i]=unseen[i+1] \Time=2

} \Time = 2n-2

usize--; \Time 1

} \Time = (n-1)(16n+1)= 16n2-15n-1

answer[n]=start; \Time =1

return answer \Time =1

Total Time Complexity = **16n2-15n-1+9+13/2n2-39/2n+2n-2 = 45/2n2-65/2n+6**

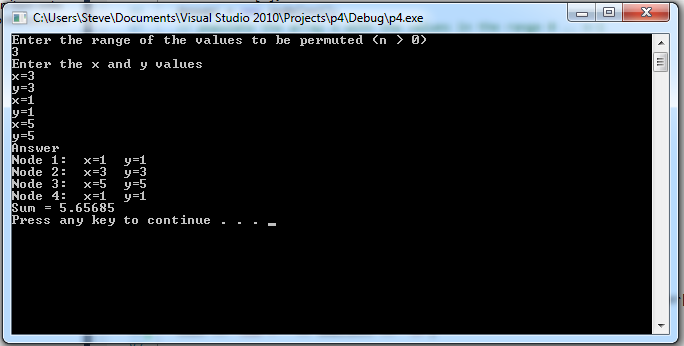
**Proving that 45/2n2-65/2n+6 is O(n2)**

C:\Users\Steve\Desktop\P1 335\Untitled42.png

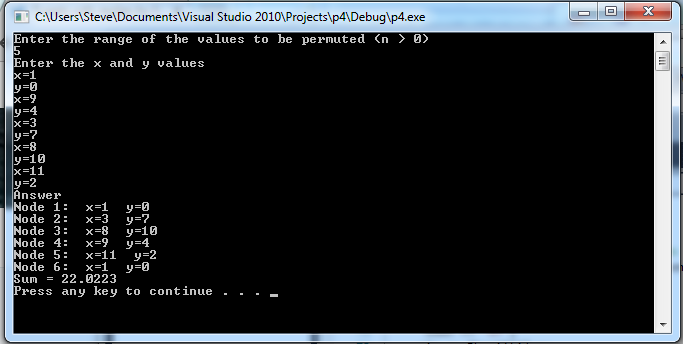
45/2 is a non-negative constant, therefore 45/2n2-65/2n+6is O(n2)

**Sample Outputs:**

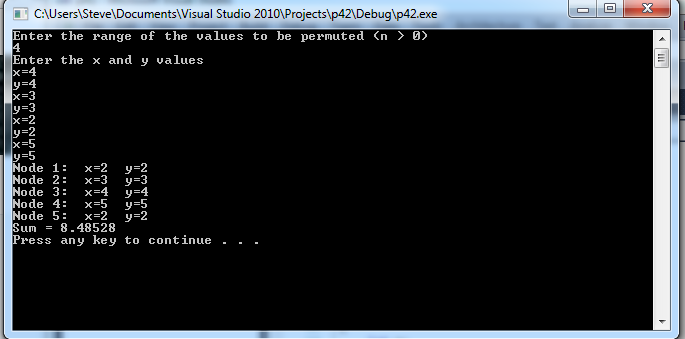
**Efficient Optimization 1:**

****

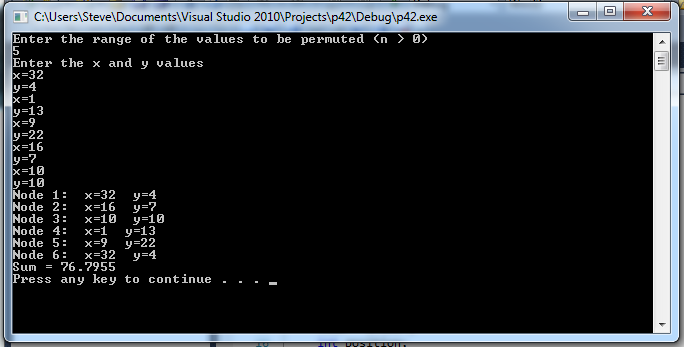
**Efficient Optimization 2:**

****

**INNA 1:**

****

**INNA 2:**

****