FINAL PROJECT

BANA 7020 – Optimization

Dr. Leonardo Lozano Sanchez

FALL 19

Submitted by:
Ayush Rai
Leila Ouellaj
Puneet Bhatia

University of CINCINNATI

PROBLEM FORMULATION

Variables-

x – Array of x coordinates

y – Array of y coordinates

Euc – Euclidean Matrix

C_T – Cost per unit of Truck

C_D – Cost per unit of Drones

Primary_nodes: Number of Primary nodes (Integer)

Secondary_nodes: Number of Secondary nodes (Integer)

Positional – Array of positions (Integer)

Truck_travel_bin: Truck binary variable

Drones_travel_bin: Drones binary variable

LIMIT – Customers to be served + 2

Objective function –

Minimize

$\sum_i^P \sum_j^P Truck\_travel\_bin(i,j)$ * Euc(i,j)* C_T + $\sum_i^P \sum_j^P Drones\_travel\_bin(i,j)$ * Euc(i,j)* C_D

Constraints –

Origin constraints-

x(1):= 0

y(1):= 0

x(LIMIT):= 0

y(LIMIT):= 0

K Drones allowed

$\sum_{j=1}^{LIMIT} Truck\_travel\_bin(i,j) + \sum_{j=1}^{LIMIT} Drones\_travel\_bin(i,j)$ <= K+1 $\forall$ i $\in$ (2 to (LIMIT – 1))

Positional Constraints

Positional(1) = 1

Positional(22) = $\sum_{i=1}^{LIMIT} \sum_{j=1}^{LIMIT} Truck\_travel\_bin(i,j)$ + 1

Positional(j) >= Positional(i) + 1 - 200*(1-Truck_travel_bin(i,j)) $\forall$ i $\in$ (1 to LIMIT), j $\in$ (1 to LIMIT)

Balance constraint out - in = 0

$\sum_{j=1}^{LIMIT} Truck\_travel\_bin(i,j) - \sum_{j=1}^{LIMIT} Truck\_travel\_bin(j,i)$ = 0 $\forall$ i $\in$ (2 to (LIMIT – 1))

Nodes are greater than 0

Primary_nodes >= 0

Secondary_nodes >= 0

Euc(p,j) = round(sqrt((y(j)-y(p))^2 + (x(j)-x(p))^2)) $\forall$ p $\in$ (1 to LIMIT), j $\in$ (1 to LIMIT)

Limit constraints

Primary_nodes + Secondary_nodes =22

$\sum_{i=1}^{LIMIT}\sum_{j=1}^{LIMIT} Drones\_travel\_bin(i,j)$ = 2*Secondary_nodes

$\sum_{i=1}^{LIMIT}\sum_{j=1}^{LIMIT} Truck\_travel\_bin(i,j)$ = Primary_nodes -1

Constraint for removing subtours of drones

Drones_travel_bin(i,j) = Drones_travel_bin(j,i) $\forall$ i $\in$ (2 to (LIMIT – 1)) j $\in$ (2 to (LIMIT – 1))

Constraint for removing subtour of trucks

Truck_travel_bin(i,j) + Truck_travel_bin(j,i) <=1 $\forall$ i $\in$ (2 to (LIMIT – 1)) j $\in$ (2 to (LIMIT – 1))

Drones_travel_bin(i,j) <= $\sum_{a=1}^{LIMIT} Truck\_travel\_bin(a,i)$ + $\sum_{a=1}^{LIMIT} Truck\_travel\_bin(a,j)$ $\forall$ i $\in$ (2 to (LIMIT – 1)) j $\in$ (2 to (LIMIT – 1))

Origin constraints

$\sum_{i=1}^{LIMIT} Truck\_travel\_bin(1,i)$ = 1

$\sum_{i=1}^{LIMIT} Truck\_travel\_bin(i,1)$ = 0

$\sum_{i=1}^{LIMIT} Drones\_travel\_bin(1,i)$ = 0

$\sum_{i=1}^{LIMIT} Drones\_travel\_bin(i,1)$ = 0

Last node constraints

$\sum_{i=1}^{LIMIT} Truck\_travel\_bin(LIMIT,i)$ = 0

$\sum_{i=1}^{LIMIT} Truck\_travel\_bin(i,LIMIT)$ = 1

$\sum_{i=1}^{LIMIT} Drones\_travel\_bin(LIMIT,i)$ = 0

$\sum_{i=1}^{LIMIT} Drones\_travel\_bin(i,LIMIT)$ = 0

Truck_travel_bin(1,LIMIT) =0

Truck_travel_bin(LIMIT,1) =0

Removing arcs on itself

$\sum_{i=1}^{LIMIT} Truck\_travel\_bin(i,i)$ = 0

$\sum_{i=1}^{LIMIT} Drones\_travel\_bin(i,i)$ = 0

All nodes should be having incoming nodes - except origin

$\sum_{j=1}^{LIMIT} Truck\_travel\_bin(j,i)$ - $\sum_{j=1}^{LIMIT} Drones\_travel\_bin(j,i)$ >= 1 $\forall$ i $\in$ (2 to LIMIT)

All nodes should be having outgoing nodes - except destination

$\sum_{i=1}^{LIMIT} Truck\_travel\_bin(j,i)$ - $\sum_{i=1}^{LIMIT} Drones\_travel\_bin(j,i)$ >= 1 ∀ j ∈ (1 to (LIMIT-1))

Constraint for not being secondary and primary at the same time

Truck_travel_bin(i,j) + Drones_travel_bin(i,j) <=1 ∀ i ∈ (1 to LIMIT) j ∈ (1 to LIMIT)

Constraint for traveling back to j if drone travels from j

Drones_travel_bin(i,j) = Drones_travel_bin(j,i) ∀ i ∈ (2 to (LIMIT – 1)) j ∈ (2 to (LIMIT – 1))

XPRESS CODE

```
model final_project
uses "mmxprs"; !gain access to the Xpress-Optimizer solver
uses "mmive" !gain access to graphical capabilities

 parameters
 LIMIT=22
 end-parameters
 setparam("XPRS_MAXTIME", 7200)


 declarations
P= 1..22
graph : integer
x: array(P) of real !declaring random X coordinates
y: array(P) of real !declaring random Y coordinates
Euc: array(P,P) of integer !declaring Euclidean matrix
Truck_travel_bin: array(P,P) of mpvar !declaring truck binary variable
Drones_travel_bin: array(P,P) of mpvar !declaring drones binary variable
Primary_nodes: mpvar !primary nodes no
Secondary_nodes: mpvar !secondary nodes no
Positional: array(P) of mpvar
end-declarations

 writeln("RANDOM numbers(", LIMIT," of them) between 1 and 20 :")

setrandseed(4)
cloud:=IVEaddplot("DISTRIBUTION OF GENERATED X and Y", IVE_BLUE)
 forall(p in 2..LIMIT) do
   x(p):= 100*random
   y(p):= 100*random
 end-do


 ! origin constraints
x(1):= 0
y(1):= 0
x(22):= 0
y(22):= 0

!type_constraints
forall( p in 1..LIMIT) do
Positional(p) is_integer
end-do

!K drones allowed constraints
forall(i in 2..21) sum(j in 1..LIMIT) Truck_travel_bin(i,j) + sum(j in 1..LIMIT) Drones_travel_bin(i,j)  <= 6

!Positional Constraints
Positional(1) = 1
Positional(22) = sum(i in 1..LIMIT, j in 1..LIMIT) Truck_travel_bin(i,j)+1

forall( i in 1..LIMIT, j in 1..LIMIT)
Positional(j) >= Positional(i) + 1 - 200*(1-Truck_travel_bin(i,j))

!Balance constraint out - in = 0
forall(i in 2..21) sum(j in 1..LIMIT) Truck_travel_bin(i,j)-sum(j in 1..LIMIT) Truck_travel_bin(j,i)=0

forall(p in 1..LIMIT,j in 1..LIMIT) do
  Truck_travel_bin(p,j) is_binary
  Drones_travel_bin(p,j) is_binary
end-do
```

```
Primary_nodes is_integer
Secondary_nodes is_integer
Primary_nodes >=0
Secondary_nodes >=0

 forall(p in 1..LIMIT,j in 1..LIMIT) do
    Euc(p,j):= round(sqrt((y(j)-y(p))^2 + (x(j)-x(p))^2))
 end-do

!Limit constraints

Primary_nodes + Secondary_nodes =22

!Primary_nodes=(Secondary_nodes/2)+2

!Primary_nodes*2=Secondary_nodes

sum(i in 1..LIMIT, j in 1..LIMIT) Drones_travel_bin(i,j) = 2*Secondary_nodes

sum(i in 1..LIMIT, j in 1..LIMIT) Truck_travel_bin(i,j) = Primary_nodes -1

! constraint for removing subtours of drones
forall(i in 2..21, j in 2..21)
   Drones_travel_bin(i,j) = Drones_travel_bin(j,i)

!Constraint for removing subtour of trucks
forall(i in 2..21, j in 2..21)
   Truck_travel_bin(i,j) + Truck_travel_bin(j,i) <=1
```

```
orall (i in 2..21, j in 2..21) Drones_travel_bin(i,j) <= sum(a in 1..22) Truck_travel_bin(a,i)+ sum(a in 1..22) Truck_travel_bin(a,j)
if S(i,j) exists no S(j,i) should exist
forall(i in 2..11) sum(j in 2..11) Drones_travel_bin(j,i) + sum(j in 2..11) Drones_travel_bin(i,j)>= 2- sum(j in 2..11)Truck_travel_bin(i,j)

 CONSTRAINT FOR Kth

! origin constraints - part 2
sum(i in 1..LIMIT) Truck_travel_bin(1,i)= 1
sum(i in 1..LIMIT) Drones_travel_bin(1,i)= 0

sum(i in 1..LIMIT) Truck_travel_bin(i,1)= 0
sum(i in 1..LIMIT) Drones_travel_bin(i,1)= 0

!last node constraints
sum(i in 1..LIMIT) Truck_travel_bin(22,i)= 0
sum(i in 1..LIMIT) Drones_travel_bin(22,i)= 0

sum(i in 1..LIMIT) Truck_travel_bin(i,22)= 1
sum(i in 1..LIMIT) Drones_travel_bin(i,22)= 0

Truck_travel_bin(1,22) =0
Truck_travel_bin(22,1) =0

!removing_arcs_on_itself
sum(i in 1..LIMIT) Truck_travel_bin(i,i) = 0
sum(i in 1..LIMIT) Drones_travel_bin(i,i) = 0

!all nodes should be have incoming nodes - except origin
forall(i in 2..LIMIT) sum(j in 1..LIMIT) Truck_travel_bin(j,i) + sum(j in 1..LIMIT) Drones_travel_bin(j,i) >=1
```

```
 !all nodes should have outgoing nodes - except destination
  forall(j in 1..21) sum(i in 1..LIMIT) Truck_travel_bin(j,i) + sum(i in 1..LIMIT) Drones_travel_bin(j,i) >=1


 ! constraint for not being secondary and primary at the same time
 forall(i in 1..LIMIT, j in 1..LIMIT)
    Truck_travel_bin(i,j) + Drones_travel_bin(i,j) <=1

! constraint for traveling back to j if drone travels from j
 forall(i in 2..21, j in 2..21)
    Drones_travel_bin(i,j) = Drones_travel_bin(j,i)


obj:=sum(i in P)sum(j in P)(Truck_travel_bin(i,j))*Euc(i,j)*20+sum(i in P)sum(j in P)(Drones_travel_bin(i,j))*Euc(i,j)*1
minimize(obj)


 writeln("Printing random numbers generated below:")
 forall(p in 1..LIMIT) do
 writeln("Number ",p," in Array X is :",x(p)," and the Number ",p," in Array  Y is:", y(p))
 end-do


 IVEzoom(0, -0, 10, 10)
 writeln(" Generating the Plot for co-ordinates below")
 graph:=IVEaddplot("Y(X)",IVE_RED)        !Create a graph
 graph2:=IVEaddplot("Y(X)",IVE_GREEN)

 forall(p in 1..LIMIT) do
 IVEdrawpoint(graph, x(p), y(p))
 IVEdrawlabel(graph, x(p), y(p), ""+p)
 end-do
```

```
forall(i in 1..LIMIT,j in 1..LIMIT |getsol(Drones_travel_bin(i,j)) > 0) do
IVEdrawline(graph2, x(i), y(i),x(j), y(j))
end-do

 forall(i in 1..LIMIT,j in 1..LIMIT |getsol(Truck_travel_bin(i,j)) > 0) do
IVEdrawline(graph, x(i), y(i),x(j), y(j))
end-do

forall(i in 1..LIMIT,j in 1..LIMIT |getsol(Truck_travel_bin(i,j)) > 0 ) do
    writeln("Send Truck from ",i," to ", j," : ",getsol(Truck_travel_bin(i,j)))
end-do

forall(i in 1..LIMIT,j in 1..LIMIT |getsol(Drones_travel_bin(i,j)) > 0 ) do
    writeln("Send Drones from ",i," to ", j," : ",getsol(Drones_travel_bin(i,j)))
    end-do

end-model
```

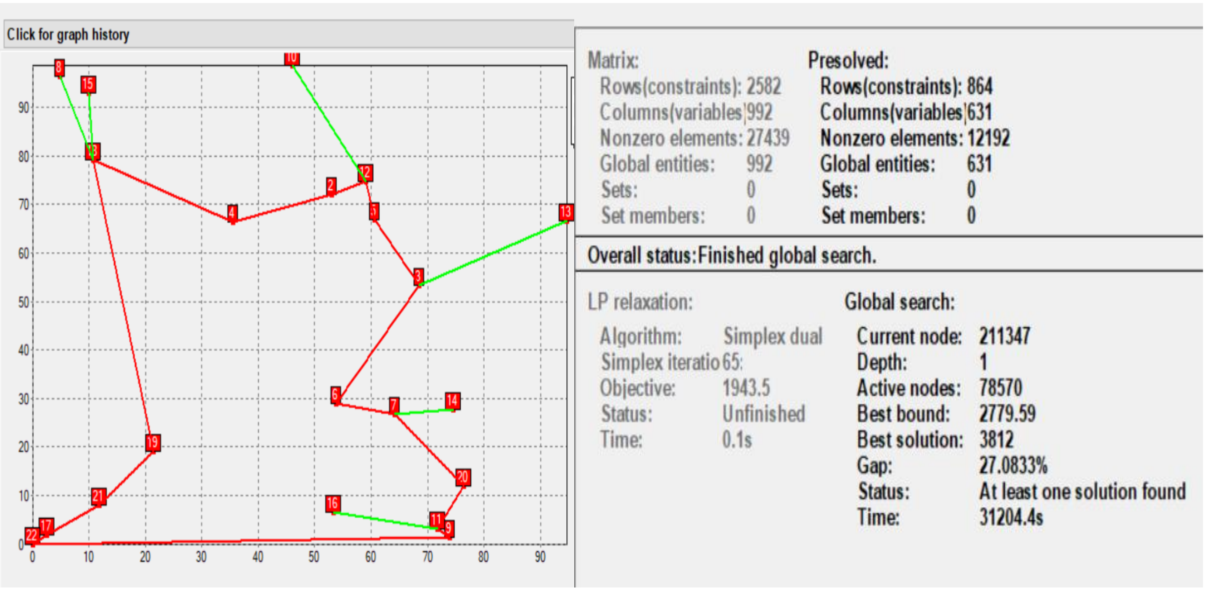
NOTE: We have also used a code with manually generated random points

SOLUTION

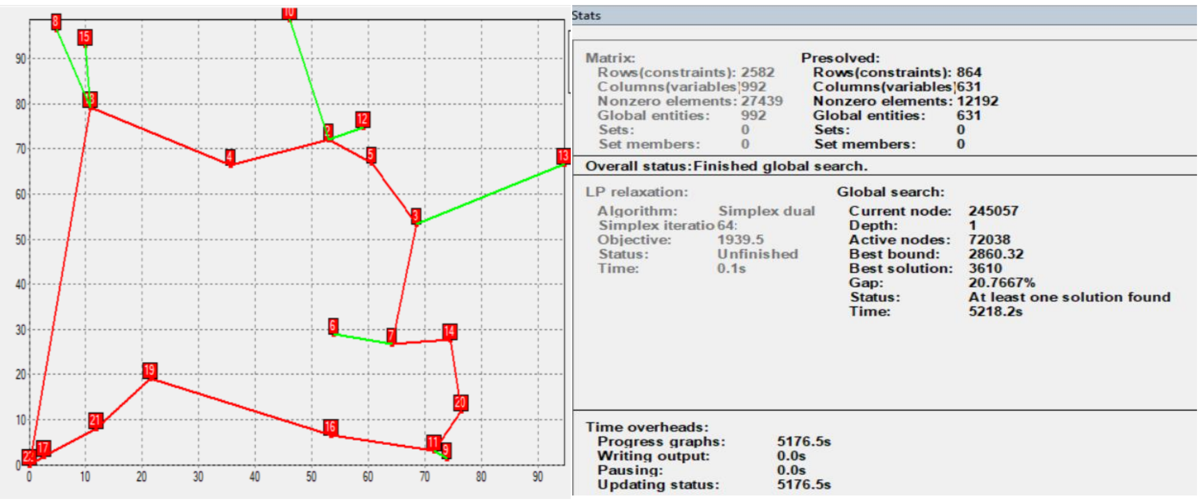
SEED = 4

COST PER UNIT DISTANCE OF TRUCK – 10 COST PER UNIT DISTANCE OF DRONES - 3

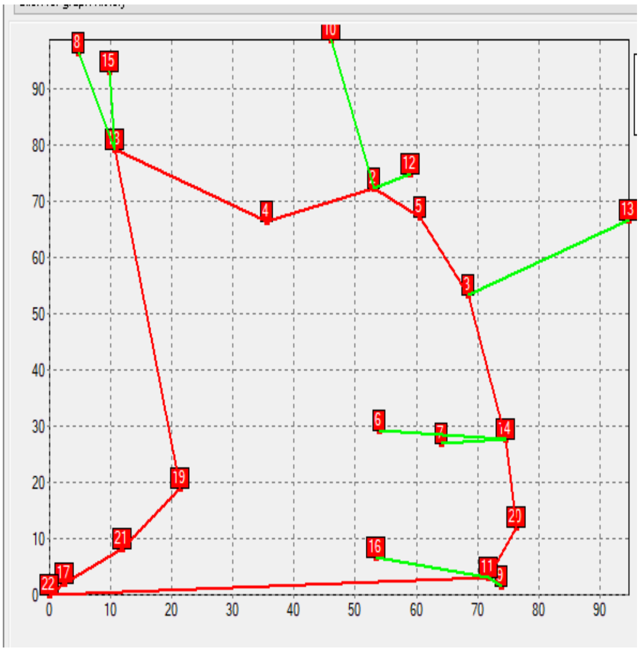
K =2 OBJECTIVE FUNCTION = 3812

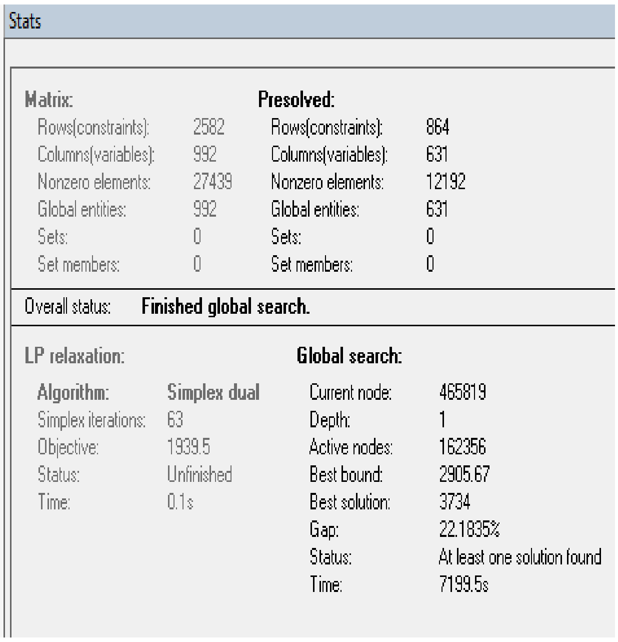


K =3 OBJECTIVE FUNCTION = 3610

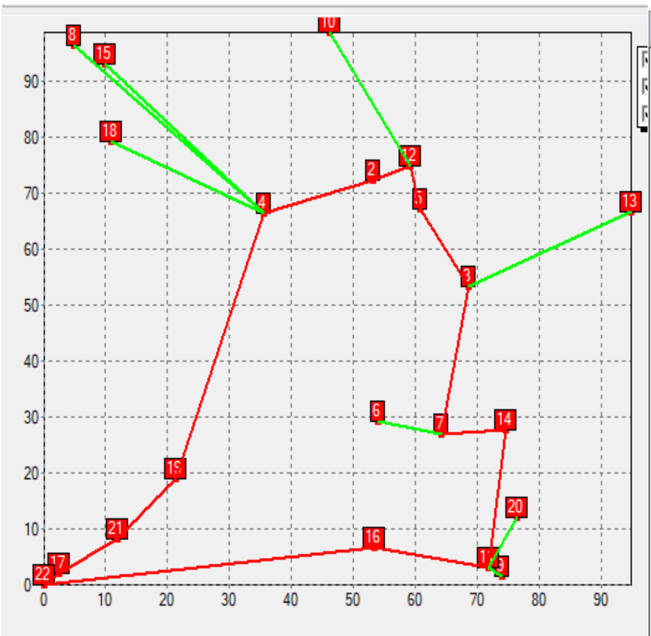


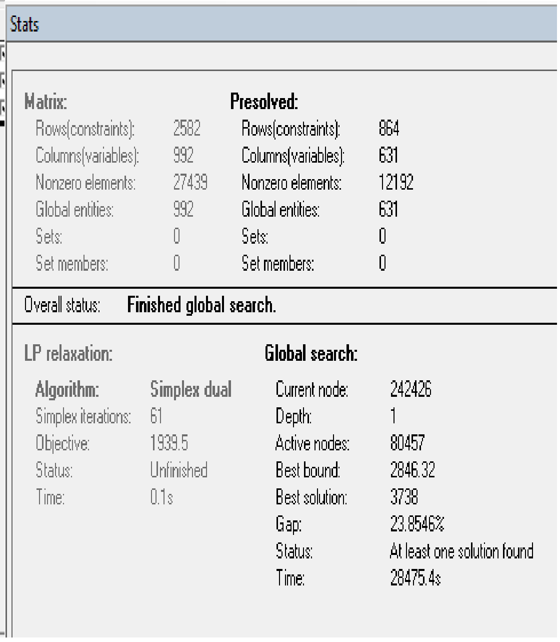
K =4   OBJECTIVE FUNCTION = 3734





K =5   OBJECTIVE FUNCTION = 3738

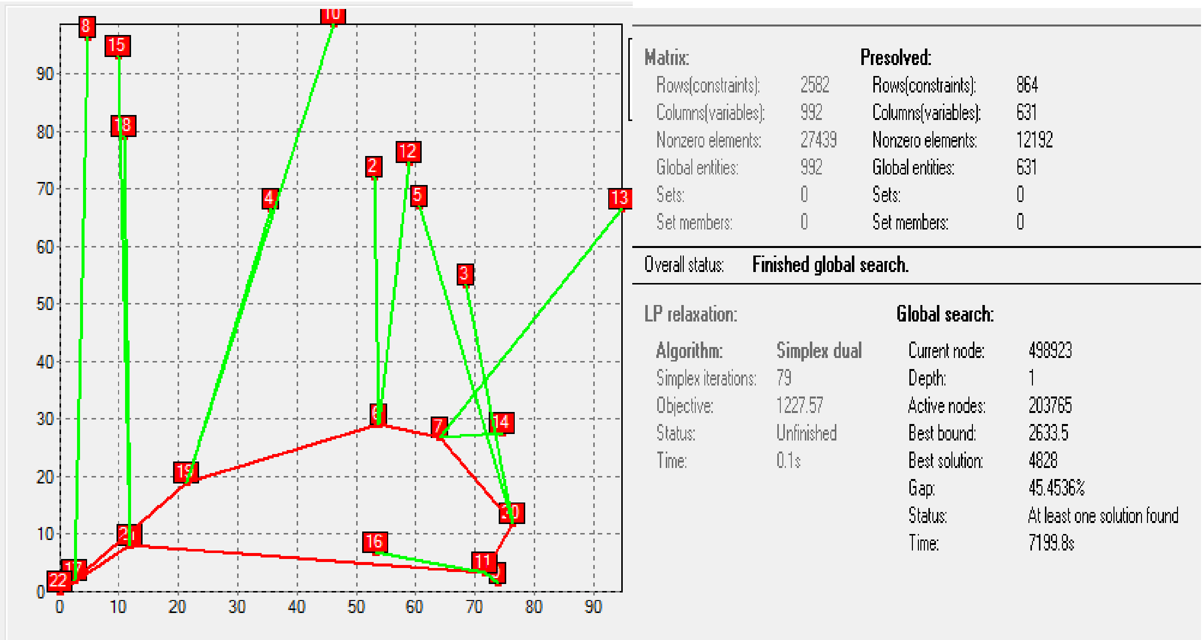




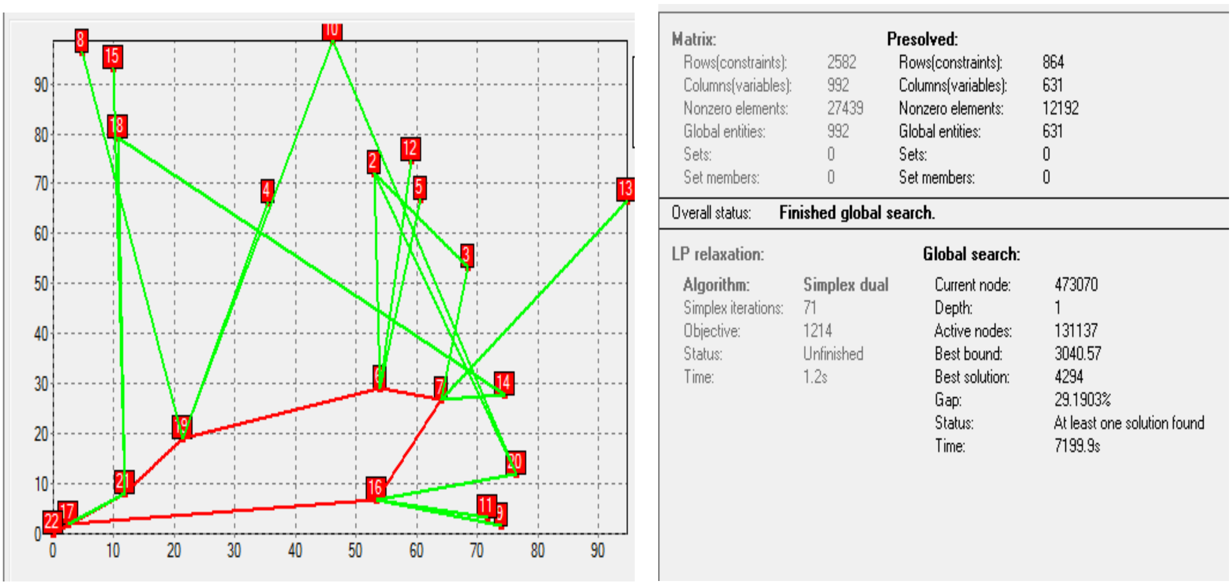
SENSITIVITY ANALYSIS

COST PER UNIT DISTANCE OF TRUCK – 20 COST PER UNIT DISTANCE OF DRONES – 1

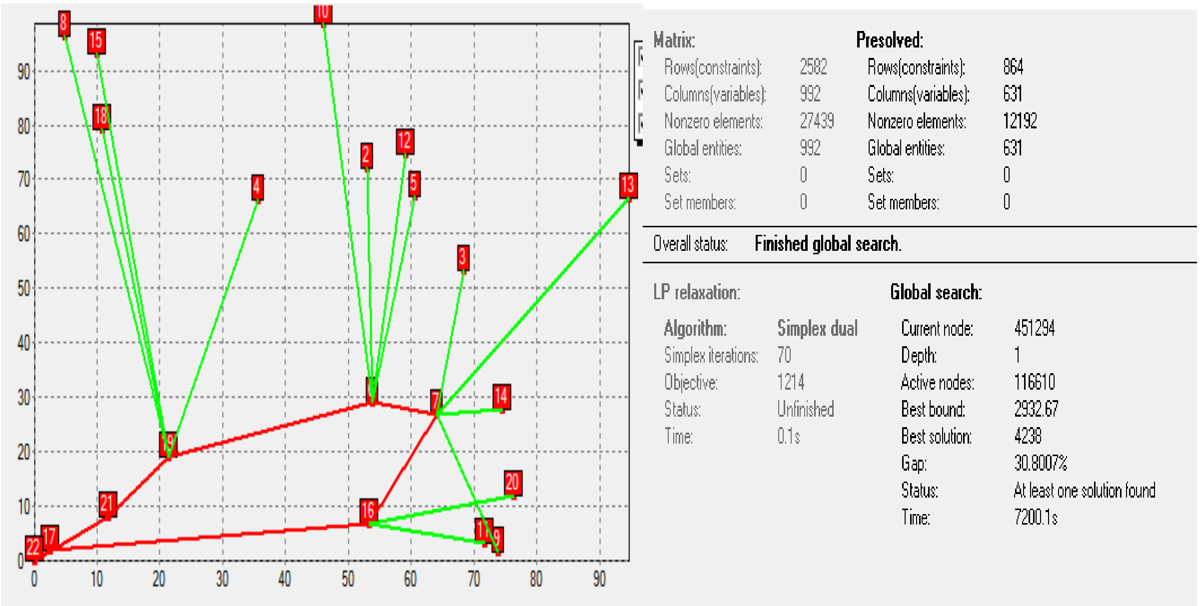
K =2   OBJECTIVE FUNCTION = 4828



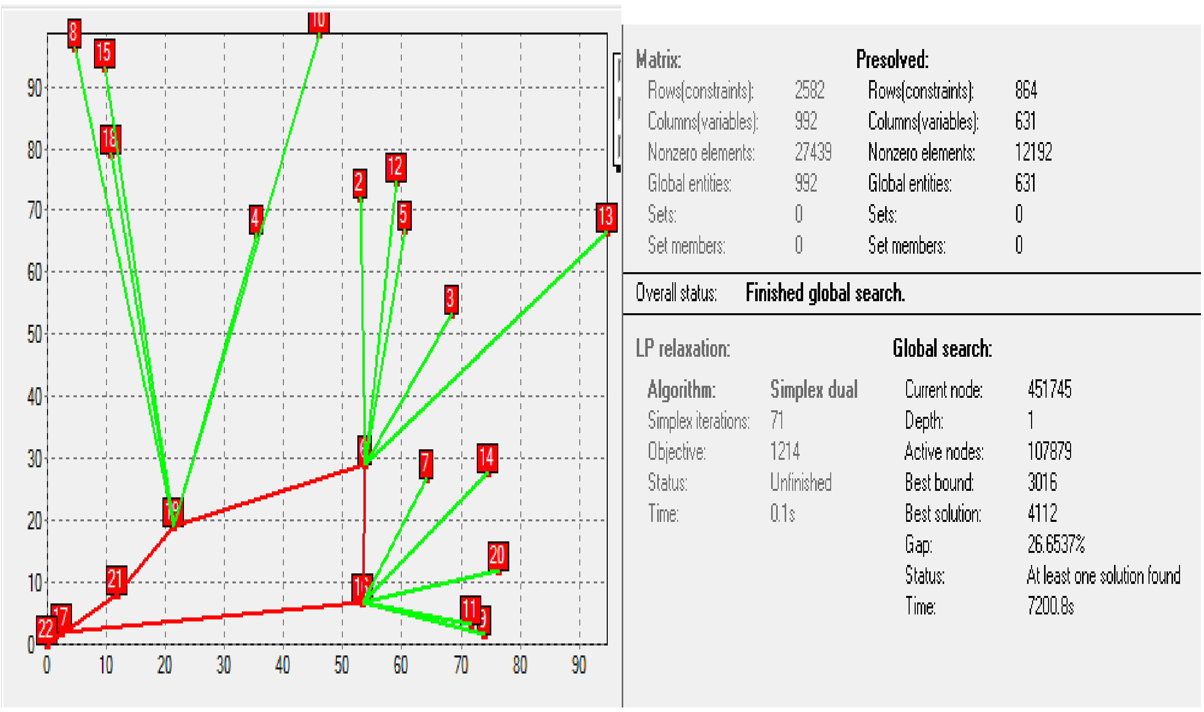
K =3   OBJECTIVE FUNCTION = 4294



K =4   OBJECTIVE FUNCTION = 4238



K =5   OBJECTIVE FUNCTION = 4112

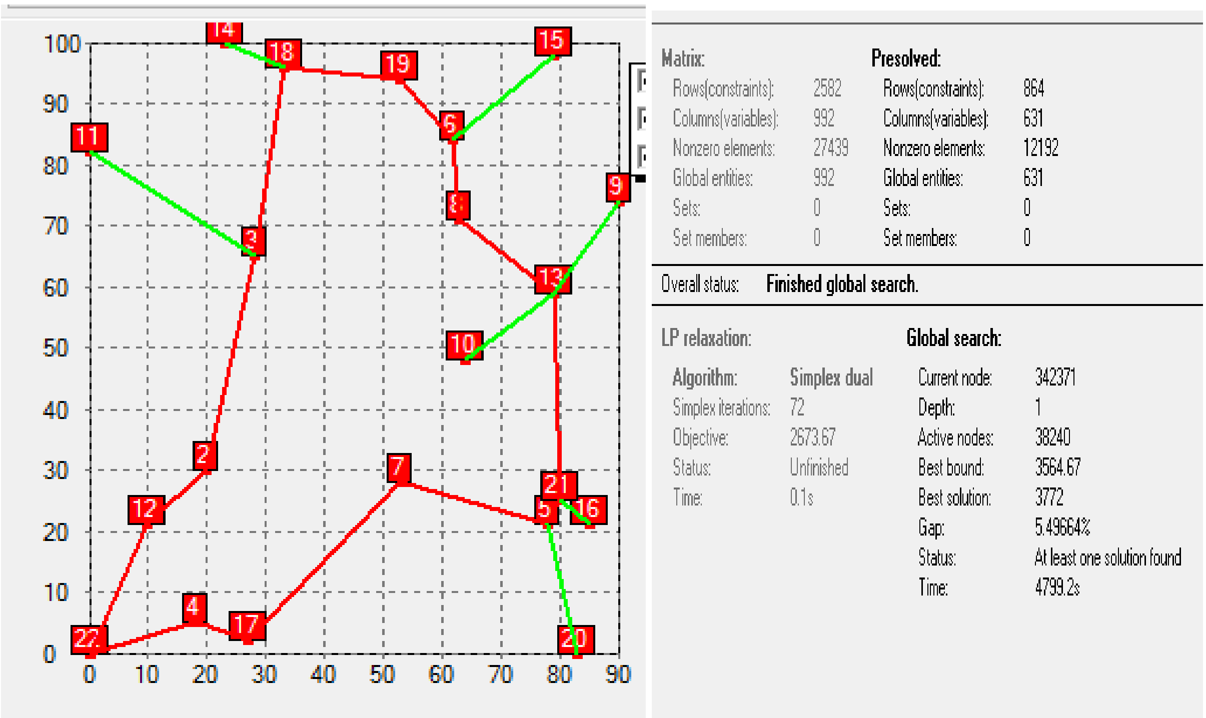


CONCLUSION: We observe by increasing the ratio of Unit cost per distance of Truck to that of Drones, the solution optimizes the route by using more Drones for every value of K

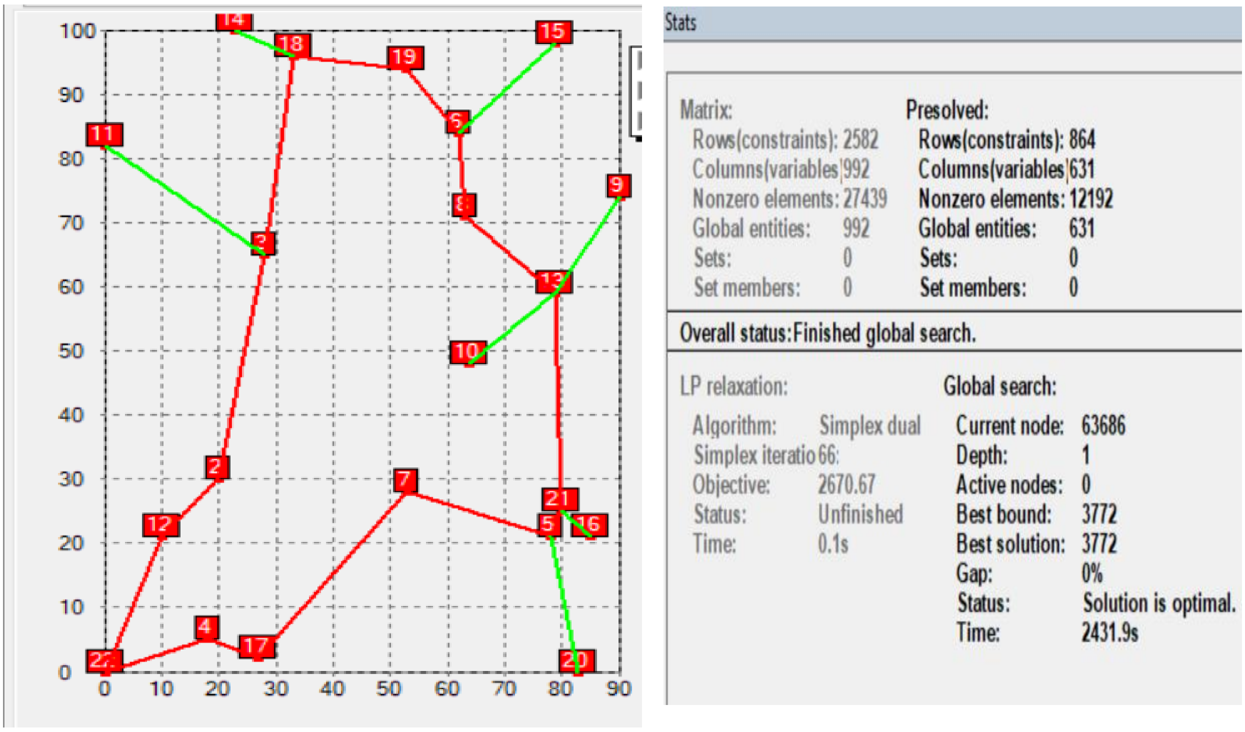
To better visualize the working of the model we have manually generated customer coordinates which are clustered near a single node (Customer location). Our result for these set of customer location is as follows-

COST PER UNIT DISTANCE OF TRUCK – 10 COST PER UNIT DISTANCE OF DRONES - 3

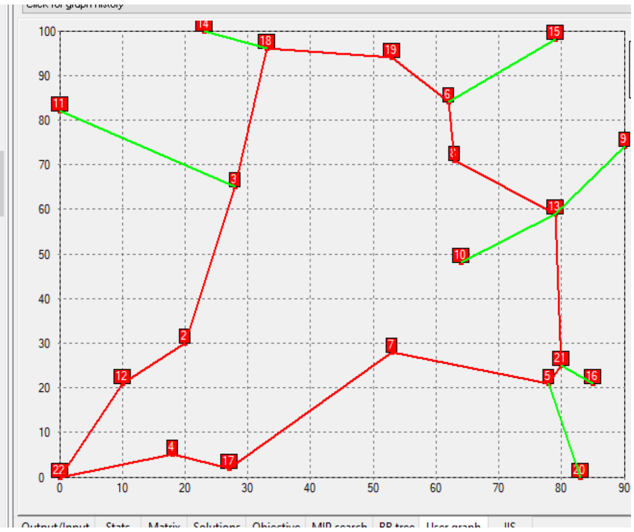
K =2   OBJECTIVE FUNCTION = 3772

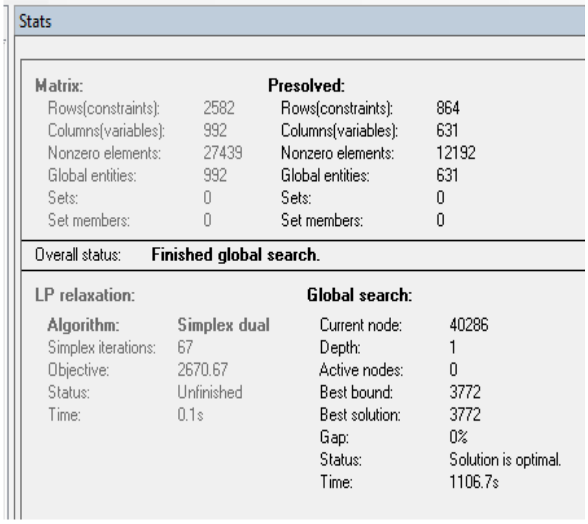


K =3   OBJECTIVE FUNCTION = 3772

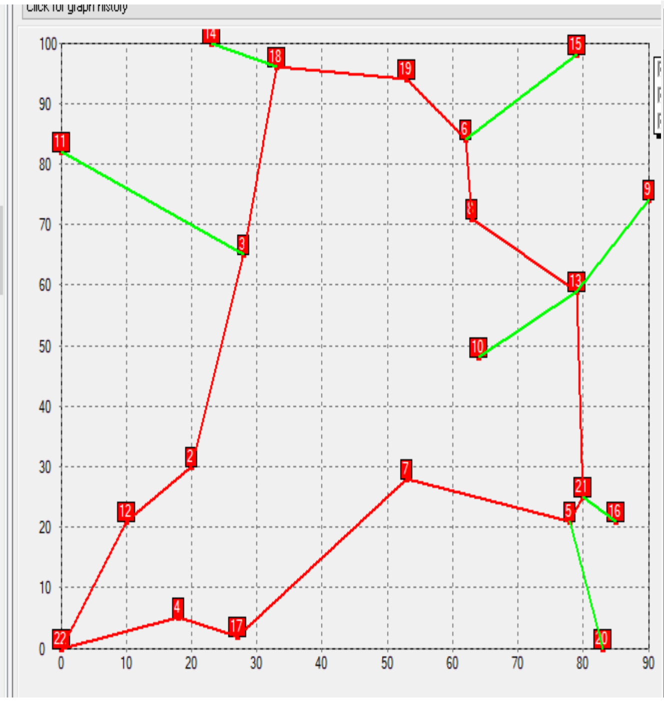


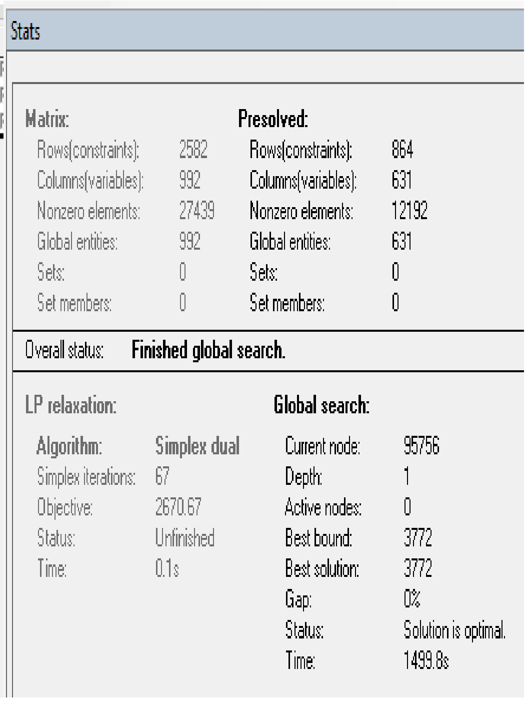
K=4   OBJECTIVE FUNCTION = 3772





K=5   OBJECTIVE FUNCTION = 3772



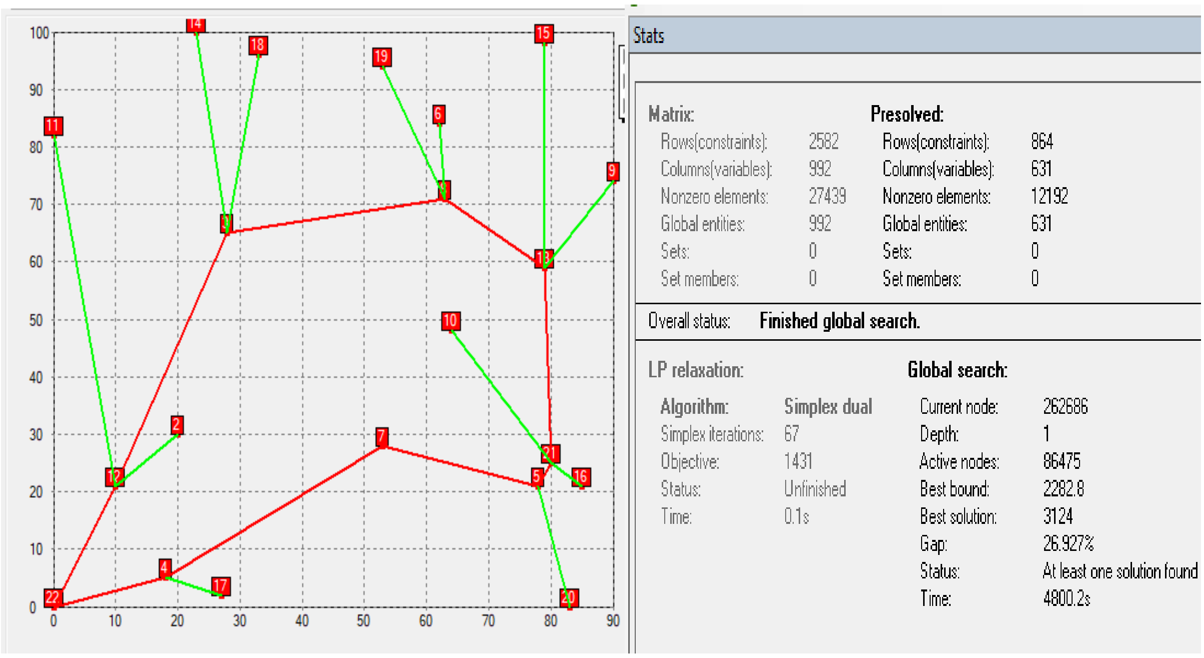


CONCLUSION: Here we observe increasing K does not decrease the objective function.

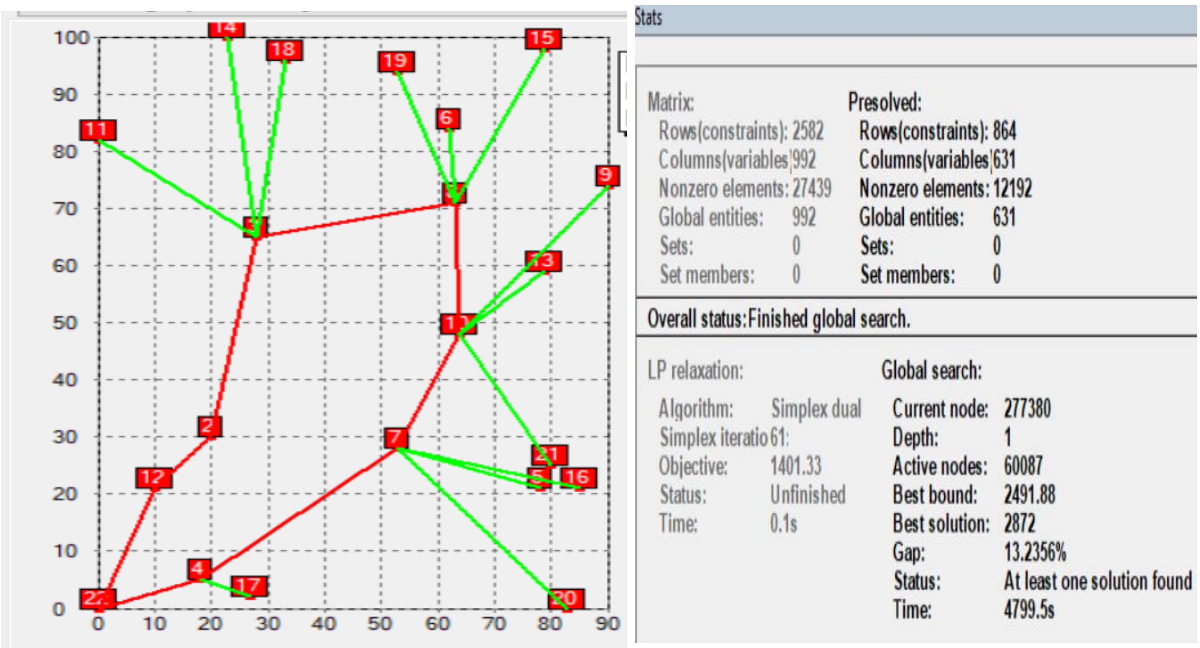
SENSITIVITY ANALYSIS

COST PER UNIT DISTANCE OF TRUCK – 10 COST PER UNIT DISTANCE OF DRONES – 1

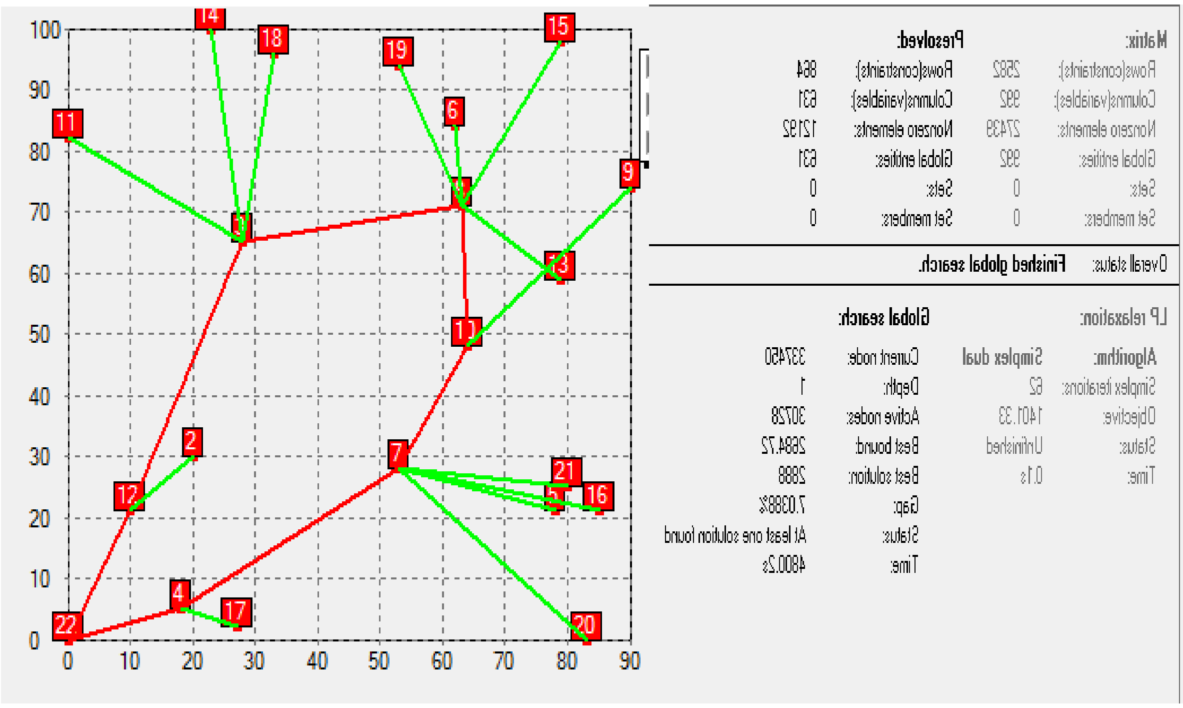
K=2 OBJECTIVE FUNCTION = 3124



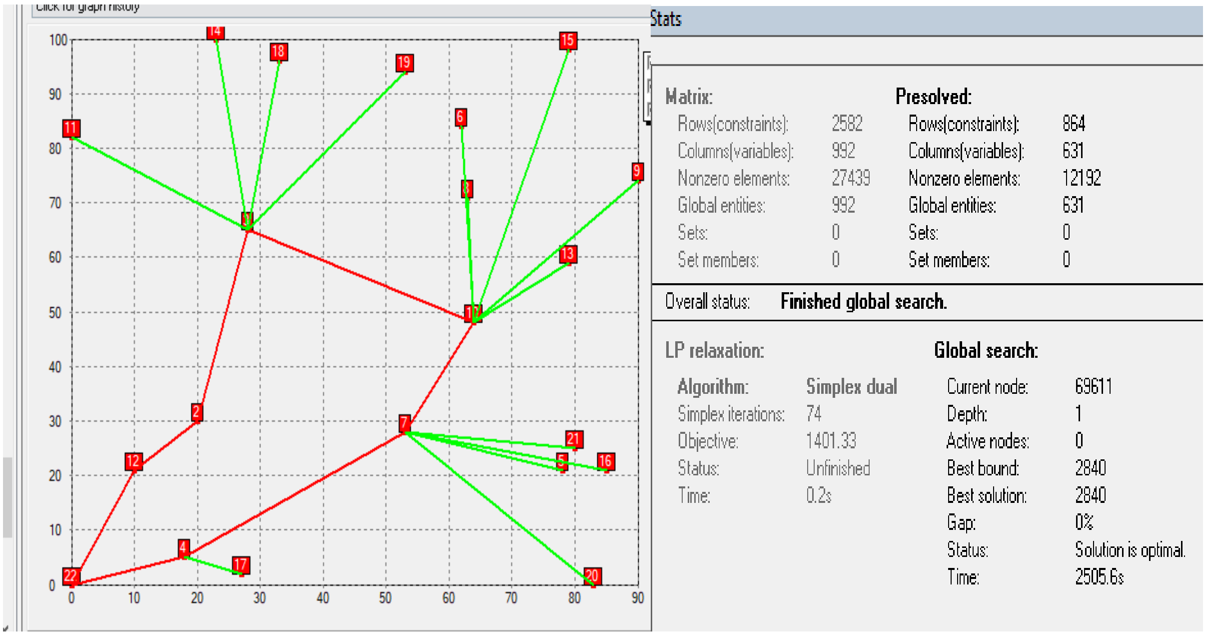
K=3 OBJECTIVE FUNCTION = 2872



K=4   OBJECTIVE FUNCTION = 2888



K=5   OBJECTIVE FUNCTION = 2840



CONCLUSION: Here we observe increasing K decreases the objective function, but not by a huge margin.