In The Name Of God

University of Tehran

Faculty of Engineering

Department of Surveying and Geomatics Engineering

Geospatial Information Systems (GIS) Division

Travelling Salesman Problem

Tabu Search

Simulated Annealing

Bee Colony

Ant Colony

Instructor : Dr.Samadzadegan

Teacher assistant : MC.s Hasani

Name :Mehran Ghandehary

Student Number : 810389010

Email : m.ghandhary@yahoo.com

Date Delivered : 18.12.2010

Apply Simulated Annealing and Tabu Search to Solve TSP

Farhad Samadzadegan, Ph.D
Geomatics Engineering - University of Tehran

Example of TSP

- There are *n* cities with symmetric distance matrix that indicates the cost of travel from each city to every other city.
- The goal is to find the shortest circular tour, visiting every city exactly once, so as to minimize the total travel cost, which includes the cost of traveling from the last city back to the first city'.

Example of TSP

- Consider 6 Indian cities –

  Mumbai, Nagpur , Calcutta, Delhi , Bangalore and Chennai and assign a number to each.

Mumbai	→	1
Nagpur	→	2
Calcutta	→	3
Delhi	→	4
Bangalore	→	5
Chennai	→	6

Example of TSP

- Cost matrix for six city example.

	1	2	3	4	5	6
1	0	863	1987	1407	998	1369
2	863	0	1124	1012	1049	1083
3	1987	1124	0	1461	1881	1676
4	1407	1012	1461	0	2061	2095
5	998	1049	1881	2061	0	331
6	1369	1083	1676	2095	331	0

Problem Representation

- Thus a path would be represented as a sequence of integers from 1 to 6. It avoids unfeasible solution.

- The path [1 2 3 4 5 6] represents a path from Mumbai to Nagpur, Nagpur to Calcutta, Calcutta to Delhi, Delhi to Bangalore, Bangalore to Chennai, and finally from Chennai to Mumbai.

Fitness Function

- The fitness function will be the total cost of the tour represented by each chromosome.

- This can be calculated as the sum of the distances traversed in each travel segment.

Minimum Fitness Function Value, The Fitter The Solution

Tabu Search For TSP

- Steps to solve problem with Tabu search:
  - Problem representation and encoding
  - Fitness function
  - Neighbor structure
  - Setting algorithm parameter, such as Tabu size, Tabu tenure,…
  - Start with initial solution and evaluate it
  - Generate neighbors and select best one

Tabu Search For TSP

- Steps to solve problem with Tabu search (Cont,):
  - Check is current movement in Tabu list:
    - If it is not in Tabu list, move to new position,
    - otherwise check aspiration* criterion:
      - If it satisfies, relax Tabu list and move to new position
      - Otherwise consider another neighbor
  - Update Tabu list
  - Continue until stopping criteria is satisfied

* Aspiration criteria can be improving best position till now

Tabu Search For TSP

Initialization

- Start with initial solution, such as

1	**2**	**3**	**4**	**5**	**6**

- Evaluate fitness function

F(Initial_tour)=7209

Tabu Search For TSP
Neighbors Structure

- Generate neighbors by swapping two node

Current Position	1	2	3	4	5	6
Neighbor	1	4	3	2	5	6

- So in *n* cities problem, there are $\frac{n(n-1)}{2}$ neighbors.
- In our example, there are 15 neighbors

Tabu Search For TSP
Neighbors Structure

- Evaluating of 15 neighbors in our example:

Movement	Fitness	Movement	Fitness	Movement	Fitness
1-2	7786	2-3	7884	3-5	8479
1-3	7462	2-4	6741	3-6	9970
1-4	7547	2-5	7804	4-5	9393
1-5	6690	2-6	8479	4-6	7462
1-6	8096	3-4	6917	5-6	6872

Best Neighbors!

Tabu Search For TSP
Movement and Update Tabu list

- By swapping first and fifth element, new position is generated:

5	**2**	**3**	**4**	**1**	**6**

- Updete Tabu list:

It means for 10 iterations Swapping 1 and 5 is forbidden.

At each iteration Tabu list elements are decrease

				10	

Tabu Search For TSP
CheckTabu list

- During iteration, after finding best neighbors, check is this movement in Tabu list or not:
  - If not, move to new position
  - Otherwise, verify aspiration criterion such as: improvement respect to best position till now
    - If satisfied, relax Tabu list and move
    - Otherwise consider another neighbors

Simulated Annealing For TSP

- Steps to solve problem with Simulated Annealing:
  - Problem representation and encoding
  - Fitness function
  - Neighbor structure
  - Setting algorithm parameter, such as initial temperature, control parameter,…
  - Start with initial solution and evaluate it
  - For specific number of iterations, generate a neighbor and compare it with current position:
    - If it makes current position better, then move
    - Else move by probability
  - Continue until stopping criteria is satisfied

Simulated Annealing For TSP
Initialization

- Start with initial solution, such as

1	**2**	**3**	**4**	**5**	**6**

- Evaluate fitness function

F(Initial_tour)=7209

- Initial tempature=2000
- $\alpha = 0.8$

Simulated Annealing For TSP
Neighbors Structure

- Generate neighbors randomly

Current Position	1	2	3	4	5	6
Neighbor	1	4	3	2	5	6

F(Current_Position)= 7209

F(Neighbor)= 6741

F(Neighbor)<F(Current_Position) -> Move to new position

Simulated Annealing For TSP
Neighbors Structure

- Generate neighbors randomly (another example)

Current Position	**1**	**2**	**3**	**4**	**5**	**6**
Neighbor	**1**	**2**	**3**	**5**	**4**	**6**

F(Current_Position)= 7209, F(Neighbor)= 9393

F(Neighbor)>F(Current_Position) -> Claculate probability

$$P = e^{-\frac{\Delta F}{T}} \longrightarrow P = e^{-\frac{9393-7209}{2000}} = 0.33$$

- If *rand<0.33* then accept solution

Simulated Annealing For TSP
Movement and update temperature

- Move to new position

1	4	3	2	5	6

- Update temperature by control parameter

$$T_{new} = \alpha T_{Old} = 0.8 \times 2000 = 1600$$

- Iterate until maximum iteration or reaching final temperature

Applying Bees Algorithm to Solve TSP

Example of TSP

- There are *n* cities with symmetric distance matrix that indicates the cost of travel from each city to every other city.
- The goal is to find the shortest circular tour, visiting every city exactly once, so as to minimize the total travel cost, which includes the cost of traveling from the last city back to the first city'.

Example of TSP

- Cost matrix for six city example.

	1	2	3	4	5	6
1	0	863	1987	1407	998	1369
2	863	0	1124	1012	1049	1083
3	1987	1124	0	1461	1881	1676
4	1407	1012	1461	0	2061	2095
5	998	1049	1881	2061	0	331
6	1369	1083	1676	2095	331	0

Problem Representation

- Thus a path would be represented as a sequence of integers from 1 to 6. It avoids unfeasible solution.

- The path [1 2 3 4 5 6] represents a path from Mumbai to Nagpur, Nagpur to Calcutta, Calcutta to Delhi, Delhi to Bangalore, Bangalore to Chennai, and finally from Chennai to Mumbai.

Bees Algorithm
Initialization

- Generate initial population randomly:

Population	Fitness
[4 5 2 3 6 1]	8686
[4 3 1 5 6 2]	6872
[6 4 5 1 3 2]	9348
[2 4 5 3 1 6]	9393
[4 2 6 5 1 3]	6872
[3 5 4 6 1 2]	9393
[6 3 5 4 1 2]	8971
[1 6 4 2 5 3]	9393
[2 3 4 1 5 6]	6404
[2 3 6 4 1 5]	8349

Bees Algorithm
Select best bees

- Sort bees according to their fitness function

Best Bees (m=5)

Population	Fitness
[2 3 4 1 5 6]	6404
[4 3 1 5 6 2]	6872
[4 2 6 5 1 3]	6872
[2 3 6 4 1 5]	8349
[4 5 2 3 6 1]	8686
[6 3 5 4 1 2]	8971
[6 4 5 1 3 2]	9348
[2 4 5 3 1 6]	9393
[3 5 4 6 1 2]	9393
[1 6 4 2 5 3]	9393

Bees Algorithm

Select best bees

- Select the best bees

	Population	Fitness
Elite Bees (e=3)	[2 3 4 1 5 6]	6404
	[4 3 1 5 6 2]	6872
	[4 2 6 5 1 3]	6872
Best Bees (m-e=2)	[2 3 6 4 1 5]	8349
	[4 5 2 3 6 1]	8686

Bees Algorithm

Recruit bees for elite sites

- Recruit bees for elite site. There are *Ne* bees for searching around each elite sites. (Ne=3)
- For example, 3 bees should search neighbors of first point. the best bee in this patch, replace first point
- Neighborhood size= swap two nodes

Elite Bee	Neighbors	Fitness
[2 3 4 1 5 6] → 6404	[2 4 3 1 5 6]	6872
	[5 3 4 1 2 6]	7026
	[2 3 6 1 5 4]	8240

Neighbor couldn't improve elite, so this point remain

Bees Algorithm

Recruit bees for other best sites

- Recruit bees for other sites: There are *Nb* bees for searching around each other best sites. (Nb=2)
- For example, 2 bees should search neighbors of point 5. the best bee in this patch, replace 5th point
- Neighborhood size= swap two nodes

Best Bees

[4 5 2 3 6 1]

8686

Neighbors

[4 1 2 3 6 5] → 7462

[5 4 2 3 6 1] → 8240

Fitness

The neighbor improve best bee, so it replace

Bees Algorithm

Scout bees-> Random search

- For remaining bee (n-m=10-5=5) explore search space randomly and replace bees which are not in the best bees

Population	Fitness
[5 2 6 4 1 3]	9502
[3 2 6 5 1 4]	6404
[2 5 6 4 3 1]	7786
[2 1 6 3 4 5]	8479
[3 5 4 2 1 6]	8862

Bees Algorithm
Iteration

- Continue with new population

Population	Fitness
[2 3 4 1 5 6]	6404
[4 3 1 5 6 2]	6872
[3 2 6 5 1 4]	6404
[2 3 4 6 1 5]	8349
[4 1 2 3 6 5]	7462
[5 2 6 4 1 3]	9502
[3 2 6 5 1 4]	6404
[2 5 6 4 3 1]	7786
[2 1 6 3 4 5]	8479
[3 5 4 2 1 6]	8862

Applying Ant Colony System to Solve Travelling Salesman Problem

www.insectpod.com

Apply ACS to Optimization Problem

- ACS steps toward solving optimization problem:
  - Problem representation by graph
  - setting parameters:
    - Number of ants
    - Initial pheromone intensity
    - Evaporation rate
    - Stopping criteria
  - Initialization
  - Fitness Function

www.insectpod.com

Apply ACS to Optimization Problem (Cont,)

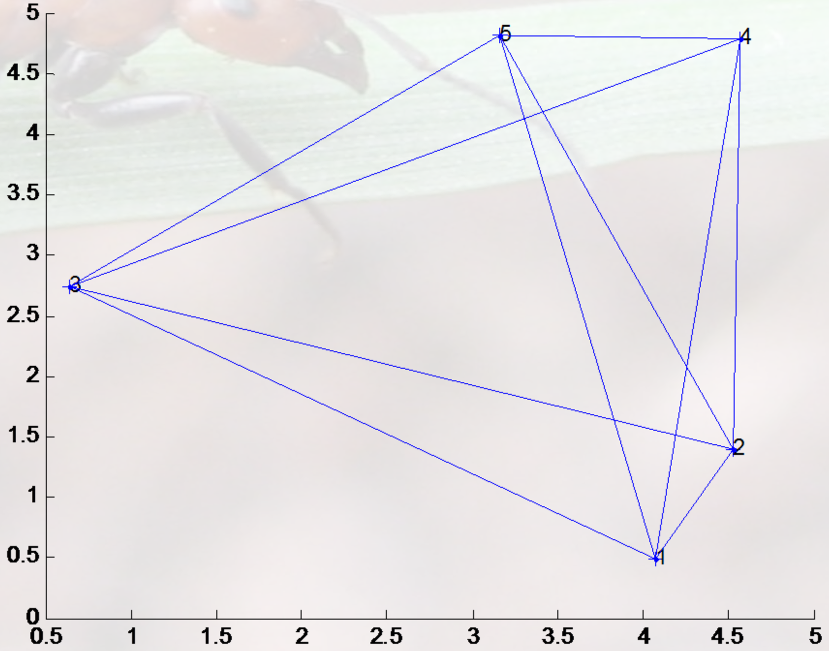
- ACS steps toward solving optimization problem:
  - Move on graph by probabilistic rules
  - Evaluate solution
  - Update pheromone intensity

www.insectpod.com

Problem Representation

- TSP represent by a graph which nodes are cities and links are connectivity between cities

NO	X	Y
1	4.0736	0.4877
2	4.5289	1.3924
3	0.6349	2.7344
4	4.5668	4.7875
5	3.1617	4.8244



www.insectpod.com

Parameters Setting

- Parameters should set experimentally for proposed problem:
  - Number of ants: 4
  - Initial pheromone: 1
  - Evaporation rate: 0.3
  - Number of best ants: 2
  - $\alpha=1$, $\beta=1$
  - $r0=0.5$
  - Maximum number of iteration: 100

www.insectpod.com

Fitness Function

- Fitness Function is computed by total distance of tour, e.g. [1 2 3 4 5]

$$Fitness = d_{12} + d_{23} + d_{34} + d_{45} + d_{51}$$

$$= 1.0129 + 4.1188 + 4.4357 + 1.4056 + 4.4316 = 15.4046$$

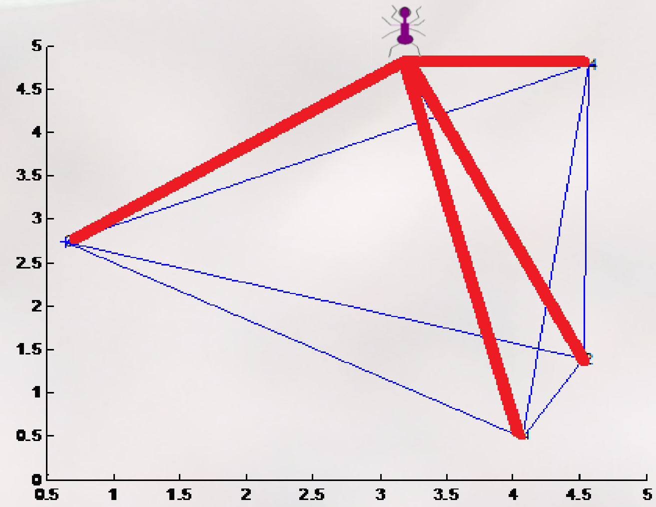
www.insectpod.com

Constructing a Solution

- Move in graph by probabilistic rules:
  - Consider ant randomly locate at 5th node
  - It can move to other 4 nodes

$$j = \begin{cases} max\{\tau \times \eta\} & if\ r \leq 0.5 \\ J & if\ r > 0.5 \end{cases}$$

- r is random, e.g. *r=0.73*



www.insectpod.com

Constructing a Solution

- Move in graph by probabilistic rules: compute probability of each link

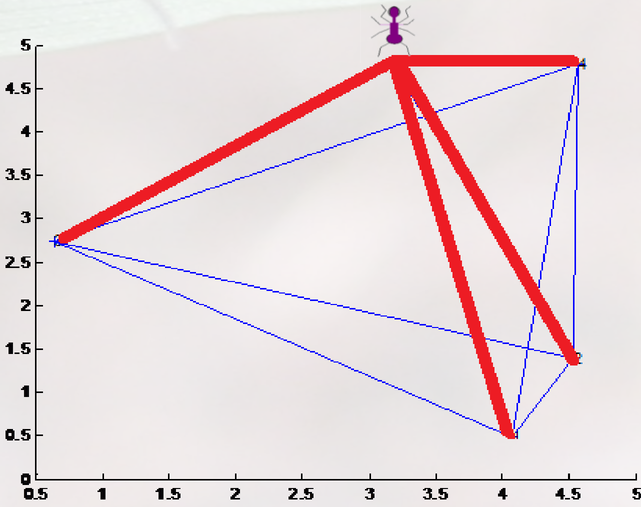
$$p_{i,J}^{k}(t) = \frac{\tau_{i,J}(t)\eta_{i,J}^{\beta}(t)}{\sum_{u \in N_i^k} \tau_{iu}(t)\eta_{iu}^{\beta}(t)}$$

$$P_{51} = \frac{1^1 \times (\frac{1}{4.4316})^1}{1^1 \times (\frac{1}{4.4316})^1 + 1^1 \times (\frac{1}{3.6942})^1 + 1^1 \times (\frac{1}{3.2792})^1 + 1^1 \times (\frac{1}{1.4056})^1} = \frac{\frac{1}{4.4316}}{\frac{1141497}{75.4589}} = 0.1492$$

$$P_{52} = \frac{1^1 \times (\frac{1}{3.6942})^1}{1^1 \times (\frac{1}{4.4316})^1 + 1^1 \times (\frac{1}{3.6942})^1 + 1^1 \times (\frac{1}{3.2792})^1 + 1^1 \times (\frac{1}{1.4056})^1} = \frac{\frac{1}{3.6942}}{\frac{1141497}{75.4589}} = 0.1789$$

$$P_{53} = \frac{1^1 \times (\frac{1}{3.2792})^1}{1^1 \times (\frac{1}{4.4316})^1 + 1^1 \times (\frac{1}{3.6942})^1 + 1^1 \times (\frac{1}{3.2792})^1 + 1^1 \times (\frac{1}{1.4056})^1} = \frac{\frac{1}{3.2792}}{\frac{1141497}{75.4589}} = 0.2016$$

$$P_{54} = \frac{1^1 \times (\frac{1}{1.4056})^1}{1^1 \times (\frac{1}{4.4316})^1 + 1^1 \times (\frac{1}{3.6942})^1 + 1^1 \times (\frac{1}{3.2792})^1 + 1^1 \times (\frac{1}{1.4056})^1} = \frac{\frac{1}{1.4056}}{\frac{1141497}{75.4589}} = 0.4703$$



www.insectpod.com

Constructing a Solution

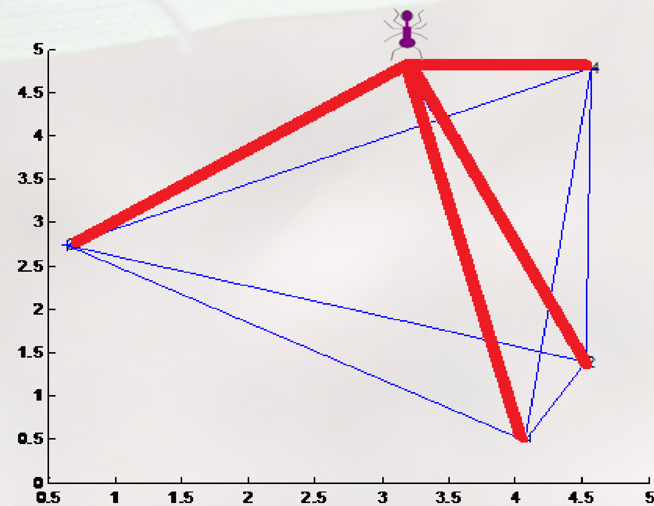
- Calculate Cumulative probability of presented path:

Cumulative probability

0.1492	**0.1789**	**0.2016**	**0.4703**

0.1492	**0.3281**	**0.5297**	**1**

- Generate random: 0.61
- So ant move to 4^{th} node



www.insectpod.com

Constructing a Solution

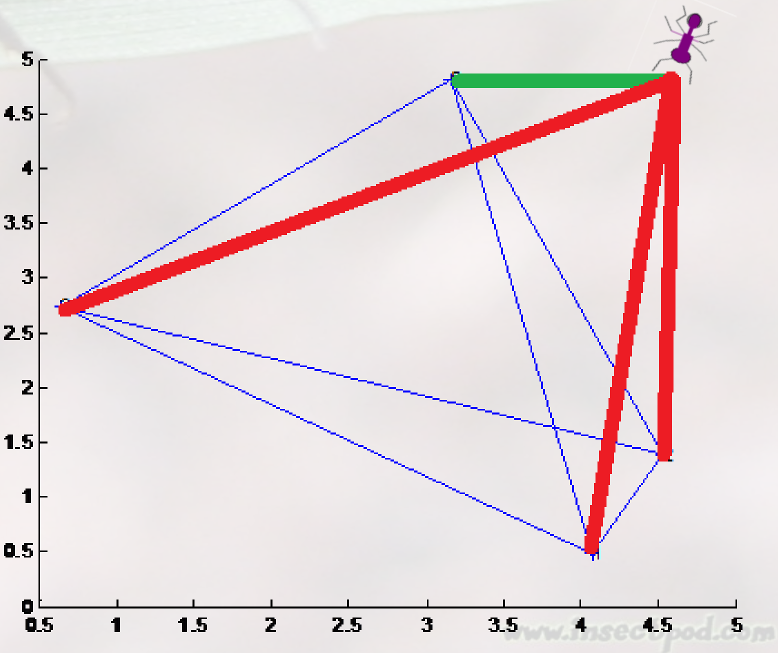
- From 4th node, 3 acceptable links are available
- 5th and 4th nodes are in Tabu list

$$j = \begin{cases} max\{\ \tau_{4i} \times \eta_{4i}\ \} & if\ r \leq 0.5 \\ J & if\ r > 0.5 \end{cases}$$

- *r=0.3*

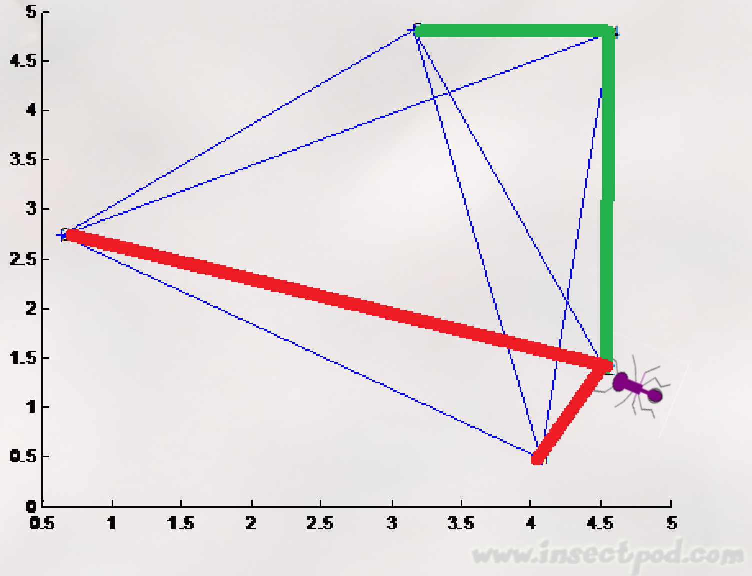
$$\begin{cases} \tau_{41} \times \dfrac{1}{d_{41}} = 1 \times \dfrac{1}{4.328} = 0.2311 \\ \tau_{42} \times \dfrac{1}{d_{42}} = 1 \times \dfrac{1}{3.3953} = 0.2945 \\ \tau_{43} \times \dfrac{1}{d_{43}} = 1 \times \dfrac{1}{4.4357} = 0.2254 \end{cases}$$

Max



Constructing a Solution

- From 2th node, 2 acceptable links are available
- 5th and 4th and 2nd nodes are in Tabu list
- Similar to previous nodes ant move on graph and complete its tour



All ants complete their solution

- Suppose in our experiment 4 ants generate their solution as following:

#Ant	Solution	Fitness
1	[5 4 2 3 1]	17.4588
2	[2 1 4 5 3]	13.7198
3	[3 2 4 1 5]	14.1443
4	[4 3 1 5 2]	20.0644

www.insectpod.com

Pheromone Updating

- Select 2 best ants to update pheromone:

#Ant	Solution	Fitness	
1	[5 4 2 3 1]	17.4588	
2	[2 1 4 5 3]	13.7198	← Best 1
3	[3 2 4 1 5]	14.1443	← Best 2
4	[4 3 1 5 2]	20.0644	

www.insectpod.com

Pheromone Updating

- Update pheromone by following equation:

$$\tau_{ij}(t+1)=(1-0.3)\tau_{ij}(t)+\sum_{l=1}^{2}\Delta\tau^{l}_{ij}$$

$$\Delta\tau_{ij}^{l}=\frac{1}{D_l}$$

- First, 70% current pheromone is evaporated, then it intensify by best ants

$$\begin{cases}\Delta\tau^{1}=\dfrac{1}{13.7198}=0.0729\\[2ex]\Delta\tau^{2}=\dfrac{1}{14.1443}=0.0707\end{cases}$$

www.insectpod.com

Pheromone Updating

1	1	1	1	1
	1	1	1	1
		1	1	1
			1	1
				1

evaporation

0.7	0.7	0.7	0.7	0.7
	0.7	0.7	0.7	0.7
		0.7	0.7	0.7
			0.7	0.7
				0.7

Intensify by 1st best ant
[2 1 4 5 3]

0.7	0.7729	0.7	0.7729	0.7
	0.7	0.7	0.7729	0.7
		0.7	0.7	0.7729
			0.7	0.7729
				0.7

Intensify by 2nd best ant
[3 2 4 1 5]

0.7	0.7729	0.7	0.8436	0.7707
	0.7	0.7	0.8436	0.7707
		0.7	0.7	0.8436
			0.7	0.7729
				0.7

www.insectpod.com

Iteration

- Start with new pheromone intensity
- Ant randomly locate at each city
- Ants complete their tour by probabilistic movement
- Evaluation of their solution
- Select best ant
- Pheromone updating
- Iteration until termination criterion

www.insectpod.com