Swarm Intelligence

Traveling Salesman Problem and Ant System

Leonardo Bezerra and Leslie Perez Caceres

IRIDIA – Université Libre de Bruxelles (ULB)
Bruxelles, Belgium
lperez@iridia.ulb.ac.be
lbezerra@iridia.ul.ac.be

Outline

1. Travelling salesman problem
   - Problem definition
   - Examples
2. Ant System Algorithm
   - Description
   - Applied to TSP
3. Practical exercise

Traveling Salesman Problem
Informal definition

- Given a set of customer cities, a salesman from his home town needs to find a shortest tour that takes him through all customers just once and then back home.

Traveling Salesman Problem (TSP)

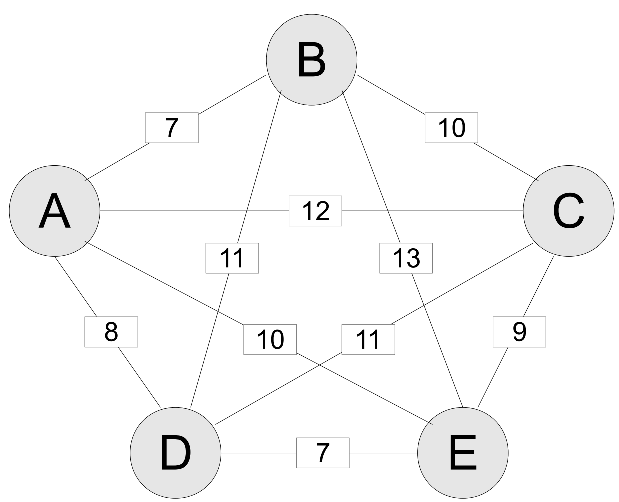
Main reasons for choosing the TSP:

- It is a classical **combinatorial optimization problem**.
- It is **NP hard**.
- It is the problem to which the Ant System algorithm was first applied.
- Often used to test new algorithms and variants.

Traveling Salesman Problem
Formal Definition

The TSP can be modeled as a Graph ***G(N,A)*** where:

- ***N*** is the set of nodes representing the cities
- ***A*** is the set of arcs
- Each arc is assign a cost value (length) ***d***
  - d_{ij} is the arc cost, or the length from city *i* to city *j*



Traveling Salesman Problem
Formal definition

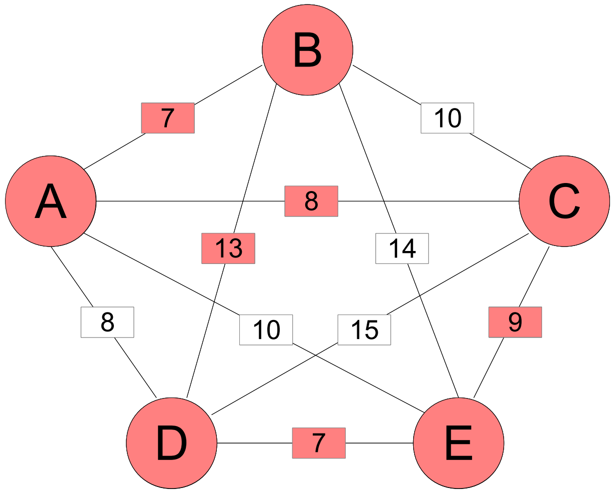
*Find a minimum length **f(π)** Hamiltonian circuit of a graph G(N,A), where n is number of nodes and **π** is a permutation of the nodes indices.*

$$f(\pi) = \sum_{i=1}^{n-1} d_{\pi(i)\pi(i+1)} + d_{\pi(n)\pi(1)}$$

Traveling Tournament Problem

First attempt to solve

- The **nearest neighborhood heuristic** is a simple greedy-type construction heuristic
  - It starts from a chosen city and always select the closest city that is not yet visited



- Initial city: C
- Closest city: A cost: 8
- Closest city: B cost: 7
- Closest city: D cost: 13
- Closest city: E cost: 7
- Return city cost: 9

Total: 44

- Lets see a more complex example

Traveling Tournament Problem
First attempt to solve

- The nearest neighbour algorithm is ***easy to implement*** and ***executes quickly.***
- Usually the last a few edges added are extremely large, due to the “*greedy*" nature.
- In some cases it even constructs the unique worst possible tour.
- How to generate a tour more intelligently?
  - Learn from the previous constructions!

Ant System

- **Ant System** is a basic ant behaviour based algorithm.
- Ants visit the cities sequentially till they obtain a tour.
- Transition from city $\boldsymbol{i}$ to $\boldsymbol{j}$ depends on:
  - Heuristic desirability to visit city $\boldsymbol{j}$ when in city $\boldsymbol{i}$, associated to a static value based on the edge-cost (distance) $\boldsymbol{\eta_{ij}}$
  - Pheromone that represents the learned desirability to visit city $\boldsymbol{i}$ when in city $\boldsymbol{j}$ associated to a dynamic value $\boldsymbol{\tau_{ij}}$

Ant System

Stochastic Solution Construction

- Use memory to remember partial tours.
- Being at a city ***i*** choose next city ***j*** probabilistically among feasible neighbors.
- Probabilistic choice depends on:
  - pheromone trails $\boldsymbol{\tau_{ij}}$
  - heuristic information $\boldsymbol{\eta_{ij} = 1/d_{ij}}$
- A common action choice rule at node ***i*** is:

$$p_{ij}^{k}(t) = [\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}$$

Ant System
Pheromone Update

- Use pheromone evaporation to avoid unlimited increase of pheromone trails and allow **forgetting** of earlier choices
  - Pheromone evaporation rate $0 < \rho \leq 1$
- Use pheromone deposite to positive feedback, reinforcing components of good solutions
  - Better solutions give more feedback

Ant System
Pheromone Update

- Example of pheromone update

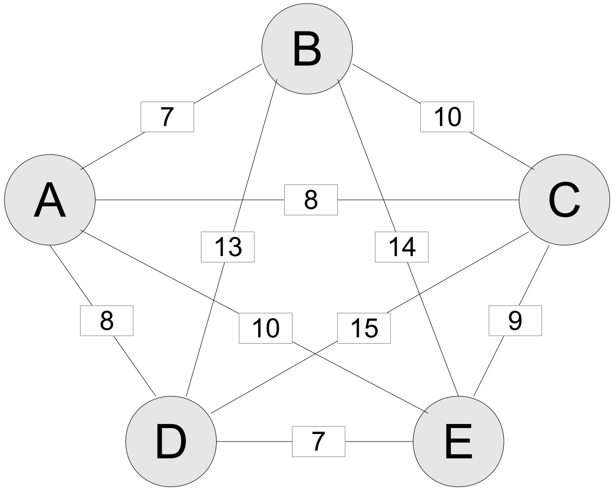
$$\tau_{ij}(t)=(1-\rho)\cdot\tau\,(t-1)+\sum_{k=1}^{m}\Delta\tau_{ij}^{k}$$

$$\Delta\tau_{ij}^{k}=\frac{1}{L_k},\ \ if\ \ arc(i,j)\ \ is\ \ used\ \ by\ \ ant\ \ k\ \ on\ \ its\ \ tour$$

  - ***L_k*:** Tour length of ant *k*
  - ***m***: *number of ants*

Ant System
Simple example

- For our example with #ants=3, α=1, β=5, ρ=0.5 and τ_0=1



  - Heuristic Information

$$\Delta \tau_{ij}^{k} = \frac{1}{L_k}$$

nij	A	B	C	D	E
A	-	1/7	1/8	1/8	1/10
B	1/7	-	1/10	1/13	1/14
C	1/8	1/10	-	1/15	1/9
D	1/8	1/13	1/15	-	1/7
E	1/10	1/14	1/9	1/7	-

  - Pheromone trails

tij	A	B	C	D	E
A	-	1	1	1	1
B	1	-	1	1	1
C	1	1	-	1	1
D	1	1	1	-	1
E	1	1	1	1	-

Ant System
Simple example

- For ant #1 we start from city **C** (random), selection probabilities

$$p_{ij}^{k}(t)=\frac{[\tau_{ij}(t)]^{\alpha}\cdot[\eta_{ij}]^{\beta}}{\sum_{l\in N_i^k}[\tau_{il}(t)]^{\alpha}\cdot[\eta_{il}]^{\beta}}$$

pij	A	B	C	D	E
A	-	0,2493	0,2454	0,2478	0,2502
B	0,2488	-	0,2472	0,2416	0,2530
C	**0,2498**	**0,2520**	**-**	**0,2637**	**0,2493**
D	0,2498	0,2439	0,2611	-	0,2475
E	0,2516	0,2548	0,2463	0,2469	-

- Select a city → rand 0.3747
  - City **B** selected

pij	A	B	C	D	E
A	-	0,0000	0,0000	0,3366	0,3333
B	**0,3317**	**-**	**0,0000**	**0,3281**	**0,3370**
C	0,0000	0,0000	-	0,0000	0,0000
D	0,3329	0,0000	0,0000	-	0,3296
E	0,3354	0,0000	0,0000	0,3353	-

- Select a city → rand 0.6216
  - City **D** selected

pij	A	B	C	D	E
A	-	0,0000	0,0000	0,0000	0,5028
B	0,0000	-	0,0000	0,0000	0,0000
C	0,0000	0,0000	-	0,0000	0,0000
D	**0,4982**	**0,0000**	**0,0000**	**-**	**0,4972**
E	0,5018	0,0000	0,0000	0,0000	-

- Select a city → rand 0.2033
  - City **E** selected

Ant System
Simple example

- First iteration we can have:
  - Ant #1: C-B-D-E-A
  - Ant #2: D-A-C-B-E
  - Ant #2: A-C-B-D-E
- Update the pheromone using this tours

$$\tau_{ij}(t) = [1-\rho] \cdot \tau(t-1) + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}$$

tij	A	B	C	D	E
A	-	0,50	0,54	0,50	0,52
B	0,50	-	0,52	0,52	0,50
C	0,50	0,54	-	0,52	0,52
D	0,54	0,50	0,50	-	0,52
E	0,52	0,52	0,50	0,52	-

- And then iterate

Ant System
Exercise #1

- Implement Ant System according to one of the provided templates.
  - C
  - Java

- The following slides give a practical view of the Ant System algorithm procedures.

Ant System Algorithm
Solution Construction

```
Procedure ConstructSolutions
    For k = 1 To m Do          #m number of ants
        For i = 1 To n Do      #n number of cities
            ant[k].visited[i] ← false
        EndFor
    EndFor
    step ← 1
    For k = 1 To m Do
        r ← random{1, . . . , n}
        ant[k].tour [step] ← r
        ant[k].visited [r] ← true
    EndFor
```

Ant System Algorithm
Solution Construction

```
    While (step < n) Do
        step ← step + 1
        For k = 1 To m Do
            ASDecisionRule(k, step)
        EndFor
    EndWhile
    For k = 1 To m Do
        ant[k].tour [n+1] ← ant[k].tour[1]
        ant[k].tour length ← ComputeTourLength(k)
    EndFor
EndProcedure
```

Ant System Algorithm
Decision Rule

```
Procedure ASDecisionRule(k, i)
 Input k % ant identifier
 Input i % counter for construction step
   c ← ant[k].tour[i-1]
   sum_prob = 0.0
   For j = 1 To n Do
      If ant[k].visited[j] Then
         selection_prob[j] ← 0.0
      Else
         selection_prob[j] ← choice_info[c][j]
         sum_prob ← sum_prob + selection_prob[j]
      EndIf
   EndFor
```

Ant System Algorithm
Decision Rule

```
    r ← random[0, sum_prob]
    j ←1
    p ← selection_prob[j]
    While (p < r ) Do
       j ← j + 1
       p ← p + selection_prob[j]
    EndWhile
    ant[k].tour[i] ← j
    ant[k].visited[j] ← true
EndProcedure
```

Ant System Algorithm

Pheromone Update

```
Procedure ASPheromoneUpdate
    Evaporate
    For k = 1 To m Do
        DepositPheromone(k)
    EndFor
    ComputeChoiceInformation
EndProcedure
```

Ant System Algorithm
Pheromone Update

```
Procedure Evaporate
  For i = 1 To n Do
    For j = i To n Do
      pheromone[i][j]←(1−ρ)·pheromone[i][j]
      pheromone[j][i]←pheromone[i][j]
      %pheromones are symmetric
    EndFor
  EndFor
EndProcedure
```

Ant System Algorithm
Pheromone Update

```
Procedure DepositPheromone(k)
 Input k % ant identifier
   Δτ ← 1/ant[k].tour_length
   For i = 1 To n Do
      j ← ant[k].tour[i]
      l ← ant[k].tour[i+1]
      pheromone[j][l] ← pheromone[j][l] + Δτ
      pheromone[l][j] ← pheromone[j][l]
   EndFor
 EndProcedure
```

Ant System Algorithm
Slides for download

- Slides are available at:
  - http://iridia.ulb.ac.be/~leonardo/teaching/INFO-H-414/lessons02-03.pdf
- Templates are available at:
  - http://iridia.ulb.ac.be/~leonardo/teaching/INFO-H-414/template_c.tar.gz
  - http://iridia.ulb.ac.be/~leonardo/teaching/INFO-H-414/template_java.tar.gz

Ant System
Exercise #2

- Test and analyse the behaviour of the algorithm.
  - Modify some parameters:
    - Number of ants
    - α, β, ρ
- What effect can you appreciate?
- What is the reason?