Solving TSP using Ant Colony Optimization

CORTAL Gustave (ID : F10815023)

18 juin 2020

Introduction

- The travelling salesman problem (TSP) is a NP-hard problem → find the shortest possible route between a list of cities knowing that we have distance between each pairs of cities.
- ACO algorithms try to mimic biological behaviors of ants in their colony → through iterations, an optimal solution has to emerge from different solutions proposed by ants.

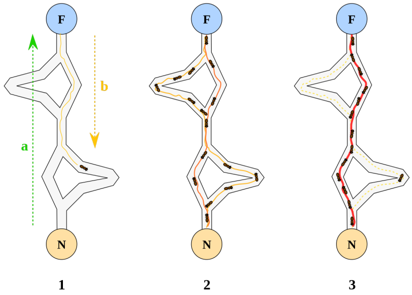

Figure 1 – Emergence of a solution through iterations

A randomized algorithm : concept

procedure ACOBasic
  Set parameters, initialize pheromone trails
  while (termination condition not met) do
    ConstructAntsSolutions
    DaemonActions % *optional*
    UpdatePheromones
  end
end

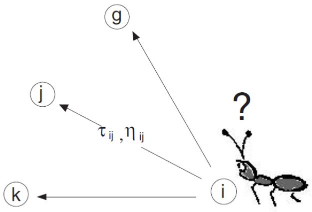

FIGURE 2 – ACO algorithm skeleton and ant choice

A randomized algorithm : probabilistic action choice rule

The probability with which ant k, currently at city i, chooses to go to city j is :

$$p_{ij}^{k} = \frac{[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{l \in N_i^k}[\tau_{il}]^{\alpha}[\eta_{il}]^{\beta}}, ifj \in N_i^k \tag{1}$$

$\eta_{ij} = 1/d_{ij}$: heuristic desirability of visiting city j directly after city i
τ_{ij} : pheromone desirability of visiting city j directly after city i
α and β : the relative influence of the pheromone trail and the heuristic information
N_i^k : the set of cities that ant k has not visited yet

A randomized algorithm : update of pheromone trails

The pheromone evaporation :

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij}, \forall (i,j) \in L \tag{2}$$

All ants deposit pheromone on the arcs they have crossed :

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}, \forall (i,j) \in L \tag{3}$$

ρ : evaporation rate ($\in [0,1]$)
$\Delta\tau_{ij}^{k}$: the amount of pheromone ant k deposits on the arcs it has visited.
$\Delta\tau_{ij}^{k} = 1/C^{k}$ if arc (i,j) belongs to T^{k} where C^{k}, the length of the tour T^{k} built by the k-th ant.

A randomized algorithm : improvements

Elitist Ant System :

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} + e\Delta\tau_{ij}^{bs}, \forall(i,j) \in L \tag{4}$$

Where, $\Delta\tau_{ij}^{bs} = 1/C^{bs}$ if arc (i,j) belongs to T^{bs} where C^{bs} is the length of the best-so-far tour T^{bs}.

Initialize pheromone matrix :

$$\tau_0 = m/C^{nn} \tag{5}$$

$$\tau_0 = (e+m)/\rho C^{nn} \tag{6}$$

Where C^{nn} is the tour length found by applying nearest neighbor (NN) algorithm at a random node.

A randomized algorithm : detect stagnation

- **Standard deviation** σ_L of the length of the tours the ants construct after every iteration (or compute **variation coefficient**).
- **Shannon entropy** $\rightarrow$ the average $\overline{\varepsilon} = \sum_{i=1}^{n} \varepsilon_i / n$ of the entropy ε_i of the selection probabilities at each node :

$$\varepsilon_i = -\sum_{j=1}^{l} p_{ij} log(p_{ij}) \tag{7}$$

Where p_{ij} is the probability of choosing arc (i, j) when being in node i, and l, $1 \leq l \leq n-1$, is the number of possibles choices.

A randomized algorithm : detect stagnation

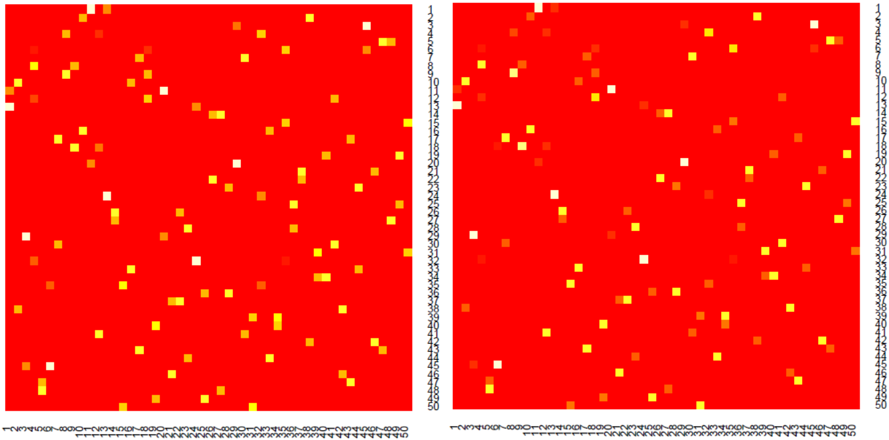

FIGURE 3 – Pheromone matrix heatmap after 100 and 300 iterations (ASE)

A neural network idea : dropout

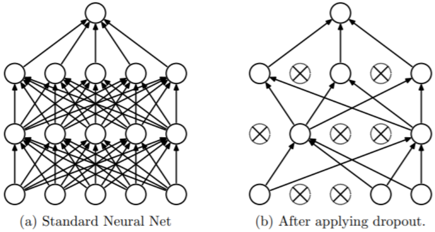

FIGURE 4 – Example of dropout in a neural network

New daemon action → randomly vanish some interesting arcs

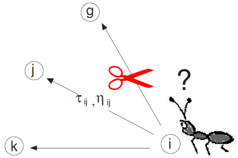

FIGURE 5 – Cut some arcs during the construction procedure

A neural network idea : dropout

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} + e\Delta\tau_{ij}^{bs} * B_{ij}^{bs}, \forall(i,j) \in L \tag{8}$$

Where $B_{ij}^{bs} \sim Bernoulli(p)$. p is a new parameter to tune which decides whether or not we add pheromones to a specific arc ($B_{ij}^{bs} = 1$ or 0).

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} + e\Delta\tau_{ij}^{bs} * N_{ij}^{bs}, \forall(i,j) \in L \tag{9}$$

Where $N_{ij}^{bs} \sim N(1, \sigma^2)$. The mean is set to 1 and σ, the standard deviation, is a new parameter to tune which decides the amount of noises we want around the best iteration tour.

A neural network idea : dropout

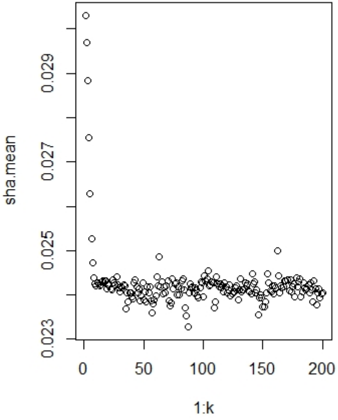

FIGURE 6 – Shannon index analysis on ASE and ASE+Drop

Experimental results

$$\alpha = 1,\ \beta = 3.5,\ \rho = 0.5,\ m = n.$$

Gap error : $(Avg - opt)/opt * 100$

TSP (opt)	Algorithm	Best	Worst	Avg	Std	Gap
eil51 (426)	ASE+Drop	430.35	437.60	433.66	2.07	1.79%
	ASE	432.33	440.83	435.74	3.12	2.29%
	AS	443.35	455.26	450.76	3.39	5.81%
	NN+2-opt	438.72	514.62	476.61	22.57	11.89%
berlin52 (7542)	ASE+Drop	7544.37	7544.66	7544.46	0.14	0.03%
	ASE	7544.37	7663.21	7558.22	36.96	0.22%
	AS	7549.29	7677.12	7622.91	53.62	1.07%
	NN+2-opt	8042.95	8894.5	8448.54	326.69	12.02%
eil76 (538)	ASE+Drop	546.19	555.90	550.89	3.49	2.39%
	ASE	547.33	558.25	551.96	3.41	2.59%
	AS	558.17	570.54	565.45	3.81	5.10%
	NN+2-opt	568.46	630.15	597.18	22.26	11.00%

FIGURE 7 – Experimental results on three different benchmarks

- Taipei Youbike stations dataset provided by Taipei City.
- Get the distance matrix between the stations → Google Maps API.

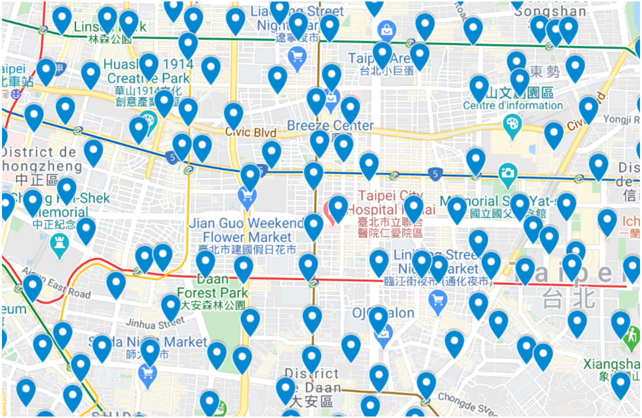

FIGURE 8 – Youbike stations in Google Maps

A practical problem : Youbike Stations

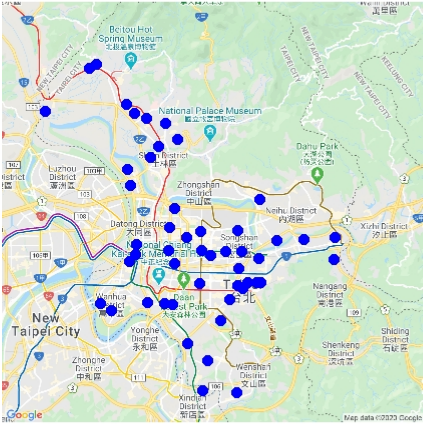

FIGURE 9 – Youbike stations (blue nodes) in Google Maps

A practical problem : results

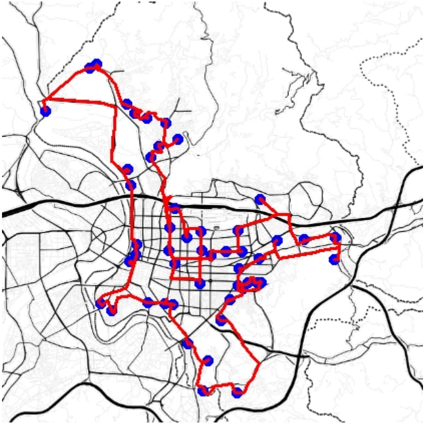

FIGURE 10 – Youbike stations (blue nodes) with best found tour in Google Maps

References

- M. Dorigo, *Optimization, Learning and Natural Algorithms*, PhD thesis, Politecnico di Milano, Italy, 1992.
- M. Dorigo, T.Stützle, *Ant Colony Optimization*, MIT Press, 2005.
- S. Mateusz, K.Michał, B.Aleksander, B.Indurkhya, K.Marek, S.Dana, L.Tom, *Multi-pheromone ant Colony Optimization for Socio-cognitive Simulation Purposes*, Procedia Computer Science, Volume 51, 2015, Pages 954-963.
- S. Nitish, C.G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, *Dropout : A Simple Way to Prevent Neural Networks from overfitting*, Journal of Machine Learning Research, 2014.
- L. Wan, M. Zeiler, S. Zhang, Y. Le Cun, R. Fergus, *Regularization of Neural Networks using DropConnect*, Proceedings of the 30th International Conference on Machine Learning, 2013.