Artificial intelligince and robot path planning

By Mehret Abraha

TABLE OF CONTENTS

Robot path planning

Particle Swarm Optimization

Genetic Algorithm

Tabu Search

Simulated Annealing

Ant Colony Algorithm

Robot path planning

Goal → Accurate procedure for path planning based on soft computing algorithms in a dynamic and unstructured environment

- ❑ Navigating a collision free path.
- ❑ Precision and automation (Minimum human assistance)

Particle SWARM OPTIMZATION

A metaheurstic algorithm (Large scale and minimum assumption)

Iterativelty improves and moves around candidate solutions in search space based on mathematical formulae

Previously found best positions influnce the particle and are constantly updated

Does not require optimization problem to be differentiable

Genetic algorithm(GA)

- Mimics the process of **natural evolution**(Inheritance, mutation, selection and crossover)
- Population of strings called chromosomes or genotypes of **genome encode candidate solution**
- Solution are represented in binary strings of 0 and 1
- **Algorithm terminates** either when **maximum number of generations or satisfying result is found**

Tabu Search

Local search method

Tend to get stuck in suboptimal regions. Many equally fit solutions.

Tabu search

Enhanced method as they use memory that describe the visited solutions or user may provide sets of rules to mark taboos to avoid repetition

Simulated Annealing

Generic metaheuristic and probabilistic algorithm

Acceptably good solution in a fixed amount of time

Replace current solution by random solution depending on difference on **corresponding function** and a **global parameter called T(Temperature)**

When T is **Large** the solution replacement is **random** but when T is **close to Zero downhill(Better) Solutions** are selected.

But the **allowance for uphill solutions saves the system from being stuck in local optima**

Simulated Annealing

When T is **Large** the solution replacement is **random** but when T is **close to Zero downhill(Better) Solutions** are selected.

But the **allowance for uphill solutions saves the system from being stuck in local optima**

Ant Colony algorithm

The approach to **alleviate stagnations** is pheromone control.

Several approaches to **reduce the influence** from **past** experience and encourage **exploration of new paths** that are non optimal

Ant Colony algorithm

Evaporation:

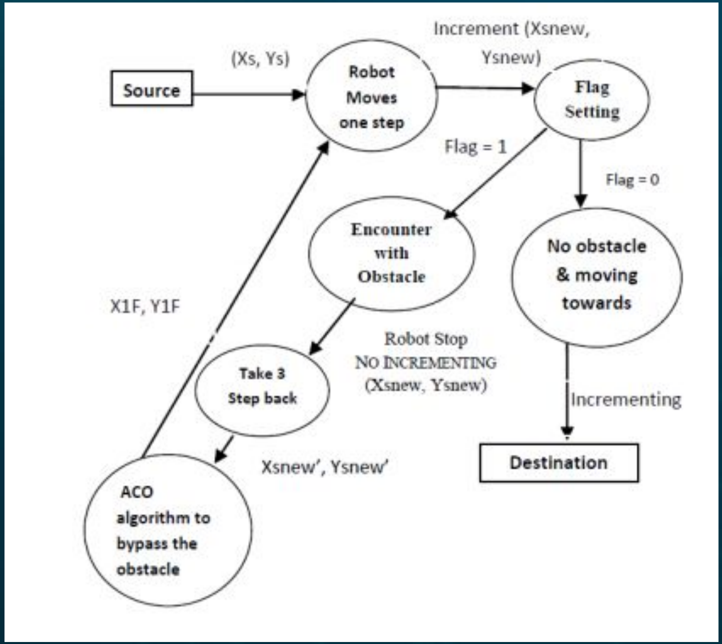
To reduce the effect of past experience and encourage new path finding

In each iteration the pheromone value τ_{ij} are decremented by a factor p

$$\tau_{ij} \Leftarrow \tau_{ij}(1-p)$$

Placing an upper value on amount of pheromones on every edge ij prevents the generation of a dominant path

Solution for Path planning



Solution for ACO Path planning

Xs and Ys are the original positions of the robot

To calculate new position

Xsnew and Ysnew as one step is taken

Xsnew=Xs+Steps*Cos(θ)

Ysnew=Ys+Steps*Sin(θ)

To calculate the angle

First, we take Xprev and Yprev that is the current position of the robot over the XY axis

θ=Tan-1 (Xprev/Yprev)

Solution for ACO Path planning

If the value flag is Zero = No obstacle and if Flag=1 there is obstacle

Number of obstacles is a fixed constraint on this paper=20

obstacles are generated in the moving space of 100*100 and whose **length and width varies between 0 1o 10 dynamically.**

Represented by a pseudo code:

```
oopsV=20;
x=100*rand(1,oopsV);
y=100*rand(1,oopsV);
 l=10*rand(1,oopsV);
w=10*rand(1,oopsV);
for m=1:oopsV
plot([x(1,m) x(1,m)+w(1,m)], [y(1,m) y(1,m)]);
plot([x(1,m) x(1,m)], [y(1,m) y(1,m)+l(1,m)]);
plot[x(1,m) x(1,m)+w(1,m)],[y(1,m)+l(1,m)y(1,m)+l(1,m)]);
plot([x(1,m)+w(1,m)x(1,m)+w(1,m)],[y(1,m) y(1,m)+l(1,m)]);
end
```

Solution for ACO Path planning

Whenever the robot faces an obstacle

Xsnew= Xprev-3*step*cos(ө) (4)

Ysnew= Yprev-3*step*sin(ө) (5)

Ө =Tan-1 (Xprev/ Yprev) (6)

The robot then reaches the destination at the point (XT,YT) after following optimal path

Ant Colony Optimization Implementation

Algorithm is implemented in two steps.

1.Ant k is located at node i, uses the pheromone Tij deposited on the edge (i,j) to compute the probability of choosing next.

$$\mathrm{Pij} = \begin{cases} \dfrac{\tau_{ij^{\alpha}}}{\sum\limits_{j \varepsilon N_{i(k)}} \tau_{ij^{\alpha}}} \; if \; j \, \varepsilon \, N_{i}^{(k)} \\ 0 \end{cases}$$

α =degree of importance of pheromone trail

Ni (k) = the set of neighbor of ant k when located at node i except the predecessor node

This will prevent the ant from traveling back to the same node

Ant Colony Optimization Implementation

The second algorithm is once the tour is completed the best route is chosen. That is the global optimization of Pheromone trial.

$$\tau_{ij} = (1-\rho)\rho + \sum_{k=1}^{N} \Delta\tau_{ij}^{(k)}$$

P ε (0,1] is the evaporation rate

It aims to increase the pheromone value with optimal path.

Ant Colony Optimization Implementation

Pheromone deposited on arc (i, j) by the best ant k is $\Delta\tau_{ij}(k)$

$$\Delta\tau_{ij}(k) = Q/L_k$$

Q is constant and Lk The length of the path transversed by the best ant

Conclusion

- Ant colony optimization (ACO) takes inspiration from the foraging behavior of ant species.
- These ants deposit pheromone on the ground in order to mark some favorable path.
- Ant colony optimization is to be applied for robot –motion control such as navigation and obstacle avoidance in an efficient manner.
- From this, money can be saved, and reliability can be increased by allowing them to adapt themselves according to the environment without further programming.

Thank you ☺️