Swarm intelligence: Ant Colony Optimisation

Lecturer: Liliana Mamani Sanchez
mamanisl@cs.tcd.ie

October 21, 2015
Simulating problem solving?

- Can simulation be used to improve distributed (agent-based) problem solving algorithms?
- Yes: directly, in a supervised fashion (e.g. Neural Nets “simulators”)
- But also, indirectly, via exploration & experimental parameter tuning
- Case Study: Ant Colony Optimisation Heuristics [Dorigo et al., 1996, Dorigo and Di Caro, 1999]
- See also [Bonabeau et al., 2000] for a concise introduction (* recommended reading)
Biological Inspiration

- Real ants foraging behaviour

- long branch is $r$ times longer than the short branch.

- left graph: branches presented simultaneously.

- right graph: shorter branch presented 30 mins. later
Pheromone trails
Ants solve TSP

The Travelling Salesman Problem:

- Let $N = \{a, ..., z\}$ be a set of cities, $A = \{(r, s) : r, s \in V\}$ be the edge set, and $\delta(r, s) = \delta(s, r)$ be a cost measure associated with edge $(r, s) \in A$.
- TSP is the problem of finding a minimal cost closed tour that visits each city once.
- If cities $r \in N$ are given by their coordinates $(x_r, y_r)$ and $\delta(r, s)$ is the Euclidean distance between $r$ and $s$, then we have an Euclidean TSP.
- If $\delta(r, s) \neq \delta(s, r)$ for at least one edge $(r, s)$ then the TSP becomes an asymmetric TSP (ATSP).
A Simple Ant Algorithm for TSP

Initialize
while (!End_condition )
{
    /* call these ‘iterations’ */
    position each ant on a node
    while (! all_ants_built_complete_solution )
    {
        /* call these ‘steps’ */
        for (each ant) 
        {
            ant applies a state transition rule to 
            incrementally build a solution
        }
    }
}
apply local (per ant) pheromone update rule
apply global pheromone update rule
Characteristics of Artificial Ants

Similarities with real ants:

- They form a colony of cooperating individuals
- Use of pheromone for stigmergy (i.e. “stimulation of workers by the very performance they have achieved” [Dorigo and Di Caro, 1999])
- Stochastic decision policy based on local information (i.e. no lookahead)
Dissimilarities with real ants

- Artificial Ants in Ant Algorithms keep internal state (so they wouldn’t qualify as purely reactive agents either)
- They deposit amounts of pheromone directly proportional to the quality of the solution (i.e. the length of the path found)
- Some implementations use lookahead and backtracking as a means of improving search performance
Environmental differences

- Artificial ants operate in discrete environments (e.g. they will “jump” from city to city)
- Pheromone evaporation (as well as update) are usually problem-dependant
- (most algorithms only update pheromone levels after a complete tour has been generated)
ACO Meta heuristic

```plaintext
acoMetaHeuristic() 
while (!terminationCriteriaSatisfied) 
{ /* begin scheduleActivities */
  antGenerationAndActivity()
  pheromoneEvaporation()
  daemonActions() # optional #
} /* end scheduleActivities */

antGenerationAndActivity() 
while (availableResources)
{
  scheduleCreationOfNewAnt()
  newActiveAnt()
}
```
Ant lifecycle

```plaintext
newActiveAnt()

initialise()

M := updateMemory()

while (currentState ≠ targetState) {
    A := readLocalRoutingTable()
    P := transitionProbabilities(A, M, constraints)
    next := applyDecisionPolicy(P, constraints)
    move(next)
    if (onlineStepByStepPheromoneUpdate) {
        depositPheromoneOnVisitedArc()
        updateRoutingTable()
    }
    M := updateInternalState()
} /* end while */

if (delayedPheromoneUpdate) {
    evaluateSolution()
    depositPheromoneOnAllVisitedArcs()
    updateRoutingTable()
}
```
How do ants decide which path to take?

► A *decision table* is built which combines pheromone and distance (cost) information:

► \( A_i = [a_{ij}(t)]_{|N_i|} \), for all \( j \) in \( N_i \), where:

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

1. \( N_i \): set of nodes accessible from \( i \)
2. \( \tau_{ij}(t) \): pheromone level on edge \((i, j)\) at iteration \( t \)
3. \( \eta_{ij} = \frac{1}{\delta_{ij}} \)
The decision table also depends on two parameters:
- \( \alpha \): the weight assigned to pheromone levels
- \( \beta \): the weight assigned to edge costs

Probability that ant \( k \) will choose edge \((i, j)\):

\[
P_{ij}^k(t) = \begin{cases} 
  a_{ij}(t) & \text{if } j \text{ has not been visited} \\
  0 & \text{otherwise}
\end{cases}
\]  

(2)
Pheromone updating (per ant)

- Once all ants have completed their tours, each ant $k$ lays a certain quantity of pheromone on each edge it visited:

$$\Delta \tau_{ij}^k(t) = \begin{cases} \frac{1}{L^k(t)} & \text{if } (i, j) \in T^k, \\ 0 & \text{otherwise} \end{cases}$$  \hspace{1cm} (3)

- $T^k(t)$: ant $k$’s tour
- $L^k(t)$: length of $T^k(t)$
Pheromone evaporation and (global) update

Let

\[ \Delta \tau_{ij}(t) = \sum_{k=1}^{m} \Delta \tau_{ij}^k(t) \]

where \( m \) is the number of ants, and \( \rho \in (0, 1] \) be a pheromone decay coefficient.

The (per iteration) pheromone update rule is given by:

\[ \tau_{ij}(t) = (1 - \rho)\tau_{ij}(t') + \Delta \tau_{ij}(t) \] (4)

where \( t' = t - 1 \)
Parameter setting

How do we choose appropriate values for the following parameters?

- $\alpha$: the weight assigned to pheromone levels
  NB: if $\alpha = 0$ only edge costs (lengths) will be considered
- $\beta$: the weight assigned to edge costs
  NB: if $\beta = 0$ the search will be guided exclusively by pheromone levels
- $\rho$: the evaporation rate
- $m$: the number of ants
Exploring the parameter space

- Different parameters can be “empirically” tested using a multiagent simulator such as Repast
- Vary the problem space
- Collect statistics
- Run benchmarks
- etc
Performance of ACO on TSP

- Ant algorithms perform better than the best known evolutionary computation technique (Genetic Algorithms) on the Asymmetric Travelling Salesman Problem (ATSP)
- ... and practically as well as Genetic Algorithms and Tabu Search on standard TSP
- Good performance on other problems such as Sequential Ordering Problem, Job Scheduling Problem, and Vehicle Routing Problem
Other applications

- Network routing [Caro and Dorigo, 1998] and load balancing [Heusse et al., 1998]
References

Protein folding in the hydrophobic-hydrophilic (hp) is np-complete.

Inspiration for optimization from social insect behaviour.
Nature, 4006.

Antnet: Distributed stigmergetic control for communications networks.

On the complexity of protein folding (abstract).

Principles of protein folding – A perspective from simple exact models.

The ant colony optimization meta-heuristic.

The Ant System: Optimization by a colony of cooperating agents.