Introduction to Artificial Intelligence

Unit # 13

Acknowledgement

- The slides of this lecture have been taken from the lecture slides of CSE307 – "Introduction to Artificial Intelligence" and CSE659 – "Computational Intelligence" by Dr. Sajjad Haider.

Swarm Intelligence

- Social Insects work without supervision.
- Their teamwork is largely self-organized, and coordination arises from the different interactions among individuals in the colony.
- Although these interactions might be primitive, taken together they result in efficient solutions to difficult problems.
- The collective behavior that emerges from a group of social insects has been dubbed ***Swarm Intelligence***.

Swarm Intelligence

- There are two popular swarm inspired methods in computational intelligence areas:
  - Ant colony optimization (ACO)
  - Particle swarm optimization (PSO)
- ACO was inspired by the behaviors of ants and has many successful applications in discrete optimization problems.
- The particle swarm concept originated as a simulation of simplified social system. The original intent was to graphically simulate the choreography of bird of a bird block or fish school. However, it was found that particle swarm model can be used as an optimizer.

Inspiration from Social Insects

- Flexibility: the colony can adapt to a changing environment
- Robustness: even when one or more individual fail, the group can still perform its task
- Self-organization: activities are neither centrally controlled nor locally supervised

Examples

MASON Demo

Working of Ant Colonies

- Certain species of ants are able to find the shortest path to a food source merely by laying and following chemical trails.
- Individual ants emit a chemical substance – a pheromone – which then attracts other ants.
- The colony's efficient behavior emerges from the collective activity of individuals following two very simple rules:
  - Lay pheromone
  - Follow the trails of others

Ant System: An Example

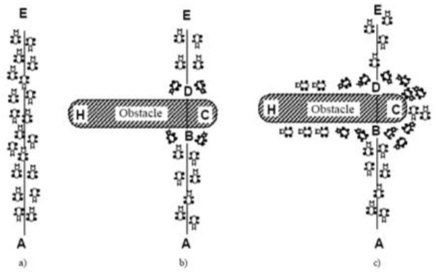

Fig. 1. An example with real ants.

a) Ants follow a path between points A and E.
b) An obstacle is interposed; ants can choose to go around it following one of the two different paths with equal probability.
c) On the shorter path more pheromone is laid down.

Ant System: An Example

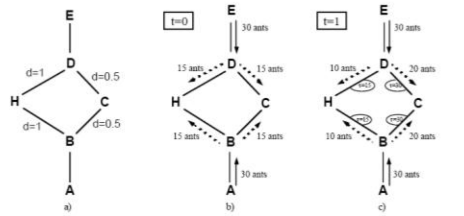

Fig. 2. An example with artificial ants.

a) The initial graph with distances.
b) At time t=0 there is no trail on the graph edges; therefore, ants choose whether to turn right or left with equal probability.
c) At time t=1 trail is stronger on shorter edges, which are therefore, in the average, preferred by ants.

Working of Ant Colonies

- In a simple case, two ants leave the nest at the same time and take different paths to a food source, marking their trail with pheromone.
- The ant that took the shortest path will return first, and this trail will now be marked with twice as much pheromone (from the nest to the food and back) as the path taken by the second ant, which has yet to return.
- Their nest mates will be attracted to the shorter path because of its higher concentration of pheromone.
- As more and more ants take that route, they too lay pheromone, further amplifying the attractiveness of the shorter trail.

Working of Ant Colonies

- The more time it takes for an ant to travel down the path and back again, the more time the pheromones have to evaporate.
- A short path, by comparison, gets marched over faster, and thus the pheromone density remains high as it is laid on the path as fast as it can evaporate.
- Pheromone evaporation has also the advantage of avoiding the convergence to a locally optimal solution.
- If there were no evaporation at all, the paths chosen by the first ants would tend to be excessively attractive to the following ones. In that case, the exploration of the solution space would be constrained.

ACO*

- A general-purpose heuristic algorithm which can be used to solve different combinatorial optimization problems.
- Not interested in simulation of ant colonies, but in the use of artificial ant colonies as an optimization tool.
- Major differences with a real (natural) ant:
  - Artificial ants will have some memory
  - They will not be completely blind
  - They will live in an environment where time is discrete

* The Ant Systems: Optimization by a colony of cooperating agents by Marco Dorigo, Vittorio Maniezzo, and Alberto Colorni

Traveling Sales Person Problem

- Given a number of cities and the costs of traveling from one city to any other city, what is the cheapest round-trip route that visits each city exactly once and then returns to the starting city?

	A	B	C	D	E	F	G	H
A	0	8	3	1	4	9	3	6
B	8	0	5	10	11	4	3	6
C	3	5	0	8	7	1	5	12
D	1	10	8	0	9	11	6	4
E	4	11	7	9	0	5	17	3
F	9	4	1	11	5	0	4	1
G	3	3	5	6	17	4	0	7
H	6	6	12	4	3	1	7	0

ACO Algorithm

- Let $b_i(t)$ (i=1, ..., n) be the number of ants in town i at time t and let m = sum [$b_i(t)$] be the total number of ants.
- Each ant is a simple agent with the following characteristics:
  - it chooses the town to go to with a probability that is a function of the town distance and of the amount of trail present on the connecting edge;
  - to force the ant to make legal tours, transitions to already visited towns are disallowed until a tour is completed (this is controlled by a tabu list);
  - when it completes a tour, it lays a substance called *trail* on each edge (i,j) visited.

ACO Algorithm

- Let $\tau_{ij}(t)$ be the ***intensity of trail*** on edge (i,j) at time t.
- Each ant at time t chooses the next town, where it will be at time t+1.
- In n iterations, each ant completes a tour.
- At this point the trail intensity is updated according to the following formula

$$\tau_{ij}(t+n)=\rho\cdot\tau_{ij}(t)+\Delta\tau_{ij} \quad (1)$$

where

ρ is a coefficient such that $(1 - \rho)$ represents the *evaporation* of trail between time t and t+n.

ACO Algorithm

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \qquad (2)$$

where $\Delta\tau_{ij}^{k}$ is the quantity per unit of length of trail substance (pheromone in real ants) laid on edge (i,j) by the k-th ant between time t and t+n; it is given by

$$\Delta\tau_{ij}^{k} = \begin{cases} \frac{Q}{L_k} & \text{if k - th ant uses edge (i, j) in its tour (between time t and t + n)} \\ 0 & \text{otherwise} \end{cases} \qquad (3)$$

where Q is a constant and L_k is the tour length of the k-th ant.

- In order to satisfy the constraint that an ant revisits a town, a data structure, called ***tabu list***, is associated with each ant that saves the towns already visited upto time t and forbids the ant to visit them again before n iterations (a tour) have been completed.

ACO Algorithm

- ***Visibility***, nij, is defined as the inverse of the length of the edge (i,j)
- The transition probability from town i to town j for the kth ant is defined as

$$p_{ij}^{k}(t) = \begin{cases} \frac{[\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum_{k \in allowed_k} [\tau_{ik}(t)]^{\alpha} \cdot [\eta_{ik}]^{\beta}} & \text{if } j \in allowed_k \\ 0 & \text{otherwise} \end{cases} \qquad (4)$$

- Alpha and Beta are parameters that control the relative importance of trail versus visibility.
- The transition probability is a trade-off between visibility and trail intensity.

Example

	A	B	C	D	E
A	0	5	2	9	3
B	5	0	8	1	9
C	2	8	0	10	2
D	9	1	10	0	4
E	3	9	2	4	0

- Initial Solutions
  - A C D B E 22
  - B A C D E 21
  - E D B C A 15
  - E B C A D 28
  - A D C E B 30
- Given
  - $\rho = 0.6$,
  - Q = 1
  - $\alpha = 0.8$
  - $\beta = 0.8$
  - $\tau_{ij} = 0$

Computing τ_{ij}

	A	B	C	D	E
A	0	(1/21)	(1/22+1/21+1/15+1/28)	(1/28+1/30)	0
B		0	(1/15)	(1/22)	(1/22+1/28+1/30)
C			0	(1/22+1/21+1/30)	(1/30)
D				0	(1/21+1/15)
E					0

- Initial Solutions
  - A C D B E 22
  - B A C D E 21
  - E D B C A 15
  - E B C A D 28
  - A D C E B 30

Computing τ_{ij} and η_{ij}

τ_{ij}

	A	B	C	D	E
A	0.000	0.048	0.195	0.069	0.000
B	0.048	0.000	0.067	0.045	0.115
C	0.195	0.067	0.000	0.126	0.033
D	0.069	0.045	0.126	0.000	0.114
E	0.000	0.115	0.033	0.114	0.000

η_{ij}

inverse of the length of the edge (i,j)

	A	B	C	D	E
A	0.00	0.20	0.50	0.11	0.33
B	0.2	0.00	0.13	1.00	0.11
C	0.5	0.13	0.00	0.10	0.50
D	0.11	1.00	0.10	0.00	0.25
E	0.33	0.11	0.50	0.25	0.00

Computing Transition Probabilities

$$p_{ij}^{k}(t) = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum_{k \in allowed_k} [\tau_{ik}(t)]^{\alpha} \cdot [\eta_{ik}]^{\beta}} & \text{if } j \in allowed_k \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

- Numerators only
  - $p_{AB}^{1} = (0.048)^{0.8} (0.20)^{0.8} = 0.024$
  - $p_{AC}^{1} = (0.195)^{0.8} (0.5)^{0.8} = 0.156$
  - $P_{AD}^{1} = (0.069)^{0.8} (0.11)^{0.8} = 0.020$
  - $P_{AE}^{1} = 0.00$
- After Normalization (dividing by the denominator)
  - $p_{AB}^{1} = 0.121$
  - $p_{AC}^{1} = 0.778$
  - $p_{AD}^{1} = 0.101$
- So Ant 1, starting from node A, would decide about the next node based on these probabilities.
- The process is repeated for the other nodes in the sequence as well.
- All the ants follow this process.