Take-Home Practice on Multiagent Systems (Non-Graded)

Note. We have not included problems on subsumption architecture or behavior nets since the quiz questions on those readings will be more conceptual in nature.

1. Consider the problem of finding the shortest route through several cities, such that each city is visited only once and in the end return to the starting city (the Travelling Salesman problem). Suppose that in order to solve this problem we use a genetic algorithm, in which genes represent links between pairs of cities. For example, a link between London and Paris is represented by a single gene 'LP'. Let also assume that the direction in which we travel is not important, so that LP = PL.

   - How many genes will be used to represent each individual in the population if the number of cities is 10?

     Answer: Each chromosome will consist of 10 genes with each gene representing the path between a pair of cities in the tour.

   - How many genes will be in the alphabet of the algorithm if number of cities is 10?
     (Hint: A gene is represented by a link between two cities. Alphabet here means the set of all possible genes. For example, if there are 3 cities A,B,C possible genes are AB, AC, and BC – note AB = BA)

     Answer: The alphabet will consist of 45 genes. Indeed, each of the 10 cities can be connected with 9 others. Thus, $10 \times 9 = 90$ is the number of ways in which 10 cities can be grouped in pairs. However, because the direction is not important (i.e. London–Paris is the same as Paris–London) the number must be divided by 2. So, we shall need $90/2 = 45$ genes in order to encode all pairs. In general the formula for n cities is: $n(n - 1)$ 2

2. Suppose a genetic algorithm uses chromosomes of the form x = abcdefgh with a fixed length of eight genes. Each gene can be any digit between 0 and 9. Let the fitness of individual x be calculated as: $f(x) = (a + b) - (c + d) + (e + f) - (g + h)$,

   Let the initial population consist of four individuals with the following fitness values:
   x_1 = 6 5 4 1 3 5 3 2 ; $f(x_1) = (6 + 5) - (4 + 1) + (3+ 5) - (3 + 2) = 9$
   x_2 = 8 7 1 2 6 6 0 1 ; $f(x_2) = (8 + 7) - (1 + 2) + (6+ 6) - (0 + 1) = 23$
   x_3 = 2 3 9 2 1 2 8 5 ; $f(x_3) = (2 + 3) - (9 + 2) + (1+ 2) - (8 + 5) = -16$
   x_4 = 4 1 8 5 2 0 9 4 ; $f(x_4) = (4 + 1) - (8 + 5) + (2+ 0) - (9 + 4) = -19$

   - Cross the fittest two individuals to produce offsprings o1 and o2, using crossover at the middle point and calculate their fitness values.

     Answer:
     Offspring 1 **= 8 7 1 2** 3 5 3 2   f(o1) = 15
     Offspring 2 **=** 6 5 4 1 **6 6 0 1**   f(o2) = 17

   - What would the individual representing the optimal solution (i.e. with the maximum fitness) be and what is the optimal fitness (remember, each gene can only take on values from 0 to 9)?

Answer:
The optimal solution should have a chromosome that gives the maximum of the fitness function.

$$max\ f(x) = max\ [(a + b) - (c + d) + (e + f) - (g + h)]$$

Because genes can only be digits from 0 to 9, the optimal solution should be:

$$X = 9\ 9\ 0\ 0\ 9\ 9\ 0\ 0$$

and the maximum fitness is $f(X) = (9 + 9) - (0 + 0) + (9 + 9) - (0 + 0) = 36$

- Based on the initial population shown above, would a GA ever reach the optimal solution without the mutation operator?

  Answer:
  No, the algorithm will never reach the optimal solution without mutation. The optimal solution is X = 9 9 0 0 9 9 0 0. If mutation does not occur, then the only way to change genes is by applying the crossover operator. Regardless of the way crossover is performed, the only outcome is an exchange of genes of parents at certain positions. Since none of the parents have any way to get a permutation like that of X (e.g., since none contain 9 in the first position), it is not possible to reach the optimal solution without mutation.

3. Consider the problem of collecting garbage from across the city of Boulder while using the least amount of fuel. Garbage needs to be collected from all locations and there is only one garbage truck available. You need to find the best route through all the locations starting from the truck depot, and ending the route back at the depot. Model this problem using ACO and GAs.

   Answer:
   The problem is similar to the traveling salesperson problem (TSP). The locations across Boulder can be represented as nodes (similar to cities in TSP). The path from one location to the next will have a certain amount of distance which is the cost to go from one location to the next. We need to minimize the total fuel consumed - which is achieved by minimizing the total distance traveled. Refer to notes and slides for how to use ACO for solving TSP.

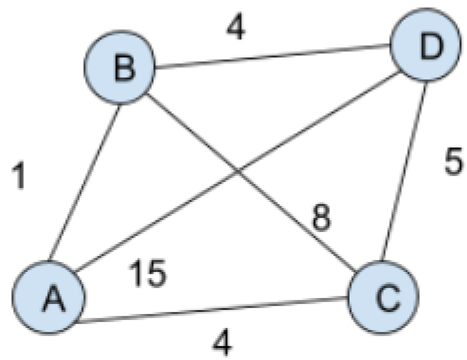
   GA:
   Here we can represent a gene as a location, so if we have N locations, we can represent them as L_1, L_2, L_3 ... L_N and L_0 can be the truck depot.

   An individual is then a string of locations representing the route in which they are visited. For example if there are 5 locations, an individual can be: L_0, L_4, L_2, L_5, L_1, L_3, L_0

   The fitness function is the total distance traveled by following the locations in an individual.

   We can create offsprings by considering two routes/individual and performing operations like crossover or mutations (swap any 2 locations except the first and last).

4. Consider an iteration in ACO being applied to the travelling salesperson problem. The graph below shows the cities, and the distances between them,



- Assume that the pheromone values for edges between all cities is 1.
  If an ant starts at city **A,** which city is it most likely to choose next?

  Hint:

  $$P_{i,j} = \frac{(\tau_{i,j})^{\alpha}(\eta_{i,j})^{\beta}}{\sum\left((\tau_{i,j})^{\alpha}(\eta_{i,j})^{\beta}\right)}$$

  where $P_{i,j}$ is the probability of going to city j from city i. Assume that both **α** and **β** are 1, and **$\tau_{i,j}$** is the pheromone density between cities i, j

  Answer: It is most likely to visit City **B** as it has the lowest edge cost, which is the only factor that matters since the pheromone values are the same for the other options.

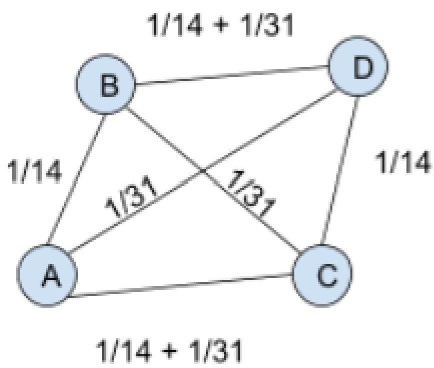
- If we change the value of **τ** on the path(**A, C)** to 5, and keep the rest of the pheromone values as 1, what city is the ant starting at A most likely to visit next?

  Answer:
  $P_{A,C} = (5)^1(1/4)^1 \ / \ ((5)^1(1/4)^1 + (1)^1(1/15)^1 + (1)^1(1)^1) = 0.53$
  $P_{A,B} = (1)^1(1)^1 / ((5)^1(1/4)^1 + (1)^1(1/15)^1 + (1)^1(1)^1) = 0.42$
  $P_{A,D} = (1)^1(1/15)^1 / ((5)^1(1/4)^1 + (1)^1(1/15)^1 + (1)^1(1)^1) = 0.02$
  It is most likely to visit City **C** as it has **P** = 0.53. Other cities have lower **P.**

- Assume that there are two ants a^1 and a^2 that both start from **A.**
  a^1 follows the path (**A -> B -> D -> C -> A)** at a cost of 14
  a^2 follows the path (**A -> D -> B -> C -> A)** at a cost of 31
  What will the pheromone density on each path look like at the end when both ants have completed their tours?

  Note: $\Delta\tau^k_{i,j} = 1/L_k$ for each ant a^k that travels along the path **(i, j)**. Where L_k is the total length of the tour for ant a^k.

Answer:

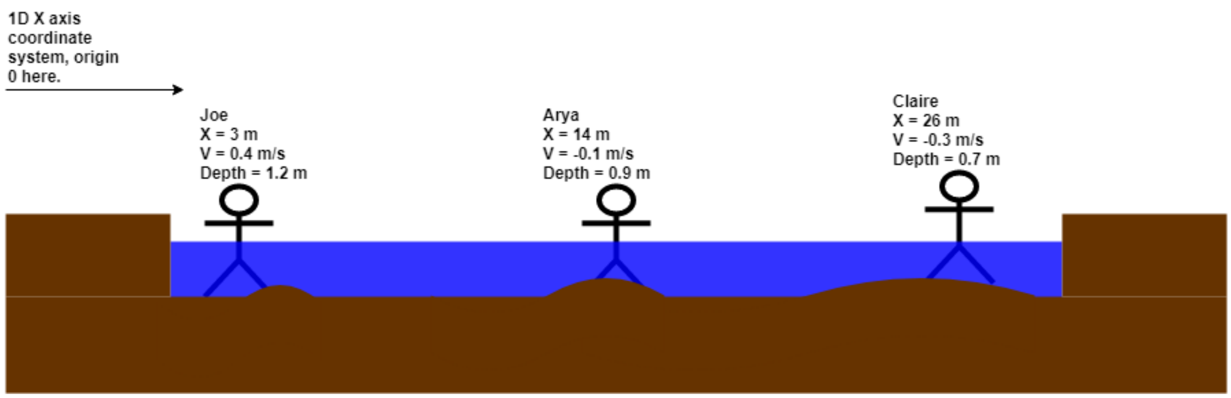


6. Boids were covered in class (see slides) as an example of how a combination of simple rules can yield interesting and effective control for multiagent systems. Remember that the rules follow boids follow are:
   - Collision avoidance
   - Velocity matching
   - Flock centering

   In the YouTube video from the slides: https://www.youtube.com/watch?v=GUkjC-69vaw a swarm of Boids are depicted chasing a green target ball and avoiding a red ball. Where in the PSO algorithm would we implement the green ball chasing and red ball avoidance behavior to produce the behavior shown in the video?

   Answer: In the particle swarm algorithm the evaluation function should include terms that increase if the particle gets closer to the target green ball and decrease if it gets closer to the red ball. That metric can then be used for evaluating the particle best location as well as the swarm best location.

7. Shown below are three surveyors who are wading a river crossing looking for the shallowest point to place their bridge anchor. Their initial positions and velocities are shown below along with the depth of each surveyor.



Assume all three surveyors are using a PSO algorithm with c1 = c2 = 0.5, complete the below table with their positions and velocities for the next 3 iterations. The random number generator returns values r1 and r2 for each iteration as shown below.

Iter			Joe			Arya			Claire		
	r1	r2	X	Vel	Depth	X	Vel	Depth	X	Vel	Depth
0	NA	NA	3	0.4	1.2	14	-0.1	0.9	26	-0.3	0.7
1	0.1	0.6	3.4	7.3	1.0	13.9	3.5	1.0	25.7	-0.3	0.6
2	0.4	0.7			1.5			1.1			0.5
3	0.2	0.4			1.3			1.2			0.6
4	0.8	0.3			1.1			0.8			0.6

Remember, for particle swarm optimization each particle's velocity and position are updated in a relatively simple loop:

1. Calculate a particle's velocity update based on its current velocity, the best position it has found so far, and the best position the swarm has found so far.
    a. $v_{t+1} = v_t + c_1 r_1(personal\ best - x_t) + c_2 r_2(swarm\ best - x_t)$
        i. Where c_1 is a individual weighting factor, increasing c1 causes a particle to increase its chances of sticking with its personal best.
        ii. Where c_2 is a social weighting factor, increasing c2 causes a particle to increase its chances of heading towards the swarm best.
        iii. Where $r_1\ and\ r_2$ are random numbers from 0 - 1
2. The particle's position is then updated based on the last velocity.
    a. $x_{t+1} = x_t + v_t$
3. Then there is bookkeeping.
    a. $x_t = x_{t+1}$
    b. $v_t = v_{t+1}$
    c. Evaluate the cost function for every particle in the swarm and communicate the information.
        i. Note we're not going to focus on the cost function evaluation in this problem and have provided the depths found at each iteration for each particle.

Answer:

Iter			Joe			Arya			Claire		
	r1	r2	X	Vel	Depth	X	Vel	Depth	X	Vel	Depth
0	NA	NA	3	0.4	1.2	14	-0.1	0.9	26	-0.3	0.7
1	0.1	0.6	3.4	7.3	1.0	13.9	3.5	1.0	25.7	-0.3	0.6
2	0.4	0.7	**10.7**	**15.1**	1.5	**17.4**	**7.6**	1.1	**25.4**	**-0.3**	0.5
3	0.2	0.4	**25.8**	**17.3**	1.3	**25.0**	**8.9**	1.2	**25.1**	**-0.3**	0.6
4	0.8	0.3	**43.1**	**8.3**	1.1	**33.9**	**4.6**	0.8	**24.8**	**-0.1**	0.6

Below are two update calculations from iteration 2 for reference:

Joe

X_2 = 3.4 + 7.3 = 10.7

V_2 = 7.3 + 0.5*0.4*(3.4-3.4) + 0.5*0.7*(25.7 – 3.4) = 15.1