MSBD6000A 2018-19 Spring Semester Assignment #1

Date assigned: Thursday, Feb 21

Due time: 23:59pm on Monday, March 4

How to submit it: Submit your written answers as a pdf file on canvas.ust.hk. Submit your code for the last question as a zip file

Penalties on late papers: 20% off each day (anytime after the due time is considered late by one day)

Problem 1. (30%) Consider a 10x10 grid without any obstacles, and a robot with the same specification as our boundary following robot: eight sensors and four actions.

- Design a reactive production system to control the robot to go to one of the four corners, wherever its initial position is.
- Show that it is not possible to have a reactive production system to make the robot visit every cell in the grid.
- Design a state machine to achieve the above task. In addition to having memory about previous action taken and previous sensory input, you can have internal variables (mental states) and actions to change the values of these variables.

Problem 2. (15%) Which boolean function does the following TLU implement? The TLU has five inputs. Its weight vector is $(1.1, 3.1, -1, -2, 0.5)$, and the threshold is 1.

Problem 3. (20%) Consider the problem of training a TLU to do logical disjunction (logical "or") using the error-correction procedure that we talked about in class. Notice that this operator takes two inputs, but to apply the procedure, you need to add one more input whose value is always 1, and use 0 as the threshold. Suppose that we start with the initial weights all equal to 0, and learning rate $c = 1$. Find a *minimal* set of training instances that will correctly train the TLU according to the procedure. Here a training set is minimal if removing any instance in it will not produce a TLU for the logical disjunction. Please show the details of your work inlcuding the converging sequence of the weights.

Problem 4. (Programming) (35%) Design and implement a genetic programming system to evolve some perceptrons that match well with a given training set. A training set is a collection of tuples of the form $(x_1, ..., x_n, l)$, where x_i's are real numbers and l is either 1 (positive example) or 0 (negative example). So for your genetic programming system, a "program" is just a tuple $(w_1, ..., w_n, \theta)$ of numbers (weights and the threshold). Answer the following questions:

1. What's your fitness function?
2. What's your crossover operator?

3. What's your copy operator?
4. What's your mutation operator, if you use any?
5. What's the size of the initial generation, and how are programs generated?
6. When do you stop the evolution? Evolve it up to a fixed iteration, when it satisfies a condition on the fitness function, or a combination of the two?
7. What's the output of your system for the training set in the next page? This training set will be uploaded to canvas as a csv file.

In addition to answer these questions, please also upload your source code and executable (in any language and platform) as a zip archive file and name it as YourStudentID.zip on canvas.

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	$Output$
0.05	0.47	0.89	0.28	0.37	0.58	0.71	0.75	0.29	0
0.62	0.2	0.71	0.61	0.01	0.63	0.52	0.04	0.94	1
0.61	0.7	0.63	0.43	0.79	0.83	0.92	0.81	0.27	0
0.75	0.42	0.03	0.02	0.23	0.42	0.13	0.81	0.51	0
0.66	0.07	0.88	0.83	0.92	0.83	0.3	0.35	0.79	1
0.57	0.56	0.36	0.94	0.51	0.7	0.56	0.01	0.78	1
0.07	0.66	0.54	0.6	0.38	0.67	0.66	0.96	0.55	0
0.75	0.43	0.11	0.04	0.35	0.48	0.25	0.59	0.5	1
0.91	0.87	0.65	0.95	0.42	0.27	0.25	0.2	0.76	1
0.46	0.47	0.13	0.99	0.82	0.56	0.72	0.16	0.62	1
0.09	0.08	0.17	0.57	0.24	0.68	0.99	0.71	0.48	0
0.34	0.5	0.26	0.34	0.06	0.86	0.58	0.41	0.74	0
0.7	0.77	0.27	0.98	0.52	0.38	0.32	0.85	0.46	0
0.66	0.57	0.22	0.77	0.4	0.1	0.61	0.38	0.21	0
0.4	0.92	0.79	0.18	0.58	0.3	0.08	0.71	0.7	1
0.3	0.85	0.36	0.23	0.68	0.15	0.21	0.01	0.31	1
0.8	0.05	0.21	0.44	0.74	0.25	0.85	0.83	0.06	0
0.26	0.32	0.24	0.18	0.74	0.36	0.09	0.48	0.18	1
0.76	0.65	0.52	0.0	0.39	0.64	0.36	0.03	0.62	1
0.4	0.46	0.71	0.73	0.34	0.29	0.75	0.93	0.19	0
0.59	0.66	0.28	0.19	0.87	0.31	0.37	0.84	0.41	1
0.35	0.73	0.43	0.46	0.48	0.49	0.48	0.86	0.1	0
0.47	0.45	1.0	0.91	0.24	0.04	0.72	0.17	0.7	0
0.86	0.11	0.28	0.19	0.96	0.07	0.48	0.92	0.91	1
0.78	0.2	0.49	0.26	0.83	0.42	0.03	0.42	0.73	1
0.76	0.01	0.63	0.82	0.5	0.8	0.6	0.52	0.63	1
0.9	0.31	0.22	0.22	0.31	0.72	0.45	0.79	0.72	0
0.84	0.72	0.09	0.67	0.95	0.54	0.05	0.97	0.16	1
0.94	0.07	0.43	0.19	0.1	0.63	0.5	0.78	0.35	0
0.51	0.55	0.27	0.84	0.41	0.38	0.75	0.91	0.53	0
0.68	0.99	0.05	0.56	0.63	0.81	0.1	0.58	0.51	1
0.31	0.77	0.85	0.04	0.32	0.87	0.88	0.04	0.42	1
0.32	0.98	0.38	0.77	0.54	0.57	0.22	0.46	1.0	1
0.57	0.1	0.98	0.44	0.5	0.37	0.7	0.54	0.46	1
0.24	0.18	0.17	0.08	0.62	0.46	0.93	0.98	0.32	0
0.39	0.84	0.92	0.31	0.34	0.14	0.72	0.85	0.24	0
0.86	0.84	0.75	0.59	0.73	0.57	0.77	0.38	0.67	1
0.0	0.77	0.97	0.4	0.64	0.02	0.46	0.84	0.14	1
0.74	0.7	0.6	0.79	0.34	0.24	0.64	0.82	0.38	0
0.71	0.36	0.95	0.23	0.08	0.46	0.96	0.04	0.28	0
0.61	0.03	0.67	0.83	0.03	0.46	0.58	0.36	0.21	0
0.72	0.87	0.45	0.48	0.47	0.61	0.67	0.28	0.45	0
0.23	0.66	0.03	0.04	0.44	0.23	0.76	0.11	0.77	0
0.35	0.21	0.69	0.32	0.41	0.91	0.44	0.13	0.49	1
0.92	0.25	0.34	0.85	0.12	0.37	0.23	0.79	0.81	0
0.58	0.68	0.26	0.83	0.97	0.78	0.3	0.14	0.9	1
0.24	0.85	0.91	0.24	0.98	0.97	0.05	0.03	0.08	1
0.35	0.72	0.87	0.66	0.58	0.79	0.19	0.86	0.85	1
0.38	0.93	0.96	0.05	0.4	0.33	0.06	0.09	0.27	1