# CS620 End-semester Exam (Spring 2021)

**Max marks: 55** **Duration: 3 hours**

- *Be brief, complete and stick to what has been asked.*
- *Untidy presentation of answers, and random ramblings will be penalized by negative marks.*
- *Unless asked for explicitly, you may cite results/proofs covered in class without reproducing them.*
- ***If you need to make any assumptions, state them clearly.***
- ***Do not copy solutions from others. Penalty for offenders: FR grade.***
- **Expected time to solve: $\leq 180$ mins.**
- **You will have an additional 30 mins to revise, scan your answer papers and upload on Moodle.**

1. Consider the neural network shown in Fig. 1. Assume that each node in the hidden and output layers uses a ReLU activation function. We will use the terminology used in the paper "On the Effectiveness of

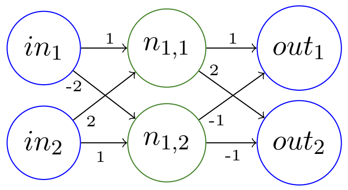

Figure 1: A simple DNN

Interval Bound Propagation for Training Verifiably Robust Models" by Sven Gowal et al, but introduce some modifications in the calculations.

Specifically, for each node $n$ in the network, we will maintain a set of intervals where the set can have at most two intervals. This allows us to be more precise in representing the possible values of a node. For example, if we say that the value of $n$ lies in $\{[1,2],[5,8]\}$, then we have $(1 \leq n \leq 2) \vee (5 \leq n \leq 8)$. Note that this is more precise than $(1 \leq n \leq 8)$, which is what we would have if we had to use a single interval, i.e. $[1,8]$, to represent the possible values of $n$.

(a) *[5 marks]* Suppose we have $y = ReLU(x)$, and we know that $x$ lies in $\{[l_1,u_1],[l_2,u_2]\}$, where $l_1 \leq u_1 \leq l_2 \leq u_2$. Find the best possible representation of the values of $y$ using a set of atmost two intervals. Give clear reasoning for your answer.

(b) *[5 marks]* Now consider $y = 2.x - 3.z + 5$, where both $x$ and $z$ lie in $\{[-1,0],[2,3]\}$. Find the best possible representation of $y$ using a set of atmost two intervals. Give clear reasoning for your answer.

(c) *[10 marks]* Suppose $y_{true}$ is the component of the output vector in the DNN shown in Fig. 1 that has the highest value. For example, if the inputs $in_1$ and $in_2$ have the value 1 each, the outputs have values $out_1 = 3$ and $out_2 = 6$, and hence $y_{true} = 2$ in this case.

For the example considered above, we wish to check if $y_{true}$ stays at 2 if $in_2$ has a value in $\{[-1,0.5],[1.5,2]\}$ (instead of being 1). Use the technique discussed above for representing values of nodes using at most two intervals to determine the $\widehat{z}_K$ vector referred to in equation (10), page 4 of the above paper.

2. In this question, we will try to synthesize a pre-emptive shield, as described in the paper "Safe Reinforcement Learning via Shielding" by Alshiekh et al.

   To simplify things, we consider a two-state specification automaton as shown in Fig. 2(a), and also a two-state abstraction of the underlying MDP, as shown in Fig. 2(b). The accepting states of the specification automaton are $\{q_0, q_1\}$, while $q_e$ is the non-accepting error state (representing violation of the desired property). Similarly, the accepting states of the MDP abstraction are $\{r_0, r_1\}$, while $r_e$ is the non-accepting error state (representing incorrectness of the abstraction).

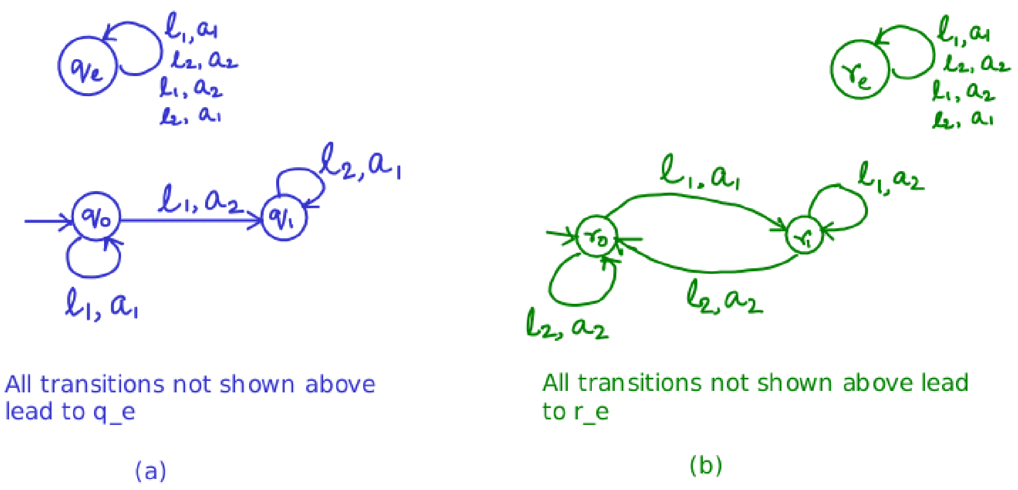

Figure 2: Specification and abstraction automata

   For both the above automata, $\{l_1, l_2\}$ represent observations of the MDP state, while $\{a_1, a_2\}$ represent agent actions.

   (a) *[10 marks]* Construct the safety game automaton $\mathcal{G}$ as described in pages 12 and 13 of the above paper. When construction $\mathcal{G}$, you can merge all states in $\{q_e\} \times \{r_0, r_1\}$ into one state (representing violation of the specification), and similarly merge all states in $\{q_0, q_1, q_2\} \times \{r_e\}$ into another state (representing incorrectness of the abstraction).

   Clearly identify the final/accepting states of your safety game automaton, as described in page 13 of the paper.

   (b) *[5 marks]* The winning region $W$ of the safety game automaton $\mathcal{G}$ is the largest subset of states of $\mathcal{G}$ such that for every state in $W$ and for every observation $l_i$ of the MDP state, there exists an action $a_j$ that takes $\mathcal{G}$ to a final/accepting state of $\mathcal{G}$.

   List the states in the winning region of $\mathcal{G}$ constructed in the previous sub-question.

   (c) *[10 marks]* A pre-emptive shield can be represented by an automaton that has the same states and transitions as $\mathcal{G}$. However, in every state, and for every observation $l_i$ of the MDP state, the shield outputs the minimally interfering set of safe actions that the learning agent can take. This is represented by the function $\lambda_S(g, l)$ in page 13 of the paper.

   Give the $\lambda_S(g, l)$ function for the pre-emptive shield obtained from the safety game automaton $\mathcal{G}$ constructed above. Thus, you must prepare a table that lists the set of safe actions for the learning agent for every combination of state of $\mathcal{G}$ and MDP state observation $l_i$.

3. *[10 marks]* In this question, we will try to reason about the learning of an automaton from an RNN $R$, as described in the paper "Extracting Automata from Recurrent Neural Networks Using Queries and Counterexamples" by Weiss et al.

   Recall that at all points of the reasoning, we have the following artifacts:

   - An automaton $\mathcal{A}$ that is proposed to mimic $R$.

- A partition $p : S \to \mathbb{N}$ of the state space $S$ of $R$, and a corresponding automaton $A^{R,p}$ that tries to approximate the behaviour of $R$ as per the partition $p$.

The technique presented in the paper essentially alternates between (i) repairing $\mathcal{A}$ (using the L* algorithm) based on a counterexample $w$ on which $\mathcal{A}$ and $R$ disagree, but on which $R$ and $A^{R,p}$ agree, and (ii) refining the partition $p$ (resulting in a new automaton $A^{R,p}$) based on a counterexample $w'$ on which $\mathcal{A}$ and $R$ agree, but on which $\mathcal{A}$ and $A^{R,p}$ disagree.

In order to keep the reasoning simple, we will assume the following:

A1: In the L* algorithm, everytime a counterexample (to the claim that the proposed automaton $\mathcal{A}$ is equivalent to the abstract automaton $A^{R,p}$) is processed, the new proposed automaton has at most $k$ additional states.

A2: Everytime the partition $p$ is refined, the count of partitions is increased by at most $t$.

Suppose the technique outlined in the paper starts with $\mathcal{A}$ having $K$ states, and $p_0$ (initial partition) having $T$ partitions.

Suppose further that application of the technique described in the paper results in 2 refinements of the partition, and 2 modifications of the proposed automaton by the L* algorithm. Note that these refinements and modifications of proposed automaton can occur in any arbitrarily interleaved manner.

What is the maximum number of states possible in the final proposed automaton?

Give clear justification for your answer.

*[Hint: You don't need to follow the details of either Algorithm 1 or Algorithm 2 given in the paper to answer this question.]*