I. Overview

The goal of this project is to acquire hands-on experience in writing code for a Markov Decision Process (MDP) in a manner that can be solved using a dynamic programming (DP) formalism. The assignment comprises of writing a Matlab-based simulation, the results of which are to be analyzed and explained. Each student, or a group of at most two students, is required to submit a printed project report. The deadline for submitting the project report is Oct. 8, 2015.

II. Problem Description: Real-Time Packet Scheduling

Internet switches and routers are often faced with the task of arbitrating service between a set of packet queues that are contending for a similar output link. For example, two users may be generating packet traffic that is destined to the same output link/port, as illustrated in Figure 1. All packets are assumed to have the same length and time is slotted equally into packet-times (i.e. a single time slot is identical in duration to the time it takes to transmit a packet).

Consider a queueing system comprising of two independent FIFO (first-in-first-out) queues, as depicted in figure 1. Each queue has a distinct packet arrival rate, ($\lambda_1=0.3$, $\lambda_2=0.6$), whereby arrivals follow a Bernoulli i.i.d. process. During each time slot only one of the queues can be served (i.e. have a packet depart from it). Note that the number of packet arrivals during a single time slot ranges from 0 to 2. The scheduling mechanism is governed by the agent and is responsible for determining which of the two queue is to be served at any given time slot. Moreover, assume each of the queues has a maximal capacity of 8 packets. For every packet that departs from the system, the agent receives 1 credit point. If, however, a packet arrives to a full queue, that packet will be dropped and a reward of -5 is incurred. The overall goal of the agent will be to minimize the queue sizes, thereby reducing the packet-loss rate.

![Figure 1: A dual-queue packet scheduling system](image-url)
Project components:

(a) Formulate the problem as an MDP (i.e. describe all components of the problem, along with how they are to be interpreted and what their boundaries are).
(b) Write Matlab code for solving the problem by means of policy-iteration based dynamic programming. Make sure you describe the structure and flow of your code.
(c) Plot and explain the progression of the policy improvement process. Provide an intuitive explanation for both beginning and ending points.
(d) Plot and describe the value function obtained for the optimal policy. How confident are you that the policy reached is indeed optimal?

III. Grading Policy

The project report should be coherent and clearly written. The following is a suggested structure for the project report:

- Cover page and Abstract
- Introduction and Background
- Design (suggested outline)
  o Design objectives
  o Design challenges
  o Technical approach (preferably a flow chart)
  o Experiments and Results - mainly addressing (a) through (d) above.
- Summary
  o What have you learned from this project? What have you achieved?
- Appendix A: Source code listings - the implementation (this item is judged based on the combination of your design and implementation).

Students are encouraged to discuss the project with each other; however individual work is to be submitted by each student/pair.