**Artificial Intelligence** **Ali Raza Syed**
**Lecture 8: Complex Decisions**

## Introduction

In this lecture, we're concerned with how agents should make decisions under uncertainty. Uncertainty arises from the non-deterministic environment: each action has a set of possible outcomes or states. But the resulting states have utilities which can be calculated. The best action can then be chosen by employing decision theory which combines probability theory and utility theory.

Suppose an action $a$ can lead to a set of states $s_a$. Then the outcome has an expected utility, $E(u(s_a)) = \sum s^{'} \in s_a u(s^{'})P(s_a = s^{'})$. The second term in the summation is the probability of getting to a given state, while the first term is utility of that state. This gives us a notion of how "good" an outcome can be achieved by taking action $a$. A rational agent chooses an action $a^*$ which maximises the expected utility. This makes sense since utilities encode preferences and thus a a rational agent chooses an action to achieve an outcome which maximises preferences.

There are other critera for choosing an action rather than maximizing utility. The maximin criterion chooses the least bad worst outcome. The maximax criterion chooses the best outcome for each criterion to minimize regret.

## Sequential Decition Problems

We assume that the environment is fully observable. An agent's utilty depends on a sequence of actions and thus a sequence of states the agent moves through. One option is to use a greedy technique so that, at any given point, the agent always makes the best decision for the next outcome. But this leads to a myopic view which ignores the longer term utility and the need for planning.

## Markovian Decision Problem

We specify a transition model for states, $s \xrightarrow{a} s^{'}$: $P(s^{'}|s, a)$, which is Markovian. In addition, the agent is rewarded at each state with the reward function, $R(s)$ (negative rewards are possible for bad outcomes). Then the utility of a run is not just related to the final outcome, but sums over all the actions and states. This leads to a Markov decision process (MDP) applicable to a fully observable, non-deterministic environment with additive rewards.

A solution to the MDP must specify the choice of action for every state and we call this the policy ($\pi(s) : s \to a$). The optimum policy is defined as the policy with highest expected utility; thus an agent chooses an action based on the optimum policy $\pi^*(s)$.

Given a run, we would like to compute the utility of that run, $U_{run}(s_1, \cdots, s_n)$. We need to consider, for our environment, whether the horizon is finite or infinite, that is, does the environment stop at some point or can the agent continue indefinitely to find the best outcome. An infinite horizon has stationary utilities: the same state always has the same value. So infinite horizons are easier to deal with when determining the optimal policy.

Given stationarity, we can use additive or discounted rewards. Additive rewards assign utilities to runs by summing the rewards: $U_{run}(s_0, s_1, \cdots) = R(s_0) + R(s_1) + \cdots$, while discounted rewards use a discount factor $0 \leq \gamma \leq 1$: $U_{run}(s_0, s_1, s_2, \cdots) = R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \cdots$. Under an additive rewards structure, the infinite horizon can lead to policy utilities of $\pm\infty$ and we are unable to compare policies. Thus we might prefer discounted rewards unless we can restrict our runs to some finite sequence.

**Bellman Equations**

The Bellman equation is $U(s) = R(s) + E(\text{best action})$. So the utility of a state is the reward for for achieving that state and the expected utility for the next state based on the optimal action to be taken. The utilities of the states can be then set up as a system equations (one for each state $s_i$). This leads to an iteration algorithm for finding the solution to our MDP. We start by assign arbitrary values to the the utilities and update the utility of each state through: $U_{i+1}(s) = R(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s^{'}|s, a) U_i(s^{'})$, where $i$ denotes the current iteration. Bellman proved that these are guaranteed to converge to the solution of our system of equations.

Given a policy, $\pi_i(s)$, we can determine the utility of each state under that policy. This allows us to tweak the policy to determine a new one, $\pi_{i+1}(2)$. The policy iteration algorithm also converges.

**POMDP**

The MDP assumes a fully observable environment so that the agent always knows what state it's in and the optimal policy depends on the current state. In the real world, we have partially observable environments, where the current state may be unclear. Instead we have a probability distribution over the set of possible states. This leads to the Partially Observable MDP. The setup for a POMDP is similar to the filtering problem, discussed in the context of Bayesian networks, with both transition and sensor models, and allows us to compute the current belief based on conditional distributions. A POMDP hinges on belief rather the a notion of a best action given a state.