Agenda for Today

1. Introduction
   - Introduction
   - Examples
   - A Sequential Decision Model

2. Model Formulation
   - Decision Epochs and Periods
   - State and Action Sets
   - Rewards and Transition Probabilities

3. Decision Rules and Policies

4. Optimality Equations and Principle of Optimality

5. Shortest Route and Critical Path Models

6. Backward Induction Algorithm to Calculate $u_t^*$ and $A_t^s$
We have learned (or will learn) many decision techniques LP, IP.

Sometimes game...

Sometimes dynamic...

- Today’s decision impact tomorrow’s...
- We should account for the relationship between present and future decisions, and present and future outcomes...

Markov decision processes (MDP) = Stochastic dynamic programs = Stochastic control problems
Examples...

- Inventory Management
- Airline scheduling
- Homeland security resource allocation
- Car/engine replacement
- Highway Maintenance problem
A MDP model consists of decision epochs, states, actions, rewards, and transition probabilities.

Generates a reward and determines the state at the next decision epoch through a transition probability function.

Policy or strategies are prescriptions of which action to choose under any eventuality at every future decision epoch.

Decision makers seek policy which are **optimal** in some sense.
An analysis of this MDP model Includes...

- Providing conditions under which there exist easily implementable optimal policies
- Determining how to recognize these policies
- Developing and enhancing algorithms for computing them; and
- Establishing convergence of these algorithms
A Sequential Decision Model

- At a specified point in time, a decision maker, observes the state of a system.
- Based on the state, the decision maker chooses an action.
- The action choice produces two results:
  - The decision maker receives an immediate reward (cost).
  - The system evolves to a new state at a subsequent point in time according to a probability distribution determined by the action choice.
- At this subsequent point in time, the decision maker faces a similar problem, but the system may be in a different state, and may have different set of actions to choose from.
Five elements of a MDP model

1. Decision epochs
2. States
3. Actions
4. Transition probabilities
5. Rewards

The decision maker’s goal is to maximize total (discounted) rewards
Let $T$ denote the set of decision epochs:

1. Continuum: decisions may be made at
   - all decision epochs (continuously)
   - random points of time when certain events occur, such as arrivals
   - Some specific time chosen by decision maker

2. Discrete time: time is divided into periods or stages
   - Finite: $T = \{1, 2, \ldots, N\}$ for some positive integer $N < \infty$
   - Infinite: $T = \{1, 2, \ldots\}$
   - We write $T = \{1, 2, \ldots, N\}$, $N \leq \infty$ to include both cases
   - Following convention that no decision was made at period $N$.
     So the last decision is made at decision epoch $N - 1$
State and Action Sets

- Denote the set of possible system states by $S$
- At state $s \in S$, the decision maker observes $s$ and choose action $a$ from the set of allowable actions in state $s$, $A_s$
- We assume $S$ and $A_s$ do not vary with $t$
- Usually we assume $S$ and $A_s$ are discrete (finite or countably infinite)
As a result of choosing action $a \in A_s$ in state $s$ at decision epoch $t$

1. The decision maker receives a reward $r_t(s, a)$
2. The system state at the next decision epoch is determined by the probability distribution $p_t(\cdot | s, a)$
Rewards

- When positive, income; when negative, cost
- It may not be immediately at time $t$;
- Could be a lump sum received at a fixed or random time prior to next decision epoch, or
- Accrued continuously throughout the current period, or
- An expectation of random quantity that depends on system state at the subsequent decision epoch
- A combination of the above
When Rewards Depends on the Next State...

- We denotes $r_t(s, a, j)$ as the reward at time $t$ when the state of system is $s$, action is $a \in A_s$, and the system occupies state $j$ at decision epoch $t + 1$.
- Its expected value at decision epoch $t$ is...

$$r_t(s, a) = \sum_{j \in S} r_t(s, a, j)p_t(j|s, a)$$
Transition Probability Function $p_t(j|s, a)$

- $p_t(j|s, a)$ denotes the probability that the system is in state $j \in S$ at time $t + 1$, when the decision maker chooses action $a \in A_s$ in state $s$ at time $t$.
- Certainly we have
  $$ \sum_{j \in S} p_t(j|s, a) = 1 $$
Final Stage $N$

- In finite horizon MDP, no decision is made at decision epoch $N$.
- So the reward at this time point is only a function of the state.
- We denote $r_N(s)$.
- Sometimes refer to it as **salvage value** or **scrap value**.
We refer to the collection of the objects

\[ \{ T, S, A_s, p_t(\cdot | s, a), r_t(s, a) \} \]

as a Markov Decision Process (MDP)

- “Markov” is used because the transition probability and reward function depend on the past only through the current state of the system, and the action selected by the decision maker in that state
- However, in some cases, the induced stochastic process need not be Markov...
Decision Rules

- A decision rule prescribes a procedure for action selection in each state at a specified decision epoch.
- Deterministic Markovian Decision rule: $d_t : S \rightarrow A_s$. For each $s \in S$, $d_t(s) \in A_s$
- History dependent decision rules: $d_t$ could be a function of the history $\{h_t = s_1, a_1, ..., s_{t-1}, a_{t-1}, s_t\}$ History $h_t$ follows the recursion $h_t = (h_{t-1}, a_{t-1}, s_t)$
A decision rule prescribes a procedure for action selection in each state at a specified decision epoch.

- Deterministic Markovian Decision rule: \( d_t : S \rightarrow A_s \). For each \( s \in S \), \( d_t(s) \in A_s \)

- History dependent decision rules: \( d_t \) could be a function of the history \( \{h_t = s_1, a_1, ..., s_{t-1}, a_{t-1}, s_t\} \) History \( h_t \) follows the recursion \( h_t = (h_{t-1}, a_{t-1}, s) \)

- Randomized decision rule specifies a probability distribution on the set of actions
We classify decision rules as:

- History dependent and randomized (HR)
- History dependent and deterministic (HD)
- Markovian and randomized (MR)
- Markovian and deterministic (MD)

We denote the set of decision rules at time $t$ by $D^K_t$, where $K \in \{HR, HD, MR, MD\}$, $D^K_t$ is called a decision rule set.
A policy, contingency plan, plan, or strategy specifies the decision rule to be used at all decision epoch.

A policy is a sequence of decision rules; i.e.,

\[ \pi = (d_1, d_2, ..., d_{N-1}) \]

where \( d_t \in D^K_t \) is the decision rule set for all \( t = 1, ..., N - 1 \)
Finite-horizon Policy Evaluation Algorithm (for fixed $\pi \in \Pi^{HD}$)

1. Set $t = N$ and $u_N^\pi(h_N) = r_N(s_N)$ for all $h_N \in H_N$
2. If $t = 1$, stop; otherwise go to step 3
3. Substitute $t - 1$ for $t$ and compute $u_t^\pi(h_t)$ for each $h_t \in H_t$ by

$$u_t^\pi(h_t) = r_t(s_t, d_t(h_t)) + \sum_{j \in S} p_t(j|s_t, d_t(h_t)) u_{t+1}^\pi(h_t, d_t(h_t), j)$$

Noting that $(h_t, d_t(h_t), j) \in H_{t+1}$
4. Return to 2
Optimality Equations

• Let

\[ u^*_t(h_t) = \sup_{\pi \in \Pi^{HR}} u^\pi_t(h_t) \]

be the optimal policy valuation

• We have \( u^*_N(h_N) = r_N(s_N) \) and

\[ u^*_t(h_t) = \sup_{a \in A_{st}} \left\{ r_t(s_t, a) + \sum_{j \in S} p_t(j|s_t, a)u^*_t+1(h_t, a, j) \right\} \]

For each \( t \) and \( h_t \), the optimal decision rule \( d^*_t(h_t) \) is such that \( u^*_t \) is attained

• Optimality equations are fundamental tools in MDP; each solution to the optimality equations are the optimal returns from period \( t \) onward for each \( t \)
Principle of Optimality

- “An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision” (Bellman, 1957)
- “There exists a policy that is optimal for every state (at every stage)” (Denardo, 1982)
A finite directed graph consists of a set of nodes and directed arcs.

- path: a sequence of arcs that connects one node to another
- The value above arc give the “distance,” “cost,” or “time.”
- Node 1 is called origin; Node 8 is destination
Shortest/Longest Route Problem

- A **shortest route problem** corresponds to finding a path from the origin to destination of minimal total value
  - Applications: optimization of shipping and communication networks

- While a **longest route problem** seeks a path of greatest total value consists of a set of nodes and directed arcs.
  - Applications: critical path analysis (in which each arc represents a necessary task and the longest route value represents the minimal time to complete all task)

- Both above could be modeled as a MDP
Backward Induction Algorithm to Calculate $u^*_t$ and $A^*_{s_t, t}$

1. Set $t = N$ and $u^*_N(s_N) = r_N(s_N)$ for all $s_N \in S$
2. If $t = 1$, stop; otherwise go to step 3
3. Substitute $t - 1$ for $t$ and compute $u^*_t(s_t)$ for each $s_t \in S$ by

$$u^*_t(s_t) = \max_{a \in A_{s_t}} \left\{ r_t(s_t, a) + \sum_{j \in S} p_t(j | s_t, a) u^*_{t+1}(j) \right\}$$

Set

$$A^*_{s_t, t} = \arg \max_{a \in A_{s_t}} \left\{ r_t(s_t, a) + \sum_{j \in S} p_t(j | s_t, a) u^*_{t+1}(j) \right\}$$

4. Return to 2
Solve the Longest Route Problem Using BIA

1. Set \( t = 4 \) and \( u^*_4(8) = 0 \)

2. Since \( t \neq 1 \). Set \( t = 4 - 1 = 3 \):
   \[
   u^*_3(5) = 1 + u^*_4(8) = 1, \quad u^*_3(6) = 2 + u^*_4(8) = 2, \\
   u^*_3(7) = 6 + u^*_4(8) = 6, \\
   A^*_{5,3} = (5, 8), \quad A^*_{6,3} = (6, 8), \quad A^*_{7,3} = (7, 8)
   \]
Continued...

3. Since $t \neq 1$. Set $t = 3 - 1 = 2$:

$$u_2^*(2) = \max \{4 + u_3^*(5), 5 + u_3^*(6)\} = \max \{5, 7\} = 7$$

$$u_2^*(3) = \max \{5 + u_3^*(5), 6 + u_3^*(6), 1 + u_3^*(7)\} = \max \{6, 8, 7\} = 8; \quad u_2^*(4) = 2 + u_3^*(7) = 8$$

Optimal action sets: $A_{2,2}^* = (2, 6), A_{3,2}^* = (3, 6), A_{4,2}^* = (4, 7)$
4. Since $t \neq 1$. Set $t = 2 - 1 = 1$:

$$u^*_1(1) = \max\{2 + u^*_2(2), 3 + u^*_2(3), 4 + u^*_2(4)\} = \max\{9, 12, 11\} = 12$$

Optimal action sets: $A^*_{1,1} = (1, 3)$

5. Since $t = 1$, stop.
Conclusion

Since the optimal action set at each state is a singleton, there is a unique optimal policy $\pi^* = (d_1^*, d_2^*, d_3^*)$ where

- $d_1^*(1) = (1, 3)$
- $d_2^*(2) = (2, 6)$, $d_2^*(3) = (3, 6)$, $d_2^*(4) = (4, 7)$,
- $d_3^*(5) = (5, 8)$, $d_3^*(6) = (6, 8)$, $d_3^*(7) = (7, 8)$

This example illustrates the Principle of Optimality:

- Starting at node 1 and using the optimal policy moves the system from node 3 to node 6 to 8
- If the system started at node 3, the optimal policy would be to go to node 6 and then node 8