An Introduction to Stochastic Dynamic Programming

Justin C. Goodson

1 Modeling

A stochastic dynamic program is a model of an optimization problem that consists of the sequence *decision, information, decision, information, decision,* ... Draw a decision tree representing this sequence.

A Markov decision process (MDP) is a type of stochastic dynamic program. It consists of the elements below. Label these elements on the decision tree.

- **Decision Epochs.** Points in time at which decisions are made.
- **States.** Information necessary to define feasible actions at an epoch, the reward for choosing an action, and the transition function.
- **Actions.** Decisions available from a particular state.
- **Information.** Possible realizations of uncertainty and the associated probability distributions.
- **Contributions** Expected rewards or costs incurred by choosing a particular action in a given state.
- **Transition Function.** How the system evolves from one state to the next given a selected action and a realization of information.
- **Objective.** A function of the contributions, typically maximization or minimization of the expected sum of contributions across all epochs.

The process proceeds as follows:

- Decisions are made at epochs $0, 1, \ldots, K$
- At epoch k the system occupies state s_k
- In state s_k, choose action x_k from the set of feasible decisions $\mathcal{X}(s_k)$
- The expected contribution accrued is $C_k(s_k, x_k)$
- A realization of random information w_{k+1} is observed
- The transition to $s_{k+1} = S(s_k, x_k, w_{k+1})$ is made
- Begin at initial state s_0 and repeat until reaching a terminal state s_K

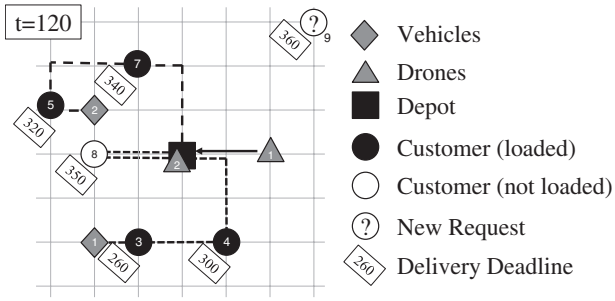
The optimization proceeds as follows:

- A policy π is a sequence of decision rules, $\pi = (X_0^\pi, X_1^\pi, \ldots, X_K^\pi)$
- A decision rule $X_k^\pi(s_k) : s_k \mapsto \mathcal{X}(s_k)$ maps states to feasible actions
- The set of all policies is Π
- The objective is to optimize the expected total contribution, conditional on the initial state:

$$\max_{\pi \in \Pi} \mathbb{E}\left[\sum_{k=0}^{K} C_k\left(s_k, X_k^\pi\left(s_k\right)\right) \middle| s_0\right]$$

Example. "Same-Day Delivery with Heterogeneous Fleets of Drones and Vehicles," Ulmer & Thomas. *Networks*, 2018:72, 475-505.

Problem Description. A combined fleet of vehicles and drones deliver goods from a depot to customers who dynamically request service over a finite operating horizon. Vehicles operate across a road network, have unlimited capacity, and travel slowly. In contrast, drones travel quickly in straight-line distances, can carry only one package, and must recharge between trips. Requests follow a known spatial-temporal distribution, but their times and locations are unknown before the request is made. Requests may be accepted or rejected. Accepted requests must be serviced by a hard deadline and may be satisfied via drone or vehicle. The objective is to maximizes the expected number of serviced requests.



Model. Identify decision epochs, states, actions, information, contributions, transition function, and objective. Briefly describe these elements in words. Mathematical formalism is not required.

2 Essential Theory

From a given state s_k, an MDP seeks a policy with value

$$\max_{\pi \in \Pi} \mathbb{E}\left[\sum_{k'=k}^{K} C_{k'}\left(s_{k'}, X_{k'}^{\pi}(s_{k'})\right) \middle| s_k\right].$$

It can be shown that this value may be calculated recursively as

This is called the *value function*. Return to the decision tree diagram. Label the value function on the diagram.

The value function is the basis for the *backward induction* algorithm to solve MDPs:

Initialize $V_K(s_K)$ for all s_K
for $k = K-1, K-2, \ldots, 0$ do

$$V_k(s_k) = \max_{x \in \mathcal{X}(s_k)} \left\{ C_k(s_k, x_k) + \mathbb{E}\left[V_{k+1}(s_{k+1}) \middle| s_k, x \right] \right\} \qquad \text{for all } s_k$$

Build the decision tree first. Begin with initial state s_0, then use the model to transition until the terminal states are reached. Then apply backward induction.

3 Policies

Solving the value functions presents several computational challenges. Label these challenges on the decision tree diagram.

The curse of dimensionality renders exact solution methods computationally intractble for many problems of practical interest. Further, it is often difficult to fully characterize the structure of an optimal policy via analysis. Thus, we often construct heuristic policies.

Recall the definition of a policy:

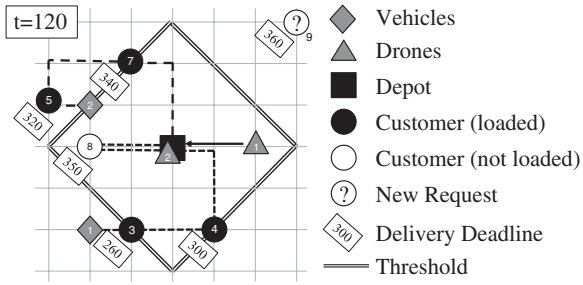
Within this definition, we can be creative with policy construction. Following are four ways to think about constructing policies.

Policy Function Approximations (PFAs) are typically simple functions that map a state to a feasible action. They're useful when you have a good idea of how to make a decision and can design a function that captures the structure of the problem.

PFA Example. "Same-Day Delivery with Heterogeneous Fleets of Drones and Vehicles," Ulmer & Thomas. *Networks*, 2018:72, 475-505.

Problem Description. A combined fleet of vehicles and drones deliver goods from a depot to customers who dynamically request service over a finite operating horizon. Vehicles operate across a road network, have unlimited capacity, and travel slowly. In contrast, drones travel quickly in straight-line distances, can carry only one package, and must recharge between trips. Requests follow a known spatial-temporal distribution, but their times and locations are unknown before the request is made. Requests may be accepted or rejected. Accepted requests must be serviced by a hard deadline and may be satisfied via drone or vehicle. The objective is to maximizes the expected number of serviced requests. For more details, see the illustrative example in Section 3.2.

Threshold PFA. Choose a time τ. Customers whose travel time from the depot is beyond τ are assigned to drones, if possible. Otherwise, customers are assigned to vehicles, if possible.



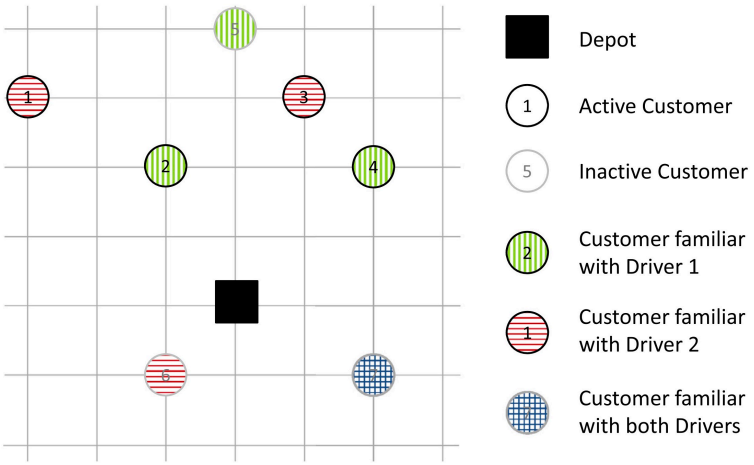
Write down the decision rule and the resulting class of policies:

Within this class of policies, search for a threshold τ that yields the best policy value.

Cost Function Approximations (CFAs) select actions using a parameterized optimization model. CFAs are like PFAs, but with explicit optimization over actions. A classical CFA is a myopic policy with a parameterized correction term, as in the example below. CFAs may be useful when there is a tractable deterministic approximation of the problem and we have some idea about how to manage uncertainty.

CFA Example. "Binary Driver-Customer Familiarity in Service Routing," Ulmer, Nowak, Mattfeld, & Kaminski. *European Journal of Operational Research*, 2020:286, 477-493.

Problem Description. Over a sequence of days, a provider serves customers with a fleet of drivers. Drivers' working time each day is limited. Customers request service intermittently. Requests are known to the provider at the beginning of each day and the driver designs routes to service these requests. Cost is determined by time required to traverse routes and service customers. Service time depends on whether the assigned driver is familiar with the customer. If the driver has previously serviced the customer, then this leads to a smaller service time. Consequently, a routing decision made today can impact service times on subsequent days. For example, consider a set of retail stores serviced from a central warehouse. Stores frequently request delivery, but not every day. A driver becomes familiar with a customer by locating the store, finding the correct unloading area, reporting to the store manager, unloading goods, and completing delivery review. On the first visit, these steps take some time, but on subsequent visits, because the driver is familiar with the customer, the process proceeds faster. The objective is to minimize expected routing and service costs.



Model.

- *Decision Epochs.* Decisions are made at the beginning of each day.
- *States.* A state consists of the set of active customers and the current familiarity between drivers and customers.
- *Actions.* An action is an assignment of active customers to drivers as well as the routing of drivers. The set of feasible actions is defined by the constraints of an uncapacitated VRP with route duration limits.
- *Information.* A set of new customer requests, revealed at the beginning of the day.
- *Contribution.* The sum of routing and service costs.
- *Transition function.* An action updates familiarity. New requests update the set of active customers.
- *Objective.* Minimize expected sum of contributions.

Familiarity CFA. The CFA is a myopic policy with a cost correction term. The contribution function is augmented with a term that captures a driver's familiarity with active customers. Cost is artificially decreased for unfamiliar customers. Write down the decision rule.

Value Function Approximations (VFAs) approximate the value of being in a state, typically by approximating the value function. VFAs are useful when we need to capture the impact of a decision now on the future, and the impact can be expressed as a well-defined function.

VFA Example. "Meso-Parametric Value Function Approximation for Dynamic Customer Acceptances in Delivery Routing," Ulmer & Thomas. *European Journal of Operational Research*, 2020:285, 183-195.

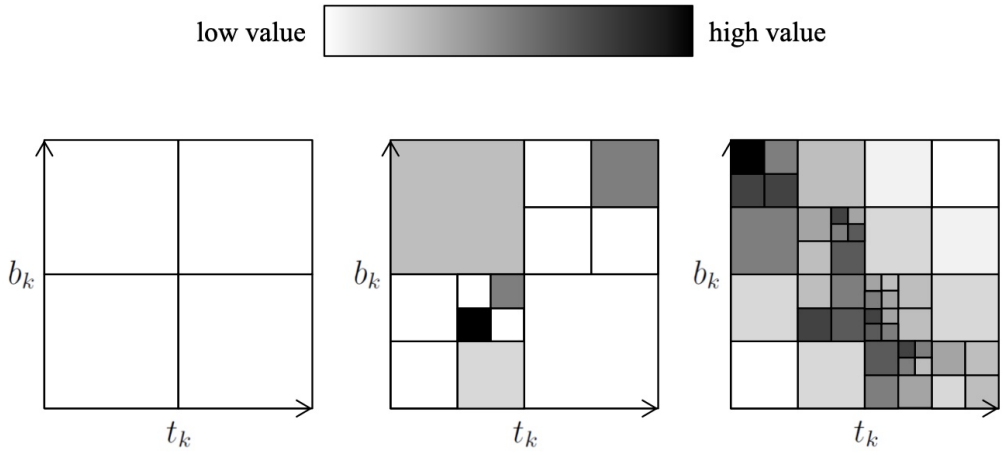
Problem Description. A dispatcher receives delivery requests at random times and from random locations across a geographic area serviced by a single vehicle. The dispatcher must decide to accept or reject requests. If accepted, a request requires some amount of the vehicle's capacity and yields some revenue. The objective is to maximize expected revenue subject to constraints on vehicle capacity and total service time.

Model.

- *Decision Epochs.* Decisions occur whenever a request for service is received.
- *States.* A state consists of the time that a request occurs; the set of already accepted orders; the new order, its location, load, and revenue; and the current route plan.
- *Actions.* An action is a pair consisting of a decision to accept or reject the request and, if accepted, a route plan to serve the request. Route plans must respect constraints on capacity and duration.
- *Information.* The time and location of a new request.
- *Contribution.* The revenue associated with the new request, if it is accepted, otherwise zero.
- *Transition function.* If an action accepts a request, then this updates the route plan and the set of already accepted orders. Exogenous information updates the state with information about the new request: time, location, capacity, and revenue.
- *Objective.* Maximize expected revenue.

Parametric VFA. Simplify the state variable by reducing it to two dimensions: a route plan's remaining time and a route plan's remaining vehicle capacity. Make a linear approximation of the value function:

Non-Parametric VFA. Use the same simplification plus simulation to estimate value across these dimensions. The results might look like this:



a. Initial Lookup Table b. Mid-Procedure c. Final Approximation

Denote the parametric or non-parametric VFA by $\hat{V}_k(s_k)$. Write down a decision rule for a policy $\hat{\pi}$ that uses the VFA:

Directy Lookaheads (DLAs) establish the value of being in a state by optimizing over some approximation of what lies ahead. DLAs can be useful when a decision now naturally requires a tentative plan for the future. Also, try DLAs when other approaches fail.

DLA Example: Truncated Decision Trees. Although backward induction across the full decision tree is often intractable, working with some portion of it may be possible. For instance, look two steps ahead, then truncate the tree. The decision rule chooses the action with the best total cost across the two-stage tree. Draw the truncated tree.

DLA Example: "A Rollout Algorithm Framework for Heuristic Solutions to Finite-Horizon Stochastic Dynamic Programs," Goodson, Thomas, & Ohlmann. *European Journal of Operational Research*, 2017:258, 216-229.

A **Rollout Policy** is a truncated decision tree plus an approximation of what's missing. Rollout policies use the value of heuristic policies to approximate the parts of the decision tree that are omitted. Denote by $\bar{\pi}$ a heuristic policy and by $V_k^{\bar{\pi}}(s_k)$ the value of the heuristic policy from state s_k. Draw and write out the one-step rollout policy decision rule.

4 Dual Bounds

Motivation. How good is a policy? We can gauge quality by comparing to benchmarks, but this leaves open the question of how much better we can do. A dual bound can help answer this question.

Dual Bound. A dual bound is some number z such that

$$z \geq V^\star = \max_{\pi \in \Pi} \mathbb{E}\left[\sum_{k=0}^{K} C_k(s_k, X_k^\pi(s_k)) \Big| s_0\right]$$

Optimality Gap. When we don't know the value of an optimal policy, we rely on the optimality gap to gauge quality. Let $V^\pi (\leq V^\star)$ be the value of our best heuristic policy:

A dual bound can be obtained by relaxing some part of the dynamic program, then optimally solving the resulting problem. Any part of the model can be relaxed: states, actions, information, transitions, contributions.

Perfect Information Relaxation. One way to calculate a dual bound is to reveal uncertainty before decisions are made. How does the decision tree change when information is perfect? Draw the modified tree, then write down the dual bound.

A perfect information relaxation is often tractable because

5 Learn More

Modeling and Policy Classes. Chapters 9, 11, 12, 13, and 16 of
"Reinforcement Learning and Stochastic Optimization: A Unified Framework for Sequential Decisions" by Warren Powell (2022).

Theory.
"Markov Decision Processes: Discrete Stochastic Dynamic Programming," by Martin Puterman (2005).

"Dynamic Programming and Optimal Control Vol. 1 and Vol. 2," by Dimitri Bertsekas (2017).

Policy Function Approximation Example.
"Same-Day Delivery with Heterogeneous Fleets of Drones and Vehicles," Ulmer & Thomas. *Networks*, 2018:72, 475-505.

Cost Function Approximation Example.
"Binary Driver-Customer Familiarity in Service Routing," Ulmer, Nowak, Mattfeld, & Kaminski. *European Journal of Operational Research*, 2020:286, 477-493.

Value Function Approximation Example.
"Meso-Parametric Value Function Approximation for Dynamic Customer Acceptances in Delivery Routing," Ulmer & Thomas. *European Journal of Operational Research*, 2020:285, 183-195.

Direct Lookahead Example.
"A Rollout Algorithm Framework for Heuristic Solutions to Finite-Horizon Stochastic Dynamic Programs," Goodson, Thomas, & Ohlmann. *European Journal of Operational Research*, 2017:258, 216-229.

Information Relaxations.
"Information Relaxations and Duality in Stochastic Dynamic Programs: A Review and Tutorial," Brown & Smith. *Foundationas and Trends in Optimization*, 2022:5:3, 246-339.