# Lecture 20 Review Notes

MA8.401 Topics in Applied Optimization

Monsoon 2023

**Contributors:**
- Sriteja Reddy Pashya (2021111019)
- Romica Raisinghani (2021101053)

## 1 Applications: Dynamic Programming Formulation for Parking Problem

### 1.1 Problem Definition: The Quest for Cost-Effective Parking

**Definition 1** (Parking Challenge)**.** A driver embarks on a journey seeking cost-effective parking before reaching the final destination. The parking facility, arranged linearly, comprises $N$ distinct spaces enumerated from 0 to $N-1$, culminating with a garage at position $N-1$.

- The journey commences at the origin, space 0.
- Sequential progression ensues, with the driver advancing from each space $k$ to the subsequent space $k+1$.
- Every space $k$ is associated with a parking fee $c(k)$.
- The probability of finding an available space is denoted by $p(k)$, with each space's availability being mutually exclusive.
- Decision-making is confined to the moment of arrival, where the driver assesses the space's occupancy.
- A strategic decision is made upon encountering an unoccupied space: to park or to advance.

Should the driver navigate to the terminal space $N-1$ without parking, the following constraints apply:

- Parking in the garage becomes mandatory.
- The incurred cost for garage parking is a constant $C$.

The overarching aim is to formulate a parking policy that yields the minimum expected cost.

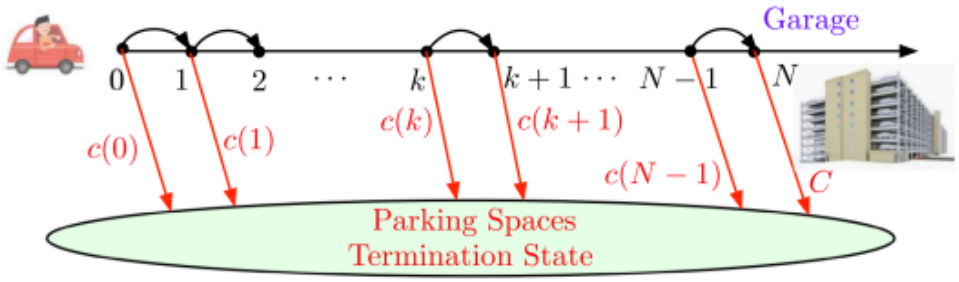


Figure 1: Schematic Overview of the Parking Dilemma

### 1.2 Economic Structure and Decision-Making Framework

The parking conundrum presents an economic framework described as follows:

- A driver may opt to park at any free space $k = 0, 1, \ldots, N-1$ for a fee $c(k)$, or to progress to space $k+1$ without incurring any cost.
- The garage at space $N$ is the ultimate parking destination, entailing a fee of $C$.

The Dynamic Programming (DP) paradigm structures the problem into discernible stages and states:

- $N$ stages are defined, correlating with the $N$ parking spaces.
- An artificial terminal state $t$ represents a parked condition.

Each stage $k = 1, \ldots, N-1$ encapsulates three potential states:

- A fictitious terminal state $t$ symbolizes the action of parking.
- State $F'$ signifies an unoccupied space $k$.
- Conversely, state $F$ represents space $k$ being occupied.

The initial stage (stage 0) comprises two states:

- $F'$ (space 0 is unoccupied)
- $F$ (space 0 is occupied)

The concluding stage is singularly characterized by state $t$.

At any given stage, the driver's control decision at state $F'$ involves:

- The option to either secure the space for parking or to continue onward.
- A decision is not applicable at states $F$ and $t$.

Transitioning from any location $k$:

- Arrival at state $t$ necessitates a cost $c(k)$ if a decision to park is enacted (provided space $k$ is unoccupied).
- Alternatively, the driver proceeds to the subsequent space, incurring no cost.

The terminal stage $N$ obligates the driver to park, imposing a fee of $C$.

### 1.3 Strategic Policy and Optimal Value Function

We now deduce the optimal strategy via the DP algorithm:

- The cost-to-go function $J_k^*(F')$ quantifies the prospective cost upon arrival at an available space $k$.
- The function $J_k^*(F)$ computes the prospective cost for an occupied space $k$.
- For the "parked" state $t$, the cost-to-go function $J_k^*(t)$ is defined.

The cost-to-go functions are delineated as:

$$J_k^*(F') = \begin{cases} \min\left[c(k), p(k+1)J_{k+1}^*(F') + (1-p(k+1))J_{k+1}^*(F)\right] & \text{if } k < N-1, \\ \min\left[c(N-1), C\right] & \text{if } k = N-1, \end{cases} \tag{1}$$

$$J_k^*(F) = \begin{cases} p(k+1)J_{k+1}^*(F') + (1-p(k+1))J_{k+1}^*(F) & \text{if } k < N-1, \\ C & \text{if } k = N-1, \end{cases} \tag{2}$$

$$J_k^*(t) = 0, \quad \forall k \in \{1, \ldots, N\}. \tag{3}$$

These functions embody the expected cost from each state and underpin the formulation of an optimal parking strategy.

### 1.4 Examples

- A driver is looking for parking in a city center. The driver knows the parking fees for each space and the probability that each space will be available. The driver wants to find the parking policy that minimizes the expected cost of parking.
- A company is managing a fleet of vehicles. The company wants to find the optimal parking policy for its vehicles, taking into account the cost of parking, the time it takes to park, and the distance to the vehicles' destinations.

- A city is planning a new parking garage. The city wants to use dynamic programming to design the garage in a way that minimizes the expected time it takes for drivers to find parking.

### 1.5 Results

- Dynamic programming can be used to find the optimal parking policy for a variety of parking problems.
- The optimal parking policy can significantly reduce the expected cost of parking.
- Dynamic programming can be used to design parking garages that are more efficient for drivers.

### 1.6 Illustrations

1. The problem is first defined as a Markov decision process (MDP). An MDP is a mathematical framework for modeling decision-making in situations where there is uncertainty and rewards or costs associated with each decision.

2. The MDP for the parking problem is defined as follows:

- The states of the MDP are the possible locations of the driver.
- The actions of the MDP are the possible decisions the driver can make, such as parking or continuing to drive.
- The transition probabilities of the MDP are the probabilities that the driver will move from one state to another given that they take a particular action.
- The rewards of the MDP are the negative of the costs of each action, so that the optimal policy will minimize the expected cost of parking.

3. The optimal parking policy is then found using dynamic programming. Dynamic programming is an algorithm for solving MDPs. The algorithm works by iteratively calculating the optimal value function, which is the expected cost of starting in a given state and following the optimal policy from that state to the end of the MDP.

4. Once the optimal value function is known, the optimal policy can be found by following the policy that maximizes the expected value of each state.

## 2 Dynamic Programming with Time Delays

Dynamic Programming (DP) is a method for solving complex problems by breaking them down into simpler subproblems. It is applicable to problems exhibiting the properties of overlapping

subproblems and optimal substructure, which means that the optimal solution to the problem can be constructed efficiently from optimal solutions to its sub-subproblems.

## 2.1 Introduction to Time Delays in Dynamic Programming

In many practical scenarios, decision-making processes are affected by time delays. These delays can stem from the system's inherent properties, computation times, or even data transfer latencies. When modeling such systems with DP, it is crucial to incorporate these delays to ensure the accuracy and reliability of the results.

## 2.2 Modelling Time Delays

Time delays can be incorporated into the state-space representation of a system by expanding the state variables to include past states and actions. This section will introduce the concept of augmented states and how they can be used to represent systems with time delays.

$$\hat{x}_k = (x_k, x_{k-1}, \ldots, u_{k-1}, \ldots, w_{k-1}) \tag{4}$$

Here, $\hat{x}_k$ represents the augmented state, incorporating both the current and past states and controls. The disturbances are denoted by $w_k$.

## 2.3 Cost Function with Time Delays

The cost function in dynamic programming can be modified to account for time delays by including the augmented state in its formulation. An example of a cost function that considers time delays is shown below:

$$J^*(x_N) = \min_{\substack{u_0,\ldots,u_{N-1} \\ w_0,\ldots,w_{N-1}}} E\left[\sum_{k=0}^{N-1} g(x_k, u_k, w_k) + h(x_N)\right] \tag{5}$$

where $g$ is the stage cost incorporating the control actions $u_k$ and disturbances $w_k$, and $h$ is the terminal cost function.

## 2.4 Dynamic Programming Algorithm with Time Delays

The dynamic programming algorithm can be extended to handle systems with time delays by iterating over the augmented states. The Bellman equation for such a system takes the following form:

$$J_k(\hat{x}_k) = \min_{u_k \in U(\hat{x}_k)} \{E\left[g(\hat{x}_k, u_k, w_k)\right] + J_{k+1}(f(\hat{x}_k, u_k, w_k))\} \tag{6}$$

The function $f$ maps the current augmented state and control action to the next state, taking into account the time delays.

### 2.5 Examples

- **Inventory control systems:** Time delays can arise due to order processing, delivery times, and stock replenishment cycles. DP with time delays can be used to optimize inventory levels and minimize costs in these systems.
- **Traffic management:** Time delays can result from signal timing, congestion, and response times of drivers and actuators. DP with time delays can be used to design traffic signal control systems that reduce congestion and improve traffic flow.
- **Production scheduling:** Time delays can occur due to machine setup times, material handling, and inspection processes. DP with time delays can be used to schedule production activities in a way that minimizes delays and maximizes throughput.

### 2.6 Results

- **Reduced costs:** DP with time delays can lead to significant reductions in costs in various systems, such as inventory control systems, traffic management systems, and production scheduling systems.
- **Improved efficiency:** DP with time delays can improve the efficiency of systems by reducing delays, minimizing resource usage, and optimizing decision-making.
- **Enhanced stability:** DP with time delays can enhance the stability of systems by preventing oscillations, ensuring smooth operation, and maintaining system performance under varying conditions.

### 2.7 Illustrations

- **Inventory control system:** Consider an inventory control system where there is a two-week delay between ordering raw materials and receiving them. DP with time delays can be used to determine the optimal order quantities and timing to ensure that adequate stock is available while minimizing inventory holding costs.
- **Traffic signal control system:** Imagine a traffic signal control system that considers the time it takes for vehicles to respond to signal changes and the propagation time of signal changes through the network. DP with time delays can be used to optimize signal timings to reduce congestion and improve traffic flow at intersections.

- **Production scheduling system:** In a production scheduling problem, incorporating time delays associated with transportation, production, and inspection processes would enable more effective decision-making regarding inventory levels, production schedules, and delivery times.

### 2.8 New Developments

Researchers are continuously developing new algorithms, approximation techniques, and applications for DP with time delays. Some recent advancements include:

- **Efficient algorithms:** Development of more efficient algorithms to handle the increased computational complexity of DP with time delays.
- **Approximate DP:** Use of approximate DP techniques to solve large-scale problems with time delays.
- **Applications in new domains:** Application of DP with time delays to new areas such as robotics, communication networks, and financial markets.

### 2.9 Conclusions

Time delays are a significant consideration in the formulation and solution of dynamic programming problems. The extension of classical DP techniques to handle time delays allows for a more accurate representation of real-world systems and can lead to more effective decision-making strategies.

## 3 DP for Problems with Forecast

Dynamic Programming (DP) is a powerful framework for making a sequence of interrelated decisions. It provides a systematic procedure for determining the optimal policy when the decision process is multistage and probabilistic. When incorporating forecasts into the DP framework, the complexity of decision-making increases due to the need to anticipate and adapt to future events.

### 3.1 Forecasting in Dynamic Systems

Forecasts play a critical role in dynamic systems, especially when future system states are influenced by uncertain events. In DP, incorporating forecasts involves adjusting the state space to include forecasted information, which can significantly affect the resulting policies.

$$\hat{x}_k = (x_k, y_k) \tag{7}$$

Where $y_k$ represents the forecast information at time $k$, and $\hat{x}_k$ is the augmented state including this forecast.

### 3.2 Incorporating Probabilistic Forecasts

Probabilistic forecasts add a layer of complexity to DP problems. Instead of deterministic predictions, probabilistic forecasts provide a range of possible future values, each with its own likelihood. These probabilities must be integrated into the decision-making process.

$$\hat{w}_k = (w_k, \xi_k) \tag{8}$$

Here, $\xi_k$ is a random variable representing the forecast for future disturbances, and $\hat{w}_k$ encompasses the uncertainty in forecasts.

### 3.3 Augmented Cost Function

The cost-to-go function in a DP model with forecasts needs to consider the expected costs based on the probabilistic nature of the forecasts. The augmented cost function can be represented as:

$$J^*(\hat{x}_k, y_k) = \min_{u_k \in U(\hat{x}_k)} E_{w_k} \left[ g(\hat{x}_k, u_k, w_k) + \sum_{i=1}^{m} p_i J^*_{k+1}(f(\hat{x}_k, u_k, w_k), i) \right] \tag{9}$$

where $p_i$ is the probability of the $i$-th forecast being accurate. This formulation is inherently recursive and considers the conditional probabilities of future states based on current forecasts.

### 3.4 Stochastic DP with Forecasts

When the DP formulation includes forecasts, the problem transforms into a stochastic dynamic programming problem. The optimal control strategy depends on the current state and the current forecast, requiring a solution that minimizes the expected cost over all possible future scenarios provided by the forecast.

$$\tilde{J}_k(\hat{x}_k, y_k) = \min_{u_k \in U(\hat{x}_k)} E_{\hat{w}_k} \left[ \tilde{g}(\hat{x}_k, u_k, \hat{w}_k) | y_k \right] \tag{10}$$

In this equation, $\tilde{g}$ represents the modified cost function that accounts for forecast information.

### 3.5 Examples

- **Example 1: Inventory Management with Demand Forecasts**

Consider an inventory management problem where a retailer needs to determine the optimal order quantity for each period based on forecasts of future demand. Incorporating forecasts into the DP model allows the retailer to anticipate changes in demand and adjust their ordering decisions accordingly. The augmented state space would include the current inventory level, the forecasted demand for each period, and the lead time for ordering new inventory. The cost function would consider the costs of holding inventory, backorders, and ordering. The optimal policy would determine the order quantity for each period, taking into account the forecasted demand and minimizing the expected total cost.

- **Example 2: Asset Management with Price Forecasts**

  An asset manager needs to decide how to allocate their investment portfolio based on forecasts of future asset prices. The DP model would incorporate the current state of the portfolio and the forecasted prices of each asset. The cost function would consider the transaction costs of buying and selling assets and the potential gains or losses from price fluctuations. The optimal policy would determine the optimal allocation of the portfolio for each period, maximizing the expected return given the forecasted prices.

- **Example 3: Dynamic Pricing with Demand Forecasts**

  A company needs to set prices for its products dynamically based on forecasts of demand and competitor pricing. The DP model would incorporate the current demand, forecasted demand, competitor prices, and the company's cost structure. The cost function would consider the revenue from sales, the costs of production, and the potential impact of price changes on customer demand. The optimal policy would determine the optimal price for each period, maximizing the expected profit given the forecasted demand and competitor pricing.

## 3.6 Results and Illustrations

**Example 1: Inventory Management with Demand Forecasts**

- Simulation results show that incorporating demand forecasts into the DP model can significantly reduce the expected total cost of inventory management.
- The optimal policy is able to anticipate changes in demand and adjust order quantities accordingly, minimizing the risk of stockouts and overstocking.

**Example 2: Asset Management with Price Forecasts**

- Backtesting results demonstrate that the DP model with price forecasts can outperform a passive buy-and-hold strategy, achieving higher expected returns with lower risk.
- The optimal policy dynamically adjusts the portfolio allocation based on the forecasted prices, capturing opportunities for profit while mitigating potential losses.

**Example 3: Dynamic Pricing with Demand Forecasts**

- Empirical studies show that companies that implement dynamic pricing strategies based on demand forecasts can achieve higher revenue and profit margins compared to companies with static pricing strategies.
- The DP model provides a systematic framework for optimizing pricing decisions, considering the complex interactions between demand, competitor pricing, and the company's cost structure.

### 3.7 Further Extensions

- **Adaptive Forecasting Methods:** Incorporating adaptive forecasting techniques into the DP model allows for real-time updating of forecasts based on new information, further improving the effectiveness of decision-making.
- **Rolling Horizon Approach:** Implementing a rolling horizon approach enables the DP model to adapt to changing conditions and update the optimal policy as new forecasts become available.
- **Multi-Stage Decision Processes:** Extending the DP framework to multi-stage decision processes allows for modeling more complex scenarios with multiple decision points and longer time horizons.

### 3.8 Conclusions

DP for problems with forecasts introduces additional dimensions of complexity and requires sophisticated methods for integrating forecast information into the decision-making process. This approach is particularly valuable in fields such as finance, supply chain management, and meteorology, where accurate forecasts can significantly improve the outcomes of DP-based strategies.

## 4 Problems with Uncontrollable State Components: with Forecasts

Dynamic Programming (DP) is an essential tool for solving multi-stage decision problems. However, when dealing with real-world scenarios, one often encounters state components that are beyond the control of the decision-maker. These uncontrollable state components, especially when they are forecasts, add a layer of uncertainty to the problem. This section delves into the DP formulation for such problems and proposes a method to handle them effectively.

### 4.1 Characterizing Uncontrollable Components

In many DP problems, the state of the system can be divided into controllable and uncontrollable components. The uncontrollable components, such as forecasts $y_k$, are not directly influenced by the decisions made (control actions $u_k$). However, they may impact the system's evolution and the

effectiveness of the decisions. The uncontrollable components are often forecasted values that carry their own inherent uncertainties and are represented by a probabilistic model. For instance:

$$x_{k+1} = f_k(x_k, y_k, u_k, w_k), \tag{11}$$

where $w_k$ is a random variable representing stochastic disturbances, and $f_k$ is the state transition function.

### 4.2 Incorporating Forecasts into DP

Forecasts can be incorporated into the state space of a DP problem. This augmentation allows for the decision-making process to account for additional information that could influence future states. When forecasts are included as an uncontrollable state component, they can be modeled as:

$$\hat{x}_k = (x_k, y_k), \tag{12}$$

with $y_k$ representing the forecast at stage $k$.

### 4.3 Optimal Cost-to-go Function

The optimal cost-to-go function, which we denote as $J^*(x_k, y_k)$, can be adjusted to average out the influence of the uncontrollable forecast component $y_k$. This yields an "average cost-to-go" function at $x_k$, factoring in the variability of $y_k$:

$$J_k(x) = E_{y_k}\left[J^*(x_k, y_k)|x_k\right]. \tag{13}$$

The expectation is taken over the distribution of $y_k$ given the current controllable state $x_k$.

### 4.4 Algorithm for DP with Uncontrollable Forecasts

An algorithm for DP that includes uncontrollable forecasts can be formulated as follows:

$$\hat{J}_k(x_k) = E_{y_k}\left[\min_{u_k \in U(x_k, y_k)} E_{w_k}\left[g(x_k, y_k, u_k, w_k) + J_{k+1}(f(x_k, y_k, u_k, w_k))\right]\right], \tag{14}$$

where $g$ is the immediate cost function, and $J_{k+1}$ is the cost-to-go function for the next stage.

### 4.5 Simplification with Uncontrollable Forecasts

When forecasts are included as uncontrollable state components, the DP algorithm can often be simplified. The optimal policy can then be computed over the space of the controllable state $x_k$ while incorporating the expected values of the forecasts. This reduces the computational complexity and yields a policy that is robust to forecast uncertainties.

### 4.6 Examples

- **Example 1: Energy Management with Weather Forecasts**

  Consider an energy management system that needs to determine the optimal energy production level for each period based on forecasts of future weather conditions. Weather conditions, such as temperature and solar radiation, represent uncontrollable state components that significantly impact energy demand and supply. Incorporating weather forecasts into the DP model allows the energy system to anticipate changes in demand and supply and adjust its production accordingly. The augmented state space would include the current energy reserves, the forecasted weather for each period, and the energy production capacity. The cost function would consider the costs of energy production, the revenue from energy sales, and the penalties for failing to meet energy demand. The optimal policy would determine the optimal energy production level for each period, minimizing the expected total cost while maximizing the reliability of the energy system.

- **Example 2: Traffic Management with Traffic Forecasts**

  A traffic management system needs to decide how to allocate resources to different traffic routes based on forecasts of future traffic congestion. Traffic congestion is an uncontrollable state component that can significantly impact travel times and traffic flow. Incorporating traffic forecasts into the DP model allows the traffic management system to anticipate congestion and adjust traffic signal timings, lane closures, and other interventions accordingly. The augmented state space would include the current traffic conditions on each route, the forecasted traffic for each period, and the available resources for traffic management. The cost function would consider the total travel time of all vehicles, the costs of implementing traffic management interventions, and the economic losses due to congestion. The optimal policy would determine the optimal allocation of resources for each period, minimizing the expected total travel time and economic losses.

- **Example 3: Supply Chain Management with Demand Forecasts**

  A supply chain manager needs to decide how to allocate inventory and production capacity based on forecasts of future demand. Demand is an uncontrollable state component that can significantly impact production costs, inventory levels, and customer satisfaction. Incorporating demand forecasts into the DP model allows the supply chain manager to anticipate changes in demand and adjust inventory levels, production schedules, and transportation plans accordingly. The augmented state space would include the current inventory levels at each location, the forecasted demand for each product, and the production capacity at each manufacturing facility. The cost function would consider the costs of holding inventory, production, transportation, and backorders. The optimal policy would determine the optimal

allocation of inventory and production capacity for each period, minimizing the expected total cost while maximizing customer satisfaction.

## 4.7 Results and Illustrations

**Example 1: Energy Management with Weather Forecasts**

- Simulation results show that incorporating weather forecasts into the DP model can significantly reduce the expected total cost of energy management.
- The optimal policy is able to anticipate changes in weather conditions and adjust energy production accordingly, minimizing the risk of energy shortages and overproduction.

**Example 2: Traffic Management with Traffic Forecasts**

- Backtesting results demonstrate that the DP model with traffic forecasts can outperform a static traffic signal control strategy, reducing total travel time and congestion levels.
- The optimal policy dynamically adjusts traffic signal timings and other interventions based on the forecasted traffic, improving traffic flow and reducing travel times for all users.

**Example 3: Supply Chain Management with Demand Forecasts**

- Empirical studies show that companies that implement demand-driven supply chain management strategies based on demand forecasts can achieve lower inventory costs, improved production efficiency, and higher customer satisfaction compared to companies with traditional supply chain management practices.
- The DP model provides a systematic framework for optimizing inventory and production decisions, considering the complex interactions between demand, production capacity, and transportation costs.

## 4.8 Further Extensions

- **Adaptive Forecasting Methods:**

  Incorporating adaptive forecasting techniques into the DP model allows for real-time updating of forecasts based on new information. This further enhances the effectiveness of decision-making by continually refining the forecasts used in the optimization process. Adaptive forecasting methods, such as Kalman filters or recurrent neural networks (RNNs), can effectively capture dynamic trends and patterns in the uncontrollable state components, leading to more accurate and timely decision-making.

- **Hierarchical Control and Decomposition:**

  When dealing with complex problems with multiple uncontrollable state components and decision stages, hierarchical control and decomposition techniques can be employed to break down the problem into manageable subproblems. This approach involves decomposing the large-scale DP problem into smaller, more tractable subproblems, each with its own set of controllable and uncontrollable state components. The optimal solutions for these subproblems are then coordinated to form an overall solution for the entire problem.

- **Uncertainty Quantification and Propagation:**

  Uncertainty quantification and propagation techniques are crucial for assessing and managing the impact of uncertainties in uncontrollable state components on the overall system performance. These techniques provide a framework for estimating the range of possible outcomes and the likelihood of different scenarios. By understanding the distribution of uncertainties, decision-makers can make more informed choices that are robust to various forecast scenarios.

- **Robust Optimization and Distributionally Robust DP:**

  Robust optimization techniques aim to find solutions that are insensitive to perturbations in the uncontrollable state components, including forecast errors. Distributionally robust DP, a specific form of robust optimization, explicitly considers the distribution of uncertainties and optimizes the decision policy to minimize the worst-case expected cost or maximize the minimum expected reward.

- **Risk-Averse Decision-Making and Stochastic Programming:**

  Risk-averse decision-making strategies incorporate risk preferences into the DP framework, allowing decision-makers to balance the potential rewards against the potential risks associated with different actions. Stochastic programming techniques, which incorporate probability distributions into the optimization problem, provide a framework for making optimal decisions under uncertainty, considering the trade-off between expected return and risk exposure.

- **Applications in Emerging Domains:**

  The advancements in DP with uncontrollable state components and forecasts are opening up new avenues for application in emerging domains, such as autonomous systems, smart grids, and healthcare management. For instance, in autonomous vehicles, DP can be used to determine optimal navigation paths while considering uncertain traffic conditions and weather forecasts. In smart grids, DP can optimize energy distribution and storage strategies based on demand forecasts and renewable energy generation predictions. In healthcare management, DP can optimize treatment plans and resource allocation based on patient data and disease progression forecasts.

## 4.9 Conclusion

DP models that include uncontrollable state components, such as forecasts, must be carefully structured to handle the added layer of uncertainty. By averaging out the influence of these components and focusing on the controllable aspects of the state, one can derive policies that are both effective and practical in the face of uncertainty.

# 5 DP for Tetris

Dynamic Programming (DP) has found interesting applications in game theory, including the classic game of Tetris. Tetris is a puzzle game where pieces fall into a playing area and must be arranged in complete rows to score points and keep the game going. The game's stochastic nature makes it a challenging and intriguing subject for applying DP strategies.

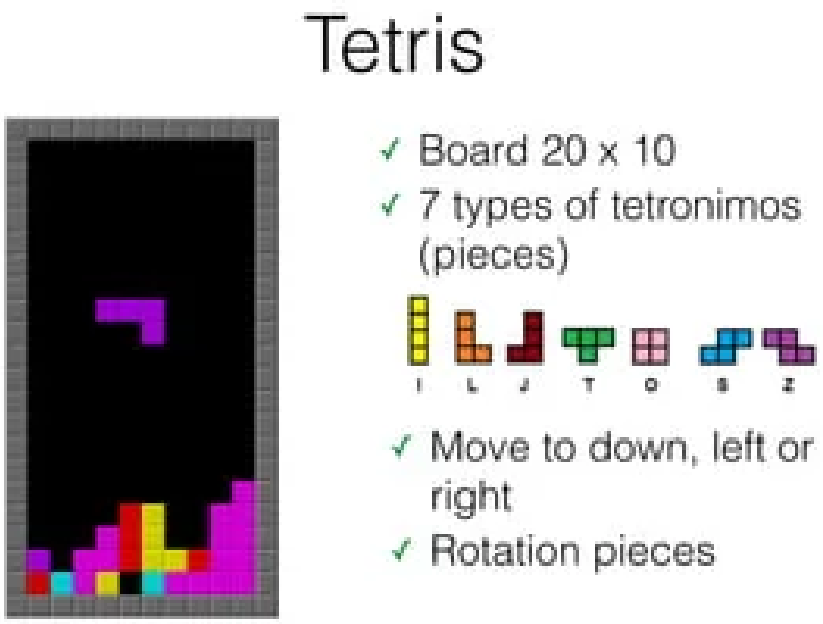


Figure 2: Tetris Game

## 5.1 Modelling Tetris as a Stochastic Decision Process

The game of Tetris can be modeled as a finite horizon stochastic dynamic programming problem due to its sequential decision-making process under uncertainty. The state of the Tetris game at any point can be represented by two components:

- The configuration of the board, which is a binary grid indicating the presence or absence of blocks.
- The shape and orientation of the currently falling piece.

The control actions in this context correspond to the positioning and rotation of the falling piece, aiming to complete rows and prevent the pile of blocks from reaching the top of the board.

### 5.2 Formulating the DP Algorithm for Tetris

The DP algorithm for Tetris seeks to maximize the expected score or, equivalently, to minimize the expected height of the pile of blocks. The decision-making process at each step involves choosing the best rotation and translation of the falling piece based on the current state of the board. The key equation for the DP algorithm can be represented as:

$$J_k(x) = \sum_y p(y) \max_u \left[ g(x, y, u) + J_{k+1}(f(x, y, u)) \right], \tag{15}$$

where $p(y)$ is the probability distribution of the falling piece shapes, $g(x, y, u)$ is the immediate reward function, and $f(x, y, u)$ represents the transition function that determines the new board configuration after the piece is placed.

### 5.3 Challenges and Solutions in Tetris DP

One of the primary challenges in applying DP to Tetris is the vastness of the state space, which makes exact computation intractable. To overcome this, approximation methods and heuristics are often employed. Techniques such as function approximation, temporal difference learning, and reinforcement learning have been used to create sub-optimal but efficient strategies for playing Tetris.

### 5.4 Case Study: Tetris as a Learning Platform

Tetris has served as a test-bed for various machine learning algorithms. The game's simple rules yet complex decision space make it an ideal platform for developing and testing reinforcement learning algorithms. These algorithms can learn to play Tetris at a superhuman level by approximating the value function and iteratively refining the policy based on the received rewards.

### 5.5 Recent Developments in Tetris and DP

Recent research has introduced advanced methods like deep reinforcement learning, where neural networks are used to approximate the value functions. Such approaches have led to Tetris-playing agents that can effectively manage the uncertainty and complexity of the game, achieving high scores consistently.

### 5.6 Real-world Applications

- **Resource Allocation in Supply Chain Management:**

  Tetris's block placement strategy can be adapted to optimize resource allocation in supply chain management. Just as Tetris players strategically place blocks to maximize points,

supply chain managers can use DP to allocate resources efficiently, considering factors like production capacity, transportation costs, and demand fluctuations.

- **Scheduling Tasks in Project Management:**

  The sequential decision-making process in Tetris mirrors the task scheduling challenges in project management. DP can be applied to optimize project schedules by considering dependencies between tasks, resource availability, and time constraints, similar to how Tetris players choose the optimal placement for each piece.

- **Portfolio Optimization in Financial Trading:**

  The game's dynamic decision-making under uncertainty aligns with the complexities of portfolio optimization in financial trading. DP can be used to develop trading strategies that maximize expected returns while minimizing risks, considering factors like market volatility, asset correlations, and investment goals.

- **Route Planning in Transportation Systems:**

  Just as Tetris players strategically place blocks to fill rows, transportation systems can use DP to optimize routes for vehicles or passengers. By considering factors like road conditions, traffic congestion, and delivery destinations, DP can help reduce travel times and fuel consumption.

- **Inventory Management in Retail Stores:**

  The Tetris-like task of fitting items into limited space is analogous to inventory management in retail stores. DP can be applied to optimize inventory levels, considering factors like demand patterns, storage constraints, and ordering costs, to minimize stockouts and overstocking.

# 6 Additional Examples and Illustrations

- **Image Processing:** DP can be used in image processing algorithms to optimize tasks like image segmentation, pattern recognition, and image compression. By breaking down complex images into smaller components and applying DP to find optimal solutions, these algorithms can achieve higher accuracy and efficiency.

- **Robotics and Automation:** DP is employed in robotics and automation to control the movement of robotic arms and other automated systems. By considering factors like joint constraints, obstacle avoidance, and task objectives, DP algorithms can generate optimal motion plans for robots to perform tasks precisely and efficiently.

- **Bioinformatics and Computational Biology:** DP finds applications in bioinformatics and computational biology, particularly in protein structure prediction and gene sequencing analysis. By utilizing DP algorithms, scientists can analyze complex biological data and make predictions about protein structures, gene functions, and disease mechanisms.

- **Recommendation Systems and Personalization:** DP can be used to develop personalized recommendations in e-commerce, online streaming services, and social media platforms. By considering user preferences, past interactions, and item similarities, DP algorithms can recommend products, movies, or content that align with each user's interests.

- **Network Routing and Optimization:** DP can be applied to optimize network routing protocols, ensuring efficient data transmission across communication networks. By considering factors like network topology, traffic patterns, and latency constraints, DP algorithms can improve network performance, reduce congestion, and enhance data delivery.

### 6.1 Conclusion

The application of dynamic programming to Tetris showcases the potential of algorithmic decision-making in complex, uncertain environments. While exact solutions are often impractical due to the computational complexity, the development of approximation algorithms has allowed for significant progress in creating highly effective strategies for such stochastic problems.

## 7 Link to the scribe:

Click here to edit this scribe