15.093 Optimization Methods

Lecture 16: Dynamic Programming
1 Outline

1. The knapsack problem
2. The traveling salesman problem
3. The general DP framework
4. Bellman equation
5. Optimal inventory control
6. Optimal trading
7. Multiplying matrices

2 The Knapsack problem

\[
\text{maximize } \sum_{j=1}^{n} c_j x_j \\
\text{subject to } \sum_{j=1}^{n} w_j x_j \leq K \\
x_j \in \{0,1\}
\]

Define

\[
C_i(w) = \text{maximize } \sum_{j=1}^{i} c_j x_j \\
\text{subject to } \sum_{j=1}^{i} w_j x_j \leq w \\
x_j \in \{0,1\}
\]

2.1 A DP Algorithm

- \(C_i(w)\): the maximum value that can be accumulated using some of the first \(i\) items subject to the constraint that the total accumulated weight is equal to \(w\)

- Recursion

\[
C_{i+1}(w) = \max \{ C_i(w), C_i(w - w_{i+1}) + c_{i+1} \}
\]

- By considering all states of the form \((i, w)\) with \(w \leq K\), algorithm has complexity \(O(nK)\)
3 The TSP

- $G = (V, A)$ directed graph with $n$ nodes
- $c_{ij}$ cost of arc $(i, j)$
- Approach: choice of a tour as a sequence of choices
- We start at node 1; then, at each stage, we choose which node to visit next.
- After a number of stages, we have visited a subset $S$ of $V$ and we are at a current node $k \in S$

3.1 A DP algorithm

- $C(S, k)$ be the minimum cost over all paths that start at node 1, visit all nodes in the set $S$ exactly once, and end up at node $k$
- $(S, k)$ a state; this state can be reached from any state of the form $(S \setminus \{k\}, m)$, with $m \in S \setminus \{k\}$, at a transition cost of $c_{mk}$
- Recursion
  
  $$C(S, k) = \min_{m \in S \setminus \{k\}} \left( C(S \setminus \{k\}, m) + c_{mk} \right), \quad k \in S$$
  
  $$C(\{1\}, 1) = 0.$$  

- Length of an optimal tour is
  
  $$\min_k \left( C(\{1, \ldots, n\}, k) + c_{k1} \right)$$

- Complexity: $O(n^2 2^n)$ operations

4 Guidelines for constructing DP Algorithms

- View the choice of a feasible solution as a sequence of decisions occurring in stages, and so that the total cost is the sum of the costs of individual decisions.
- Define the state as a summary of all relevant past decisions.
- Determine which state transitions are possible. Let the cost of each state transition be the cost of the corresponding decision.
- Write a recursion on the optimal cost from the origin state to a destination state.

The most crucial step is usually the definition of a suitable state.
5 The general DP framework

- Discrete time dynamic system described by state $x_k$, $k$ indexes time.
- $u_k$ control to be selected at time $k$. $u_k \in U_k(x_k)$.
- $w_k$ randomness at time $k$
- $N$ time horizon
- Dynamics:
  \[ x_{k+1} = f_k(x_k, u_k, w_k) \]
- Cost function: additive over time
  \[ E \left( g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k) \right) \]

5.1 Inventory Control

- $x_k$ stock available at the beginning of the $k$th period
- $u_k$ stock ordered at the beginning of the $k$th period
- $w_k$ demand during the $k$th period with given probability distribution. Excess demand is backlogged and filled as soon as additional inventory is available.
- Dynamics
  \[ x_{k+1} = x_k + u_k - w_k \]
- Cost
  \[ E \left( R(x_N) + \sum_{k=0}^{N-1} (r(x_k) + cu_k) \right) \]

6 The DP Algorithm

- Define $J_k(x_k)$ to be the expected optimal cost starting from stage $k$ at state $x_k$.
- Bellman’s principle of optimality
  \[ J_N(x_N) = g_N(x_N) \]
  \[ J_k(x_k) = \min_{u_k \in U_k(x_k)} \left\{ g_k(x_k, u_k, w_k) + J_{k+1}(f_k(x_k, u_k, w_k)) \right\} \]
- Optimal expected cost for the overall problem: $J_0(x_0)$. 

7 Inventory Control

- If \( r(x_k) = ax_k^2, w_k \sim N(\mu_k, \sigma_k^2) \), then
  \[
  u_k^* = c_kx_k + d_k, \quad J_k(x_k) = b_kx_k^2 + f_kx_k + e_k
  \]
- If \( r(x_k) = p\max(0, -x_k) + h\max(0, x_k) \), then there exist \( S_k \):
  \[
  u_k^* = \begin{cases}
  S_k - x_k & \text{if } x_k < S_k \\
  0 & \text{if } x_k \geq S_k
  \end{cases}
  \]

8 Optimal trading

- \( \overline{S} \) shares of a stock to be bought within a horizon \( T \).
- \( t = 1, 2, \ldots, T \) discrete trading periods.
- Control: \( S_t \) number of shares acquired in period \( t \) at price \( P_t, t = 1, 2, \ldots, T \)
- Objective:
  \[
  \min E \left[ \sum_{t=1}^T P_t S_t \right], \quad \text{s.t. } \sum_{t=1}^T S_t = \overline{S}
  \]
- Dynamics:
  \[
  P_t = P_{t-1} + \alpha S_t + \epsilon_t
  \]
  where \( \alpha > 0, \epsilon_t \sim N(0,1) \)

8.1 DP ingredients

- State: \( (P_{t-1}, W_t) \)
  \( P_{t-1} \) price realized at the previous period
  \( W_t \) # of shares remaining to be purchased
- Control: \( S_t \) number of shares purchased at time \( t \)
- Randomness: \( \epsilon_t \)
- Objective:
  \[
  \min E \left[ \sum_{t=1}^T P_t S_t \right]
  \]
- Dynamics:
  \[
  P_t = P_{t-1} + \alpha S_t + \epsilon_t \quad W_t = W_{t-1} - S_{t-1}, \quad W_1 = \overline{S}, \quad W_{T+1} = 0
  \]
Note that \( W_{T+1} = 0 \) is equivalent to the constraint that \( \overline{S} \) must be executed by period \( T \)
8.2 The Bellman Equation

\[ J_t(P_{t-1}, W_t) = \min_{S_t} E_t \left[ P_t S_t + J_{t+1}(P_{t+1}) \right] \]

\[ J_T(P_{T-1}, W_T) = \min_{S_T} E_T[P_T W_T] = (P_{T-1} + \alpha W_T) W_T \]

Since \( W_{T+1} = 0 \) \( \Rightarrow S_T^* = W_T \)

8.3 Solution

\[ J_{T-1}(P_{T-2}, W_{T-1}) = \]

\[ = \min_{S_{T-2}} E_{T-1} \left[ P_{T-2} S_{T-2} + J_T(P_{T-1}, W_T) \right] \]

\[ = \min_{S_{T-2}} E_{T-1} \left[ (P_{T-2} + \alpha S_{T-2} + \epsilon_{T-1}) S_{T-2} + J_T \left( P_{T-2} + \alpha S_{T-2} + \epsilon_{T-1}, W_{T-1} - S_{T-2} \right) \right] \]

\[ S_{T-1}^* = \frac{W_{T-1}}{2} \]

\[ J_{T-1}(P_{T-2}, W_{T-1}) = W_{T-1}(P_{T-2} + \frac{3}{4} \alpha W_{T-1}) \]

Continuing in this fashion,

\[ S_{T-k}^* = \frac{W_{T-k}}{k + 1} \]

\[ J_{T-k}(P_{T-k-1}, W_{T-k}) = W_{T-k}(P_{T-k-1} + \frac{k + 2}{2(k + 1)} \alpha W_{T-k}) \]

\[ S_1^* = \frac{S}{T} \]

\[ J_1(P_0, W_1) = P_0 S + \frac{\alpha S^2}{2} \left( 1 + \frac{1}{T} \right) \]

\[ S_1^* = S_2^* = \cdots = S_T^* = \frac{S}{T} \]
8.4 Different Dynamics

\[
    P_t = P_{t-1} + \alpha S_t + \gamma X_t + \epsilon_t, \quad \alpha > 0
\]

\[
    X_t = \rho X_{t-1} + \eta_t, \quad X_1 = 1, \quad \rho \in (-1, 1)
\]

where \( \epsilon_t \sim N(0, \sigma^2) \) and \( \eta_t \sim N(0, \sigma^2) \)

8.5 Solution

\[
    S^*_T = \frac{W_{T-k}}{k+1} + \frac{\rho b_{k-1}}{2a_{k-1}} X_{T-k}
\]

\[
    J_{T-k}(P_{T-k-1}, X_{T-k}, W_{T-k}) = P_{T-k-1}W_{T-k} + a_k W^2_{T-k} + b_k X_{T-k}W_{T-k} + c_k X^2_{T-k} + d_k
\]

for \( k = 0, 1, \ldots, T - 1 \), where:

\[
    a_k = \frac{\alpha}{2} \left( 1 + \frac{1}{k+1} \right), \quad a_0 = \alpha
\]

\[
    b_k = \gamma + \frac{\alpha \rho b_{k-1}}{2a_{k-1}}, \quad b_0 = \gamma
\]

\[
    c_k = \rho^2 c_{k-1} - \frac{\rho^2 b^2_{k-1}}{4a_{k-1}}, \quad c_0 = 0
\]

\[
    d_k = d_{k-1} + c_{k-1} \sigma^2, \quad d_0 = 0
\]

9 Matrix multiplication

- Matrices: \( M_k : n_k \times n_{k+1} \)
- Objective: Find \( M_1 \cdot M_2 \cdots M_N \)
- Example: \( M_1 \cdot M_2 \cdot M_3 \); \( M_1 : 1 \times 10, M_2 : 10 \times 1, M_3 : 1 \times 10 \).

\( M_1 (M_2 M_3) \) 200 multiplications;

\( (M_1 M_2) M_3 \) 20 multiplications.

- What is the optimal order for performing the multiplication?
• $m(i, j)$ optimal number of scalar multiplications for multiplying $M_i \ldots M_j$.
• $m(i, i) = 0$
• For $i < j$:
  \[
m(i, j) = \min_{1 \leq k < j} (m(i, k) + m(k + 1, j) + n_k n_{k+1} n_{j+1})\]