Introduction to Dynamic Programming

A bad and painless presentation

Emanuele Concas
emanuele.concas@enpc.fr
Outline

Introduction
  History
  What is Dynamic Programming
  An example: TSP
  Types of Dynamic Programming

Deterministic Dynamic Programming
  Formulation
  Principle of Optimality
  Algorithm
  Extensions

More about DP
Who

- Invented by Richard Bellman
- RAND was hired by the USAF
- The SecDef was Wilson
- Need to hide the secret
  - Dynamic: it is dynamic
  - Programming: find an optimal program schedule
What

- Optimization approach
  - complex problem $\rightarrow$ sequence of simpler problem
- Decision are made in stage
- Decision may be not fully predictable

Key ideas

- A decision cannot be seen in isolation
  - Balance between cost now vs cost in the future
- Optimal subproblems
- Overlapping subproblems
An example: Travelling salesman problem

- Shortest Hamiltonian circuit in a graph
TSP brute-force

- Take all the possible combinations
- Calculate the cost

\[ A \rightarrow B \rightarrow C \rightarrow D \rightarrow A \]
\[ A \rightarrow B \rightarrow D \rightarrow C \rightarrow A \]
\[ A \rightarrow C \rightarrow B \rightarrow D \rightarrow A \]
\[ A \rightarrow C \rightarrow D \rightarrow B \rightarrow A \]
\[ A \rightarrow D \rightarrow B \rightarrow C \rightarrow A \]
\[ A \rightarrow D \rightarrow C \rightarrow B \rightarrow A \]

- Overlapping subproblems
Depart
min. cost = 13
min. path = ?

Arrive
Types of Dynamic Programming

Exact DP

- Can formulate any optimization problem
- Deterministic and stochastic
- Combinatorial (discrete)
- One decision maker and two player games
- Finite or infinite states
- BUT:
  - Curse of dimensionality
  - Need for a mathematical model
## Types of Dynamic Programming (cont.)

### Curse of dimensionality

- Linear in time
- Exponential in the states
- Exponential in the decision
- Exponential in the disturbances

### Example (w/o disturbances)

- 5 states, 5 controls with a horizon of 50
- State and controls discretized in 100 values \((100^5)\)
- Flops \(\approx O(10^{22})\)
- It takes 31709 years with a PC with \(10^{10}\) flops/s
Approximate DP

- Approximation
- Simulation
- Very problem dependent
Deterministic Dynamic Programming

System that evolves according to equations

\[ x_{k+1} = f_k(x_k, u_k), \quad k = 0, 1, \ldots, N - 1 \]  

where \( u_k \in U_k(x_k) \)

Cost function that is accumulated in time

\[ J(x_0; u_0, \ldots, u_{N-1}) = g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k) \]
Minimize (2), over all control sequences that satisfy the constraint

\[ J^*(x_0) = \min_{u_k \in U_k(x_k), k=0,...,N-1} J(x_0, u_0, \ldots, u_{N-1}) \]  

Formalize TSP

- States = partial tours (e.g. A, AB, ABC, ...)
- Controls = cities component (e.g. A, B, C, D)
- Constraints = visit all cities one time
- Cost function = cost between two cities
Principle of Optimality

The tail of an optimal sequence is optimal for the tail subproblem.
**Principle of Optimality (cont.)**

<table>
<thead>
<tr>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>- We want to go from Paris to Milano by car</td>
</tr>
<tr>
<td>- The optimal path pass through Lyon</td>
</tr>
</tbody>
</table>

Paris $\rightarrow$ Lyon $\rightarrow$ Milano

Tail optimal

Optimal sequence

- If we start our journey from Lyon, the optimal route to Milano is the same?

**Idea**

The optimal cost function can be constructed backwards

E. Concas · *Introduction to DP*
Algorithm

Algorithm idea

- Solve all the tail subproblems for all the states, using the solution of the previously sub-subproblems

- Given a $x_k$: take every $u_k \in U_k(x_k)$ and resolve the tail subproblem at state $x_{k+1} = f_k(x_k, u_k)$. We get the cost $g_k(x_k, u_k)$

- Minimize the cost by optimizing over the $u_k$
Deterministic Dynamic Programming  

Algorithm (cont.)

Algorithm

■ Start with final stage $N$

$$J^*_N(x_N) = g_N(x_N), \quad \forall x_N$$ (4)

■ For $k = 0, \ldots, N - 1$ and $\forall x_k$

$$J^*_k(x_k) = \min_{u_k \in U_k(x_k)} [g_k(x_k, u_k) + J^*_{k+1}(f_k(x_k, u_k))]$$

■ We go backward and resolve the tail subproblems

$$J_N(x_N), J_{N-1}(x_{N-1}), \ldots, J_0(x_0) = J^*_0(x_0)$$
Optimal control sequence

What we have

- We obtained the optimal cost function (min. cost in TSP)
- We need to find the optimal controls (route in TSP)
Construct the optimal control sequence

- Start from $x_0$, the first optimal control is
  \[
  u_0^* \in \arg \min_{u_0 \in U_0(x_0)} [g_0(x_0, u_0) + J_1^*(f_0(x_0, u_0))]
  \]

- We find $x_1^* = f_0(x_0, u_0^*)$

- We continue going forward for $k = 1, 2, \ldots, N - 1$
  \[
  u_k^* \in \arg \min_{u_k \in U_k(x_k^*)} [g_k(x_k^*, u_k) + J_{k+1}^*(f_k(x_k^*, u_k))]
  \]

- We find $x_{k+1}^* = f_k(x_k^*, u_k^*)$
Recap algorithms

Offline vs Online
- Offline: we compute the optimal cost functions with the Bellman equation
- Online: we compute the optimal control sequence when needed

Reminder
- The state space could be too large
- The cost function could be difficult to compute
- We can extend to approximate value function $\tilde{J}_k$
Extensions

**Stochastic DP**
- States affected by random parameter
- More difficult than deterministic

**Infinite horizon**
- Infinite number of stages
- Stationary system

**Stochastic partial state information**
- States not perfectly known
- Very difficult to solve
More about DP


This presentation was very inspired by all of these works.