Dynamic Programming + Data Structures
Reminders

- Homework B: Complexity & Recursion
  - Out: Thu, Nov. 8
  - Due: Tue, Nov. 20 at 11:59pm

- Quiz 1: Logic & Proofs; Computation
  - Mon, Nov. 19, in-class
  - Covers Lectures 1 – 6
Q&A
RECURSION
Example: Greedy Search

Goal:
- Search space consists of nodes and weighted edges
- Goal is to find the lowest (total) weight path from root to a leaf

Greedy Search:
- At each node, selects the edge with lowest (immediate) weight
- Heuristic method of search (i.e. does not necessarily find the best path)
Example: Greedy Search

Goal:
- Search space consists of nodes and weighted edges
- Goal is to find the lowest (total) weight path from root to a leaf

Greedy Search:
- At each node, selects the edge with lowest (immediate) weight
- Heuristic method of search (i.e. does not necessarily find the best path)
Example: Greedy Search

**Goal:**
- Search space consists of nodes and weighted edges
- Goal is to find the lowest (total) weight path from root to a leaf

**Greedy Search:**
- At each node, selects the edge with lowest (immediate) weight
- Heuristic method of search (i.e. does not necessarily find the best path)
Example: Decision Trees

ID3 vs Search

- age
- P? =
- CP
- CP
- CP

Search space: all possible trees

ID3: greedy search, maximizing info gain at each split

\[ \text{Search for smallest tree consistent with the training data} \]

"Inductive bias" of ID3

Ockam’s Razor: prefers the simplest hypothesis that explains the data.
(i.e. 1300 smallest exp is best)
Proof by Induction

Chalkboard:

– Weak Induction
  • basis case
  • inductive hypothesis
  • inductive step
– Example: sum of powers of two
– Why does proof by induction work?
  • propositional logic interpretation
Proof by Induction

In-Class Exercise
Prove the following statement by induction.

\[ \sum_{i=1}^{n} i = \frac{n(n + 1)}{2} \]
Recursion

Chalkboard:

– Example: Factorial (iterative implementation)
– Example: Factorial (recursive implementation)
– Strong Induction
  • multiple basis cases
  • complete assumption
– Proof of recursive factorial correctness
Recursion

**Chalkboard:**

- Definition: Sorted Array
- Example: Insertion Sort (iterative implementation)
- Example: Insertion Sort (recursive implementation)
- Big Idea: Divide and Conquer
- Example: Merge Sort
DYNAMIC PROGRAMMING
A Hidden Markov Model (HMM) provides a joint distribution over the tunnel states / travel times with an assumption of dependence between adjacent tunnel states.

\[ p(O, S, S, O, C, 2m, 3m, 18m, 9m, 27m) = (0.8 \times 0.08 \times 0.2 \times 0.7 \times 0.03 \times \ldots) \]
Forward-Backward Algorithm: Finds Marginals

\[ p(v, a, n) = \frac{1}{Z} \times \text{product weight of one path} \]

- Marginal probability \( p(Y_2 = a) \)
- \( Z \) is the total weight of all paths through all \( a \)

Motivating Example
Constituency Parsing

• **Variables:**
  - Constituent type (or $\emptyset$) for each of $O(n^2)$ substrings

• **Interactions:**
  - Constituents must describe a binary tree
  - Tag bigrams
  - Nonterminal triples (parent, left-child, right-child)

[these factors not shown]
Dynamic Programming

Chalkboard:

– Big Idea: Dynamic Programming
– Example: Fibonacci with and without dynamic programming
  • Recursive Fibonacci’s computational complexity
  • Dynamic programming Fibonacci’s computational complexity
– Types of Dynamic Programming
  • Tabulation (bottom-up)
  • Memoization (top-down)
– Example: Matrix Product Parenthesization
DATA STRUCTURES FOR ML
Abstractions vs. Data Structures

Abstractions
- List
- Set
- Map
- Queue (FIFO)
- Stack (LIFO)
- Graph
- Priority Queue

Data Structures
- Array (fixed size)
- Array (variable size)
- Linked List
- Doubly-Linked List
- Multidimensional Array
- Tensor
- Hash Map
- Binary Search Tree
- Balanced Tree
- Trie
- Stack
- Heap
- Graph
- Bipartite Graph
- Sparse Vector
- Sparse Matrix
Data Structures for ML

• Data:
  – Dense feature vector (array)
  – Sparse feature vector (sparse vector)
  – Design matrix (multidimensional array)

• Models:
  – Decision Trees (tree)
  – Bayesian Network (directed acyclic graph)
  – Factor Graph (bipartite graph)

• Algorithms:
  – Greedy Search (weighted graph)
  – A* Search (priority queue/heap)
  – Forward-backward for HMM (trellis)
Trees

Chalkboard:

– Binary Tree
  • Representation
  • Depth First Search
    – pre-order traversal
    – in-order traversal
    – post-order traversal
  • Breadth First Search

– Decision Tree
  • Representation
Sparse Vectors

**Chalkboard:**

- Sparse Vector
  - Representation
  - Sparse Dot Product
  - Addition of dense vector and sparse vector
Data Structures & Algorithms

Chalkboard:

– Weighted Directed Acyclic Graph
  • Representation
  • Greedy Search
  • Dijkstra’s Algorithm
  • A* Search

– Binary Search Tree
  • Representation
  • Average vs. Worst Case Time Complexity
  • Search
  • Insertion
  • Deletion