Richard Bellman pioneered Dynamic Programming in the 50’s

Dynamic Programming works via the Principle of Optimality: An optimal sequence of decisions is obtained iff each subsequence of decisions is optimal.

I.e. we can build large optimization solutions out of small optimization solutions.
Dynamic Programming is decomposing a problem into subproblems whose solutions are stored for later use.

The two biggest categories of Dynamic Programming are top-down and bottom-up.
Top-Down

- Starting with the large problem, decompose it into smaller problems
- Compute and store the solution to the smaller problems if they haven’t been computed yet
- Use the solutions to the subproblems to compute the solution to the large problem
Top-Down Example

```java
int marker = 1;
arr[]; arr[0] = 0; arr[1] = 1;

topDownFib(int n){
    if(n > marker){
        int temp = topDownFib(n-1)+topDownFib(n-2);
        marker = n;
        arr[n] = temp;
    }
    return arr[n];
}
```
Bottom-Up

Combine the solutions to small subproblems to obtain the solutions to subproblems of increasing size. Continue in this fashion until we arrive at the solution to the original problem.

fibonacci(3) =>

Fib[0] = 0
Fib[1] = 1
Bottom-Up Example

```java
int marker = 1; //the highest index we’ve calculated so far
HashMap<Integer, Integer> map; //Add 0 at 0th index and 1 at 1st index

bottomUpFib(int n) {
    if (n > marker) {
        int i = marker;
        for (; i < n ; i ++) {
            map.add(i+1, map.get(i-1) + map.get(i-2));
        }
        marker = n;
    }
    return map.get(n);
}
```
Top-Down vs Bottom-Up

- Top-down has recursive overhead
- Bottom-up may compute more sub-problems than necessary.
Theoretical Computer Science

Theoretical Computer Science concerns itself with three chief categories:

- Algorithm Design
- Algorithm Analysis
- Modes of Computation

Dynamic Programming is part of Algorithm Design

- Optimization Algorithm
  - Speed vs. Memory
Current Theory

- Increased speed
- Limited by difficulty in decomposing a large problem into smaller subproblems
- Favors objects that are linearly ordered and cannot be rearranged
  - Characters in a string, matrices in a chain, left-to-right order of leaves in a search tree
- Problems evaluating subproblems in an efficient order
- Uses more memory
Exercise

Try and think of a problem where the dynamic programming approach would be useful. The problem doesn’t necessarily need to be framed in terms of coding and computers.

Discuss with your table for a minute (or so).

(Hint: what’s a problem where it’s worth remembering the answer once in awhile?)
Real-World Application I

- **Bioinformatics**
  - Interdisciplinary field that develops methods and software tools for understanding biological data
    - DNA/RNA Sequence Alignment

- **Control Theory**
  - Interdisciplinary branch of engineering and mathematics that deals with the behavior of dynamic systems with inputs and how their behavior is modified by feedback
    - AI computing approaches
Real-World Application II

- **Information Theory**
  - Studies the quantification, storage, and communication of information
    - Cryptography
    - Gambling

- **Operations Research**
  - Deals with the application of advanced analytical methods to help make better decisions
    - Optimal search
    - Routing
    - Scheduling

- **Computer Science**
Real-World Application III

- **Diff Algorithm**
  - Reports differences between two files, expressed as a minimal list of line changes to bring either file into agreement with the other
    - Solves the longest common subsequence problem to find the lines that do not change between files

- **Viterbi Algorithm**
  - Decodes convolutional codes used in both CDMA and GSM digital cellular, dial-up modems, satellite, deep-space communications, and 802.11 wireless LANs
    - Decoding algorithm for convolutional codes over noisy digital communication links
    - Commonly used in speech recognition, speech synthesis, diarization, keyword spotting, computational linguistics, and bioinformatics
Real-World Application IV

- **Smith-Waterman Algorithm**
  - Performs local sequence alignment for determining similar regions between two strings or nucleotide or protein sequences
  - Compares segments for all possible lengths and optimizes the similarity measure

- **Bellman-Ford Algorithm**
  - Detects a negative cycle if any exists in a weighted digraph
  - Finds the shortest simple path if no negative cycles exist

- **Cocke-Kasami-Younger Algorithm**
  - Parsing algorithm for context-free grammars given in Chomsky Normal Form
Ongoing Research I

- Researchers at MIT’s Computer Science and Artificial Intelligence Laboratory (CSAIL) working on parallelize algorithms that utilize dynamic programming called Bellmania
- Programs run on multicore chips
- Operate up to 11 times faster than other parallel processing technique
Ongoing Research II

● Limitations
  ○ Typically 10 times longer than single-core version
  ○ Individual lines of code are more complex
    ■ So complex that errors occur
  ○ Problems efficiently accessing the stored solutions in parallel
    ■ Memory management more complex
Backup Exercise


https://www.cs.princeton.edu/courses/archive/spr05/cos423/lectures/06dynamic-programming.pdf

http://www.academia.edu/2875080/Dynamic_Programming_Its_Principles_Applications_Strengths_and_Limitations

