Algorithmic Paradigms

Common Paradigms

Standard patterns include the following,

Common Paradigms

Standard patterns include the following,

- Brute Force

Common Paradigms

Standard patterns include the following,

- Brute Force
- Greedy

Common Paradigms

Standard patterns include the following,

- Brute Force
- Greedy
- Divide and Conquer

Common Paradigms

Standard patterns include the following,

- Brute Force
- Greedy
- Divide and Conquer
- Dynamic Programming

Common Paradigms

Standard patterns include the following,

- Brute Force
- Greedy
- Divide and Conquer
- Dynamic Programming
- Randomized

Common Paradigms

Standard patterns include the following,

- **Brute Force**
- **Greedy**
- **Divide and Conquer**
- Dynamic Programming
- **Randomized**

You’ve already seen a lot of these in our class!

Brute Force

Solve a problem by trying every possible solution.

Brute Force

Solve a problem by trying every possible solution.

This includes,

Brute Force

Solve a problem by trying every possible solution.

This includes,

- TSP

Brute Force

Solve a problem by trying every possible solution.

This includes,

- TSP
- Backtrackers

Brute Force

Solve a problem by trying every possible solution.

This includes,

- TSP
- Backtrackers
- Branch and Bound

Brute Force

Solve a problem by trying every possible solution.

This includes,

- TSP
- Backtrackers
- Branch and Bound

Difficult to prove runtimes when pruning is involved.

Greedy

Solve a problem by trying only one choice at each decision

Greedy

Solve a problem by trying only one choice at each decision; never undo work.

Greedy

Solve a problem by trying only one choice at each decision; never undo work.

Examples include,

Greedy

Solve a problem by trying only one choice at each decision; never undo work.

Examples include,

- MST

Greedy

Solve a problem by trying only one choice at each decision; never undo work.

Examples include,

- MST
- Event Sweeps

Greedy

Solve a problem by trying only one choice at each decision; never undo work.

Examples include,

- MST
- Event Sweeps
- Huffman Encoding

Greedy

Solve a problem by trying only one choice at each decision; never undo work.

Examples include,

- MST
- Event Sweeps
- Huffman Encoding
- Dijkstra’s Shortest Path Algorithm

Greedy

Solve a problem by trying only one choice at each decision; never undo work.

Examples include,

- MST
- Event Sweeps
- Huffman Encoding
- Dijkstra’s Shortest Path Algorithm

Typically involve some sort call or a sorted data structure.

Divide and Conquer

Solve a problem by turning the problem into subproblems and solving them.

Divide and Conquer

Solve a problem by turning the problem into subproblems and solving them.

Common examples,

Divide and Conquer

Solve a problem by turning the problem into subproblems and solving them.

Common examples,

- Binary Search

Divide and Conquer

Solve a problem by turning the problem into subproblems and solving them.

Common examples,

- Binary Search
- Quick Sort

Divide and Conquer

Solve a problem by turning the problem into subproblems and solving them.

Common examples,

- Binary Search
- Quick Sort
- Merge Sort

Divide and Conquer

Solve a problem by turning the problem into subproblems and solving them.

Common examples,

- Binary Search
- Quick Sort
- Merge Sort
- Karatsuba Algorithm

Divide and Conquer

Solve a problem by turning the problem into subproblems and solving them.

Common examples,

- Binary Search
- Quick Sort
- Merge Sort
- Karatsuba Algorithm
- Linear Median Finding

Divide and Conquer

Solve a problem by turning the problem into subproblems and solving them.

Common examples,

- Binary Search
- Quick Sort
- Merge Sort
- Karatsuba Algorithm
- Linear Median Finding
- Fast Expo

Divide and Conquer

Solve a problem by turning the problem into subproblems and solving them.

Common examples,

- Binary Search
- Quick Sort
- Merge Sort
- Karatsuba Algorithm
- Linear Median Finding
- Fast Expo

Mostly recursive problems.

Dynamic Programming

Solve a problem by turning the problem into subproblems, solving them, and storing their solutions.

Dynamic Programming

Solve a problem by turning the problem into subproblems, solving them, and storing their solutions.

Common Examples,

Dynamic Programming

Solve a problem by turning the problem into subproblems, solving them, and storing their solutions.

Common Examples,

- Fast(ish) Fibonacci

Dynamic Programming

Solve a problem by turning the problem into subproblems, solving them, and storing their solutions.

Common Examples,

- Fast(ish) Fibonacci
- Knapsack and variants

Dynamic Programming

Solve a problem by turning the problem into subproblems, solving them, and storing their solutions.

Common Examples,

- Fast(ish) Fibonacci
- Knapsack and variants
- Matrix Chain Multiplication

Dynamic Programming

Solve a problem by turning the problem into subproblems, solving them, and storing their solutions.

Common Examples,

- Fast(ish) Fibonacci
- Knapsack and variants
- Matrix Chain Multiplication
- Floyd-Warshall’s Algorithm (Floyd’s)

Dynamic Programming

Solve a problem by turning the problem into subproblems, solving them, and storing their solutions.

Common Examples,

- Fast(ish) Fibonacci
- Knapsack and variants
- Matrix Chain Multiplication
- Floyd-Warshall’s Algorithm (Floyd’s)
- Memoization (not memorization)

Dynamic Programming

Solve a problem by turning the problem into subproblems, solving them, and storing their solutions.

Common Examples,

- Fast(ish) Fibonacci
- Knapsack and variants
- Matrix Chain Multiplication
- Floyd-Warshall’s Algorithm (Floyd’s)
- Memoization (not memorization)

Things you might not have seen before

Randomized!!!

Solve a problem by trying random solutions and checking the value of the solution.

Randomized!!!

Solve a problem by trying random solutions and checking the value of the solution.

Two broad categories

Randomized!!!

Solve a problem by trying random solutions and checking the value of the solution.

Two broad categories

- Monte Carlo (Guessing)

Randomized!!!

Solve a problem by trying random solutions and checking the value of the solution.

Two broad categories

- Monte Carlo (Guessing)
- Las Vegas (Waiting)

Randomized!!!

Solve a problem by trying random solutions and checking the value of the solution.

Two broad categories

- Monte Carlo (Guessing)
- Las Vegas (Waiting)

Example

Randomized!!!

Solve a problem by trying random solutions and checking the value of the solution.

Two broad categories

- Monte Carlo (Guessing)
- Las Vegas (Waiting)

Example

- Genetic Algorithms (Monte Carlo; not guaranteed to be best)

Randomized!!!

Solve a problem by trying random solutions and checking the value of the solution.

Two broad categories

- Monte Carlo (Guessing)
- Las Vegas (Waiting)

Example

- Genetic Algorithms (Monte Carlo; not guaranteed to be best)
- Randomized Hill Climbing (Monte Carlo; not guaranteed to be best)

Randomized!!!

Solve a problem by trying random solutions and checking the value of the solution.

Two broad categories

- Monte Carlo (Guessing)
- Las Vegas (Waiting)

Example

- Genetic Algorithms (Monte Carlo; not guaranteed to be best)
- Randomized Hill Climbing (Monte Carlo; not guaranteed to be best)
- Randomized Skip List (Las Vegas; always correct)

Randomized!!!

Solve a problem by trying random solutions and checking the value of the solution.

Two broad categories

- Monte Carlo (Guessing)
- Las Vegas (Waiting)

Example

- Genetic Algorithms (Monte Carlo; not guaranteed to be best)
- Randomized Hill Climbing (Monte Carlo; not guaranteed to be best)
- Randomized Skip List (Las Vegas; always correct)
- Randomized Quick Sort (Las Vegas; always correct)