09/22/2020

CSCI 310 - Advanced Algorithms

Scribes: Collin Bauer, Alex Bailey

9:55 AM: lecture started

Exhausive Search and Introduction to Tree Search

Q: *What is exhausitve search?*

- search everything
- brute force approach

Brute Force Algorithm - Revisit

- Straightforward approach
- often checks all possible answers - exhausitvely checks
- inefficient

Examples discussed so far:

- linear search
- travelling salesman
- bubble sort
- selection sort
- "string algorithm" (find substring pattern)

Q: *If an algorithm is O(n^2), what tipped you off?*

- Nested loops
- Summation
- *Note: Intuition helps find the order of growth, but to prove it, solve the summation.*

Combinatorics and Optimization

- Mathematical problems dealing with combinations, permutations or optimizing parameters
- General problems seeking specific solutions
- BRUTE FORCE: generate *all* solutions, select the *best*

> Q: *How can I maximize my profite and minmize my loss?*
> $Z = \max(x_1 + x_2 + x_3)$ where x_1, x_2, x_3 representes the number of each product
> $0 <= x_1 < 100$
> $0 <= x_2 < 10000$
> $0 <= x_3 < 500$
>
> *Just by inspection...when will Z be maximum?*

> - when $x_1 = 99$; $x_2 = 9999$, $x_3 = 499$
>
> What about if we slightly change the problem?
> $Z = \max(x_1 + x_2 - x_3)$
>
> - Now the optimized solution is $x_1 = 99$, $x_2 = 9999$, $x_3 = 0$

Many real-world problems reduce to combinatorial versions (e.g. routing, schedulting, logitical planning

- Combinatorial problems eeven show up on technical reviews.
- Two of today's problems represent a special class of problems: *NP-Hard*
  - NP-Hard problems have no known polynomial-time solution (i.e. $O(n^2)$)
  - Arguably the greatest open question in computing

Traveling Salesperson Problem

- A salesman needs to visit n cities without repeating any.
  What is the cheapest route?
- In mathematical terms: Given a weighted graph of n vertices, find a cycle that travels throughe every vertex exactly once.

Ghosh demonstrated the salesperson problem to high schoolers with "balls" and "strings"
This problem is heavily based in graph theory, and it gets infinitley more complicated with every "ball" added

Knapsack Problem

- A thief is choosing from a set of n objects, where each object has a value v_i and a weight w_i. With a knapsack capacity c, find the most valuable subset of objects.
- In mathematical terms: Given a set of n objects with value v_i and weight w_i and a maximum cpacity c, find the subset of object with the highest total value v and a total weight $w < c$.

Should have brought a bigger sack.

This problem actually has a polynomial solution $O(n^4)$! Can you find it?

Eight Queens Problem

- How many different ways can you arrange 8 queens without endangering any of them?
- 92 unique solutions
- How many different positions are there so that:
  - No two queesn are on the same square?
  - No two queesn are on the same row?
  - No two queesn are on the same row OR in the same column?

Q: *Is this an optimization or a combinatorics?*

- Combinatorics. Why?
- *It is not optimization because there is no function for maximization/minimization.*
- *It is combinatorics because you can represent the problem in graphs, and the rules can be represented as edges.*

Exhaustive (Graph) Search

- Graphs are a very common construction in computer science
- Specially structured graphs called trees are a common way to store data
- How do we search graph vertices in an ordered fashion?
    - Breadth-First Search
    - Depth-First Search

> **Graphs in terms of nodes only or edges only**
> Set of edges = {(n_1, n_2), (n_2, n_4), (n_3, n_1)} where n_1, n_2, n_3, n_4 are nodes
> Directed or undirected graph

Mathematical Graph

- A set of vertices and edges that connect them
- trees are graphs with no cycles (loops)
- Common representation
    - Adjacency matrix
    - Adjacency list

Depth-First Search

- Search vertices by moving as far as possible from the starting node
- Produce a set of trees containing the searched vertices called a Depth-frist Search Forest
- Implemented with a stack (least in first out) data structure

Why it is important

- Methematical: graph connectiviety, acyclicity, etc.
- Foundation for iterative deepening tree search - Applications in SAT solving
- Often applied to AI search spaces (e.g., determining which move to make in chess, solving a maze)
- File Systems arranged as trees can be traversed using DFS-based Database algorithms

> *What is SAT?*
> SAT = a V b V c, where a, b, c are Boolean values
>
0	0	0	=	0
0	0	1	=	1
0	1	0	=	1
0	1	1	=	1
1	0	0	=	1
1	0	1	=	1
1	1	0	=	1
1	1	1	=	1

> SAT = Satisfiability
> a, b -> 4 cases
> a, b, c -> 8 cases
> a, b, c, d -> 16 cases
> and so on...
>
> Satisfiability is built on boolean values, and allegedly implies a truth table...or something
> *I got a little lost during this epxlanation.*

...

10:55 AM - moved on from slides to talk about mathematical induction

> Mathematical induction... So far, what we learned is
>
> - weak form of induction
> - strong form of induction
>
> **Weak form**
> Let P be predicate on non-negative integer if:
>
> - P(0) is true and
> - for all n ∈ $\mathbb{N}$, P(n) -> P(n+1)
>   -> P(m) is true for all m ∈ $\mathbb{N}$
>
> **Strong form**
> Let P be predicate on non-negative integer if:
>
> - P(0) is true and
> - for all n ∈ $\mathbb{N}$, P(0), P(1), ..., P(n)
>   -> P(m) is true for all m ∈ $\mathbb{N}$

...

Something about Book of Proof by R. Hammack

Read the following sections (from our usual book):

- 3.1
- 3.2
- 3.3
- 3.4