Design and Analysis of Algorithm [R1UC407B]

Module-II: Brute Force

Dr. A K Yadav

GALGOTIAS UNIVERSITY

School of Computer Science and Engineering
Plat No 2, Sector 17A, Yamuna Expressway
Greater Noida, Uttar Pradesh - 203201

February 14, 2025

Contents

Brute Force Approach

- Brute Force is a straightforward method used in algorithmic problem-solving that checks every possible solution until the correct one is found.
- Brute Force Algorithms function by searching each element sequentially until the desired result is found or all options are exhausted.

Pros & Cons of Brute Force

Pros:

- The brute force approach is a guaranteed way to find the correct solution by listing all the possible candidate solutions for the problem.
- It is a generic method and not limited to any specific domain of problems.
- The brute force method is ideal for solving small and simpler problems.
- It is known for its simplicity and can serve as a comparison benchmark.

Cons:

- The brute force approach is inefficient. For real-time problems, algorithm analysis often goes above the O(N!) order of growth.
- This method relies more on compromising the power of a computer system for solving a problem than on a good algorithm design.
- Brute force algorithms are slow.
- Brute force algorithms are not constructive or creative compared to algorithms that are constructed using some other design paradigms.

Naive string-matching algorithm

NAIVE-STRING-MATCHER(T,P)

```
$n = length(T)$
$m = length(P)$
for $s = 0$ to $n - m$
    if $P[1..m] = T[s+1..s+m]$
        print pattern occurs at position $s+1$.
```

NAIVE-STRING-MATCHER(T,P)

```
$n = length(T)$
$m = length(P)$
for $s = 0$ to $n - m$
    $valid = 1$
    for $j = 1$ to $m$
        if $P[j] \neq T[s + j]$
            $valid = 0$
    if $valid = 1$
        print pattern occurs with sift $s$.
```

Complexity of the algorithm is $O((n - m + 1)m)$

Traveling Salesman Problem

Given a set of cities and distances between every pair of cities, the problem is to find the shortest possible route that visits every city exactly once and returns to the starting point.

Algorithm:

1. Consider city 1 as the starting and ending point. Since the route is cyclic, we can consider any point as a starting point.
2. Generate all (n-1)! permutations of cities.
3. Calculate the cost of every permutation and keep track of the minimum cost permutation.
4. Return the permutation with minimum cost.

Knapsack Problem

Given a bag with maximum weight capacity of W and a set of items, each having a weight and a value associated with it. Decide the number of each item to take in a collection such that the total weight is not more than the capacity W and the total value is maximized.

Algorithm:

1. Create a list of all possible combinations of items.
2. For each combination, calculate the total weight and total value of the items in the combination.
3. If the total weight is less than or equal to the weight capacity, update the maximum value found so far.
4. Return the maximum value found.

Assignment problem

You are the head of a firm and you have to assign jobs to people. You have N persons working under you and you have N jobs that are to be done by these persons. Each person has to do exactly one job and each job has to be done by exactly one person. Each person has his own capability (in terms of time taken) to do any particular job. Your task is to assign the jobs to the persons in such a way that the total time taken is minimum. A job can be assigned to only one person and a person can do only one job.

Algorithm:

1. Create a list of all possible assignments of tasks to agents.
2. For each assignment, calculate the total cost of the assignment by summing the costs of the tasks assigned to each agent.

3 Keep track of the minimum cost found so far.

4 Return the minimum cost found.

Thank you

Please send your feedback or any queries to
ashokyadav@galgotiasuniversity.edu.in