Competitive Study
Of
Dynamic Programing & Greedy Algorithm

Algorithm Lab

Course Code: CSE215

Submitted To:

Subroto Nag Pinku

Department Of CSE

Daffodil International University

Submitted By:

Susmoy Paul

ID- 191-15-12910

Section – O13

Group - 910

Department Of CSE

Daffodil International University

Competitive Study Of Dynamic Programing & Greedy Algorithm

Nazam Been Saddi – 191-15-124170

***INTRODUCTION**:*

This paper discusses relationships between two approaches to optimal solution to problems: Greedy algorithm and dynamic programming. Greedy algorithm has a local choice of the sub-problems whereas Dynamic programming would solve the all sub-problems and then select one that would lead to an optimal solution. Greedy algorithm takes decision in one time whereas Dynamic programming takes decision at every stage. This paper discus about Dynamic Programming and Greedy Algorithm to solve the Knapsack Problem where one has to maximize the benefit of items in a knapsack without extending its capacity. The paper discusses the comparison of each algorithm in terms of time different Item values.

DYNAMIC PROGRAMMING AND GREEDY ALGORITHM:

Dynamic algorithm is an algorithm design method, which can be used, when the problem breaks down into simpler sub problems; it solves problems that display the properties of overlapping sub problems. In general, to solve a problem, it's solved each sub problems individually, then join all of the sub solutions to get an optimal solution.

The dynamic algorithm solves each sub problem individually, once the solution to a given sub problem has been computed, it will be stored in the memory, since the next time the same solution is needed, it's simply looked up. Distinctly, a Dynamic algorithm guarantees an optimal solution.

Dynamic Programming is a technique for solving problems whose solutions satisfy recurrence relations with overlapping sub-problems. Dynamic Programming solves each of the smaller sub-problems only once and records the results in a table rather than solving overlapping sub-problems over and over again. To design a dynamic programming algorithm for the 0/1 Knapsack problem, we first need to derive a recurrence relation that expresses a solution to an instance of the knapsack problem in terms of solutions to its smaller instances.

WHAT IS DYNAMIC PROGRAMMING?

Dynamic Programming is a technique in computer programming that helps to efficiently solve a class of problems that have overlapping subproblems and optimal substructure property. Such problems involve repeatedly calculating the value of the same subproblems to find the optimum solution.

Whereas Algorithms are the way to find that solution in easiest method. There are different kind of algorithm that used to find the solution of any problem like searching algorithm, greedy algorithm, in packing algorithm and so on. Algorithms are use according to the basis of requirement of any program.

Divide & Conquer algorithm partition the problem into disjoint subproblems solve the subproblems recursively and then combine their solution to solve the original problems. Dynamic Programming is the most powerful design technique for solving optimization problems. Dynamic Programming is used when the subproblems are not independent, when they share the same subproblems. In this case, divide and conquer may do more work than necessary. But unlike, divide and conquer, these sub-problems are not solved independently because it solves the same sub problem multiple times.

Susmoy Paul – 191-15-12910

USEAGE OF DYNAMIC PROGRAMMING:

Dynamic programming is used where we have problems, which can be divided into similar sub-problems, so that their results can be re-used. Mostly, these algorithms are used for optimization. Before solving the in-hand sub-problem, dynamic algorithm will try to examine the results of the previously solved sub-problems. The solutions of sub-problems are combined in order to achieve the best solution. So, we can say that -

- *The problem should be able to be divided into smaller overlapping sub-problem.*
- *An optimum solution can be achieved by using an optimum solution of smaller sub-problems.*
- *Dynamic algorithms use Memorization technique.*
- *Dynamic programming used in both top-down and bottom-up manner.*

The following computer problems can be solved using dynamic programming approach –

- ✓ *Fibonacci number series*
- ✓ *Knapsack problem*
- ✓ *Tower of Hanoi*
- ✓ *All pair shortest path by Floyd-Warshall*
- ✓ *Shortest path by Dijkstra*
- ✓ *Project scheduling*

Dynamic Programming solves each subproblems just once and stores the result in a table so that it can be repeatedly retrieved if needed again. Dynamic Programming is a Bottom-up approach- we solve all possible small problems and then combine to obtain solutions for bigger problem

From Dynamic programming point of view, Dijkstra's algorithm for the shortest path problem is a successive approximation scheme that solves the dynamic programming functional equation for the shortest path problem by the Reaching method.

Kajali Chakma – 191 -15-12923

WHAT IS GREEDY ALGORITHM?

Greedy Approach or Technique, As the name implies, this is a simple approach which tries to find the best solution at every step. Thus, it aims to find the local optimal solution at every step so as to find the global optimal solution for the entire problem. Consider that there is an objective function that has to be optimized (maximized/ minimized). This approach makes greedy choices at each step and makes sure that the objective function is optimized.

The greedy algorithm has only one chance to compute the optimal solution and thus, cannot go back and look at other alternate solutions. However, in many problems, this strategy fails to produce a global optimal solution.

USEAGE OF GREEDY ALGORITHM

For a problem with the following properties, we can use the greedy technique, Greedy Choice Property. This states that a globally optimal solution can be obtained by locally optimal choices.

Optimal Sub-Problem: This property states that an optimal solution to a problem, contains within it, optimal solution to the sub-problems. Thus, a globally optimal solution can be constructed from locally optimal sub-solutions.

Feasible Solution: This can be referred as approximate solution (subset of solution) satisfying the objective function and it may or may not build up to the optimal solution. Optimal Solution: This can be defined as a feasible solution that either maximizes or minimizes the objective function.

This algorithm may not be the best option for all the problems. It may produce wrong results in some cases. This algorithm never goes back to reverse the decision made. This algorithm works in a top-down approach. The main advantage of this algorithm is -

- *The algorithm is easier to describe.*
- *This algorithm can perform better than other algorithms (but, not in all cases).*

The following computer problems can be solved using greedy approach –

- ✓ *Sorting: Selection sort, Topological sort*
- ✓ *Priority Queues: Heap sort*
- ✓ *Huffman coding compression algorithm*
- ✓ *Prim's and Kruskal's algorithms*
- ✓ *Sorted Path in Weighted Graph (Dijkstra's)*
- ✓ *Coin change problem*
- ✓ *Job scheduling algorithms*

Kazi Israt Zahan – 191-15-12402

Greedy algorithm take decision in one time whereas Dynamic programming take decision at every stage. Greedy algorithm work based on choice property whereas Dynamic programming work based on principle of optimality. Greedy algorithm follows the top-down strategy whereas Dynamic programming follows the bottom-up strategy.

TOP-DOWN:

Top-down approach also known as stepwise design and stepwise refinement and in some cases used as a synonym of decomposition is essentially the breaking down of a system to gain insight into its compositional sub-systems in a reverse engineering fashion. In a top-down approach an overview of the system is formulated, specifying, but not detailing, any first-level subsystems. Each subsystem is then refined in yet greater detail, sometimes in many additional subsystem levels, until the entire specification is reduced to base elements. A top-down model is often specified with the assistance of "black boxes", which makes it easier to manipulate. However, black boxes may fail to clarify elementary mechanisms or be detailed enough to realistically validate the model. Top down approach starts with the big picture. It breaks down from there into smaller segments.

BOTTOM-UP:

Bottom-up approach is the piecing together of systems to give rise to more complex systems, thus making the original systems sub-systems of the emergent system. Bottom-up processing is a type of information processing based on incoming data from the environment to form a perception. From a cognitive psychology perspective, information enters the eyes in one direction (sensory input, or the "bottom"), and is then turned into an image by the brain that can be interpreted and recognized as a perception (output that is "built up" from processing to final cognition). In a bottom-up approach the individual base elements of the system are first specified in great detail. These elements are then linked together to form larger subsystems, which then in turn are linked, sometimes in many levels, until a complete top-level system is formed. This strategy often resembles a "seed" model, by which the beginnings are small but eventually grow in complexity and completeness. However, "organic strategies" may result in a tangle of elements and subsystems, developed in isolation and subject to local optimization as opposed to meeting a global purpose.

Kazi Abu Sayem Md. Nayem – 191-15-12910

Differences between Dynamic programming and Greedy algorithm discussing in next page.

FEATURE	*DYNAMIC PROGRAMMING*	*GREEDY METHOD*
Feasibility	*In Dynamic Programming we make decision at each step considering current problem and solution to previously solved sub problem to calculate optimal solution.*	*In a greedy Algorithm, we make whatever choice seems best at the moment in the hope that it will lead to global optimal solution.*
Optimality	*It is guaranteed that Dynamic Programming will generate an optimal solution as it generally considers all possible cases and then choose the best.*	*In Greedy Method, sometimes there is no such guarantee of getting Optimal Solution.*
Recursion	*A Dynamic programming is an algorithmic technique which is usually based on a recurrent formula that uses some previously calculated states.*	*A greedy method follows the problem-solving heuristic of making the locally optimal choice at each stage.*
Memorization	*It requires table for memorization and it increases its memory complexity.*	*It is more efficient in terms of memory as it never looks back or revise previous choices*
Time Complexity	*Polynomial*	*Polynomial but use less time than dynamic programming*
Memory Complexity	*Use DP table to store the answer to use them recursively*	*More effective because never have to look back for Other answer.*

Susmoy Paul – 191-15-12910

CONCLUSION*:*

Both Greedy and dynamic programming algorithms construct an optimal solution of a subproblem based on optimal solutions of smaller subproblems. However, the main difference is that greedy algorithms have a local choice of the subproblem that will lead to an optimal answer. where dynamic programming would solve all dependent subproblems and then select one that would lead to an optimal solution. Both require an optimal solution of current subproblem is based on optimal solutions of dependent subproblems.

The objective of the paper is to present a comparative study of the dynamic programming, and greedy algorithms. Greedy strategy is to point out the initial state of the problem, through each of the greedy choice and get the optimal value of a problem-solving method. Dynamic programming is an optimization approach that transforms a complex problem into a sequence of simpler problems; its essential characteristic is the multistage nature of the optimization procedure.

More so than the optimization techniques described previously, dynamic programming provides a general framework for analyzing many problem types and the best possible way for solving the problem.

THANK YOU