iJRASET
International Journal For Research in
Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 8 **Issue:** VI **Month of publication:** June 2020

DOI: http://doi.org/10.22214/ijraset.2020.6361

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 8 Issue VI June 2020- Available at www.ijraset.com

Problem Solving Optimization using Dynamic Programming Approach

P. V. Siva Kumar[1], M. Bharath Reddy[2], V. Venu[3]
[1]Associate Professor, [2, 3]UG Student, Computer Science and Engineering, VNR Vignana Jyothi Institute of Engineering and Technology Hyderabad, India-500090.

Abstract: Dynamic programming approach is an optimization approach that can solve the complex problem by dividing into overlapping sub-problems and solve those sub-problems stage by stage to get the optimal solution of the problem. Even though we have many problems solving approaches , dynamic programming is the approach the best to get the optimal solution of the problem. In this paper, we would like to discuss about the use of dynamic programming approach and how it is different from other problem solving techniques like divide and conquer, greedy method, branch and bound and backtracking. We discussed the dynamic programming approach in detail and given some of the applications of dynamic programming where we can apply this problem solving technique. Finally we discussed about the limitations of dynamic programming approach.
Keywords: dynamic programming, divide and conquer, greedy method, optimization, backtracking, branch and bound.

I. INTRODUCTION

We have many problem solving techniques to solve the real time problems. We have many problem solving approaches like divide and conquer, greedy method, backtracking, branch and bound and so on. Here we will discuss about each problem solving approach in brief.

A. Divide and Conquer Approach

In this divide and conquer technique mainly there are three parts i.e divide, conquer and combine. In this divide part we are going to divide problem in to sub problems. This is a recursive procedure to divide the entire problem into sub problems. In conquer part, recursively calling the sub problem until that sub problem solved. In the third part i.e. combine part we are combining all sub problem solutions to get the solution of the entire problem. Some of the applications problems which can be solved by using this divide and conquer approach is Binary search, merge sort, quick sort, Stassen's matrix multiplication and so on. The main advantage of this divide and conquer approach is that it is used to solve all sub problems independently and parallel so that it reduced the time complexity to solve the problem. The disadvantage of this method is it requires lot of memory because all the sub problems are solved using recursion.

B. Greedy Approach

In this greedy approach we are going to define the objective function for the given problem and we are going to choose all possible choices of solutions for the given problem. From that solution set we are going to select the optimal solution of the problem. The advantage of this technique is the implementation of this approach is easy. However, the main disadvantage of this technique is that it is not guaranteed that the greedy method can provide the globally optimal solution of the problem. The example problems which can solved by this greedy approach are knapsack problem, job scheduling problem, graph vertex cover and map coloring problem, travelling salesman problem, Prim's and Kruskal's algorithms to find the minimal spanning tree and so on.

C. Backtracking Technique

In this backtracking technique we use recursive calling to find the solution by constructing a solution step by step. In each step it find the all possible solutions of the problem and remove the solutions which can not lead for the solution of the problem based on the problem constraints.

Generally we can apply this backtracking technique to solve the decision problems, optimisation problem and enumeration problem. We find the feasible solution of the decision problem, the best solution for the optimisation problem and set of all feasible solutions for enumeration problem. Some of the examples problems solved using backtracking techniques are N-Queen problem, maze problem, Sudoku puzzle and so on.

D. Branch and Bound Approach

Branch and Bound technique is generally used for solving combinational optimization problems. The main disadvantage of this technique is that it requires finding all possible permutations in the worst case and the time complexity is also required in exponential.

We can apply this technique for the problems like traveling salesman problem, 0/1 knapsack problem and so on.

As each method of problem solving technique has its own advantage and disadvantages, we can select the technique based on the problem constraints.

After studying of these techniques it is motivated to do more work on dynamic programming approach, as this technique gives us the optimal solution of the problem based on the problem constraints. Compare to all the problem solving techniques the dynamic programming approach is taking less time complexity also. With this motivation we studied more about the dynamic programming approach.

II. DYNAMIC PROGRAMMING APPROACH

An American mathematician Richard Bellman described about dynamic programming approach in 1940 to solve the mathematical problems or computer programming in optimal way. As prescribed by Richard Bellman, this dynamic programming approach is consists of sequence of decisions.

In dynamic programming the problem can be divided in to number of subproblems and solving these subproblems we get the optimal solution of the problem. In dynamic programming mainly there are two principles we need to keep in mind i.e principle of optimality and overlapping subproblems[1-5].

Principle of optimality defined that if each subproblem has optimal solution then by selecting that optimal solution in each level of subproblem we can achieve the optimal solution of the problem. Overlapping subproblems [3]means that if we can define the recursive method to solve the subproblems the decision should be taken in each stage based on the previous level subproblem decision. So in optimization problem solving if we define the overlapping subproblems then the dynamic programming uses solve the overlapping subproblems only once and the results will be saved in to the table and used it for later. Because of this overlapping subproblem solving we will solve the problems in polynomial time where as if we solve those problems using navie method we require exponential time to solve that problem.

Generally there are four steps we need to develop in dynamic approach. Step 1: we need to identify the optimal solution for the problem. Step 2: identify the overlapping subproblems and solve those subproblems using recursion. Step 3: By using bottom up approach find the optimal solution. Step 4: find the optimal solution by considering the previous subproblems solutions[6].

III. APPLICATIONS OF DYNAMIC PROGRAMMING

There are many areas where we can find the optimal solution of the problem using dynamic programming are bioinformatics, control theory, information theory, operations research and many applications of computer science like artificial intelligence graphics[6,7] and so on.

It is also using for many other problems like Longest Common Subsequence (LCS), Longest Increasing Subsequence (LIS), Assembly Line Scheduling Problem, weighted interval scheduling, global sequence alignment segmented least squares problem, scoring alighments, sum of subset problem, optimal capacity expansion, discounting future returns, coin change problem, 0/1 knapsack problem, optimial stopping problems, optimal binary search tree, flow shop schedurling, travelling salesman problem, single source shortest path problem, single pair shortest path problem, single destination shortest path problem, all pairs shortest path problem[6,7] and so on.

IV. CONCLUSION

The dynamic programming approach is very useful and has many advantages to find the optimal solution for the problem. The main advantage of dynamic programming is it stores all the solutions of subproblems in the table. So we can retrieve the solution from the table whenever requires. Because of that the same subproblem need not to compute again and again. Due to this the total time complexity of the problem will reduce. That's why dynamic programming approach will take polynomial time complexity whereas the other problem solving techniques will require exponential time complexity to solve the same problem[6-8].

The limitations of this approach is we need to apply the principle of optimality in each level of subproblem solving. It can use only if we can able to define the overlapping suproblems of the problem[6-8].

REFERENCES

[1] Baker, K.R., Schrage, L.E.: Finding an Optimal Sequence by Dynamic Programming: An Extension to Precedence-Constrained Tasks; Operations Research 26 (1978) 111–120
[2] Denardo, E.V.: *Dynamic Programming - Models and Applications*; Prentice-Hall, Englewood Cliffs, 1982
[3] Gallo, G., Pallottino, S.: Shortest Path Methods–A Unifying Approach; Math. Programming Study 26 (1986) 38–64
[4] Lawler, E.L.: Efficient Implementation of Dynamic Programming Algorithms for Sequencing Problems; Report BW 106/79, Mathematisch Centrum, Amsterdam, 1979
[5] M. Ke, J. Chen and H. Jin, "Research of Using Dynamic Programming in the Nodes Encoding Optimization," 2009 International Conference on Information Engineering and Computer Science, Wuhan, 2009, pp. 1-4, doi: 10.1109/ICIECS.2009.5363278.
[6] Bhowmik, Biswajit. (2010). DYNAMIC PROGRAMMING – ITS PRINCIPLES, APPLICATIONS, STRENGTHS, AND LIMITATIONS. International Journal of Engineering Science and Technology. 2.
[7] Neumann K. (1993) Dynamic Programming Basic Concepts and Applications. In: Frauendorfer K., Glavitsch H., Bacher R. (eds) Optimization in Planning and Operation of Electric Power Systems. Physica, Heidelberg
[8] Abdulla, Sajil. (2018). Dynamic programming. 10.13140/RG.2.2.20186.39368.