International Journal of Research in Engineering, Science and Management
Volume-3, Issue-3, March-2020
www.ijresm.com | ISSN (Online): 2581-5792

Interpretation of Learning Strategies for Algorithms

Arun Padmanabhan
Assistant Professor, Dept. of Computer Applications, Saintgits College of Applied Sciences, Kottayam, India

***Abstract*: This paper introduces real world ideas for effectively learning the basic concepts of algorithms. Design and Analysis of Algorithm are useful for study of computer science and information technology and most important for social science and industry. Algorithms are clearly specified means to solve any problem. Problem solving and resource utilization skills are one among many judging factors of any algorithm. It is also thereby important to lean algorithm wisely so as to design or analyze it effectively.**

***Keywords*: Algorithms, Interpretation, Learning ideas, Teaching style.**

1. Introduction

Algorithms and its techniques have to be learned wisely in order to implement it effectively. A good algorithm can solve any task using minimal requirements and time. It is therefore necessary to understand the concept of algorithms in detail so as to implement it in effect for bringing out good algorithms. The first step towards an understanding of why the study and knowledge of algorithms are so important is to define exactly what we mean by an algorithm. However, for most of us, complex algorithms are best studied so we can use them as building blocks for more efficient logical problem solving in the future.

Learning wisely is what required for building an efficient algorithm. Steps to be followed for learning algorithm easily are:

1. Selection of learning approach
2. Recognizing the concept of algorithm
3. Solving the algorithm
4. Implementation of algorithm

2. Selection of learning approaches

Let's first discuss the different approaches to figure out the best and most efficient way (the optimal way) to learn Algorithms for Beginners

A. Brute force approach

To brute the optimal way, learners can try every possible combinations of learning, and figure out the best one that suits them that is the optimal way. The time complexity for this approach would be very high as they have to figure out every possible combination of learning.

B. Greedy approach

A greedy approach would be to make a locally optimized choice i.e. to pick a way that the learners think would be the best for them at that point of time and assume that it leads to a globally optimized choice (the optimal way). It will be difficult to prove the correctness though. This approach would take time proportional to the number of optimizations you need to reach the optimal way. Hence, if we say there are `n` optimizations to make, then this approach would take O (n) time. Sub section heading and modify the heading.

C. Divide and conquer approach

To apply a divide and conquer approach we need to make one observation. (1) The optimal way is not always the same for everyone. With that in mind and given an optimal way of a programmer you can divide (work on specific parts of the given way) and conquer (figure out if that part works out for you or not or it needs some modifications). Finally, you merge the sub problems to get your optimal way. This approach would take O (nlogn) time irrespective of the modifications required. Space Complexity would be high for this approach.

D. Dynamic programming approach

Explore all the different ways of learning algorithms, take the one that best suites you. This approach takes exponential time O (2^n) where `n` is the number of ways, hence, this approach would take time but would give you the optimal way. Finally, a DP approach to get the optimal way would be to just to apply what you did except that you have to use memorization i.e. to remember the optimizations (ways) that worked out for you and the ones that didn't. This would bring down the time complexity from exponential time to polynomial time, about O (n^2).
So what would be the best and the most efficient way?

As observed, the optimal way is not always the same for two programmers. So, an optimal choice cannot be given. But most programmers use the greedy approach combined with dynamic programming technique.

3. Recognizing the concept of algorithm

Having not understood the concept of algorithm, it is impossible to learn or develop it. Hence learning the concept of any algorithm is mandatory for developing it to solve any problem. After the selection of right approach is made, learners

have to spend most of their time in understanding the concept of algorithm. Once this stage is done, learners may learn about the input requirement, output, design strategy, complexity, performance analysis etc.

4. Solving the algorithm

Ability to learn solving the algorithm produces solution for any problem. It can be achieved only through following a series of sub steps.

A. Exemplify: Create the example of the algorithm

Before thinking about the logic of algorithm, we should understand the question clearly. Write at least two examples, indicating input and output. This two-minute initial work will remove the uncertainty of misunderstanding the question and thinking in wrong direction.

B. Pattern matching

We have to consider what problems the algorithm is similar to; we need to figure out if we can modify the solution to develop an algorithm for the given problem.

C. Simplify and generalize

Changing constraint (e.g. size, length, data type) to simplify the problem. For example, changing the data type from double to int, make the problem smaller. Write algorithm for int data type and then generalize for double.

D. Base case and build

This approach is most widely used in the recursive algorithm. Solve the algorithm first for a base case (e.g., just one element). Then, try to solve it for elements one and two, assuming that we have the answer for element one. Then, try to solve it for elements one, two and three, assuming that we have the answer to elements one and two.

5. Implementation of algorithm

A. Problem definition phase

Know all the phases of the problem, as far as the learner can, to do this, he or she need to search what is X, learn about it, and learn more and more about the problem.

B. Solution phase

In the last phase learners searched for “what is X”, now they will search for “what are the possible solutions for X”, they will search for that in order to know, what did great people do so far, search a lot, read papers, search for conferences, collect as many solutions and as newest solutions as they can, study them will, compare them, understand them.

C. Decision phase

Now, the learners had learnt about the problem and its most recently solutions, ask yourself, do you think “I need” or “I can” make better? If you can be satisfied with one of the most recently developed solutions, then we are done, if not continue.

D. Development phase

This is the last and longest phase, here, learners need to start thinking about a solution which is faster than or better than the others, they may use one of them also and complete it, or they may use it as a reference, but this phase really don’t have a typical way to go through it, as they know, Archimedes had discovered the floating principle just by chance, the same lot of inventions and discoveries, you just have you, yourself, your mind, the nature and the luck to invent the algorithm which is the last phase in the algorithm of creating your algorithm.

6. Conclusion

Learning concepts discussed above may be beneficial not only for beginners but advanced learners also. For learning any subject prior understanding of the topic is mandatory. Algorithm studies can be made interesting and effective by following the above explained stages. Different learners tend to use different strategies based on their interest and commitment. Detailed explanation of various strategies described above can be useful for any learners who are passionate in learning algorithms.

References

[1] Ellis Horowitz, Sartaj Sahni, Sanguthevar Rajasekaran, 1998, Fundamentals of Computer Algorithms in Galgotia Publications Pvt. Ltd.
[2] Robert Sedgewick and Kevin Wayne, Algorithms, 4th Edition, Pearson Publications.
[3] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein, “Introduction to Algorithms,” MIT Press, 1990.