Algorithms Play a Vital and Helping Function in Developing Long, Efficient, Outstanding Programs

Ping He*

Department of Architecture, University of Zhaoqing, China

ping.123@gmail.com

Received: November 30, 2022, Manuscript No. MATHLAB-23-86764; **Editor assigned:** December 2, 2022, PreQC No. MATHLAB-23-86764 (PQ); **Reviewed:** December 16, 2022, QC No MATHLAB-23-86764; **Revised:** December 21, 2022, Manuscript No. MATHLAB-23-86764 (R); **Published:** December 28, 2022

Introduction

In mathematics and computer science, an algorithm is a finite sequence of exact instructions, typically used to solve a particular class of problems or perform computations. Algorithms are used as specifications to perform computations and data processing. More advanced algorithms perform automatic reasoning (called automatic reasoning) and use mathematical and logical tests to redirect code execution along different routes (called automatic decision-making). The metaphorical use of human characteristics as machine descriptors was already practiced by Alan Turing with terms such as “memory”, “retrieval” and “stimulus”.

Description

In contrast, heuristics are problem-solving approaches that may not guarantee correct or optimal results, especially in problem domains that are either not fully specified or where correct or optimal results are not apparently defined. Effectively, algorithms can be expressed in a well-defined formal language for computing functions in finite space and time. Starting with an initial state and an initial input (possibly empty), the instruction traverses a well-defined finite number of consecutive states at run time, eventually producing an “output” and a final ending state. Describes the computation to end. Some algorithms, called randomized algorithms, involve random inputs. There are many different recipes for preparing certain dishes that look different, but end up tasting the same. The same applies to algorithms. If a recipe calls for many complex ingredients that you don’t have, it won’t work as well as a simple recipe. When we think of algorithms as problem-solving tools, we often want to know how long it takes a computer to solve a problem using a particular algorithm. When writing an algorithm, we want it to take as little time as possible so that we can solve the problem as quickly as possible. In cooking, some recipes are more difficult to implement than others because they take the time or have a lot to keep track of. The same applies to algorithms, and algorithms are better the easier they are for computers to work with. The measure of algorithmic difficulty is called complexity. When we ask how complex an algorithm is, we often want to know how long it takes the computer to solve the problem it has to solve. In computer science, a sorting algorithm is an algorithm that sorts the elements of a list. The most commonly used orders are numeric and lexicographic, ascending or descending. Efficient sorting is important for optimizing the efficiency of other algorithms that require the input data to be sorted lists, such as search and merge algorithms. Sorting is also useful for normalizing data and producing human-readable output. Formally, the output of any sorting algorithm must satisfy two conditions: The output is in monotonic order (each element is neither lesser nor greater than the previous element, according to the desired order). The output is a permutation of the input (permuted while preserving all original elements). For optimal efficiency, input data should be stored in data structures that allow random access, rather than data structures that allow only sequential access. In some cases, it may be advantageous for a program to exhibit non-deterministic behavior. For example, the behavior of the card shuffler the program used in the game of blackjack should not be predictable to the player, even if the source code of the program is visible [1-4].

Conclusion

Using a pseudo-random number generator is often not enough to make the shuffle results unpredictable by the player. A clever player can guess exactly which numbers the generator will choose, predetermine the entire contents of the deck, and cheat. For example, Reliable Software Technologies’ Software Security Group was able to do this in their implementation of Texas Hold’em Poker distributed by ASF Software, Inc., allowing them to consistently predict hand outcomes in advance. These problems can be partially avoided by using a cryptographically secure pseudo-random number generator, but an unpredictable random seed must be used to initialize the generator. This requires a nondeterministic source, such as that provided by a hardware random number generator.

Acknowledgement

None.

Conflict of interest

The authors are grateful to the journal editor and the anonymous reviewers for their helpful comments and suggestions.

References

1. Rahman CM, Rashid TA (2019) Dragonfly algorithm and its applications in applied science survey. Comput Intell Neurosci 2019:9293617.
2. Howard J (2022) Algorithms and the future of work. Am J Ind Med 65(12):943-952.
3. Aziz S (2021) ECG-based machine-learning algorithms for heartbeat classification. Sci Rep 11(1):18738.
4. Zhang Y (2022) Unsupervised multi-class domain adaptation: Theory, algorithms, and practice. EEE Trans Pattern Anal Mach Intell 44(5):2775-2792.