Proceedings of the 2020 Winter Simulation Conference
K.-H. Bae, B. Feng, S. Kim, S. Lazarova-Molnar, Z. Zheng, T. Roeder, and R. Thiesing, eds.

MODELING AND SIMULATION FOR TIME-ACCURATE AND STOCHASTIC ANALYSIS OF ALGORITHMS

Abdurrahman Alshareef

Information Systems Department
College of Computer and Information Sciences
King Saud University
P.O. Box 4545
Riyadh, Riyadh 11451, SAUDI ARABIA

ABSTRACT

We propose a framework to perform algorithms analysis based on modeling and simulation. The framework supports time-accurate analysis of the target algorithm for a given input or set of inputs. It also can support a stochastic analysis by extension to allow the model to assign time advances according to some probability distribution. The framework facilitates performing the analysis of computational models within systems in a modularized manner. The granularity of the analyzed instruction as well as the time structure can be determined during the modeling process using the Activity abstraction and DEVS-based support for the simulation. The initial portion of the simulation models is generated automatically and then modified afterward to account for more concrete and operational instructions.

1 INTRODUCTION

Asymptotic analysis (Knuth 1976) has been widely used for evaluating the computational complexity of algorithms. It is fundamental in evaluating and promoting the performance of system with computational elements. It can be, however, extended in various ways to further facilitate the analysis of algorithms. We select three ways in which modeling and simulation can add up to such analysis. The first one is to account for the specific input samples in order to make the analysis more rigorous with respect to the notion of sufficiently large input. The second one is to make algorithms analysis subject to stochasticity, which can be expected in the actual computational environment. A probabilistic distribution assigns to the time advance function in the simulation in order to create such account for the time complexity analysis. The third one is to allow for specialized time allocations for different instructions and steps in the algorithm. In overall, the level of granularity in such accounts can be determined by the modeler and the level of details provided using the proposed abstraction as well as in the corresponding models.

2 MODELING THE TARGET ALGORITHM

In order to create a suitable entry point for modeling, we start by drawing an Activity for the algorithm that is being target of the analysis. We have previously proposed an Activity specification (Alshareef 2019) using the DEVS (Discrete Event System Specification) formalism (Zeigler et al. 2018). The Activity specification maps different elements in the abstraction to different DEVS models according to predefined set of rules. The Activity is mapped to a coupled model. The actions and control nodes are mapped to atomic models for handling delays, synchronization, and selection across the flow. We choose the quicksort algorithm as a commonly discussed example for sorting algorithms as well as their complexity analysis. Figure 1 shows the Activity that corresponds to the algorithm. The diagram highlights the steps starting by receiving an array, traversing from the lowest element to the highest element, and initializing the variable *i*

and *pivot*. Then, the partitioning procedure starts with a loop from j^{th} to *high* element by incrementing *i* and swapping i^{th} element with j^{th} element after making sure the j^{th} element is less than *pivot*. Then, it swaps $(i+1)^{th}$ element with high element and repeat for the resulting two partitions until the end. After completing the diagram, the code generation takes place automatically and results in a set of Java classes suitable for the initial simulation in DEVS-Suite simulator as atomic and coupled models. For simpler models that might be sufficient. However, for relatively advanced algorithms, some modifications and refinements have to take place on the resulting models in order to set some parameters and specializes the model with the necessary low-level details and operations that are not suitable to be specified in the higher level abstraction.

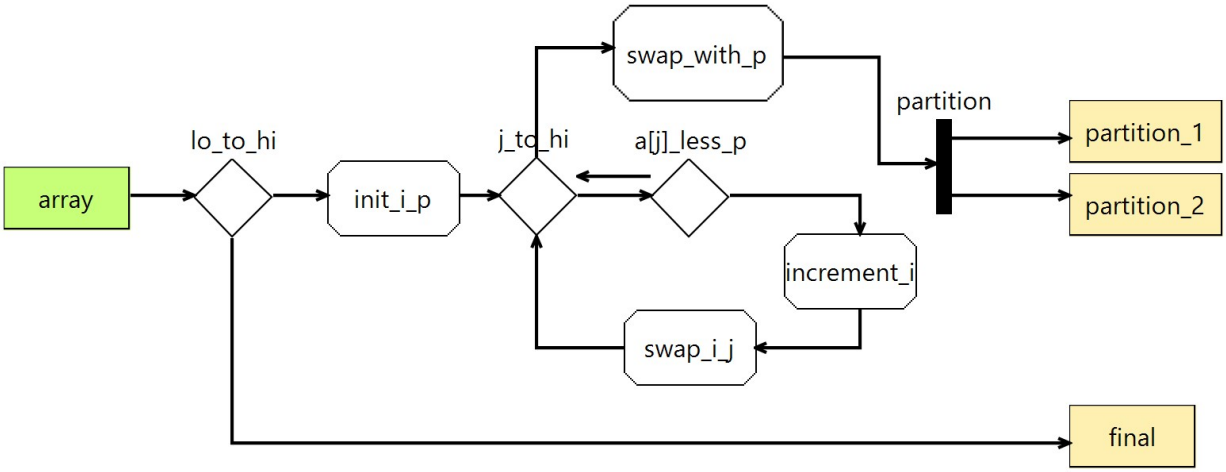

Figure 1: An activity abstraction of the quicksort algorithm.

The algorithm performs according to the Big O notation $O(n^2)$ in the worst case and $O(n \log n)$ in the average case. We conduct a simulation with different settings according to the input, input size, and timing assignments. The result of a typical simulation is consistent with the result of the formal analysis given by the Big O notation. Nevertheless, the results can differ based on the changes and calibrations made at a finer level of details regarding the time needed per step as well as the determined mean and distribution for the identified micro steps of the procedure as highlighted in the abstraction. It also highlights specific cases of inputs and input sizes. Table 1 shows the results of some of the conducted simulations.

Table 1. The termination time of the simulation for various cases.

Input size	Post-ordered Balanced Binary Search Tree	Random-case	Worst -case
10	71	118	180
100	1576	1367	15343

In conclusion, the framework can facilitate performing analyses for systems with computational element to extend the notion of the computational complexity analysis in a broader, inter-disciplinary context for complexity in systems, and system of systems. We show some aspects of the framework with demonstrative example. We plan to provide further details with other examples, observations, and analyses in future work.

REFERENCES

Alshareef, A. 2019. *Activity Specification for Time-based Discrete Event Simulation Models*. Ph.D. Dissertation, Arizona State University, Tempe, Arizona. https://repository.asu.edu/items/55474, accessed 9th November 2020.

Knuth, D.E. 1976. "Big Omicron and Big Omega and Big Theta". *ACM SIGACT News* 8(2):18-24.

Zeigler, B.P., A. Muzy, and E. Kofman. 2018. *Theory of Modeling and Simulation: Discrete Event & Iterative System Computational Foundations*. 3rd ed. Academic Press.