A Visual and Interactive Learning Tool
for CPU Scheduling Algorithms

Sukanya Suranuwarat

Graduate School of Applied Statistics, National Institute of Development Administration,
Bangkok 10240, Thailand

Abstract

CPU scheduling is an important topic in operating systems courses. In this paper, a tool implemented as a Java application
and designed as an auxiliary instrument for both classroom teaching and independent study of CPU scheduling algorithms is
presented. This tool uses graphical animation to convey the concepts of various CPU scheduling algorithms. The tool is
unique in a number of respects. First, it uses a more realistic process execution model that can be configured easily by the
user. Second, it graphically depicts each process in terms of what the process is currently doing against time. By using this
representation, it becomes much easier to understand what is going on inside the system and why a different set of processes is
a candidate for the allocation of the CPU at different times. Third, the tool allows the user to test and increase his
understanding of the concepts studied by making his own scheduling decisions and receive immediate feedback on the test
problems.

Keywords: Educational Software, Animation Tool, Computer Science Education, CPU Scheduling Algorithms, Operating
System.

1. Introduction

In the past two decades, a number of visualization and animation tools have been developed and used in many
areas of computer science and engineering education [1]-[8]. Experiments carried out with various visualization
and animation tools have provided evidence indicating that carefully designed visualizations and animations can
have beneficial learning effects. For example, engagement of the learners attention [9]-[11] and the ability to control
the pace of the visualization [12] appear to be key factors in building effective visualization and animation tools.
Keeping these in mind, the author has developed an interactive Java-based simulator that uses graphical
animation to convey the concepts of various CPU scheduling algorithms for a single CPU. CPU scheduling
can be defined as the art of determining when and for how long each process runs on the CPU when there are
multiple runnable processes. It is central to an operating-system's design and constitutes an important topic in the
computer science curriculum.

In addition to providing a visual and animated view as an alternative to a static representation provided by
textbooks, the simulator is unique in a number of respects. First, it uses a more realistic process execution model —
the execution of a process consists of alternating CPU bursts and I/O bursts, as opposed to a simplified model
used in textbooks examples — only one CPU burst per process. Through a graphical user interface of the
simulator, the user can configure several sets of processes easily and use them in observing simulations of various
CPU scheduling algorithms. By using a more realistic process execution model, users will be able to gain insight
into exactly how the algorithms work in real operating systems. Second, the simulator graphically depicts each
process' state versus time. The state of a process describes the current activity of that process such as "the
process is waiting for an I/O operation to complete" or "the process is currently using the CPU". Various events
can cause a process to change states; the simulator shows these events. By using this representation, it becomes
much easier to understand what is going on inside the system, why, at any given time, some processes are
candidates for the allocation of the CPU and some are not, and why the currently running process can continue using
the CPU or why it cannot. Third, the simulator allows the user to practice and test his understanding of the concepts
studied by making his own scheduling decisions (i.e., by deciding when and for how long each process runs)
through an easy-to-use graphical user interface of the simulator, and receive immediate feedback on the test
problems.

The simulator can be used as an auxiliary instrument for both classroom teaching and independent study of CPU
scheduling algorithms.

The remainder of this paper is organized as follows: section 2 is a brief overview of the process state and
scheduling algorithms used in the simulator, section 3 gives a description of the simulator, section 4 discusses
versions and availability of the simulator, section 5 discusses related work, and section 6 draws some
conclusions.