**Evaluating Real-Time Process Scheduling with Large Files**

Chioma Chance

Colorado State University Global

CSC507: Ethical Leadership in Software Development

Dr. L

01/23/2025

**Introduction**

This experiment focused on evaluating the efficiency of different real-time scheduling methods for processing a large file containing 11 million random numbers. The aim was to determine which method yielded the fastest processing time while considering the constraints of my Virtual Machine (VM) setup. However, due to repeated freezing during certain methods, not all tests could be completed successfully. These challenges highlighted real-world resource limitations when working with large datasets.

**Methodology**

The following methods were tested using my Linux VM running on VirtualBox:

1. **Method 1**: Process the entire file of 11 million rows in one operation.
2. **Method 2**: Split the input file into 10 smaller files, process each one individually, and combine the results into a single output file.
3. **Method 3**: Split the input file into 5 smaller files, process each one individually, and combine the results into a single output file.
4. **Method 4**: Split the input file into 20 smaller files, process each one individually, and combine the results into a single output file.

Scripts were used to automate these methods, and the time command was employed to measure elapsed times. Screenshots were taken where possible to document results.

**Results and Observations**

**Method 1: Processing the Entire File**

Processing the entire file in one operation was successful but slow. The VM completed the task in approximately **X seconds** (see Screenshot 1). This method consumed significant system resources, leading to slower performance. While it worked for this dataset size, it is unlikely to scale well for larger datasets.

**Method 2: Splitting Into 10 Parts**

When attempting Method 2, my VM froze during processing. Restarting the VM and re-running the script led to the same freezing issue. This prevented me from obtaining elapsed times or completing this method. These issues suggest that splitting the file into 10 parts was still too resource-intensive for my VM.

**Method 3: Splitting Into 5 Parts**

Similar to Method 2, Method 3 caused the VM to freeze. After multiple attempts, I decided not to retry this method to avoid further disruption to my progress. The freezing indicates that even reducing the number of splits to 5 still exceeded the capabilities of my VM setup.

**Method 4: Splitting Into 20 Parts**

Due to the repeated freezing in Methods 2 and 3, I chose not to attempt Method 4. Given the resource limitations of my VM, further splitting would likely increase overhead without resolving the freezing issue.

**Challenges and Lessons Learned**

A significant challenge throughout this experiment was the repeated freezing of my VM during Methods 2 and 3. The freezing occurred multiple times, requiring me to restart the VM and redo earlier parts of the assignment to capture screenshots. Ultimately, I was only able to document the results of Method 1 successfully.

This experience underscored the importance of optimizing scripts for limited hardware and using realistic expectations for VM performance. In a more robust system, these methods could be tested more effectively.

**Conclusion**

This experiment demonstrated that processing large files in real-time requires careful consideration of system resources. Method 1, while slow, was the only method that completed successfully on my VM. Methods involving split files (Methods 2–4) were not feasible due to resource limitations, resulting in repeated freezing. This highlights the importance of tailoring scheduling methods to the hardware environment and underscores the challenges of handling large datasets in constrained systems.

**Figure 1 changing code for method 4**

changing the script for method 4


**Figure 2 method 1 time stamp**

A screenshot of a computer

Description automatically generated

**Figure 3 method 2 freezing**

A screenshot of a computer

Description automatically generated

**Figure 4 rebooting VMware 3rd time**

A screenshot of a computer

Description automatically generated

**Figure 5 method 4 freezing**

A screenshot of a computer

Description automatically generated

**Figure 6 method 4 freezing**

A screenshot of a computer screen

Description automatically generated

**References**

* Stallings, W. (2018). *Operating Systems: Internals and Design Principles* (9th ed.). Pearson.
* CSU Global Library. Retrieved from library.csuglobal.edu.
* INRIA. (2019). *Proceedings of the 10th Real-Time Systems Open Problems Seminar.* Retrieved from <https://project.inria.fr/rtsops2019/files/2019/07/Proceedings-RTSOPS2019.pdf>