CS 3220-AM,BM,CM: Operating Systems

Quiz 2

Apr 22nd, 2021

(**Total Score**: 10 points)

(**Questions**: 3)

(**Duration**: 15 minutes)

Instructions:

Please turn off all your phones and electronic devices and keep them inside. This is a Closed Book/ Closed Notes Quiz. Write your Name and Student ID on all Quiz sheets. Usage of Scientific Calculators is allowed. Labels of all figures and tables are mentioned below the figures and tables respectively. Show all your working where required. Provide all numerical results with two digits of precision only.

Q 1. Assume that you are trying to download a large 4 GB file from Internet. The file is available from a set of Mirror Servers, each of which can deliver a subset of the file's bytes. Assuming that a given request specifies the starting and ending bytes of the file, explain how you might use Threads to improve the download time. **(4 points)**

Q 2. Under what circumstances is a single-threaded system better than a multi-threaded system? **(4 points)**

Q 3. Why would a thread ever voluntarily give up the CPU by calling $thread\_yield$? After all, if there is no periodic clock interrupt, it may never get the CPU back. **(2 points)**

A 1. The downloading Process can assign each of its Threads the task to request a subset of the file's bytes from the Mirror Servers. In this way, instead of downloading a large 4 GB file sequentially, it will be downloaded using pseudoparallelism, and hence will reduce the total download time. Later on, these subsets can be merged with the help of another software to re-develop the original 4 GB file.

A 2. A single-threaded system is better than a multi-threaded system when Processes running in the system are CPU-intensive, and thus do not require I/O much (do not get blocked often), and hence do not require Threads. Also, if functions inside a Process are dependent on one another, and can not execute independently (in parallel), there is no need of Threads, and in result a multi-threaded system.

A 3. *thread_yield* is usually used for Threads that are expected to finish their work in short durations, and eventually return the control to the calling Thread. The developer designs the program flow in order to efficiently finish the program execution, where *thread_yield* might be a helping component.