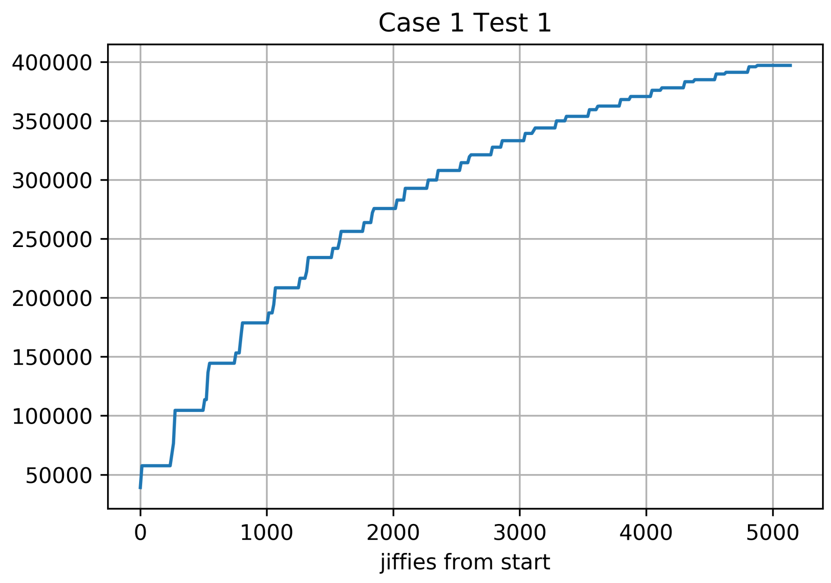
CS423 MP3 Report

Ziang Wan, ziangw2

April 13, 2018

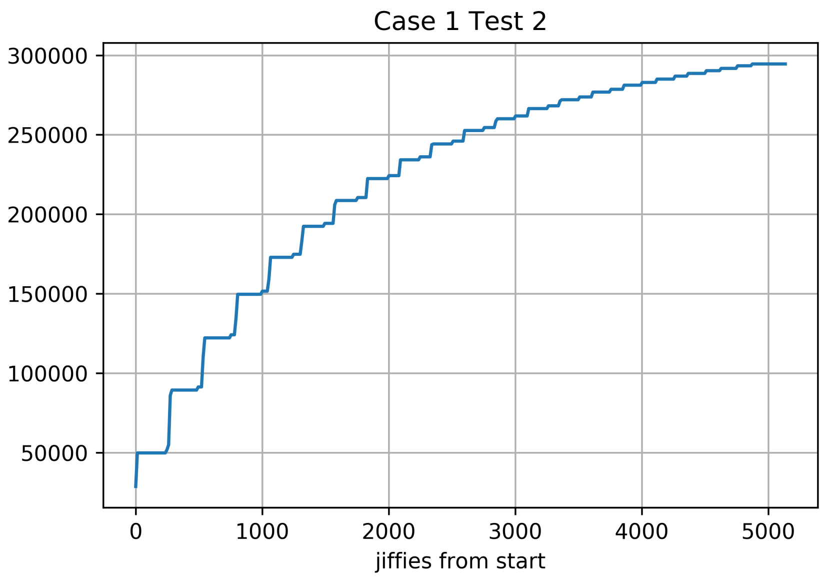
Case Study 1

(1) The first run:
Work process 1: 1024MB Memory, Random Access, and 50,000 accesses per iteration
Work process 2: 1024MB Memory, Random Access, and 10,000 accesses per iteration



NOTE:
The graph is time vs. accumulated minor page fault. There is no major page fault happened during the first run.

(2) The second run:
Work process 3: 1024MB Memory, Random Locality Access, and 50,000 accesses per iteration
Work process 4: 1024MB Memory, Locality-based Access, and 10,000 accesses per iteration



NOTE:
The graph is time vs. accumulated minor page fault. There is also no major page fault happened during the second run.

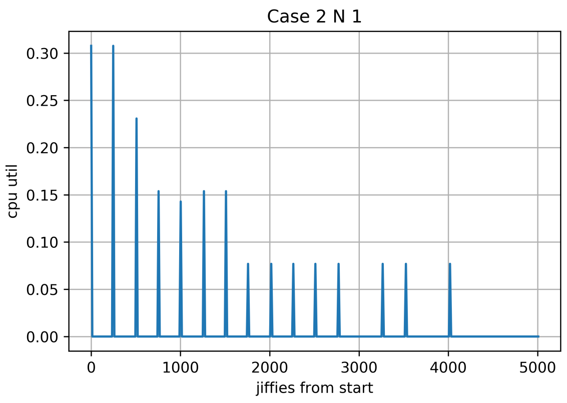
Analysis

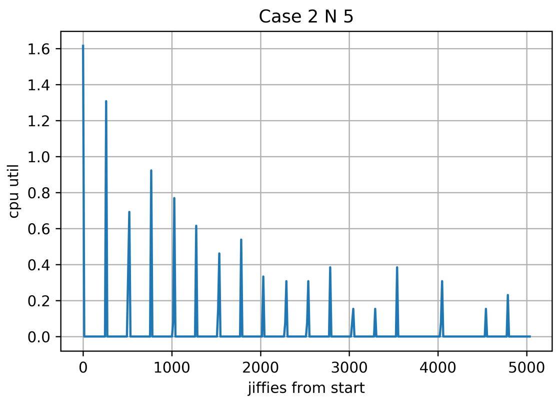
There is more minor page faults happened during the first run than the second run. The reason is that, in the second run, we make process 4 to have locality-based access; the memory access pattern of the second run will exhibit more locality than the that of the first run, leading to less minor page faults.

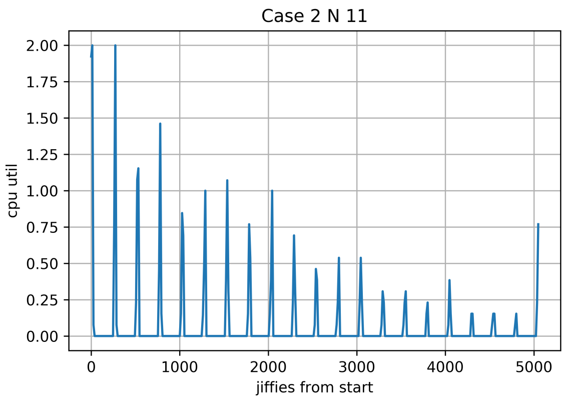
The completion times of the two run are almost the same. Memory access pattern does not affect completion time.

Case Study 2

(1) Three runs: we concurrently run 1, 5, and 11 the following working processes: Work process 5: 200MB Memory, Random Locality Access, and 10,000 accesses per iteration





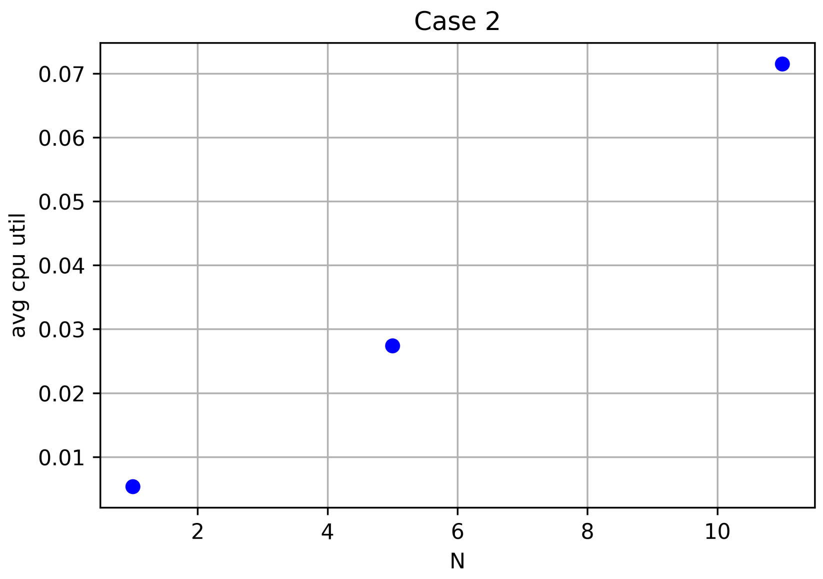


NOTE: The graphs are time vs. CPU utilization rate. CPU utilization rate is expressed as:

$$\frac{\text{utime} + \text{stime}}{\text{wall time}}$$

per each 50ms time interval. The maximum of CPU utilization rate is 2.0, which means 200% as there are two cores in this VM.

(2) Put them all together, I have a graph of average CPU utilization rate vs. N:



Analysis

Average CPU utilization rate goes up as N goes up. There is no thrashing observed. The work processes exhibit a CPU usage peak every second. As the degree of multi-programming goes up, the magnitude of the peak of CPU usage also goes up.

With more working processes running concurrently, the completion time of the whole run gets longer. At N=1, CPU utilization rate goes to zero earlier than at N=11.