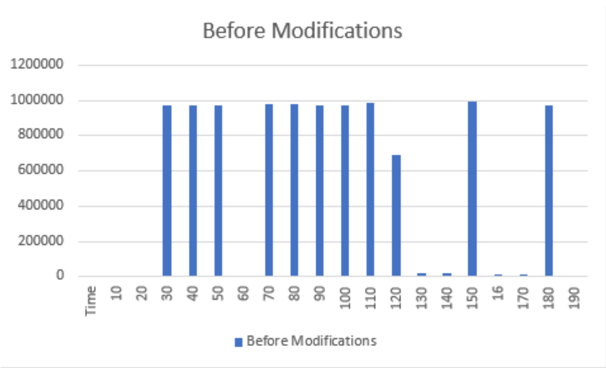
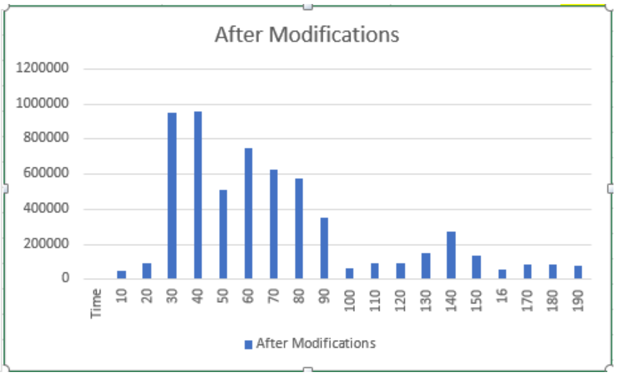
Assignment 3 - Implementation of Slim Chance Page Replacement Algorithm for FreeBSD Memory Management System

Assignment By: Ahmed SalahEldin Farouk Elkashef 1410216

Findings, Graphs, and numbers

Total number of pages scanned:



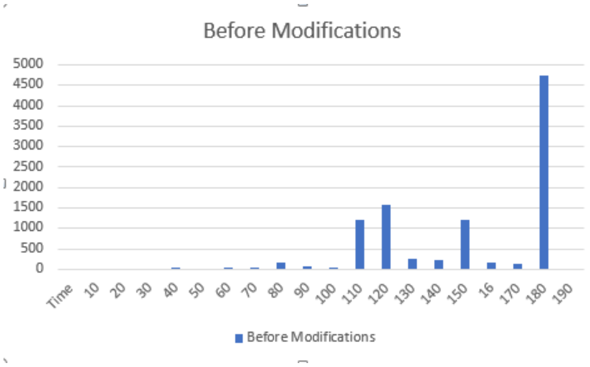


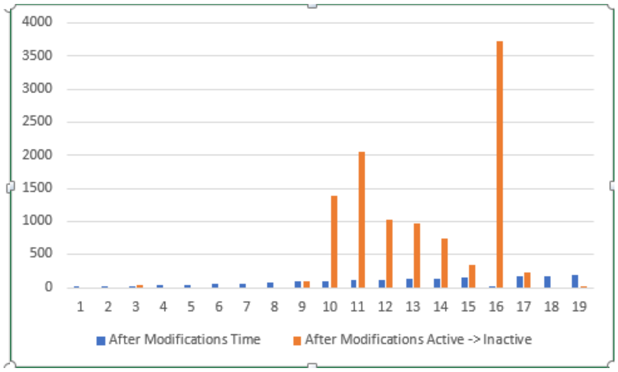
Average – Before Modifications: 553003.1579

Average – After Modifications: 314242.3158

Since the activity count is being halfed every turn instead of just doing simple subtraction, it seems like the average number of total pages is also closer to the half.

Active to Inactive:



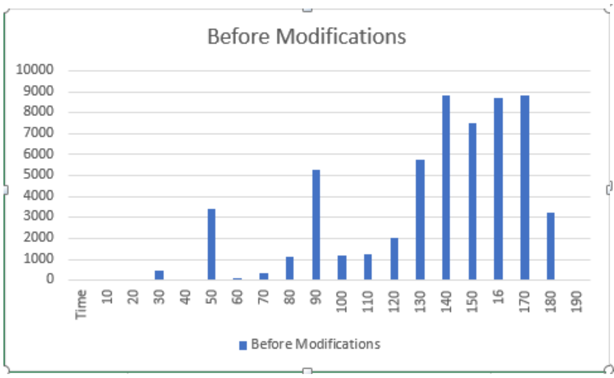


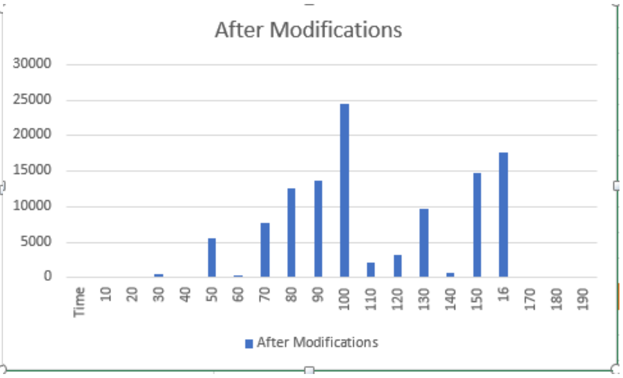
Average – Before Modifications: 520.2631579

Average – After Modifications: 561.5789474

According to the definition of "slim chance algorithm" this algorithm does not give the page a lot of chances like its relative "second chance" and thus it makes sense that after modification, more pages will become inactive than before.

Inactive to Cache:



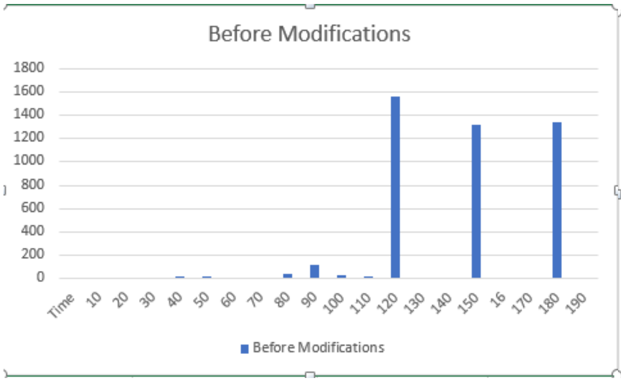


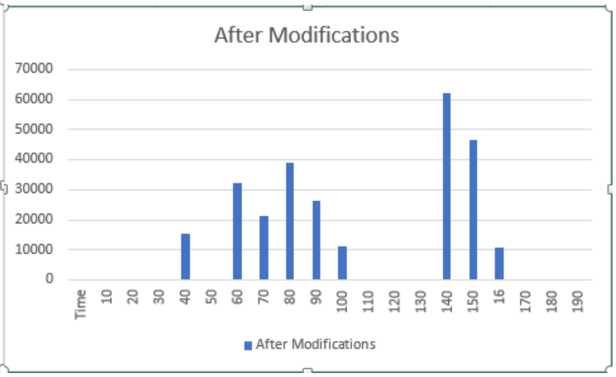
Average – Before Modifications: 3050.842105

Average – After Modifications: 5963.789474

It seems like the slim chance algorithm is not an efficient algorithm, from the numbers, the number of pages being moved to cache is very big compared to the second chance algorithm that the freebsd is already utilizing.

Queued for Flush:





Average – Before Modifications: 231.8947368

Average – After Modifications: 13961.68421

In the slim chance algorithm, the number being moved to flush is huge compared to the default implementation. Slim chance does not give as much "chances" to those pages and thus, its expected to see a larger number of pages being flushed.