Members: Mackenzie Huynh (Captain)
Anthony Lin,
Khachatur Mirijanyan,
Priya Calaimany
Team: Evanescence
Date: March 19, 2019

Benchmarks

System Configuration:

The benchmarks were performed on the following hardwares: 1 CPU, 2 GB RAM (2048 MB).

***NOTE**: The logging will only be logged when the ./benchmark.sh script is run, as this was an easier way for us to collect results from the log when we ran the benchmarks ourselves. The logs are logged into the /var/log/debug.log file. Also, we provided our .logs results in the /logs folder.

In order to run our tests, we first ran the benchmarks (./benchmark.sh) 5 times using the original paging daemon algorithm implemented by FreeBSD and collected the results from the debug.log. Our benchmark uses the stress tester recommended on Piazza[1], so the "stress" program must be installed to run the benchmark.

Our benchmark will turn on debugging by changing a system variable, **vm.do_logging**; then it will run stress. After the stress test, our benchmark runs vmstat to print statistics about the system, and it turns off logging. This is how we collected our results for each benchmark run.

After testing the original page daemon implemented by FreeBSD, we restarted the VM and switched to the Bon Chance algorithm by running the switch script: ./switch 1, which changed the paging daemon algorithm to Bon Chance. Then we ran the benchmarks (./benchmark.sh) 5 times again and collected the results from the debug.log.

A comparison of our results from both page daemon algorithms is below.

[1] https://people.seas.harvard.edu/~apw/stress/

Logging:

In order to record our results, we used the field *vm_pagequeue->pg_cnt* and created 5 other, original variables:

Original Variable	Purpose
scanned	number of pages scanned in that pageout
active_to_inactive_counter	number of pages from active queue to inactive queue
inactive_to_active_counter	number of pages from inactive queue to active queue
inactive_to_free_counter	number of pages from inactive queue to free list
active_scan_occured / inactive_scan_occured	is set whenever a scan actually occurs in the queues respectively.

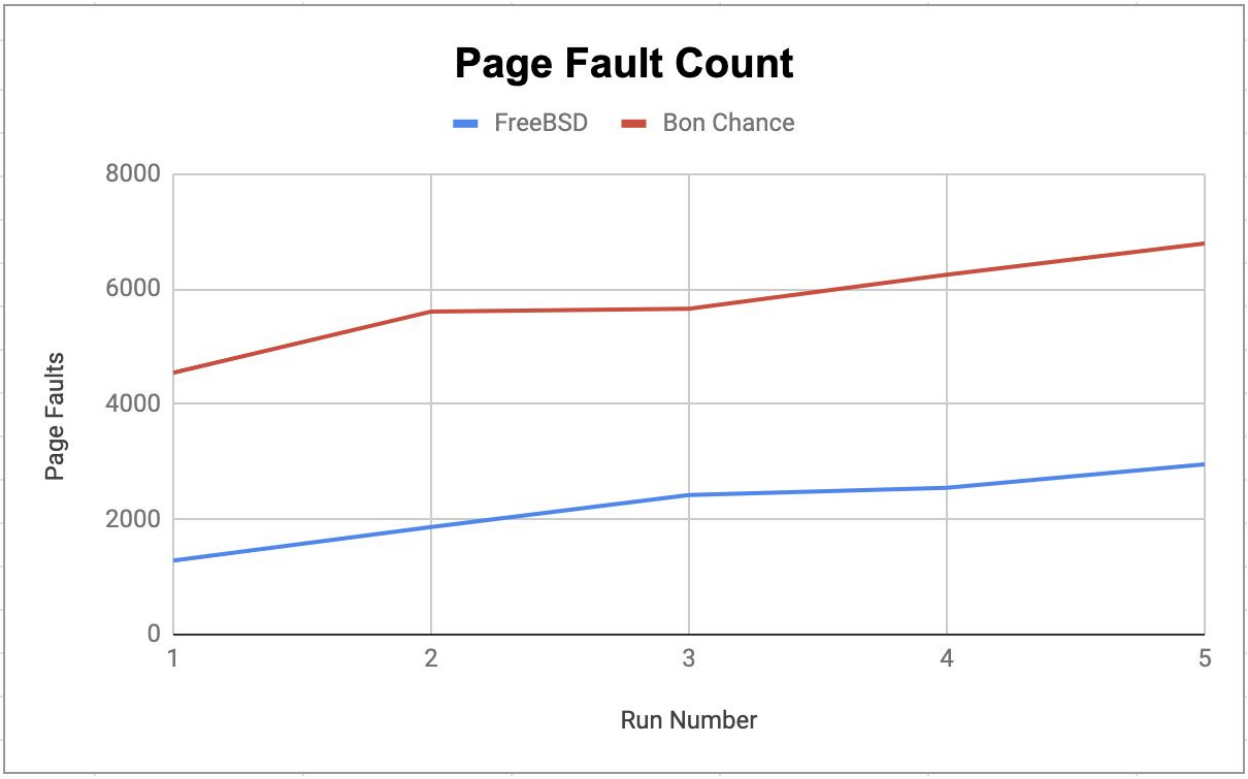
NOTE*: The values for page counts in the Active, Inactive, and Free Queues are similar or the same in the Bon Chance implementation and the default FreeBSD implementation. This can be expected because the number of pages in each queue should not be different, since we are just changing how the pages get put onto the lists. The pages still get put onto the queues, just in a different order; therefore, the total number of pages will remain around the same.

Results

Page Fault Count

Extraction of Data:
We obtained this data from running the stress tester and then running *vmstat*.

Analysis:
Our Bon Chance implementation had more than double the amount of page faults throughout the five runs of the stress tester. This tells us that Bon Chance is much worse than FreeBSD's default implementation. This is due to the fact that we are decaying the activity count much faster for active pages, so pages that are still needed may be swapped out due to the activity count reaching zero too quickly. Additionally, we are randomly placing pages in either the front or back of the free and inactive lists.



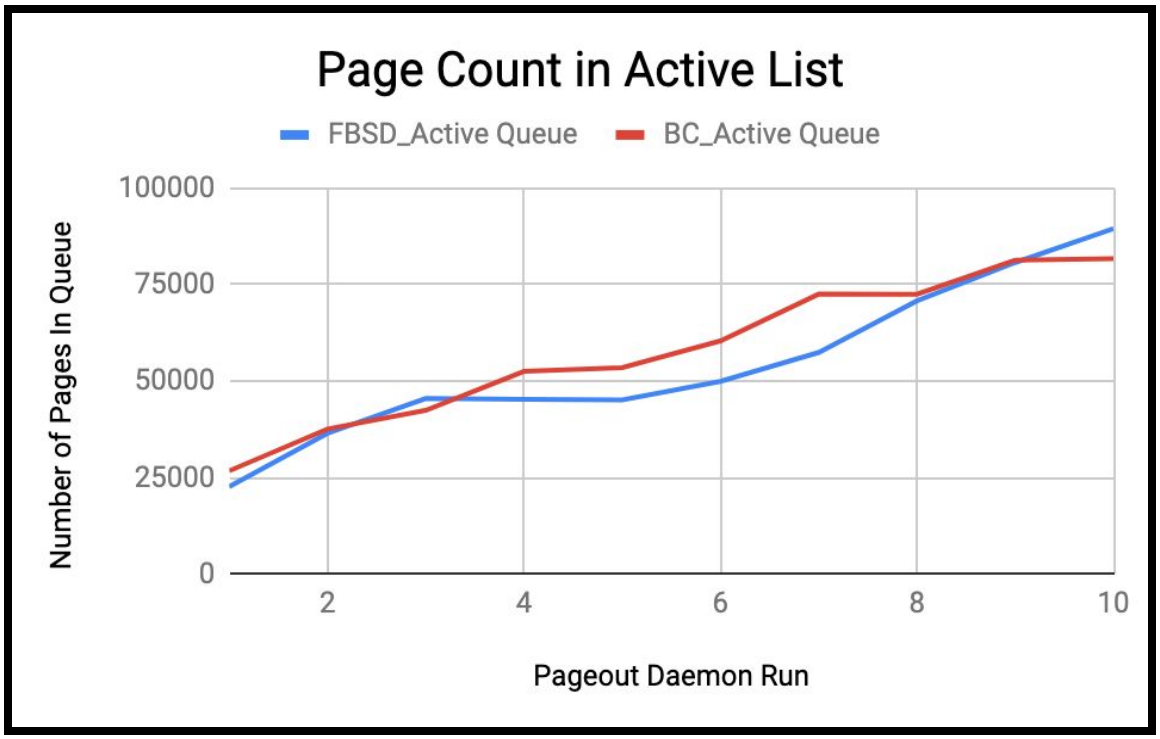
Page Count in Active List

Extraction of Data:
We extracted this data from the field *vm_pagequeue->pg_cnt.*

Analysis:
In general, the number of pages in the Active Queue increases with each run of the page daemon for both the Bon Chance algorithm and FreeBSD's algorithm. Because pages are referenced repeatedly in consequent passes as the page daemon runs in both algorithms, they are moved to the lower queue (a.k.a. Active Queue).

The average number of pages in the Active Queue when implementing the Bon Chance algorithm is slightly greater than the average number of pages in the Active Queue when implementing FreeBSD's algorithm. We believe, however, that this result is just anomalous. We do not believe that pages are transferred to the Active Queue at a higher rate for Bon Chance than for FreeBSD's algorithm.

Averages:
(FreeBSD) FBSD_Active Queue Average: **54346.1**
(Bon Chance) BC_Active Queue Average: **58138.1**



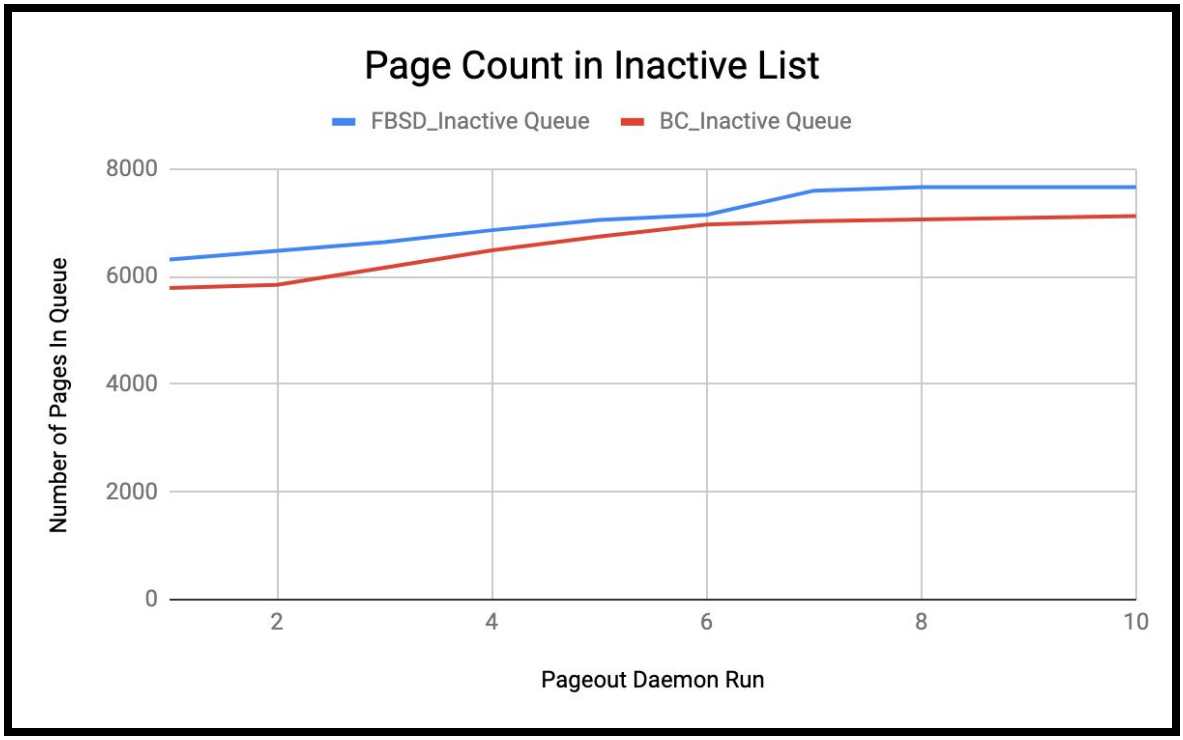
Page Count in Inactive List

Extraction of Data:
We extracted this data from the field *vm_pagequeue->pg_cnt.*

Analysis:
In general, the number of pages in the Inactive Queue increases with each run of the page daemon for both the Bon Chance algorithm and FreeBSD's algorithm. This is because pages are

The average number of pages in the Inactive Queue for Bon Chance is consistently lower than that for FreeBSD. This could be an anomaly between runs, or it could indicate that the random nature of Bon Chance leads to less pages in the Inactive Queue for this set of runs.

Averages:
(FreeBSD) FBSD_Inactive Queue Average: **7106.5**
(Bon Chance) BC_Inactive Queue Average: **6629.2**



Page Count in Free List

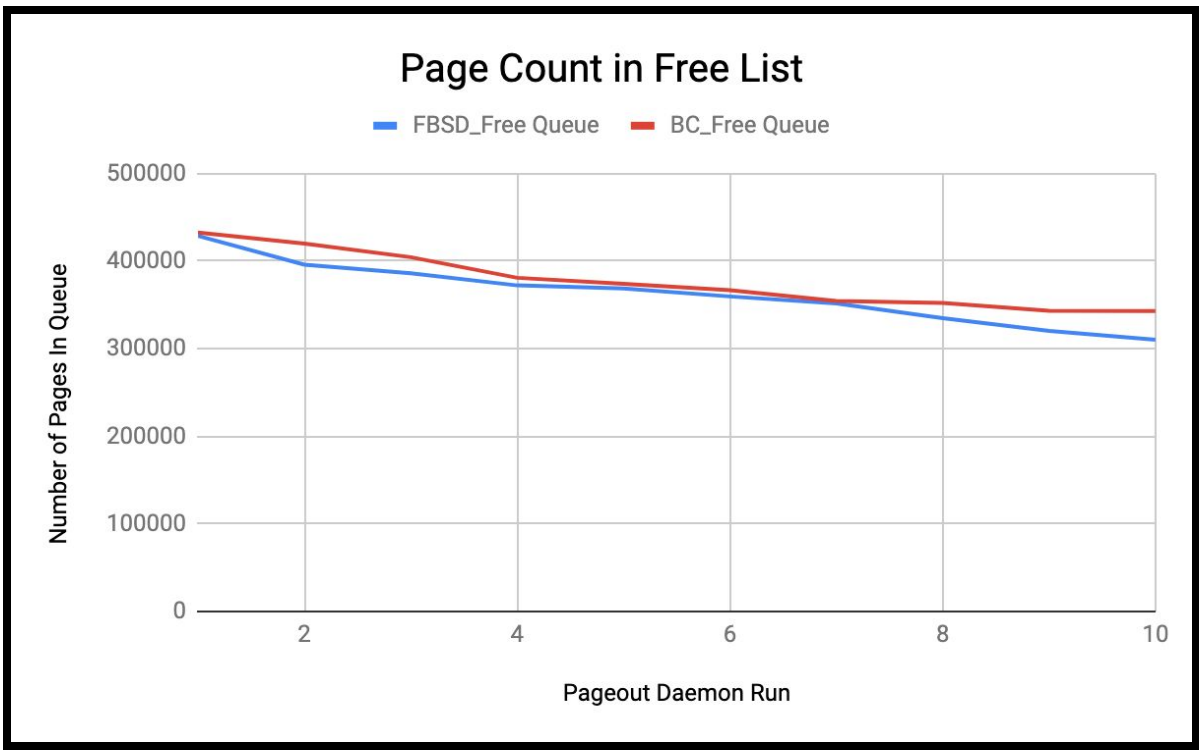
Extraction of Data:
We extracted this data from the field *vm_pagequeue->pg_cnt.*

Analysis:

Both algorithms started at about the same value for number of pages in the Free Queue. Bon Chance, however, has slightly more pages in the Free Queue than FreeBSD at the end of the pageout runs. This is also apparent near the beginning of the benchmark, so it is possible that a correspondingly larger number of pages are being moved back to the Free Queue for Bon Chance. It could also just be an anomaly.

Conversely, Bon Chance has a more randomized method of deciding when to transfer pages from the Inactive Queue to the Free Queue and sometimes puts recently Inactive pages on the back of the Free Queue. This could also explain the difference in page count and why there are slightly more pages in the Free Queue for Bon Chance.

Averages:
(FreeBSD) FBSD_Free Queue Average: **362845.9**
(Bon Chance) BC_Free Queue Average: **377166.3**



Page Count in Laundry List

Extraction of Data:
We extracted this data from the field *vm_pagequeue->pg_cnt.*

Analysis:

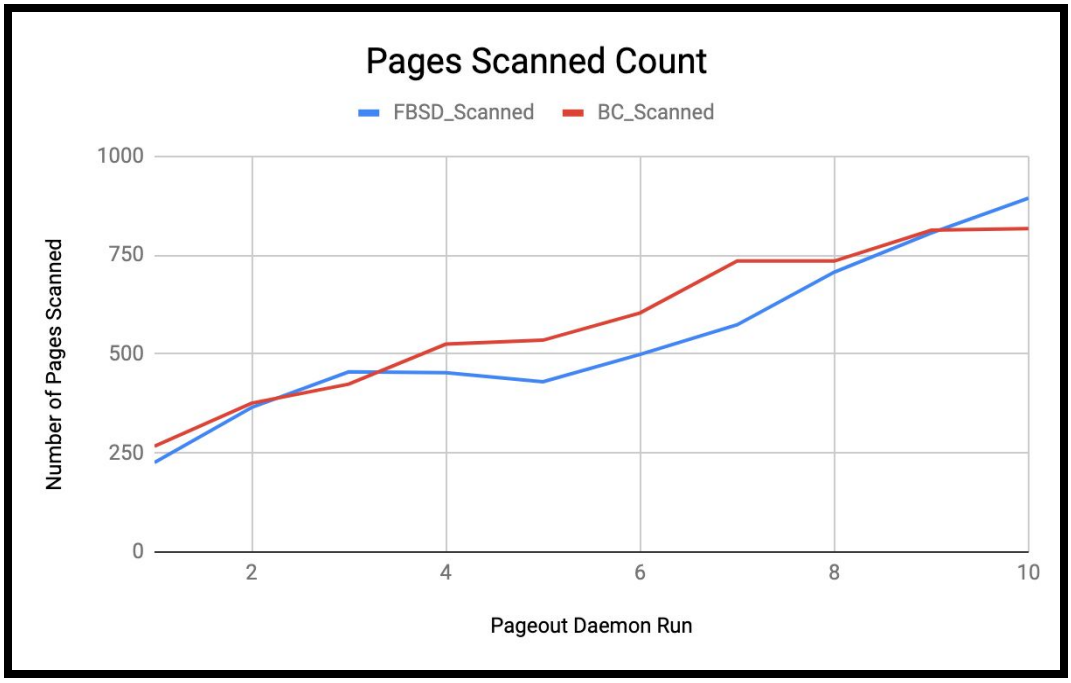
We did not include a graph for the number of pages in the Laundry Queue. For both algorithms, we found that the number of pages in the Laundry Queue is 0 throughout all pageout daemon runs.

Pages Scanned Count

Extraction of Data:
We extracted this data from a variable we created, *scanned*. We placed *scanned* in the "act_scan" section of the *vm_pageout_scan_active()* and *vm_pageout_scan_inactive()* functions. By doing this, we obtained the total number of pages scanned.

Analysis:
The Bon Chance average for number of pages scanned is higher than the FreeBSD average. This is because in the Bon Chance implementation, the scans may need to be run for longer on each queue, since the order in which we are placing pages on the queue is random and not optimal.

Averages:
(FreeBSD) FBSD_Scanned: **540.9**
(Bon Chance) BC_Scanned: **583.1**



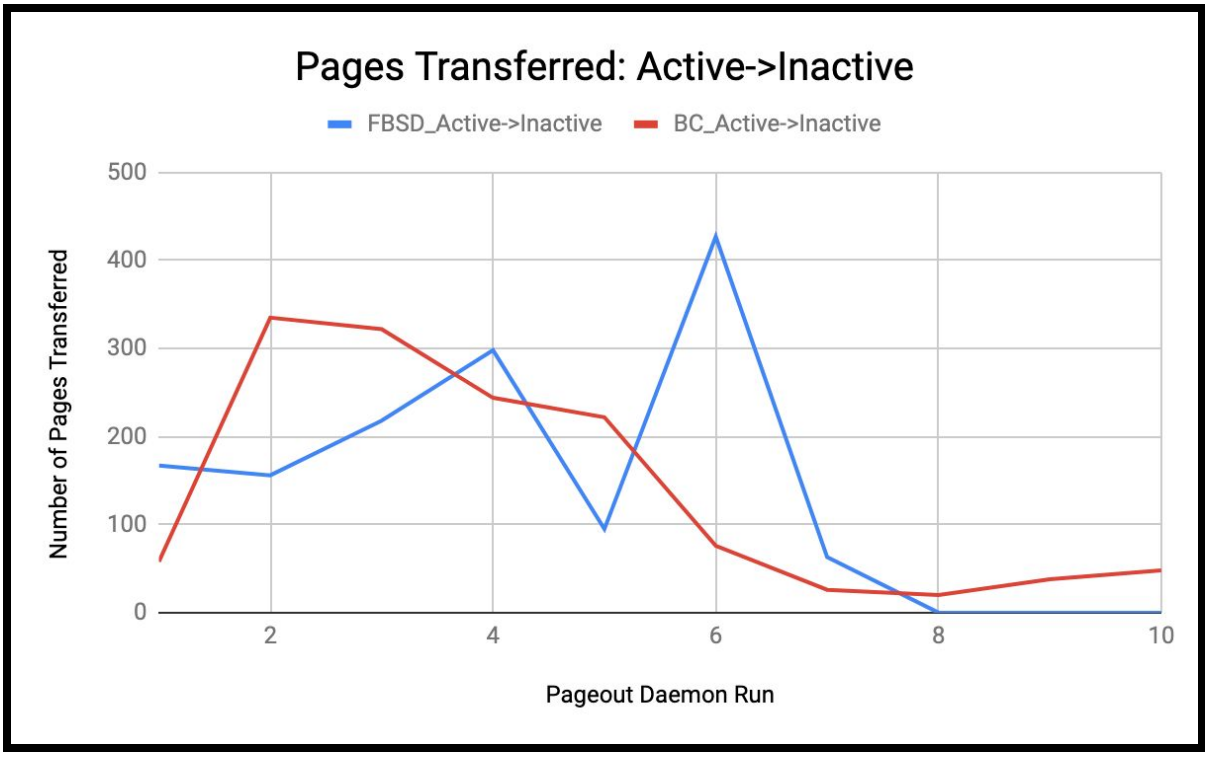
Pages Transferred from Active (Lower) to Inactive (Higher)

Analysis:

Extraction of Data:
We extracted this data from a variable we created, *active_to_inactive_counter*. We placed *active_to_inactive_counter* in the "act_scan" section of the *vm_pageout_scan_active()* function.

Trend:
The number of pages transferred from the Active Queue to the Inactive Queue spikes sharply at the beginning of the runs because most of the pages have not been referenced yet. As pages are referenced more frequently, the number of pages being transferred from the Active Queue to the Inactive Queue decreases. For the FreeBSD algorithm, the number of pages being transferred appears to spike and drop regularly. For the Bon Chance algorithm, because of the way in which pages are selected, the number of pages transferred follows a smoother curve.

Averages:
(FreeBSD) FBSD_Active->Inactive: **142.4**
(Bon Chance) BC_Active->Inactive: **138.9**



Pages Transferred from Inactive (Higher) to Free

Analysis:

Extraction of Data:
We extracted this data from a variable we created, *inactive_to_active_counter*. We placed *inactive_to_active_counter* in the "recheck" section of the *vm_pageout_scan_inactive()* function.

Trend:
The number of pages transferred from the Inactive Queue to the Free Queue spikes sharply at the beginning of the runs because most of the pages have not been referenced yet. As pages are referenced more frequently, the number of pages being transferred from the Inactive Queue to the Free Queue decreases. For the FreeBSD algorithm, the number of pages being transferred appears to spike and drop regularly. For the Bon Chance algorithm, because of the way in which pages are selected, the number of pages transferred follows a smoother curve.

Averages:
(FreeBSD) FBSD_Inactive->Free: **3487.7**
(Bon Chance) BC_Inactive->Free: **1804.4**