Changing Linux Page Replacement Algorithm

Zihan Wang

Overview

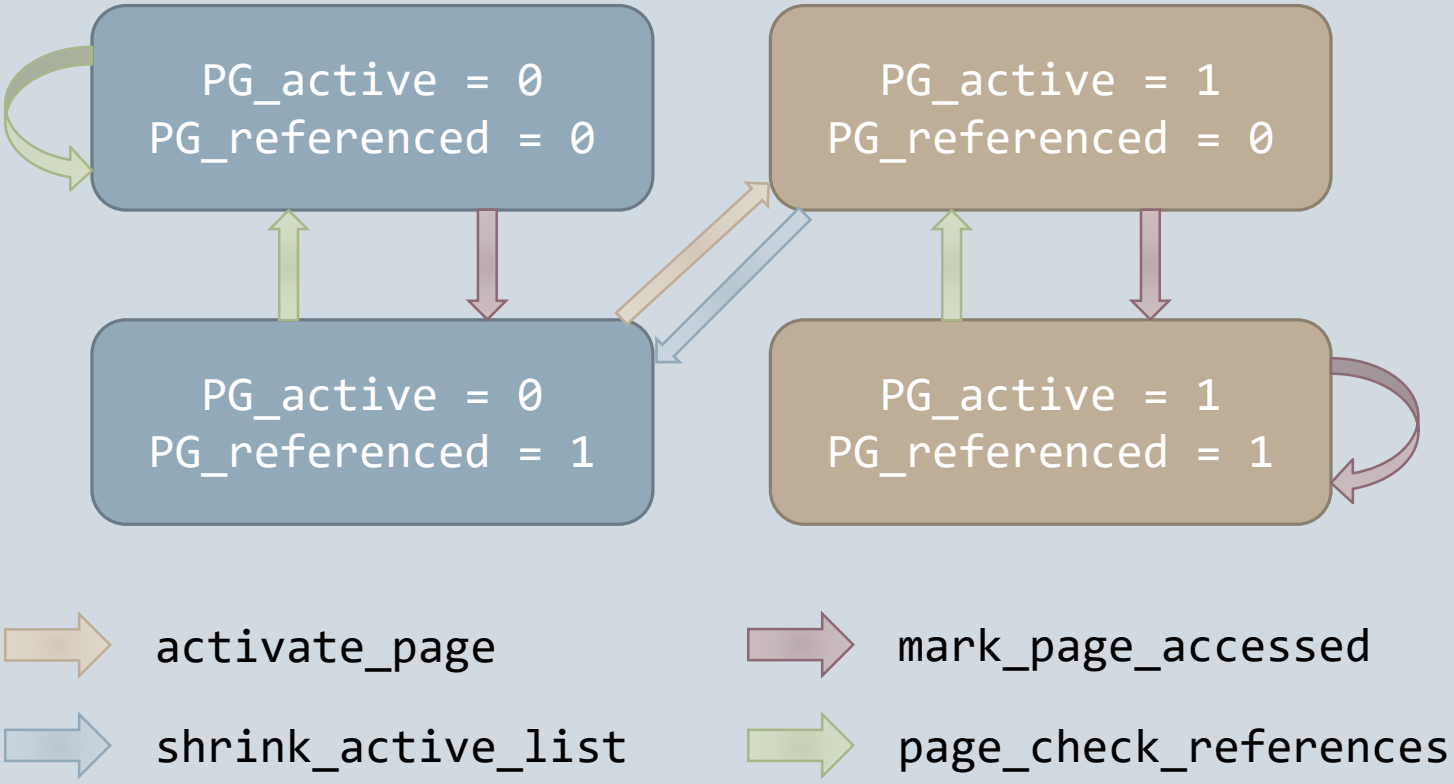
- Study of the Original Algorithm
- Changing to an LFU-Based Algorithm
- Test Results

Study of the Original Algorithm

Basic Ideas

- Reclaim ‘unused’ pages only.
- Associate a counter storing the age of the page with each page in RAM.

State Diagram

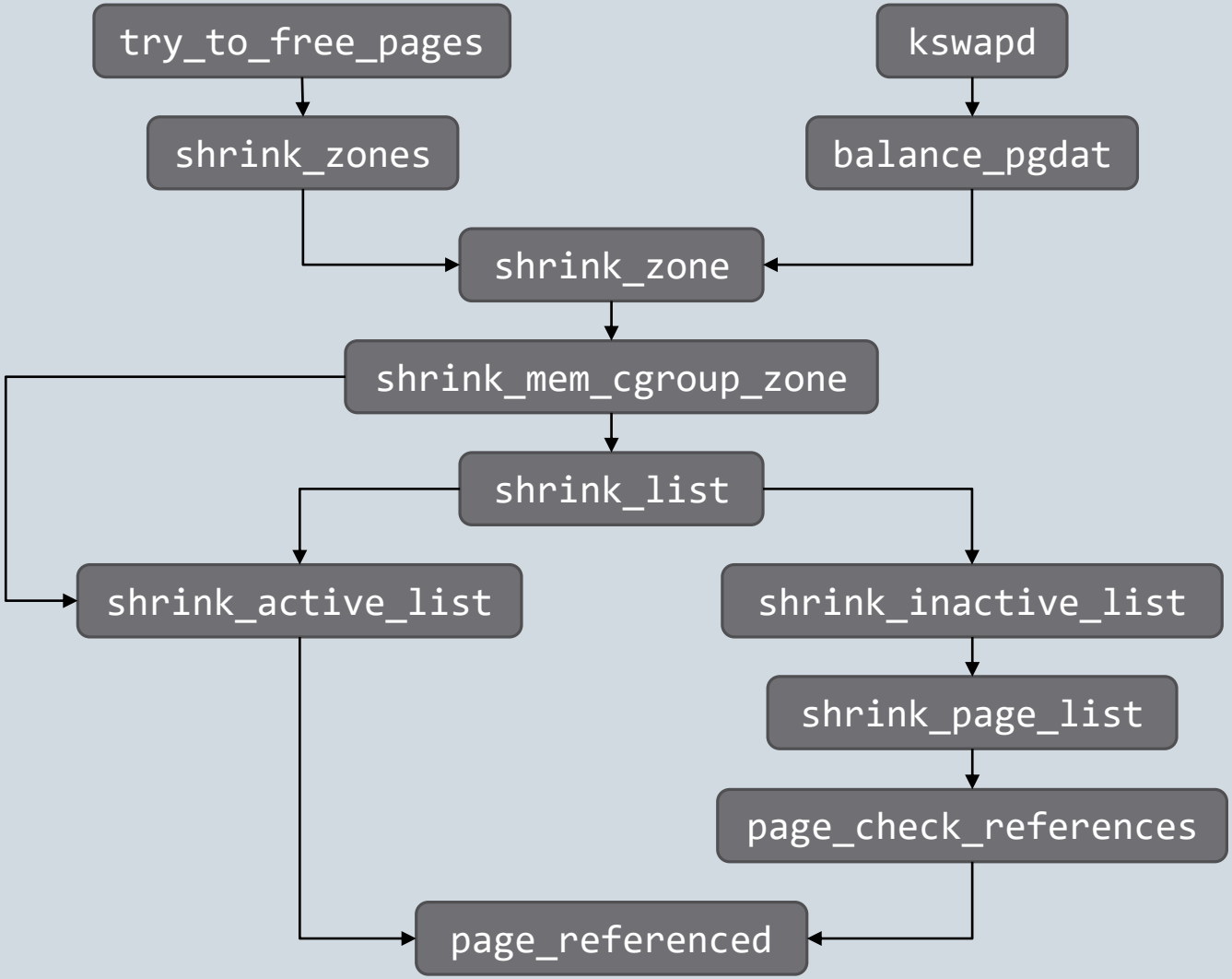


Condition of Reclaiming

- Low on memory: The kernel has difficulty allocating new pages.
- Periodic: Kernel threads are invoked periodically to check whether free pages are above a certain threshold.

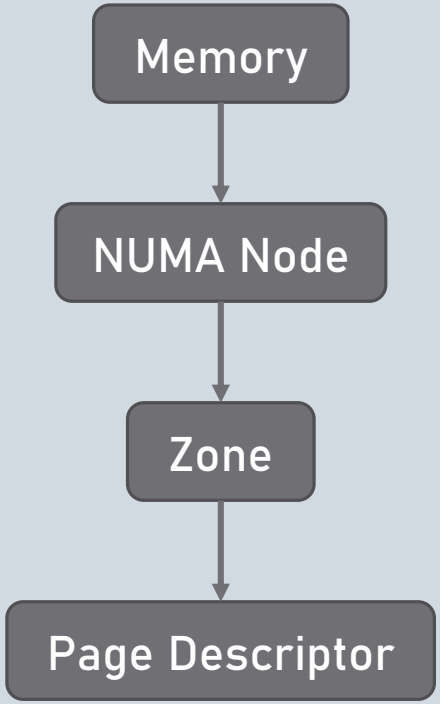
LOW ON MEMORY RECLAIMING

PERIODIC RECLAIMING



Memory-Related Structs

- NUMA: The time required to access different memory locations is different.
- Memory Zone: Accounting for hardware constraints of the type of data stored in pages.



Changing to an LFU-Based Algorithm

Scope of Work

- The algorithm involved just select the candidates to be replaced.
- We just need to change how the status of the pages are transitioned.

Comparison of Two Algorithms

Task	Original	LFU
Refresh page	If PG_referenced is 0, set it. Otherwise clear it and add to active list.	Left shift PG_referenced and add by an offset.
Age page	If PG_referenced is 0, clear it. Otherwise set it and add page to inactive list.	Right shift PG_referenced.

Condition	Original	LFU
Add to active list	PG_referenced is 1 and the page is referenced recently.	PG_referenced is above some threshold.
Add to inactive list	PG_referenced is 0 and the page is not referenced recently.	PG_referenced is below some threshold.

Parameters

- The offset to add to `PG_referenced` when page is refreshed: 1
- The threshold of moving pages between lists: to be tested
- The maximum limit of `PG_referenced`: necessary, also to be tested

Test Results

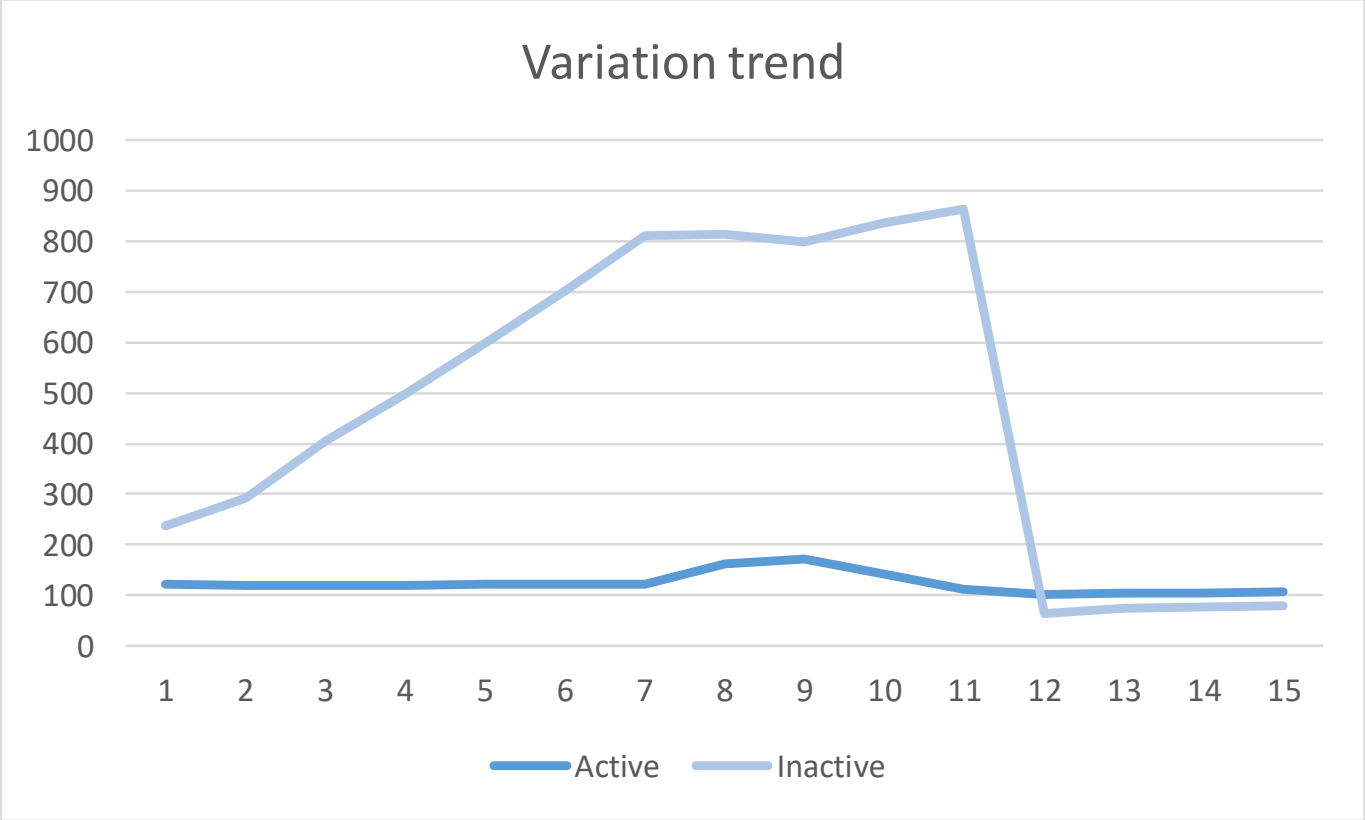
Test the Algorithm

- Write a program to occupy as much memory as possible.
- Access `/proc/meminfo` periodically to get the sizes of active and inactive lists.

Notice

- `kswapd` invokes `lowmemkiller`, which kills processes to reclaim memory space.
- The test program can no longer run while the pages are being reclaimed.

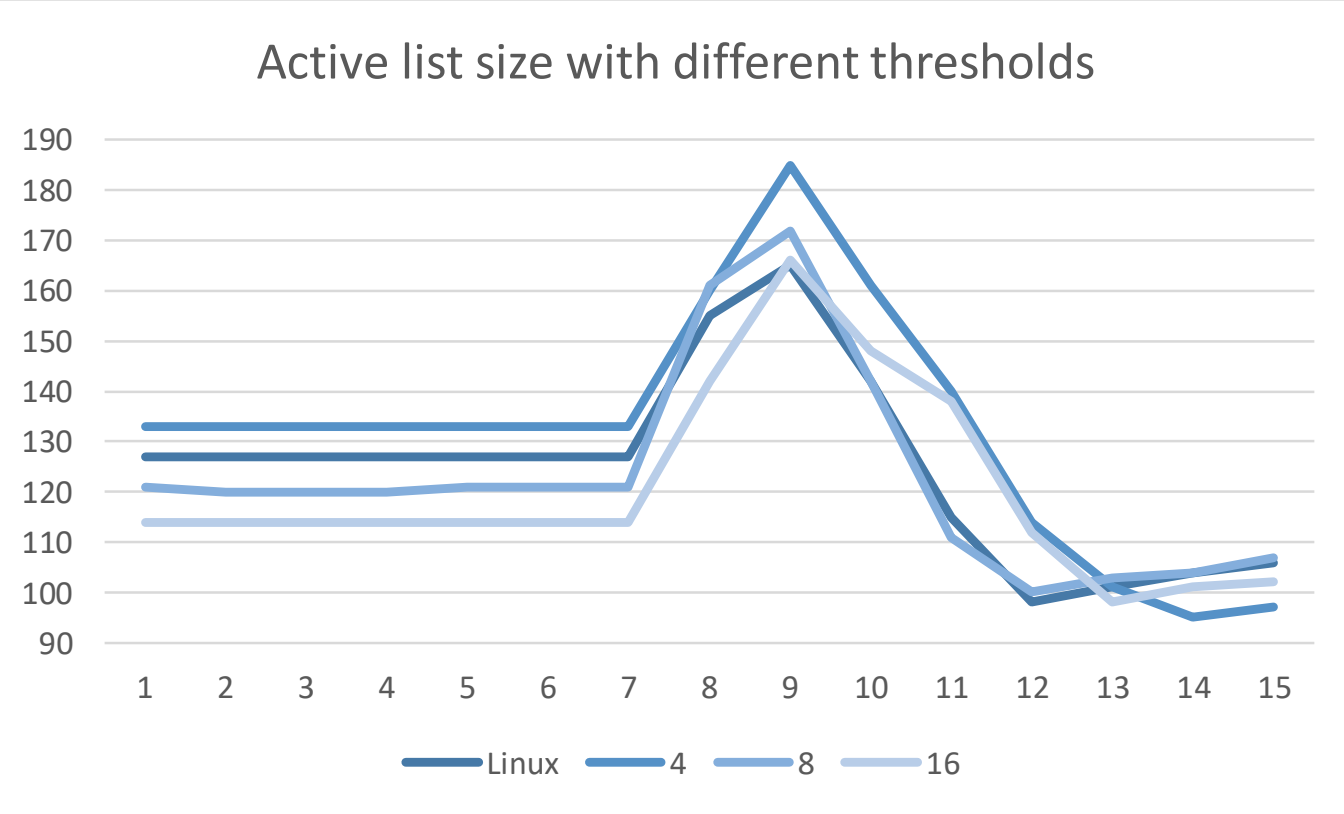
Variation Trend



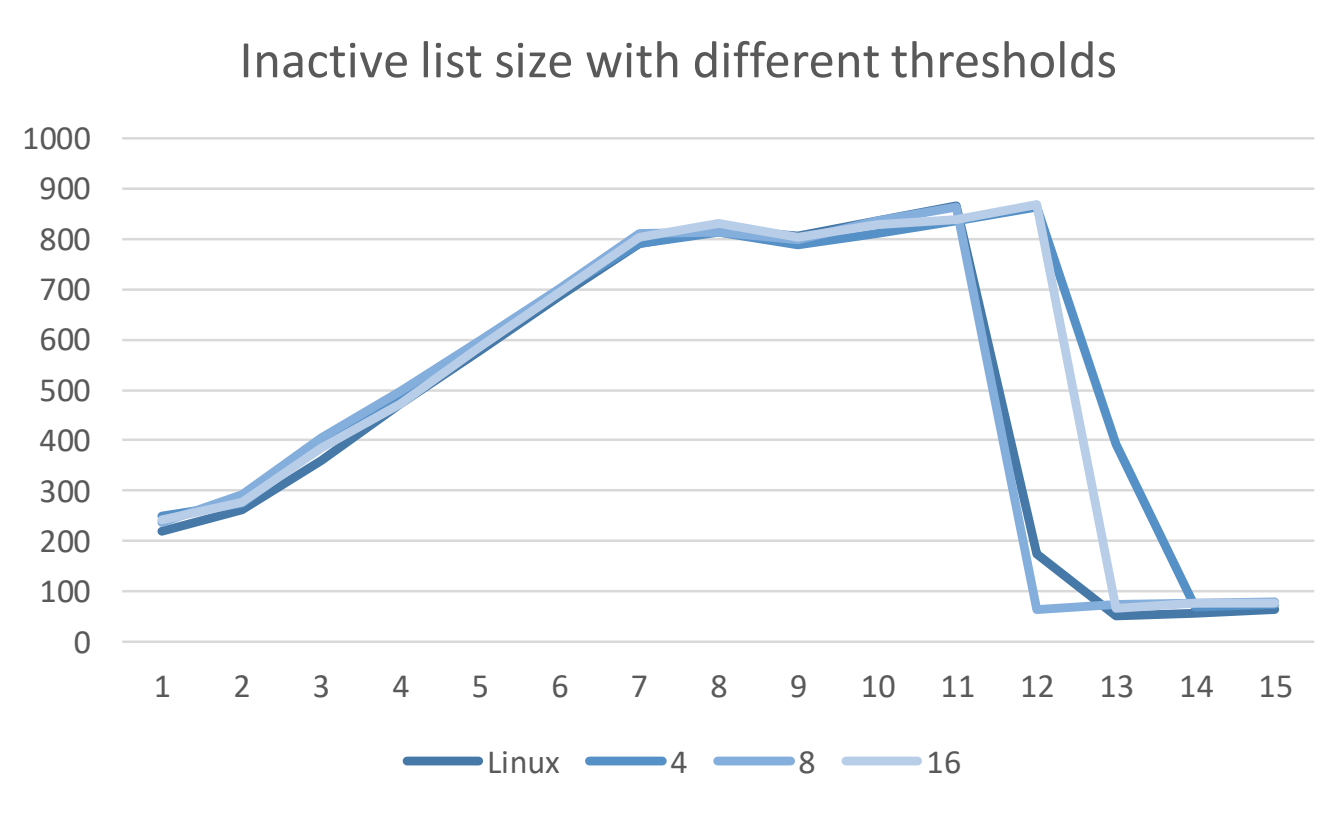
Variation Trend

Time	Trend	Explanation
1-6s	Inactive list continues to grow. Active list remains.	Test program access every page only once.
7-9s	Inactive list grows more slowly. Active list begins to grow.	Page replacement procedures are started.
9-11s	Inactive list grows, while active list shrinks.	Some pages are aged in the replacement algorithm.
11-12s	Inactive list drops dramatically.	Some processes, including the test program, are killed.

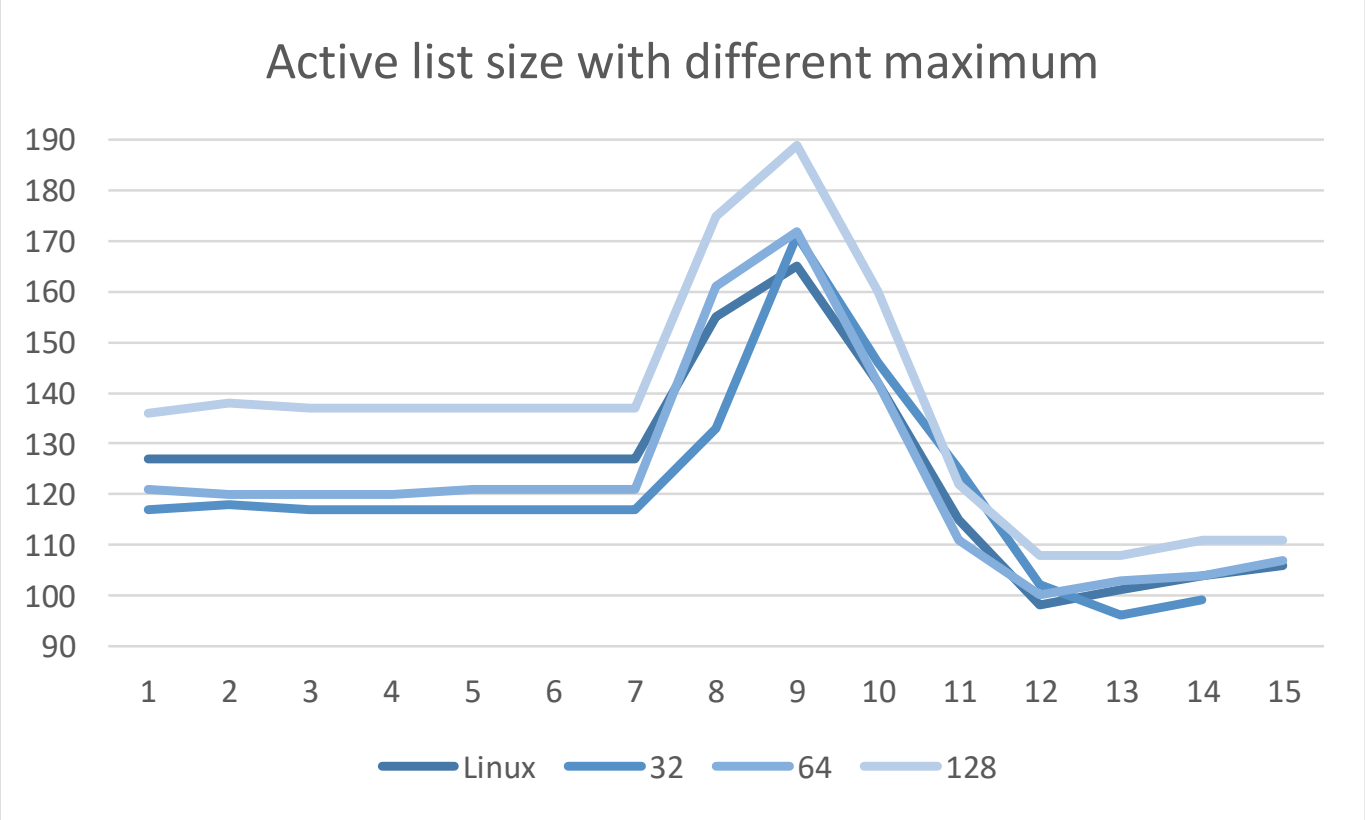
The Impact of Threshold



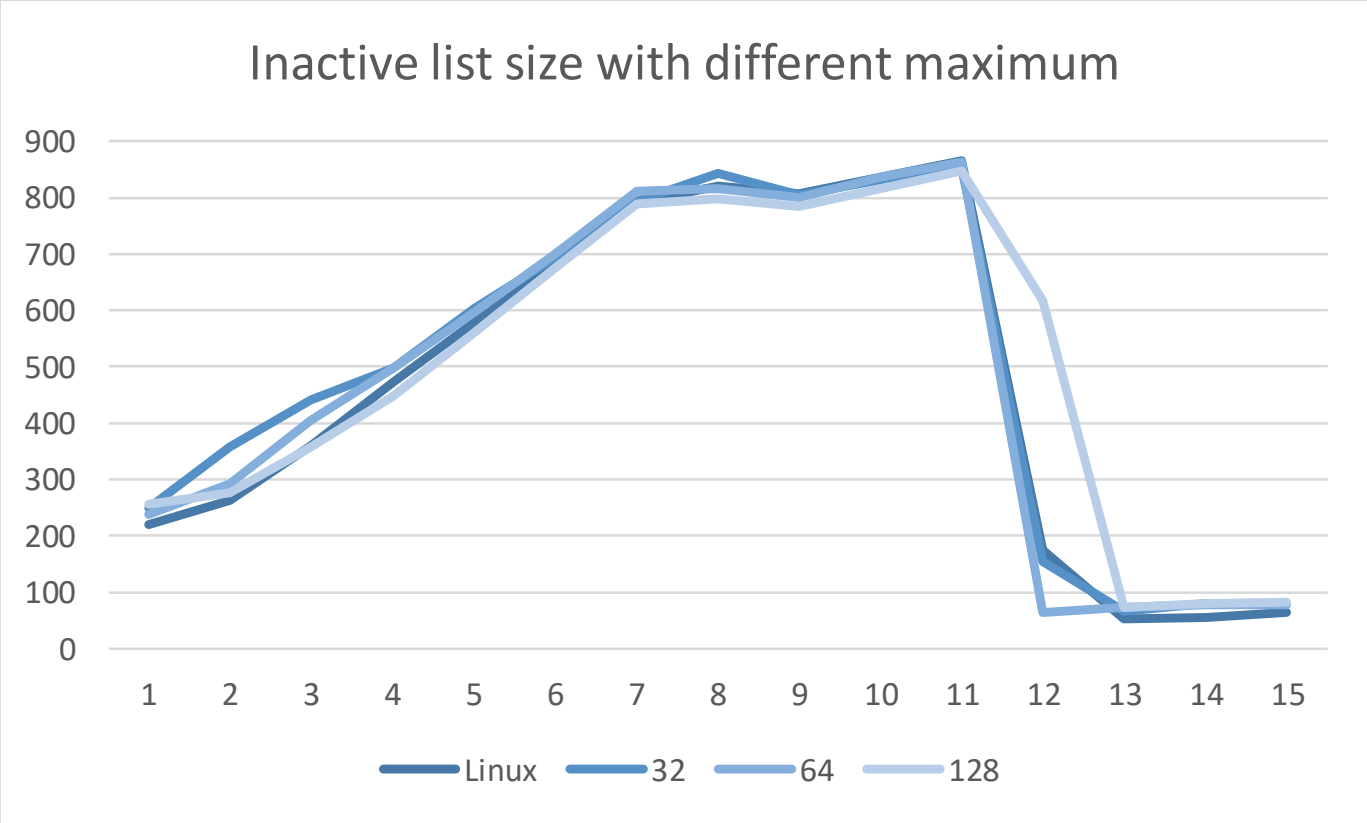
The Impact of Threshold



The Impact of Maximum



The Impact of Maximum



Observation

- The smaller the threshold, the larger the active list.
- The higher the maximum, the larger the active list.
- The size of active list varies more greatly in the new algorithm, compared with the original.
- The parameters seem to have little impact on the size of inactive list.

Choice of Parameters

Criterion	Threshold	Maximum
Similar to the original	8	64
Smaller active list size	16 or higher	Threshold × 2

Reference

- Understanding the Linux Kernel. Daniel Bovet and Marco Cesati. O'Reilly Media.

Thanks