Tarea Doughh

Sistemas Operativos y Redes

Universidad de los Andes

Profesor Claudio Álvarez
Ayudante Cristóbal Griffero
201720

Margarita Estévez
José Tomás González

13/10/2017

Introducción

En el presente informe se explica el funcionamiento de los algoritmos de reemplazamiento (*swapping*) de páginas de memoria en marcos. Las páginas de memoria son fracciones de data que se almacena en la memoria física del computador. Para poder ejecutar, leer y/o escribir (ERW) las páginas es necesario acceder a ellas y traerlas a memoria utilizable en ejecución. Para esto se emplean los marcos, una estructura que almacena las páginas y que puede ser accedida para ERW. Los marcos suelen ser de menor capacidad que la cantidad de páginas, por lo que es necesario saber qué páginas están en los marcos y en cual. También es necesario el *swapping* de páginas dentro de los marcos para poder acceder a las páginas necesarias.

Para el *swapping* se implementan los algoritmos que eligen que pagina sacar del marco. Los algoritmos implementados son *First in first out* (FIFO), *Random* (LRU) y uno diseñado por los integrantes (Custom)

El objetivo de la tarea es observar el cambio en la eficiencia según la cantidad de marcos en el cálculo. La experiencia es ejecutar los algoritmos de reemplazamiento en tres funciones SORT, SCAN y FOCUS con cien páginas y variando los marcos desde dos hasta cien.

Desarrollo

El objetivo de esta tarea consiste en implementar paginación bajo demanda con con distintos algoritmos de reemplazo de páginas. Se pide entonces, implementar dos algoritmos conocidos y diseñar uno propio para luego simular su funcionamiento y analizar su rendimiento. A continuación se explica la implementación de cada uno de los algoritmos.

FIFO

El algoritmo reemplaza la página que lleva más tiempo en los marcos. Para medir lo anterior, se define una variable global que es accedida en busca del índice de la última página ingresada. Para comenzar se define la variable como menos uno (-1) para expresar que no hay una última página guardada. De lo anterior, se obtiene que sumandole uno al índice, el marco en donde voy a guardar la primera página es cero. Siguiendo con las primeras lecturas que no necesitan *swapping* el índice llega a ser la cantidad de marcos. Con ésta información, se sabe que la página que lleva más tiempo es la del índice igual a cero. Con la operación módulo de la cantidad de marcos, el índice vuelve a cero, el marco de la página más antigua y así sucesivamente.

Random

El algoritmo calcula un marco aleatorio para el reemplazo. Nuevamente se inicia la variable del último índice en menos uno (-1). Luego se calcula un número aleatorio con la función *rand().* Como esta función depende del tiempo, esta genera el mismo número si el reloj es el mismo. Para cambiar la restricción anterior, se le suma el índice del último frame utilizado. Se calcula el módulo de la cantidad de marcos y se obtiene el nuevo índice para el *swapping*.

Custom

El algoritmo diseñado es una variación del algoritmo FIFO en el que se repite la misma posición después un *swapp*. Se toma el índice iniciado en menos uno (-1), se le suma uno, y se calcula el módulo de la cantidad de marcos. En la siguiente iteración no se suma 1, por lo que se reemplaza el que acaba de guardarse.

Con estos algoritmos se calcula el número del marco en el que se guardará la página solicitada por el sistema operativo.

Resultados

Para la obtención de los resultados, cada combinación de algoritmos con los tres programas entregados fue ejecutada en un ciclo 99 veces variando la cantidad de marcos disponibles entre 2 y 100; la cantidad de páginas se mantuvo constante en 100 para todas las ejecuciones. En cada ejecución, se obtuvieron métricas como *cantidad de faltas de página, cantidad de lecturas a disco y cantidad de escrituras a disco.* A continuación se presenta el análisis segmentado según el problema a resolver utilizando los tres algoritmos en 99 iteraciones *(Gráficos adjuntos al final del documento).*

SORT

	FIFO	CUSTOM	LRU
Faltas de Página	986	1031	1008
Lecturas a Disco	595	615	604
Escrituras a Disco	389	385	383

Los resultados obtenidos en la ejecución de las iteraciones para SORT que el algoritmo de reemplazamiento FIFO es el más eficiente en cuanto faltas de página y lecturas a disco. Por otro lado LRU mostró ser más eficiente en cuanto escrituras a disco. Los gráficos correspondientes muestran curvas bastante similares, exceptuando FIFO que muestra un comportamiento constante en varias secciones, pero manteniendo comportamiento similar. El algoritmo CUSTOM fue el con peor desempeño en faltas de página y en lecturas a disco, lo que puede mostrar lo mucho que afecta el pequeño cambio realizado al algoritmo FIFO.

SCAN

	FIFO	CUSTOM	LRU
Faltas de Página	1190	896	918
Lecturas a Disco	1090	796	818
Escrituras a Disco	99	97	99

Analizando el gráfico podemos observar que utilizando el algoritmo FIFO las métricas permanecen constantes hasta igualar la cantidad de frames con la de páginas por lo que existe una página por marco y las faltas se reducen a cero. Utilizando el algoritmo CUSTOM se puede ver una mejoría en la cantidad de faltas de página, lecturas a disco, cuando se tienen sobre 50 marcos. Interesante ver que la cantidad de escrituras a disco permanece constante reduciéndose sólo a partir de 92 marcos en adelante. Finalmente, utilizando el algoritmo LRU se produce una caída exponencial de la cantidad de errores

y lecturas a disco. Nuevamente la cantidad de escrituras a disco se mantiene constante durante el incremento de marcos. En promedio, se ve claramente que el algoritmo más eficiente fue CUSTOM, seguido por LRU y finalmente FIFO.

FOCUS

	FIFO	CUSTOM	LRU
Faltas de Página	382	382	389
Lecturas a Disco	233	224	230
Escrituras a Disco	147	132	141

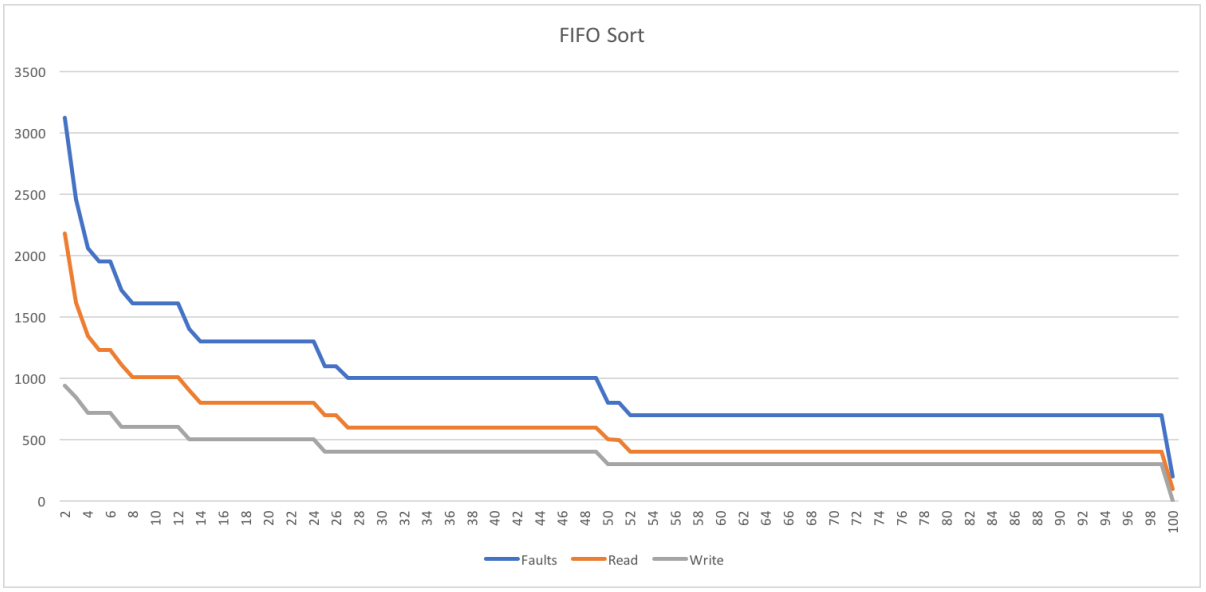
Los resultados obtenidos utilizando los tres algoritmos sobre el problema FOCUS presentan estadísticas bastante similares. Se presenta una tendencia a la baja en la cantidad de faltas y llamadas a medida que aumenta la cantidad de marcos. En cuanto a rendimientos individuales, se puede apreciar que el algoritmo FIFO y CUSTOM presentan la misma cantidad de faltas de página, sin embargo CUSTOM realiza menos llamadas a disco tanto de lectura como escritura. LRU pese a tener más faltas de página que FIFO, presenta un mejor rendimiento en cuanto a lecturas y escrituras a disco, sin embargo no mejora las estadísticas del algoritmo CUSTOM.

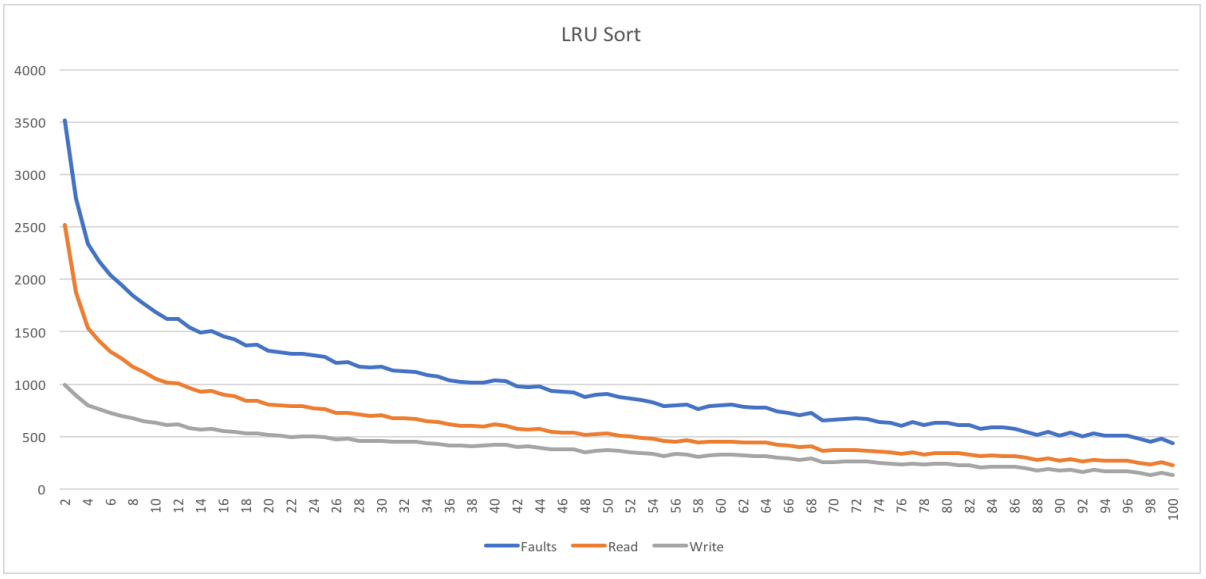
Es importante mencionar que por problemas de implementación, el algoritmo LRU produjo diferencias en el resultado del programa FOCUS por lo que los resultados obtenidos son de dudosa calidad para éste. Sin embargo, FIFO y CUSTOM no presentaron este problema por lo que la comparación mutua sigue siendo completamente válida.

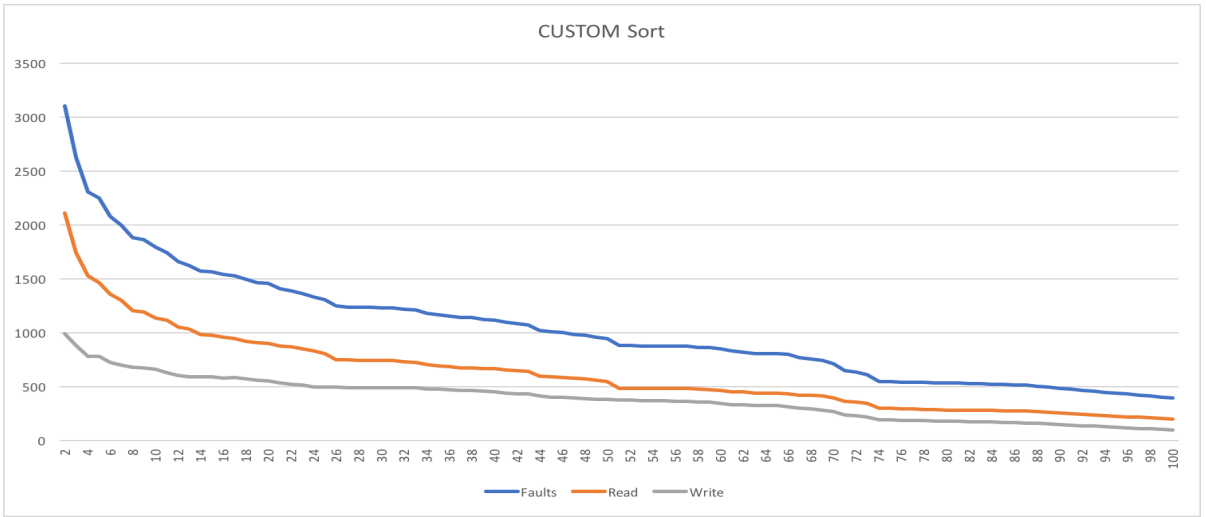
Conclusión

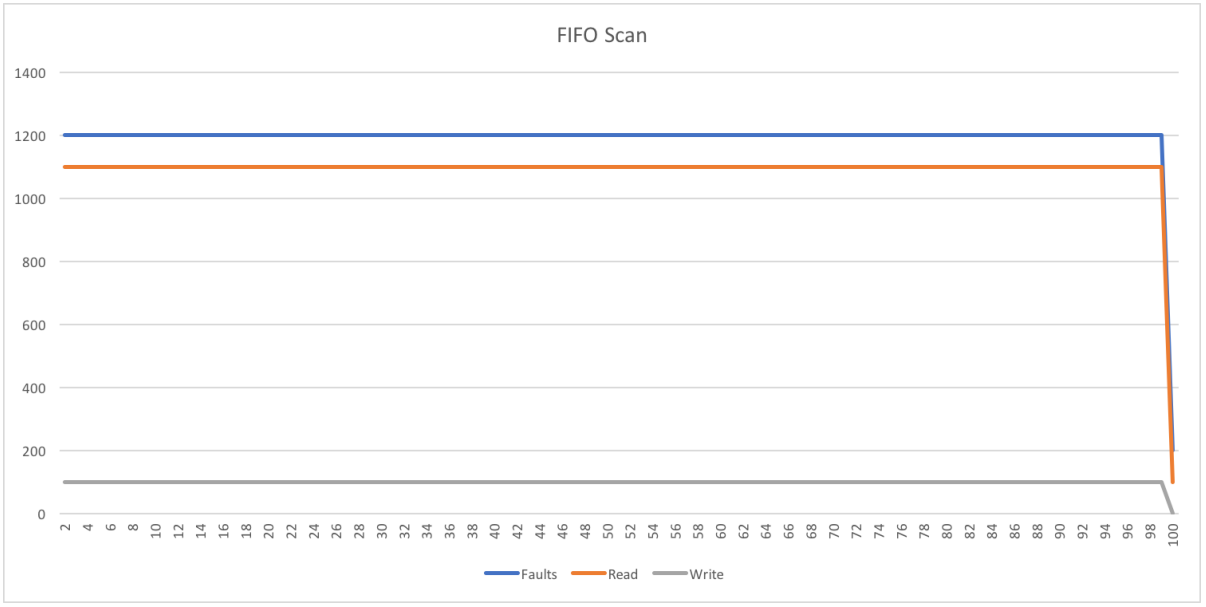
A partir de los resultados obtenidos, se puede considerar que el comportamiento de los algoritmos depende de la necesidad del sistema. Durante la ejecución de Sort utilizando el algoritmo FIFO se obtiene un comportamiento superior en cantidad de faltas, pero no así en SCAN y FOCUS. El algoritmo CUSTOM muestra una gran mejoría en SCAN y FOCUS con respecto a SORT, por lo que podemos considerar mejor el algoritmo CUSTOM que FIFO y LRU al superar a ambos en dos de las tres experiencias.

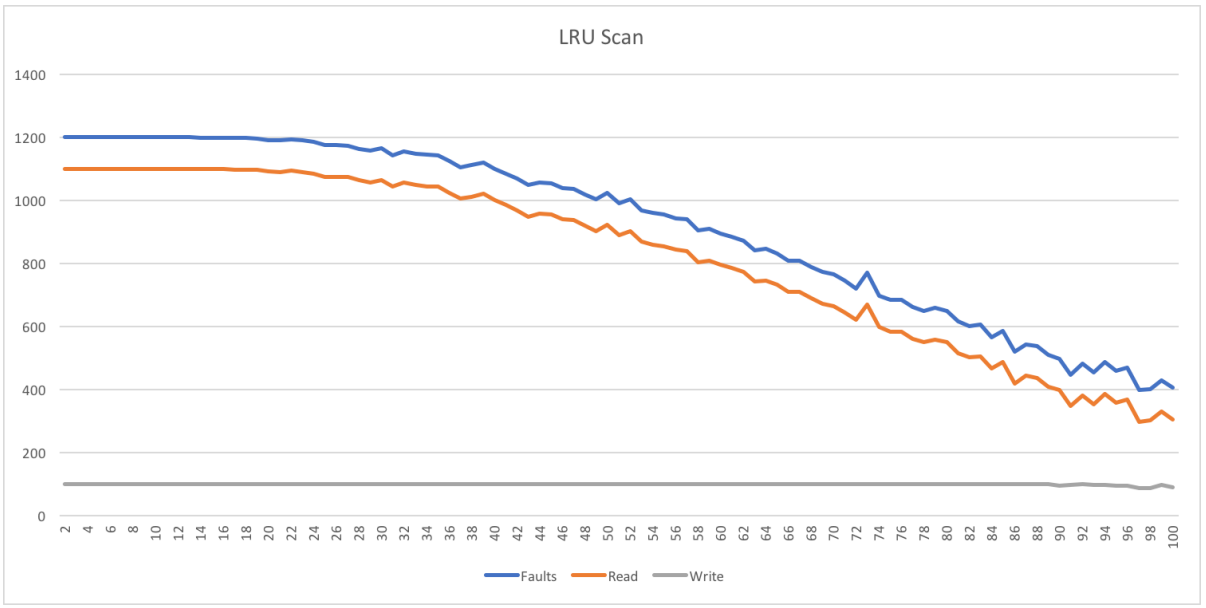
Gráficos obtenidos

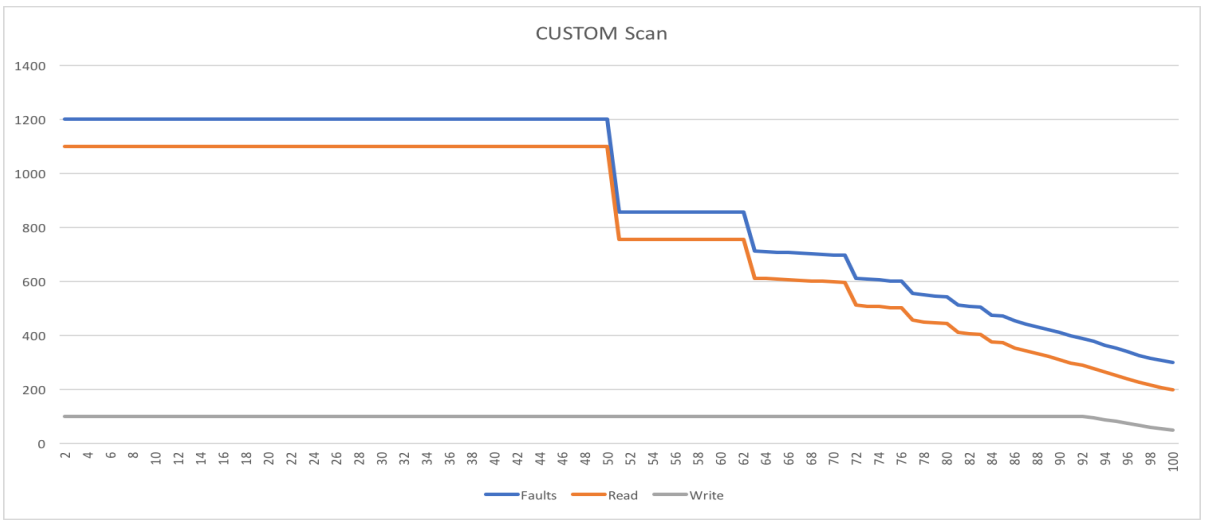


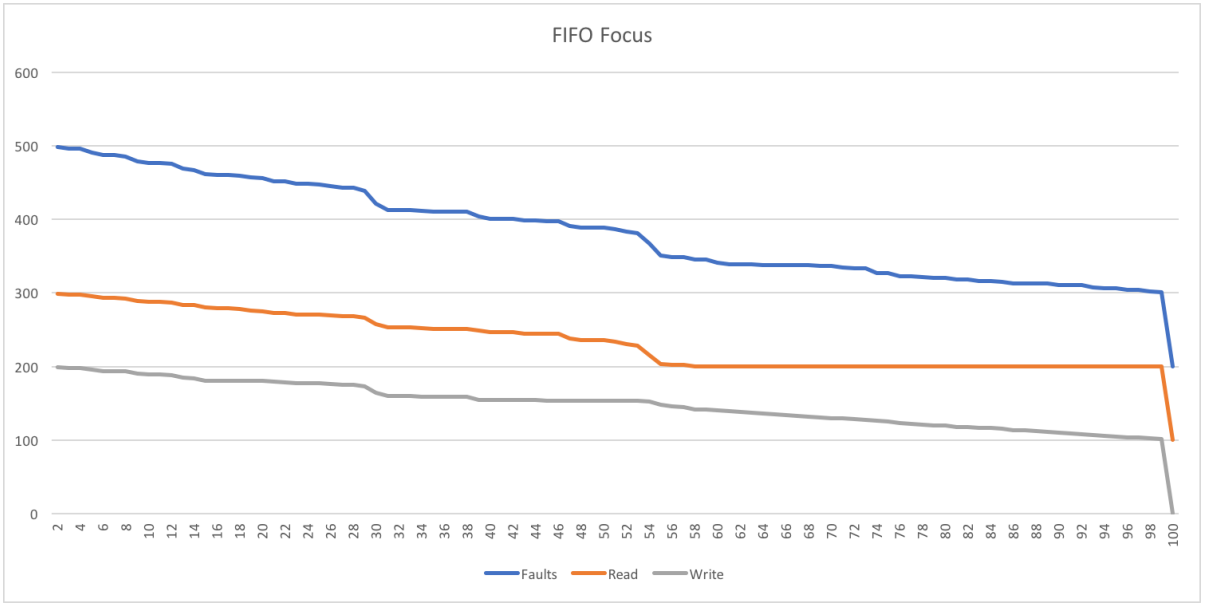


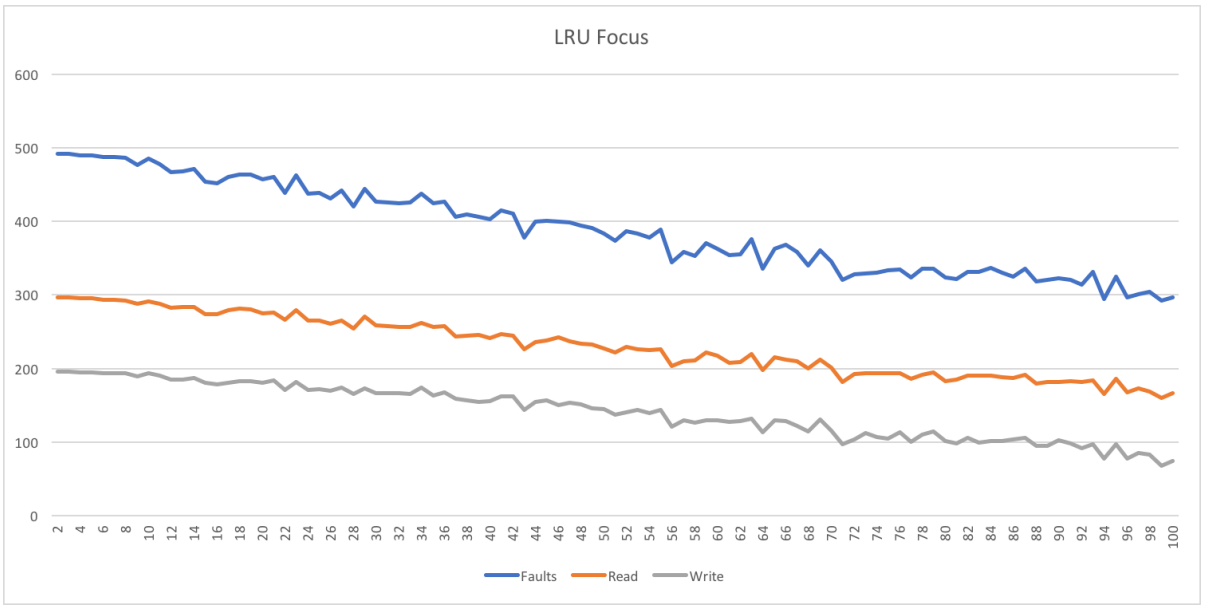


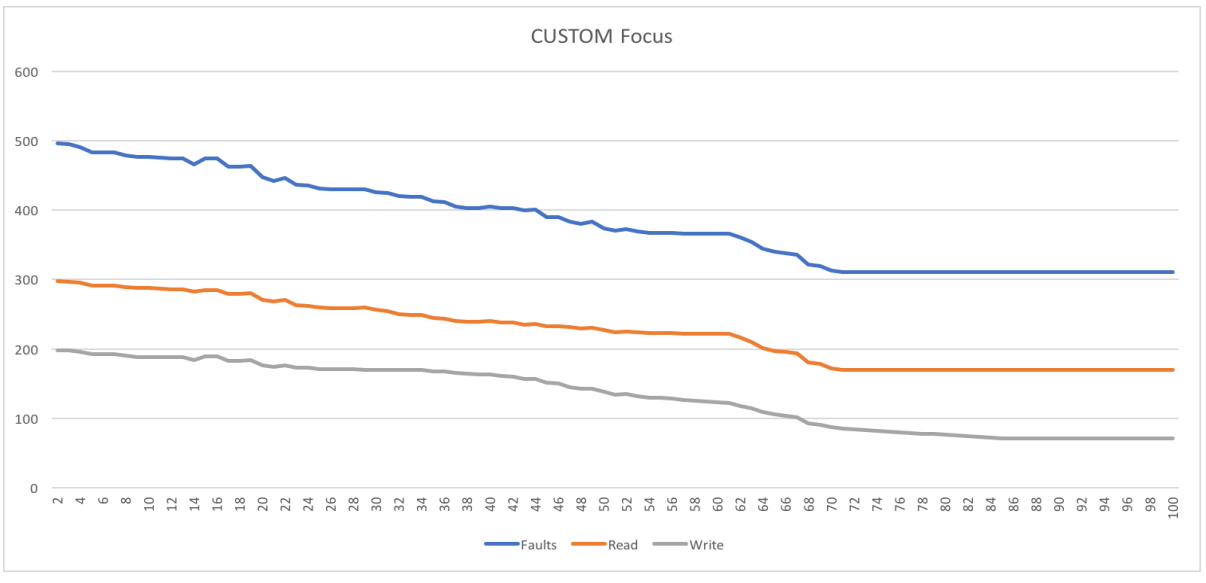

FIFO Scan
1400
1200
1000
800
600
400
200
0
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66 68 70 72 74 76 78 80 82 84 86 88 90 92 94 96 98 100
Faults
Read
Write


LRU Scan
1400
1200
1000
800
600
400
200
0
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66 68 70 72 74 76 78 80 82 84 86 88 90 92 94 96 98 100
Faults
Read
Write


CUSTOM Scan
1400
1200
1000
800
600
400
200
0
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66 68 70 72 74 76 78 80 82 84 86 88 90 92 94 96 98 100
Faults
Read
Write


FIFO Focus
600
500
400
300
200
100
0
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66 68 70 72 74 76 78 80 82 84 86 88 90 92 94 96 98 100
Faults
Read
Write


LRU Focus
600
500
400
300
200
100
0
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66 68 70 72 74 76 78 80 82 84 86 88 90 92 94 96 98 100
Faults
Read
Write


CUSTOM Focus
600
500
400
300
200
100
0
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66 68 70 72 74 76 78 80 82 84 86 88 90 92 94 96 98 100
Faults
Read
Write