Projeto
Algoritmo Two Handed Clock

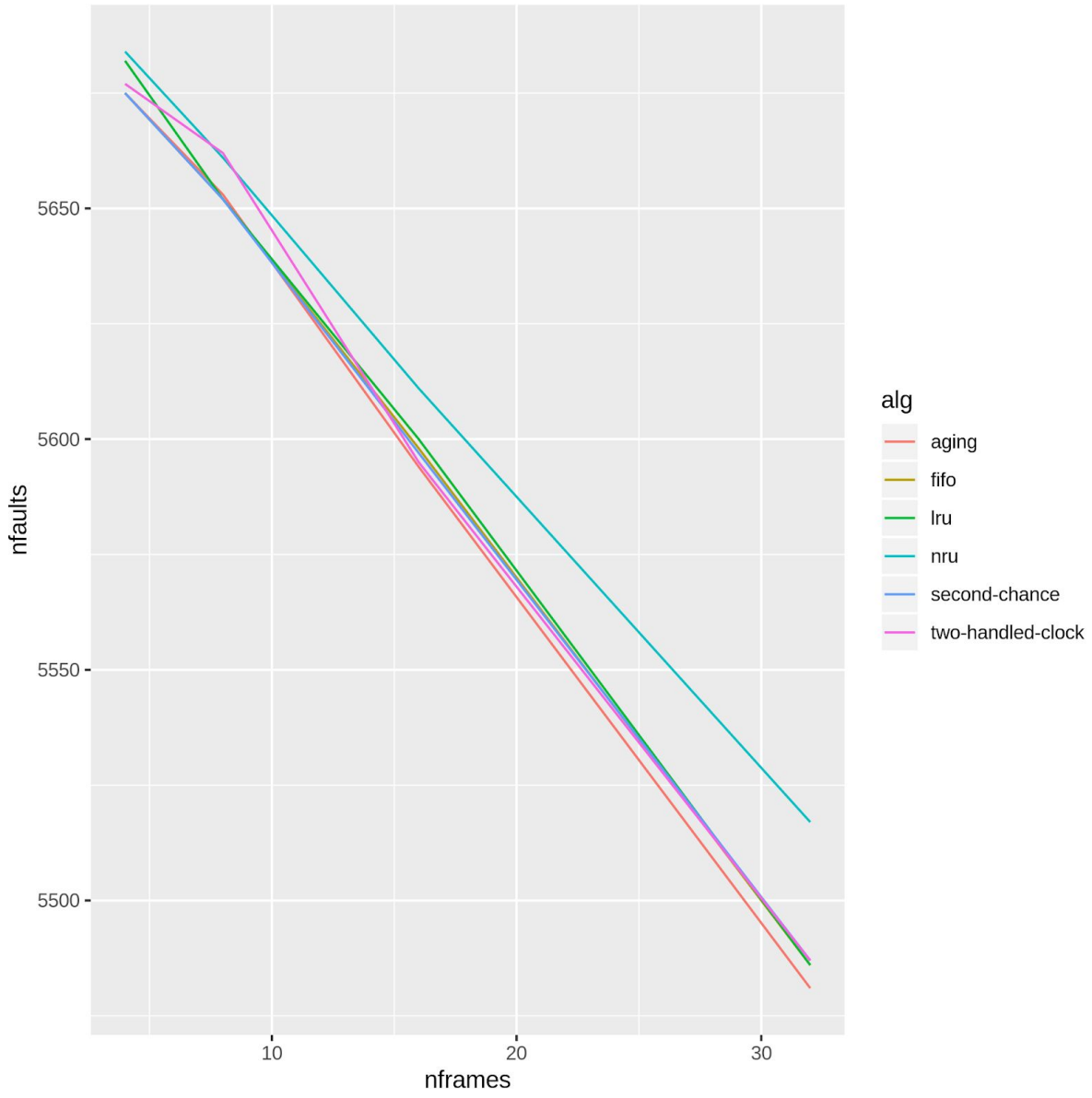
(Avaliando reposição de páginas)

Grupo: {Bruno, Gustavo, Hiago}

Neste relatório vamos discutir os resultados da análise feita sobre a política de reposição de páginas Two Handed Clock, que idealmente, funciona da seguinte forma: ao invés de ter apenas um ponteiro que aponta para a página que será substituída, como no algoritmo tradicional "Clock" (implementação otimizada da FIFO), neste algoritmo, um segundo ponteiro é adicionado, cujo objetivo é procurar páginas com bit de referência/uso que foram "limpos". Durante a execução, o segundo ponteiro procura as páginas que o primeiro ponteiro "limpou", alocando-as no final da fila, (dando uma "segunda chance").

Uma vantagem em relação a segunda chance, é que, uma página de “curto prazo”, por exemplo, uma página que foi referenciada apenas uma vez desde que entrou na lista, leva até dois ciclos na fila para ser eliminada, por outro lado, no Two Handed Clock em apenas um ciclo a eliminação é realizada devido ao segundo ponteiro.

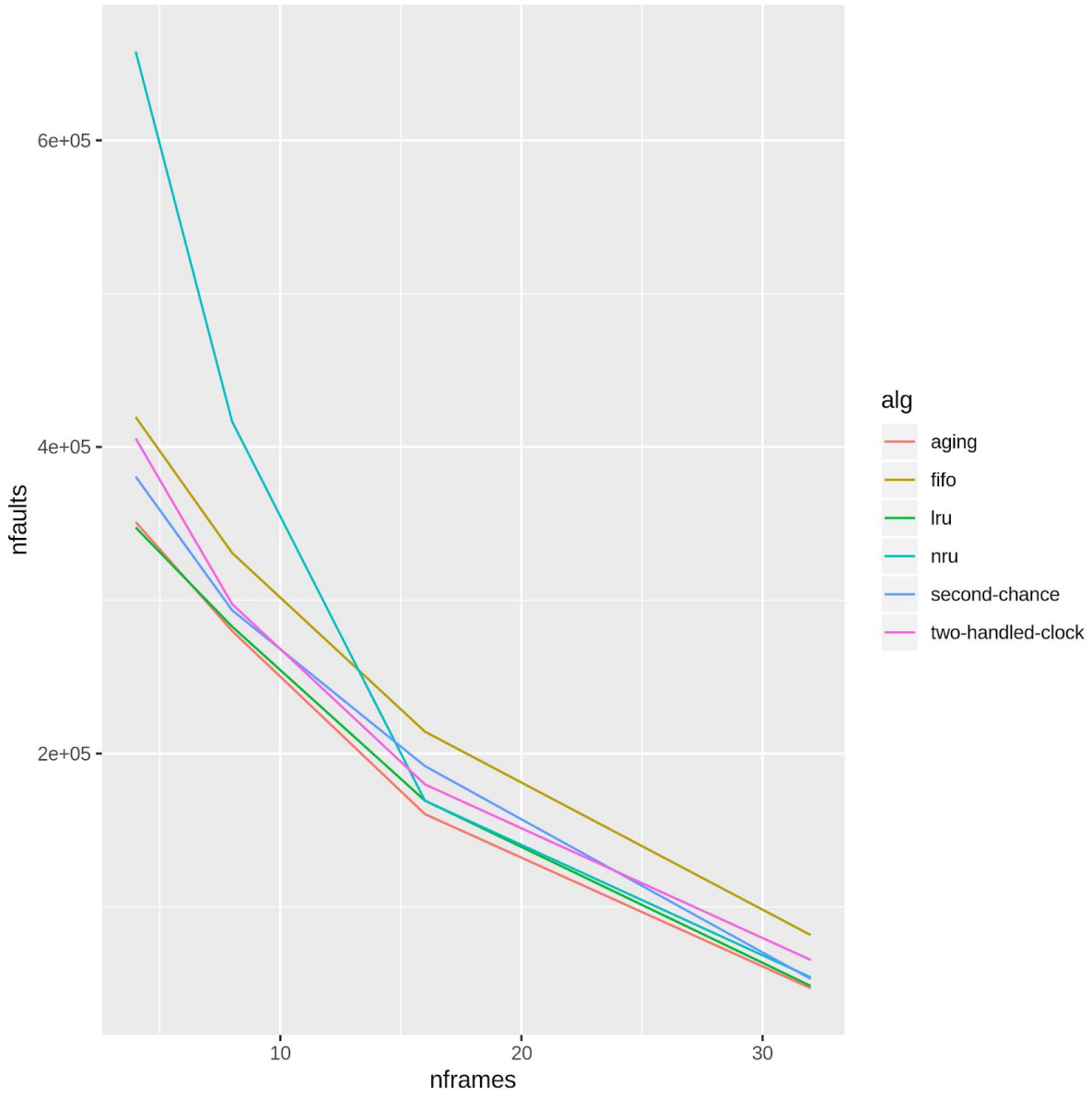
Realizamos testes a partir de três datasets diferentes para termos uma melhor compreensão do comportamento/performance do algoritmo comparado a outros abordados no laboratório 4 da disciplina, o primeiro dataset consiste em uma amostragem do memory trace de um servidor da Azure, que foi disponibilizado publicamente pela microsoft, o segundo dataset consiste em dados de acesso a páginas aleatórios que também foram usados no laboratório 4 e o terceiro dataset foi produzido para simular acessos de páginas de dois processos diferentes, que concorrem por memória. Abaixo, alguns plots demonstram a quantidade de page faults em relação a quantidade de frames:



Plot feito a partir de teste realizado com o dataset real capturado de servidores da Microsoft.

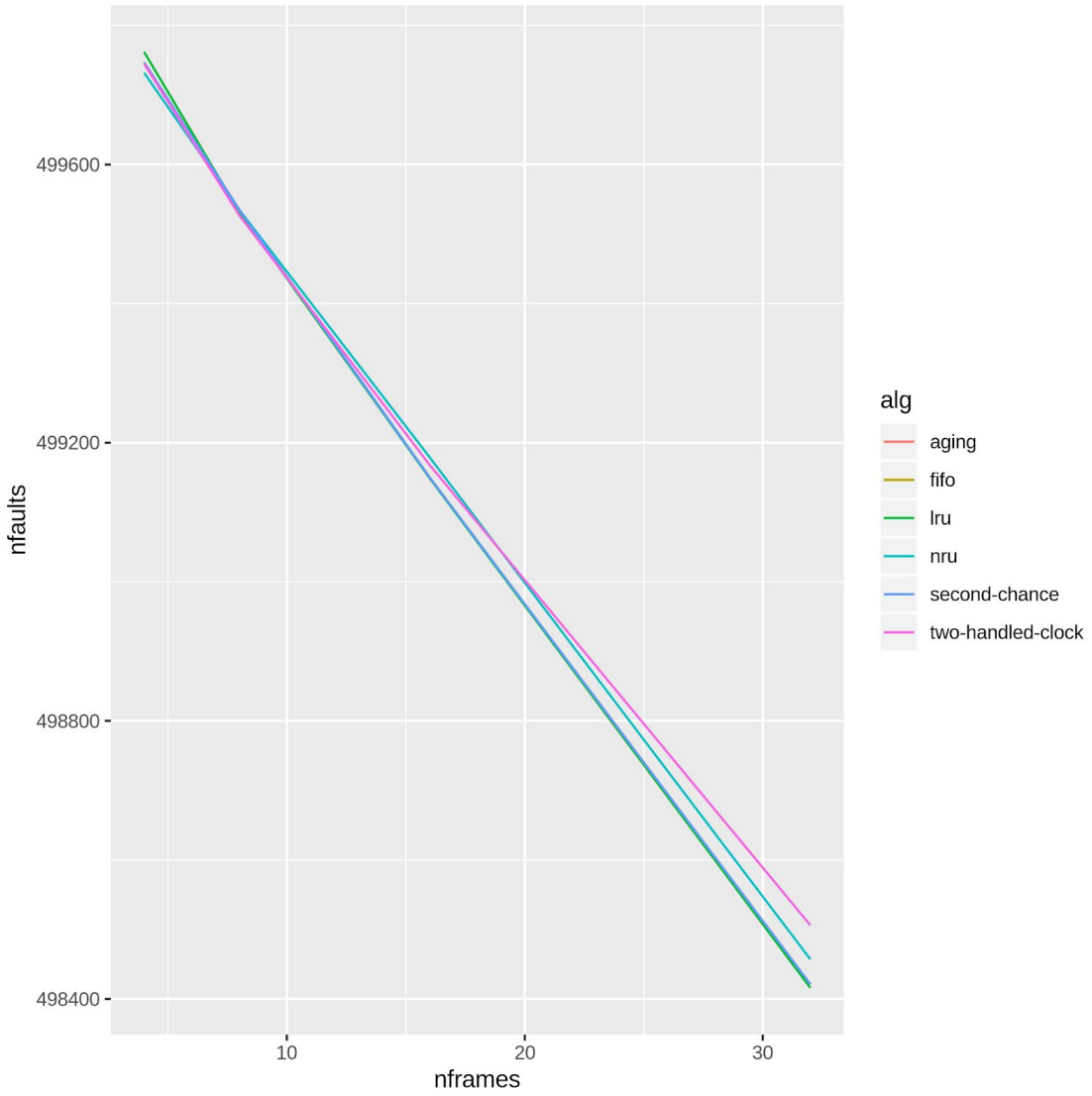
Nesse dataset temos a alta presença de operações de leitura (read), o algoritmo NRU mostrou-se afetado devido a grande quantidade dessas operações, fazendo com que páginas recentemente usadas que estão na classe mais baixa (lido e não escrito) sejam descartadas, os outros algoritmos mantiveram sua performance esperada.

No início, com poucos frames, os algoritmos ainda mantiveram o esperado, no entanto, em algum momento o Two Handed Clock apresentou-se com a pior performance, ao chegar perto da quantidade de 10 frames, porém quando a quantidade de frames aumenta sua performance entra dentro do esperado que é estar a frente de maioria deles.



Plot feito com o dataset padrão do exercício de laboratório 4 (trace 2), de acordo com as linhas as performances dos algoritmos estão dentro do padrão esperado, sendo o Aging o melhor de todos, seguido de LRU, NRU, Two Handed Clock, Second Chance e FIFO, e o Two Handed Clock variando entre Second Chance e NRU, esperado, já que o Two Handed Clock é uma aproximação do NRU.

Para um número de frames pequeno, é observado que o comportamento não é esperado, no plot é visível que o algoritmo NRU apresenta um número alto de page faults para poucos frames, o que não é consensual, afinal ele é melhor que os algoritmos FIFO, Second Chance e Two Handed Clock, porém, ao ocorrer um aumento gradual no número de frames, os algoritmos apresentam o comportamento esperado. O que pode ser explicado pela ordem de requisição das páginas, ou mesmo pelos ticks de clock, que influencia totalmente no algoritmo.



Plot produzido a partir do experimento de usar os algoritmos de reposição de página com dois processos distintos que disputam por memória.

Neste plot podemos ver um comportamento atípico do Two Handed Clock, no qual o mesmo vai perdendo performance com o acréscimo de frames, o que nos leva a crer que o problema está no chaveamento de processos. Imaginemos o seguinte, o processo 1 é alocado e ocupa completamente a lista de frames em memória quando ocorre a troca para o processo 2 ele tenta alocar páginas e assim o algoritmo de reposição executa e substituirá as páginas do processo 1, lembrando que o segundo ponteiro modificará o bit de referência o deixando 0 de tempos em tempos, dado que o processo 1 não está mais executando, seus bits de referência nunca serão atualizados para 1 e então serão os mais propensos a serem descartados, logo em algum momento é possível que as páginas do processo 1 em memória sejam poucas ou até nenhuma, portanto quando ocorrer outro chaveamento para o processo 1, ele terá de alocar suas páginas na memória novamente, o que ocasionará em uma ocorrência inesperada de page faults.

Conclusões:

Ao analisar cada plot, percebemos que em várias situações o Two Handed Clock se aproxima do algoritmo Second Chance, (sendo na literatura uma implementação otimizada deste).

Em casos em que ocorre a disputa de memória entre diferentes processos, a algoritmo apresenta certa sensibilidade ao número de frames apresentando muitos page faults devido a um processo estar eliminando páginas muito usadas do outro.

Também é possível afirmar que o Two Handed Clock possui um resultado parecido com o NRU, sendo uma implementação otimizada do Second Chance (que é em performance um pouco menor que o NRU), sendo o O Two Handed Clock uma aproximação menos custosa do NRU.

Quando a quantidade de leituras e escritas são bem distribuídas, não há diferença considerável na performance de algoritmos que precisam desses parâmetros de maneira que o comportamento das políticas de reposição são bem similares ao que vemos na teoria.

Os plots ilustram que o Two Handed Clock, Second Chance e NRU são equiparáveis alternando-se as vezes devido a variação da quantidade de frames, que pode ser algo que altera muito a performance de algoritmos que usam a estrutura de dados Fila. Mesmo com o dataset de workloads reais da Azure e o outro com dois processos disputando por memória os resultados continuam similares.

Referências:

https://cs.stackexchange.com/questions/29092/whats-the-difference-between-clock-and-second-chance-page-replacement-algorithm [1]

http://gwa.ewi.tudelft.nl/datasets/gwa-t-12-bitbrains [2]

http://www.pb.utfpr.edu.br/ribas/so/2012-2/slides-p2/substituicao-de-paginas.pdf [3]