ZRAM: USO DE COMPRESSÃO PARA AUMENTAR O DESEMPENHO DE SISTEMAS DE MEMÓRIA VIRTUAL

Felipe Duarte Silva - GRR20231957
Guilherme Eduardo Gonçalves da Silva - GRR20231950
Gustavo Benitez Frehse - GRR20235087
Universidade Federal do Paraná – UFPR

***Resumo*—Este estudo avalia o impacto da utilização do ZRAM em sistemas de memória virtual, focando na compressão da RAM para otimizar o desempenho em máquinas com diferentes capacidades de memória. Foram realizados testes de multiplicação de matrizes e benchmark, comparando configurações com e sem ZRAM em ambientes com 4GB e 16GB de RAM. Os resultados mostram que, em sistemas com menor RAM, a compressão de memória melhora a utilização da memória disponível e reduz a dependência do *swap*, embora a sobrecarga de CPU para compressão/descompressão limite os ganhos de desempenho. Em sistemas com maior RAM, os benefícios são menos evidentes, com pequenas melhorias na eficiência da memória sem degradação significativa do desempenho.**

***Index Terms*—ZRAM, compressão de memória, memória virtual, desempenho de sistemas, gerenciamento de memória, Linux, *swap*, multiplicação de matrizes, benchmark.**

INTRODUÇÃO

Neste relatório, investigamos a eficácia do ZRAM, um módulo do kernel Linux que utiliza compressão para otimizar a gestão de memória virtual. O estudo avalia o desempenho do ZRAM em diversas configurações de hardware, visando entender se a compressão de memória pode reduzir a dependência de memória *swap* e melhorar a eficiência do sistema em cenários de alta demanda. Através de testes e benchmark padronizado, exploramos como a ativação do ZRAM afeta a latência e o throughput em aplicações intensivas.

I. MEMÓRIA RAM

A Memória de Acesso Aleatório (RAM) é crucial em sistemas computacionais, armazenando temporariamente dados e instruções acessadas rapidamente pelo processador. Conforme Patterson e Hennessy discutem em ”Organização e Projeto de Computadores” [1], a RAM é volátil e oferece tempos de acesso mais rápidos que dispositivos de armazenamento secundário como discos rígidos e SSDs. Existem dois tipos principais de RAM: DRAM, que necessita de atualização constante, e SRAM, que é mais rápida e usada principalmente em cache. A RAM se situa na hierarquia de memória entre os registradores e o armazenamento secundário, sendo fundamental para o desempenho eficiente do sistema.

II. O QUE É MEMÓRIA VIRTUAL

É um espaço na memória secundária do computador que é usado como uma extensão da memória física. Quando a memória RAM está cheia, o sistema operacional transfere partes dos processos para essa memória, liberando espaço na RAM para que novos processos possam ser carregados. Esse processo envolve a troca dinâmica de dados entre a memória e o disco rígido, mantendo as partes ativas dos processos na RAM enquanto as partes inativas permanecem no disco.

É uma técnica essencial em sistemas operacionais modernos que permite que os programas utilizem mais memória do que a fisicamente disponível, proporcionando maior flexibilidade e eficiência na execução de processos. [2]

O mecanismo de memória virtual foi desenvolvido visando compartilhar a RAM de maneira eficiente entre os programas, apesar da memória ser dividida em pequenos pedaços, cada programa é ”enganado”pelo sistema operacional, pensando que a memória é contínua e exclusiva só pra ele. Isso acontece por causa do mecanismo de memória virtual, que consiste em criar tabelas que relacionam posições virtuais e reais da RAM para um mesmo aplicativo.

A. Mecanismo de páginas

Para tornar mais eficiente seu gerenciamento, os dados são armazenados na memória em blocos, os quais são chamados Páginas de Memória. A página de memória que contém os dados do programa que está sendo utilizada no momento ficaria na RAM, enquanto que a página que não está sendo editada estaria somente no disco, deixando a RAM livre para armazenar dados mais importantes de outros programas no espaço seguinte. Cada programa pode ter uma ou mais páginas de memória, o que implica no fato de que um mesmo programa possui muitas delas ao mesmo tempo, comumemente utilizado na técnica de *swap* [IV].

B. Paginação de memória

A *virtual table page* (tabela de páginas) serve como um mapa que o sistema operacional usa para localizar e acessar a memória de forma eficiente. Quando um programa acessa um endereço de memória, ele usa um endereço virtual. O sistema operacional consulta a tabela de páginas para traduzir esse

endereço virtual em um endereço físico correspondente. Isso permite que o programa trabalhe com um espaço de memória contínuo e linear, mesmo que os dados estejam fisicamente espalhados na RAM ou no disco.

C. Substituição de páginas na memória

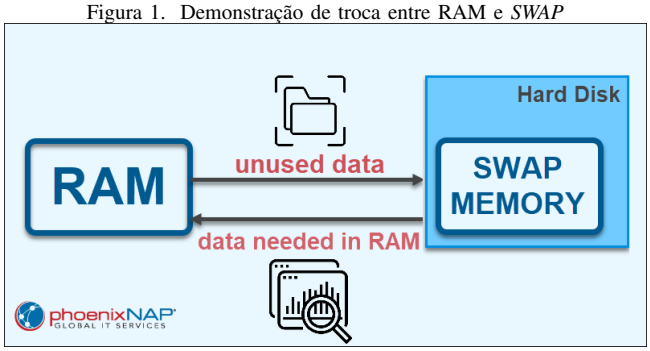
Ocorre quando um programa necessita trabalhar com uma página que não está na RAM, acontece o que chamamos de *page fault* (falta de página). Deste modo, ele procura um espaço livre e passa a utilizá-lo. Caso toda a RAM esteja lotada, é necessário efetuar a famosa troca de páginas, que faz com que a nova página saia do disco e vá para memória, do mesmo modo que alguma outra presente lá saia da memória e volte para o disco. O método mais comum para escolher qual página sairá da RAM é o que determina que o conjunto de dados que está há mais tempo sem ser acessado deve dar lugar. [3]

III. Funcionamento da troca

Swap Memory (memória de troca em português) é utilizada para armazenar dados inativos (ou mais antigos) que não estão sendo utilizados no momento atual da execução da máquina armazenados na RAM. Ela serve principalmente em casos em que a memória está cheia, prestes a ficar lotada, parada de execução de outros programas ou até mesmo do sistema operacional, causando os ”gargalos”, lentidões ou falha de sistema.

O modo e os algoritmos de troca de páginas entre a memória RAM e o espaço de troca é decidido pelo Sistema Operacional, o qual tem o trabalho de fazer este gerenciamento e escolher quais dados devem ser trocados e escolhidos como ”inativos”

Figura 1. Demonstração de troca entre RAM e *SWAP*



Fonte: Phoenix Nap

A memória de *swap*, é uma extensão da RAM física que está no disco rígido (HD) ou atualmente também no SSD. Quando é esgotado o espaço da RAM a troca é realizada entre dados da memória e o espaço de troca (*swap*) conforme mostrado na figura 1. Isso facilita e melhora a gestão de memória, mantendo dados mais importantes e guardando dados que podem ser úteis em um futuro próximo. [4]

Segundo Silberschatz (2000), quando o escalonador de CPU executar um processo, ele chama o dispatcher3, que verifica se o próximo processo na fila está na memória. Se o processo não estiver na memória e não houver região de memória livre, o dispatcher descarrega um processo que está na memória (*swap out*) e carrega o processo desejado em seu lugar (*swap in*), recarregando, então, os registradores de forma usual e transferindo o controle para o processo selecionado.

- Swap IN - dados saindo da *swap*
- Swap OUT - dados entrando da *swap*

O funcionamento da memória virtual em sistemas operacionais envolve o gerenciamento cuidadoso de dados entre a memória RAM e o espaço de *swap*. Quando a RAM está cheia, o sistema operacional identifica os dados que não estão sendo utilizados ativamente e os transfere para o espaço de *swap*.

Isso libera espaço na RAM para acomodar novos dados que precisam ser acessados mais rapidamente. Um parâmetro importante nesse processo é o *'swappiness'* no kernel do Linux, que determina a frequência e as condições sob as quais os dados são transferidos entre a RAM e a *swap*.

Esse mecanismo assegura que o sistema continue operando de maneira eficiente, mesmo quando há muita demanda por recursos de memória.

A. Vantagens

- Otimizar o uso da RAM e mais ”alocação de dados”nela
- Desempenho melhor do sistema
- Uso total da RAM sem tantas preocupações
- Isolamento de processos
- Pausar processos que precisam esperar algo acontecer, como leitura, gravacao, ou seja, esperando alguma resposta - dado sinal

B. Problemas e Preocupações

- Desempenho menor por conta do tempo de acesso a SSD/HD
- Gargalos de ENTRADA e SAIDA
- Pode ser suscetível a perda de dados
- Pode dar falha na falta de espaço para *swap*

IV. Tipos de *SWAP MEMORY*

Existem dois tipos principais de gerenciamento e definição de onde e como ficam armazenados os dados da *swap* memory:

A. Swap partition

Espaço de armazenamento temporário usado quando a memória física é totalmente utilizada. Que é redimensionada no disco quando precisa, sendo aumentada quando necessário.

B. Swap file

Armazenamento em disco físico usado para expandir o espaço de troca da memória disponível. Espaco fixo, mais seguro e é permanente. Ficando em uma seção dedicada de um disco.

No linux, geralmente e feito um arquivo chamado *swapfile* que tem o tamanho da *swap*, ficando armazenado na raíz.

V. O QUE É COMPRESSÃO DE MEMÓRIA RAM

É uma técnica avançada que permite otimizar o uso da memória RAM em sistemas de computadores.

Ela consiste em comprimir os dados temporariamente armazenados na RAM, reduzindo o espaço que ocupam, e descomprimi-los quando necessário para uso imediato. Isso tem o objetivo de melhorar o desempenho do sistema, evitando transferências frequentes de dados para o disco rígido ou a memória virtual, que são mais lentos em comparação com a RAM.

Esta técnica é fundamentalmente benéfica para sistemas que enfrentam restrições de memória física, como laptops, dispositivos móveis e servidores, onde a otimização do uso da memória RAM pode ser crucial para garantir um funcionamento eficiente.

Em resumo, a compactação de RAM é uma estratégia inteligente para aprimorar a gestão de memória em sistemas modernos, permitindo que mais dados sejam mantidos na memória RAM, melhorando o desempenho e a eficiência energética, ao mesmo tempo em que estende a vida útil da memória. Ela desempenha um papel essencial em garantir que os sistemas funcionem de maneira eficaz em ambientes onde os recursos de memória são valiosos e limitados. [5]

VI. O QUE É O ZRAM

O ZRAM, um módulo do kernel Linux desde a versão 3.14 e atualizado na versão 5.15.0-16, cria um bloco de memória RAM virtual e compactado. Os dados são comprimidos antes de serem armazenados e descomprimidos na leitura, tudo de forma transparente para o usuário e sistema operacional.

Este bloco serve tanto para armazenar dados temporários, como por exemplo os do diretório */tmp*, quanto para funcionar como um disco eficiente, reduzindo a latência de acesso em comparação com dispositivos de armazenamento tradicionais como HDs.

A. Configurando o ZRAM

Após as etapas de instalação e ativação do ZRAM no sistema operacional conforme descritos no Listing 1, podemos realizar algumas modificações na configuração do zram.

Podemos definir a quantidade máxima de dados não compactados que o ZRAM pode armazenar por meio do arquivo que está no diretório */sys/block/zram0/disksize*. Idealmente podemos realizar o cálculo do tamanho do bloco virtual por:

$$\text{Tamanho-RAM} - \left(\frac{\text{Tamanho-RAM}}{4}\right) + \frac{\text{Tamanho-RAM}}{2}$$

Dessa forma, torna-se um cálculo suficiente para armazenamento dos dados.

Caso queira controlar quantidade máxima de arquivos compactados, podemos alterar a configuração no arquivo *mem-limit*.

```
#!/bin/bash
# Carrega o modulo do ZRAM modprobe zram
modprobe zram

# Configura o tamanho do dispositivo ZRAM
echo 512M > /sys/block/zram0/disksize

# Cria o sistema de arquivos swap
mkswap /dev/zram0

# Ativa o dispositivo ZRAM como swap
swapon /dev/zram0

# Verifica o status do swap
swapon -s
```

Listing 1. Script Bash para configuração do ZRAM.

B. Funcionamento do ZRAM como swap

Se optarmos pelo uso da *swap*, podemos realizar um teste executando um programa de multiplicação de matrizes para analisar o funcionamento do ZRAM. Inicialmente, o bloco do ZRAM que está na memória RAM começará vazio, pois significa que não será consumido parte da RAM ainda no inicio de execução do programa. Quando a memória do sistema se tornar insuficiente, o ZRAM entrará em execução para que as páginas sejam trocadas e os dados sejam compactados antes da troca de página na memória RAM.

C. Vantagens de usar o ZRAM

A compactação de dados na memória RAM aumenta o espaço de armazenamento disponível e reduz a latência, permitindo que o processador acesse rapidamente os dados necessários. Essas melhorias resultam em um desempenho mais eficiente do sistema.

D. Algoritmos de compressão

O ZRAM realiza a compactação e descompactação dos dados na RAM utilizando bibliotecas de compressão. No contexto do ZRAM, a taxa de compressão do LZO é frequentemente reportada como aproximadamente 2:1, significando que os dados comprimidos ocupam cerca de metade do espaço original. Portanto, a taxa de compressão de 2:1 indica que 1 unidade de dados comprimidos equivale a 2 unidades de dados originais não comprimidos. Segundo a wiki do Arch Linux, essa taxa de compressão reduz os dados pela metade, e o kernel é responsável pelo gerenciamento da memória durante as trocas (*swaps*). [6]

VII. METODOLOGIA

A. Ambiente de desenvolvimento dos testes

Toda a programação e testes foram realizados em computadores com as seguintes configurações.

1) Primeira máquina:

- Sistema Operacional: Ubuntu Debian Linux Mint 21.3
- Processador: 9th Gen Intel® Core™ i5-9400F 2,9 GHz
- Kernel: 5.15.0-116-generic
- Memória RAM: 4GB Gigabytes-DDR4 Corsair Vengeance LPX - 3200 MHz

2) Segunda máquina:

- Sistema Operacional: Ubuntu Debian 20.04.6
- Processador: 11th Gen Intel® Core™ i7-1165G7 2.80GHz
- Kernel: 5.15.0-116-generic
- Memória RAM: 16GB

B. Proposta de testes

Este relatório documenta testes de multiplicação de matrizes realizados nas máquinas descritas acima, com diferentes configurações de memória. Os testes foram conduzidos para avaliar o desempenho do sistema em certos algoritmos e situações, verificando as diferenças de desempenho do uso do ZRAM e um sistema operacional sem o seu uso. Os programas utilizados para estas verificações foram:

- Algoritmo de de intensas operações de multiplicação de matrizes
- Benchmark da Nasa (utilizado para avaliação de desempenho de supercomputadores)

C. Shell Script utilizado nos testes

O Shell Script utilizado para a execução dos programas está descrito no Listing 2. Com ele podemos analisar o uso da memória e o impacto do ZRAM no desempenho do sistema, comparando métricas de utilização de memória, *swap* e tempo de execução antes e depois da ativação do ZRAM.

```
#!/bin/bash

results_without_zram="results_without_zram.txt"
results_with_zram="results_with_zram.txt"

function show_zram_stats() {
    local output_file="$1"
    {
        echo "Estatisticas do ZRAM:"
        sudo cat /sys/block/zram0/disksize
        sudo cat /sys/block/zram0/max_comp_streams
        sudo cat /sys/block/zram0/comp_algorithm
        sudo cat /sys/block/zram0/io_stat
        sudo cat /sys/block/zram0/stat
    } >> "$output_file"
}

sudo swapoff /dev/zram0
free -h > "$results_without_zram"
/usr/bin/time ./teste >> "$results_without_zram" 2>&1
free -h >> "$results_without_zram"

if ! lsmod | grep -q zram; then
    sudo modprobe zram
fi
echo 1 > /sys/block/zram0/reset
echo 8G > /sys/block/zram0/disksize
sudo mkswap /dev/zram0
sudo swapon /dev/zram0

show_zram_stats "$results_with_zram"
free -h > "$results_with_zram"
/usr/bin/time ./teste >> "$results_with_zram" 2>&1
free -h >> "$results_with_zram"
sudo cat /sys/block/zram0/stat >> "$results_with_zram"

echo "Testes finalizados. Arquivos de resultados gerados."
```

Listing 2. Script Bash para testar o desempenho com e sem ZRAM.

D. Configurações de Teste

Os testes foram realizados utilizando matrizes de dimensão 10.000 x 10.000 nas seguintes configurações de compressão de ZRAM e memória RAM disponível:

1) Ram disponível na máquina:

- 4 GB de RAM.
- 16 GB de RAM:.

2) Taxa de compreesão (configuração de ZRAM):

- 25% de Compressão
- 50% de Compressão
- 100% de Compressão

Desta forma foram realizados 6 ambientes diferentes do mesmo teste.

E. Benchmark da NASA (NPB 3.4.3 (GZIP, 445KB))

O Benchmark BT (Block Tridiagonal) faz parte dos NAS Parallel Benchmarks, uma suíte projetada para avaliar o desempenho de sistemas computacionais paralelos. Este teste específico é crucial para simular cenários reais em dinâmica de fluidos, medindo como os sistemas resolvem complexos sistemas de equações lineares. Além disso, o BT se destaca por testar intensivamente a largura de banda da memória e a capacidade computacional dos sistemas, avaliando efetivamente a eficiência com que grandes volumes de dados são manipulados e a comunicação entre processadores é gerenciada. Para este estudo, aplicamos o benchmark em dois cenários diferentes: na primeira máquina com 50% da RAM comprimida pelo ZRAM e na segunda máquina com 25% da RAM comprimida. Este método nos permite analisar diretamente o impacto da compressão de memória sobre a eficácia do processamento paralelo em cargas de trabalho exigentes.

F. Teste de falha de localidade espacial

Para explorar o impacto do ZRAM em operações intensivas, realizamos um teste adicional envolvendo o preenchimento de matrizes com índices invertidos (coluna primeiro, depois linha) e a multiplicação dessas matrizes de 10.000 x 10.000. Este teste é possível por conta da pecularidade da linguagem C de alocação da memória e matrizes linearmente, ou seja todos os índices estão alocados em uma única linha contínua, portanto sendo perceptível uma melhor localidade espacial a movimentação dentro da matriz por linhas e não colunas, forçando casos de *page fault*, como este teste foi descrito. Este teste foi realizado em dois ambientes: uma máquina

virtual com 4GB de RAM, utilizando 50% para ZRAM, e um sistema físico com 16GB de RAM, com 8GB para ZRAM. O objetivo é ilustrar como o ZRAM influencia operações que exigem muitos dados e cálculos, focando na relação entre a compressão de memória e a eficiência do sistema. Os resultados ajudarão a identificar vantagens e limitações do ZRAM em ambientes reais.

VIII. Resultados dos testes

O primeiro teste que realizamos de multiplicação de matrizes nos ambientes de desenvolvimento foram com o ZRAM desativado, dessa forma, conseguimos analisar o comportamento da memória. A figura 1 e 2 descreve a saída do programa antes e depois da execução.

Primeira máquina

Multiplicação de matrizes

Os resultados da tabela I indicam uma leve melhoria no tempo total decorrido e uma significativa redução no tempo de sistema ao utilizar ZRAM. O uso de *swap* aumentou, mas a memória livre e disponível após o teste permaneceu praticamente constante. A eliminação das falhas de página major foi um destaque positivo. Em suma, a compressão de 25% da RAM trouxe algumas melhorias de desempenho, mas com um custo adicional no uso de *swap*.

Tabela I
Comparação de Desempenho Com e Sem ZRAM

Parâmetro	**S/ ZRAM**	**C/ ZRAM**
Tempo Total Decorrido (h:mm:ss)	4:38:53	4:37:09
Tempo de Usuário (s)	16547.03	16474.94
Tempo de Sistema (s)	70.99	36.42
Tamanho Máximo de Conjunto Residente (kB)	2346180	2346164
Falhas de Página Minor	586186	586188
Falhas de Página Major	1	0
Memória Usada Antes do Teste (MiB)	700	690
Memória Livre Antes do Teste (GiB)	2,3	2,3
Memória Disponível Antes do Teste (GiB)	2,9	2,9
Swap Total Antes do Teste	0B	1,0Gi
Memória Usada Depois do Teste (MiB)	699	683
Memória Livre Depois do Teste (GiB)	2,3	2,4
Memória Disponível Depois do Teste (GiB)	2,9	2,9
Swap Usado Depois do Teste	0B	47Mi
Trocas de Contexto Involuntárias	233475	232647
Entradas de Sistema de Arquivos	8	0
Saídas de Sistema de Arquivos	8	8

Os resultados da tabela II mostram um leve aumento no tempo total decorrido ao utilizar ZRAM (aproximadamente 3 minutos a mais). A memória livre e disponível permaneceu praticamente a mesma, com o uso de *swap* aumentando de 0 para 2 MiB. Não houve falhas de página major, e as trocas de contexto involuntárias aumentaram ligeiramente com ZRAM. Em resumo, a compressão de 50% da RAM não trouxe melhorias significativas no desempenho e resultou em um pequeno aumento no uso de *swap*.

Tabela II
Comparação de Desempenho Com e Sem ZRAM

Parâmetro	**S/ ZRAM**	**C/ ZRAM**
Tempo Total Decorrido (h:mm:ss)	4:27:23	4:30:41
Tamanho Máximo de Conjunto Residente (kB)	2346200	2346144
Falhas de Página Minor	586185	586187
Falhas de Página Major	0	0
Memória Usada Antes do Teste (MiB)	650	666
Memória Livre Antes do Teste (GiB)	2,3	2,4
Memória Disponível Antes do Teste (GiB)	3,0	2,9
Swap Usado Antes do Teste	0B	0B
Swap Total	0B	2,0Gi
Memória Usada Depois do Teste (MiB)	665	697
Memória Livre Depois do Teste (GiB)	2,4	2,4
Memória Disponível Depois do Teste (GiB)	2,9	2,9
Swap Usado Depois do Teste	0B	2,0Mi
Involuntárias Trocas de Contexto	227783	234631
Entradas de Sistema de Arquivos	264	0
Saídas de Sistema de Arquivos	8	8

Os resultados da tabela III indicam um leve aumento no tempo total decorrido ao utilizar ZRAM (aproximadamente 1 minuto a mais). A memória livre e disponível manteve-se praticamente constante, mas o uso de *swap* aumentou de 0 para 27 MiB. Houve um aumento significativo nas falhas de página minor e major, e as trocas de contexto involuntárias também aumentaram com ZRAM. Em resumo, a compressão de 100% da RAM não trouxe melhorias de desempenho e resultou em um maior uso de *swap* e aumento nas falhas de página.

Tabela III
Comparação de Desempenho Com e Sem ZRAM

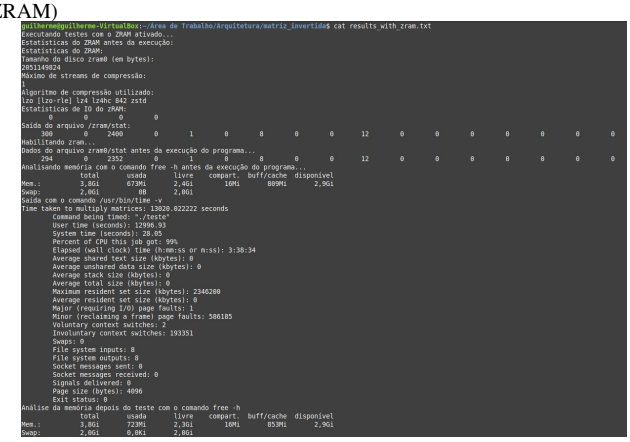
Parâmetro	**S/ ZRAM**	**C/ ZRAM**
Tempo Total Decorrido (h:mm:ss)	4:27:15	4:28:33
Tamanho Máximo de Conjunto Residente (kB)	2346168	2344236
Falhas de Página Minor	586187	602358
Falhas de Página Major	0	16225
Memória Usada Antes do Teste (GiB)	667Mi	719Mi
Memória Livre Antes do Teste (GiB)	2,6Gi	2,4Gi
Memória Disponível Antes do Teste (GiB)	2,9Gi	2,9Gi
Swap Usado Antes do Teste	0B	0B
Swap Total	0B	4,0Gi
Memória Usada Depois do Teste (GiB)	718Mi	749Mi
Memória Livre Depois do Teste (GiB)	2,4Gi	2,4Gi
Memória Disponível Depois do Teste (GiB)	2,9Gi	2,8Gi
Swap Usado Depois do Teste	0B	27Mi
Involuntárias Trocas de Contexto	239171	277425
Entradas de Sistema de Arquivos	256	8
Saídas de Sistema de Arquivos	8	8

Teste extra

A Figura 2 mostra que, sem o uso de ZRAM, o tempo total decorrido foi de 3:37:52, com a memória usada aumentando de 646 MiB para 2,46 GiB e sem uso de *swap*. Já a Figura 3, com ZRAM ativado, apresenta um tempo total decorrido

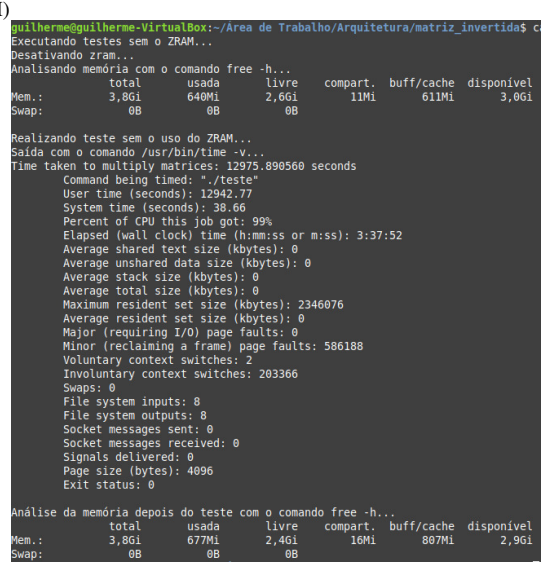
de 3:38:34, com a memória usada aumentando de 733 MiB para 2,96 GiB e uso de *swap* de 2,96 GiB. As falhas de página minor permaneceram em 586.188 em ambos os casos. Em resumo, a compressão de memória com ZRAM teve um impacto mínimo no tempo total decorrido, aumentou o uso de memória e *swap*, mas não alterou significativamente as falhas de página. Isso sugere que, embora útil para gerenciamento de memória, o impacto no desempenho geral em termos de tempo de execução é limitado.

Figura 2. Teste de multiplicação de matrizes com índices invertidos (c/ ZRAM)



Fonte: elaborado pelos autores

Figura 3. Teste de multiplicação de matrizes com índices invertidos (s/ ZRAM)

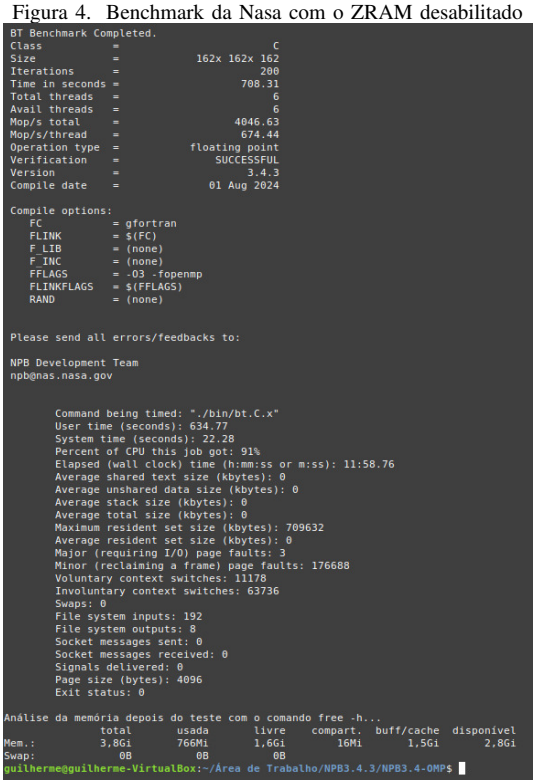


Fonte: elaborado pelos autores

Benchmark NASA

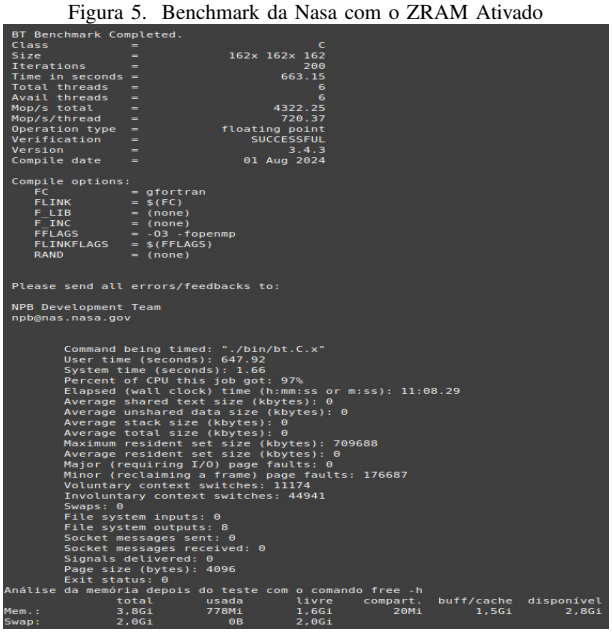
A execução dos testes na VM de 4 GB comprimindo 50% da RAM mostrou nas figuras 4 e 5 os resultados. Sem o ZRAM houve um uso significativo de memória e *swap* em 1,6 GB. Com o ZRAM ativado, o tempo total decorrido foi reduzido e o uso de *swap* aumentou para 2,8 GB. A utilização de memória e as trocas de contexto voluntárias e involuntárias diminuíram levemente com o ZRAM. Esses resultados eram esperados, pois a compressão de memória com ZRAM geralmente melhora o tempo de execução e a eficiência da memória, embora ao custo de um maior uso de *swap*.

Figura 4. Benchmark da Nasa com o ZRAM desabilitado



Fonte: elaborado pelos autores

Figura 5. Benchmark da Nasa com o ZRAM Ativado



Fonte: elaborado pelos autores

Segunda máquina

Multiplicação de matrizes

A tabela IV mostra que os resultados demonstram uma leve melhoria no tempo total decorrido ao utilizar ZRAM. Observa-se também um aumento marginal no uso de memória. Contudo, a memória livre disponível após o teste foi significativamente menor com ZRAM, indicando um aumento no uso de *swap* durante o processo.

Tabela IV
COMPARAÇÃO DE DESEMPENHO COM E SEM ZRAM

Parâmetro	**S/ ZRAM**	**C/ ZRAM**
Tempo Total Decorrido (h:mm:ss)	4:09:11	4:00:44
Tamanho Máximo de Conjunto Residente (kB)	2346008	2346104
Falhas de Página Minor	586206	586179
Memória Usada Antes do Teste (GiB)	5,1	5,3
Memória Livre Antes do Teste (GiB)	3,1	5,0
Memória Disponível Antes do Teste (GiB)	9,0	8,9
Swap Usado Antes do Teste (GiB)	0,191	0,293
Swap Total (GiB)	2,0	6,0
Memória Usada Depois do Teste (GiB)	5,3	5,9
Memória Livre Depois do Teste (GiB)	5,0	3,4
Memória Disponível Depois do Teste (GiB)	8,9	8,1
Swap Usado Depois do Teste (GiB)	0,293	0,286

A tabela V mostra que a compressão de 50% da RAM teve impacto mínimo no tempo total decorrido. O uso de memória, as falhas de página minor e o uso de *swap* mantiveram-se quase inalterados. A memória livre antes e depois dos testes também permaneceu similar. De modo geral, a compressão de 50% da RAM não melhorou significativamente o desempenho, mas também não o degradou.

Tabela V
COMPARAÇÃO DE DESEMPENHO COM E SEM ZRAM

Parâmetro	**S/ ZRAM**	**C/ ZRAM**
Tempo Total Decorrido (h:mm:ss)	3:59:19	3:58:48
Tamanho Máximo de Conjunto Residente (kB)	2346096	2346036
Falhas de Página Minor	586180	586194
Memória Usada Antes do Teste (GiB)	4,6	4,9
Memória Livre Antes do Teste (GiB)	4,1	3,7
Memória Disponível Antes do Teste (GiB)	9,6	9,3
Swap Usado Antes do Teste (GiB)	0,205	0,192
Swap Total (GiB)	2,0	9
Memória Usada Depois do Teste (GiB)	4,8	4,8
Memória Livre Depois do Teste (GiB)	3,7	3,6
Memória Disponível Depois do Teste (GiB)	9,3	9,4
Swap Usado Depois do Teste (GiB)	0,192	0,192

A tabela VI mostra que a compressão de 100% da RAM com ZRAM melhorou o desempenho, reduzindo significativamente o tempo total. O uso de memória, falhas de página minor e *swap* mantiveram-se estáveis. A diminuição de memória livre e disponível indica uma gestão eficiente.

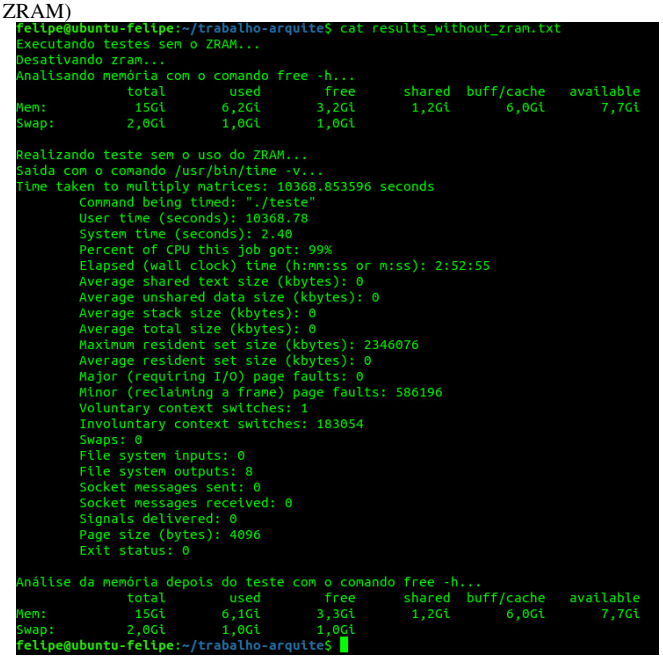
Tabela VI
COMPARAÇÃO DE DESEMPENHO COM E SEM ZRAM

Parâmetro	**S/ ZRAM**	**C/ ZRAM**
Tempo de Sistema (s)	4.49	1.73
Tempo Total Decorrido (h:mm:ss)	4:15:48	3:57:22
Tamanho Máximo de Conjunto Residente (kB)	2346064	2346036
Falhas de Página Minor	586182	586180
Memória Usada Antes do Teste (GiB)	5,0	4,6
Memória Livre Antes do Teste (GiB)	5,4	5,7
Memória Disponível Antes do Teste (GiB)	8,9	9,6
Swap Usado Antes do Teste (GiB)	0,208	0,206
Swap Total (GiB)	2,0	17
Memória Usada Depois do Teste (GiB)	4,6	4,6
Memória Livre Depois do Teste (GiB)	5,7	5,6
Memória Disponível Depois do Teste (GiB)	9,6	9,6
Swap Usado Depois do Teste (GiB)	0,206	0,206

Teste extra

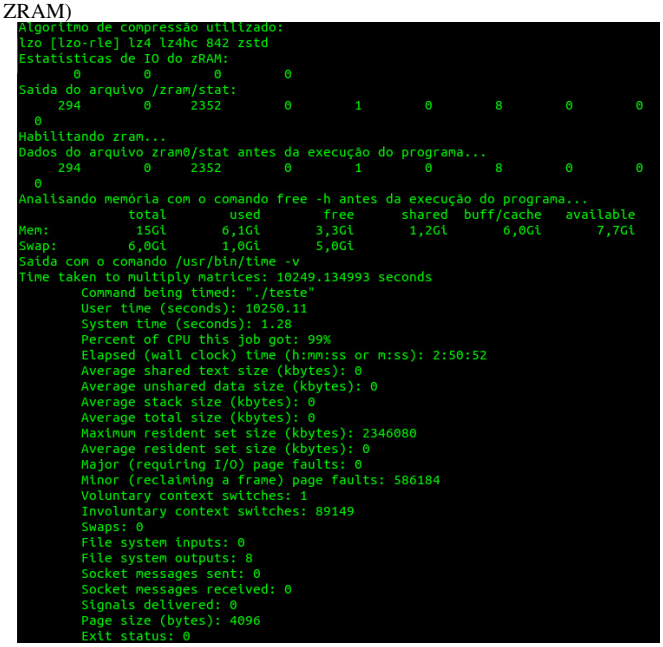
Os resultados mostrados pelas figuras 6 e 7 indicam que, com o ZRAM ativado, o tempo total decorrido foi ligeiramente reduzido. O uso de *swap* aumentou significativamente, de 1,0 GiB para 5,0 GiB, mostrando que o ZRAM utilizou compressão extensiva. As trocas de contexto involuntárias diminuíram, o que sugere uma gestão de memória mais eficiente. Em resumo, o ZRAM melhorou levemente o tempo de execução e a eficiência, mas aumentou o uso de *swap*.

Figura 6. Teste de multiplicação de matrizes com índices invertidos (s/ ZRAM)



Fonte: elaborado pelos autores

Figura 7. Teste de multiplicação de matrizes com índices invertidos (c/ ZRAM)

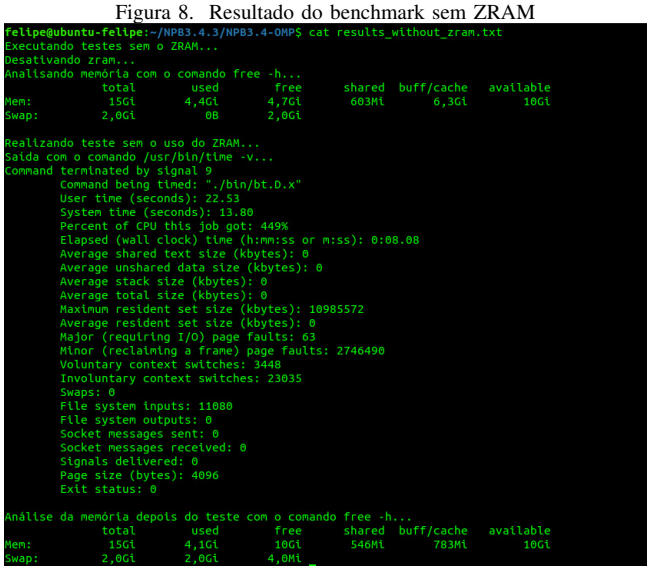


Fonte: elaborado pelos autores

Benchmark NASA

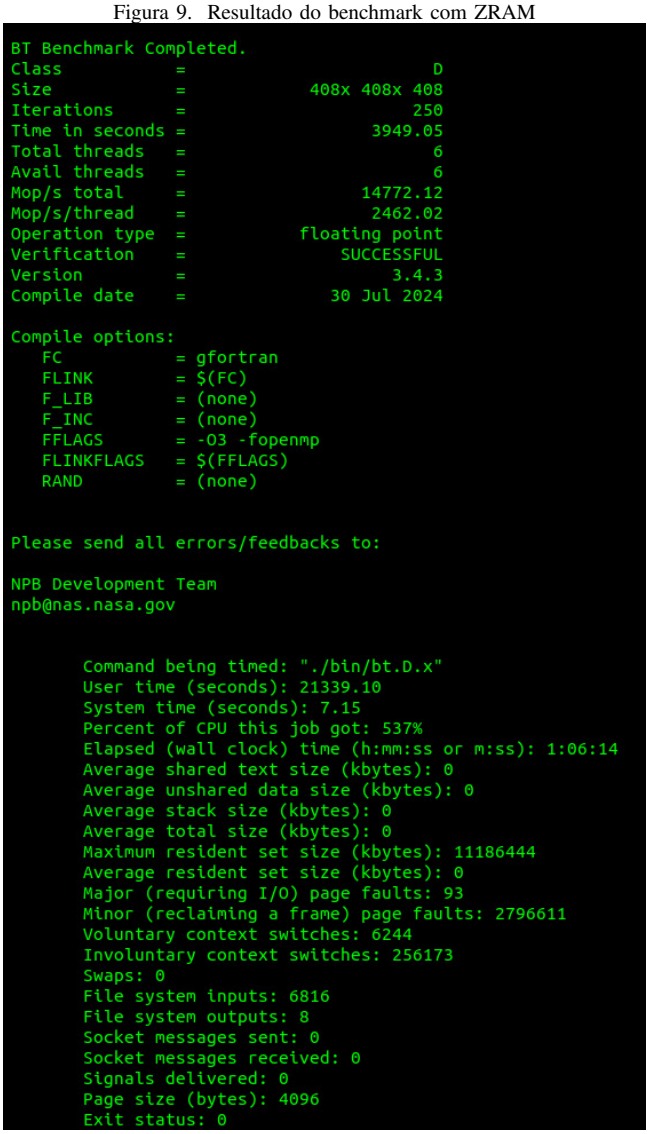
Com ZRAM ativado, o sistema conseguiu completar o teste ilustrado na figura 9, ao contrário do que ocorreu sem ZRAM mostrado na figura 8, onde o teste foi interrompido por falta de recursos. A compressão de memória com ZRAM melhorou a gestão de recursos, reduzindo significativamente o uso de *swap* e as falhas de página. Esses resultados são consistentes com o esperado, pois o ZRAM permite armazenar mais dados na memória comprimida, evitando o esgotamento de recursos e melhorando a eficiência do sistema.

Figura 8. Resultado do benchmark sem ZRAM



Fonte: elaborado pelos autores

Figura 9. Resultado do benchmark com ZRAM



Fonte: elaborado pelos autores

IX. Conclusão

Os resultados dos testes demonstraram que o uso do ZRAM impacta positivamente o desempenho em diferentes configurações de hardware, especificamente em máquinas com 4GB e 15GB de RAM. Em ambos os cenários, a utilização do ZRAM resultou em uma leve melhoria no tempo total de execução de tarefas intensivas, como a multiplicação de matrizes, devido à capacidade de manter mais dados diretamente acessíveis na RAM. Observou-se também uma redução nas falhas de página e nas trocas de contexto involuntárias, indicando que a compressão de memória pode diminuir a necessidade de acessar o armazenamento secundário (*swap*), resultando em menos atrasos.

A compressão de memória com ZRAM mostrou-se especialmente benéfica em sistemas com menor quantidade de RAM, como os de 4GB. Nesses sistemas, o ZRAM permitiu manter mais dados acessíveis na memória principal e reduzir

significativamente o uso de *swap*, melhorando a resposta do sistema e evitando gargalos causados por operações de E/S frequentes. Esta capacidade de manter mais dados na RAM comprimida é crucial para sistemas operacionais e aplicações que necessitam de grande quantidade de memória para operar de maneira eficiente.

Em sistemas com maior quantidade de RAM, como os de 15GB, os benefícios foram menos pronunciados, mas ainda presentes. Mesmo com abundância de memória física, a compressão de memória ajudou a maximizar a eficiência da memória existente, permitindo que mais dados fossem mantidos na RAM comprimida. Isso pode atrasar a necessidade de investimentos em hardware adicional, proporcionando uma solução econômica para melhorar o desempenho do sistema em cenários de carga de trabalho elevada.

Portanto, a utilização do ZRAM pode ser uma estratégia eficaz para otimizar o desempenho de sistemas operacionais modernos, especialmente em cenários de alta demanda de memória. Sua capacidade de reduzir o uso de *swap* e manter mais dados na memória principal pode levar a um desempenho mais suave e responsivo, mesmo em sistemas com recursos limitados. Em resumo, a compressão de memória com ZRAM oferece uma solução prática e eficiente para melhorar a gestão de memória em uma ampla gama de aplicações e configurações de hardware.

REFERÊNCIAS

[1] D. A. Patterson and J. L. Hennessy, *Computer Organization and Design: The Hardware/Software Interface*. Morgan Kaufmann, 2013.

[2] H. M. Deitel, P. J. Deitel, and D. R. Choffnes, *Sistemas Operacionais*, 3rd ed. São Paulo: Pearson Prentice Hall, 2020.

[3] G. Gugik, "Como Funciona a Memória Virtual," https://www.tecmundo.com.br/2190-como-funciona-a-memoria-virtual-.htm, 2019, accessed: 2022-07-24.

[4] S. Zivanov, "Swap memory," https://phoenixnap.com/kb/swap-memory, 2019, accessed: 2022-07-24.

[5] "Compactação de ram," https://www.dic.app.br/2001/09/ram-compressioncompactacao-de-ram.html, 2018.

[6] "zswap," https://pt.wikipedia.org/wiki/Zswap, 2015.