Caracterização de Operações de E/S por meio da Análise de Logs*

Thomas S. Wiederkehr[1], **Philippe O. A. Navaux**[1], **Cristiano A. Künas**[1]

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)

{tswiederkehr, navaux, cakunas}@inf.ufrgs.br

Resumo. *A popularidade de aplicações que exigem uma vasta quantidade de recursos computacionais, como algoritmos de Deep Learning (DL) e Machine Learning (ML), retoma a necessidade de pesquisar e otimizar os sistemas de alta performance computacional (HPC). Neste contexto, analisar e categorizar operações de E/S se torna crucial para realizar ajustes de performance, visto que estas ainda são um dos principais gargalos de desempenho em sistemas HPC. Neste artigo a abordagem de caracterizar operações de E/S por meio da analise de logs é demonstrada pelo estudo de aplicações de DL executadas no supercomputador Santos Dumont (SDumont).*

1. Introdução

O crescente número de aplicações que precisam lidar com vastas quantidades de dados vem tornando os sistemas de alta performance computacional (HPC) cada vez mais relevantes. Algoritmos de inteligência artificial (AI) como ML e DL são alimentados com volumosos datasets para ajustar os pesos de suas redes neurais durante o treinamento, realizando diversas operações de leitura ao longo de sua execução [Devarajan et al. 2020]. Estudos apontam que quanto mais complexos estes modelos se tornam, maiores são as suas capacidades de aprender e obter bons resultados [Amodei et al. 2016].

Operações de E/S ainda são um dos principais gargalos em aplicações HPC e, consequentemente, de algoritmos de AI. Isto se dá pela diferença que existe entre a velocidade de processamento e a velocidade de acesso a dados [Pavan et al. 2019]. Neste contexto, analisar e categorizar os diferentes tipos de operações de E/S que ocorrem nestas aplicações se torna crucial para realizar ajustes em seus parâmetros e melhorar sua performance.

Este artigo aborda uma forma de caracterizar a carga de trabalho de E/S encontrada em supercomputadores por meio da análise de logs. Para tal, estudamos o comportamento de dois tipos conhecidos de aplicações de DL, Classificação de Imagens e Tradução de Máquina, no supercomputador SDumont e utilizando arquivos de log gerados pelo Darshan. O restante do artigo é separado em: Seção 2, onde apresentamos a ferramenta Darshan; Seção 3, onde são descritos os modelos de DL utilizados, o processo de coleta de logs e as especificações de hardware do supercomputador; Seção 4, que demonstra os resultados obtidos; e Seção 5, que conclui o trabalho.

*Este estudo foi parcialmente apoiado pela Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001, pela Petrobras sob número 2020/00182-5 e pelo edital CNPq/MCTI/FNDCT - Universal 18/2021 sob número 406182/2021-3. Os autores agradecem ao Laboratório Nacional de Computação Científica (LNCC/MCTI, Brasil) pelo fornecimento de recursos de HPC do supercomputador SDumont, que contribuíram para os resultados da pesquisa relatados neste artigo. URL: http://sdumont.lncc.br

2. Ferramenta Darshan

O Darshan é uma ferramenta de caracterização de E/S que coleta informações de aplicações sem afetar consideravelmente o seu desempenho. Desenvolvida no laboratório de Argonne, é muito popular em estudos que utilizam seus logs para analisar cargas de trabalho de supercomputadores e caracterizar o comportamento das operações de E/S [Pavan et al. 2019, Devarajan et al. 2020, Chien et al. 2020].

A medida que uma aplicação é executada, os módulos de instrumentação do Darshan geram arquivos que caracterizam a carga de trabalho da aplicação dentre diferentes interfaces de E/S: MPI-IO, POSIX e STDIO [Paul et al. 2021]. MPI-IO é uma interface utilizada para operações de E/S paralelas, que são muito comuns em sistemas de arquivo paralelos (PFS) de computadores HPC. Já POSIX e STDIO são interfaces de E/S que oferecem, respectivamente, controle de baixo e alto nível a arquivos. Quando a aplicação é finalizada, cada módulo organiza, comprime e escreve seus arquivos de forma coletiva no log. MPI-IO é gravado para cada chamada do tipo *MPI_File_read()* e *MPI_File_write()*. O módulo POSIX grava cada chamada *read()* e *write()*, enquanto que o módulo STDIO cuida das funções da família *stdio.h*, como *fopen()*, *fprint()* e *fscan()*.

Diversos trabalhos de caracterização de operações de E/S em sistemas HPC já foram realizados no passado. [Paul et al. 2021] analisaram logs de Darshan a fim de entender o comportamento de E/S de aplicações de ML. Mais de 23,000 jobs de ML, obtidos durante o período de um ano no supercomputador Summit da IBM, foram analisados, classificando-os de acordo com o domínio cientifico de cada aplicação. Ao final, mostrou-se que as cargas de trabalho de ML geram um grande número de pequenas operações de leitura e escrita. Também existem abordagens que utilizam AI para caracterizar dados que já foram coletados, como [Pavan et al. 2019] que agruparam jobs de acordo com seus padrões de E/S utilizando o algoritmo de agrupamento k-means.

3. Metodologia

Como forma de demonstrar a abordagem apresentada na introdução, estudamos os comportamentos de E/S de algoritmos de DL através de logs gerados pelo Darshan. Após coletar os dados, é possível sumarizar o conteúdo dos logs por meio do comando `darshan-job-summary` que gera um pdf contendo informações sobre todos os acessos realizados pela aplicação.

3.1. Seleção das Aplicações

Os benchmarks foram implementados e executados em Python, que ainda é a linguagem mais popular para manipular aplicações de AI como ML e DL. Isto se dá pela sua implementação prática e pela vasta quantidade de bibliotecas (Caffe, Tensorflow, Torch, and MXNet).

Para cobrir uma maior variedade de cargas de trabalho e complexidades, selecionamos dois tipos distintos de aplicações do TBD Suite[1] (um suíte de benchmarks para treinamento de redes neurais profundas (DNN)): **Classificação de Imagens** e **Tradução de Máquina**. Classificação de Imagens é uma aplicação de DL clássica que utiliza o modelo Inception-v3 implementado com MXNet. Tradução de Máquina envolve análises sequenciais de dados e emprega redes neurais recorrentes (RNN), usando células LSTM como principal algoritmo. Para esta última, escolhemos o modelo implementado com as bibliotecas Seq2Seq e Sockeye, desenvolvidas em PyTorch. Ambas aplicações foram

[1] https://github.com/tbd-ai/tbd-suite

executadas com apenas uma época devido ao limite de tempo de 20 minutos das partições de desenvolvimento do sistema de agendamento de tarefas do SDumont.

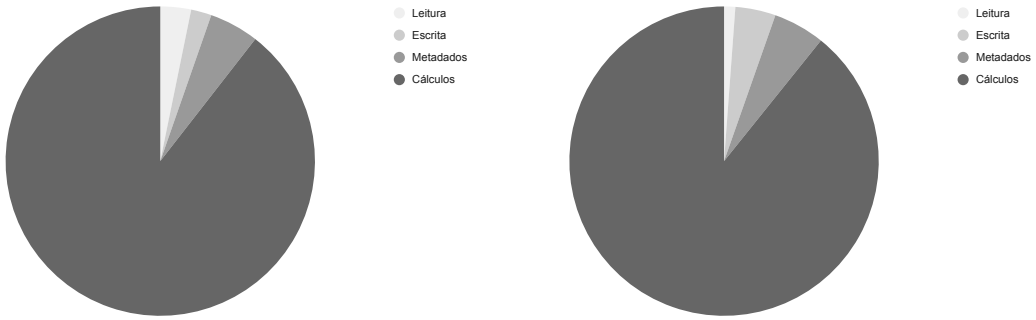
3.2. Especificações do Hardware

Os experimentos descritos foram realizados no Supercomputador SDumont, do Laboratório Nacional de Computação Científica (LNCC). Foi utilizado um único nodo **Bull Sequana X1120**, que consiste de 2 processadores Intel Xeon Cascade Lake Gold 6252 (2.1 GHz), 48 cores físicos, 384 GB de RAM e 4 GPUs NVIDIA Tesla V100-SXM2-32GB (apenas uma destas foi usada). SDumont tem um sistema de arquivos paralelo Lustre, v2.1, instalado em um ClusterStor Xyratex/Seagate v1.5.0.

4. Resultados

Esta seção apresenta uma análise das operações de E/S dos algoritmos de DL introduzidos anteriormente a partir dos dados coletados pelo Darshan nos dois tipos de aplicação.

O algoritmo de Classificação de Imagens levou um total de 813 segundos para executar, dos quais 26.1 segundos foram gastos com operações de leitura, 17.41 com operações de escrita, 42.1 com metadados e o restante com outros cálculos. No total, apenas 85.6 segundos foram gastos com alguma operação de E/S, mostrando que este algoritmo é dominado pelos cálculos, como pode-se verificar na Figura 1(a). Ao analisar essas informações podemos supor que a causa para este comportamento esteja ligado a execução de apenas uma época, ou pelo fato do dataset ser lido inteiramente e armazenado em memória.

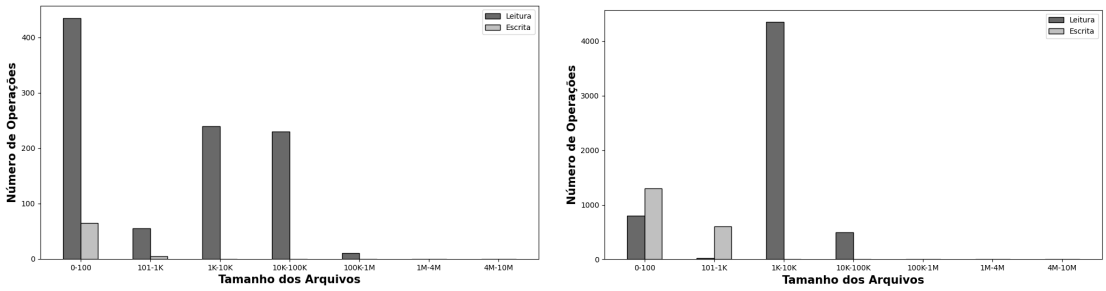


(a) Classificação de Imagem. (b) Tradução de Máquina.

Figura 1. Tempo gasto em leitura, escrita, metadados e cálculos (em %).

O histograma da Figura 2(a) foi obtido através da sumarização gerada pelo Darshan e exibe o tamanho dos acessos realizados pela interface POSIX assim como a quantidade de vezes que esses acessos foram realizados na aplicação de Classificação de Imagem. Como pode-se notar, o tamanho dos acessos varia entre 0-1 MB, sendo que a maior quantidade de acessos são operações de leitura de até 100 bytes, o que mostra que este algoritmo de Classificação de Imagens realiza uma série de pequenos acessos e raramente executa operações de escrita consideráveis. Em aplicações HPC tradicionais ocorre o inverso, são realizadas menos operações de E/S, porém, com tamanhos maiores [Bez et al. 2020].

Por outro lado, a aplicação Tradução de Máquina levou um total de 1116 segundos para completar sua execução e transferiu 44.4 MB pela interface POSIX e 1115 MB pela interface STDIO. A porcentagem de tempo de execução para leitura, escrita, metadados e



(a) Classificação de Imagem. (b) Tradução de Máquina.

Figura 2. Quantidade de operações x tamanho de acessos.

cálculos é apresentada na Figura 1(b). É possível verificar que a maior parte do tempo de execução segue sendo gasto com a própria aplicação, que equivale a cerca de 89%. Diferente do algoritmo de Classificação de imagens, neste *benchmark* a maioria dos acessos de POSIX se encontra no intervalo de 1k-10k, e os acessos também ocorrem com muito mais frequência. Mesmo que a quantidade de operações de leitura se mantenha maior que a de escrita, estas são mais frequentes neste algoritmo, e variam para tamanhos de acesso maiores como pode ser visualizado pela Figura 2(b).

5. Conclusão

Neste artigo foi caracterizado o comportamento de operações de entrada e saída pela análise de logs. Esta abordagem é uma prática comum de se obter informações importantes sobre o desempenho de E/S de supercomputadores, afim de aplicar otimizações a estes sistemas HPC. Ferramentas de caracterização, como o Darshan, possuem recursos para extrair dados importantes de aplicações sem afetar o desempenho destas de forma significativa. Estes dados podem ser refinados utilizando métodos da própria ferramenta ou por meio de softwares de terceiros como, por exemplo, os gráficos apresentados na Seção 4 que foram gerados tanto pelo Darshan quanto por Excel e Python.

Referências

Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Cheng, Q., Chen, G., et al. (2016). Deep speech 2: End-to-end speech recognition in english and mandarin. In *ICML*, pages 173–182. PMLR.

Bez, J. L., Carneiro, A. R., Pavan, P. J., Girelli, V. S., Boito, F. Z., Fagundes, B. A., Osthoff, C., da Silva Dias, P. L., Méhaut, J.-F., and Navaux, P. O. (2020). I/O performance of the Santos Dumont supercomputer. *IJHPCA*, 34(2):227–245.

Chien, S. W., Podobas, A., Peng, I. B., and Markidis, S. (2020). tf-Darshan: Understanding fine-grained I/O performance in machine learning workloads. In *2020 IEEE CLUSTER*, pages 359–370. IEEE.

Devarajan, H., Zheng, H., Sun, X.-H., and Vishwanath, V. (2020). Understanding I/O behavior of scientific deep learning applications in HPC systems.

Paul, A. K., Karimi, A. M., and Wang, F. (2021). Characterizing machine learning i/o workloads on leadership scale hpc systems. In *2021 29th MASCOTS*, pages 1–8. IEEE.

Pavan, P. J., Bez, J. L., Serpa, M. S., Boito, F. Z., and Navaux, P. O. (2019). An unsupervised learning approach for i/o behavior characterization. In *2019 31st SBAC-PAD*, pages 33–40. IEEE.