PerfEnerPy - Uma ferramenta para a avaliação da performance e eficiência energética de ferramentas *Dataframes*

André Martins[1] and Ricardo Vilaça[1][0000−0002−6957−1536]

HASLab - High-Assurance Software Lab, INESC TEC & U. Minho, Portugal
andre.c.martins@inesctec.pt, rmvilaca@di.uminho.pt

1 Introdução

Dataframes [7] são estruturas de dados, frequentemente utilizadas em linguagens de scripting, como o Python, que implementam algoritmos para análise de dados e que são capazes de representar princípios de álgebra relacional e linear. As *dataframes* tornaram-se uma estrutura de dados de eleição para a análise de dados, sendo o **pandas** [5] a ferramenta mais utilizada a nível global.

Apesar de ser a *framework* mais popular, o **pandas** possui bastantes limitações tanto a nivel da memória de uma máquina, como de paralelismo. Esta limitação revela-se um entrave para a análise de conjuntos de dados massivos, e , naturalmente, surgiram diversas ferramentas para a análise de dados massivos [8,12,1,16,13,18,15,21,4].

A comparação destas ferramentas não é uma tarefa linear e é necessária uma ferramenta de *benchmarking* que seja capaz de analisar e comparar as diferentes ferramentas escaláveis. Esta ferramenta permitirá de forma mais homogénea, fácil e fiável a avaliação das diferentes plataformas. Em detalhe esta ferramenta deverá: utilizar tanto conjuntos de dados reais como produzidos sinteticamente; ser extensível à implementação de novas ferramentas; recolher diversas estatísticas da execução assim como sobre os recursos, nomeadamente sobre consumos energéticos.

Neste artigo é apresentada uma ferramenta de *benchmarking* da performance e eficiência energética de um conjuntos de ferramentas escaláveis de processamento de dados, Secção 3. A ferramenta está disponível em código aberto em https://github.com/accunhamartins/PerfEnerPy, sendo facilmente extensível a novas *frameworks* e conjuntos de operações. Assim, esperamos que a ferramenta possa ser melhorada e expandida a novas áreas da análise de dados.

Na Secção 2, discutimos o trabalho relacionado, abordando as ferramentas de *benchmark* e estudos de performance de *dataframes*. Na Secção 4, realizamos uma pequena conclusão acerca do trabalho realizado, bem como o trabalho futuro.

2 Trabalho Relacionado

O trabalho de *benchmarking* de *dataframes* previamente existente é dirigido a conjuntos de dados e operações simples [3,15,17]. Existe trabalho de *benchmar-*

king em áreas similares como o YCSB [2] para *NoSQL/key-value stores* ou o TPCx-BB Express Benchmark[9] para medir o desempenho dos sistemas de Big Data baseados em Hadoop, executando interrogações analíticas. No entanto, estes não representam da melhor forma os conjuntos de operações e de dados mais utilizados em ambientes de utilização de *dataframes*. O FuzzyData[10] é um primeiro passo para *benchmarks* mais complexas, disponibilizando um sistema extensível de geração de dados sintéticos e de *workflows* para *dataframes* e que usámos neste trabalho.

Apesar da existência de algumas *benchmarks* com a preocupações de eficiência energética [11,19,14] as ferramentas de *benchmark* existentes para *dataframes* focam-se apenas na avaliação de performance. O artigo [14] apresenta um estudo do consumo de energia de três ferramentas de processamento de *dataframes* (pandas, Vaex e Dask), não apresentando operações complexas nem o impacto de diferentes configurações de *hardware*. Adicionalmente, não apresenta uma ferramenta de *benchmarking* que possa ser extensível a novas ferramentas e facilmente ajudar os programadores e cientistas de dados, a realizar estudos noutras configurações de *hardware*.

3 PerfEnerPy

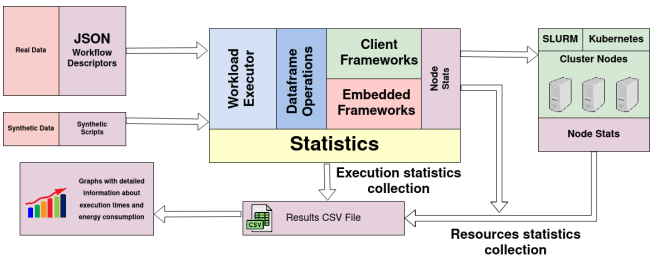

Figura 1. Arquitetura da ferramenta.

Esta secção descreve a ferramenta de benchmarking desenvolvida. Em detalhe, apresenta a arquitetura da mesma, os parâmetros que a mesma é capaz de avaliar, bem como os conjuntos de dados utilizados.

3.1 Arquitetura da ferramenta

Na Figura 1 é possível observar a arquitetura da nossa ferramenta de benchmarking. Para uma melhor compreensão, iremos descrever cada componente principal da mesma:

- ***Workload Executor*:** Responsável pela execução de todas as operações da *benchmark*, tanto com dados reais, como com dados sintéticos. Irá coordenar e orquestrar a execução das *frameworks* pretendidas, assim como garantir a escrita de resultados de performance e eficiência energética.
- ***Client Frameworks*:** *Frameworks* cuja execução recorre à utilização de *workers* externos à nossa ferramenta, e na ferramente apenas reside o cliente.
- ***Embedded Frameworks*:** *Frameworks* cuja execução é realizada dentro da nossa ferramenta, como, por exemplo, o Polars e o próprio pandas.
- **Resultados:** Todos os tempos de execução, bem como a coleção de estatísticas de recursos (CPU, memória, consumo energético, ...) são recolhidos tanto nos nós da benchmark, como onde correr as "client frameworks"e são guardadas em ficheiros, que são posteriormente analisados para gerar gráficos.

3.2 Workloads

Micro: Este modo realiza a avaliação por operação isolada, para um dado conjunto de operações pré-definidas. Estas operações são flexíveis à adição de novas ferramentas e operações, tornando a nossa ferramenta mais abrangente e de fácil utilização para novas frameworks. De momento são suportadas um total de 7 operações elementares de manipulação e análise de *dataframes*: **join** (junção de dois *dataframes*), **mean** (cálculo da média de uma dada coluna), **sum** (cálculo do somatório de uma dada coluna), **groupby** (agrupamento de dados de uma dada coluna aplicando a mean como função de agregação), **multiple_groupby** (agrupamento de dados de duas colunas aplicando a mean como função de agregação), **unique_rows** (cálculo de valores únicos de uma dada coluna) e **sort** (ordenação de uma dada coluna).

Macro: Este modo pretende avaliar *workflows* completos de análise e tratamento de dados. Estes tanto podem ser definidos pelo utilizador em formato JSON, como podem ser gerados de forma aleatória com base num conjunto de operações pré-definidas , com recurso à aplicação do Fuzzydata. De momento está pre-definido um *workflow* para análise e tratamento de dados do NYC-TLC. Essas mesmas operações estão encadeadas e conjugadas de maneira a que sejam todas executadas e no final tenhamos um resultado mais próximo de operações utilizadas em análise de dados, ao contrário do modo de *micro-benchmarking* em que apenas se avaliam as operações isoladas.

3.3 Fases

A nossa ferramenta de benchmarking pode ser dividida em 4 fases distintas de execução:

1. **Deployment** Foram desenvolvidas "scripts"para automatizar a instalação da bibliotecas necessárias, assim como para o caso de *client frameworks* o *deployment* de clusters da respetiva *framework*, assim como descarregamento dos conjundos de dados reais necessários e geração de dados sintéticos. O

script **setup** cria todos os ambientes virtuais Conda e Python necessários e instala todas as dependências Python necessárias para executar cada framework. Estas scripts para além do apoio ao *deployment* em si servem também inicio dos processos/micro-serviços necesários para a recolha de métricas sobre os recursos (memória, CPU, GPU, energia) durante a fase de execução da mesma.
2. **Carregamento de dados** Nesta fase são carregados os conjuntos de dados, tanto reais, como sintéticos, com que iremos realizar todas as operações. Cada *framework* implementada na ferramenta irá realizar o carregamento dos dados no seu próprio ambiente virtual e com os seus algoritmos e técnicas próprias.
3. **Execução** Como referido anteriormente, esta fase diz respeito à execução da *workload* definida para cada ferramenta a avaliar. Cada *framework* irá realizar o mesmo conjunto de operações e os seus tempos de execução serão guardados num ficheiro CSV. Para garantir que os resultados são credíveis e estão sustentados, cada conjunto de operações é realizado múltiplas vezes.
4. **Análise de resultados** Nesta fase são analisados os ficheiros CSV com as estatísticas sobre tempos de execução e consumos energéticos. Para facilitar a visualização e interpretação dos mesmos, são gerados gráficos que representam a informação recolhida. Esses gráficos permitem fazer uma comparação direta entre todas as ferramentas analisadas.

3.4 Ferramentas suportadas

Existem diversas ferramentas de processamento de dados escaláveis na comunidade Python. Neste artigo a *benchmark* suporta as seguintes ferramentas: pandas, Vaex, Modin [8] (Ray [6], Dask e MPI como *backend engines*), Dask [12], RAPIDS [21], PySpark [20], Polars, Datatable, DuckDB, Bodo e PyArrow.

4 Conclusão e trabalho futuro

De forma a responder à diversidade de ferramentas escaláveis para tratamento de *dataframes* que apareceram nos ultimos tempos, e que lidam com a maior complexidade e dimensão dos dados analisados, este artigo propõe o PerfEnerPy, uma ferramenta para avaliação de diferentes ferramentas escaláveis de processamento de dados. Esta ferramenta permite avaliar os compromissos entre performance e eficiência energética das diferentes ferramentas. Assim, esperamos que a ferramentas se revele útil para cientistas de dados e toda a comunidade que utilizada regularmente este tipo de ferramentas e que ajude na escolha de ferramentas mais adequadas ao caso de estudo onde forem inseridas.

Como trabalho futuro, seria interessante finalizar a implementação da utilização de Kubernetes para facilitar o *deployment*. Uma vez que a nossa ferramenta foi desenvolvida implementando a facilidade de inserção de novas ferramentas, seria interessante ver a mesma a ser expandida com novas adições.

Referências

1. Breddels, M.A., Veljanoski, J.: Vaex: Visualization and exploration of out-of-core dataframes. Astrophysics Source Code Library pp. ascl–1810 (2018)
2. Cooper, B.F., Silberstein, A., Tam, E., Ramakrishnan, R., Sears, R.: Benchmarking cloud serving systems with ycsb. In: Proceedings of the 1st ACM Symposium on Cloud Computing. p. 143–154. SoCC '10, Association for Computing Machinery, New York, NY, USA (2010). https://doi.org/10.1145/1807128.1807152, https://doi.org/10.1145/1807128.1807152
3. H2O.ai: Database-like ops benchmark. https://h2oai.github.io/db-benchmark// (2021), accessed: 2023-01-02
4. Kunft, A., Stadler, L., Bonetta, D., Basca, C., Meiners, J., Breß, S., Rabl, T., Fumero, J., Markl, V.: Scootr: Scaling r dataframes on dataflow systems. In: Proceedings of the ACM Symposium on Cloud Computing. p. 288–300. SoCC '18, Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3267809.3267813, https://doi.org/10.1145/3267809.3267813
5. McKinney, W., et al.: pandas: a foundational python library for data analysis and statistics. Python for high performance and scientific computing **14**(9), 1–9 (2011)
6. Moritz, P., Nishihara, R., Wang, S., Tumanov, A., Liaw, R., Liang, E., Elibol, M., Yang, Z., Paul, W., Jordan, M.I., Stoica, I.: Ray: A distributed framework for emerging ai applications. In: Proceedings of the 13th USENIX Conference on Operating Systems Design and Implementation. p. 561–577. OSDI'18, USENIX Association, USA (2018)
7. Petersohn, D.: Dataframe Systems: Theory, Architecture, and Implementation. Ph.D. thesis, Ph. D. Dissertation. EECS Department, University of California, Berkeley ... (2021)
8. Petersohn, D., Macke, S., Xin, D., Ma, W., Lee, D., Mo, X., Gonzalez, J.E., Hellerstein, J.M., Joseph, A.D., Parameswaran, A.: Towards scalable dataframe systems. arXiv preprint arXiv:2001.00888 (2020)
9. Poggi, N., Montero, A., Carrera, D.: Characterizing BigBench queries, hive, and spark in multi-cloud environments. In: Performance Evaluation and Benchmarking for the Analytics Era, pp. 55–74. Springer International Publishing (dec 2017). https://doi.org/10.1007/978-3-319-72401-0$_5$,
10. Rehman, M.S., Elmore, A.: Fuzzydata: A scalable workload generator for testing dataframe workflow systems. In: Proceedings of the 2022 Workshop on 9th International Workshop of Testing Database Systems. p. 17–24. DBTest '22, Association for Computing Machinery, New York, NY, USA (2022). https://doi.org/10.1145/3531348.3532178, https://doi.org/10.1145/3531348.3532178
11. Rivoire, S., Shah, M.A., Ranganathan, P., Kozyrakis, C.: Joulesort: A balanced energy-efficiency benchmark. In: Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data. p. 365–376. SIGMOD '07, Association for Computing Machinery, New York, NY, USA (2007). https://doi.org/10.1145/1247480.1247522, https://doi.org/10.1145/1247480.1247522
12. Rocklin, M.: Dask: Parallel computation with blocked algorithms and task scheduling. In: Proceedings of the 14th python in science conference. vol. 130, p. 136. SciPy Austin, TX (2015)
13. Shan, K., Perera, N., Lenadora, D., Zhong, T., Kumar Sarker, A., Kamburugamuve, S., Amila Kanewela, T., Widanage, C., Fox, G.: Hybrid cloud and hpc approach to high-performance dataframes. In: 2022 IEEE International Conference on Big Data (Big Data). pp. 2728–2736 (2022). https://doi.org/10.1109/BigData55660.2022.10020958

14. Shanbhag, S., Chimalakonda, S.: An exploratory study on energy consumption of dataframe processing libraries. In: Proceedings of the 20th International Conference on Mining Software Repositories. MSR '23, Association for Computing Machinery, New York, NY, USA (2023), https://drive.google.com/file/d/1SR1fXAx1OQZYNRFKS6iUKs5AUFn60Wr9/view?usp=sharing
15. Sinthong, P., Carey, M.J.: Aframe: Extending dataframes for large-scale modern data analysis. In: 2019 IEEE International Conference on Big Data (Big Data). pp. 359–371. IEEE Computer Society, Los Alamitos, CA, USA (dec 2019). https://doi.org/10.1109/BigData47090.2019.9006303, https://doi.ieeecomputersociety.org/10.1109/BigData47090.2019.9006303
16. Sinthong, P., Carey, M.J.: Polyframe: A retargetable query-based approach to scaling dataframes. Proc. VLDB Endow. **14**(11), 2296–2304 (jul 2021). https://doi.org/10.14778/3476249.3476281, https://doi.org/10.14778/3476249.3476281
17. Thiruvathukal, G.K., Christensen, C., Jin, X., Tessier, F., Vishwanath, V.: A benchmarking study to evaluate apache spark on large-scale supercomputers. CoRR **abs/1904.11812** (2019), http://arxiv.org/abs/1904.11812
18. Totoni, E., Anderson, T.A., Shpeisman, T.: Hpat: High performance analytics with scripting ease-of-use. In: Proceedings of the International Conference on Supercomputing. ICS '17, Association for Computing Machinery, New York, NY, USA (2017). https://doi.org/10.1145/3079079.3079099, https://doi.org/10.1145/3079079.3079099
19. Young, E., Cao, P., Nikolaiev, M.: First tpc-energy benchmark: Lessons learned in practice. In: Nambiar, R., Poess, M. (eds.) Performance Evaluation, Measurement and Characterization of Complex Systems. pp. 136–152. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)
20. Zaharia, M., Xin, R.S., Wendell, P., Das, T., Armbrust, M., Dave, A., Meng, X., Rosen, J., Venkataraman, S., Franklin, M.J., Ghodsi, A., Gonzalez, J., Shenker, S., Stoica, I.: Apache spark: A unified engine for big data processing. Commun. ACM **59**(11), 56–65 (oct 2016). https://doi.org/10.1145/2934664, https://doi.org/10.1145/2934664
21. Zedlewski, J.: End-to-End data science on GPUs with RAPIDS. USENIX Association (Jul 2020)