Assignment 1 – Exploratory data analysis

Rui Santos - 2020225542 Miguel Pedroso - 2019218176
Francisco Gouveia - 2023186817

3 de novembro de 2023

Resumo

Estudo estatístico realizado *datasets* relacionado com *workload logs* de um sistema paralelo de 128 nós - o NASA Ames iPSC/860, no âmbito da Unidade Curricular de Métodos Estatísticos em Informática. O esforço em garantir a execução em paralelo de vários *jobs* é grande, porém o número de trabalhos sequenciais é maior do que o número de trabalhos paralelos. O *runtime* chegou ao máximo de quatro horas. Grande parte das submissões são efetuadas entre as nove e dezoito horas, graças ao horário laboral em vigor, onde o *user group* 1 tem o destaque pela execução de mais tarefas. A média de trabalhos a executar em simultâneo é "1.7", o que significa que temos, maior parte do tempo, mais do que um *job* a executar simultâneamente e adicionalmente a média do *runtime*, por *job* é de "764"segundos, ou aproximadamente de "12"minutos. Certas falhas que possam aparecer na paralelidade do sistema devem-se à sua arqitetura de alocação de nós por sub-cubos.

1 Introdução

No âmbito da disciplina de Metodologias Experimentais em Informática, foi realizado o primeiro *assignment*, relacionado com análise exploratória de dados. Neste foi realizada a análise de um *dataset* entitulado "The NASA Ames iPSC/860 log", retirado de um repositório mantido pelo Experimental Systems Lab da Hebrew University em Israel, que contém diversos *datasets* reais de *workload logs* de sistemas paralelos.

2 Ferramentas Utilizadas

De forma a seguir o recomendado pelo Docente, os discentes decidiram utilizar uma ferramenta EDA (*Exploratory Data Analysis*, que ficasse responsável pela análise mais superficial do *dataset*, enquanto que, para o seu tratamento e consequente análise mais aprofundada, decidiu-se utilizar duas diferentes bibliotecas da linguagem de programação Python - Matplotlib, para efetuar o *plotting* dos diferentes gráficos que serão demonstrados posteriormente e por último, Pandas, para o tratamento, manipulação, limpeza e exploração dos *logs*.

3 O *dataset* NASA Ames iPSC/860

O sistema responsável pela produção destes *workload logs* é o IPSC/860 de 128 nós conectado numa arquitetura de *hypercube*. Os elementos constituintes deste sistema são microprocessadores Intel i860 ou Intel 80386 que operam à velocidade de 40-50 MHz. Cada nó tem ao seu dispor 8MB de memória. O log deste sistema, localizado na Divisão de Sistemas de Simulação Aerodinâmica Numérica (NAS) do Centro de Pesquisa Ames da NASA, contêm aproximadamente três meses de registros de aerociência computacional. Para além de trabalhos associados a esta àrea, também contêm dados relacionados às operações de mantimento e gestão do sistema, muita das vezes executados por tanto equipas de desenvolvimento como equipas de suporte da NAS. Nove *jobs* poderiam ser executados ao mesmo tempo, utilizando sub-cubos. Devido a isto mesmo, o tamanhos dos diferentes *jobs* são limitados a potências de dois.

Este sistema, naturalmente possuí regras de uso, maioritariamente associadas ao tempo alocado ao processo de *job* e às *queues* e o número de nós alocadas à mesma. É importante também, contextualizar

Figura 1: Intel iPSC/860 32-node.

que o sistema foi definido como tendo um *prime time*, onde certas filas e o *NQS scheduler*, responsável por, como indica o nome, manter, gerir e impor os limites das diferentes filas e dos *jobs* colocados nessas mesmo, se pode comportar de formas diferentes. Nos próximos pontos, iremos tentar abordar estes aspetos anteriormente mencionados, de forma a aumentar a compreensão sobre os mesmos.

3.1 Tratamento do Dataset

De modo a facilitar a exploraçao dos dados, o ficheiro .swf foi convertido em .csv, utilizando um *script* de Python. O *log* original tem uma job por linha, com diversos campos separados por espaço (*user, job, number of nodes, run time e start time*). Posteriormente este foi convertido para o formato *.swf* ou (*standard workload format*). Depois deste procedimento, o *dataset* continua a ter um *job* por linha, mas passa assim a ter 18 colunas:

- Job number: contagem de *jobs*, começa no valor "1".
- Start time: em segundos, começa na submissão do primeiro *job*.
- Wait time: a diferença entre o tempo de submissão do *job* e o tempo em que ele começou a correr.
- Runtime: a diferença entre o tempo de acabo e tempo de começo do *job*.
- Nodes allocated: número de processadores que o *job* utiliza.
- Cpu used: a média do tempo em que o processador foi utilizado, em segundos.
- Memory used: a média, por processador de memória utilizada em kilobytes.
- Processors requested: como o nome indica, o número de processadores requisitados pelo *job*.
- Requested time: este valor pode representar tanto o *runtime*, tal como a média do tempo usado de CPU, por processador.
- Memory requested: memória requisitada, por processador, em kilobytes.
- Status: 1 se o *job* foi concluido, 0 se falhou e 5 se foi cancelado.
- User: número natural entre um e o número de users distintos.
- User group: número natural entre um e o número de grupos distintos.
- Executable number: número natural entre um e o número de aplicações diferentes no workload.
- Queue number: número natural, entre um e o número de queues do sistema.
- Partition: número natural entre um e o número de partições do sistema.

- Previous job: número do *job* anterior no workload.
- Think time: número de segundos que deve decorrer entre o fim do *job* anterior e a submissão do *job* atual.

Neste ficheiro *.swf*, verificou-se que várias das colunas acima ficaram totalmente preenchidas com o valor de -1". Consultando posteriormente o *Standard Workload Format*, documento que pretende estandardizar todos os *logs* fornecidos pela Hebrew University chegou-se à conclusão de que não era possível, de facto, extrair esta informação do log original - assim as seguintes colunas permaneceram: Wait time, Cpu used, Memory used, Processors requested, Time requested, Memory requested, Status, Queue number, Partition, Previous job e Think time.

4 Análise Exploratória de Dados

Em primeiro lugar, decidimos analisar um número percentual que expressasse a quantidade de trabalhos realizados de forma sequencial, ou de forma paralela. Como conseguimos visualizar pela tabela seguinte da figura 1, o número de *jobs* sequenciais é maior (sensivelmente "17%").

Tipo de Trabalho	**Percentagem**
Sequencial	58.33%
Paralelo	41.66%

Tabela 1: *jobs* sequenciais *vs jobs* paralelos

Na tabela abaixo podemos verificar que em 25.4 porcento do tempo, o sistema esteve sem qualquer utilização. Só em cerca de 15.5 porcento do tempo todos os nós foram utilizados.

Número de nós	**Percentagem**
0	25.39%
128	15.47%

Tabela 2: número de nós utilizados em intervalos de 10 minutos

No gráfico apresentado abaixo, podemos observar o número de nós alocados por cada *job*. É importante observar que o número de nós alocados segue um padrão, sendo o número de nós alocados sempre uma potência de dois, como mencionado no ponto 3.1. Considerando que, como o sistema só possui um total de 128 nós, a maioria dos *jobs* opta só por alocar um número reduzido de nós. Isto é feito estrategicamente para evitar a ocupação total do sistema e permitindo a execução de outros jobs simultaneamente.

A análise do gráfico revela que a maioria das alocações de nós está na faixa inferior, evidenciada pela baixa percentagem de casos em que mais de trinta e dois nós foram alocados. Especificamente, menos de vinte nós, em média, foram alocados por *job*. Além disso, ao considerarmos a mediana dos dados, encontramos que metade dos *jobs* alocam apenas quatro nós ou menos.

Essas descobertas destacam a eficiência na utilização dos recursos do sistema, já que a maioria dos *jobs* optam por alocar apenas a quantidade necessária de nós, deixando espaço para outras tarefas. Essa estratégia contribui para a otimização do sistema, permitindo a execução paralela de diversas tarefas de forma eficaz. A média de nós alocados foi inferior a vinte, a mediana foi de quatro nós.

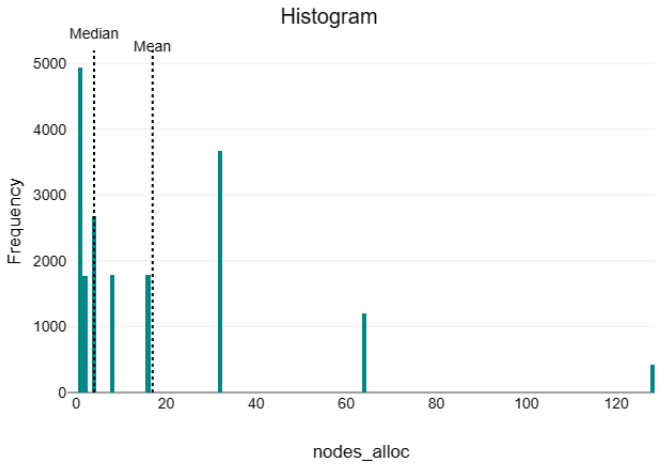

Figura 2: número de nós alocados por *job*

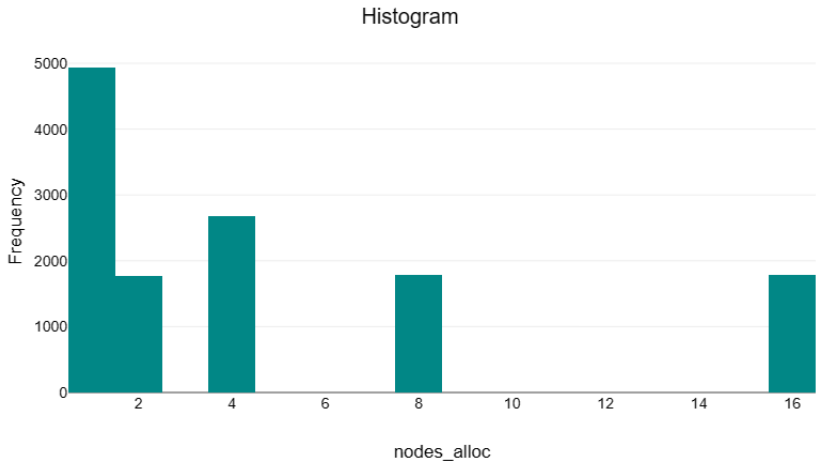

Figura 3: número de nós alocados por *job* (zoom até 16 nós)

Na figura 3 é possível observar a relação entre o número de nós alocados por cada *job* e o seu runtime. Os *jobs* que requerem todos os nós do sistema têm um *runtime* inferior a "15000"segundos ou cerca de quatro horas. A grande maioria teve um *runtime* inferior a "30000"segundos, ou oito horas e trinta minutos, com alguns *outliers* com um tempo superior. O job com maior runtime demorou cerca de "65000"segundos e alocou trinta e dois nós. Vários *jobs* tiveram um *runtime* de poucos segundos.

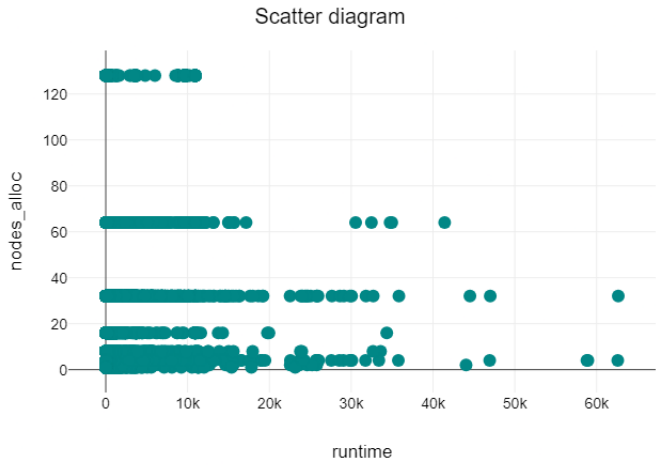

Figura 4: *runtime* e número de nós alocados por job

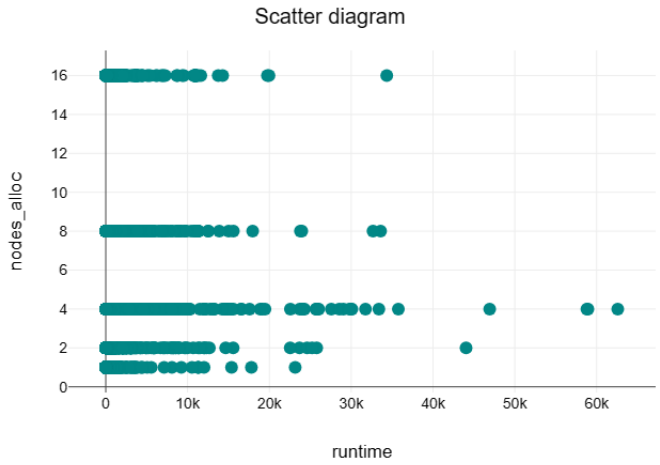

Figura 5: *runtime* e número de nós alocados por job (zoom até 16 nós)

Variável	**Média (em *s*)**
Runtime	764.887

Tabela 3: média de *runtime* por *job* em segundos

As figuras seguintes (4 e 5) representam os nós alocados pelo user group 1 e 2 e os runtimes dos respetivos grupos. Através da análise das mesmas, conseguimos retirar que o user group 2 consegue ter runtimes menores com menos recursos, o que nos leva a concluir que os *jobs* realizados pelos users deste grupo não requerem tanto poder computacional.

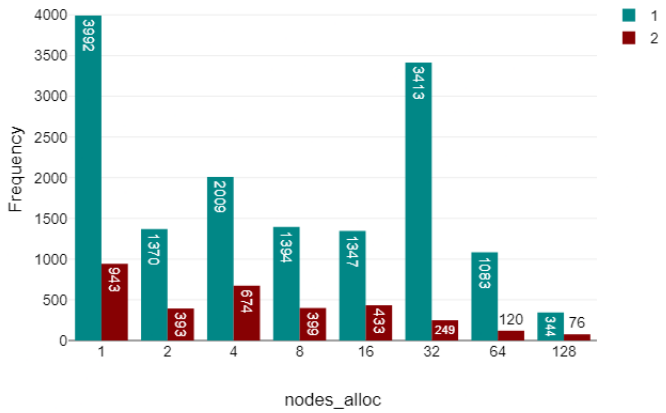

Figura 6: número de nós alocados por *user group*

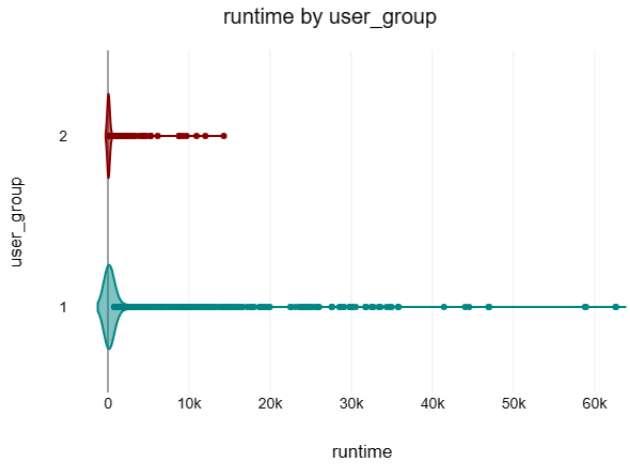

Figura 7: *runtime* por *user group*

Na figura 6 representamos a variação do número de jobs submetidos em função do dia da semana. Como podemos observar o número de submissões aumenta gradualmente atingindo o pico na quarta feira onde, a partir desse dia, começa a diminuir. Verifica-se também uma grande descida das submissões no fim de semana. Há partida este comportamento é verificado devido ao *Prime Time* - espaço de tempo definido de segunda-feira até sexta-feira, onde as filas disponíveis para executar *jobs* são as q16s, q16m, q32s, and q32m, nas quais o limite máximo de nós alocados seria 32 e uma hora de execução. No fim de semana, os *NQS jobs* poderiam eventualmente utilizar todo o cubo como eventualmente todas as queues que necessitassem.

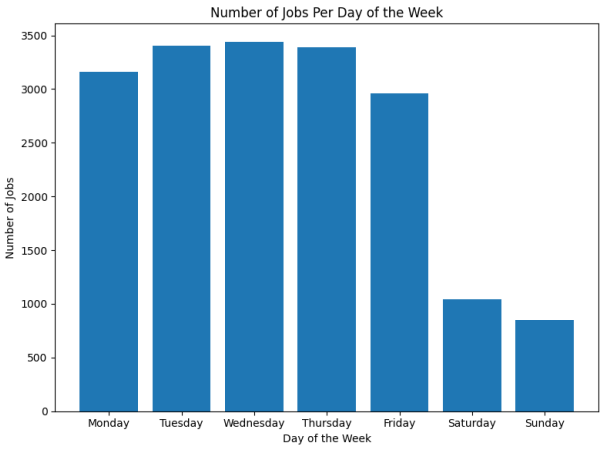

Figura 8: número de submissões por dia da semana

Na figura 9 podemos observar o número médio de nós em uso durante cada dia da semana. Tal como no gráfico acima, é possível observar uma discrepância entre os nós ocupados durante a semana e o fim de semana. O pico de uso é na quinta-feira, provalvemente relacionado com o facto de o maior número de submissões ser feita na quarta-feira, enquanto que o domingo é o dia em que o sistema está menos ocupado.

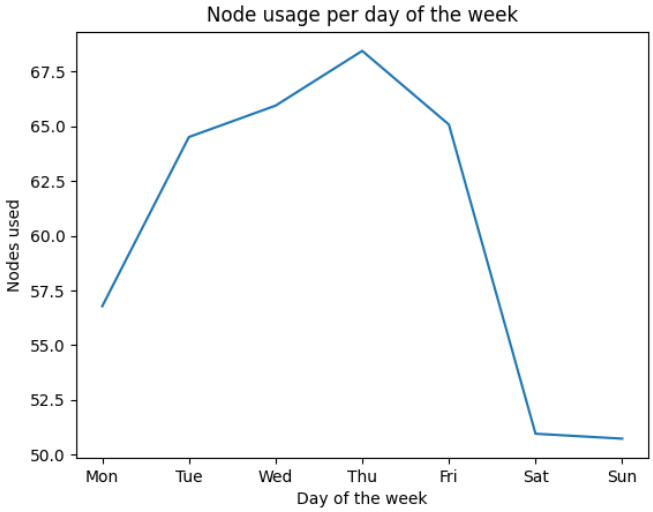

Figura 9: número médio de nós alocados por dia da semana

Na próxima figura está representada a variação da submissão de *job* durante as diferentes horas do dia, comparando também essa variação durante os dias de trabalho e o fim de semana. O maior número de submissões ocorre durante o horário de trabalho, entre as nove e as dezoito horas, atingindo o mínimo durante a noite. Durante o fim de semana, a variação entre as diferentes horas é substancialmente menor, havendo um pico de submissões às treze horas.

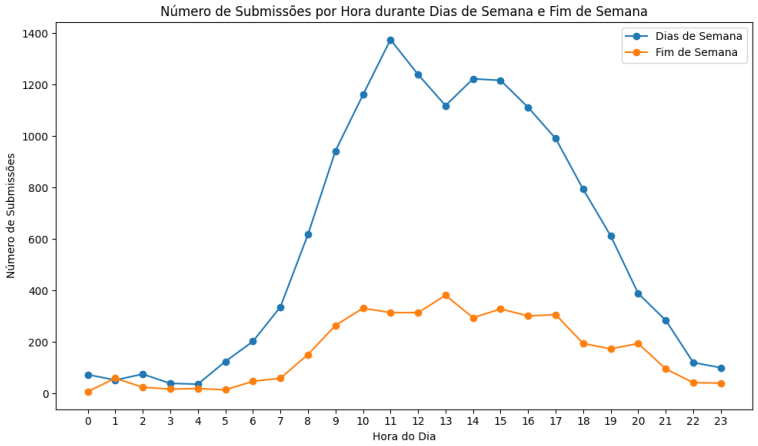

Figura 10: número de submissões por hora durante o dia (*weekdays* vs *weekend*)

Podemos observar, outra vez, um menor número de submissões ao fim de semana comparativamente aos dias de trabalho. Para além disto, o gráfico revela uma subida no número de submissões durante a manhã atingindo o pico por volta das onze da manhã. Durante a tarde o número de submissões diminui, mantendo-se baixo durante toda a noite. É também de notar uma pequena baixa no número de submissões entre as onze e as quatorze horas (hora de almoço).

Atentando nos diferentes fusos-horários referenciados no *dataset* e na sua respetiva documentação, podemos afirmar que o pico não se deve em grande parte ao que é evidente pelo gráfico, mas sim pelo facto de que os utilizadores de outros fusos horários, denominadamente da Costa Leste, começam o seu dia de trabalho por volta das nove horas da manhã. Aqui no gráfico, devido a questões de conversão de fuso-horário, o pico localiza-se por volta das onze horas.

A figura seguinte representa o número de nós alocados por hora do dia. Como podemos observar, este gráfico apresenta uma tendência muito semelhante à do gráfico anterior, ou seja, as horas de maior pico de submissões são também as horas onde são alocados mais nós.

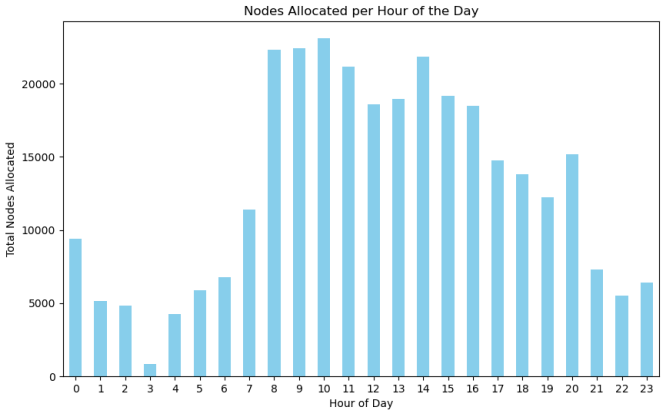

Figura 11: processadores alocados por dia

No que toca ao comportamento dos *users* (cerca de setenta), na figura abaixo podemos observar que a maioria deles submete relativamente poucos trabalhos. No entanto, o "user 4"destaca-se por submeter acima de "2500".

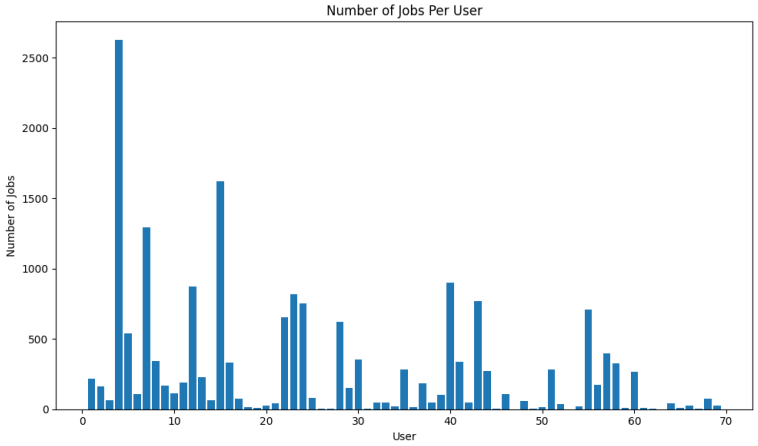

Figura 12: número de submissões por *user*

Na figura 13, conseguimos observar o número de *jobs* totais, e os nós, também no seu total, alocados por utilizador. Esta tem o fundamento de complementar a anterior figura, onde exploramos o número de submissões por utilizador. É necessário ter em consideração, primeiramente, que no eixo das ordenadas, observamos uma contagem. Verificamos mais uma vez, o destaque do utilizador 4 que alocou mais de "60000"nós no total. A sua contagem de número de *jobs* também é maior do que os restante dos utilizadores. Observa-se também a dispariedade previsível entre a quantidade de nós alocados *versus* a quantidade de *jobs* alocados, visto que um *job* normalmente implica a alocação de uma grande quantidade de nós, algo relacionado, inerentemente com a sua natureza.

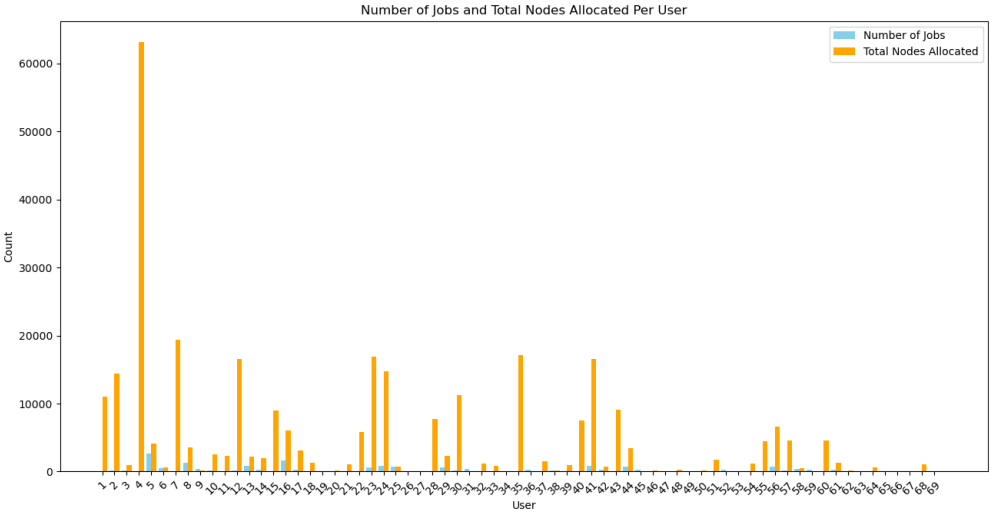

Figura 13: número de jobs e nós totais a serem executados por utilizador

Na figura 14, tentando acabar esta análise mais atómica, ligada ao utilizador e a sua interação com o sistema, temos um gráfico que demonstra, mais uma vez, a contagem de nós alocados por utilizador. Verifica-se mais uma vez, a predominância do utilizador número quatro, que apresenta mais de "60000"nós alocados.

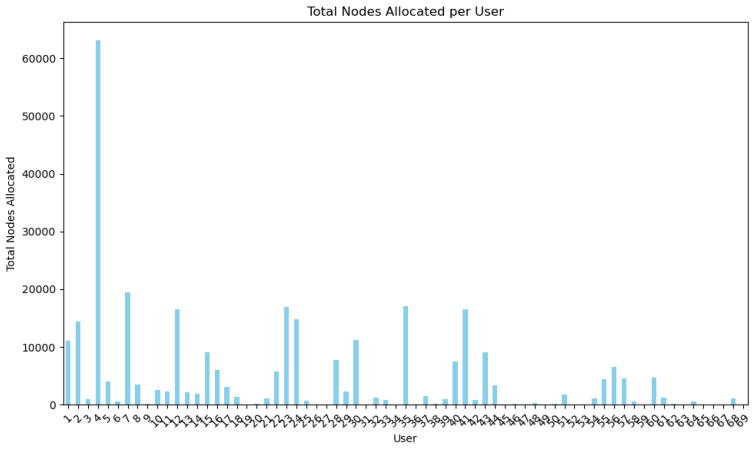

Figura 14: total de nós alocados por utilizador

Na figura seguite temos o número de jobs a decorrer ao mesmo tempo. Podemos observar que no máximo temos nove *jobs* a decorrer ao mesmo tempo, devido à limitação dos sub-cubos, com uma média de "1.74"não aparentando ter um padrão específico. Um detalhe importante mas fácilmente ignorado é falhas que conseguimos visualizar na região inferior no gráfico. Em relação a essas, referimo-nos no próximo gráfico.

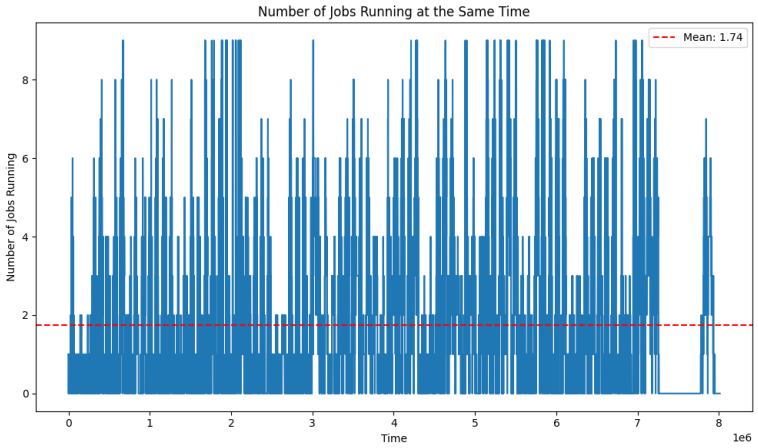

Figura 15: *jobs* a correr em paralelo

Como conseguimos evidenciar pelo seguinte gráfico de setores, existe uma grande percentagem de *downtime*. A semântica desta palavra, refere-se não a momentos onde o sistema esteja completamente desligado, mas onde não haja uma nova submissão, ou um intervalo de tempo que não tenha um novo *job* a executar. Aproximadamente vinte e cinco por cento dos três meses contidos no *dataset* são correspondentes a períodos de *downtime*. Relacionando algum contexto teórico com a arquitetura da máquina alvo de estudo estes "tempos mortos"são explicados pela falta de existência de *jobs* que sejam o pequenos suficientes, em termos de recursos computacionais, para que consigam preencher estes vazios. Em suma, falando de forma hipotética, se ocorrer uma situação onde temos um *job* de 64 nós a ser executado simultâneamente com um *job* de 4 nós, um outro *job* de 64 nós não pode ser executado - este comportamento é viável para diferentes cenários. Isto é explicado pela arquitetura de sub-cubos do iPSC/860, onde os *jobs* são obrigados a alocar sub-cubos com potências de dois.

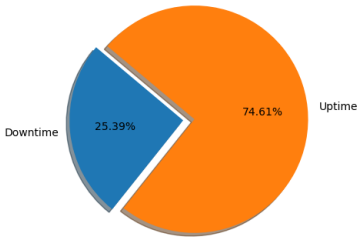

Figura 16: tempo de *downtime vs uptime*

Por fim, na figura 17 podemos comprovar que, apesar de haver uma grande quantidade de intervalos sem qualquer uso de nós, o número de intervalos em que todos os nós disponíveis no sistema estão a ser utilizados é consideravelmente superior. Existe um grande espaço de tempo sem atividade no fim do gráfico, que sugere que o sistema foi desligado, em princípio para manutenção. Em geral, pode-se concluir que o sistema foi bem aproveitado, uma vez que a maioria dos intervalos teve mais de metade dos nós a ser utilizados.

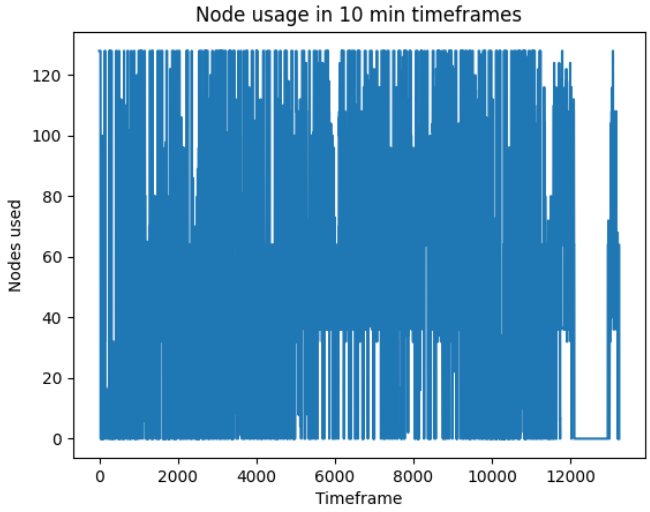

Figura 17: nós usados em cada intervalo de 10 minutos

5 Conclusões

Após a análise dos dados e gráficos produzidos previamente, podemos concluir que existe uma grande preocupação em garantir a execução em paralelo de vários *jobs*, tentando minimizar ao máximo o impacto dos que requeiram muitos nós, limitanto o seu *runtime* a um máximo de quatro horas. Mesmo assim, mostrou-se que a quantidade de *jobs* sequenciais é maior do que os que executam de forma paralela, sendo que os paralelos são os mais intensivos no que toca aos recursos computacionais e arquiteturais disponíveis. Também se verificou que em grande parte do tempo, os *jobs* a correrem em paralelo ocorrem mais frequentemente, mesmo que o *downtime* do sistema, tenha apresentado valores que são relativamente altos.

Em relação a horários, utilização por dias da semana, e por grupos de utilizador é percetível que a grande maioria das submissoes são feitas durante a semana e entre as 9 e as 18 horas, correspondendo ao horário laboral, diminuindo substancialmente durante a noite e ao fim de semana. Isto tornou-se evidente depois de algumas conversões no fuso-horário do *dataset* e *cross-referencing* com o *prime-time*

definido no sistema. Temos um destaque para o *user group* 1, que consome grande parte dos recursos computacionais.

Em suma, podemos afirmar que a grande parte dos aspetos abordados neste estudo estatístico se verificou através dos conceitos mais técnicos relacionados com as diferentes particulariedades do sistema, tanto a nível computacional como arquitetural. Expressa-se assim também ligeiras dificuldades no tratamento dos dados, especialmente pelo facto de que diversas colunas da versão limpa do *dataset* apresentavam valores ilegíveis, o que dificultou o alcançe do objetivo proposto pelo Docente. Porém, todos os discentes consideram que o mesmo foi tornado possível, e que os assuntos discutidos no presente relatório são de grande importância para a próxima etapa da Unidade Curricular.

6 Divisão de Tarefas

- Rui Santos: Análise Gráfica; elaboração de gráficos
- Miguel Pedroso: Análise Gráfica; elaboração de gráficos
- Francisco Gouveia: Análise Gráfica; Contextualização Teórica

Referências

[1] Britannica. (n.d.). Hypercube. Retrieved November 1, 2023, from https://www.britannica.com/technology/hypercube

[2] Chapin, S. J., Cirne, W., Feitelson, D. G., Jones, J. P., Leutenegger, S. T., Schwiegelshohn, U., Smith, W., & Talby, D. (1999). Benchmarks and Standards for the Evaluation of Parallel Job Schedulers. In Job Scheduling Strategies for Parallel Processing, D. G. Feitelson & L. Rudolph (Eds.), Springer-Verlag, 1999, Lect. Notes Comput. Sci. vol. 1659, pp. 66-89. DOI: 10.1007/3-540-47954-6_4.

[3] Intel iPSC - Wikipedia. (n.d.). Retrieved November 1, 2023, from https://en.wikipedia.org/wiki/Intel_iPSC

[4] Feitelson, D. G., & Nitzberg, B. (1995). Job characteristics of a production parallel scientific workload on the NASA Ames iPSC/860. In Job Scheduling Strategies for Parallel Processing, D. G. Feitelson & L. Rudolph (Eds.), Springer-Verlag, 1995, Lect. Notes Comput. Sci. vol. 949, pp. 337-360.

[5] Ramachandramurthi, S. (1996). iPSC/860 Guide. Computational Science Education Project at Oak Ridge National Laboratory. http://www.phy.ornl.gov/csep/CSEP/IP/IP.html