Análise de Envelhecimento de Software em uma Plataforma de Blockchain

Douglas Dias[1], **Ermeson Andrade**[1]

[1] Departamento de Computação – Universidade Federal Rural de Pernambuco (UFRPE)
Recife, PE – Brasil

{douglas.dias,ermeson.andrade}@ufrpe.br

Abstract. *Software aging is a phenomenon that plagues many long-running complex computer systems, which exhibit performance degradation or an increasing failure rate. Such a phenomenon may also be present in blockchain platforms. However, studies that focus on analyzing this phenomenon on these platforms are scarce. Thus, we adopted the Cardano blockchain to analyze software aging due to the presence of this technology in critical projects, its open-source nature and for being a sustainable solution. Considering the analysis of running a Cardano node on two computers with different configurations, we found evidence of software aging through memory degradation that was confirmed by the Mann-Kendall test. By analyzing the running processes, we confirmed that cardano-node (the main process of the platform) is the process possibly responsible for such degradation.*

Resumo. *O envelhecimento de software é um fenômeno que assola muitos sistemas computacionais complexos de longa execução, os quais exibem degradação de desempenho ou uma taxa de falha crescente. Tal fenômeno pode também estar presente em plataformas de blockchain. Porém, são escassos os estudo que focam em analisar esse fenômeno nessas plataformas. Assim, adotamos a blockchain Cardano para analisar o envelhecimento de software devido a presença dessa tecnologia em projetos críticos, sua natureza open-source e por ser uma solução sustentável. Considerando a análise da execução de um nó da Cardano em dois computadores com diferentes configurações, encontramos evidências de envelhecimento de software através da degradação da memória que foi confirmada pelo teste de Mann-Kendall. Através da análise dos processos em execução, confirmamos que o cardano-node (o principal processo da plataforma) é o processo possivelmente responsável por tal degradação encontrada.*

1. Introdução

O envelhecimento de *software* é um fenômeno inerente ao *software.* Assim como as pessoas, os *softwares* podem envelhecer, e apesar de não ser sempre possível prevenir sua ocorrência, seus efeitos podem ser limitados, seus danos podem ser revertidos e suas causas podem ser entendidas [Parnas 1994]. Cada vez mais, *softwares* permeiam nossas vidas, inclusive em áreas críticas como segurança e saúde, o que motiva não apenas o desenvolvimento de programas mais eficientes e mais seguros, mas também o acompanhamento constante de seu funcionamento por longos períodos. Sendo um problema

que muitas vezes é silencioso até o momento em que uma falha crítica ocorre, o envelhecimento de *software* vem sendo investigado e monitorado nos mais diversos sistemas. Porém, para a *blockchain*, uma tecnologia baseada em *hashing*, que está na base das plataformas para negociação de criptomoedas e na execução de contratos inteligentes [Pierro 2017], existem poucos estudos abordando este fenômeno e suas implicações em tais ambientes.

Apesar de *blockchains* estarem fortemente associadas com criptomoedas desde 2008, que foi o ano no qual Satoshi Nakamoto publicou o famoso *white paper* do Bitcoin [Nakamoto 2008], a blockchain em si nada mais é do que uma estrutura de dados que permite o registro seguro de transações. Entretanto, a implementação do Bitcoin em 2009 e sua eventual popularização demonstraram as possibilidades para tal tecnologia e influenciou o surgimento de vários projetos com as mais diversas aplicações [Baiod et al. 2021]. *Medicalchain*, por exemplo, é uma companhia sediada em Londres que utiliza a *blockchain* para armazenar o histórico médico de pacientes, dando a seus usuários o poder de armazenar e controlar os dados relacionados à saúde (ex. histórico de consultas ou exames), além de permitir o fácil compartilhamento desses dados com organizações médicas de maneira privada e segura. *BlockPharma*, por outro lado, utiliza *blockchain* para rastrear medicamentos e prevenir falsificações [Jeffery 2019]. Cada vez mais, diferentes áreas produzem soluções construídas com essa tecnologia, mostrando a sua relevância e a colocando em evidência. Assim, é essencial analisar os aspectos de confiabilidade dessas plataformas, especialmente em execuções de longa duração.

Apesar de existirem várias plataformas de *blockchain* (ex: *Hyperledger Fabric*, *Ethereum* e *Corda*), a Cardano se destaca por ser uma das maiores e mais importantes *blockchains* que utilizam o sistema *proof-of-stake*. Por tanto, este estudo almeja investigar a existência de sintomas de envelhecimento de *software* e suas possíveis causas na plataforma Cardano através de uma avaliação experimental da execução de um *cardano-node*. O nó, conectado à *mainnet* da Cardano, executou tarefas de sincronização da *blockchain* considerandos diferentes ambientes e cargas de trabalho provenientes de requisições de informações feitas ao nó. Nós coletamos várias métricas de desempenho do sistema, como uso de memória, uso de *Swap* e de CPU, e aplicamos o teste de Mann-Kendall para confirmar as suspeitas de envelhecimento de *software* no *cardano-node*. Além disso, identificamos os processos suspeitos da causa do envelhecimento do *software.* Nossas descobertas, bem como o método de análise, podem ser úteis para usuários ou desenvolvedores no processo de tomada de decisão de neutralizar problemas de envelhecimento de *software* encontrados em ambientes de *blockchain*.

O artigo está organizado da seguinte forma. A Seção 2 brevemente descreve o cardano. A Seção 3 detalha os trabalhos relacionados. A Seção 4 detalha nossos experimentos. A Seção 5 mostra os resultados dos experimentos e da análise estatística. A Seção 6 apresenta as conclusões e brevemente descreve os trabalhos futuros.

2. Cardano

Cardano é uma das várias plataformas *blockchain* de código aberto disponíveis. A plataforma foi desenvolvida com o objetivo de implementar um ecossistema sustentável e equilibrado que facilite as necessidades de usuários e também de outros sistemas que desejam realizar uma integração. A Cardano possui características atrativas como escala-

bilidade, interoperabilidade, descentralização e baixo consumo de energia quando comparado a outras plataformas, como o Bitcoin, devido a classe do seu mecanismo de consenso ser o *Proof of Stake*, ao invés do *Proof of Work*, que depende da resolução de problemas complexos e do consumo intenso de energia

A Cardano possui duas redes nas quais as aplicações e os sistemas de terceiros podem se integrar: a *testnet*, que é utilizada para testes de integração, e a *mainnet*, que é a principal rede da Cardano. As redes são formadas por nós, chamados de *cardano-nodes*, e que estão interconectados e trabalham em conjunto para validar as transações e os blocos por meio de consenso. Vale destacar que o *cardano-node* é o componente chave da Cardano, uma vez que sustenta a rede. Esse componente é implantando através de um contêiner que é executado nos computadores conectados e implementa vários componentes da plataforma, como o *ledger*, a rede, as configurações, os serviços de registro de atividades e o monitoramento, além do conjunto de protocolos *proof-of-stake Ouroboros*. Os nós se comunicam entre si através de um conjunto de mecanismos *IPC* (*Inter-Process Communication*).

3. Trabalhos relacionados

O envelhecimento de *software* é um fenômeno que assola muitos sistemas computacionais complexos de longa execução, os quais exibem degradação de desempenho ou uma taxa de falha crescente [Cotroneo et al. 2014]. Devido a suas implicações práticas, tanto a academia quanto a indústria têm estudado cada vez mais esse tema, resultando em numerosas publicações científicas e patentes registradas [Valentim et al. 2016]. À medida em que novas tecnologias são criadas e adotadas, o número de publicações envolvendo o envelhecimento de *software* em sistemas que as utilizam também cresce, a exemplo da computação em nuvem [Pietrantuono and Russo 2020]. Porém, ainda são escassos os estudos que almejem analisar o envelhecimento de *software* em plataformas de *blockchain*.

Os estudos sobre o envelhecimento de *software* são amplamente divididos em abordagens baseadas em modelos [Andrade et al. 2011] e em abordagens baseadas em medições [Andrade et al. 2021]. Muitos pesquisadores investigaram o envelhecimento de *software* em sistemas computacionais explorando modelos estocásticos como redes de Petri estocásticas (SPN), cadeias de Markov de tempo contínuo (CTMC) ou diagramas de blocos de confiabilidade (RBD). Por outro lado, a abordagem baseada em medições têm sido empregada em muitos estudos para caracterizar empiricamente fenômenos de envelhecimento de *software* em sistemas computacionais. Nessa abordagem, os dados de medição são coletados do sistema sob estudo para inferir a tendência de consumo de recursos ou degradação de desempenho causada pelo envelhecimento do *software.*

O único trabalho disponível na literatura que analisa indicativos de envelhecimento de software em uma blockchain é o trabalho apresentado em [Melo et al. 2022], onde os autores focam na plataforma Hyperledger Fabric. Diferentemente, o nosso trabalho tem como objetivo investigar a existência de sintomas de envelhecimento de *software* na plataforma Cardano através da verificação da degradação da memória durante a execução de um nó da rede *mainnet*. Também são considerados diferentes configurações de ambientes e cargas de trabalho. Adicionalmente, os processos suspeitos da causa do envelhecimento do *software* são identificados.

4. Experimentos

Nesta seção, primeiramente detalhamos as perguntas de pesquisa adotadas neste trabalho. Em seguida, a configuração experimental é apresentada. Posteriormente, são detalhados os testes de estresse adotados. Por fim, as métricas e os métodos de análise estatística são explicadas.

4.1. Perguntas de pesquisa

O objetivo deste estudo é investigar a ocorrência de eventos que afetam e provocam o envelhecimento de *software* na plataforma Cardano a partir da avaliação da execução do *cardano-node* e considerando diferentes configurações de ambiente e cargas de trabalho. Também visa investigar as causas principais de tal fenômeno, caso sejam observadas. Com base nesse objetivo, os experimentos foram planejados de acordo com as seguinte perguntas:

PP1: A execução do *cardano-node* apresenta algum problema relacionado ao envelhecimento de software?

PP2: Quais são as causas potenciais do envelhecimento do *software* na presença de algum fenômeno de envelhecimento de *software* na plataforma Cardano?

Para responder as perguntas acima, realizamos os experimentos, análises estatísticas e análise de processos que são detalhados nas próximas seções.

4.2. Configuração Experimental

Para a realização dos experimentos, configuramos dois computadores com a versão 1.29.0 do *cardano-node* (um com a configuração miníma e outro com a configuração recomendada), além da interface de linhas de comando *cardano-cli*, a partir do código fonte *cardano-src*. O *cardano-node* foi configurado para se conectar à *mainnet*, a rede principal da plataforma Cardano, a qual é formada por diversos outros nós em execução em diversos outros dispositivos. Vale destacar que cada uma das rodadas de experimentos só foi iniciada após a completa sincronização do *node* com a rede, no caso, a *mainnet*.

Em cada computador responsável pela execução do nó, também foram adicionados *scripts* para a obtenção de dados dos recursos e dos processos em execução, além de um programa com uma *API* criada com o *microframework* Flask, em Python, com um único *endpoint*, responsável por receber as requisições que consistem em solicitações das informações sobre o último bloco da *blockchain* O programa, por sua vez, utiliza o *cardano-cli* para solicitar estes dados, e então, os retorna para o cliente, que é um outro programa (gerador de cargas) escrito em Python com o propósito de estressar os computadores executando o *cardano-node*. O gerador de cargas foi instalado em uma Raspberry Pi e todas as requisições foram enviadas para os computadores por meio de uma rede local sem fio. As especificações dos componentes do sistema são dadas a seguir:

- Notebook executando o cardano-node nos requisitos mínimos: Dell Vostro 5470-A20, Intel Core i5-4200U (3 MB cache, dual core @2.6 GHz, 4 threads), 8 GB RAM 1600Mhz DDR3, SSD 240GB, SATA, leitura 500MB/s, escrita 350MB/s, Ubuntu 20.04.3
- Notebook executando o cardano-node nos requisitos recomendados: Acer Aspire VX15, Intel Core i7-7700HQ, (6 MB cache, quad core @3.8 GHz, 8 threads), 16 GB RAM, HDD 1TB, SATA, 5400 RPM, Linux Mint 20.3

- Raspberry Pi 3 B+, 1GB LPDDR2 SDRAM, 16GB SD, Raspberry Pi OS

A versão do *cardano-node* testada, 1.29.0, foi a última versão do *cardano-node* que os requisitos exigiam, no mínimo, 8GB de RAM para o computador que executaria um nó [IOHK 2022]. Portanto, o computador Dell Vostro 5470-A20 descrito acima atende aos requisitos mínimos de tal versão e representa a *baseline* dos experimentos.

4.2.1. Teste de Estresse

Para acelerar os potenciais sintomas de envelhecimento de *software* em cada computador executando o *cardano-node*, executamos testes de estresse usando um gerador de cargas, a partir de outro computador. Aplicamos três intensidades de cargas: baixa (requisições síncronas com o envio de no máximo 3 requisições por segundo), média (requisições síncronas com o envio de no máximo 8 requisições por segundo) e alta (requisições assíncronas com o envio de 20 requisições a cada segundo). Para cada experimento, executamos o sistema por 72 horas ou até que o *cardano-node* seja interrompido em decorrência de uma falha crítica ou interrupção do mesmo pelo próprio sistema operacional. Como resultado, teríamos 3 resultados de teste de longa duração para as cada configurações dos computadores, resultando em 6 resultados de teste no total. Porém, um dos computadores, que atendia apenas os requisitos mínimos, suportou apenas o teste de baixa carga, e portanto, obtivemos 4 resultados de teste ao final dos experimentos.

4.2.2. Métricas e análises

Para cada experimento, coletamos dados de monitoramento do sistema e analisamos indicadores de envelhecimento. Conforme mencionado em [Trivedi et al. 2010], os indicadores de envelhecimento referem-se a variáveis do sistema que podem ser medidas diretamente e podem estar relacionadas ao fenômeno de envelhecimento do *software.* Para os indicadores de envelhecimento, várias métricas foram coletadas, incluindo o uso de CPU, de memória e de *Swap*, os processos em execução e seus respectivos uso de recursos, além do tempo de resposta no dispositivo cliente. Neste estudo, iremos focar na investigação da presença de degradação de memória como indicador da existência de envelhecimento de *software.* Portanto, a utilização de memória e de *Swap* serão as métricas adotadas neste trabalho. Para ambas, foi aplicado o teste de Mann-Kendall de análise de tendência, uma vez que uma tendência de crescimento no consumo de memória é um forte indicador de envelhecimento de *software.* Para esse teste, foi utilizado o pacote ”trend”da linguagem R. O resultado do teste de Mann-Kendall retorna alguns valores, dos quais os mais relevantes em nosso estudo são o *p-value* (ou valor de p), cujo valor indica a existência de uma tendência no conjunto de dados, e o *S*, cujo valor indica uma tendência crescente, decrescente ou nula dos dados analisados.

O teste de Mann-Kendall adota duas hipóteses: a hipótese nula, de que não há tendência presente nos dados, e a hipótese alternativa, de que existe uma tendência temporal estatisticamente significativa nos dados, crescente ou decrescente. Se o valor do *p-value* retornado do teste for inferior ao nível de significância de 0,05, rejeitamos a hipótese nula e aceitamos a hipótese alternativa, pois isso indica que há evidência significativa de que uma tendência existe. Para definir se a tendência é de crescimento ou decrescimento,

Memória					
Computador/Carga de Trabalho	**Média**	**Desvio Padrão**	**Intervalo de Confiança (nível de confiança = 95%)**		
			Limite Inferior	**Limite Superior**	**Erro Padrão**
Dell Vostro / Baixa	89,76991	22,83787	88,12286	91,41695	0,83897
Acer Aspire / Baixa	58,53823	6,48198	58,34771	58,72875	0,09718
Acer Aspire / Média	64,37922	4,95793	64,23321	64,52522	0,07447
Acer Aspire / Alta	65,30512	3,94030	65,18756	65,42268	0,05996

Tabela 1. Uso médio de memória RAM

é analisado o valor de S. Se ele for positivo, a tendência é de crescimento, e se for negativo, a tendência é de decrescimento. Se o valor de S for zero, não existe tendência alguma. Por fim, os cinco processos em execução que mais utilizaram a memória foram coletados e analisados a fim de identificar os potenciais causadores do envelhecimento de *software.*

5. Resultados

Nesta seção, apresentamos os resultados dos experimentos e a análise estatística que realizamos para responder as perguntas de pesquisa.

5.1. Análises Preliminares

Nesta subseção, apresentamos alguns resultados preliminares sobre a execução dos experimentos, de modo que o uso de memória RAM e de *Swap* são analisados a partir dos dados obtidos durante os períodos de execução. A Tabela 1 detalha as os resultados relacionados ao uso da memória RAM, enquanto a Tabela 2 apresenta os resultados relacionados ao uso do Swap.

Para o Dell Vostro, sob uma baixa carga de trabalho (3 requisições síncronas por segundo), a média do uso de memória foi de 89,7% com o limite superior do intervalo de confiança em 91,4%. O alto valor para o desvio padrão, de 22,8%, denuncia a grande dispersão nos dados durante a execução do experimento nesse dispositivo. Vale ser destacado que o uso de memória se manteve alto durante toda a execução do experimento, sendo que o *cardano-node* foi prematuramente finalizado por volta da décima primeira hora devido a degradação da memória. Como o limite máximo do computador foi atingido já na execução da carga de trabalho mais baixa, testes equivalentes para as cargas média e alta não foram possíveis no Dell Vostro. Conforme descrito nos requisitos de sistema do *cardano-node*, 8GB de RAM é de fato um requisito mínimo para a execução da versão 1.29.0, porém a finalização do processo após poucas horas de execução indicam que tal quantidade de memória RAM é insuficiente para essa versão do *cardano-node*.

Para o Acer Aspire, a média do consumo de memória apresentou variações entre si de acordo com a carga de trabalho aplicada. Isto é, houve uma influência direta entre a carga de trabalho e o consumo de memória RAM. É notável a diferença significativa que se apresentou entre os experimentos de baixa (3 requisições por segundo) e de média (8 requisições por segundo) cargas de trabalho, sobretudo quando comparados o da média e o da alta (20 requisições assíncronas por segundo). Adicionalmente, vale destacar que os intervalos de confiança não se sobrepõem, significando que as médias são estatisticamente diferentes.

O uso de *Swap* no Acer Aspire, em todas as cargas de trabalho, foi muito baixo. Embora o experimento com carga média tenha consumido mais *Swap* do que o experi-

Swap					
Computador/Carga de Trabalho	**Média**	**Desvio Padrão**	**Intervalo de Confiança (nível de confiança = 95%)**		
			Limite Inferior	**Limite Superior**	**Erro Padrão**
Dell Vostro / Baixa	38,80729	5,19909	38,43233	39,18224	0,19099
Acer Aspire / Baixa	0,00000	0,00000	0,00000	0,00000	0,00000
Acer Aspire / Média	0,10000	0,00000	0,10000	0,10000	0,00000
Acer Aspire / Alta	0,03323	0,06981	0,03115	0,03532	0,00106

Tabela 2. Uso médio de Swap

mento com carga alta, ele não variou no decorrer das horas e já demonstrava o uso de 0,1% desde o começo, além do intervalo de confiança nos mostrar que os limites inferiores e superiores são iguais a média e o desvio padrão é 0. Portanto, para a carga média, consideramos que a execução do *cardano-node* não influenciou no consumo de *Swap*. Já sob alta carga de trabalho, o uso de *Swap* apresentou um leve crescimento com o passar do tempo, apesar desse consumo ser extremamente baixo. Com relação ao Dell Vostro, por outro lado, os resultados mostram um considerável consumo de *Swap*, sobretudo considerando que o *Swap*, normalmente, só é utilizado quando o sistema fica sem memória principal disponível, o que explica a interrupção do *cardano-node* explicado acima.

5.2. Análises de Indícios de Envelhecimento de *software*

Nesta subseção, iremos analisar as perguntas de pesquisa apresentadas acima. Primeiramente, analisaremos as tendências de degradação na memória disponível (PP1). Caso a suspeita de envelhecimento de *software* seja confirmada, nossa próximo objetivo é saber quais são os potenciais processos causadores de tais degradações da memória (PP2). Vale destacar que para o Dell Vostro, a carga de trabalho utilizada foi a baixa, visto que as outras não foram suportadas pelo computador. Para o Acer Aspire, por outro lado, foram utilizadas as cargas baixa, média e alta.

5.2.1. Dell Vostro

A Figura 1 apresenta a análise de memória e *Swap* para o Dell Vostro. É possível notar um rápido crescimento para ambos logo na primeira hora de execução. Em seguida, o consumo de memória cai levemente, mas se mantém elevado, enquanto o consumo de *Swap* cresce cada vez mais. A décima hora marca o período de pico de uso do *Swap*, próximo de 45%, enquanto que o uso de memória ainda se mantinha acima de 90%, como esteve desde o início do experimento. Depois da décima hora, ambos passam a cair em consequência da finalização da execução do processo *cardano-node* pelo OOM Killer[1].

A Tabela 3 exibe os resultados do teste de tendência de Mann-Kendall, considerando até a décima hora, ou seja, antes da finalização do processo. Conforme mostram os resultados, para ambas as métricas, a tendência é de crescimento tanto para o uso de memória quando de *Swap*, visto que os valores de p são menores do que 0,05, o que indica a existência de tendência, e o valor de S foi positivo, o que indica uma tendência de crescimento. Em outras palavras, observamos uma tendência de degradação estatisticamente

[1]O *Out of Memory Killer* é o processo empregado pelo kernel do Linux quando o sistema está com a quantidade de memória disponível criticamente baixa. Ele analisa os processos em execução e finaliza um ou mais deles para liberar memória e manter o sistema em pleno funcionamento.

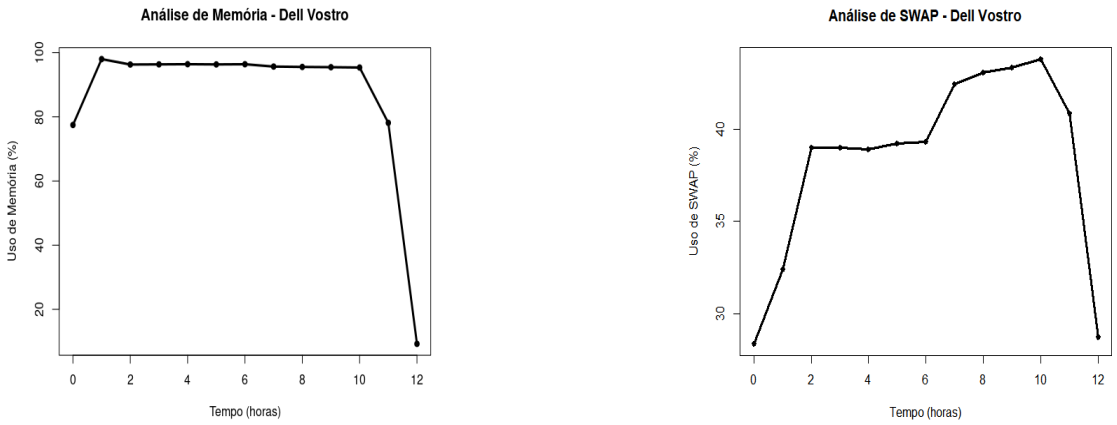

Figura 1. Utilização da Memória e *Swap* - Dell Vostro

Teste MK - Dell Vostro			
Métrica	**S**	***p-value***	**Tendência**
Memória	484	1,829e-08	Crescimento
SWAP	1,98e+06	2,22e-16	Crescimento

Tabela 3. Testes de Mann-Kendall para o Dell Vostro

Top 5: Processos que Mais Consumiram Memória - Dell Vostro	
PID	**Descrição**
6139 (cardano-node)	O processo do *cardano-node*.
1627, 1164 (gnome-shell)	Shell gráfico do ambiente da área de trabalho do GNOME.
7442 (unattended-upgrade)	Processo em *background* para instalação automática de atualizações de segurança (e outras) no Ubuntu.
1450 (Xauthority-backgroundnone)	Um dos processos do Xorg, *software* de sistema e protocolo que fornece uma base para GUIs.

Tabela 4. Descrição dos processos que mais consumiram memória durante o experimento no Dell Vostro.

significativa de memória e de *Swap* disponíveis para carga de trabalho baixa, sugerindo a suspeita de envelhecimento de *software.*

Para investigar melhor as causas subjacentes do fenômeno suspeito de envelhecimento, realizamos uma análise de processo para o Cardano. Criamos um programa em Python para coletar informações sobre os processos em execução no sistema a cada hora. Este programa usa o comando *ps* com *eo* para recuperar informações do processo, como Número de Identificação do Processo (PID), Tamanho do Conjunto Residente (RSS) e Tamanho do Conjunto Virtual (VSZ). A Figura 2 exibe os cinco processos que mais consumiram memória no Dell Vostro, enquanto a Tabela 4 apresenta a descrição de tais processos. Os resultados revelam que o processo do *cardano-node* consumiu em média mais de 80% da memória disponível, enquanto os quatro outros processos com maior consumo, todos eles parte do sistema operacional (neste caso, Ubuntu), utilizaram pouca memória. Desta forma, podemos concluir que a execução do *cardano-node* contribuiu com grande parcela para a exaustão de memória ocorrida durante o experimento, o que nos indica que ele é o principal suspeito do envelhecimento de *software.*

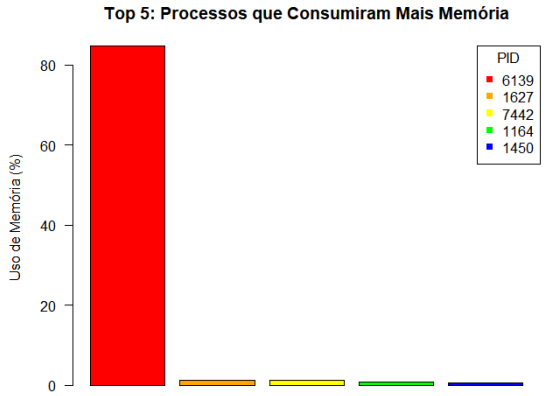

Figura 2. Top 5 dos processos que mais consumiram memória durante o experimento - Dell Vostro.

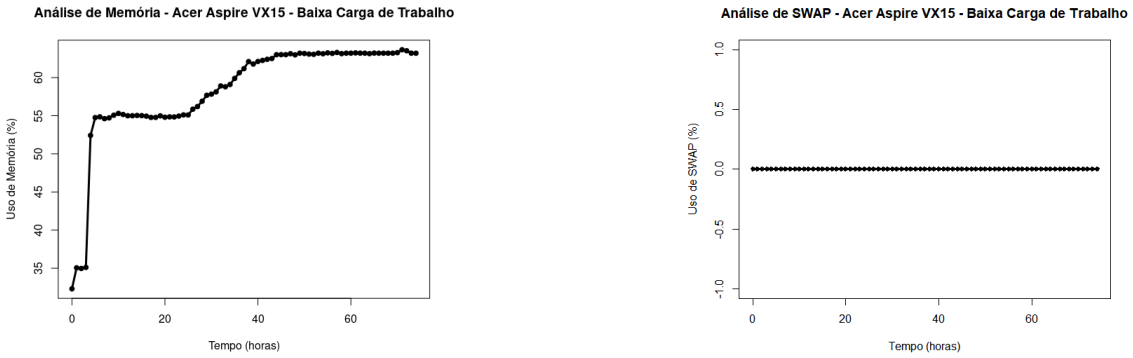

Figura 3. Utilização de Memória e *Swap* - Acer Aspire VX15 - Baixa Carga de Trabalho

5.2.2. Acer Aspire

A Figura 3 apresenta a análise de memória e *Swap* para o Acer Aspire, considerando a carga de trabalho baixa. É possível observar um grande aumento no consumo de memória a partir da quarta hora de execução do experimento. Antes disso, o uso de memória passou por um pequeno incremento na primeira hora de execução, seguido por duas horas de relativa estabilidade. Depois da quarta hora, o comportamento se repetiu por algum tempo: leve crecimento na quinta hora, seguido de um período de certa estabilidade, até passar a um consumo visivelmente crescente conforme o tempo avançava. Por outro lado, não houve variação para o uso de *Swap*, o qual se manteve em 0% do início ao fim do experimento.

A Tabela 5 apresenta os resultados do teste de tendência de Mann-Kendall para o Acer Aspire sob baixa carga de trabalho. Conforme mostram os resultados dos testes, o uso de memória possui uma tendência de degradação, visto que o *p-value* é menor do que 0,05, o que indica a presença de uma tendência. Já o valor de S é maior do que zero, indicando a tendência de crescimento. Por outro lado, o uso de *Swap* não possui tendência alguma, dado que o *p-value* é zero. A Figura 4 exibe os cinco processos que mais con-

Teste MK - Acer Aspire, Baixa Carga de Trabalho			
Métrica	**S**	***p-value***	**Tendência**
Memória	2,03e+03	2,22e-16	Crescimento
SWAP	0	0	Sem tendência

Tabela 5. Testes de Mann-Kendall para o Acer Aspire - Carga Baixa

Top 5: Processos que Mais Consumiram Memória - Acer Aspire, Baixa Carga de Trabalho	
PID	**Descrição**
4085 (cardano-node)	O processo do *cardano-node*.
1356 (cinnamon-replace)	Comando que inicia/reinicializa o Cinnamon, gerenciador de janelas.
862 (Xorg-core)	Um dos processos do Xorg, *software* de sistema e protocolo que fornece uma base para GUIs.
152020, 1787000 (cinnamon-screensaver)	O protetor de tela padrão em um desktop GNOME.

Tabela 6. Descrição dos processos que mais consumiram memória durante o experimento no Acer Aspire - baixa carga de trabalho

sumiram memória no Acer Aspire para a carga de trabalho baixa, enquanto a Tabela 6 apresenta a descrição de tais processos. Novamente, o *cardano-node* liderou com folga entre os processos que mais consumiram memória, utilizando em cerca de 8GB dos 16GB disponíveis no computador. Os demais processos fazem parte do sistema operacional (Linux Mint, neste caso) e consumiram pouca memória, mesmas características presentes na execução do experimento no Dell Vostro.

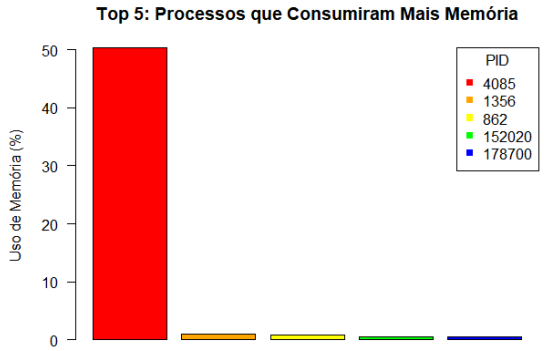

Figura 4. Top 5 dos processos que mais consumiram memória durante o experimento - Acer Aspire, baixa carga de trabalho.

A Figura 5 apresenta a análise de memória e *Swap* para o Acer Aspire, considerando a carga de trabalho média. É notável o rápido incremento no uso de memória durante este experimento, se compararmos com a execução do experimento anterior (ver Figura 3). Além de atingir um consumo médio de RAM superior ao final da primeira hora, cerca de 47,5% contra 35% no experimento sob baixa carga, o patamar de 60% de uso de memória foi atingido na décima hora, enquanto que, sob baixa carga de trabalho, isso só aconteceu pouco antes da quadragésima hora de execução. Para ambos, o uso de memória se encontrou em certa estabilidade pouco acima dos 60%. Porém, para o experimento da carga média, houve uma utilização um pouco acima de 65%, valor o qual não foi atingido durante a execução do teste de baixa carga. O consumo do *Swap*, apesar de não ser nulo,

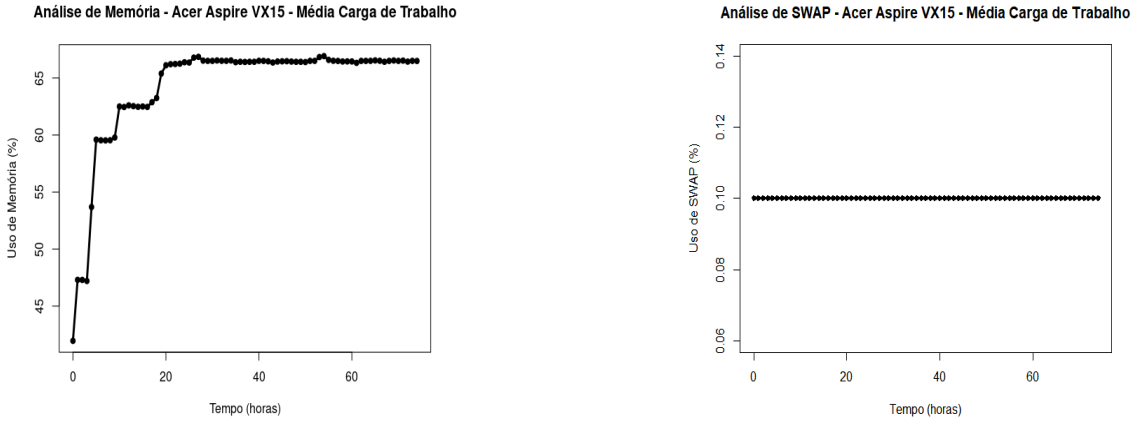

Figura 5. Utilização de Memória e *Swap* - Acer Aspire VX15 - Média Carga de Trabalho

Teste MK - Acer Aspire, Média Carga de Trabalho			
Métrica	**S**	***p-value***	**Tendência**
Memória	1,39e+03	2,015e-12	Crescimento
SWAP	0	0	Sem tendência

Tabela 7. Testes de Mann-Kendall para o Acer Aspire VX15 - Carga Média

não se alterou durante a execução do experimento, o que indica que este pequeno uso de 0,1% não é consequência da execução do *cardano-node*.

A Tabela 7 mostra os resultados do teste de tendência de Mann-Kendall para o Acer Aspire, considerando a carga de trabalho média. Assim como no experimento sob baixa carga de trabalho, existe uma tendência de degradação no uso de memória: o *p-value* é menor do que 0,05, o que indica que existe uma tendência nos dados, e o valor de S é positivo, indicando uma tendência de crescimento e evidenciando a ocorrência de degradação de memória. Para o uso de *Swap*, não existe tendência alguma, pois o *p-value* e o valor de S são iguais a zero. Analisando os cinco processos que mais consumiram memória no Acer Aspire com o experimento sob uma carga de trabalho média (ver Figuras 6 e Tabela 8), podemos concluir que o processo do *cardano-node* foi, novamente, o que mais consumiu memória RAM, e assim como no experimento de baixa carga, os demais processos com maior consumo fazem parte do sistema operacional, onde cada um deles consumiu uma quantidade muito pequena em comparação com o processo de execução do nó.

Top 5: Processos que Mais Consumiram Memória - Acer Aspire, Média Carga de Trabalho	
PID	**Descrição**
30658 (cardano-node)	O processo do *cardano-node*.
1412 (cinnamon-replace)	Comando que inicia/reinicializa o Cinnamon, gerenciador de janelas.
880 (Xorg-core)	Um dos processos do Xorg, *software* de sistema e protocolo que fornece uma base para GUIs.
280167, 478758 (mint-refresh-cache)	Um dos processos do Gerenciador de Atualizações do Linux Mint.

Tabela 8. Descrição dos processos que mais consumiram memória durante o experimento no Acer Aspire - média carga de trabalho

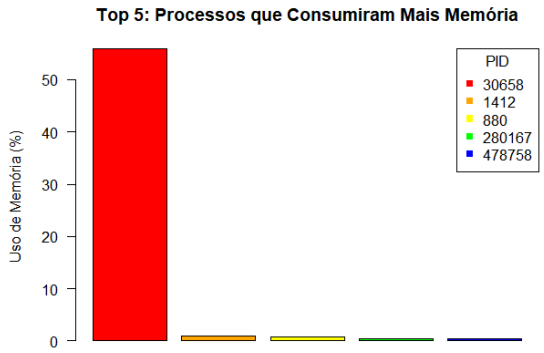

Figura 6. Top 5 dos processos que mais consumiram memória durante o experimento - Acer Aspire, média carga de trabalho.

Para o experimento no Acer Aspire sob alta carga de trabalho, cujos resultados da utilização de memória e *Swap* estão representados na Figura 7, o incremento no consumo de memória foi ainda mais rápido do que na execução do experimento sob carga de trabalho média, atingindo mais de 60% de uso antes do final da terceira hora de execução. Apesar do crescimento no uso de memória ter desacelerado acima dos 60%, ele continuou apresentando alguns períodos de incremento seguidos de períodos de relativa estabilidade, assim como a execução do experimento sob carga média, porém, atingindo valores mais altos, chegando perto de 70% de uso. Adicionalmente, para esse experimento, o uso de *Swap* exibiu uma pequena variação positiva, passando a crescer próximo da sexagésima hora de execução, chegando a um período de estabilidade antes de crescer novamente e alcançar estabilidade em um novo patamar. Porém, o consumo apresentado foi relativamente baixo, atingindo apenas 0,2% ao final do experimento.

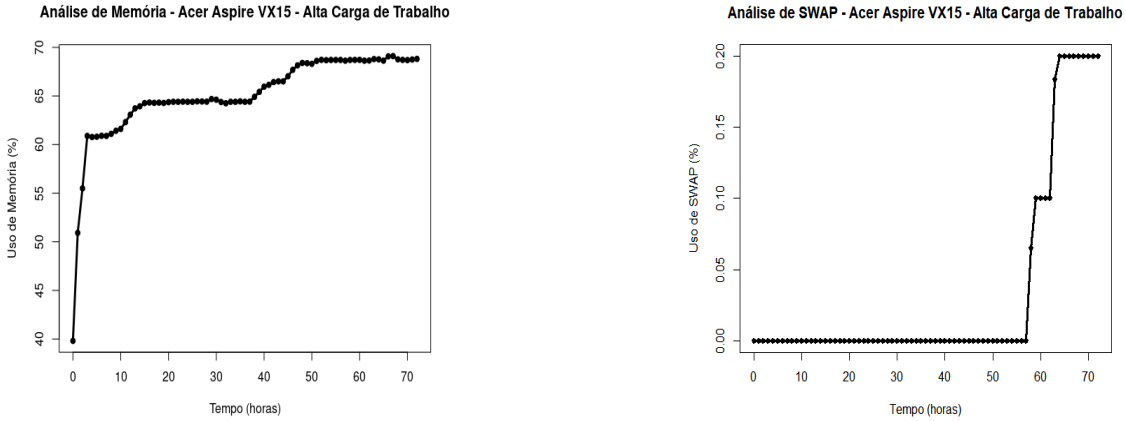

Figura 7. Consumo de Memória e *Swap* - Acer Aspire VX15 - Alta Carga de Trabalho

Os resultados do teste de tendência de Mann-Kendall para o Acer Aspire sob alta carga de trabalho, exibidos na Tabela 9, nos mostram uma tendência de degradação tanto para a memória quanto para o *Swap*.Em ambos os casos, o *p-value* foi menor do que 0,05,

indicando a existência de uma tendência, e o valor de S foi positivo, definindo a tendência como crescente.

Teste MK - Acer Aspire, Alta Carga de Trabalho			
Métrica	**S**	***p-value***	**Tendência**
Memória	2,14e+03	2,22e-16	Crescimento
SWAP	2,53e+06	2,22e-16	Crescimento

Tabela 9. Testes de Mann-Kendall para o Acer Aspire VX15 - Carga Alta

Por fim, a Figura 8 exibe os cinco processos que mais consumiram memória no Acer Aspire para a carga de trabalho alta, enquanto a Tabela 10 apresenta a descrição de tais processos. As características apresentadas nos experimentos anteriores se repetiram aqui. O processo *cardano-node* consumiu a maior parte da memória RAM, enquanto os demais processos consumiram pouca memória. Além disso, o processo do *cardano-node* atingiu o valor mais elevado do que havia atingido na execução do experimento sob cargas de trabalho mais leves.

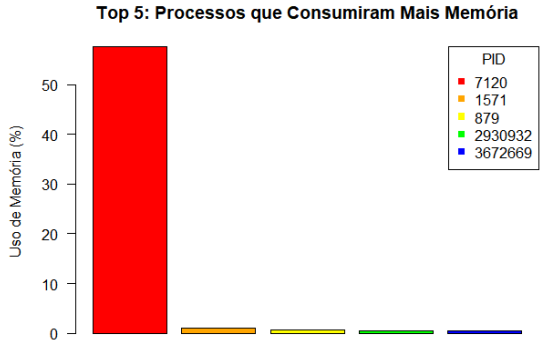

Figura 8. Processos em execução durante o experimento no Acer Aspire, alta carga de trabalho.

Top 5: Processos que Mais Consumiram Memória - Acer Aspire, Alta Carga de Trabalho	
PID	**Descrição**
7120 (cardano-node)	O processo do *cardano-node*.
1571 (cinnamon-replace)	Comando que inicia/reinicializa o Cinnamon, gerenciador de janelas.
879 (Xorg-core)	Um dos processos do Xorg, *software* de sistema e protocolo que fornece uma base para GUIs.
2930932, 3672669 (mint-refresh-cache)	Um dos processos do Gerenciador de Atualizações do Linux Mint.

Tabela 10. Descrição dos processos que mais consumiram memória durante o experimento no Dell Vostro

6. Conclusões

Este trabalho investigou a existência de envelhecimento de *software* na plataforma de *blockchain* Cardano através da análise da degradação de memória durante a execução da mesma sob vários ambientes e cargas de trabalho. Nós confirmamos estatisticamente indicativos de envelhecimento de *software*, acompanhado de tendências de degradação no uso de memória em todos os testes, além do aumento no uso do *Swap* em alguns casos. Os

resultados, acompanhados da análise de processos, são fortes indicativos que a plataforma Cardano possui de fato sintomas de envelhecimento de *software*, e que, dos processos da Cardano em execução, o próprio cardano-node é o responsável. Investigações mais profundas serão executadas para incluir outros indicados como tempo de resposta, consumo de energia, utilização de CPU, etc.

Referências

Andrade, E., Machida, F., Pietrantuono, R., and Cotroneo, D. (2021). Memory degradation analysis in private and public cloud environments. In *2021 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, pages 33–39. IEEE.

Andrade, E. C., Machida, F., Kim, D.-S., and Trivedi, K. S. (2011). Modeling and analyzing server system with rejuvenation through sysml and stochastic reward nets. In *2011 Sixth International Conference on Availability, Reliability and Security*, pages 161–168. IEEE.

Baiod, W., Light, J., and Mahanti, A. (2021). Blockchain technology and its applications across multiple domains: A survey. pages 78–119. Journal of International Technology and Information Management.

Cotroneo, D., Natella, R., Pietrantuono, R., and Russo, S. (2014). A survey of software aging and rejuvenation studies. *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, 10(1):1–34.

IOHK (2022). Cardano node releases. `https://github.com/input-output-hk/cardano-node/releases`. Accessed: 2022-04-18.

Jeffery, E. (2019). Blockchain beyond cryptocurrency. IBM.

Melo, C., Oliveira, F., Dantas, J., Araujo, J., Pereira, P., Maciel, R., and Maciel, P. (2022). Performance and availability evaluation of the blockchain platform hyperledger fabric. *The Journal of Supercomputing*, pages 1–23.

Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. Decentralized Business Review.

Parnas, D. (1994). Software aging. In *Proceedings of 16th International Conference on Software Engineering*, pages 279–287.

Pierro, M. D. (2017). What is the blockchain? pages 92–95. Computing in Science Engineering.

Pietrantuono, R. and Russo, S. (2020). A survey on software aging and rejuvenation in the cloud. *Software Quality Journal*, 28(1):7–38.

Trivedi, K. S., Grottke, M., and Andrade, E. (2010). Software fault mitigation and availability assurance techniques. *International Journal of System Assurance Engineering and Management*, 1(4):340–350.

Valentim, N. A., Macedo, A., and Matias, R. (2016). A systematic mapping review of the first 20 years of software aging and rejuvenation research. In *2016 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, pages 57–63. IEEE.