SISTEMAS OPERACIONAIS

AVALIAÇÃO DE DESEMPENHO

Apresentação:

Foi proposto, no segundo trabalho da disciplina de Sistemas Operacionais, o desenvolvimento de um código em linguagem C que cria processos no Linux utilizando a chamada **fork()** de duas formas: uma formando uma **árvore binária cheia** e outra, uma **cadeia linear**, ambas com altura definida via linha de comando. Cada processo criado deve exibir seu PID e PPID, além de uma mensagem ao finalizar. O programa também deve medir e comparar o tempo de criação das duas estruturas utilizando **clock_gettime()**, garantindo que os processos pais aguardem a finalização dos filhos para evitar a criação de processos zumbis.

Códigos utilizados:

Foram utilizados dois programas durante o desenvolvimento deste relatório.
primeiramente foi criado o programa proposto com o nome de **diogo_rocha_marques-t2-SO.c** que foi utilizado para gerar os processos de árvore binária e de cadeia coletando os PIDs e tempos de execução, este código citado seguem em anexo no envio da atividade, já para criar as tabelas e rodar o código anterior várias vezes com o intuito de gerar uma amostra mais satisfatória foi criado um código no shell do linux que rodaria o .c 20 vezes, o mesmo foi chamado de **executar_20_vezes.sh**, segue o código usado com comentários

Código Bash:

```
#!/bin/bash

# Nome do executável
EXECUTAVEL="./diogo_rocha_marques-t2-SO"

# Arquivos de saída
ARQUIVO_RESULTADOS="resultados.txt"
ARQUIVO_ANALISE="analise_resultados.txt"

# Cabeçalho dos arquivos
echo "Execução | Tempo Árvore (ms) | Tempo Cadeia (ms)" >
"$ARQUIVO_RESULTADOS"
echo "Análise dos Resultados" > "$ARQUIVO_ANALISE"
echo "======================" >> "$ARQUIVO_ANALISE"

# Loop de 20 execuções
for i in $(seq 1 20); do
        echo "Executando $i..."

        TEMPO_ARVORE=$($EXECUTAVEL 3 | grep "Tempo total de criação
da árvore" | awk '{print $7}')
        TEMPO_CADEIA=$($EXECUTAVEL 3 | grep "Tempo total de criação
da cadeia" | awk '{print $7}')

        if [[ -z "$TEMPO_ARVORE" || -z "$TEMPO_CADEIA" ]]; then
```

```
        echo "Erro ao obter tempos para a execução $i. Ignorando..."
        continue
        fi
        echo "$i | $TEMPO_ARVORE | $TEMPO_CADEIA" >>
"$ARQUIVO_RESULTADOS"
done
```

Análise dos Resultados:

Com base nas respostas obtidas durante as 20 execuções do programa, foram extraídas as seguintes estatísticas:

Média de tempo:

- **ÁRVORE:** 5,65 ms
- **CADEIA:** 1,59 ms

Desvio Padrão:

- **Árvore:** 1.26 ms

- **Cadeia:** 0.65 ms

Número de Execuções:

- **20 execuções válidas**

Tabela de execuções:

Execução:	Tempo Árvore:	Tempo Cadeia:
1	4.591	1.140
2	3.661	1.112
3	4.528	1.705
4	6.035	2.333
5	7.605	1.032
6	5.319	2.320

7	5.053	1.921
8	5.100	3.549
9	6.329	2.360
10	7.124	1.380
11	6.069	1.755
12	4.330	0.852
13	4.615	0.849
14	4.696	1.478
15	6.860	1.127
16	4.030	1.919
17	8.410	1.530
18	7.119	1.404
19	5.905	0.904
20	5.715	1.140

Tendências Observadas:

A maioria das execuções para a **estrutura em árvore** ficou entre **4,0 ms e 7,6 ms**, o que está relativamente próximo da média observada. Já para a **estrutura em cadeia**, os tempos variaram de **0,849 ms a 3,549 ms**, com a maioria das execuções ocorrendo abaixo de **2 ms**.

A **execução 17** apresentou o maior tempo para a árvore, com **8,410 ms**, enquanto a mais rápida foi a **execução 2**, com **3,661 ms**. Para a cadeia, o maior tempo foi registrado na **execução 8** (**3,549 ms**) e o menor na **execução 13**, com apenas **0,849 ms**.

Conclusões:

A partir dos testes realizados, foi possível observar diferenças significativas entre os dois métodos de criação de processos: em árvore binária cheia e em cadeia linear. Ambos demonstraram comportamentos distintos em relação à eficiência, estabilidade e exigência de recursos do sistema operacional.

A criação de processos em **cadeia** se destacou pela **eficiência**, apresentando tempos de execução consideravelmente menores. Em média, foi cerca de **3,5 vezes mais rápida** do que a criação em árvore. Além disso, demonstrou **maior estabilidade**, com baixa variação entre os tempos de execução nas 20 repetições realizadas, refletindo um comportamento mais previsível e consistente.

Por outro lado, a criação de processos em **árvore binária cheia** mostrou-se mais onerosa em termos de desempenho. A necessidade de bifurcações simultâneas — dois processos filhos a partir de cada processo pai — gera uma sobrecarga maior para o sistema operacional, tanto no controle quanto na sincronização desses processos. Isso

justifica os tempos de execução mais altos observados.

As **variações pontuais** nos tempos de criação, em ambos os métodos, são compreensíveis e esperadas. Elas decorrem de fatores como a carga momentânea do sistema e o funcionamento do escalonador de processos do sistema operacional, que podem afetar ligeiramente a performance em diferentes execuções.

Em resumo, o experimento evidenciou que a criação em cadeia é mais leve e rápida, sendo mais adequada quando se deseja minimizar o tempo e o custo de criação de processos. Já a criação em árvore, embora mais custosa, é útil em cenários onde a paralelização mais ampla de processos for necessária.