

Juliano Cesar Petini
Michel Gomes de Souza

Manipulação de threads Laboratório 03

Relatório técnico de atividade prática solicitado pelo professor Rodrigo Campiolo na disciplina de Sistemas Operacionais do Bacharelado em Ciência da Computação da Universidade Tecnológica Federal do Paraná.

Universidade Tecnológica Federal do Paraná – UTFPR
Departamento Acadêmico de Computação – DACOM
Bacharelado em Ciência da Computação – BCC

Campo Mourão
Junho / 2021

Resumo

O Presente trabalho tem como objetivo compreender o uso das threads no **Linux** e como podemos manipular e criar novos fluxos na linguagem de programação *C*.

Sumário

1	Introdução	4
2	Parte 1	4
2.1	Identifique no seu sistema Linux quantas threads estão em execução? Qual o processo com o maior número de threads?.	4
2.2	Qual o número máximo de threads que o seu sistema suporta?.	4
2.3	Verifique o tempo de execução do programa da questão 3, parte 2, considerando: 1 thread, 2 threads, 4threads, 8 threads e 16 threads.	5
3	Conclusões	5

1 Introdução

Nesse projeto iremos realizar vários testes no Linux a fim de compreender a fundo o que são *threads* e como manipulá-las, para isso utilizaremos a linguagem de programação *C* e uma maquina virtual. A Maquina utilizada para realizar os experimentos foi a seguinte:

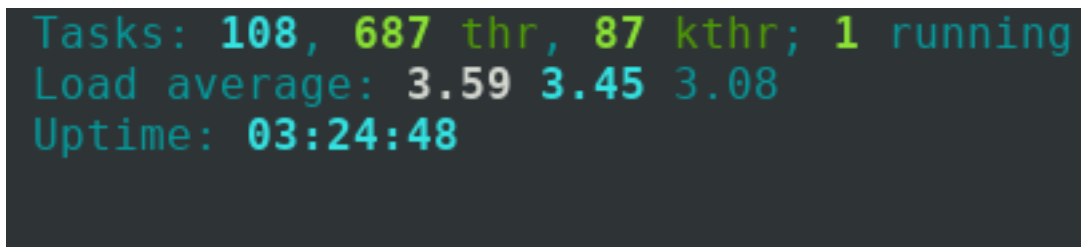
Arquitetura: x86_64
Modo(s) operacional da CPU: 32-bit, 64-bit
CPU(s): 2 Ncleos Fisicos 4 Threads
Nome do modelo: Intel(R) Core(TM) i3-3217U CPU @ 1.80GHz
Memoria: 8GB

2 Parte 1

Nessa seção iremos apresentar os comandos que foram realizados para contemplar os objetivos proposto pela parte 1 da atividade.

2.1 Identifique no seu sistema Linux quantas threads estão em execução? Qual o processo com o maior número de threads?.

Com o auxilio do `htop` conseguimos identificar que são aproximadamente 87 threads no modo kernel(núcleo) e 687 threads no modo usuário. Podemos ver esses dados na Figura 1.



```
Tasks: 108, 687 thr, 87 kthr; 1 running
Load average: 3.59 3.45 3.08
Uptime: 03:24:48
```

Figura 1 – Resultado obtido ao utilizar o `htop`.

Já utilizando o comando `ps -eLf`, conseguimos identificar que o processo com maior numero de threads foi *firefox* com um total de 85 de threads, como demonstrado na Figura 2.

2.2 Qual o número máximo de threads que o seu sistema suporta?.

Utilizando o comando `cat /proc/sys/kernel/threads-max`, conseguimos obter o valor de 61048, no qual representa o máximo de threads que o sistema suporta.

```

mallone 1364      1 1364 0      1 20:15 ?      00:00:00 /bin/sh -c firefox
mallone 1365 1364 1365 5      85 20:15 ?      00:04:12 /usr/lib/firefox/firefox
mallone 1365 1364 1374 0      85 20:15 ?      00:00:00 /usr/lib/firefox/firefox
mallone 1365 1364 1375 0      85 20:15 ?      00:00:00 /usr/lib/firefox/firefox
mallone 1365 1364 1376 0      85 20:15 ?      00:00:00 /usr/lib/firefox/firefox
mallone 1365 1364 1377 0      85 20:15 ?      00:00:00 /usr/lib/firefox/firefox
mallone 1365 1364 1378 1      85 20:15 ?      00:01:11 /usr/lib/firefox/firefox
mallone 1365 1364 1379 0      85 20:15 ?      00:00:01 /usr/lib/firefox/firefox
mallone 1365 1364 1380 0      85 20:15 ?      00:00:00 /usr/lib/firefox/firefox
mallone 1365 1364 1381 1      85 20:15 ?      00:01:25 /usr/lib/firefox/firefox
mallone 1365 1364 1382 0      85 20:15 ?      00:00:00 /usr/lib/firefox/firefox
mallone 1365 1364 1384 0      85 20:15 ?      00:00:00 /usr/lib/firefox/firefox
mallone 1365 1364 1385 0      85 20:15 ?      00:00:00 /usr/lib/firefox/firefox
mallone 1365 1364 1402 0      85 20:15 ?      00:00:00 /usr/lib/firefox/firefox
mallone 1365 1364 1403 0      85 20:15 ?      00:00:00 /usr/lib/firefox/firefox
mallone 1365 1364 1404 0      85 20:15 ?      00:00:00 /usr/lib/firefox/firefox
mallone 1365 1364 1405 0      85 20:15 ?      00:00:00 /usr/lib/firefox/firefox
mallone 1365 1364 1406 0      85 20:15 ?      00:00:00 /usr/lib/firefox/firefox
mallone 1365 1364 1408 0      85 20:15 ?      00:00:02 /usr/lib/firefox/firefox

```

Figura 2 – Resultado obtido ao utilizar o ps -eLf.

2.3 Verifique o tempo de execução do programa da questão 3, parte 2, considerando: 1 thread, 2 threads, 4threads, 8 threads e 16 threads.

Para executar essa atividade fizemos uso de uma matriz 100x100 gerado com valores randômicos e analisamos o tempo de execução em milissegundos com quantidades específicas de threads, lembrado que quanto menor for o tempo resultante melhor é o resultado e também o hardware utilizado para fazer estes teste esta descrito no início da seção.

Para começar o teste testamos 1x threads, conseguimos obter um resultado de **23/Ms**. Na sequencia utilizando 2x threads conseguimos obter o resultado de **25/Ms**. Para o próximo teste utilizamos 4x threads e já adianto que, foi o melhor resultado obtido com uma taxa de **21/Ms**. E Para finalizar o exercício fizemos o teste utilizando 8x e 16x threads e os resultados foram de **22/Ms** e de **31/Ms** respectivamente.

Relação de numero de threads com tempo de execução

Numero de Threads	Tempo em milissegundos.	Tamanho da Matriz.
1	23/Ms	100x100
2	25/Ms	100x100
4	21/Ms	100x100
8	22/Ms	100x100
16	31/Ms	100x100

3 Conclusões

Nesta atividade foi apresentado alguns comandos para visualizar as threads no sistema operacional e também foram criados algoritmos para fixar o conteúdo teórico

apresentado em aula. E de fato conhecer esse recurso é de vital importância para a carreira de um programador, a possibilidade de melhorar o paralelismo e o tempo para realizar algumas tarefas podem diminuir utilizando multitarefas, porém não necessariamente aumentar o numero de threads em uma aplicação vai fazer ela executar mais rapidamente.