**Nome:** Rodrigo Siveris Klein **Matrícula:** 2010100987  
Texto

Descrição gerada automaticamenteRelatório Técnico

1. **Identificação do processador**

Processador Intel Core i7 7700; 4 núcleos; 8 threads; tecnologia Hyper-Threading; Frequência base de 3.60GHz; TDP de 65 Watts.

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

1. **Registro das informações coletadas no processador ao executar a sequência de códigos apresentados no Tsuna's Blog**
   1. **Resultados do código cpubench.sh**

./cpubench.sh

model name : Intel(R) Core(TM) i7-7700 CPU @ 3.60GHz

1 physical CPUs, 4 cores/CPU, 2 hardware threads/core = 8 hw threads total

-- No CPU affinity --

10000000 system calls in 5074002096ns (507.4ns/syscall)

2000000 process context switches in 7518264555ns (3759.1ns/ctxsw)

2000000 thread context switches in 7239323909ns (3619.7ns/ctxsw)

2000000 thread context switches in 7218825982ns (3609.4ns/ctxsw)

-- With CPU affinity --

10000000 system calls in 5074429784ns (507.4ns/syscall)

2000000 process context switches in 4739219508ns (2369.6ns/ctxsw)

2000000 thread context switches in 4315853933ns (2157.9ns/ctxsw)

2000000 thread context switches in 4294561171ns (2147.3ns/ctxsw)

-- With CPU affinity to CPU 0 --

sed: can't read /sys/bus/node/devices/node0/cpumap: No such file or directory

taskset: bad usage

* 1. **Resultados do código timectxsw**

./timectxsw

2000000 process context switches in 7381308143ns (3690.7ns/ctxsw)

* 1. **Resultados do código timesyscall**

./timesyscall

10000000 system calls in 5081153555ns (508.1ns/syscall)

* 1. **Resultados do código timetctxsw**

./timetctxsw

2000000 thread context switches in 7864612262ns (3932.3ns/ctxsw)

* 1. **Resultados do código timetctxsw2**

./timetctxsw2

2000000 thread context switches in 6368631441ns (3184.3ns/ctxsw)

* 1. **Resultados do código timetctxswws.c**

./timetctxswws 1000

2000 memset on 4000K in 161782296ns (80.9ns/page)

8000 process context switches (wss:4000K) in 252407395ns (31550.9ns/ctxsw)

1. **Para cada resultado encontrado explique o que o programa que o originou faz e o que pretende medir (destaquem os conceitos vistos na disciplina que motivaram o autor);**
   1. **Resultados do código timectxsw**

O programa mede o tempo e a eficiência das trocas de contexto de processo no sistema operacional, simulando um cenário de troca de contexto, onde os processos estão cooperando e compartilhando dados através de uma região de memória compartilhada. O programa está medindo o número de trocas de contexto de processo realizadas durante a execução do loop, calculando o tempo total decorrido durante todas as trocas de contexto de processo e calculando a média do tempo por troca de contexto.

* 1. **Resultados do código timesyscall**

O programa mede o tempo e a eficiência das chamadas de sistema em um sistema operacional, fornecendo métricas sobre o número total de chamadas de sistema e o tempo médio gasto em cada chamada. Neste caso específico, ele executa 10 milhões de chamadas de sistema e relata o tempo total gasto e o tempo médio por chamada de sistema.

* 1. **Resultados do código timetctxsw**

O programa também mede a eficiência das trocas de contexto de thread no sistema operacional, fornecendo métricas sobre o número total de trocas de contexto e o tempo médio gasto em cada uma delas. Neste caso específico, ele executa 2 milhões de trocas de contexto de thread e relata o tempo total gasto e o tempo médio por troca de contexto.

* 1. **Resultados do código timetctxsw2**

O programa também está medindo a eficiência das trocas de contexto de thread, fornecendo métricas sobre o número total de trocas de contexto e o tempo médio gasto em cada uma delas. Neste caso específico, ele executa 2 milhões de trocas de contexto de thread e relata o tempo total gasto e o tempo médio por troca de contexto, com a política de agendamento de threads definida como FIFO.

* 1. **Resultados do código timetctxswws.c**

O programa também está medindo o número total de trocas de contexto de processo realizadas durante a execução, calculando o tempo total decorrido durante todas as trocas de contexto de processo e calculando a média do tempo por troca de contexto.

Passando 1000 como argumento, foram realizadas 2000 operações de memset em um working set de 4000 KB. O tempo total gasto nessas operações foi de 161782296 nanossegundos, resultando em uma média de aproximadamente 80.9 nanossegundos por página. Durante a execução do programa, foram realizadas 8000 trocas de contexto de processo.

1. **Para cada resultado, assim como o autor faz, registre a análise dos dados;**

**a. Resultados do código timectxsw**

O resultado indica o tempo médio necessário para uma troca de contexto de thread no sistema. Uma troca de contexto eficiente é crucial para a multitarefa em sistemas operacionais, e uma média de 3690.7 nanossegundos parece razoável.

**b. Resultados do código timesyscall**

Foram realizadas 10000000 chamadas de sistema e a média de tempo por chamada de sistema foi de 508.1 nanossegundos. Portanto, o resultado parece estar dentro do esperado para uma chamada de sistema em um ambiente típico.

**c. Resultados do código timetctxsw:**

Foram realizadas 2000000 trocas de contexto de thread. A média de tempo por troca de contexto foi de aproximadamente 3932.3 nanossegundos, o que parece razoável para este tipo de operação.

**d. Resultados do código timetctxsw2:**

Nesse caso o código também mede o tempo de execução das trocas de contexto de thread, mas com uma abordagem diferente de agendamento de threads. Foram realizadas 2000000 trocas de contexto de thread. A média de tempo por troca de contexto foi de aproximadamente 3184.3 nanossegundos. Podemos ver a eficiência das trocas de contexto de thread com uma política de agendamento FIFO. A média de 3184.3 nanossegundos é razoável e indica um bom desempenho.

**e. Resultados do código timetctxswws.c:**

Neste caso foram realizadas 8000 trocas de contexto de processo. A média de tempo por troca de contexto foi de aproximadamente 31550.9 nanossegundos. Essa média mostra como o tamanho do conjunto de trabalho pode afetar o desempenho do sistema durante as trocas de contexto.

1. **Gere um gráfico similar ao que consta no Tsuna's Blog relacionando: o tempo de troca de contexto X tamanho do Working Set X Tempo para escrever uma página. Explique o gráfico.**

Gráfico gerado “grafico.png” a partir do arquivo graficos.py.

