Humboldt University

Computer Science Department
Systems Architecture Group
http://sar.informatik.hu-berlin.de

Operating Systems Principles

Cachegrind

Valgrind - Memcheck

- Example: Matrix
  - inc_matrix: increases each element of a matrix
  - 2 Versions:
    - Row first
    - Col first

- Run

```
% valgrind --tool=cachegrind –cachegrind-out-file=cg.out –log-file=cg.log ./cachetest_x
```

- Output

```
==3496== I   refs:      600,313,026
==3496== I1  misses:            758
==3496== LLi misses:            752
==3496== I1  miss rate:        0.00%
==3496== LLi miss rate:        0.00%
==3496==
==3496== D   refs:      200,103,179  (100,080,175 rd   + 100,023,004 wr)
==3496== D1  misses:      6,262,125  (  6,261,926 rd   +         199 wr)
==3496== LLd misses:      6,251,817  (  6,251,627 rd   +         190 wr)
==3496== D1  miss rate:         3.1% (       6.2%     +         0.0%  )
==3496== LLd miss rate:         3.1% (       6.2%     +         0.0%  )
```


Systems Architecture Group
http://sar.informatik.hu-berlin.de

Cachegrind

- Detailed information:
  - Summary
  - Data-Cache-Refs
  - Data-Cache-Misses
  - Instructions-Cache-Refs
  - Instruction-Cache-Misses

```
% cat ./cachetest_x
```

```
==3496== I   refs:      600,313,026
==3496== I1  misses:            758
==3496== LLi misses:            752
==3496== I1  miss rate:        0.00%
==3496== LLi miss rate:        0.00%
==3496==
==3496== D   refs:      200,103,179  (100,080,175 rd   + 100,023,004 wr)
==3496== D1  misses:      6,262,125  (  6,261,926 rd   +         199 wr)
==3496== LLd misses:      6,251,817  (  6,251,627 rd   +         190 wr)
==3496== D1  miss rate:         3.1% (        6.2%     +         0.0%  )
==3496== LLd miss rate:         3.1% (        6.2%     +         0.0%  )
```


Systems Architecture Group
http://sar.informatik.hu-berlin.de

Cachegrind

- More information:
  - Information per line of code
  - Data-Cache-Refs
  - Data-Cache-Misses
  - Instructions-Cache-Refs
  - Instruction-Cache-Misses

```
% cg_annotate cg.out --auto=yes cachetest.c
```

mc [robert@jetli]:~/Programmieren/C/Projects/cachetest

```
        .   .   .          .          .          .          .   .   .  #define x_size SIZE
        .   .   .          .          .          .          .   .   .  #define y_size SIZE
        .   .   .          .          .          .          .   .   .  #endif
        .   .   .          .          .          .          .   .   .
        .   .   .          .          .          .          .   .   .    register int l;
        1   0   0          0          0          0          1   0   0    register int sum = 0;
        1   0   0          1          0          0          0   0   0    register int *m1 = m;
        .   .   .          .          .          .          .   .   .    register int *mat;
        .   .   .          .          .          .          .   .   .
        .   .   .          .          .          .          .   .   .  #ifdef X_FIRST
        .   .   .          .          .          .          .   .   .    for ( l = 0; l < y_size; l++, m1+=x_size) {
        .   .   .          .          .          .          .   .   .      mat = m1 + x_size-1;
        .   .   .          .          .          .          .   .   .      for ( ; mat >= m1; mat--) {
        .   .   .          .          .          .          .   .   .        (*mat)++;
        .   .   .          .          .          .          .   .   .        //sum += (*mat);
        .   .   .          .          .          .          .   .   .      }
        .   .   .          .          .          .          .   .   .    }
        .   .   .          .          .          .          .   .   .  #else
   40,004   1   1          0          0          0          0   0   0    for ( l = 0; l < x_size; l++, m1++) {
   10,000   0   0          0          0          0          0   0   0      mat = m1 + (x_size*y_size)-x_size;
300,030,000 0   0          0          0          0          0   0   0      for ( ;mat >= m1; mat -= x_size) {
300,000,000 0   0 100,000,000 100,000,000 6,259,999 100,000,000   0   0        (*mat)++;
        .   .   .          .          .          .          .   .   .        //sum += (*mat);
        .   .   .          .          .          .          .   .   .      }
```


Systems Architecture Group
http://sar.informatik.hu-berlin.de

Hyper-Threading

Systems Architecture Group
http://sar.informatik.hu-berlin.de

Hyper-Threading

- Hardwareseitiges Multithreading (Intel-Prozessoren)
- 2 Logische Prozessoren auf einer physikalischen CPU
- Jeder log. bearbeitet einen Thread
- Register-Satz ist doppelt vorhanden
- Andere Komponenten werden geteilt (L1-Cache,ALU,FPU,…)
- Zweiter Thread T_2 nutzt Komponenten, welche der erste Thread T_1 nicht benutzt, z.B.:
  - ALU, während T_1 Daten holt
  - ….

Humboldt Universität Informatik
Systems Architecture Group
http://sar.informatik.hu-berlin.de

Hyper-Threading

- Problem:
  - Performance stark von Threads abhängig (Was machen diese?)
  - Thread müssen verschiedene Komponenten der CPU nutzen
  - L1-Cache wird von beiden genutzt
    - Daten holen ermöglicht weiteren Thread zu arbeiten, aber
    - Threads „überschreiben“ sich Cache-Inhalte
  - Scheduler kann Auslastung der CPU (welche Komponenten) nicht erkennen → ungünstiges Scheduling
  - Programmierer gehen von mehreren Kernen aus → vergessen Abhängigkeiten (L1-Cache,…)

Unix-API

- Programmierer steuert zuordnung von Thread zu CPU
- API:
  - pthread_setaffinity_np
  - pthread_getaffinity_np
  - …
  - Weiter Funktionen ermöglichen die Ermittlung der Anzahl der CPUs/Kerne

Experiment

- ht_test.c
  - Mehrere Threads
  - Verschiedene Zuordnung von Threads zu Cores
  - Threads können unterschiedliche Tasks ausführen
    - Primzahl bestimmen (ALU)
    - Primzahl bestimmen (FPU)
    - Matrix inkrementieren
  - Indirekt: Task-zuordnung pro Core

- Testsystem
  - Core i5: 2 physikalische Kerne + HT = 4 logische Kerne

ht_test

- Optionen:
  - no_thread: Anzahl der Threads
  - no_loops: Anzahl der Durchläufe (Primzahl)
  - Mode:

Mode	Core 0	Core 1	Core 2	Core 3
0	Thread 0 & 1	Thread 2 & 3		
1	Thread 0 & 2	Thread 1 & 3		
2	Thread 0	Thread 1	Thread 2	Thread 3
3	Thread 0	Thread 2	Thread 1	Thread 3
4	?	?	?	?

  - Task-Mode

Mode	Thread 0	Thread 1	Thread 2	Thread 3
0	Task 0	Task 0	Task 0	Task 0
1	Task 1	Task 1	Task 1	Task 1
2	Task 0	Task 1	Task 0	Task 1
3	Task 0	Task 0	Task 1	Task 1
4	Task 2	Task 2	Task 2	Task 2
5	Task 0	Task 2	Task 0	Task 2

ht_test

Task	,CPU	Core 0	Core 1	Core 2	Core 3
0	2	TH0:**T0**	TH1 **T0**	TH2 **T0**	TH3 **T0**
0	3	TH0 **T0**	TH2 **T0**	TH1 **T0**	TH3 **T0**
0	4	?	?	?	?
2	2	TH0 **T0**	TH1 **T1**	TH2 **T0**	TH3 **T1**
2	3	TH0 **T0**	TH2 **T0**	TH1 **T1**	TH3 **T1**
2	4	?	?	?	?
3	2	TH0 **T0**	TH1 **T0**	TH2 **T1**	TH3 **T1**
3	3	TH0 **T0**	TH2 **T1**	TH1 **T0**	TH3 **T1**
3	4	?	?	?	?

T-Mode	Thread 0	Thread 1	Thread 2	Thread 3
0	Task 0	Task 0	Task 0	Task 0
1	Task 1	Task 1	Task 1	Task 1
2	Task 0	Task 1	Task 0	Task 1
3	Task 0	Task 0	Task 1	Task 1
4	Task 2	Task 2	Task 2	Task 2
5	Task 0	Task 2	Task 0	Task 2

Mode	Core 0	Core 1	Core 2	Core 3
0	Thread 0 & 1	Thread 2 & 3		
1	Thread 0 & 2	Thread 1 & 3		
2	Thread 0	Thread 1	Thread 2	Thread 3
3	Thread 0	Thread 2	Thread 1	Thread 3
4	?	?	?	?