CS 377P: Programming for Performance

Assignment 1: Performance counters

Due date: February 7, 2018, 9:00PM

You should do this assignment in teams of two.
Late submission policy: Submissions can be at most 1 day late. There will be a 10% penalty for late submissions.

Description

Write C code for the 6 variants of matrix-matrix multiply (MMM) you can generate by permuting loops in the standard three-nested loop version of MMM. The data type in the matrix should be doubles.

- Instrument your implementations to use PAPI to measure:
  - Total cycles
  - Total instructions
  - Total Load Store Instructions
  - Total Floating Point Instructions
  - L1 data cache accesses and misses
  - L2 data cache accesses and misses
- Compile your code in ICC with flags '-O3 -fp-model precise'.
- Collect these measurements for the following 8 matrix sizes: 50x50, 100x100, 200x200, 400x400, 800x800, 1200x1200, 1600x1600 and 2000x2000. To ensure that there is no interference, make sure that you are the only one running experiments on the machine, and do one measurement at a time.
- You can collect measurements on the following 10 CS machines (with your CS login): orcrist-20.cs.utexas.edu, orcrist-21, orcrist-22, orcrist-24, orcrist-25, orcrist-26, orcrist-27, orcrist-28, orcrist-29 and orcrist-30.
- Create a table in which the rows correspond to the loop-order variant (i-j-k, j-i-k, j-k-i, k-j-i, i-k-j, k-i-j) and the columns correspond to the matrix sizes, and fill in each position in the table with four values: L1 and L2 miss rate, total load and store instructions, and number of committed floating point instructions.You can create four separate tables if you prefer.
- Answer the following questions.
  - For the smallest matrix size, do the L1 and L2 miss rates vary for the different loop-order variants? Do they vary for the larger matrix sizes? Is there any difference in behavior between the different problem sizes? Can you explain intuitively the reasons for this behavior?
  - Re-instrument your code by removing PAPI calls, and using clock_gettime with CLOCK_THREAD_CPUTIME_ID to measure the execution times for the six versions of MMM and the 8 matrix sizes specified above. How do your timing measurements compare to the execution times you obtained from using PAPI? Repeat this study using CLOCK_REALTIME. Explain your results briefly.

Hint: To check cache sizes on the machine, run: lscpu

Deliverables

Submit (in canvas) the following two files:

- A .tar.gz file with your code, a README.txt and a Makefile.
  - The README.txt describes how to run your program and what the output will be. A reasonable output will be pairs of "name of measured event, value".
  - With the Makefile, your code should be compiled on the 10 CS machines by running only "make".
- A report (in .pdf) containing the tables, and the answers to the questions. Clearly list both teammates' names in the report.

Grading

- Code: 40 points
- Measurements (plots): 40 points
- Explanation: 20 points

PAPI:

To see which papi counters are available on a host, run:

```
papi_avail
```

To see which papi counters can be collected at the same time, run:

```
papi_event_chooser
```

Read the PAPI manual http://icl.cs.utk.edu/projects/papi/wiki/PAPIC:EventSets for more information, including example code.

"Warning! num_cntrs is more than num_mpx_cntrs" can be ignored.

ICC:

To run ICC on the indicated CS machines, run:

```
export PATH=$PATH:/opt/intel/bin
icc [compiler commands]
```

To check the availability of icc, run:

```
icc -v
```