# Indian Institute of Information Technology, Vadodara

## Parallel Programming(cs403)

Prof. Reshmi Mitra

Name - Dilip Puri ID - 201351014 Collaborator - Hemant Kumar(201352026)

Lab 01

Submission Date - August 13, 2016

Deadline - Due Aug 15, 4:00 PM

## Peak Performance

Problem 1 Consider a memory system with a level 1 cache of 32 KB and DRAM of 512 MB with the processor operating at 1 GHz. The latency to L1 cache is 1 cycle and the latency to DRAM is 100 cycles. In each memory cycle, the processor fetches four words (cache line size is 4 words). What is the peak achievable performance of a dot product of two vectors?

```
/* dot product loop */
for (i = 0; i < dim; i++)
   dot_prod += a[i] * b[i];</pre>
```

#### Answer:

```
/* dot product loop */
for (i = 0; i < dim; i++)
  dot_prod += a[i] * b[i];</pre>
```

Problem 2 Now consider the problem of multiplying a dense matrix with a vector using a two-loop dot-product formulation. The matrix is of dimension 4K x 4K. (Each row of the matrix takes 16 KB of storage.) What is the peak achievable performance of this technique using a two-loop dot-product based matrix-vector product?

```
/* matrix-vector product loop */
for (i = 0; i < dim; i++)
  for (j = 0; j < dim; j++)
    c[i] += a[i][j] * b[i];</pre>
```

#### Answer:

```
/* dot product loop */
adf
adf
```

### Linux commands

\$top

| top - 02:47:51 up 12:02, 2 users, load average: 0.41, 0.43, 0.44                                  |       |    |      |         |       |       |   |      |      |                      |           |
|---|-------|----|------|---------|-------|-------|---|------|------|----------------------|-----------|
| Tasks: 187 total, 2 running, 185 sleeping, 0 stopped, 0 zombie                                    |       |    |      |         |       |       |   |      |      |                      |           |
| %Cpu(s  |       |    |      |         |       |       |   |      |      | hi, <b>0.0</b> si, ( | 0.0 st    |
| KiB Me  |       |    |      |         |       |       |   |      |      | 23852 buffers        |           |
| KiB Swap: <b>1547260</b> total, <b>240476</b> used, <b>1306784</b> free. <b>303060</b> cached Mem |       |    |      |         |       |       |   |      |      |                      | em        |
|   |       |    |      |         |       |       |   |      |      |                      |           |
|   | USER  | PR | NI   | VIRT    | RES   |       |   | %CPU |      | TIME+ COMMA          |           |
|   | dilip |    |      | 1552804 |       | 22560 |   |      | 10.2 | 11:11.04 compi       | Z         |
|   | root  | 20 | 0    |         | 59880 | 25448 |   | 6.0  | 4.0  | 18:38.43 Xorg        |           |
|   | dilip |    | 0    | 631520  | 16880 | 12040 |   | 2.3  |      | 0:00.92 gnome        |           |
|   | dilip |    | 0    | 1465000 |       | 41320 |   |      | 30.4 | 28:09.40 firef       |           |
|   | root  | 20 | 0    | 0       | 0     | 0     |   | 0.3  | 0.0  | 0:29.07 rcuos        |           |
|   | root  | 20 | 0    |         | 3676  | 2248  |   | 0.3  | 0.2  | 0:01.61 Netwo        |           |
|   | mysql | 20 | 0    | 550096  | 8144  | 804   |   | 0.3  | 0.5  | 0:19.51 mysql        |           |
| 6716  |       | 20 | 0    | Θ       | 0     | 0     |   | 0.3  | 0.0  | 0:09.86 kwork        |           |
| 7584  | dilip | 20 | 0    | 651572  | 19280 | 12760 | S | 0.3  | 1.3  | 0:00.81 gnome        | -terminal |
| 1   | root  | 20 | 0    | 33756   | 2108  | 836   | S | 0.0  | 0.1  | 0:02.23 init         |           |
| _   | root  | 20 | 0    | 0       | 0     | 0     |   | 0.0  | 0.0  | 0:00.02 kthre        |           |
| _   | root  | 20 | 0    | 0       | 0     | 0     |   | 0.0  | 0.0  | 0:00.43 ksoft        |           |
| 5   | root  | 0  | - 20 | 0       | 0     | 0     |   | 0.0  | 0.0  | 0:00.00 kwork        | er/0:0H   |
| 7   | root  | 20 | 0    | 0       | 0     | 0     |   | 0.0  | 0.0  | 0:38.26 rcu_s        | ched      |
| 9   | root  | 20 | 0    | 0       | 0     | 0     |   | 0.0  | 0.0  | 0:28.37 rcuos        | /1        |
| 10  | root  | 20 | 0    | Θ       | 0     | 0     | S | 0.0  | 0.0  | 0:00.00 rcuos        | /2        |
|   | root  | 20 | 0    | 0       | 0     | 0     |   | 0.0  | 0.0  | 0:00.00 rcuos        | /3        |
| 12  | root  | 20 | 0    | 0       | 0     | 0     | S | 0.0  | 0.0  | 0:00.00 rcu_b        | h         |
| 13  | root  | 20 | 0    | 0       | 0     | 0     |   | 0.0  | 0.0  | 0:00.00 rcuob        | /0        |
| 14  | root  | 20 | 0    | Θ       | 0     | 0     |   | 0.0  | 0.0  | 0:00.00 rcuob        | /1        |
| 15  | root  | 20 | 0    | 0       | 0     | 0     | S | 0.0  | 0.0  | 0:00.00 rcuob        | /2        |
| 16  | root  | 20 | 0    | 0       | 0     | 0     | S | 0.0  | 0.0  | 0:00.00 rcuob        |           |
| 17  | root  | rt | 0    | 0       | 0     | 0     | S | 0.0  | 0.0  | 0:00.56 migra        |           |
| 18  | root  | rt | 0    | 0       | 0     | 0     | S | 0.0  | 0.0  | 0:00.17 watch        |           |
| 19  | root  | rt | 0    | 0       | 0     | 0     | S | 0.0  | 0.0  | 0:00.15 watch        | dog/1     |
|   |       |    |      |         |       |       |   |      |      |                      |           |

Figure 1: top: provides dynamic real-time view of individual jobs running on the system

\$gnome-system-monitor

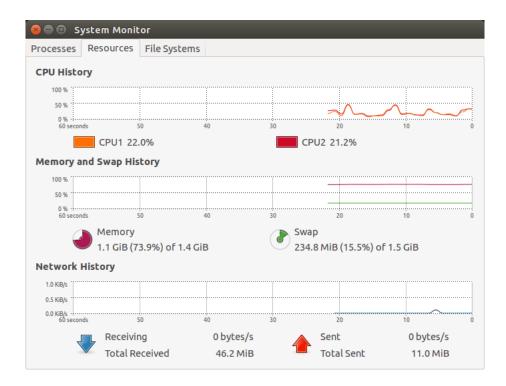


Figure 2: gnome-system-monitor: shows which programs are running and how much processor time, memory, and disk space are being used. This gives an overall system view whereas the "top" instruction represents a detailed perspective

\$lscpu
We can also get the same information from \$cat /proc/cpuinfo

Semester - 7 2 Parallel Programming

```
Architecture:
                     x86_64
                     32-bit, 64-bit
CPU op-mode(s):
Byte Order:
                     Little Endian
CPU(s):
On-line CPU(s) list: 0,1
Thread(s) per core: 1
Core(s) per socket:
Socket(s):
NUMA node(s):
Vendor ID:
                     AuthenticAMD
CPU family:
                     18
Model:
Stepping:
CPU MHz:
                     800.000
BogoMIPS:
                     4392.08
Virtualization:
                     AMD-V
L1d cache:
                     64K
L1i cache:
                     64K
L2 cache:
                     1024K
NUMA nodeO CPU(s): 0,1
dilip@dilip-notebook-pc:~$ lscpu
Architecture:
                     x86_64
CPU op-mode(s):
                     32-bit, 64-bit
Byte Order:
                     Little Endian
CPU(s):
On-line CPU(s) list: 0,1
Thread(s) per core: 1
Core(s) per socket: 2
Socket(s):
NUMA node(s):
Vendor ID:
                     AuthenticAMD
                     18
CPU family:
Model:
                     1
Stepping:
CPU MHz:
                     800.000
BogoMIPS:
                     4392.08
Virtualization:
                     AMD-V
L1d cache:
                     64K
L1i cache:
                     64K
L2 cache:
                     1024K
NUMA nodeO CPU(s):
                     0,1
```

```
$time ./a.out
```

```
dilip@dilip-notebook-pc:~$ time
real 0m0.000s
user 0m0.000s
sys 0m0.000s
```

Figure 3: time: Get total program execution time in the shell

### Lab Problems

- 1. Familiarize yourself with the Linux commands and POSIX thread code given in this handout.
- 2. Solutions for Problems 1-2 on peak performance.
- 3. Using the basic Linux commands find the cache size, bandwidth number of processors on your system.
- 4. Write a C-code using POSIX threads to create an unbalanced load using sleep command and hello.c. The sample sleep times are given below:
  - (a) thread-1: 1000 sec, thread-2: 5000 sec, thread-3: 20 sec, thread-4: 1200 sec.
  - (b) Measure the total time taken for the complete execution of code with and without the additional sleep command.
    - Hint: You will require to include unistd.h for successful compilation.
- 5. Write a C-code using POSIX threads about matrix multiplication.
  - (a) Take overall execution time measurement using time command for different application size and thread count for the serial and parallel code.
  - (b) Observe gnome-system-monitor output as your fire up different thread counts.
  - (c) Use the overall execution time measurements to plot and comment upon the speed-up.