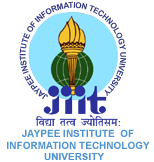
**OS MINOR PROJECT REPORT**

**GRID COMPUTING**



SUBMITTED BY:

NKITA JAIN(09104735)

ANUSHREE PAI (09104737)

YATHARTH SHARMA(09104743)

**Preface to First Phase**

In the first phase we showed how distributed computing works. We created a sort algorithm that can do parallel computing on different number of processes on type same computer and different processes on different computers.

We connect 2-3 computers and parallel computed the sorting algorithm. The pre-requisite for this was that all computers should be connected via LAN. The library used is- MPI (Message Passing Interface)

**Improvement over first phase**

**Problem:**

Error in observation when 8 processes were used on the same computer, the time did not reduce further, instead it increased.

**Solution:**

The printf() statements were removed. They were causing I/O delay resulting in increased time.

**Result:**

The time reduced as we kept on increasing the processes on the same computer as well as on different computers as should have been the desired output.

**WORK DONE IN SECOND PHASE**

A scheduler has been designed that performs scheduling as well as load balancing at the core level.

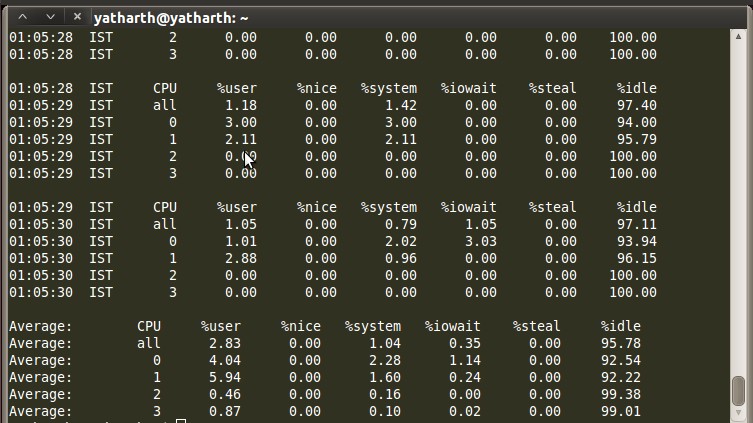
It was observed in the last phase that when the code was made to run on 1 process, it used only one core in spite of the fact that the other cores were ideal. This point has been taken into consideration in this phase.

The programs to be executed have been categorized on the basis of memory intensive and priority. The following priority order has been decided as per our algorithm used in designing the scheduler. (Designed as per i3 processor i.e. 4 cores)

1. High priority low memory intensive -2 cores
2. High priority high memory intensive - 4 cores
3. Low priority low memory intensive- 1 core
4. Low priority high memory intensive- 3 cores

The priority can be random, but here in our project we have set the priority to illustrate the code.

A terminal command called ‘sar’ is used to look at the status of the cores.



**WHAT OUR CODE DOES**

We have taken 5 programs in our code which are memory intensive and non-memory intensive and have explicitly set their priorities.

Whenever a program arrives from the ready queue, it checks for the available number of cores (i.e. number of ideal cores). If the amount of cores needed by the program is less than or equal to the amount of ideal cores, the process is made to run on it.

Otherwise the process gets stored in the waiting queue.

When next process arrives from the ready queue, it first checks the waiting queue. If there is a process having higher priority and needs core that is available at that time, the process in the waiting queue is executed first. This checking continues for all the processes in the waiting queue.

**OBSERVATION**

The following observations have been made keeping the priorities as follows:

Process 1=3;

Process 2=4;

Process 3=1;

Process 4=2;

Process 5 =0;

The higher the number the higher is the priority.

1. When all the 5 processes are not memory intensive

Nikita2= 1,00,000

Anuf1= 1,00,000

|  |  |  |
| --- | --- | --- |
| OBSERVATION # | OS | OUR |
| 1 | 109 | 77 |
| 2 | 108 | 75 |
| 3 | 107 | 74 |
| 4 | 110 | 76 |
| 5 | 114 | 75 |

AVERAGE :

When all the 2 processes are not memory intensive and 3 are memory intensive

Nikita2=3,00,000

Anuf1=1,00,000

|  |  |  |
| --- | --- | --- |
| OBSERVATION # | OS | OUR |
| 1 | 550 | 205 |
| 2 | 540 | 185 |
| 3 | 556 | 187 |
| 4 | 548 | 194 |
| 5 | 542 | 190 |

AVERAGE :

1. When all the 5processses are memory intensive

Nikita2=3,00,000

Anuf1=3,00,000

|  |  |  |
| --- | --- | --- |
| OBSERVATION # | OS | OUR |
| 1 | 870 | 315 |
| 2 | 869 | 313 |
| 3 | 823 | 316 |
| 4 | 853 | 322 |
| 5 | 865 | 325 |

AVERAGE :

**CONCLUSION**

We successfully distributed the processes into various cores depending on the usage of the cores. The processes were divided based on their priority and memory intensity. It means no core will be ideal unless all the processes in the ready queue have been successfully processed.

**References**

1.http://www.google.co.in/#sclient=psy&hl=en&rlz=1W1ADFA\_en&source=hp&q=how%20to%20schedule%20processes%20of%20cores%20in%20CPU&rlz=1W1ADFA\_en&pbx=1&oq=&aq=&aqi=&aql=&gs\_sm=&gs\_upl=&bav=on.2,or.r\_gc.r\_pw.&fp=fb14f34d6fd262fe&biw=1366&bih=526&pf=p&pdl=3000

2. http://www.mattheaton.com/?p=222

3. http://en.wikipedia.org/wiki/Processor\_affinity

4. http://www.cyberciti.biz/tips/setting-processor-affinity-certain-task-or-process.html

5. http://www.cyberciti.biz/tips/how-do-i-find-out-linux-cpu-utilization.html

6. http://en.wikipedia.org/wiki/Multi-core\_processor

7.http://www.google.co.in/#hl=en&cp=48&gs\_id=42&xhr=t&q=display+performance+of+each+core+in+CPu+in+linux&pf=p&sclient=psy&rlz=1R2ADFA\_enIN442&source=hp&pbx=1&oq=display+performance+of+each+core+in+CPu+in+linux&aq=f&aqi=&aql=&gs\_sm=&gs\_upl=&bav=on.2,or.r\_gc.r\_pw.&fp=fb14f34d6fd262fe&biw=1366&bih=526

8. http://stackoverflow.com/questions/3342889/how-to-measure-separate-cpu-core-usage-for-a-process

9. <http://www.thegeekstuff.com/2011/03/sar-examples/>

10. http://www.google.co.in/#sclient=psy&hl=en&rlz=1R2ADFA\_enIN442&source=hp&q=manage+core+processes+linux+using+C&rlz=1R2ADFA\_enIN442&pbx=1&oq=manage+core+processes+linux+using+C&aq=f&aqi=&aql=&gs\_sm=e&gs\_upl=25725l29203l4l30233l15l14l6l0l0l0l327l2244l27.1l8l0&bav=on.2,or.r\_gc.r\_pw.&fp=fb14f34d6fd262fe&biw=1366&bih=526

11. <http://stackoverflow.com/questions/663958/how-to-control-which-core-a-process-runs-on>

12. <http://tldp.org/LDP/lki/lki-2.html>

13. <http://www.ibm.com/developerworks/linux/library/l-affinity.html>

14. <http://www.scribd.com/doc/4838281/Operating-System-Scheduling-on-multicore-architectures>

15.http://www.google.co.in/#hl=en&cp=28&gs\_id=2m&xhr=t&q=how+to+balance+load+in+cores&pf=p&sclient=psy&r

**ACKNOWLEDGEMENT**

We owe a great many thanks to a great many people who helped and supported us during the project work we undertook.

Our deepest thanks to our teachers, Ms. Chetna Dabbas and Mr. Abhishek Swaroop for guiding and correcting various issues of project with attention and care. They have taken pain to go through the project and make necessary correction as and when needed.