# CHAPTER 6

# RESULTS AND DISCUSSIONS

## 6.1 Experimental Results

As explained in the preceeding chapter, experiments were conducted in two phases:

1. The initial experiment, in order to find out the effect of weight ratio of host domain to guest domains for the performance of guest domain for different types of workload.
2. Experiment with new scheduler algorithm, to test the improvement in PCPU share of guest domains according to dynamic workloads generated.

### 6.1.1 The Initial Experiment

The initial experiment, as described in the methodology section, was done in order to experiment the performance of mainly two schedulers SEDF scheduler and Credit scheduler in the xenserver. The experimental setup for the purpose is explained in the preceding section. In this section, the results of the experiments are obtained and discussed.

The experiment was conducted for mainly 3 types of workload, web server throughput using httperf, network throughput using iperf and disk throughput using dd utility. Tables 6.1 to 6.6 show the results obtained from these experiments.

The following parameters were altered during the experiment:

1. Types of workload
2. CPU scheduler for each workload
3. Weight ratio of host domain to guest domain

As the result of these variations, the throughput obtained was recorded and tabulated as below for mainly two modes: Work conserving mode and Non work conserving mode.

**Work Conserving Mode:**

In this mode, the CPU is always busy unless there is no any runnable task going. For e.g., when two clients with equal weight are running and a situation occurs when one of the client gets blocked, the other client can consume the entire CPU. The work conserving mode in SEDF algorithm is implemented by setting the flag as 1 in the third parameter. In Credit Scheduler, the work conserving mode is set by setting the cap as 0, which tells the scheduler that there is no limit to the cap value.

The results obtained for the WC mode for 3 different types of workloads are given below:

| | | | | | | |
|---|---|---|---|---|---|---|
| Dom0 weight relative to DomU weight | 0.5 | 1 | 1.5 | 2 | 2.5 | 3.0 |
| Throughput for SEDF (Req/Sec) | 835 | 712 | 624 | 572 | 530 | 512 |
| Throughput for Credit(Req/Sec) | 886 | 510 | 486 | 472 | 460 | 456 |

Table 6.1 Scheduler performance for web server workload

| | | | | | | |
|---|---|---|---|---|---|---|
| Dom0 weight relative to DomU weight | 0.5 | 1 | 1.5 | 2 | 2.5 | 3.0 |
| Throughput for SEDF (Mbits/Sec) | 250 | 255 | 240 | 235 | 236 | 234 |
| Throughput for Credit(Mbits/Sec) | 220 | 215 | 209 | 201 | 196 | 188 |

Table 6.2 Scheduler performance for iperf

| | | | | | | |
|---|---|---|---|---|---|---|
| Dom0 weight relative to DomU weight | 0.5 | 1 | 1.5 | 2 | 2.5 | 3.0 |
| Throughput for SEDF (MB/Sec) | 43.2 | 42.4 | 44.5 | 46.7 | 46.5 | 46.6 |
| Throughput for Credit (MB/Sec) | 42.1 | 43.1 | 44.6 | 46.6 | 48.2 | 48.1 |

Table 6.3 Scheduler performance for disk throughput

The discussions regarding the above experimental observations are made in the next section.

**Non-Work Conserving Mode:**

In this mode, the CPU is shared by clients with caps. This means each client will own a fraction of the CPU whatever the situation is. For e.g., in a case of two clients with equal weights, each client will get up to 50% even if the rest of the CPU is idle. In SEDF, it is maintained by setting the flag as 0 in the third parameter. In case of Credit scheduler, the cap value is given. In my experiment, I have set the cap for the guest domains only, whereas the host domain, namely Dom0 is uncapped. The cap is set to 50 for the guest domain, so that it can only get the 50% share of the CPU at maximum.

The results for the NWC mode for 3 types of workloads as obtained are tabulated below:

| | | | | | | |
|---|---|---|---|---|---|---|
| Dom0 weight relative to DomU weight | 0.5 | 1 | 1.5 | 2 | 2.5 | 3.0 |
| Throughput for SEDF (Req/Sec) | 812 | 750 | 526 | 386 | 359 | 304 |
| Throughput for Credit (Req/Sec) | 604 | 460 | 398 | 316 | 340 | 362 |

Table 6.4 Scheduler performance for web server workload

| | | | | | | |
|---|---|---|---|---|---|---|
| Dom0 weight relative to DomU weight | 0.5 | 1 | 1.5 | 2 | 2.5 | 3.0 |
| Throughput for SEDF (Mbits/Sec) | 165 | 222 | 203 | 186 | 176 | 162 |
| Throughput for Credit (Mbits/Sec) | 98 | 154 | 162 | 168 | 148 | 133 |

Table 6.5 Scheduler performance for iperf

| | | | | | | |
|---|---|---|---|---|---|---|
| Dom0 weight relative to DomU weight | 0.5 | 1 | 1.5 | 2 | 2.5 | 3.0 |
| Throughput for SEDF (MB/Sec) | 28.2 | 37.4 | 42.8 | 44.6 | 39.3 | 36.9 |
| Throughput for Credit (MB/Sec) | 27.9 | 37.1 | 38.2 | 34.6 | 32.8 | 29.2 |

Table 6.6 Scheduler performance for disk throughput

### 6.1.2 Discussions on the results of Initial Experiments:

Tables 6.1 to 6.6 show the performance of the three workloads for the two schedulers in the WC and NWC modes respectively. From the analysis of these results from the experiments, we can summarize below:

• I/O intensive applications are highly sensitive to the amount of CPU allocated to *Dom0*. The problem of adequate CPU allocation to $Dom_0$ and efficient CPU scheduling becomes even harder when multiple VMs with diverse set of applications are competing for I/O processing in $Dom_0$;

• Application performance varies significantly under different schedulers even when the schedulers are configured with the same CPU allocation shares;

• Application performance is significantly worse under NWC mode when compared to WC-mode (when using similar shares). NWC-mode is an operational requirement for workload managers (it is used to support performance isolation and to deliver resource guarantees between applications). Optimizing CPU schedulers to support a more efficient CPU allocation under NWC-mode is an often overlooked problem.

Thus, the choice of the CPU scheduler and its configuration can significantly impact application performance despite supporting similar resource allocation models. In an environment where different servers may potentially run different CPU schedulers with varying configurations, the job of the workload manager becomes even more complex: migrating a VM to a different node with more resources does not necessarily result in better application performance. Hence, one interesting open question is whether virtualization environments must employ a single CPU scheduler with fixed parameters to successfully manage heterogeneous workloads.

### 6.1.3 Experiments with the modified credit scheduler algorithm:

From the initial experiment, the weight parameter of the domains was found as a significant factor in the performance of virtual machines. Virtual machines with a higher weight value will get more share of PCPUs with them, so it is obvious that VMs with more workload should get more weight value than the VMs with lesser workload. Keeping this in mind, the credit scheduler was modified in order to update the weights of the running VMs dynamically based upon the workload they have. The modified algorithm is shown in the previous chapter, and experiments were carried out in order to test if the results would come as expected or not.

For the experiment, two kinds of scenarios for each original credit scheduler and dynamic weight update based credit scheduler were tested. In the first scenario, the domain 1, domain 2 and domain 3 guests were given a workload ratio of 20%, 60% and 80% of the CPU they were allocated in the virtual machine. In the second scenario, the domain 1, domain 2 and domain 3 guests were given a workload ratio of 10%, 15% and 90% of the CPU they were allocated. The results were taken for a period of 120 seconds, with 10 second being the sampling time interval. The xen tools, xentop and xenmon[21], were used to collect the data. The results of the experiments carried out are tabulated in the Tables 6.7 to 6.10.

| Time Duration(Sec) | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 | 110 | 120 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PCPU share for Domain 1 | 10 | 18 | 25 | 30 | 31 | 32 | 31 | 30 | 32 | 31 | 32 | 31 |
| PCPU share for Domain 2 | 40 | 36 | 33 | 32 | 31 | 32 | 33 | 33 | 31 | 31 | 32 | 32 |
| PCPU share for Domain 3 | 45 | 41 | 36 | 34 | 33 | 32 | 33 | 33 | 32 | 33 | 31 | 32 |

**Table 6.7: PCPU share for original credit scheduler with 20%, 60% and 80% workloads for Dom1, Dom2, Dom3 respectively**

| Time Duration(Sec) | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 | 110 | 120 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PCPU share for Domain 1 | 8 | 12 | 14 | 13 | 13 | 12 | 11 | 12 | 13 | 12 | 11 | 12 |
| PCPU share for Domain 2 | 24 | 33 | 39 | 37 | 36 | 35 | 37 | 38 | 36 | 37 | 37 | 36 |
| PCPU share for Domain 3 | 50 | 55 | 53 | 52 | 50 | 48 | 47 | 48 | 46 | 48 | 47 | 48 |

**Table 6.8: PCPU share for modified credit scheduler with 20%, 60% and 80% workloads for Dom1, Dom2, Dom3 respectively**

| Time Duration(Sec) | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 | 110 | 120 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PCPU share for Domain 1 | 9 | 16 | 23 | 28 | 30 | 31 | 30 | 32 | 32 | 31 | 31 | 30 |
| PCPU share for Domain 2 | 10 | 18 | 25 | 30 | 31 | 32 | 31 | 32 | 31 | 32 | 33 | 32 |
| PCPU share for Domain 3 | 80 | 65 | 50 | 37 | 35 | 34 | 32 | 33 | 33 | 34 | 33 | 34 |

**Table 6.9: PCPU share for original credit scheduler with 10%, 15% and 90% workloads for Dom1, Dom2, Dom3 respectively**

| Time Duration(Sec) | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 | 110 | 120 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PCPU share for Domain 1 | 10 | 9 | 8 | 8 | 9 | 10 | 10 | 9 | 8 | 9 | 8 | 9 |
| PCPU share for Domain 2 | 15 | 14 | 14 | 13 | 13 | 12 | 13 | 13 | 14 | 13 | 12 | 13 |
| PCPU share for Domain 3 | 68 | 72 | 78 | 80 | 79 | 79 | 78 | 78 | 76 | 78 | 79 | 78 |

**Table 6.10: PCPU share for modified credit scheduler with 10%, 15% and 90% workloads for Dom1, Dom2, Dom3 respectively**

Figures 6.1 to 6.4 show the graphical comparison of the three domains under experiment. The discussions regarding the observations in the tables, and figures are made in the following section.

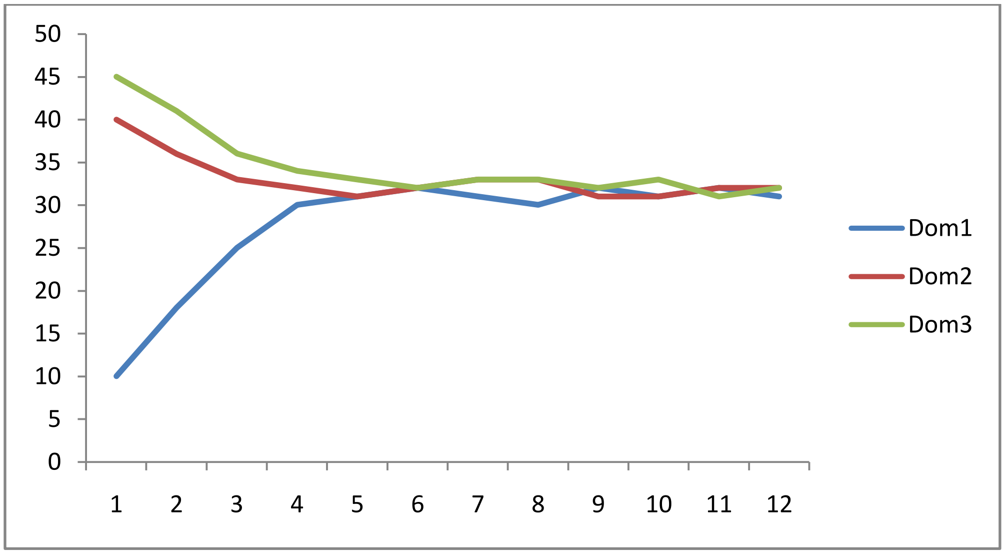


*Figure 6.1 Comparison of three guest domains for PCPU share by credit scheduler for Case 1*

### 6.1.4 Discussions on the results of experiments in modified Credit Scheduler:

Table 6.7 and 6.9 show the experimental results of the three user domains when they are scheduled by the original Credit scheduler. As we can see that the actual allocated PCPU shares of these three user domains are all converged to around 32-33%. It is because the original Credit scheduler allocates the PCPU share only according to the predefined weight. As a result, in the

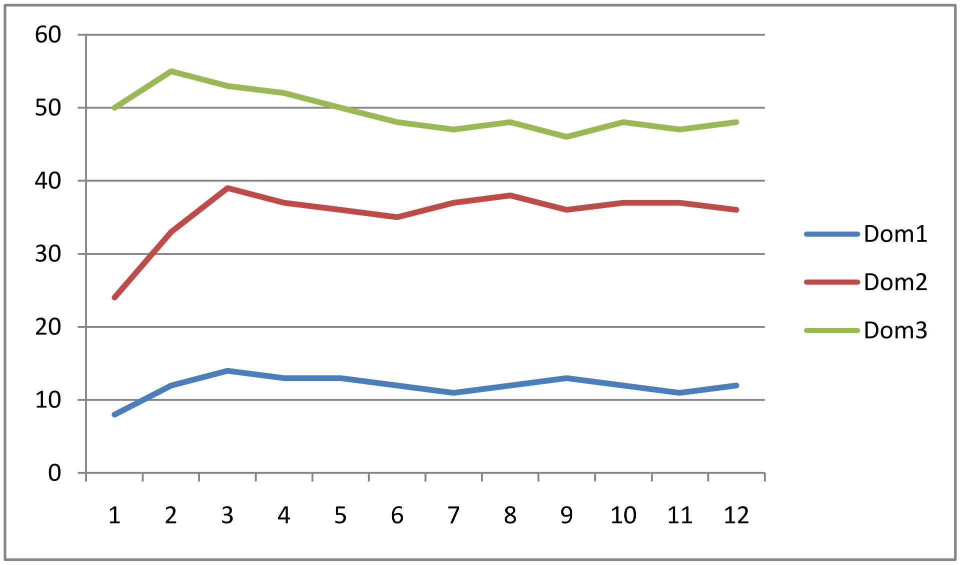


*Figure 6.1 Comparison of three guest domains for PCPU share by modified credit scheduler for Case 1*

first case, the *dom3* could not get enough PCPU share although the workload of it is greater than other two domains while the *dom*1 got around equal amount of PCPU share as *dom2* and *dom*3 although its workload is lesser than those two domains. Similar is the second case, when the *dom*3 has significantly higher amount of workload than the other two domains, but still, the PCPU allocation is still the same. It was observed that the PCPU share would converge around the average value of the three domains in all the cases, regardless of the workload.

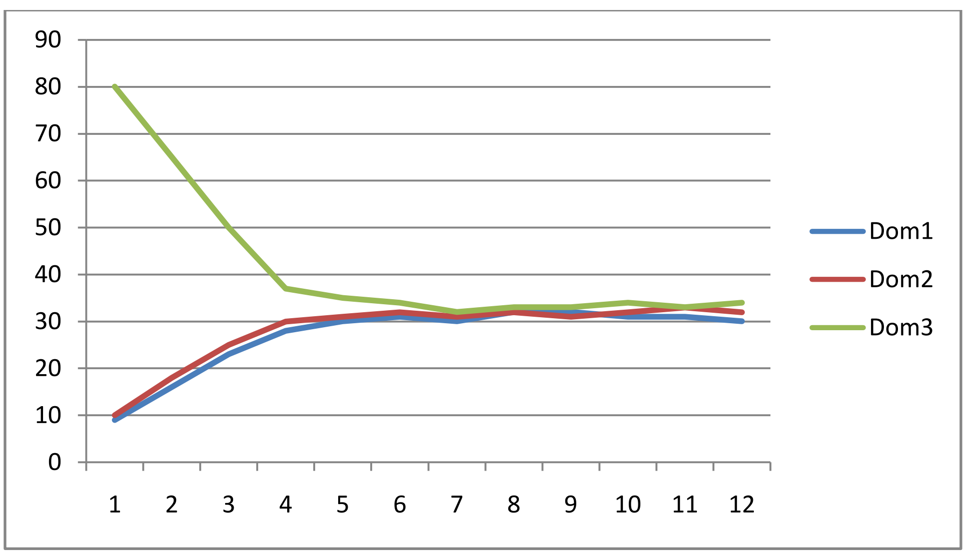


*Figure 6.3: Comparison of three guest domains for PCPU share by credit scheduler for Case2*

Table 6.8 and 6.10 show the experimental results under our proposed modified credit scheduler with dynamic weight update. In the first case, the generated workload of the three domains are 20%, 60% and 80%, their expected PCPU shares should be 20/(20+60+80) ~= 13% , 60/(20+60+80) ~= 37% and 80/(20+60+80) ~= 50% respectively. The results in Table 6.8 show that the actual PCPU shares of *dom*1, *dom*2 and *dom*3 are converged to around 12% , 37% and 48%, respectively. Similarly, in the second case, the generated workload of the three domains are 10%, 15% and 90%, their expected PCPU shares should be 10/(10+15+90) ~= 9% , 15/(10+15+90) ~= 14% and 19/(10+15+90) ~= 80% respectively. The results in Table 6.10 show that the actual PCPU shares of *dom*1, *dom*2 and *dom*3 are converged to around 8% , 13% and 78%, respectively.

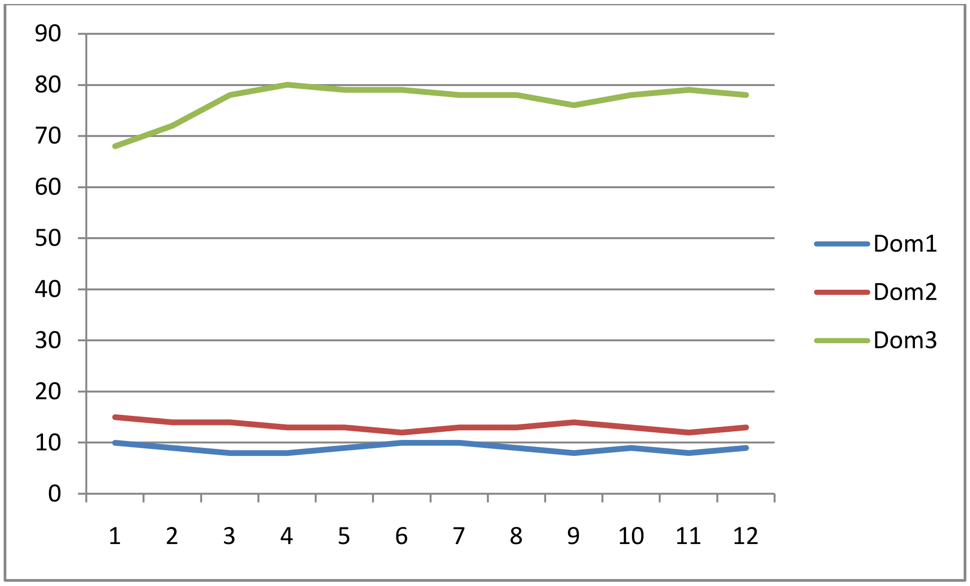


*Figure 6.4 Comparison of three guest domains for PCPU share by modified credit scheduler for Case 2*

The above results show that the modified credit scheduler with dynamic weight update outperforms the original Credit scheduler, and it is consistent with the expected value. This is because the proposed modified credit scheduler with dynamic weight update allocates the proportional fair PCPU share based on the dynamic workload while the original Credit scheduler based on the predefined weight values. Therefore, a user domain with heavier workload can obtain more share of the PCPU with the implementation of the modified credit scheduler with dynamic weight update.