UNIVERSITY OF CAMBRIDGE

Cloud Computing

VM Scheduling

Eva Kalyvianaki
ek264@cam.ac.uk

Contents

- Resource management and scheduling.
- Policies and mechanisms.
- Case-study of CPU schedulers:
  - Xen CPU schedulers: BVT, SEDF, Credit

- Material is from the paper:

  "Comparison of the Three CPU Schedulers in Xen",

  by L. Cherkasova, D. Gupta, A. Vahdat

  published in ACM SIGMETRICS Performance Evaluation Review, Volume 35 Issue 2, September 2007, pages 42-51

Cloud Resource Management Policies

1. **Admission control** → prevent the system from accepting workload in violation of high-level system policies.
2. **Capacity allocation** → allocate resources for individual activations of a service.
3. **Load balancing** → distribute the workload evenly among the servers.
4. **Energy optimisation** → minimisation of energy consumption.
5. **Quality of service (QoS) guarantees** → ability to satisfy timing or other conditions specified by a Service Level Agreement (SLA).

Resource Management and Scheduling

- Scheduling in a computing system → deciding how to allocate resources of a system, such as CPU cycles, memory, secondary storage space, I/O and network bandwidth, between users and tasks.
- What is a scheduler? → it is a program that implements a particular scheduling algorithm.
- Critical function of any man-made system.
- It affects the three basic criteria for the evaluation of a system:
  1. Functionality (does it work as it should?)
  2. Performance (does it perform well?)
  3. Cost (how much does it cost?)
- Policies and mechanisms for resource allocation.
  - **Policy** → principles guiding decisions.
  - **Mechanisms** → the means to implement policies.

Motivation

- Cloud resource management is challenging because:
  - It requires complex policies and decisions for multi-objective optimisation.
  - The complexity of the system makes it impossible to have accurate global state information.
  - Cloud service providers are faced with large fluctuating loads which challenge the claim of Cloud elasticity.
  - Affected by unpredictable interactions with the environment, e.g., system failures, attacks.

Scheduling Algorithms

- Scheduling → responsible for resource sharing and multiplexing at several levels. For example:
  - A server can be shared among several virtual machines.
  - A virtual machine could support several applications.
- A scheduler decides:
  - The amount of resources allocated
  - The duration of the particular resource allocation
- A scheduling algorithm should be:
  - efficient, fair, and starvation-free.

Scheduling Algorithms (cont'd)

- The objectives of a scheduler can vary according to application:
  - Batch system → maximise throughput.
  - Real-time system → meet the deadlines.
- Common scheduling algorithms:
  - Round-robin, First-Come-First-Serve (FCFS), Shortest-Job-First (SJF).
- Multimedia applications (e.g., audio and video streaming)
  - Have soft real-time constraints.
  - Require statistically guaranteed maximum delay and throughput.
- Real-time applications have hard real-time constraints.

Deadlines

- **Hard deadlines** → if the task is not completed by the deadline, other tasks which depend on it may be affected and there are penalties; a hard deadline is strict and expressed precisely as milliseconds, or possibly seconds.

- **Soft deadlines** → more of a guideline and, in general, there are no penalties; soft deadlines can be missed by fractions of the units used to express them, e.g., minutes if the deadline is expressed in hours, or hours if the deadlines is expressed in days.

Resource Management in Virtualized servers

Considerations:

1. Many different workloads
2. Finite numbers of workloads can be hosted on each server
3. Time-varying workload requirements
4. Performance and resource isolation among the workloads

"Comparison of the Three CPU Schedulers in Xen",

by L. Cherkasova, D. Gupta, A. Vahdat

published in ACM SIGMETRICS Performance Evaluation Review, Volume 35 Issue 2, September 2007, pages 42-51

Terminology for CPU Schedulers

- Many schedulers are Proportional Share (PS) schedulers:

  PS scheduling allocates CPU in proportion to the number of shares (weights) that VMs have been assigned.

  - Tend to provide instantaneous form of sharing

- Evaluation criteria:

  1. **Fairness:** time interval over which the scheduler provides fair CPU allocation
  2. **Allocation error:** the difference between the allocation and real demanded allocation.

- Fair-Share vs Proportional Scheduling:

  - Fair-share schedulers attempt to provide a ***time-averaged*** form of proportional sharing based on the actual usage measured over long time periods.

Example of Fair and PS Schedulers

Problem:

Two clients C1 and C2 share a system with equal CPU shares. Assume that C1 is using the CPU for some time and that C2 is blocked.

Question: What happens when C2 becomes active again?

Solution:

1. A fair-scheduler will allocate a large CPU share to C2 to "catch up" with C1
2. A PS scheduler will treat C1 and C2 equally, because it is unfair to penalise C1 for consuming otherwise idle resources

Terminology for CPU Schedulers (con't)

1. **Work-conserving (WC-mode)**
   - CPU shares are merely guarantees: as long as there is some work to be done and all clients have used their shares the CPU will be used.
2. **Non work-conserving (NWC-mode)**
   - CPU shares are caps. Clients will get their share of CPU and only that.

Terminology for CPU Schedulers (con't)

1. **Non-preemptive schedulers**
   - These schedulers allow running clients to finish their CPU slice. These schedulers only make decisions when the running client gives up the CPU.

2. **Preemptive schedulers**
   - Running clients can be preemptive for others clients to run. These schedulers rerun their scheduling decisions for when a new client arrives.

   - Preemption is good for I/O intensive workloads, why?

Workload Management

Workload management is the ability to **precisely** assign (CPU, memory, I/O) resources to applications. Applications provide service levels on their desired performance and the workload managers assign resources to comply to these levels.

Workload management approaches:

1. **Static:**
   - Resources are estimated once and are assigned statically to applications despite their varying workloads
   - Problem of over-provisioning!!!!
2. **Dynamic:**
   - Dynamically allocate resources to match application workload resource demands.

➢ Workload managers usually use PS schedulers in NWC

- Why? → for performance isolation reasons

Case study: Xen CPU Schedulers

Over the years, there has been proposed and used different CPU schedulers for the Xen VMM:

1. Borrowed Virtual Time (BVT)
2. Simple Earliest Deadline First (SEDF)
3. Credit Scheduler

- These algorithms were used as in the order above. Today, Xen uses the Credit Scheduler.

Borrowed Virtual Time (BVT) (very briefly)

- **Objective** – is a fair-share scheduler based on the concept of virtual time, dispatching the thread with the runnable VM with the lowest virtual time first.
- Allows latency-sensitive applications to wrap back in virtual time to gain scheduling priority and so applications borrows virtual time from its future allocation.
- The VM effectively "borrows" virtual time from its future and thus does not disrupt long-term CPU sharing.
- CPU allocation.
- BVT has the following features:
  - Preemptive, WC-mode only
  - Optimally-fair
  - Low-overhead implementation on uni- and multiprocessors
- But, it does not support NWC-mode and so it limits isolation

Simple Earliest Deadline First (SEDF)

- In SEDF each domain specifies three values: (s_i, p_i, x_i), where:
  - s_i := is the slice
  - p_i := is the period
  - A Dom_i will receive s_i units of time every period of length p_i.
  - x_i := indicates if the domain can receive extra CPU time (WC-mode)
  - If WC-mode is enabled, SEDF distributes slack time fairly among all runnable domains.

  - Example: how can we assign 30% of CPU to a domain?
    - (3ms, 10ms, 0), or
    - (30ms, 100ms, 0)
    - Does it make a difference if we choose a period of 10ms or 30ms?

Simple Earliest Deadline First (SEDF)

- For each domain, SEDF tracks two more values (d_i, r_i), where:
  - d_i := is the time at which Dom_i's current period ends, i.e., the deadline
  - r_i := is the remaining CPU time of Dom_i's current period

  - **Policy** → The runnable domain with the earliest deadline is picked to be scheduled next.

  - SEDF characteristics:
    1. Preemptive, WC and NWC modes
    2. Fairness depends on a value of a period
    3. Implements per CPU queue; but, lacks of global load balancing on multiprocessors

Credit Scheduler

- Credit scheduler is Xen's latest PS scheduler featuring automatic load balancing of virtual CPUs across physical CPUs on an SMP host.
- Before a physical CPU goes idle, the scheduler will consider other CPUs with runnable vCPUs to run.

- **Goal** → This approach guarantees that no CPU idles when there is runnable work in the system.

- For each VM, the scheduler assigns a *weight* and a *cap*.
- If cap is 0, then the VM can receive extra time (WC-mode)
- If cap is non-0, it shows the max time it can receive (NWC)

Credit Scheduler

- The scheduler works with 30ms of slice: a vCPU receives 30ms before being preempted by another VM.
- Every slice (30ms) the priorities/credits of all runnable VMs are recalculated.
- The scheduler monitors CPU usage every 10ms.

- Credit Scheduler characteristics:
    1. Non-preemptive, WC and NWC modes
    2. Global load balancing on multiprocessors

BVT → SEDF → Credit

- BVT → SEDF : NWC mode
- BVT, SEDF → Credit: automatic, load balancing in SMPs

- In theory there are distinct differences among the schedulers.
- But, what about setting their different parameters? Would these affect their performance?

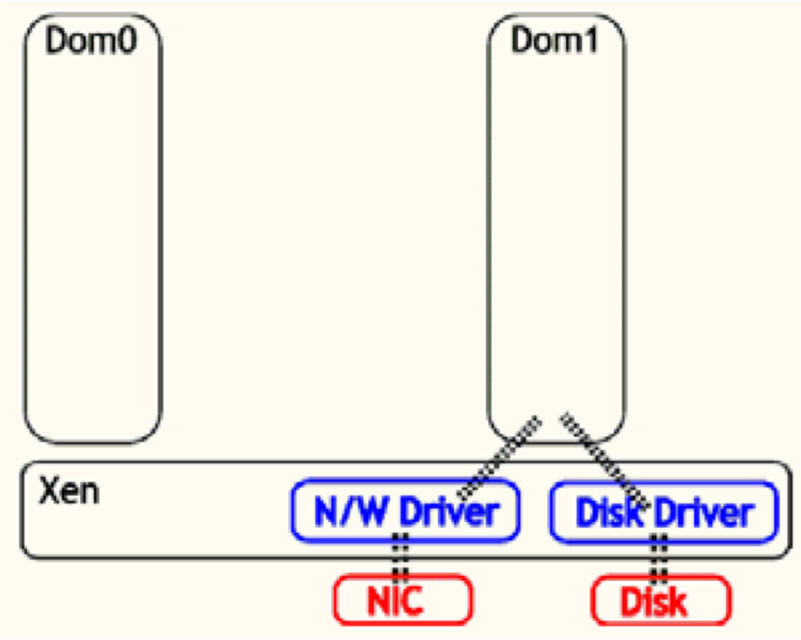
Experimental Comparison of the Three Schedulers

- Experimental evaluation using three benchmarks:
  1. **Web server** application to measure web server throughput
  2. **Iperf** to measure the maximum achievable network throughput
  3. **Disk read** to measure disk read throughput

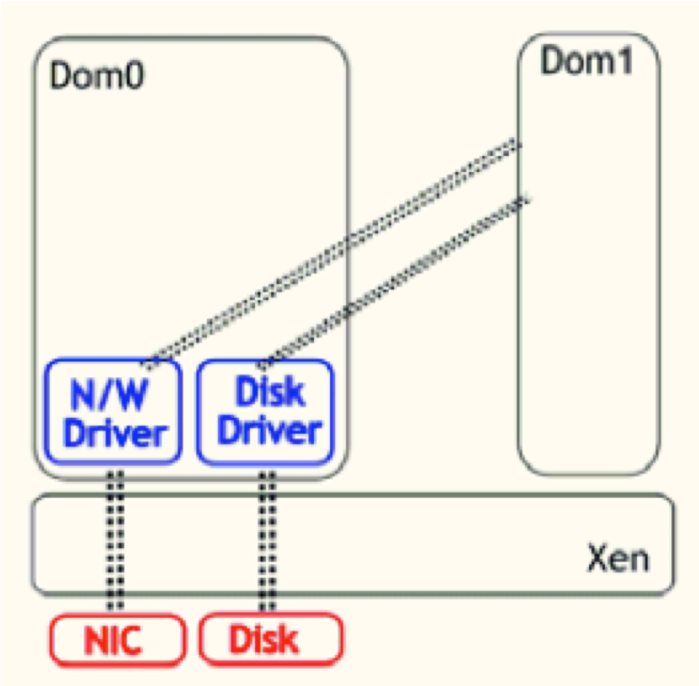
- Machines' specs:

  Dual CPU HP workstations LP200R, with 1-GHz PIII processors, 2GB RAM and 1 Gbits NICs (network interface cards) running Xen 3.0.3

I/O model for VMs in Xen (Figure, 1b from paper)



(a) Initial I/O Model



(b) "Current" I/O Model in Xen

- Dom_0 performs I/O processing on behalf of the guest domains
- This design shows that Dom_0 also requires a certain CPU (resource) allocation to perform I/O for the VMs.
- **But**, how much allocation is good enough?

Effect in performance of SEDF parameters' values

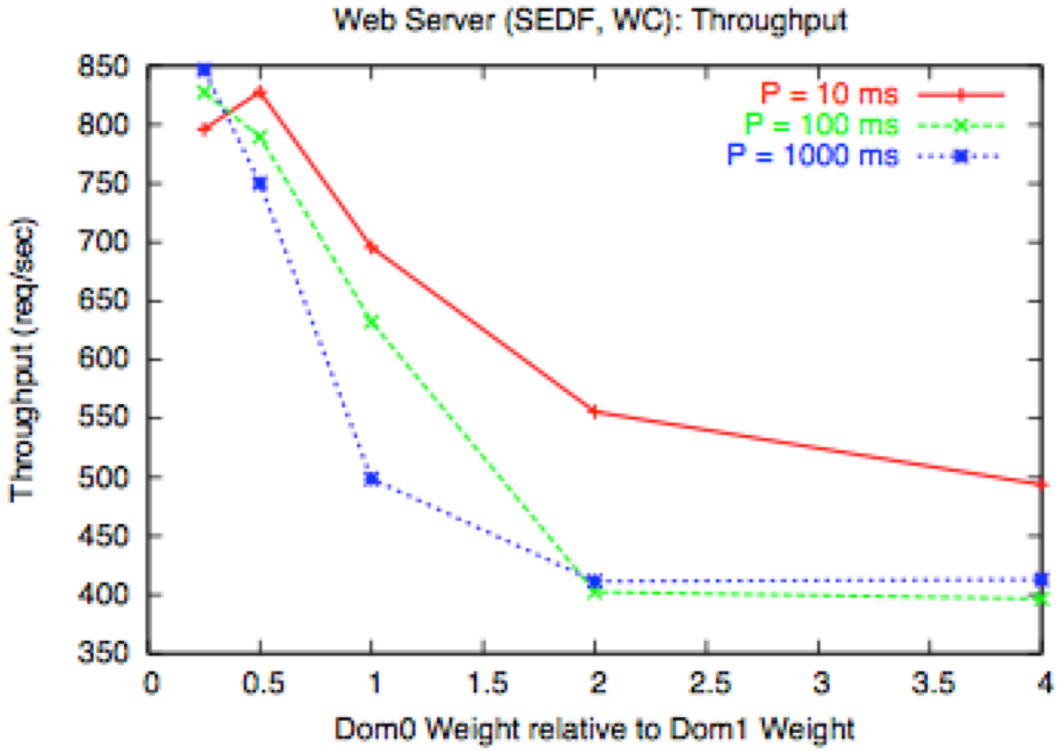
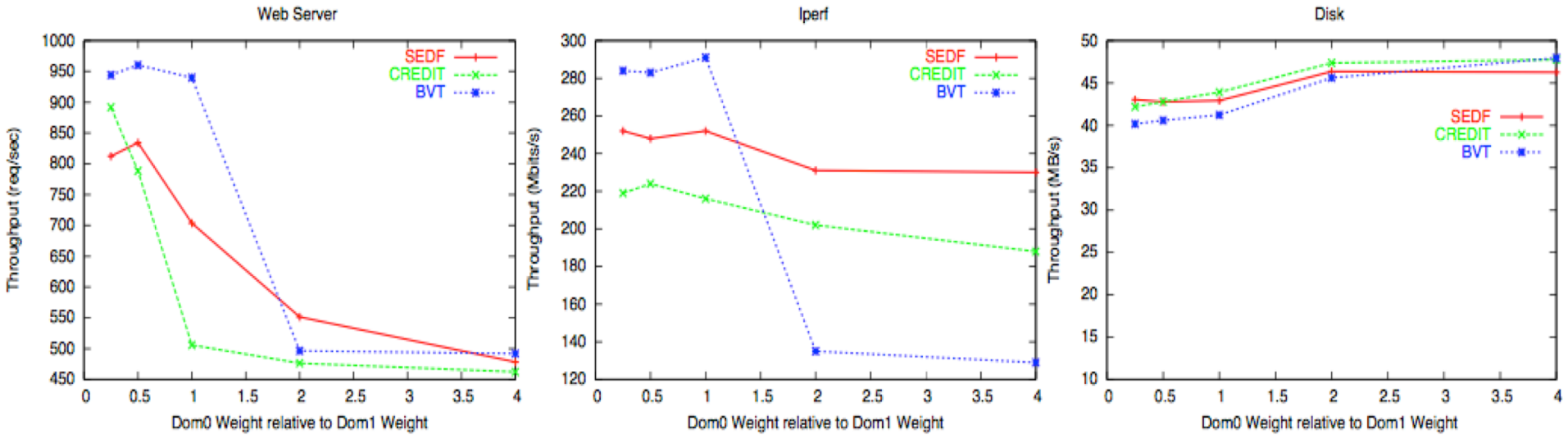


Figure 2.b from paper

- Dom_0 performs I/O processing on behalf of the guest domains
- $Dom_0 = x * Dom_1$ (across the x axis)
- Web server throughput decreases to increasing weights
- Web server throughput decreases to increasing P values, why?
- MSc content Table 2 from paper

Comparison of schedulers for workloads



(a) Web server (b) iperf (c) Disk

Figure 5: Evaluating the three schedulers (WC-mode) for different workloads.

Figure 5 from paper

- Application performance varies significantly under different schedulers
- I/O applications are highly sensitive to the amount of CPU given to Dom_0.

Performance comparison of the schedulers in the SMP case

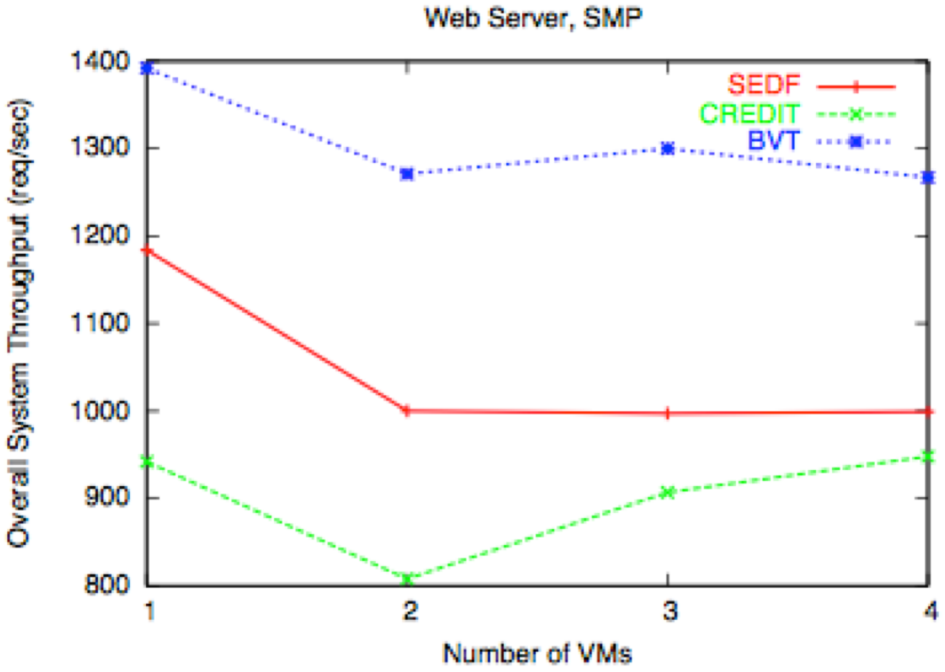


Figure 10.c from paper

(c) Web server, WC-mode, 1 to 4 VMs.

- Machine specs: dual CPU HP workstation, 2 vCPUs for dom_0, dom_1
- Each VM is given 2 vCPUs; runs a web server and all domains are given equal weights.
- The Credit scheduler is able to increase throughput with more VMs
- Credit throughput is lower than SEDF and BVT. Further analysis showed a high allocation error for Credit (Figure 11)

Summary

- Resource management and scheduling
- Policies and mechanisms.
- Case-study of CPU schedulers:
  - Xen CPU schedulers: BVT, SEDF, Credit

- Material is from the paper:

  "Comparison of the Three CPU Schedulers in Xen",

  by L. Cherkasova, D. Gupta, A. Vahdat

  published in ACM SIGMETRICS Performance Evaluation Review, Volume 35 Issue 2, September 2007, pages 42-51