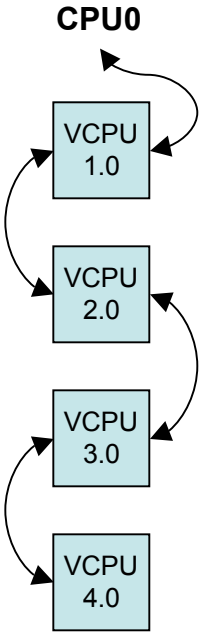
The Xen Credit CPU Scheduler

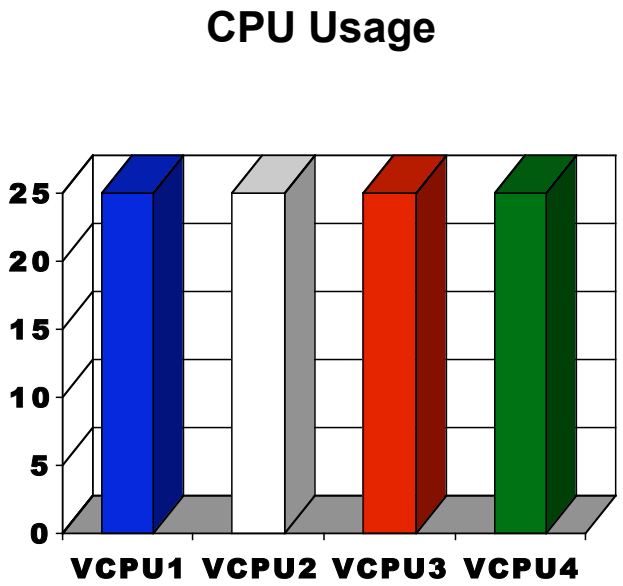
Emmanuel Ackaouy
ack@xensource.com

Goals

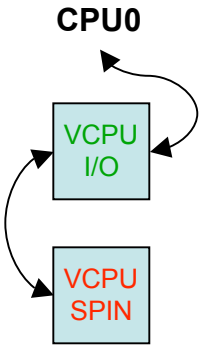
- **Maintainability**
  - Simple & Fast
- Zero-admin **good default behavior**
  - **Fairness**: Weight based
  - **I/O *versus* CPU** intensive VCPUs
  - Tuning via per-VM ***weight*** & ***cap***
- Better **SMP support**
  - **Work conserving**
  - Transparent cross **CPU migration**
  - **System wide** accounting & fairness

Round Robin

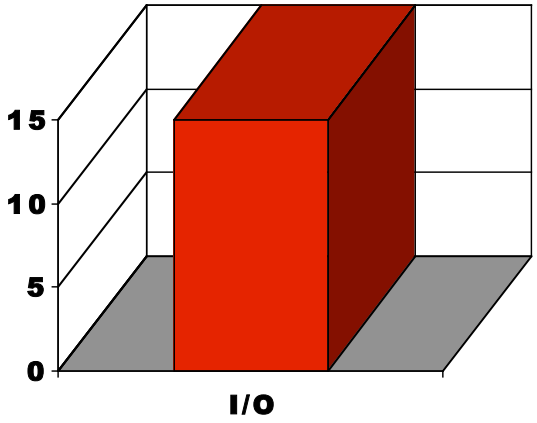

CPU0
VCPU 1.0
VCPU 2.0
VCPU 3.0
VCPU 4.0

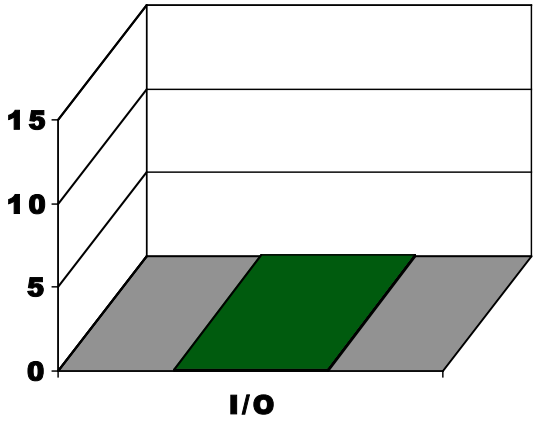

CPU Usage
25
20
15
10
5
0
VCPU1
VCPU2
VCPU3
VCPU4

Preemption

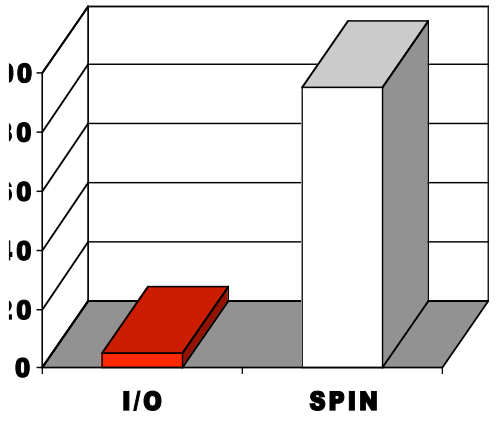


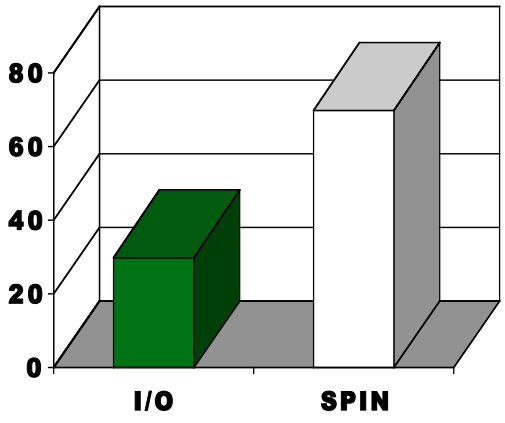
Latency to Run



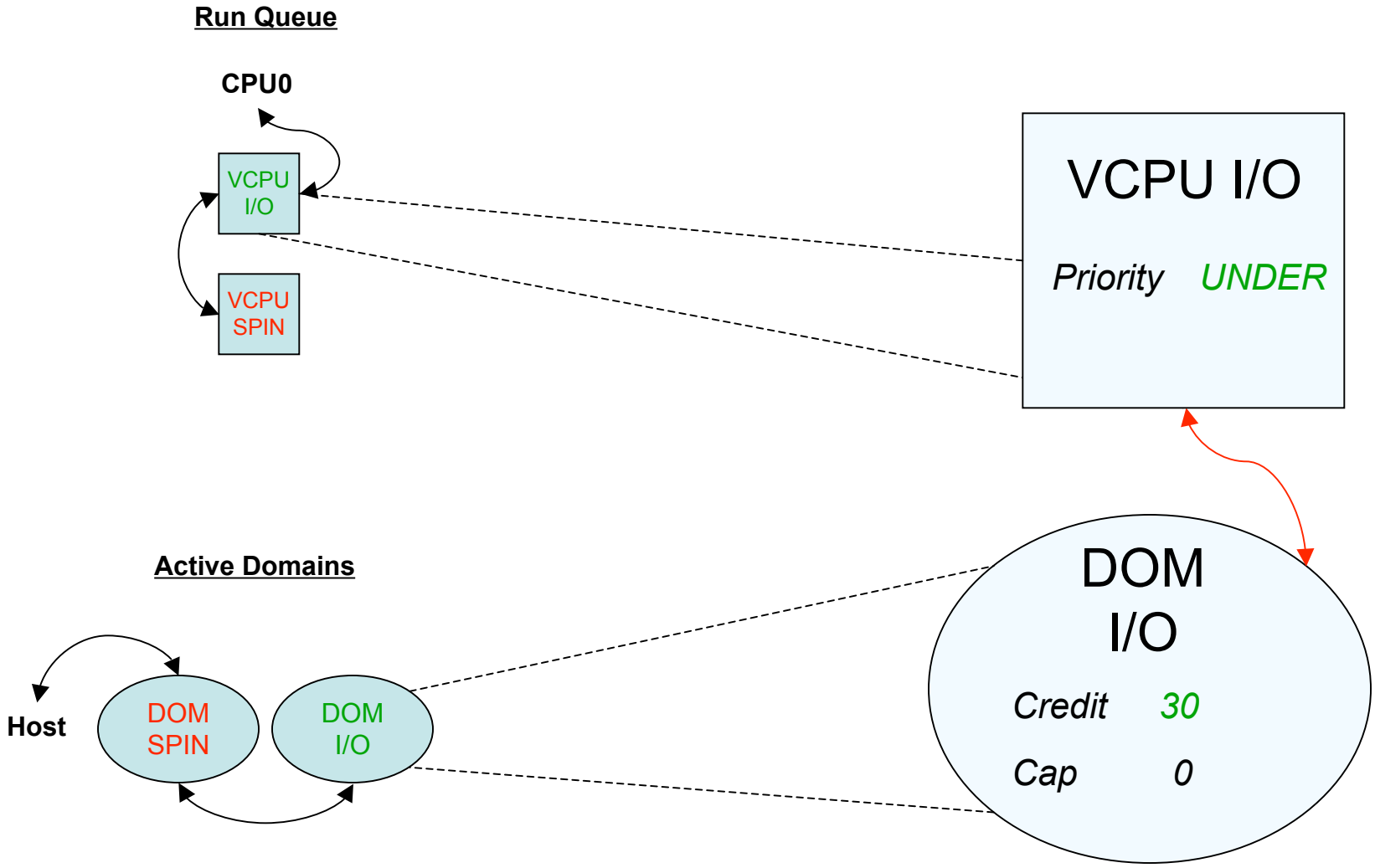


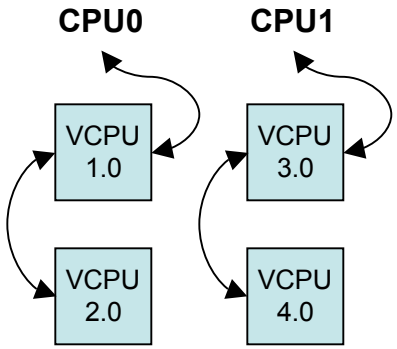
CPU Usage

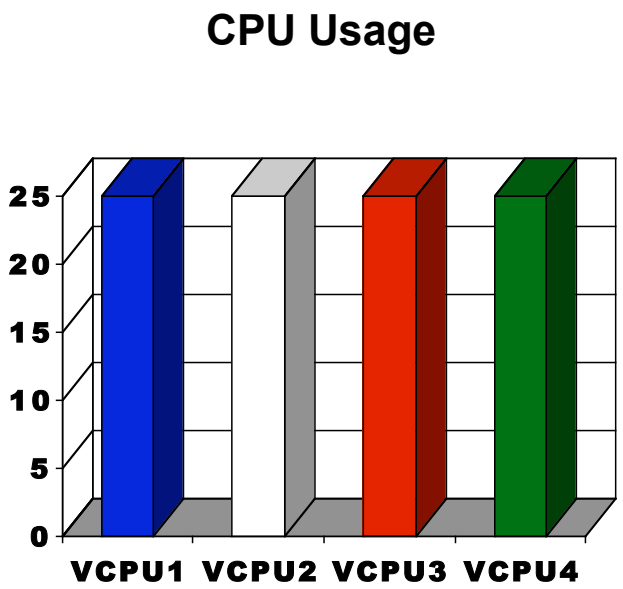


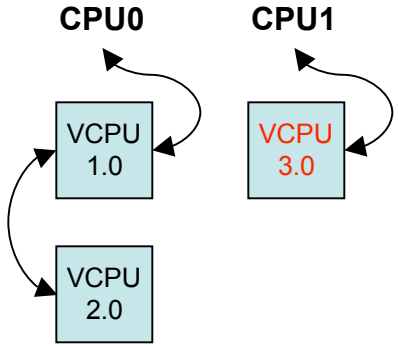


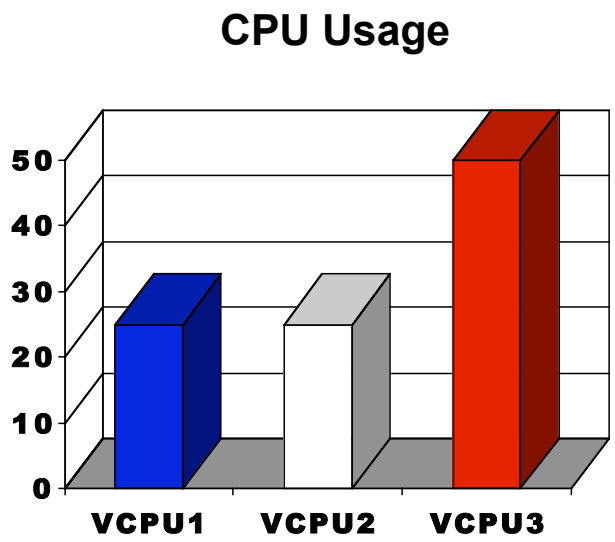
Priorities!

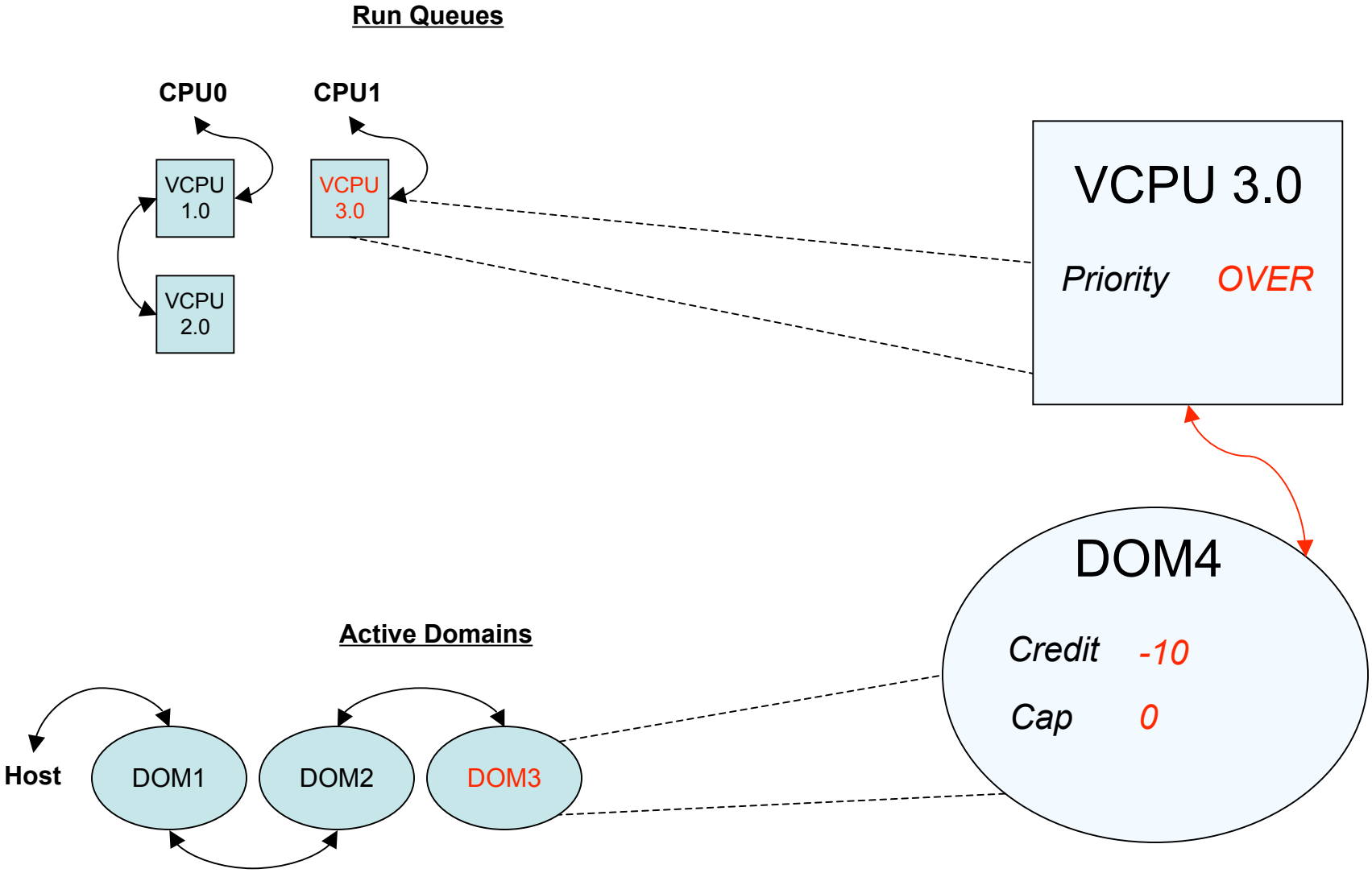

Run Queue
CPU0
VCPU I/O
VCPU SPIN
VCPU I/O
Priority UNDER
Active Domains
Host
DOM SPIN
DOM I/O
DOM I/O
Credit 30
Cap 0


CPU0
CPU1
VCPU 1.0
VCPU 3.0
VCPU 2.0
VCPU 4.0

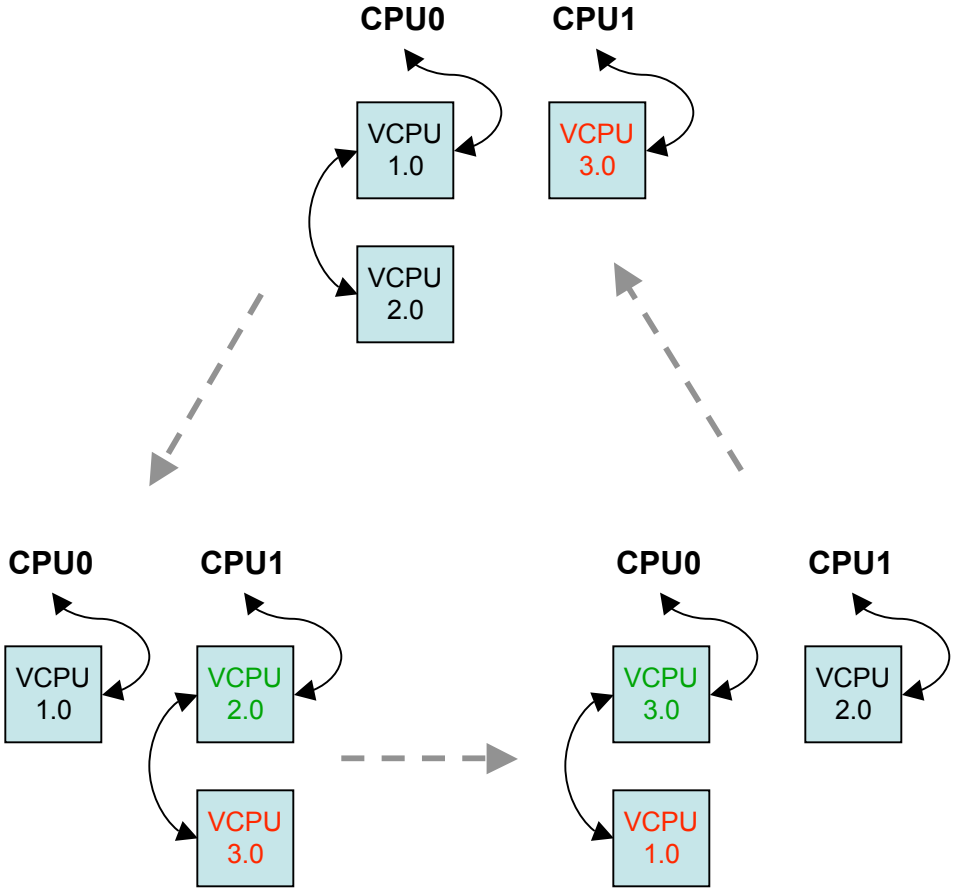

CPU Usage
25
20
15
10
5
0
VCPU1 VCPU2 VCPU3 VCPU4

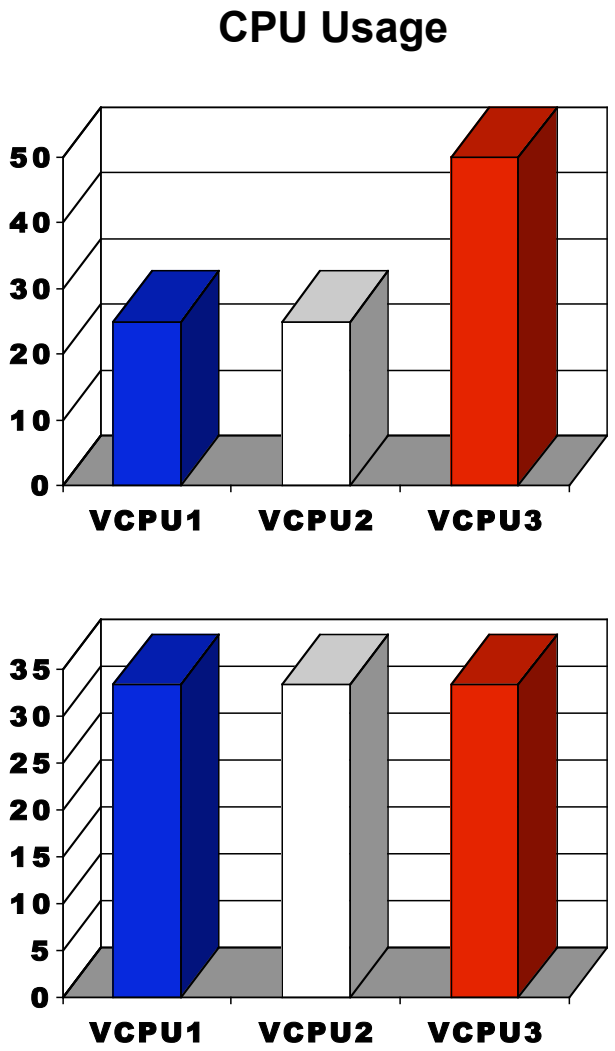

CPU0
CPU1
VCPU 1.0
VCPU 3.0
VCPU 2.0


CPU Usage
50
40
30
20
10
0
VCPU1
VCPU2
VCPU3


Run Queues
CPU0
CPU1
VCPU 1.0
VCPU 3.0
VCPU 2.0
VCPU 3.0
Priority OVER
DOM4
Credit -10
Cap 0
Active Domains
Host
DOM1
DOM2
DOM3

Self Balancing!





Global Accounting

- Periodically recompute priorities for all active VCPUs
  - UNDER (below fair share usage)
  - OVER (above fair share)
- VCPU fair shares derived from VM weight and cap in proportion to all active VMs
- Accounting cost
  - A function of # active VCPUs in system
  - **NOT** a function of # scheduling decisions

Local CPU Scheduling

- Per-CPU sorted runq
  - 2 VCPU priorities: OVER and UNDER fair share
- Clock tick
  - Charge running VCPU
- Scheduling decision
  - End of time slice or VCPU blocks
  - Common case: run next local VCPU under fair share
  - Otherwise: pick queued remote under fair share VCPU, local over VCPU, or remote over VCPU (in that order)
- Scheduling cost
  - Common case: Pick next VCPU on local runq

Credit Scheduler Internals