Toward SLO Complying SSDs Through OPS Isolation

October 23, 2015

~~Hongik University~~
UNIST
(Ulsan National Institute of Science & Technology)

Sam H. Noh

Outline

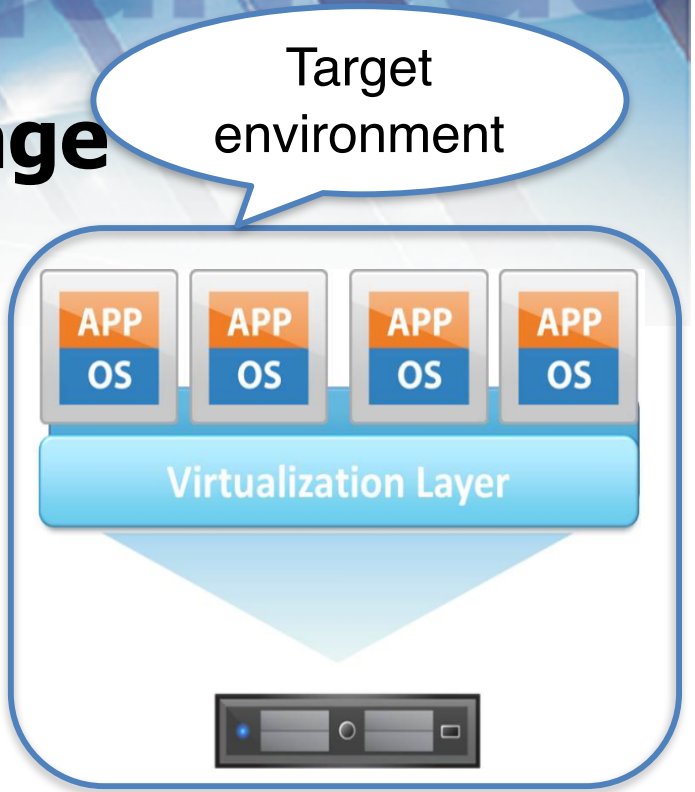
- **Part 1: FAST 2015**
- **Part 2: Beyond FAST**

Outline

- **Part 1: FAST 2015**
  - 김재호 박사
- **Part 2: Beyond FAST**

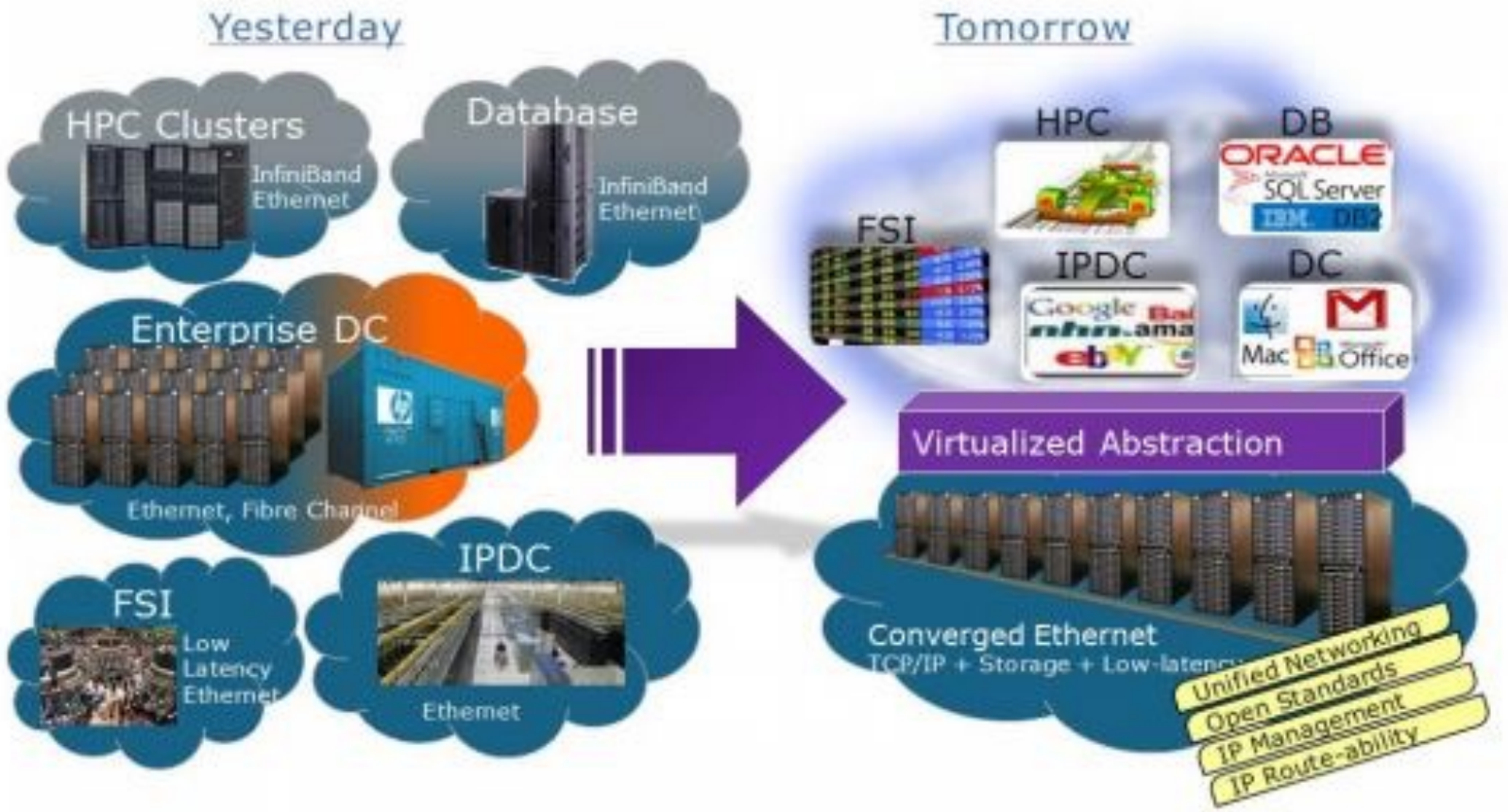
Flash Memory Everywhere

- **From embedded to server storage**



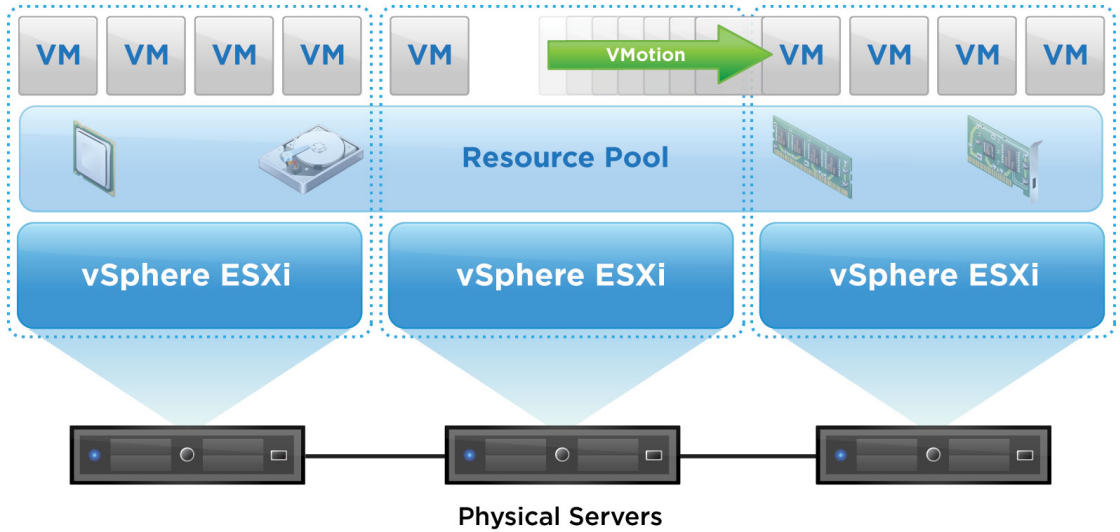
Introduction

- **Infrastructure convergence**
  - Virtualized servers

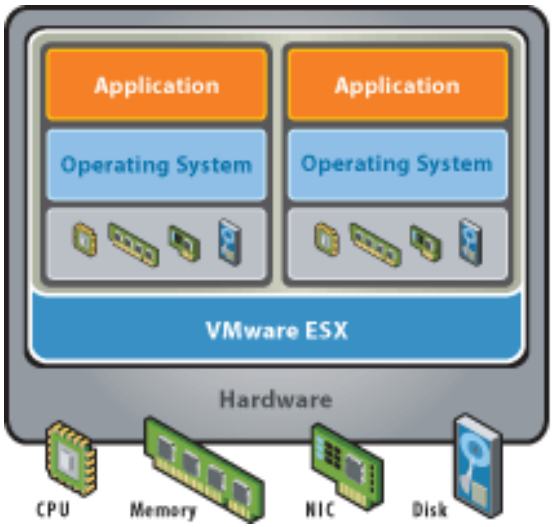


Motivation

- **Virtualization system**
  - Need to satisfy Service Level Objective (SLO) for each VM
  - SLO is provided through hardware resource isolation
- **Existing solutions for isolating CPU and memory**
  - Distributed resource scheduler [VMware inc.]
  - Memory resource management in VMware ESX server [SIGOPS OSR 2002]



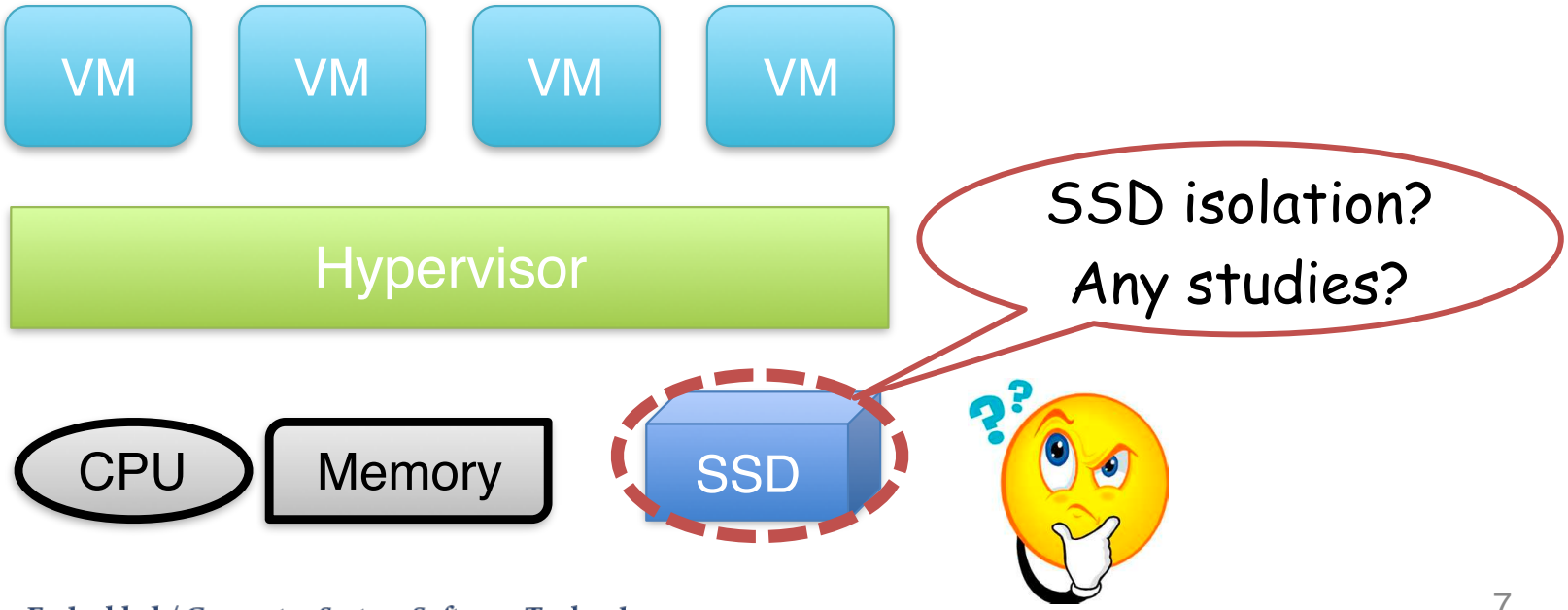
[Distributed resource scheduler]



[ESX server]

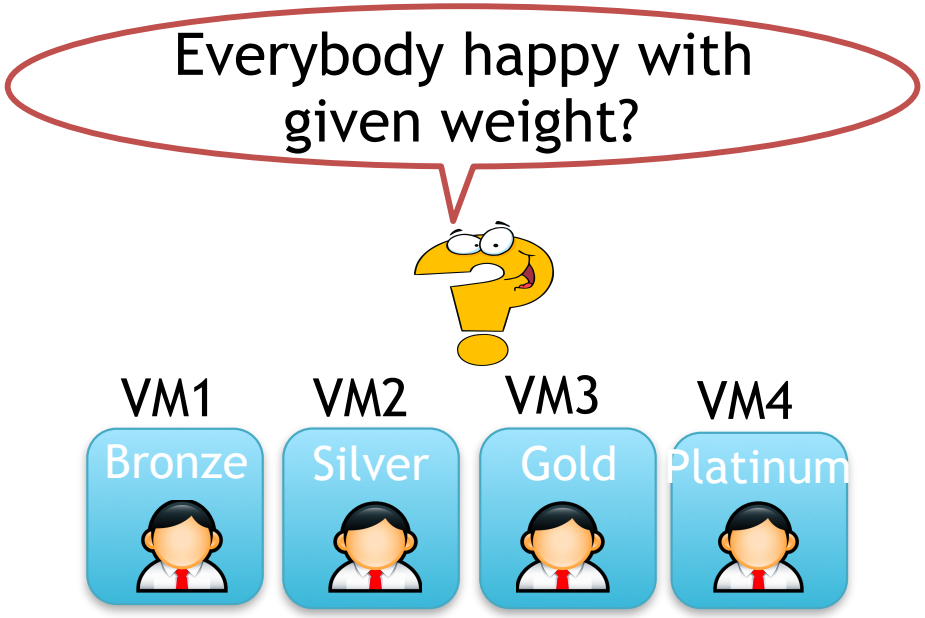
Motivation

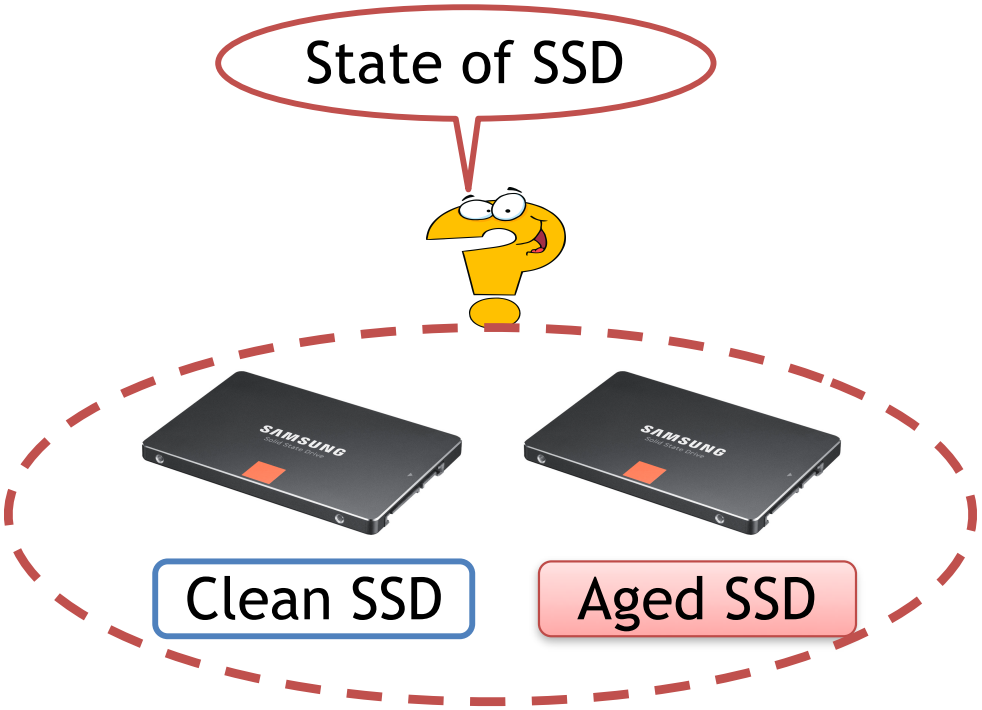
- **Few studies on SSD resource isolation**
  - S-CAVE [PACT'13], vCacheShare [USENIX ATC'14]
- **Challenges for isolating SSDs**
  - Performance is quite sensitive to workload characteristics
  - More complex architecture than HDD



Questions Raised

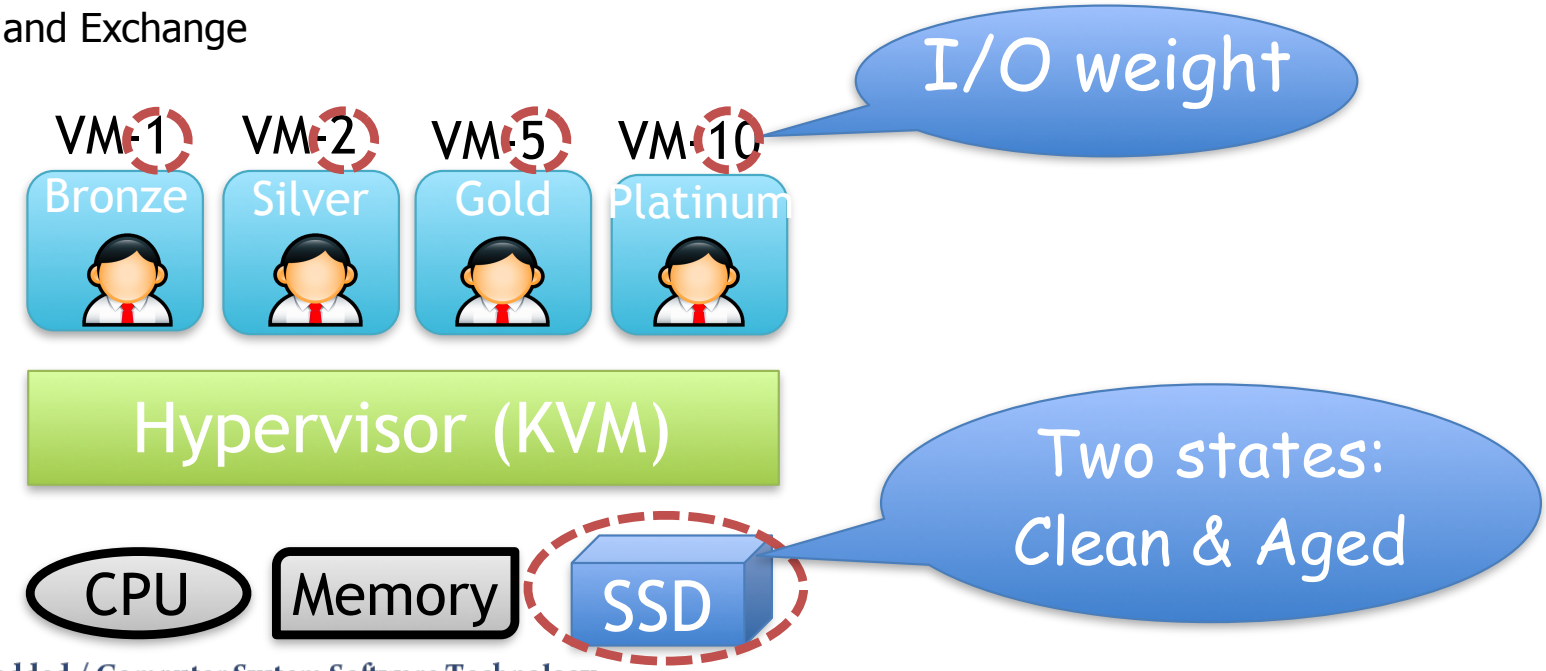
- **Is I/O bandwidth of the shared SSD proportionally distributed among the VMs?**
- **How does state of the SSD affect proportionality?**





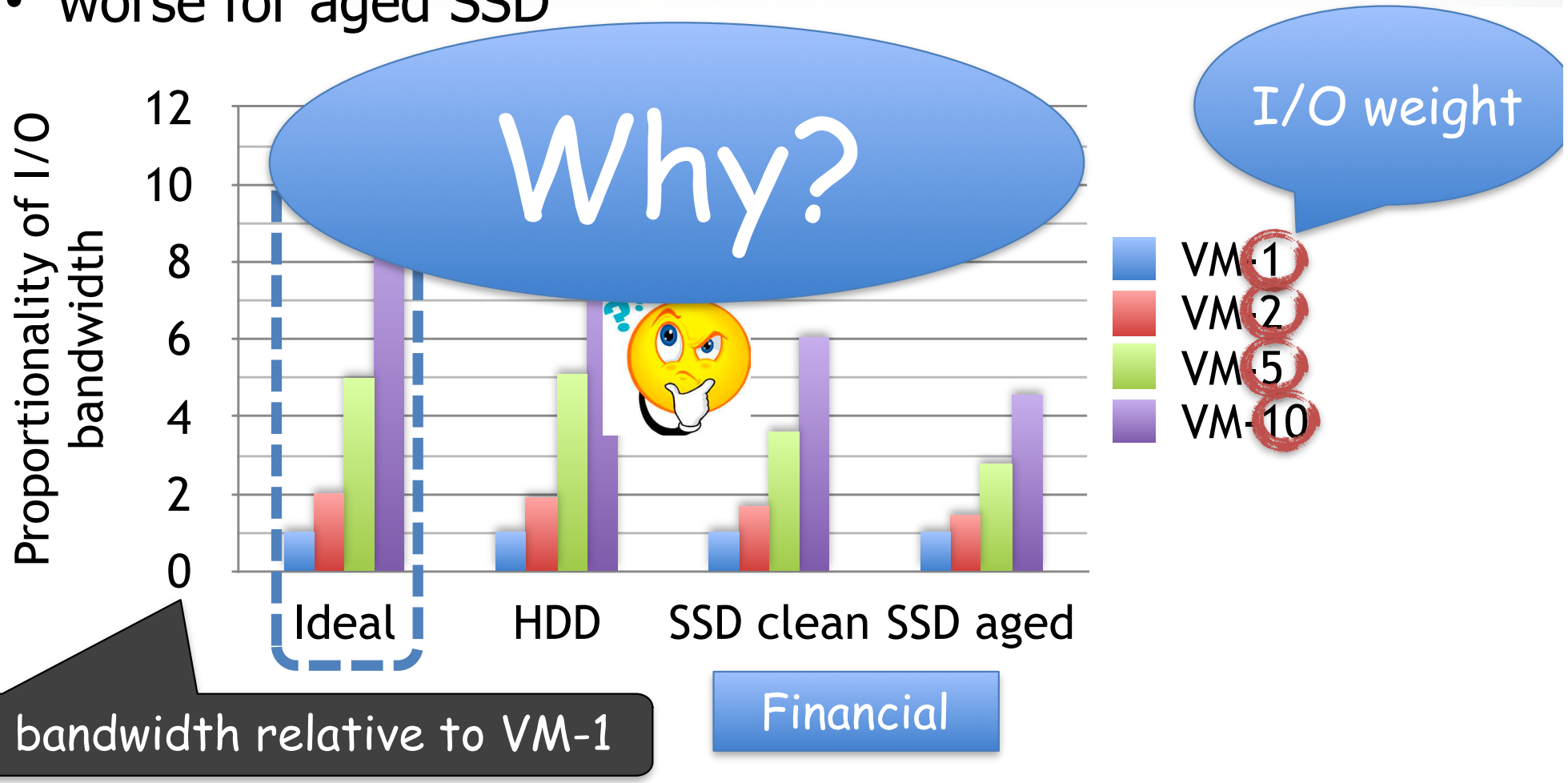
Experiments on Commercial SSD

- Linux kernel-based virtual machine (KVM) on 4 VMs
- Assign proportional I/O weight
  - Using Cgroups feature in Linux kernel 3.13.x
  - VM-x: x is I/O weight value (Higher value: Allocate higher throughput)
- SSD as shared storage
  - 128GB capacity, SATA3 interface, MLC Flash
  - clean SSD: empty SSD
  - aged SSD: full SSD (busy performing garbage collection)
- Each VM runs the same workload concurrently
  - Financial, MSN, and Exchange



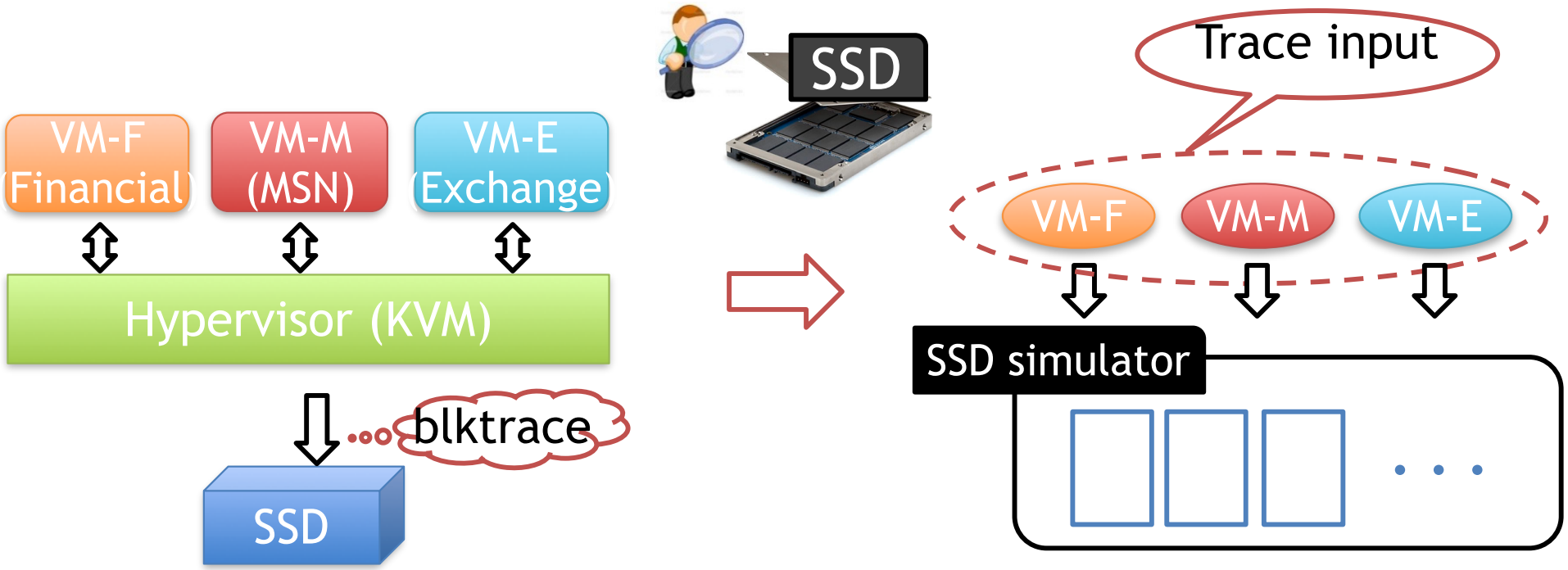
Results

- **HDD: Proportionality close to I/O weight**
- **Not so, for SSD**
  - worse for aged SSD



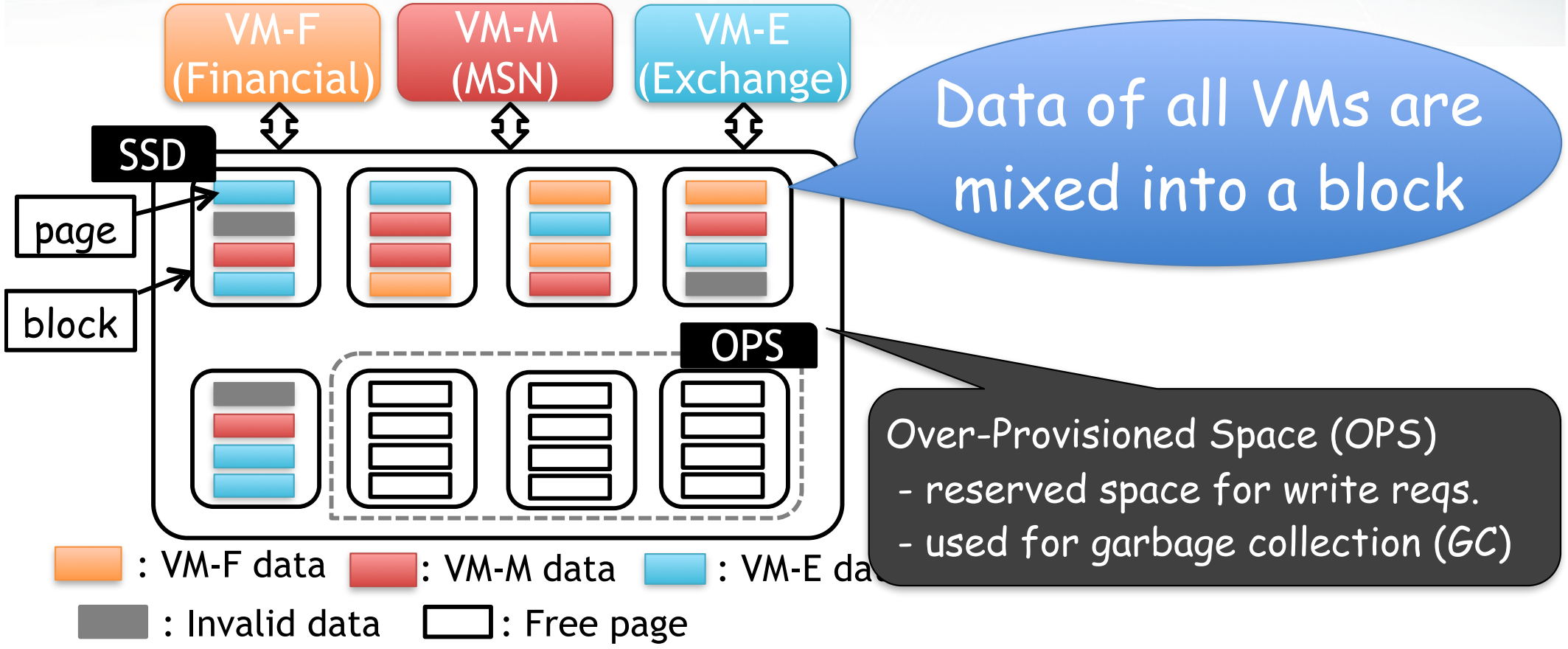
Monitor Internal Workings of SSD

- **Commercial SSD: Proprietary, black box SSDs**
- **Monitor using simulator**
  - SSD simulator: DiskSim SSD Extension
  - Workloads: Financial, MSN, and Exchange
    - Traces are captured as VMs run concurrently on real system



Analysis #1 : Mixture of Data

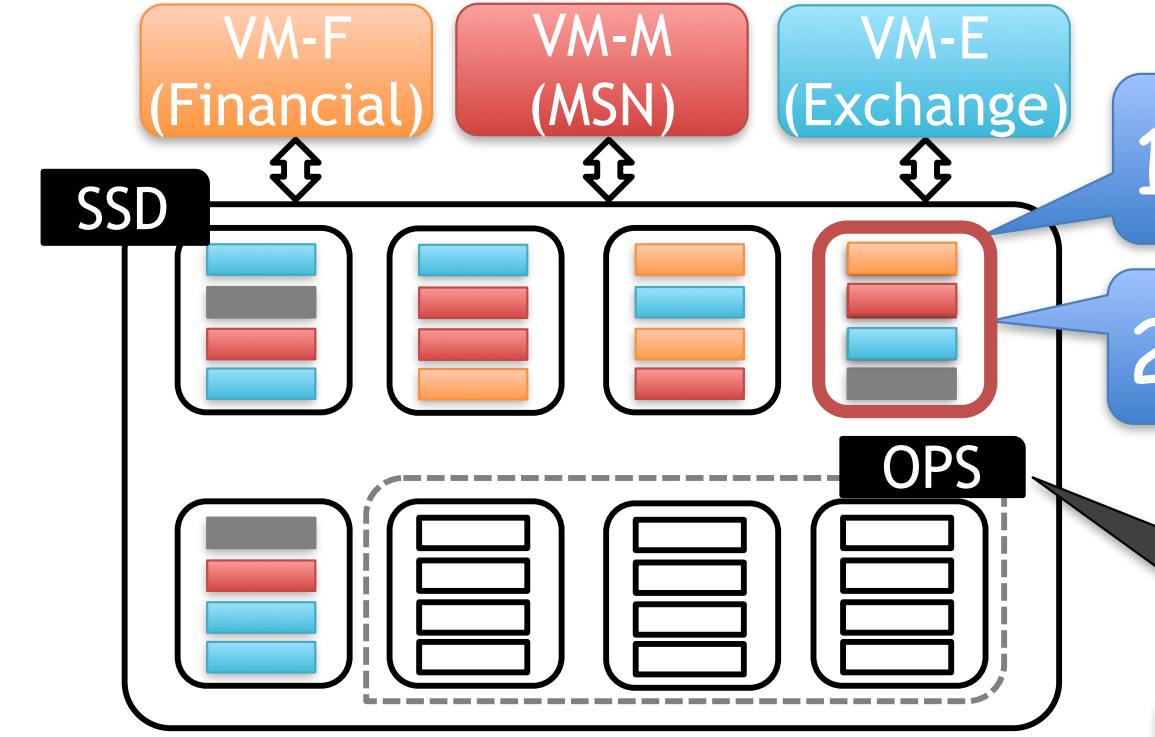
- **Within block (GC unit): mixture of data from all VMs**



Data layout of conventional SSD

Analysis #2 : Interference among VMs during GC

- Movement of data: live pages of workloads other than the one invoking GC



1) Victim block for GC

2) Pages moved to OPS

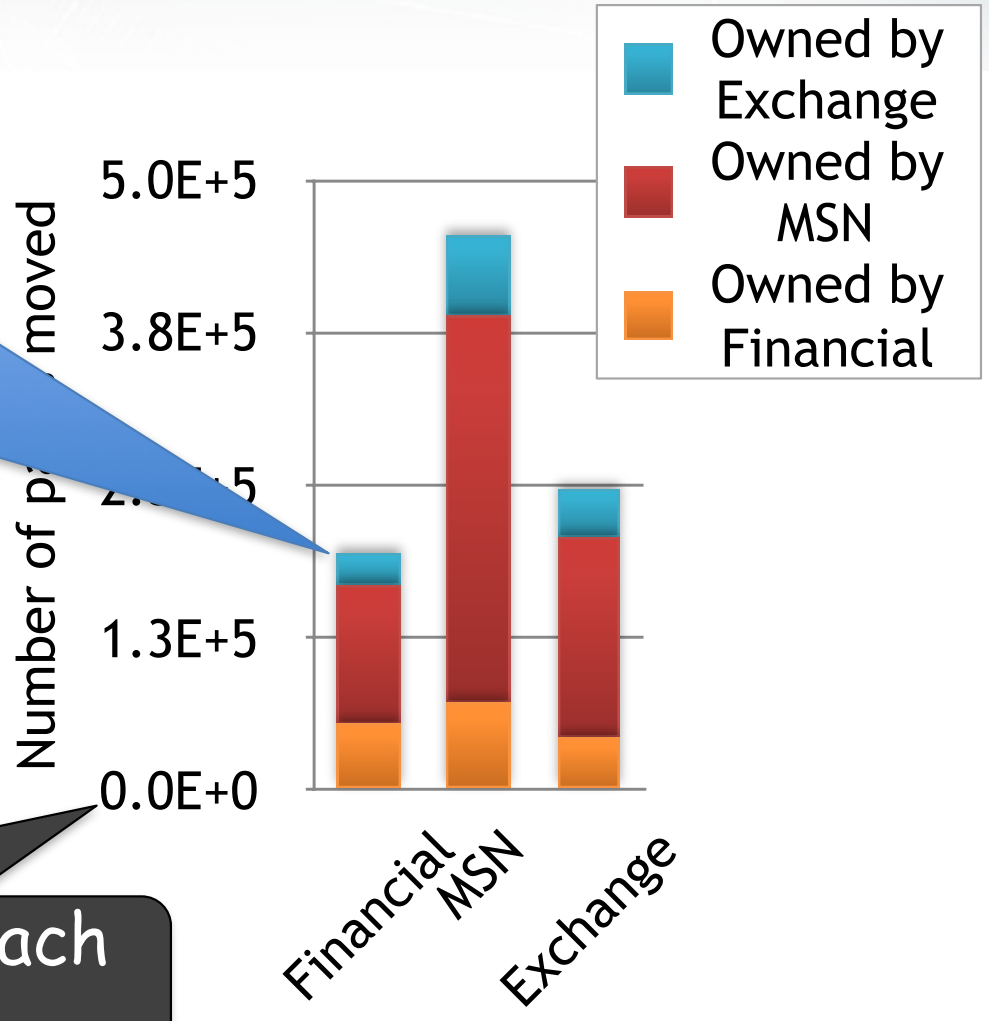
Over-Provisioned Space (OPS)
- reserved space for write reqs.
- used for garbage collection (GC)

Data layout of conventional SSD

Analysis #3: Work induced by other VMs

- **From one VM's viewpoint: doing unnecessary work induced by other workloads**

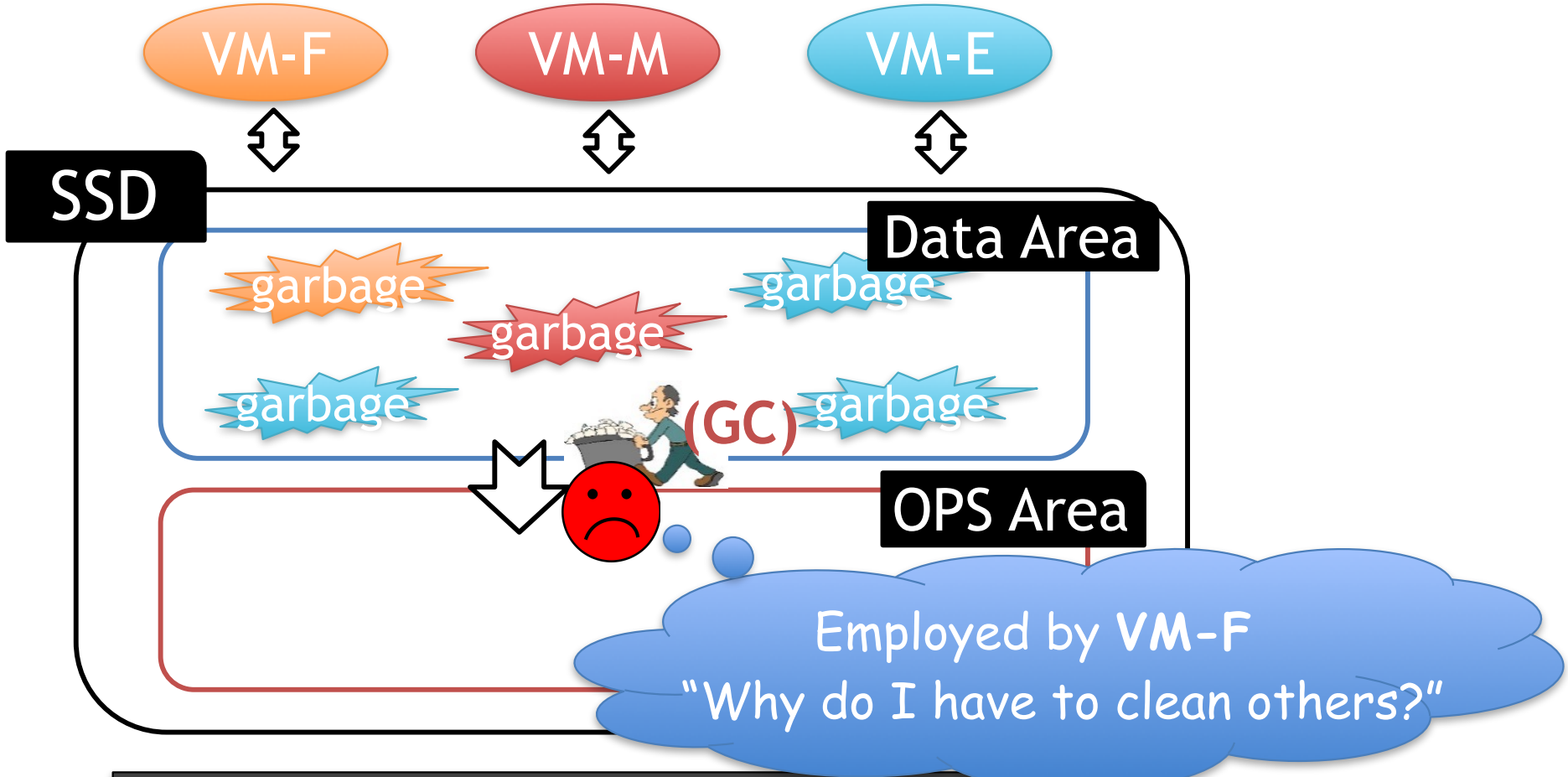
While executing VM-F (Financial) workload, only 30% of moved pages are its own



Number of pages moved for each workload during GC

More Closely

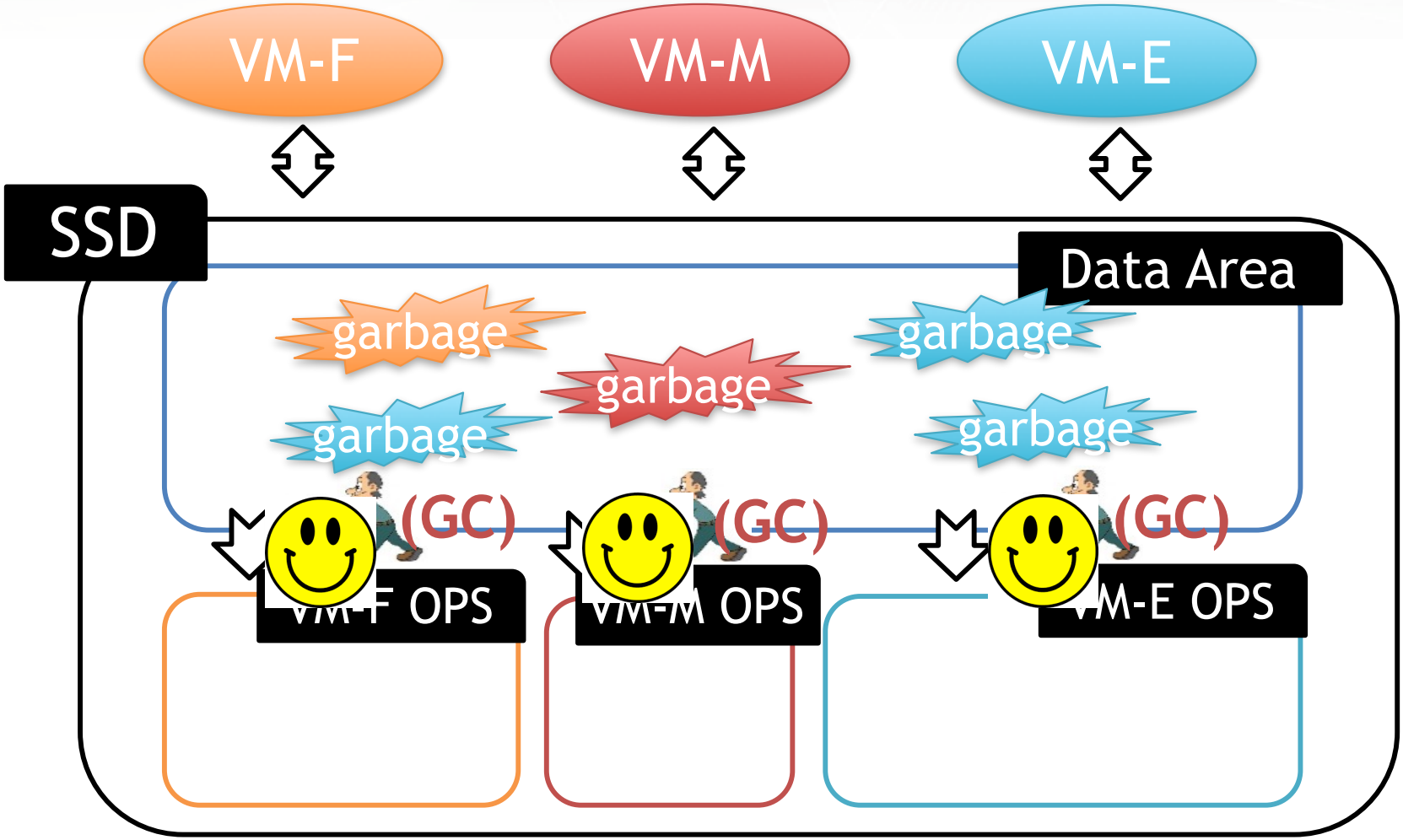
- GC leads to interference problem among VMs
- GC operation employed by one VM is burdened with other VM's pages



GC operation in conventional SSD

Avoiding Interference

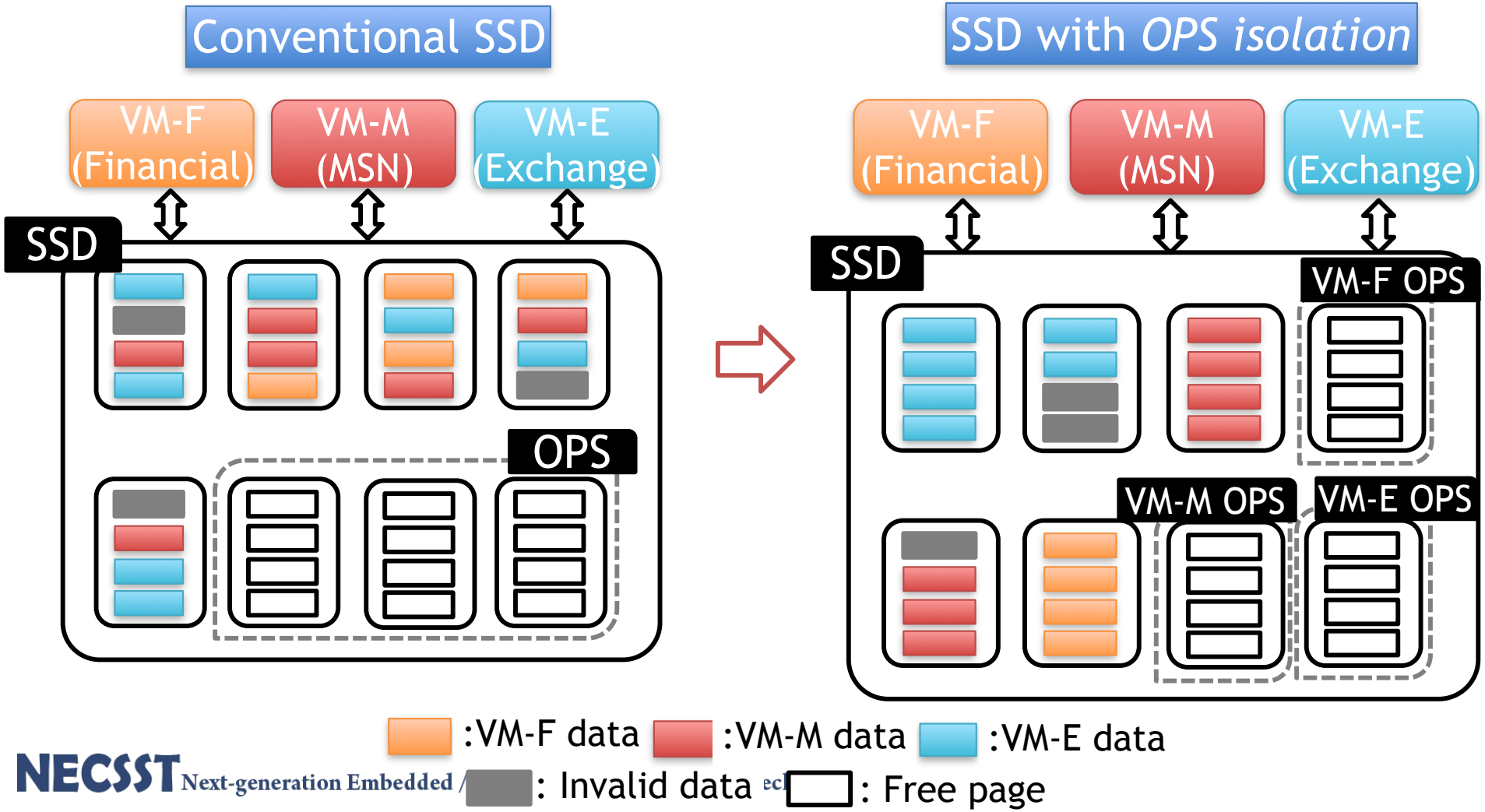
- Cost of GC is major factor in SSD I/O performance
- Each VM should pay only for its own GC operation



Proposed scheme: OPS isolation

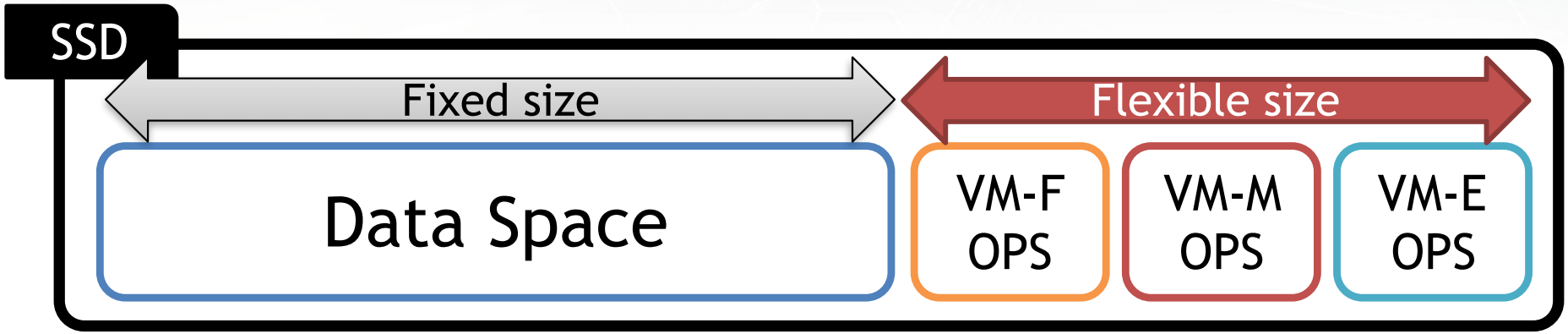
- **Dedicate flash memory blocks, including OPS, to each VM separately when allocating pages to VMs**

  → **Prevent interference during GC**



VM OPS Allocation

- How much OPS for each VMs to satisfy SLO?



IOPS of SSD

SSD

IOPS = 1 / (tGC + tPROG + tXfer)

Constant value

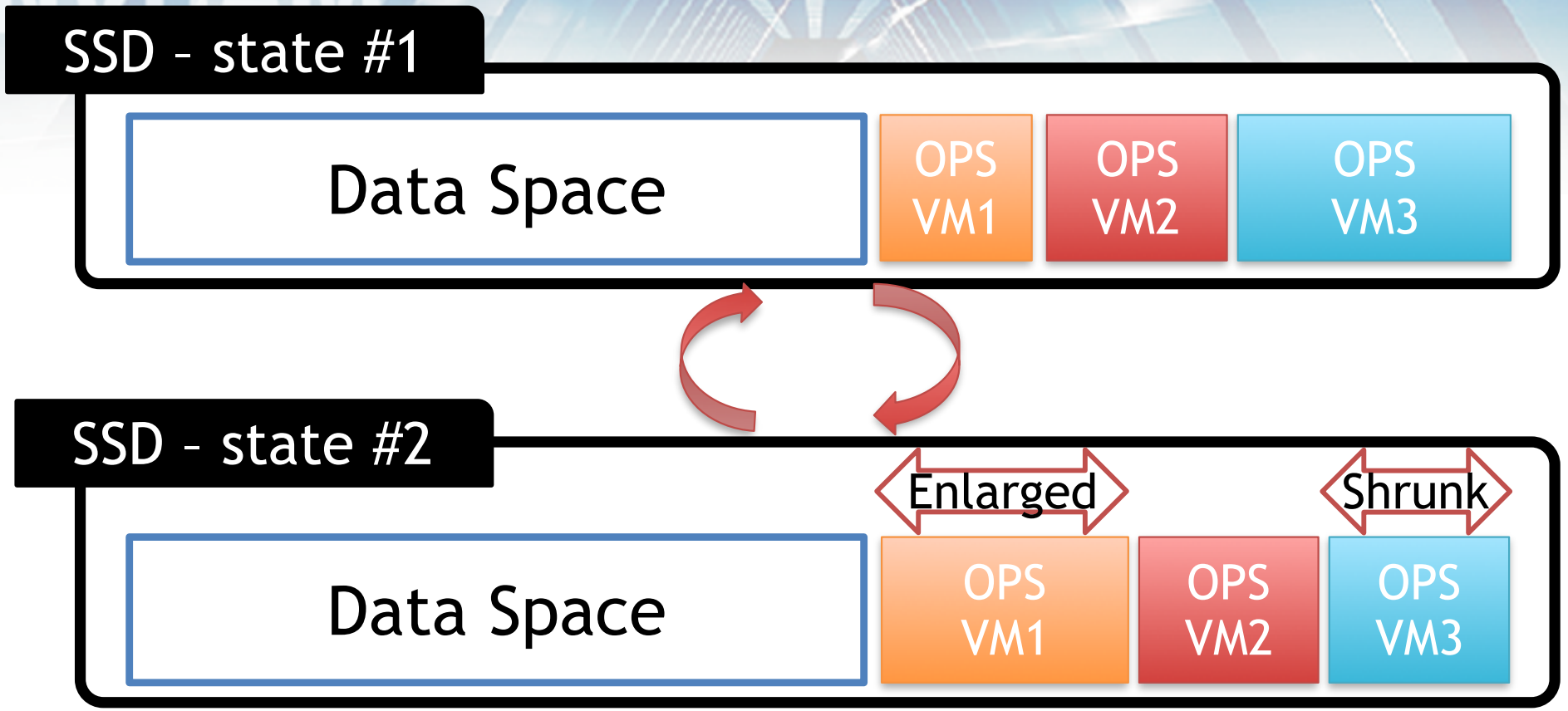
Constant value

Variable value
(Crucial factor for IOPS)

Determined by OPS size

Parameter	Meaning
tGC	Time to GC (depends on utilization (u) of **victim block at GC**)
tPROG	Time for programming a page (constant value)
tXfer	Time for transferring a page (constant value)

How to meet SLO (IOPS) of each VM?
: Dynamically adjust OPS



IOPS of state #2

IOPS of VM1 = Prev. IOPS + Δ
IOPS of VM2 = Prev. IOPS
IOPS of VM3 = Prev. IOPS - Δ

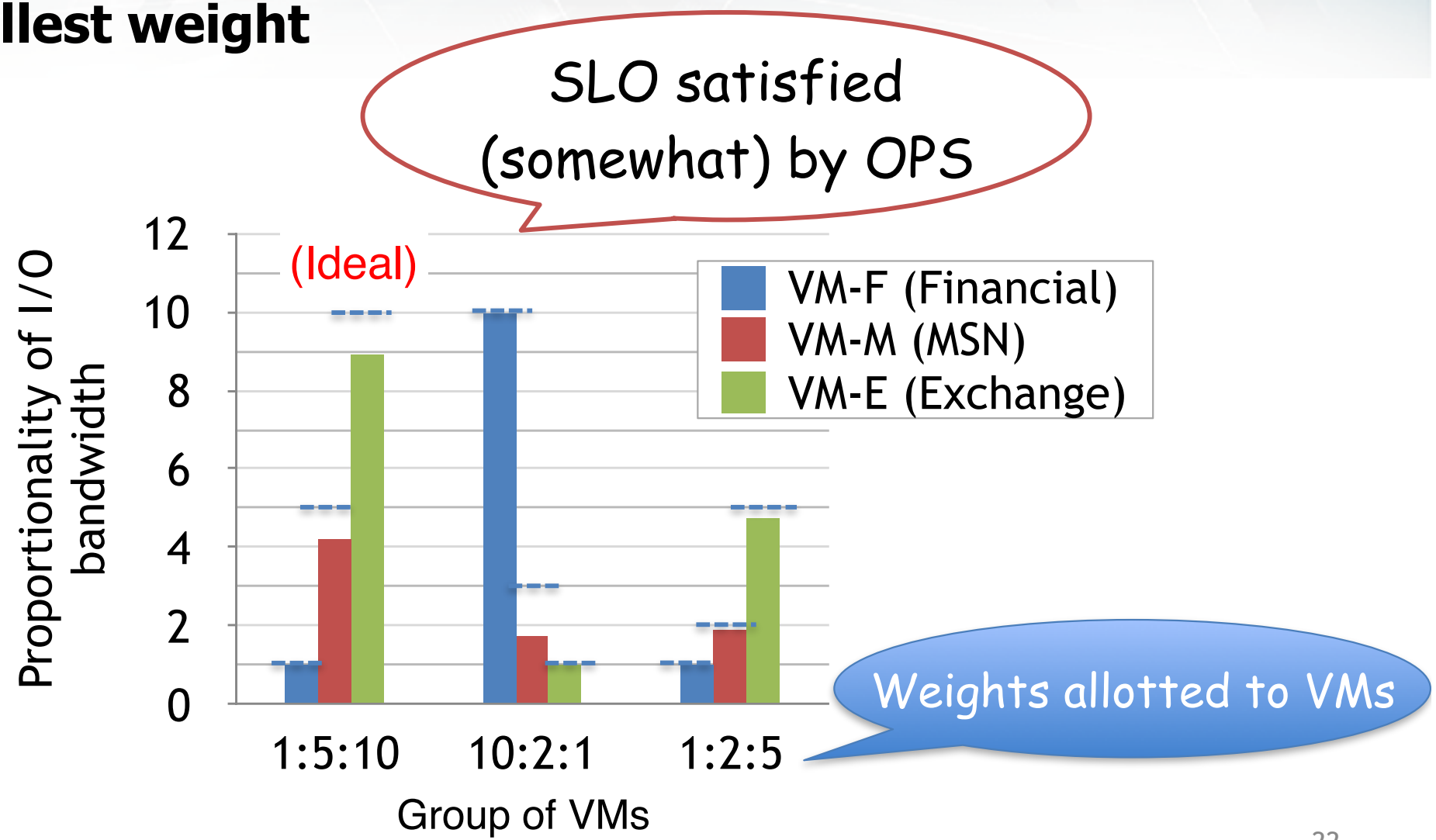
Evaluation of OPS isolation

- **Evaluation environment**
  - SSD simulator: DiskSim SSD Extension
    - FTL: Page-mapped FTL
    - GC: Greedy policy
    - Aged state SSD
  - Workloads:
    - Financial, MSN, and Exchange
      - Traces captured as VMs run concurrently on real system
  - Host interface
    - Tags of VM ID are informed to SSD

Parameter	Description
Page size	4KB
Block size	512KB
Page read	60us
Page write	800us
Block erase	1.5ms
Xfer latency (Page unit)	102us
OPS	5%

Results

- **x-axis: groups of VMs that are executed concurrently**
- **y-axis: proportionality of I/O bandwidth relative to smallest weight**



Conclusion of Part I

- **Performance SLOs can not be satisfied with current commercial SSDs**
  - Garbage collection interference between VMs
- **Propose OPS isolation**
  - Allocate blocks to VM in isolation via OPS allocation
  - Do not allow mix of pages in same block
  - Size of OPS is dynamically adjusted per VM
- **OPS isolation: "effective" in providing performance SLOs among competing VMs**

- Is OPS isolation satisfactory?

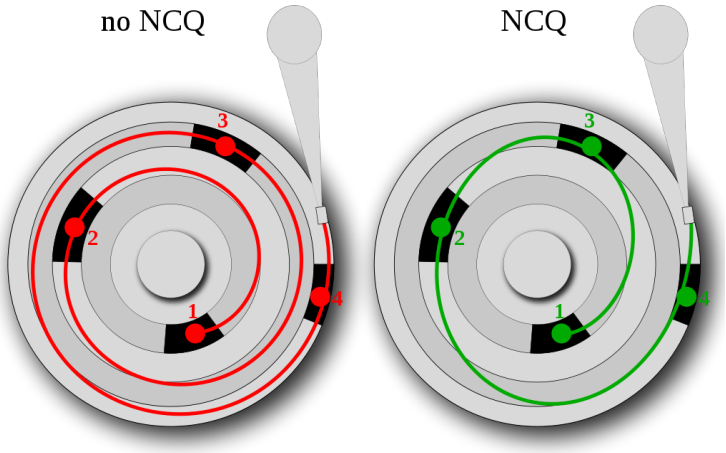
- Is OPS isolation satisfactory?
- What about other resources?
  - **Channels, Buffer, NCQ**, etc

Outline

- **Part 1 FAST 2015**
- **Part 2: Beyond FAST**
  - Still on-going

SSD Components Considered

- **Channel**
  - Data bus to connect controller to flash memory package
- **Write cache**
  - Small amount of DRAM used as volatile cache
- **NCQ**
  - SATA technology used to internally optimize the execution order of received disk read and write commands



From: wikipedia

Devices

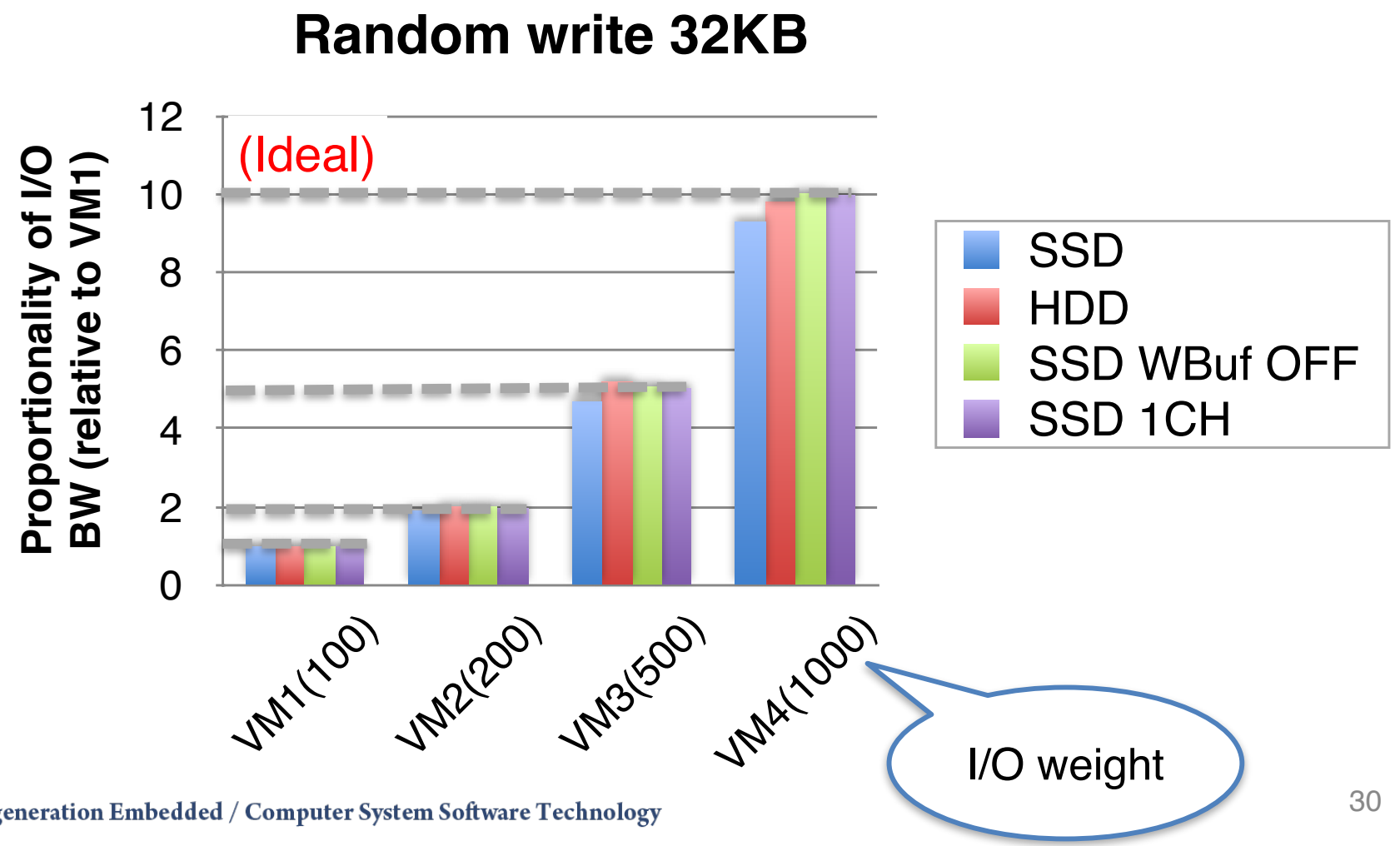
- **SSD**
  - Samsung 850 PRO 256GB
- **HDD**
  - Seagate Barracuda 1TB 7200 RPM
- **SSD WBuf OFF**
  - Samsung 850 PRO 256GB
  - SSD with disabling write cache
- **SSD 1CH**
  - Commercial Samsung SSD
  - Set to use 1-channel

Workloads

- **Micro benchmark**
  - Random write
- **Macro benchmark**
  - Fileserver workload from fio benchmark
- **Traces**
  - Proj trace from MSR Cambridge
  - Exchange trace from MS corporate mail

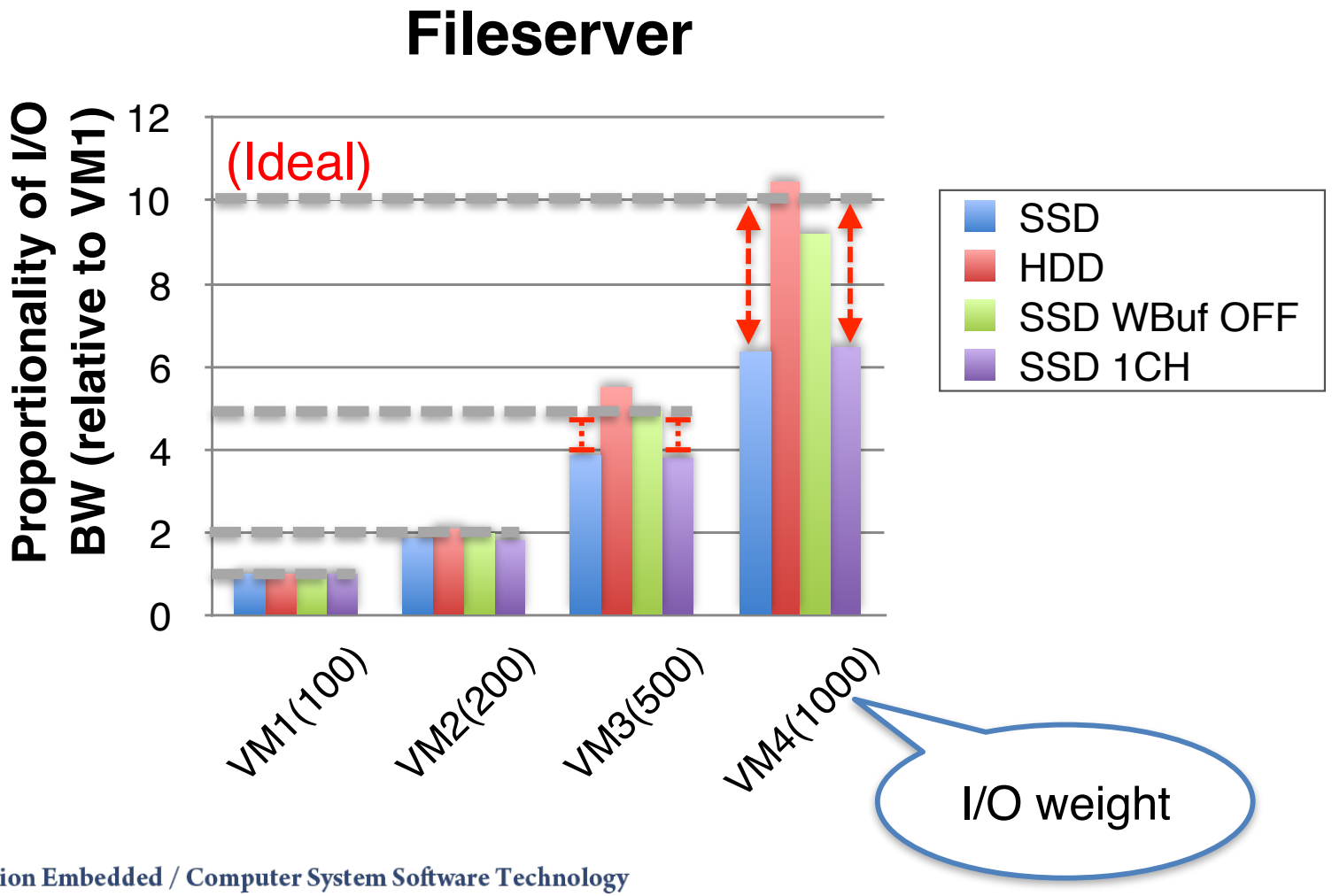
Channel Parallelism & Write Cache

- **Random writes**
  - proportionality, generally, close to I/O weight

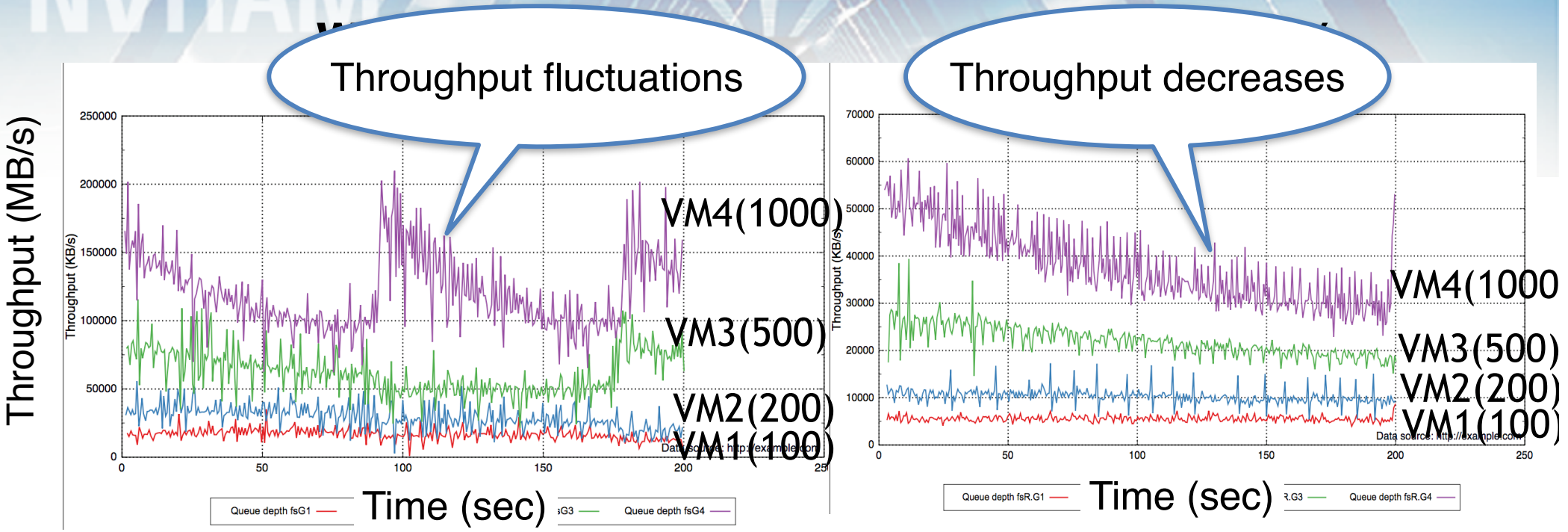


Channel Parallelism & Write Cache

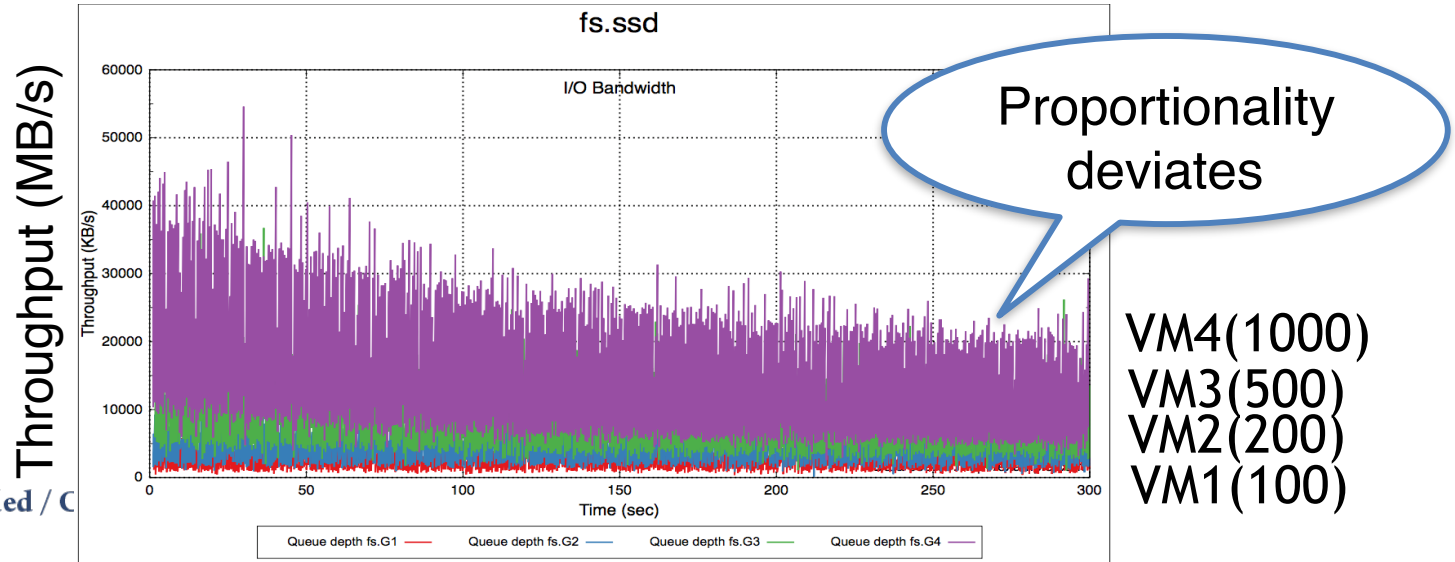
- **Fileserver**
  - proportionality deviates for SSD and SSD 1CH



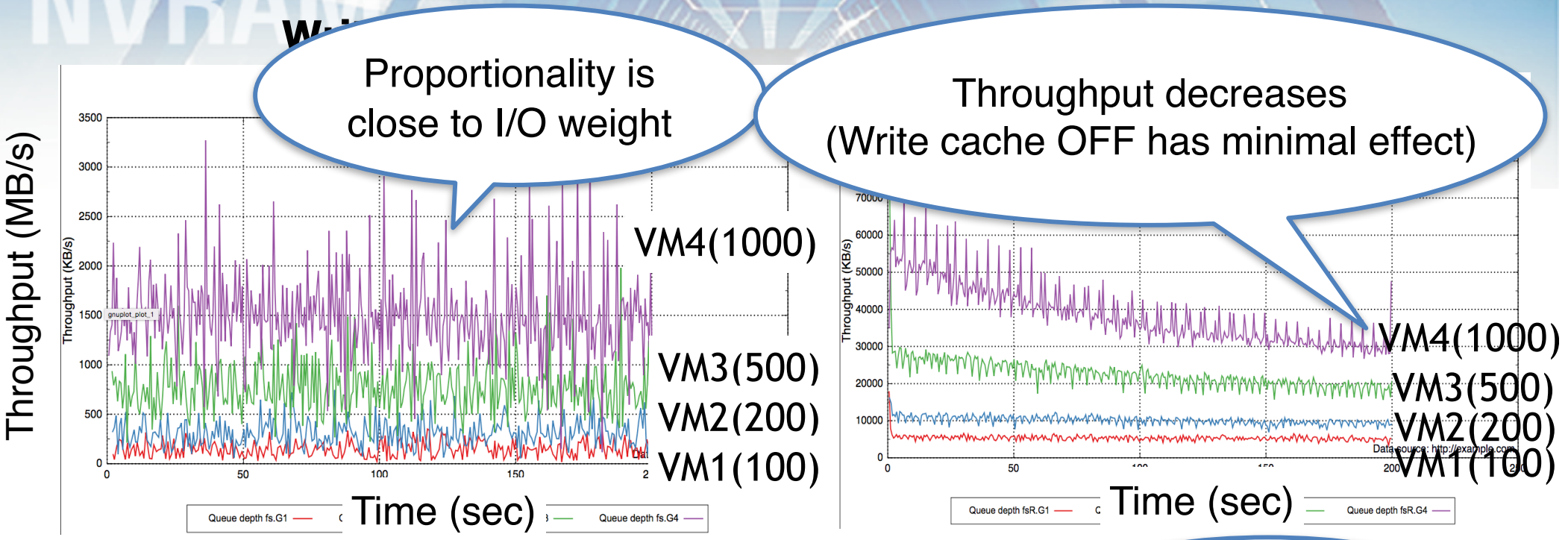
Time Series Analysis of Fileserver on SSD



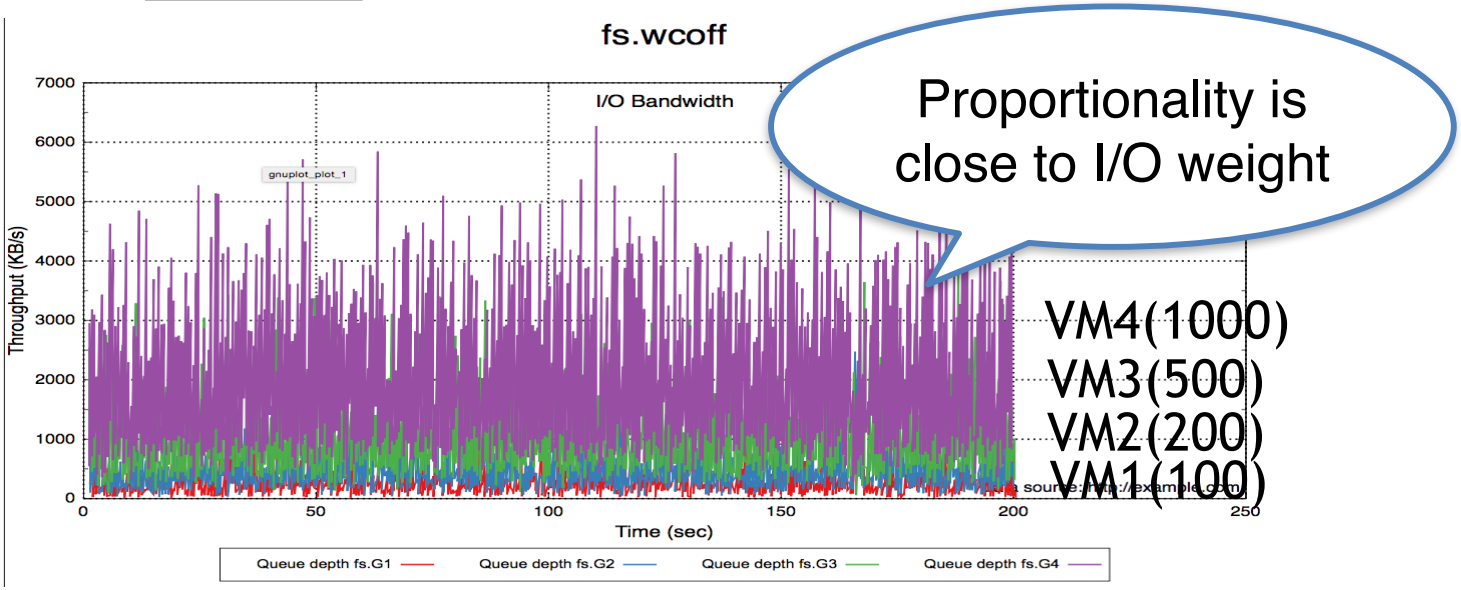
Read: 40%



Time Series Analysis of Fileserver on SSD with Write Cache OFF



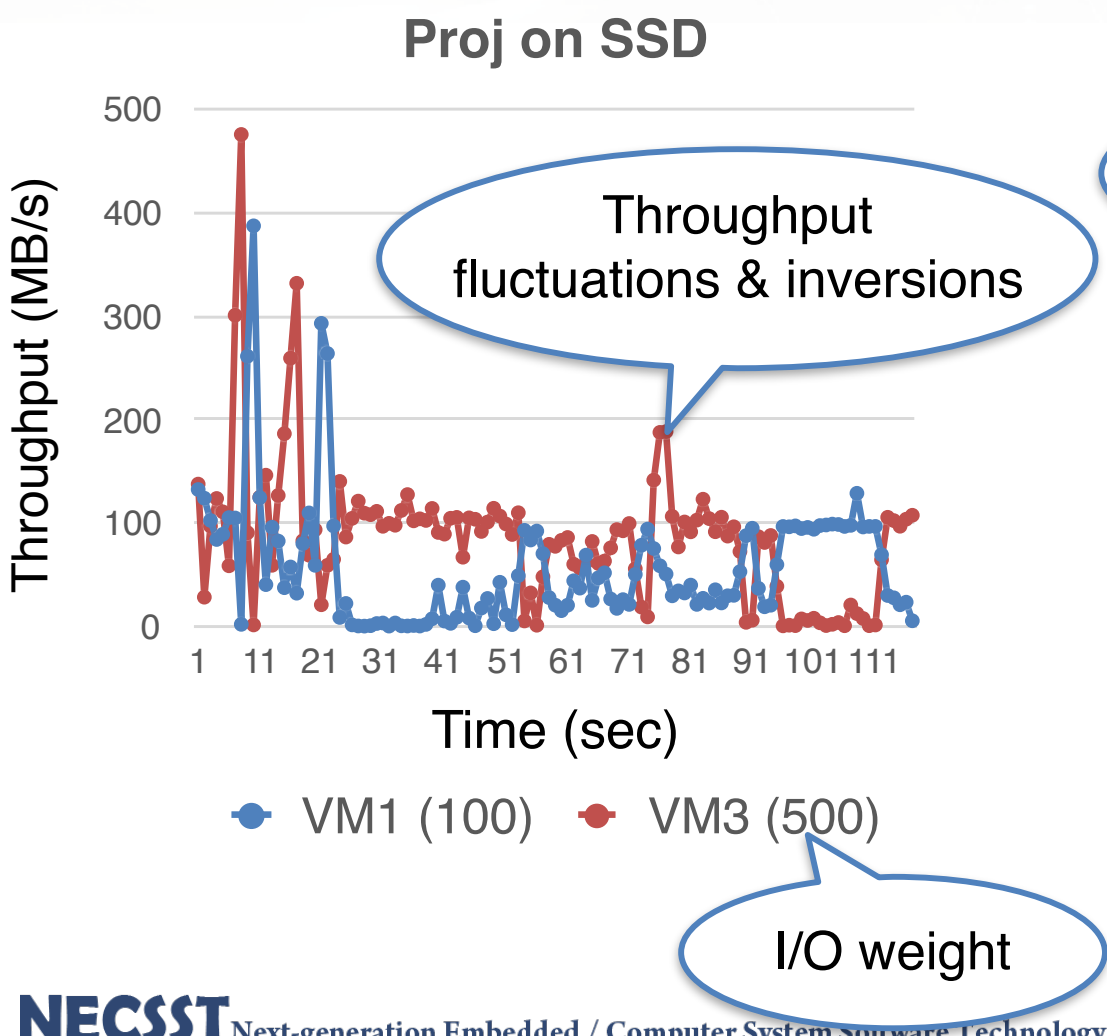
Read: 40%

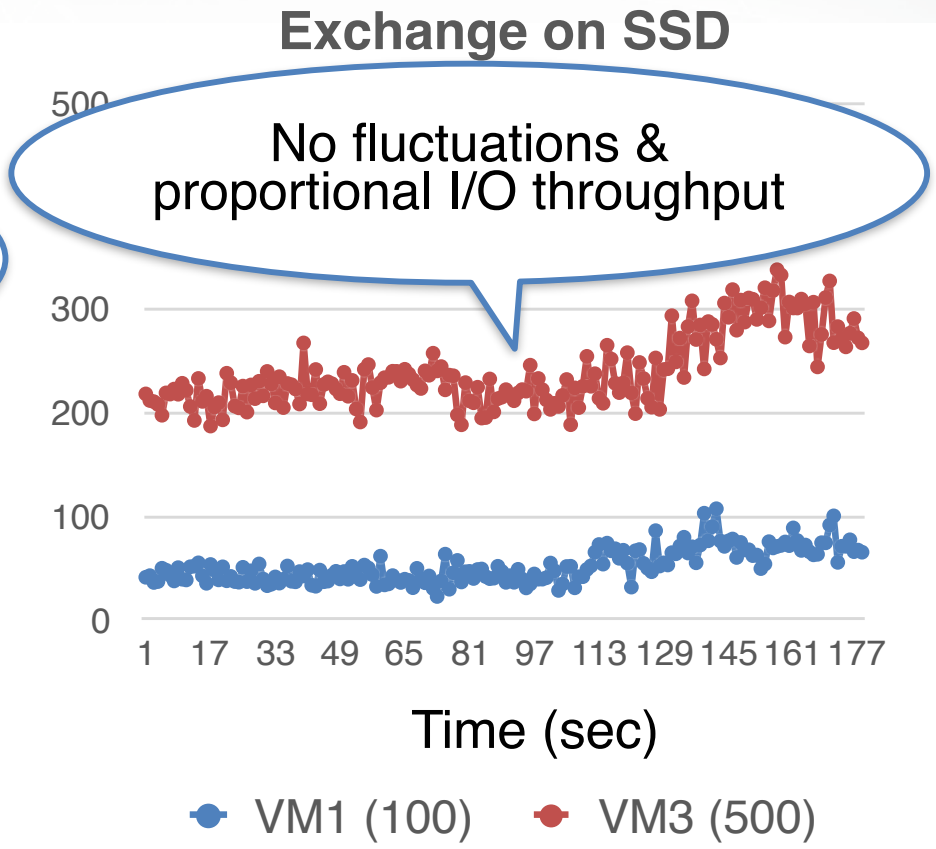


NECSST Next-generation Embedded

Effect of NCQ

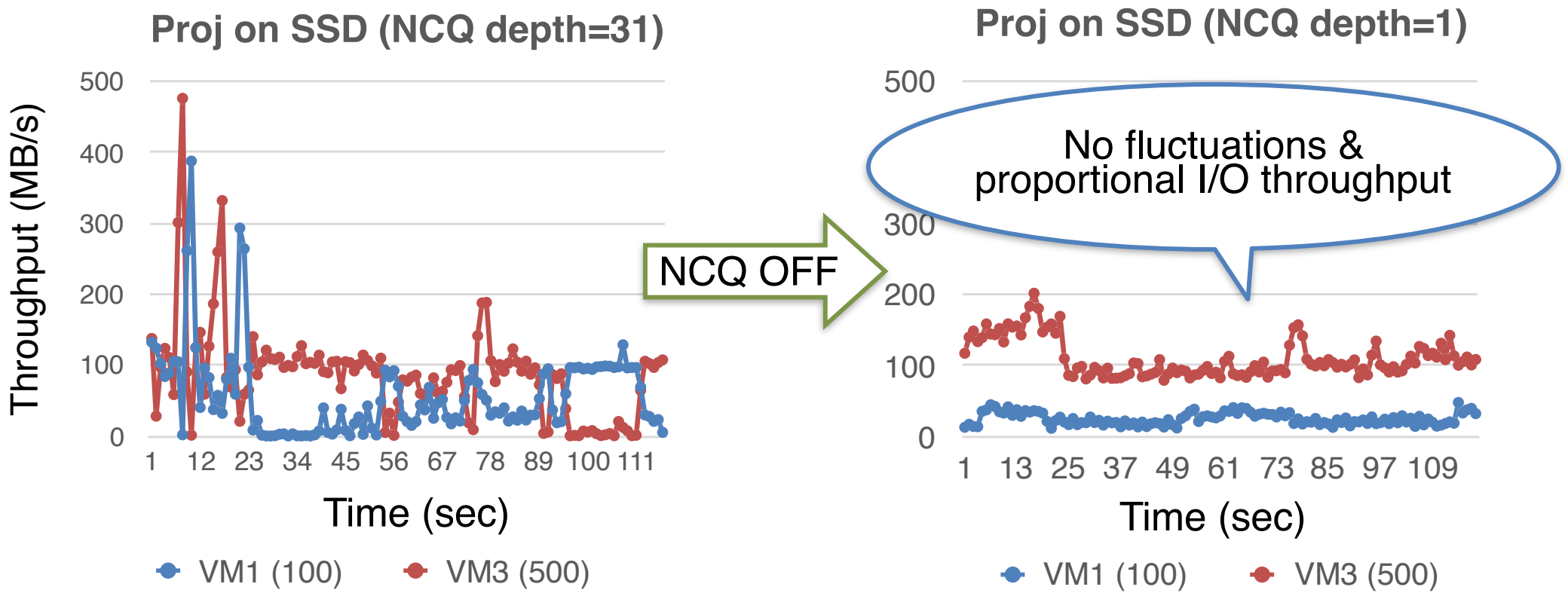
- **Time series analysis with 2 VMs**





Proj Trace on SSD with NCQ OFF

- Effect of NCQ in SSD



Conclusion

- **Conducted analysis of I/O SLO through examining major components of SSD**
  - GC, Write cache, Channel, and NCQ
- **SSD components affect I/O SLO under various workloads**
- **Future work**
  - Analyze OS components for I/O SLO on SSD

Thank you!!!