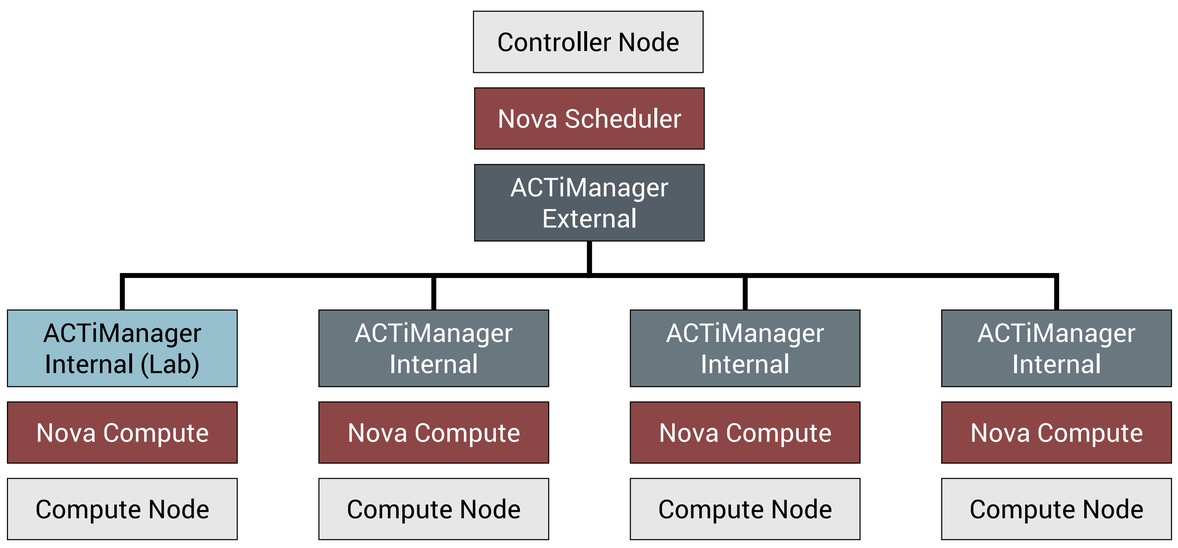
ACTiManager:
An end-to-end interference-aware cloud resource manager

Stratos Psomadakis, Stefanos Gerangelos, Dimitrios Siakavaras, Ioannis Papadakis, Marina Vemmou, Aspa Skalidi,
Vasileios Karakostas, Konstantinos Nikas, Nectarios Koziris, Georgios Goumas
National Technical University of Athens, ICCS
actimanager@cslab.ece.ntua.gr

1. Motivation & Background

- **The Problem**: Cloud Service Providers host numerous workloads on their facilities
  - Resource sharing → interference, performance unpredictability (e.g., Openstack) [1]
- **Current Solution**: High Quality of Service: simplistic resource management policies → low resource utilization (e.g., Socket Isolation) [2]
- **Can we do better?**
  - *Interference avoidance*: predict the potential to create or suffer from interference [3]
  - *Interference mitigation*: detect interference accurately at runtime [4]
  - *Prioritization* between latency-critical (Gold) and best effort (Silver) workloads [5]
    - Simplistic approach: avoid oversubscription for Gold VMs (Gold Not Oversubscribed - GNO)
- **Our solution**: ACTiManager - a practical, interference-aware cloud resource manager:
  - Performs both interference avoidance and mitigation, requiring no offline application profile
  - Optimizes for datacenter profit

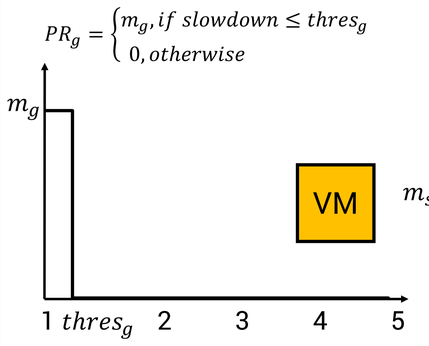
2. ACTiManager Architecture



3. VM Prioritization and Pricing

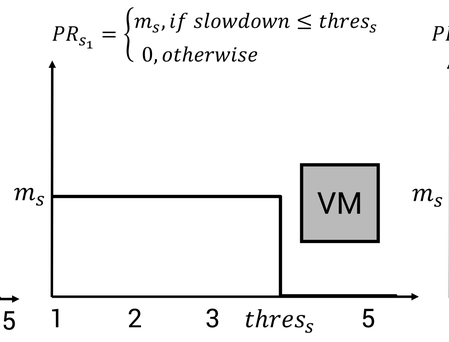
Profit Rate (PR) as a function of $Slowdown = \frac{perf_{isolation}}{perf_{attained}}$

$$PR_g = \begin{cases} m_g, if\ slowdown \le thres_g \\ 0, otherwise \end{cases}$$



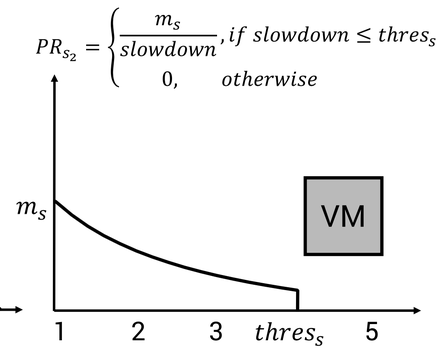
Gold VMs

$$PR_{s_1} = \begin{cases} m_s, if\ slowdown \le thres_s \\ 0, otherwise \end{cases}$$



Silver VMs (constant time)

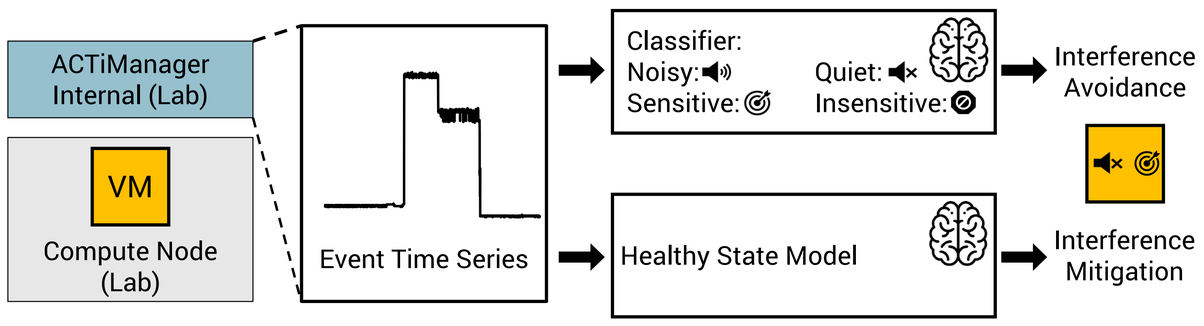
$$PR_{s_2} = \begin{cases} \frac{m_s}{slowdown}, if\ slowdown \le thres_s \\ 0, \quad otherwise \end{cases}$$



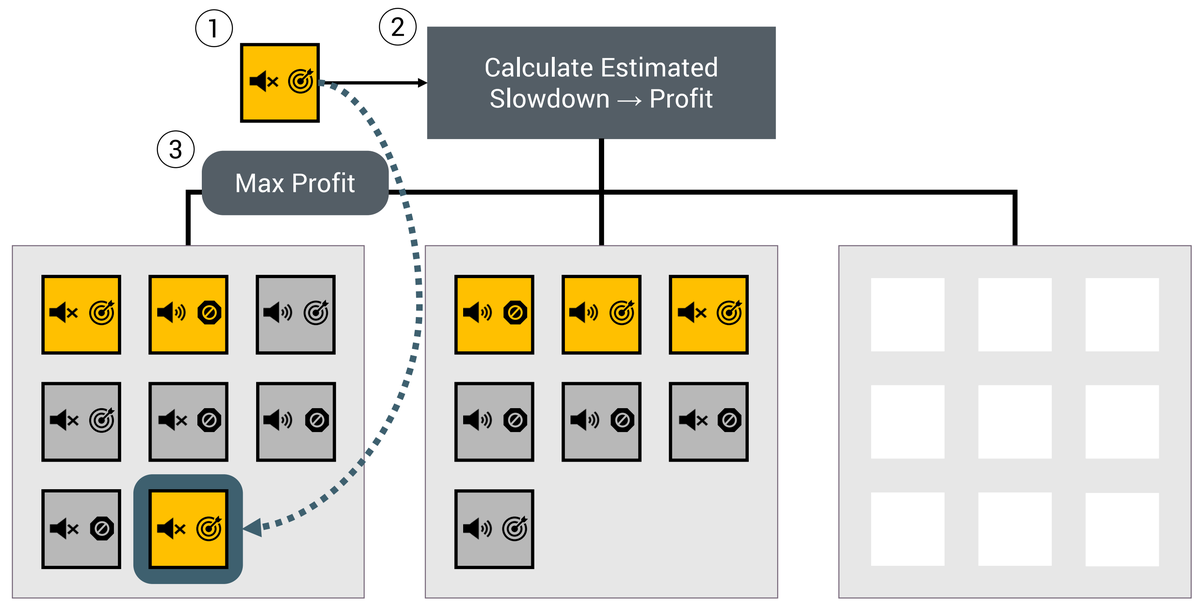
Silver VMs (constant work)

$$Profit_{VM} = Number\ of\ vCPUs * Execution\ Time * PR(slowdown)$$

4. VM Characterization



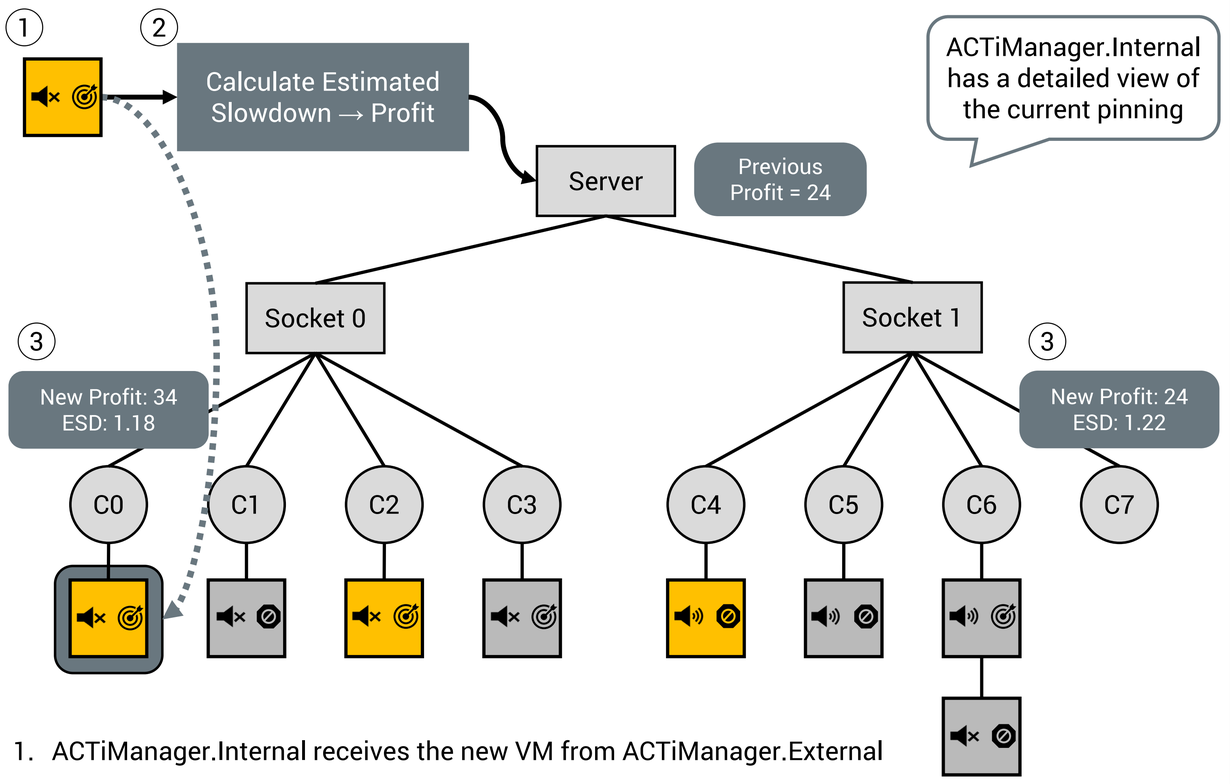
5. ACTiManager.external: Allocating VMs to Servers



1. Receives the VM from the Lab
2. Calculates the Estimated Slowdown coarsely for each server
3. Places the VM in the server with the maximum profit

\+ Handles server overload messages by migrating VMs

6. ACTiManager.internal: Allocating VMs within servers

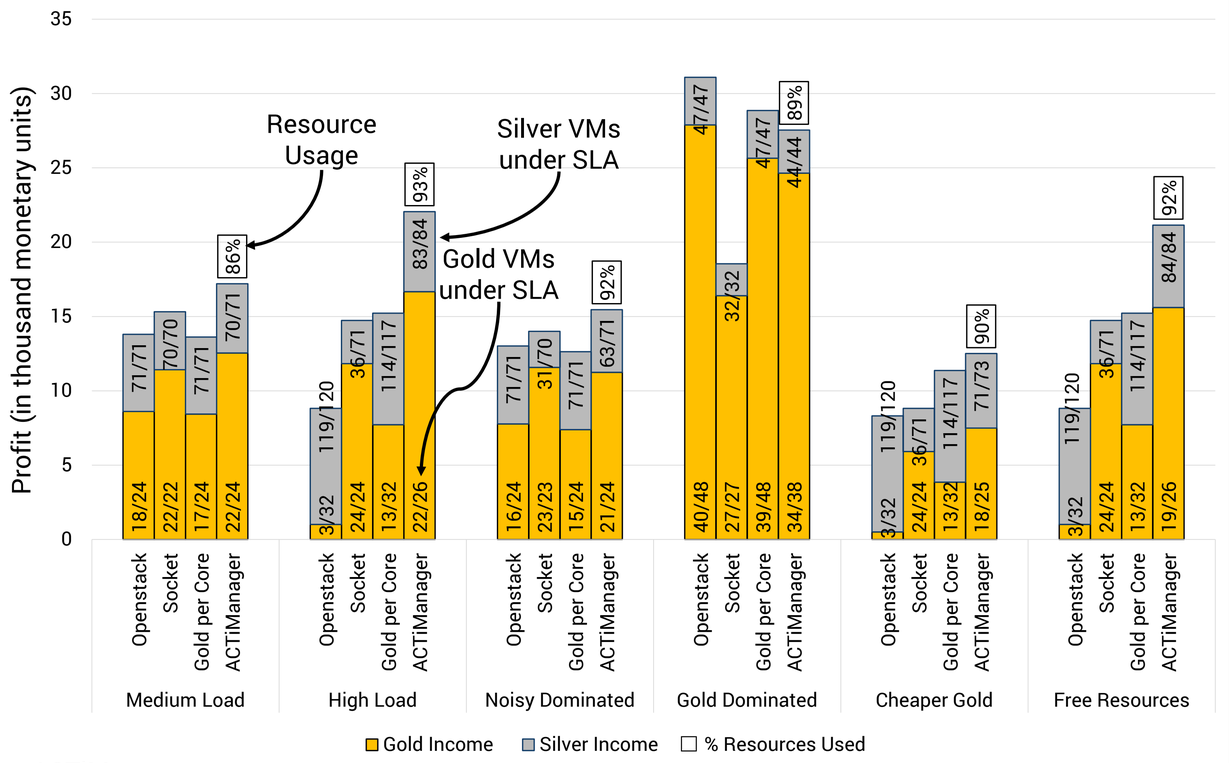


1. ACTiManager.Internal receives the new VM from ACTiManager.External
2. Calculates the estimated slowdown for all the server's cores
3. Selects the core with the maximum total profit

\+ Monitors the VMs and checks for performance anomalies

7. Evaluation

- 4 dual-socket (10 cores per socket) server cluster, 16 benchmarks from the SPEC 2006 suite.



- ACTiManager:
  - Enforces successfully the prioritization of VMs (more VMs respect their SLOs)
  - Reduces resource usage vs other policies that always use 100% of resources

References

[1] Delimitrou et al.: Paragon: QoS-aware Scheduling for Heterogeneous Datacenters. *In ASPLOS '13*
[2] Delimitrou et al.: Quasar: Resource-efficient and QoS-aware Cluster Management. *In ASPLOS '14*
[3] Haritatos et al.: A resource-centric Application Classification Approach. *In COSH@HiPEAC '16*
[4] Kannan et al.: Proctor: Detecting and Investigating Interference in Shared Datacenters. *In ISPASS '18*
[5] Lo et al.: Heracles: Improving resource efficiency at scale. *In ACM SIGARCH Computer Archit. News '15*

Acknowledgements

This research has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement no 732366 (ACTiCLOUD).