University of Colorado **Boulder**

Escra: Event-driven, Sub-second Container Resource Allocation

Greg Cusack, Maziyar Nazari, Sepideh Goodarzy, Erika Hunhoff, Prerit Oberai, Eric Keller, Eric Rozner, Richard Han

ICDCS - July 11, 2022
Bologna, Italy

Containerized Infrastructure

- Light-weight
- Rich orchestration systems
- Development workflow integration
- Strong resource isolation

Developers must make tradeoffs!

Developers must make tradeoffs

Prioritize Performance

Overallocate compute resources

Prioritize Resource Efficiency

Underallocate compute resources

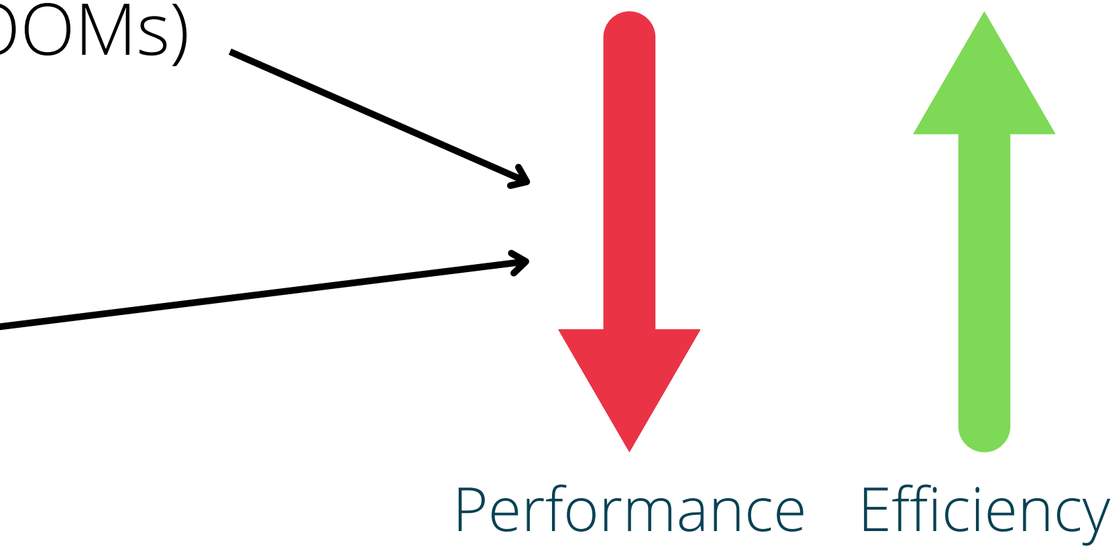
Developers must make tradeoffs

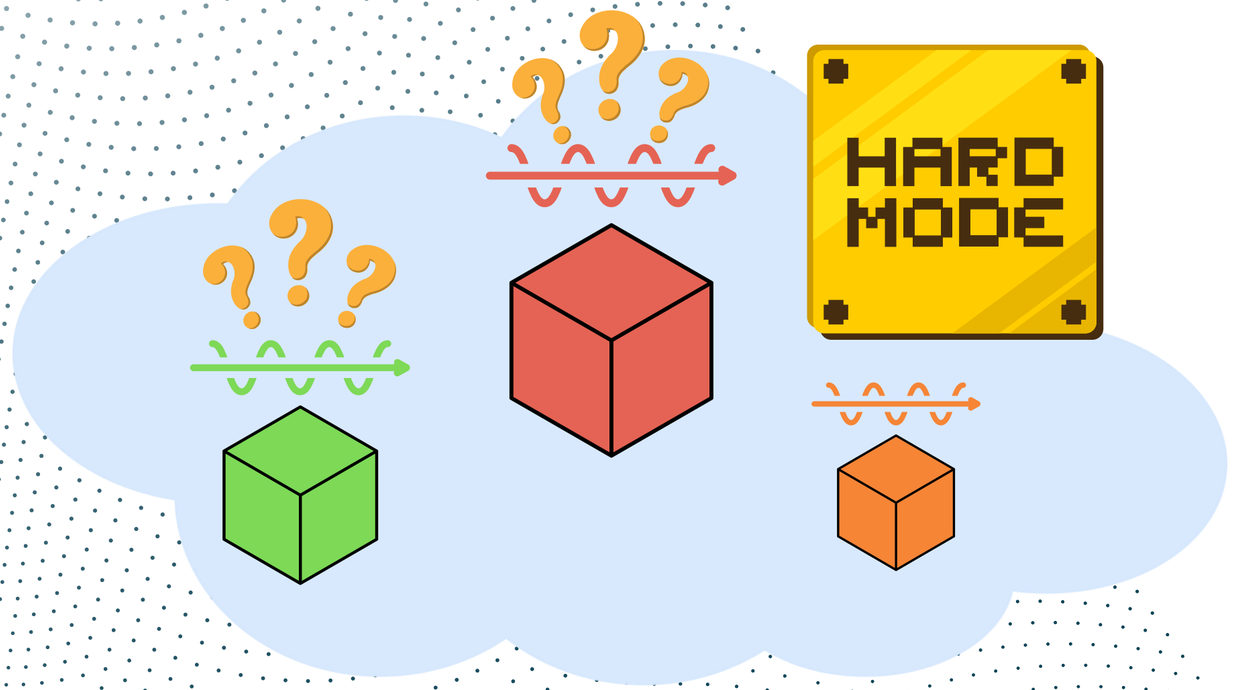
Prioritize Resource Efficiency

Underallocate compute resources

Out of Memory Events (OOMs)

CPU Throttles





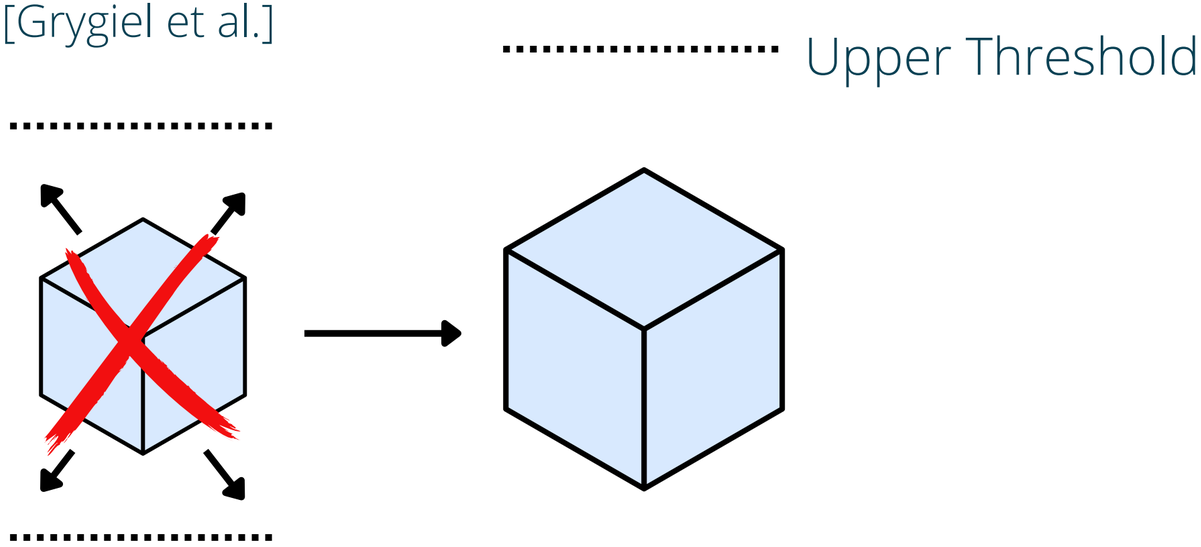
Setting Container Limits is Difficult!

- Requires representative workload
- Resource needs vary over time
- Tools aggregate usage over coarse-grained timescales
- **Result: Over provisioned containers**

Recent Work

Threshold Scaling

VPA
[Grygiel et al.]



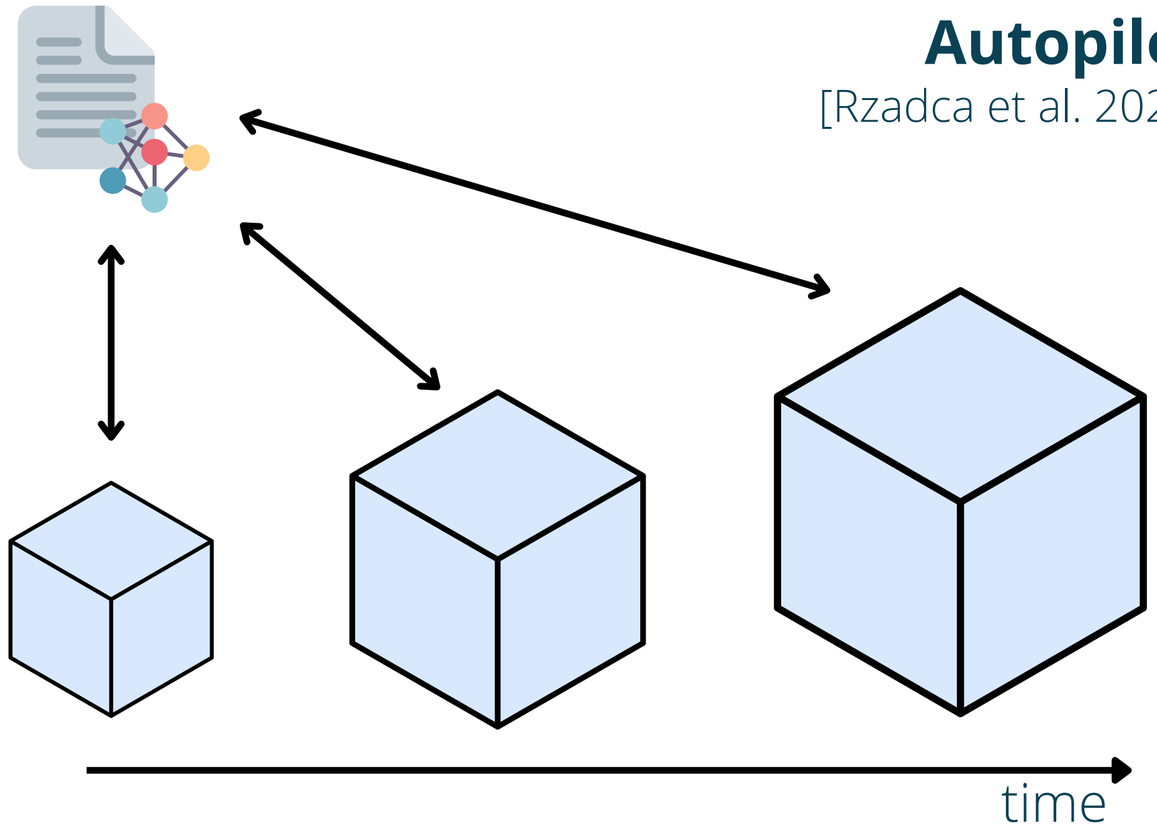
Shortcomings

- Container restart to scale
- Scale once per minute
- High slack

Recent Work

ML-based Scaling

Autopilot
[Rzadca et al. 2020]



Shortcomings

- Cannot respond to quick changes in workload
- Must allocate to maximum prediction over next 5-minute interval
- Unable to correct inaccurate predictions
- Not suitable for short-lived containers (e.g. serverless)

Key Insight

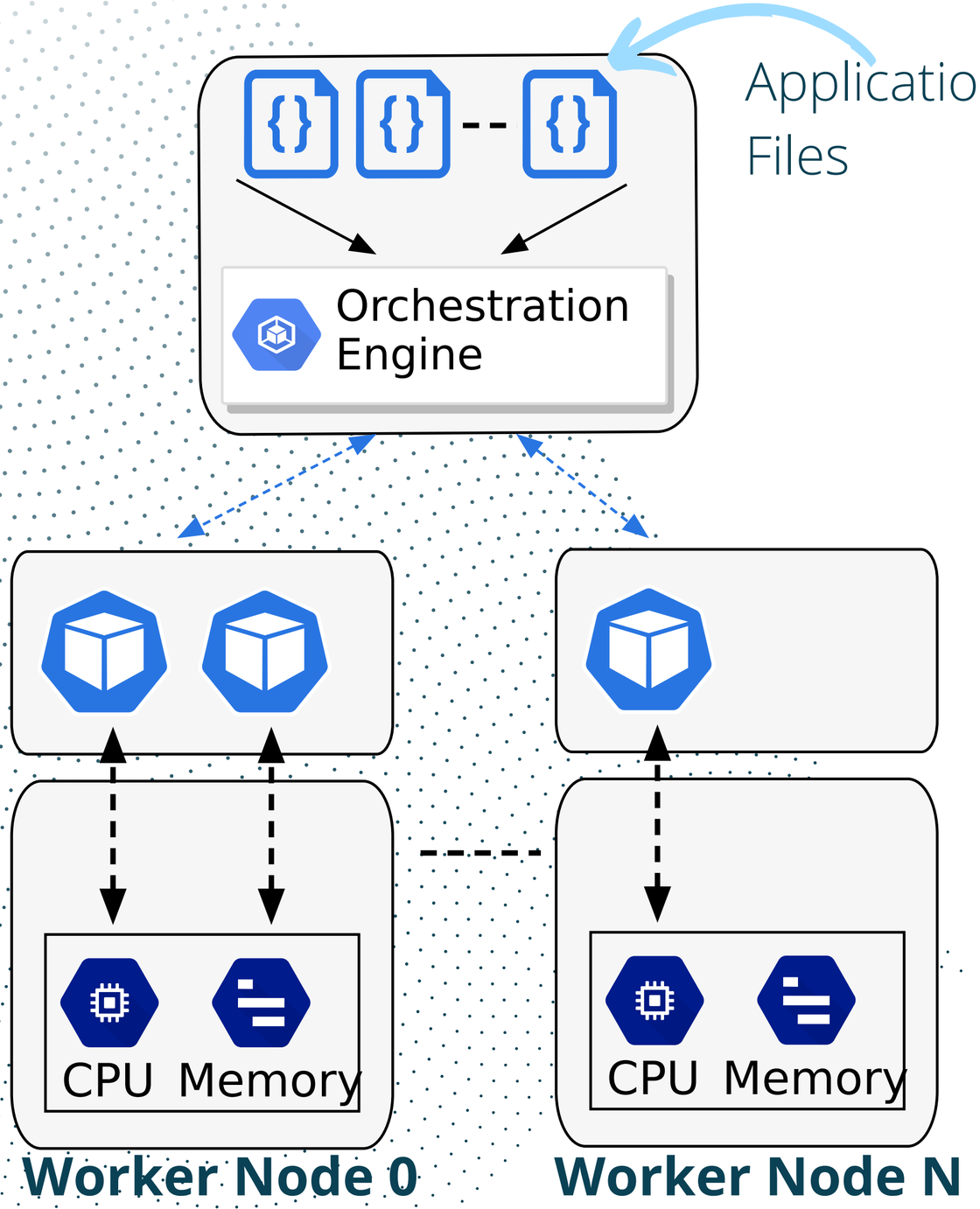
If we can make resource allocation fine grained, we don’t need to predict; we can react to workload changes.

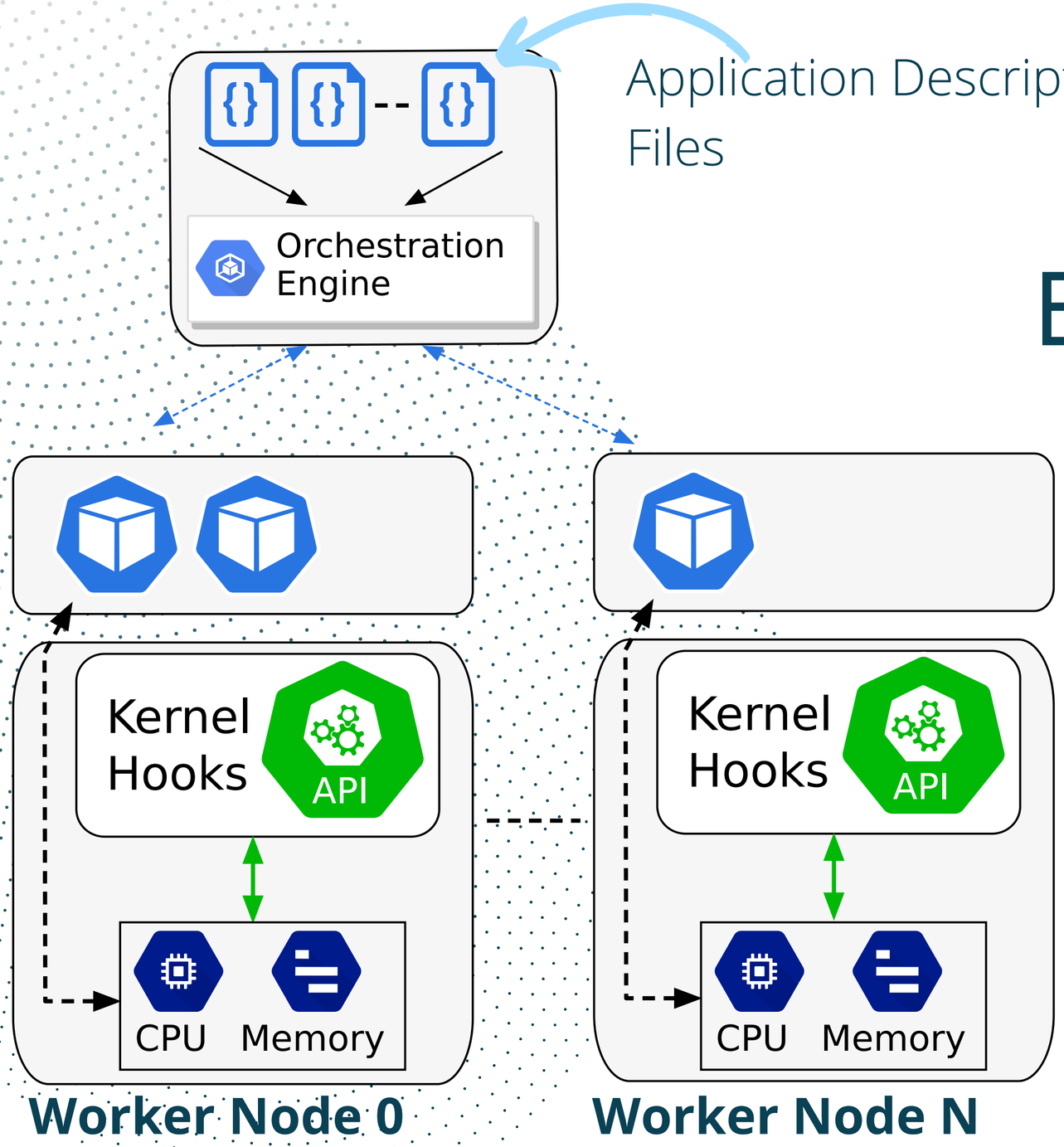
Escra

- Distributed resource allocation
- Sub-second interval scaling within and across hosts
- OOM prevention and scaling
- Immediate CPU throttle response
- No performance penalty

University of Colorado **Boulder**

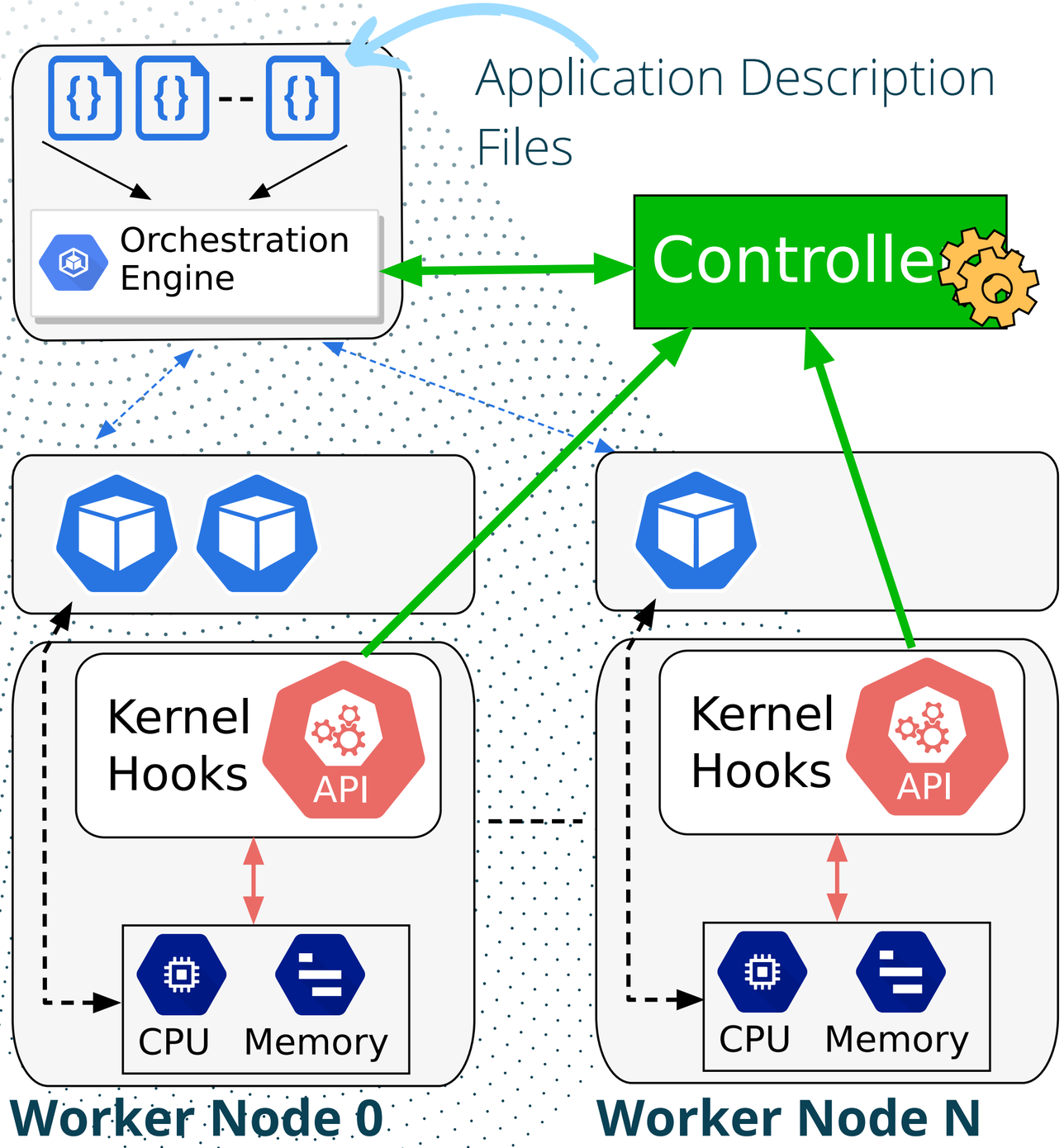
Simplified Cloud Orchestration Architecture





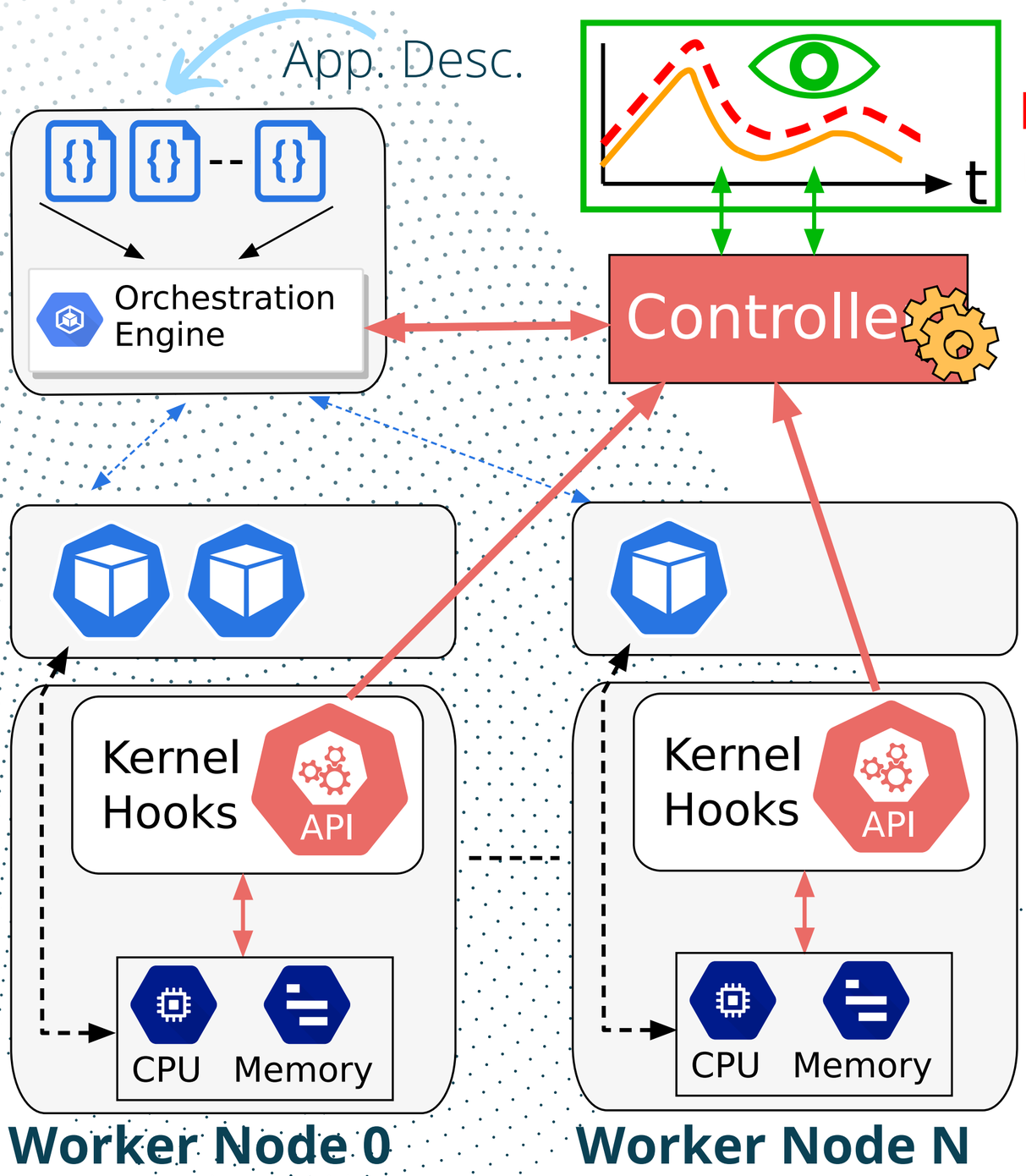
Escra Architecture

- **Kernel Hooks**



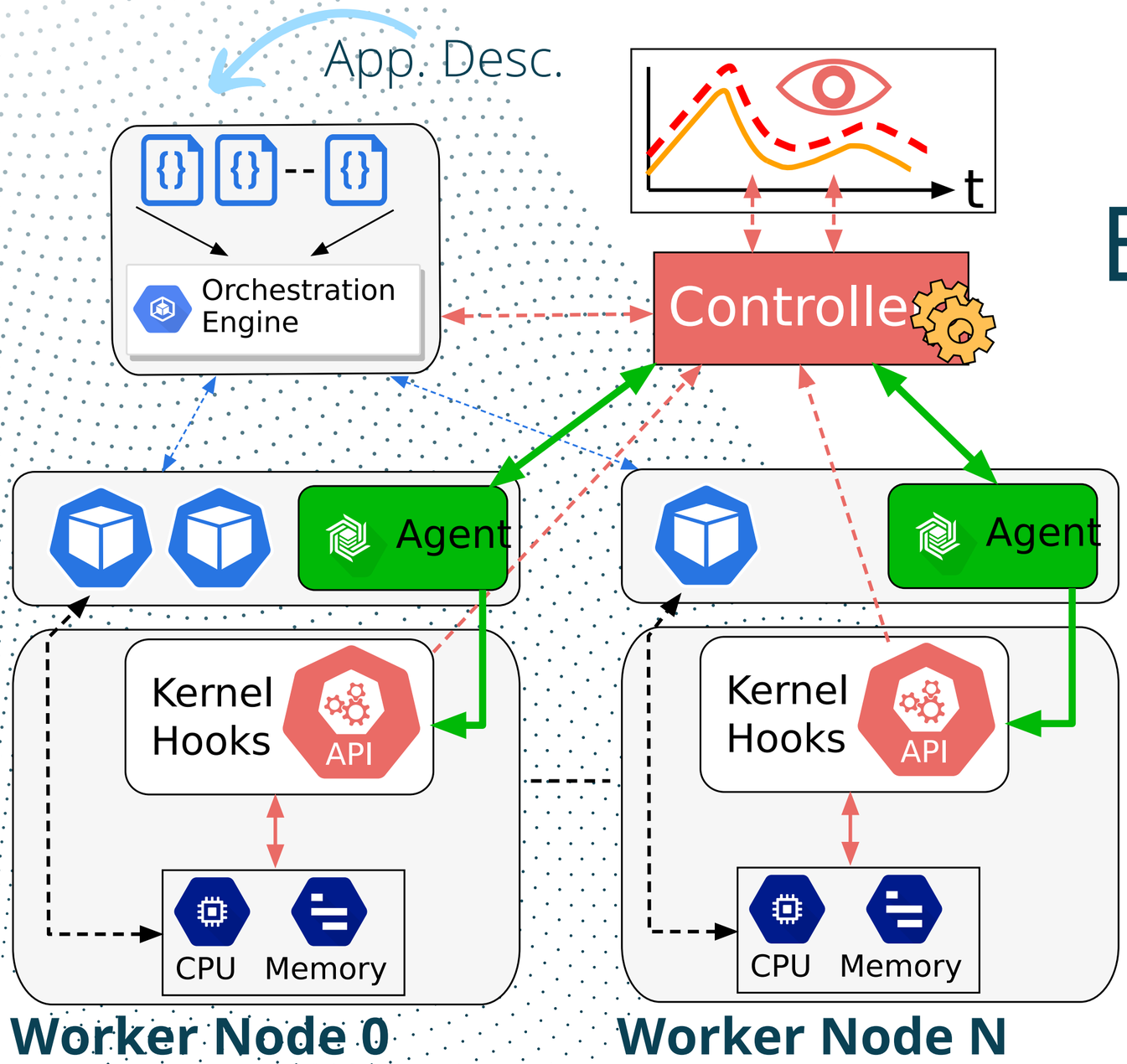
Escra Architecture

- Kernel Hooks
- **Controller**



Escra Architecture

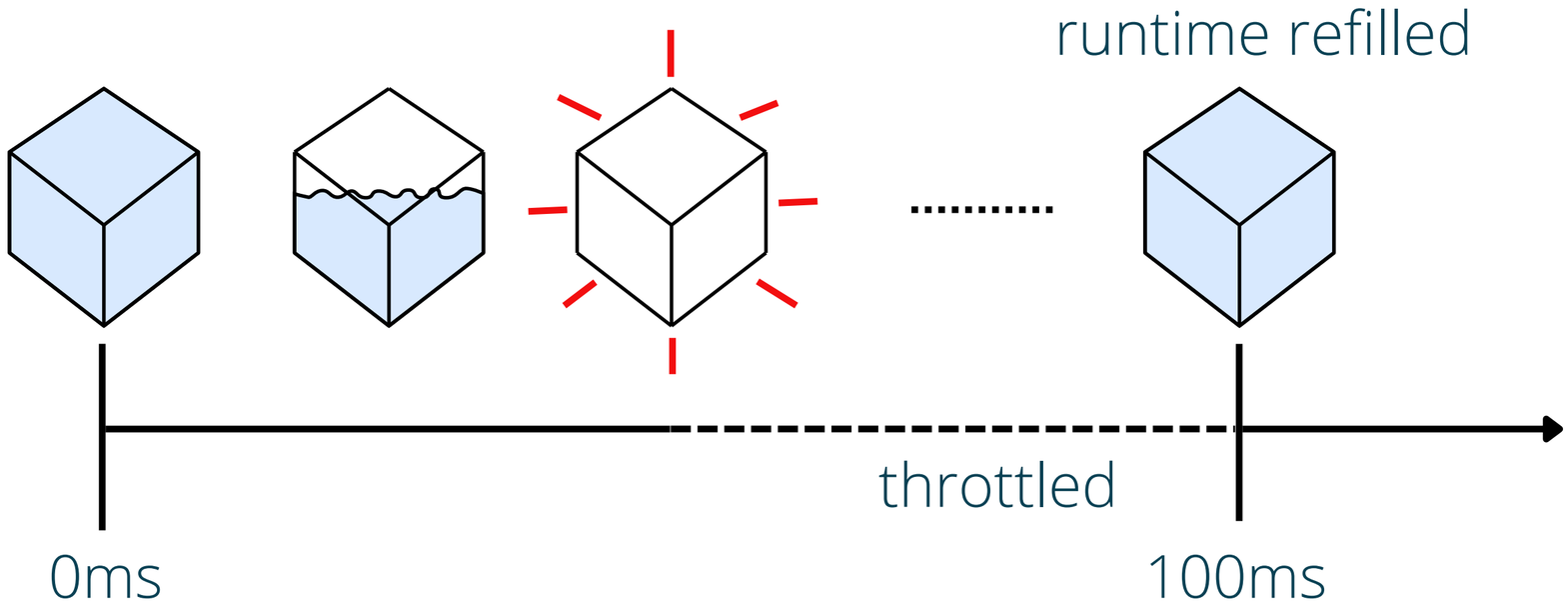
- Kernel Hooks
- Controller
- **Resource Allocator**



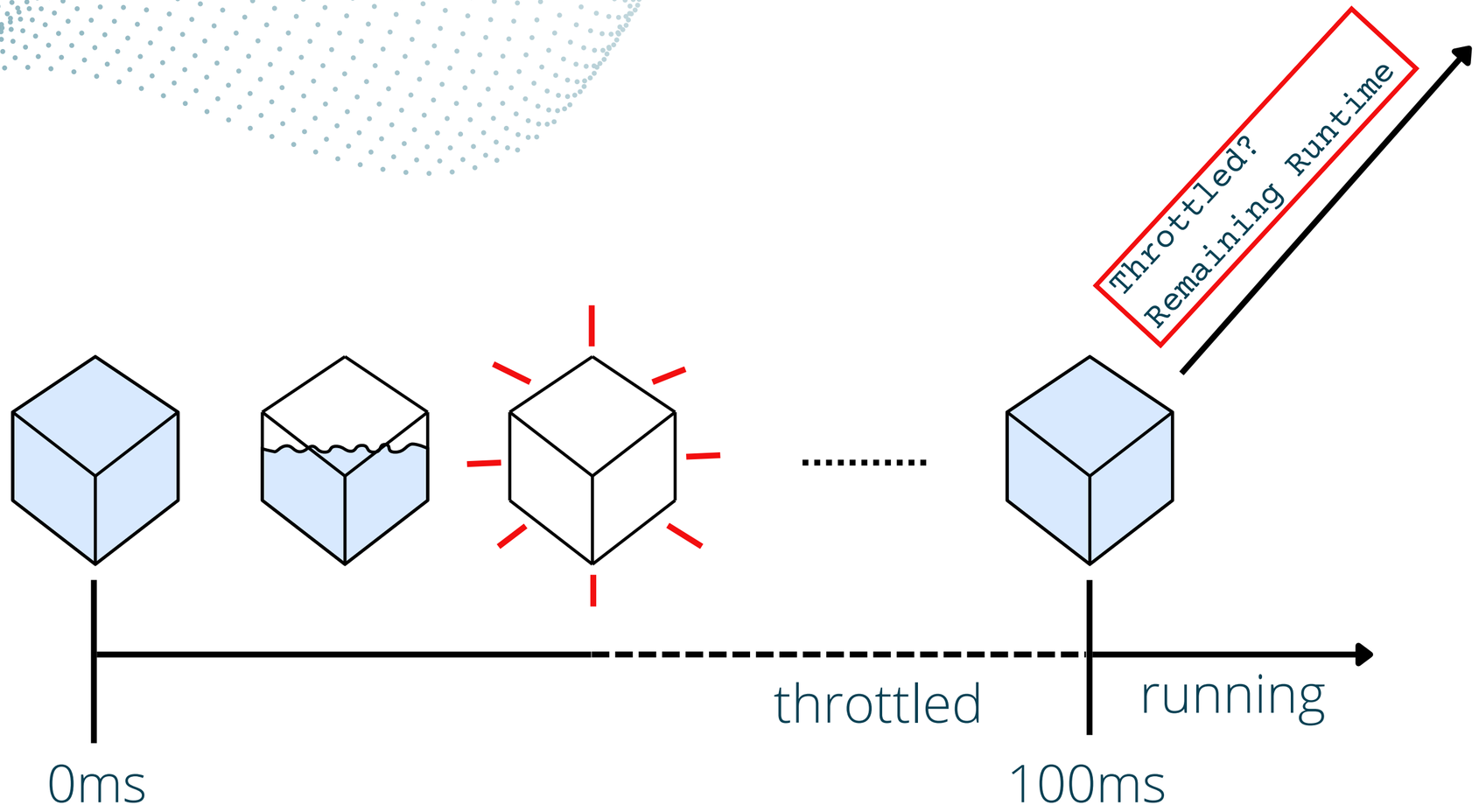
Escra Architecture

- Kernel Hooks
- Controller
- Resource Allocator
- **Agent**

CPU Telemetry and Scaling

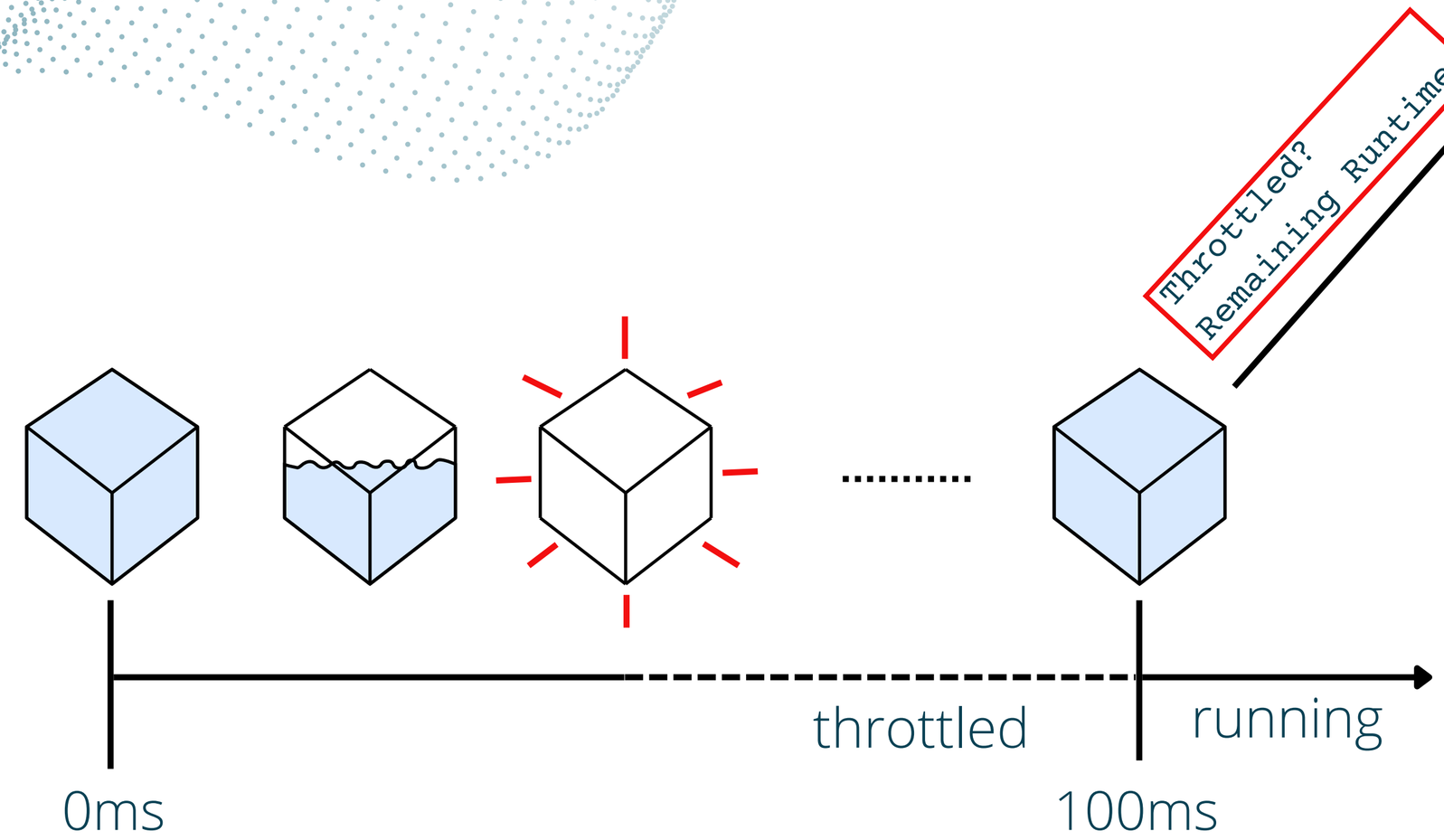


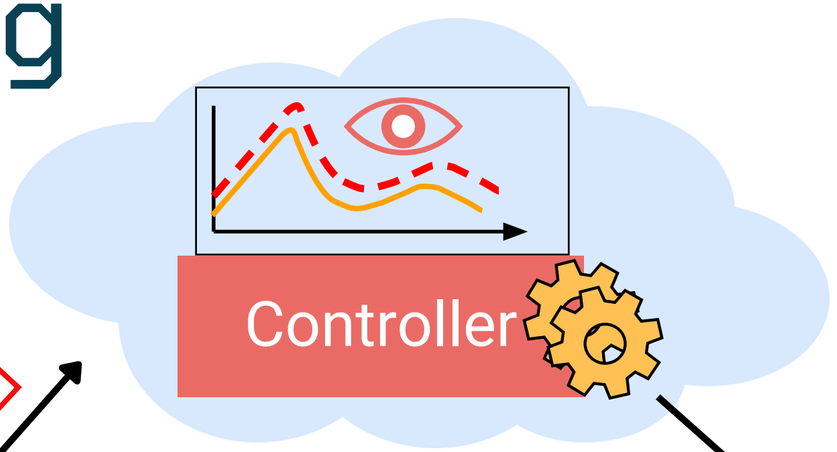
CPU Telemetry and Scaling



University of Colorado **Boulder**

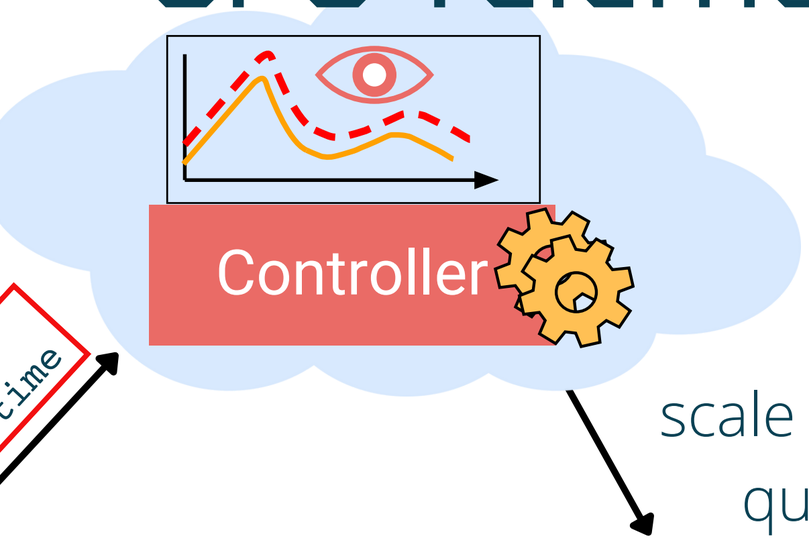
CPU Telemetry and Scaling





scale container
quota up

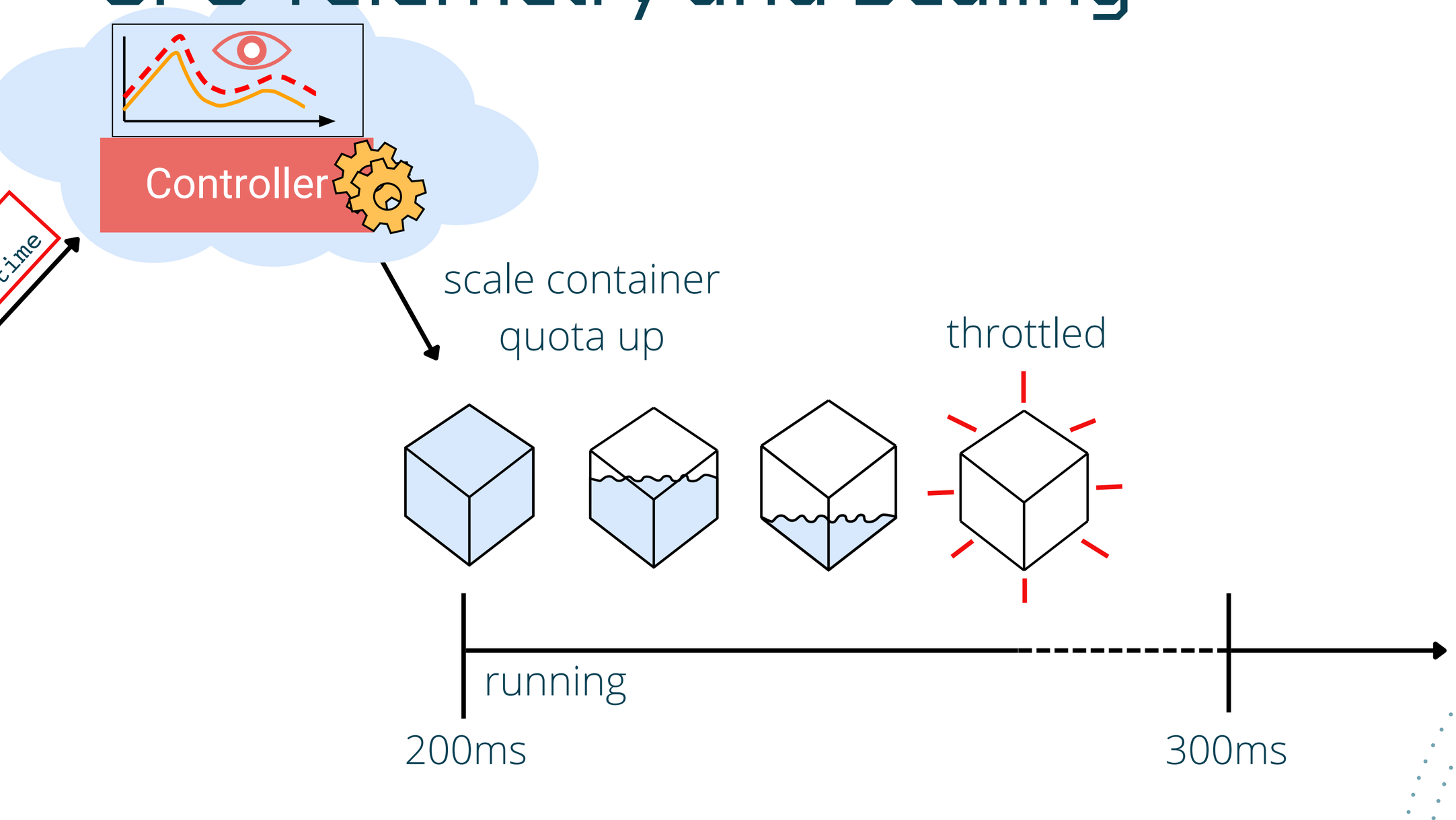
CPU Telemetry and Scaling



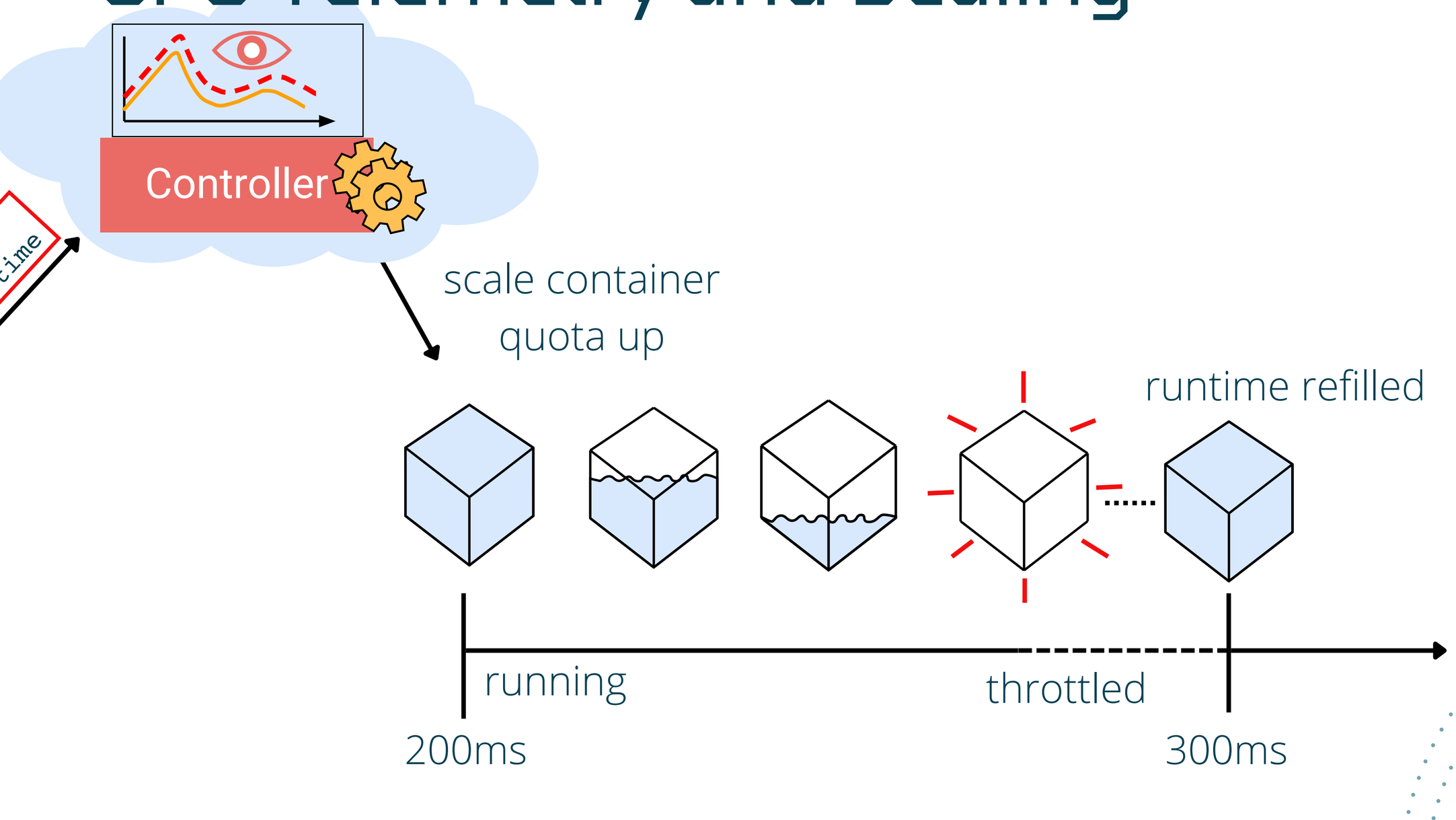
scale container
quota up

$$C(i)_q[t+1] = C(i)_q[t] + \frac{\sum_{t=0}^{n} C(i)_{th}[t]}{n} * \Upsilon(\Omega_l - \sum_{i=0}^{\lambda} C(i)_q[t])$$

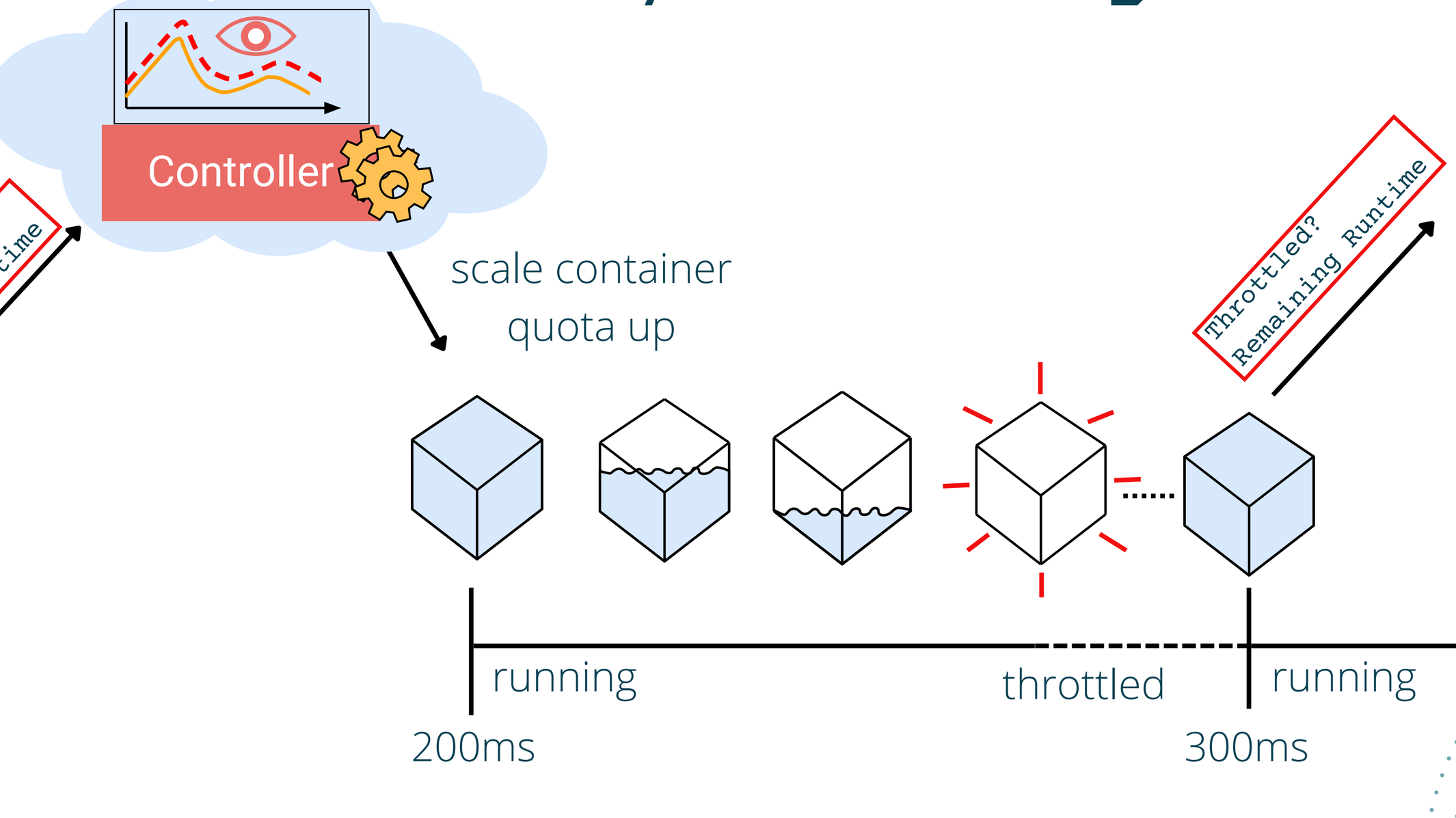
CPU Telemetry and Scaling



CPU Telemetry and Scaling

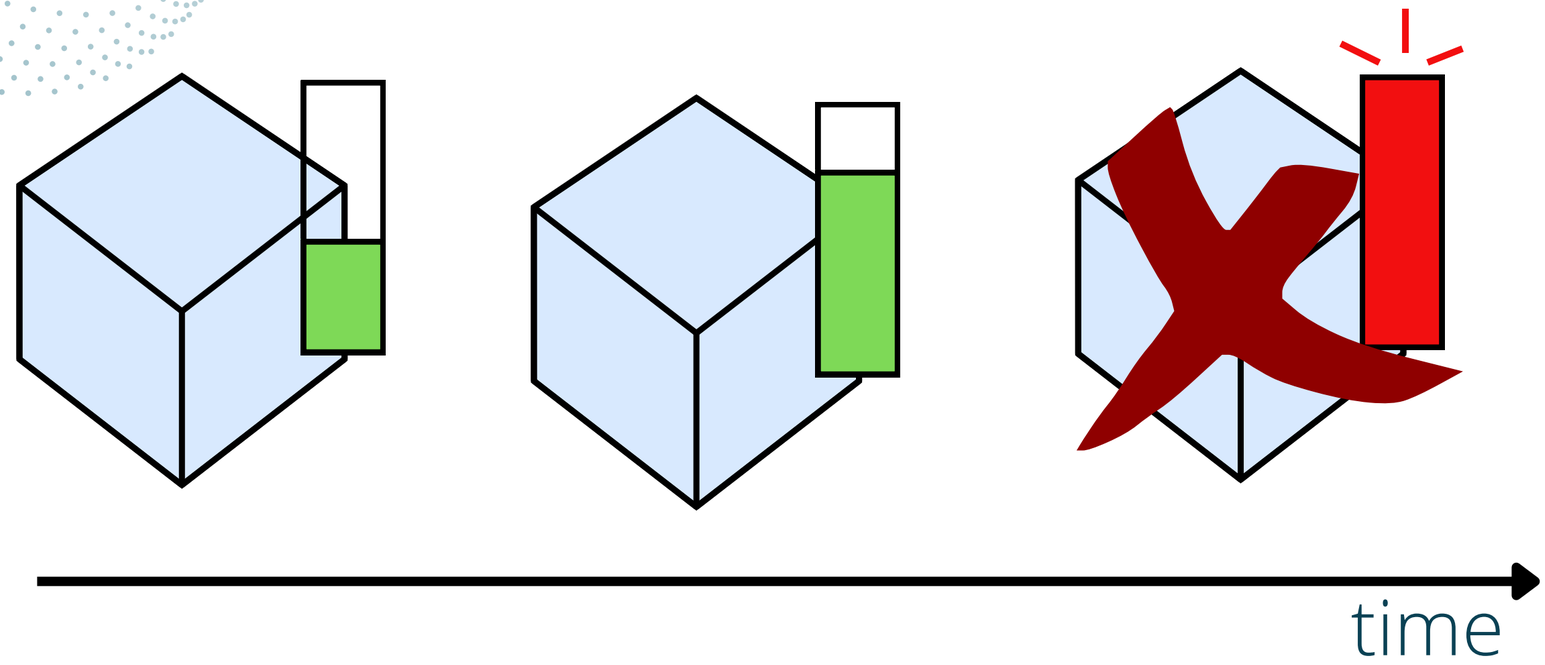


CPU Telemetry and Scaling



Event-Based Memory Scaling

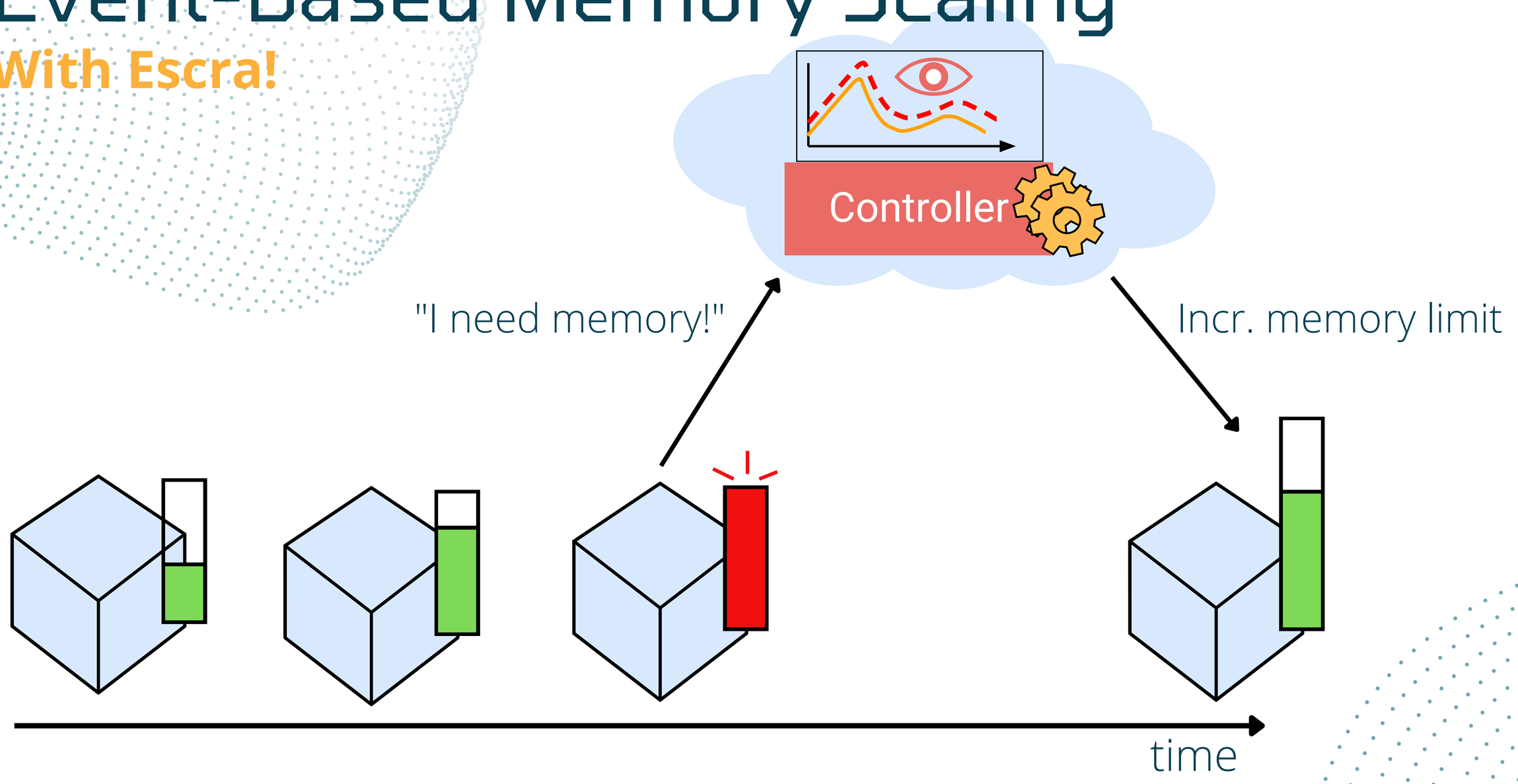
Typical Scenario



University of Colorado **Boulder**

Event-Based Memory Scaling

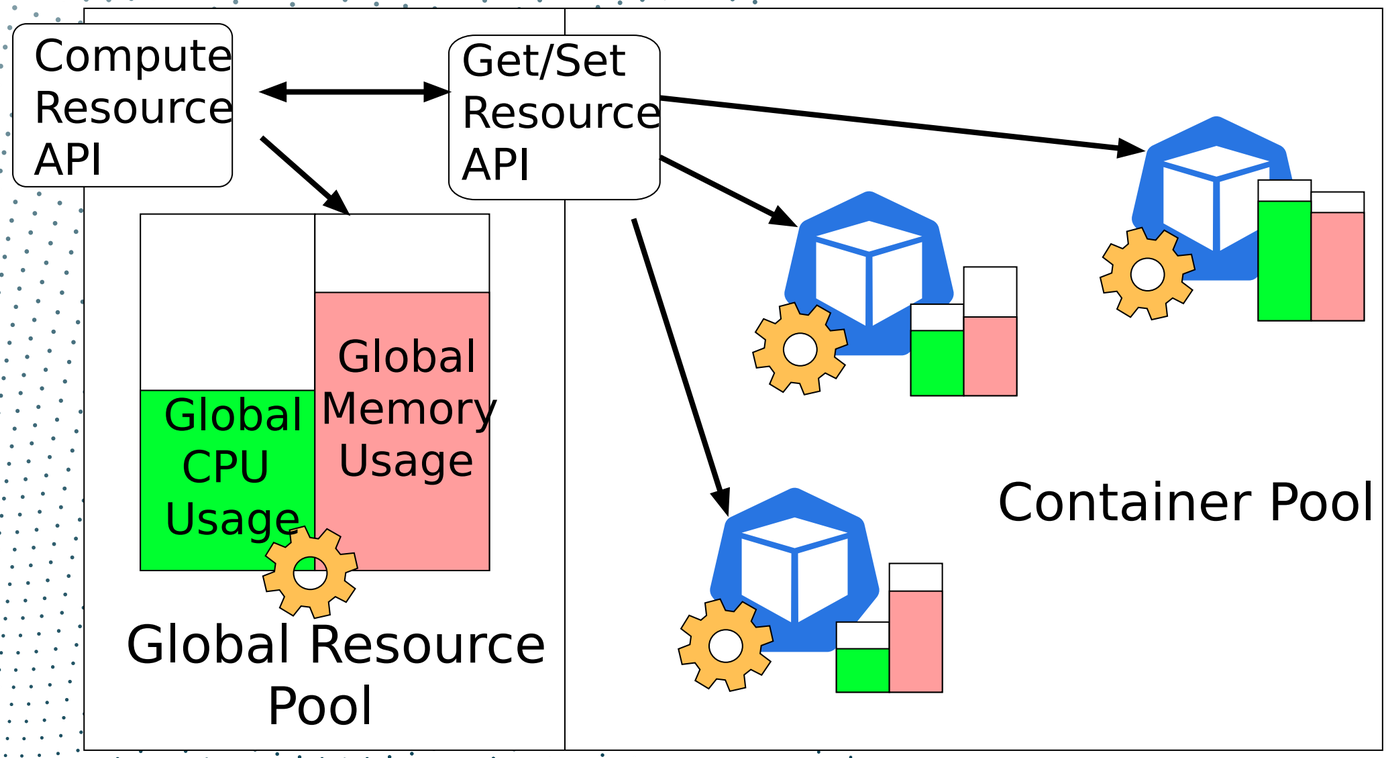
With Escra!



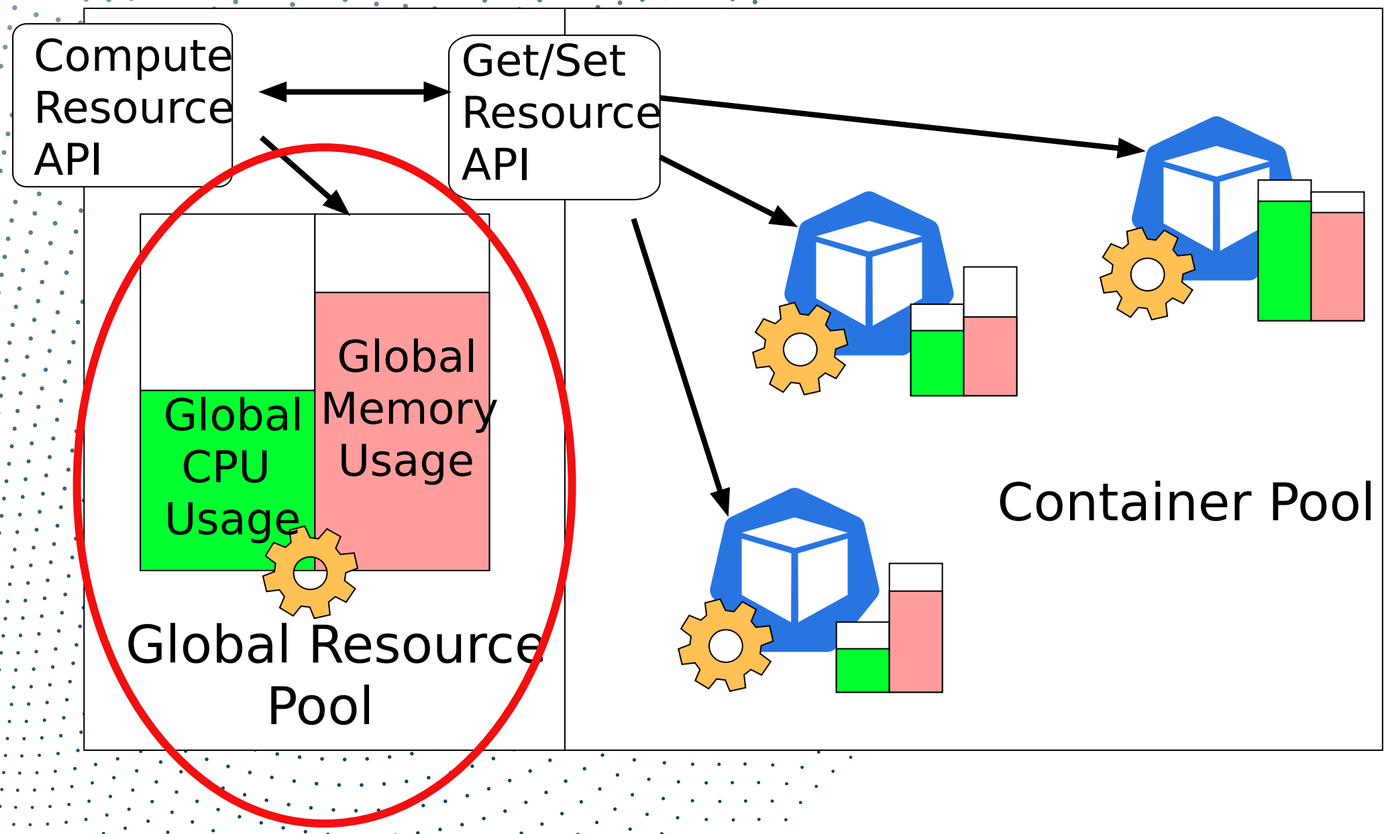
University of Colorado **Boulder**

Distributed Container

Distributed Container

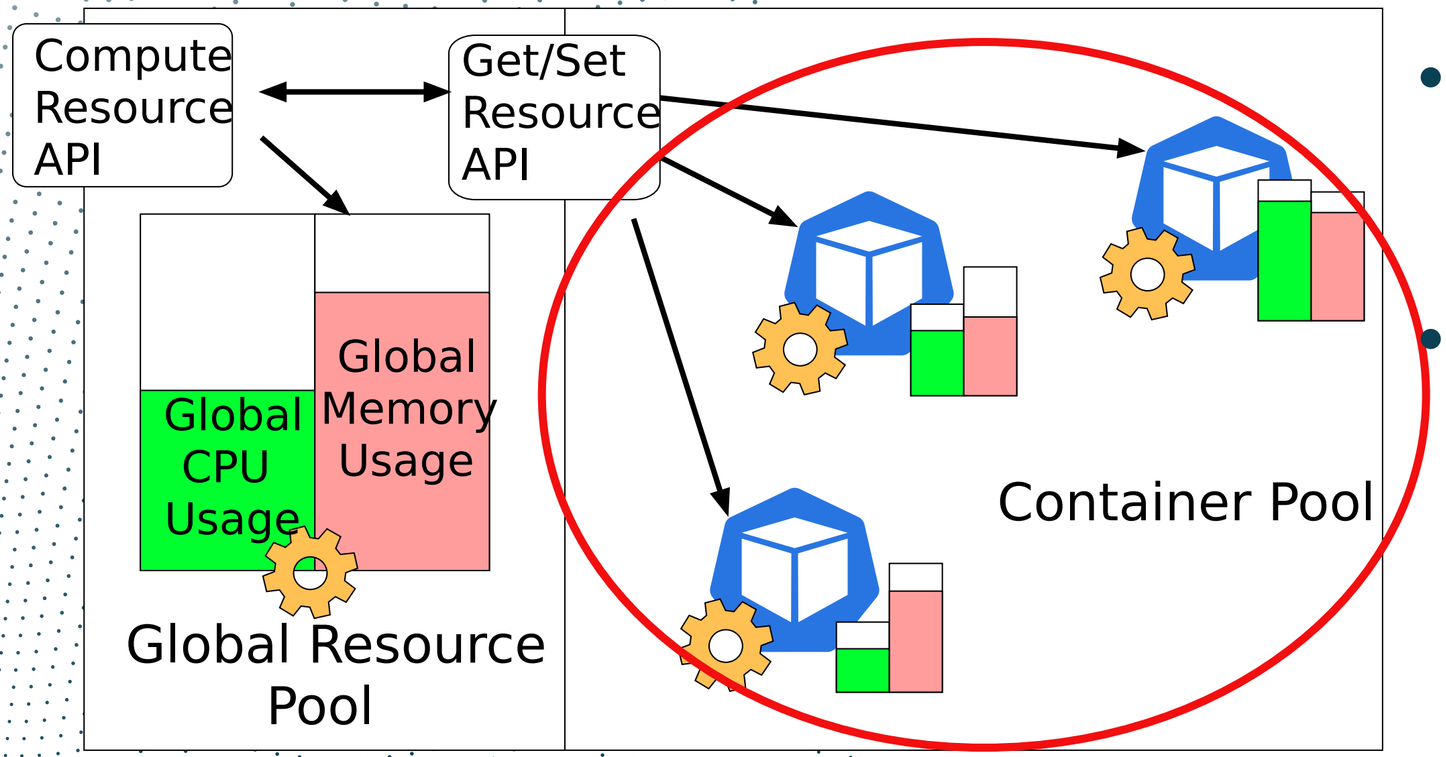


Distributed Container



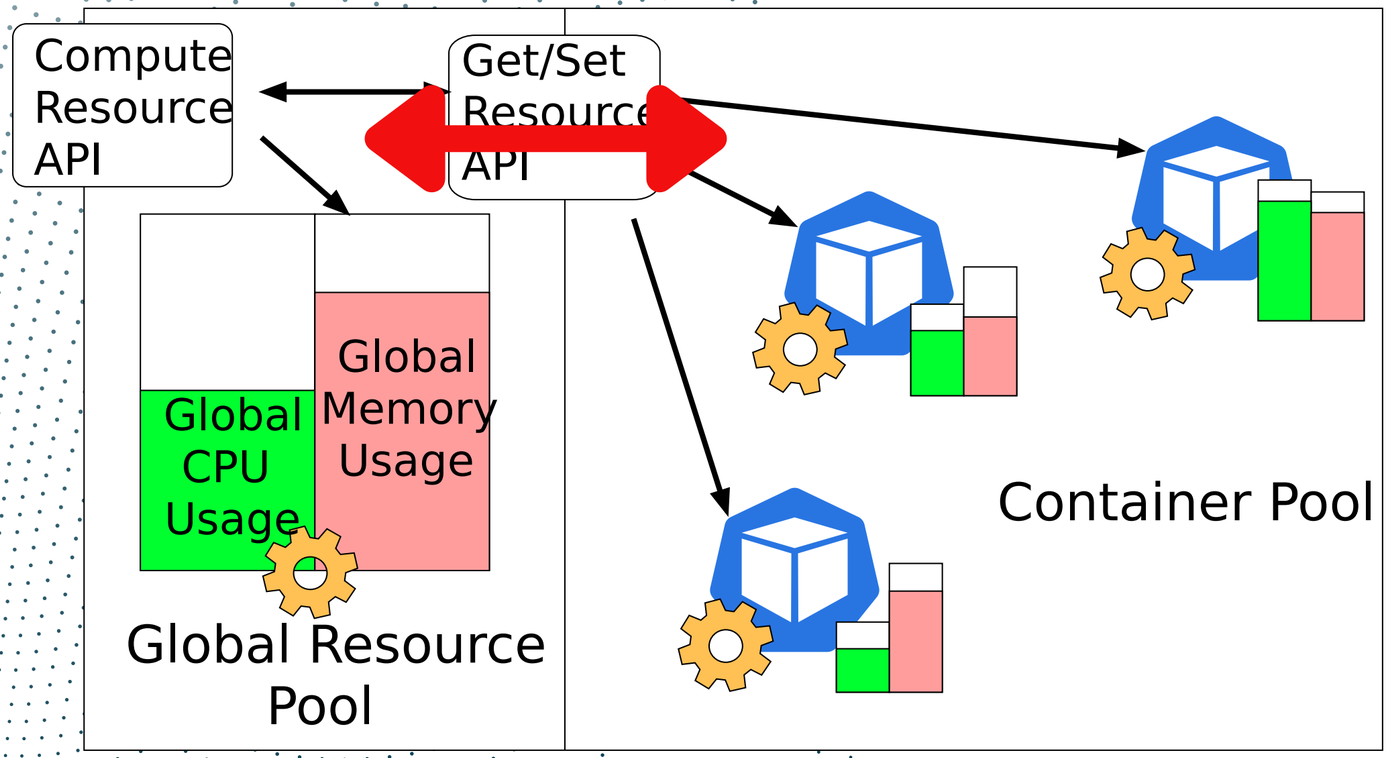
- Enforces per-application resources limits

Distributed Container



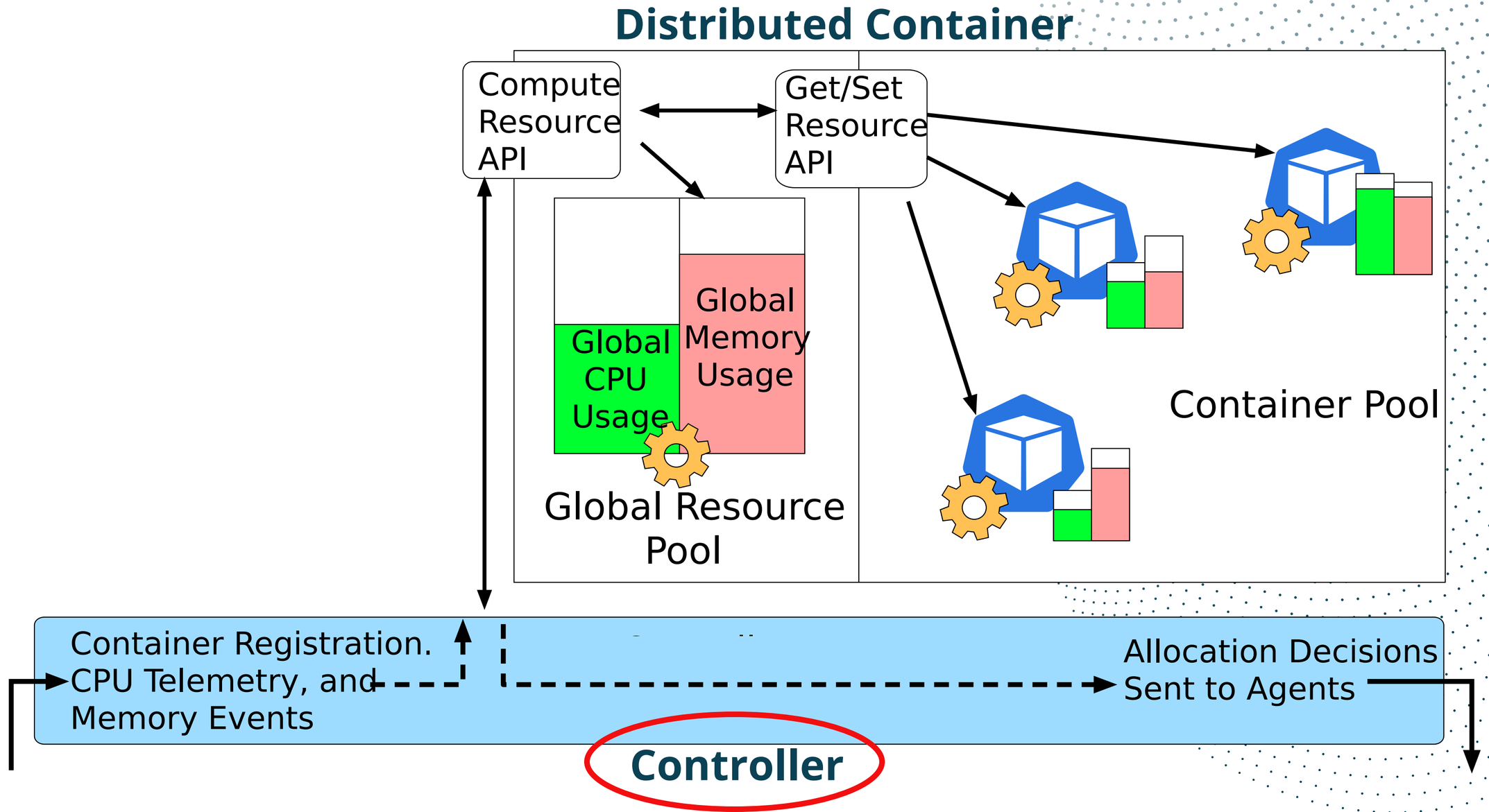
- Enforces per-application resources limits
- Per-container resource usage and limit tracking

Distributed Container

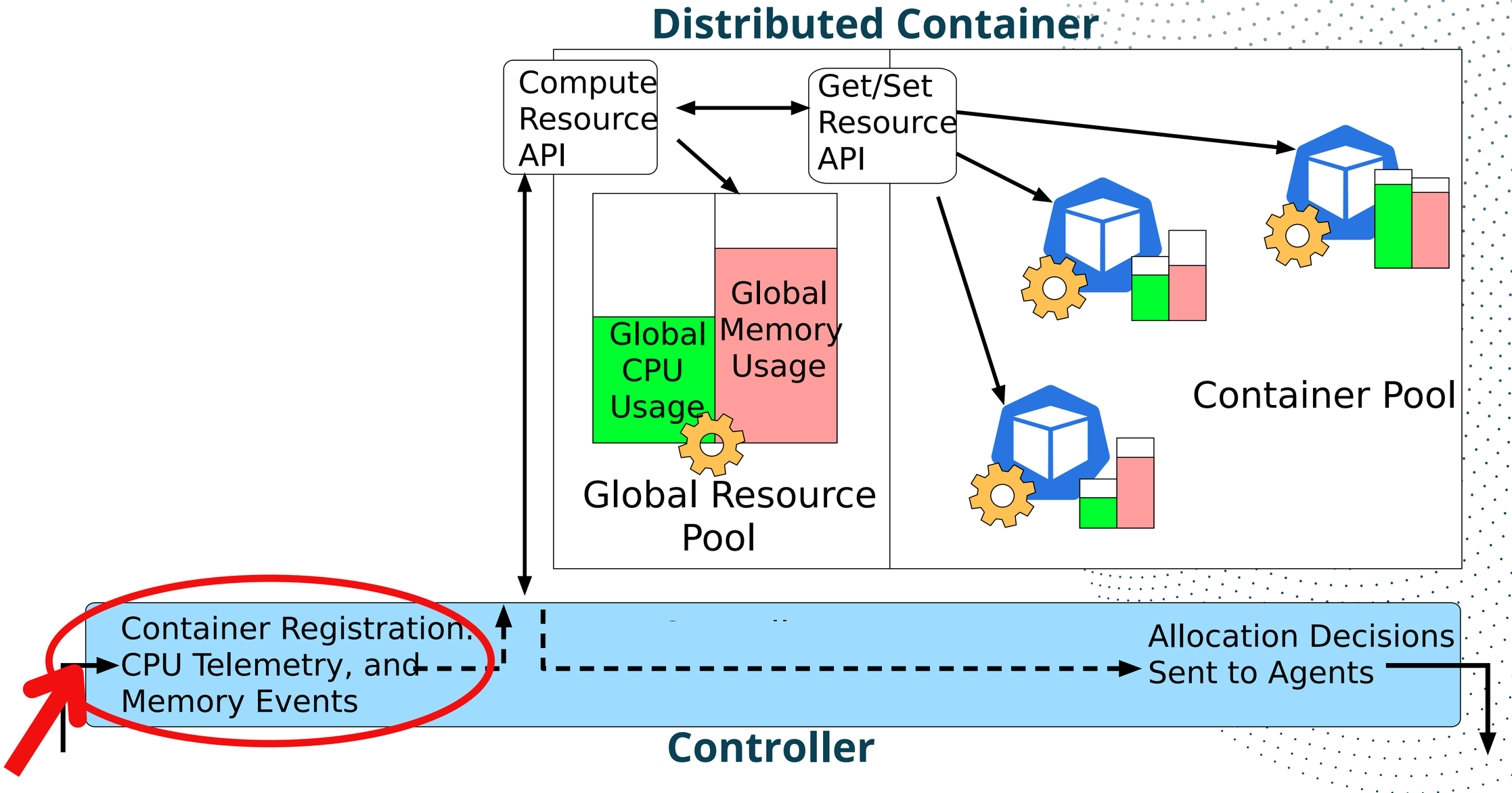


- Enforces per-application resources limits
- Per-container resource usage and limit tracking
- Containers dynamically share compute resources at runtime

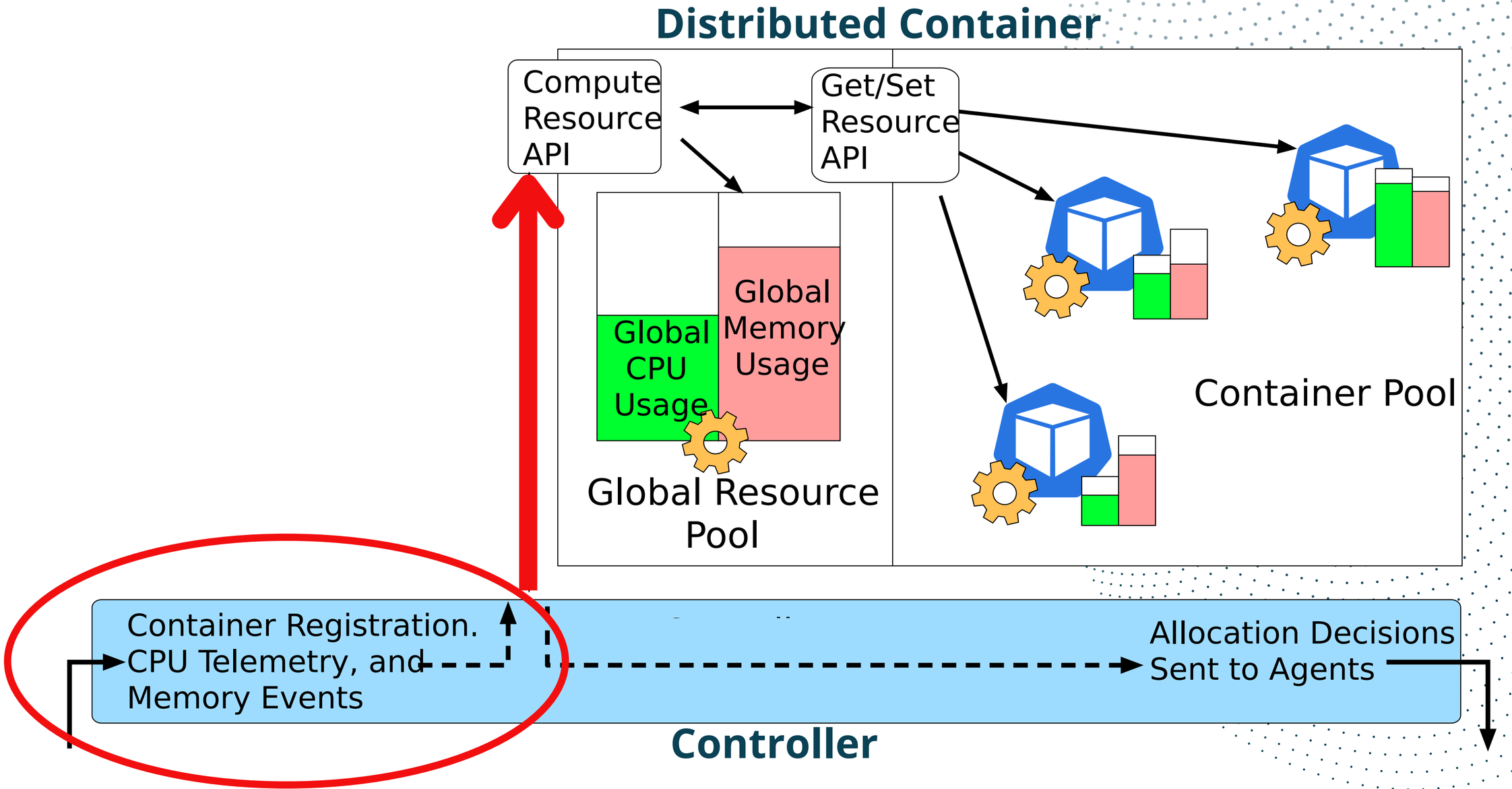
Escra Controller



Escra Controller

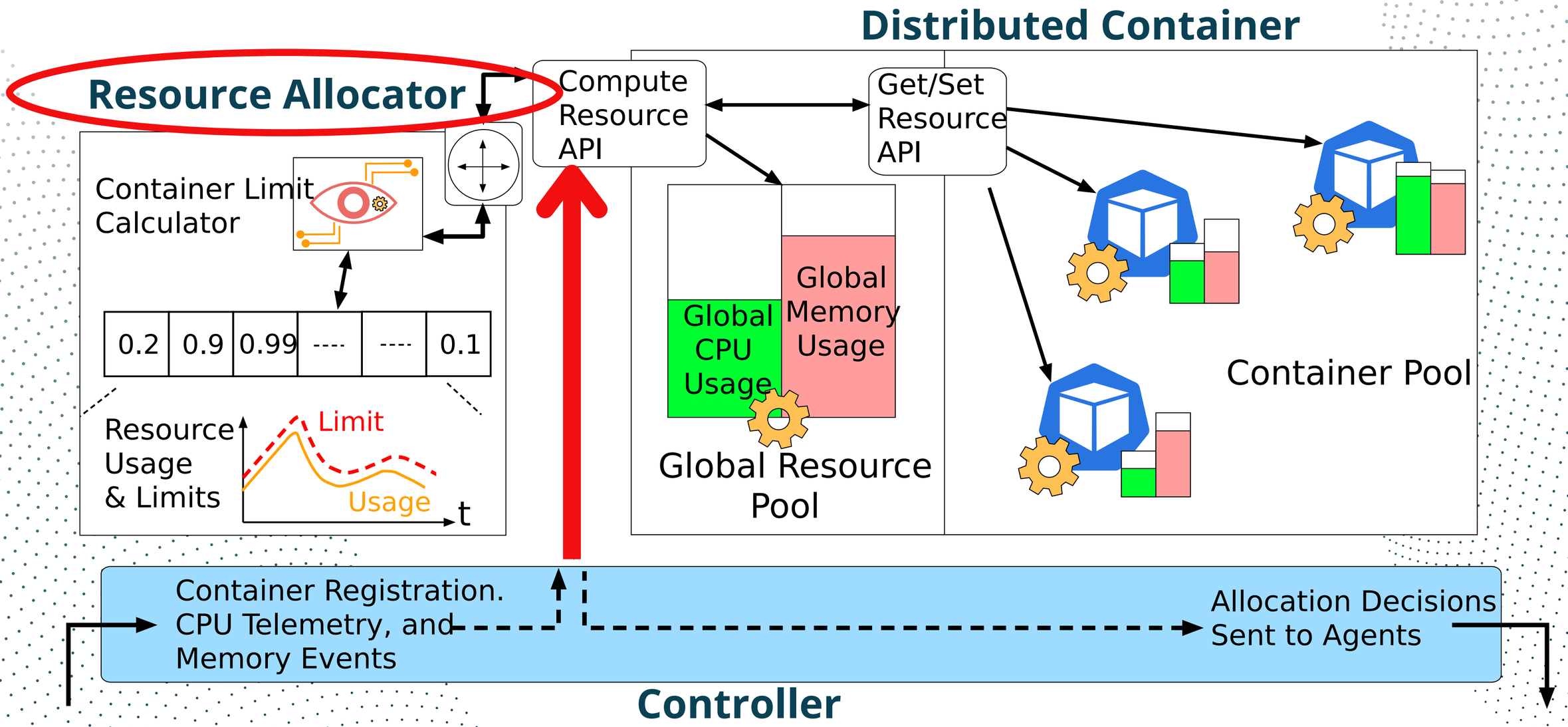


Escra Controller



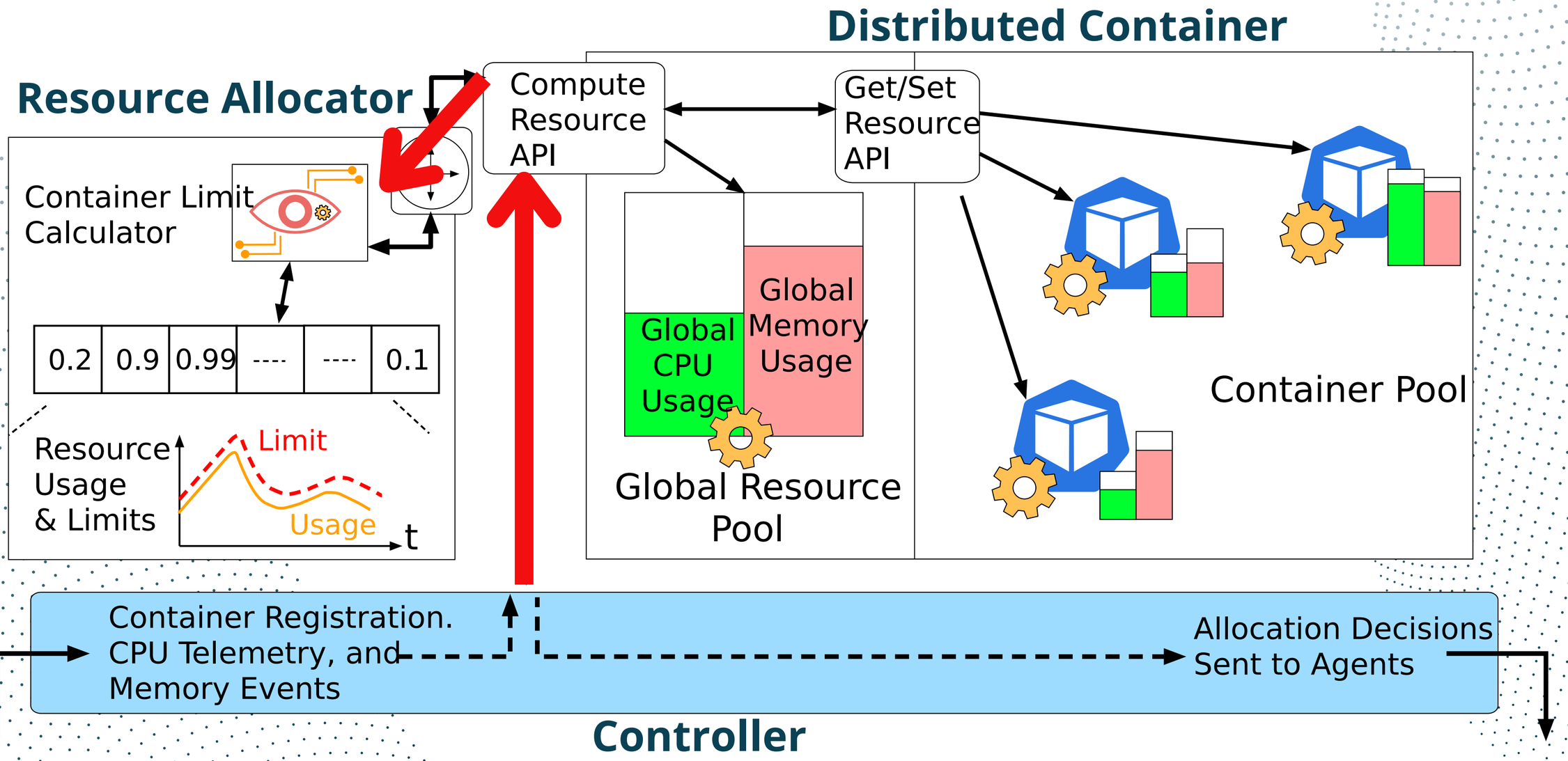
University of Colorado **Boulder**

Escra - Control Node

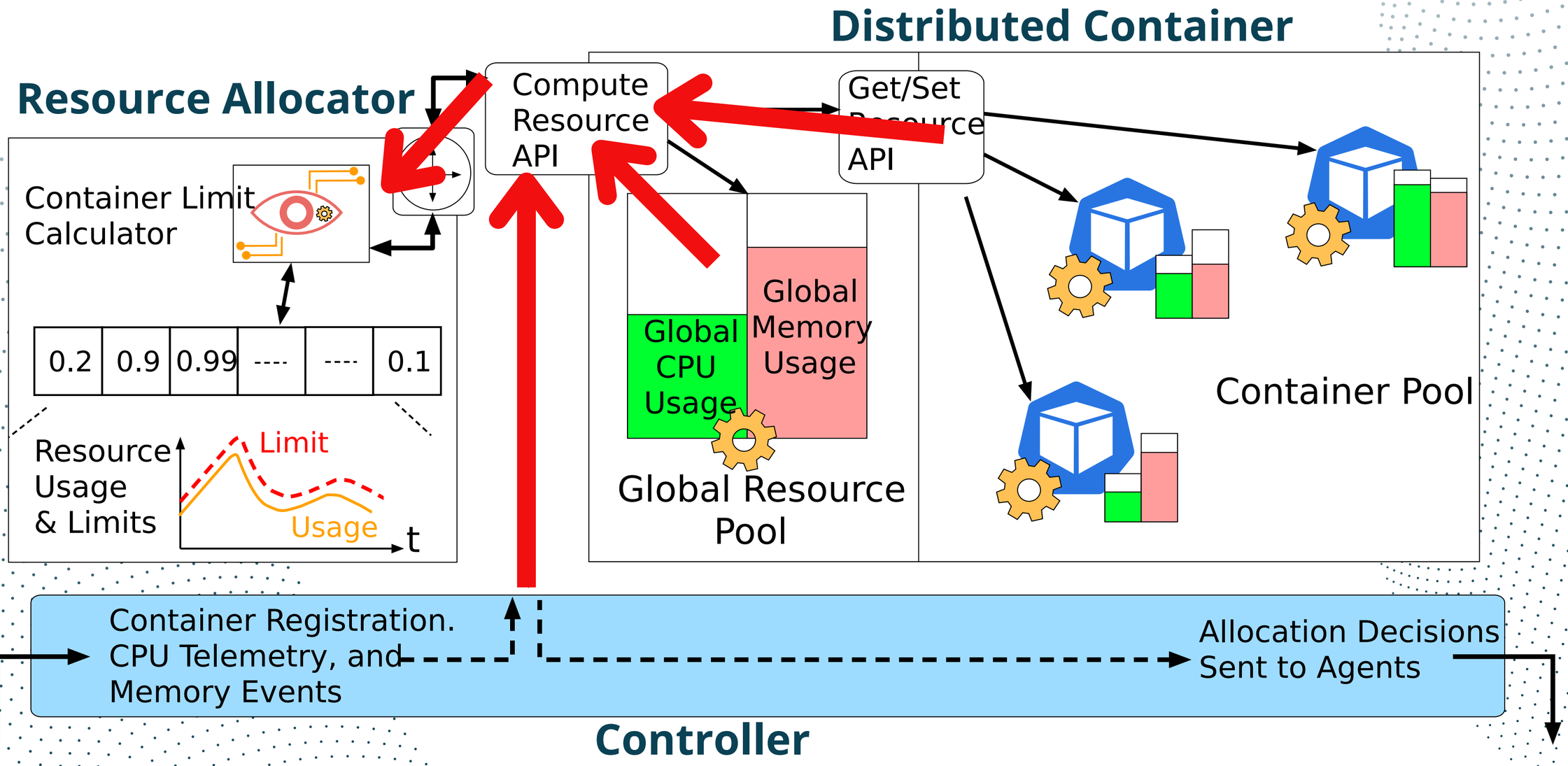


University of Colorado **Boulder**

Escra - Control Node

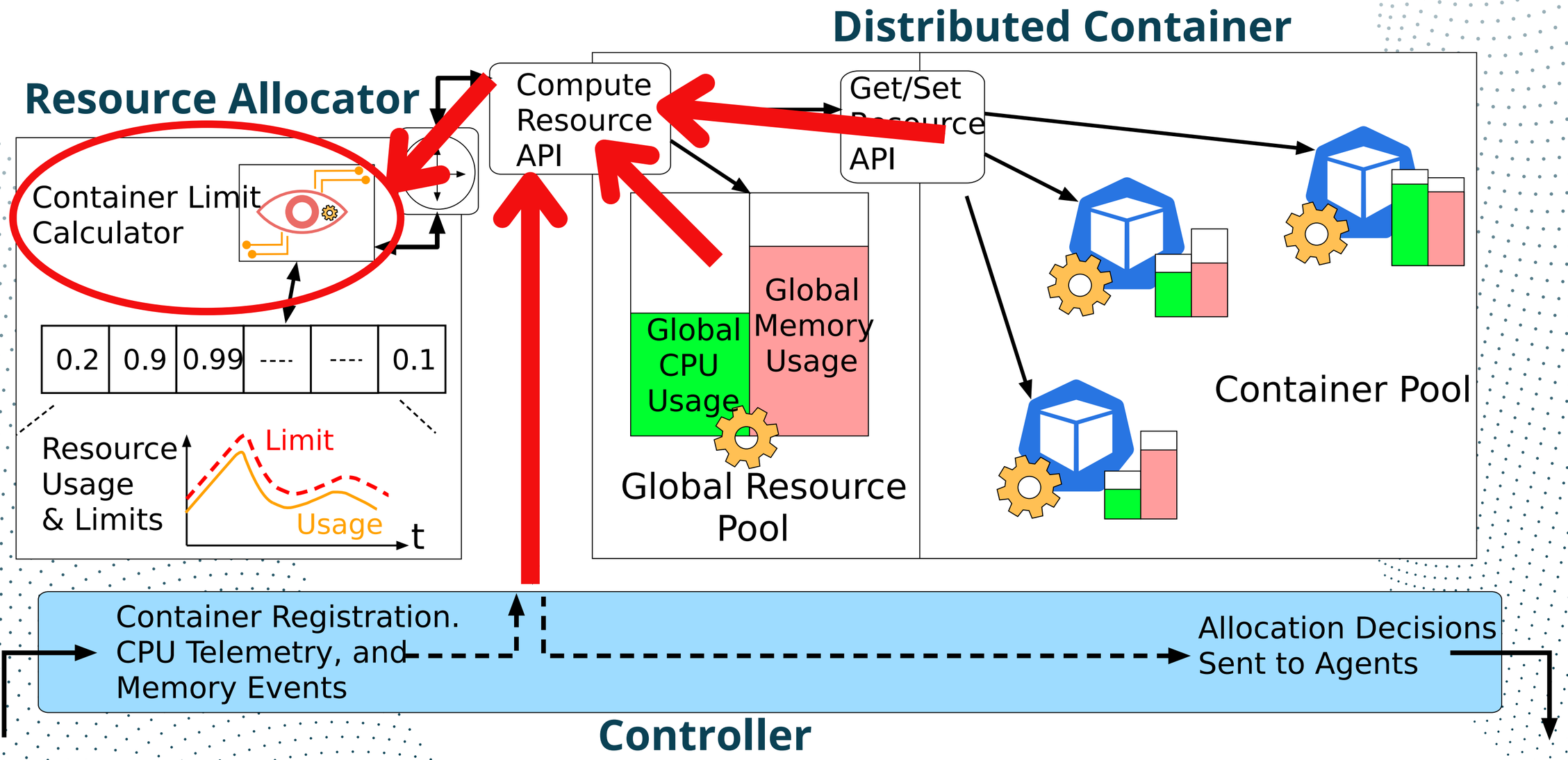


Escra - Control Node

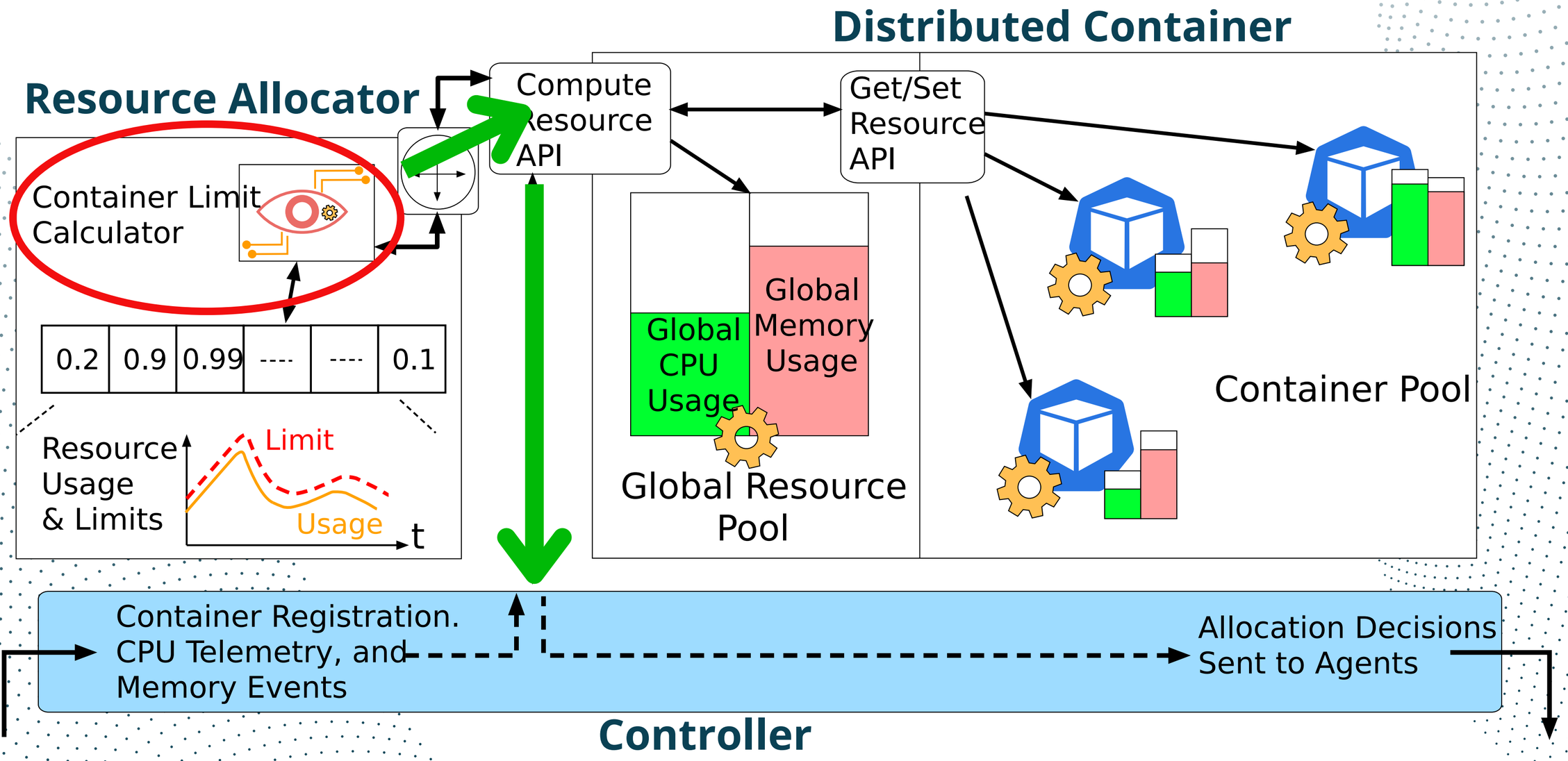


University of Colorado **Boulder**

Escra - Control Node

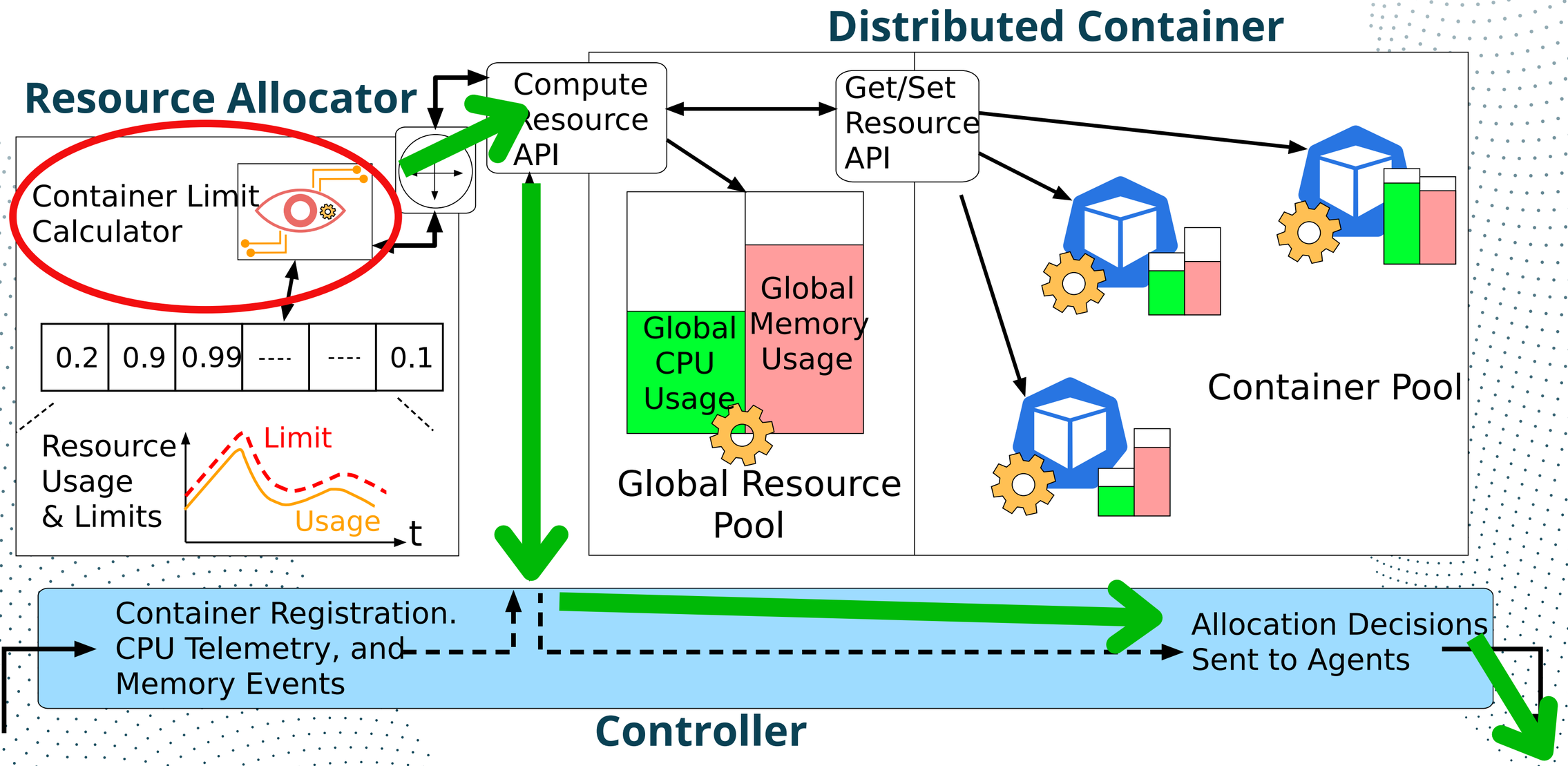


Escra - Control Node



University of Colorado **Boulder**

Escra - Control Node



Evaluation - Microservices

Evaluation - Microservices

Metrics

Absolute Slack

Container Limit - Container Usage

Application Throughput

Successful requests/second

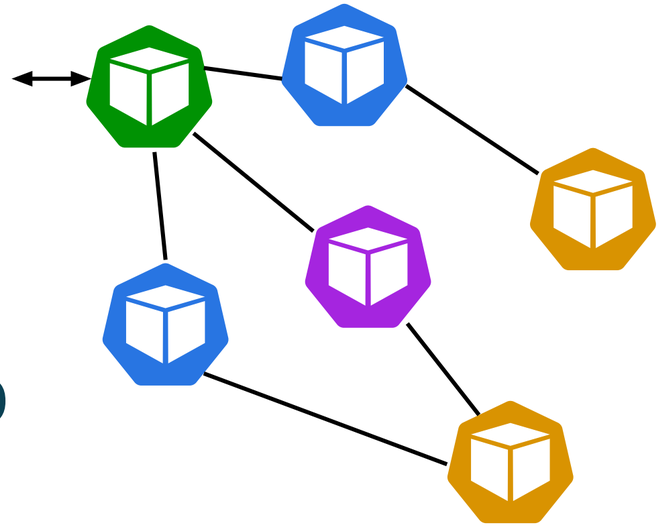
Application 99.9%ile Latency

99.9%ile end-to-end latency

Applications & Workloads

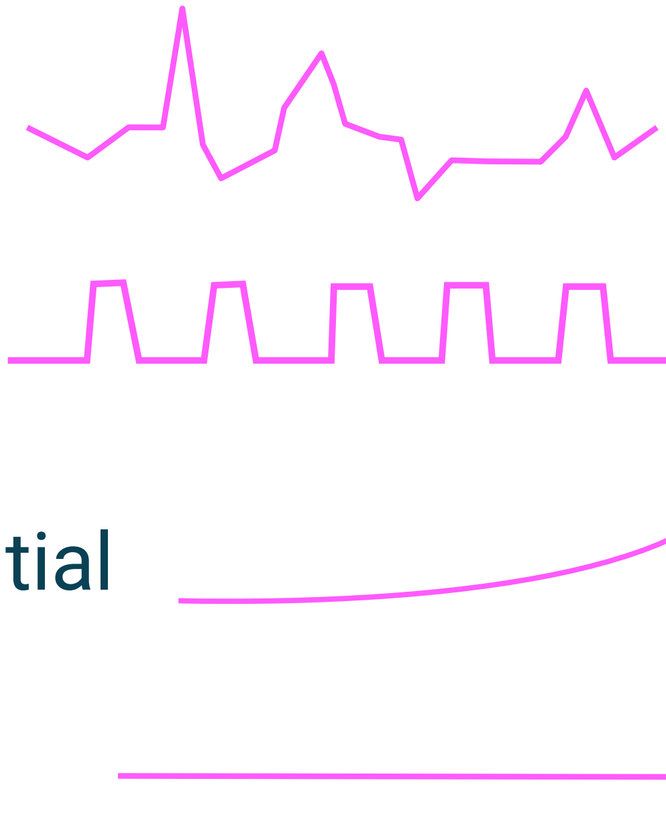
Applications

- TrainTicket
- TeaStore
- HipsterShop
- MediaMicroservice



Workloads

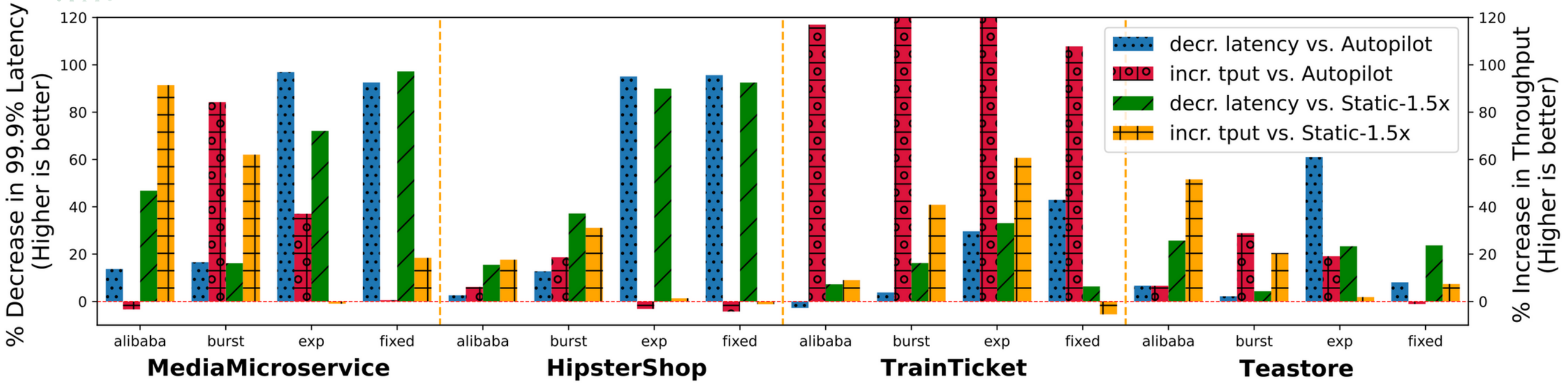
- Alibaba
- Burst
- Exponential
- Fixed



University of Colorado **Boulder**

Change in Latency and Throughput

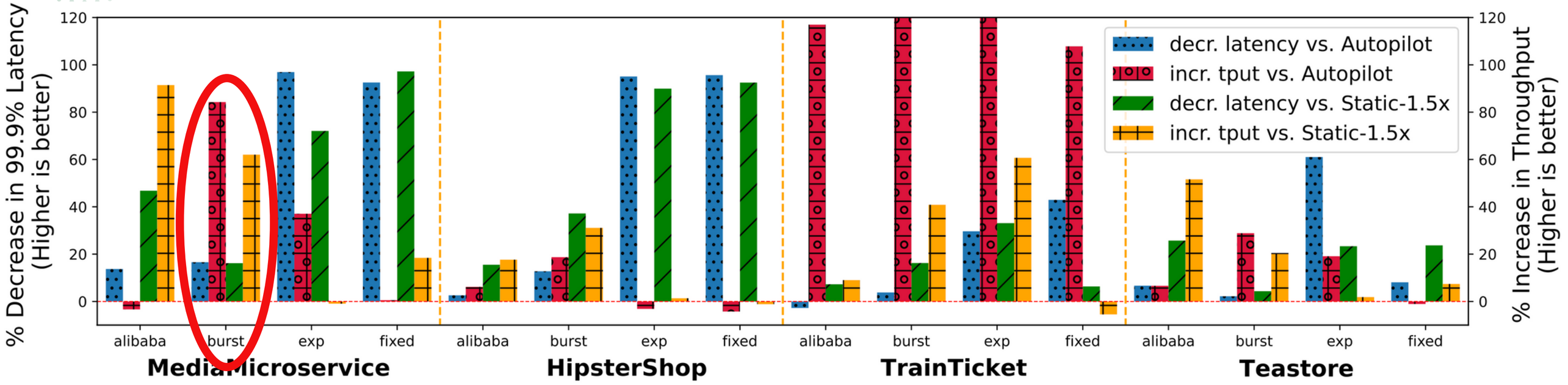
Microservices



University of Colorado **Boulder**

Change in Latency and Throughput

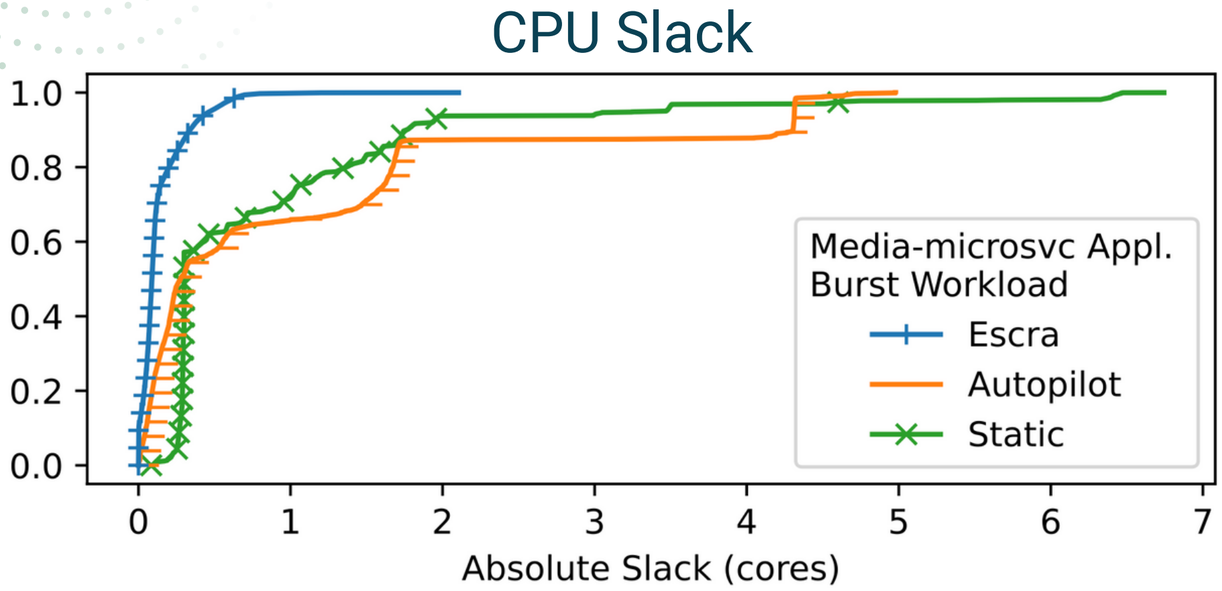
Microservices



University of Colorado **Boulder**

Escra vs. Autopilot

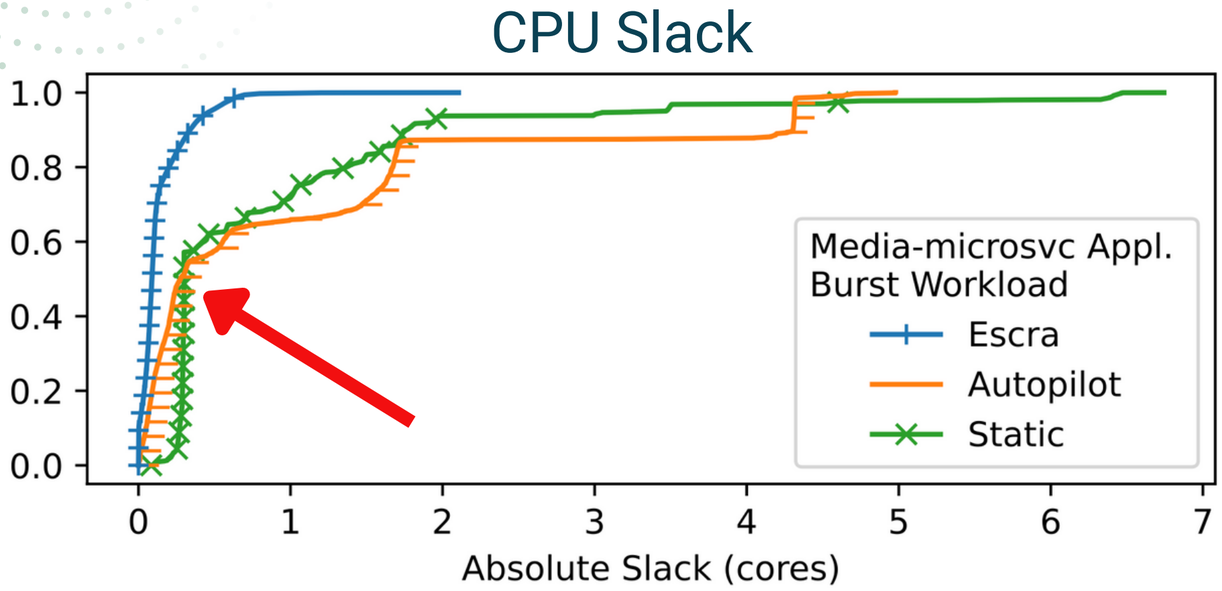
Microservices - MediaMicroservice/Burst Workload



University of Colorado **Boulder**

Escra vs. Autopilot

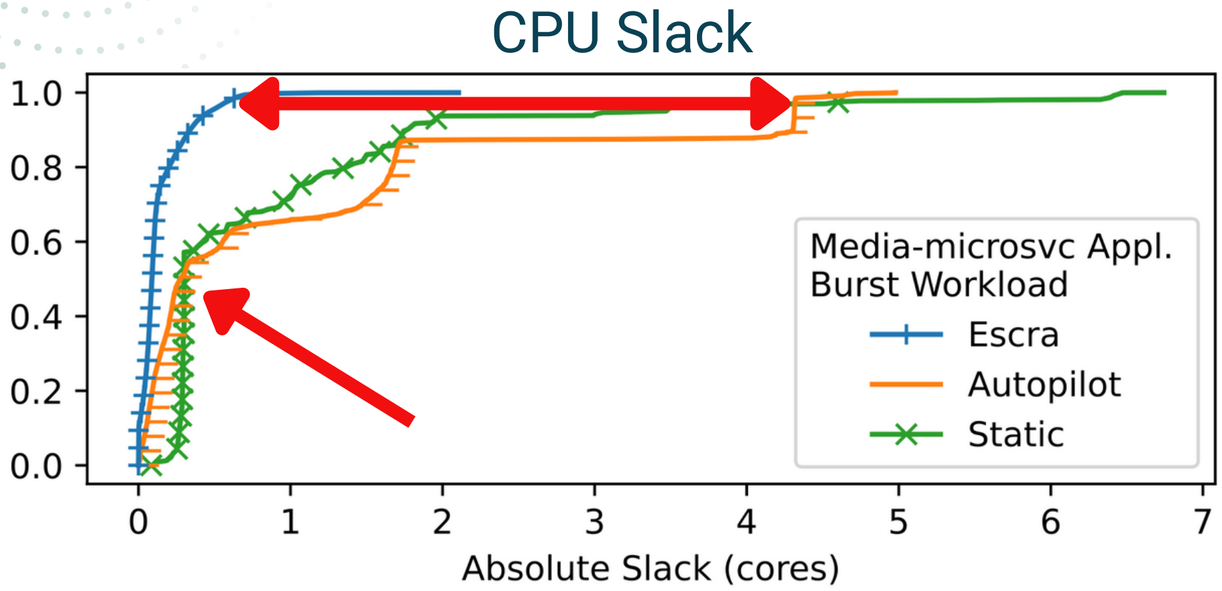
Microservices - MediaMicroservice/Burst Workload



University of Colorado **Boulder**

Escra vs. Autopilot

Microservices - MediaMicroservice/Burst Workload

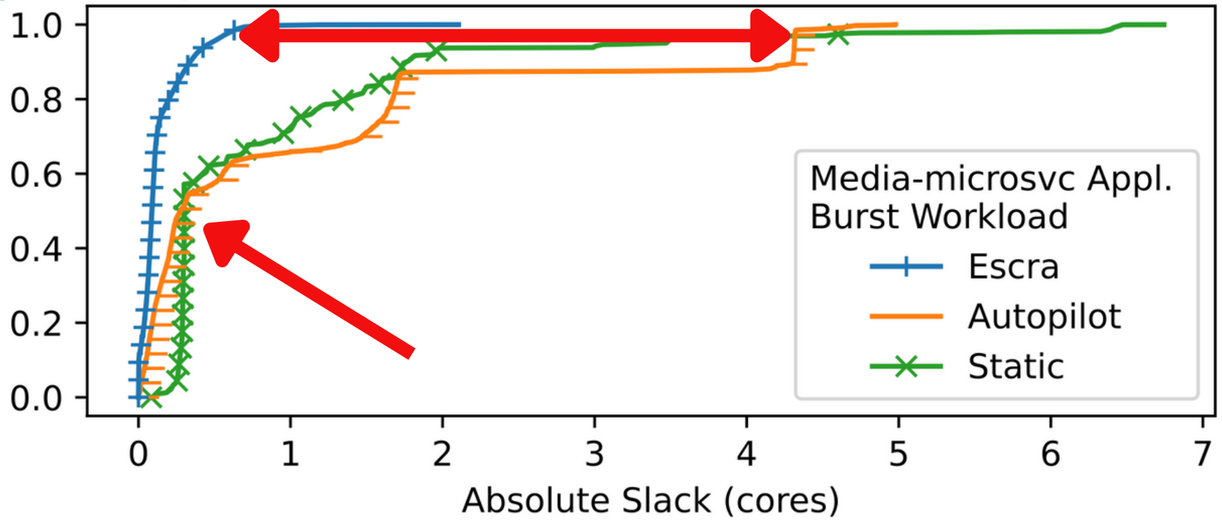


University of Colorado **Boulder**

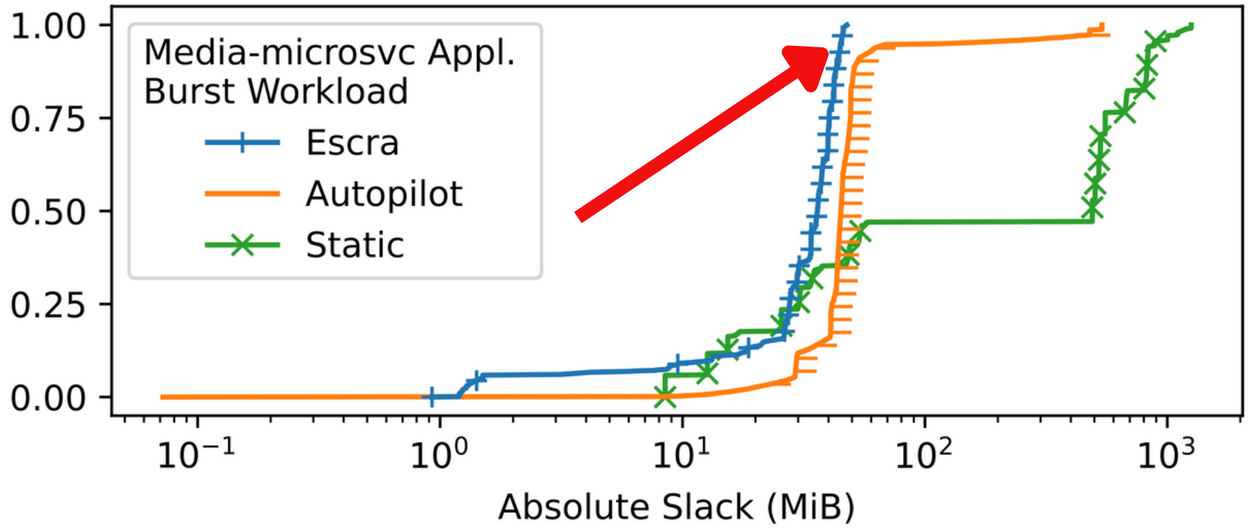
Escra vs. Autopilot

Microservices - MediaMicroservice/Burst Workload

CPU Slack



Memory Slack

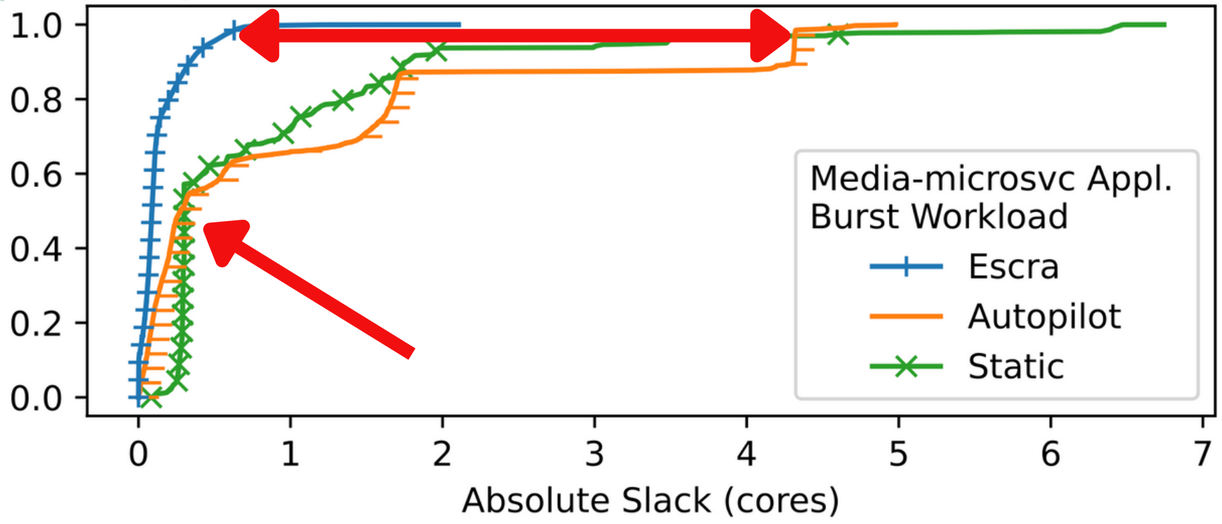


University of Colorado **Boulder**

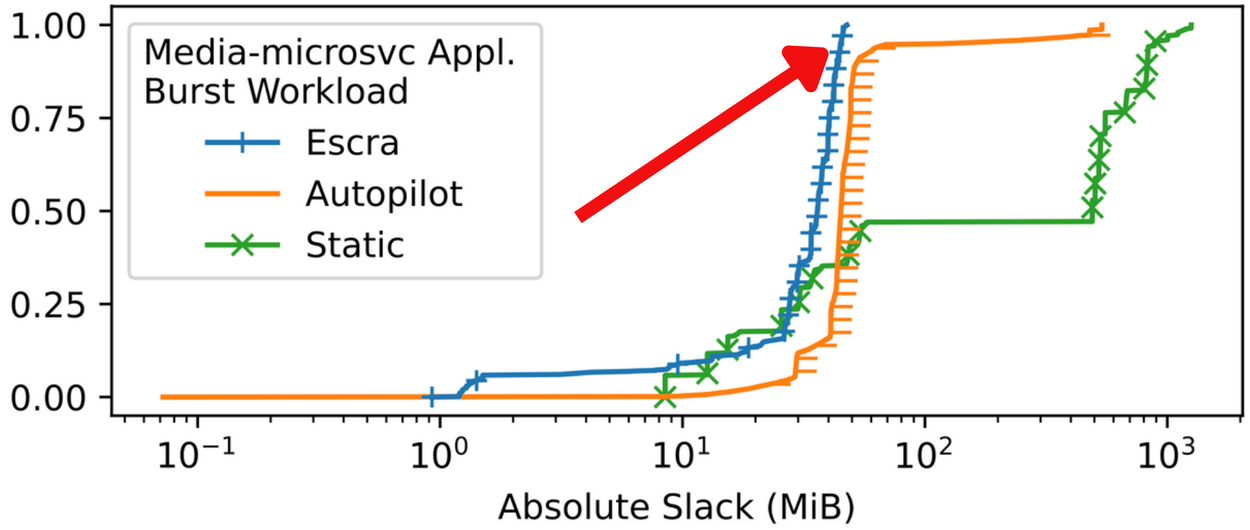
Escra vs. Autopilot

Microservices - MediaMicroservice/Burst Workload

CPU Slack



Memory Slack



- **Latency Decrease: 16.6%**
- **Throughput Increase: 84.3%**

Evaluation - Serverless

Evaluation - Serverless

Metrics

Aggregate Limits

Σcontainer limits - Σcontainer usages

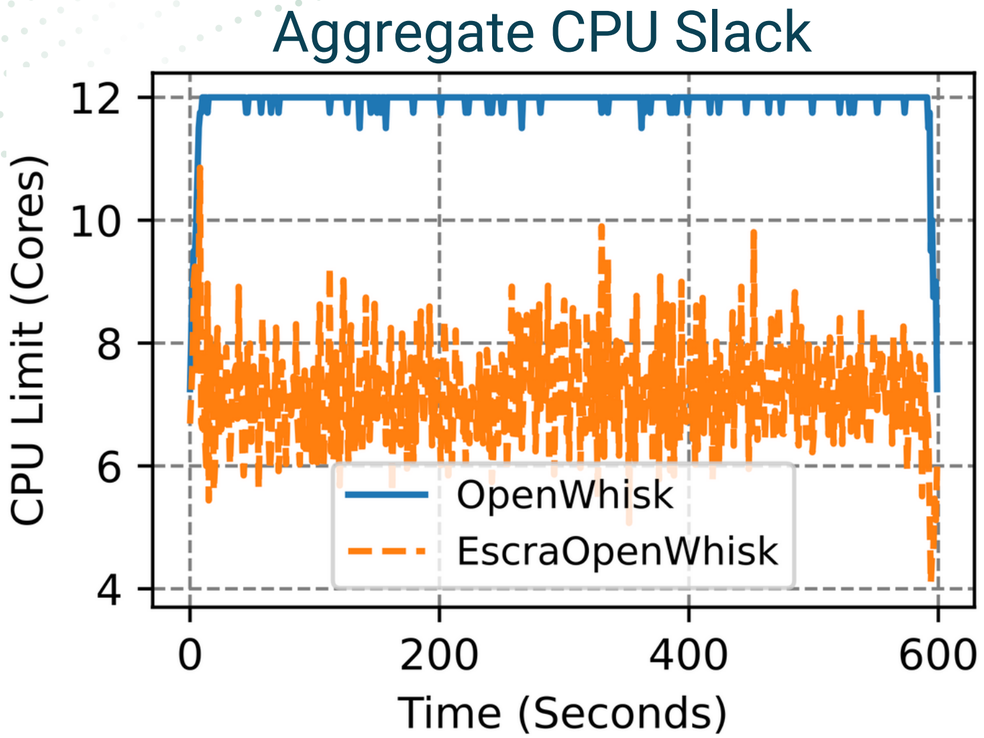
Application Latency

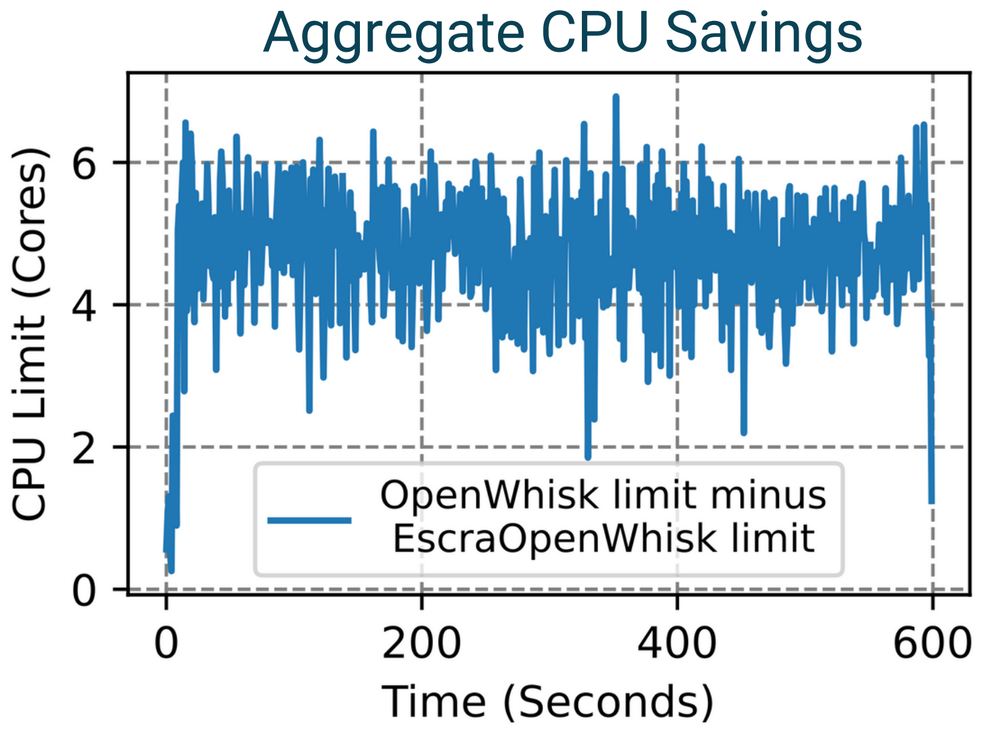
End-to-end latency per request or job

University of Colorado **Boulder**

ImageProcess

Serverless



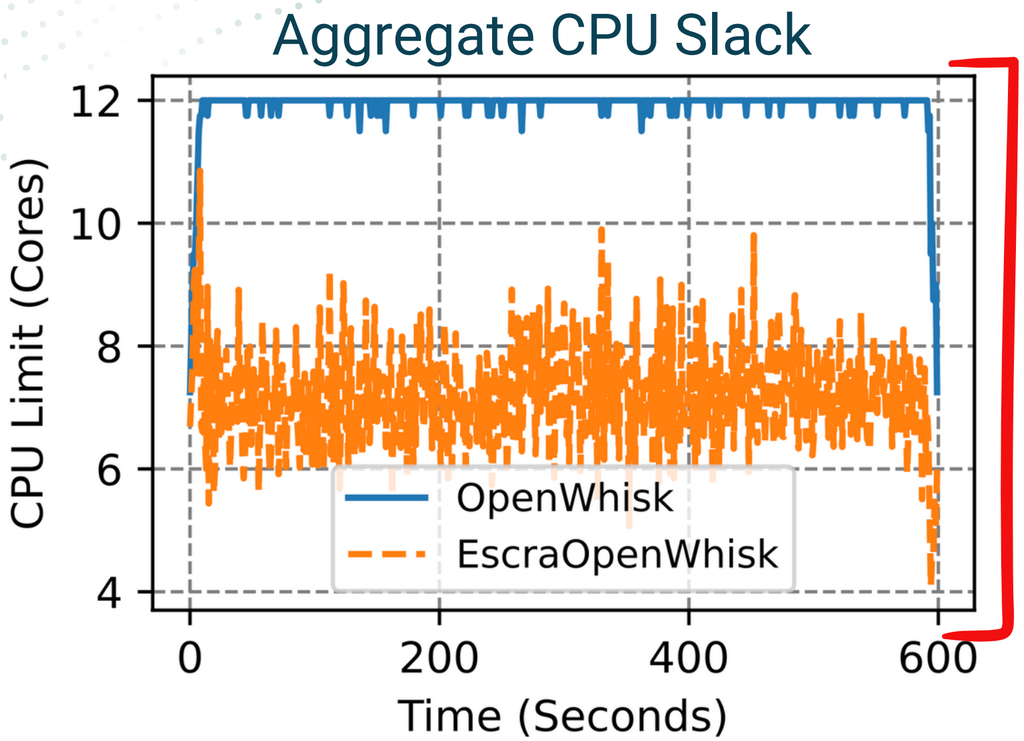


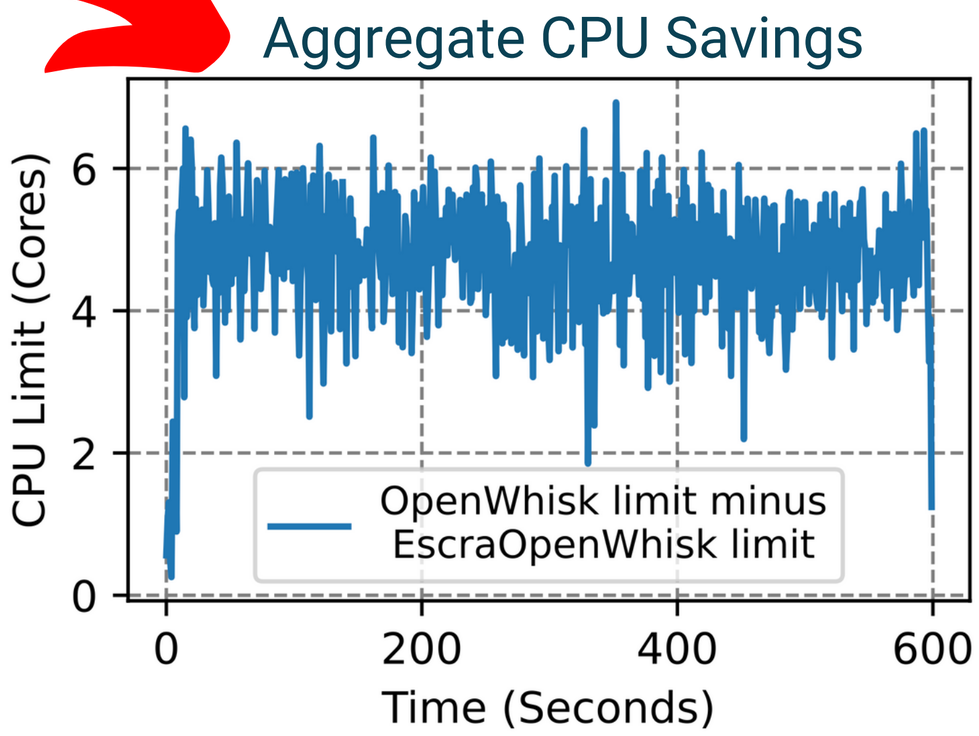
University of Colorado **Boulder**

ImageProcess

Serverless

Difference!



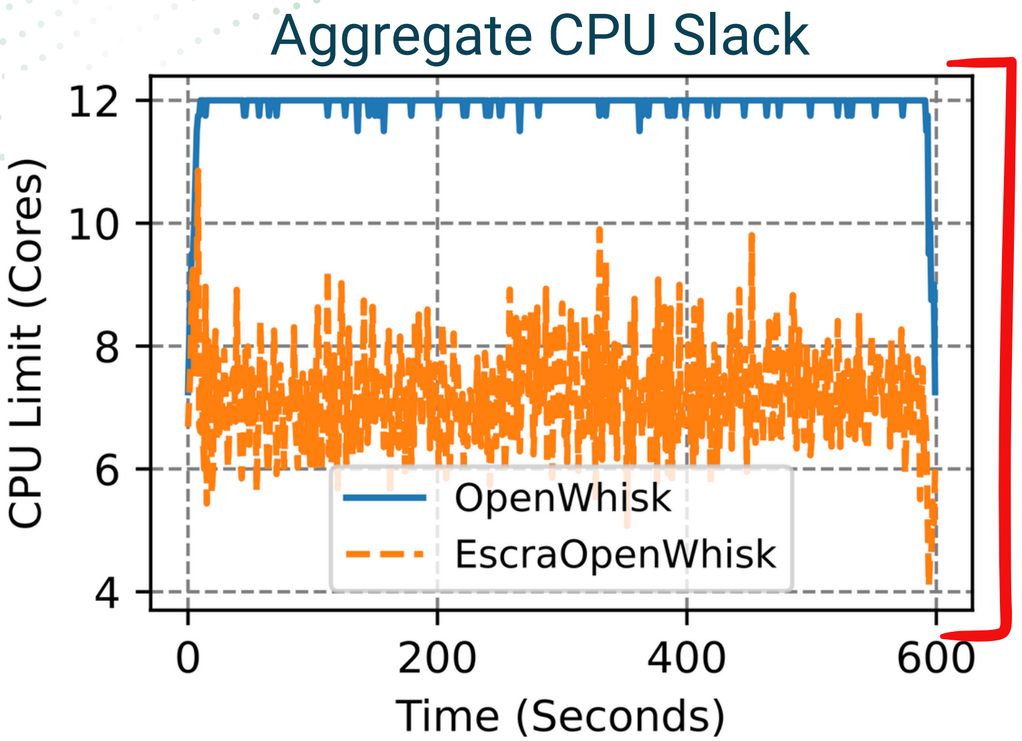


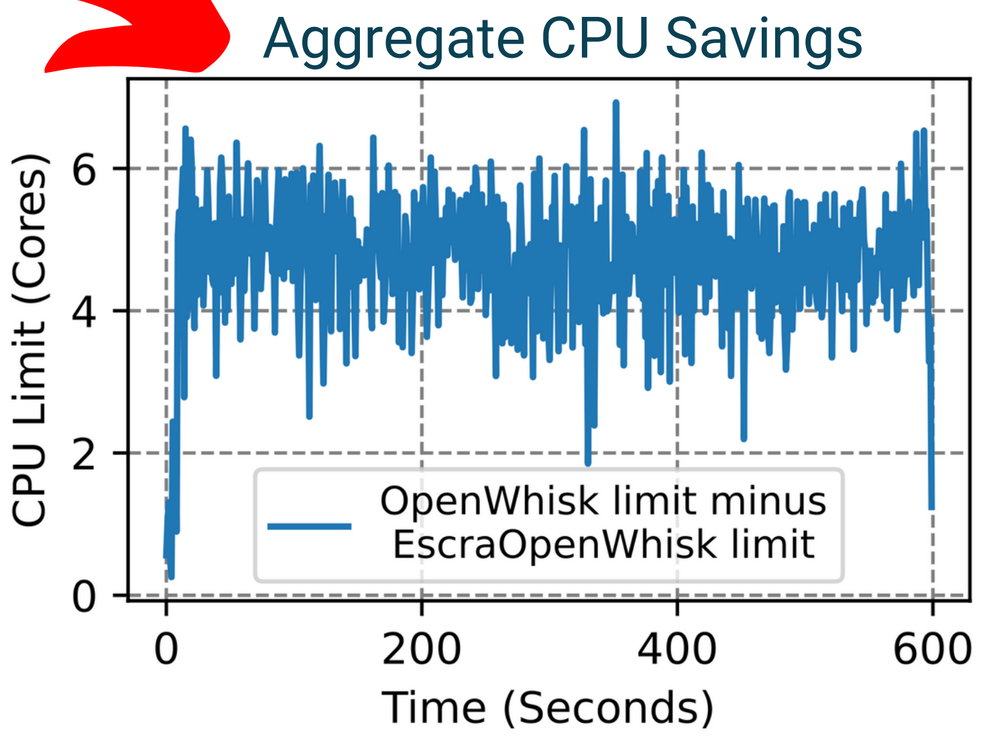
University of Colorado **Boulder**

ImageProcess

Serverless

Difference!



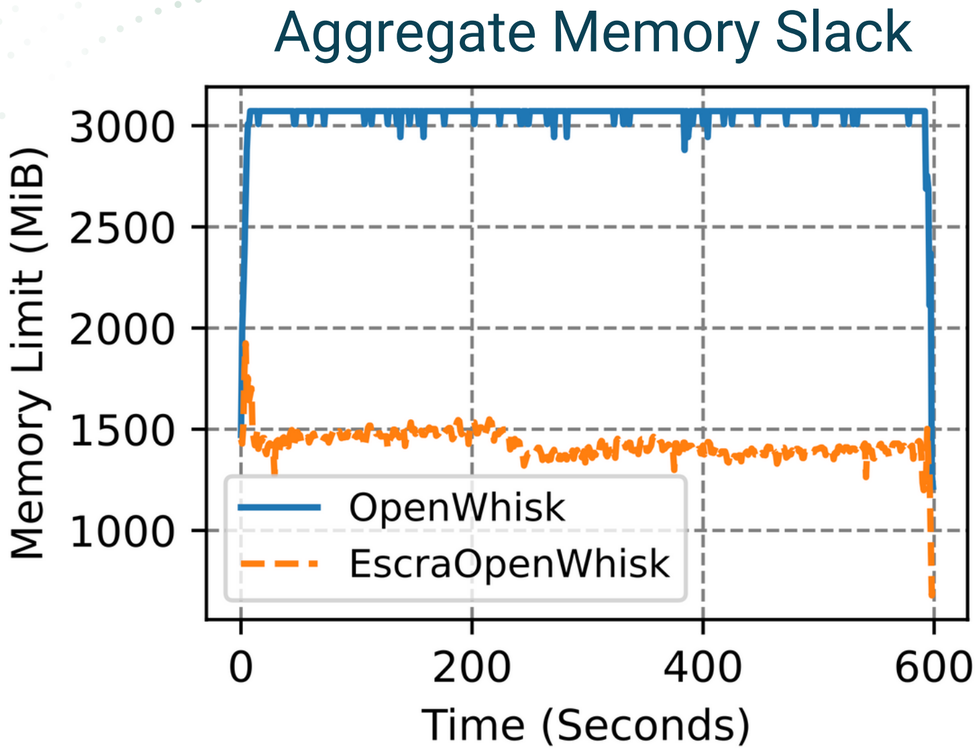


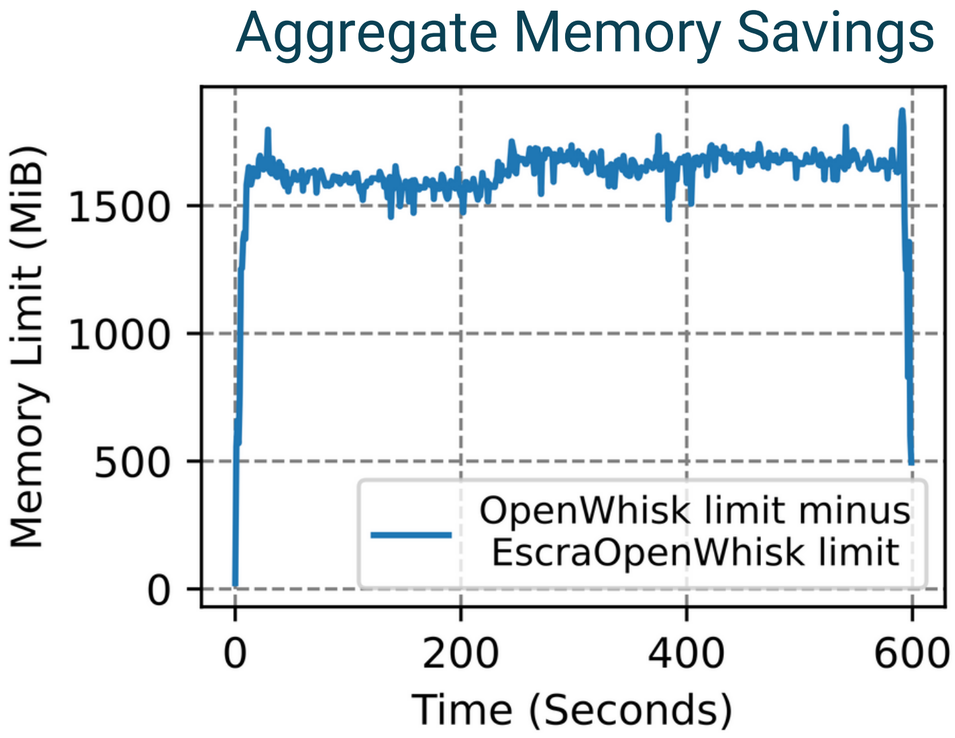
- **Escra + Openwhisk avg vCPU limit: 7 vCPU**
- **Openwhisk avg vCPU limit: 12v CPU**

University of Colorado **Boulder**

ImageProcess

Serverless



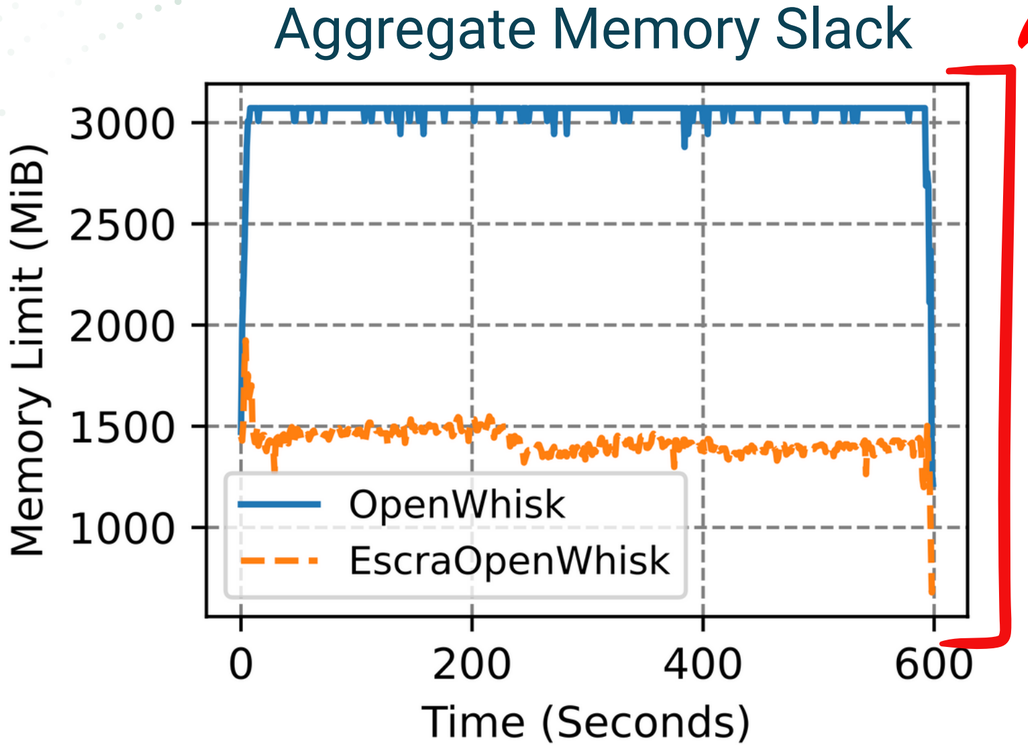


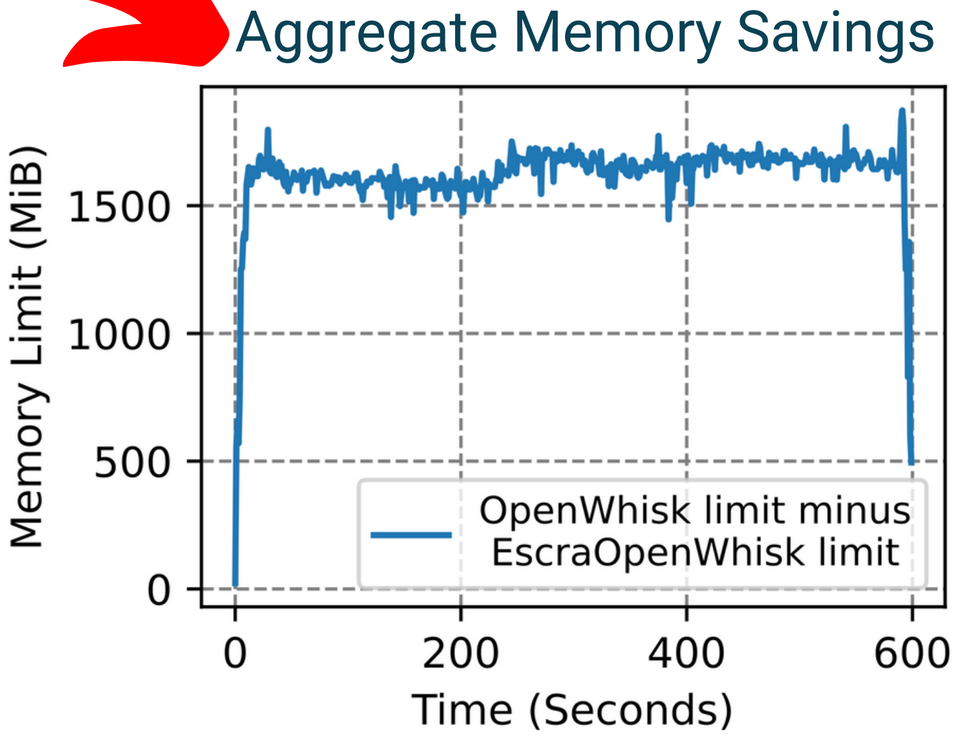
University of Colorado **Boulder**

ImageProcess

Serverless

Difference!



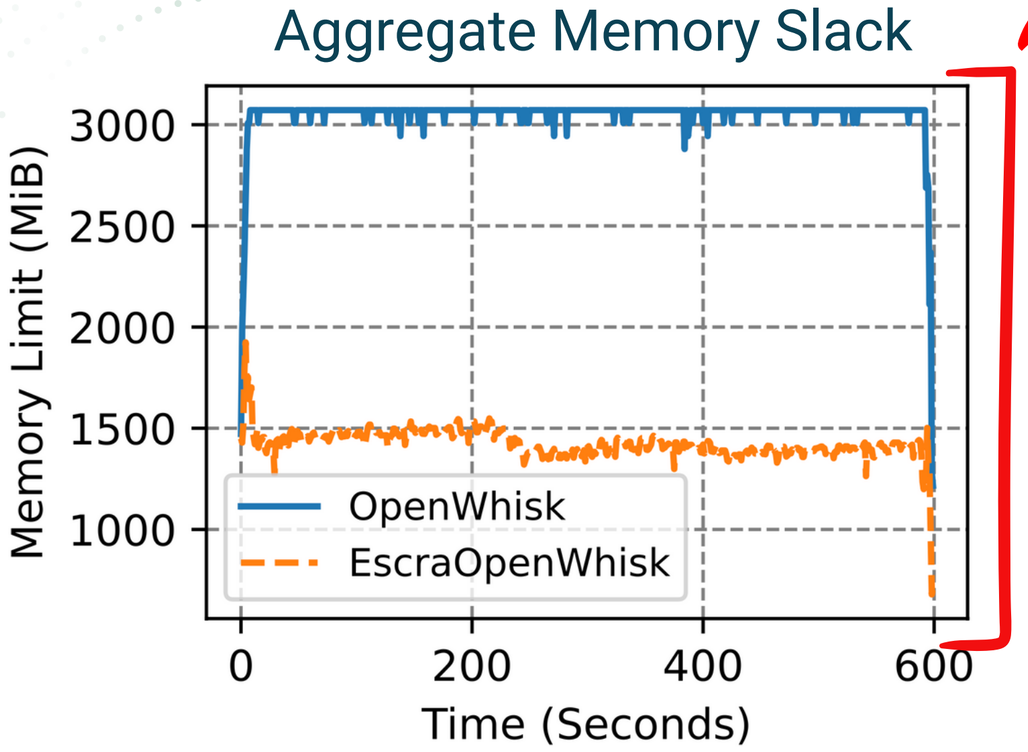


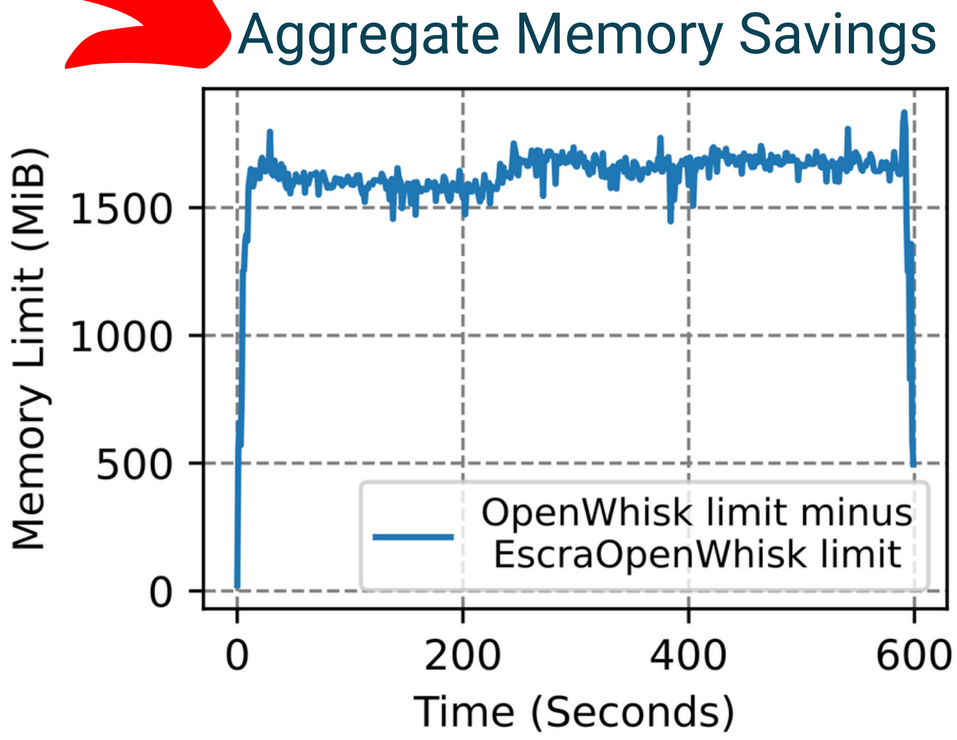
University of Colorado **Boulder**

ImageProcess

Serverless

Difference!



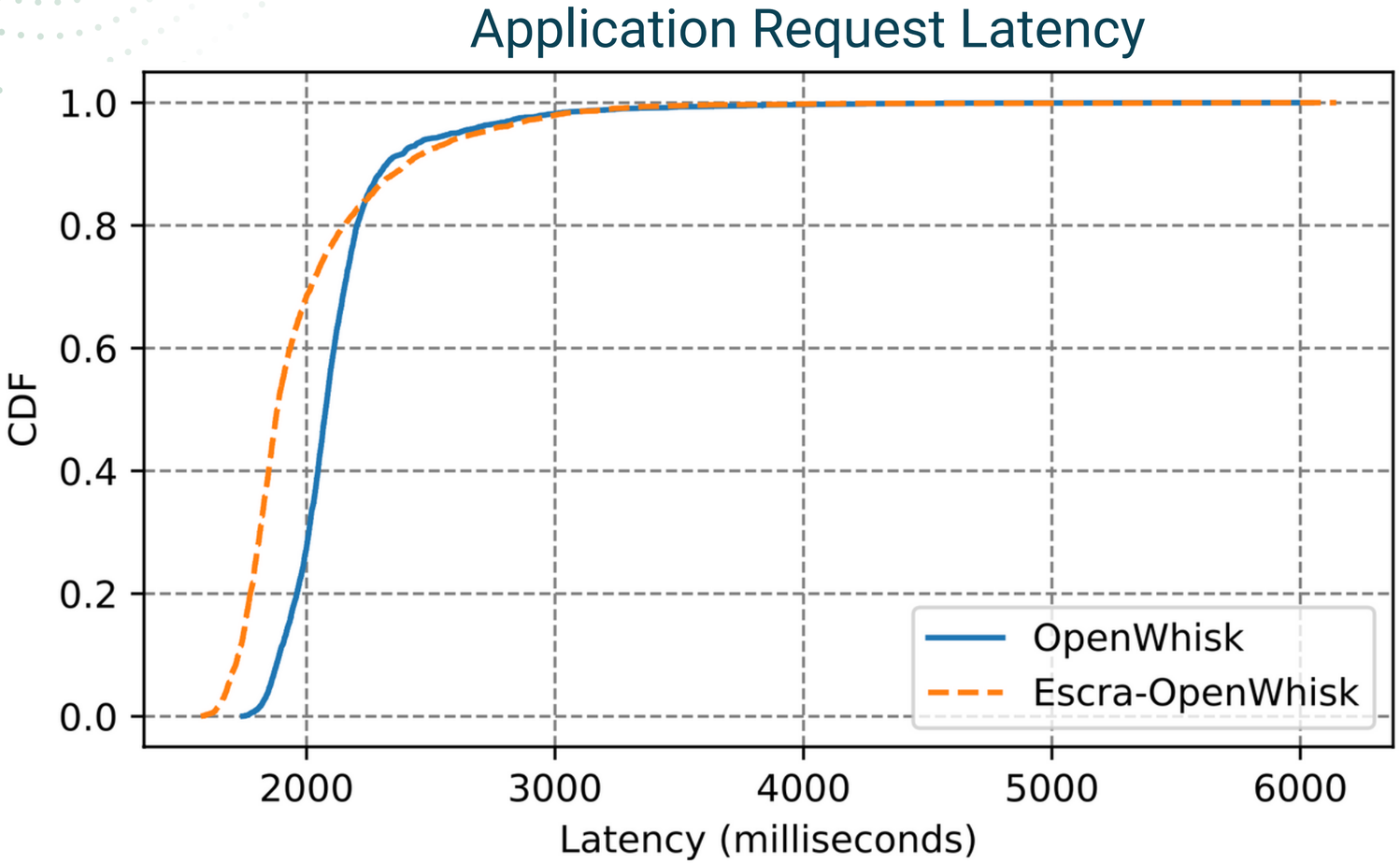


Avg. memory savings: 1550 MiB

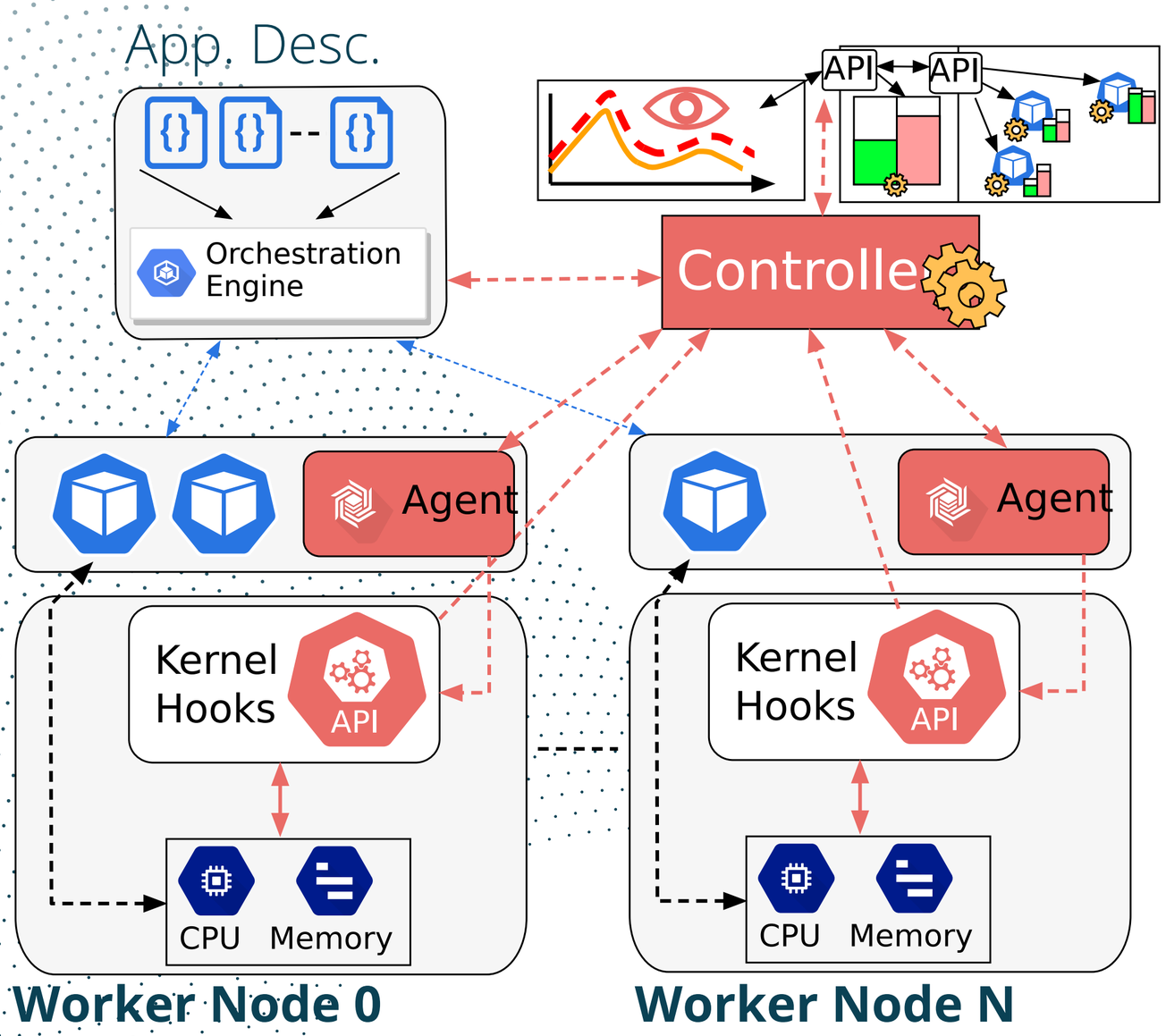
University of Colorado **Boulder**

ImageProcess

Serverless



- 80% of the time, Escra + Openwhisk sees a strong performance gain
- 99th%-ile latency remains similar for both



In Conclusion...

- Fine-grained allocations allows us to react to, not predict, workload changes
- High efficiency and high performance!
- Outperforms state of the art!
- Significant resource saving in serverless deployments

University of Colorado **Boulder**

Questions?


github.com/gregcusack/Escra


@GregoryCusack


gregory.cusack@colorado.edu


linkedin.com/in/gregorycusack

ICDCS - July 11, 2022
Bologna, Italy

Huge Thank You to:

Maziyar Nazari,
Sepideh Goodarzy,
Erika Hunhoff,
Prerit Oberai,
Eric Keller,
Eric Rozner,
Richard Han

vmware®