Themis: Fair and Efficient GPU Cluster Scheduling

Kshiteej Mahajan[1], Arjun Balasubramanian[1], Arjun Singhvi[1],
Shivaram Venkataraman[1], Aditya Akella[1], Amar Phanishayee[2], Shuchi Chawla[1]

1 WISCONSIN UNIVERSITY OF WISCONSIN–MADISON

2 Microsoft Research

Deep Learning at a Large Enterprise

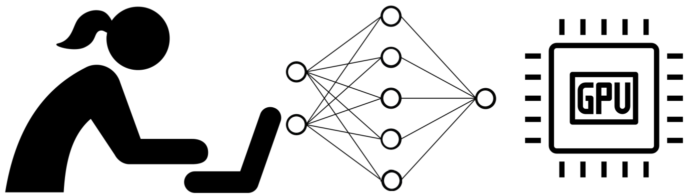
Speech, Image, Ads, NLP, Web Search…

Deep Learning at a Large Enterprise

Speech, Image, Ads, NLP, Web Search…

Cortana XBOX

Office 365 Bing

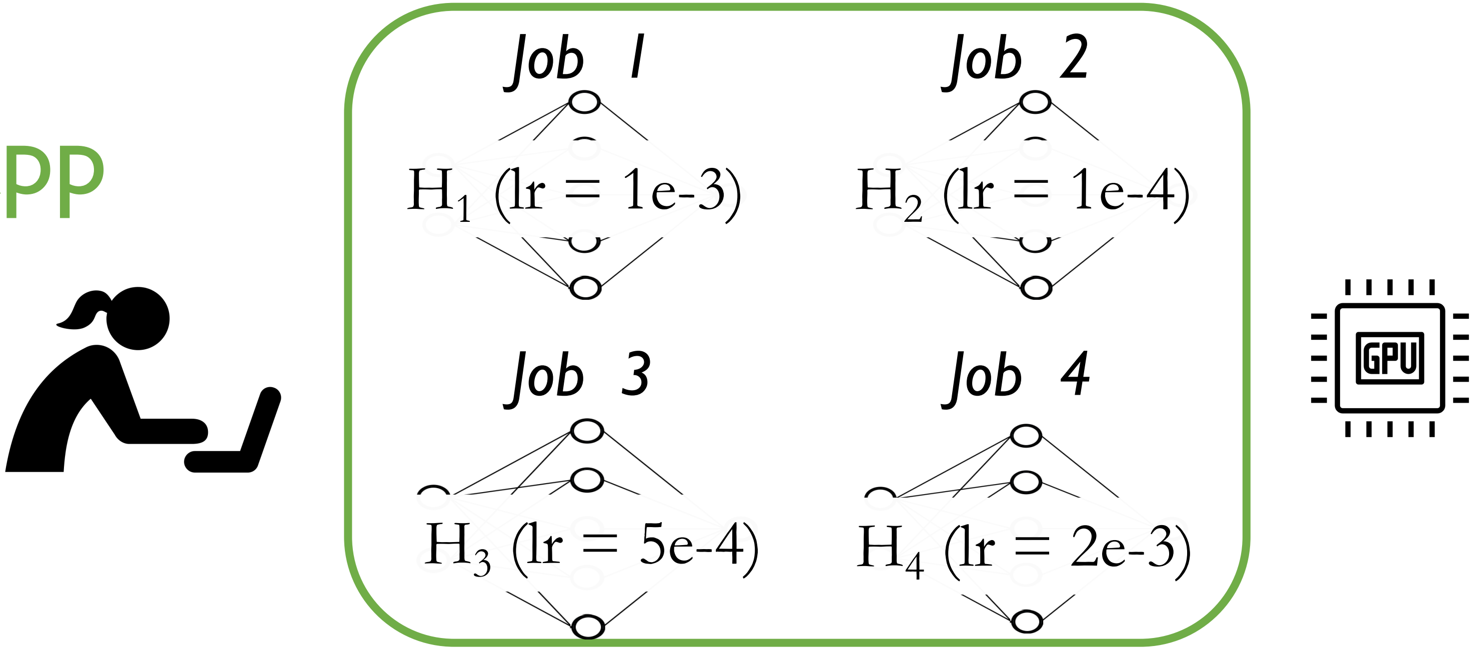


Innovate and Train newer DL models on GPUs

Deep Learning at a Large Enterprise

- Hyperparameter Optimization is typical – train same DL *model (M)* with different hyperparameters (H)

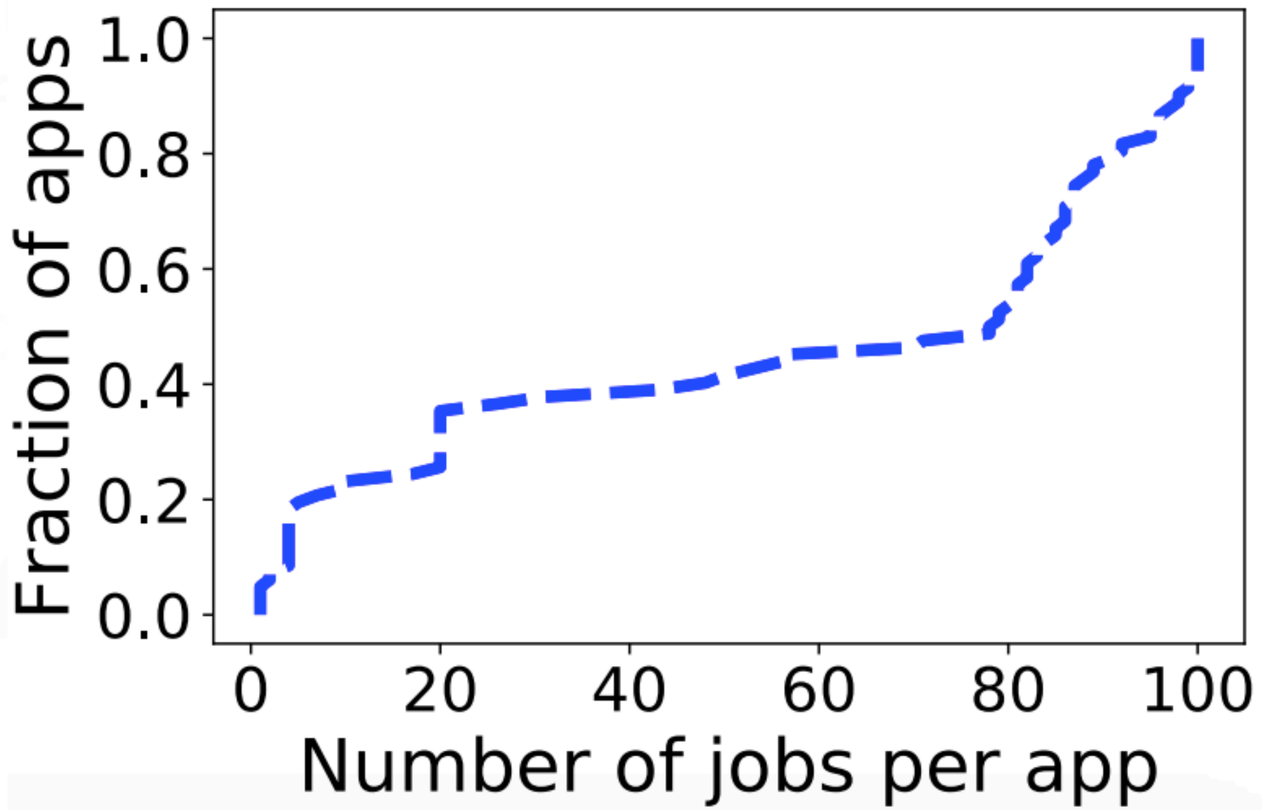
- DL App



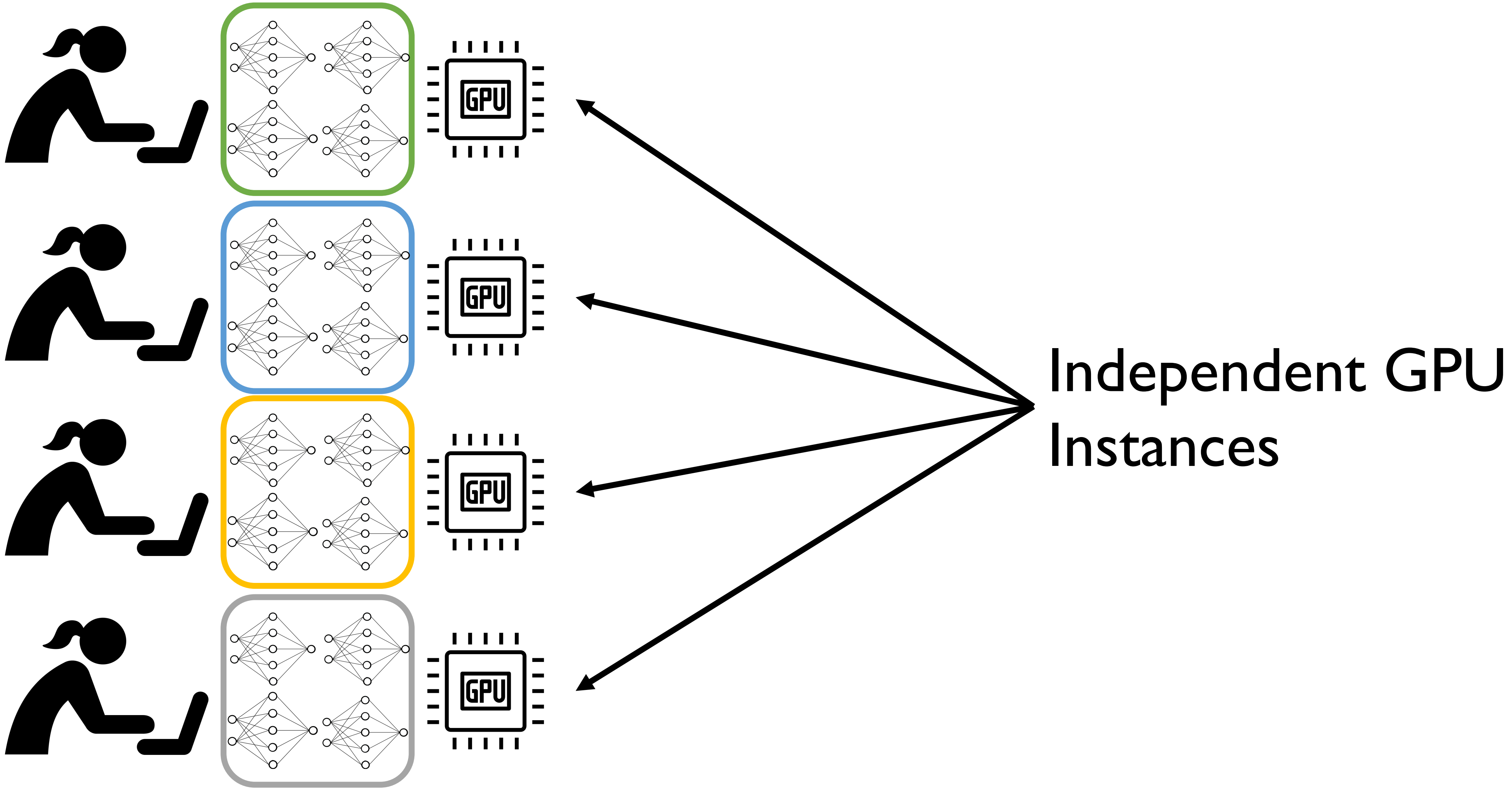
DL App, *Model M*

DL Apps at a Large Enterprise

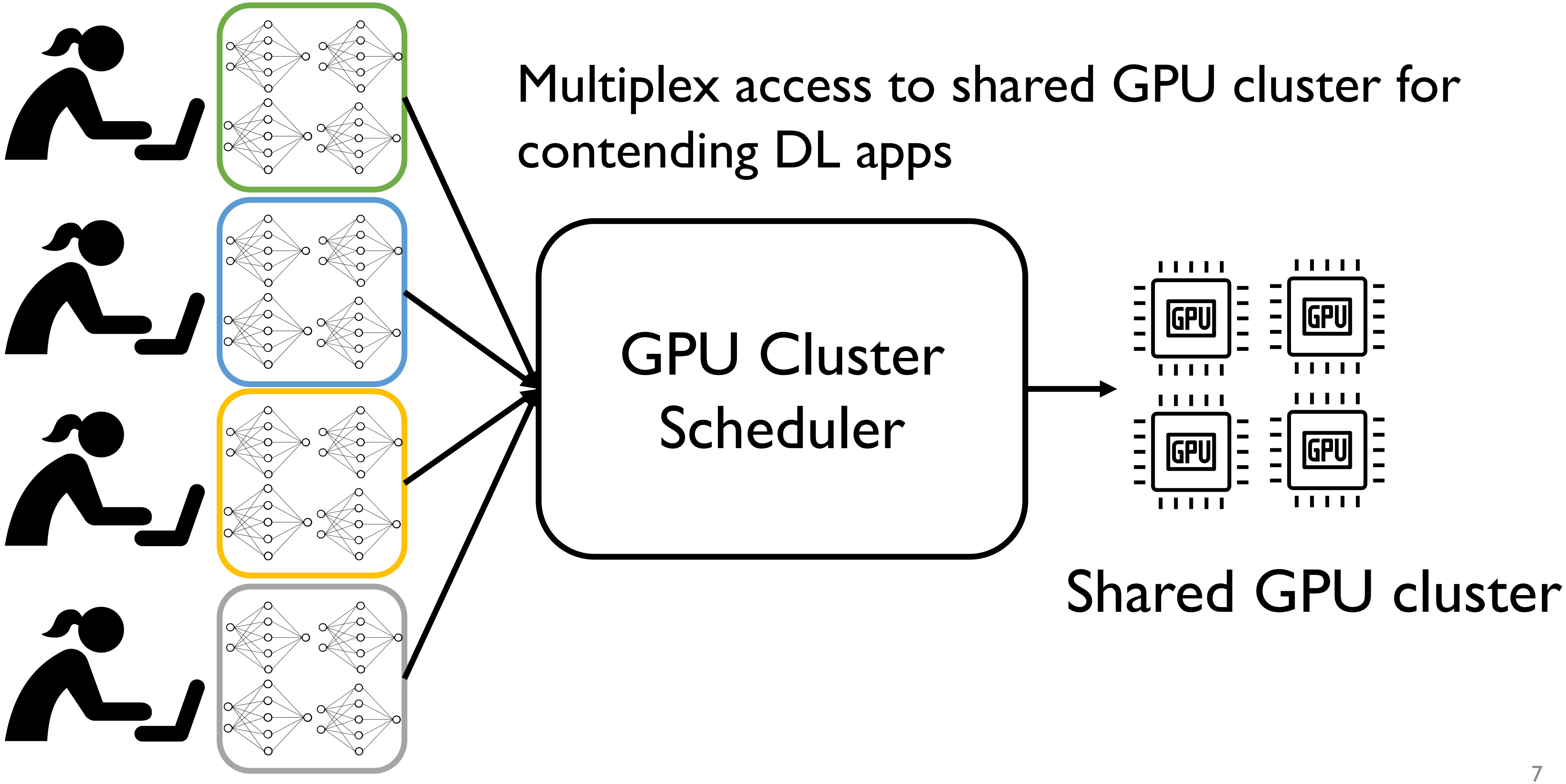
- Hyperparameter Optimization is typical –



Deep Learning at a Large Enterprise

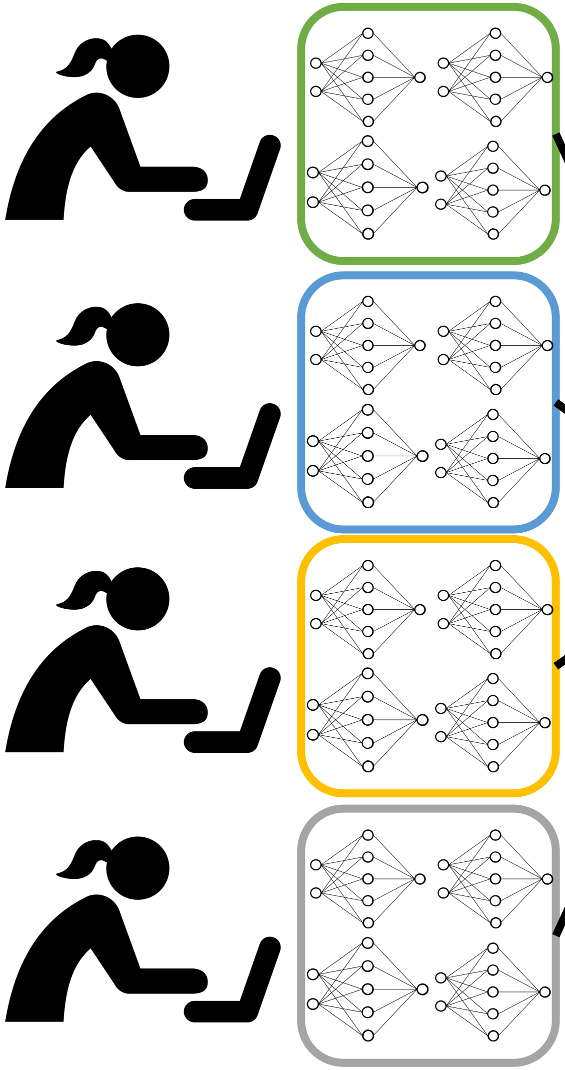


GPU Cluster Scheduler: Goal



GPU Cluster Scheduler: Goal

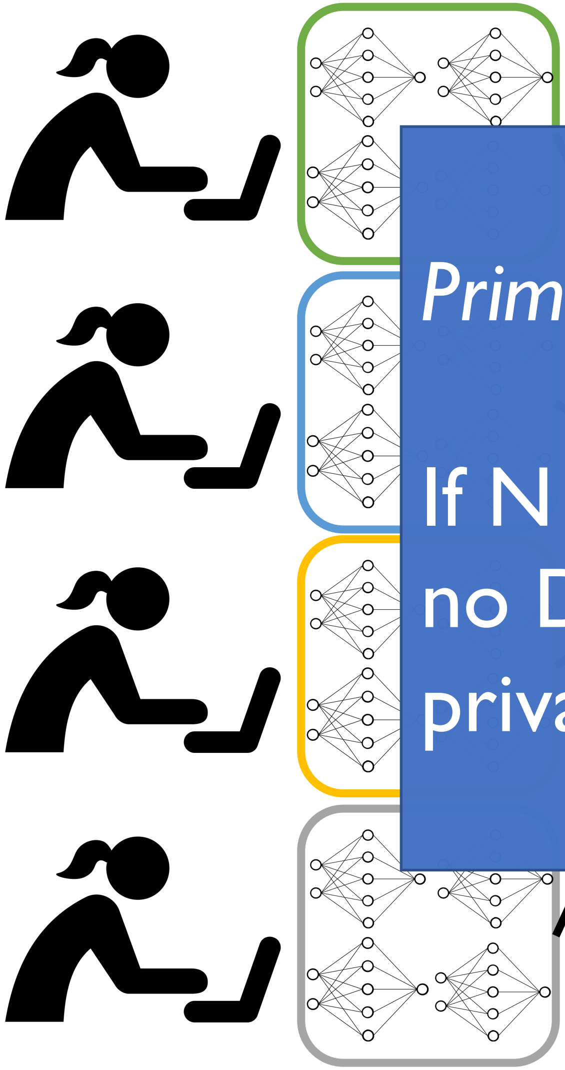
Multiplex access to shared GPU cluster for contending DL apps



Desired Goal – Incentivize sharing of GPU resources

Shared GPU cluster

GPU Cluster Scheduler: Goal



Multiplex access to shared GPU cluster for contending DL apps

Primary Goal – Sharing Incentive (SI)

If N DL apps are sharing a cluster then no DL app should run slower than on a private cluster with 1/N resources.

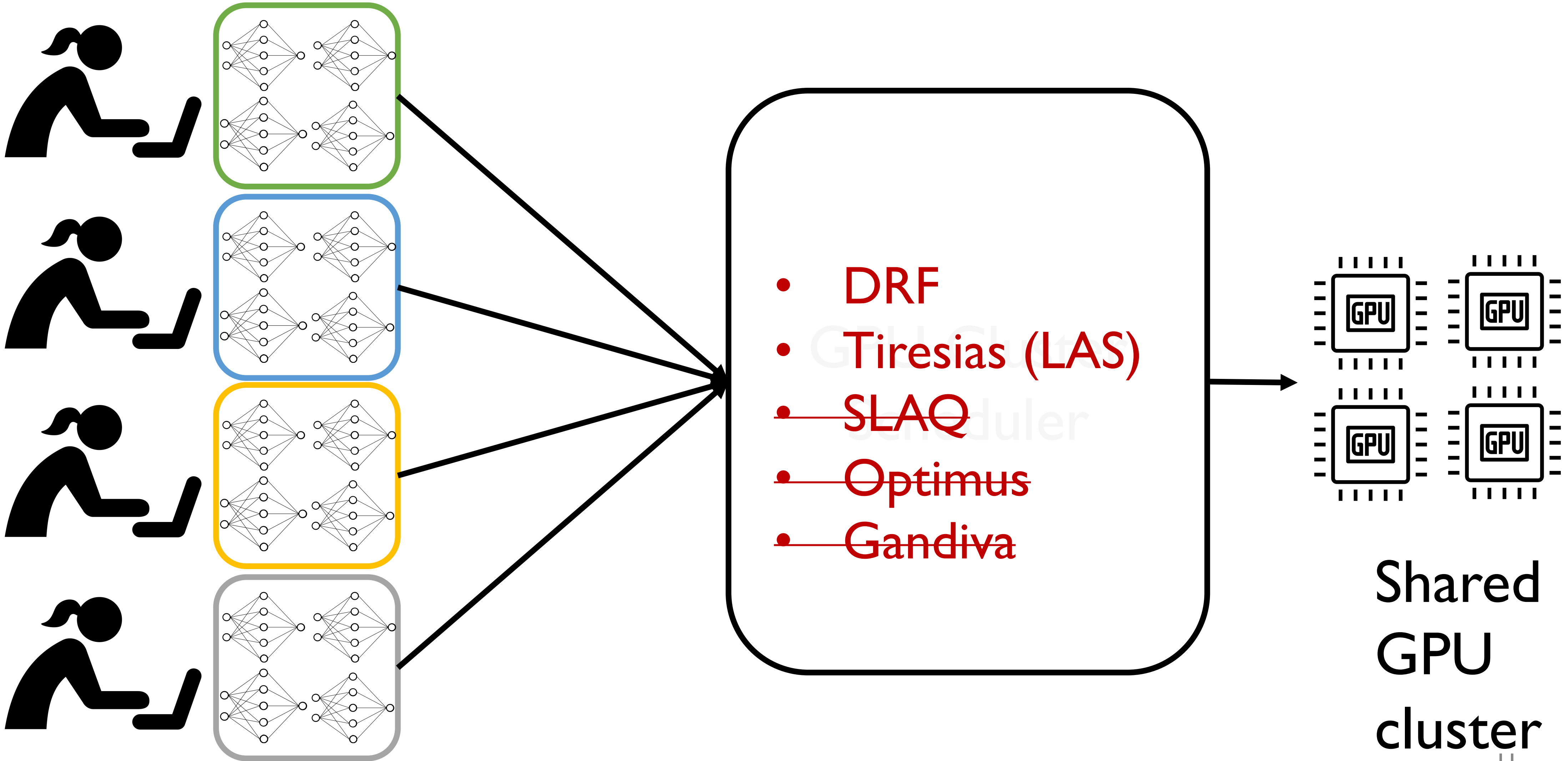
Shared GPU cluster

Overview

- Existing GPU Cluster Schedulers
  - Do not give Sharing Incentive
  - DL App Properties
  - Drawbacks
  - Requirements

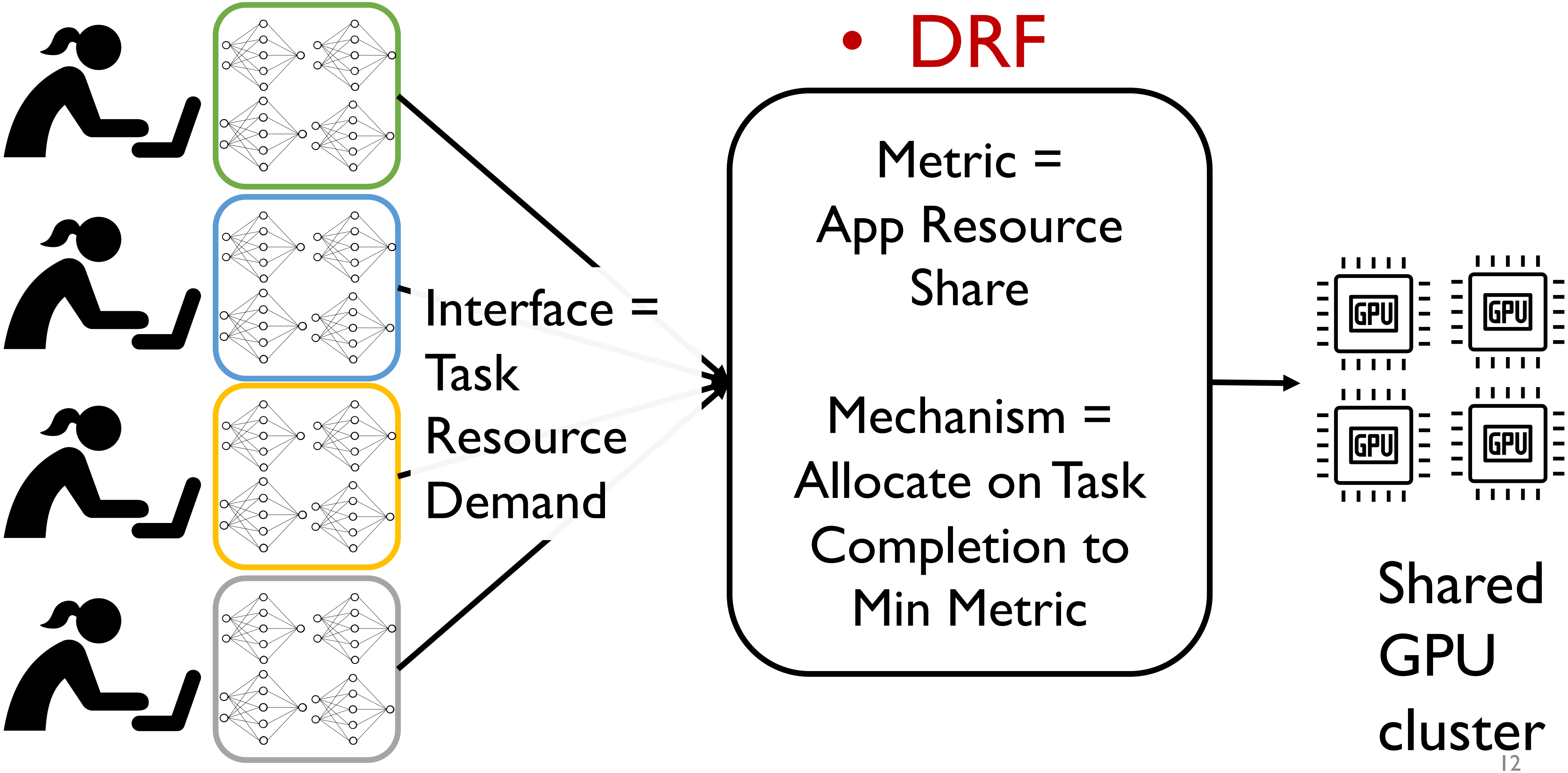
- Themis
  - Design
  - Implementation
  - Evaluation

Existing GPU Cluster Schedulers



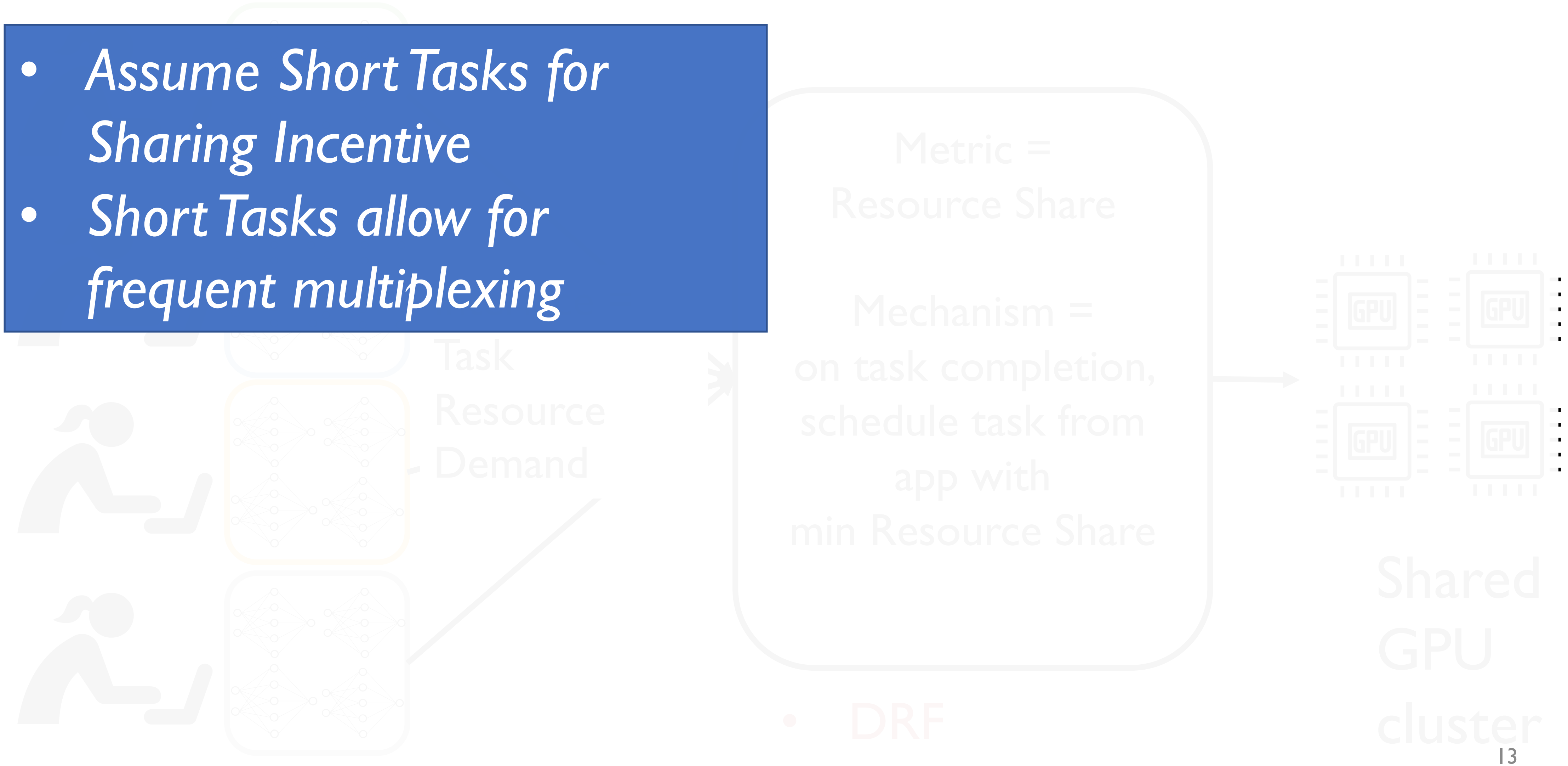
GPU Cluster Scheduler: Drawback 1

- DRF



GPU Cluster Scheduler: Drawback 1

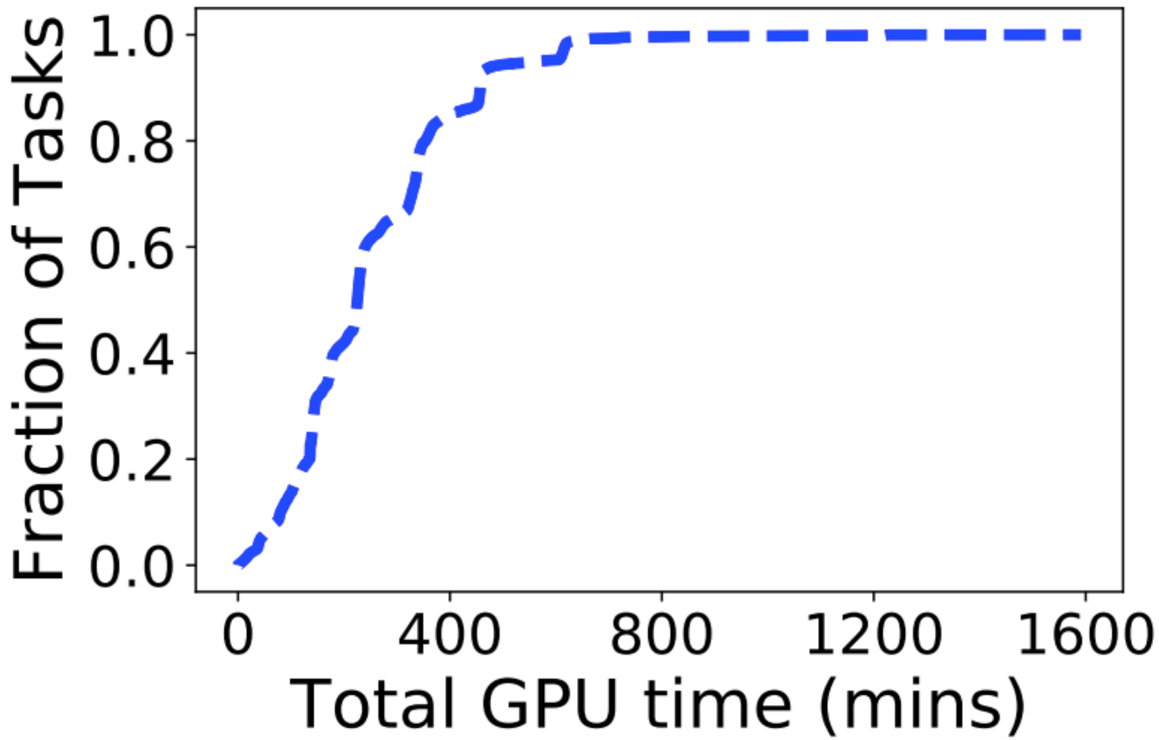
- *Assume Short Tasks for Sharing Incentive*
- *Short Tasks allow for frequent multiplexing*



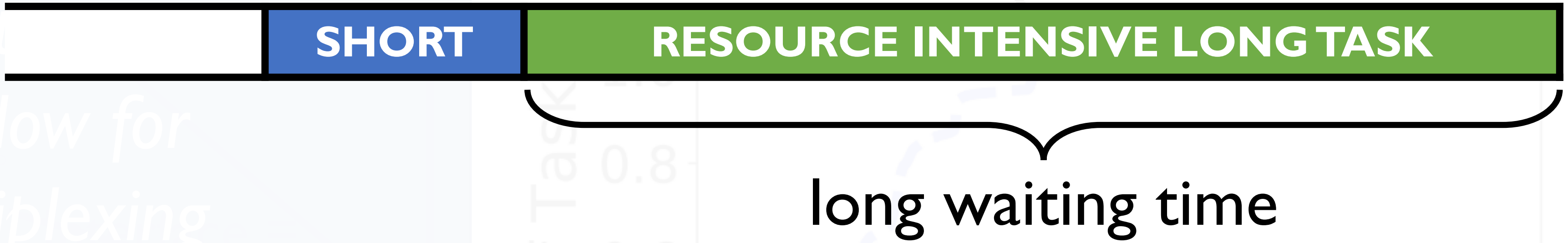
GPU Cluster Scheduler: Drawback 1

- *Assume Short Tasks for Sharing Incentive*
- *Short Tasks allow for frequent multiplexing*

- *ML median task duration – 3.75 hours*
- *Lot of apps with 5X shorter and 5X longer tasks*



GPU Cluster Scheduler: Drawback 1



- *Long waiting time for Short Apps*
- *No SI for Short Apps*

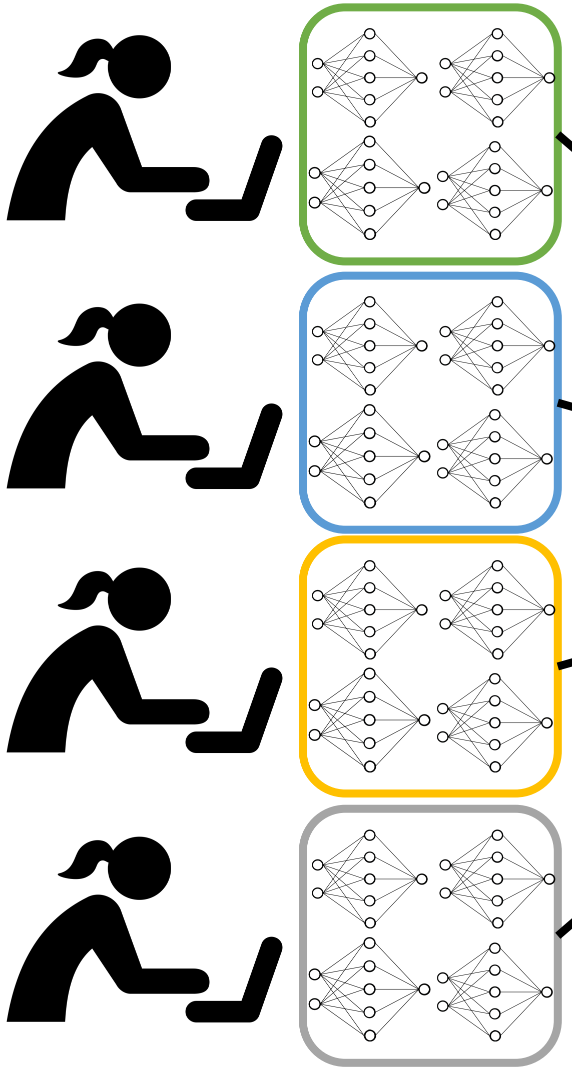
GPU Cluster Scheduler: Requirement 1

schedule short task on preemption of long task

- *SI for ML Apps => Preemption is necessary*
- *Allocate any task for at most lease duration*

GPU Cluster Scheduler: Drawback 2

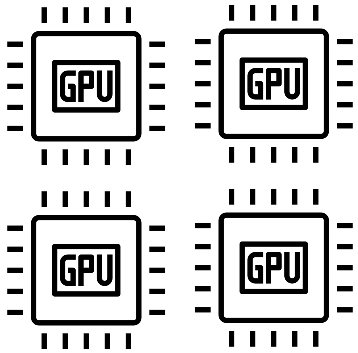
- Tiresias (LAS)



Interface =
Task
Resource
Demand

Metric =
App Attained
Service
(= #GPUs * time)

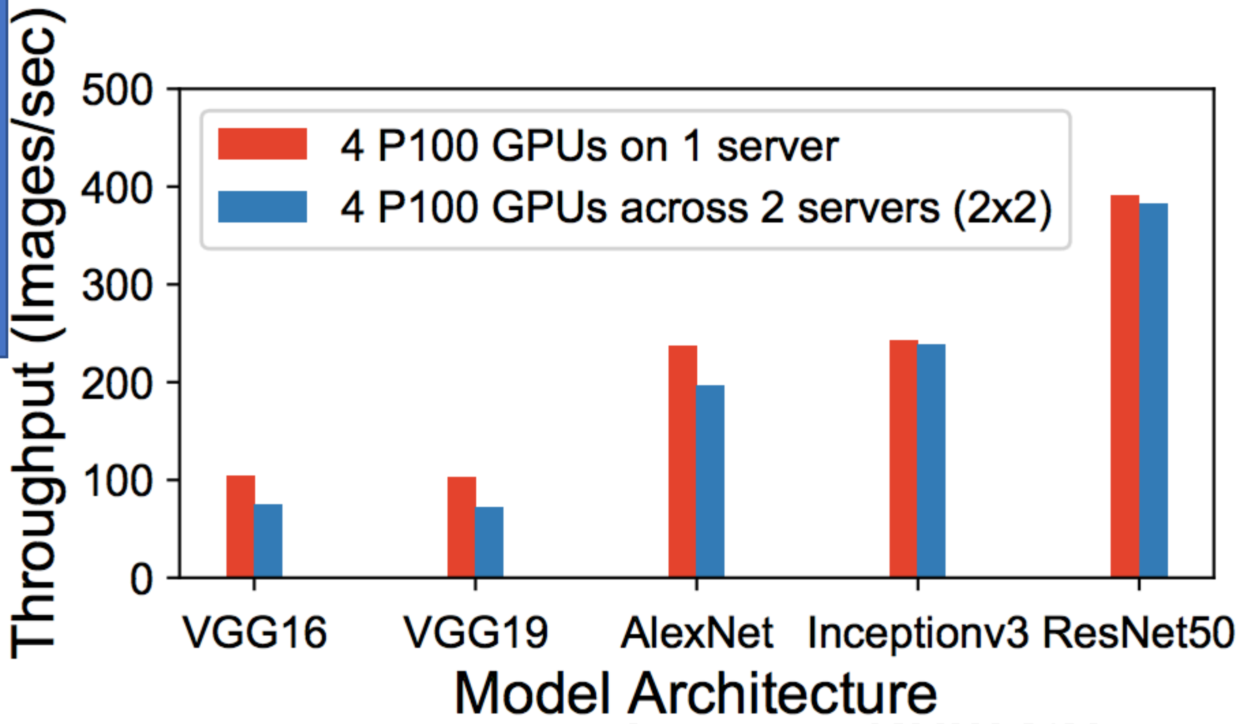
Mechanism =
Allocate for *lease*
duration to Min
Metric



Shared
GPU
cluster

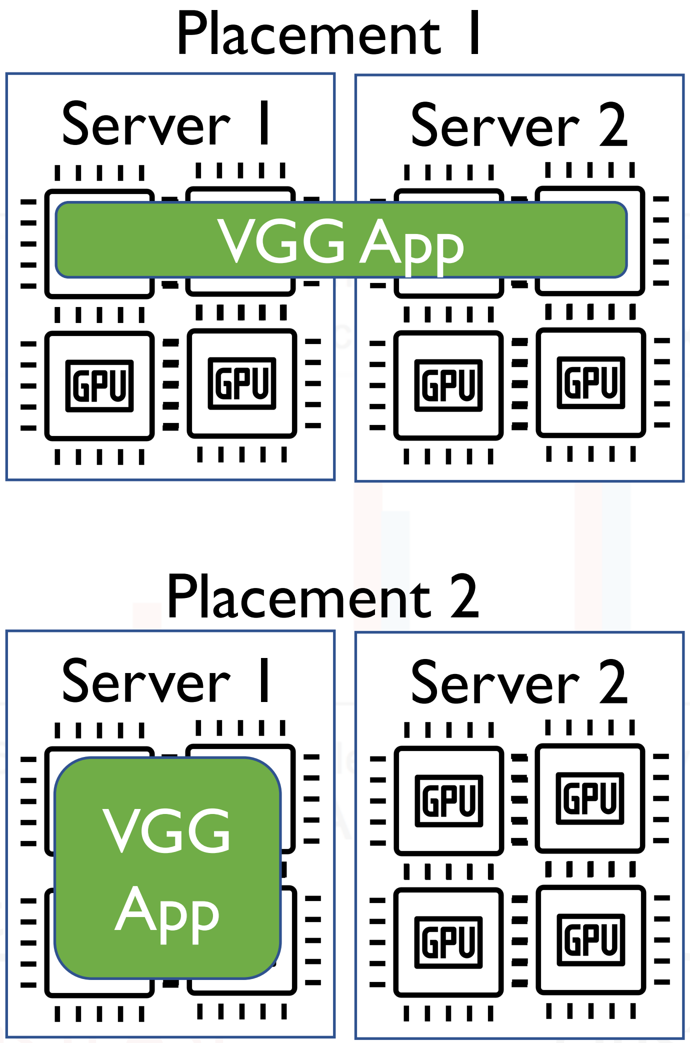
GPU Cluster Scheduler: Drawback 2

- *DL apps have a placement preference*
- *E.g.: VGG model family prefers dense placement*



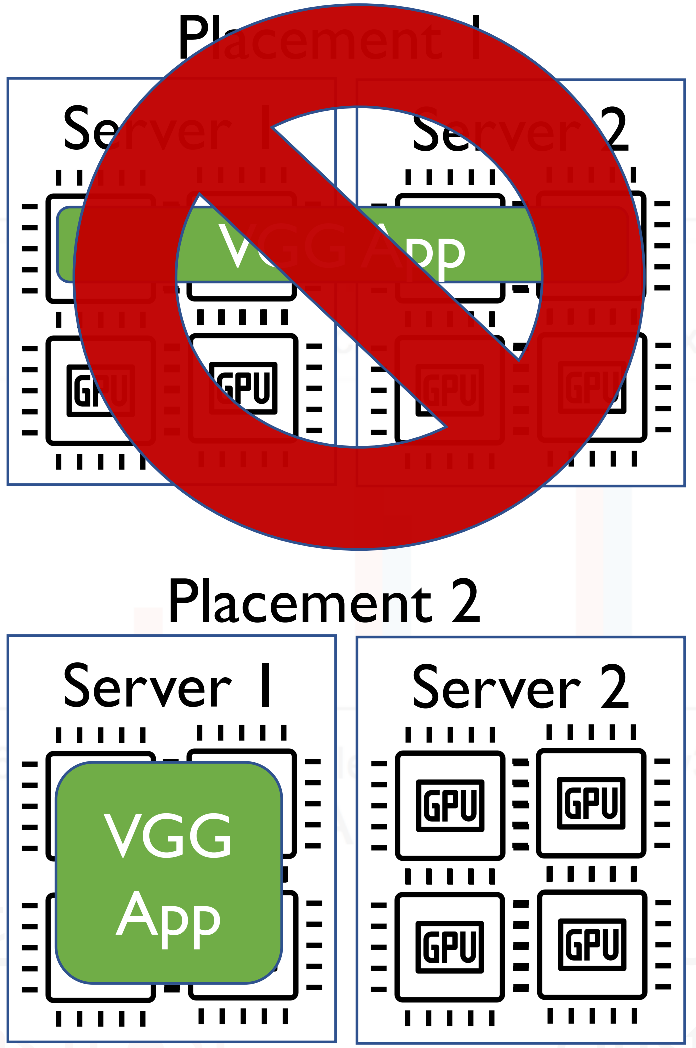
GPU Cluster Scheduler: Drawback 2

- *Attained Service is equal in both placements (= 4 GPUs * time)*
- *Both placements are equivalent*
- *Poor placement => slower execution time*
- *VGG app would rather prefer its own server*
- *No SI*



GPU Cluster Scheduler: Drawback 2

- *Binary Placement Enforcement – Strict Consolidation (wait for Placement 2)*
- *Partial Progress can be made with Placement 1*
- *Long wait time without progress*
- *SI is violated*



GPU Cluster Scheduler: Requirement 2

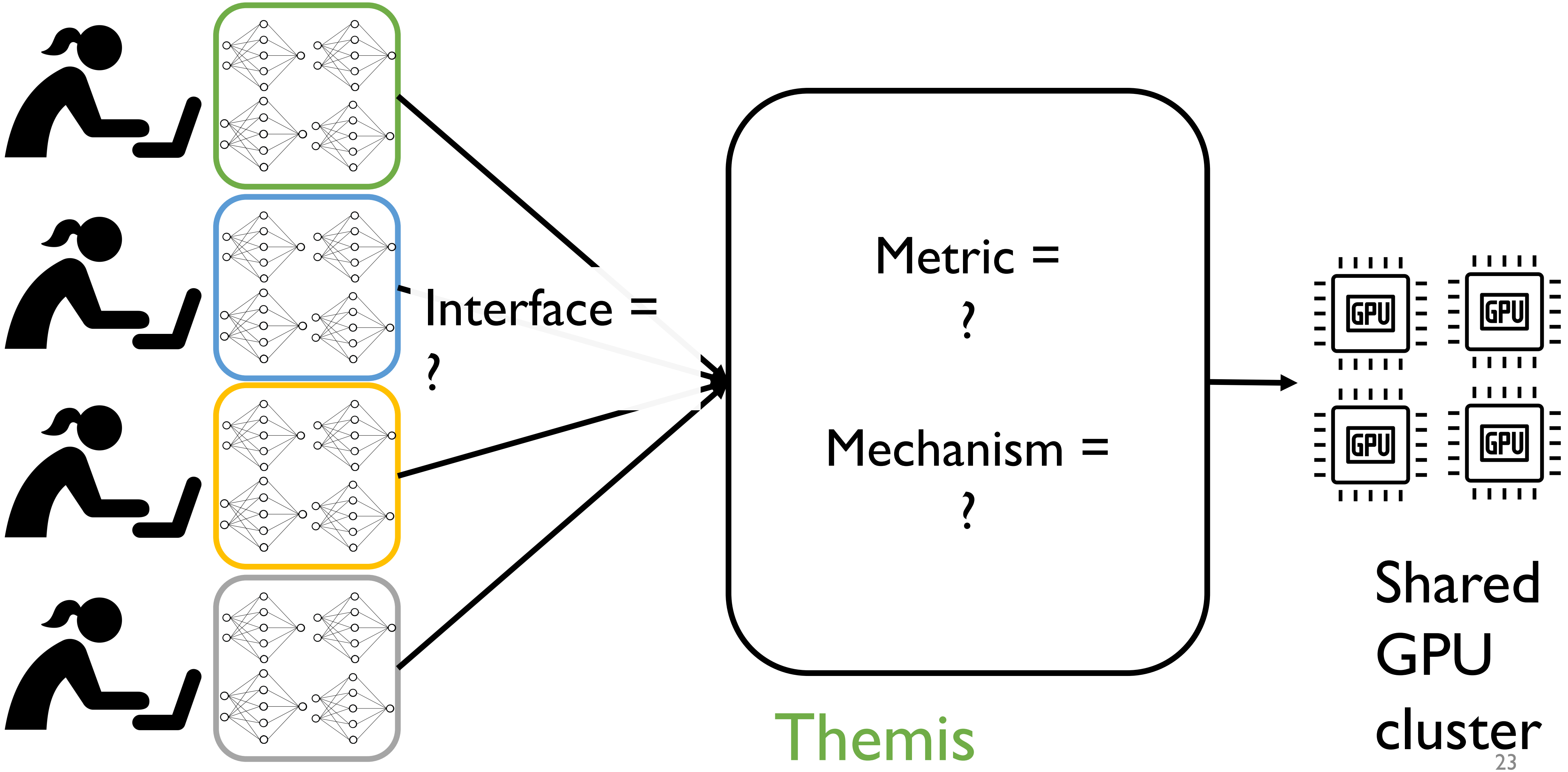
- *SI for ML Apps =>*
  Fine-grained Placement Preference

Overview

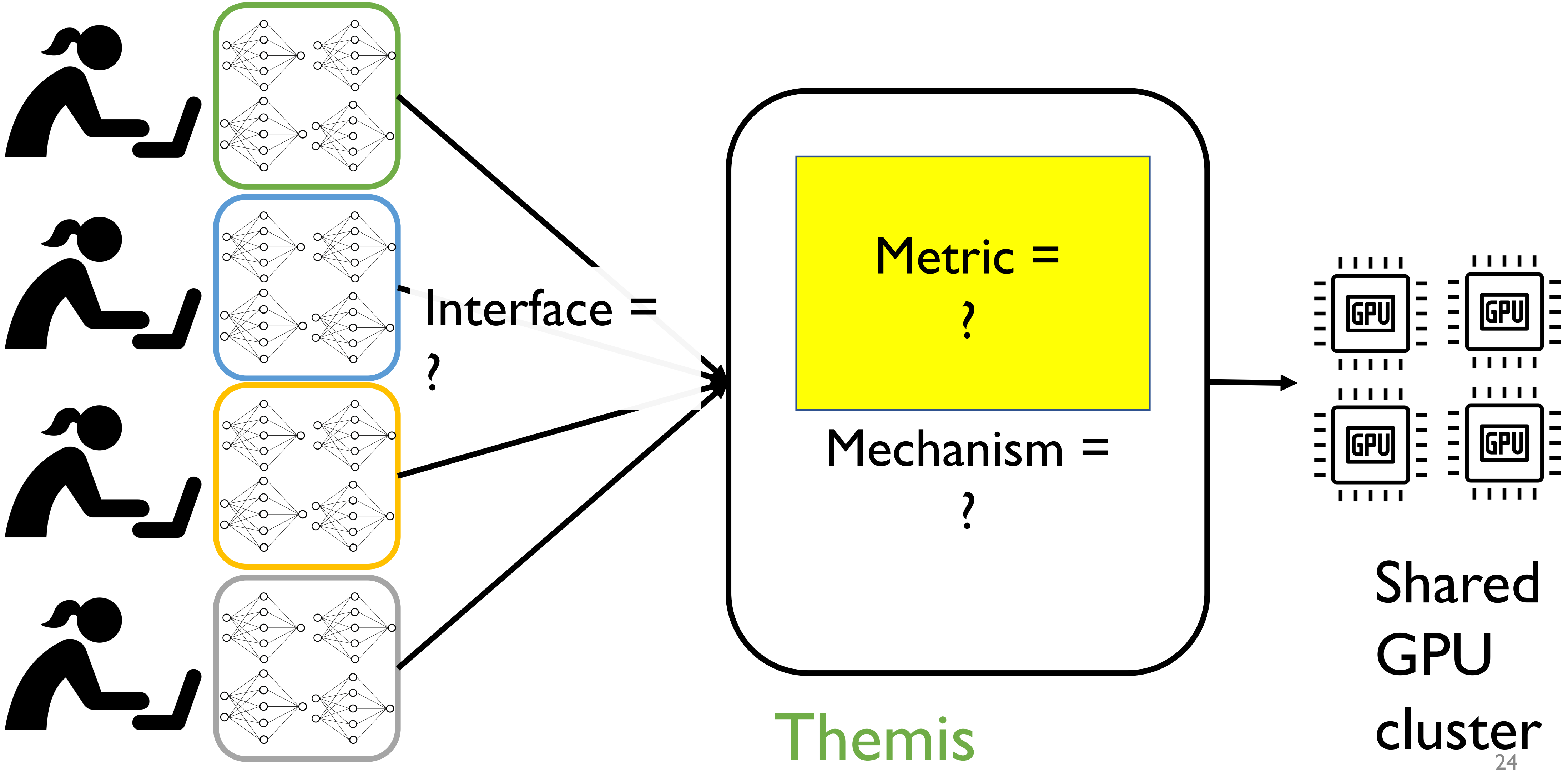
- Existing GPU Cluster Schedulers
  - Do not give Sharing Incentive
  - DL App Properties
  - Drawbacks
  - Requirements

- Themis
  - Design
  - Implementation
  - Evaluation

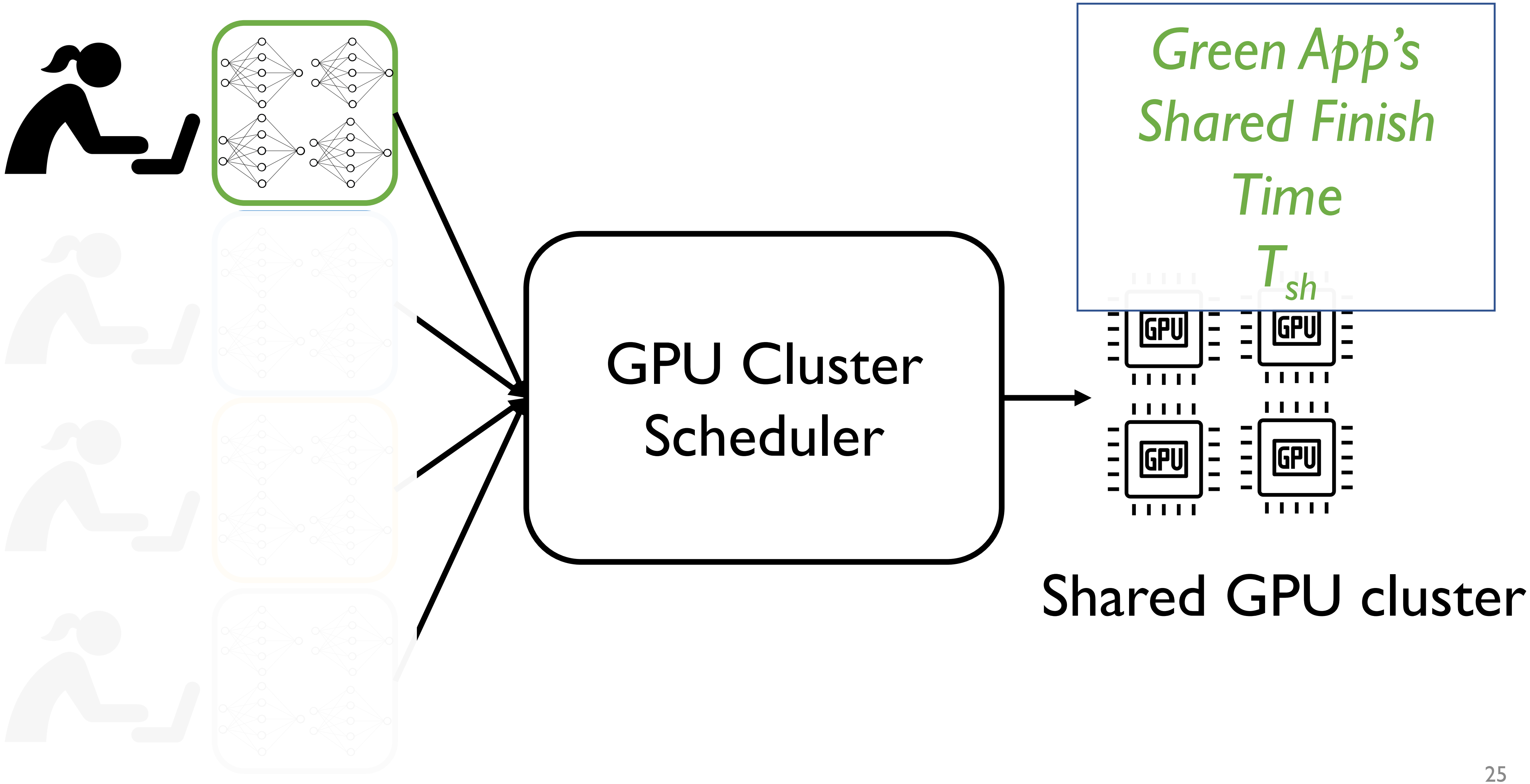
Towards a new GPU Cluster Scheduler



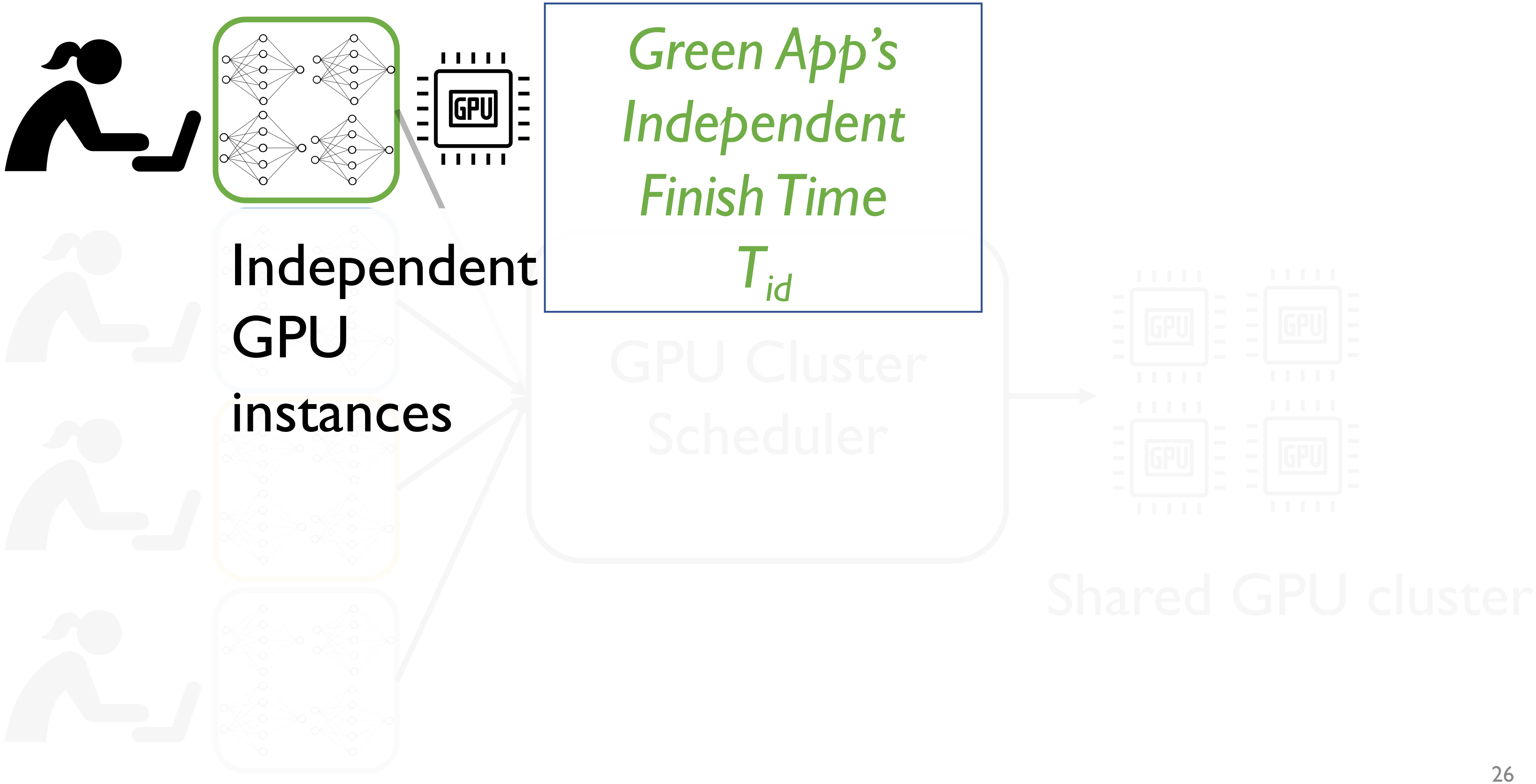
Themis: Metric



Themis: Metric



Themis: Metric



Themis: Metric

Green App's Independent Running Time T_{id}

>=

Green App's Shared Running Time T_{sh}

Primary Goal – Sharing Incentive (SI)

$T_{sh} <= T_{id}$

Themis: Finish-Time Fairness Metric

- $\rho = T_{sh} / T_{id}$
  - T_{sh}: finish-time of app in shared cluster
  - T_{id}: finish-time of app in exclusive 1/N share of cluster
  - N: Average contention during app lifetime

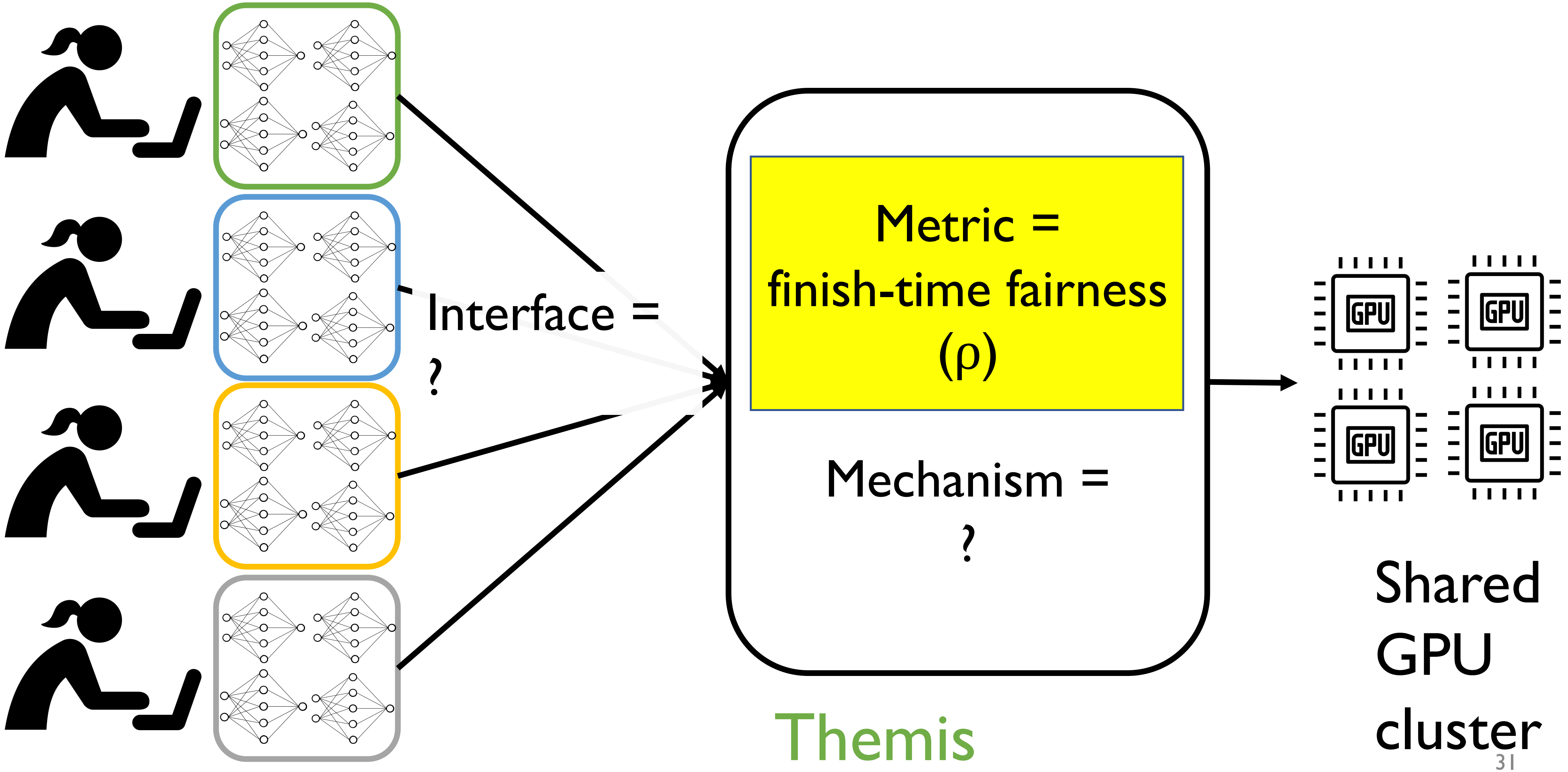
Themis: Finish-Time Fairness Metric

- $\rho = T_{sh} / T_{id}$
  - T_{sh}: finish-time of app in shared cluster
  - T_{id}: finish-time of app in exclusive 1/N share of cluster
  - N: Average contention during app lifetime
- SI: for all apps, $\rho <= 1$

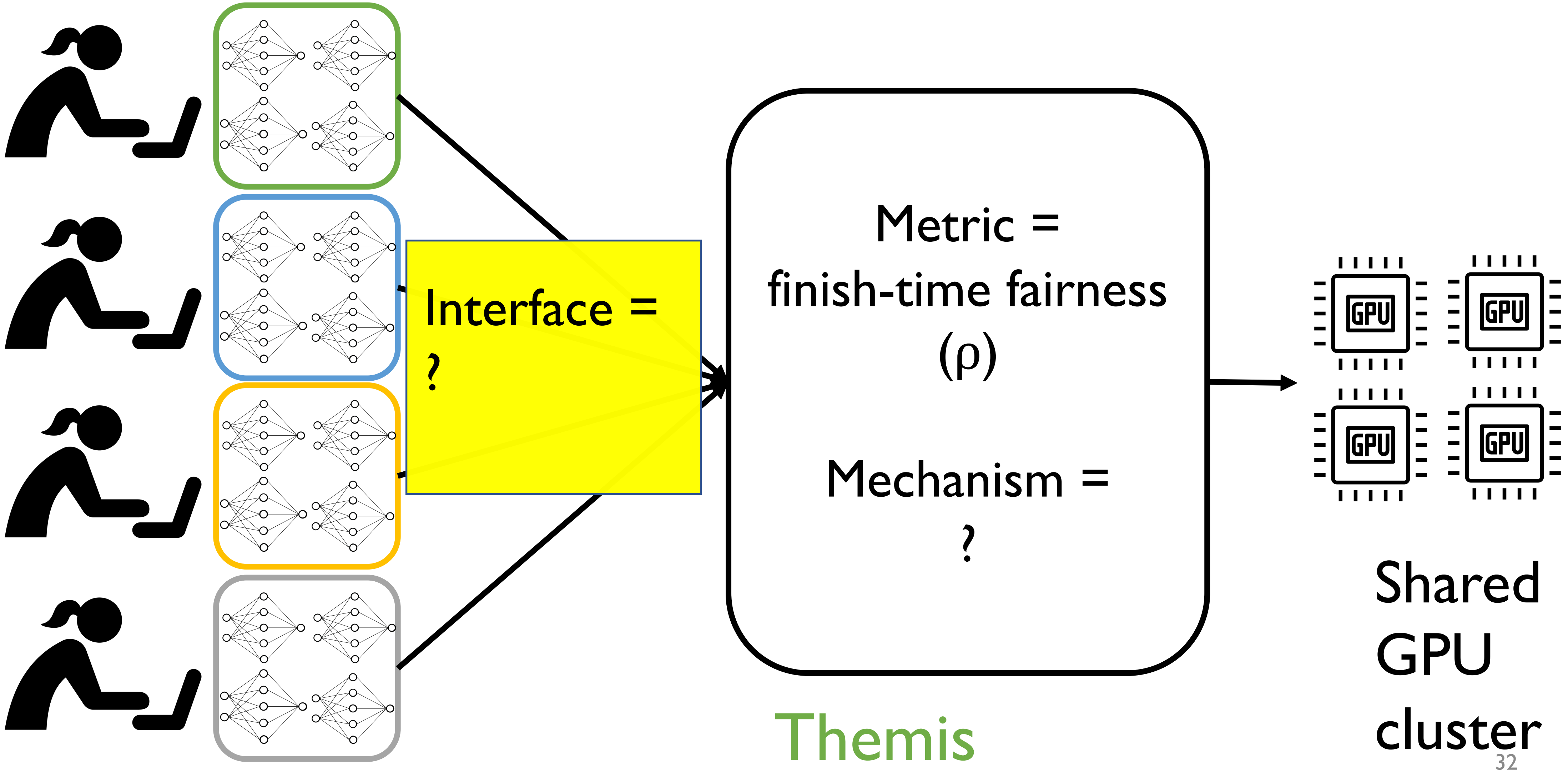
Themis: Finish-Time Fairness Metric

- $\rho = T_{sh} / T_{id}$
  - T_{sh}: finish-time of app in shared cluster
  - T_{id}: finish-time of app in exclusive 1/N share of cluster
  - N: Average contention during app lifetime
- SI: for all apps, ρ <= 1
- Fine-Grained Placement Preferences –
  - Excessive queueing or bad placements worsens T_{sh} and hence ρ

Themis: Metric



Themis: Interface



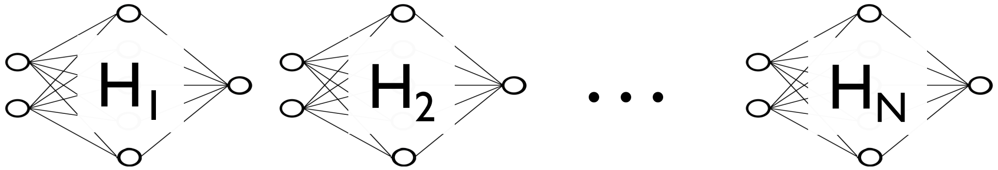
Themis: Interface

- Key Purpose: Enable book-keeping of ρ
- Who calculates ρ – the app or the scheduler?

Themis: Interface

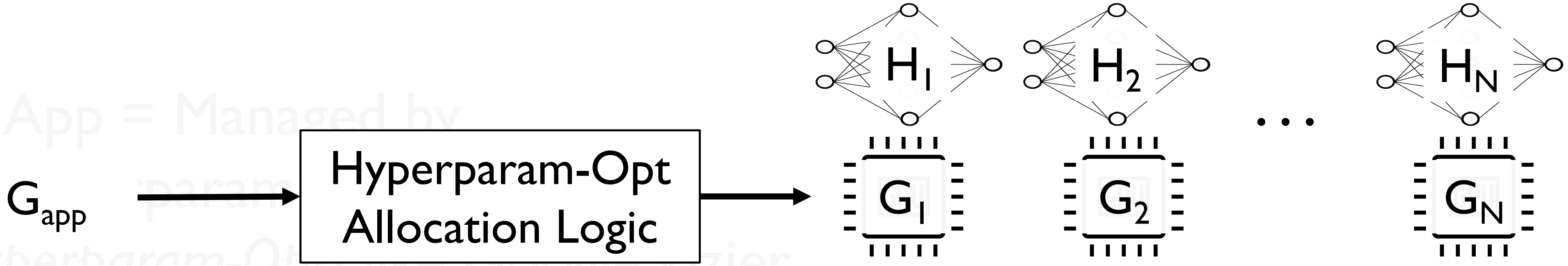
- DL App = Managed by
  an Hyperparameter Optimizer
  (*Hyperparam-Opt*) like Google Vizier

Themis: Interface



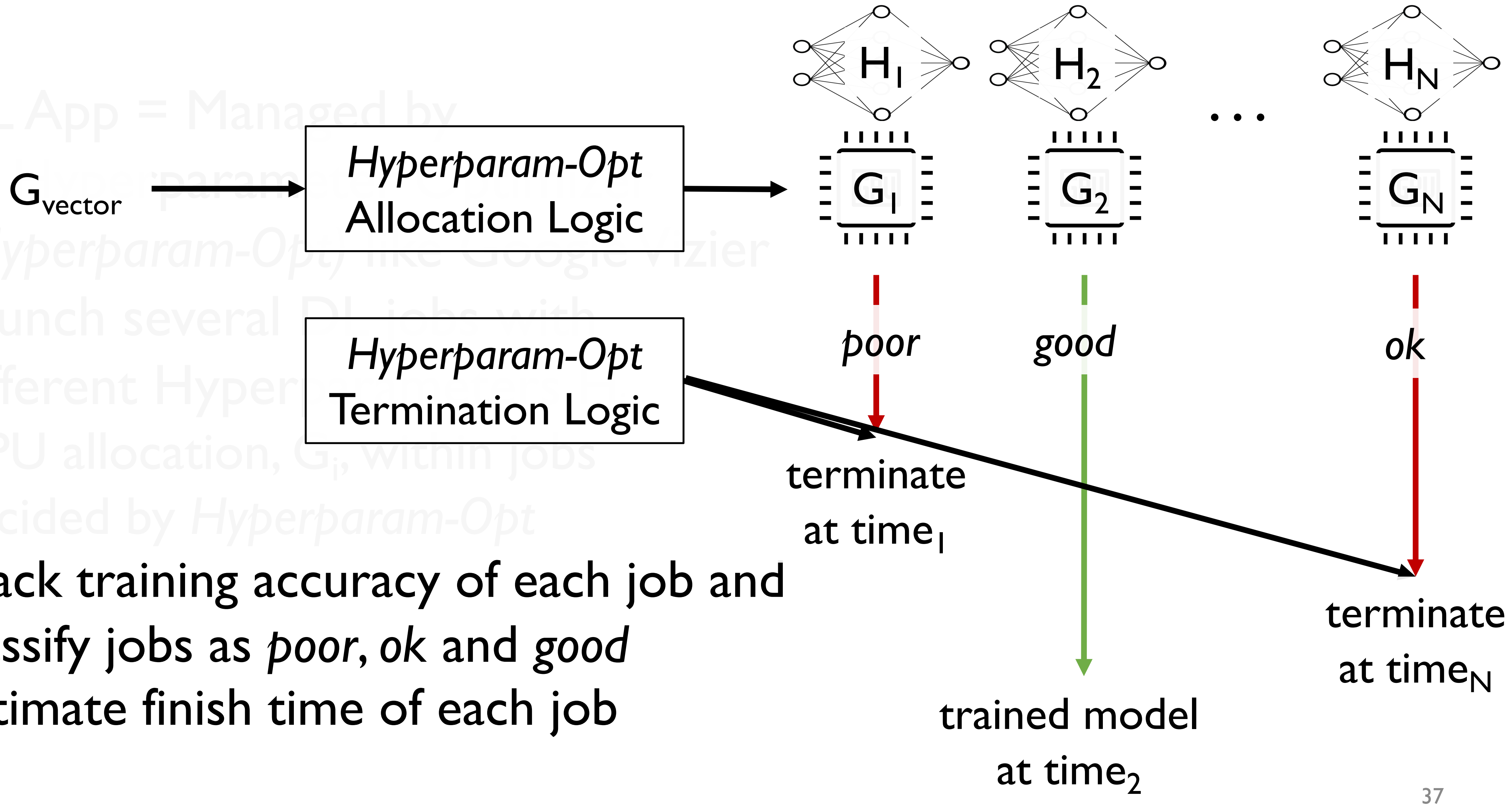
- DL App = Managed by an Hyperparameter Optimizer (*Hyperparam-Opt)* like Google Vizier
- Launch several DL jobs with different Hyperparameters H_i

Themis: Interface



- DL App = Managed by an Hyperparameter Optimizer (*Hyperparam-Opt*) like Google Vizier
- Launch several DL jobs with different Hyperparameters H_i
- GPU allocation, G_i, within jobs decided by *Hyperparam-Opt*

Themis: Interface

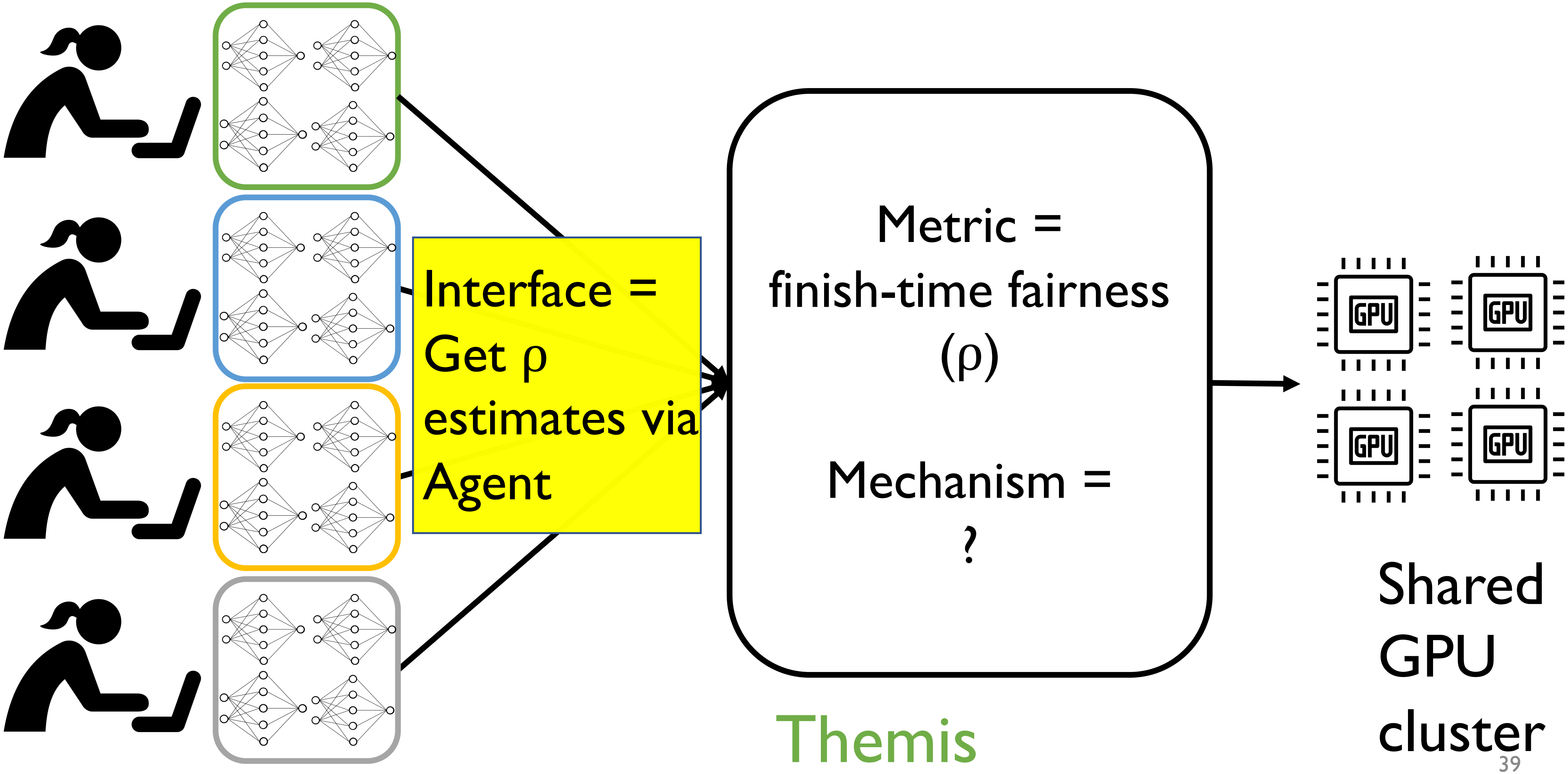


- Track training accuracy of each job and classify jobs as *poor*, *ok* and *good*
- Estimate finish time of each job

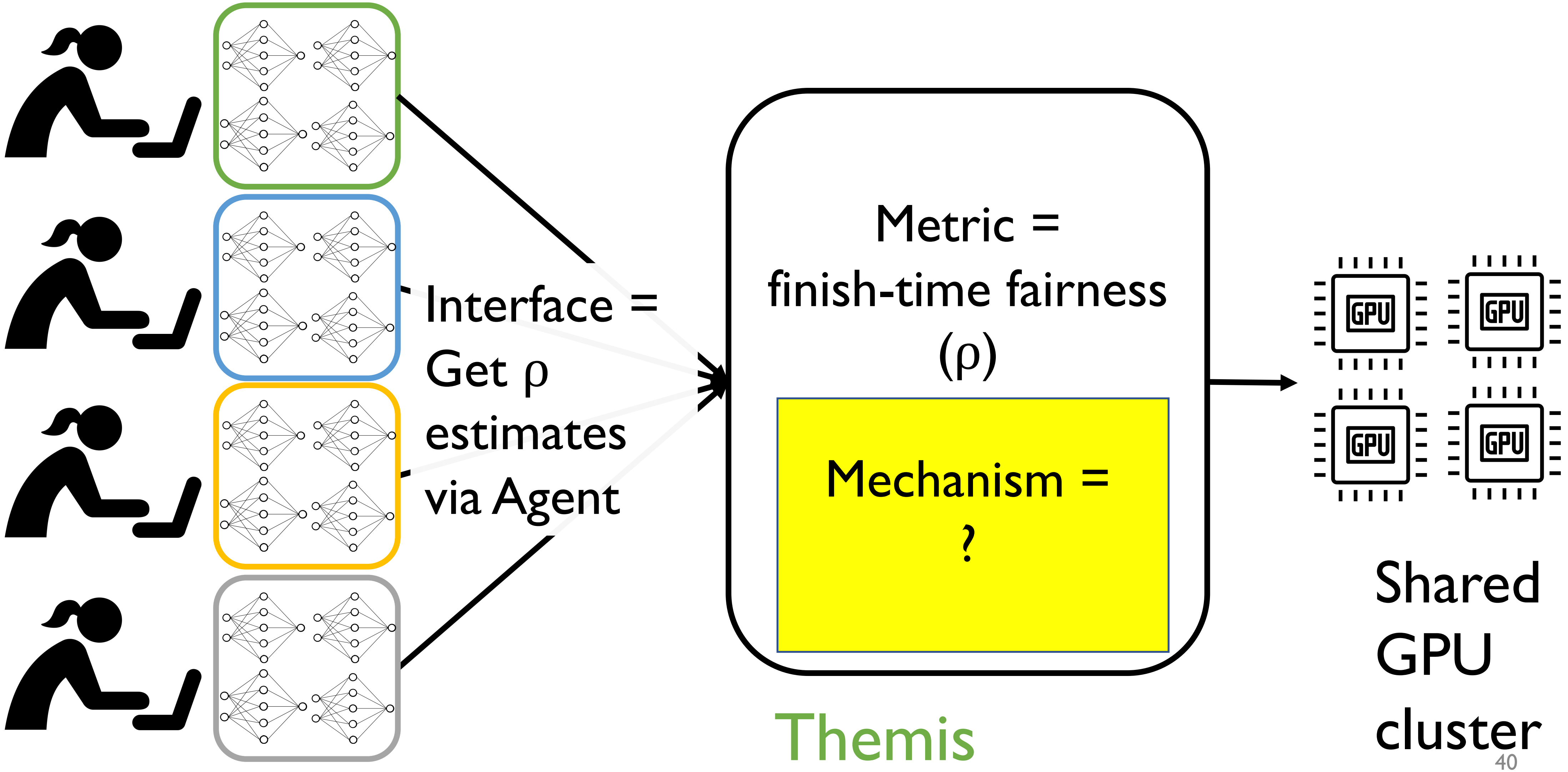
Themis: Interface

- App's Hyperparam-Opt tracks per-job progress
- *App does calculation of ρ*
- Scheduler pulls updated values of ρ from the *Agent* co-located with App's Hyperparam-Opt
- Details in the paper

Towards a new GPU Cluster Scheduler



Towards a new GPU Cluster Scheduler

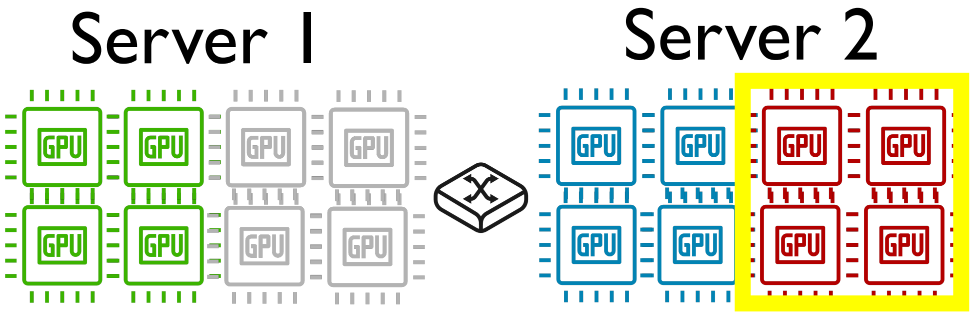


Themis: Mechanism

- Key Goal: Sharing Incentive
- SI: for all apps, ρ <= 1
- Difficult to guarantee with online arrivals
- Our focus: *min (max ρ)*:
  empirically keeps ρ's ≈ 1 without admission control

Strawman Mechanism

SI Objective – min (max ρ)

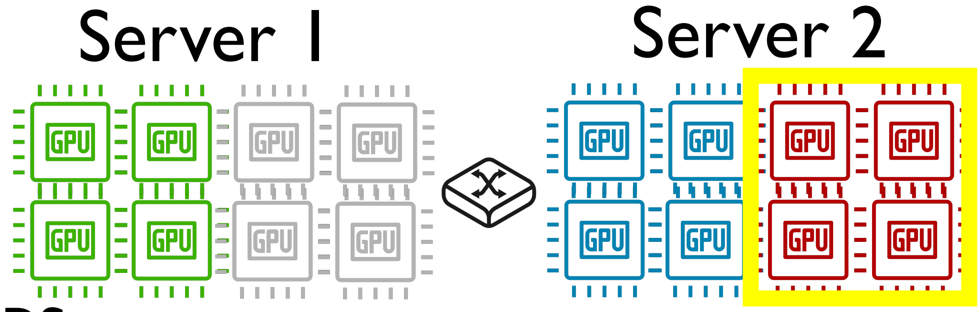


Red GPUs become available

Strawman Mechanism

SI Objective – min (max ρ)

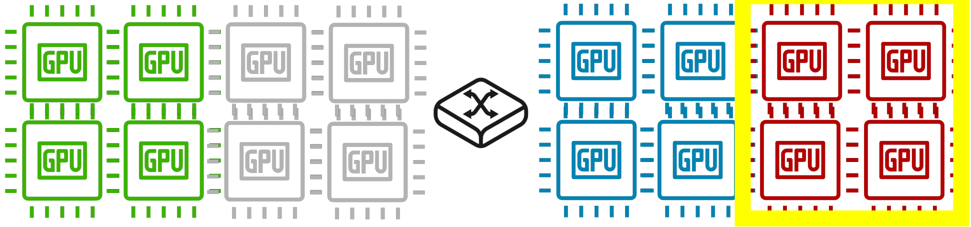
Interface: Get ρ estimates from all apps



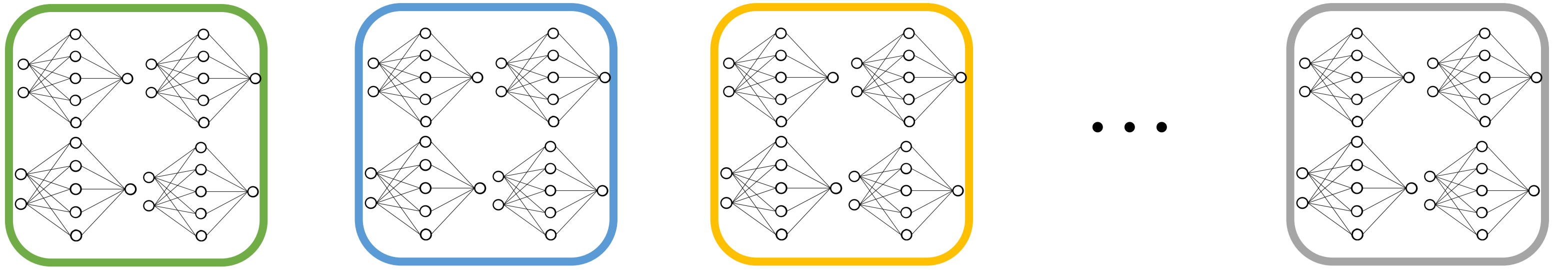
Red GPUs become available

Strawman Mechanism

SI Objective – min (max ρ)



Interface: Get ρ estimates from all apps

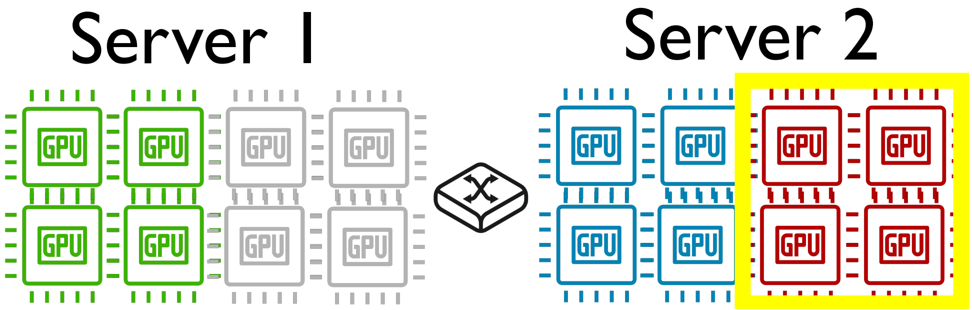


$\rho_1 > \rho_2 > \rho_3 > \rho_N$

Sort in decreasing order of ρ

Allocate to app with highest ρ (green app) for *lease* duration

Strawman Mechanism: Issues



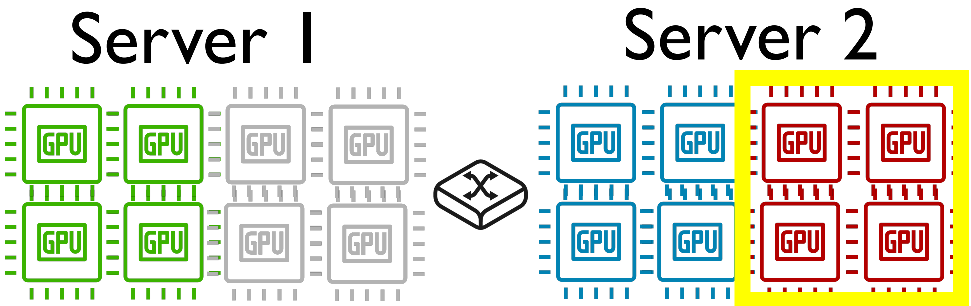
1. Inefficient Allocation – Red GPUs are not co-located with Green Apps GPUs

2. Lying Apps – Apps can lie with high ρ values to hoard GPU resources

Sort in decreasing order of ρ
Allocate to app with highest ρ (green app) for *lease* duration

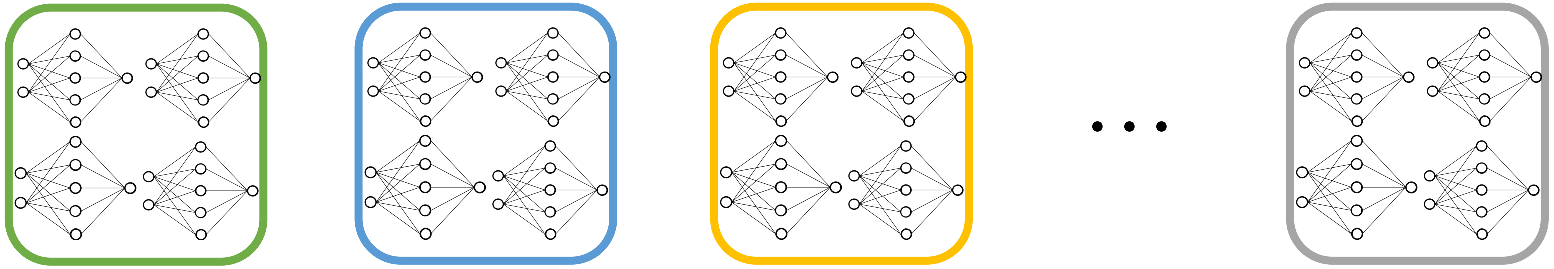
Themis: Mechanism

SI Objective – min (max ρ)



Red GPUs become available

Interface: Get ρ estimates from all apps



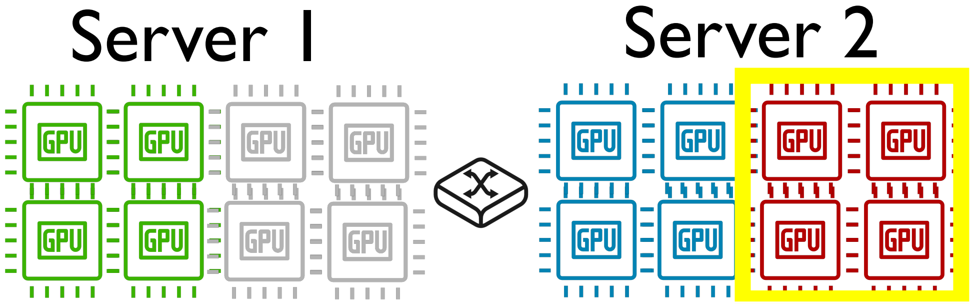
$$\underbrace{\rho_1 > \rho_2}_{1-f} > \underbrace{\rho_3 > \rho_N}_{f}$$

1. Filter *1 – f* apps with max ρ values
2. Allocate to one or more of *1 – f* apps for *lease* duration using Partial Allocation Auctions

Themis: Mechanism

SI Objective – min (max ρ)

Interface: Get ρ estimates from all apps

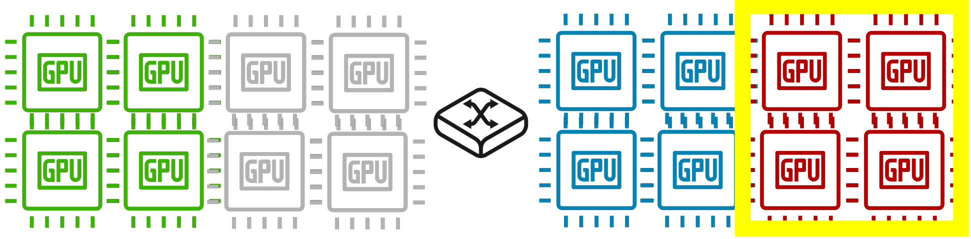


1. Tradeoff SI for Efficiency – f → 0 => More apps to allocate resources => Better opportunity to match placement preference of apps to resources. Our sensitivity analysis suggests f = 0.8 gives a good tradeoff.

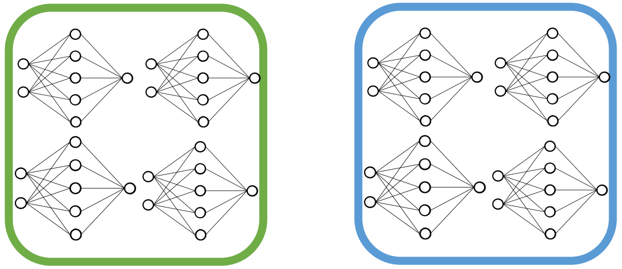
2. Partial Allocation Auction within 1 – f apps – Incentivizes truth telling of ρ

1. Filter *1 – f* apps with max ρ values
2. Allocate to one or more of *1 – f* apps for *lease* duration (Red GPUs can potentially go to Blue App) using Auctions

Themis: Mechanism: Partial Allocation Auction



Red GPUs become available

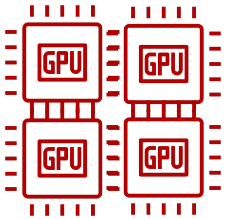


$\rho_1 > \rho_2$

$1 - f$

Offer

Bids



Resource Alloc	ρ_{new}
+ 0	ρ_{old}
+1 Red GPU	ρ_{old}-δ
+2 Red GPUs	ρ_{old}-2δ

Themis: Mechanism: Partial Allocation Auction

- Input: Valuation Tables from filtered apps
- Pareto efficiency (PE) – max $\prod_i 1/\rho_{i,\, new}$ – proportional fair allocations
- Strategy Proofness (SP) – Allocate a fraction of this per app for *lease* duration – rest is "hidden payment"
- More lying => higher hidden payments => incentivizes truth-telling
- Leftover Allocation – Allocate hidden payments to unfiltered apps at random to avoid unallocated resources and enable work conservation

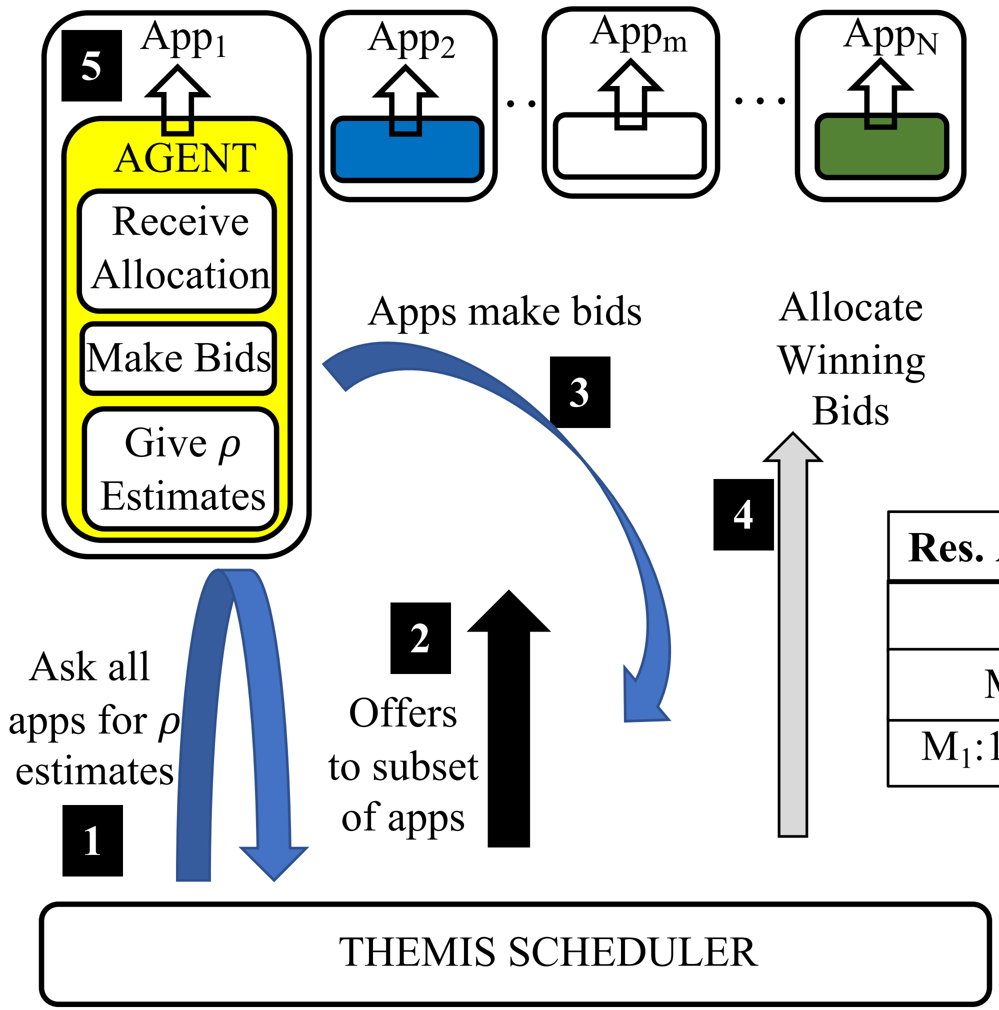
Themis: Overall Design

Hyperparam-opt at the top

Agent:
Shim layer added to Apps to enable interaction (estimate ρ, make bids) with Scheduler

Scheduler:
Finish-Time Fair Metric (ρ);

Mechanism:
SI + PE + SP



Res. Alloc.	ρ_{new}
0	ρ_{old}
M_1:2G	$\rho_{old} - 2\delta$
M_1:1G,M_2:1G	$\rho_{old} - \delta$

Themis: Implementation

Hyperparam-opt at the top

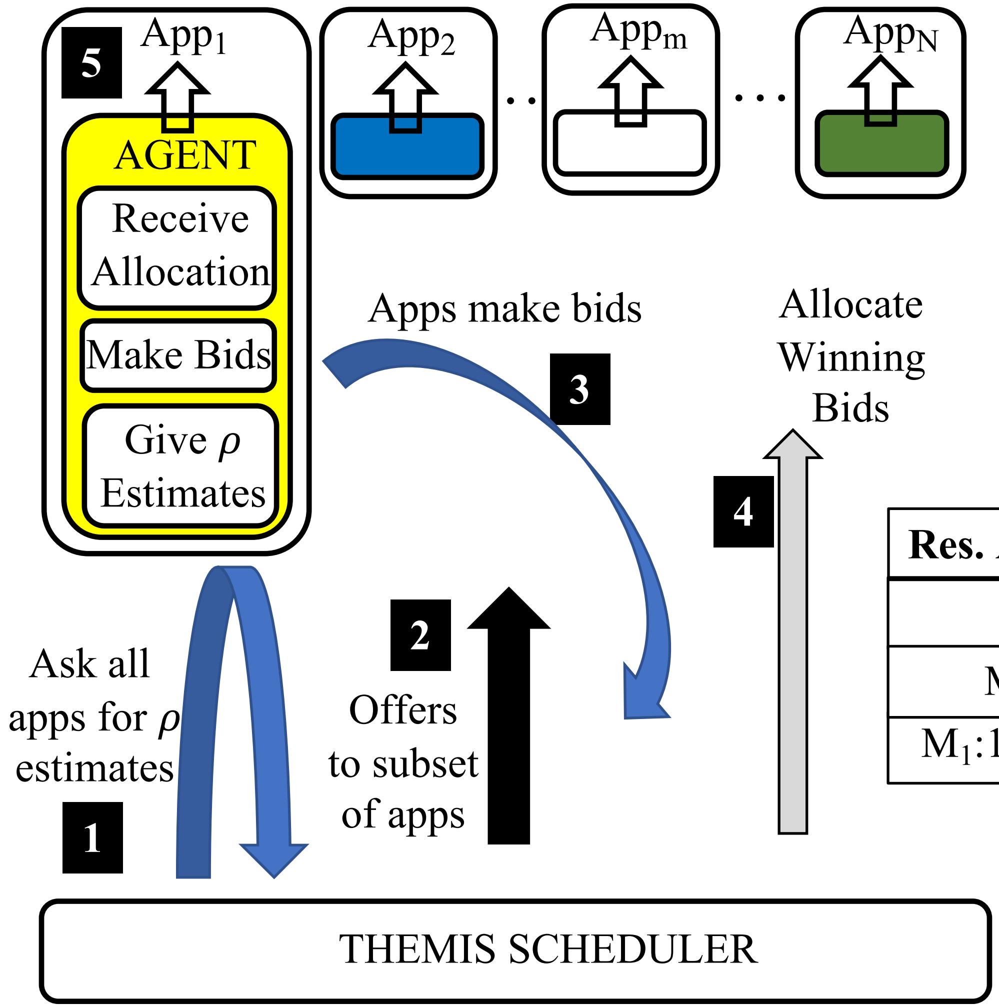
Agent:
Shim layer added to Apps to enable interaction with Scheduler

Submarine AM

Scheduler:
Finish-Time Fair Metric;

Mechanism:
SI + PE + SP

Hadoop 3.2.0 YARN RM



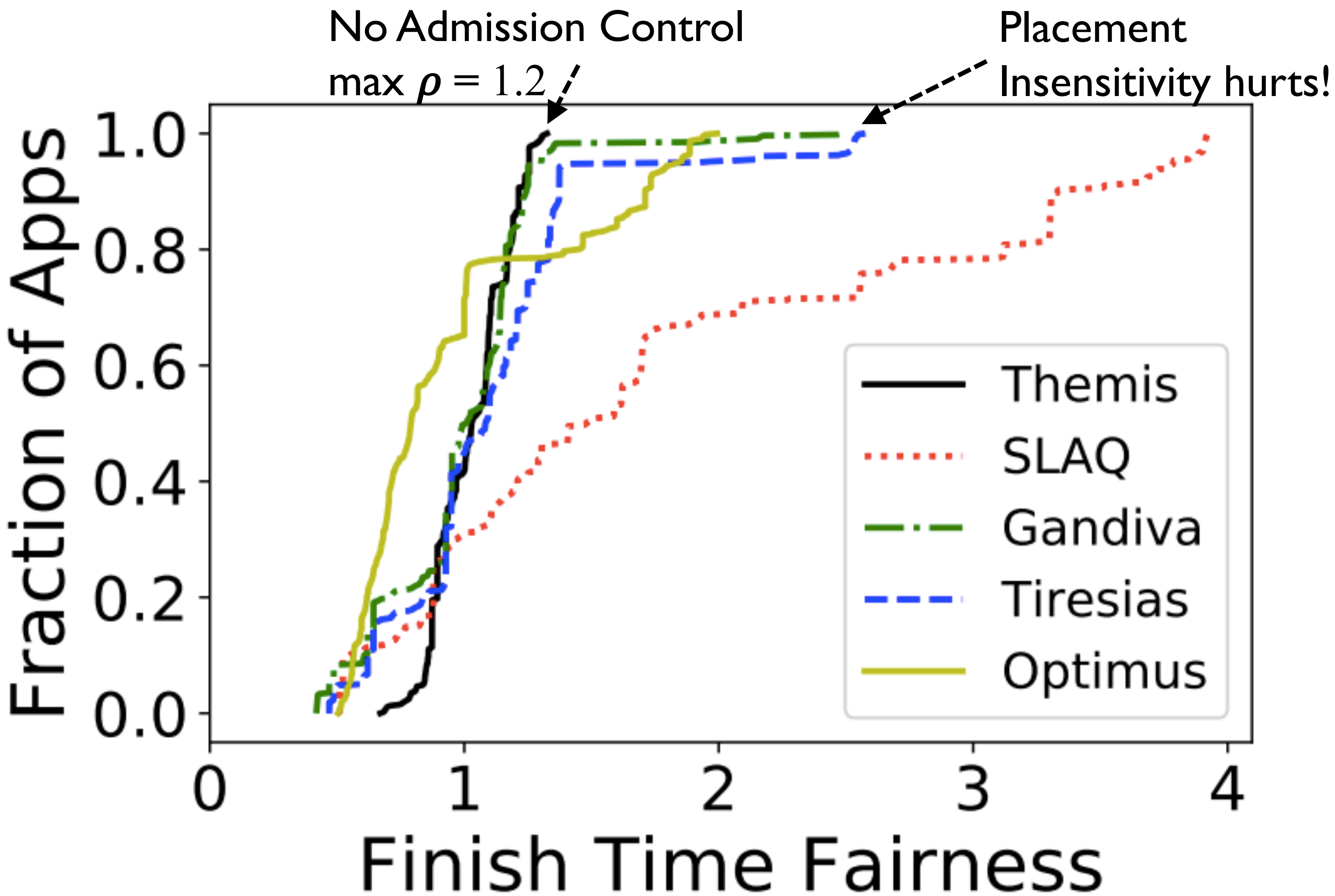
Res. Alloc.	ρ_{new}
0	ρ_{old}
M_1:2G	$\rho_{old} - 2\delta$
M_1:1G,M_2:1G	$\rho_{old} - \delta$

Themis: Evaluation

- 20 machine, 64 GPU cluster
  - 8 instances each with 2 Tesla K80 GPUs and
  - 12 instances each with 4 Tesla K80 GPUs
- A publicly available trace of DL apps from Microsoft
- Baselines:
  - **Tiresias** – Least Attained Service Job First
  - **Optimus** – Best Throughput Scaling First
  - **Gandiva** – Best Packing Job First
  - **SLAQ** – Best Loss Gradient Job First

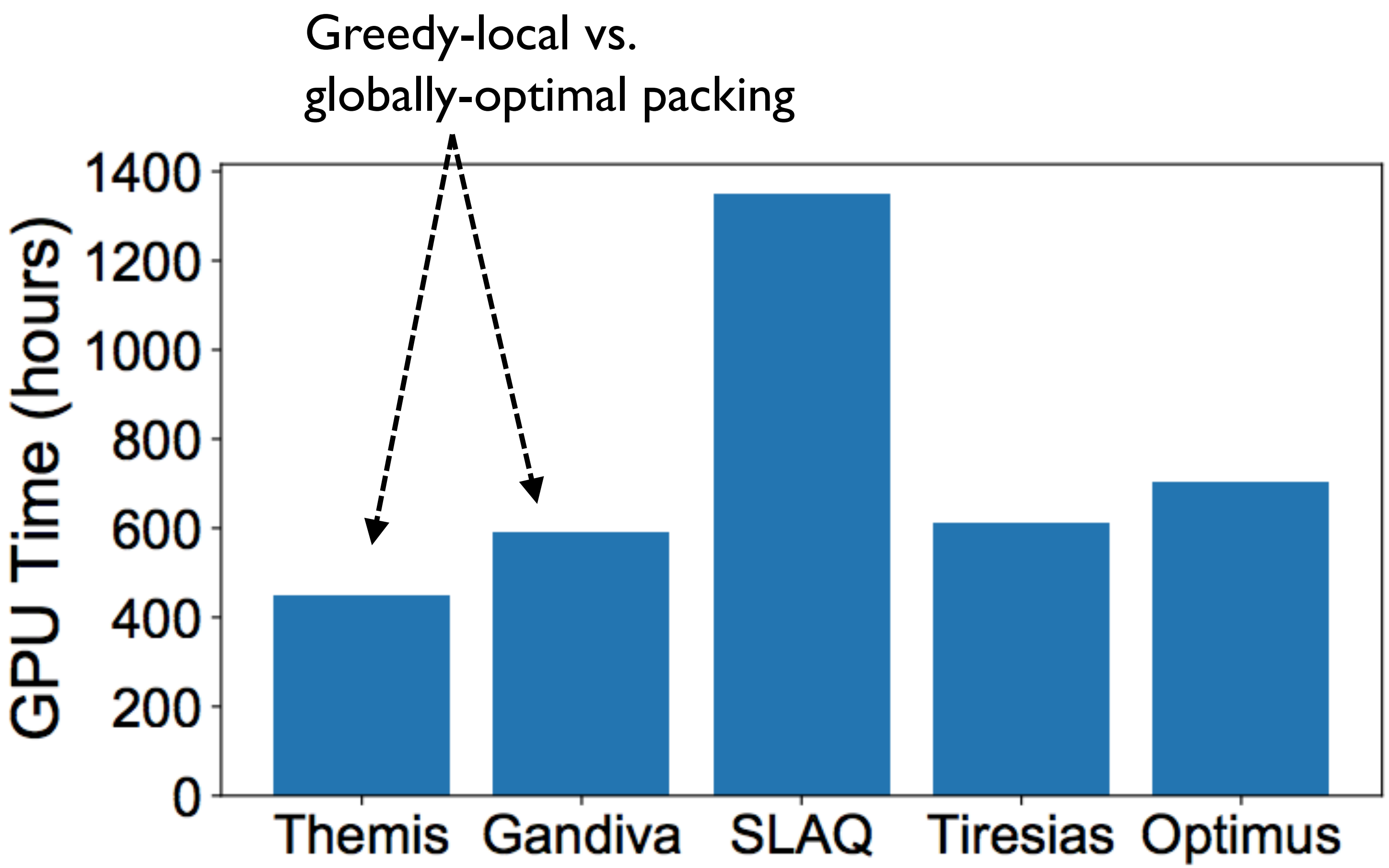
Macrobenchmark: Sharing Incentive

- CDF of ρ for all apps in the workload
- max ρ = 1.2 (~1) with Themis
- ρ distribution has long tail without Themis



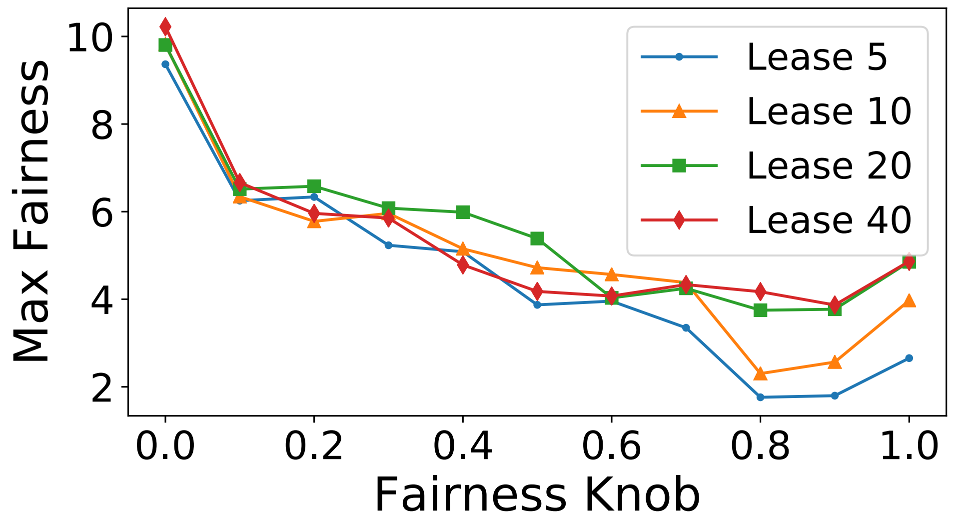
Macrobenchmark: Efficiency

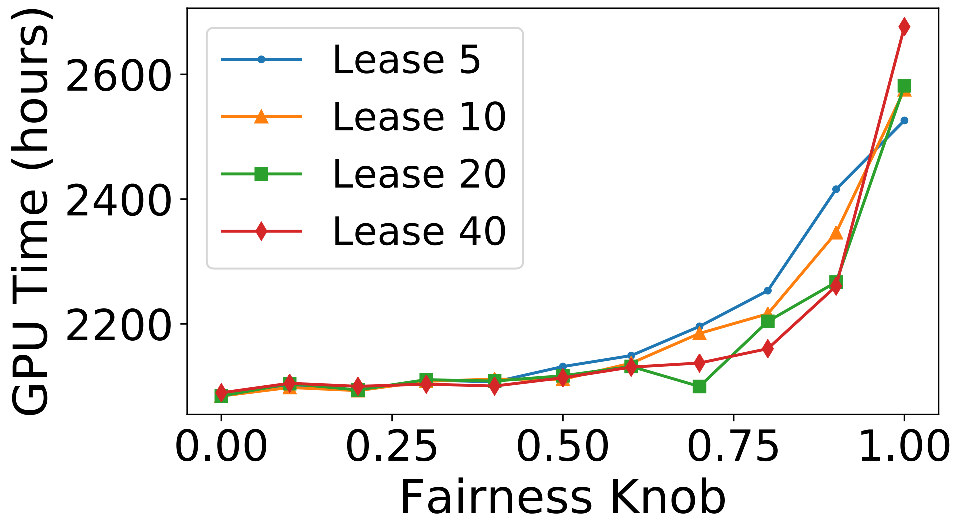
- GPU Time to execute workload
- Themis better than Gandiva
- Auctions enable globally optimal packing



Sensitivity Analysis/Tradeoffs

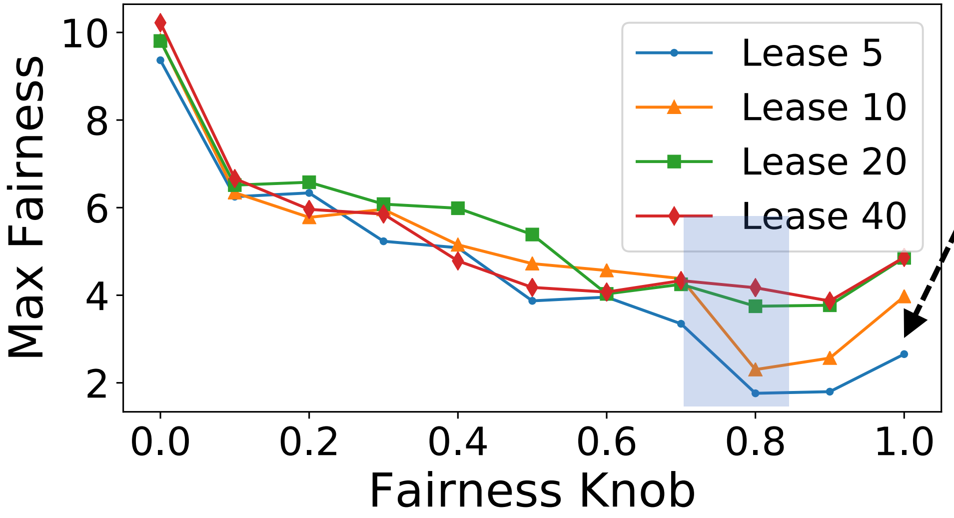
- max finish-time fair metric (ρ) and GPU time for different values of fairness knob (*f*)





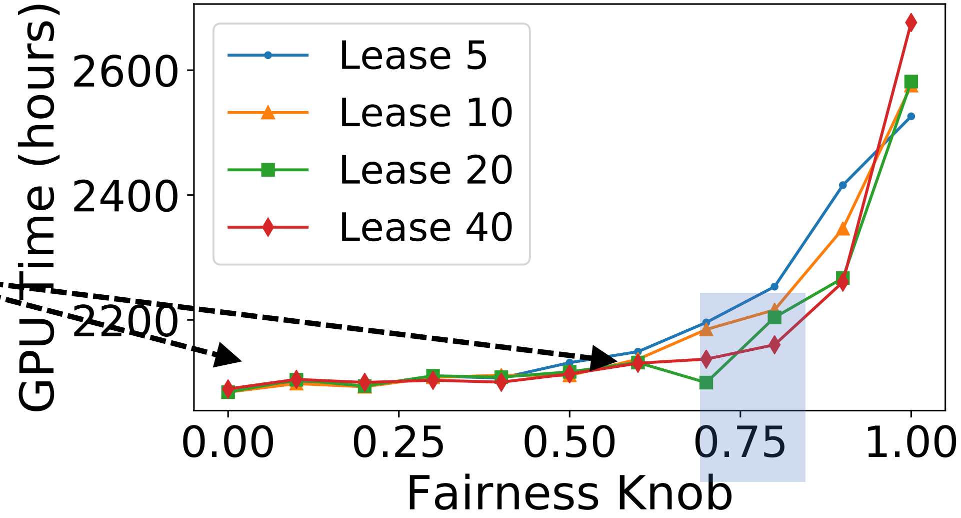
Sensitivity Analysis/Tradeoffs

- max finish-time fair metric (ρ) and GPU time for different values of fairness knob (*f*)
- *f = 0.8* maximizes sharing incentive without degrading efficiency



Sub-optimal placement

More choice; pack better



Conclusion

- Consolidation of GPUs => Sharing Incentive is key
- DL App properties => existing schedulers violate SI
- Themis proposes a new metric finish-time fairness that captures SI
- Filtering + Partial Allocation Auctions => Themis performs better than existing schedulers on SI and Efficiency