Tiresias: A GPU Cluster Manager for Distributed Deep Learning

NSDI'19

Guanbin Xu

2019-03-27, Reading Group

.

Tiresias

A **GPU Cluster Manager** for **Distributed Deep Learning**

NSDI'19

.

Outline

1. Deep Learning

2. GPU Cluster Manager & Weakness of them

3. Tiresias:

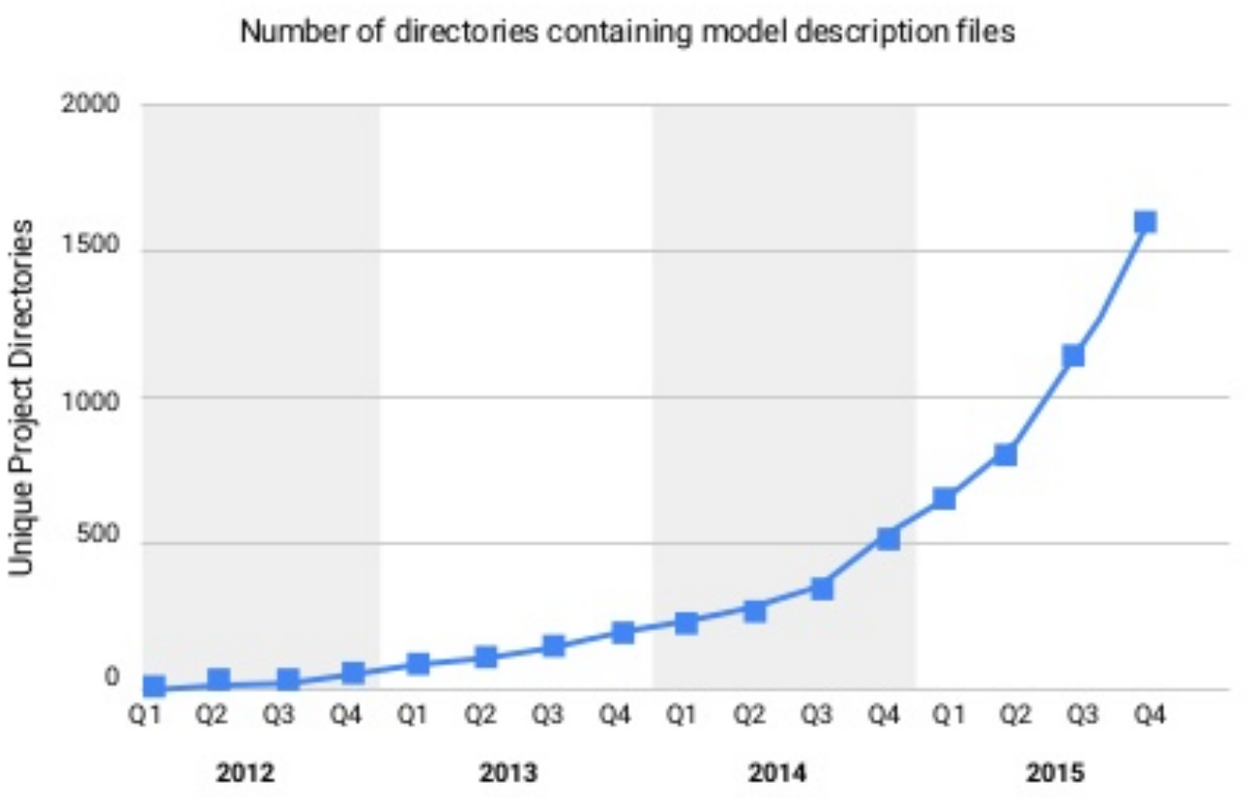
    1.Scheduler

    2. Placement

4. Tiresias: Evaluation

Deep Learning

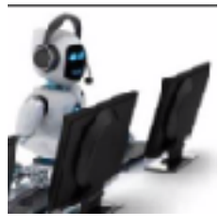
Growing Use of Deep Learning at Google



Across many products/areas

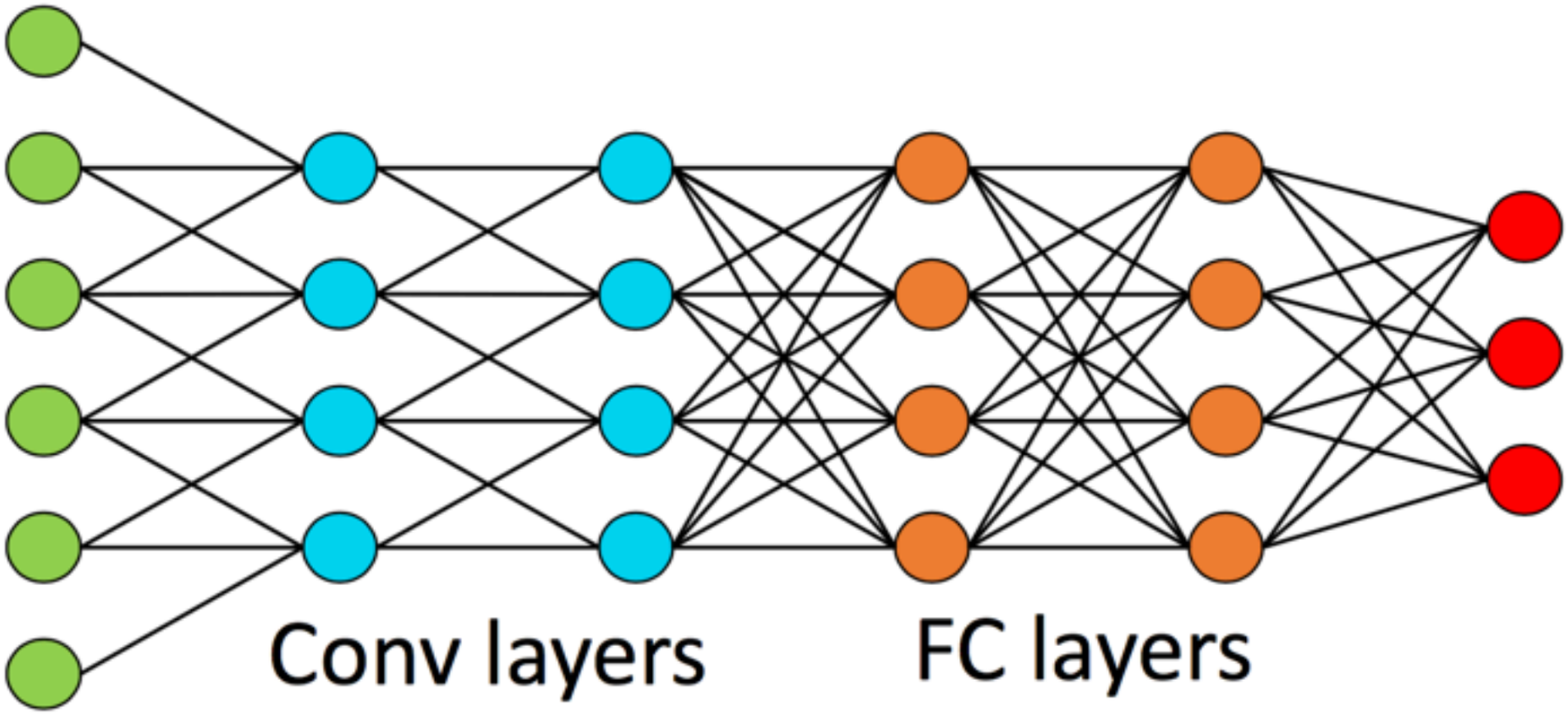
- Apps
- Maps
- Photos
- Gmail
- Speech
- Android
- YouTube
- Translation
- Robotics Research
- Image Understanding
- Natural Language Understanding
- Drug Discovery

Google

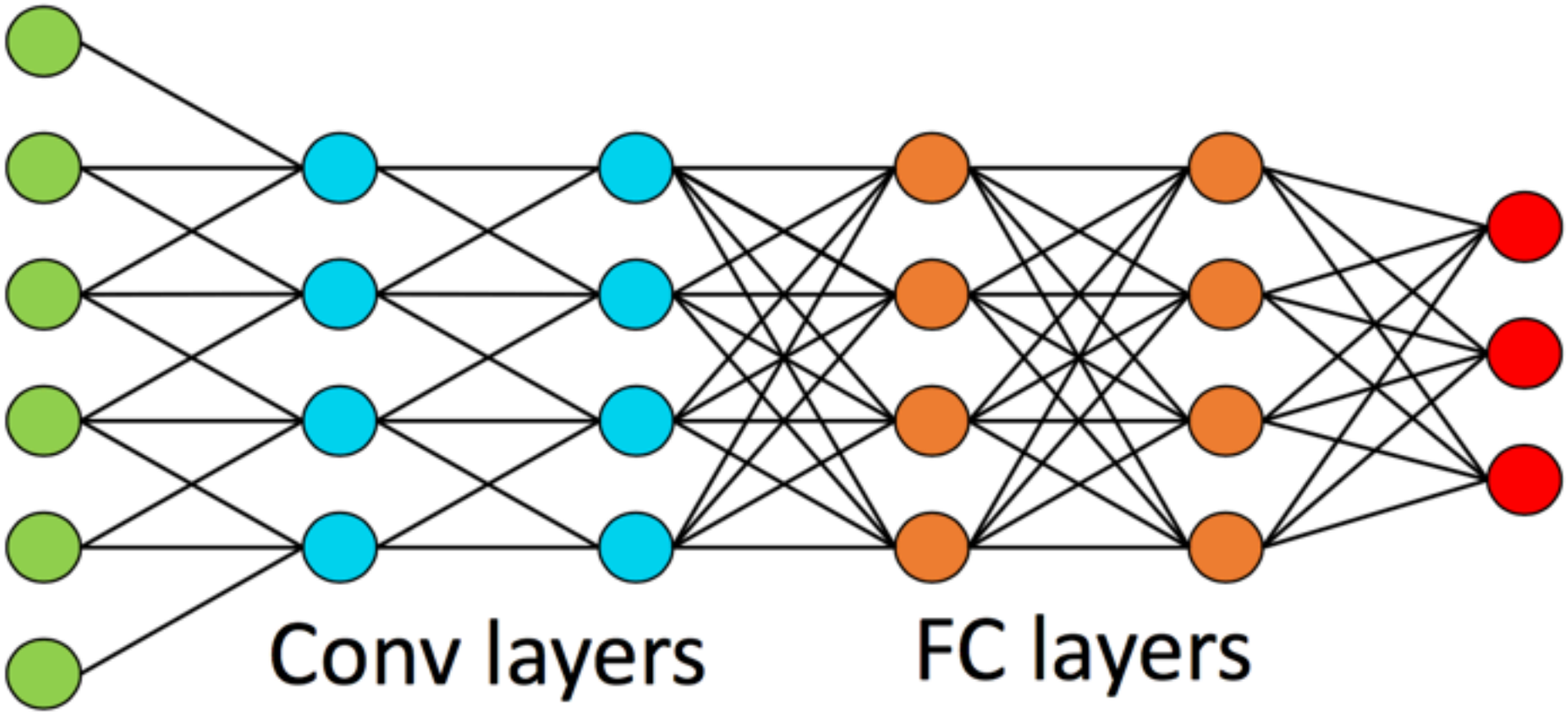


Large-Scale Deep Learning for Intelligent Computer Systems, Google Research

Deep Learning

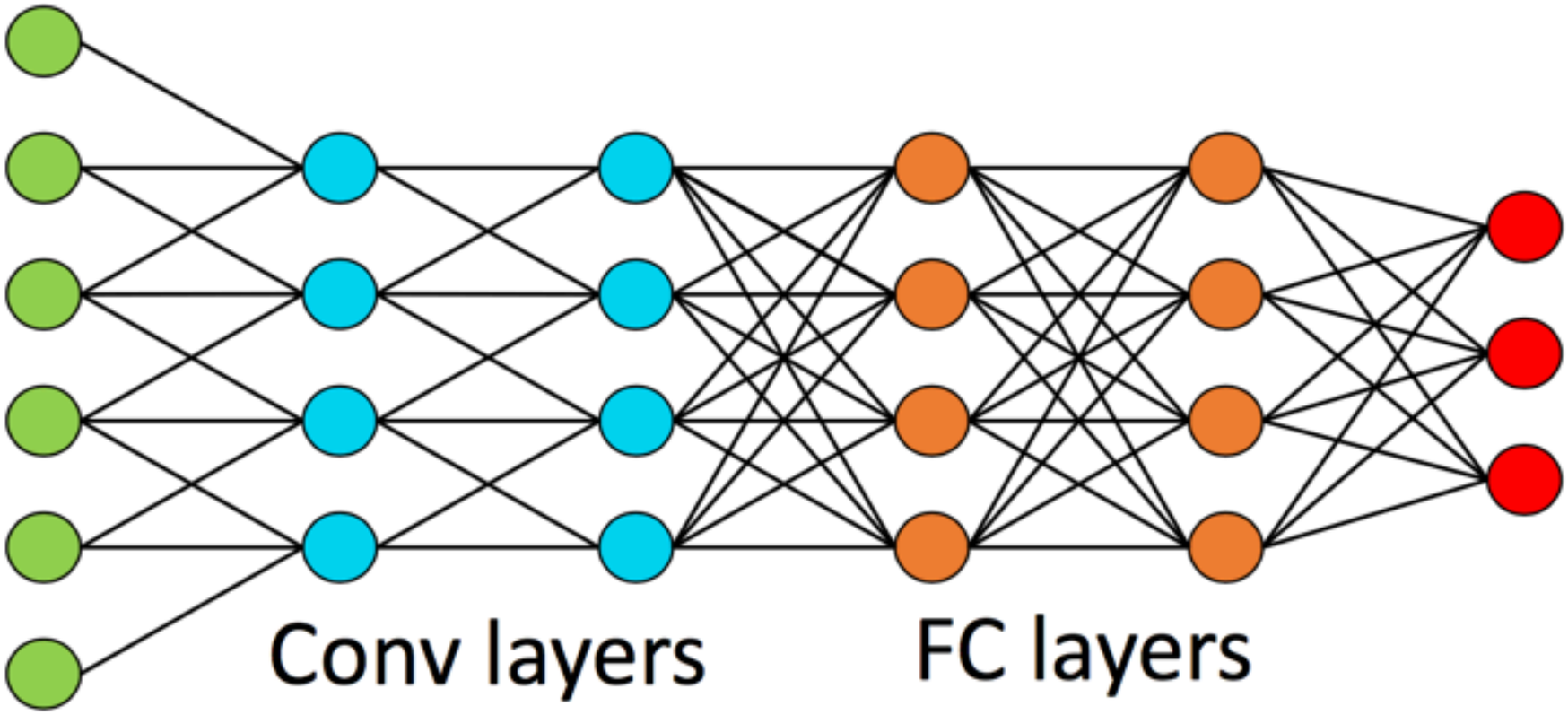


Deep Learning

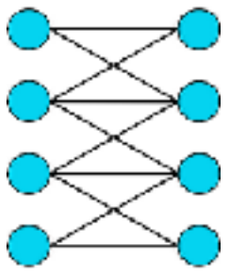


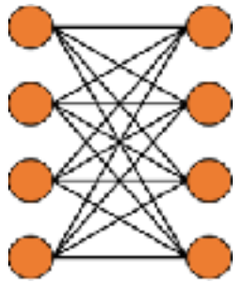
f（w，x）

Deep Learning

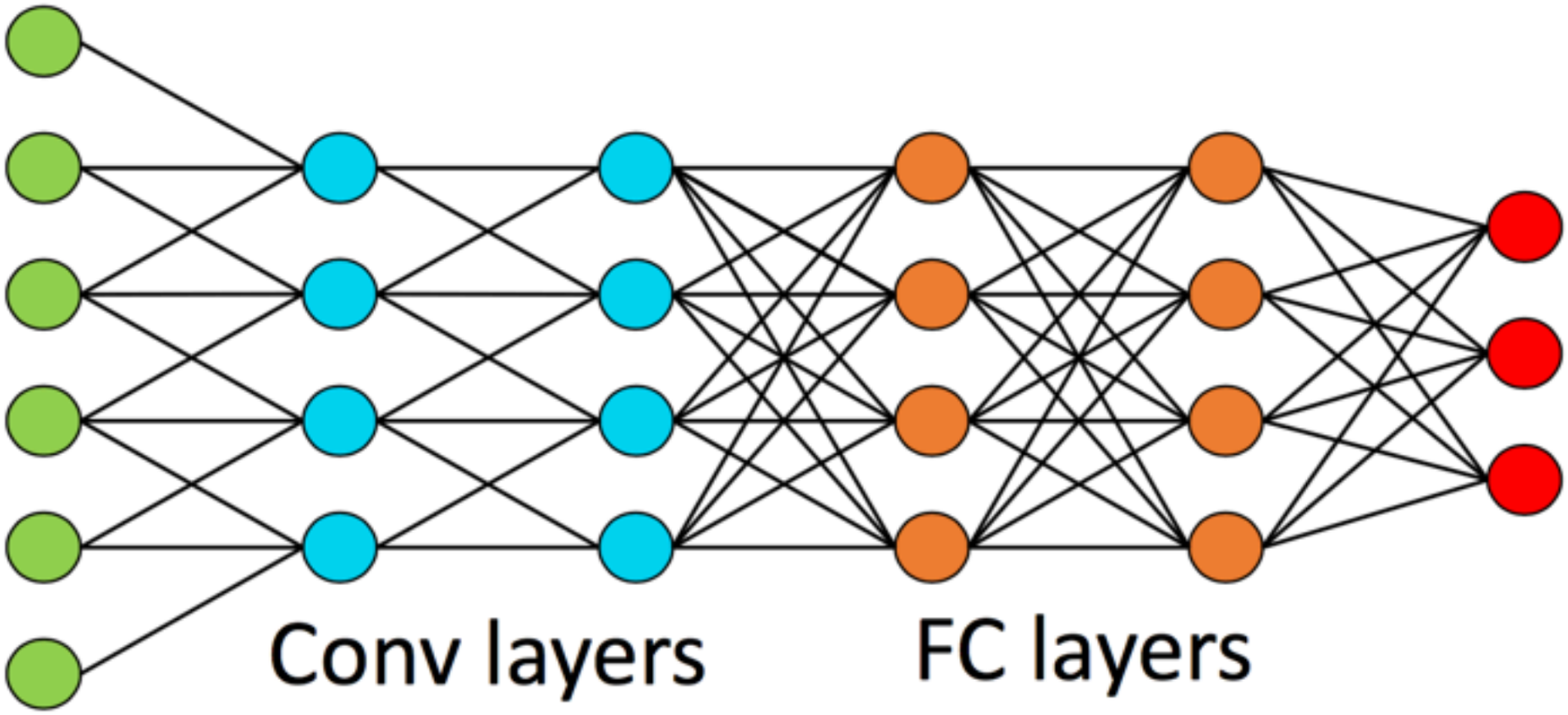


f（w，x）

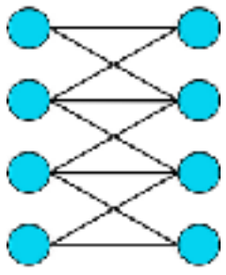


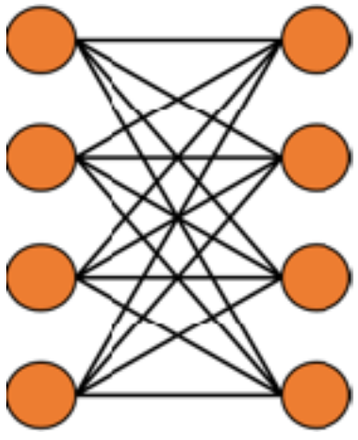


Deep Learning

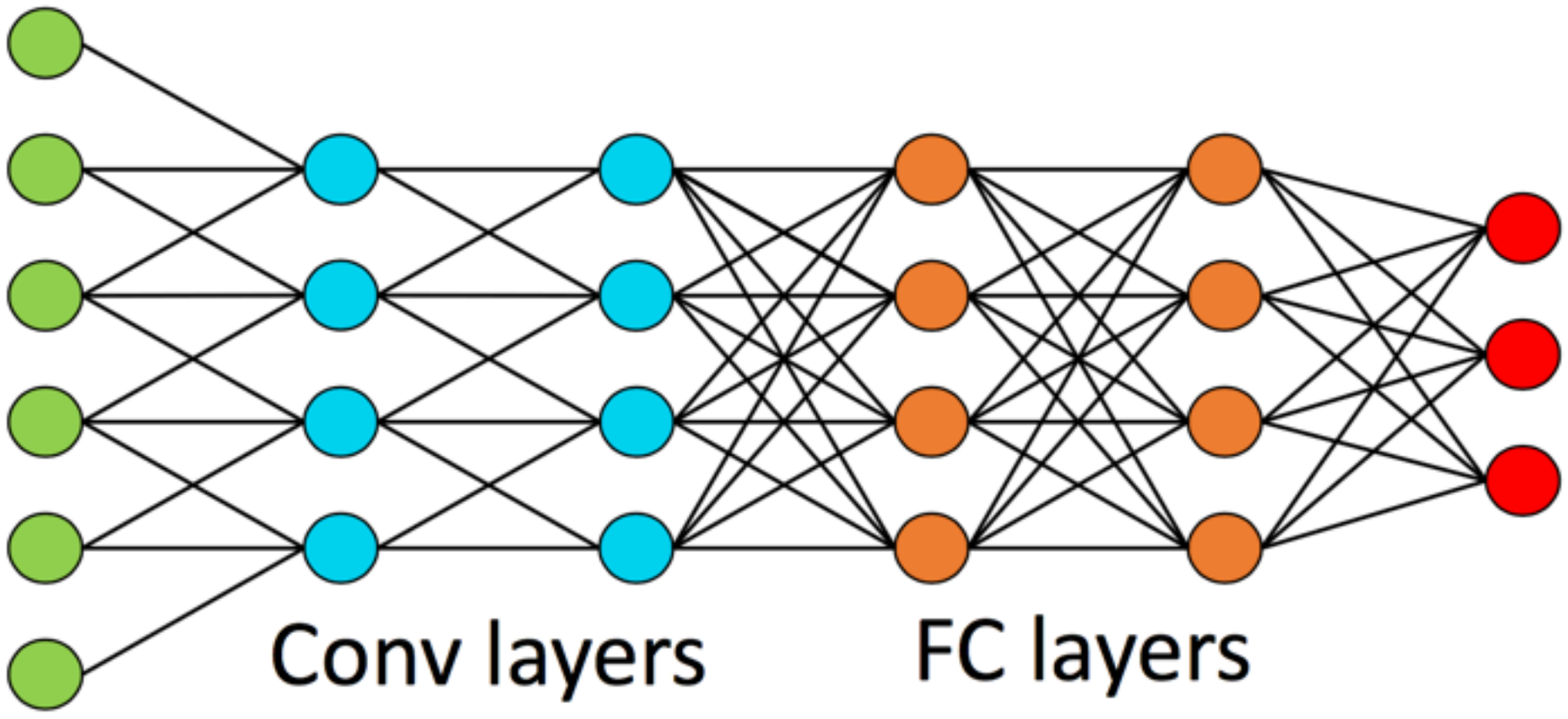


f（w，x）





Deep Learning



Deep Learning

SIMD

Deep Learning

Outline

1. Deep Learning

2. GPU Cluster Manager & Weakness of them

3. Tiresias:

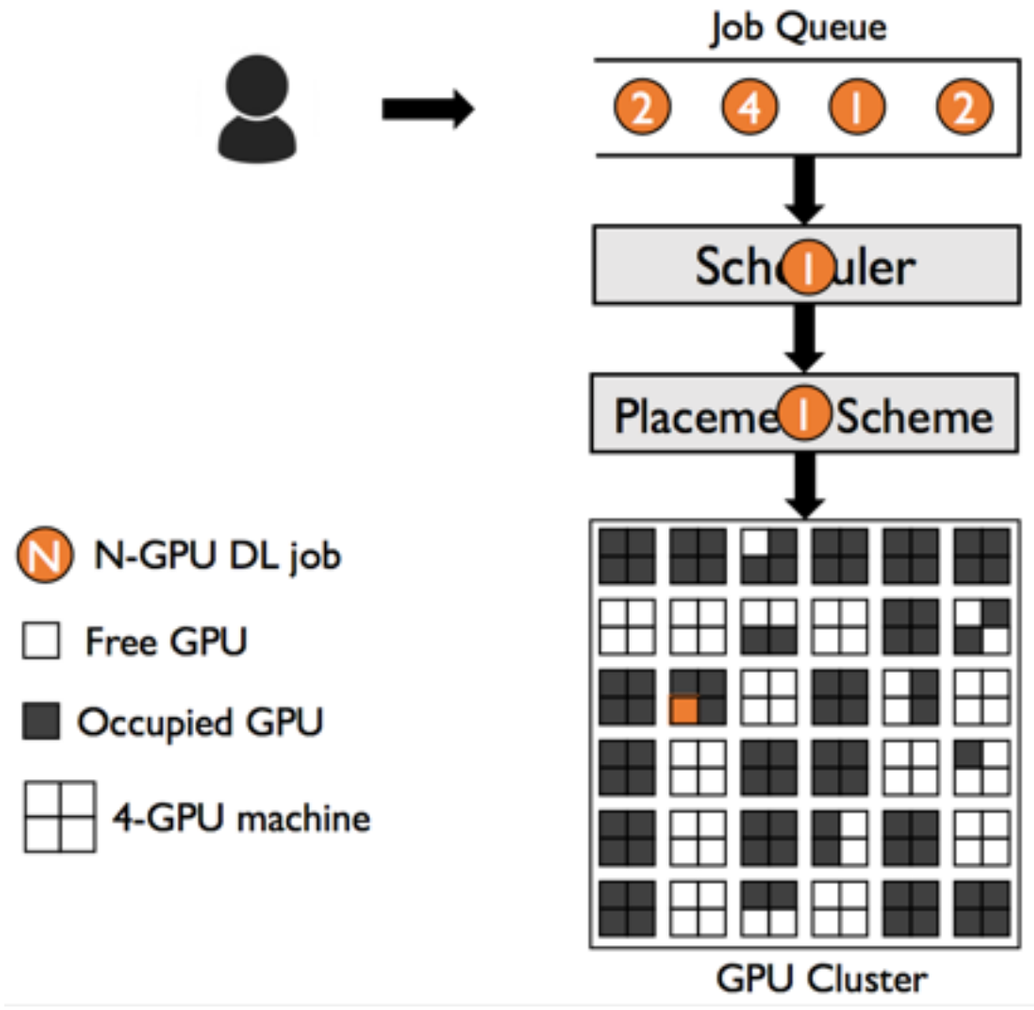
    1.Scheduler

    2. Placement

4. Tiresias: Evaluation

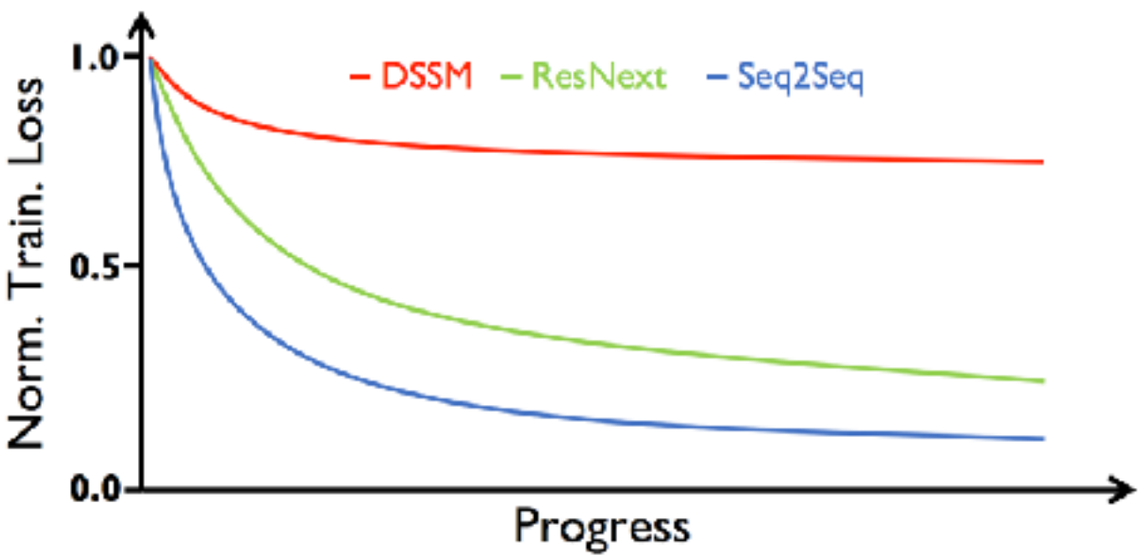
GPU Cluster Manager

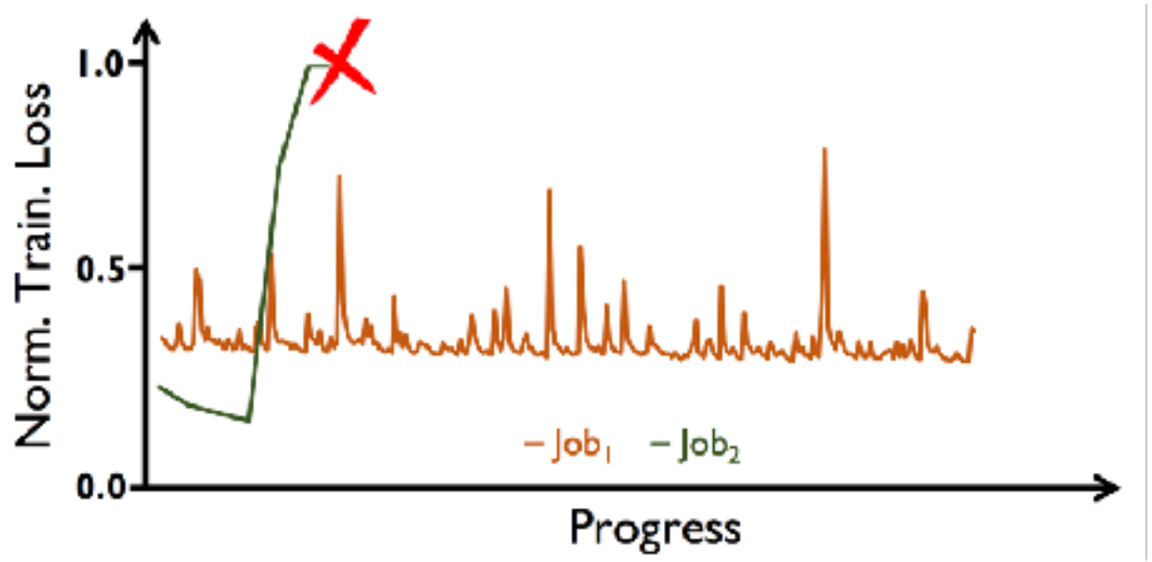
- Design Objectives
  - Minimize Cluster-Wide Average Job Completion Time (JCT)
- Achieve
  - High Resource (GPU) Utilization



Challenge 1: Unpredictable Training Time

- **Unknown execution time of DL training jobs**
  - **Job execution time is useful when minimizing JCT**
- **Predict job execution time**
  - **Use the smooth loss curve of DL training jobs (Optimus [1])**

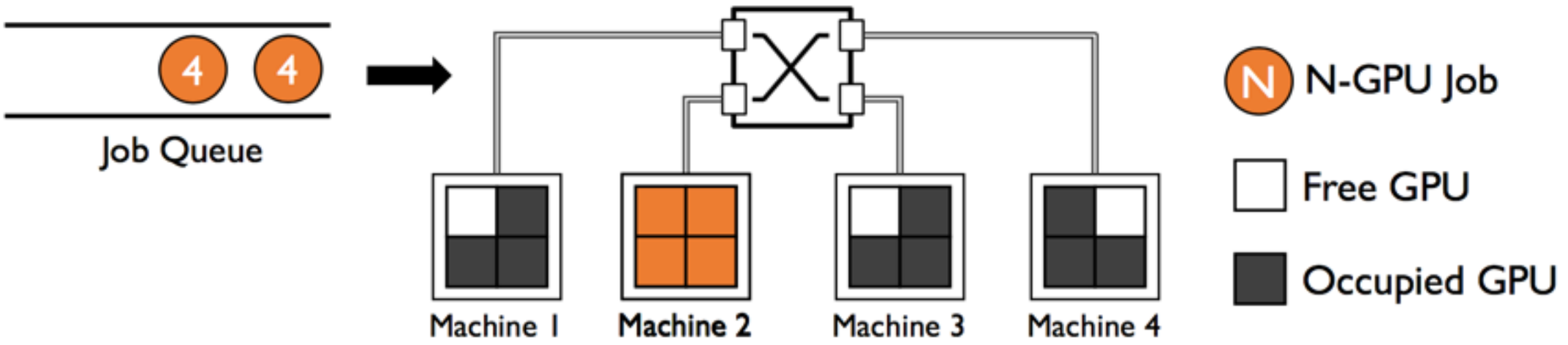




[1]. Optimus: An Efficient Dynamic Resource Scheduler for Deep Learning Clusters, EuroSys'18

Challenge 2: Over-Aggressive Job Consolidation

- Network overhead in DDL training
  - Consolidated placement for good training performance
    - Fragmented free GPUs in the cluster
    - Longer queuing delay



Prior Solutions

	I. *Unpredictable Training Time* (*Scheduling*)	II. *Over-Aggressive Job Consolidation* (*Job Placement*)
Optimus[1]	None	None
YARN-CS	*FIFO*	None
Gandiva[2]	*Time-sharing*	*Trial-and-error*

[1]. Optimus: An Efficient Dynamic Resource Scheduler for Deep Learning Clusters, EuroSys'18
[2]. Gandiva: Introspective Cluster Scheduling for Deep Learning, OSDI'18

Outline

1. Deep Learning

2. GPU Cluster Manager & Weakness of them

3. Tiresias:

    1.Scheduler

    2. Placement

4. Tiresias: Evaluation

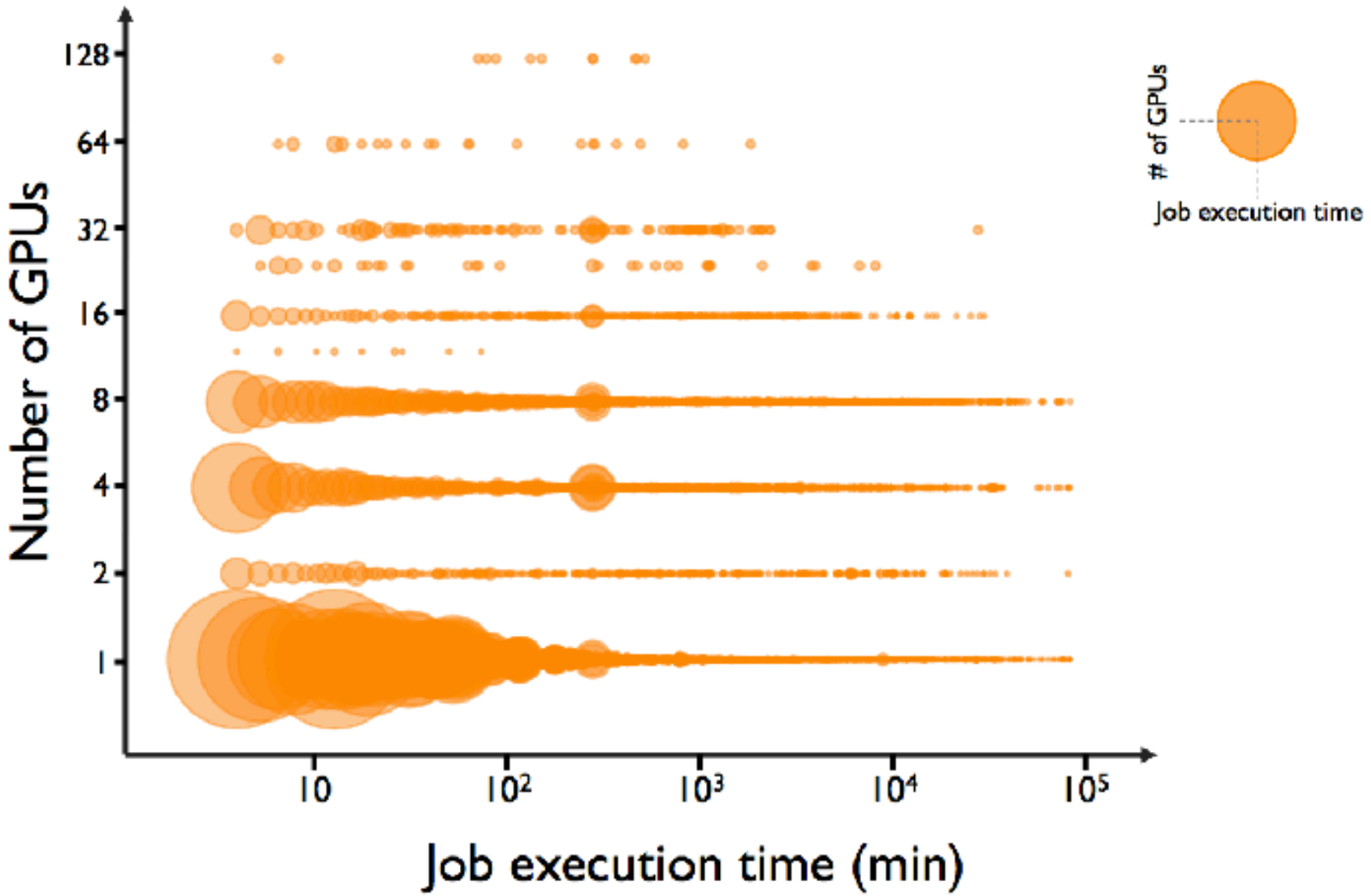
Tiresias:

A GPU cluster manager for Distributed Deep Learning Without Complete Knowledge

- Age-Based Scheduler → Minimize JCT without complete knowledge of jobs
- Model Profile-Based Placement → Place jobs without additional information from users

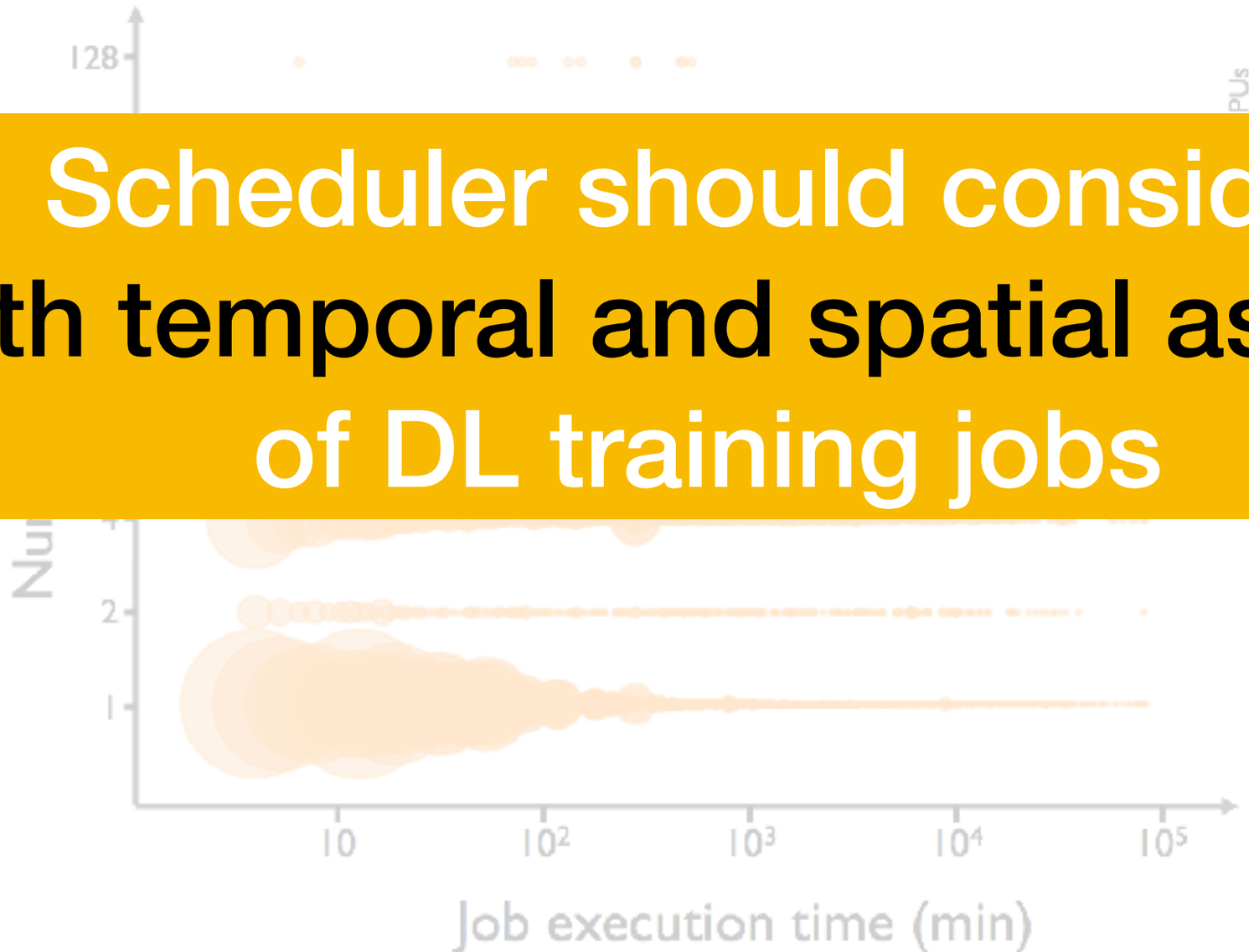
Observation 1

- Variations in both temporal and spatial aspects



Observation 1

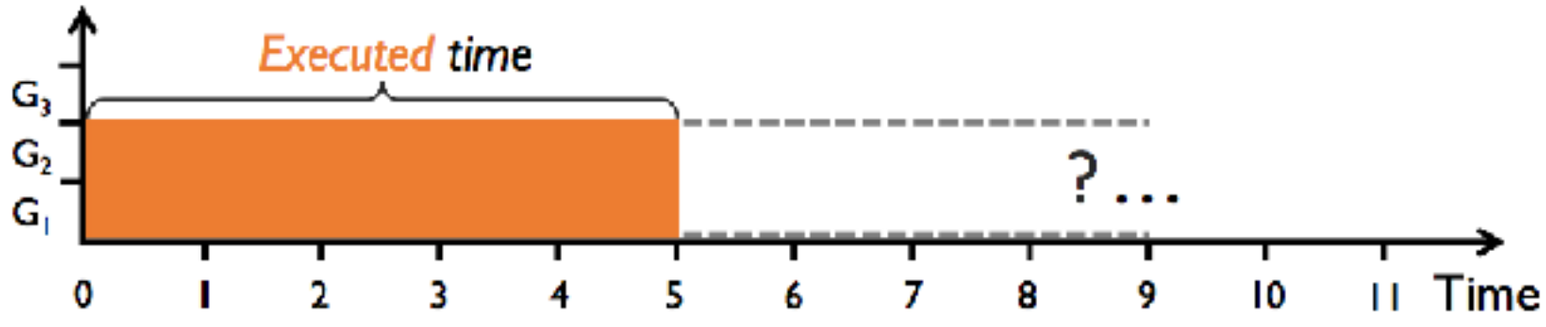
- Variations in both temporal and spatial aspects



Scheduler should consider both temporal and spatial aspects of DL training jobs

Observation 1

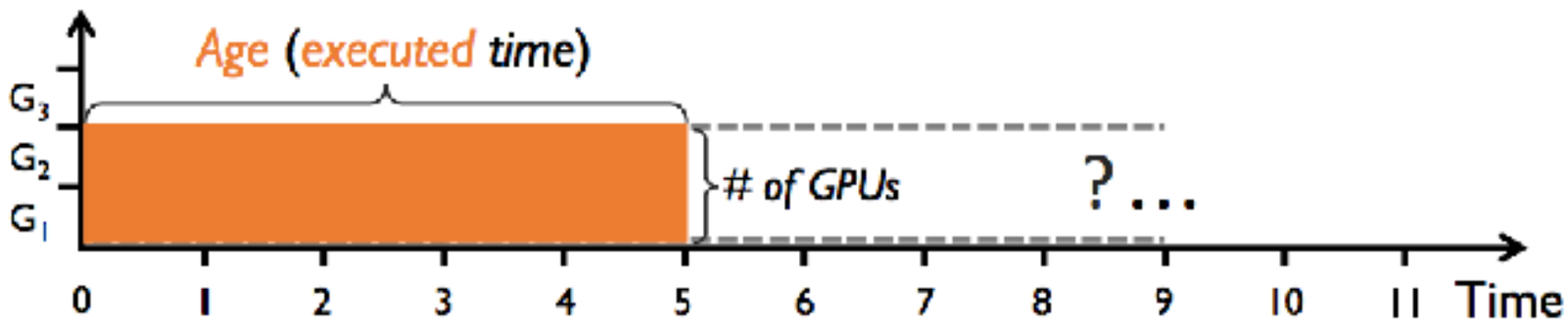
- Spatial: number of GPUs
- Temporal: executed time
- distribution of job execution time**(maybe)**



Tiresias: Scheduler

~~**Short Remaining-Time First(SRTF)**~~

- Least-Attained Service[1] (LAS) ~~**Short Job First(SJF)**~~
  - Prioritize job that has the shortest executed time
- Gittins Index policy[2]
  - Need the distribution of job execution time
  - Prioritize job that has the highest probability to complete in the near future



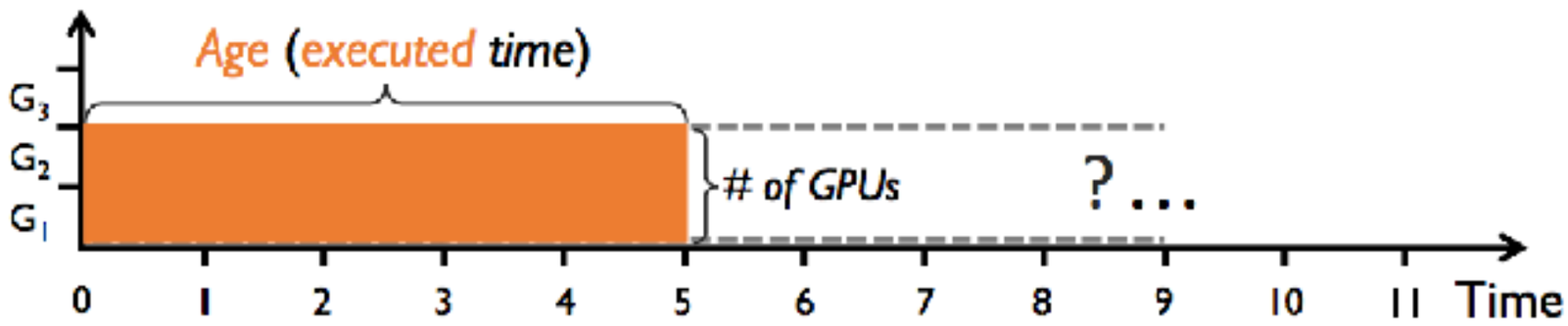
[1]. Feedback queueing models for time-shared systems. JACM, 1968
[2]. Multi-armed bandit allocation indices. Wiley, Chichester, 1989

Tiresias: Scheduler

- Least-Attained Service[1] (LAS)
  - Prioritize job that has the shortest executed time

$$\frac{1}{age}$$

- Gittins Index policy[2]
  - Need the distribution of job execution time
  - Prioritize job that has the highest probability to complete in the near future



[1]. Feedback queueing models for time-shared systems. JACM, 1968
[2]. Multi-armed bandit allocation indices. Wiley, Chichester, 1989

Gittins Index

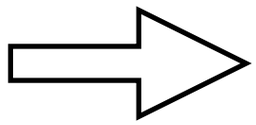
$$GI_J = \sup_{\Delta>0} \frac{P(S - a_J \le \Delta \mid S > a_J)}{E[\min\{S - a_J, \Delta\} \mid S > a_J]}$$

- ***P*** is the probability that ***J*** can complete with in ***Δ***
- ***E*** is the expected service (cost) of ***J*** to be complete with in ***Δ***
- ***Δ*** is the next service quantum
- ***P*** and ***E*** are calculated from the distribution of job GPU time

Gittins Index

- Higher probability to complete (Gittins Index), higher priority

	# of GPUs	Execution time
J_1	2	2
J_2	1	8
J_3	2	6

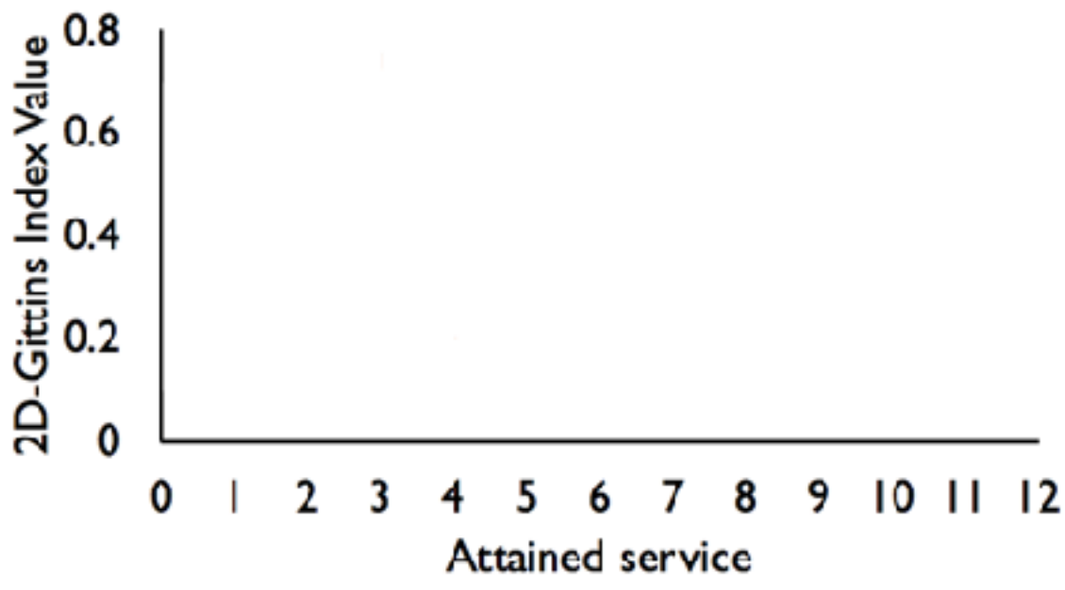


	# of GPUs	Distribution
J_1	2	2
J_2	1	(4, 8, 12)
J_3	2	6

Gittins Index

- Higher probability to complete (Gittins Index), higher priority

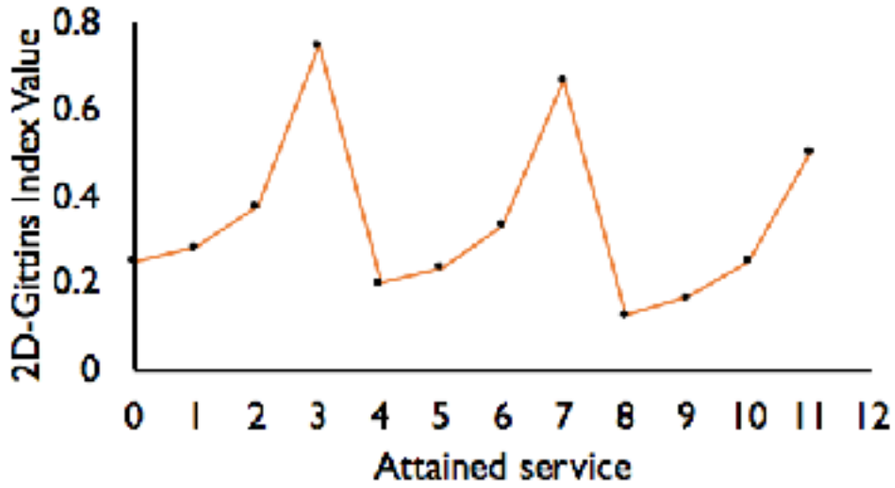
	# of GPUs	Distribution	Attained Service
J_1	2	(4, 8, 12)	0
J_2	1		0
J_3	2		0



Gittins Index

- Higher probability to complete (Gittins Index), higher priority

	# of GPUs	Distribution	Attained Service	Gittins Index
J_1	2	(4, 8, 12)	0	0.25
J_2	1		0	0.25
J_3	2		0	0.25



$$GI_J = \sup_{\Delta>0} \frac{P(S - a_J \le \Delta \mid S > a_J)}{E[\min\{S - a_J, \Delta\} \mid S > a_J]}$$

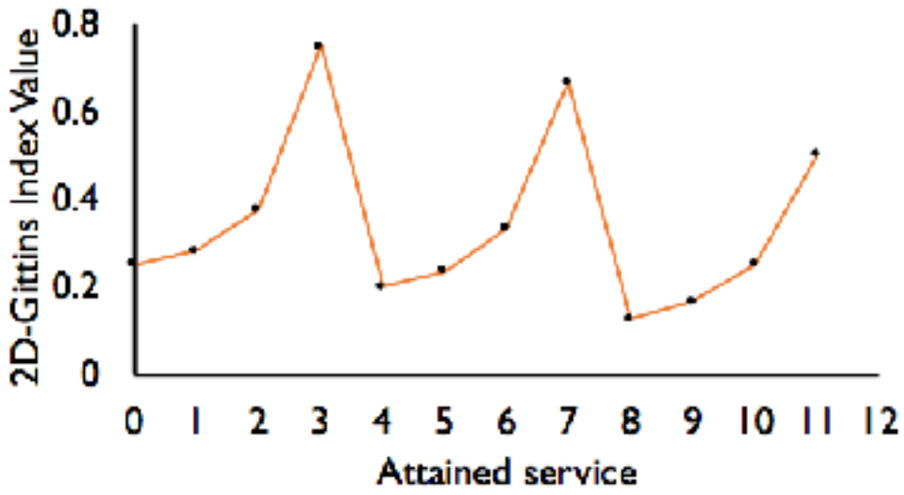
- ***Δ=4***
- ***distribution=(4,8,12)***

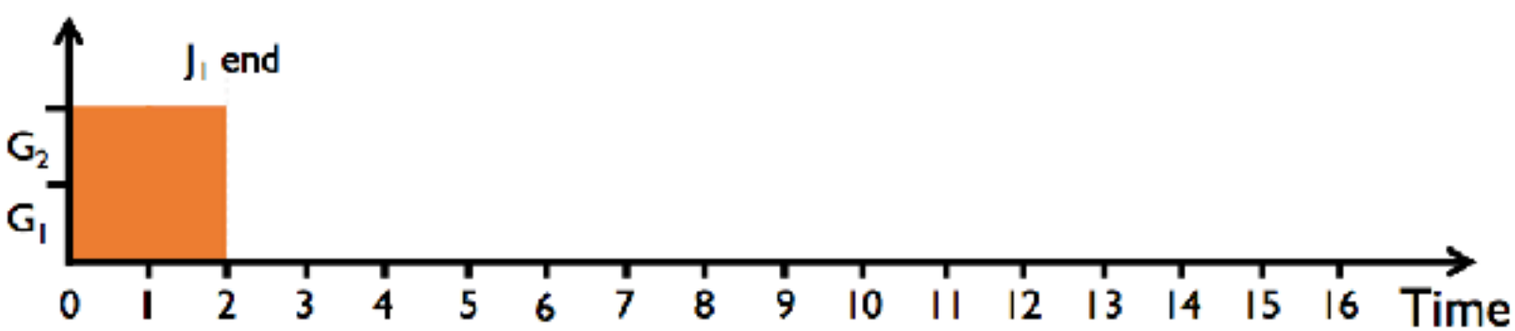
$$G_{j_1} = \frac{P_{s=4}}{min(4-0,\delta) * 1/3 + min(8-0,\delta) * 1/3 + min(12-0,\delta) * 1/3} = \frac{1/3}{1/3 * 4} = 0.25$$

Gittins Index

- Higher probability to complete (Gittins Index), higher priority

	# of GPUs	Distribution	Attained Service	Gittins Index
J_1	2	(4, 8, 12)	4	0.2
J_2	1		0	0.25
J_3	2		0	0.25

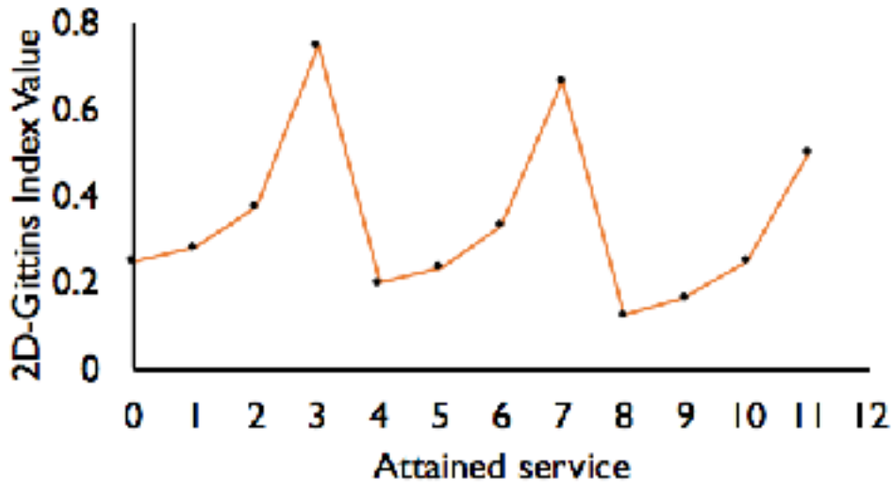


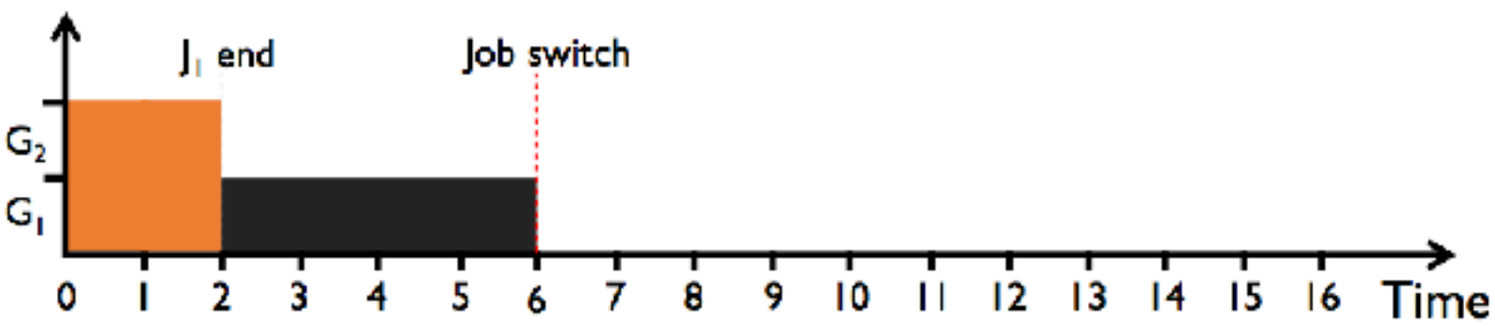


Gittins Index

- Higher probability to complete (Gittins Index), higher priority

	# of GPUs	Distribution	Attained Service	Gittins Index
J_1	2	2	4	0.2
J_2	1	(4, 8, 12)	4	0.2
J_3	2	6	0	0.25

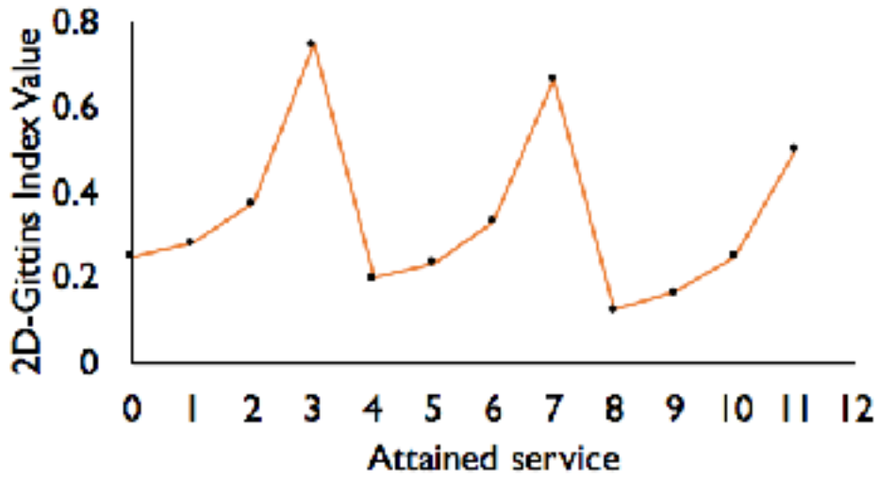


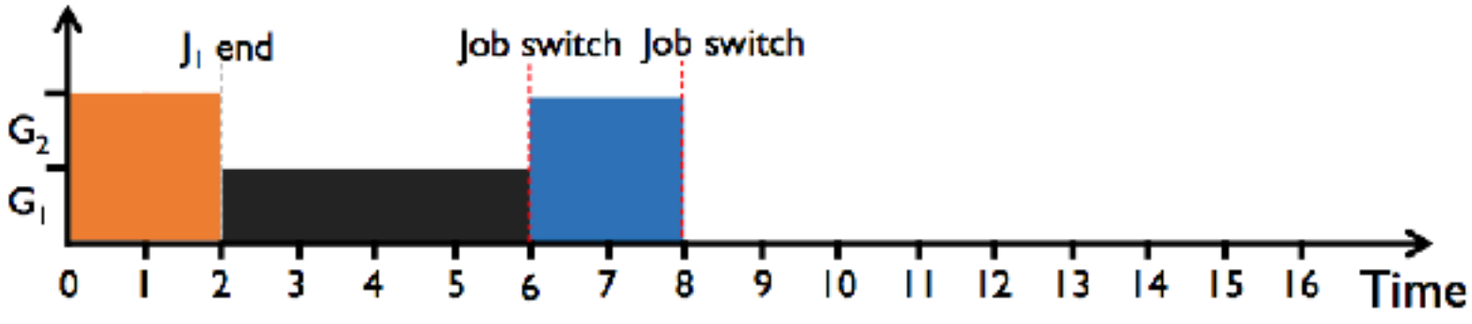


Gittins Index

- Higher probability to complete (Gittins Index), higher priority

	# of GPUs	Distribution	Attained Service	Gittins Index
J_1	2	2	4	0.2
J_2	1	(4, 8, 12)	4	0.2
J_3	2	6	4	0.2

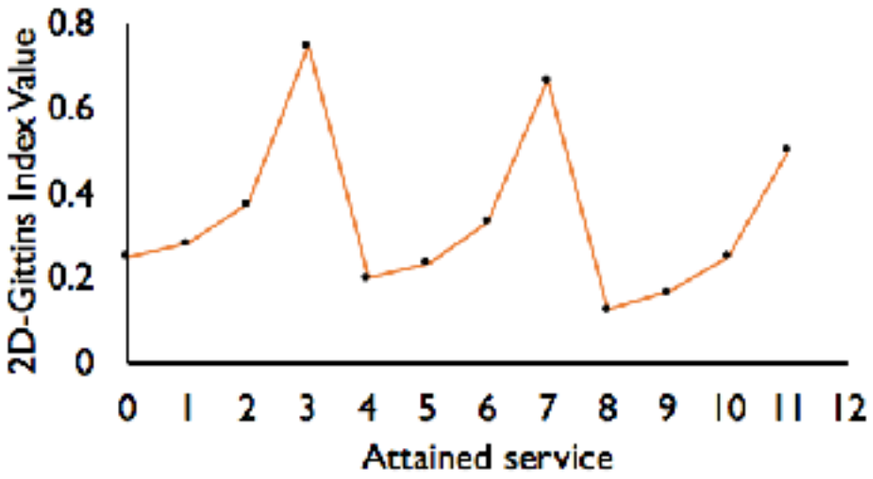


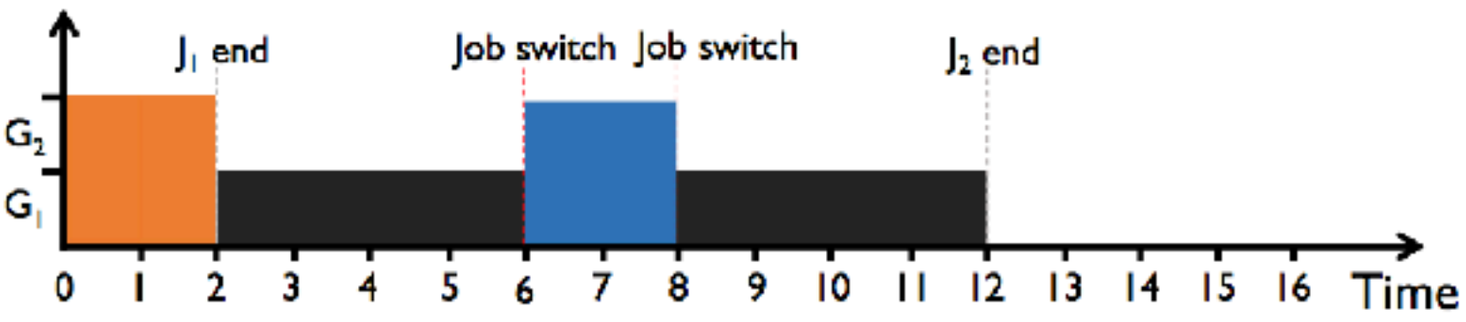


Gittins Index

- Higher probability to complete (Gittins Index), higher priority

	# of GPUs	Distribution	Attained Service	Gittins Index
J_1	2	(4, 8, 12)	4	0.2
J_2	1		8	0.125
J_3	2		4	0.2

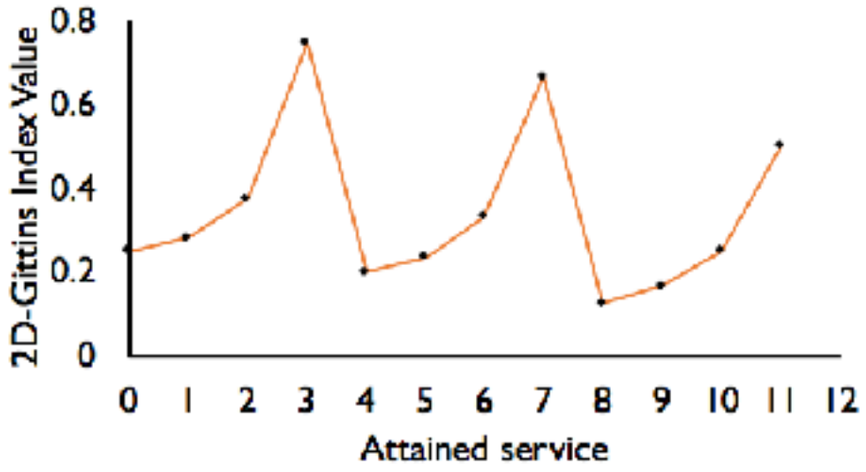


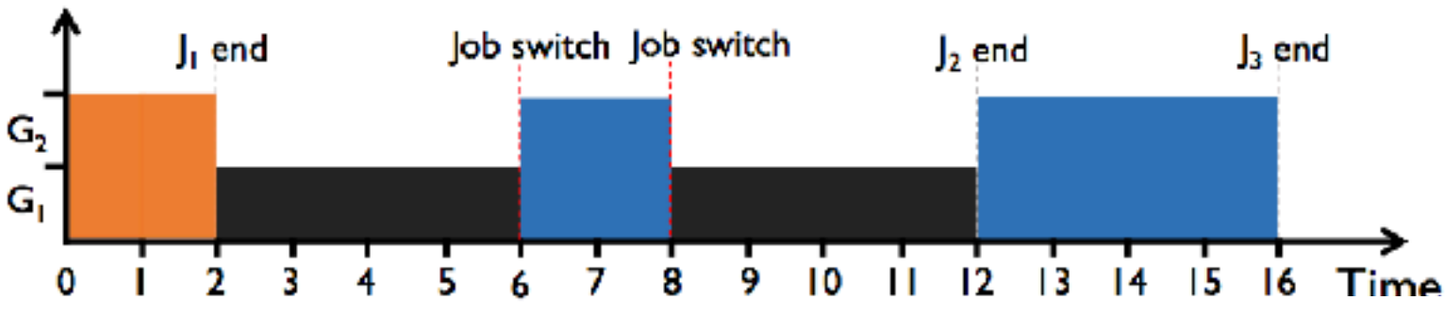


Gittins Index

- Higher probability to complete (Gittins Index), higher priority

	# of GPUs	Distribution	Attained Service	Gittins Index
J_1	2	2	4	0.2
J_2	1	(4, 8, 12)	8	0.125
J_3	2	6	12	N/A





	Extra Information	Avg. JCT
2D-Gittins Index	GPU time distribution	10.0
2D-LAS	None	11.7

$$\mathrm{GI}_J = \sup_{\Delta>0} \frac{\mathrm{P}\left(S - a_J \le \Delta \,\middle|\, S > a_J\right)}{\mathrm{E}\left[\min\{S - a_J, \Delta\} \,\middle|\, S > a_J\right]}$$

Two-Dimensional Age-Based Scheduler (2DAS)

- Age calculated by two-dimensional attained service
  - i.e., a job's total executed GPU time (# of GPUs × executed time)
- No prior information
  - 2D-LAS
- With partial information: distribution of job GPU time
  - 2D-Gittins Index

Two-Dimensional Age-Based Scheduler (2DAS)

- Age calculated by two-dimensional attained service
  - i.e., a job's total executed GPU time (# of GPUs × executed time)

Fewer job switches:

Priority discretization: Discretized-2DAS

  - 2D-LAS
- With partial information: distribution of job GPU time
  - 2D-Gittins Index

Prior Solutions

	I. *Unpredictable Training Time* (*Scheduling*)		II. *Over-Aggressive Job Consolidation* (*Job Placement*)
Optimus[1]	None		None
YARN-CS	*FIFO*		None
Gandiva[2]	*Time-sharing*		*Trial-and-error*
Tiresias	*LAS*	*Gittins Index*	*?*

[1]. Optimus: An Efficient Dynamic Resource Scheduler for Deep Learning Clusters, EuroSys'18
[2]. Gandiva: Introspective Cluster Scheduling for Deep Learning, OSDI'18

Outline

1. Deep Learning

2. GPU Cluster Manager & Weakness of them

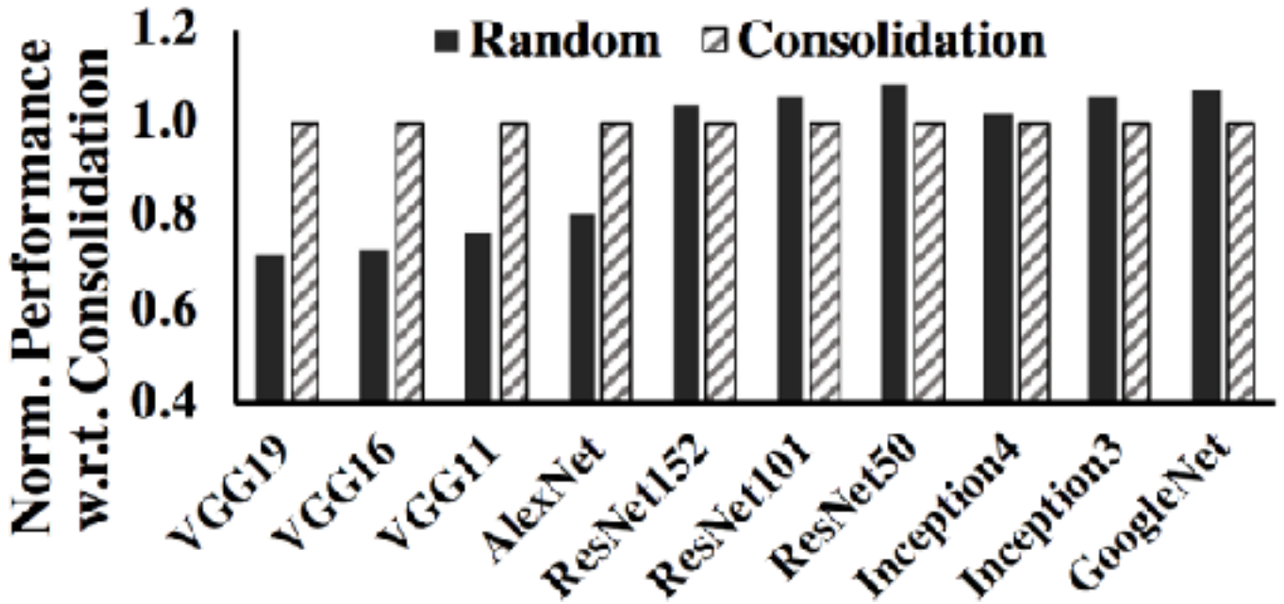
3. Tiresias:

    1.Scheduler

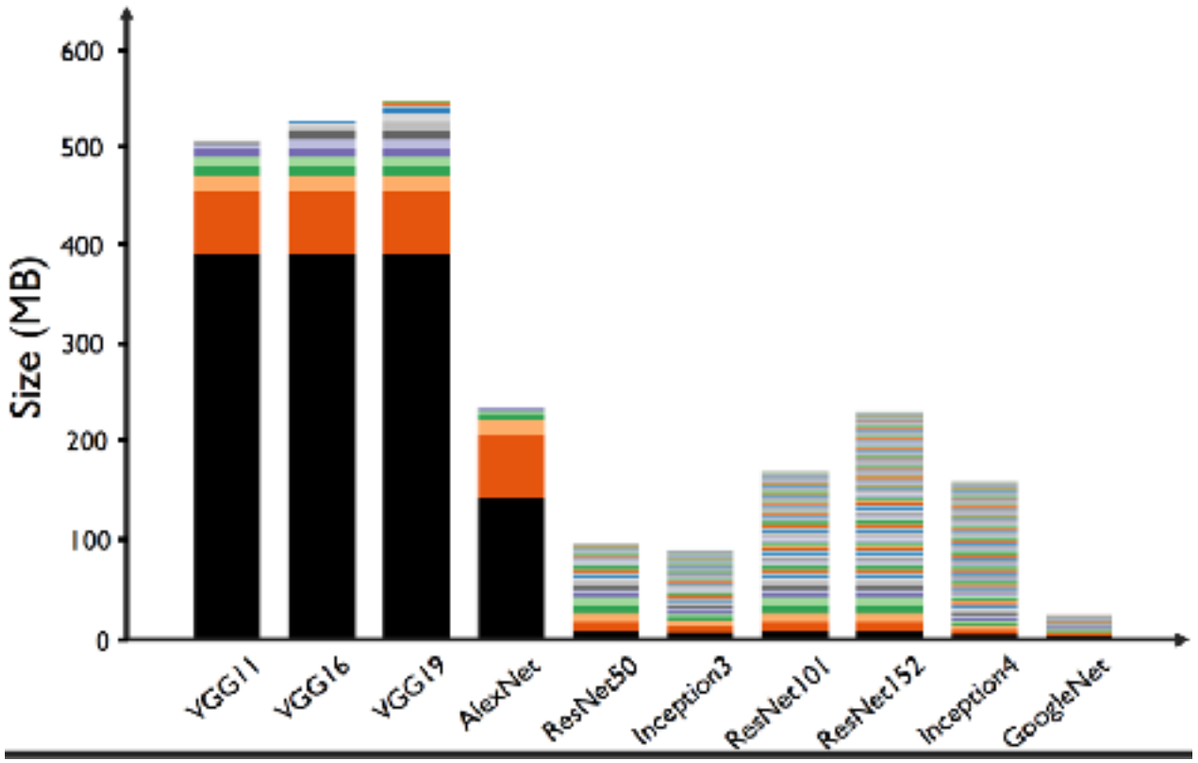
    2. Placement

4. Tiresias: Evaluation

Observation 2



4 concurrent 8-worker jobs with different placement schemes.



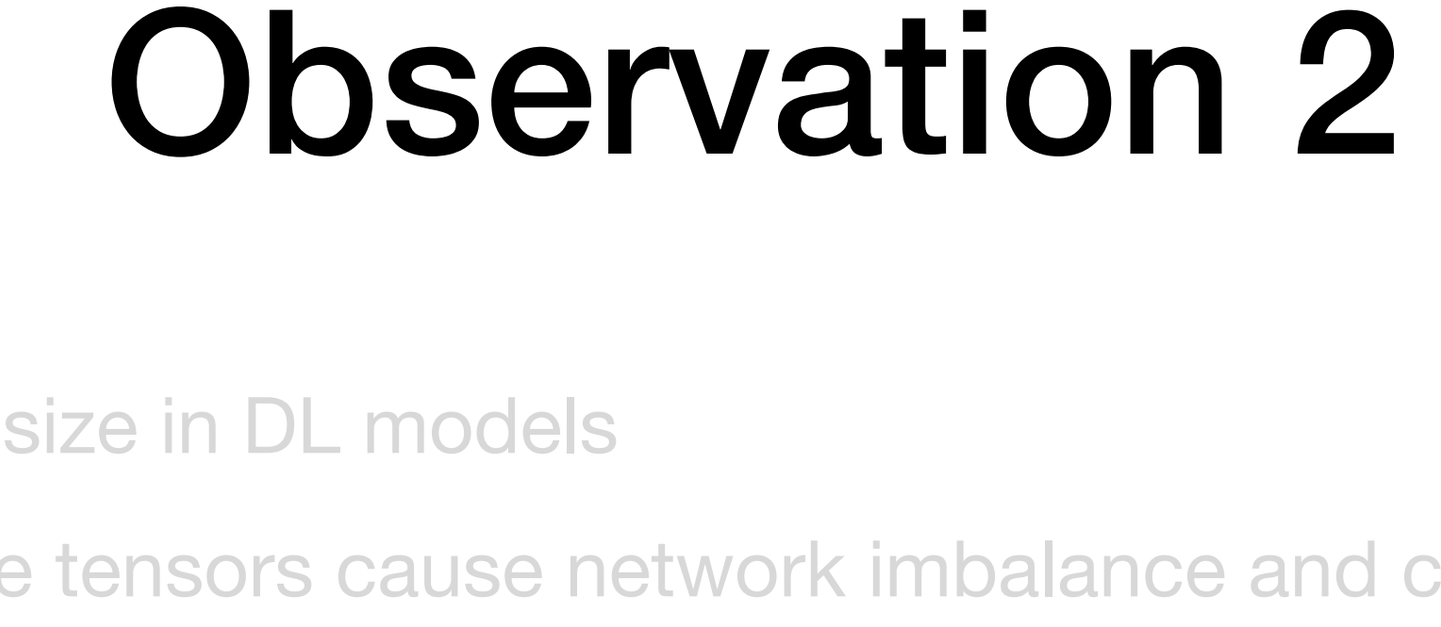
Tensor size in DL models

- Large tensors cause network imbalance and contention

Observation 2

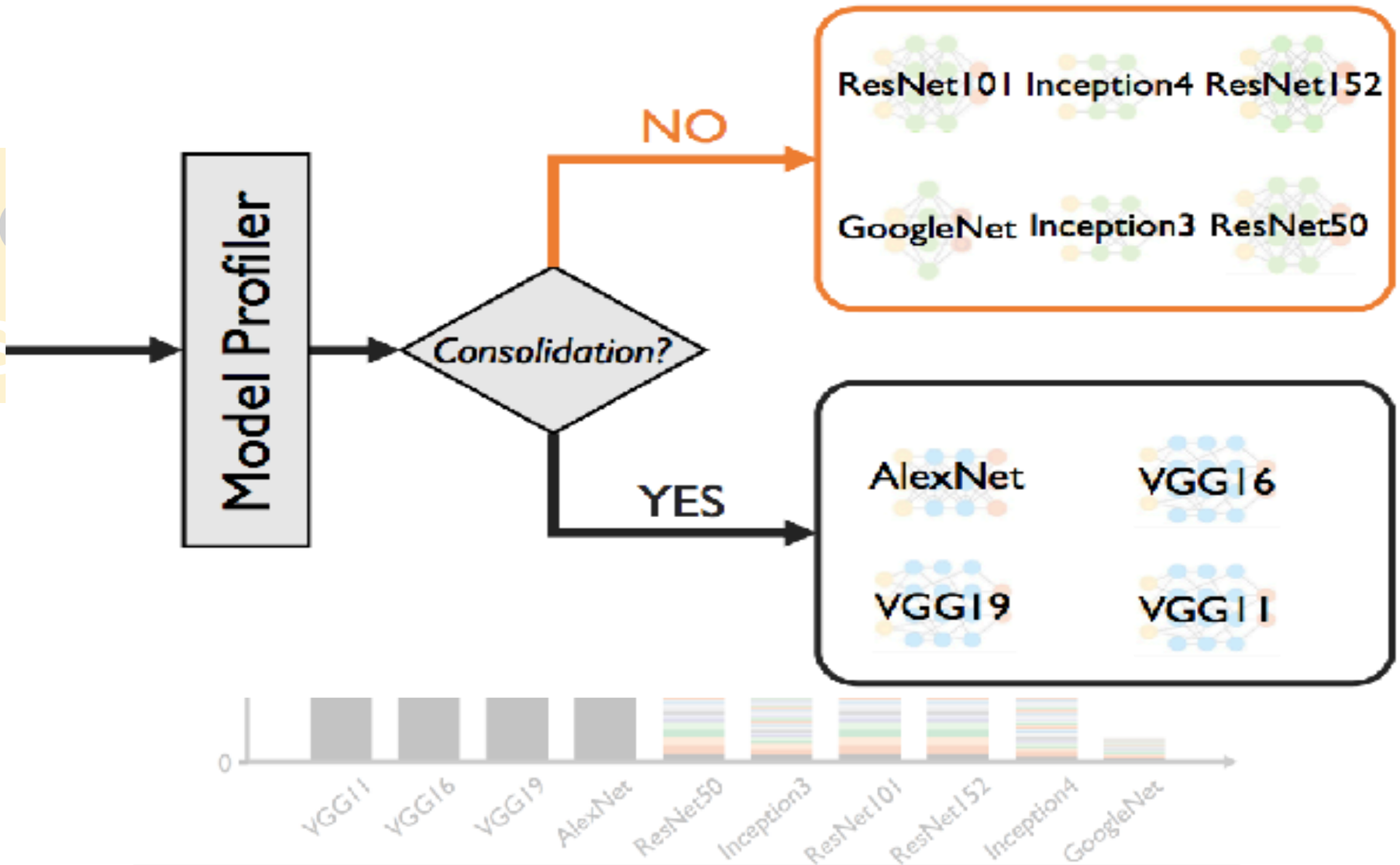
- Tensor size in DL models
  - Large tensors cause network imbalance and contention

Consolidated placement: when the model is highly skewed in its tensor size



Observation 2

- Tensor size in DL models
  - ion



Outline

1. Deep Learning

2. GPU Cluster Manager & Weakness of them

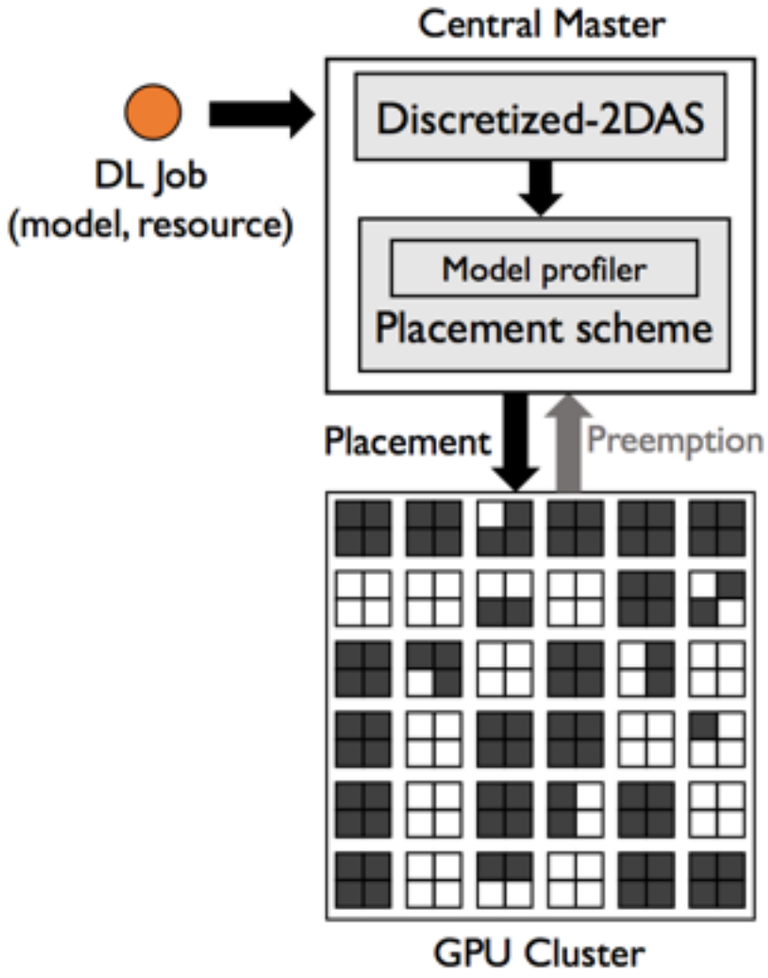
3. Tiresias:

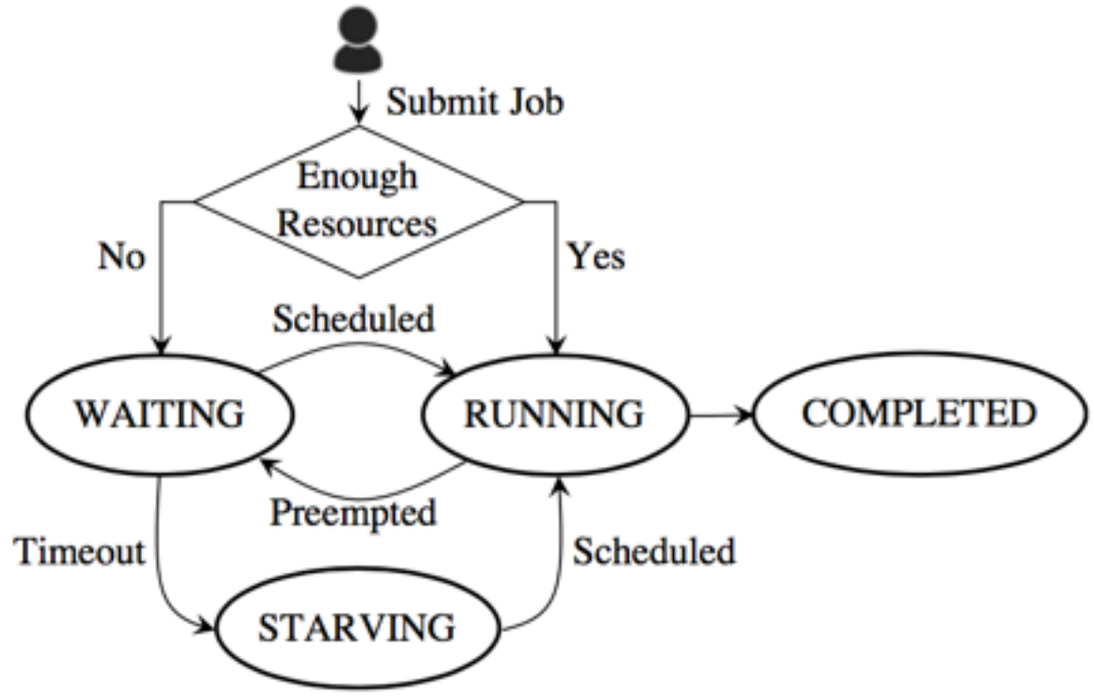
    1.Scheduler

    2. Placement

4. Tiresias: Evaluation

Tiresias





https://github.com/SymbioticLab/Tiresias

Evaluation - Setup

- Testbed Experiment(Michigan ConFlux cluster)
  - 100 Gbps EDR Mellanox IB + RDMA protocol
  - 15*(4-NVIDIA Tesla P100s with NVlink + 256GB DDR4)
  - GPFS(1.2GB/s)
- Large-scale Trace-driven(from Microsoft) Simulation
  - Information: job arrival, completion, demotion, propotion, preemption, #GPU, training time
- #queues: k=2
- threshold=$\Delta=1$

Evaluation - Workload

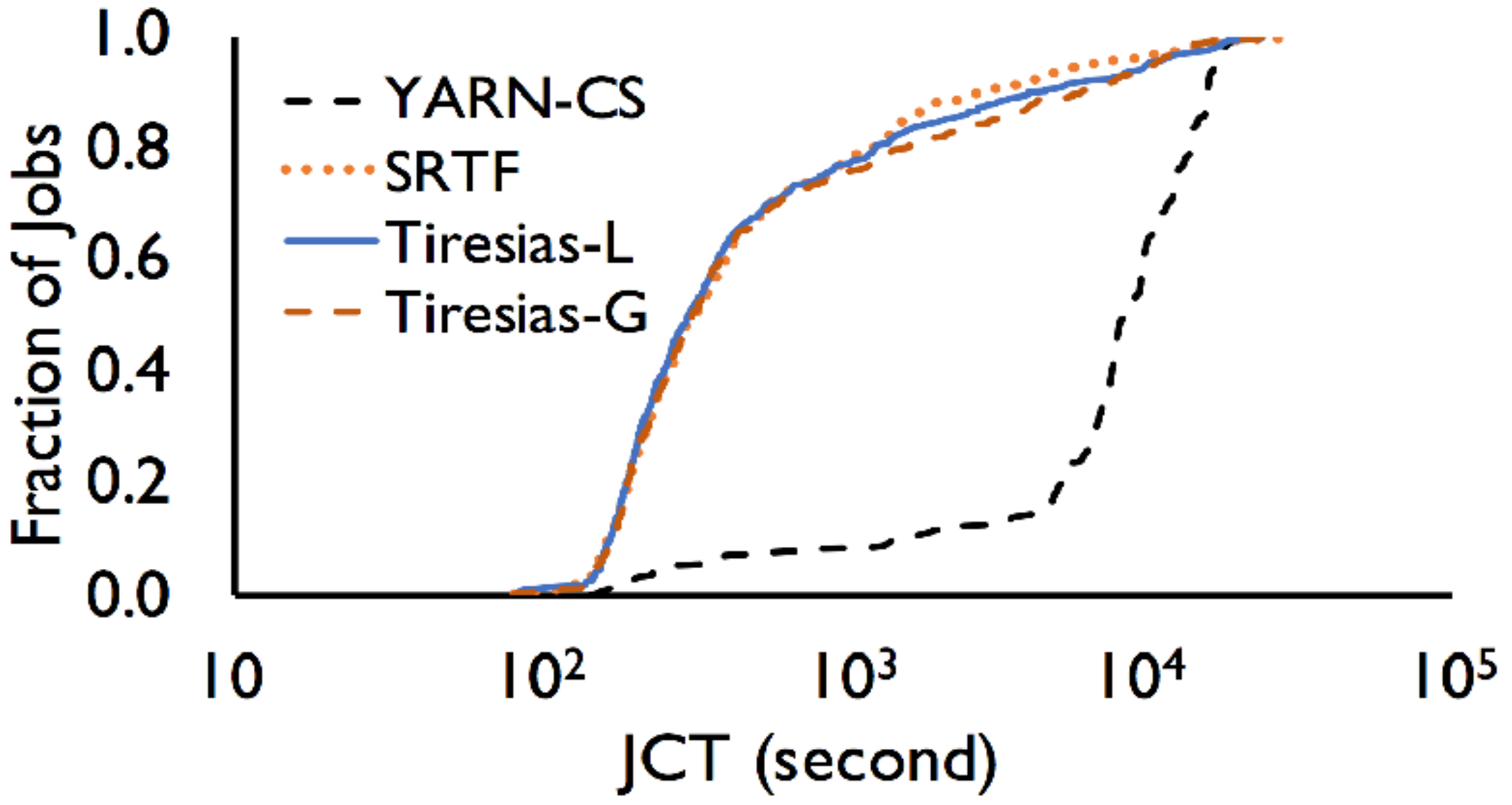
- 480 jobs
  - 240 * 1-GPU jobs
  - 40 * 2-GPU jobs
  - 80 * 4-GPU jobs
  - 90 * 8-GPU jobs
  - 25 * 16-GPU jobs
  - 5 * 32-GPU jobs

Small: <4GPUs

Short: <=800s

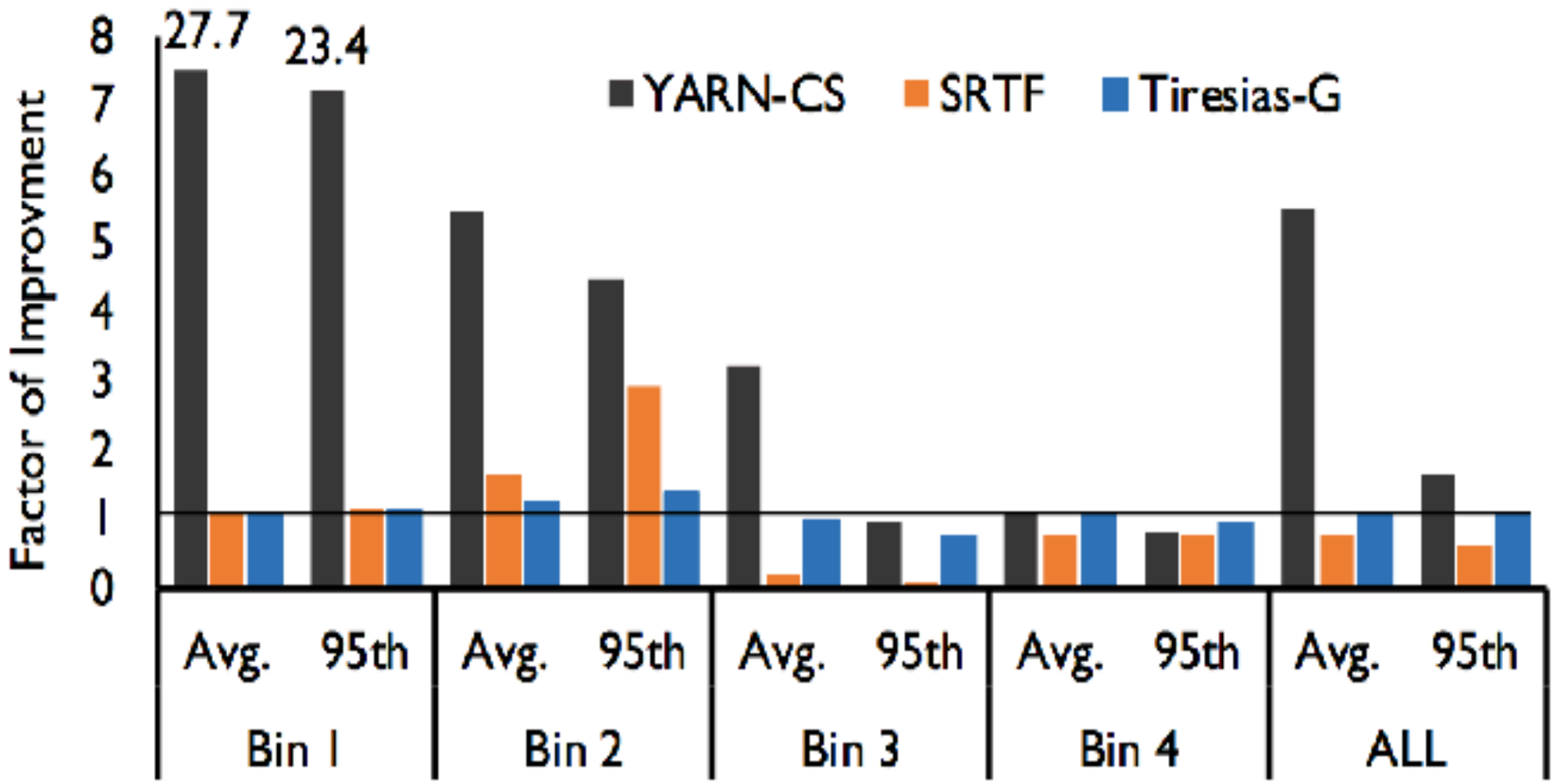
Bin	1 (**S**$\mathbb{S}$)	2 (**S**$\mathbb{L}$)	3 (**L**$\mathbb{S}$)	4 (**L**$\mathbb{L}$)
% of Jobs	63.5%	12.5%	16.5%	7.5%

JCT Improvements in Testbed Experiment



- Avg. JCT improvement (w.r.t.YARN-CS): 5.5×
- Comparable performance to SRTF

JCT in Testbed Experiment



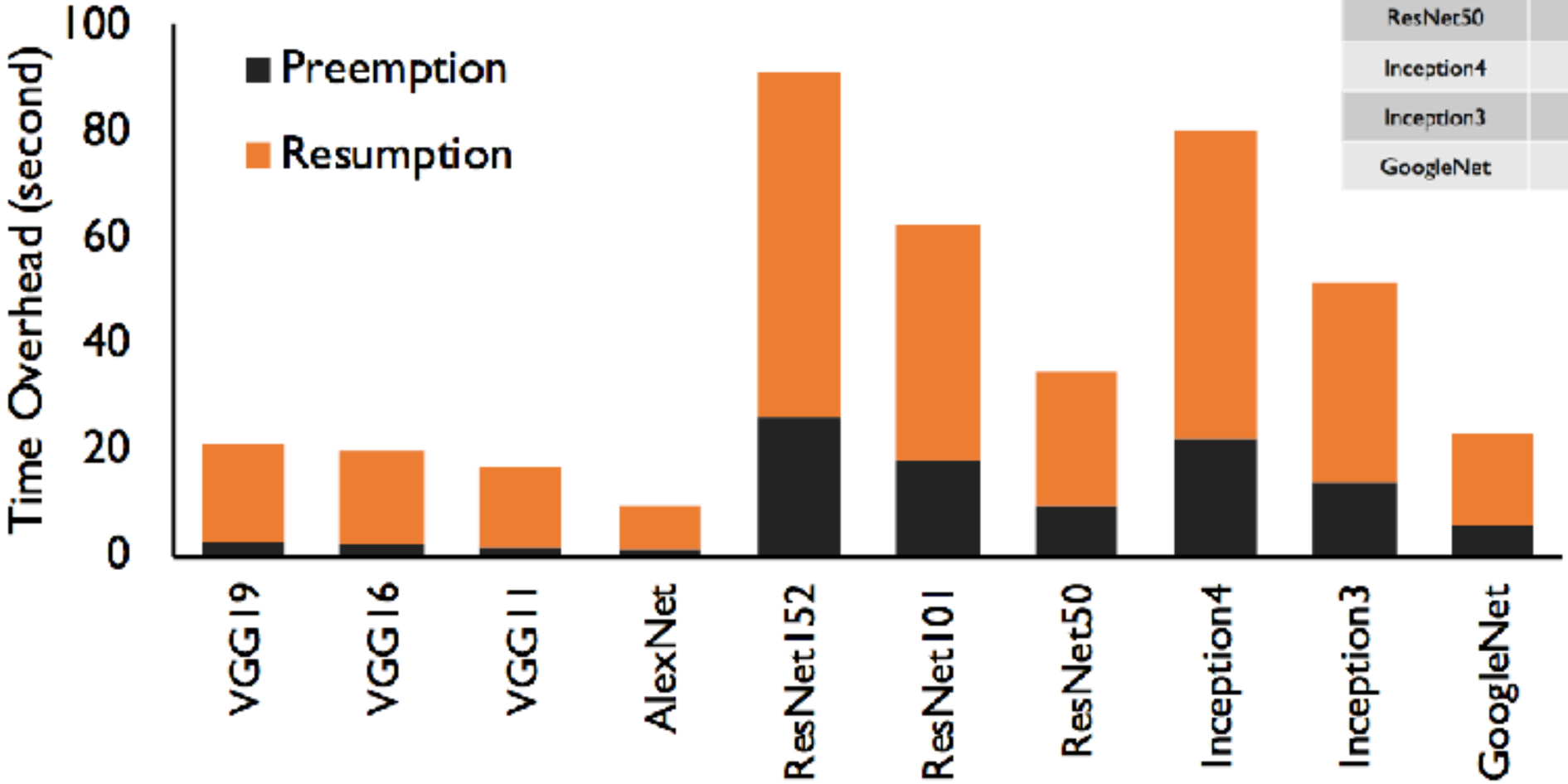
Bins	1(Small-Short)	2(Small-Long)	3(Large-Short)	4(Large-Long)
% of Jobs	63.5%	12.5%	16.5%	7.5%

Queuing Delay inTestbed Experiment

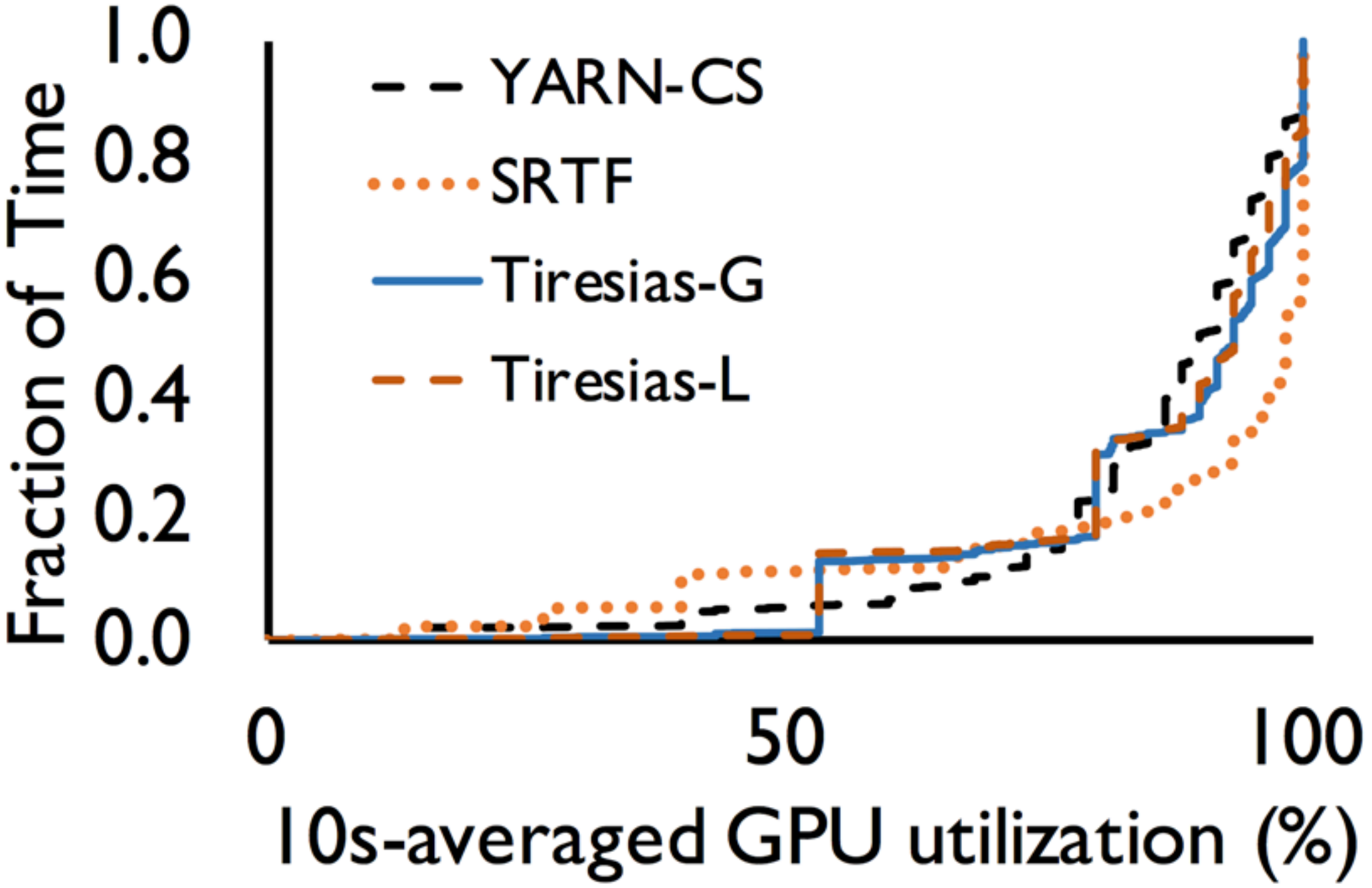
	Average	Median	95th
YARN-CS	8146s	7464s	15327s
SRTF	593s	32s	3133s
Tiresias-G	1005s	39s	7933s
Tiresias-L	963s	13s	7755s

Time overhead of Job switch

Model	Total size (MB)	Largest tensor (MB)
VGG19	548	382
VGG16	527	392
VGG11	506	392
AlexNet	235	144
ResNet152	230	9
ResNet101	170	9
ResNet50	98	9
Inception4	163	6
Inception3	91	8
GoogleNet	27	4

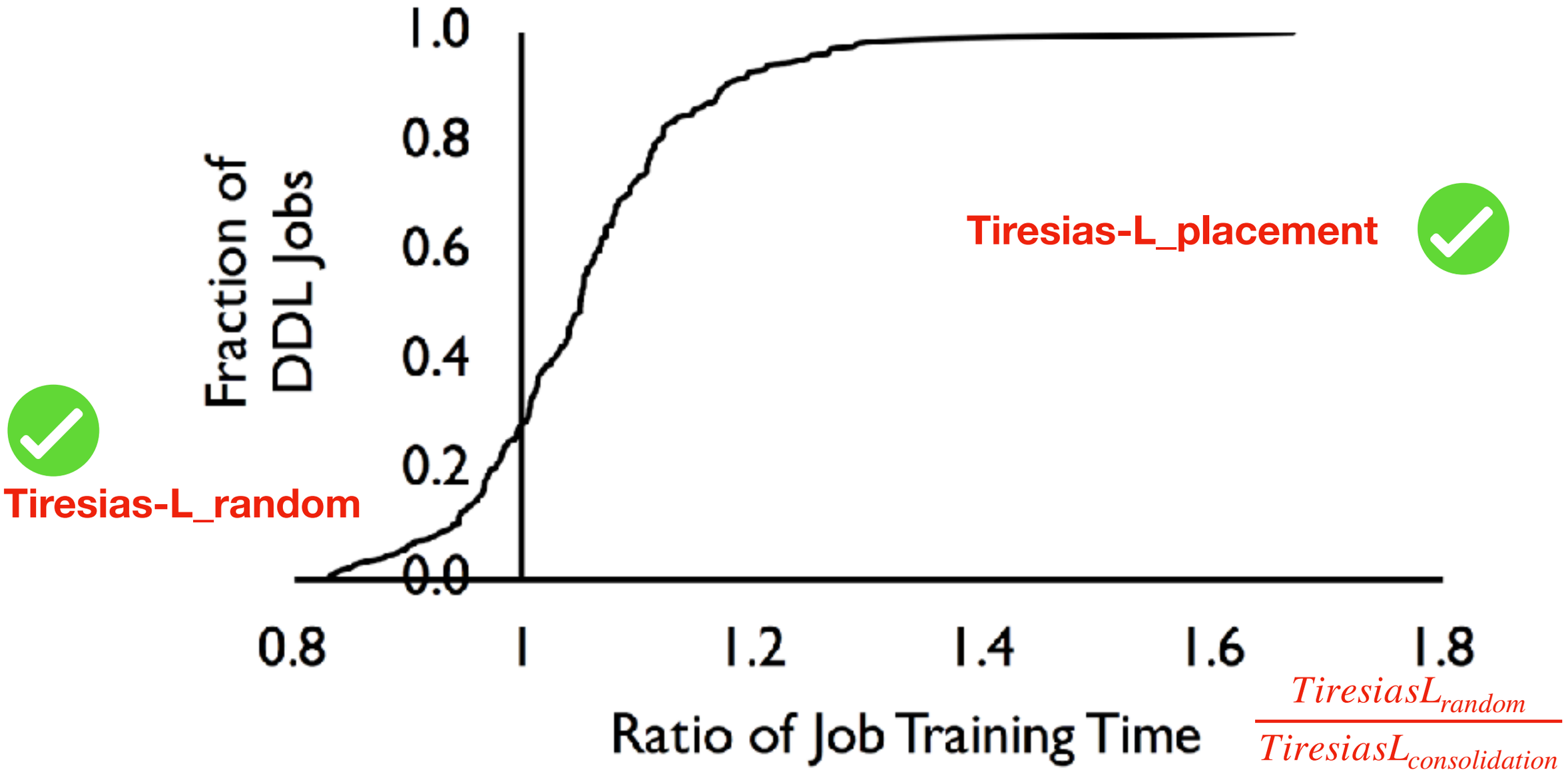


GPU Utilization in Testbed



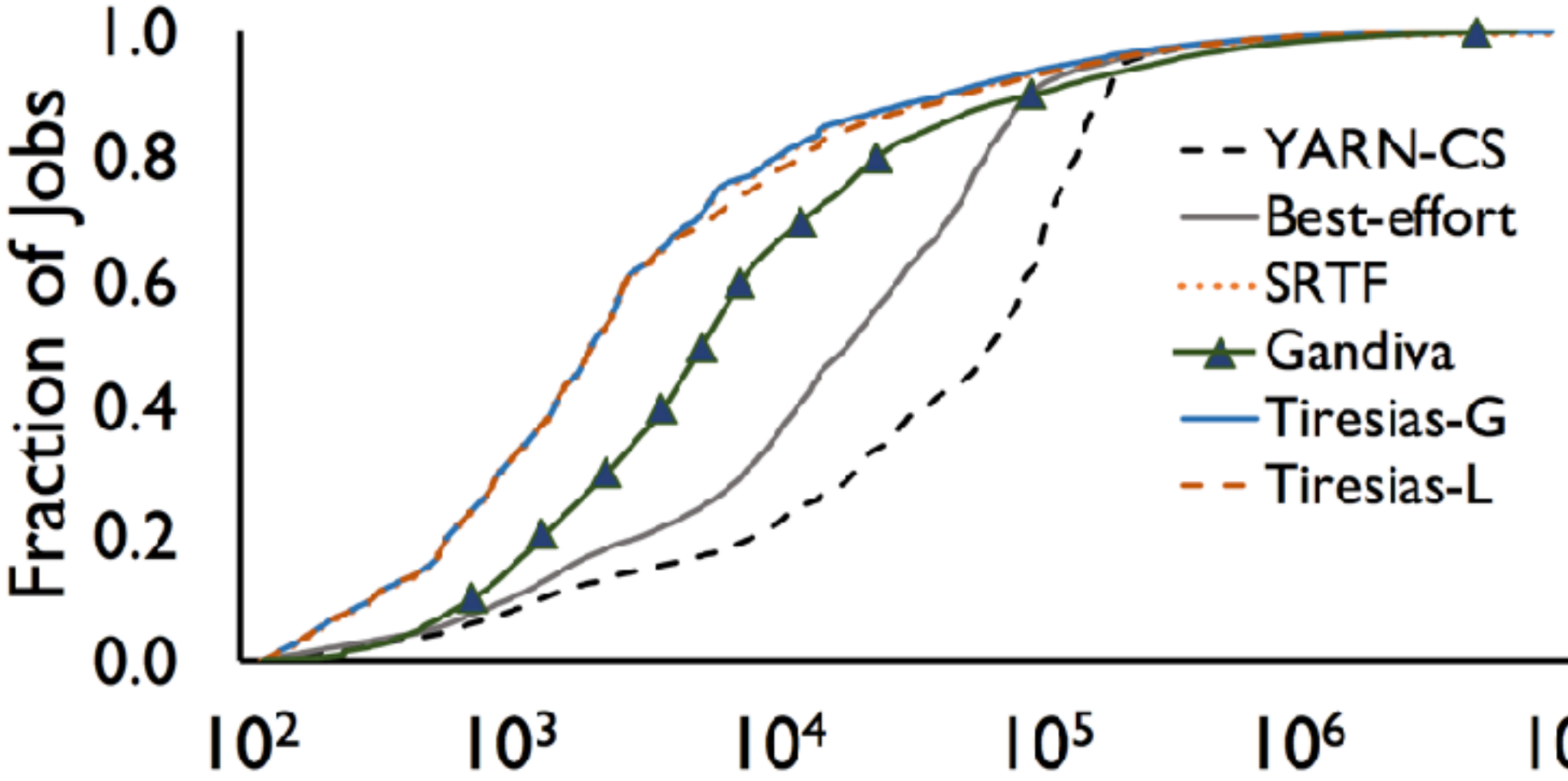
- The makespan is improved by 1.21× (w.r.t. YARN-CS)

Training Performance in Testbed Experiment



- Training time whenTiresias-L running with and without placement

JCT Improvements in Trace-Driven **Simulation**



- 10-week job trace from Microsoft • 2,000
- Avg. JCT improvement (w.r.t. Gandiva): 2

	Average	Median	95th
YARN-CS	2.41×	30.85×	1.25×
SRTF	1.00×	1.00×	0.84×
Gandiva	2.00×	2.59×	2.08×
Tiresias-G	0.97×	1.00×	0.85×

Sensitivity Analysis of 2D-LAS

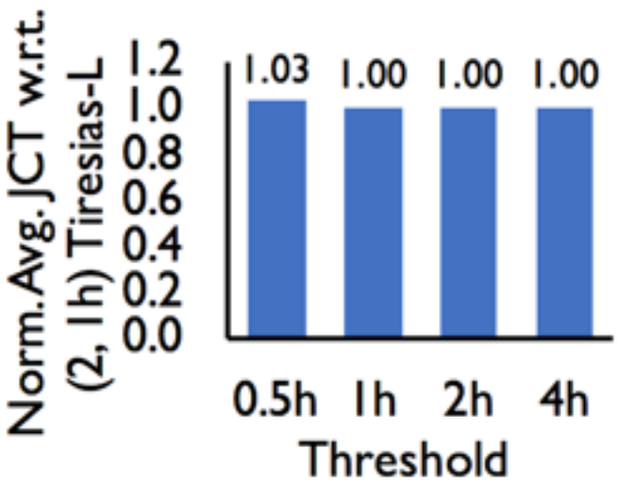


Fig14a

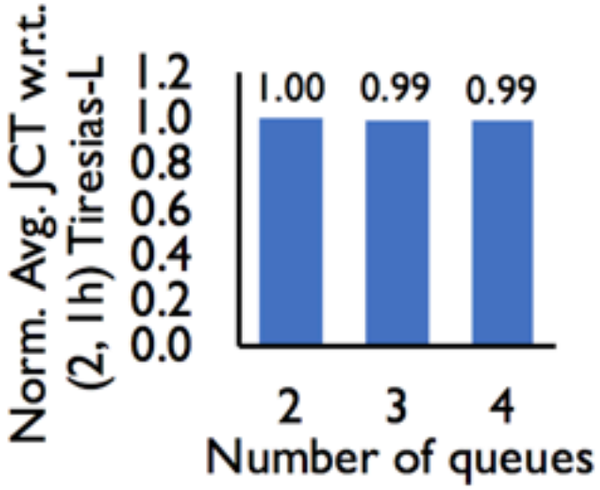


Fig14b

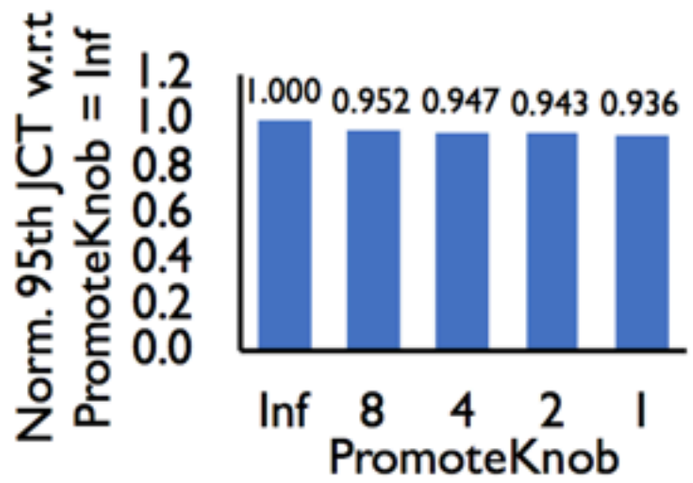


Fig14c

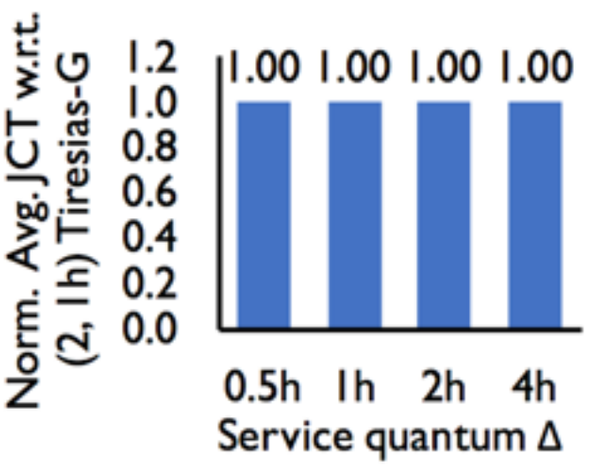


Fig15a

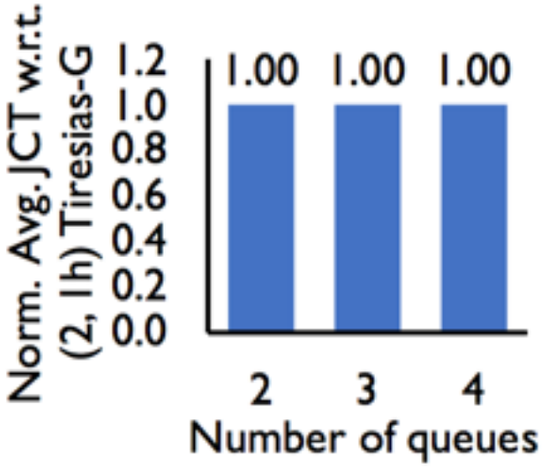


Fig15b

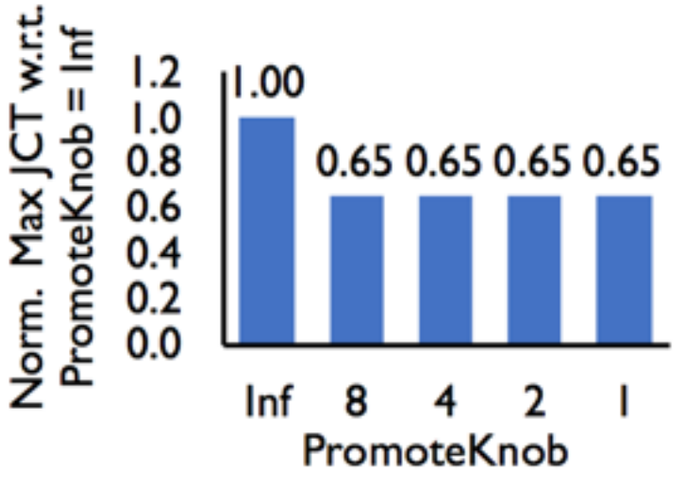


Fig15c

Tiresias: A GPU Cluster Manager for Distributed Deep Learning

NSDI'19

Q&A

.

Backup

- N/A