Fluid: Resource-aware Hyperparameter Tuning Engine

Peifeng Yu†, Jiachen Liu†, Mosharaf Chowdhury

† Equal contribution

SymbioticLab

UNIVERSITY OF MICHIGAN

Outline

1. Background and Motivation
2. Abstraction and Algorithms
3. Evaluation

Hyperparameter Tuning Today

- Hyperparameters
  - # of layers/# of neurons
  - Dropout rate
  - # of channels
  - Learning rate
  - Optimizer parameters
  - Etc.

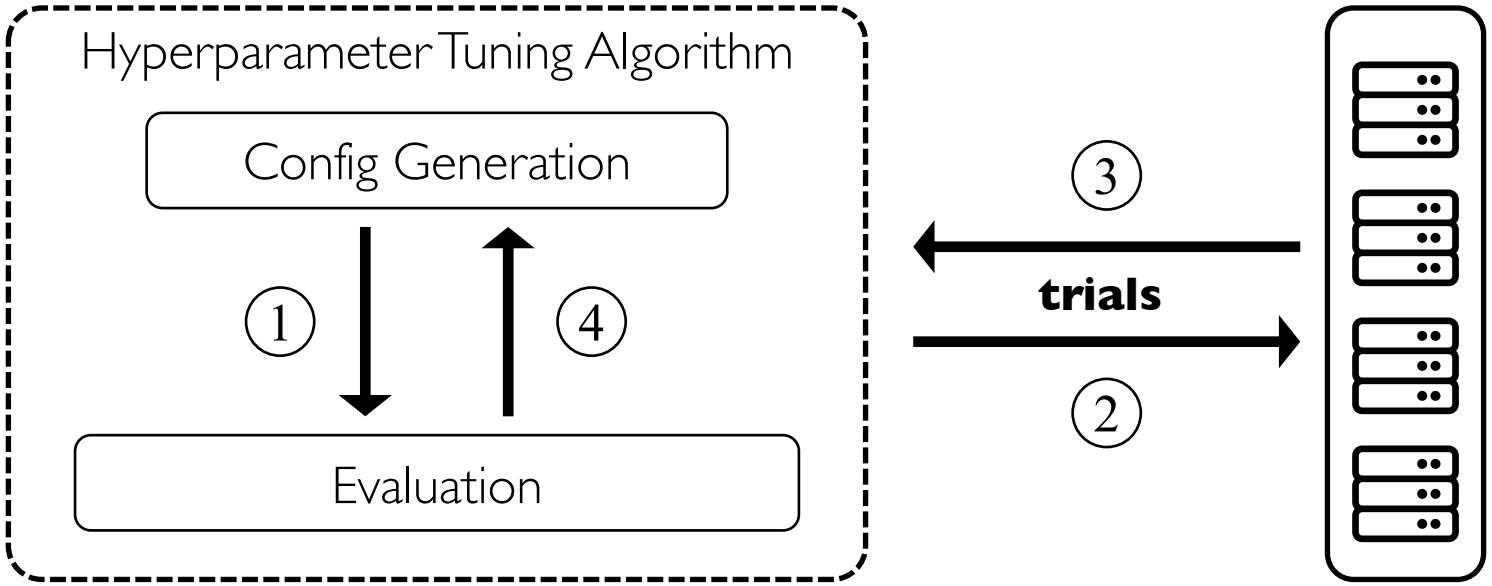
- Non-differentiable & high dimensional search space

Hyperparameter Tuning Today

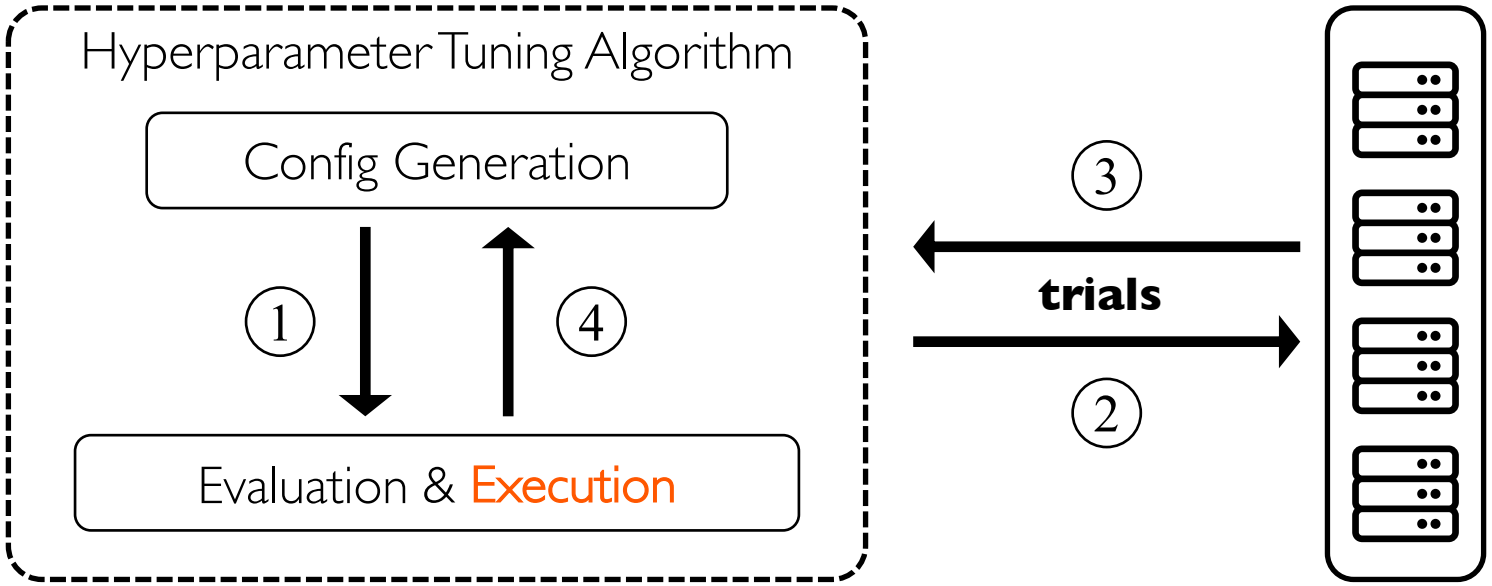
- Evaluation of hyperparameters is time/resource consuming
  - train a model to know if it works

- Many algorithms & techniques
  - Random/Grid
  - Model-based config generation
  - Early stopping/successive halving
  - Many others

Hyperparameter Tuning Today

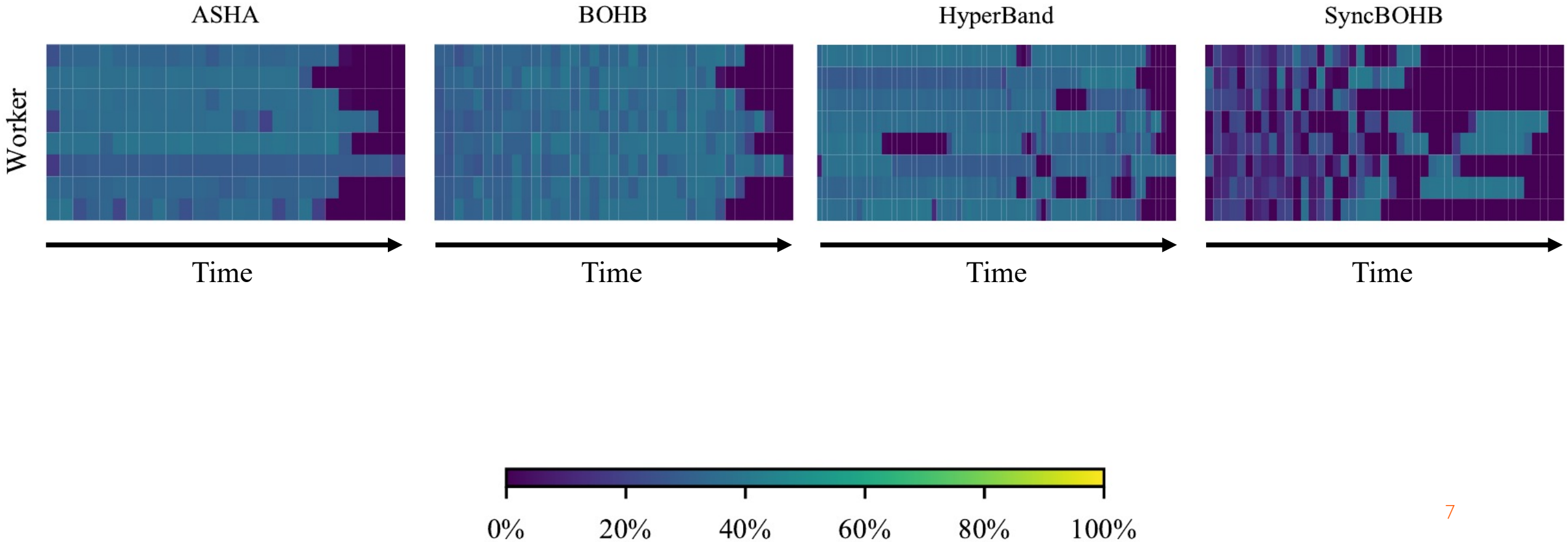


Hyperparameter Tuning Today



- **Direct** interaction with the cluster to execute **trials**
- Trials gets executed in FIFO order

Trials Execution



Trials Execution

- Lack of elasticity reduces utilization
  - Existing trials can not easily use new idle workers
- High utilization != useful work
  - Asynchronous Successive Halving (ASHA)
    - Trial concurrency == # of workers
    - Can not scale up beyond a certain # of workers

- *Yet another* hyperparameter tuning algorithm?

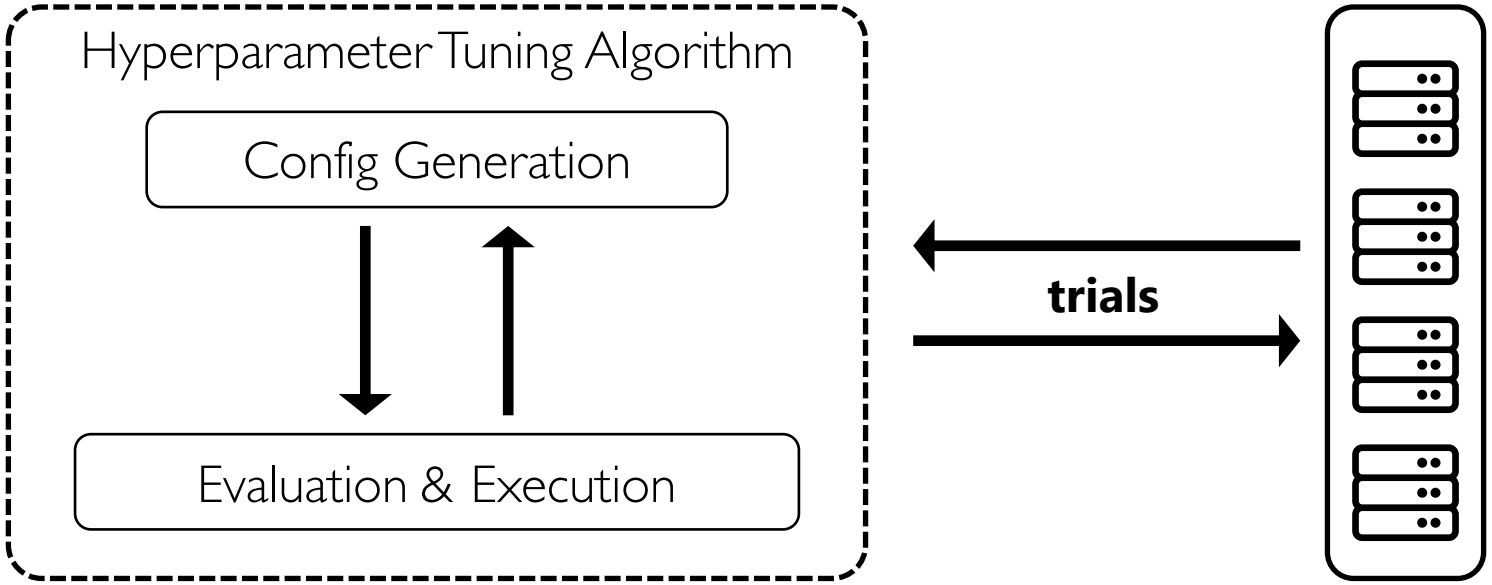
Trials Execution

- What about existing algorithms?
  - Mostly still FIFO
- Execution is tightly coupled with algorithm
  - Hard to apply to other algorithms
  - Hard to improve w/o deep knowledge of the algorithm itself

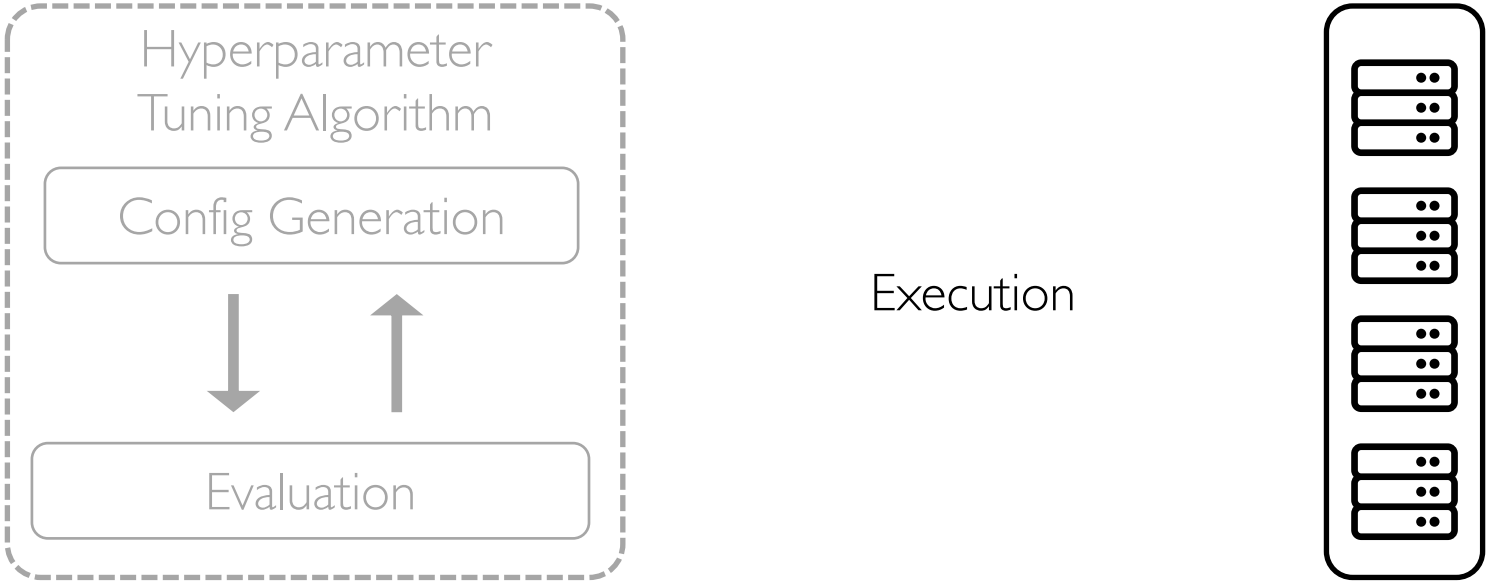
All problems in computer science can be solved by another level of indirection

--- *Butler Lampson*

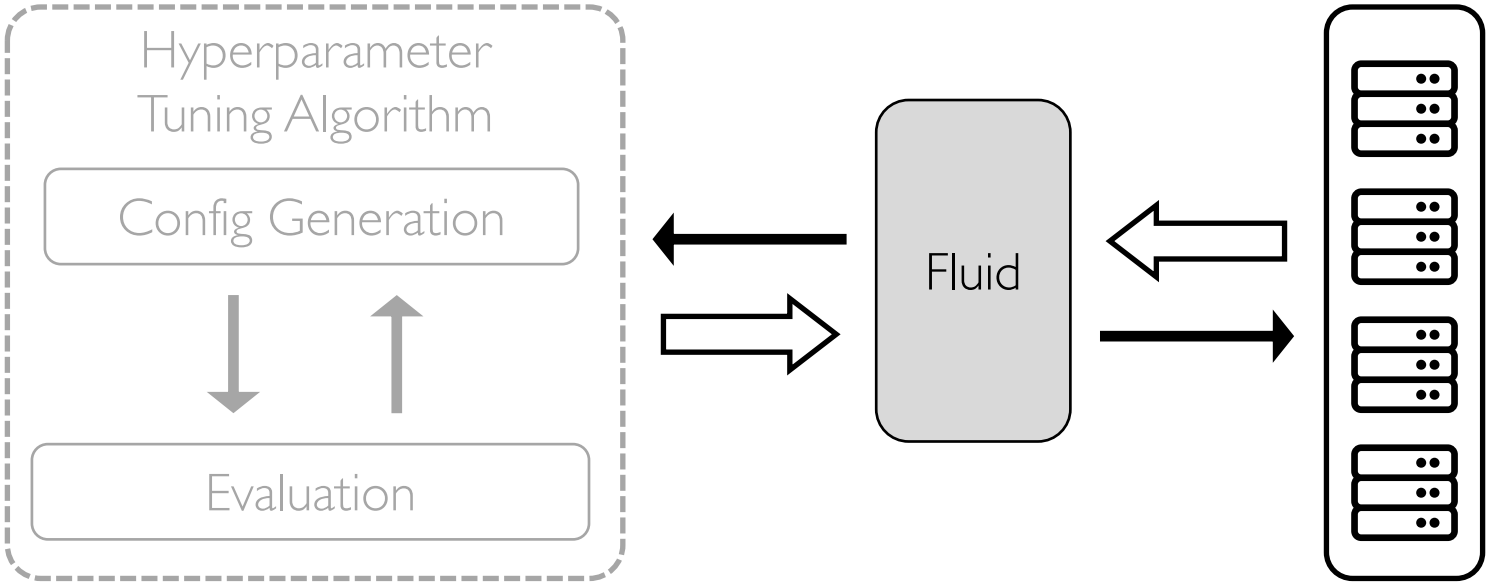
Hyperparameter Execution Engine: Fluid



Hyperparameter Execution Engine: **Fluid**



Hyperparameter Execution Engine: Fluid



- Wide variety of tuning algorithms
  - Random/Iterative/Sequential
  - ✓ TrialGroup

- Heterogeneity & dynamicity
- ✓ Multiple source of parallelism
  - Inter-GPU: elastic distributed training
  - Intra-GPU: Nvidia MPS
- ✓ `StaticFluid/DynamicFluid`

Outline

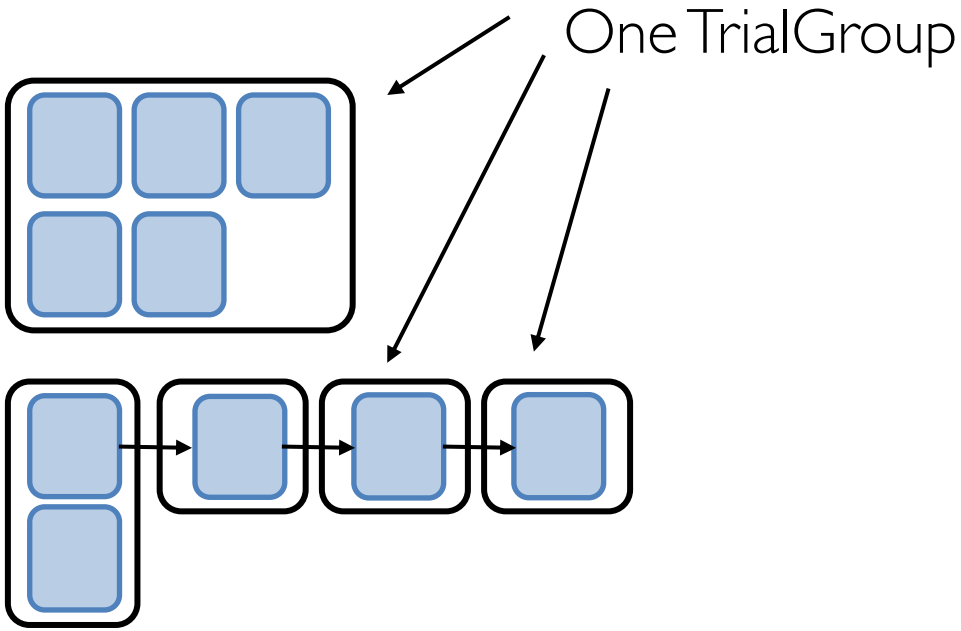
1. Background and Motivation
2. Abstraction and Algorithms
3. Evaluation

The Interface: TrialGroup

- Definition

  A group of training trials with a training budget associated to each trial.

- Example
  - Given 5 trials to evaluate:  x5
  - Grid/random search:
  - Sequential model-based algorithms:



The Interface: TrialGroup

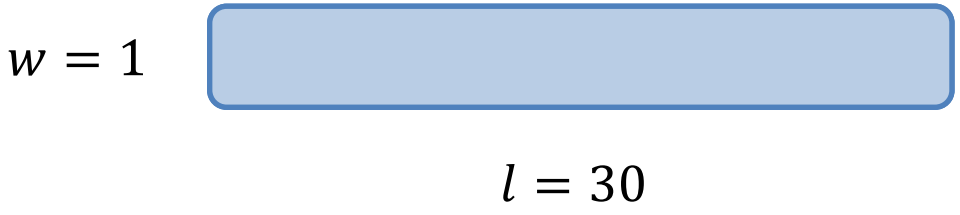
- Definition

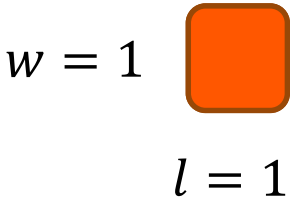
  A group of training trials with a training budget associated to each trial.

- Generalization
  - All kinds of hyperparameter tuning algorithms could be expressed by a sequence of TrialGroup and executed by Fluid.

Problem Definition: Strip Packing

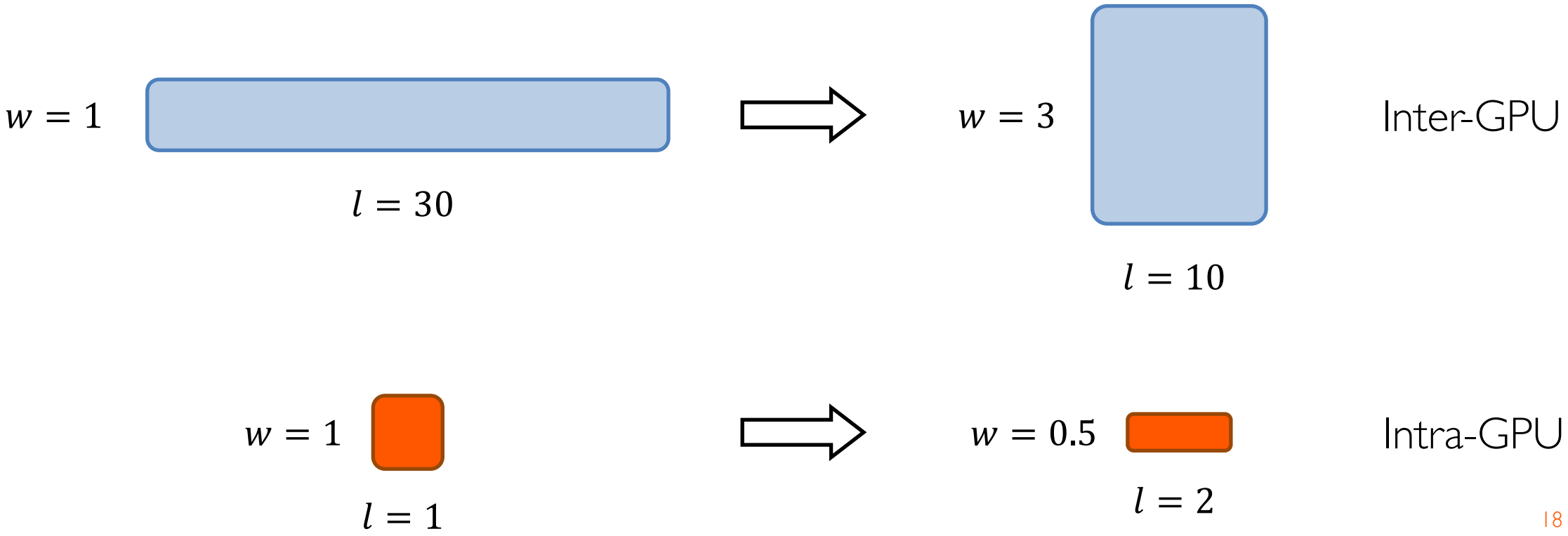
- Input: TrialGroup $A = \{a_1, a_2, \cdots, a_k\}$, resources $M = \{m_1, m_2, \cdots, m_n\}$
- Output: resource allocation $W = \{w_1, w_2, \cdots, w_n\}$
- Goal: minimize the length $\mathbf{L}$ of strips



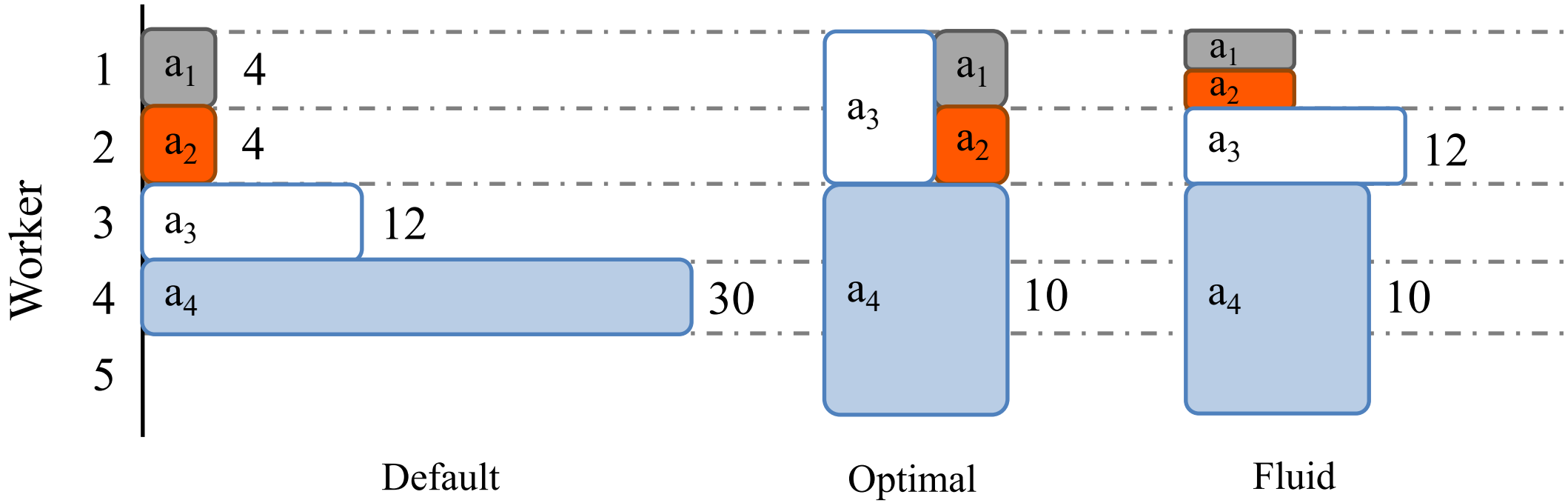


Problem Definition: Strip Packing

- Input: TrialGroup $A = \{a_1, a_2, \cdots, a_k\}$, resources $M = \{m_1, m_2, \cdots, m_n\}$
- Output: resource allocation $W = \{w_1, w_2, \cdots, w_n\}$
- Goal: minimize the length L of strips



Toy Example



Different solutions to execute 4 trials (1 TrialGroup) scheduled on 5 workers

Fully utilize the resources and mitigate the straggler

Algorithm: `StaticFluid`

$$w_i = \min\left(\max\left(\left\lfloor \frac{h_{i,1}}{\sum_j h_{j,1}} n \right\rfloor, \frac{1}{c}\right), d\right)$$

Intra-GPU overhead

Inter-GPU overhead

Budget ratio

- h: trial training budget
- n: available resources
- c: maximum intra-GPU parallelism (# of packing trials)
- d: maximum inter-GPU parallelism (# of distributed workers)

Algorithm: `DynamicFluid`

$$w_i = \min(\max(\left\lfloor \frac{h_{i,1}}{\sum_j h_{j,1}} n \right\rfloor, \frac{1}{c}), d)$$

Minimize the makespan of multiple TrialGroups

Dynamically changing resources

Outline

1. Background and Motivation
2. Abstraction and Algorithms
3. Evaluation

Evaluation Setup

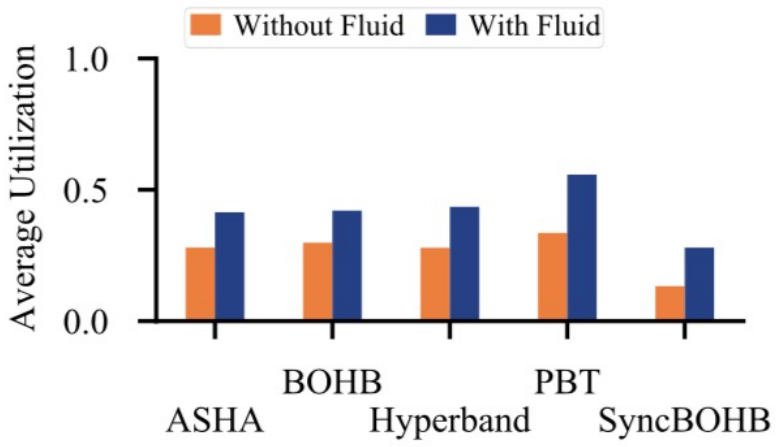
- Implementation: an alternative Ray[1] executor
- Workloads

Task	Base Model	# of Params.	Target
CIFAR-10	AlexNet	7	Acc. >= 90%
WLM	RNN	10	PPL <= 140
DCGAN	CNN	2	Inception >= 5.2

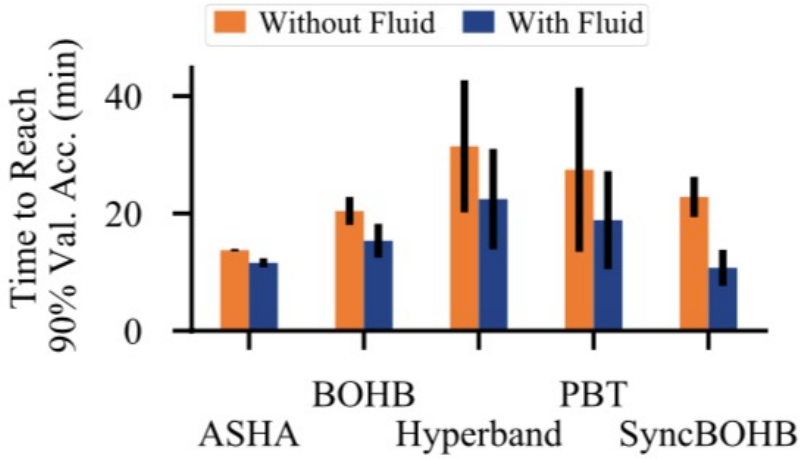
[1]: https://github.com/ray-project/ray

Evaluation Results

- Average resource utilization: **10%-100%** improvement
- Average job completion time: **10%-70%** improvement



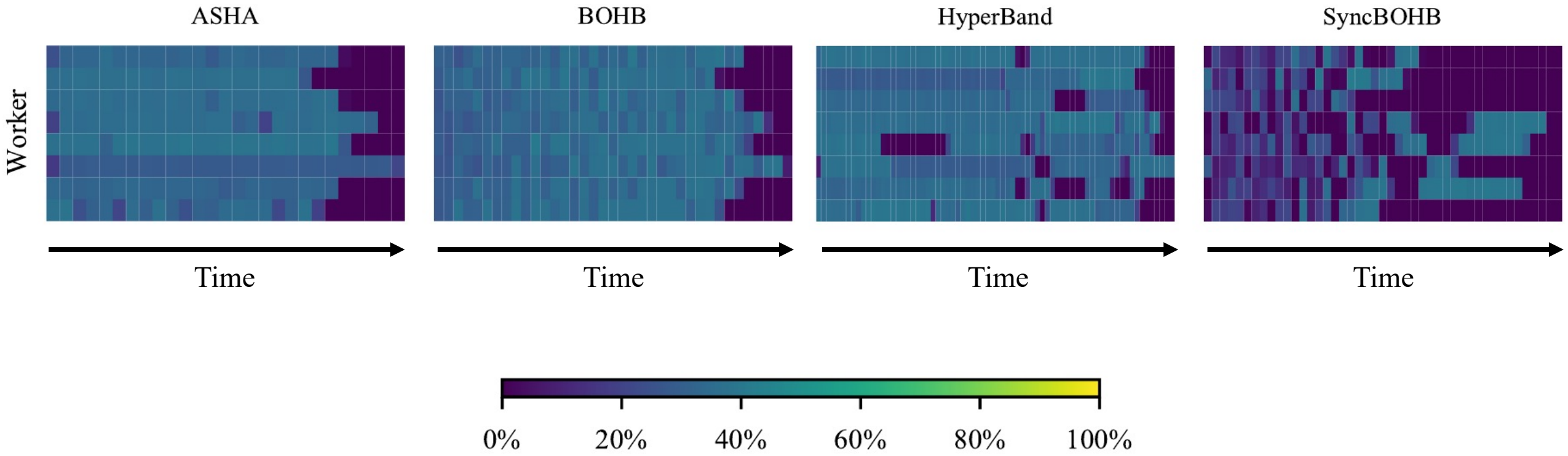
(a) CIFAR-10



(b) CIFAR-10

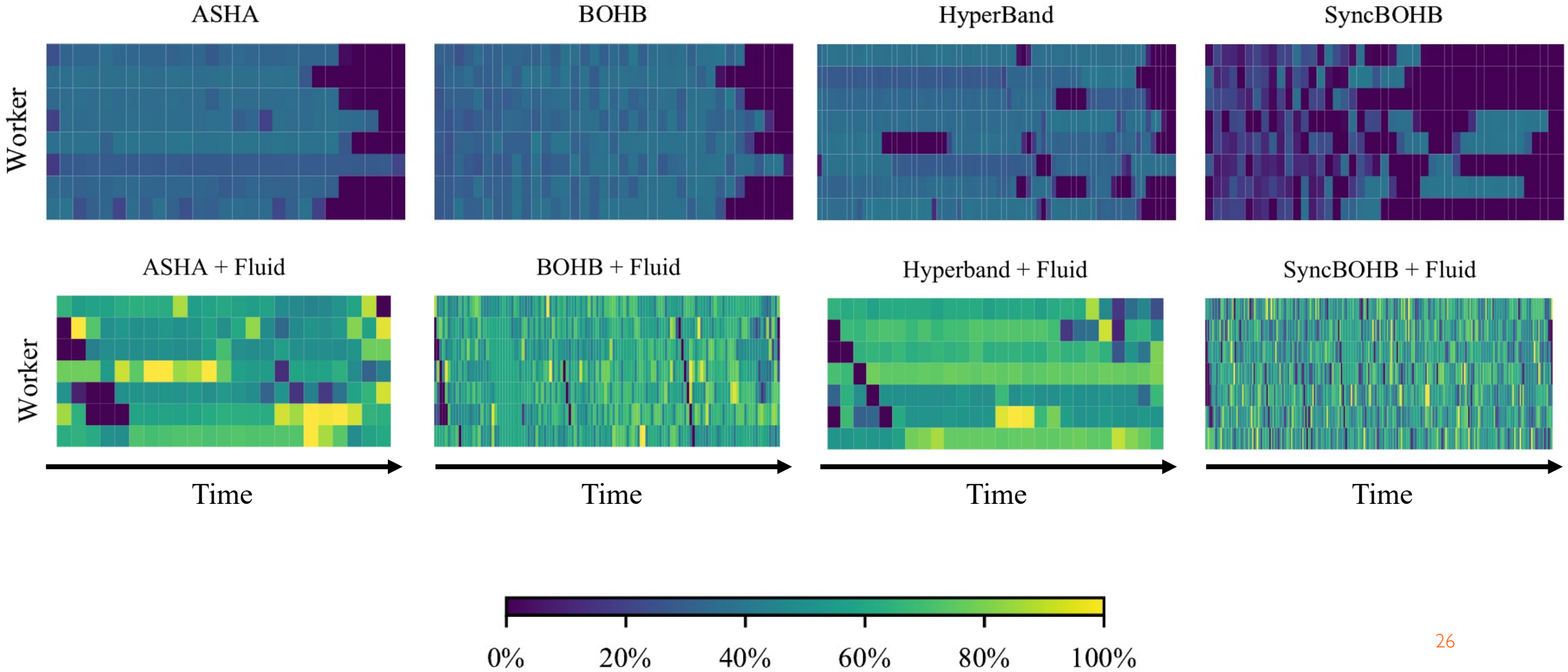
Evaluation Results: Visualization

Resource utilization over time



Evaluation Results: Visualization

Resource utilization over time



Conclusion

- Fluid
  - Hyperparameter tuning execution engine
  - Can be combined with most tuning algorithms
  - Improve utilization and end-to-end tuning time
- Open source
  - https://github.com/SymbioticLab/fluid
- Q&A

SymbioticLab

UNIVERSITY OF MICHIGAN