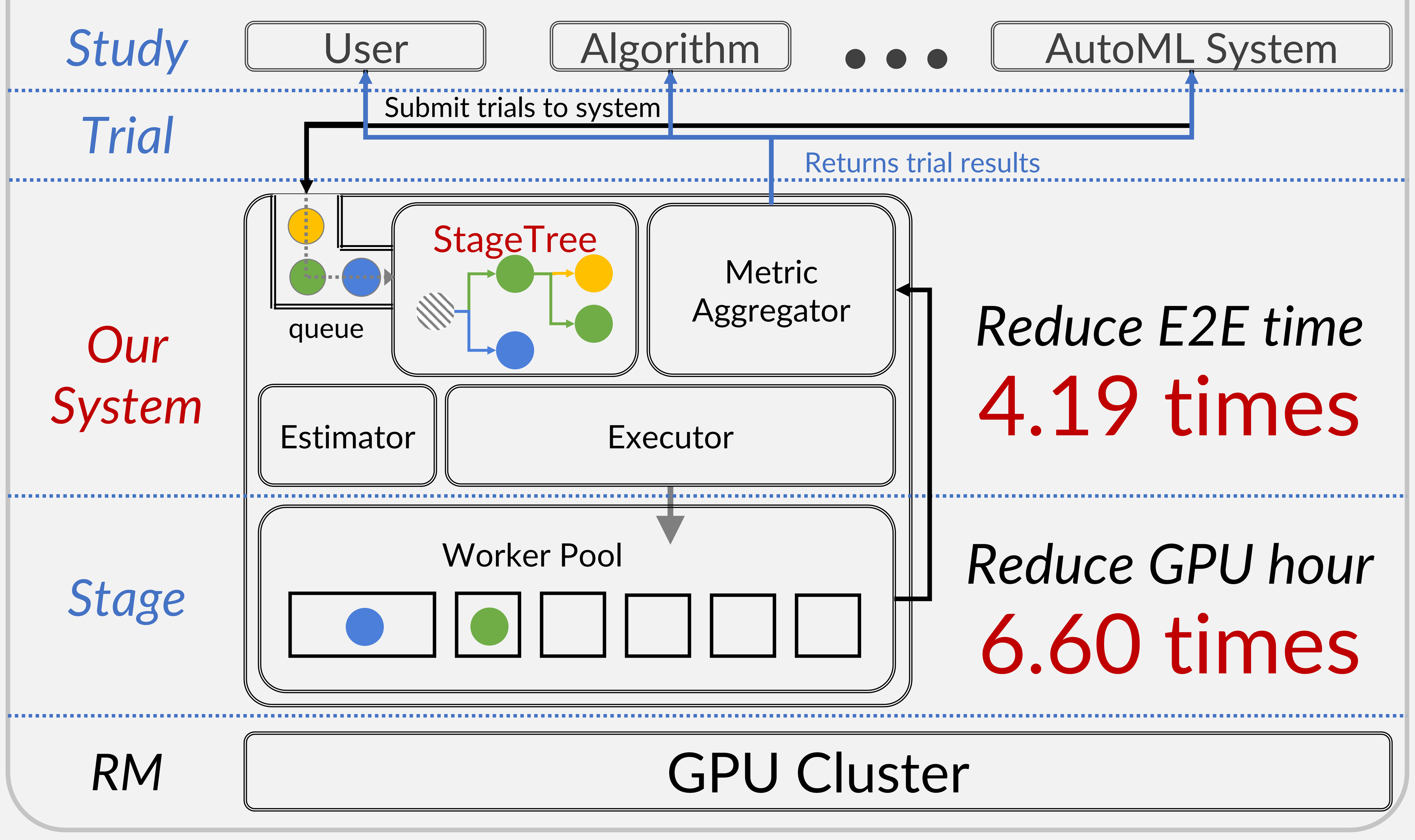
Stage-based Hyper-parameter Optimization for Deep Learning

Ahnjae Shin, Dong-Jin Shin, Sungwoo Cho, Do Yoon Kim, Eunji Jeong, Gyeong-In Yu, Byung-Gon Chun

Seoul National University *Software Platform Lab*

Hyper-parameter Tuning as a Job



Reduce E2E time
4.19 times

Reduce GPU hour
6.60 times

Hyper-parameters are Sequences

Observation: *State-of-the-art DL use Hyper-parameter as sequences*

{ *learning-rate, drop-out ratio, optimizer, momentum, batch size, image augmentation parameters, training image input size, input sequence length, network architecture parameters* }

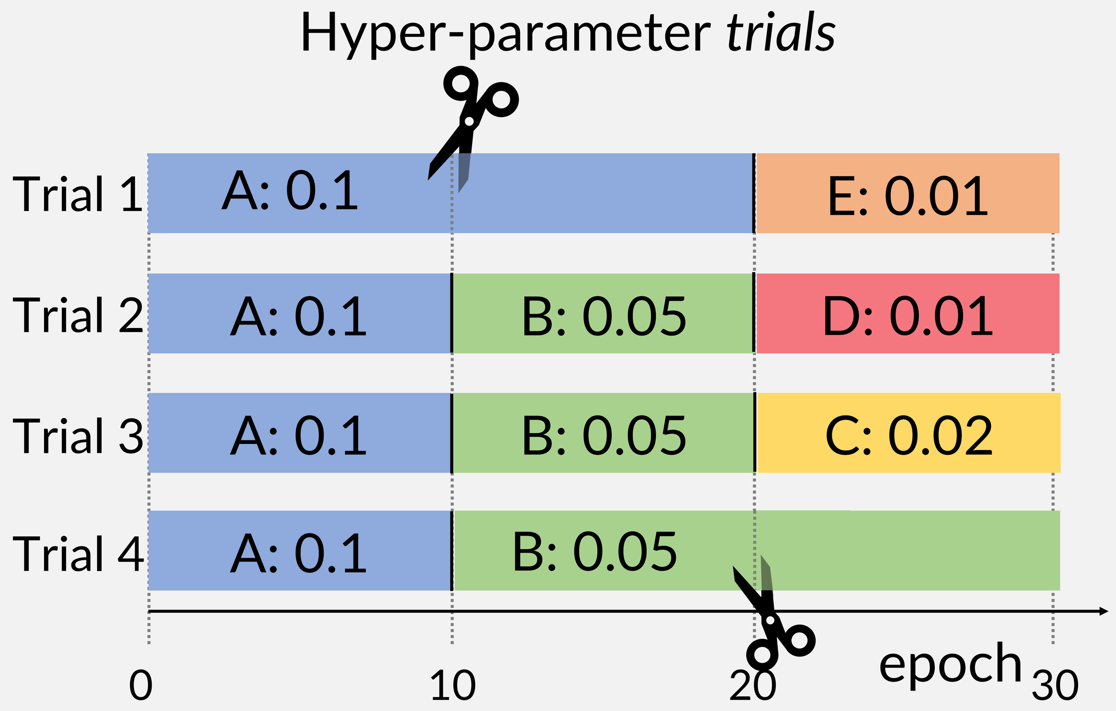
→ Each value has different behavior, use sequences for hybrid approach
→ Such hyper-parameter sequences are *parameterized*
E.g. Learning rate schedule functions

Idea: *Different sequences may have* ***same prefix***

- Most sequences are piece-wise constant
- Continuous-valued sequence (learning-rate) is also eligible for merging E.g. Warm-up, Cyclic learning rate
- When there is no mergeable configurations, still behave similarly to existing systems

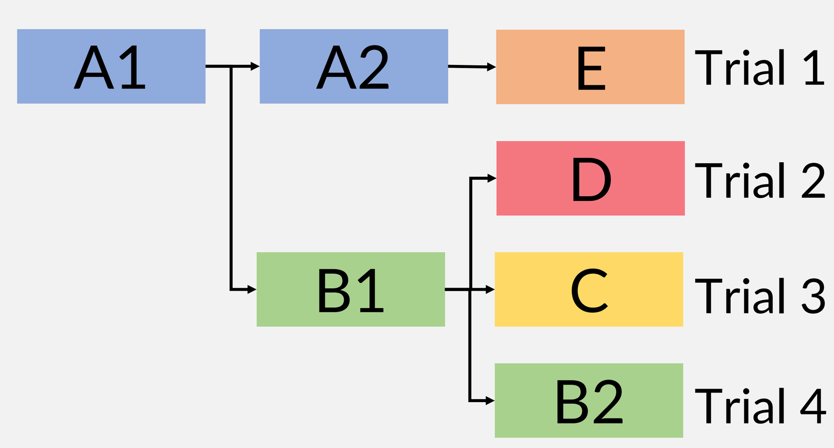
Stage-based Execution

Hyper-parameter *trials*



HP optimization job train multiple *trials*
Trials have common prefixes.
Common prefixes are merged to build a tree

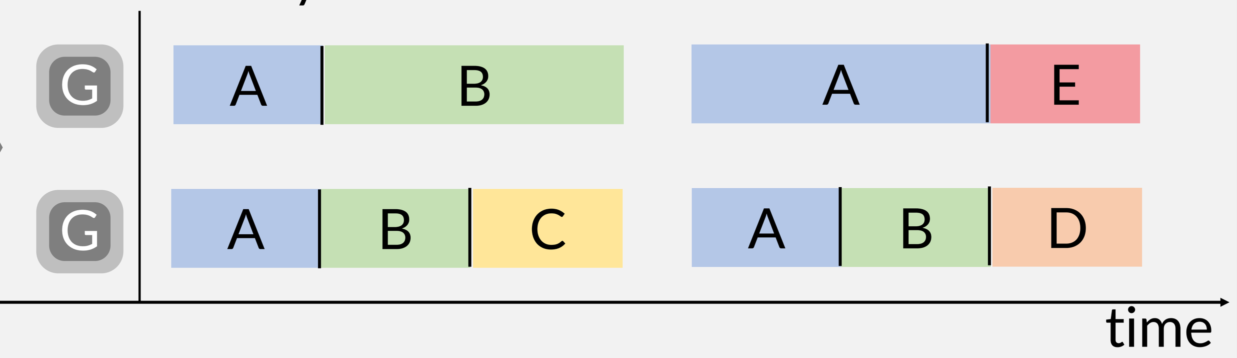
Same search space in *StageTree*



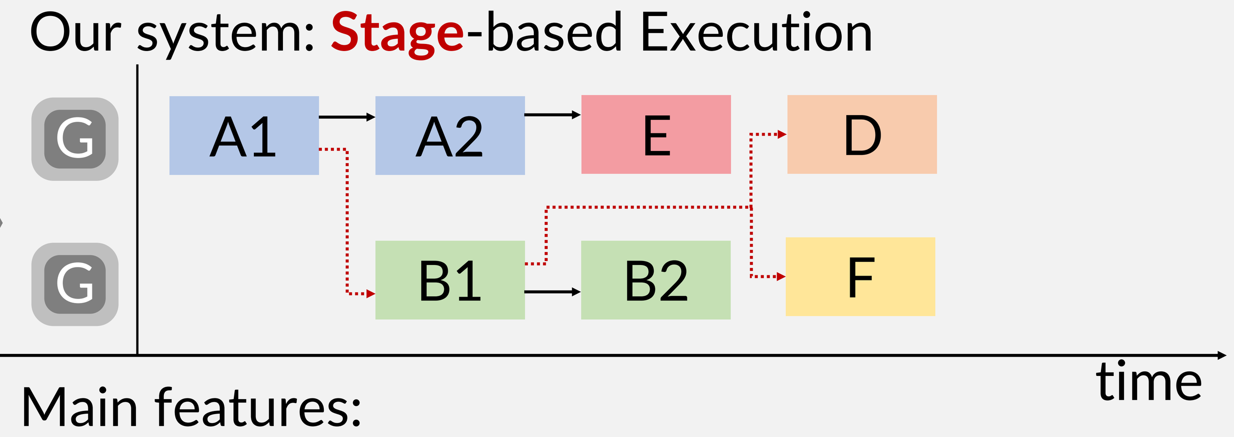
Merged stages can be trained only once
➔ Computation reuse happens

Each stage is **homogenous** from system perspective

Previous systems: **Trial**-based Execution



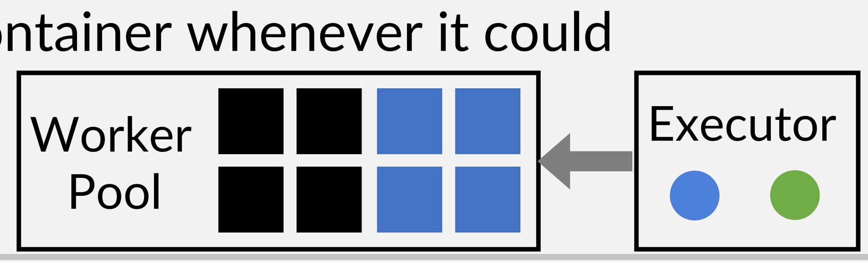
Our system: **Stage**-based Execution



Main features:
1. spilt & merge hyper-parameter sequences
2. automatically save & restore checkpoints

Reuse container

- Each task in hyper-parameter tuning job have same environment
- System maintains its worker pool, and reuse container whenever it could



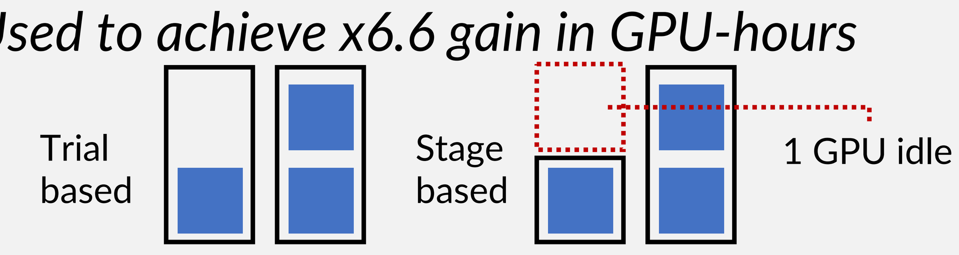
Reuse process

- Checkpoint overhead can be mitigated by continuing training in same process
- Stages are scheduled workers so that reusing can be maximized

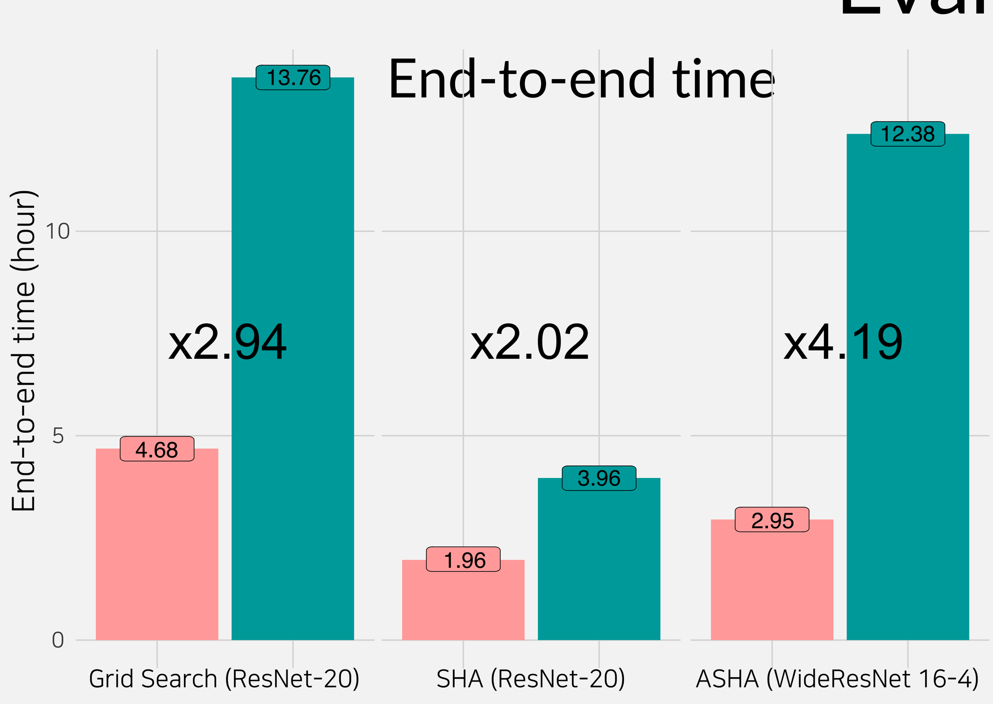
Bad: A1 → B1 Good: A1 → A2

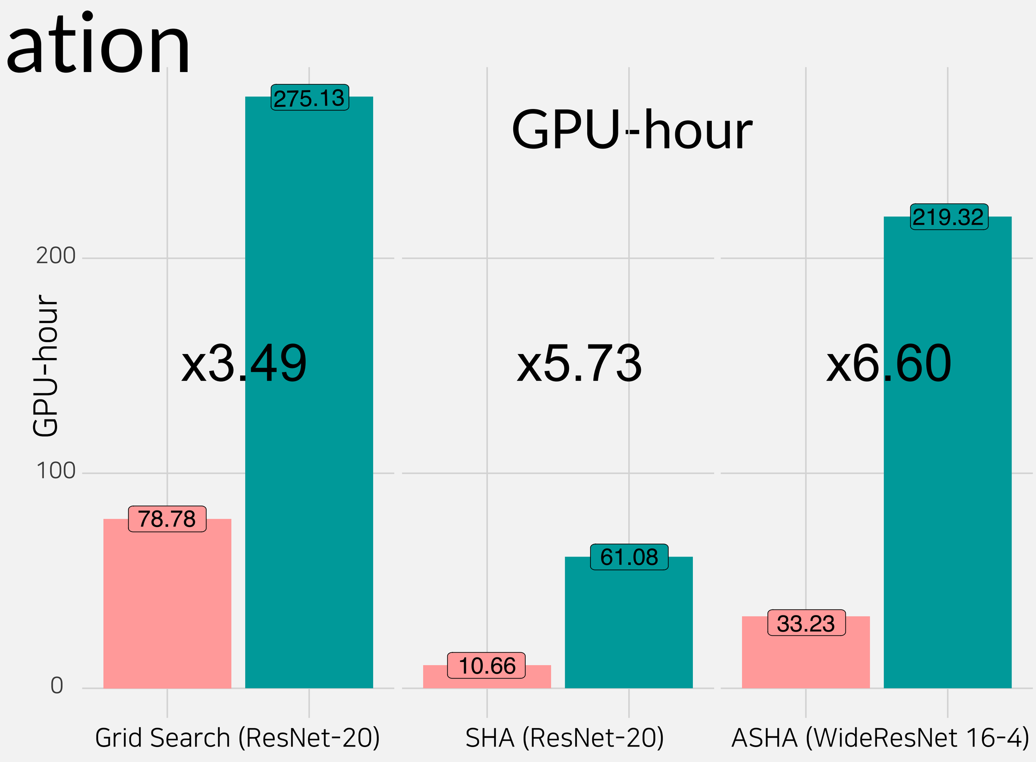
Just-fit resource allocation

- Stages have homogenous resource usage.
- When resource requirement changes, system adjusts allocated resource.
  → *Used to achieve x6.6 gain in GPU-hours*



Evaluation





- Python + gRPC + Docker Implementation
- 20 NVIDIA GeForce TITAN Xp GPU
- CIFAR-10 (40K/10K/10K split)
- Tune learning rate for Grid Search / SHA
- Tune batch size for ASHA

- Resnet-20: test error 8.24% with 4/5 of training data (original ResNet-20 report 8.75%) up to **5.73x** save of GPU-hour
- WideResNet 16-4: test error 5.2% (original 5.6%) up to **6.6x** save of GPU-hour

Future Work

1. Evaluate with other models / datasets using various hyper-parameters. Expect larger gains
   Continuous-valued sequences
   Data augmentation / Network architecture

2. Multi-study optimization
   Merging trials between multiple studies
   Cooperation between multiple studies
   Meta-learning between multiple studies

3. New hyper-parameter optimization algorithm
   Algorithm that can maximize use of StageTree
   Algorithm that exploit multi-study use case

Do you have any troubles in hyper-parameter tuning?
Let us know ☺