Jolteon: Unleashing the Promise of Serverless for Serverless Workflows

Zili Zhang, **Chao Jin**, Xin Jin

PEKING UNIVERSITY

Serverless computing

AWS Lambda | Azure Functions | Google Cloud Functions | Knative

Fine-grained resource elasticity

- Auto-scaling
- Concurrency from 1 to 1,000

Fine-grained billing

- 1 MB memory granularity
- 1 ms time granularity

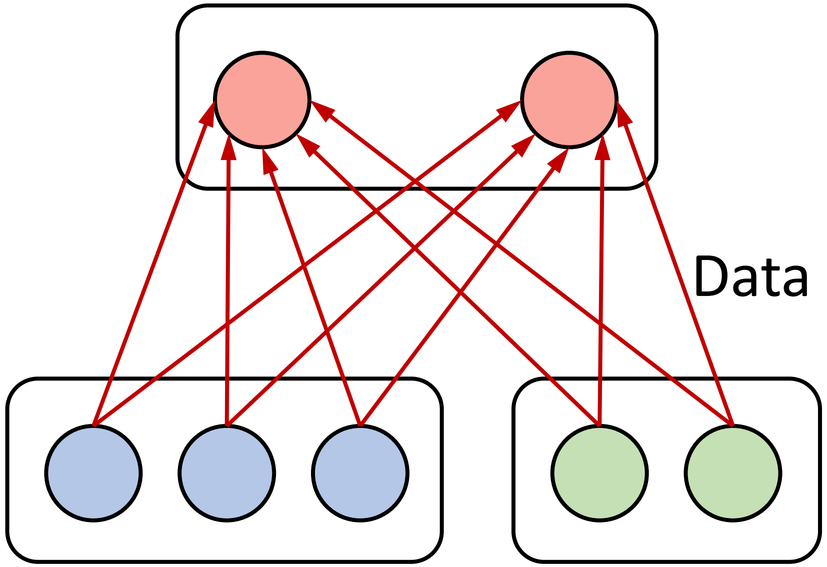
Serverless workflow

AWS Step Function

Azure Logic App

Google Cloud Workflow

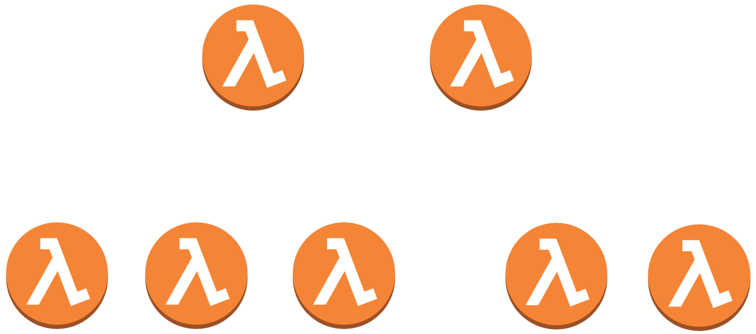
Job Execution DAG



Data parallelism

Deploy

Serverless functions



Workflow execution latency

Cost ($\sum_{\text{funcs}} \text{time} \times \text{memory}$)

Resource configuration: a new problem

Fine-grained resource elasticity

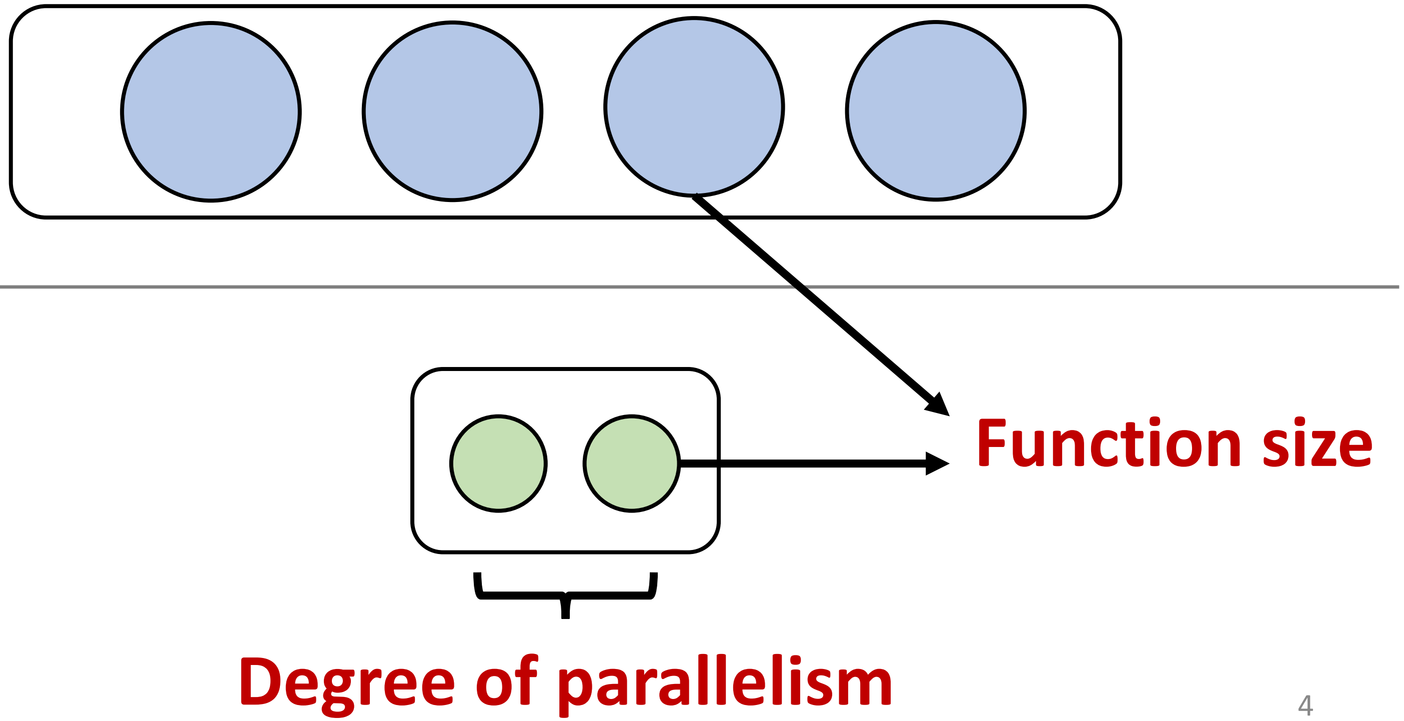
Can we decide the resource configuration automatically to satisfy application-level requirements for serverless workflows?

More resources

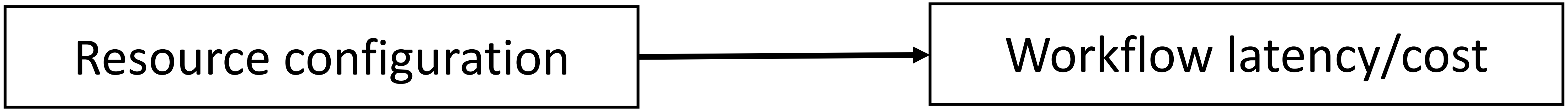
Faster, higher cost

Less resources

Slower, lower cost

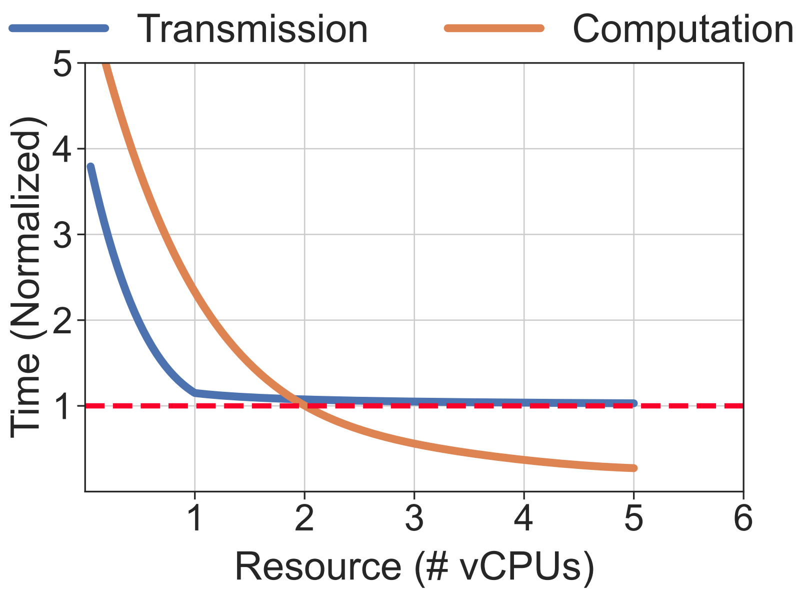


Performance model



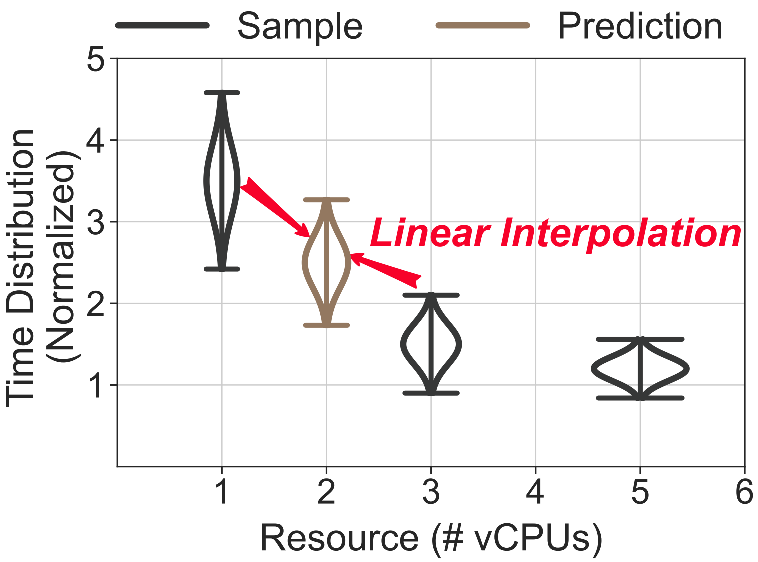
White-box model (Ditto, SIGCOMM'23)

Capture the characteristics step-by-step

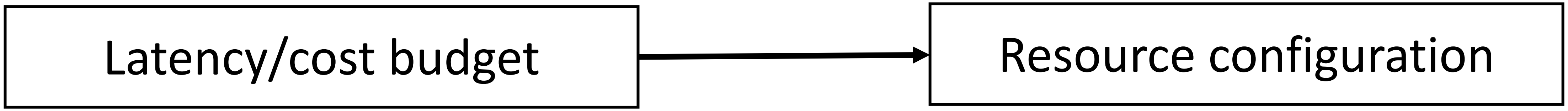


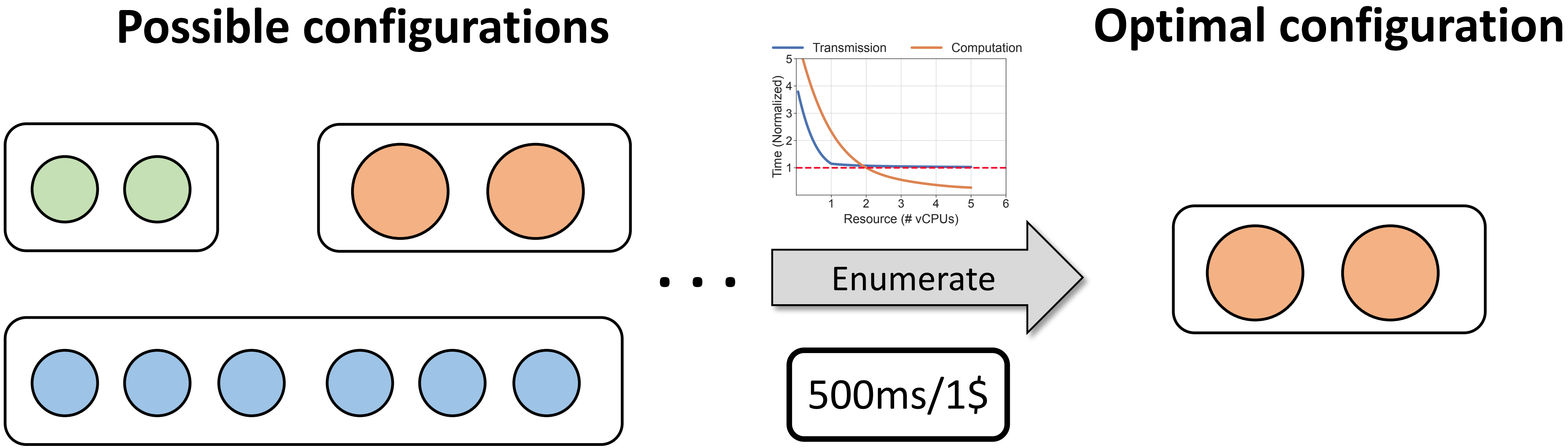
Black-box model (Orion, OSDI'22)

Capture the performance variability



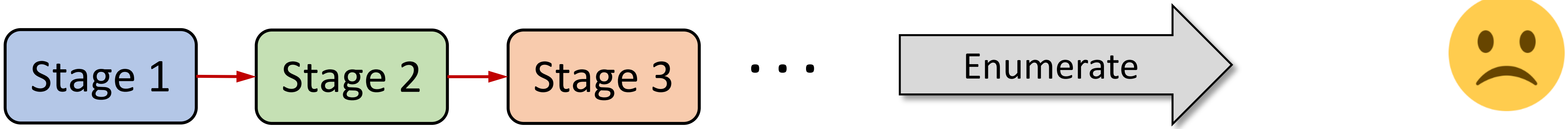
Solve the optimization problem

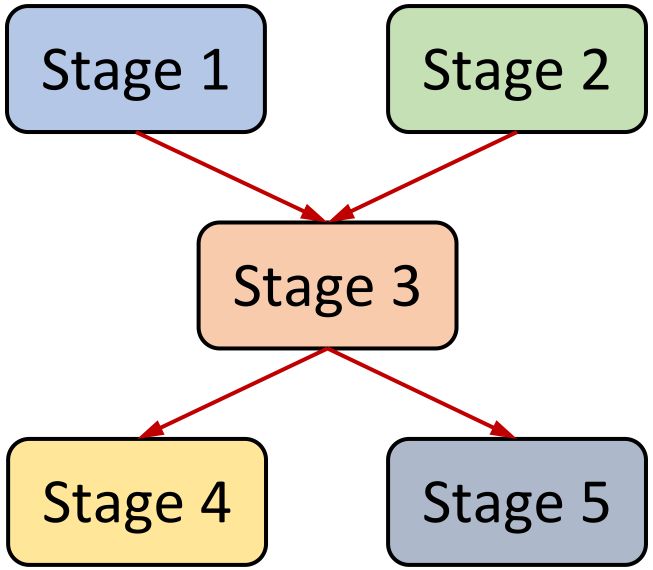




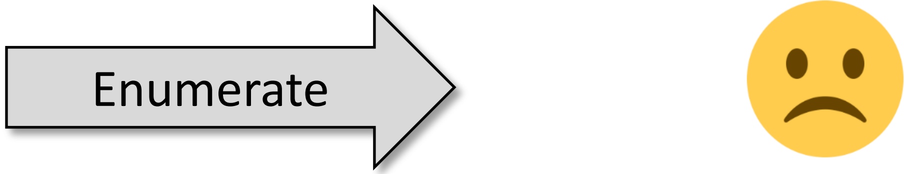
Solve the optimization problem

Large configuration space





Complex performance model



Jolteon design outline

Challenge 1: How to build the performance model?

- Analytical model → Fast and accurate prediction on average time
- Distribution-aware model → Guarantee performance bound

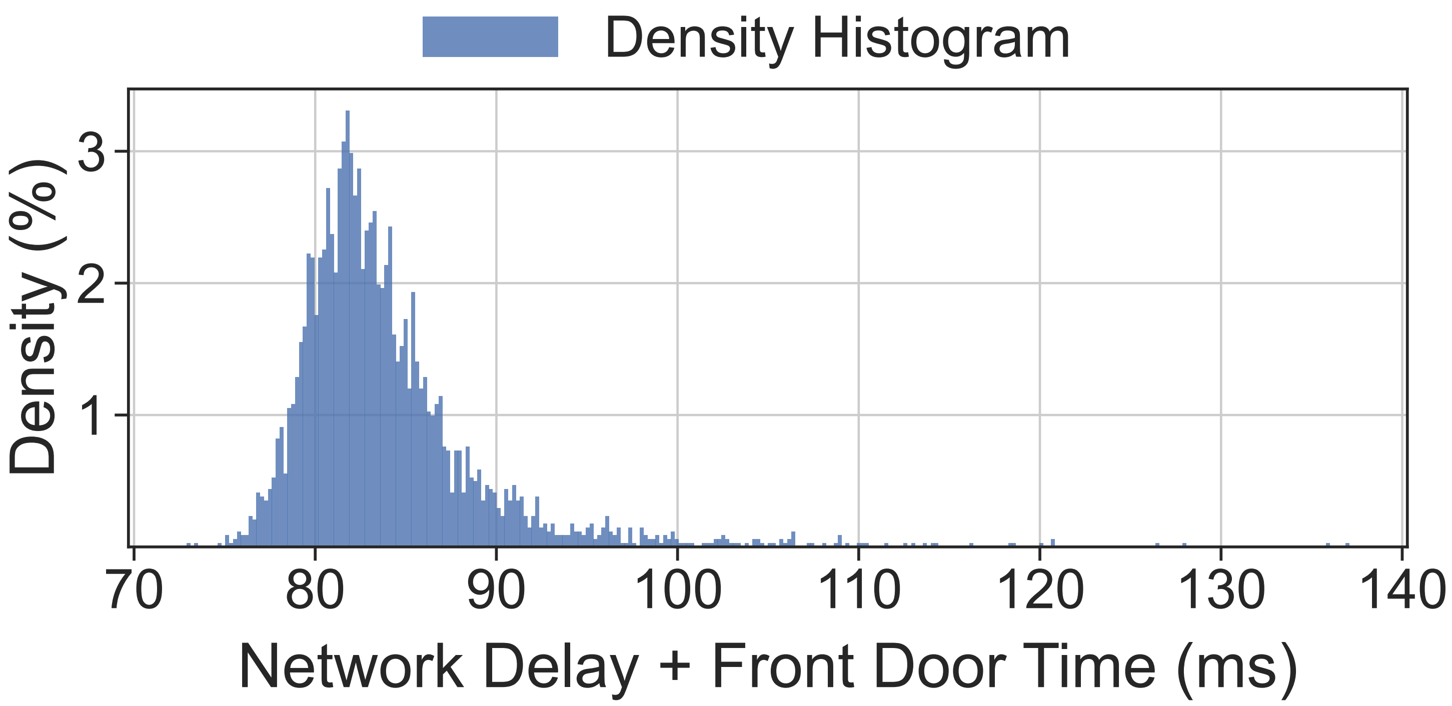
Challenge 2: How to optimize the optimization problem?

- Formulate the optimization
- Fast solve the problem with optimal result

Performance model: initialization

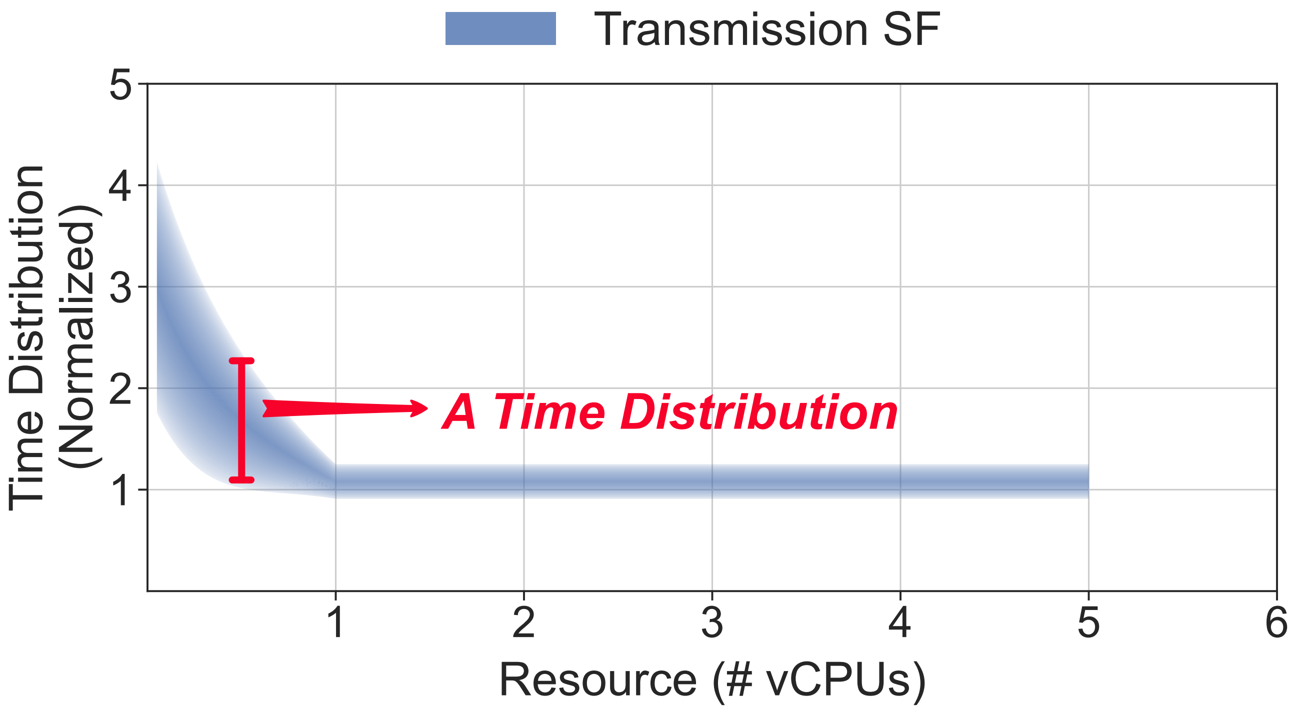
- Network delay
- Image transmission
- Front door execution
- Load container

$\rightarrow D + G$



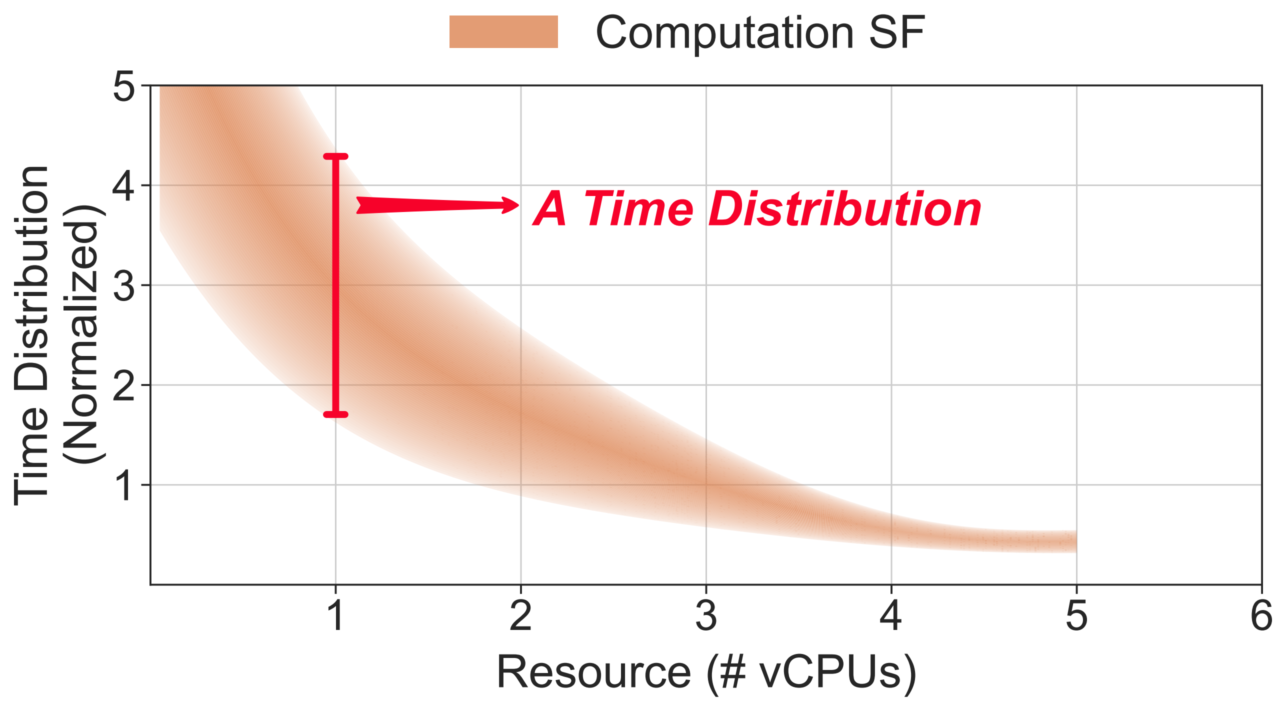
Performance model: transmission

$$T(d,v) = \frac{S}{d \times min(v \times W,\ B)} + O_T$$



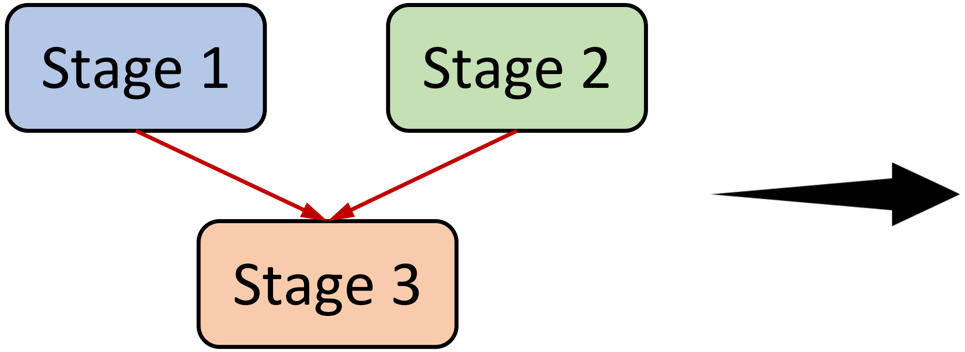
Performance model: computation

$$C(d,v) = \sum_{i=0}^{l} \left(A_i \times \left(\frac{S}{dv}\right)^i + \ln \frac{S}{dv} \times \left(\sum_{i=0}^{m} B_i \times \left(\frac{S}{dv}\right)^i\right)\right).$$



Performance model: workflow

Stochastic performance model: analytic formulas with random variables



$$Time = \max\{stage\ 1, stage\ 2\} + stage3$$

Problem Solver: problem formulation

- Objective: minimize cost
- Guarantee the latency bound ε with confidence level δ

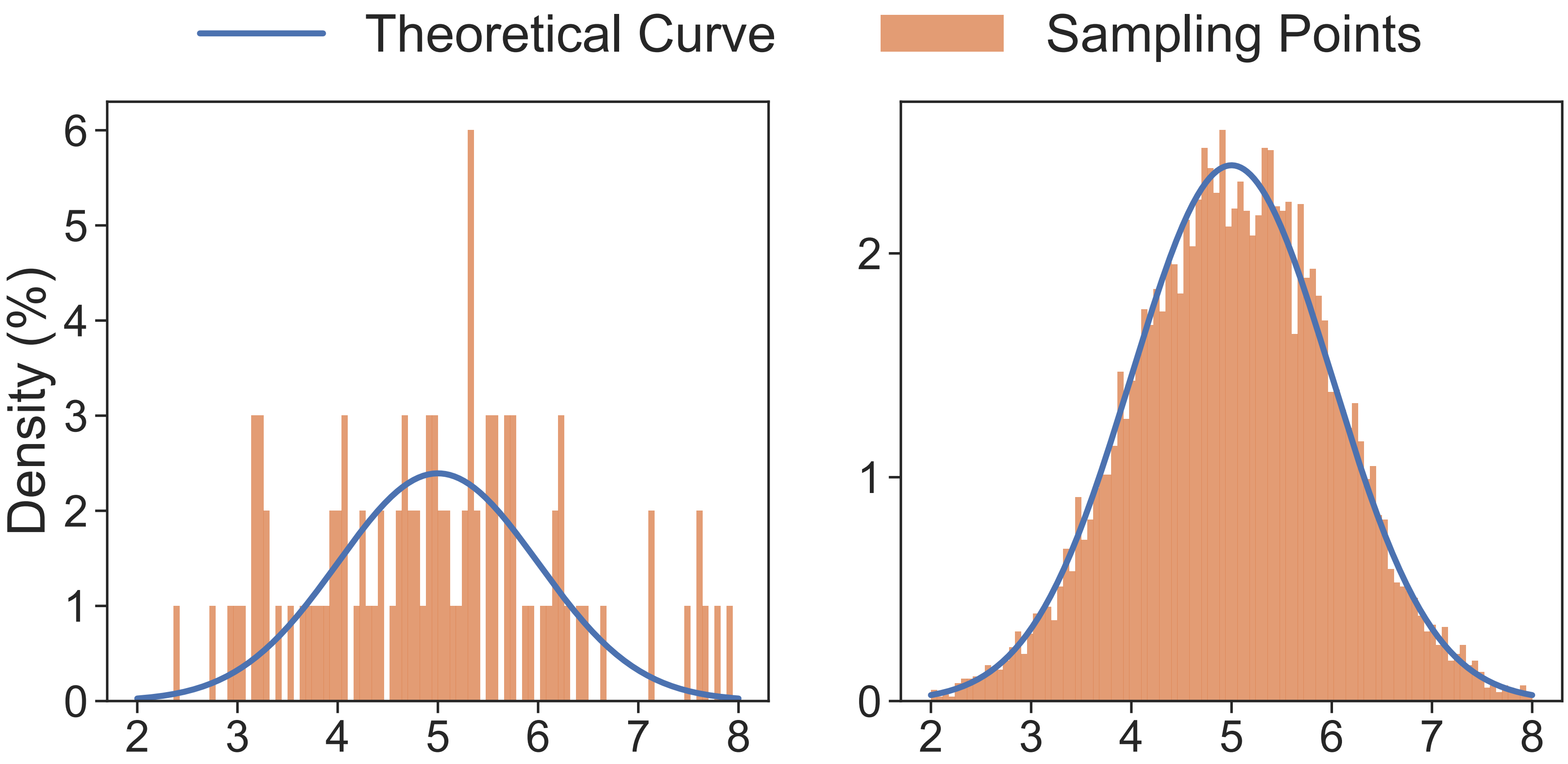
$$\begin{cases} Minimize \quad Cost(\boldsymbol{d}, \boldsymbol{v}) \\ St. \quad Confidence(Latency(\boldsymbol{d}, \boldsymbol{v}) < \varepsilon) \geq \delta \end{cases}$$

Problem Solver: bound guaranteed sampler

$$Confidence(Latency(\boldsymbol{d},\boldsymbol{v}) < \varepsilon) \geq \delta$$

$$\left\{\begin{array}{ll} Latency(\boldsymbol{d},\boldsymbol{v}) < \varepsilon & \textit{Sample 1} \\ Latency(\boldsymbol{d},\boldsymbol{v}) < \varepsilon & \textit{Sample 2} \\ Latency(\boldsymbol{d},\boldsymbol{v}) < \varepsilon & \textit{Sample 3} \\ \quad\cdots & \end{array}\right.$$

Problem Solver: bound guaranteed sampler



(a) sample size = 100

(b) sample size = 10000

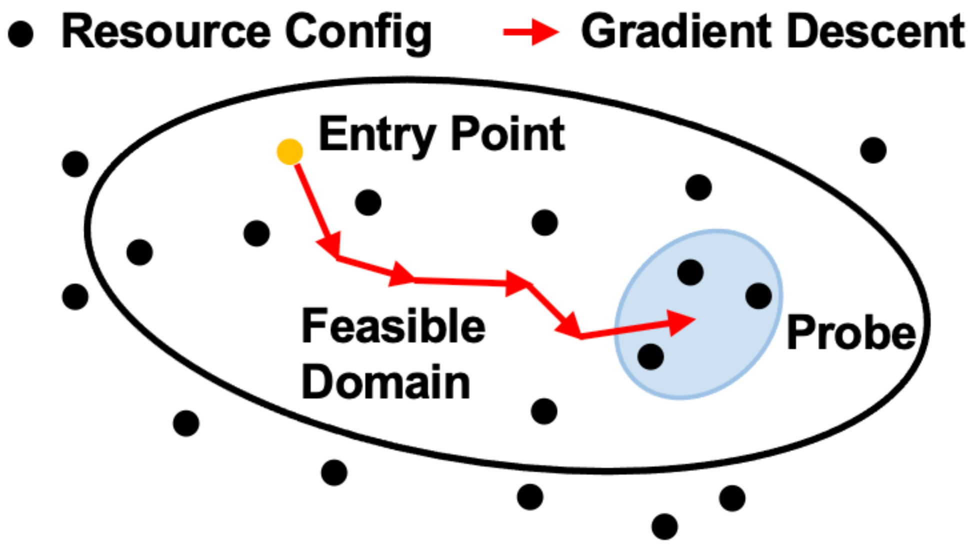
Problem Solver: bound guaranteed sampler

- The minimal sample size to guarantee the performance bound with confidence level δ

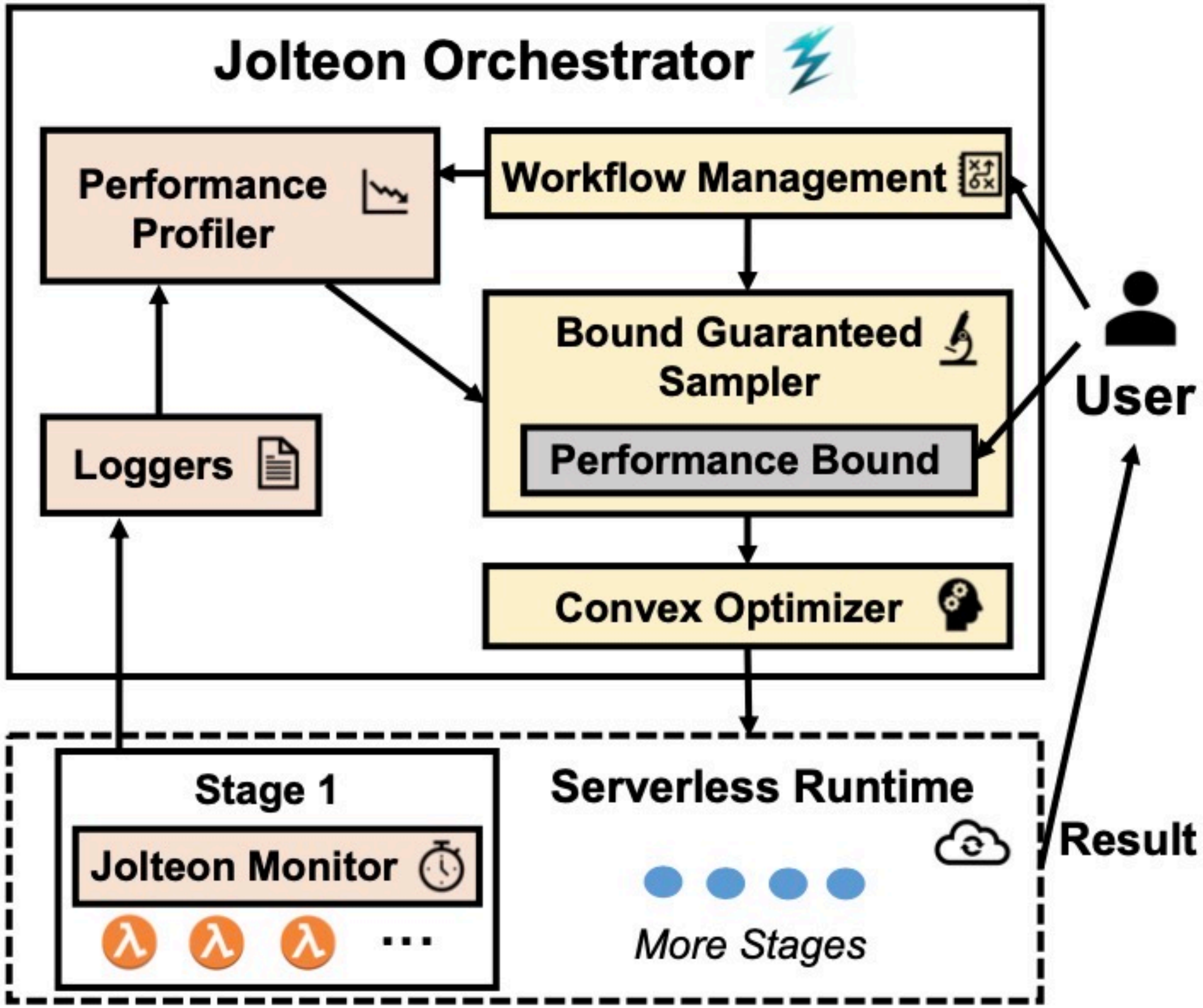
$$\frac{1}{2 \times (1 - percentile)^2} log(\frac{|D|}{1 - \delta})$$

Problem Solver: solving algorithm with convexity

- Gradient descent algorithm with convexity
- Probe to calibrate the result

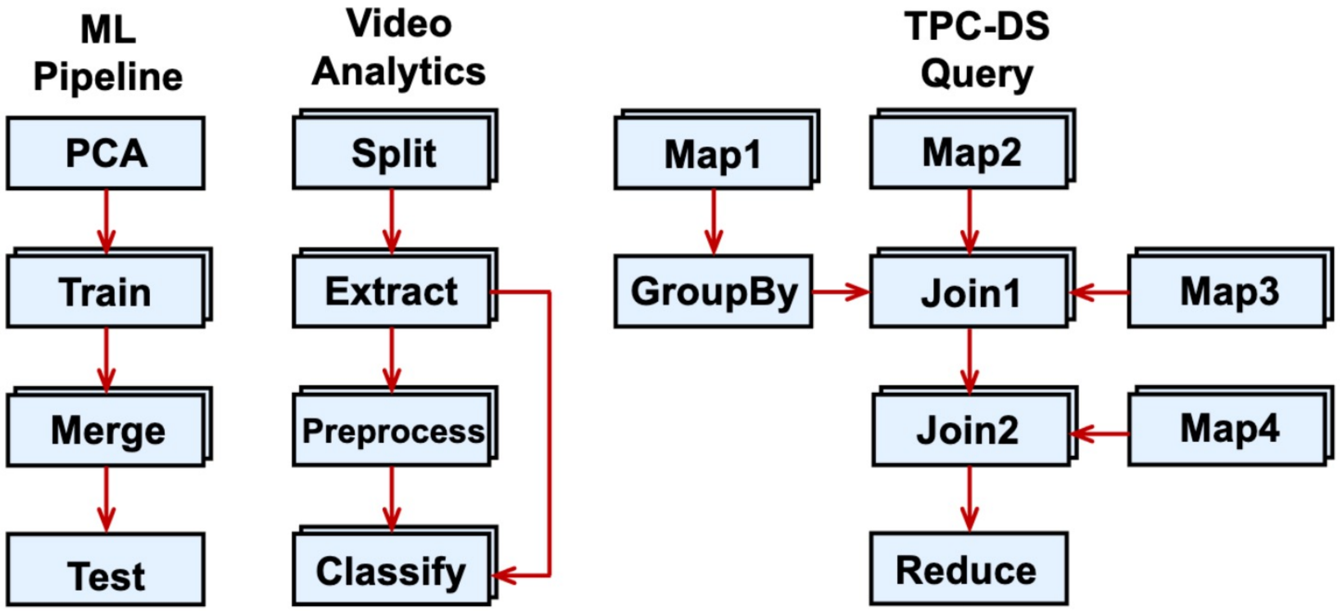


Jolteon system



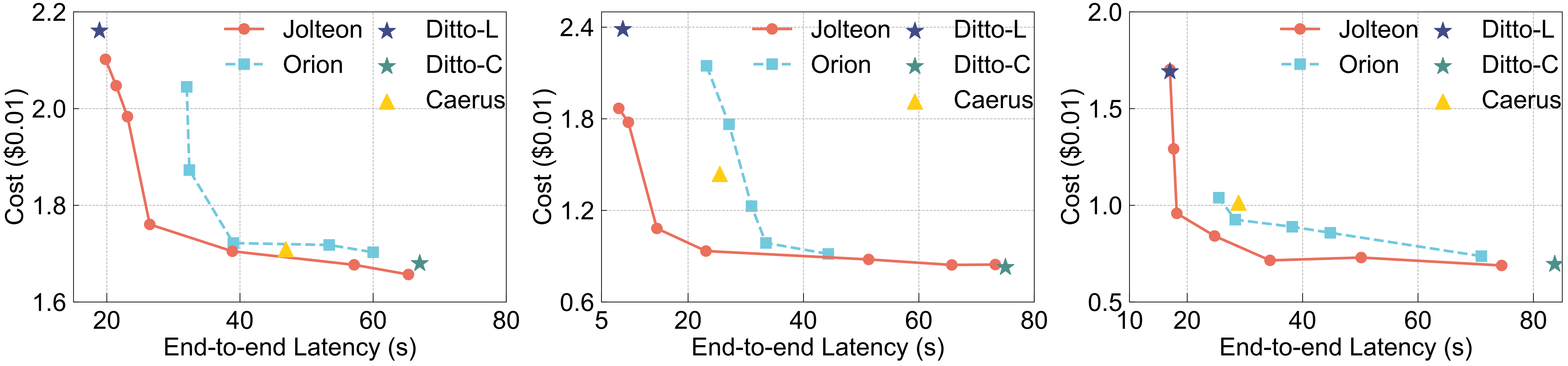
Evaluation

- Setup on AWS
  - Workflow orchestrator: one AWS c5.12xlarge EC2 server
  - Compute: AWS Lambda function
  - Storage: AWS S3



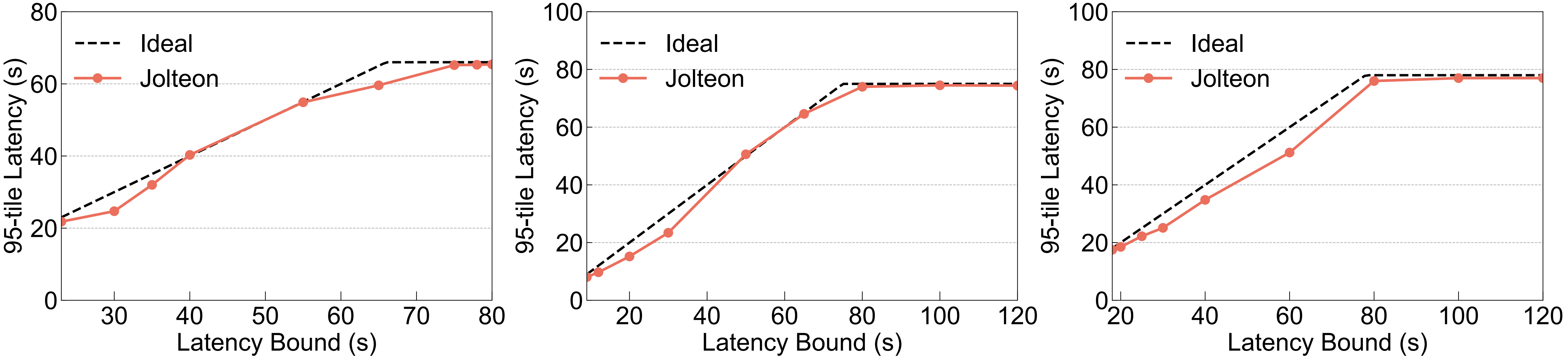
Evaluation

- Jolteon outperforms Orion by up to 2.3× on cost and 2.1× on latency
- Compared to Ditto, Jolteon reduce cost by 1.8× or latency by 3.3×, with a ≤11% reduction on the other metric.



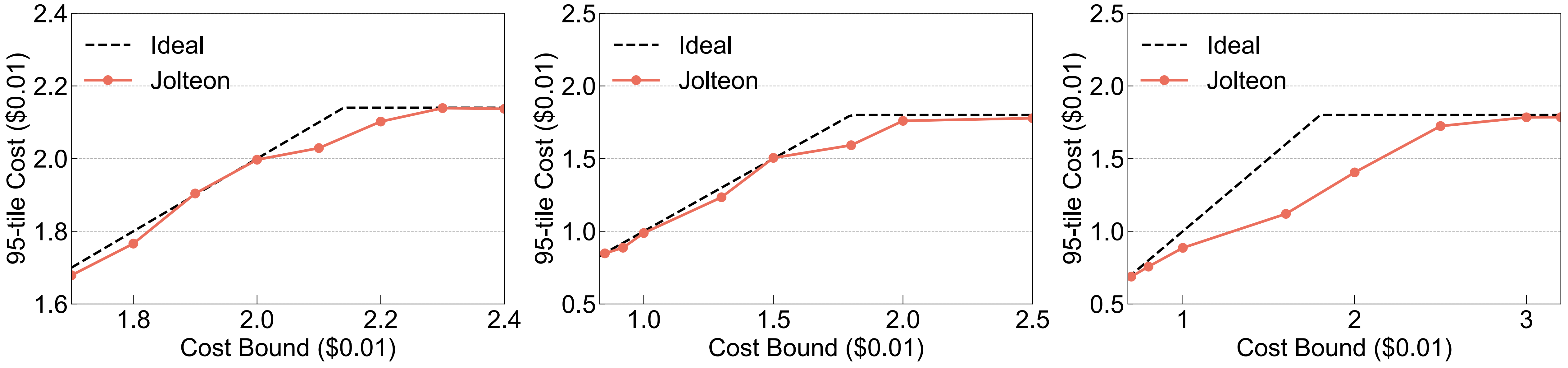
Evaluation

- Jolteon is able to guarantee the latency bound



Evaluation

- Jolteon is able to guarantee the cost bound



Evaluation

- Accuracy of the performance model
- Optimization problem solving time
- Performance model fit time
- Sensitivity of problem solver

Conclusion

- Serverless workflow orchestrator that provides automatic resource configuration to satisfy application-level requirements

- Jolteon uses stochastic performance model to form an optimization problem, which **minimize the cost** under a latency bound or **minimize the latency** under a cost bound.

- Jolteon outperforms Orion by up to 2.3× on cost and 2.1× on latency. Compared to Ditto, Jolteon reduce cost by 1.8× or latency by 3.3×, with a ≤11% reduction on the other metric.

Thank you!

chaojin@pku.edu.cn