ACM Symposium on Cloud Computing

16th ACM Symposium on Cloud Computing 2025
November 19–21, 2025

METIS: A Non-Clairvoyant, Workflow-Aware OS Scheduler for Serverless Applications

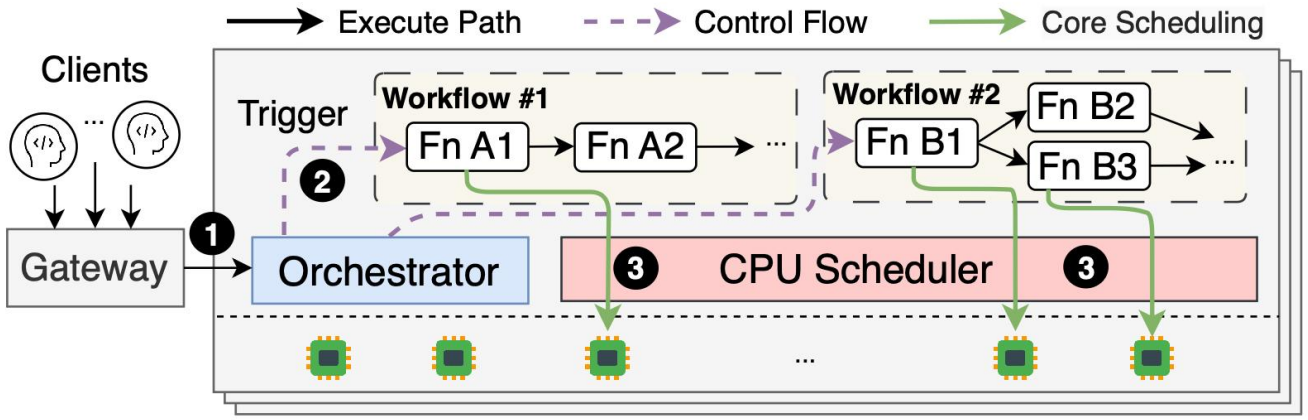
Wenda Tang, Yanan Yang and Jie Wu

China Telecom Cloud Computing Research Institute, Beijing, China

中国电信 CHINA TELECOM

Contents

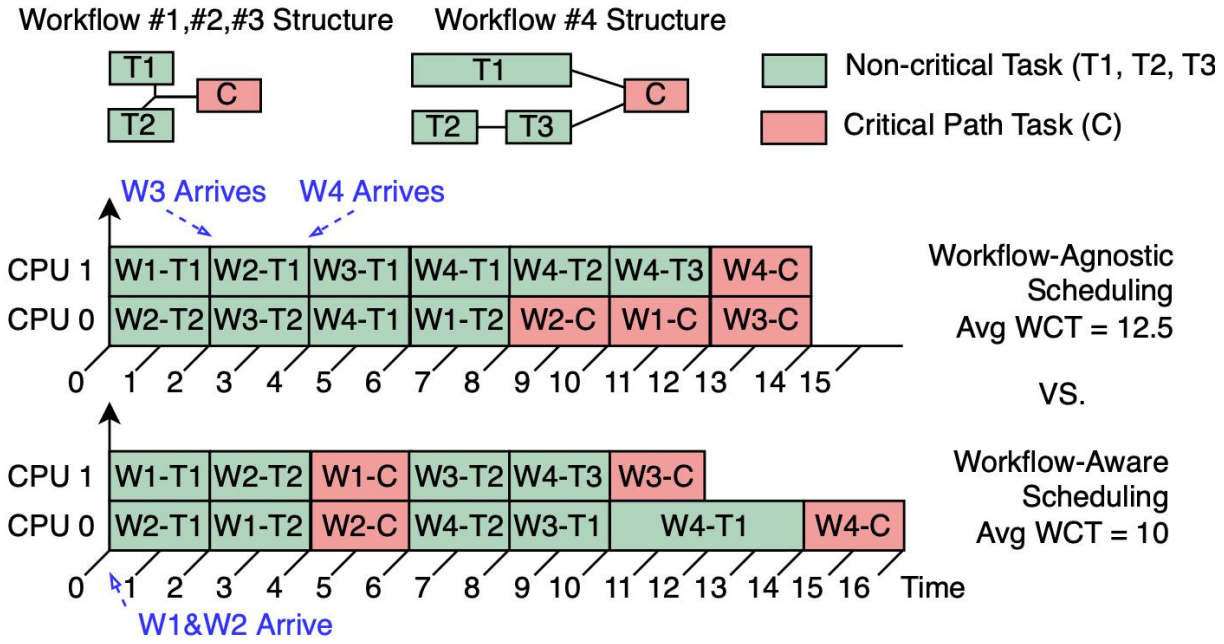
Serverless Workflow Execution Model



- **Host OS Scheduler**
  - Each function runs as an independent task (container/microVM) enqueued in the OS runqueue.
  - Linux CFS (typical) allocates CPU without workflow context.

- **Gateway**: Handles auth, validation, routing; forwards triggers to the workflow orchestrator.
- **Orchestrator**: Manages a DAG: nodes = functions, edges = data/control dependencies.
  - Dispatches a function only when dependencies are satisfied; exploits parallel branches.

Workflow-OS Scheduling Mismatch (Case Study)



Comparison of workflow-agnostic (up) and workflow-aware (down) scheduling for 4 concurrent workflows on a dual-core system.

- Workflow-agnostic scheduling:
  - Schedules functions independently, without workflow context
  - **Intra-workflow blocking:** Critical-path tasks can be delayed by non-critical tasks **within the same workflow**
  - **Inter-workflow blocking:** Critical tasks may also be blocked by functions from **other workflows** competing for CPU
  - Increased workflow completion time (WCT)
- Workflow-aware scheduling:
  - Prioritizes critical-path tasks within workflows
  - Eliminates unnecessary blocking & reduces overall workflow latency

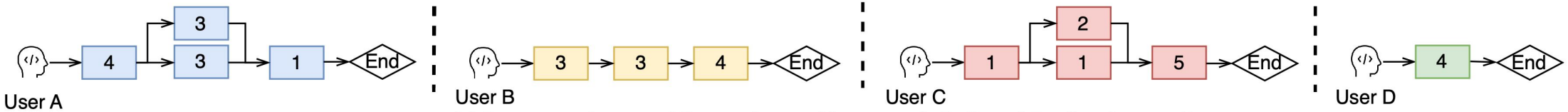
Related Work

- Existing research focuses on:
  - Cold start latency reduction [SONIC(ATC'21), AlloyStack(EuroSys'25)...]
  - Elastic resource scaling [Jiffy(EuroSys'22), Faascale(SoCC'24)...]
  - I/O optimization [SAND(ATC'18), SPRIGHT(SIGCOMM'22), RMMap(EuroSys'24)...]
- Function-centric scheduling:
  - Each function is treated independently [SFS(SC'22), ALPS(ATC'24)...]
  - Workflow-level dependencies and objectives are overlooked
- Gap:
  - Lack of workflow-aware scheduling for end-to-end performance

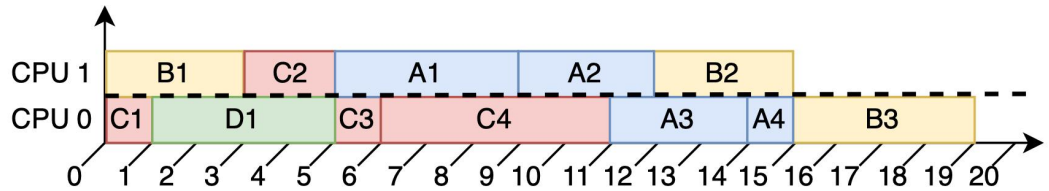
Problem Definition & Goals

- Problem:
  - OS schedulers lack workflow awareness in serverless environments
  - Critical-path tasks are often blocked, increasing workflow completion time (WCT)
  - Existing solutions require prior knowledge of workflow structures or task durations
- Goals:
  - Enable OS schedulers to optimize workflows as a whole, not just individual functions
  - Minimize end-to-end workflow latency
  - Achieve non-clairvoyant scheduling—no need for prior workflow knowledge

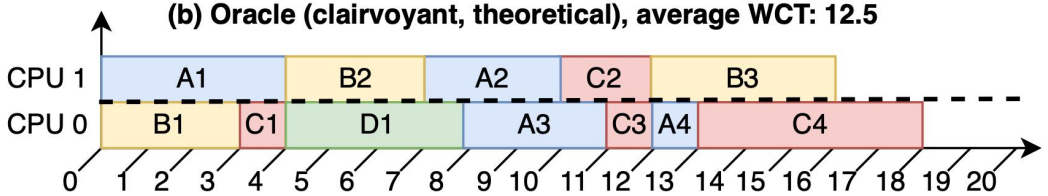
Key Insight: Workflow-Aware Least-Attained Service (WLAS)



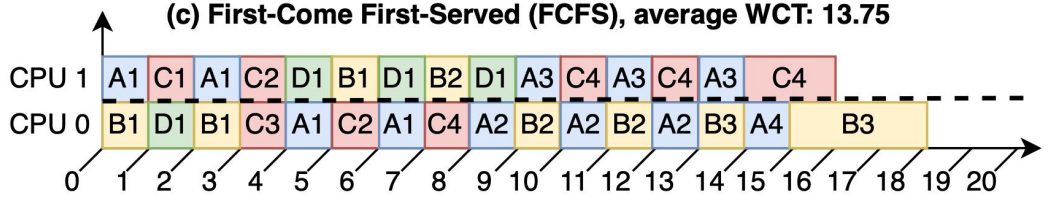
(a) Four serverless workflows, triggered by users A, B, C, and D, all arrive at t=0



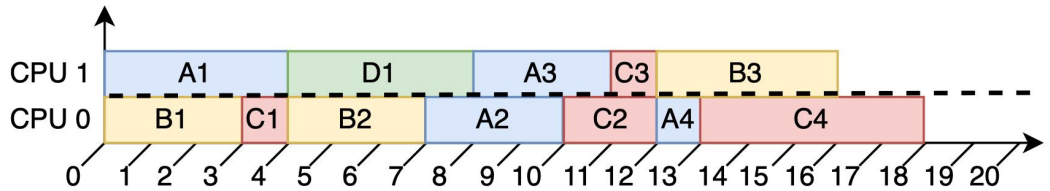
(b) Oracle (clairvoyant, theoretical), average WCT: 12.5



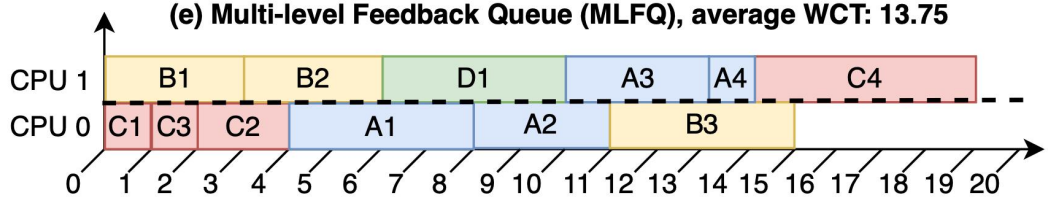
(c) First-Come First-Served (FCFS), average WCT: 13.75



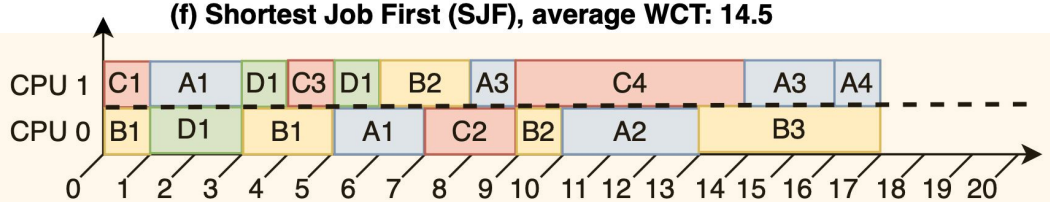
(d) Processor Sharing (PS), average WCT: 14.5



(e) Multi-level Feedback Queue (MLFQ), average WCT: 13.75



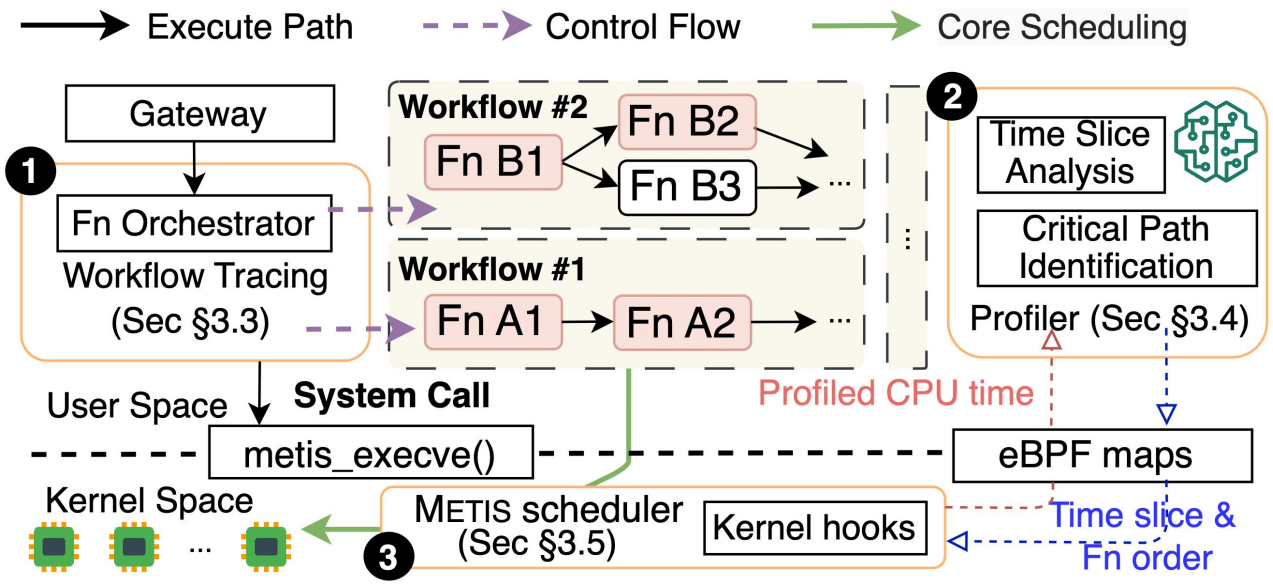
(f) Shortest Job First (SJF), average WCT: 14.5



(g) Workflow-Level Attained Service (WLAS), average WCT: 13.5

METIS Design

Overview of *Metis*

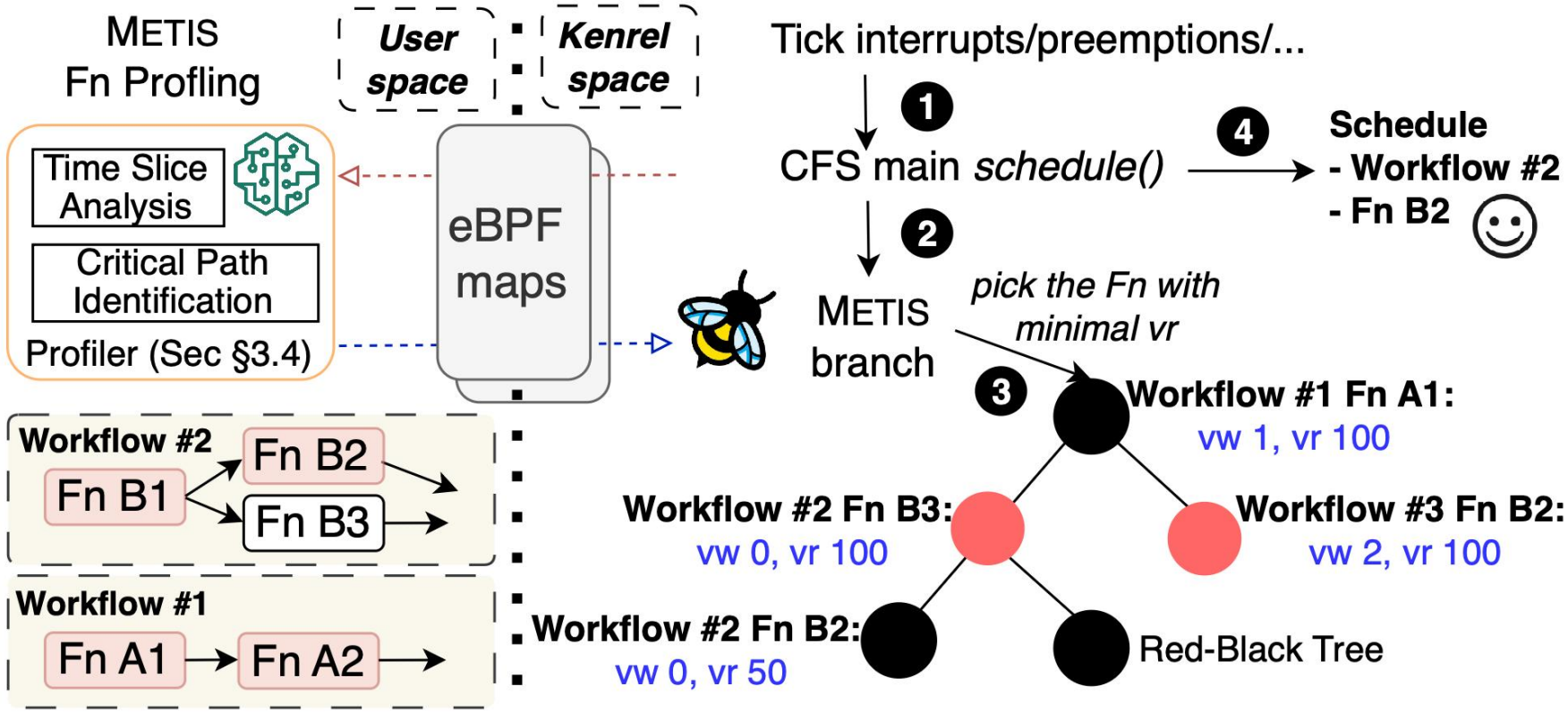


- Three main modules:
  - Workflow tracking—— Identity structure
  - Metric collection—— **critical path estimation**
  - Scheduling decision——ebpf scheduling hints

Key Features of *Metis*

- Workflow-aware scheduling:
  - Dynamically adapts to workflow structures and dependencies
- Non-clairvoyant design:
  - No need for prior knowledge of workflow structure or task durations
- Low-overhead deployment:
  - Uses eBPF for lightweight, minimally invasive integration with the OS
- Compatibility:
  - Works with mainstream operating systems and serverless platforms

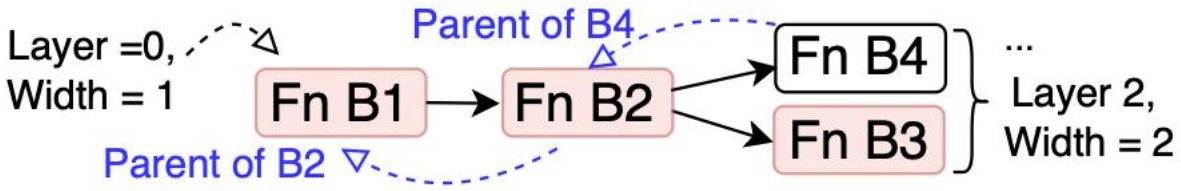
Self-Clocked Workflow-Aware Scheduling



Function scheduling lifecycle in Metis based on workflow-aware prioritization.

Critical Path Estimation & Function Ordering.

- **Critical Path:**
  - The longest path through the workflow's Directed Acyclic Graph (DAG).
  - Determines the minimum possible completion time for the workflow.
- **Estimation Method:**
  - Traverse the DAG to identify the path with the maximum cumulative execution time.



$$CP(w_i) = \max_{\text{path } P \in G_i} \sum_{f \in P} t_f$$

function f's execution time,

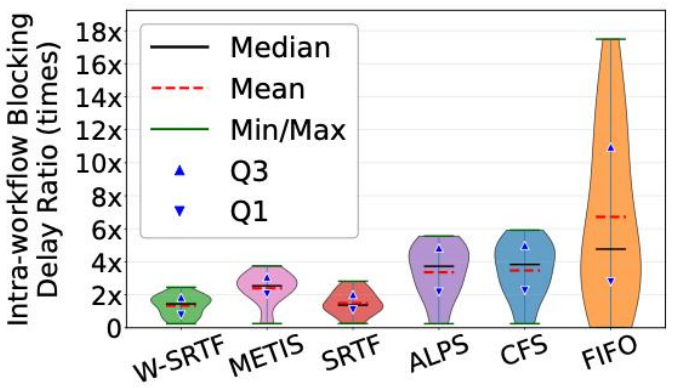
$$p(f_j) = \max_{f_k \in \mathcal{P}(f_j)} \{p(f_k) + t_k\} + \gamma \cdot \text{Width}(L_j)$$

set of parent nodes of f_j

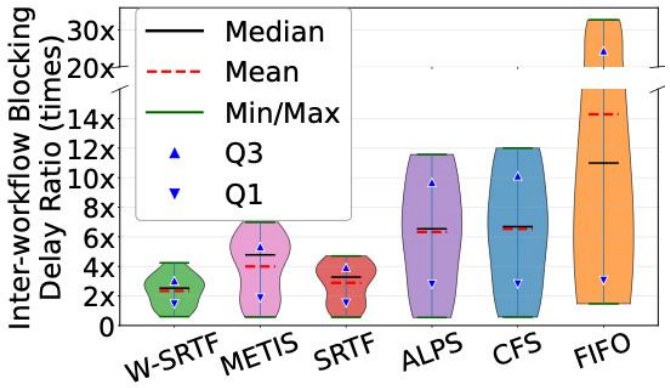
Simulation Results

- Baselines:
  - Compared against Linux CFS, FIFO, and ALPS[ATC'24] schedulers
- Evaluation methodology:
  - Trace-driven simulations with diverse workflow patterns and scales
- Key results:
  - Metis reduces average workflow completion time (WCT) by 31.3%
  - 95th percentile latency reduced by 14.3%
  - Significant improvement in tail latency (99th percentile).

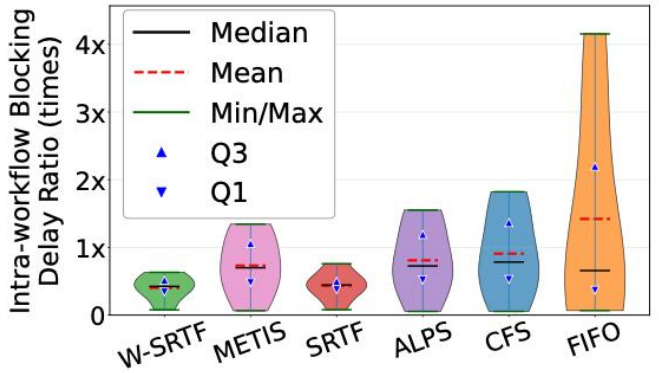
Synthetic Trace-driven Analysis



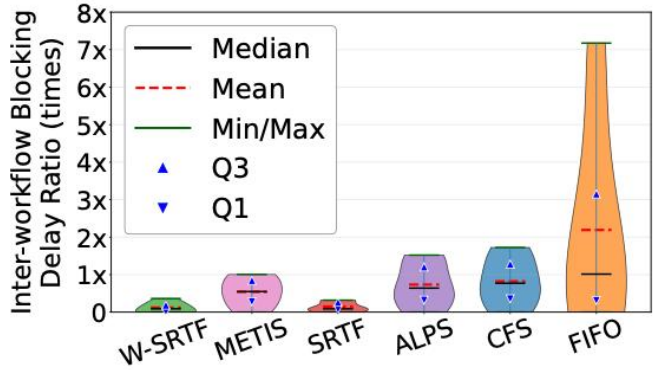
Intra-workflow@Medium



Inter-workflow@Medium



Intra-workflow@Small



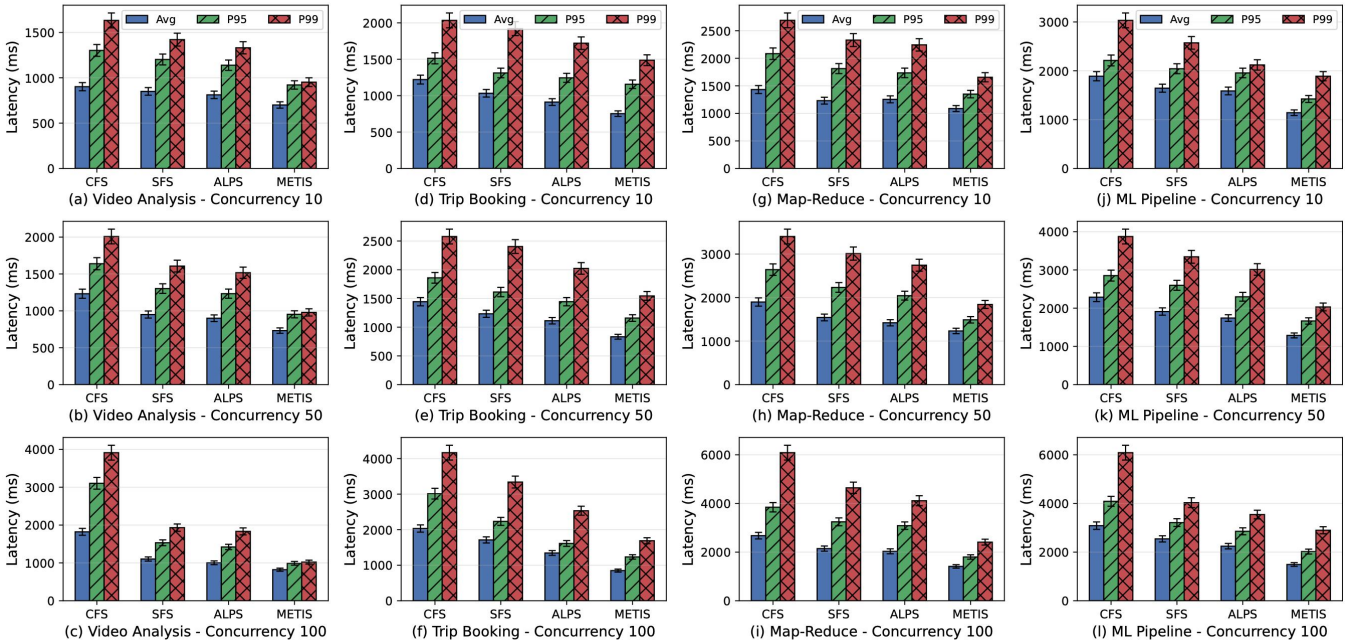
Inter-workflow@Small

End-to-End Workload Performance Comparison

- Baselines:
  - Compared against Linux CFS, SFS, and ALPS [ATC'24] schedulers
- Experimental Setup:
  - Four representative workflows (Video Analysis, Trip Booking, MapReduce, ML Pipeline)
  - Tested at 10, 50, 100 concurrency levels
- Results:
  - Metis reduces average WCT by **31.3%** vs. ALPS
  - 95th percentile latency improved by **14.3**
  - 99th percentile latency reduced by up to **73.9%**
  - Over **80%** of workflows complete within **10**

Key characteristics of selected SeBS-Flow benchmarks [EuroSys'25]

Workload	# Func.	Max Par.	DAG Depth	Complexity
Video Analysis	21	16	3	★★
MapReduce	13	10	4	★★★
Trip Booking	8	4	7	★★★★
ML Pipeline	11	8	2	★



Conclusions

- Workflow-centric serverless applications require both fair and efficient OS-level scheduling.
- Metis introduces a non-clairvoyant, workflow-aware scheduler leveraging the novel WLAS algorithm to optimize latency and fairness.
- Comprehensive evaluation demonstrates that Metis consistently surpasses state-of-the-art schedulers.
- Metis achieves up to 58.2% lower workflow completion time and 73.9% better tail latency across diverse workloads.

Thank you!

Any questions?

Email: tangwd1@chinatelecom.cn