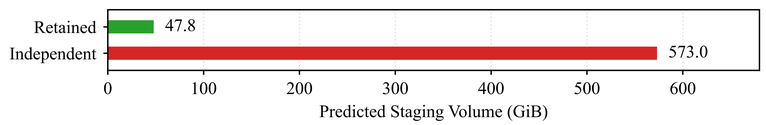
StageOnce: Speculative Cross-Job Data Staging for LLM Agent Fleets

Izzet Yildirim Xian-He Sun Anthony Kougkas
iyildirim@hawk.illinoistech.edu, sun@illinoistech.edu, akougkas@illinoistech.edu
Illinois Institute of Technology, Chicago, IL, USA

Gnosis Research Center @ ILLINOIS TECH

Fleets stage the same bytes many times over

Independent LLM jobs repeatedly stage the *same* immutable dataset, competing for bandwidth. Traces cover 72 sessions (6 workloads, 4 models):



12.0× predicted amplification, **91.7%** of bytes avoidable. Genomics tracing: 33% precision by file, **99.9% by byte**.

Crucially, the working set is **not declared at submit time**. It emerges from the model's *thinking phase*, before it issues any I/O, so the overlap is reachable only *mid-run*.

Speculative staging needs coordination across jobs

- **AgentStage** [4], but private to one session
- **Burst buffers** [1], but organized by job rather than agent intent
- **Data diffusion** [2], but reuse keyed to known inputs
- **Speculative agent execution** [3], but within one agent

The gap is **coordinating speculative dataset hints across independent jobs** while preserving version and output isolation.

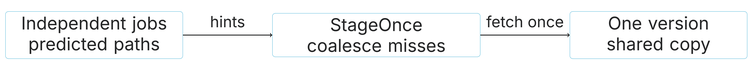
Hints arrive speculatively and out of order

Concurrent, out-of-order predictions require three guarantees:

1. deduplicate **concurrent** predictions
2. preserve private outputs and version boundaries
3. keep coordination RPCs **off** the read path

Fetch once, publish atomically, no per-read RPC

AgentStage [4] predicts paths while a model thinks. Independent jobs submit these hints to one **node-local daemon** before tool I/O begins.



Misses for `(project, dataset, version)` coalesce. The daemon fetches each file once and publishes it atomically. Tools read the shared copy directly, with **no per-read RPC**. Outputs stay private.

Measured: **1.03× bytes** for one staged copy and **0 source bytes** re-read by later jobs.

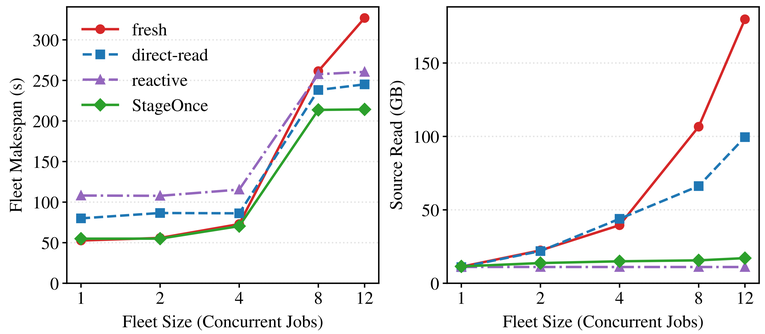
Two controls carry the argument

Conditions	fresh, direct-read, reactive, prestage, StageOnce
Fleet	$N = 1, 2, 4, 8, 12$, synchronized, 2–3 reps
Backends	Ares: SSD ∼518 MB/s, HDD ∼85 MB/s. NVMe cache
Domains	genomics 10.7 GB, astronomy 11.3 GB
Estimator	paired within repetition

reactive uses the *same* shared one-copy cache and coalescing, but stages at point-of-use behind a blocking wait. **prestage** bulk-copies the dataset once in a job prologue.

Recorded sessions replace **live LLMs**. We pair sessions within each repetition, redraw them between repetitions, and compare paired makespan ratios.

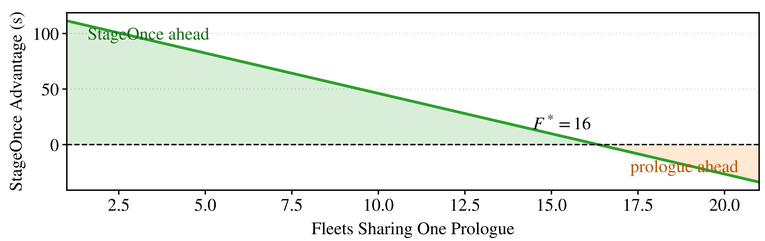
Makespan levels off at larger fleets



Against independent staging at $N = 12$, across *both* domains and *both* tiers: **−31.3%** and **−46.9%** (genomics, SSD/HDD), **−46.5%** and **−50.7%** (astronomy, SSD/HDD), sign-consistent in *every* repetition. Measured source reads at $N = 12$ (genomics/SSD): **17 GB vs. 180 GB**, a **10.5×** reduction.

Prologue setup must be amortised across fleets

Prologue setup is measured separately from fleet makespan. Who wins depends on how many fleets share that setup cost.



At $N = 12$, F^* grows with the dataset-to-working-set ratio: **13–21** fleets at 26.9 GB over a 10.7 GB working set, **2–4** when the working set *is* the dataset.

Speculation and sharing are separable



A shared cache alone does not explain the result. Two controls isolate speculation and sharing. Drop **speculative** and you get *reactive*: the *identical* shared cache, staged at point-of-use behind a blocking wait. Drop **cross-job** and you get *fresh*: speculative, but private per job.

Across four workload/tier groups, speculation yields **8–49%** reductions, with consistent direction in **18 of 20** cells. The plot shows genomics/SSD: purple isolates speculation, red sharing.

At $N = 1$ in genomics/SSD, cross-job reuse is absent, yet reactive takes **2× as long**. Both stage the same logical bytes. Reactive has 5–10× bandwidth headroom. This control isolates the timing benefit: blocking stage-in exposes latency that speculation hides.

Three regimes where something simpler wins

Regime	What wins	Margin
506 MB per job	direct read	StageOnce +4.5% at $N = 12$
Working set ≈ dataset	prologue bulk copy	3–7%
$N = 1$ and most $N = 2$ cases	fresh	direct-read only

StageOnce beats independent staging when fleet demand exceeds what reasoning windows can hide and reuse amortises coordination overhead. Against a prologue, the comparison depends on working-set size and setup amortisation. Reuse is **node-local**, so jobs must be co-scheduled.

Acknowledgment

This material is based upon work supported in part by the National Science Foundation (NSF) under Awards OAC 2411318 and OAC 2313154.

References

[1] Michael J. Brim, Adam T. Moody, Seung-Hwan Lim, Ross Miller, Swen Boehm, Cameron Stanavige, Kathryn M. Mohror, and Sarp Oral. UnifyFS: A user-level shared file system for unified access to distributed local storage. In *2023 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 290–300. IEEE, 2023.

[2] Ioan Raicu, Yong Zhao, Ian T. Foster, and Alex Szalay. Accelerating large-scale data exploration through data diffusion. In *Proceedings of the 2008 International Workshop on Data-aware Distributed Computing (DADC)*, pages 9–18. ACM, 2008.

[3] Yifan Sui, Han Zhao, Rui Ma, Zhiyuan He, Hao Wang, Jianxun Li, Kaiqiang Xu, Kai Chen, and Yuqing Yang. Parallelizing tool execution and LLM generation for low-latency agent serving. arXiv preprint arXiv:2603.18897, 2026.

[4] Izzet Yildirim, Xian-He Sun, and Anthony Kougkas. AgentStage: Exploiting LLM thinking for data staging in scientific agents. In *22nd IEEE International Conference on eScience (eScience)*, 2026. Accepted, main track.