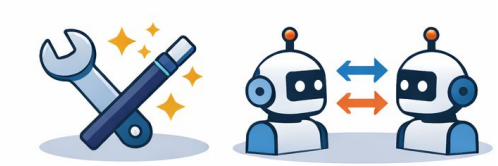
Towards Agentic Performance Management

Leo Chen, Gongqi Huang, Amit Levy
Princeton University

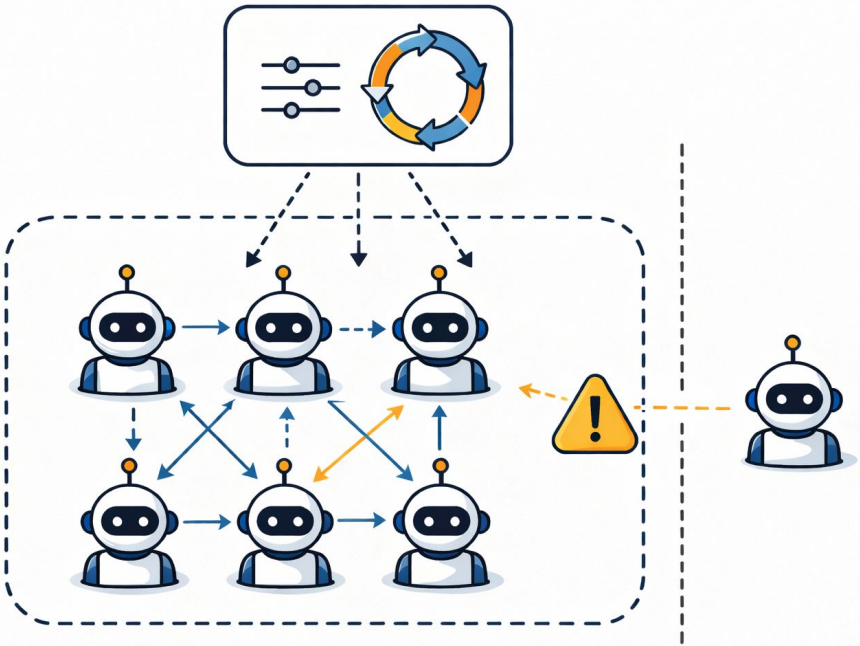
Agents Today

- AI Agents Today Are Designed…
  - Goal-Oriented
  - Self-Evolving
  - Open, Nondeterministic, Environment
- Agentic Execution Model
  - Dynamic Execution Graphs
    - e.g., Tool Usage
  - Fluid Cross-Agent Interaction

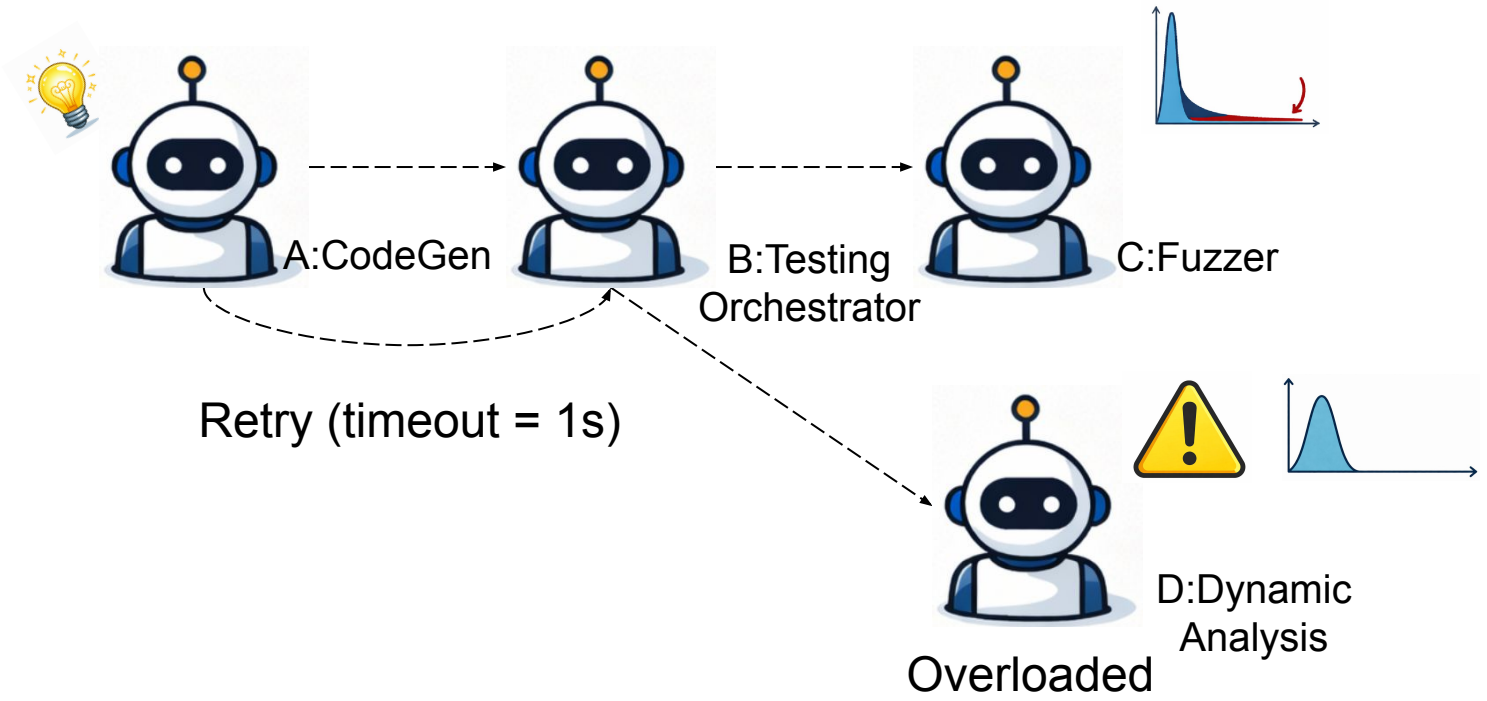


Challenges in Performance Management for Agents

- Existing Methods Can't **Evolve**
  - Stable Objectives and System Structures
  - Largely Static Policy Spaces
  - Reactive Control w/o Extensible Models
    - Local feedback loops
    - Reactive autoscaling
    - Monolithic ML policies
  - Require Manual Tuning/Costly Retraining

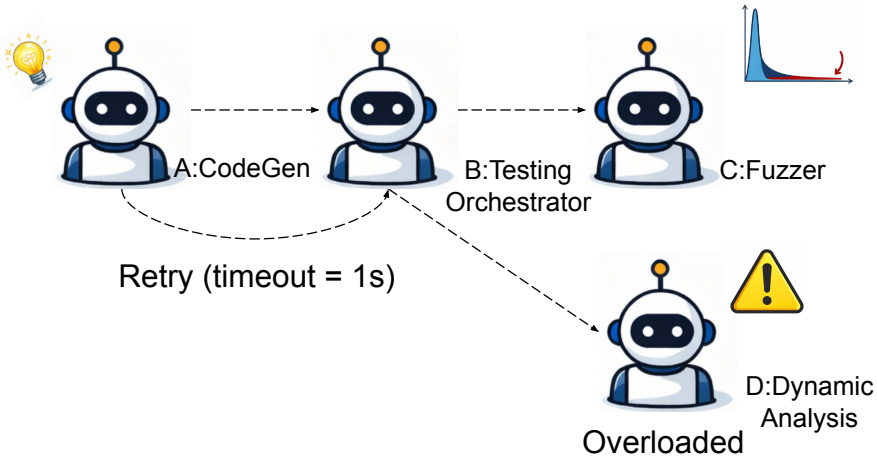


A Synthetic Example: A CodeGen Pipeline



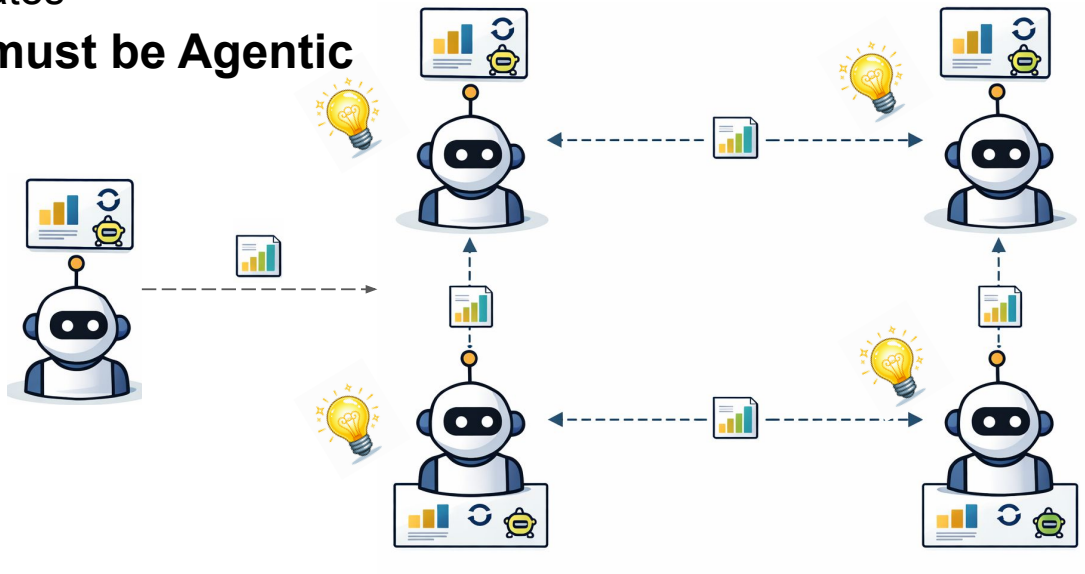
Examples of Failures in Existing Methods

- A increases timeout
  - Suffer from long tails
- D rate-limits B
  - Increase latencies for A/B
  - Hurt other agents using B
- D autoscales
  - Waste resources
  - Decreased goodput
  - Reinforce retry storms/metastability

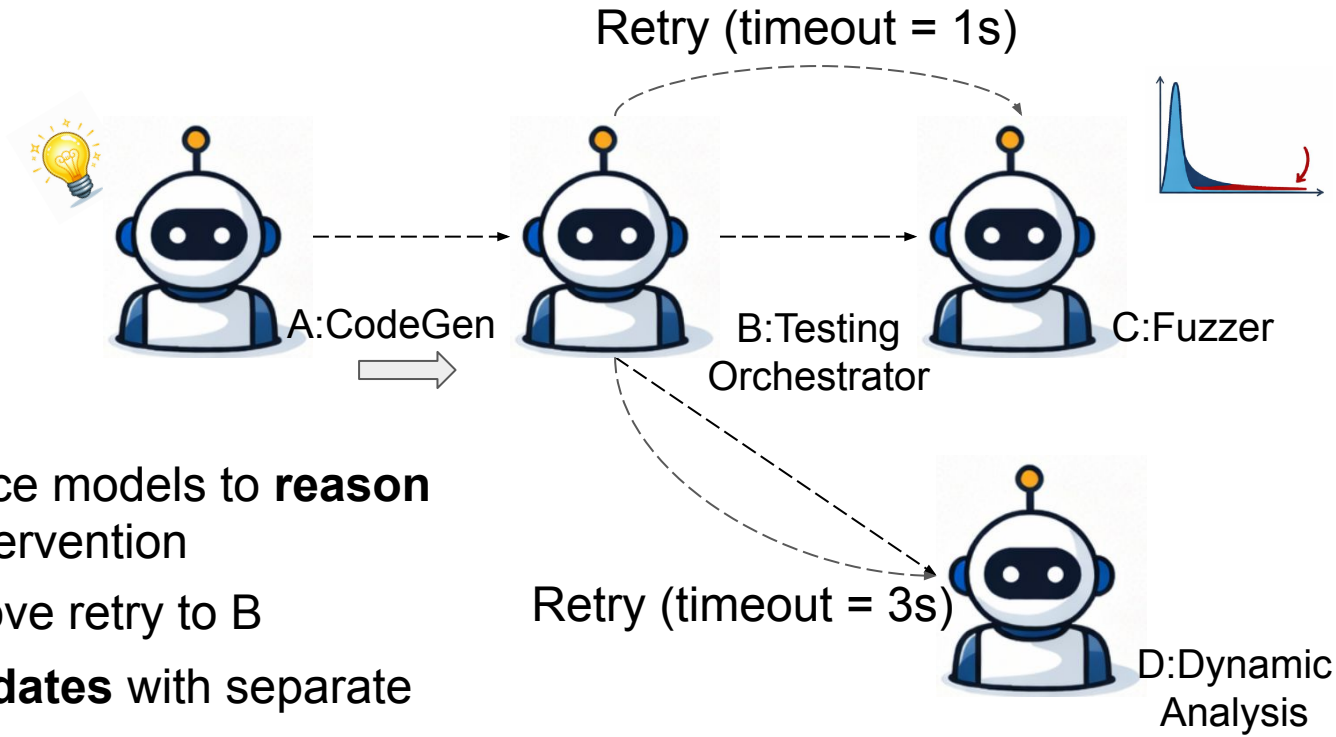


The Need for Evolvable Performance Management

- Performance Management for Agents must
  - Have Evolvable Objectives and Policies
  - Adapt to Changing System Structures
  - Coordinate for Optimal Updates
- **Performance Management must be Agentic**



An Idea Solution with Evolution



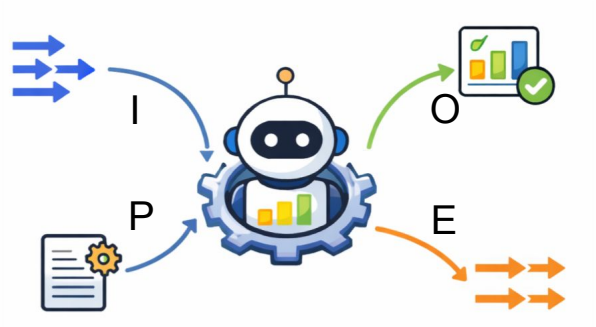
- A uses performance models to **reason** about potential intervention
- A **proposes** to move retry to B
- B accepts and **updates** with separate timeouts for C/D

Agentic Performance Management

- Each agent exposes its model via a **Performance Interface** (PI):

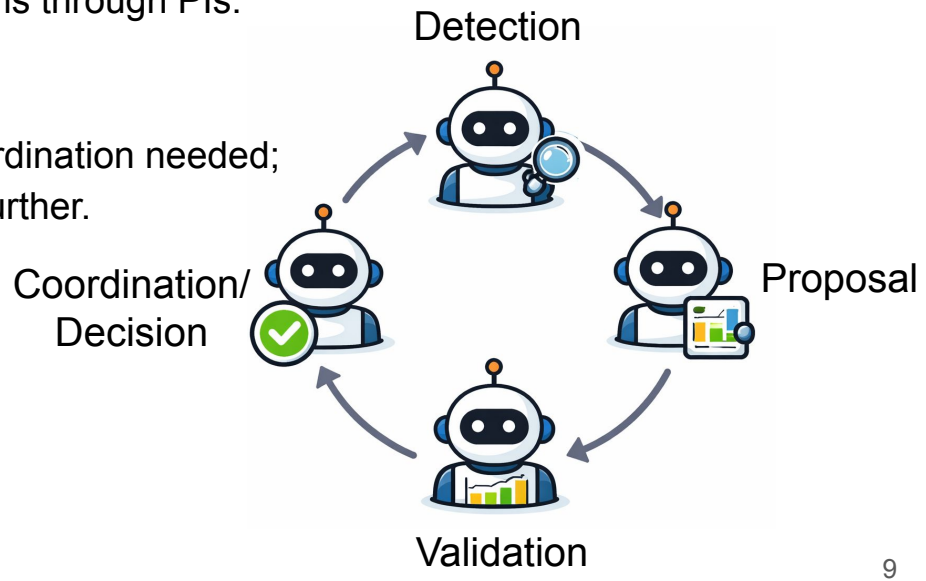
$$I \times P \Rightarrow O \times E$$

  - I : Environmental Inputs (e.g., rate of incoming loads)
  - P: Policies of the agent & other relevant agents
  - O: Observable performance metrics (e.g., p99 latencies)
  - E: Effects on the environment (e.g., rate of outgoing loads)



Agentic Performance Management

- The Coordination State Machine
  - **Detection**: Detect objective violation via the observable metrics.
  - **Plan Generation**: Formulate candidate interventions.
  - **Plan Validation**: Evaluate intervention plans through PIs.
    - Filter/prioritize plans
  - **Coordination/Decision**:
    - Send requests to other agents if coordination needed;
    - Receivers accept/reject/coordinate further.



Challenges

- How to maintain **accurate** and **lightweight** performance models?
  - Combine analytical methods (network calculus, queueing theory), ML-based models and simulation to explore good trade-offs between fidelity, extensibility and cost.
  - How to reduce the cost of simulation?
  - How/When to update the models?
- How to auto-evolve under complex dynamics and **open** policy spaces?
  - Use LLM agents themselves!
  - LLMs are good at generating performant policies [1].
  - How to avoid expensive LLM costs?
    - Avoid LLM costs on the critical path.

[1] Cheng, Audrey, et al. "Let the Barbarians In: How AI Can Accelerate Systems Performance Research." *arXiv preprint arXiv:2512.14806* (2025).

THANK YOU !!