# OPTIMAS: Globally Aligned Local Rewards for Compound AI Systems

Presenters: Zeezoo Ryu and Binhong Ye

Georgia Tech

# From Monolithic Models to Compound Systems

- Modern AI systems have evolved past single LLMs. They are now complex architectures integrating diverse, specialized components working in sequence to solve real-world problems.
  - Large Language Models
  - Retrievers and Vector Databases
  - Specialized APIs and Tools
  - Traditional Machine Learning Classifiers

# Cascading Fragility with Interconnected Power

- Compound systems are highly sensitive to the failure of individual components.
- A single misinterpretation early in the pipeline cascades into system-wide failure.
- If an LLM misinterprets an input query, it retrieves irrelevant information, causing subsequent tool calls to operate on incorrect inputs, ultimately producing unreliable outputs across the entire system.

# Difficulty in Handling Heterogeneous Pipelines

- We cannot simply use standard backpropagation to fix a compound system.
- **Non-Differentiable**
  - The system relies on hard breaks-text generations, API calls, and tool usage.
- **Highly Heterogeneous**
  - Components are tuned differently. How do you simultaneously optimize a text prompt, continuous model weights, and a discrete hyperparameter?
- **Prohibitively Expensive**
  - Running the full system to test every minor tweak consumes massive time and compute.

# Formalizing the System

We define a compound AI system mathematically as a DAG, $G = (C, E)$.

- **Components** $(C_k)$: The task nodes in the network.
- **Inputs** $(x_k)$ & Outputs $(y_k)$: The flow of information.
- **Configurations** $(v_k)$: The optimizable policy controlling each component (prompts, weights, search depths).

- The system executes components in topological order:

$$y_k = C_k(\{y_i | C_i \in pa(C_k)\}; v_k)$$

# Global Optimization Objective

Our goal is to find the perfect joint configuration policy $v^*$ that maximizes the expected global reward R across the entire system.

Across all data

Global system success

$$\mathbf{v}^\star = \arg\max_{\mathbf{v}} \mathbb{E}_{x\sim\mathcal{D}}\left[R\left(x, f\left(x;\mathbf{v}\right)\right)\right].$$

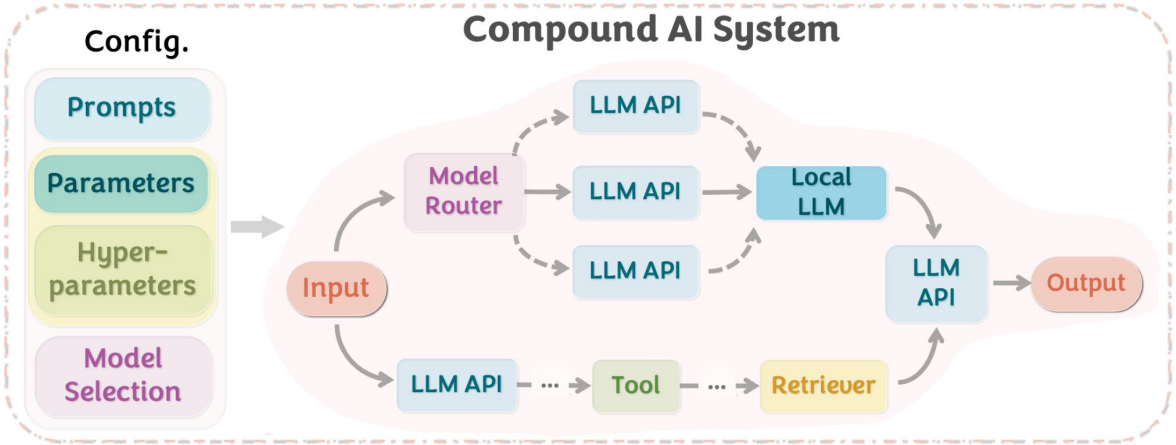
The optimal setup

Where $f(x; v)$ is the overall system execution and D is the dataset of initial inputs.

Georgia Tech

# OPTIMAS

- A unified framework for end-to-end optimization of compound AI systems
- Globally Aligned Local Reward Functions (LRFs)
  - Align each component's optimization with global system performance
  - Enables efficient, decentralized optimization while ensuring that local improvements contribute meaningfully to global rewards, backed by formal theoretical guarantees.




Georgia Tech

# Local Reward Function (LRF)

The core of OPTIMAS is maintaining one Local Reward Function per component. Instead of waiting for the final system output to determine success, an LRF evaluates a specific component's output $(y_k)$ immediately, given its specific input $(x_k)$.

$r_k : (x_k, y_k) \rightarrow \mathbb{R}$

# Evaluator Implementation

To scale efficiently across many components, OPTIMAS uses a shared neural architecture. A shared LLM backbone ($\phi$) processes the concatenated text inputs, while separate linear projection heads $(h_k)$output component-specific rewards.

$$r_k(x_k, y_k) = h_k \cdot \phi([x_k, y_k])$$

# The Guarantee: Local-Global Alignment

- • Decentralization introduces a critical risk. If every component optimizes strictly for its own isolated metric, the system fractures.
- A retriever might maximize "document recall" by pulling thousands of pages, overwhelming the downstream LLM and crashing the global system performance.
- Local success must strictly correlate with global success.
- An LRF is only valid if it acts as a proxy for the global reward.
- We require a Local-Global Alignment Property: If a component produces a "better" output according to the local reward, the expected global reward of the entire downstream system must also increase.

# Mathematically Guaranteeing Systemic Alignment

- We mathematically guarantee this alignment. For any two candidate outputs $y_k^+ and y_k^-$ from component $C_k$

$$r_k(x_k, y_k^+) \geq r_k(x_k, y_k^-) \Rightarrow \mathbb{E}_{\mathbb{down}}[R(x, f(x; v_{-k})|y_k^+)] \geq \mathbb{E}_{\mathbb{down}}[R(x, f(x; v_{-k})|y_k^-)]$$

Georgia Tech

# Generating Preference Data

To teach the LRF this alignment, we construct a preference dataset $\mathcal{D}_k(v)$:

1. Execute the system up to component $C_k$.
2. Sample two candidate outputs for $C_k$.
3. Estimate their downstream global impact via Monte Carlo sampling.
4. Label the output with the higher expected global reward as the winner $(y_k^+)$ and the other as the loser $(y_k^-)$.

# Learning the Alignment

We train each local reward function to obey the alignment property using a pairwise log-sigmoid ranking loss over our preference data.

$$\mathcal{L}_{k}\left(D_k(v)\right) = -\mathbb{E}_{(x_k, y_k^+, y_k^-)\sim \mathbb{D}_k(v)}\left[\log \sigma\left(r_k(x_k, y_k^+) - r_k(x_k, y_k^-)\right)\right]$$

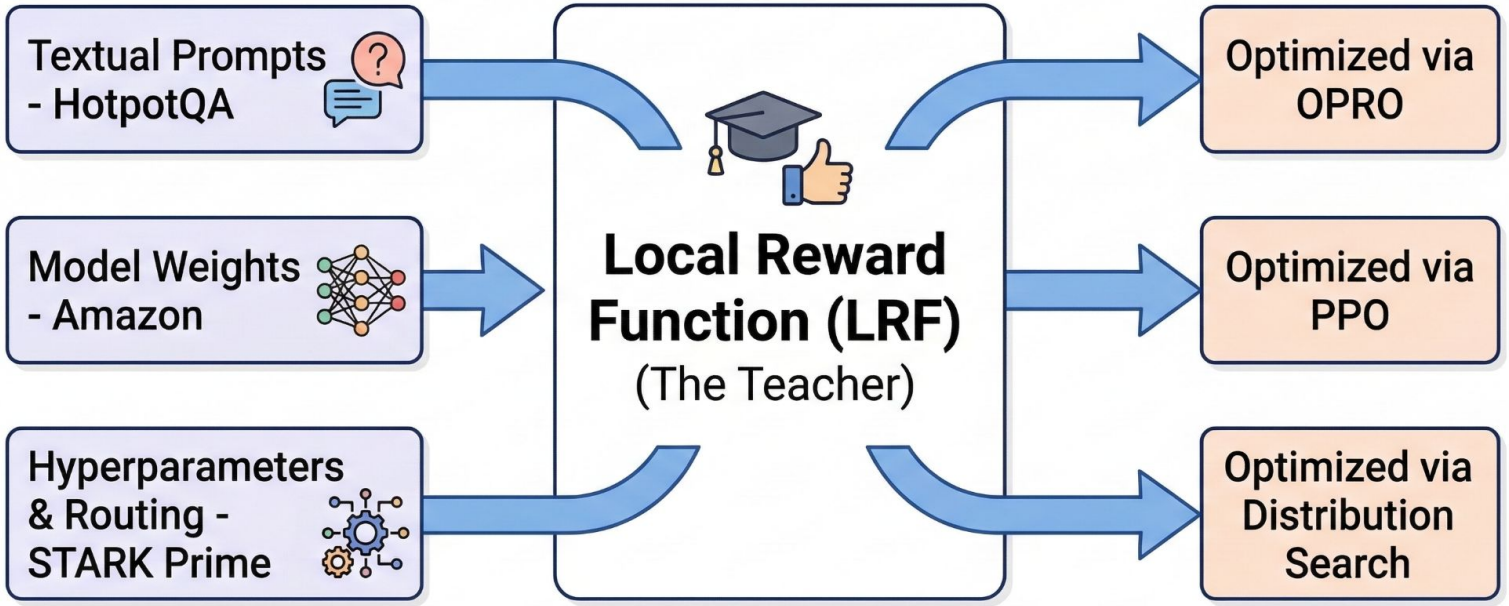
This forces the LRF to correctly rank outputs exactly as the global system would.

Georgia Tech

# The OPTIMAS Adapative Loop

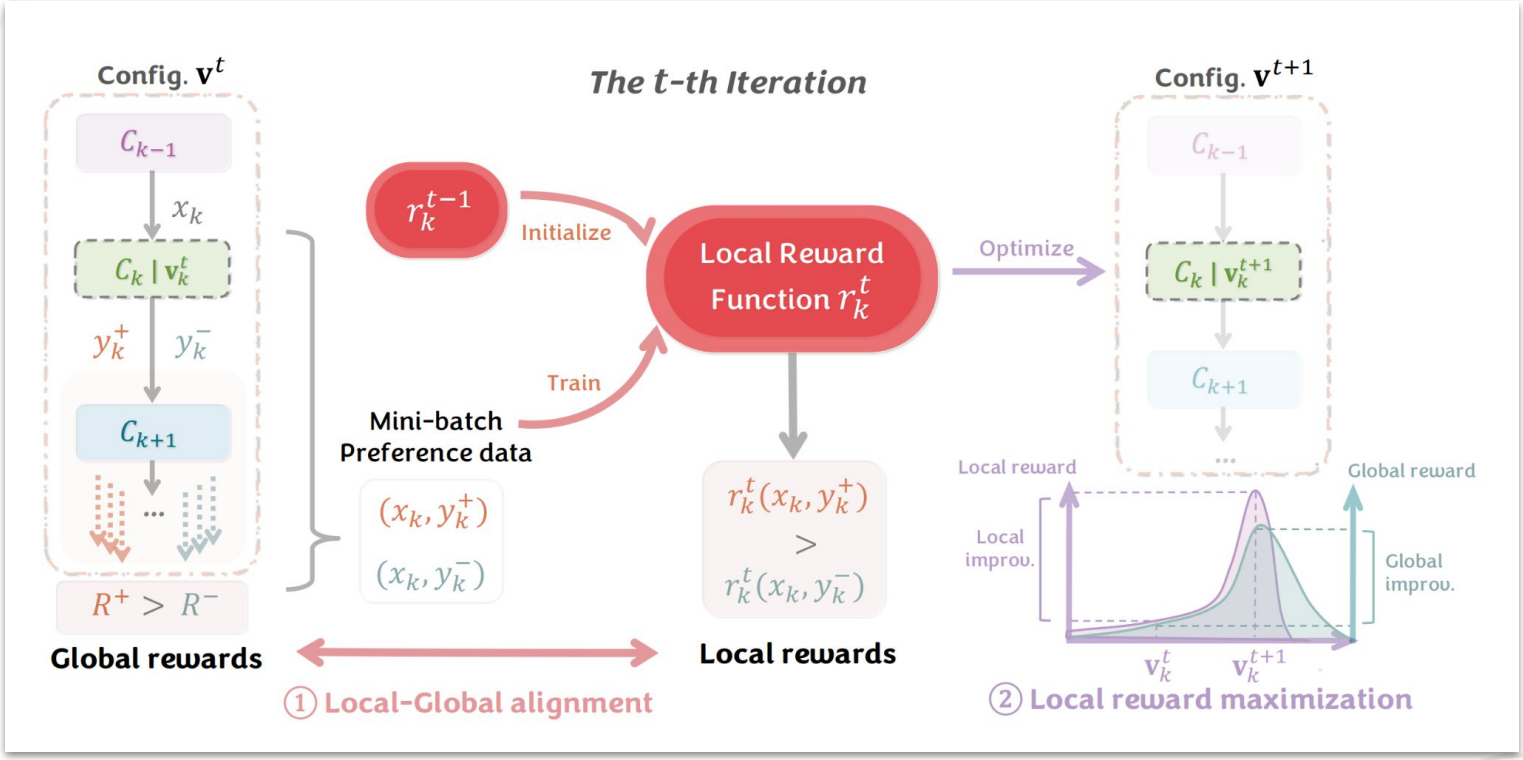
- By maintaining globally aligned, adaptive Local Reward Functions, OPTIMAS transforms a non-differentiable, heterogeneous DAG into a living, optimized entity.
- We can now apply specialized optimization methods (PPO for weights, OPRO for prompts) locally to each component, drastically reducing the need for expensive full-system runs while mathematically guaranteeing global improvement.

# Specialized Local Optimization




Georgia Tech

# The OPTIMAS Algorithm




Georgia Tech

# Guaranteeing Alignment

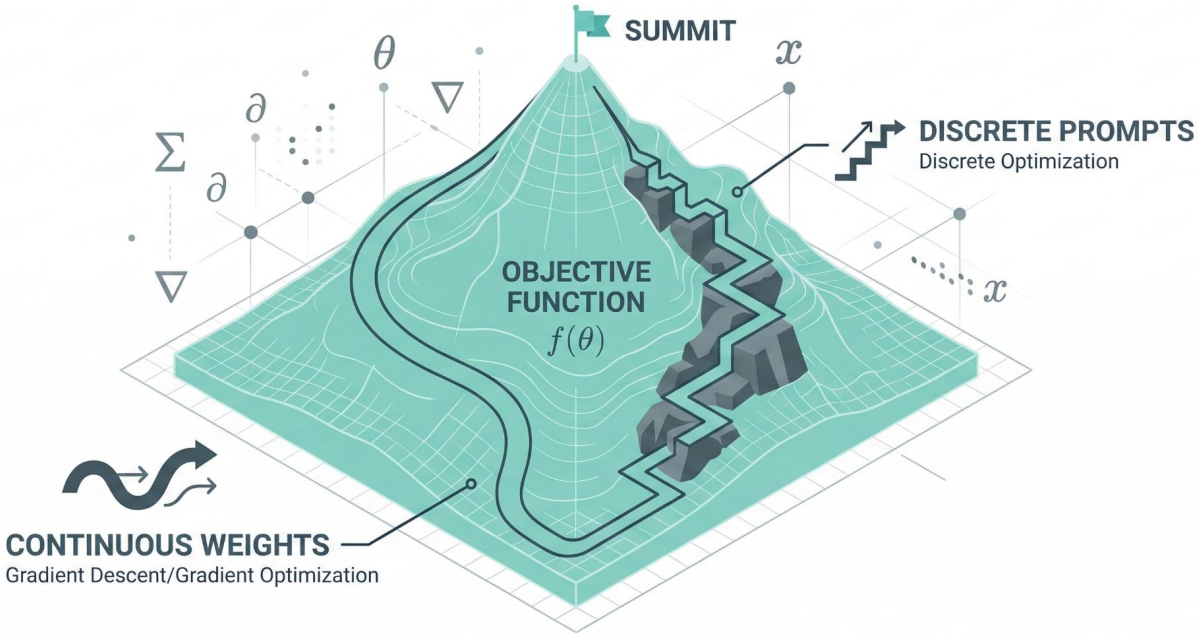
Equation 3 (Local-Global alignment property) from the paper:

$$r_k(y_A) > r_k(y_B) \Rightarrow \mathbb{E}[R \mid y_A] > \mathbb{E}[R \mid y_B]$$

Simplified theorem notation:

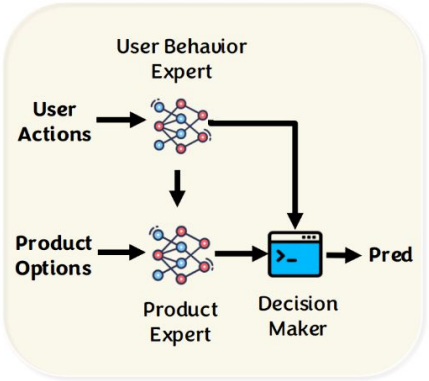
$$\arg\max_{v_k} r_k = \arg\max_{v_k} R$$

# Convergence Analysis

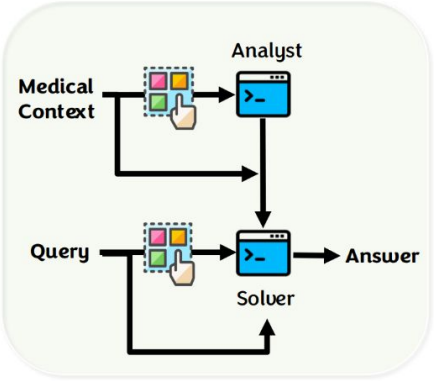



Georgia Tech

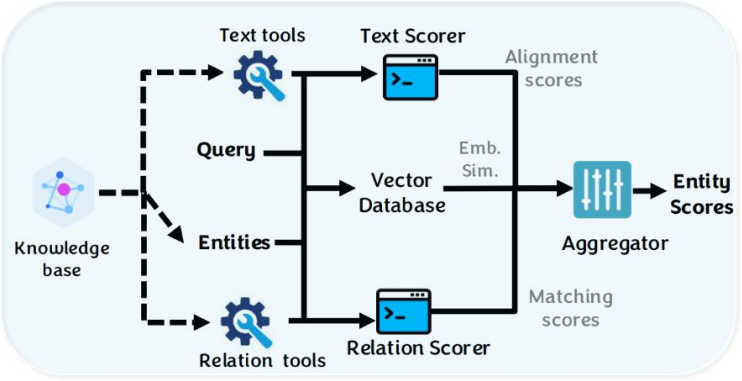
# Real-World Compound Systems



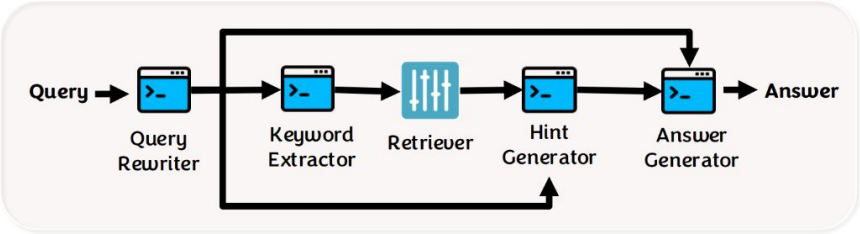

(a) Product Rec. (Amazon)



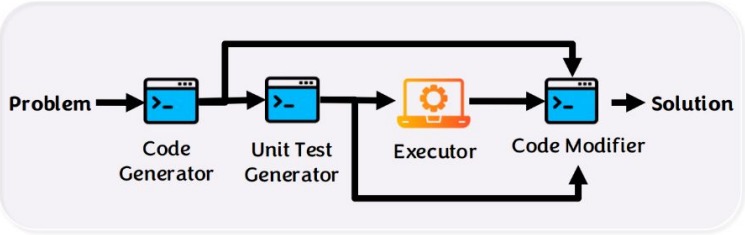

(b) Medical Analysis (PubMed)




(c) Semi-Structured Knowledge Base Retrieval (STaRK)



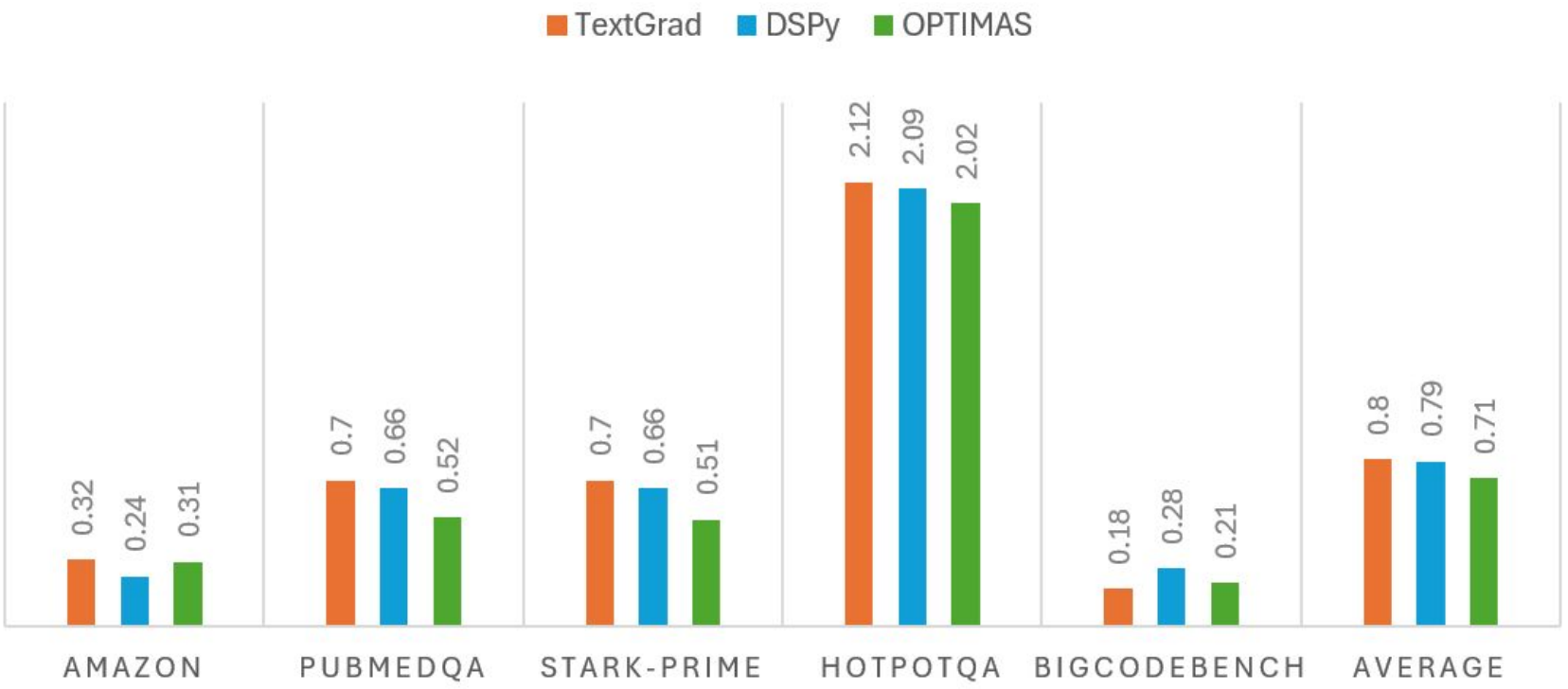

(d) General RAG (HotpotQA)




(e) Self-Verified Code Generation (BigCodeBench)

Georgia Tech

# Baselines and Cost Control



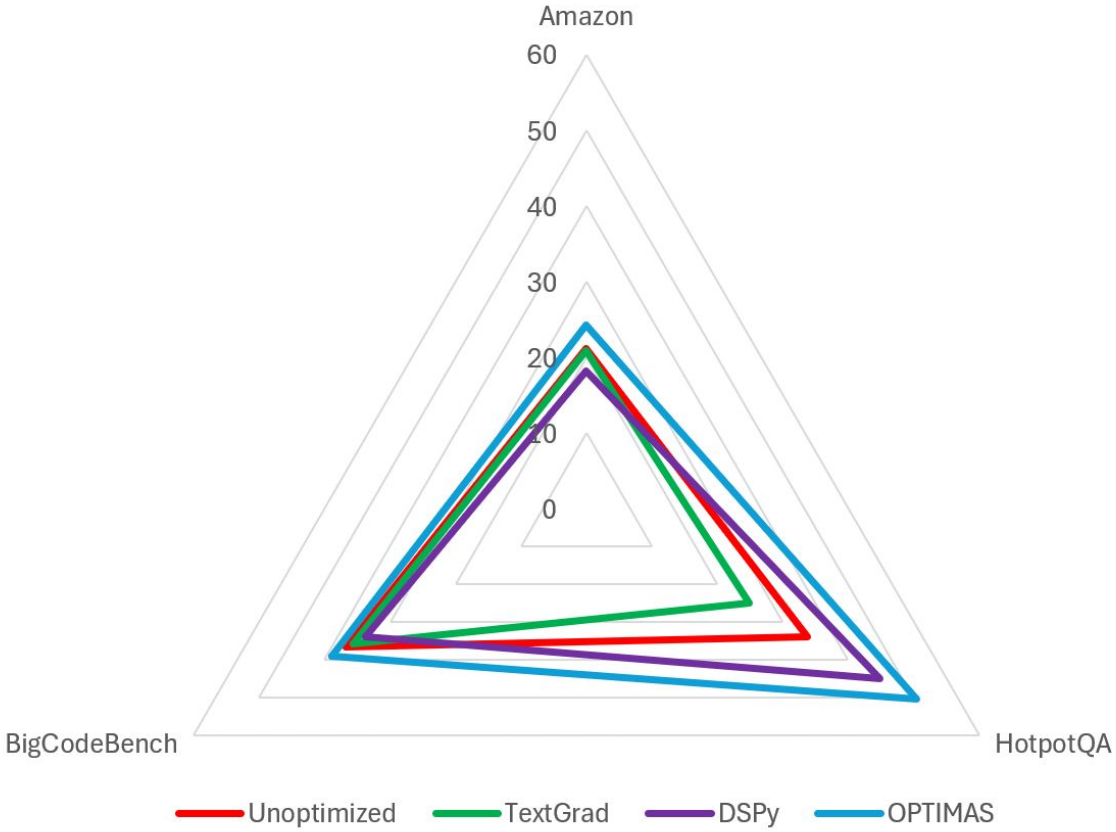

Georgia Tech

# Main Results - Performance Gains

| | AMAZON Product Rec. (Acc.) | PUBMEDQA Medical Analysis (Acc.) | STARK-PRIME Complex Retrieval (MRR) | HOTPOTQA RAG (F1) | BIGCODEBENCH Verified Code Gen. (Pass Rate) |
|---|---|---|---|---|---|
| Single LLM | 20.20±1.43 | 54.13±2.73 | 0.00±0.00 | 21.58±1.24 | 35.47±0.34 |
| Unoptimized | 21.21±3.78 | 57.46±0.75 | 40.73±0.64 | 33.80±1.51 | 36.67±1.35 |
| REINFORCE | 21.89±2.65 | - | - | - | - |
| LLMSelector | - | 67.93±0.09 | - | - | - |
| HBC | 21.55±2.07 | 58.80±0.58 | 36.95±0.59 | 21.16±0.97 | 27.78±2.08 |
| TextGrad | 20.88±3.53 | 56.96±2.24 | 41.31±1.67 | 24.86±1.19 | 35.71±0.10 |
| DSPy | 18.18±0.82 | 60.26±0.40 | 41.40±0.04 | 44.90±0.32 | 33.81±2.75 |
| OPTIMAS | **24.24±0.82** | **69.13±0.33** | **50.54±0.70** | **50.48±1.48** | **38.92±0.36** |
| Rel. Improv. | 14.3% | 1.8% | 22.1% | 12.4% | 9.0% |

Georgia Tech

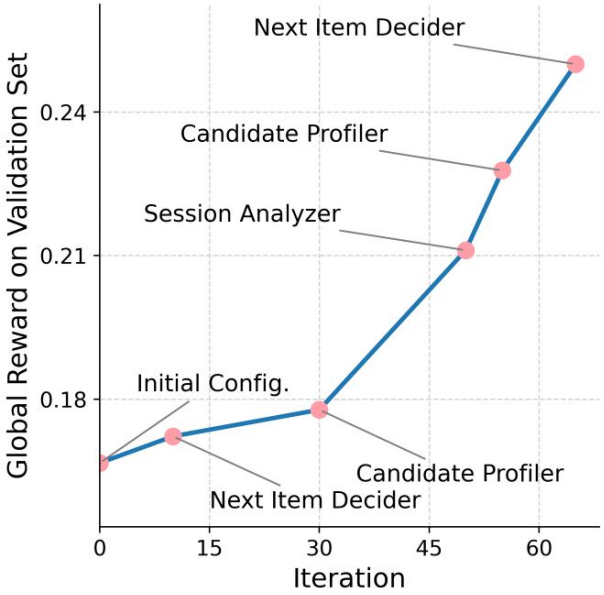
# Consistency Across Tasks

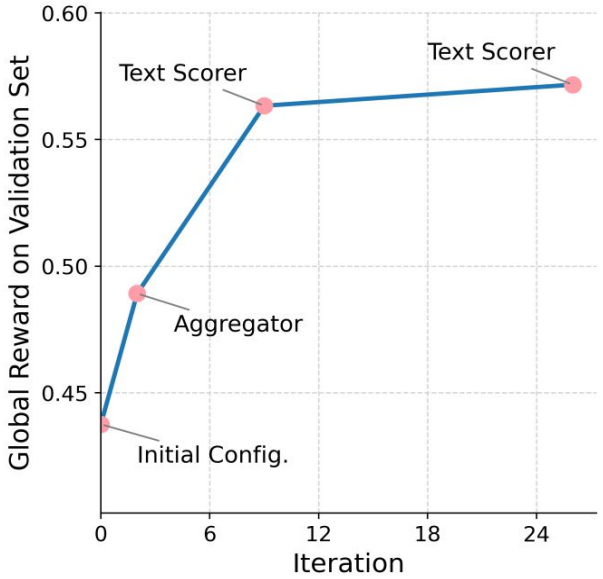


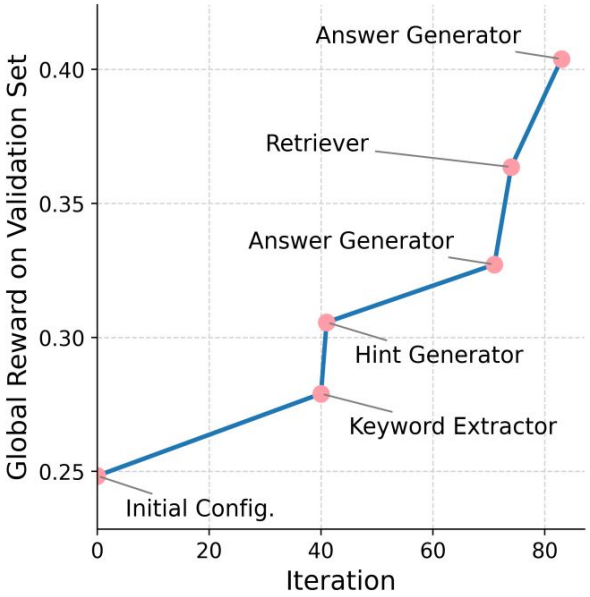

Georgia Tech

# Optimization Trajectory

## Amazon / STARK-Prime / HotpotQA




Georgia Tech

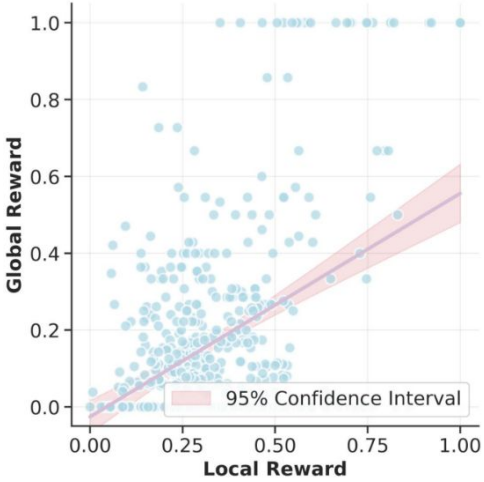
# Alignment Quality

| | AMAZON | PUBMEDQA | STARK-PRIME | HOTPOTQA | BIGCODEBENCH | Avg. |
|---|---|---|---|---|---|---|
| LLM Judge | 51.25% | 49.54% | 54.37% | 50.00% | 42.45% | 49.52% |
| OPTIMAS | 84.93% | 65.28% | 76.64% | 72.40% | 90.57% | 77.96% |

| $k$ | 1 | 2 | 3 | 5 | 10 | 15 | 25 |
|---|---|---|---|---|---|---|---|
| Local reward | 0.4247 | 0.5578 | 0.5695 | 0.6124 | 0.6117 | 0.5949 | 0.5123 |
| Global reward | 0.3398 | 0.3493 | 0.3325 | 0.3598 | 0.3645 | 0.3568 | 0.3465 |

Georgia Tech

# Interpretability




(a) Local-Global Reward Correlation

*Before optimizing with LRF:*

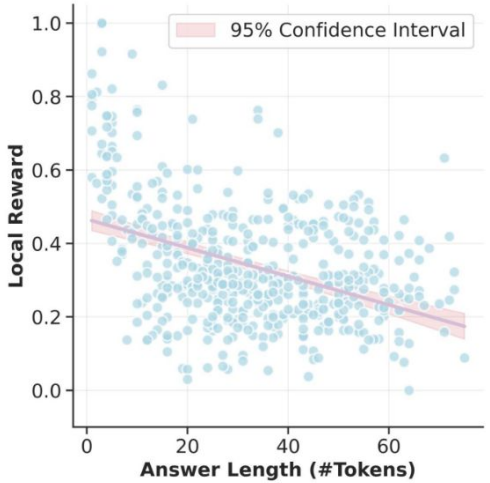
*Given some hints,*
*directly answer the query with a short answer.*

*(local reward=0.17, global reward=0.32)*

*After:*

*Given 3 to 5 clear and relevant hints,*
*provide a direct and concise answer to the*
*query in no more than 10 words*
*....*
*(local reward=0.22, global reward=0.36)*

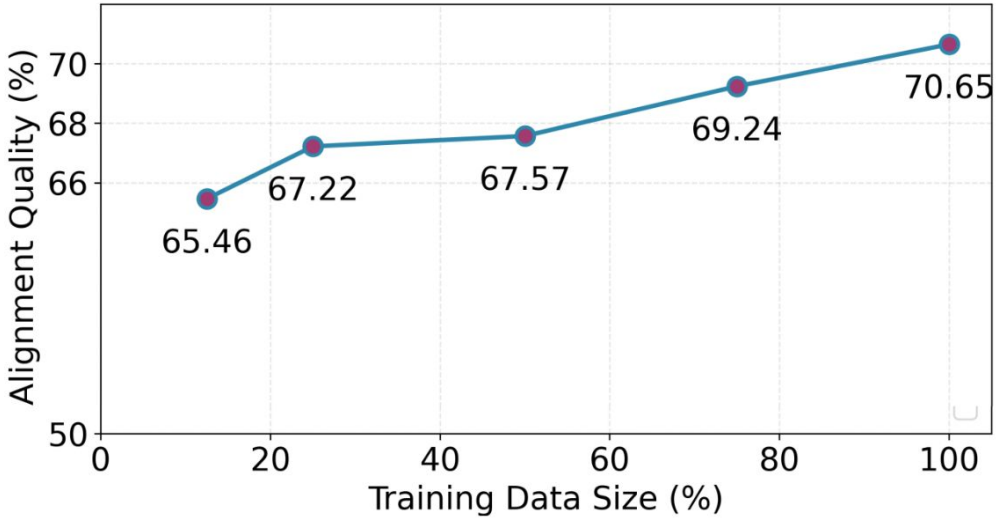
(b) Prompts for Answer Generator on HotpotQA




(c) Length-Local Reward Correlation

Georgia Tech

# Data & Compute Efficiency




| Backbone size | Alignment quality (%) |
|---|---|
| 1B | 72.58 |
| 3B | 70.04 |
| 8B | 71.20 |

Georgia Tech

# Conclusion

- Unified optimization for heterogeneous systems
- Globally aligned local rewards
- High data/compute efficiency