SCOPE: Cost-Efficient Model Selection for Compound AI Systems under Quality Constraints

Yiqian Huang, Shiqi Zhang, Tianyuan Jin, Xiaokui Xiao

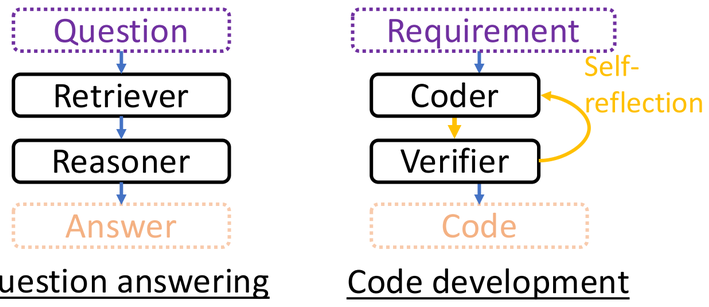
NUS National University of Singapore

香港科技大学(广州) THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

KDD 2026 August 9–13, 2026 Jeju, Korea

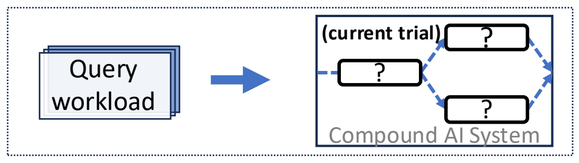
Compound AI System

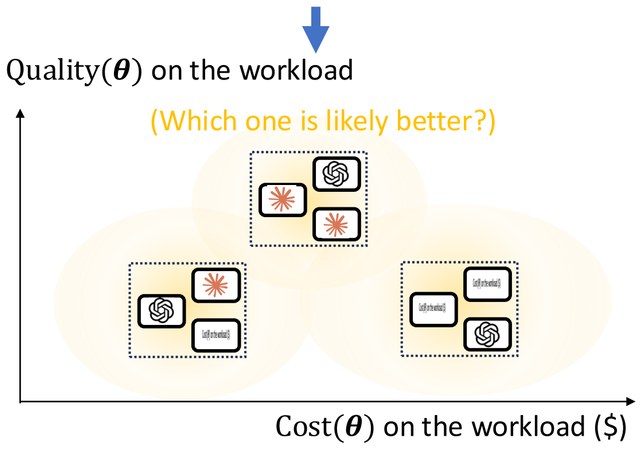
It is a system made up of multiple AI agents



Existing Solutions' Recipe

Existing solutions use *the whole workload* to figure out which configuration to try next



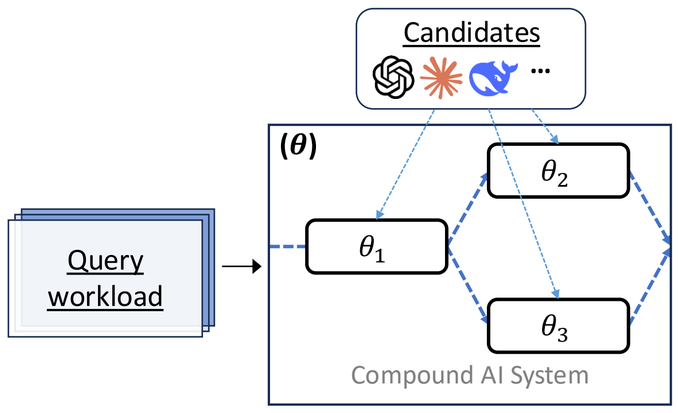


Using trial and error, existing methods are finding better configurations as the next incumbents (**General baselines**: Bayesian optimization, bandits)

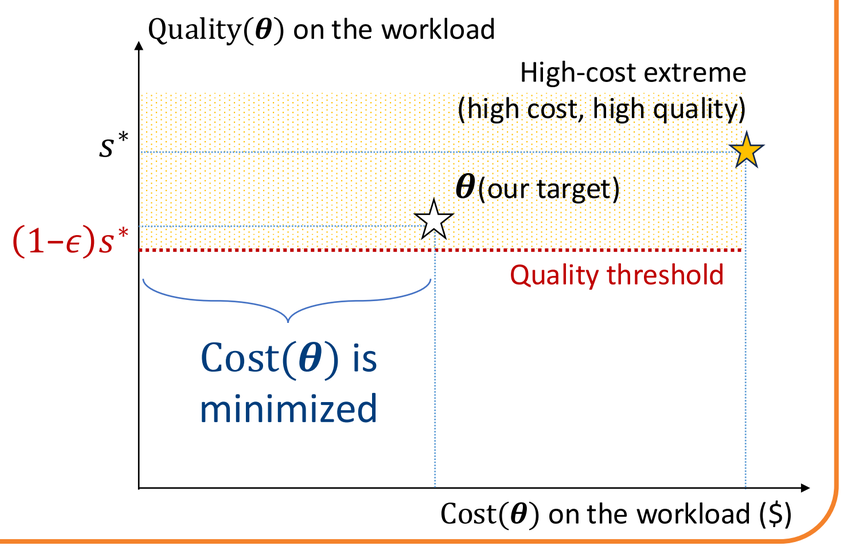
Problem Definition

Setup (left): Compound AI system + Query workload + Candidate LLMs

Input (right, on Y-axis): s^* (quality of High-cost extreme), ϵ (allowable quality degradation)



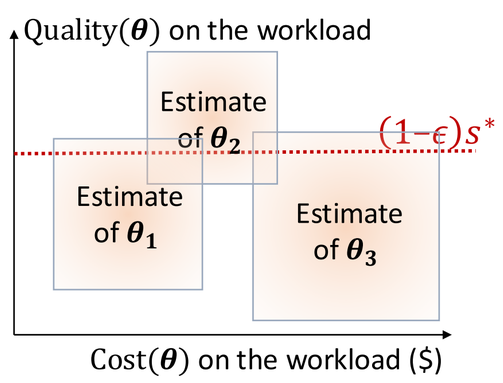
Objective: Minimize $\text{Cost}(\boldsymbol{\theta})$ subject to $\text{Quality}(\boldsymbol{\theta}) \geq (1-\epsilon)s^*$



Existing Solutions: Bandit as an example

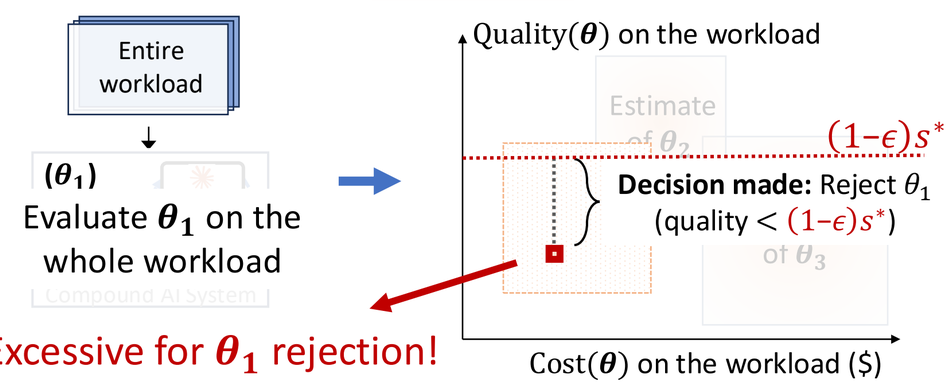
(Bandit methods keep estimates for configurations)

Left: Cost- and quality- estimate ranges are visualized as squares



Right: It evaluates $\boldsymbol{\theta_1}$ to get its cost and quality precisely *for the whole workload*, making the estimate of $\boldsymbol{\theta_1}$ shrink to a point

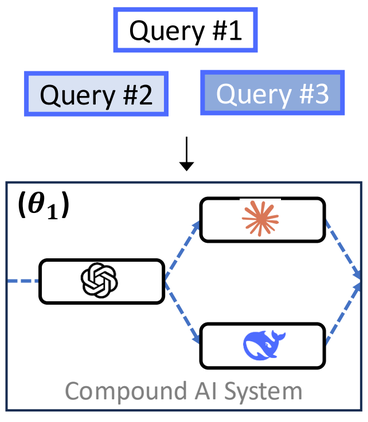
Issue: it wastes money by unnecessarily shrinking squares



Excessive for $\boldsymbol{\theta_1}$ rejection!

SCOPE: Which configuration and query to try next

SCOPE's motivation: chooses queries to shrink the square **most quickly**

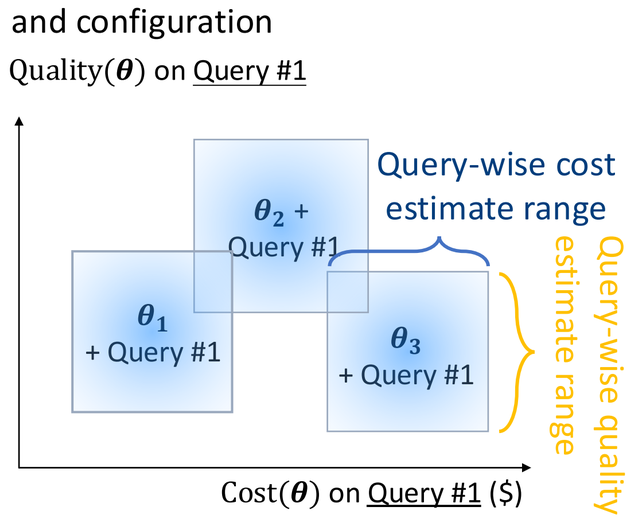


(**Recall:** Existing methods can only choose $\boldsymbol{\theta_1}$)

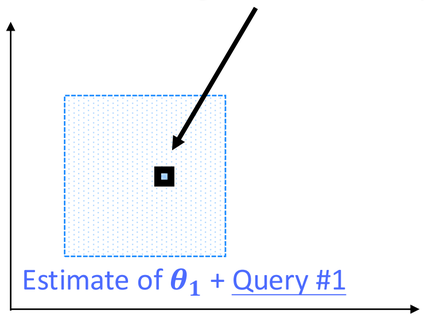
Pseudocode of **SCOPE**

1. For (trials $t = 1,2,\ldots$ while budget remains):
2. Select a configuration $\boldsymbol{\theta}^{(t)}$
3. While NOT (all queries are evaluated):
4. Select the query q with the largest side length
5. Evaluate $\boldsymbol{\theta}^{(t)}$ on query q
6. Break while when either stopping rule is met

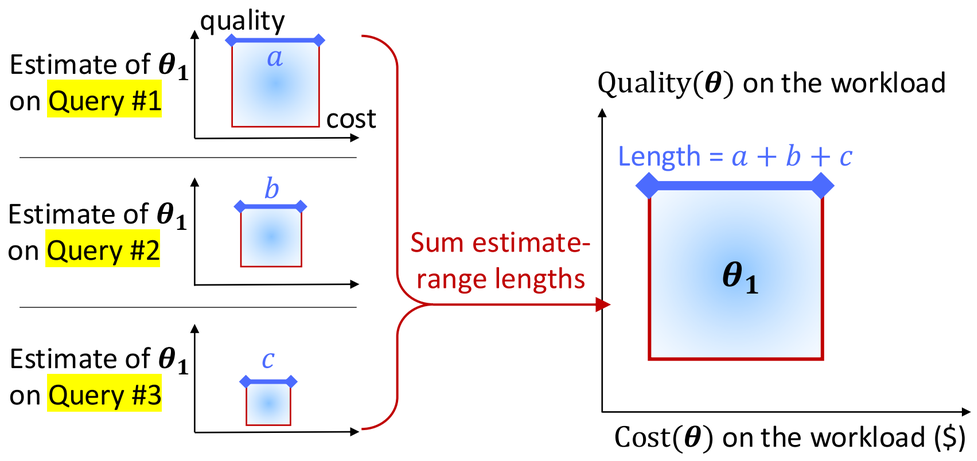
1. SCOPE keeps estimates for each query and configuration



E.g., after SCOPE evaluates $\boldsymbol{\theta_1}$ on Query #1, estimate ranges shrink to a point



2. SCOPE sums query estimate ranges into workload estimate ranges

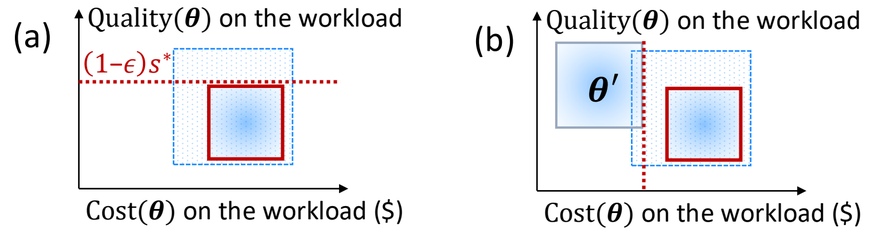


3. SCOPE *greedily* evaluates the query with the **largest side length** (Apparently, this is the right way to implement SCOPE's motivation!)

4. SCOPE stops each evaluation (Line 6) based on two stopping rules

a. The quality estimate falls below $(1-\epsilon)s^*$

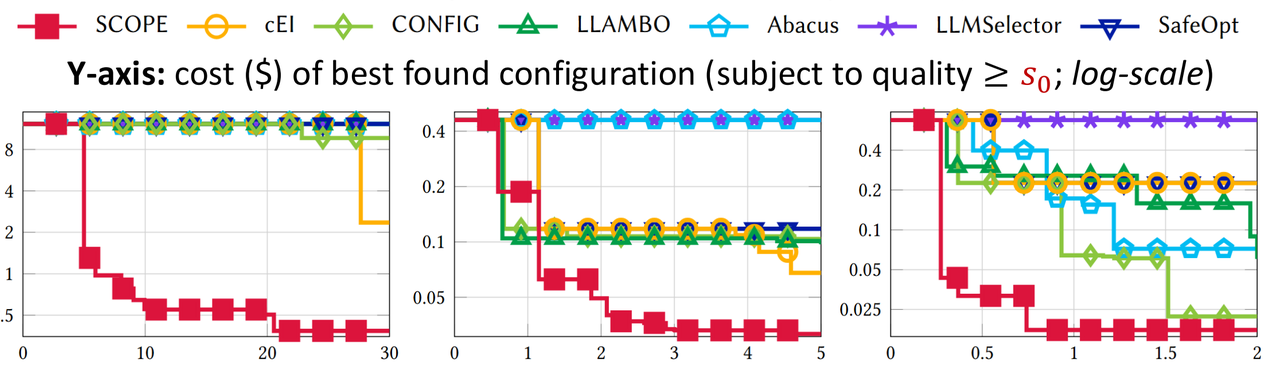
b. The cost estimate is dominated by the incumbent



Experiments

3 systems and **23 candidate LLMs** (⇒ up to ~6,400,000 possible configurations)

(We compare how fast a curve drops below)



Theoretical Results

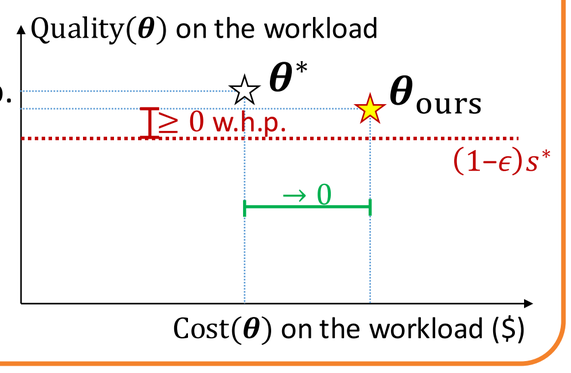
Let $\boldsymbol{\theta}^*$ be the *best feasible* configuration at the lowest cost:

$$\boldsymbol{\theta}^* \in \underset{\boldsymbol{\theta}:\ \text{Quality}(\boldsymbol{\theta}) \geq (1-\epsilon)s^*}{\arg\min} \text{Cost}(\boldsymbol{\theta})$$

SCOPE's output ($\boldsymbol{\theta}_{\text{ours}}$) satisfies:

1. $\text{Quality}(\boldsymbol{\theta}_{\text{ours}}) \geq (1-\epsilon)s^*$ w.h.p.
2. $|\text{Cost}(\boldsymbol{\theta}_{\text{ours}}) - \text{Cost}(\boldsymbol{\theta}^*)| \to 0$ when we increase # of trials

(Both hold even when evaluations include noise)



Code: **github.com/waetr/SCOPELLM-optimizer/**

Email: **yiqian@comp.nus.edu.sg**