UNIVERSITY OF CAMBRIDGE

CherryPick: Adaptively Unearthing the Best Cloud Configurations for Big Data Analytics

Authors: Omid Alipourfard, *Yale University;* Hongqiang Harry Liu and Jianshu Chen, *Microsoft Research;* Shivaram Venkataraman, *University of California, Berkeley;* Minlan Yu, *Yale University;* Ming Zhang, *Alibaba Group*

Presented by Teodora

Big Data Analytics jobs are very popular "nowadays" (2017)

Many cloud providers

- Azure
- AWS
- Google Cloud

Many machine types

- CPUs, GPUs, configurable VMs

Cluster Size

- Tens of possible configurations

HUNDREDS OF POSSIBLE CONFIGURATIONS

UNIVERSITY OF CAMBRIDGE

$$Good\ configuration = Better\ performance + Lower\ \text{cost}$$

$$Same\ \text{cost} \Rightarrow \frac{worst\ running\ time}{best\ running\ time} = 3$$

$$Same\ performance \Rightarrow \frac{the\ most\ expensive\ configuration}{cheapest\ configuration} = 12$$

UNIVERSITY OF CAMBRIDGE

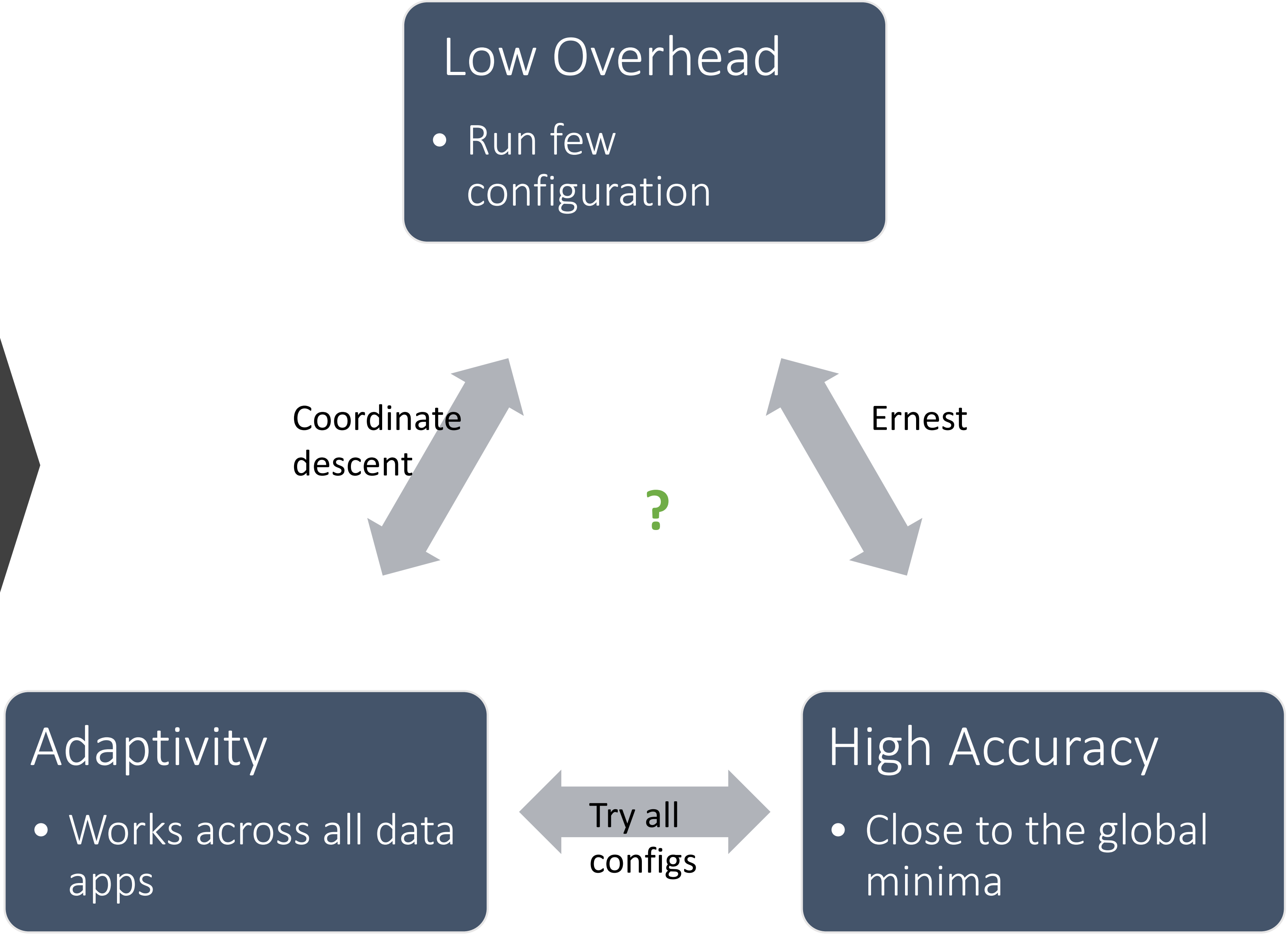
Problem

Find **best cloud configuration** (**minimizes cost** for a given **performance threshold**) for a **recurring jobs**, given its **representative workload**.

40% of the jobs is cloud are **recurring**.

UNIVERSITY OF CAMBRIDGE

Challanges



UNIVERSITY OF CAMBRIDGE

Challanges

Low Overhead

- Run few configuration

Adaptivity

- Works across all data apps

High Accuracy

- Close to the global minima

Coordinate descent

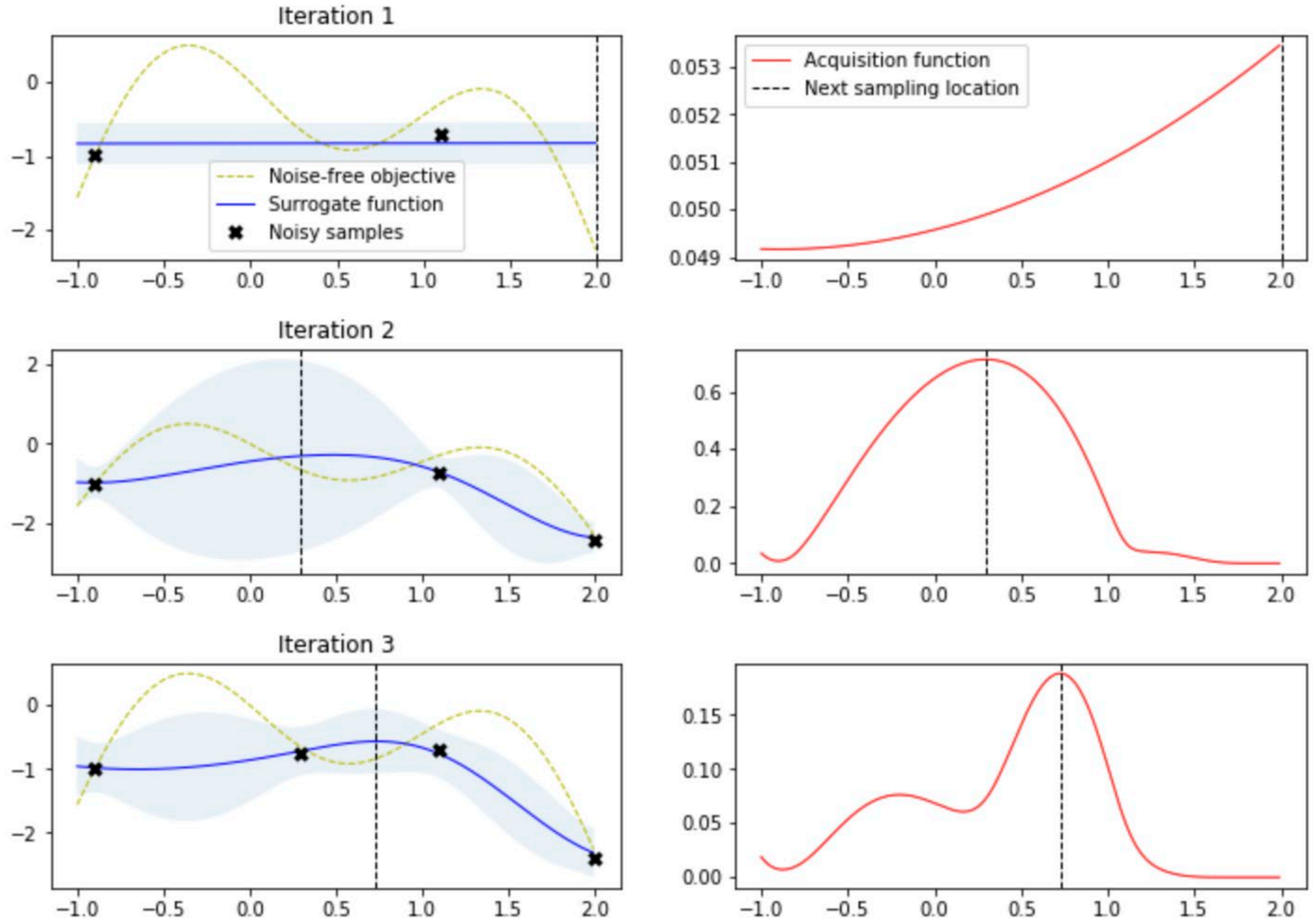
Ernest

Try all configs

Bayesian Optimization

UNIVERSITY OF CAMBRIDGE

Bayesian Optimisation



UNIVERSITY OF CAMBRIDGE

Cloud has multiplicative noise

- cloud resources are shared, so if we run the same workload, same job, with same configuration, the running time and cost might not be the same

Log(cost)

Price per unit of time → $P(\vec{x})$; Time → $T(\vec{x})$

$$\underset{\vec{x}}{\text{minimize}} \quad C(\vec{x}) = P(\vec{x}) \times T(\vec{x})$$

$$\text{subject to} \quad T(\vec{x}) \leq \mathscr{T}_{max}$$

$$\tilde{T}(\vec{x}) = T(\vec{x})(1 + \varepsilon_c)$$

$$\tilde{C}(\vec{x}) = C(\vec{x})(1 + \varepsilon_c)$$

$$\underset{\vec{x}}{\text{minimize}} \quad \log C(\vec{x}) = \log P(\vec{x}) + \log T(\vec{x})$$

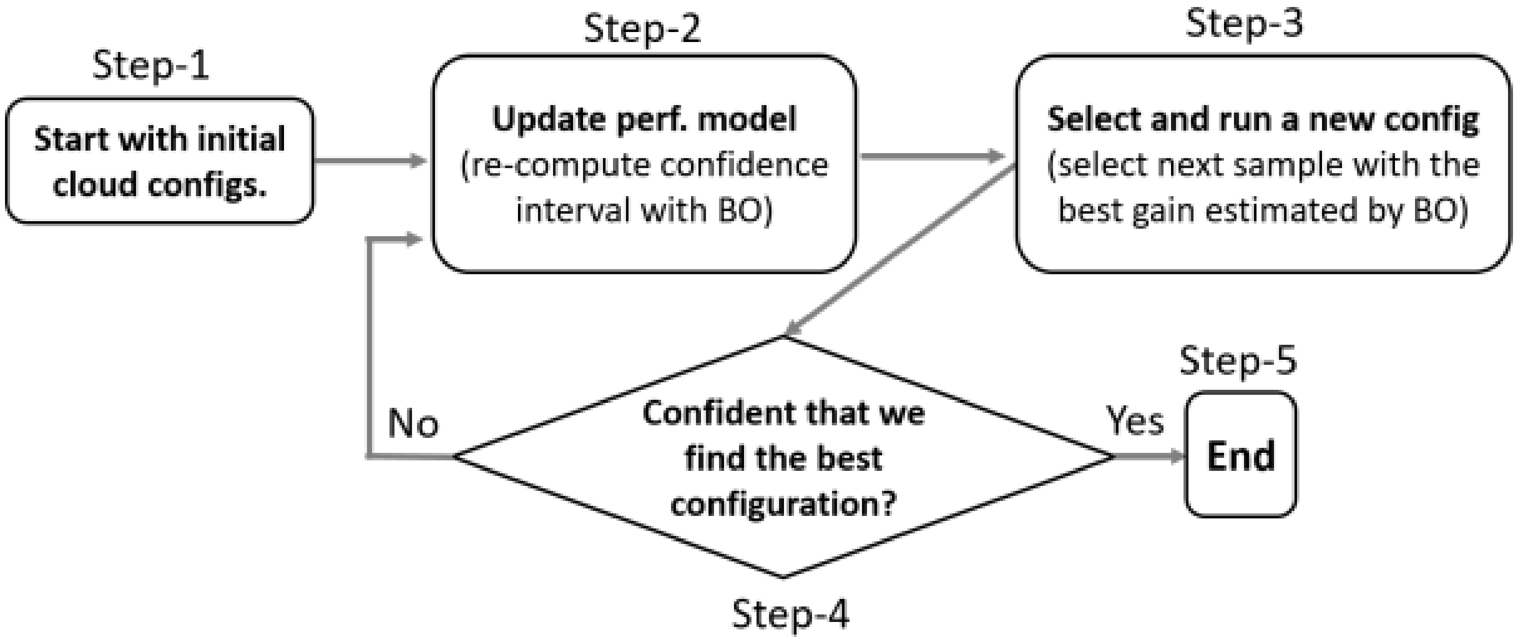
$$\text{subject to} \quad \log T(\vec{x}) \leq \log \mathscr{T}_{max}$$

We use BO to minimize $\log C(\vec{x})$ instead of $C(\vec{x})$

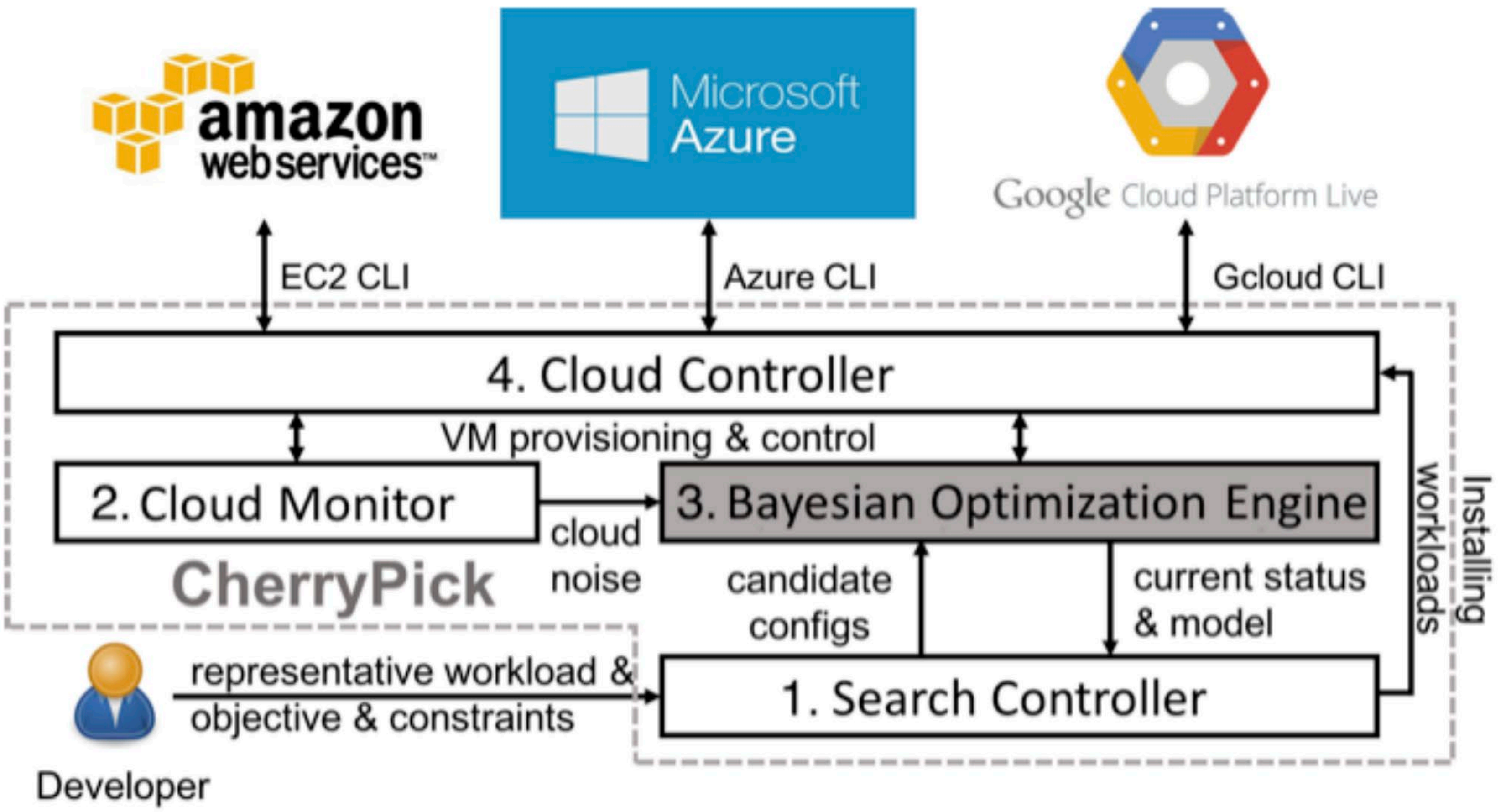
$$\log \tilde{C}(\vec{x}) = \log C(\vec{x}) + \log(1 + \varepsilon_c)$$

UNIVERSITY OF CAMBRIDGE

Workflow



UNIVERSITY OF CAMBRIDGE



amazon webservices
Microsoft Azure
Google Cloud Platform Live
EC2 CLI
Azure CLI
Gcloud CLI
4. Cloud Controller
VM provisioning & control
2. Cloud Monitor
3. Bayesian Optimization Engine
cloud noise
CherryPick
candidate configs
current status & model
Installing workloads
representative workload & objective & constraints
1. Search Controller
Developer

Evaluation

Input:
- Five popular analytical jobs
- 66 reasonable configurations, of four families in Amazon EC2

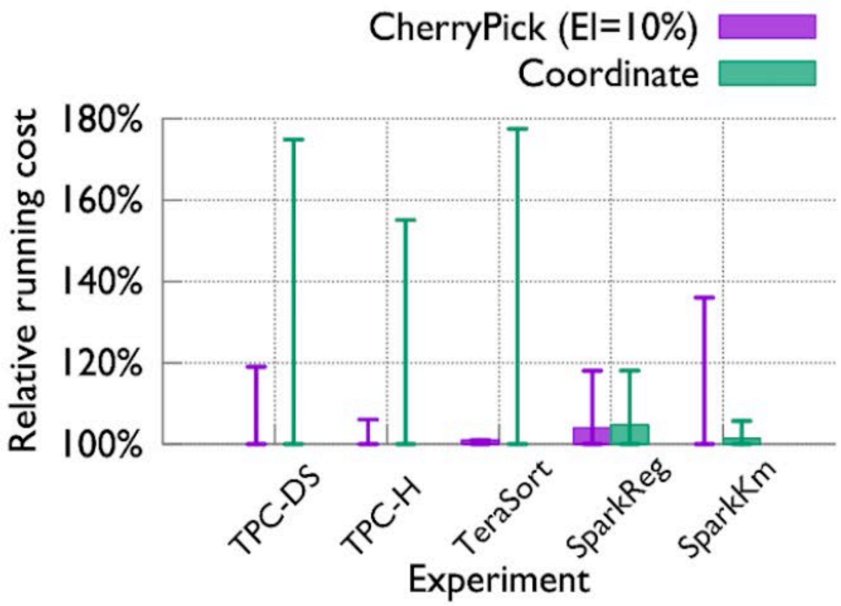
Objective
- Minimize cost, within a performance threshold

Results:

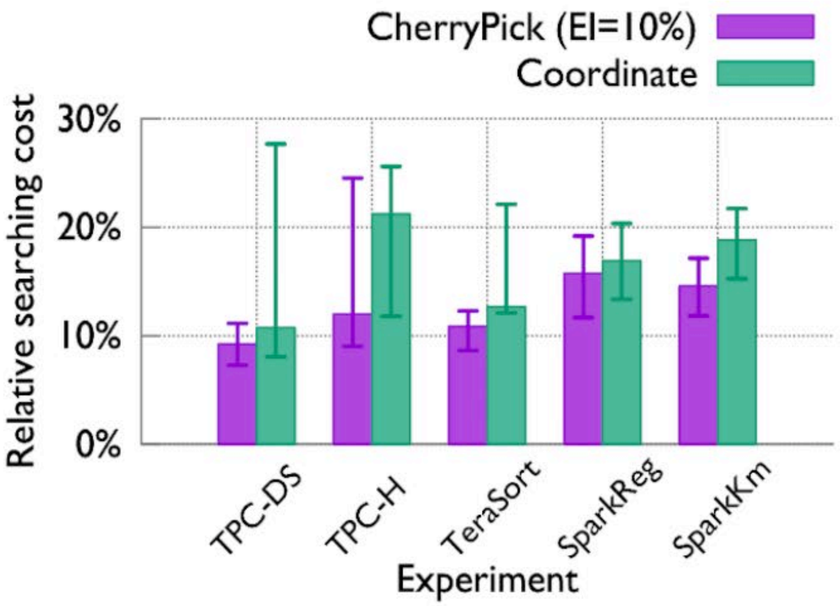
CheryPick = 45-90% to pick optimal solution, otherwise finds a solution within 5%

Alternatives = 75% more time to get to 45% overhead

Results



(a) Running cost



(b) Search cost

Figure 7: Comparing *CherryPick* with coordinate descent. The bars show 10th and 90th percentile.

UNIVERSITY OF CAMBRIDGE

Criticism

The algorithm is strong but the way they phrase it makes it seem weaker "45-90% chance to find the optimal".

Prior set to GP and **acquisition to Expected Improvement,** a bit restrictive? (ex. conjugate distribution for prior)

Representative workloads are needed. How can one get them?

Does it actually converge to a minima for noisy prior?

UNIVERSITY OF CAMBRIDGE


Questions?