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
Good configuration = Better performance + Lower cost

Same cost ⇒ \[\frac{\text{worst running time}}{\text{best running time}} = 3\]

Same performance ⇒ \[\frac{\text{the most expensive configuration}}{\text{cheapest configuration}} = 12\]
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**.
Challenge

Low Overhead
• Run few configuration

High Accuracy
• Close to the global minima

Adaptivity
• Works across all data apps

Coordinate descent
Try all configs
Ernest
Low Overhead
- Run few configuration

High Accuracy
- Close to the global minima

Adaptivity
- Works across all data apps

Bayesian Optimization

Challenges
Bayesian Optimisation
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
minimize $\bar{x}$ \hspace{1cm} C(\bar{x}) = P(\bar{x}) \times T(\bar{x})$

subject to \hspace{1cm} T(\bar{x}) \leq \mathcal{T}_{max}$

$\bar{T}(\bar{x}) = T(\bar{x})(1 + \epsilon_c)$

$\bar{C}(\bar{x}) = C(\bar{x})(1 + \epsilon_c)$

minimize $\bar{x}$ \hspace{1cm} \log C(\bar{x}) = \log P(\bar{x}) + \log T(\bar{x})$

subject to \hspace{1cm} \log T(\bar{x}) \leq \log \mathcal{T}_{max}$

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

$\log \bar{C}(\bar{x}) = \log C(\bar{x}) + \log (1 + \epsilon_c)$
Workflow

Step-1: Start with initial cloud configs.

Step-2: Update perf. model (re-compute confidence interval with BO)

Step-3: Select and run a new config (select next sample with the best gain estimated by BO)

Confident that we find the best configuration?

Step-4: No

Step-5: Yes

End
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.
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?
Questions?