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

Omid Alipourfard, Hongqiang Harry Liu, Jianshu Chen, Shivaram Venkataraman, Minlan Yu, Ming Zhang

Presented by Luou Wen lw658
Background

- Hundreds of possible instance types and instance count combinations
  - different machine types, providers, cluster sizes
- Bad cloud configuration – can cost 12x more and take 3x longer running time
- Worse for recurring jobs – (40% of analytics jobs)
- Best cloud configuration – complex task
  - High accuracy, low overhead, and good adaptivity
Existing work

- Coordinate descent on each resource one at a time
  - Not accurate – resources can be dropped early
- Modelling
  - Not adaptive
  - Ernest – performance model, but tightly bound to the particular structure of ML jobs
- Random search
  - High overhead
- Exhaustive search
  - Long running time
Key idea

- *Just accurate enough* system → near-optimal configurations
- Tolerate inaccuracy → low overhead and good adaptivity
CherryPick

- Bayesian Optimization
  - Black-box modelling – adaptivity
  - Modelling for ranking configurations – good enough accuracy
  - Interactive searching – low overhead
Bayesian Optimisation

- Prior function
  - Black box modelling
  - Confidence interval
- Acquisition function
  - Ranks and chooses the next configuration
  - Calculates expected improvement based on prior function
Further customizations

• Stopping condition – ensures that search is not stopped too soon
• Starting points – give the Bayesian optimisation engine an estimate about the shape of the cost model
• Normalise and discretise most features – reduce the search space
CherryPick 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)

Step-4
Confident that we find the best configuration?

Step-5
End
Implementation

- Search controller
- Cloud Monitor
- Bayesian Optimization Engine
- Cloud Controller
Evaluation

• TPC-DS, TPC-H, TeraSort, The SparkReg, SparkKm
• 66 cloud configurations on Amazon EC2
• Exhaustive search – 6-9 times more search cost and 5-9.5 times more running time
• More stable than coordinate descent
• Better configurations with more stability compared to random search
• Lower search cost and time compared to Ernest with similar running time.
Review

- Shows a significant improvement in search cost and running time compared to existing methods
- 45-90% chance to find optimal configurations – seems quite broad
- The paper does not discuss worst cases where near-optimal solution is never found.
Since publication

- 237 citations
- State of the art at the time
- Scout – aims to address fragility of methods like CherryPick
- PARIS – user defined goals for performance-cost trade-off
Questions?