BoGraph: Structured Bayesian Optimization From Logs for Expensive Systems with Many Parameters

- Sami Alabed
- Eiko Yoneki

- University of Cambridge, UK
- The Alan Turing Institute, UK
The landscape of computer systems

Databases
- MySQL
- PostgreSQL

Simulators of computer design
- gem5

Data processors
- Spark
- Apache Storm
- Apache Kafka
There are many parameters in a typical computer system

Need auto-tuner that scales to large number of parameters
Tune to meet system objective (e.g., latency)
Evaluation takes a long time and is expensive to run.

Methods requiring many evaluations are ill-suited for this task. E.g., Reinforcement Learning, Hill-Climbing, Evolution strategy, Random.

Benchmarks:
- TPC-H Factor 20 (~120 min)
- MachSuite (~30 min)
- YCSB (~10-30 min)
- RocksDB Bench (~15 min-24 hour)
Parameters depend on each other

Traditional dimensionality reduction methods are ineffective
E.g., Principal component analysis (PCA), Factor Analysis (FA)
Designing expert model is difficult and time consuming

Hand designing custom model of the whole system is complicated
E.g., Bespoke Structured Bayesian Optimization and Causal Bayesian Optimization

```python
class BayesianRegression(PyroModule):
    def __init__(self, in_features, out_features):
        super().__init__()
        self.linear = PyroModule(nn.Linear)(in_features, out_features)
        self.linear.weight = PyroSample(dist.Normal(0.0, 1.0).expand([out_features, in_features])
        self.linear.bias = PyroSample(dist.Normal(0.0, 10.0).expand([out_features]).to_event(1))

    def forward(self, x, y=None):
        sigma = pyro.sample("sigma", dist.Uniform(0.0, 10.0))
        mean = self.linear(x).squeeze(-1)
        with pyro.plate("data", x.shape[0]):
            obs = pyro.sample("obs", dist.Normal(mean, sigma), obs=y)
        return mean
```
BoGraph is an efficient auto-tuner that leverages both expert knowledge and logs.
BoGraph is an efficient auto-tuner that leverages both expert knowledge and logs.
BoGraph is an efficient auto-tuner that leverages both expert knowledge and logs.
BoGraph is an efficient auto-tuner that leverages both expert knowledge and logs.
BoGraph is an efficient auto-tuner that leverages both expert knowledge and logs.
BoGraph is an efficient auto-tuner that leverages both expert knowledge and logs.
Bayesian Optimization

Real System

Observations / parameters

Model

Builds knowledge of the system

Optimizer

Finds the "best" parameters

Bayesian Optimization

BO reduces the number of communications with the real system
Context reduces the dimensionality of the problem

The maximum dimension is 3

\[ P(Y|Z1, Z2) \]
\[ P(Z1|X1, X2) \]
\[ P(Z2|X3, X4, X5) \]

The maximum dimension is 5

\[ P(Y|X1, X2, X3, X4, X5) \]
BoGraph learns the context automatically from logs and expert knowledge

Parse
Prune
Group
Standardize

Learn Structure (Causal learning)

Add expert knowledge (Optional)
System logs and metrics provide insight into what impacts the objective.

<table>
<thead>
<tr>
<th>System logs</th>
<th>Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td># number of prefetches that crossed the page</td>
<td># number of prefetches that crossed the page</td>
</tr>
<tr>
<td>system.cpu.icache.prefetcher.pfSpanPage 5852</td>
<td>pg_stat_database.blks_hit 5055</td>
</tr>
<tr>
<td># Cycle average of tags in use</td>
<td>pg_stat_database.tup_returned 2164</td>
</tr>
<tr>
<td>system.l2.tags.tagsinuse 7674.85</td>
<td>pg_stat_database.blks_write_time 0.0</td>
</tr>
<tr>
<td># Total number of bytes read from DRAM</td>
<td>pg_stat_bgwriter.checkpoints_timed 0.3</td>
</tr>
<tr>
<td>system.mem_ctrls.bytesReadDRAM 3151</td>
<td></td>
</tr>
<tr>
<td>... 1000 additional metrics</td>
<td>... 200 additional metrics</td>
</tr>
</tbody>
</table>

Logs are often used to monitor the system's health; BoGraph leverages them to contextualize what is impacting the system objective.
Logs need to be processed: parsed, pruned, standardized, and grouped

Remove uninformative (low variance) group related metrics (manual or automatic) and standardize the result (avoid skew)
Causal structure learning and injecting expert’s knowledge

Learning the causal structure from the processed logs.

Causal structure learning using NoTears[1]

Statistical dependency graph of components

An informative blueprint to build a probabilistic DAG
Taking a structure and mapping it to a probabilistic graph

For each node, we approximate it using a Gaussian Process or expert’s model.
Structured Bayesian Optimization of BoGraph’s DAG

Complicated posterior: not possible to condition on parent’s full posterior

Using BoTorch’s [2] quasi-acquisition function to optimize over DAG’s samples

Instead, using a sampler to generate many samples to approximate the possible results of the posterior in the model

Evaluation setup

• Optimized the design of system accelerator gem5-Aladdin
• Tuned 20 parameters that have $2^{64}$ unique design combination
• Machsuite benchmark (mix of data and compute workloads)
• Optimizing Energy-Delay-Product (EDP)

\[
EDP = energy \times latency^2
\]
BoGraph finds better optimization than any other auto-tuner for all the tasks.

After 100 evaluation steps, Higher EDP improvement is better. The scale is an X-factor improvement from the default.
The identified structure speeds up discovering optimal configurations.

BoGraph converges faster and find better configuration than other auto-tuners.
BoGraph can be applied to a variety of other systems

**Databases**
- MySQL
- PostgreSQL

**Stream Processor**
- Apache Storm
- Spark
- Kafka

Your system?
In summary, BoGraph’s pipeline simplifies using structured Bayesian Optimization to optimize the system in the fewest evaluations.

- Logs contain useful information to decompose the system.
- Structure learning combined with pre-defined expert knowledge leads to fast convergence.

Questions?
Email: sa894@cam.ac.uk