CAPSANDRUNS: An Improved Method for Approximately Optimal Algorithm Configuration

Gellért Weisz  András György  Csaba Szepesvári

DeepMind

ICML 2019
Problem: find good parameter settings (configurations) for general purpose solvers.
   - No structure assumed over the parameter space.
Introduction

- Problem: find good parameter settings (configurations) for general purpose solvers.
  - No structure assumed over the parameter space.

- Zillions of practical algorithms ⇔ Little theory
  Want theoretical guarantees on the runtime of
  - the chosen configuration; and
  - the configuration process.
Introduction

- **Problem:** find good parameter settings (configurations) for general purpose solvers.
  - No structure assumed over the parameter space.

- **Zillions of practical algorithms ⇔ Little theory**
  - Want theoretical guarantees on the runtime of
    - the chosen configuration; and
    - the configuration process.

- **Goal:** find a near-optimal configuration solving $1 - \delta$ fraction of the problems in the least expected time.
  - Since some instances ($\delta$ fraction) are hopelessly hard; don’t want to solve those.
Problem formulation

Given: \( n \) configurations, distribution \( \Gamma \) of problem instances.
Problem formulation

Given: $n$ configurations, distribution $\Gamma$ of problem instances.
Problem formulation

Given: \( n \) configurations, distribution \( \Gamma \) of problem instances.

- Runtime of configuration \( i \) is \((\delta, \beta)\)-optimal if \( R(\delta)(i) \leq (1 + \delta) \frac{OPT}{2} \).

Previous work (Kleinberg et al., 2011; Weisz et al., 2011): no capping of \( OPT \) using \( OPT_0 \) instead of \( \frac{OPT}{2} \).
Problem formulation

Given: $n$ configurations, distribution $\Gamma$ of problem instances.

Runtime of the optimal capped configuration:

$$\text{OPT}_\delta = \min_i R^\delta(i)$$
Problem formulation

Given: \( n \) configurations, distribution \( \Gamma \) of problem instances.

Runtime of configuration \( i \) is \((\varepsilon, \delta)\)-optimal if

\[ R^\delta(i) \leq (1 + \varepsilon) \text{OPT}_{\delta/2}. \]

Runtime of the optimal capped configuration:

\[ \text{OPT}_{\delta} = \min_i R^\delta(i) \]
Problem formulation

Given: \( n \) configurations, distribution \( \Gamma \) of problem instances.

Configuration \( i \) is \((\varepsilon, \delta)\)-optimal if \( R^\delta(i) \leq (1 + \varepsilon)\text{OPT}_{\delta/2} \).

Runtime of the optimal capped configuration:

\[
\text{OPT}_\delta = \min_i R^\delta(i)
\]

Previous work (Kleinberg et al., 2017; Weisz et al., 2018): no capping of \( \text{OPT} \): using \( \text{OPT}_0 \) instead of \( \text{OPT}_{\delta/2} \).
Guarantees

- Previous work (Kleinberg et al., 2017; Weisz et al., 2018): with high probability,
  (i) the algorithm finds an \((\varepsilon, \delta)\)-optimal configuration;
  (ii) with total work
  \[
  \tilde{O} \left( \text{OPT}_0 \frac{n}{\varepsilon^2 \delta} \right).
  \]

- Worst case lower bound: \(\Omega \left( \text{OPT}_0 \frac{n}{\varepsilon^2 \delta} \right)\) (Kleinberg et al., 2017).
Guarantees

- Previous work (Kleinberg et al., 2017; Weisz et al., 2018): with high probability,
  (i) the algorithm finds an \((\varepsilon, \delta)\)-optimal configuration;
  (ii) with total work
  \[
  \tilde{O}\left(\text{OPT}_0 \frac{n}{\varepsilon^2 \delta}\right).
  \]

  ★ Worst case lower bound: \(\Omega\left(\text{OPT}_0 \frac{n}{\varepsilon^2 \delta}\right)\) (Kleinberg et al., 2017).

- This work with high probability finds an \((\varepsilon, \delta)\)-optimal configuration:
  ▶ Total work (simplified version):
  \[
  \tilde{O}\left(n\text{OPT}_{\delta/2} \left(\frac{1}{\delta} + \max\left\{\frac{\sigma^2}{\max\{\varepsilon^2, \Delta^2\}}, \frac{r}{\max\{\varepsilon, \Delta\}}\right\}\right)\right),
  \]
  where
  ★ \(\Delta \sim\) gap between the best two configurations
  ★ \(\sigma^2 \sim\) runtime variances,
  ★ \(r \sim\) range of runtimes.
CAPSANDRUNS algorithm

\[ \tilde{O}\left(n\OPT_{\delta/2}\left(\frac{1}{\delta} + \max\left\{\frac{\sigma^2}{\max\{\varepsilon^2, \Delta^2\}}, \frac{r}{\max\{\varepsilon, \Delta\}}\right\}\right)\right), \]

Phase I:
- For each configuration \( i \) find a runtime cap \( \tau_i \)
  - that solves between \( 1 - \delta \) and \( 1 - \delta/2 \) fraction of problem instances,
  - not wasting time on bad configurations.
CAPSANDRUNS algorithm

\[
\tilde{O} \left( n \text{OPT} \delta/2 \left( \frac{1}{\delta} + \max \left\{ \frac{\sigma^2}{\max\{\varepsilon^2, \Delta^2\}}, \frac{r}{\max\{\varepsilon, \Delta\}} \right\} \right) \right),
\]

Phase I:
- For each configuration \( i \) find a runtime cap \( \tau_i \)
  - that solves between \( 1 - \delta \) and \( 1 - \delta/2 \) fraction of problem instances,
  - not wasting time on bad configurations.

Phase II:
- Run a Bernstein race (Mnih et al., 2008) over the configurations.
  - Evaluate configurations in parallel, giving preference to better ones, shrinking confidence regions using Bernstein’s inequality.
Experiments

Configuring SAT solvers (Weisz et al., 2018):

Factor of total runtime improvement from LEAPSANDBOUNDS to CAPSANDRUNS for various values of $\varepsilon$ and $\delta$.

<table>
<thead>
<tr>
<th>STRUCTURED PROCRASTINATION</th>
<th>LEAPSANDBOUNDS</th>
<th>CAPSANDRUNS</th>
</tr>
</thead>
<tbody>
<tr>
<td>20643 (±5) days</td>
<td>1451 (±83) days</td>
<td>586 (±7) days</td>
</tr>
</tbody>
</table>
Thank you!

Poster #201