Procrastinating with Confidence: Near-Optimal, Anytime, Adaptive Algorithm Configuration

Robert Kleinberg, Kevin Leyton-Brown, Brendan Lucier, Devon Graham

INRODUCTION
- Given a parameterized algorithm \( \mathcal{A} \), Algorithm Configuration is the problem of finding "good" values for the parameters of \( \mathcal{A} \).
- A "good" setting of \( \mathcal{A} \)'s parameters is one that performs well on some underlying distribution of input instances.
- We present Structured Procrastination with Confidence (SPC), an anytime algorithm configuration procedure that is optimal up to log factors.
- SPC is also adaptive, in that its runtime guarantee is better on easier problem instances.

PROBLEM SETUP
- \( \mathcal{A} \): A parameterized algorithm
- \( N \): Set of potential parameter configurations for \( \mathcal{A} \)
- \( M \): Set of potential inputs to \( \mathcal{A} \)
- \( \Gamma \): Distribution over elements of \( M \)
- \( R(i,j) \): Runtime of \( \mathcal{A} \) with configuration \( i \in N \) on input \( j \in M \).
- \( R(i) = E_{j \sim \Gamma}[R(i,j)] \): Expected runtime of \( i \)
- \( R_d(i) = E_{j \sim \Gamma}[\min\{R(i,j), \theta\}] \): Expected capped runtime of \( i \)
- \( OPT = \min_{i \in N} E_{j \sim \Gamma}[R(i,j)] \): Expected runtime of the optimal configuration

(\( \epsilon, \delta \))-Optimality
- Ideally we would find a configuration \( i^* \) close to the best one: \( R(i^*) \leq (1 - \epsilon)OPT \).
- If runtime distribution is heavy tailed, this is prohibitively hard.
- Instead, we seek an \( \epsilon \)-approximate configuration over the fastest \( (1 - \delta) \) fraction of input instances:
  - Goal: find a configuration \( i^* \) s.t.
  \[ R_d(i^*) \leq (1 + \epsilon)OPT \quad \text{and} \quad \Pr_{j \sim \Gamma} [R(i,j) > \theta] \leq \delta \]
  for some \( \theta \).

EXPERIMENTAL RESULTS
- SPC is able to guarantee an \( (\epsilon, \delta) \)-optimal configuration much sooner than other methods:

RUNTIME GUARANTEE
- \( S \) is the set of \( (\epsilon, \delta) \)-optimal configurations.
- Each \( i \notin S \) is \( (\epsilon_i, \delta_i) \)-suboptimal, with \( \epsilon_i \geq \epsilon \) and \( \delta_i \geq \delta \).
- Define \( B(\epsilon, \delta, t) = e^{-\delta t} \log(t \log(1/\delta)) \).
- Then if the time spent running SPC is
  \[ \Omega \left( R(i^*) \left( |S| B(\epsilon, \delta, t) + \sum_{i \notin S} B(\epsilon_i, \delta_i, t) \right) \right) \]
  then SPC will return an \( (\epsilon, \delta) \)-optimal configuration.

RELATED WORK