Kraken: An Iterative Partitioning Approach for Per-Instance Algorithm Configuration

André Biedenkapp, Marius Lindauer, Frank Hutter
University of Freiburg

Holger Hoos
Leiden University

Kevin Leyton-Brown
University of British Columbia

Albert-Ludwigs-Universität Freiburg

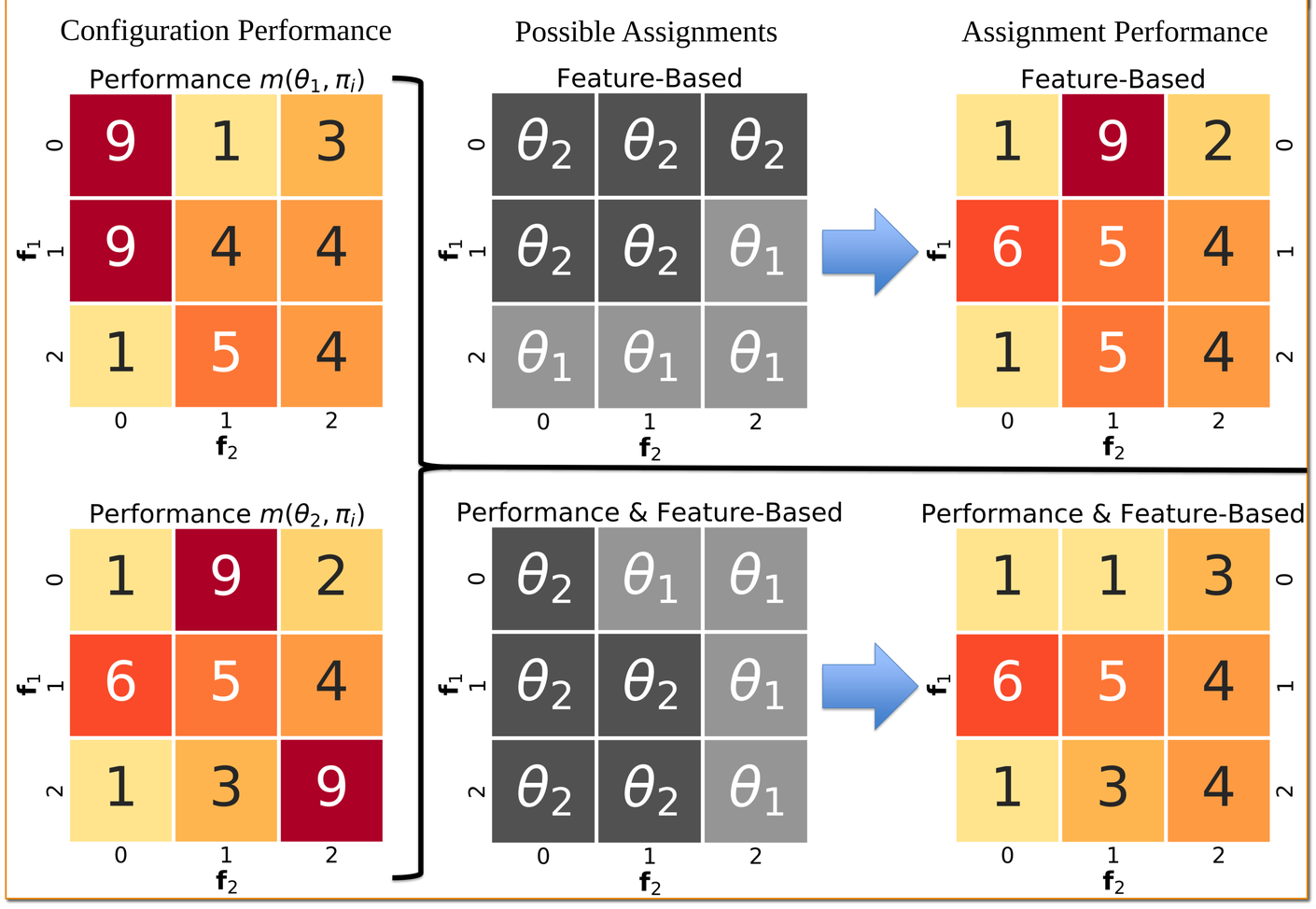
In a Nutshell

- **Algorithm configuration (AC)**, often necessary to achieve peak performance over a set of instances (e.g. AI Planning, SAT and ASP)
- **Algorithm selection (AS)**, predict most promising solver for an instance at hand
- **Per-instance algorithm configuration (PIAC)** combines the two approaches
- The problem of PIAC is to **learn a selector** that is able to *select a configuration for a given instance* such that the configuration performs best on the instance

Related Work

- **Stochastic Offline Programming** [Malitsky and Sellmann ICTAI 2009]: iterative PIAC system, modification of distance metric in feature space, partition instances into subsets
- **Hydra** [Lin et al. AAAI 2010]: iterative PIAC system, sequentially adds complementary configurations to portfolio, purely performance based, instances not partitioned
- **Instance Specific Algorithm Configuration** [Kadioglu et al. ECAI 2010]: one-shot PIAC system, g-means clustering, partition instances into subsets

Taking Performance into Account



Kraken

1. Profile Expected Improvement [Ginsbourger et al. JUQ 2014, Bossek et al. GECCO 2015] to explore the target performance
2. Cost-sensitive hierarchical clustering [Malitsky et al. IJCAI 2013] to determine partitions of the instances based on EPM predictions
3. Run an AC tool on the found partitions to determine well performing configuration
4. Iterate steps 2 and 3 until a predefined budget is exhausted

Cost-Sensitive Hierarchical Clustering

- Initially all instances belong to the same partition
- For each partition:
  - Draw R random splits
  - Search for a well performing configuration on each split
  - Compute the loss for a split as follows:

$$l(\mathbf{h}, \theta_{\mathbf{h}}) := \sum_{\pi \in \mathbf{h}} \hat{m}(\theta_{\mathbf{h}}, \pi) - \min_{\theta' \in \mathbf{\Theta}} \hat{m}(\theta', \pi)$$

$$\text{where } \theta_{\mathbf{h}} = \arg\min_{\theta \in \mathbf{\Theta}} \sum_{\pi \in \mathbf{h}} \hat{m}(\theta, \pi)$$

$\mathbf{h}$: Partition θ: Configuration $\mathbf{\Theta}$: Configuration Space
π: Instance $\hat{m}$: Empirical performance model

  - The best split with the lowest loss is kept

Configuring SATenstein on Handcrafted and Industrial