Automatic Selection of Tuning Plugins in PTF Using Machine Learning

Robert Mijaković (mijakovic@lrz.de) Leibniz Supercomputing Centre (LRZ)
Michael Gerndt (gerndt@in.tum.de) Technische Universität München (TUM)
• Large-scale machines help solve some of the greatest challenges humanity faces.
• Writing an HPC application for such system requires extensive knowledge of particular scientific subjects and a good knowledge of the underlying system.
• Increased complexity of systems, e.g. accelerators.
• Systems change every few years.
• New system requires reoptimizing the code.
The Periscope Tuning Framework

- Autotuner that combines performance analysis and tuning
- Tune codes to improve performance and energy efficiency
- Plugin creates a sequence of tuning scenarios
  - the variant,
  - the context for which the objective gets measured, and
  - the objective
- Searches for the optimal scenario: combination of parameters and their values
- Tuning plugins optimize many different aspects:
  - Compiler Flags Selection (CFS),
  - MPI Parameters,
  - Dynamic Voltage and Frequency Scaling (DVFS), etc.
Adaptive Sequence of Tuning Plugins

- Tuning with plugins is time consuming
- Predicting tuning potential from program signature
- Traditional methods depend on white-box signatures
  - More in-depth knowledge, better predictions
  - Depend on tuning plugin

- Alternative: Black-box signature
  - Prediction based on the sensitivity of the application for the tuning plugin
  - Generic for all tuning plugins
Approach - Overview

**Training**

- Historical Tuning Results
  - Signature Scenarios Identification
  - Retrieve Historical Tuning Results
  - Signatures and Tuning Results
  - Predictor Training

**Tuning**

- Application
  - Create Signature
  - Signature
  - Prediction of Tuning Result
  - High Tuning Potential
  - Apply Tuning Plugin

**Prediction**

- Prediction Model
Example: CFS Plugin

Signature Scenarios Identification

Build Signature

Prefetch

Vectorization

Sensitivity

6.5 significant potential
Signature Scenarios Identification

**Challenge:** Find minimal set of signature scenarios providing good predictions for the existing applications

**Used methods:**

- Information Gain:
  - Entropy(parent) – average entropy(children)

- k-Medoids clustering
Optimal Number of Signature Scenarios: The Elbow Method
Predictor

- **Three modes of work:**
  - Classification: label program - sufficient or insufficient improvement
  - Regression: predicts what is the improvement of the objective, e.g., 50%.
  - Regression-based Classification: e.g., 50% improvement > user expected improvement = (in)sufficient

- **Support Vector Machine**
  - Challenge: Which kernel to select?

- **Selection of kernels:**
  - Linear (no kernel)
  - Polynomial
  - Radial Basis Function (RBF)
  - Sigmoid
Historical Tuning Results

• Compiler Flags Selection Plugin:
  • 52 applications
  • 500 scenarios
  • 26000 points

• MPI Parameters Plugin:
  • 35 applications
  • 500 scenarios
  • 17500 points
Classification: Quality of Predictions

- Compiler Flags Selection plugin
- MPI Parameters plugin
Saved Time: CFS

<table>
<thead>
<tr>
<th>Settings</th>
<th>Threshold</th>
<th>Kernel</th>
<th>Signature</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.18%</td>
<td>Poly.</td>
<td>Info. Gain</td>
<td>Classificat.</td>
</tr>
<tr>
<td>2</td>
<td>7.92%</td>
<td>RBF</td>
<td>Info. Gain</td>
<td>Regression</td>
</tr>
<tr>
<td>3</td>
<td>2.14%</td>
<td>RBF</td>
<td>Info. Gain</td>
<td>Regression</td>
</tr>
<tr>
<td>4</td>
<td>26.42%</td>
<td>RBF</td>
<td>k-Medoids</td>
<td>Regression</td>
</tr>
<tr>
<td>5</td>
<td>23.94%</td>
<td>RBF</td>
<td>Info. Gain</td>
<td>Regression</td>
</tr>
<tr>
<td>6</td>
<td>33.35%</td>
<td>RBF</td>
<td>Info. Gain</td>
<td>Regression</td>
</tr>
</tbody>
</table>
Saved Time: MPI Parameters

<table>
<thead>
<tr>
<th>Settings</th>
<th>Threshold</th>
<th>Kernel</th>
<th>Signature</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3.55%</td>
<td>Sigmoid</td>
<td>Info. Gain</td>
<td>Classificat.</td>
</tr>
<tr>
<td>2</td>
<td>36.33%</td>
<td>RBF</td>
<td>Info. Gain</td>
<td>Classificat.</td>
</tr>
<tr>
<td>3</td>
<td>57.76%</td>
<td>RBF</td>
<td>Info. Gain</td>
<td>Regression</td>
</tr>
<tr>
<td>4</td>
<td>118.7%</td>
<td>RBF</td>
<td>Info. Gain</td>
<td>Regression</td>
</tr>
<tr>
<td>5</td>
<td>54.20%</td>
<td>RBF</td>
<td>Info. Gain</td>
<td>Regression</td>
</tr>
<tr>
<td>6</td>
<td>479.26%</td>
<td>RBF</td>
<td>Info. Gain</td>
<td>Regression</td>
</tr>
</tbody>
</table>

![Bar chart showing time saved for different settings and scenarios]
Regression Quality

- Compiler Flags Selection plugin
- MPI Parameters plugin
Conclusion

• Black-box signatures work well

• Information gain is superior to k-Medoids
  • Better predictions from smaller signature

Classification:
• Best kernel: RBF
• Short training time
• CFS: High F1-score between 0.90 and 1, average 0.96
• MPI Params: F1-score between 0.83 and 1, average 0.94
• Regression-based classification is superior
• Significant time saved by meta-plugin predictions

Regression:
• Best kernel: Polynomial or Linear
• Polynomial: best quality high training time
• Linear: slightly worse quality much shorter training time
• CFS: Average value = 2.4, Stand. error = 11.85%
• MPI Params: Average value = 5.2, Stand. error = 7.81%
Questions?
Send an email to: robert.mijakovic@lrz.de