Evolution of Algorithm Portfolio for Solving Strategies

Alice Tarzariol

Università degli Studi di Udine

June 21, 2019
1. A Scenario

2. Previous Approach

3. Algorithm Portfolio: Selection and Scheduling

4. Algorithm Configuration

5. Combination
Consider a “difficult” problem...

- Many frequent instances that follow a distribution.
- E.g., finding a delivery path that minimizes the time/distance of its tour ($TSP$).
Consider a “difficult” problem...

- Many frequent instances that follow a distribution.
- E.g., finding a delivery path that minimizes the time/distance of its tour (TSP).
- Models:
  - Answer Set Programming
  - Constraint Programming
  - Boolean Satisfiability
  - …
Consider a “difficult” problem...

- Many frequent instances that follow a distribution.
- E.g., finding a delivery path that minimizes the time/distance of its tour (*TSP*).
- Models:
  - Answer Set Programming
  - Constraint Programming
  - Boolean Satisfiability
  - …
- Run a solver to explore the searching space and find a solution.
Solver Selection

- Given a portfolio of solvers $\mathcal{P} = \{A, B, C\}$ suitable for this problem.
- And a new problem instance $x$, drawn from the previous distribution.
- Which solver should be run on $x$?
Solver Selection

- Given a portfolio of solvers $P = \{A, B, C\}$ suitable for this problem.
- And a new problem instance $x$, drawn from the previous distribution.
- Which solver should be run on $x$?

- Knowledge obtained from previous problem instances.
- Solved by elements in $P$ and measured according to a certain performance metric.
Previous Approach
Winner Takes All

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>$i_1$</td>
<td>55</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>$i_2$</td>
<td>40</td>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td>$i_3$</td>
<td>6</td>
<td>100</td>
<td>150</td>
</tr>
<tr>
<td>$i_4$</td>
<td>4</td>
<td>70</td>
<td>109</td>
</tr>
<tr>
<td>$i_5$</td>
<td>45</td>
<td>5</td>
<td>150</td>
</tr>
<tr>
<td>avg</td>
<td>30</td>
<td>56.6</td>
<td>83</td>
</tr>
</tbody>
</table>

Simplifications

- Instances: Larger dataset.
- Values: Approximate performance of algorithms.
Previous Approach

Winner Takes All

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>$i_1$</td>
<td>55</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>$i_2$</td>
<td>40</td>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td>$i_3$</td>
<td>6</td>
<td>100</td>
<td>150</td>
</tr>
<tr>
<td>$i_4$</td>
<td>4</td>
<td>70</td>
<td>109</td>
</tr>
<tr>
<td>$i_5$</td>
<td>45</td>
<td>5</td>
<td>150</td>
</tr>
<tr>
<td>avg</td>
<td>30</td>
<td>56.6</td>
<td>83</td>
</tr>
</tbody>
</table>

Simplifications
- Instances: Larger dataset.
- Values: Approximate performance of algorithms.
Algorithm Portfolio: Selection and Scheduling
## Portfolio Approach

### Solver A

### Solver B

### Solver C

### Portfolio Algorithm

### Features Extraction

**Static and/or Semi-static and/or Dynamic.**

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$A$</th>
<th>$B$</th>
<th>$C$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a$</td>
<td>$b$</td>
<td>55</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>$a$</td>
<td>$b$</td>
<td>40</td>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td>$c$</td>
<td>$d$</td>
<td>6</td>
<td>100</td>
<td>150</td>
</tr>
<tr>
<td>$a$</td>
<td>$d$</td>
<td>4</td>
<td>70</td>
<td>109</td>
</tr>
<tr>
<td>$c$</td>
<td>$b$</td>
<td>45</td>
<td>5</td>
<td>150</td>
</tr>
</tbody>
</table>
Portfolio Approach: Algorithm Selection

Features Extraction

Static and/or Semi-static and/or Dynamic.

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a$</td>
<td>$b$</td>
<td>55</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>$a$</td>
<td>$b$</td>
<td>40</td>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td>$c$</td>
<td>$d$</td>
<td>6</td>
<td>100</td>
<td>150</td>
</tr>
<tr>
<td>$a$</td>
<td>$d$</td>
<td>4</td>
<td>70</td>
<td>109</td>
</tr>
<tr>
<td>$c$</td>
<td>$b$</td>
<td>45</td>
<td>5</td>
<td>150</td>
</tr>
</tbody>
</table>
Algorithm Selection

Solver A

Solver B

Solver C

Problem Instance

FEATURE EXTRACTION

Algorithm Selection

Previous Cases

Solver B

Portfolio Algorithm
Algorithm Selection

Solver A

Solver B

Solver C

Portfolio Algorithm

Problem Instance $y$

Feature Extraction

Algorithm Selection

Previous Cases

Solver C
Algorithm Scheduling

Solver A

Solver B

Solver C

Algorithm Scheduling

Problem Instance $\times$

Feature Extraction

Previous Cases

A C B

Portfolio Algorithm
Considerations

- Algorithm Scheduling is more robust.
- Useful if many instances are solved within brief time by different solvers.
- **Parallel Portfolio**: scheduling with multi-core architectures.
Considerations

- Algorithm Scheduling is more robust.
- Useful if many instances are solved within brief time by different solvers.
- Parallel Portfolio: scheduling with multi-core architectures.

### Algorithm Selection
- SATzilla
- ME-ASP
- Claspfolio

### Algorithm Scheduling
- CPHydra
- SUNNY (parallel)
- ASPEED (parallel)

- Both approaches: Claspfolio 2.
### Implementations

- Several approaches: *Regression/Classification, Eager/Lazy*, etc.
- Further components and statistical techniques:
  - *Pre-Solvers*
  - *Backup Solver*
  - *Subset Portfolio Selection*
  - etc…
Considerations (Cont.)

Implementations

- Several approaches: *Regression/Classification, Eager/Lazy*, etc.
- Further components and statistical techniques:
  - *Pre-Solvers*
  - *Backup Solver*
  - *Subset Portfolio Selection*
  - etc...

Portfolio Design

- Outperform the Single Best Solver even if sub-optimal solution.
- The portfolio solvers must present complementary strengths.
Algorithm Configuration
Before defining a portfolio of solvers...

• How we can find a solver that “performs well” for the considered distribution?
Before defining a portfolio of solvers...

- How we can find a solver that “performs well” for the considered distribution?
- Many possibilities: e.g. local search techniques, DPLL, CDCL, etc.
• Beside choosing the solving “core”, the hyper-parameters of solvers further guide the behavior of algorithms.
Hyper-parameters

• Beside choosing the solving “core”, the **hyper-parameters** of solvers further guide the behavior of algorithms.

• Huge configuration space: mix of continuous, integer and categorical domains.
Hyper-parameters

• Beside choosing the solving “core”, the hyper-parameters of solvers further guide the behavior of algorithms.

• Huge configuration space: mix of continuous, integer and categorical domains.
• Manually tuned by domain experts: considerable time and efforts.
• Usually optimal configurations are not considered.
Algorithm Configuration

- Find automatically a configuration that performs optimally over the problem instance distribution.

Empirical studies shown that it outperform hyperparameters tuning.
Algorithm Configuration

- Find automatically a configuration that performs optimally over the problem instance distribution.

- Empirical studies shown that it outperform hyperparameters tuning.
• Find automatically a configuration that performs optimally over the problem instance distribution.

• Empirical studies shown that it outperform hyperparameters tuning.
Combination
- Algorithm Selection that does not require a pre-defined portfolio of algorithm, but just a parametrized solver $A$. 
Hydra

- Algorithm Selection that does not require a pre-defined portfolio of algorithms, but just a parametrized solver $A$.

- Algorithm Configuration extends a portfolio of $A$ configurations.

![Diagram showing the process of Hydra with nodes for Algorithm Configurator, AS Model, Algorithm Selector, and Algorithm Portfolio.]
Hydra

- Algorithm Selection that does not require a pre-defined portfolio of algorithm, but just a parametrized solver $A$.

- Algorithm Configuration extends a portfolio of $A$ configurations.
- Then it applies the Algorithm Selector to learn an AS Model.
Hydra

- Algorithm Selection that does not require a pre-defined portfolio of algorithm, but just a parametrized solver $A$.

  - Algorithm Configuration extends a portfolio of $A$ configurations.
  - Then it applies the Algorithm Selector to learn an AS Model.
  - Continue until a stopping criterion is met.
**AutoFolio**

- Algorithm Configuration over Algorithm Selection systems with hyperparameters.
- Define the best setting that will exploit the portfolio solvers.

**Algorithm Configurator**

**Algorithm Selector**

**Portfolio**

- **Solver A**
- **Solver B**
- **Solver C**
• Algorithm Portfolio allows to exploit the whole set of solvers.
• But knowledge is required to find a suitable portfolio.
• Algorithm Configuration extends these systems.
Algorithm Portfolio allows to exploit the whole set of solvers.
But knowledge is required to find a suitable portfolio.
Algorithm Configuration extends these systems.
Outstanding results: portfolio systems are not allowed to participate to “standard” solving competitions.
ASlib: train and evaluate new systems.
Evolution of Algorithm Portfolio for Solving Strategies

• Algorithm Portfolio allows to exploit the whole set of solvers.
• But knowledge is required to find a suitable portfolio.
• Algorithm Configuration extends these systems.
• Outstanding results: portfolio systems are not allowed to participate to “standard” solving competitions.
• ASlib: train and evaluate new systems.

Questions?