Data Science meets Optimization

ODS 2017
EURO Plenary
Patrick De Causmaecker
EWG/DSO EURO Working Group
CODeS
KU Leuven/ KULAK
Science Paradigms

- Thousand years ago: science was **empirical** describing natural phenomena
- Last few hundred years: **theoretical** branch using models, generalizations
- Last few decades: a **computational** branch simulating complex phenomena
- Today: **data exploration** (eScience) unify theory, experiment, and simulation
  - Data captured by instruments or generated by simulator
  - Processed by software
  - Information/knowledge stored in computer
  - Scientist analyzes database/files using data management and statistics
Two Quotes

Reports on insights found in data aren't that useful to business.
The role of data science is to enable data based decision making
(Jean Francois Puget)

‘Data are not taken for museum purposes; they are taken as a basis for doing something. If nothing is to be done with the data, then there is no use in collecting any. The ultimate purpose of taking data is to provide a basis for action or a recommendation for action.’

W. Edwards Deming, 1942
A particular problem in High Energy Physics: $gg \rightarrow gg \ldots$ (since 1983)

- $4g$ (4 diagrams)
- $5g$ (25 diagrams)
- $6g$ (220 diagrams)
Heuristics

“1960’s heuristics”

PORTFOLIO SELECTION: A HEURISTIC APPROACH*

Geoffrey P. Clarkson and Allan H. Meltzer
Carnegie Institute of Technology

I. Introduction

The problem of selecting a portfolio can be divided into two components: (1) the analysis of individual securities and (2) the selection of a portfolio or group of securities based on the previous analysis. Up to now, the majority of writers have focused on the first part of the problem and have developed several, well-accepted methods of analysis. Little attention has been paid to the second phase of the problem. It is to this second part of the portfolio selection process that this paper is principally devoted.

ODS 2017, Sorrento
Automated algorithm construction

BUSH JONES (University of Michigan-Dearborn, 4901 Evergreen Road, Dearborn, MI. 48128 (U.S.A.))


The fulltext of this document has been downloaded 24 times since 2006

The use of a computer to automatically generate original algorithms is discussed in this paper. The problems in the construction of a computer algorithm finder are discussed. An actual computer algorithm finder is described and demonstrated.

Cfr talks of Alberto Franzin and Frederico Pagnozzi on the application of irace for Algorithm Construction

GA’s were suggested by Alan Turing 1950 and first implemented by Baricelli in 1954…

ODS 2017, Sorrento
An early example: A hyperheuristic framework

Example 1
Data Science FOR Optimization: Automated Algorithm Configuration

Configuring irace using surrogate configuration benchmarks (GECCO 2017, ECOM track)
Configuring irace using surrogate configuration benchmarks

Nguyen Dang ¹, Leslie Pérez Cáceres ², Thomas Stützle ², Patrick De Causmaecker ¹

¹ CODeS, ITEC-imec, KU Leuven, Belgium
² IRIDIA, CoDE, Université Libre de Bruxelles (ULB), Belgium
Offline configuration

- Parameters Definition
  - name
  - type
  - possible values

- Calls with candidate configuration

- Configurator
  - Returns solution cost

- Best configuration to be used

- Software to be tuned
  - Tackles
  - Set of instances
    - 1
    - 2
    - 3
    - 4
    - ...


ODS 2017, Sorrento
Configuring configurators

What about configuring automatically the configurator? … and configuring the configurator of the configurator?

- can be done (example, see (Hutter et al., 2009)), but …
- it is costly and iterating further leads to diminishing returns


ODS 2017, Sorrento
Return solution cost

Target Algorithm

Problem instances

tackle

Call with candidate configurations

Configurator

Return solution cost

Best parameter configuration to be used

Parameters
- Type
- Domain
- Condition

Performance measure
- Maximize solution quality (within given computation time)
- Minimize computation time (to reach optimality)
Problem instances tackle

Target Algorithm

Call with candidate configurations

Configurator

Return solution cost

Parameters
- Type
- Domain
- Conditions

Best parameter configuration to be used

- irace (López-Ibáñez et al, 2011)
- SMAC (Hutter et al, 2011)
- ParamILS (Hutter et al, 2009)
- GGA (Ansótegui et al, 2009)
- SPO (Bartz-Beielstein et al 2005, 2010)
- ...

KU LEUVEN
What about the tuning engine?
What about the tuning engine?
Meta-tuning

- Real benchmarks: extremely expensive
  - An irace run on SPEAR-IBM (budget: 5000 runs) → 2 CPU days
  - A meta-tuning on SPEAR-IBM (budget: 5000 irace runs) → 27.5 CPU years

- Artificial benchmark set
  - Unclear how to generate
  - Unclear how to match characteristics of real configuration tasks

- Surrogate benchmarks
  - A prediction model: configuration x instance → performance value
  - Build on real benchmark data
Surrogate configuration benchmarks

• Meta-tuning becomes computationally feasible
  - An irace run on SPEAR-IBM (budget: 5000 runs)
    → 2 CPU days 5 CPU minutes
  - A meta-tuning on SPEAR-IBM (budget: 5000 irace runs)
    → 27.5 CPU years 7.5 CPU days

• Useful for the development of configurators
  - Study configurator’s parameters
  - Gain insights into configurator’s behaviours
  - Better performing configurators
Content

Motivation

irace’s parameters

Surrogate configuration benchmarks

Configuring irace using surrogate benchmarks

Conclusions
irace: iterated racing

Iteration 1
Parameter configurations

Instance 1
Instance 2
Instance 3
Instance 4
Instance 5

Statistical test

Iteration 2
Sampling model
## irace: parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Domain</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N^{iter}$</td>
<td>int</td>
<td>[1, 50]</td>
<td>$2 + \log_2 n^{\text{params}}$</td>
</tr>
<tr>
<td>$\mu$</td>
<td>int</td>
<td>[1, 20]</td>
<td>5</td>
</tr>
<tr>
<td>$T_{first}$</td>
<td>int</td>
<td>[4, 20]</td>
<td>5</td>
</tr>
<tr>
<td>$N^{min}$</td>
<td>int</td>
<td>[1, 20]</td>
<td>$2 + \log_2 n^{\text{params}}$</td>
</tr>
<tr>
<td>$test_type$</td>
<td>categorical</td>
<td>{F-test, t-test, t-test-holm, t-test-bonferroni}</td>
<td>F-test</td>
</tr>
<tr>
<td>$confidence_level$</td>
<td>real</td>
<td>[0.5, 0.99]</td>
<td>0.95</td>
</tr>
<tr>
<td>$enable_soft_restart$</td>
<td>categorical</td>
<td>{true, false}</td>
<td>true</td>
</tr>
<tr>
<td>$enable_elitist$</td>
<td>categorical</td>
<td>{true, false}</td>
<td>true</td>
</tr>
<tr>
<td>$elitist_instances$</td>
<td>int</td>
<td>[1, 10]</td>
<td>1</td>
</tr>
</tbody>
</table>
### irace: parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Domain</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N^{\text{iter}}$</td>
<td>int</td>
<td>[1, 50]</td>
<td>$2 + \log_2 n^{\text{params}}$</td>
</tr>
<tr>
<td>$\mu$</td>
<td>int</td>
<td>[1, 20]</td>
<td>5</td>
</tr>
<tr>
<td>$T^{\text{first}}$</td>
<td>int</td>
<td>[4, 20]</td>
<td>5</td>
</tr>
<tr>
<td>$N^{\text{min}}$</td>
<td>int</td>
<td>[1, 20]</td>
<td>$2 + \log_2 n^{\text{params}}$</td>
</tr>
<tr>
<td>test_type</td>
<td>categorical</td>
<td>{F-test, t-test, t-test-holm, t-test-bonferroni}</td>
<td>F-test</td>
</tr>
<tr>
<td>confidence_level</td>
<td>real</td>
<td>[0.5, 0.99]</td>
<td>0.95</td>
</tr>
<tr>
<td>enable_soft_restart</td>
<td>categorical</td>
<td>{true, false}</td>
<td>true</td>
</tr>
<tr>
<td>enable_elitist</td>
<td>categorical</td>
<td>{true, false}</td>
<td>true</td>
</tr>
<tr>
<td>elitist_instances</td>
<td>int</td>
<td>[1, 10]</td>
<td>1</td>
</tr>
</tbody>
</table>
irace: parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Domain</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N^{\text{iter}}$</td>
<td>int</td>
<td>[1, 50]</td>
<td>$2 + \log_2 n^{\text{params}}$</td>
</tr>
<tr>
<td>$\mu$</td>
<td>int</td>
<td>[1, 20]</td>
<td>5</td>
</tr>
<tr>
<td>$T_{\text{first}}$</td>
<td>int</td>
<td>[4, 20]</td>
<td>5</td>
</tr>
<tr>
<td>$N^{\text{min}}$</td>
<td>int</td>
<td>[1, 20]</td>
<td>$2 + \log_2 n^{\text{params}}$</td>
</tr>
<tr>
<td>test_type</td>
<td>categorical</td>
<td>{F-test, t-test, t-test-holm, t-test-bonferroni}</td>
<td>F-test</td>
</tr>
<tr>
<td>confidence_level</td>
<td>real</td>
<td>[0.5, 0.99]</td>
<td>0.95</td>
</tr>
<tr>
<td>enable_soft_restart</td>
<td>categorical</td>
<td>{true, false}</td>
<td>true</td>
</tr>
<tr>
<td>enable_elitist</td>
<td>categorical</td>
<td>{true, false}</td>
<td>true</td>
</tr>
<tr>
<td>elitist_instances</td>
<td>int</td>
<td>[1, 10]</td>
<td>1</td>
</tr>
</tbody>
</table>
# irace: parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Domain</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N^{\text{iter}}$</td>
<td>int</td>
<td>$[1, 50]$</td>
<td>$2 + \log_2 n^{\text{params}}$</td>
</tr>
<tr>
<td>$\mu$</td>
<td>int</td>
<td>$[1, 20]$</td>
<td>5</td>
</tr>
<tr>
<td>$T^{\text{first}}$</td>
<td>int</td>
<td>$[4, 20]$</td>
<td>5</td>
</tr>
<tr>
<td>$N^{\text{min}}$</td>
<td>int</td>
<td>$[1, 20]$</td>
<td>$2 + \log_2 n^{\text{params}}$</td>
</tr>
<tr>
<td>test_type</td>
<td>categorical</td>
<td>{F-test, t-test, t-test-holm, t-test-bonferroni}</td>
<td>F-test</td>
</tr>
<tr>
<td>confidence_level</td>
<td>real</td>
<td>$[0.5, 0.99]$</td>
<td>0.95</td>
</tr>
<tr>
<td>enable_soft_restart</td>
<td>categorical</td>
<td>{true, false}</td>
<td>true</td>
</tr>
<tr>
<td>enable_elitist</td>
<td>categorical</td>
<td>{true, false}</td>
<td>true</td>
</tr>
<tr>
<td>elitist_instances</td>
<td>int</td>
<td>$[1, 10]$</td>
<td>1</td>
</tr>
</tbody>
</table>

**Diagram:**

The diagram illustrates the sampling model and iterations, with parameter configurations and statistical tests for different instances.
Motivation
irace’s parameters
**Surrogate configuration benchmarks**
Configuring irace using surrogate benchmarks
Conclusions
## Surrogate algorithm configuration benchmarks

### Empirical Performance Model - EPM (Hutter, Xu, Hoos & Leyton-Brown, 2014)
- Random Forest regression model
- Training: 1000 random configurations $\times$ all problem instances

<table>
<thead>
<tr>
<th>Algorithm</th>
<th># Parameters (int/real/categorical/conditional)</th>
<th>Problem instance set (#instances)</th>
<th>Performance measure</th>
<th>Surrogate model</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPLEX</td>
<td>76 (18/7/45/4)</td>
<td>BIGMIX (1000) REG (1000) CORLAT (1000) RCW (1000)</td>
<td>Running time</td>
<td>Random forest for log transformed CPU times</td>
</tr>
<tr>
<td>SPEAR</td>
<td>28 (4/12/10/9)</td>
<td>IBM (765) SWV (604)</td>
<td>Running time</td>
<td>Random forest for log transformed CPU times</td>
</tr>
<tr>
<td>Ant Colony Optimization</td>
<td>11 (4/4/3/5)</td>
<td>TSP (50)</td>
<td>Solution quality</td>
<td>Random forest</td>
</tr>
</tbody>
</table>
Analysis on surrogate configuration benchmarks

Prediction accuracy

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Pearson's correlation coefficient (10-fold)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPLEX-BIGMIX</td>
<td>0.859</td>
</tr>
<tr>
<td>CPLEX-CORLAT</td>
<td>0.875</td>
</tr>
<tr>
<td>CPLEX-RCW</td>
<td>0.724</td>
</tr>
<tr>
<td>CPLEX-REG</td>
<td>0.701</td>
</tr>
<tr>
<td>SPEAR-IBM</td>
<td>0.946</td>
</tr>
<tr>
<td>SPEAR-SWV</td>
<td>0.900</td>
</tr>
<tr>
<td>ACOTSP</td>
<td>0.999</td>
</tr>
</tbody>
</table>

Scatter plots of real (x-axis) versus predicted (y-axis) performance
Analysis on surrogate configuration benchmarks

Homogeneity (Kendall’s concordance coefficient W)

- $m_{\text{original\_real}}$: on the real performance of the original 1000 configurations
- $m_{\text{original\_prediction}}$: on the predicted performance of the original 1000 configurations
- $\text{mean}_{1000\_\text{configurations}}$: on the predicted performance of 1000 random configurations (10 runs)

Homogeneity (surrogate) > Homogeneity (real)
Analysis on surrogate configuration benchmarks

Homogeneity (Kendall’s concordance coefficient $W$)

- $m_{original\_real}$: on the real performance of the original 1000 configurations
- $m_{original\_prediction}$: on the predicted performance of the original 1000 configurations
- $\text{mean}_{1000\_configurations}$: on the predicted performance of 1000 random configurations (10 runs)

Consistent homogeneity ranking between benchmarks
Importance of algorithm parameters using fANOVA (Hutter, Hoos & Leyton-Brown, 2014)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Original performance dataset</th>
<th>Predicted performance dataset</th>
</tr>
</thead>
</table>
| ACOTSP       | Sum of fractions for main effects 53.03%  
Sum of fractions for pairwise interaction effects 19.50%  
44.56% due to main effect: localsearch | Sum of fractions for main effects 50.47%  
Sum of fractions for pairwise interaction effects 22.77%  
40.03% due to main effect: localsearch |
| CPLEX-REG    | Sum of fractions for main effects 33.26%  
Sum of fractions for pairwise interaction effects 22.44%  
19.68% due to main effect: mip_strategy_subalgorithm  
7.56% due to main effect: mip_strategy_variableselect | Sum of fractions for main effects 34.46%  
Sum of fractions for pairwise interaction effects 21.61%  
21.22% due to main effect: mip_strategy_subalgorithm  
7.65% due to main effect: mip_strategy_variableselect |
| SPEAR-IBM    | Sum of fractions for main effects 76.16%  
Sum of fractions for pairwise interaction effects 8.64%  
74.13% due to main effect: sp-var-dec-heur | Sum of fractions for main effects 76.10%  
Sum of fractions for pairwise interaction effects 8.92%  
74.35% due to main effect: sp-var-dec-heur |
Content

Motivation
Irace’s parameters
Surrogate configuration benchmarks
Configuring irace using surrogate benchmarks
Conclusions
Experimental settings

Meta-tuning budget: 5000/10,000 irace runs

Configuration budget:
- 10,000 runs
- 5000 runs
Use irace to configure irace on single surrogate benchmarks
Analysis on irace's parameter importance

<table>
<thead>
<tr>
<th></th>
<th>$N_{iter}$</th>
<th>$N_{min}$</th>
<th>confidence_level</th>
<th>enable_elitist</th>
<th>elitist_instances</th>
<th>test_type</th>
<th>$\mu$</th>
<th>$T_{first}$</th>
<th>enable_soft_restart</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACO-TSP</td>
<td>15</td>
<td>3</td>
<td>0.87</td>
<td>false</td>
<td></td>
<td>t-test</td>
<td>11</td>
<td>9</td>
<td>false</td>
</tr>
<tr>
<td>SPEAR-IBM</td>
<td>26</td>
<td>4</td>
<td>0.53</td>
<td>false</td>
<td></td>
<td>t-test-bonferroni</td>
<td>5</td>
<td>20</td>
<td>true</td>
</tr>
<tr>
<td>SPEAR-SWV</td>
<td>32</td>
<td>1</td>
<td>0.79</td>
<td>false</td>
<td></td>
<td>t-test-bonferroni</td>
<td>20</td>
<td>17</td>
<td>false</td>
</tr>
<tr>
<td>CPLEX-BIGMIX</td>
<td>33</td>
<td>2</td>
<td>0.74</td>
<td>false</td>
<td></td>
<td>t-test</td>
<td>9</td>
<td>8</td>
<td>true</td>
</tr>
<tr>
<td>CPLEX-CORLAT</td>
<td>14</td>
<td>5</td>
<td>0.81</td>
<td>false</td>
<td></td>
<td>t-test</td>
<td>7</td>
<td>5</td>
<td>false</td>
</tr>
<tr>
<td>CPLEX-RCW</td>
<td>35</td>
<td>1</td>
<td>0.82</td>
<td>false</td>
<td></td>
<td>t-test</td>
<td>3</td>
<td>5</td>
<td>false</td>
</tr>
<tr>
<td>CPLEX-REG</td>
<td>35</td>
<td>2</td>
<td>0.52</td>
<td>false</td>
<td></td>
<td>t-test-holm</td>
<td>4</td>
<td>6</td>
<td>true</td>
</tr>
<tr>
<td>CPLEX-all</td>
<td>37</td>
<td>1</td>
<td>0.65</td>
<td>false</td>
<td></td>
<td>t-test</td>
<td>5</td>
<td>5</td>
<td>true</td>
</tr>
<tr>
<td>default-ACO-TSP</td>
<td>5</td>
<td>5</td>
<td>0.95</td>
<td>true</td>
<td>1</td>
<td>t-test</td>
<td>5</td>
<td>5</td>
<td>true</td>
</tr>
<tr>
<td>default-SPEAR</td>
<td>6</td>
<td>6</td>
<td>0.95</td>
<td>true</td>
<td>1</td>
<td>t-test</td>
<td>5</td>
<td>5</td>
<td>true</td>
</tr>
<tr>
<td>default-CPLEX</td>
<td>8</td>
<td>8</td>
<td>0.95</td>
<td>true</td>
<td>1</td>
<td>t-test</td>
<td>5</td>
<td>5</td>
<td>true</td>
</tr>
</tbody>
</table>
### Analysis on irace’s parameter importance

#### Higher-order interaction effect

<table>
<thead>
<tr>
<th>Method</th>
<th>$N_{iter}$</th>
<th>$N_{min}$</th>
<th>confidence_level</th>
<th>enable_elitist</th>
<th>elitist_instances</th>
<th>test_type</th>
<th>$\mu$</th>
<th>$T_{first}$</th>
<th>enable_soft_restart</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACO-TSP</td>
<td>15</td>
<td>3</td>
<td>0.87</td>
<td>false</td>
<td></td>
<td>t-test</td>
<td>11</td>
<td>9</td>
<td>false</td>
</tr>
<tr>
<td>SPEAR-IBM</td>
<td>26</td>
<td>4</td>
<td>0.53</td>
<td>false</td>
<td></td>
<td>t-test-bonferroni</td>
<td>5</td>
<td>20</td>
<td>true</td>
</tr>
<tr>
<td>SPEAR-SWV</td>
<td>32</td>
<td>1</td>
<td>0.79</td>
<td>false</td>
<td></td>
<td>t-test-bonferroni</td>
<td>20</td>
<td>17</td>
<td>false</td>
</tr>
<tr>
<td>CPLEX-BIGMIX</td>
<td>33</td>
<td>2</td>
<td>0.74</td>
<td>false</td>
<td></td>
<td>t-test</td>
<td>9</td>
<td>8</td>
<td>true</td>
</tr>
<tr>
<td>CPLEX-CORLAT</td>
<td>14</td>
<td>5</td>
<td>0.81</td>
<td>false</td>
<td></td>
<td>t-test</td>
<td>7</td>
<td>5</td>
<td>false</td>
</tr>
<tr>
<td>CPLEX-RCW</td>
<td>35</td>
<td>1</td>
<td>0.82</td>
<td>false</td>
<td></td>
<td>t-test</td>
<td>3</td>
<td>5</td>
<td>false</td>
</tr>
<tr>
<td>CPLEX-REG</td>
<td>35</td>
<td>2</td>
<td>0.52</td>
<td>false</td>
<td></td>
<td>t-test-holm</td>
<td>4</td>
<td>6</td>
<td>true</td>
</tr>
<tr>
<td>CPLEX-all</td>
<td>37</td>
<td>1</td>
<td>0.65</td>
<td>false</td>
<td></td>
<td>t-test</td>
<td>5</td>
<td>5</td>
<td>true</td>
</tr>
<tr>
<td>default-ACO-TSP</td>
<td>5</td>
<td>5</td>
<td>0.95</td>
<td>true</td>
<td></td>
<td>1</td>
<td>t-test</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>default-SPEAR</td>
<td>6</td>
<td>6</td>
<td>0.95</td>
<td>true</td>
<td></td>
<td>1</td>
<td>t-test</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>default-CPLEX</td>
<td>8</td>
<td>8</td>
<td>0.95</td>
<td>true</td>
<td></td>
<td>1</td>
<td>t-test</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>
Results on real configuration benchmarks

- Tuned vs default: $p = 5.7 \times 10^{-6}$
- Tuned-CPLEX-all vs default: $p = 2.1 \times 10^{-4}$

- Tuned-SPEAR-SWV vs default: $p = 1.9 \times 10^{-6}$
- Tuned-SPEAR-IBM vs default: $p = 1.9 \times 10^{-6}$
- Tuned-CPLEX-all vs default: $p = 8.3 \times 10^{-3}$
Content

Motivation
irace’s parameters
Surrogate configuration benchmarks
Configuring irace using surrogate benchmarks
Conclusions
Conclusions

Meta-tuning on the surrogate benchmarks indicates that there is room for improvement on irace’s performance over its current default configuration.

Future work:

• Improve the surrogate modelling
• Build a representative library of surrogate benchmarks
• Study other state-of-the-art configurators
• Provide more guidelines for algorithm configurators
• Algorithm selection for algorithm configurators
Example 2
Data Science FOR Optimization: Using Data Science Engineering an Algorithm

• Characterization of neighborhood behaviours in a multi-neighborhood local search algorithm, Dang et al., International Conference on Learning and Intelligent Optimization, 2016

• Based upon
The swap-body vehicle routing problem
A multi-neighborhood local search for the Swap-body Vehicle Routing problem

**Neighborhoods (42)**

<table>
<thead>
<tr>
<th>Neighborhood</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cheapest insertion</td>
<td>(11)</td>
</tr>
<tr>
<td>Swap</td>
<td>(1)</td>
</tr>
<tr>
<td>Intra-route 2-opt</td>
<td>(1)</td>
</tr>
<tr>
<td>Inter-route 2-opt</td>
<td>(1)</td>
</tr>
<tr>
<td>Change swap location</td>
<td>(1)</td>
</tr>
<tr>
<td>Merge routes</td>
<td>(1)</td>
</tr>
<tr>
<td>Split to sub-routes</td>
<td>(1)</td>
</tr>
<tr>
<td>Ruin recreate</td>
<td>(2)</td>
</tr>
<tr>
<td>Remove route</td>
<td>(1)</td>
</tr>
<tr>
<td>Remove sub-route</td>
<td>(1)</td>
</tr>
<tr>
<td>Remove sub-route with cheapest</td>
<td></td>
</tr>
<tr>
<td>insertion</td>
<td>(1)</td>
</tr>
<tr>
<td>Remove chains</td>
<td>(8)</td>
</tr>
<tr>
<td>EachSequenceCheapestInsert</td>
<td>(3)</td>
</tr>
<tr>
<td>Convert to route</td>
<td>(1)</td>
</tr>
<tr>
<td>Convert to sub-route</td>
<td>(1)</td>
</tr>
<tr>
<td>Add sub-route</td>
<td>(1)</td>
</tr>
<tr>
<td>Ejection chain</td>
<td>(6)</td>
</tr>
</tbody>
</table>
Research question

Groups of similar neighborhoods?
1. Characterize each neighborhood behaviours as a feature vector (*based on information collected from different algorithm runs*)
2. Clustering neighborhoods
Characterize neighborhood’s behaviours

Observables:

try to reflect the changes of neighborhood’s behaviours according to the hardness of different solution quality regions

➢ Probability of improving \( r_{\text{improve}} \), worsening \( r_{\text{worsen}} \), or doing nothing \( r_{\text{nothing}} \)

\[ r_{\text{improve}} + r_{\text{worsen}} + r_{\text{nothing}} = 1 \]
Characterize neighborhood’s behaviours

Step 2: identify solution quality’s regions based on the total number of times all neighborhoods are applied on each interval.

![Graph showing the sum of iterations over intervals with regions labeled as easy to reach, easy to escape, easy to reach, hard to escape, and hard to reach.](image-url)
Clustering neighborhoods

Each neighborhood is represented as a vector of 

\[ \text{#instances} \times \text{#regions} \times \text{#observables} \]

elements

High-dimensional low-sample size (42 individuals, 150 dimensions)
Clustering neighborhoods

Clustering result: 9 clusters

Ejection-chain 3, 4, 5; Remove-chain 1, 2, 3, 6, 7, 8; Remove-sub-route-with-cheapest-insertion;
Swap; Inter-route-two-opt
Cheapest-insertion 10, 15, 20, 25, 35, 50; Each-sequence-cheapest-insertion (2,5), (4,4), (5,2); Remove-chain 4
Cheapest-insertion 1, 2, 3, 4, 5
Change-swap-location; Merge-route
Add-sub-route; Convert-to-sub-route
Ejection-chain 10, 15, 35; Remove-chain 5; Intra-route-two-opt
Ruin-recreate 2, 3
Convert-to-route; Remove-sub-route; Remove-route; Split-to-sub-route
Applied to algorithm tuning

original vs clustered : $p = 0.00216206$

basic vs clustered : $p = 0.009258918$
Example 3
Optimization FOR Data Science: Community detection in graphs

- Many applications can be cast as graph theoretical problems
  - cell phone calls,
  - e-mail connections,
  - influencer graphs in social media,
- Number of nodes:
  - 1,000 – 100,000 – 1,000,000 – 100,000,000 – …

Community detection in graphs

- Objective function: quality of a partition of the set of nodes
  => Optimisation problem to find the best partition

- Evolving network -> evolving community structure

- Hypothesis: no brisk changes in the network structure
  => protect against brisk changes in community structure
  (think how an advisory system for communication would react to brisk changes)

  => Dynamic Community Problem (DCP)

Community detection in graphs

- Snapshot cost $SC$: absolute cost present situation
  - (from clustering theory)
- Temporal cost $TC$: cost of the amount of change

$\Rightarrow$ weighted cost function

$$cost = \alpha \times SC + (1 - \alpha) \times TC$$

$\Rightarrow$ leaves us with:
- $TC$: proper model, information theory
- Parameter alfa
- Parameters optimizer (NGSA-II)
Community detection in graphs

- Data set size: 100 – 1000 nodes (small)
- Complexity:

\[ O(g \times p \times \log(p) \times (n \times \log(n) + m) ) \]

- \( p \) is population size
- \( n \) is number of nodes
- \( m \) is number of edges
- \( + : n \log(n) \)
- \( - : g \) (number of generations)
- \( - : \) large constant factor from the genetic algorithm
Community detection in graphs

- Scalability between
  - 128 and 4096 nodes
  - 2938 to 65256 edges

- + : standard optimisation technique (NGSA-II) solves a problem from data science

*However: large graphs?*
Further examples:
Separating knowledge from problem solving

• Knowledge based system and optimization (with Marc Denecker)
  o expressive modeling language -> ‘knowledge base’ (KB)
  o Inference engine -> solves tasks starting from KB

  o Combinatorial tasks: performance issues
    • Discover neighborhoods automatically (symmetry properties, PhD thesis Jo Devriendt)
  o Describe neighborhoods in the KB
    • San Pham and Jo Devriendt
Further examples:
Formalize neighborhoods for local search using predicate logic

- Use local search to improve performance
- Two approaches
  - Discover neighborhoods automatically
    - Analysis of symmetry properties
    - PhD thesis Jo Devriendt
  - Describe neighborhoods in the KB and let the solver use these
    - San Pham and Jo Devriendt
Further examples:

Multi Armed Bandit for algorithm selection

- Given a set of algorithms, and a particular problem instance, which algorithm to run?
- Limited computation time and resources.
- Process may improve over time
  - Multi armed bandit learning
  - Start with a training set of historical instances
  - Improve over time while solving new real world problems

- Online Algorithm Selection, joint work with
  - Hans Degroote, Bernd Bischl and Lars Kothoff.
So Data Science

• Artificial Intelligence
• Data mining
• Machine Learning
• Pattern recognition
• Constraint programming
• Natural language understanding
• …
And Optimization

- Continuous optimization
- Discrete optimization
- Linear (Integer) programming
- Quadratic programming
- Non Linear programming
- Metaheuristics
- Hyperheuristics
- ...

ODS 2017, Sorrento
Techniques from OR and AI can be combined to produce versatile and efficient solutions

BUT

• Development time
• Reliability
• Robustness
• Understanding
• Toolset
• Theory
• …
Join the DSO working group

https://www.euro-online.org