HeuristicLab

A Paradigm-Independent and Extensible Environment for Heuristic Optimization

Algorithm and Experiment Design with HeuristicLab

An Open Source Optimization Environment for Research and Education

G. Kronberger and M. Kommenda
Heuristic and Evolutionary Algorithms Laboratory (HEAL)
School of Informatics/Communications/Media, Campus Hagenberg
University of Applied Sciences Upper Austria

HEAL
Heuristic and Evolutionary Algorithms Laboratory

Josef Ressel-Zentrum
HEUREKA!

Instructor Biographies

- Gabriel Kronberger
  - Full professor for business intelligence (since 2011)
    University of Applied Sciences Upper Austria
  - Research associate (2005 – 2011)
    University of Applied Sciences Upper Austria
  - Architect of HeuristicLab
  - Member of the HEAL research group
  - http://heal.heuristiclab.com/team/kronberger

- Michael Kommenda
  - Research associate (since 2007)
    University of Applied Sciences Upper Austria
  - Architect of HeuristicLab
  - Member of the HEAL research group
  - http://heal.heuristiclab.com/team/kommenda

Agenda

- Objectives of the Tutorial
- Introduction
- Where to get HeuristicLab?
- Plugin Infrastructure
- Graphical User Interface
- Available Algorithms & Problems

- **Demonstration Part I: Working with HeuristicLab**
- **Demonstration Part II: Data-based Modeling**

- Some Additional Features
- Planned Features
- Team
- Suggested Readings
- Bibliography
- Questions & Answers

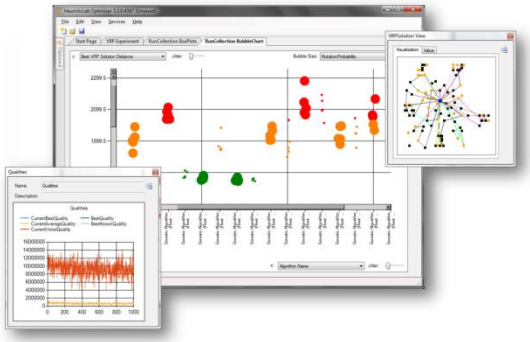
Objectives of the Tutorial

HeuristicLab

- Introduce general motivation and design principles of HeuristicLab
- Show where to get HeuristicLab
- Explain basic GUI usability concepts
- Demonstrate basic features
- Demonstrate analysis of optimization experiments
- Demonstrate data-based modeling features
- Outline some additional features

Introduction

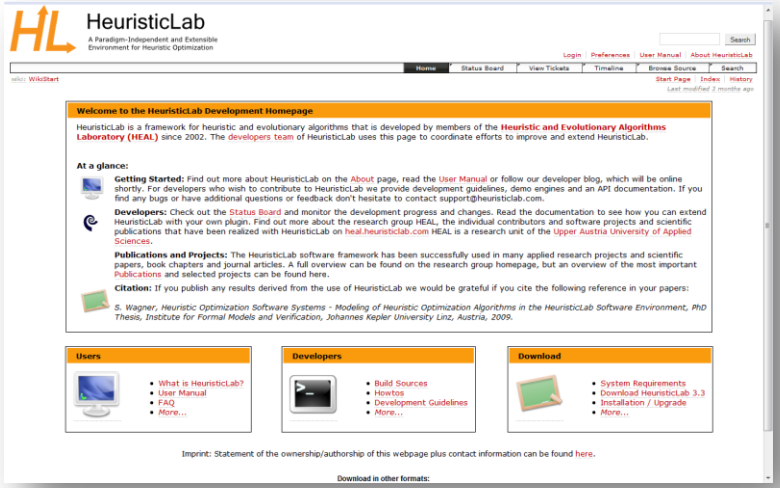
- Motivation and Goals
  - graphical user interface
  - paradigm independence
  - multiple algorithms and problems
  - large scale experiments and analyses
  - parallelization
  - extensibility, flexibility and reusability
  - visual and interactive algorithm development
  - multiple layers of abstraction



- Facts
  - development of HeuristicLab started in 2002
  - based on Microsoft .NET and C#
  - used in research and education
  - second place at the *Microsoft Innovation Award 2009*
  - open source (GNU General Public License)
  - version 3.3.0 released on May 18th, 2010
  - latest version 3.3.9 released on October 11th, 2013

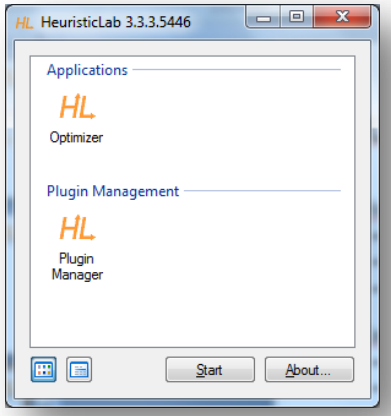
Where to get HeuristicLab?

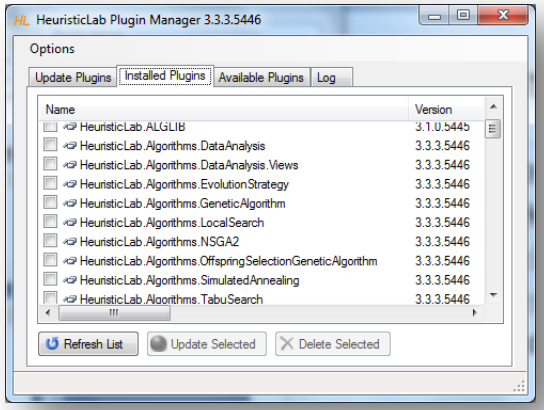
- Download binaries
  - deployed as ZIP archives
  - latest stable version 3.3.9
    - released on October 11th, 2013
  - daily trunk builds
  - http://dev.heuristiclab.com/download
- Check out sources
  - SVN repository
  - HeuristicLab 3.3.9 tag
    - http://dev.heuristiclab.com/svn/hl/core/tags/3.3.9
  - Stable development version
    - http://dev.heuristiclab.com/svn/hl/core/stable
- License
  - GNU General Public License (Version 3)
- System requirements
  - Microsoft .NET Framework 4.0 Full Version
  - enough RAM and CPU power ;-)



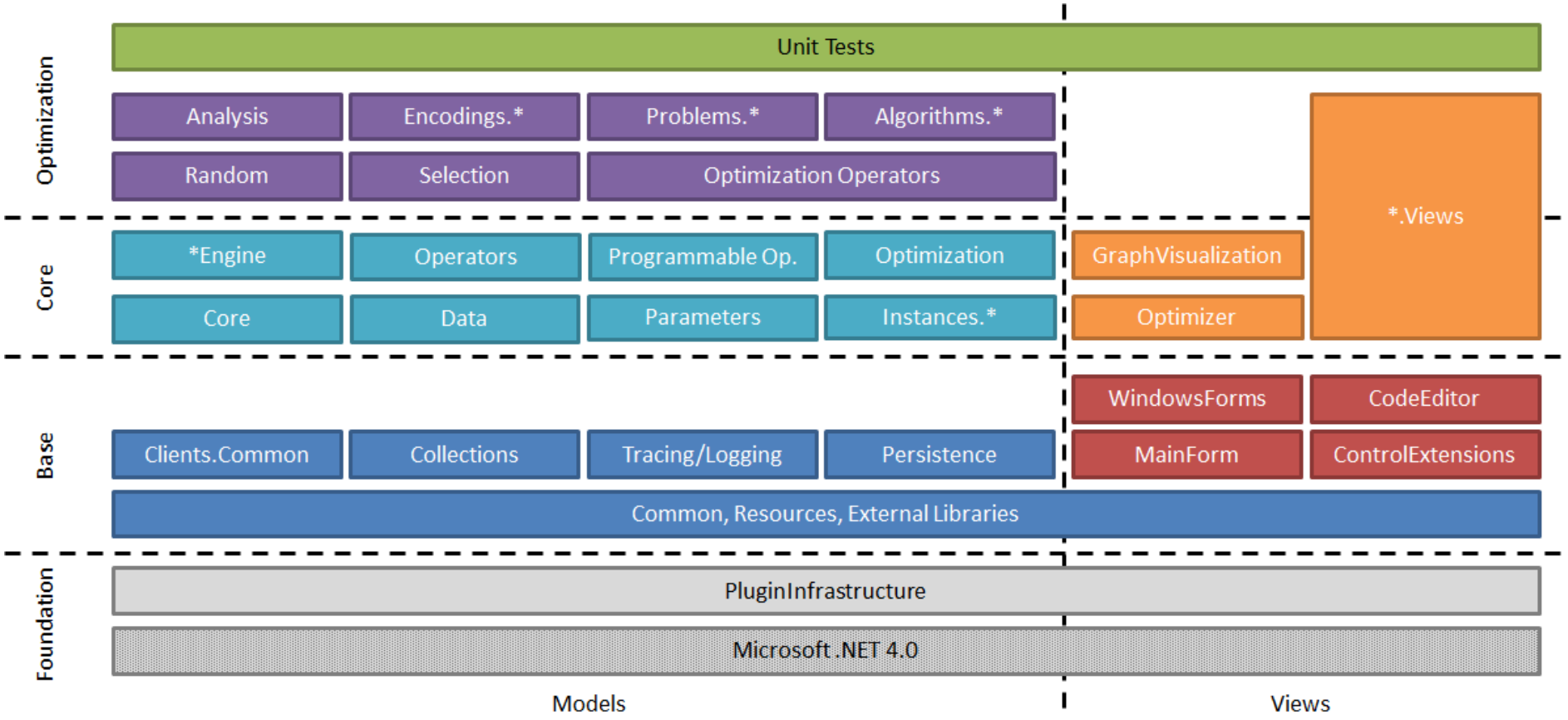
Plugin Infrastructure

- HeuristicLab consists of many assemblies
  - 132 plugins in HeuristicLab 3.3.9
  - plugins can be loaded or unloaded at runtime
  - plugins can be updated via internet
  - application plugins provide GUI frontends
- Extensibility
  - developing and deploying new plugins is easy
  - dependencies are explicitly defined, automatically checked and resolved
  - automatic discovery of interface implementations (service locator pattern)
- Plugin Manager
  - GUI to check, install, update or delete plugins





Plugin Architecture



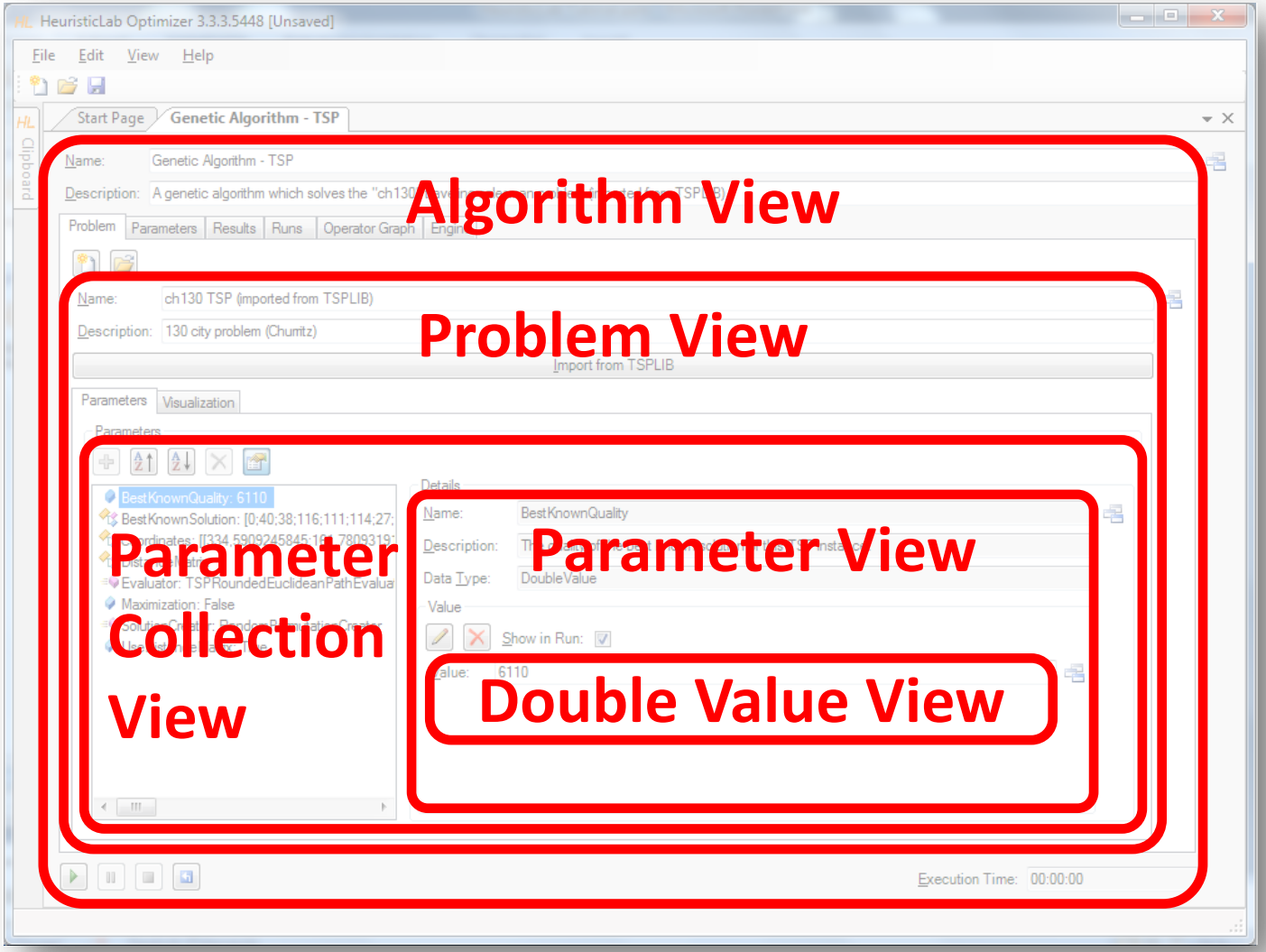
Graphical User Interface

HeuristicLab

- HeuristicLab GUI is made up of views
  - views are visual representations of content objects
  - views are composed in the same way as their content
  - views and content objects are loosely coupled
  - multiple different views may exist for the same content

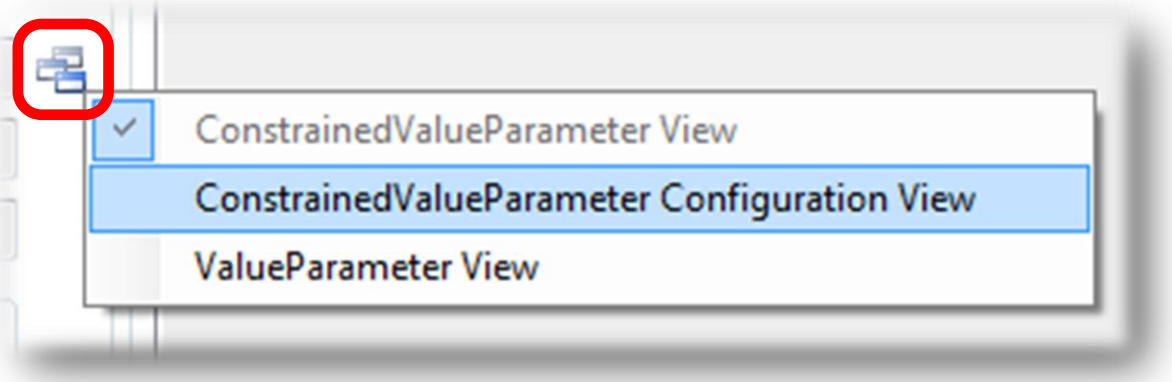
- Drag & Drop
  - views support drag & drop operations
  - content objects can be copied or moved (shift key)
  - enabled for collection items and content objects

Graphical User Interface



Graphical User Interface

- ViewHost
  - control which hosts views
  - right-click on windows icon to switch views
  - double-click on windows icon to open another view
  - drag & drop windows icon to copy contents



Available Algorithms

Population-based

- CMA-ES
- Evolution Strategy
- Genetic Algorithm
- Offspring Selection Genetic Algorithm
- Island Genetic Algorithm
- Island Offspring Selection Genetic Algorithm
- SASEGASA
- Relevant Alleles Preserving GA (RAPGA)
- Genetic Programming
- NSGA-II
- Scatter Search
- Particle Swarm Optimization

Trajectory-based

- Local Search
- Tabu Search
- Robust Taboo Search
- Variable Neighborhood Search
- Simulated Annealing

Data Analysis

- Linear Discriminant Analysis
- Linear Regression
- Multinomial Logit Classification
- k-Nearest Neighbor
- k-Means
- Neighbourhood Component Analysis
- Artificial Neural Networks
- Random Forests
- Support Vector Machines
- Gaussian Processes

Additional Algorithms

- User-defined Algorithm
- Performance Benchmarks
- Hungarian Algorithm
- Cross Validation
- LM-BFGS,

Available Problems

Combinatorial Problems

- Traveling Salesman
- Vehicle Routing
- Knapsack
- Job Shop Scheduling
- Linear Assignment
- Quadratic Assignment
- OneMax

Genetic Programming Problems

- Symbolic Classification
- Symbolic Regression
- Symbolic Time-Series Prognosis
- Artificial Ant
- Lawn Mower

Additional Problems

- Single-Objective Test Function
- User-defined Problem
- External Evaluation Problem (Anylogic, Scilab, MATLAB)
- Regression, Classification, Clustering
- Trading

Agenda

- Objectives of the Tutorial
- Introduction
- Where to get HeuristicLab?
- Plugin Infrastructure
- Graphical User Interface
- Available Algorithms & Problems

- **Demonstration Part I: Working with HeuristicLab**
- **Demonstration Part II: Data-based Modeling**

- Some Additional Features
- Planned Features
- Team
- Suggested Readings
- Bibliography
- Questions & Answers

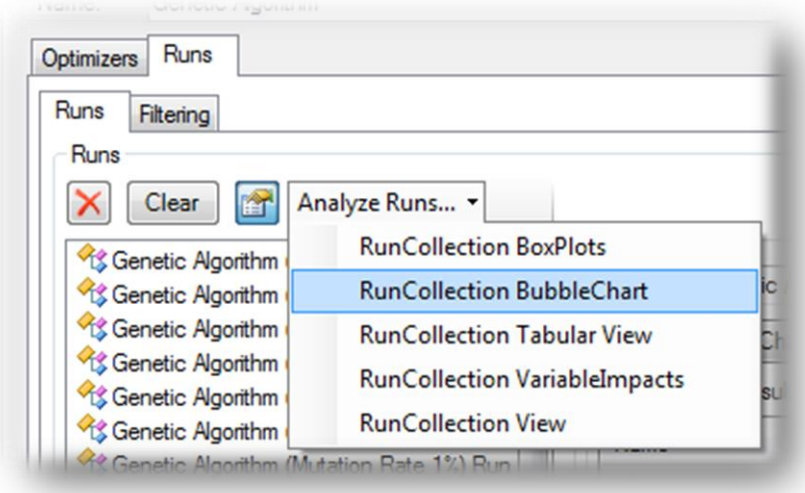
Demonstration Part I: Working with HeuristicLab

- Create, Parameterize and Execute Algorithms
- Save and Load Items
- Create Batch Runs and Experiments
- Analyze Runs

Analyze Runs

- HeuristicLab provides interactive views to analyze and compare all runs
  - textual analysis
    - Tabular View
  - graphical analysis
    - Bubble chart
    - Box plots
    - Multi-line chart

- Filtering is automatically applied to all open run collection views



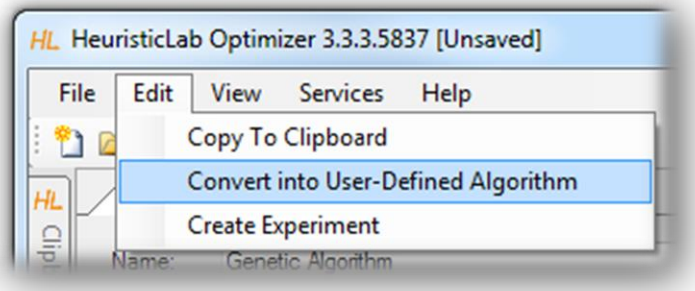
Multi-core CPUs and Parallelization

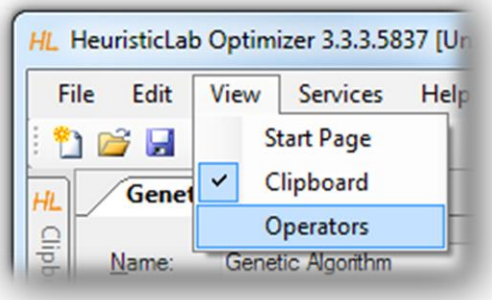
HeuristicLab

- Parallel execution of optimizers in experiments
  - optimizers in an experiment are executed sequentially from top to bottom per default
  - experiments support parallel execution of their optimizers
  - select a not yet executed optimizer and start it manually to utilize another core
  - execution of one of the next optimizers is started automatically after an optimizer is finished
- Parallel execution of algorithms
  - HeuristicLab provides special operators for parallelization
  - engines decide how to execute parallel operations
  - sequential engine executes everything sequentially
  - parallel engine executes parallel operations on multiple cores
  - Hive engine (under development) executes parallel operations on multiple computers
  - all implemented algorithms support parallel solution evaluation

Building User-Defined Algorithms

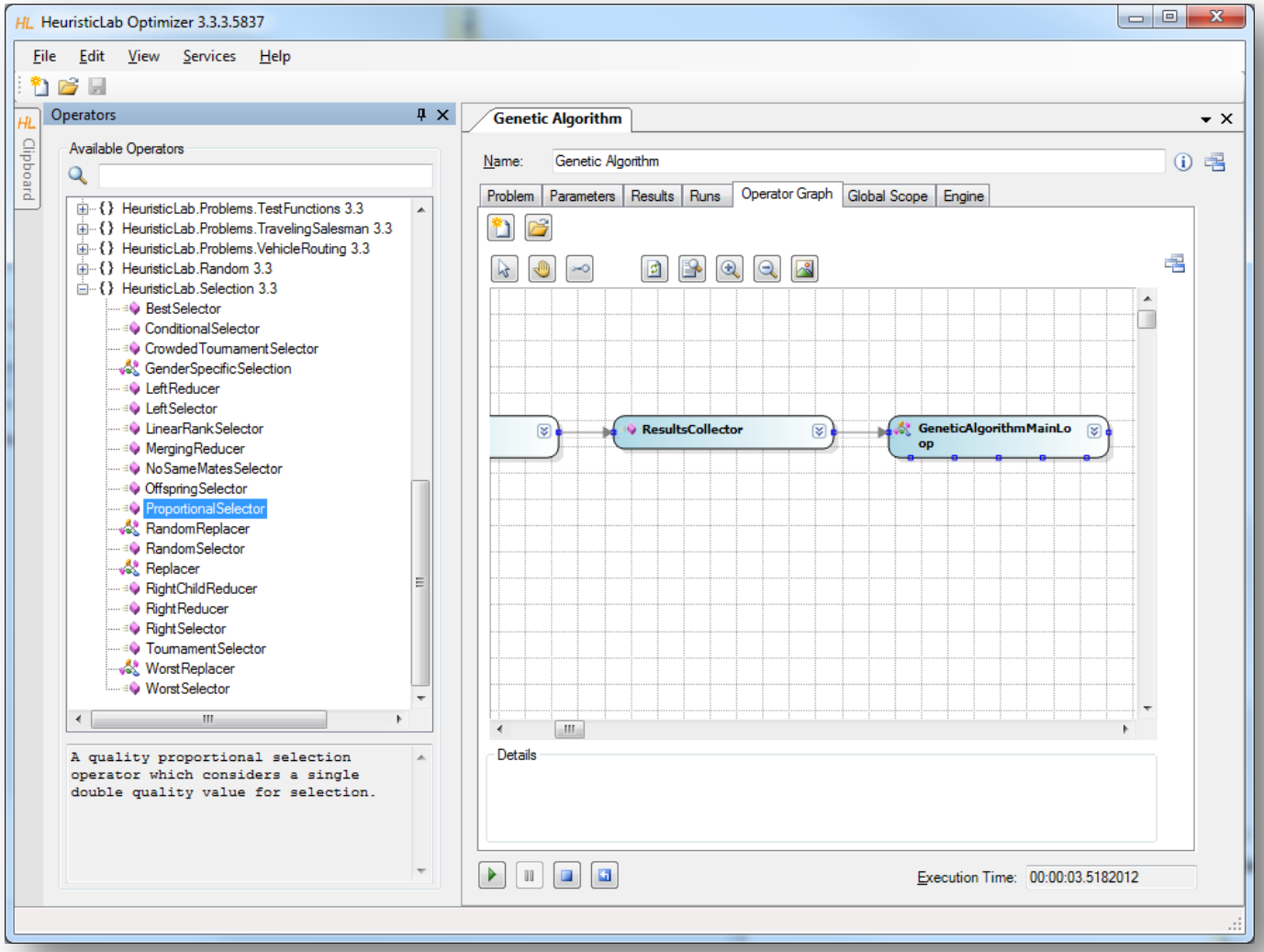
- Operator graphs
  - algorithms are represented as operator graphs
  - operator graphs of user-defined algorithms can be changed
  - algorithms can be defined in the graphical algorithm designer
  - use the menu to convert a standard algorithm into a user-defined algorithm



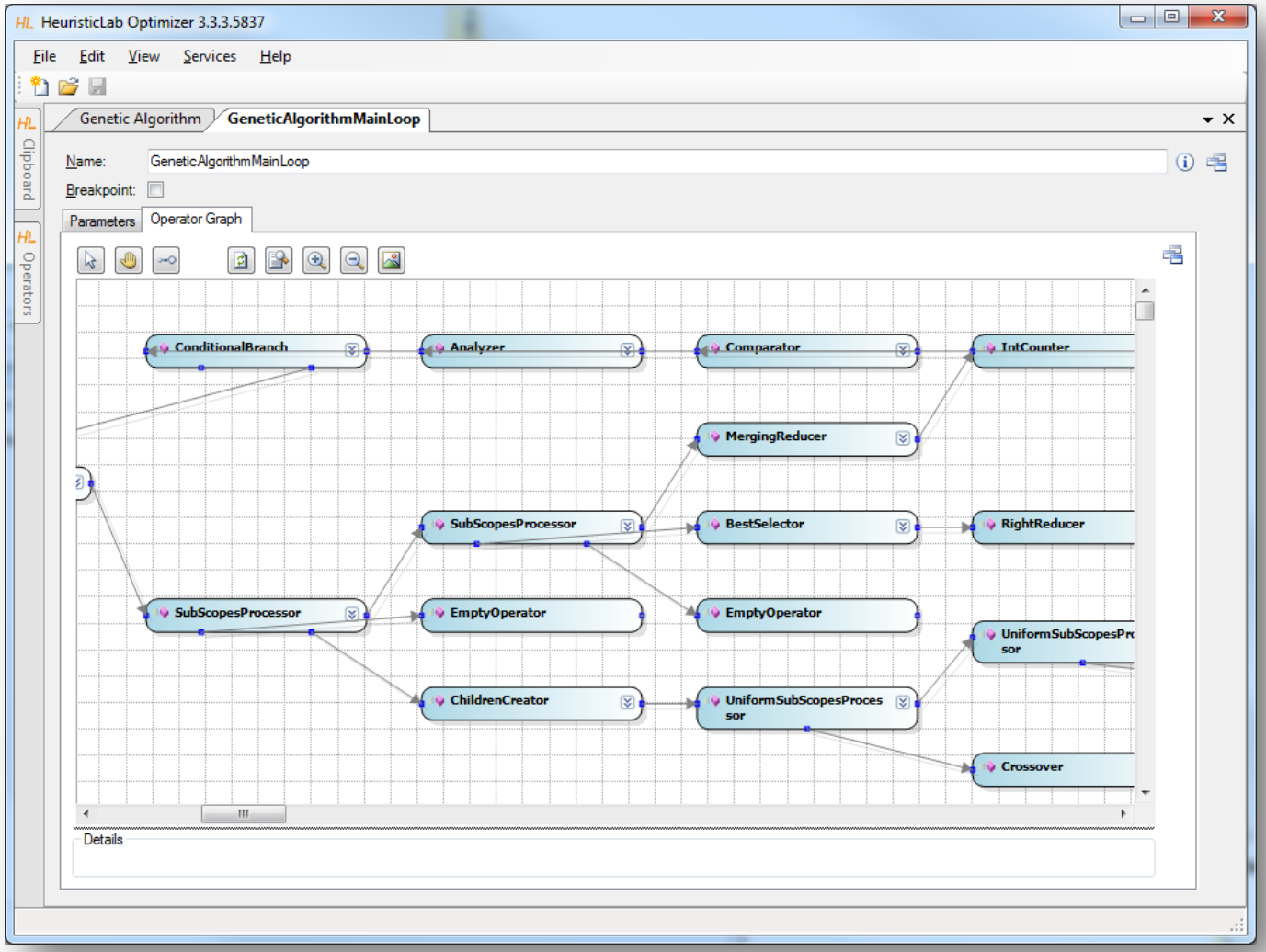


- Operators sidebar
  - drag & drop operators into an operator graph
- Programmable operators
  - add programmable operators in order to implement custom logic in an algorithm
  - no additional development environment needed
- Debug algorithms
  - use the debug engine to obtain detailed information during algorithm execution

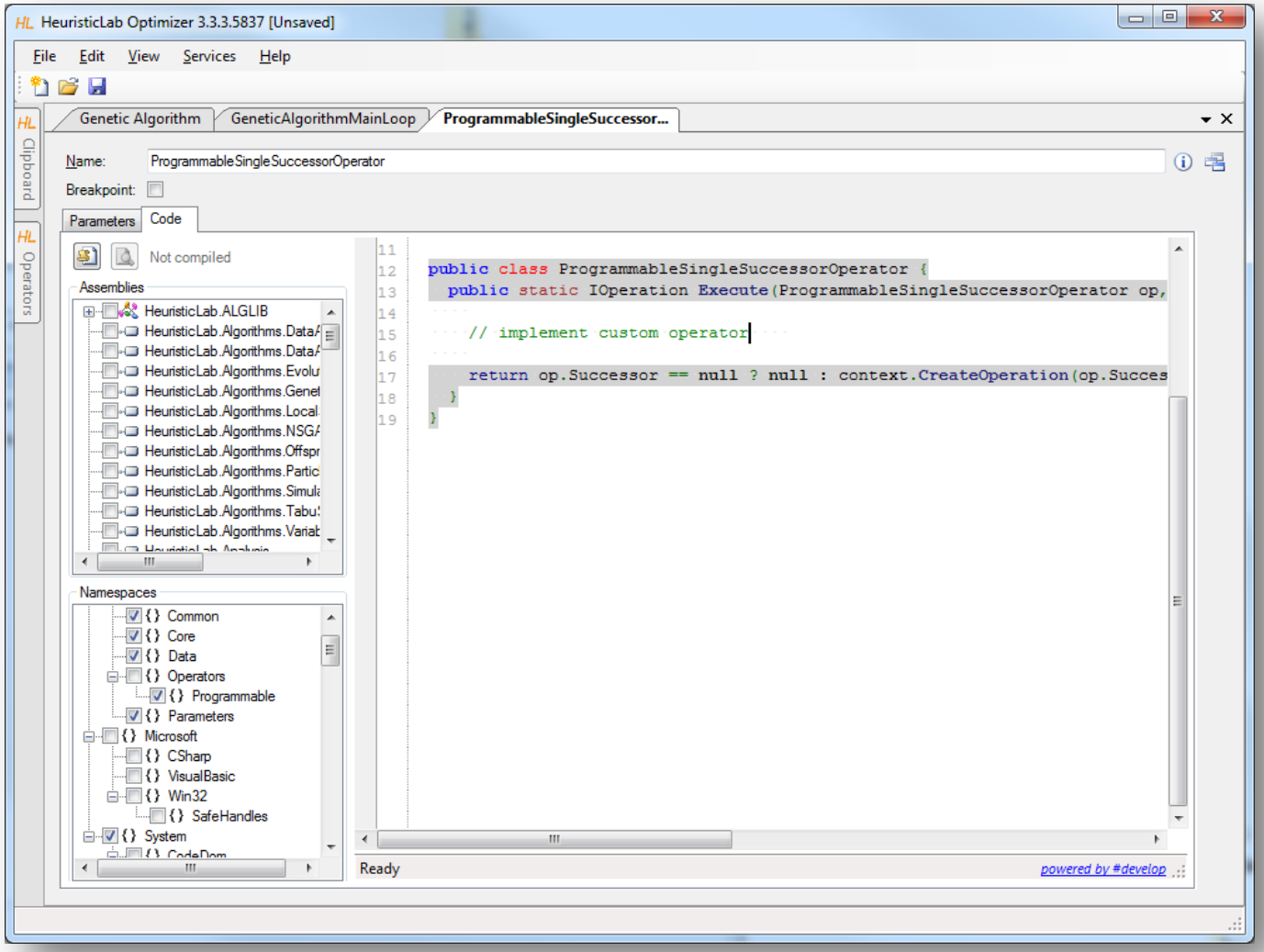
Building User-Defined Algorithms



Building User-Defined Algorithms



Programmable Operators



Agenda

- Objectives of the Tutorial
- Introduction
- Where to get HeuristicLab?
- Plugin Infrastructure
- Graphical User Interface
- Available Algorithms & Problems

- **Demonstration Part I: Working with HeuristicLab**
- **Demonstration Part II: Data-based Modeling**

- Some Additional Features
- Planned Features
- Team
- Suggested Readings
- Bibliography
- Questions & Answers

Demonstration Part II: Data-based Modeling

- Introduction
- Regression with HeuristicLab
- Variable relevance analysis
- Model simplification and export

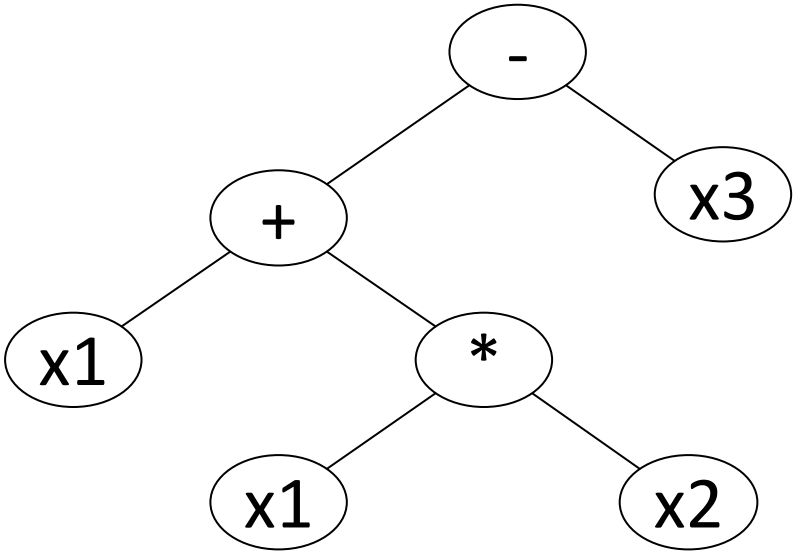
Data-based Modeling Algorithms in HeuristicLab

- Symbolic regression and classification using genetic programming

- External Libraries:
  - Linear Regression, Linear Discriminate Analysis
  - K-Means clustering
  - Support Vector Machines

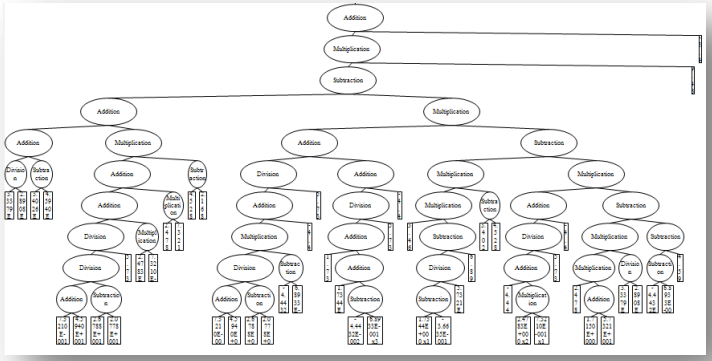
Symbolic Regression with HeuristicLab

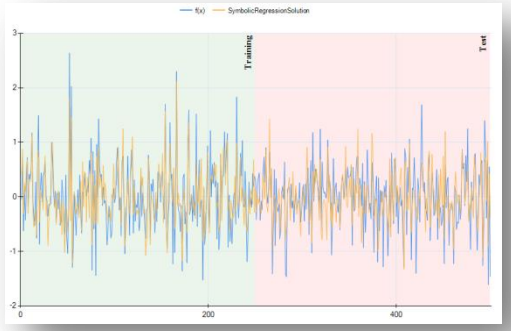
- Genetic programming
  - evolve variable-length models
  - model representation: symbolic expression tree
  - structure and model parameters are evolved side-by-side
  - white-box models



Symbolic Regression with HeuristicLab

- Demonstration
  - problem configuration
  - function set and terminal set
  - model size constraints
  - evaluation
- Algorithm configuration
  - selection
  - mutation
- Analysis of results
  - model accuracy
  - model structure and parameters





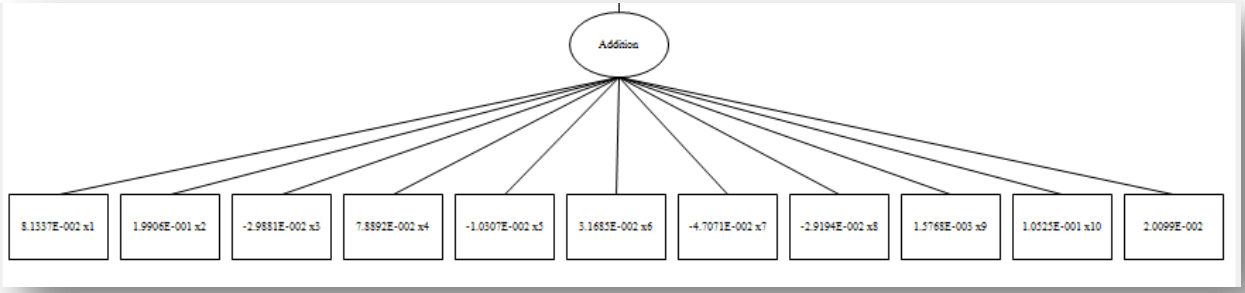
Poly-10 benchmark problem

10 input variables x_1 … x_{10}

$$y = x_1 x_2 + x_3 x_4 + x_5 x_6 + x_1 x_7 x_9 + x_3 x_6 x_{10}$$

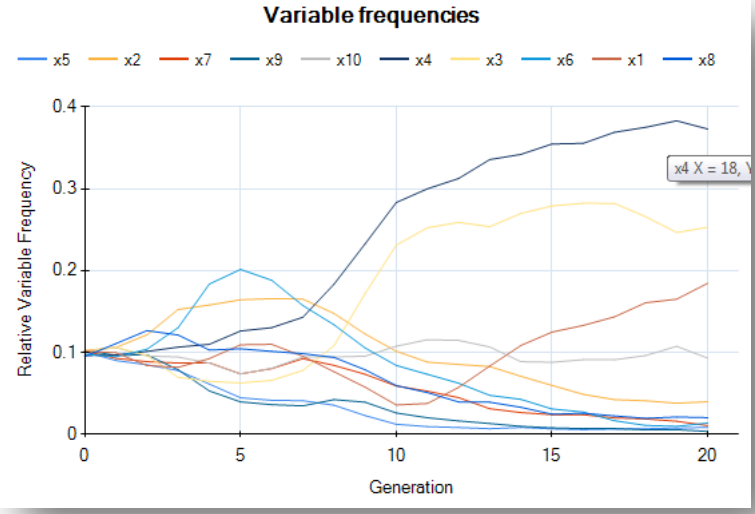
non-linear modeling approach necessary

frequently used in GP literature



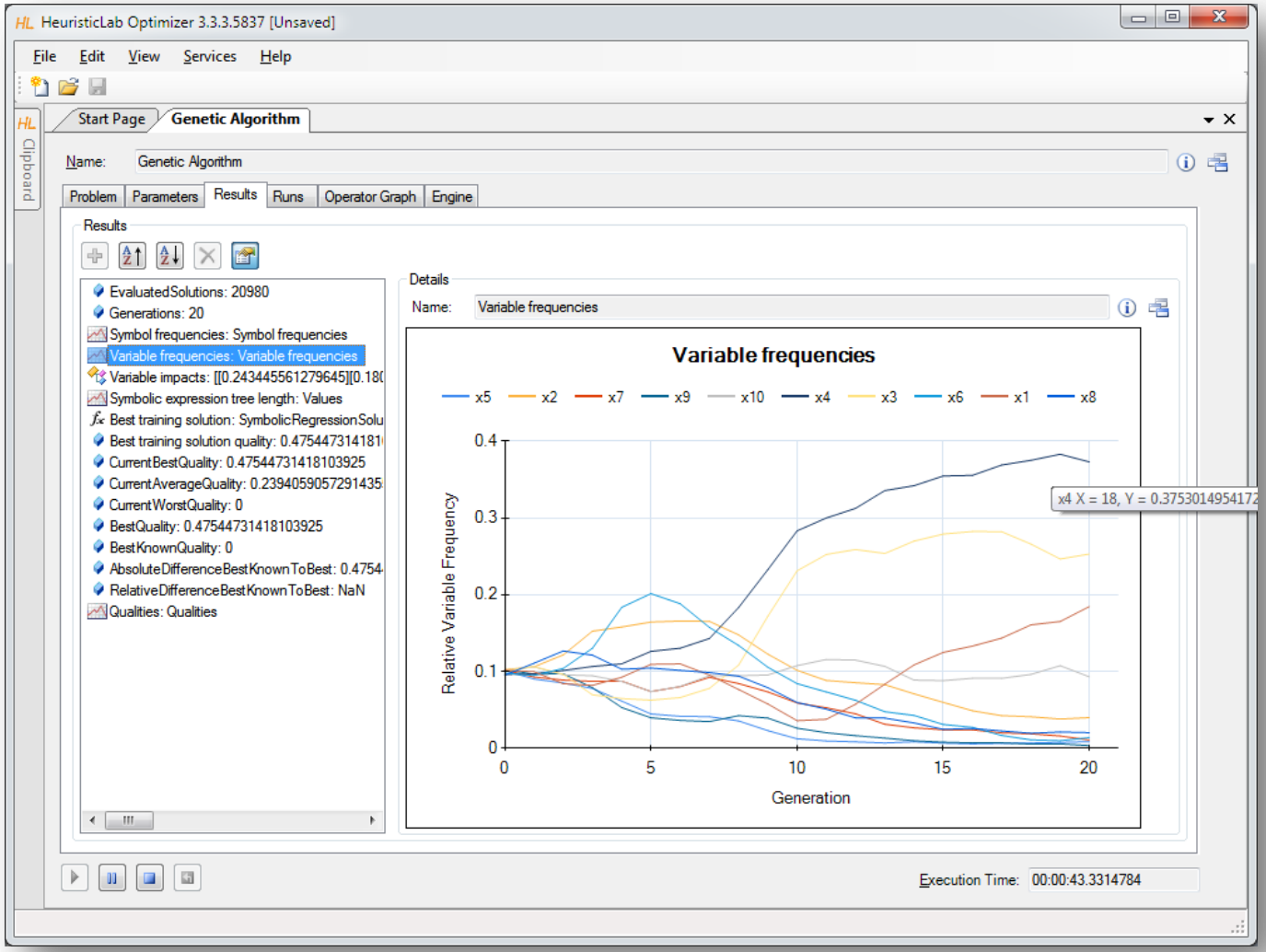
Variable Relevance Analysis

- Which variables are important to predict classes correctly?
- Demonstration
  - Variable frequency analyzer
  - symbol frequency analyzer
  - variable impacts

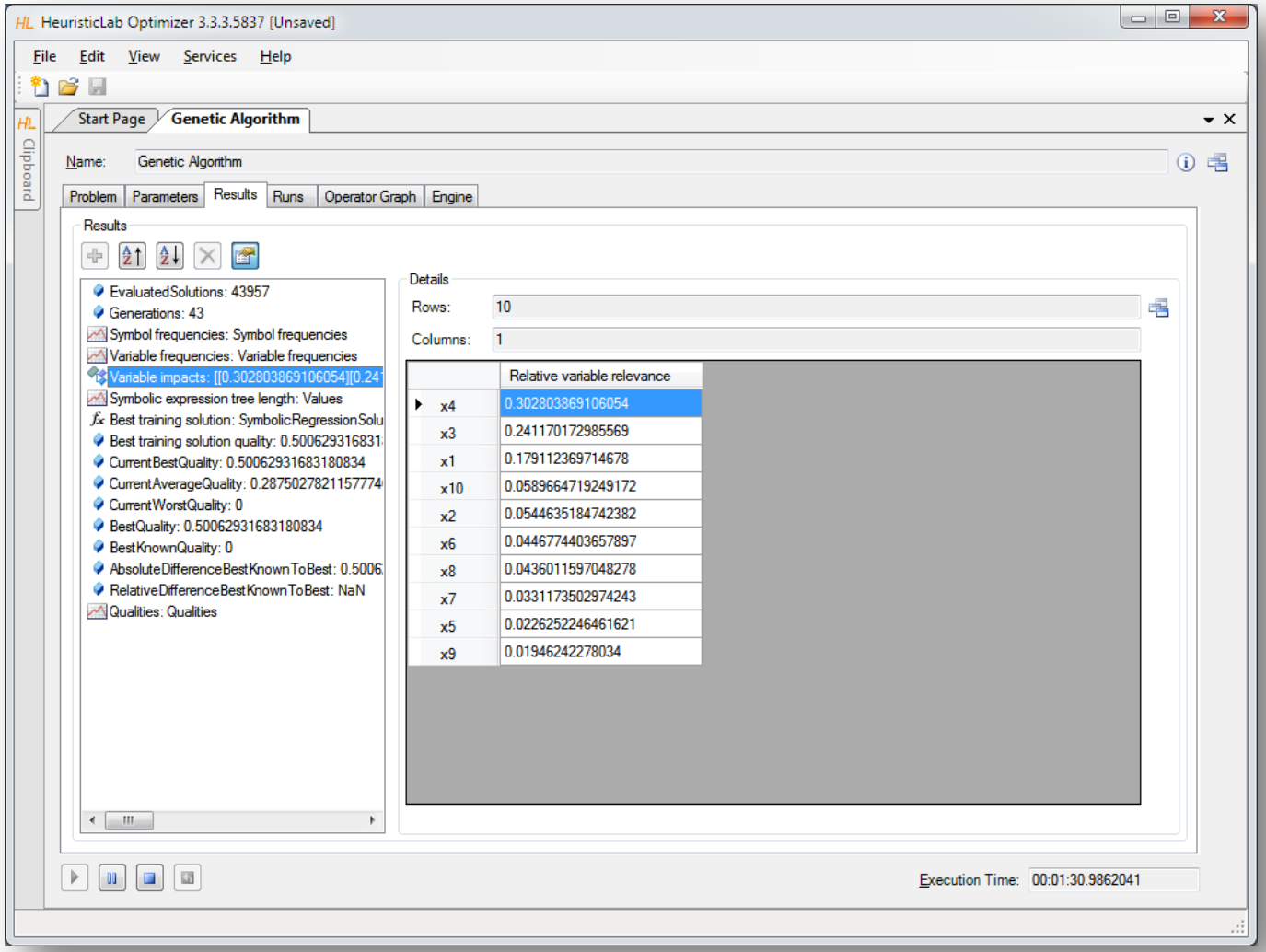


	Relative variable relevance
x4	0.302803869106054
x3	0.241170172985569
x1	0.179112369714678
x10	0.0589664719249172
x2	0.0544635184742382
x6	0.0446774403657897
x8	0.0436011597048278
x7	0.0331173502974243
x5	0.0226252246461621
x9	0.01946242278034

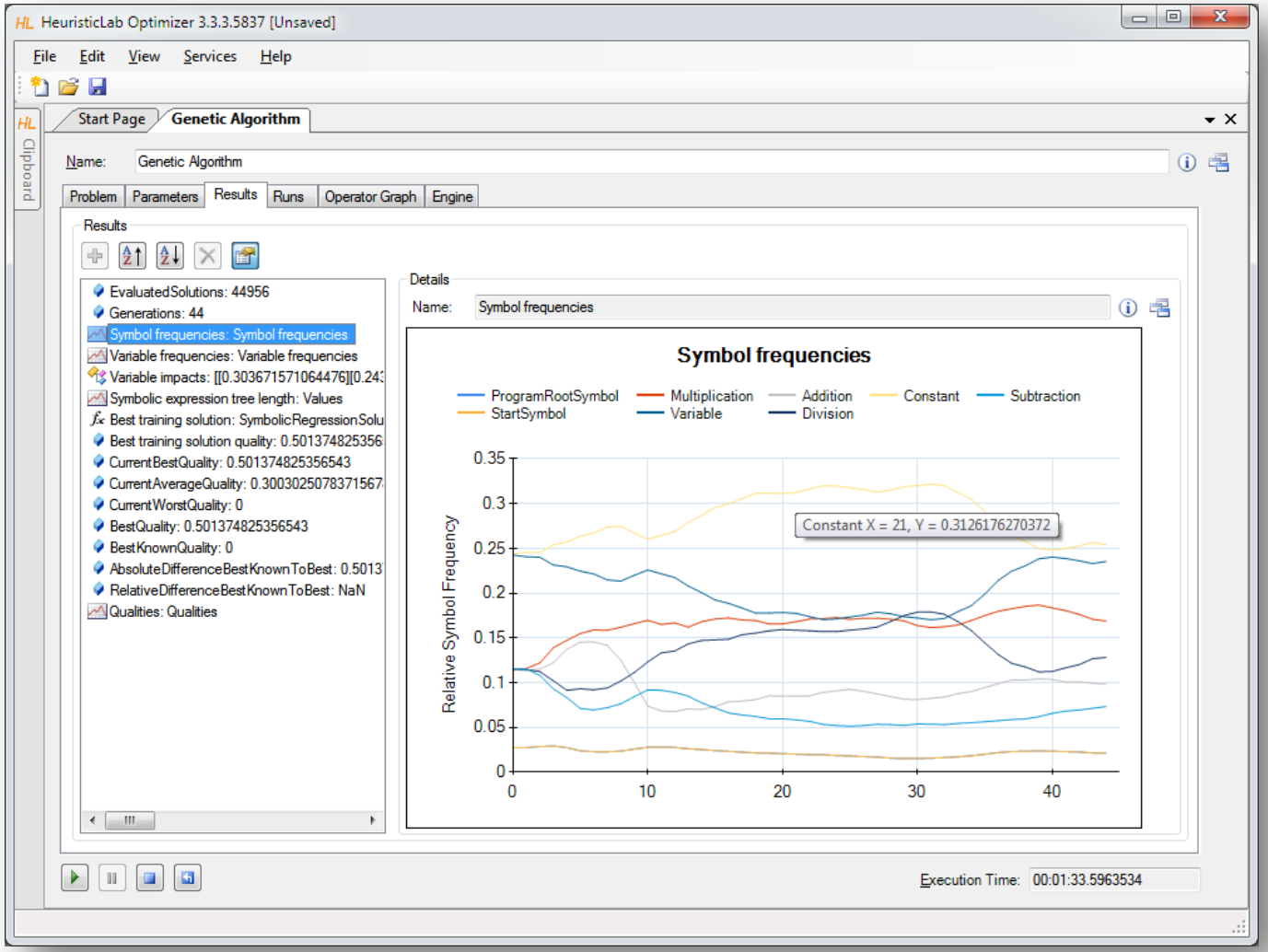
Inspect Variable Frequency Chart



Inspect Variable Impacts

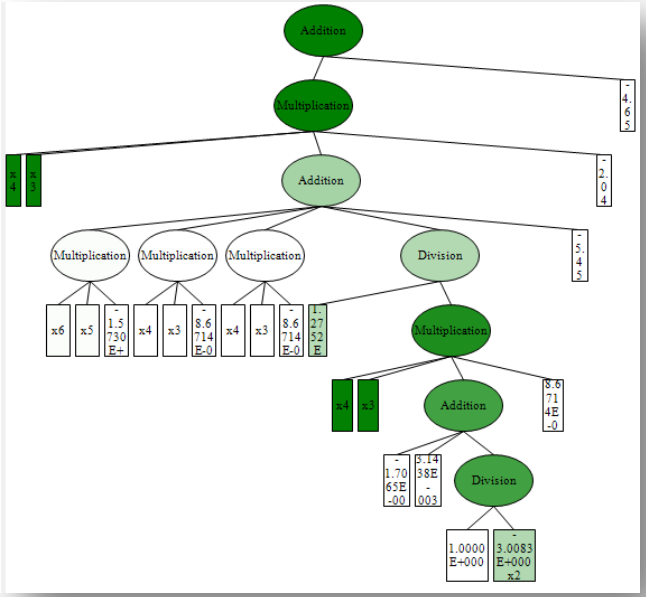


Inspect Symbol Frequencies



Model Simplification and Export

- Demonstration
  - automatic simplification
  - visualization of node impacts
  - manual simplification
    - online update of results
  - model export
    - Excel
    - MATLAB
    - LaTeX



$$Result = x4(t) \cdot x3(t) \cdot c_{20} \qquad (13)$$
$$\cdot \left(x6(t) \cdot x5(t) \cdot c_4 + x4(t) \cdot x3(t) \cdot c_7 + x4(t) \cdot x3(t) \cdot c_{10} + \frac{c_{11} x1(t)}{x4(t) \cdot x3(t) \cdot \left(c_{14} x4(t) + c_{15} x5(t) + \frac{1}{c_{17} x2(t)} \right) \cdot c_{18}} + c_{19} \right) + c_{21}$$
$$(14)$$

Agenda

- Objectives of the Tutorial
- Introduction
- Where to get HeuristicLab?
- Plugin Infrastructure
- Graphical User Interface
- Available Algorithms & Problems

- **Demonstration Part I: Working with HeuristicLab**
- **Demonstration Part II: Data-based Modeling**

- Some Additional Features
- Planned Features
- Team
- Suggested Readings
- Bibliography
- Questions & Answers

Some Additional Features

- HeuristicLab Hive
  - parallel and distributed execution of algorithms and experiments on many computers in a network
- Optimization Knowledge Base (OKB)
  - database to store algorithms, problems, parameters and results
  - open to the public
  - open for other frameworks
  - analyze and store characteristics of problem instances and problem classes
- External solution evaluation and simulation-based optimization
  - interface to couple HeuristicLab with other applications (MATLAB, AnyLogic, …)
  - supports different protocols (command line parameters, TCP, …)
- Parameter grid tests and meta-optimization
  - automatically create experiments to test large ranges of parameters
  - apply heuristic optimization algorithms to find optimal parameter settings for heuristic optimization algorithms

Planned Features

- Algorithms & Problems
  - steady-state genetic algorithm
  - unified tabu search for vehicle routing
  - estimation of distribution algorithms
  - …
- Cloud Computing
  - port HeuristicLab Hive to Windows Azure
- Statistics
  - implement statistical tests and automated statistical analysis
- Have a look at the HeuristicLab roadmap
  - http://dev.heuristiclab.com/trac/hl/core/roadmap
- Any other ideas, requests or recommendations?
  - join our HeuristicLab Google group heuristiclab@googlegroups.com
  - write an e-mail to support@heuristiclab.com

HeuristicLab Team

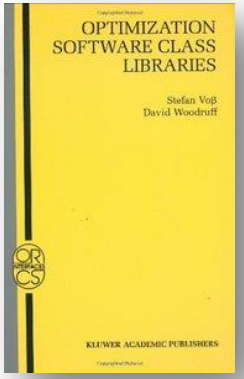
Heuristic and Evolutionary Algorithms Laboratory (HEAL)
School of Informatics, Communications and Media
University of Applied Sciences Upper Austria

Softwarepark 11
A-4232 Hagenberg
AUSTRIA

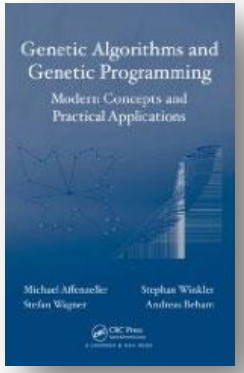
WWW: http://heal.heuristiclab.com

Suggested Readings

- S. Voß, D. Woodruff (Edts.)
  Optimization Software Class Libraries
  Kluwer Academic Publishers, 2002



- M. Affenzeller, S. Winkler, S. Wagner, A. Beham
  Genetic Algorithms and Genetic Programming
  Modern Concepts and Practical Applications
  CRC Press, 2009



Bibliography

- S. Wagner, M. Affenzeller
  HeuristicLab: A generic and extensible optimization environment
  Adaptive and Natural Computing Algorithms, pp. 538-541
  Springer, 2005

- S. Wagner, S. Winkler, R. Braune, G. Kronberger, A. Beham, M. Affenzeller
  Benefits of plugin-based heuristic optimization software systems
  Computer Aided Systems Theory - EUROCAST 2007, Lecture Notes in Computer Science, vol. 4739, pp. 747-754
  Springer, 2007

- S. Wagner, G. Kronberger, A. Beham, S. Winkler, M. Affenzeller
  Modeling of heuristic optimization algorithms
  Proceedings of the 20th European Modeling and Simulation Symposium, pp. 106-111
  DIPTEM University of Genova, 2008

- S. Wagner, G. Kronberger, A. Beham, S. Winkler, M. Affenzeller
  Model driven rapid prototyping of heuristic optimization algorithms
  Computer Aided Systems Theory - EUROCAST 2009, Lecture Notes in Computer Science, vol. 5717, pp. 729-736
  Springer, 2009

- S. Wagner
  Heuristic optimization software systems - Modeling of heuristic optimization algorithms in the HeuristicLab software environment
  Ph.D. thesis, Johannes Kepler University Linz, Austria, 2009.

- S. Wagner, A. Beham, G. Kronberger, M. Kommenda, E. Pitzer, M. Kofler, S. Vonolfen, S. Winkler, V. Dorfer, M. Affenzeller
  HeuristicLab 3.3: A unified approach to metaheuristic optimization
  Actas del séptimo congreso español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados (MAEB'2010), 2010

- S. Wagner, G. Kronberger, A. Beham, M. Kommenda, A. Scheibenpflug, E. Pitzer, S. Vonolfen, M. Kofler, S. Winkler, V. Dorfer, M. Affenzeller
  Architecture and Design of the HeuristicLab Optimization Environment
  Advanced Methods and Applications in Computational Intelligence, vol. 6, pp. 197-261, Springer, 2014

- Detailed list of all publications of the HEAL research group: http://research.fh-ooe.at/de/orgunit/detail/356#showpublications

Questions & Answers

HeuristicLab

http://dev.heuristiclab.com

heuristiclab@googlegroups.com

http://www.youtube.com/heuristiclab

http://www.facebook.com/heuristiclab