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

S. Wagner, G. Kronberger
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

OBERÖSTERREICH

Josef Ressel-Zentrum
HEUREKA!

# Instructor Biographies

HeuristicLab

- Stefan Wagner
  - Full professor for complex software systems (since 2009) University of Applied Sciences Upper Austria
  - Co-founder of the HEAL research group
  - Project manager and chief architect of HeuristicLab
  - PhD in technical sciences (2009) Johannes Kepler University Linz, Austria
  - Associate professor (2005 – 2009) University of Applied Sciences Upper Austria
  - http://heal.heuristiclab.com/team/wagner

- Gabriel Kronberger
  - Full professor for business intelligence (since 2011) University of Applied Sciences Upper Austria
  - Member of the HEAL research group
  - Architect of HeuristicLab
  - PhD in technical sciences (2010) Johannes Kepler University Linz, Austria
  - Research assistant (2005 – 2011) University of Applied Sciences Upper Austria
  - http://heal.heuristiclab.com/team/kronberger



# Agenda

HeuristicLab

- 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 editing and analysis of optimization experiments
- Demonstrate custom algorithms and graphical algorithm designer
- 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 3.3.6 released on January 3rd, 2012



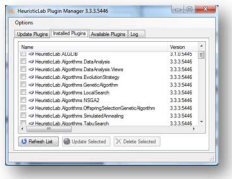

# Where to get HeuristicLab?

- Download binaries
  - deployed as ZIP archives
  - latest stable version 3.3.6
    - released on January 3rd, 2012
  - daily trunk builds
  - http://dev.heuristiclab.com/download
- Check out sources
  - SVN repository
  - HeuristicLab 3.3.6 tag
    - http://dev.heuristiclab.com/svn/hl/core/tags/3.3.6
  - current development trunk
    - http://dev.heuristiclab.com/svn/hl/core/trunk
- License
  - GNU General Public License (Version 3)
- System requirements
  - Microsoft .NET Framework 4.0 Full Version
  - enough RAM and CPU power ;-)



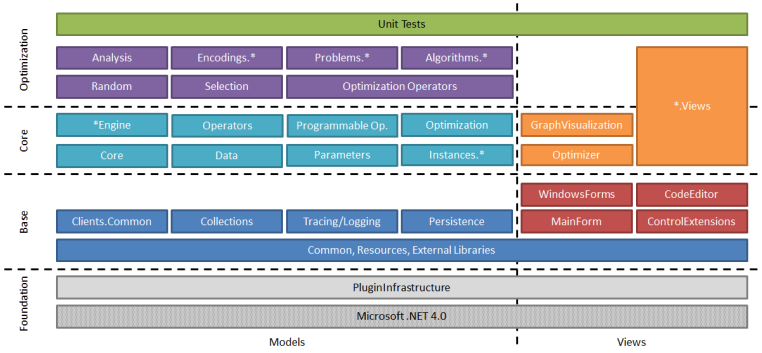
# Plugin Infrastructure

- HeuristicLab consists of many assemblies
  - 94 plugins in HeuristicLab 3.3.6
  - 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

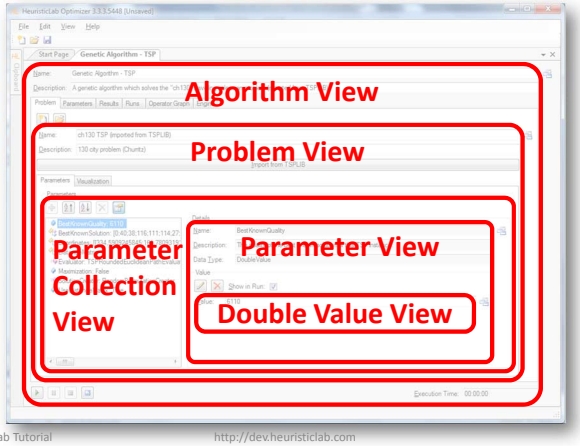


# Graphical User Interface

- HeuristicLab 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



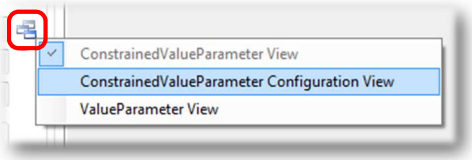
# Graphical User Interface






# 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 & Problems

**Algorithms**

- Evolution Strategy
- Genetic Algorithm
- Genetic Programming
- Island Genetic Algorithm
- Island Offspring Selection Genetic Algorithm
- Local Search
- NSGA-II
- Offspring Selection Genetic Algorithm
- Particle Swarm Optimization
- Robust Taboo Search
- SASEGASA
- Simulated Annealing
- Tabu Search
- User-defined Algorithm
- Variable Neighborhood Search
- Performance Benchmarks
- Cross Validation
- k-Means
- Linear Discriminant Analysis
- Linear Regression
- Multinomial Logit Classification
- Nearest Neighbor Regression and Classification
- Neural Network Regression and Classification
- Random Forest Regression and Classification
- Support Vector Regression and Classification

**Problems**

- Artificial Ant
- Classification
- Clustering
- External Evaluation Problem
- Knapsack
- OneMax
- Quadratic Assignment
- Regression
- Single-Objective Test Function
- Symbolic Classification
- Symbolic Regression
- Traveling Salesman
- User-defined Problem
- Vehicle Routing

# 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

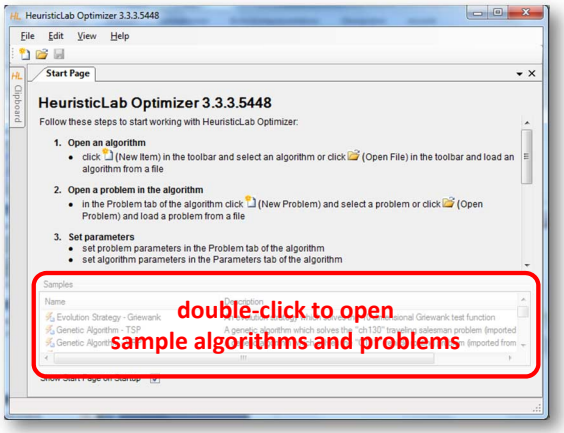


# Demonstration Part I: Working with HeuristicLab

- Create, Parameterize and Execute Algorithms
- Save and Load Items
- Create Batch Runs and Experiments
- Multi-core CPUs and Parallelization
- Analyze Runs
- Analyzers
- Building User-Defined Algorithms

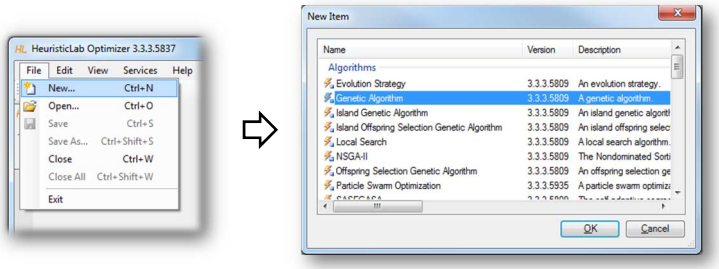


# HeuristicLab Optimizer

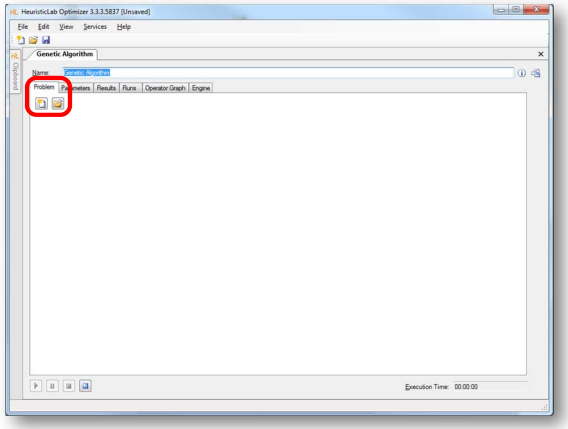





# Create Algorithm

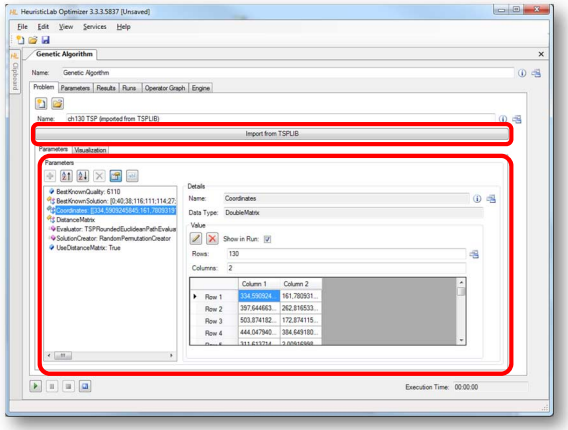


# Create or Load Problem



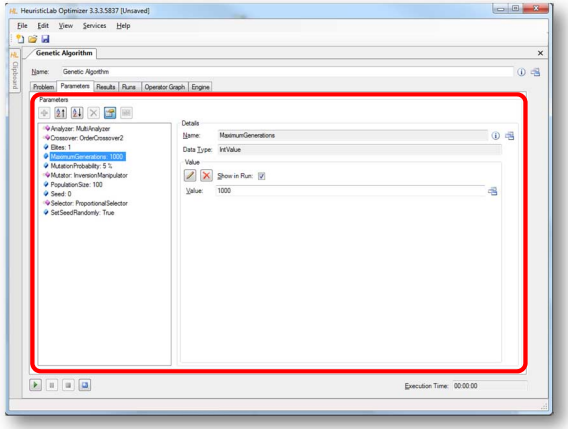


# Import or Parameterize Problem Data



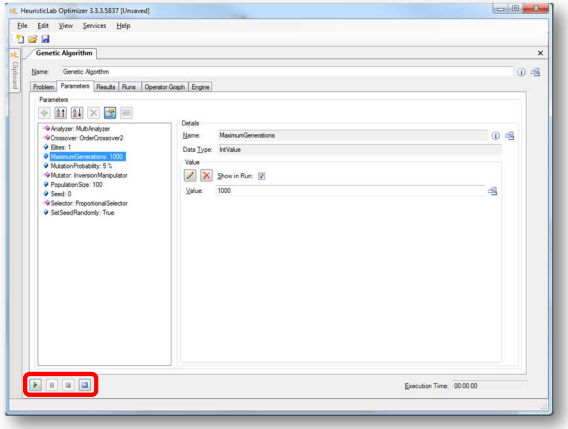


# Parameterize Algorithm

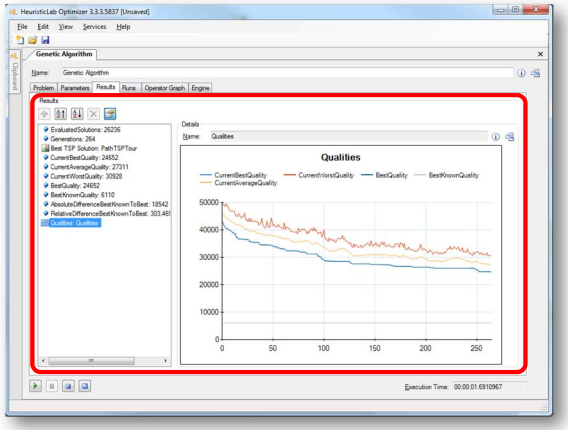




# Start, Pause, Resume, Stop and Reset



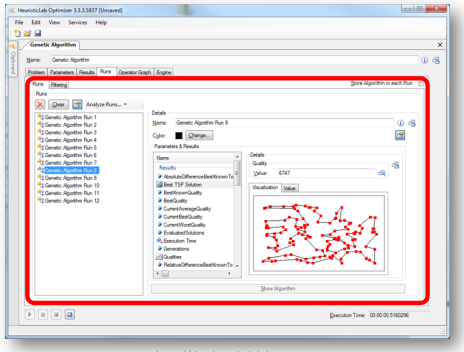
# Inspect Results






# Compare Runs

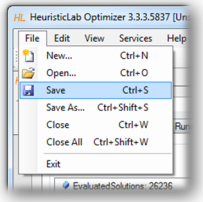
- A run is created each time when the algorithm is stopped
  - runs contain all results and parameter settings
  - previous results are not forgotten and can be compared





# Save and Load

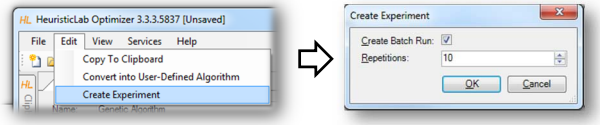
- Save to and load from disk
  - HeuristicLab items (i.e., algorithms, problems, experiments, …) can be saved to and loaded from a file
  - algorithms can be paused, saved, loaded and resumed
  - data format is custom compressed XML
  - saving and loading files might take several minutes
  - saving and loading large experiments requires some memory



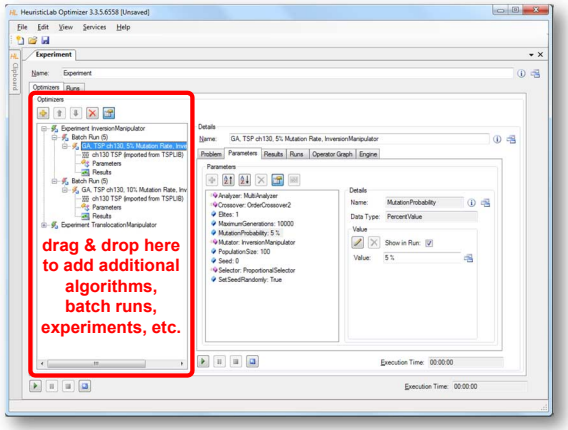



# Create Batch Runs and Experiments

- Batch runs
  - execute the same optimizer (e.g. algorithm, batch run, experiment) several times
- Experiments
  - execute different optimizers
  - suitable for large scale algorithm comparison and analysis
- Experiments and batch runs can be nested
- Generated runs can be compared afterwards

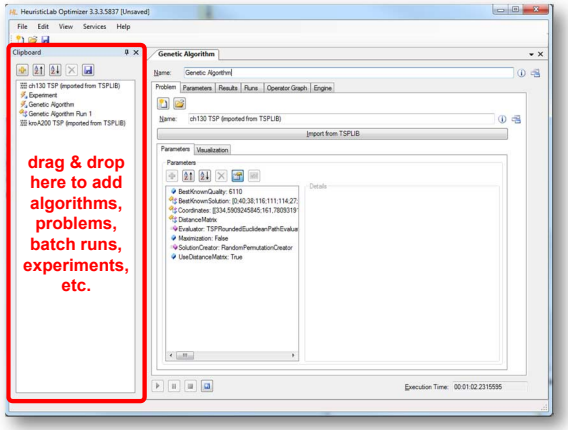


# Create Batch Runs and Experiments





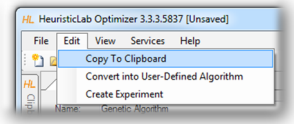

# Clipboard






# Clipboard

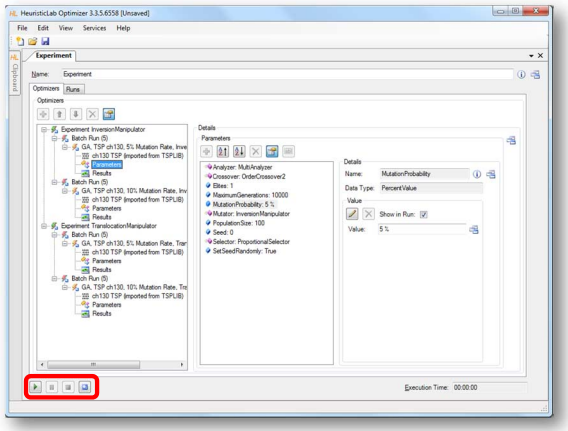
- Store items
  - click on the buttons to add or remove items
  - drag & drop items on the clipboard
  - use the menu to add a copy of a shown item to the clipboard



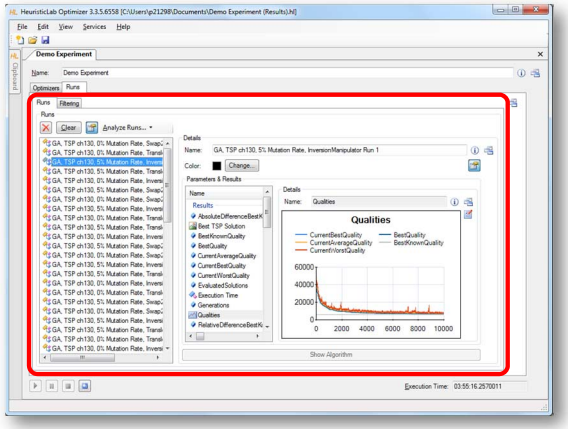

- Show items
  - double-click on an item in the clipboard to show its view
- Save and restore clipboard content
  - click on the save button to write the clipboard content to disk
  - clipboard is automatically restored when HeuristicLab is started the next time



# Start, Pause, Resume, Stop, Reset



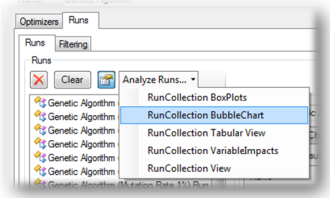
# Compare Runs





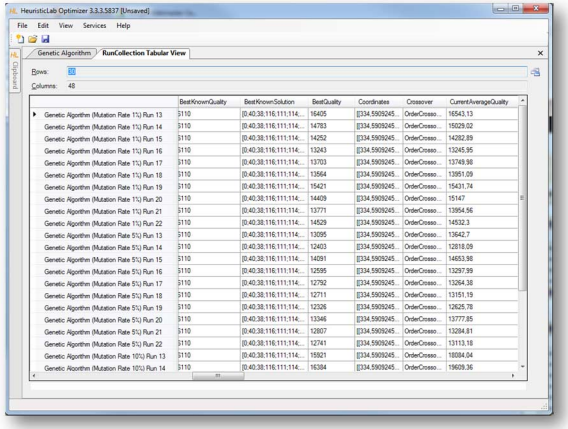
# Analyze Runs

- HeuristicLab provides interactive views to analyze and compare all runs of a run collection
  - textual analysis
    - RunCollection Tabular View
  - graphical analysis
    - RunCollection BubbleChart
    - RunCollection BoxPlots
- Filtering is automatically applied to all open run collection views






# RunCollection Tabular View

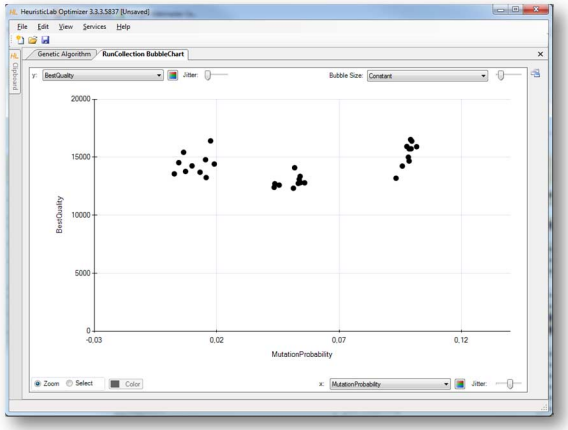




# RunCollection Tabular View

- Sort columns
  - click on column header to sort column
  - Ctrl-click on column header to sort multiple columns
- Show or hide columns
  - right-click on table to open dialog to show or hide columns
- Compute statistical values
  - select multiple numerical values to see count, sum, minimum, maximum, average and standard deviation
- Select, copy and paste into other applications

# RunCollection BubbleChart



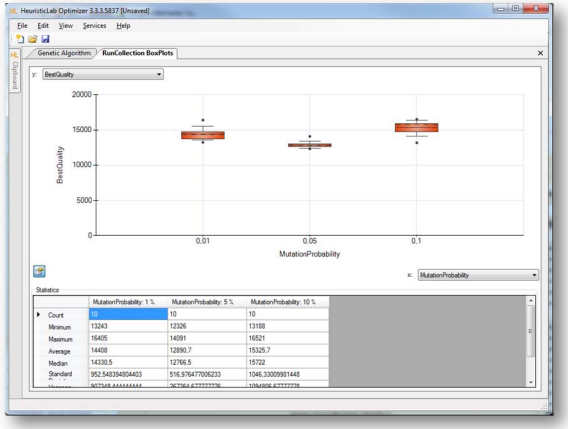



# RunCollection BubbleChart

- Choose values to plot
  - choose which values to show on the x-axis, the y-axis and as bubble size
  - possible values are all parameter settings and results
- Add jitter
  - add jitter to separate overlapping bubbles
- Zoom in and out
  - click on Zoom and click and drag in the chart area to zoom in
  - double click on the chart area background or on the circle buttons beside the scroll bars to zoom out
- Color bubbles
  - click on Select, choose a color and click and drag in the chart area to select and color bubbles
  - apply coloring automatically by clicking on the axis coloring buttons
- Show runs
  - double click on a bubble to open its run
- Export image
  - right-click to open context menu to copy or save image
  - save image as pixel (BMP, JPG, PNG, GIF, TIF) or vector graphics (EMF)
- Show box plots
  - right-click to open context menu to show box plots view



# RunCollection BoxPlots

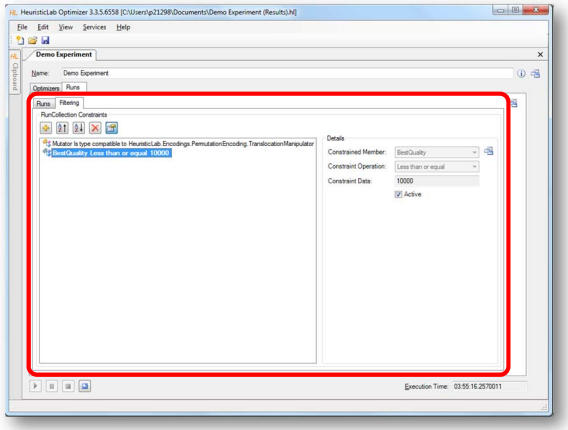





# RunCollection BoxPlots

- Choose values to plot
  - choose which values to show on the x-axis and y-axis
  - possible values are all parameter settings and results
- Zoom in and out
  - click on Zoom and click and drag in the chart area to zoom in
  - double click on the chart area background or on the circle buttons beside the scroll bars to zoom out
- Show or hide statistical values
  - click on the lower left button to show or hide statistical values
- Export image
  - right-click to open context menu to copy or save image
  - save image as pixel (BMP, JPG, PNG, GIF, TIF) or vector graphics (EMF)

# Filter Runs



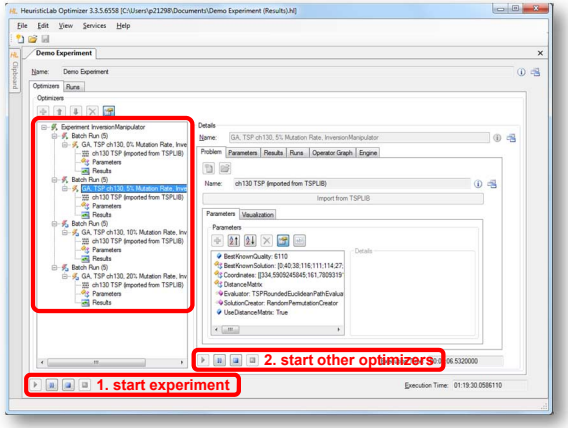


# Multi-core CPUs and Parallelization

- 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

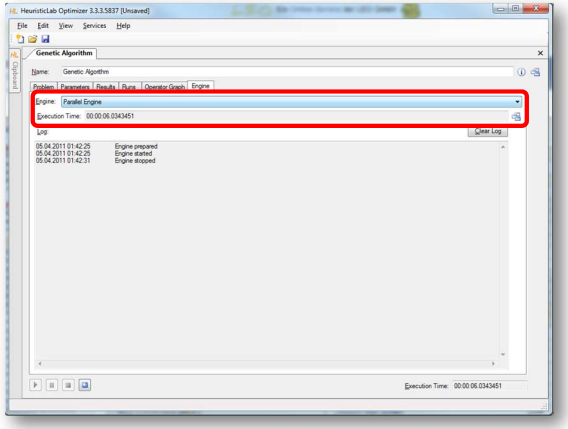


# Parallel Execution of Experiments






# Parallel Execution of Algorithms

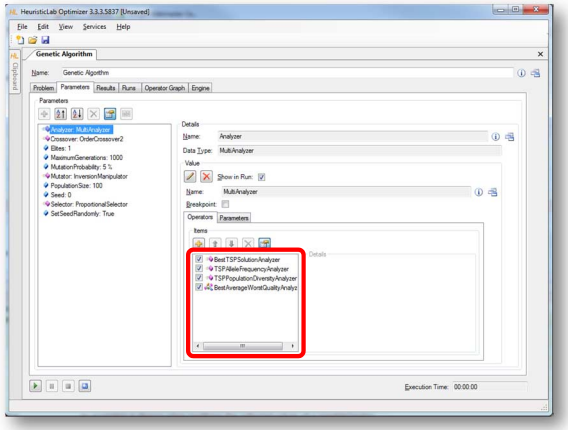


# Analyzers

- Special operators for analysis purposes
  - are executed after each iteration
  - serve as general purpose extension points of algorithms
  - can be selected and parameterized in the algorithm
  - perform algorithm-specific and/or problem-specific tasks
  - some analyzers are quite costly regarding runtime and memory
  - implementing and adding custom analyzers is easy
- Examples
  - TSPAlleleFrequencyAnalyzer
  - TSPPopulationDiversityAnalyzer
  - SuccessfulOffspringAnalyzer
  - SymbolicDataAnalysisVariableFrequencyAnalyzer
  - SymbolicRegressionSingleObjectiveTrainingBestSolutionAnalyzer
  - …

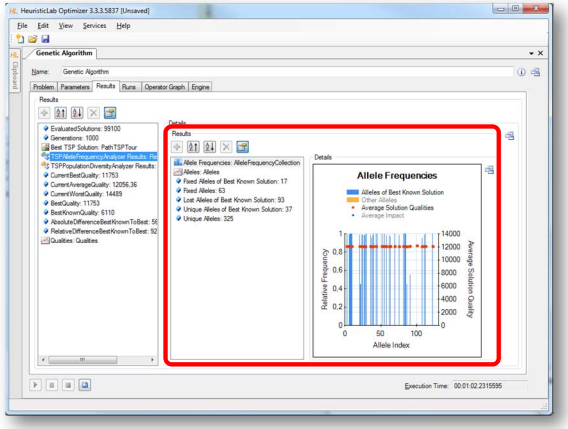


# Analyzers



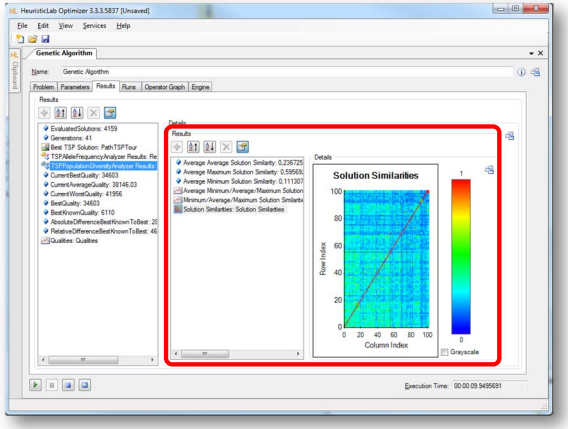


# TSPAlleleFrequencyAnalyzer



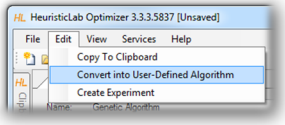


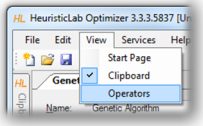

# TSPPopulationDiversityAnalyzer



# 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

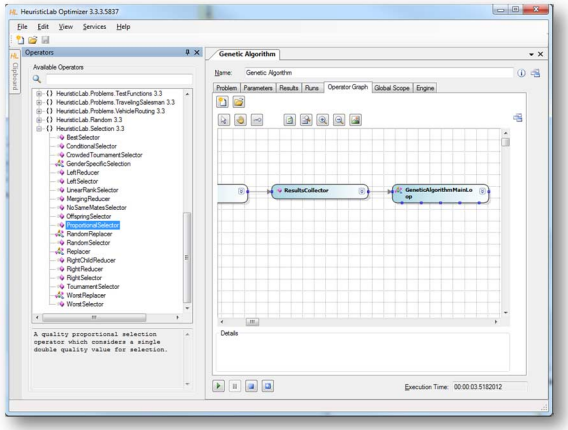



- Operators sidebar
  - drag & drop operators into an operator graph
- Programmable operators
  - add programmable operators in order to implement custom logic in an algorithm
  - no additional development environment needed
- Debug algorithms
  - use the debug engine to obtain detailed information during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

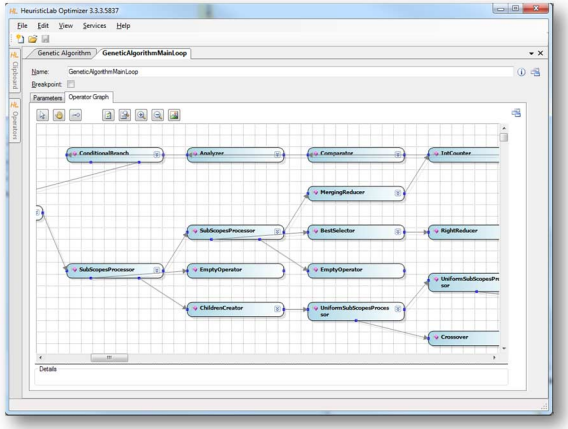


# Building User-Defined Algorithms



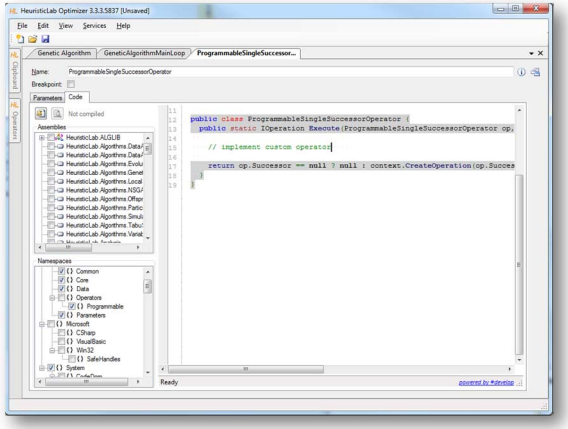


# Building User-Defined Algorithms

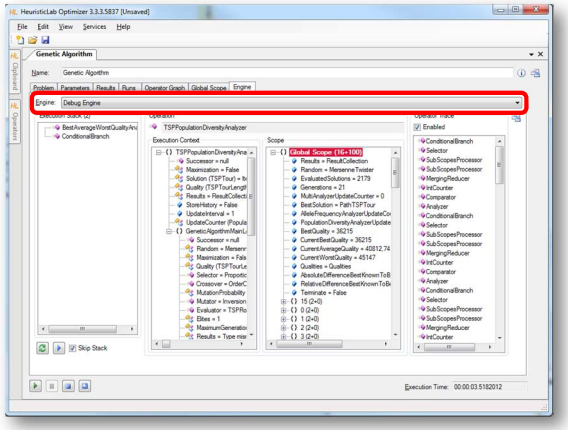




# Programmable Operators



# Debugging Algorithms





# 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
- Model simplification and export
- Variable relevance analysis
- Classification with HeuristicLab



# Introduction to Data-based Modeling

- Dataset: Matrix $(x_{i,j})$ $_{i=1..N, j=1..K}$
  - N observations of K input variables
  - $x_{i,j}$ = i-th observation of j-th variable
  - Additionally: Vector of labels $(y_1 \ldots y_N)^T$
- Goal: learn association of input variable values to labels
- Common tasks
  - Regression (real-valued labels)
  - Classification (discrete labels)
  - Clustering (no labels, group similar observations)

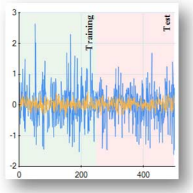
## Data-based Modeling Algorithms in HeuristicLab

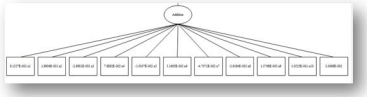
- Symbolic regression and classification based on genetic programming
- External Libraries:
  - Support Vector Machines for Regression and Classification
  - Linear Regression
  - Linear Discriminate Analysis
  - K-Means clustering

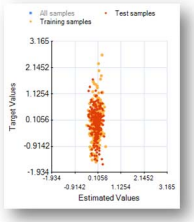


## Case Studies

- Demonstration
  - problem configuration
    - data import
    - target variable
    - input variables
    - data partitions (training and test)
  - analysis of results
    - accuracy metrics
    - visualization of model output







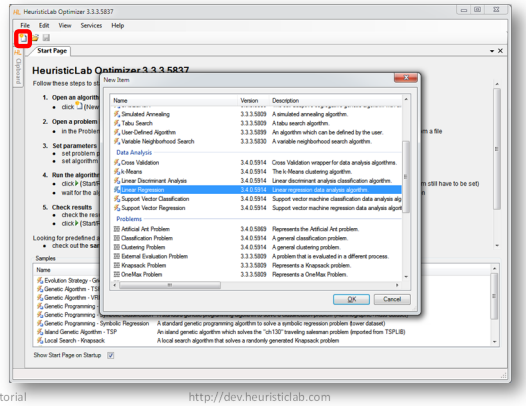

## Case Study: Regression

- Poly-10 benchmark problem dataset
  - 10 input variables $x_1 \dots x_{10}$
  - $y = x_1 \cdot x_2 + x_3 \cdot x_4 + x_5 \cdot x_6 + x_1 \cdot x_7 \cdot x_9 + x_3 \cdot x_6 \cdot x_{10}$
  - non-linear modeling approach necessary
  - frequently used in GP literature
  - download http://dev.heuristiclab.com/AdditionalMaterial#GECCO2012

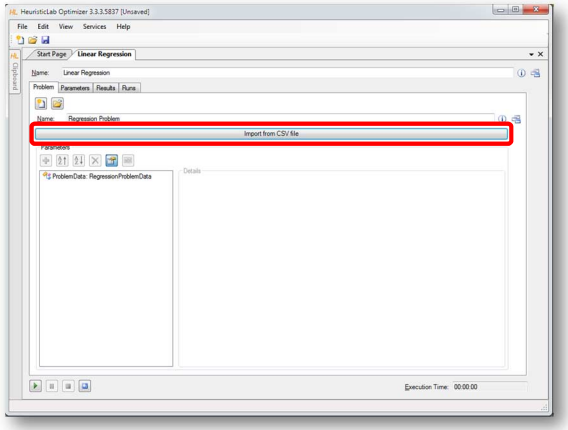


## Linear Regression

- Create new algorithm

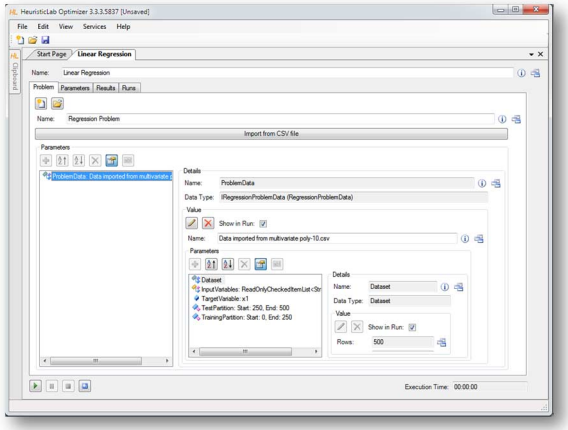


# Import Data from CSV-File



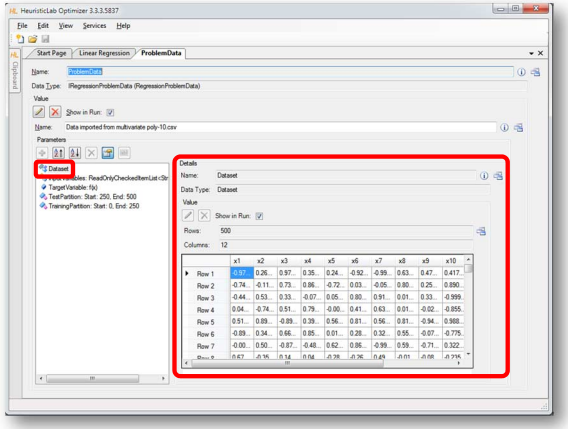


# Inspect and Configure Dataset



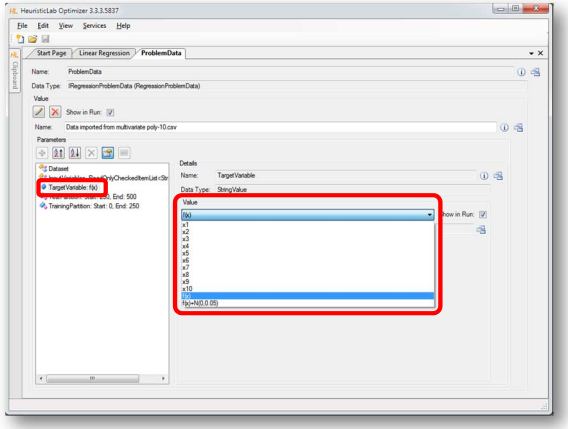


# Inspect Imported Data

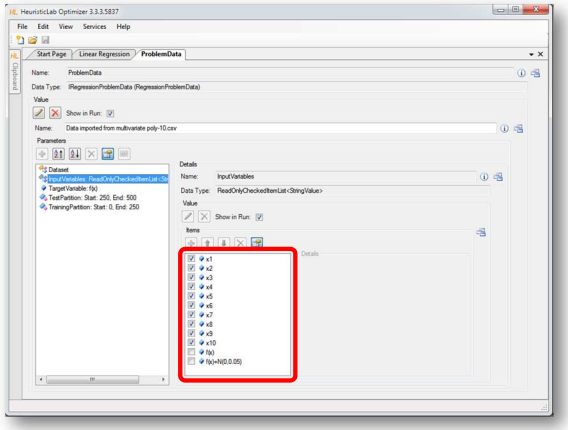




# Set Target Variable

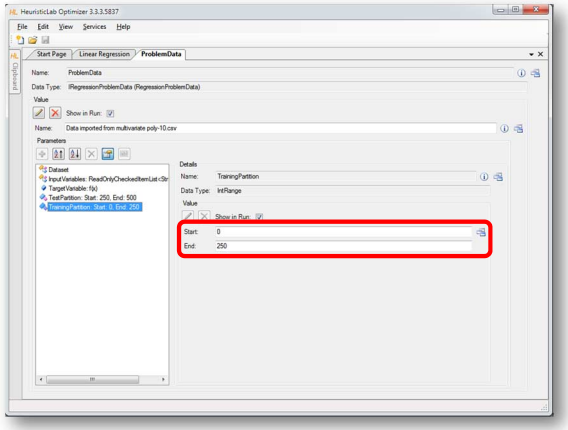


# Select Input Variables



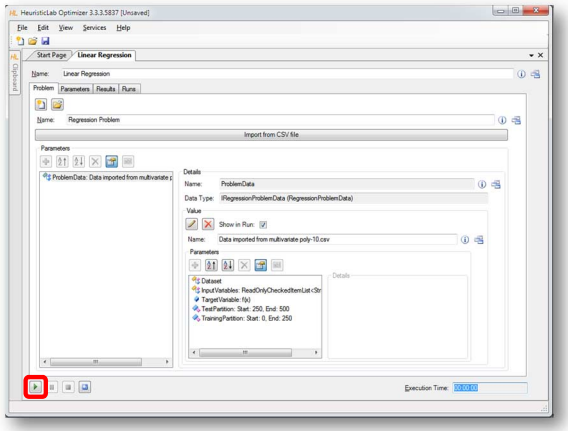


# Configure Training and Test Partitions



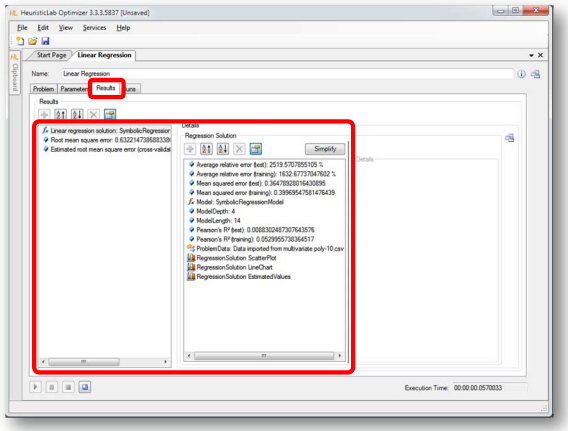


# Run Linear Regression

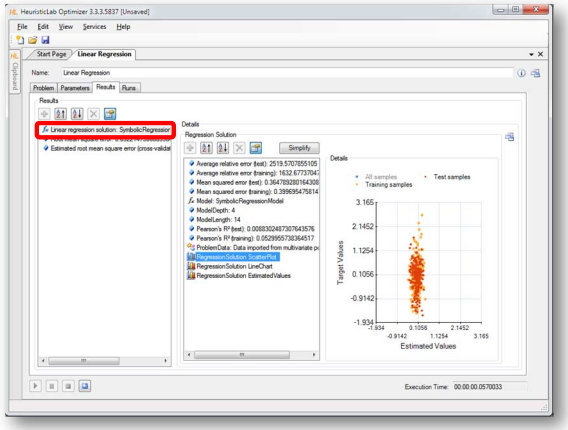




# Inspect Results

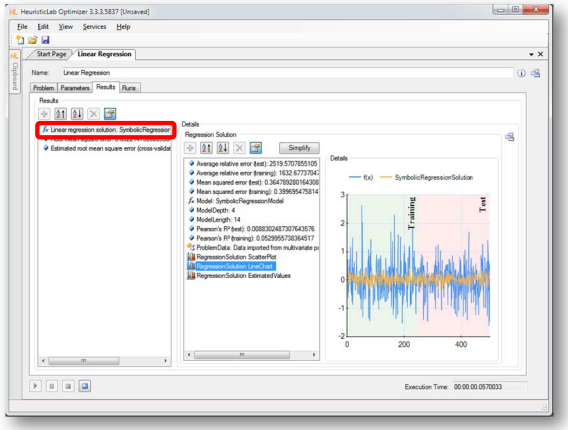


## Inspect Scatterplot of Predicted and Target Values



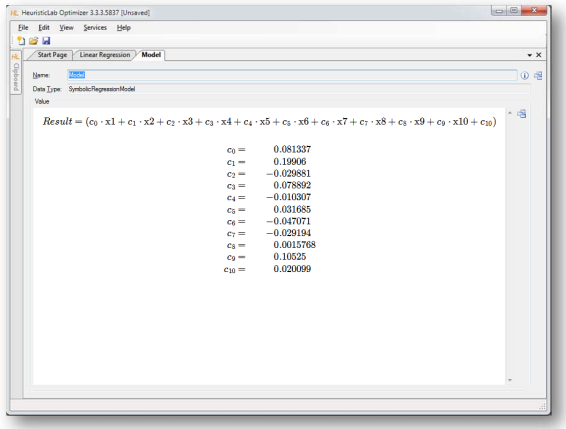


## Inspect Linechart





## Inspect the Model

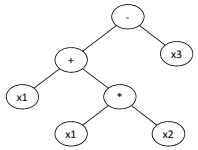




## Symbolic Regression with HeuristicLab

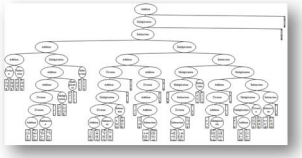
- Linear regression produced an inaccurate model.
- Next: produce a nonlinear symbolic regression model using genetic programming

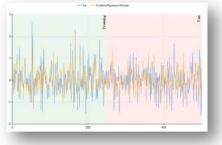
- Genetic programming
  - evolve variable-length models
  - model representation: symbolic expression tree
  - structure and model parameters are evolved side-by-side
  - white-box models



# Symbolic Regression with HeuristicLab

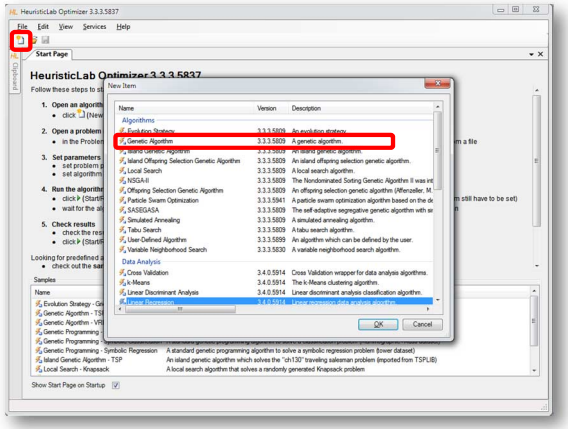
- Demonstration
  - problem configuration
  - function set and terminal set
  - model size constraints
  - Evaluation
- Algorithm configuration
  - selection
  - Mutation
- Analysis of results
  - model accuracy
  - model structure and parameters





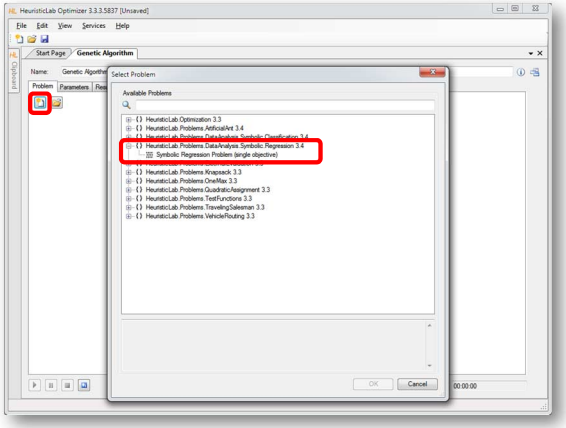


# Create New Genetic Algorithm



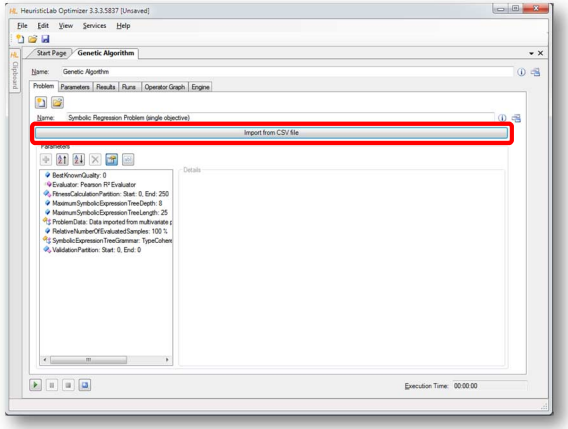


# Create New Symbolic Regression Problem

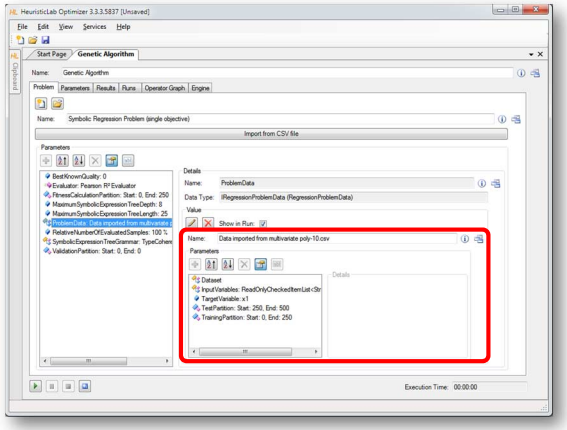




# Import Data

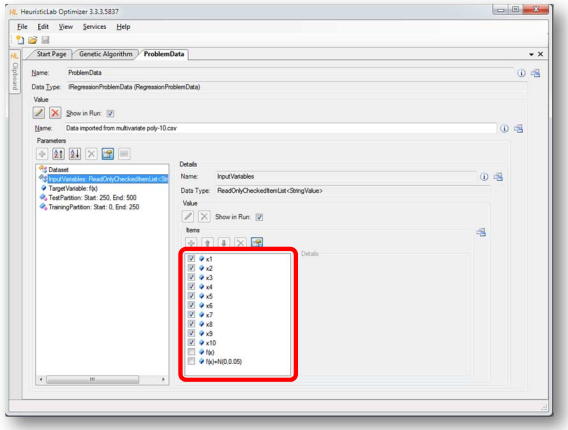


## Inspect Data and Configure Dataset



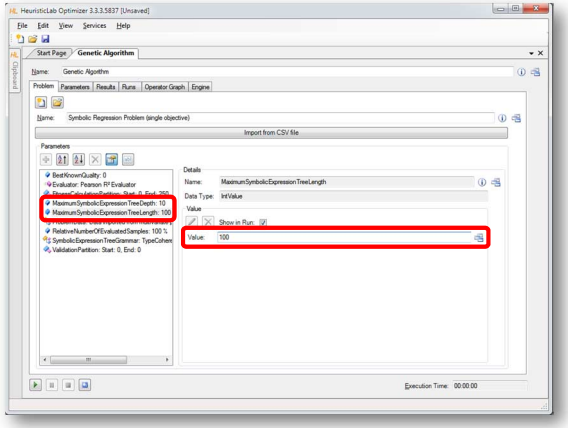


## Set Target and Input Variables



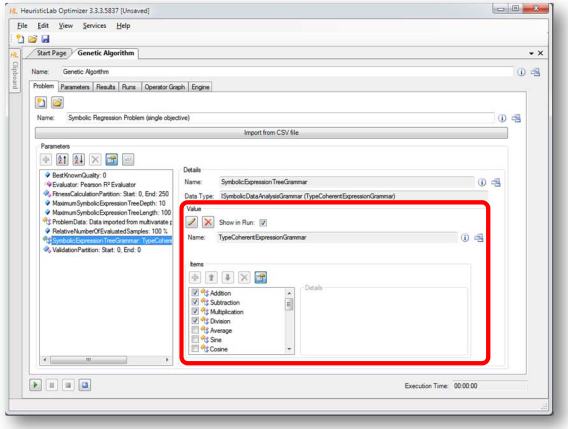


## Configure Maximal Model Depth and Length

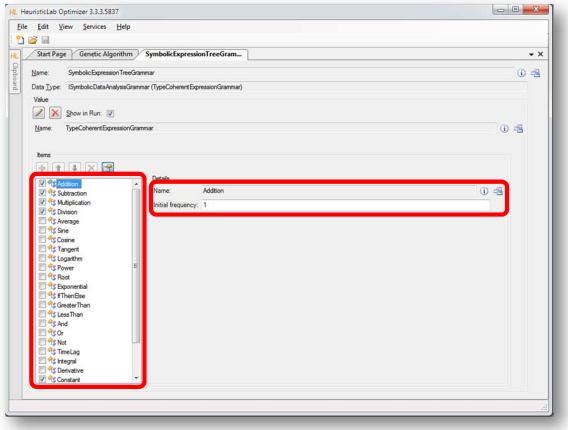




## Configure Function Set (Grammar)

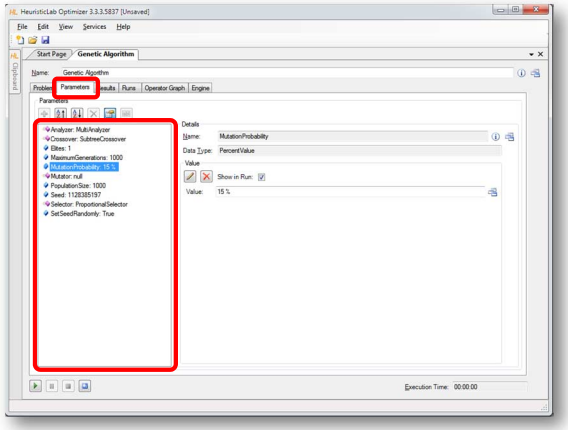


# Configure Function Set (Grammar)



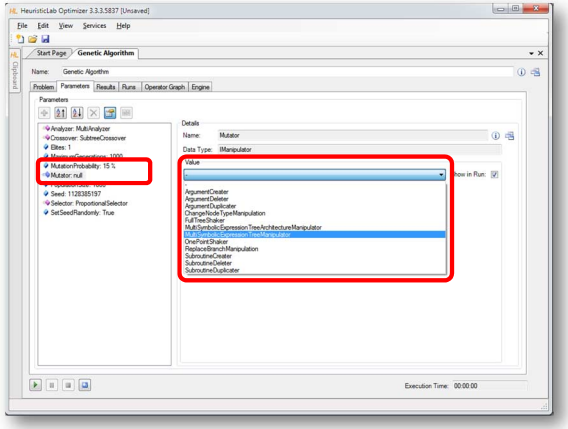


# Configure Algorithm Parameters



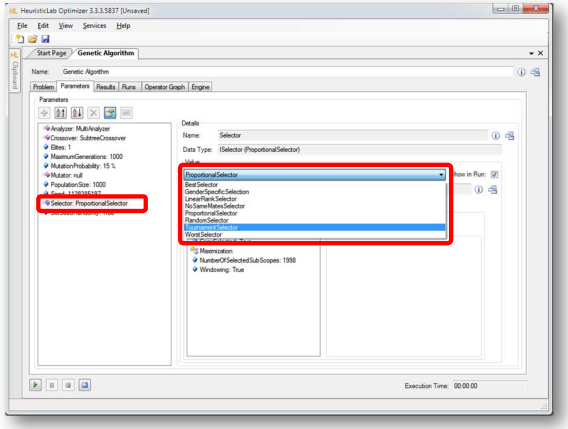


# Configure Mutation Operator

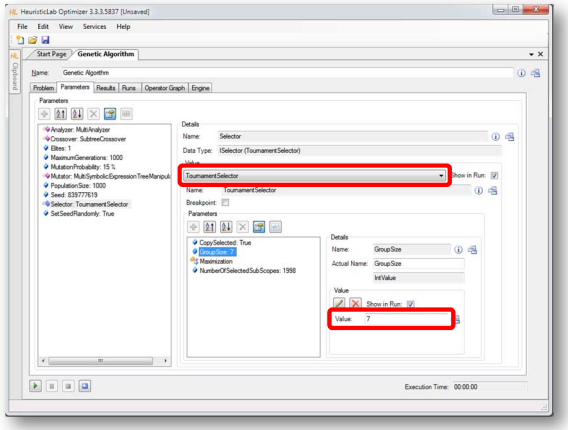




# Configure Selection Operator

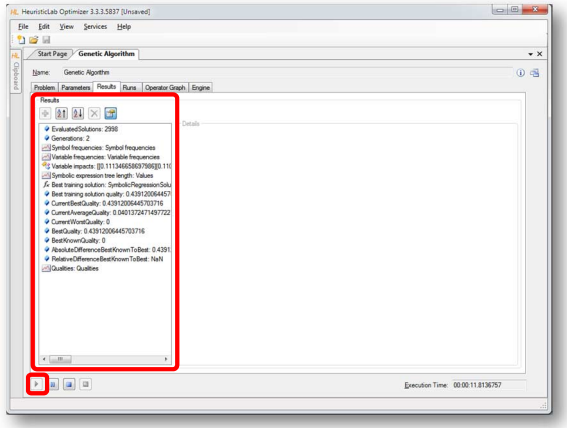


# Configure Tournament Group Size



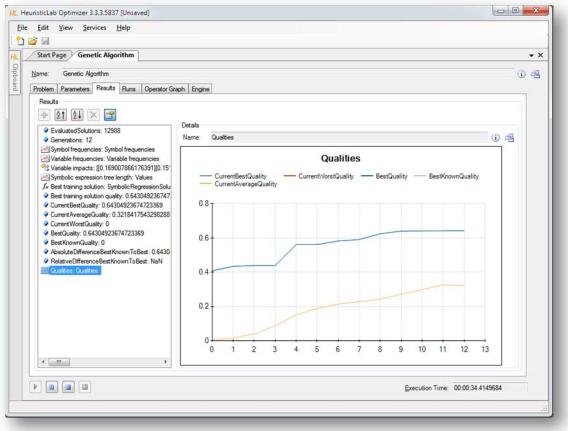


# Start Algorithm and Inspect Results



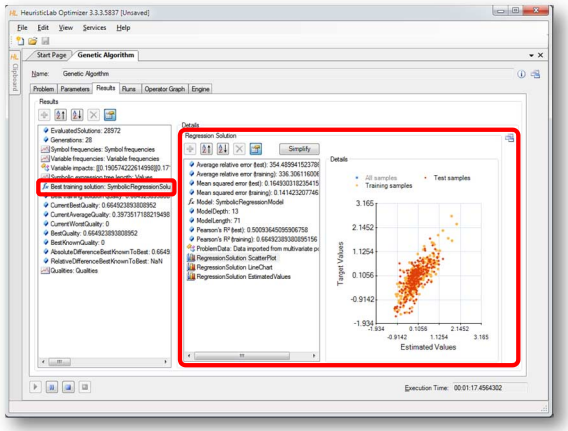


# Inspect Quality Chart

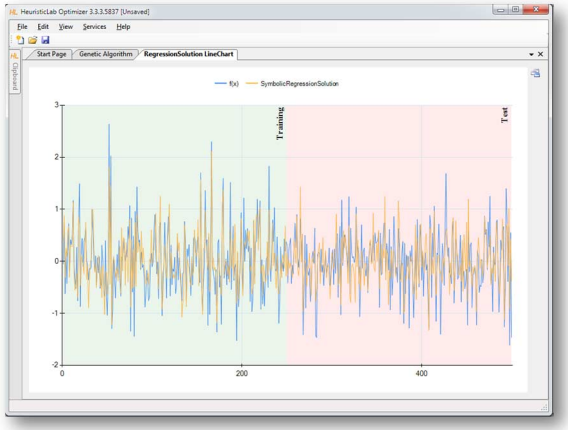





# Inspect Best Model on Training Partition

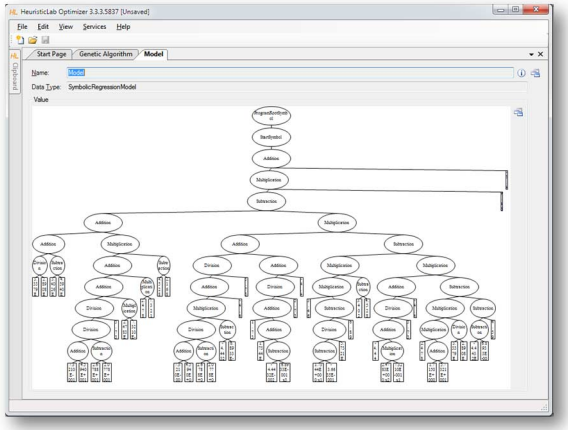


## Inspect Linechart of Best Model on Training Partition





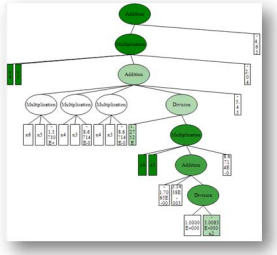

## Inspect Structure of Best Model on Training Partition

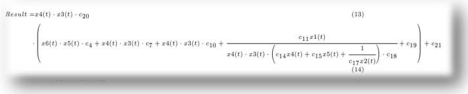




## Model Simplification and Export

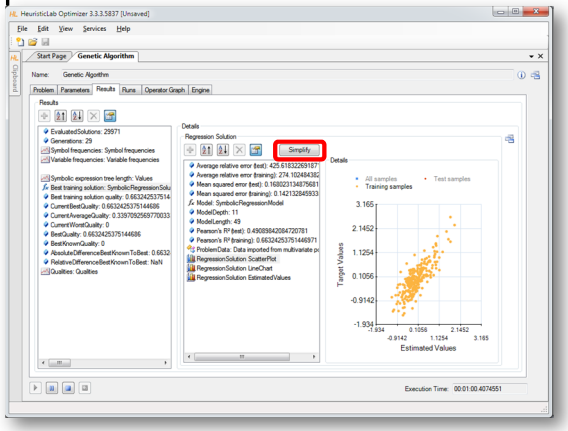
- Demonstration
  - automatic simplification
  - visualization of node impacts
  - manual simplification
    - online update of results
  - model export
    - MATLAB
    - LaTeX



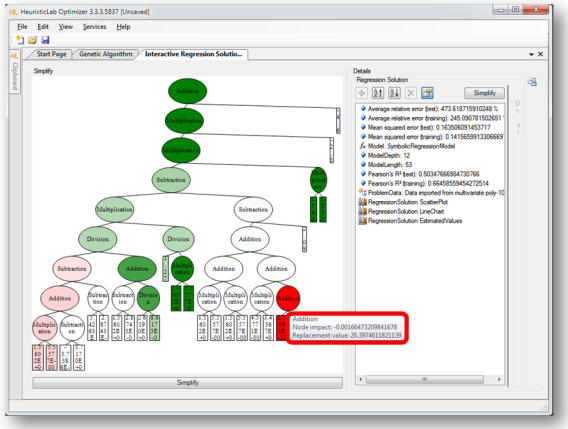




## Detailed Model Analysis and Simplification

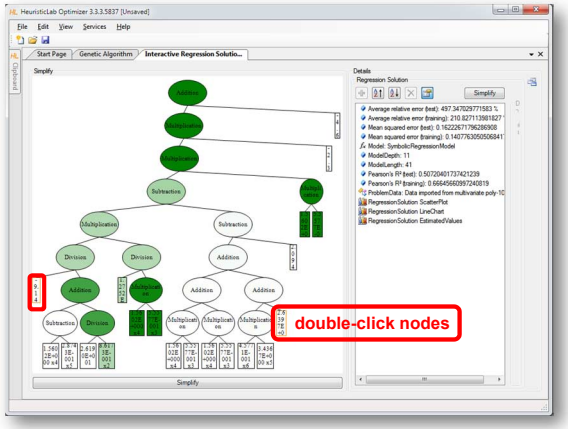


# Symbolic Simplification and Node Impacts



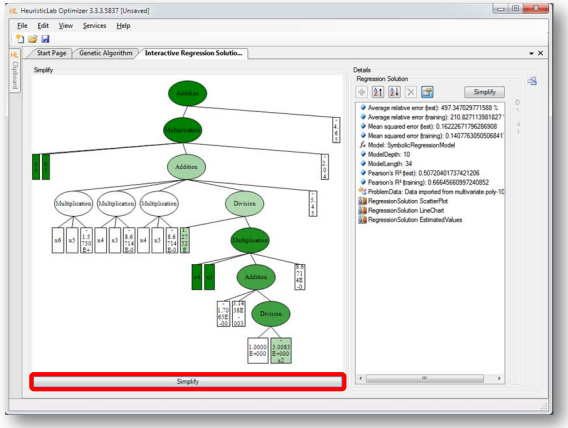


# Manual Simplification





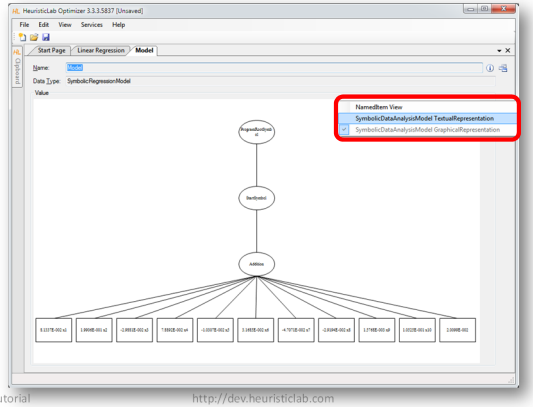

# Automatic Symbolic Simplification



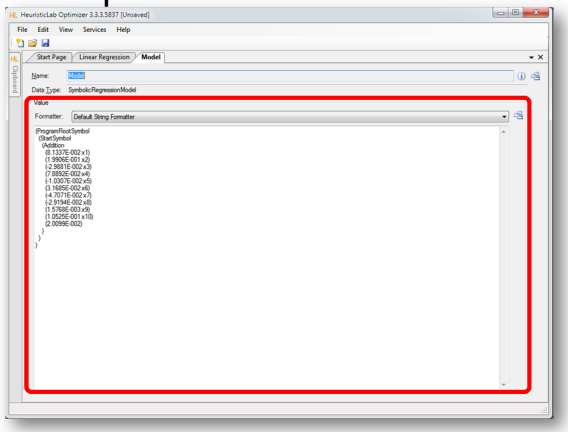


# Textual Representations Are Also Available

- Use *ViewHost* to switch to textual representation view.

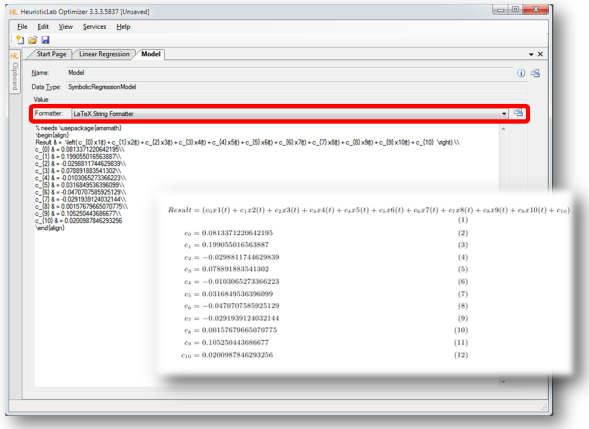


# Default Textual Representation for Model Export



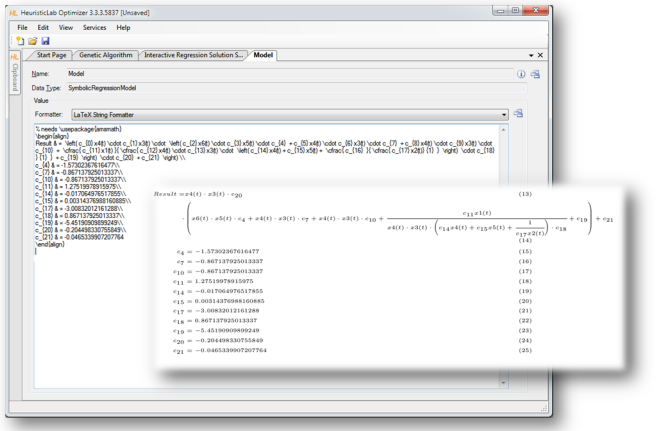


# Textual Representation for Export to LaTeX





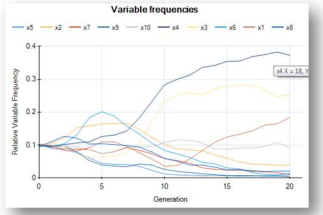
# LaTeX Export





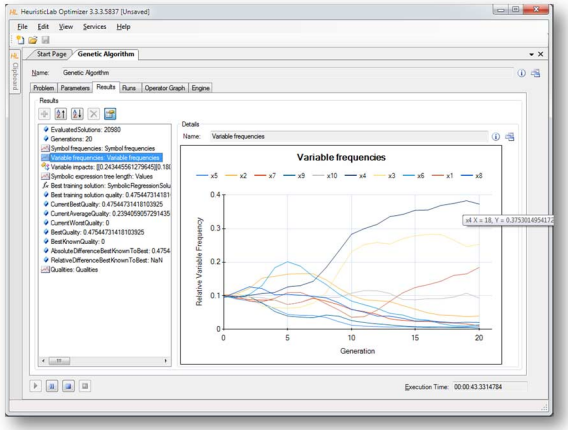
# Variable Relevance Analysis

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



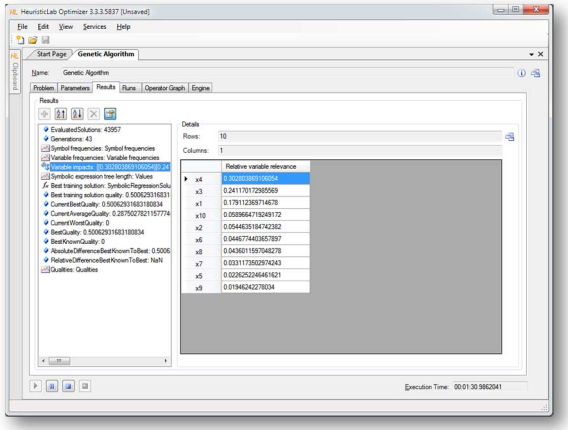

| | Relative variable relevance |
|---|---|
| x4 | 0.302803869106054 |
| x3 | 0.241170172985569 |
| x1 | 0.179112369714678 |
| x10 | 0.0589664719249172 |
| x2 | 0.0544635184742382 |
| x6 | 0.0446774403657897 |
| x8 | 0.0436011597048278 |
| x7 | 0.0331173502974243 |
| x5 | 0.0226252246461621 |
| x9 | 0.0194624278034 |

# Inspect Variable Frequency Chart



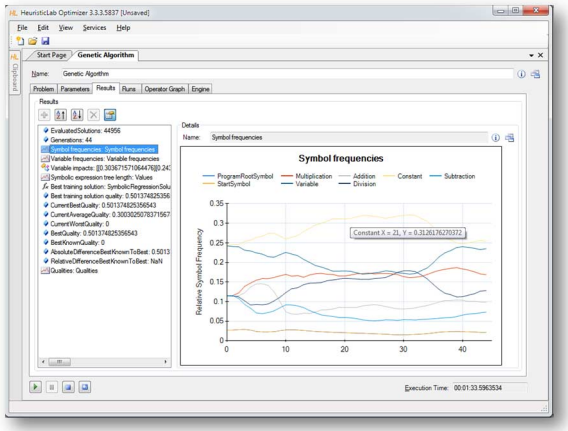



# Inspect Variable Impacts





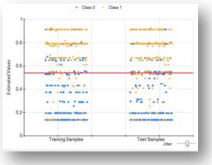
# Inspect Symbol Frequencies

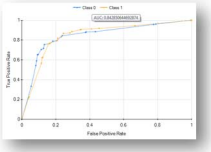





# Classification with HeuristicLab

- Symbolic classification
  - evolve discriminating function using GP
  - find thresholds to assign classes
- Demonstration
  - real world medical application
  - model accuracy
  - visualization of model output
    - discriminating function output
    - ROC-curve
    - confusion matrix





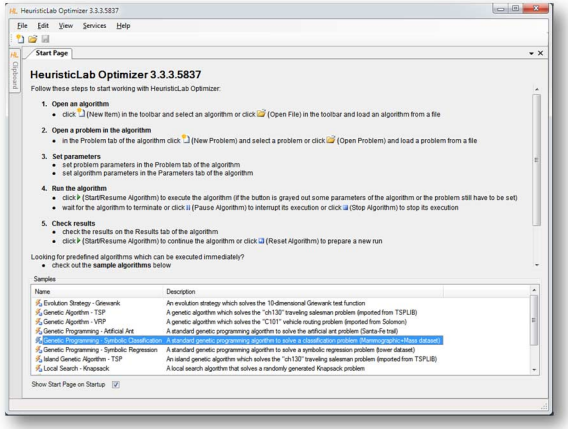
| | Actual Class 0 | Actual Class 1 |
|---|---|---|
| Predicted Class 0 | 197 | 29 |
| Predicted Class 1 | 64 | 190 |

# Case Study: Classification

- Real world medical dataset (*Mammographic Mass*) from UCI Machine Learning Repository
  - data from non-invasive mammography screening
  - variables:
    - patient age
    - visual features of inspected mass lesions: shape, margin, density
  - target variable: severity (malignant, benign)
  - download
    http://dev.heuristiclab.com/AdditionalMaterial#GECCO2012

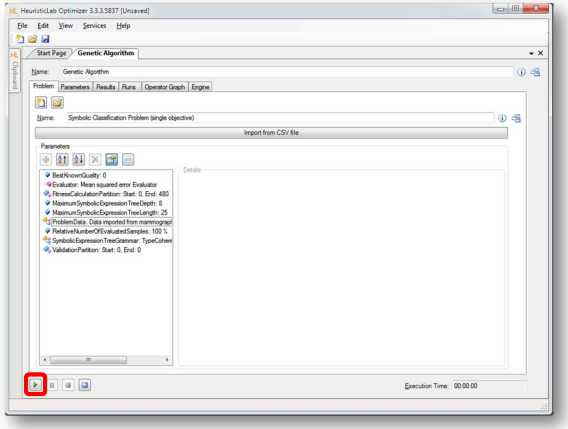


# Open Sample



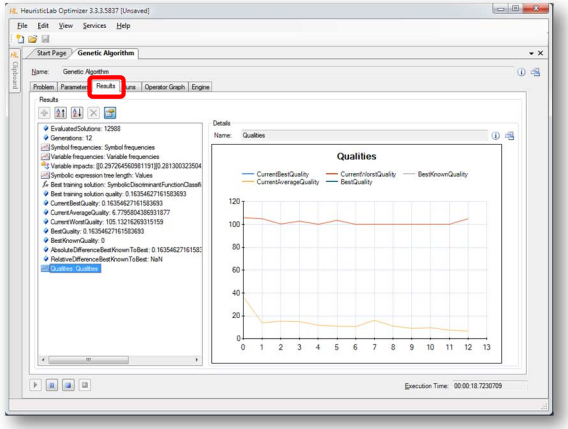



# Configure and Run Algorithm

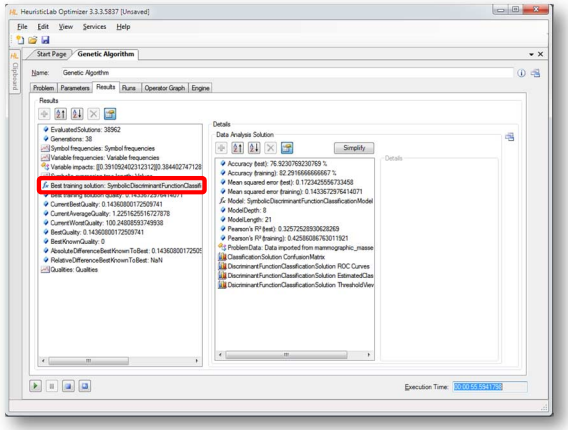





# Inspect Quality Linechart

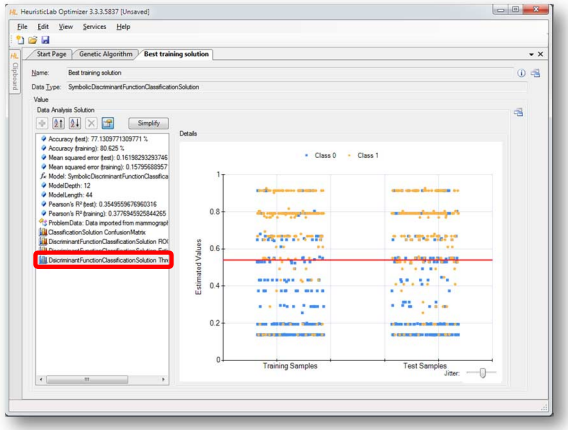


# Inspect Best Training Solution



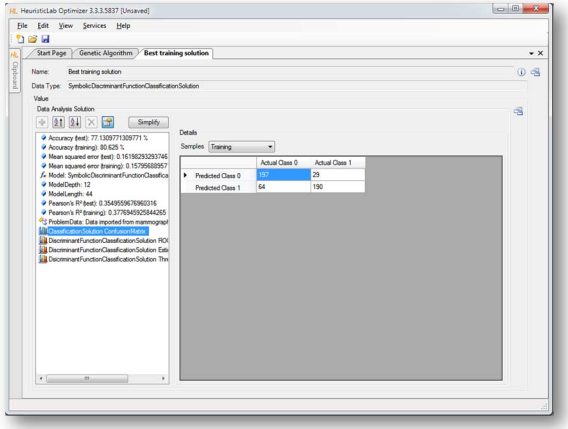


# Inspect Model Output and Thresholds



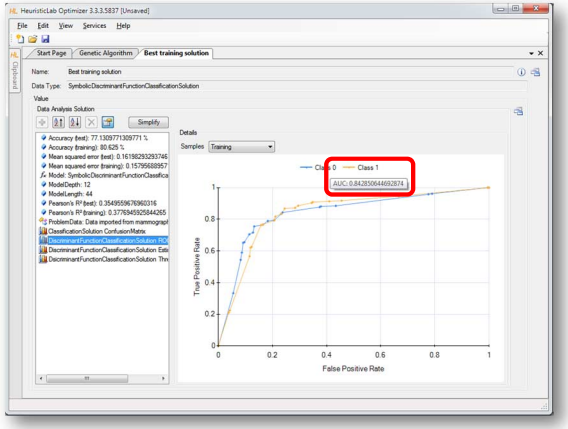


# Inspect Confusion Matrix



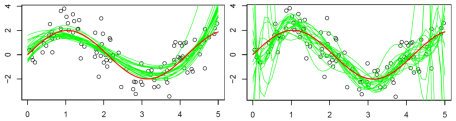


# Inspect ROC Curve

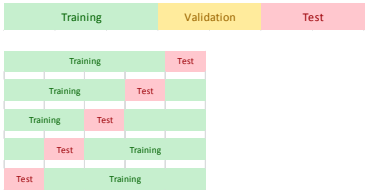


# Validation of Results

- Overfitting = memorizing data



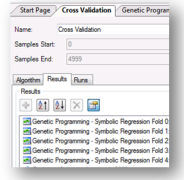
- Strategies to reduce overfitting
  - validation partition
  - cross-validation

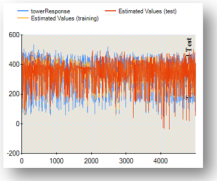


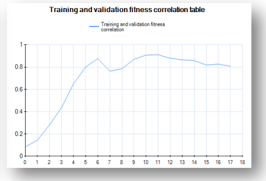



# Validation of Results

- Demonstration
  - Configuration of a validation set
  - Inspection of best solution on validation set
  - Analysis of training- and validation fitness correlation
  - Cross-validation
    - Configuration
    - Analysis of results

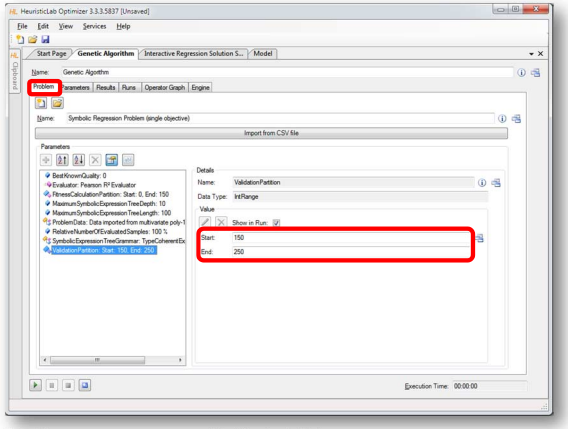






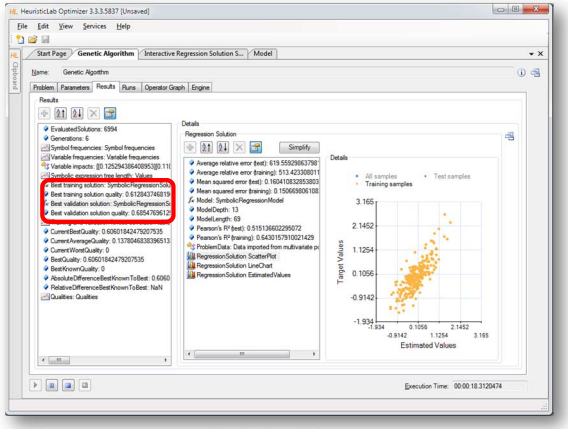



# Configuration of Validation Partition

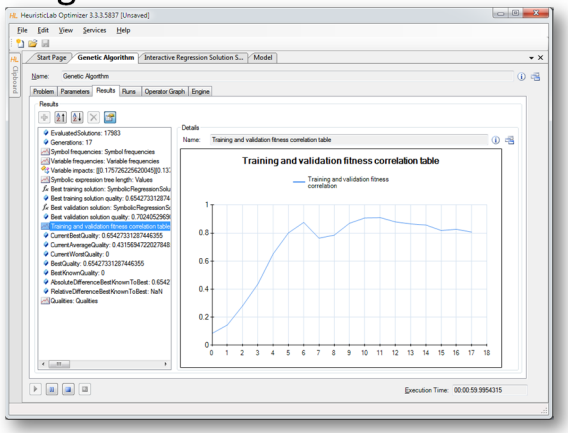




# Inspect Best Model on Validation Partition



# Inspect Linechart of Correlation of Training and Validation Fitness






# 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
  - scatter search
  - …
- Cloud Computing
  - port HeuristicLab Hive to Windows Azure
- Linux
  - port HeuristicLab to run on Mono and Linux machines
- 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 or
  - 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

HEAL
Heuristic and Evolutionary Algorithms Laboratory

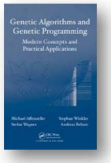
FH OBERÖSTERREICH



# 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 Metaheuristicas, Algoritmos Evolutivos y Bioinspirados (MAEB'2010), 2010
- Detailed list of all publications of the HEAL research group: http://research.fh-ooe.at/de/orgunit/detail/356#showpublications



# Questions & Answers

http://dev.heuristiclab.com

heuristiclab@googlegroups.com