Poznań University of Technology
Faculty of Computing
Institute of Computing Science

RULERANK ULTIMATE DESKTOP EDITION USER MANUAL 1.0

Author: Piotr Jówko

Poznań, September 2, 2018

About RUDE application

RuleRank Ultimate Desktop Edition (RUDE) is JavaFX application with uses java Rough Set library (jRS). Application supports Windows and Linux operating systems. It requires Java 10 to run.

It provides rich user interface to create own experiments solving ranking problems using Dominance-based Rough Set and Variable Consistency Dominance-based Rough Set Approach.

First chapter describes RUDE installation process on local computer. Rest of the document describes all features of this application.

Contents

1 Installation

Application supports Windows and Linux. It requires Java 10 JRE or JDK to be installed. The newest version of Java can be downloaded here.

RUDE application doesn't require installation. You simply need to download archived application and extract it to some directory. You can download it from here.

Archive content description:

- **data directory** - it contains configuration files, see Data configuration

- **workspace** - it is default directory for storing all files related with experiments. It also contains some example experiments with result files.

- **readme.txt** - it contains information about requirements and running application

- **RuleRank jar file** - it contains compiled sources for this application

- **run.* files** - it contains files with will run application after double click. Bat file is for Windows, sh file is for Linux.

You can run application by double clicking on run file. It will open console/terminal and run UI application. In case of any problems with running RUDE, error logs will be displayed in console/terminal. If you encounter any errors, it is recommended to execute run file in cmd/powershell or terminal, instead of double clicking. It will not close console/terminal after application exit.

2 Application overview

In this chapter we will provide general overview of main application window.

Figure 1: Main window
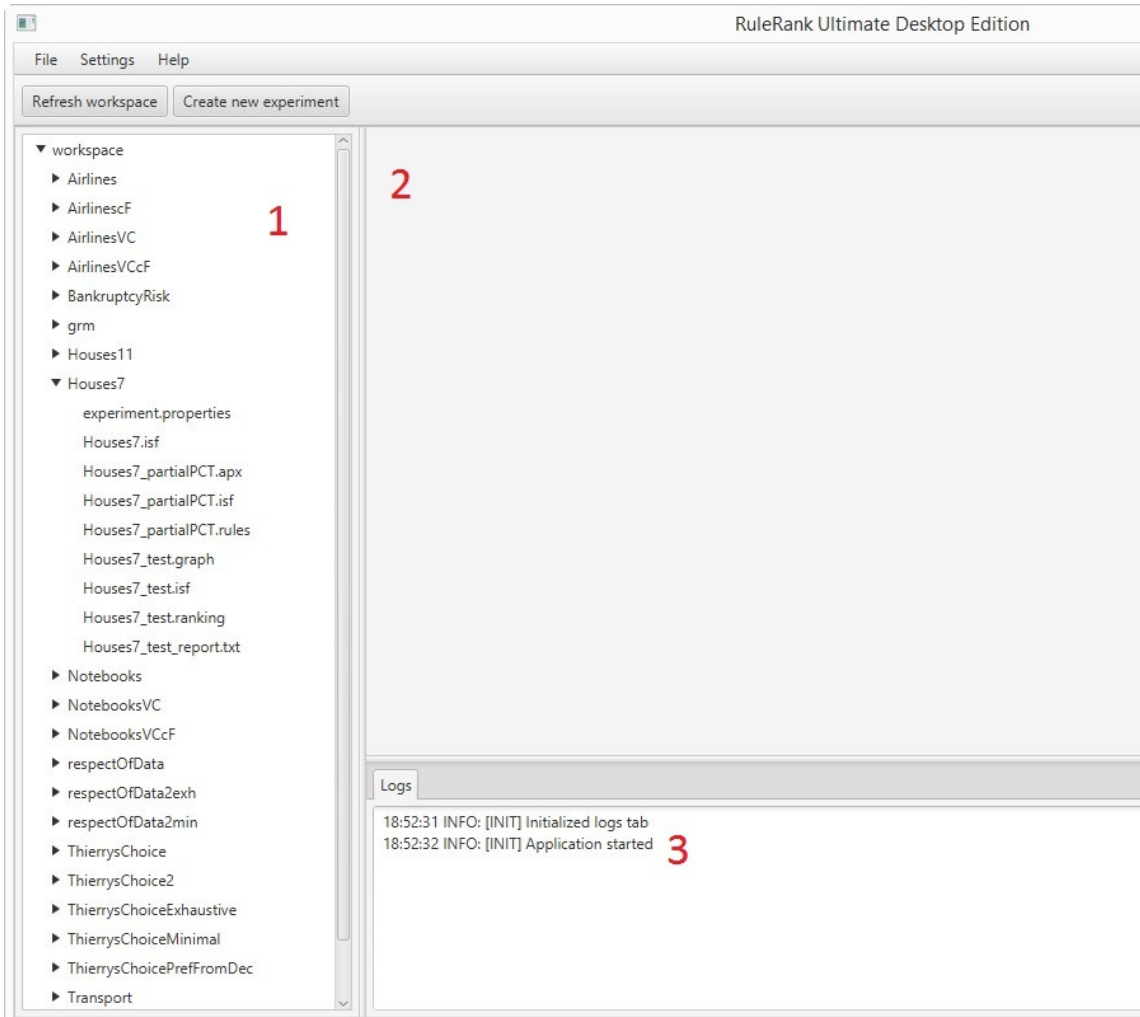
There are three main panels in RUDE application:

- **workspace (1)** - it manages all files

- **upper panel (2)** - it displays most of the information. It can display multiple tabs like web browser. Most of the work will be done here.

- **lower panel (3)** - it contains logs and can display additional information for some upper tabs. It can display multiple tabs like web browser.

On top of main window you can find menu:

- **File** - it currently enables to quit program

- **Settings** - it contains user settings for RUDE application

- **Help** - it contains About and Help modal dialogs.

Below menu you can find toolbar with actions, with will be described in Workspace management section.

2.1 Tabs management

New tab is created after opening file from workspace. You can do this by double clicking on file in workspace tree. Many tabs can be opened at once, but only one for each file. After pressing right click on tab header, you will see context menu containing actions with can close many tabs at once. If you made some changes in editable tab form, you will be asked for confirmation, if you try to close it by context menu action. Same confirmation appears when you try to exit application or close single tab.

Logs tab display all logs from application. Currently there are three logs levels:

- **INFO** - displays some useful information about performed actions

- **WARN** - indicates some misconfiguration or action aborting in certain conditions. It also indicates errors handled by application.

- **ERROR** - displays errors not handled properly by application

Logs can be copied by selecting text and pressing Ctrl + C. All logs can be cleared by right clicking on logs panel and choosing "Clear logs" option.

Next sections describes all application windows with are available for user.

2.2 High level workflow

This section provides high level steps for performing experiment. As a result of experiment, application will produce ranking of objects, from best to worse, basing on gain/cost criteria configured in isf tables and initial pairs comparison/ranking.

1. Create experiment directory with will contain all experiment files. See Workspace section.

2. Create learning data file (.isf) containing objects on with ranking will be created. See Isf edition section.

3. (Optional) Create test data file (.isf) See Isf edition section.

4. Create experiment properties file and configure experiment. See Properties section.

5. Create sample ranking or pairs comparisons in experiment properties form. See Ranking configuration section, see Pairs configuration section

6. Run experiment in properties form. See Experiment running section.

7. See logs and generated files for experiment results.

Experiment will perform analysis using Dominance-based Rough Set Approach (DRSA) or Variable Consistency Dominance-based Rough Set Approaches (VC-DRSA) (depending on options selected).

At first, it will generate Pairwise Comparison Table (PCT), where all objects are compared between each other on all criteria. Partial PCT table will be saved to .isf file, because full table is to big to save. See Partial comparison table section.

From pairwise comparison table dominance cones are calculated, with represents approximate outranking and non-outranking relations in PCT. Also some metrics are calculated. There are saved in text format to .apx file. See Partial comparison table section.

In next step application induces rules from Pairwise Comparison Table. S relation indicates outranking relation, Sc indicates non-outranking relation. Rules can be certain or possible. Also rule statistics are generated and saved in .rules file next to induced rules. See Rules section.

Next, preference graph is generated, in with nodes represents objects from isf table and arcs represents S or Sc relations. Arcs in graph can be also weighted, where weight represents satisfaction degrees of the preference relations. Weight for relation (arc) is calculated by aggregating confidence of covering rules. Preference graph is saved to .graph file. See Graph visualisation section.

As the last step, final ranking is calculated from preference graph. It contains objects positions and scores. Ranking is saved to .ranking file. In some cases also report can be generated and saved to .txt file. See Ranking section.

3 User settings

In this chapter we will describe process of basic application configuration.

To configure application, go to menu in main window, choose Settings and then User settings. RUDE will open modal window. Default values are presented on image below.
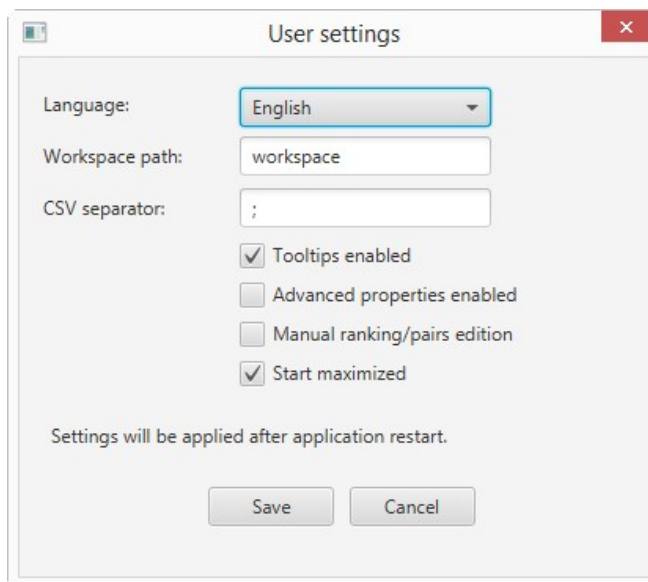

Figure 2: User settings modal dialog

From this window you can configure following options:

- **Language** - by default only English language can be chosen. Currently other languages are not supported. But you can add your own language translation, see Custom language support section.

- **Workspace path** - you can configure directory path for your workspace. All your projects files should be in this directory. Path is validated on form save.

- **CSV separator** - configures separator for csv data export. You can export data to csv in some tables by right clicking on it and selecting "Copy selected rows".

- **Tooltips enabled** - when this option is active, tooltips will be shown on experiment properties form with helping information.

- **Advanced properties enabled** - when this option is active, all experiment properties are visible on form. By default, this properties are hidden and panel needs to be expanded to access them.

- **Manual ranking/pairs edition** - enables manual edition of pairs and ranking on experiment properties form. It should be used only when importing manually created experiment.properties file. Application expects strict format and can not accept values with are not compatible with UI application. UI dialogs for pairs and ranking should be sufficient to configure this fields.

- **Start maximized** - when this option is set to true, application starts with maximum supported resolution.

All changes in configuration will be visible after application restart.

4 Workspace management

In this chapter we will describe all features of workspace panel.

Workspace panel displays all files and directories (items) from workspace directory. Items are displayed in hierarchical structure representing hierarchy in file system. It also allows to do some basic file management. Thanks to this features, user doesn't have to exit program when he want to manage files. Path to workspace can be configured in user settings. See User settings section.
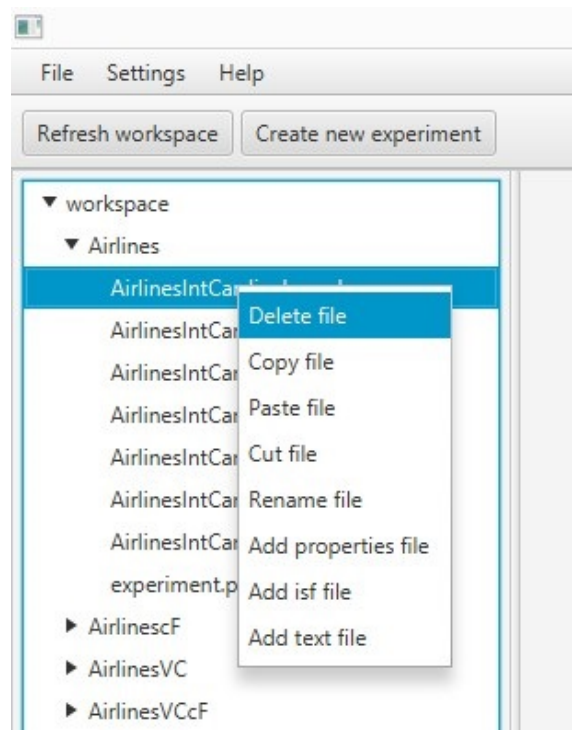

Figure 3: Application workspace from left panel

Most of workspace functionality is provided by context menu, with is accessible by right clicking on workspace item (file or directory). If directory contains files or other directories, it will have arrow on left side indicating, that item can be expanded/hided. Buttons on toolbar are also used for workspace management. They contain actions with are not related with specific item. File can be opened by double clicking on it.

Some actions support keyboard shortcuts. To perform action by shortcut, you have to select item and press specific combination of keyboard buttons described below:

- **Ctrl + C** - copies selected file to user clipboard. Copying directories is not supported yet.

- **Ctrl + V** - paste file to selected directory. If file was selected, file from clipboard will be copied to same directory as selected file.

- **Ctrl + X** - cuts file to user clipboard. If you paste file, old file will be removed.

- **Ctrl + R** - open dialog to rename file.

Toolbar currently supports two actions:

- **Refresh workspace** - it refresh first level of workspace tree. If some directories were expanded deeper, they will be hided.

- **Create new experiment** - it allows to create new experiment with example isf table and experiment properties file. It will open dialog with shows workspace. You should create new directory there or choose existing directory.

It is recommended to create separate directory for each experiment. Learning or test data table (isf files) can be stored in commonly shared folders. Each directory should contain only one experiment properties file. This file is needed for other files to extract information about experiment. Also it is recommended to store results files in same directory as experiment properties file.

Below is supported file type list:

- **.properties** - contains experiment properties with is used to configure experiment. See Experiment properties section.

- **.isf** - contains tabular data with can be editable or read only depending on data type. Here you can configure learning or test data for experiment. You can also view partial pairwise comparison table generated by experiment. See Isf table edition section and Partial PCT section.

- **.apx** - contains approximations of S and Sc relations in Pairwise Comparison Table, for now it is only displayed as text. See Partial PCT section.

- **.rules** - contains generated rules and rules statistics. See Rules section.

- **.graph** - contains preference graph visualization. See Graph visualization section.

- **.ranking** - contains result ranking for experiment. See Ranking window section.

- **.txt** - it can be used to store some notes or to show experiment report

All other file types are not supported. Not supported files are treated as non editable text files.

5 Editable isf table

In this section we will describe isf table edition window.

Learning and test data tables are stored in isf files. Columns represents objects attributes and rows represents examples of objects. RUDE application support full table edition from UI level. To open editable isf file, you need to double click on isf file in workspace. Isf file with is not editable, represents Partial Pairwise Comparison Table. In this chapter we describe only editable isf table.

Figure 4: Isf table edition window for Houses7 experiment

On top of the window, there is expandable panel with form actions. You can remove attribute (column) from there, if you select attribute in ComboBox and click on Remove selected attribute button. You can also clear table by removing all examples.

Below actions panel you can see editable isf table. In table header attribute name, type and cost (-)/gain (+) criterion is displayed. If you hover over column header additional information will be presented in tooltip. ID column is generated by application and it is used only to provide row number. It is not saved in isf file. Table columns can be also colored, when they are inactive (red) or are decision attributes (green).

5.1 Table edition

You can reorder attributes (columns) freely. You can also sort values in attributes. This changes will be saved to isf file.

You can edit examples by double clicking on cell in table. Changes will be saved after pressing Enter button or clicking on other cell. Edition will be canceled after pressing Escape button.

Each field type allows different set of characters:

- **String** - stores alphanumerical text, mostly used for description attributes.

- **Integer** - stores integer values.

- **Cardinal** - stores positive integer values.

- **Decimal** - stores floating point values, scientific notation is supported here.

- **Enum** - stores text values with are selectable from list.

Table supports context menu actions. Context menu can be opened by right clicking on table. It contains following actions:
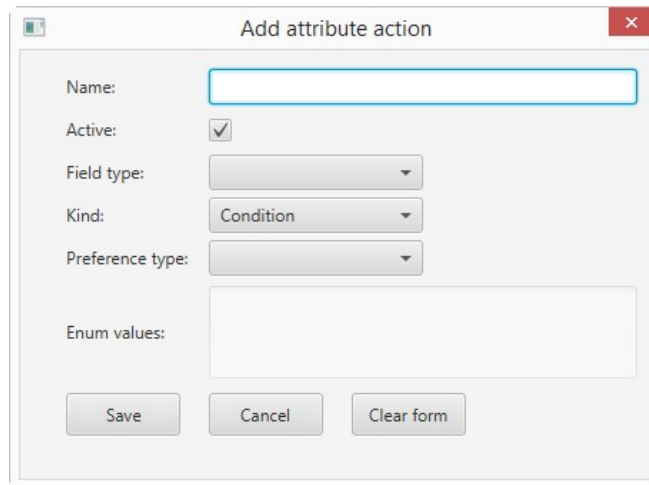
- **Add example** - adds new empty example (row).

- **Customize attributes** - opens modal dialog for editing attributes (columns).

- **Add attribute** - opens modal dialog for adding new attribute.

- **Remove selected examples** - remove selected examples (rows) from table. Multi-select is enabled.

- **Copy selected rows** - exports selected rows to CSV format. Attribute (columns) names will be used for header. Exported rows will be copied to user clipboard.

5.2 Adding new attribute

New attribute can be added by choosing "Add attribute" option from context menu. All fields are required in this form.

Field type was described earlier in Table edition section. By default, field is active, with means that it is not excluded from experiment. If attribute condition type will be selected, attribute will be used in experiment as normal field. If decision type was selected and has gain/cost preference type, it can be used to calculate reference ranking from examples in isf table. It will be used when running experiment.

Description field kind is not used in experiment and serves only for informational purposes. Preference type is used to determine if values in field have cost or gain criterion. Enum values field is only used for enum field type. If enum field type is chosen, list of values can be provided there by separating them by comma.
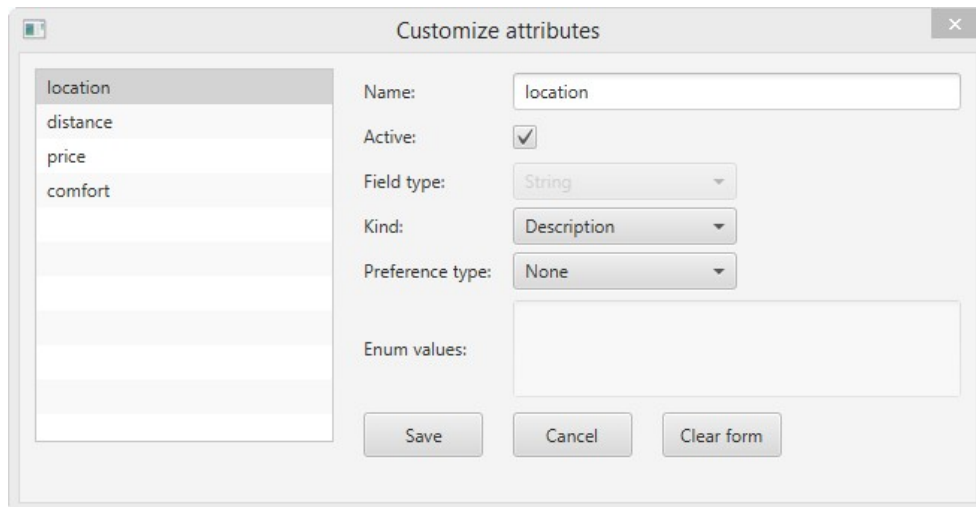
Figure 5: Add new attribute dialog

5.3 Customizing existing attribute

Existing attributes (columns) can be edited by choosing "Customize attributes" option from context menu.

Figure 6: Customize attributes dialog

Form in this dialog is similar to "Add attribute" dialog. Field type is disabled to avoid conversion problems between attributes, like conversion from string to integer. On the left side of dialog, you can choose attribute to edit. Changes in all attributes will be saved after clicking on save button.

6 Experiment properties

In this chapter we will describe experiment configuration. To edit experiment configuration, you have to double click on .properties file in workspace tree.
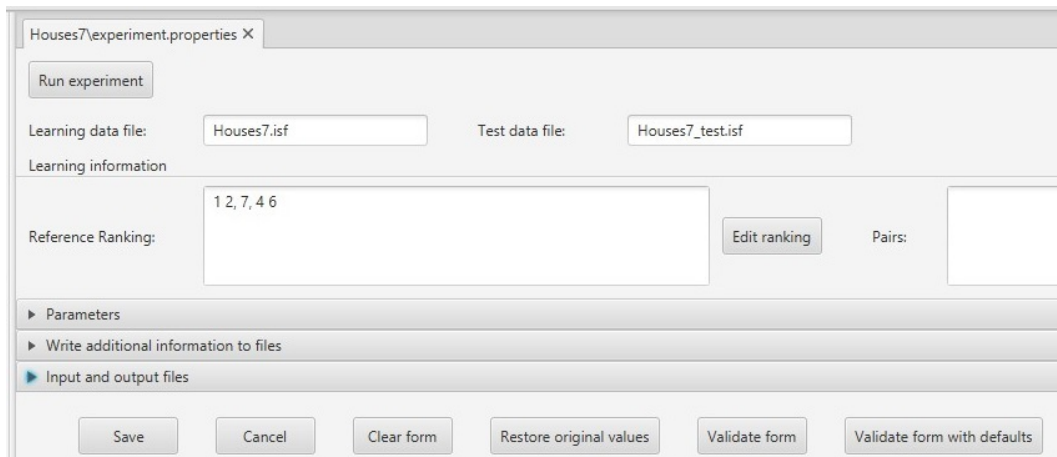
Four windows are used to configure experiment:

- **Properties form** - this is tab with opens after double click on properties file. It allows to edit experiment configuration.

- **Default properties form** - it is special case of properties form, where default properties for all experiments all configured.

- **Ranking dialog** - this modal dialog supports user friendly object ranking edition.

- **Pairs dialog** - this modal dialog supports user friendly comparison pairs edition

All windows are described in sections below.

6.1 Properties form

With default user settings, properties form will be opened with hidden sections of properties. This sections can be expanded by clicking on them. You can change this settings in User settings dialog. See User settings section.

Figure 7: Properties form with default settings

On top of the form, there is learning and test data table path. Learning data file is required field. If you hover on field with default user settings, tooltip with help will be displayed. You should also configure ranking or pairs before running experiment.

15

On bottom of the screen following actions can be performed:

- **Clear form** - clears all fields.

- **Restore original values** - restores values for all fields, as they were loaded from file again.

- **Validate form** - performs validation on form for fields. Certain options can not be selected together. If problem will be found, it will be displayed in modal dialog.

- **Validate form with defaults** - performs validation on form for all fields, including default ones. When running experiment, empty values from properties are replaced with default ones. It can discover issues with configuration, when values from form and default properties are not correctly set. If you find such issue, you should change settings in your experiment or in default properties.

Below, we describe meaning of all parameters.

- **Learning data file** - absolute or relative path to learning isf file. Directories must be separated by \ character.

- **Test data file** - absolute or relative path to test isf file. Directories must be separated by \ character. If not configured, it is assumed, that learning file is also test file.

- **Reference ranking** - initial ranking of objects in experiment.

- **Pairs** - pairs of object in outranking (S) or non-outranking (Sc) relation.

- **Type of family criteria** - can be consistent or any.

- **Type of rules** - can be certain of possible.

- **Considered set of rules** - if exhaustive, virtual exhaustive set of rules will be considered, if minimal, explicit minimal set of rules induced by VC-DomLEM algorithm will be considered.

- **Consistency measure** - measure used to calculate lower approximations of S and Sc relations.

- **Consistency measure threshold** - floating point number from [0,1] for epsilon, epsilon* and rough membership, or greater or equal than 0 for epsilon'.

- **Ranking procedure** - method used for calculating ranking from preference graph.

- **Dominance** - dominance relation used to calculate approximations of S and Sc relations in PCT.

- **Dominance for pairs of ordinal values** - indicator of the considered definition of dominance pairs of ordinal values in PCT.

- **Satisfaction degrees in preference graph** - weights in the preference graph. Can be fuzzy (from [0,1]) or crisp (0 or 1). Fuzzy satisfaction degree cannot be used with DRSA with exhaustive set of possible rules and rough membership.

- **Fuzzy satisfaction degree calculation method** - method for calculating fuzzy satisfaction degree in preference graph. Can be maximum of credibility over covering rules (max credibility) or maximum product of credibility and coverage factor over covering rules.

- **Negative examples treatment for VCDRSA** - sets strategy for covering negative examples by rules in VC-DRSA.

- **Rule conditions selection method in VCDomLEM** - strategy of rule conditions selection, either base (it can employ only elementary conditions built using evaluations of a single pair of objects) or mix (denotes that each rule can employ elementary conditions built using evaluations of different pairs of objects).

- **Optimize rule consistency in VCDomLEMWrt** - set of pairs of objects with respect to which value of rule consistency measure optimized in VC-DomLEM algorithm is calculated. Can be lower (or upper) approximation of the preference relation for which a certain (or possible, respectively) rule is generated or entire preference relation (for certain rules only) – either approximation or set.

- **Allow empty rules in VCDomLEM** - if true, rules with empty condition part can be generated if they have good consistency.

- **Use edge regions in VCDomLEM** - if true, only pairs of objects from EDGE region will be used to create rules condition.

- **Write domination information** - if true, sections [P-dominating sets] and [P-dominated sets] will be saved to .apx file.

- **Write rule statistics** - if true, rule statistics will be saved in .rule file.

- **Write learning positive examples** - if true, learning positive examples will be written to .rules file.

- **Precision** - denotes precision of floating point numbers when saving files. -1 disables rounding.

- **PCT file** - absolute or relative path on with isf file for Partial Pairwise Comparison Table will be saved. Directories must be separated by \ character. If not given, name of the file will be set using learning data file name.

- **PCT Apx file** - absolute or relative path to .apx file, where approximations from PCT table will be saved. Directories must be separated by \ character. If not given, name of the file will be set using learning data file name.

- **PCT rules file** - absolute or relative path on with rules will be saved. Directories must be separated by \ character. If not given, name of the file will be set using learning data file name.

- **Preference graph file** - absolute or relative path on with graph will be saved. Directories must be separated by \ character. If not given, name of the file will be set using test data file name.

- **Ranking file** - absolute or relative path on with ranking file will be saved. Directories must be separated by \ character. If not given, name of the file will be set using test data file name.

6.2   Default properties

When performing experiment, all properties must be set. So if user leave empty fields, they must be replaced with some default values. RUDE application have configurable default properties, with should be located in main workspace directory with file name: default.properties. You can edit this file freely, but it is recommended to fill all fields from parameters and write additional information sections.

Figure 8: Properties with default values set from default.properties file

Default properties are loaded on each properties form. They are displayed to user in form: default (default value) for ComboBox fields, and as gray placeholder in other fields if they are empty. If you configure field in own experiment, value from your configuration will replace default one when performing experiment. All properties with are used in experiment are displayed in logs before experiment run.

6.3   Ranking configuration

Ranking field represents initial ranking of objects in experiment. It can be edited by clicking on "Edit ranking" button.

Ranking can be configured manually in big text field, but it is not recommended and disabled by default. See User settings section. Objects on left list are sorted by object ID.

Figure 9: Ranking edition modal dialog

By default, objects ID (number of object) is displayed. If you created description attribute, it can be used as label for objects. You can change this in "Displayed label" field.

You can drag and drop objects from left list to right tree. If you place object on empty cell, new rank position will be created automatically. If you drag object to rank or element in rank, dragged object will be added to existing rank.

On right tree, you can perform two actions, with can be also invoked on selected item by context menu or keyboard shortcut:

- **Remove selected (Delete)** - removes selected object from ranking. Removed object will return to list on the left. If rank was selected, all objects from rank will be removed to.

- **Add Rank below (A)** - adds new rank position below selected position.

19

6.4 Pairs configuration

Pairs field represents comparison pairs of object in Pairwise Comparison Table. They can be edited by clicking on "Edit pairs" button.

Pairs can be configured manually in big text field, but it is not recommended and disabled by default. See User settings section.
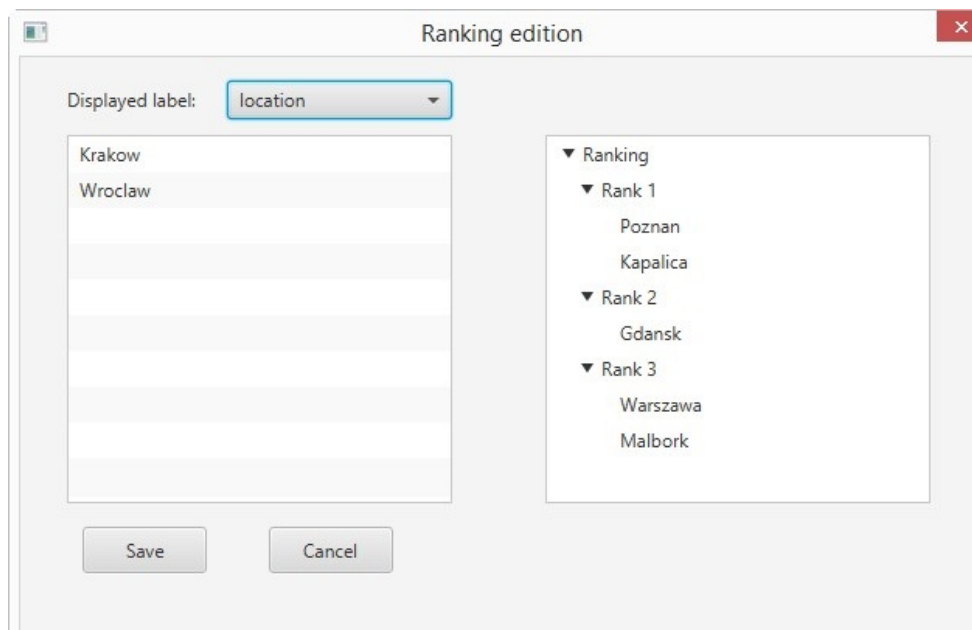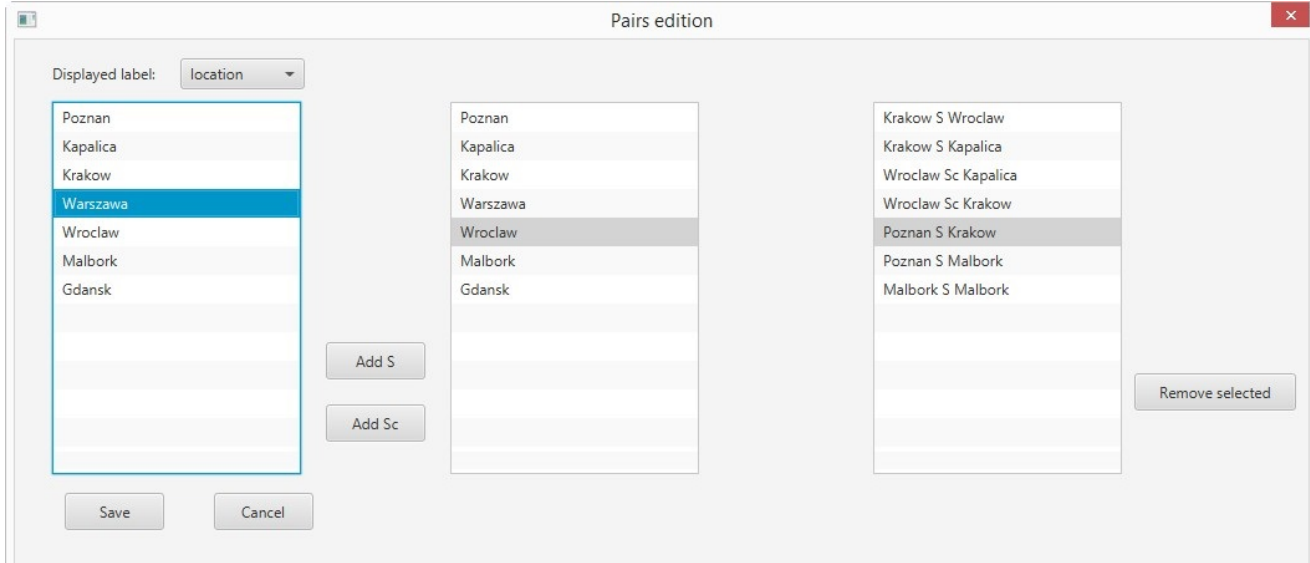

Figure 10: Pairs edition modal dialog

By default, objects ID (number of object) is displayed. If you created description attribute, it can be used as label for objects. You can change this in "Displayed label" field.

First and second list contains all objects from experiment. Third list contains compared pairs of this objects with relation S (outranking) or Sc (non-outranking).

Three actions are available for this screen. Each can be invoked by pressing button, choosing option from context menu or by keyboard shortcut. First two actions are for first two lists, third is for pairs lists:

- **Add S (S)** - adds selected pair of objects from first and second list to pairs lists. Objects are added in outranking relation.

- **Add Sc (C))** - adds selected pair of objects from first and second list to pairs lists. Objects are added in non-outranking relation.

- **Remove selected (Delete)** - removes selected pair from third list.

7   Running experiment

In this chapter we will provide instructions for running experiment.

Before running experiment, learning data file (test data file is optional) should be created See Isf edition section. It is recommended to fill all fields in active attributes. Condition attributes with empty values cannot be used in experiment, so experiment doesn't start with such values.

Also experiment properties should be configured. See Properties section. All fields in properties should be set, either explicitly or by default values.

Experiment can be run from properties form. To do this, press "Run Experiment" button on top of properties form. After this isf table and experiment properties will be validated. If RUDE finds any misconfiguration error, it will display it on modal window and experiment doesn't start. If you provided multiple information sources (ranking, pairs or decision attribute), you will be asked to choose one of them.
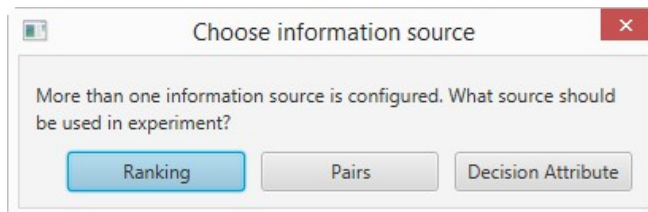
Figure 11: Choose information source dialog

Before experiment start, all properties will be logged (chosen properties and default ones for empty fields). When experiment finishes, all result files will be saved. In case of any errors, logs will be written to logs tab. Workspace tree will be automatically refreshed. If file names were not configured, they names will be calculated basing on learning and test data file name. The suffix partialPCT in some files is used to indicate that we deal with a "partial" PCT.

8 Partial Pairwise Comparison Table

In this section we will describe generated .isf and .apx file for Partial Pairwise Comparison Table. You can open this files by double clicking on them in workspace tree.

8.1 Isf file

This isf file is read only. Each row represents comparison of pair of examples. Each column represents attribute (criteria) on with comparison was performed.

ID	Pair_of_examples Pair	diff_distance Integer (-)	diff_price Integer (-)	comfort Pair (+)	Relation Decimal (+)
1	(1,1)	0	0	(Good,Good)	S
2	(1,2)	-32	30	(Good,Good)	S
3	(1,7)	-2	-10	(Good,Medium)	S
4	(1,4)	-7	-30	(Good,Basic)	S
5	(1,6)	-47	10	(Good,Medium)	S
6	(2,1)	32	-30	(Good,Good)	S
7	(2,2)	0	0	(Good,Good)	S
8	(2,7)	30	-40	(Good,Medium)	S
9	(2,4)	25	-60	(Good,Basic)	S
10	(2,6)	-15	-20	(Good,Medium)	S
11	(7,1)	2	10	(Medium,Good)	Sc
12	(7,2)	-30	40	(Medium,Good)	Sc

Figure 12: Read only Partial Pairwise Comparison Table from Houses7

In table header, field types and cost/gain criterion indicator is displayed. If you hover over column header, more information can be displayed. Columns can be ordered and sorted, but it will not be saved to file, because it is read only. You can also export selected rows to CSV format. You can do this by selecting rows and choosing "Copy selected rows" option from context menu.

8.2 Apx file

In this file additional information is saved in text format. User interface for this file is not created yet. It contains domination cons (P-dominating sets, P-dominated sets), approximations, decision classes and some metrics, like accuracy or quality of sorting.

9 Rules

In this section rules and rule statistic window is described. Rules and statistics are stored in .rules file. You can open this file by double clicking on it in workspace tree.

In some cases, rules can't be generated. In such case .no-rules file will be saved instead of rules file. Rules are generated from Pairwise Comparison Table.

9.1 Rules window

ID	Decision	<=	Condition 1		Condition 2
1	x S y	<=	(diff_price <= -30)		
2	x S y	<=	(diff_distance <= -32)		
3	x S y	<=	(diff_distance <= 0)	&	{PAIR(comfort) D (Good,Good)}
4	x Sc y	<=	{(Medium,Good) D PAIR(comfort)}		
5	x Sc y	<=	(diff_distance >= 45)		
6	x Sc y	<=	(diff_distance >= 5)	&	{(Basic,Medium) D PAIR(comfort)}

Figure 13: Rules tab from Houses7

In this window all saved rules are displayed. Each rule consists of a conjunction of elementary conditions and a decision. Decision is always displayed on the left side. Each rule can have many conditions. Also, number of conditions in rules can vary for each rule. Values in columns can be sorted.

You can also export selected rules (rows) to CSV format. You can do this by selecting rows and choosing "Copy selected rows" option from context menu. Columns names will be used for CSV header.

If you click on the rule, rule statistic tab will be displayed below. It displays additional information about rule and rule statistics.

9.2 Rule statistics window

In this tab rules statistics are displayed with some additional information about rule. Rules can be of certain or possible type. Indexes provided in statistics represents position of pair in PCT file.

Some most important statistics are:

- **Support** - number of pairs that supports this rule. Pair support rule if they belong to suggested relation and satisfies all elementary conditions.

- **Strength** - support / number of pairs.

- **Confidence** - also called certainty factor, defined as support / number of pairs of objects that satisfy all elementary conditions of rule.

- **Coverage factor** - support / number of pairs that belong to relation suggested by rule.

Logs	Statistics of Houses7_partialPCT.rules ✕		
Rule type: CERTAIN	Usage type: AT LEAST	Relation:	S

Support:	4
Supporting examples:	2, 5, 15, 20
Strength:	0.16
Confidence:	1.0
Coverage factor:	0.235
Coverage:	4
Covered examples:	2, 5, 15, 20
Negative coverage:	0
Negative examples:	
Inconsistency measure:	0.0
f-confirmation measure:	1.0
A-confirmation measure:	0.09
Z-confirmation measure:	1.0
l-confirmation measure:	Infinity

Figure 14: Rules statistic tab from Houses7

10   Graph visualization

In this section graph visualization window is described. Graph data is stored in .graph file. You can open this file by double clicking on it in workspace tree.

Preference graph is created from induced rules. Vertices of graph represents objects from isf table. Arcs represents outranking (S) or non-outranking (Sc) relations between objects. If arc is present between x and y vertices, it means that pair of x and y object is covered by a decision rule with suggest assignment to relation S or Sc relation. Arcs can be also weighted, where weight represents satisfaction degrees in preference relations. Rule confidence is used for arc weight. If arc is covered by multiple rules, arc weight is aggregated, for example by taking maximum confidence.
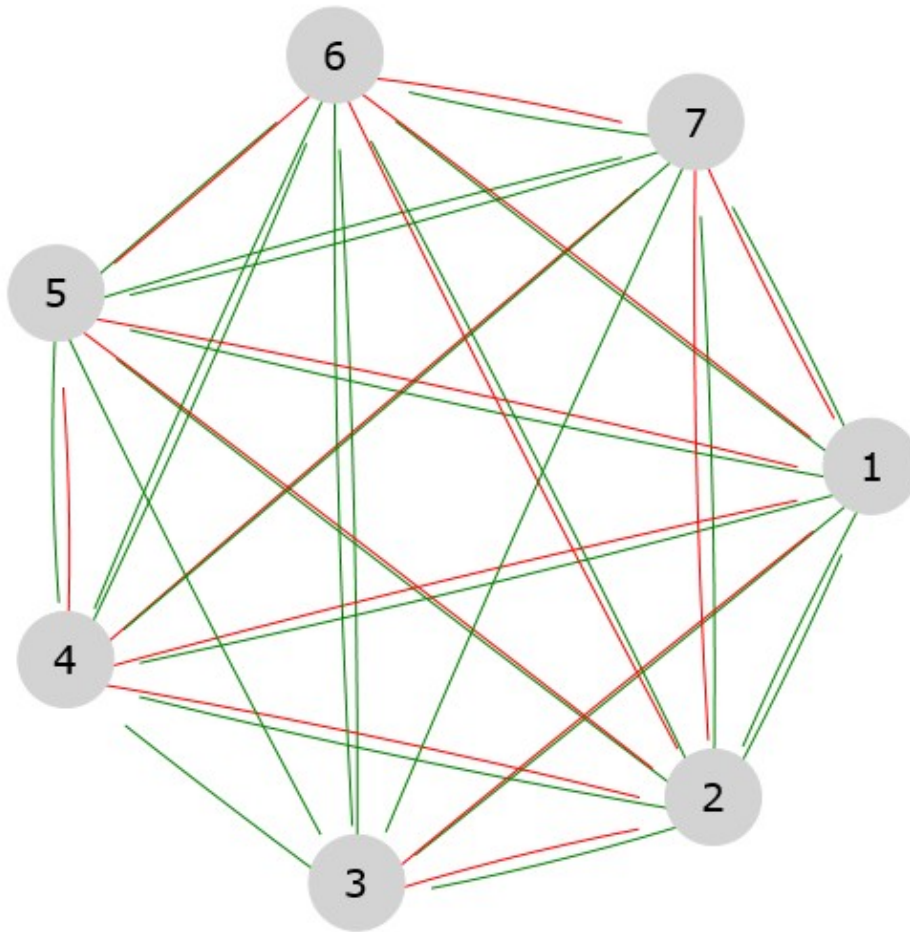

Figure 15: Graph visualization for Houses7

Between each pair of vertices, up to four relations can be present:

- x S y

- x Sc y

- y S x

- y Sc y

Relation type is marked with line color. Green is for outranking relation (S) and red for non-outranking relation (Sc). So relation x S y will be drawn as green line between this two vertices.

All arcs are directed. Direction of arc is marked with line end. Lines ends before target vertex. So x S y relation will be drawn as line from x vertex to y vertex with line ending slightly before y.

To improve arcs visibility, only up to two arcs are drawn between vertices. If verticles contain relations: x S y, x Sc y - they will be replaced with one line with gray color. Same reduction is made for y S x, y Sc x.

You can navigate on graph by using scrollbars, zoom in and zoom out by mouse wheel. You can also drag vertices to change their position. If you click on vertex, it will be selected and summary for arcs will be displayed in tab below. All arcs weights can be displayed there (if they exists) in square brackets.
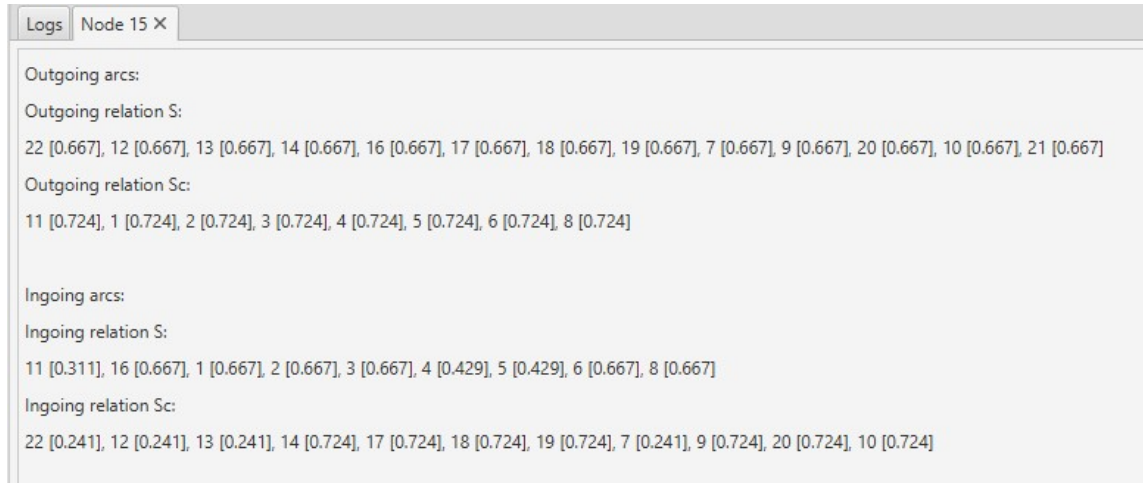

Figure 16: Vertex arcs for NotebooksVCcF with weights

11   Ranking window

In this section ranking window is described. Ranking data is stored in .ranking file. You can open this file by double clicking on it in workspace tree.

Ranking is created by chosen ranking method, with exploits preference graph. Created ranking is a solution to ranking problem.

Position	Evaluation	location String	distance Integer (-)	price Integer (-)	comfort Enum (+)	
1	10.0	Poznan	3	60	Good	
1	10.0	Kapalica	35	30	Good	
2	1.0	Wroclaw	5	60	Medium	
2	1.0	Gdansk	5	70	Medium	
3	-5.0	Krakow	7	85	Medium	
4	-8.0	Malbork	50	50	Medium	
5	-9.0	Warszawa	10	90	Basic	

Figure 17: Ranking for Houses7

Each row in table represents position of object in ranking. Also objects attributes are displayed as additional columns. Columns for attributes contains field type and cost/gain criterion indicator. If you hover over column header, tooltip will be displayed with additional information.

Columns can be reordered and are sortable. This is not saved to file, because this files are read only. You can also export selected rows to CSV format. You can do this by selecting rows and choosing "Copy selected rows" option from context menu. Columns names will be used for CSV header.

12 Data configuration directory

Data directory in RUDE application contains all configuration files. It shouldn't be edited manually. The only exception is adding own language translation to application.

12.1 Custom language support

RUDE can support user created translations for other languages. Currently only English is supported officially.

All languages translations are stored in map (key-value), where key represents language code and value represents displayed text (translation). This maps are stored in JSON files. So files after changes should be valid JSON. You can use online JSON validator for validating.

To add own language, you have to perform this steps:

1. Close RUDE application if it is running.

2. Make backup copy of data directory.

3. Open languages.json file.

4. Add own language code and language name to file. Each entry must be separated by comma. Text value for language is displayed in user settings as possible option in language configuration field.

5. Open labels.json file. Copy and paste content in first brackets: "ENG" : {...}. Remember, that each language entry must be separated by comma.

6. Replace "ENG" with own language code from languages.json file.

7. Translate text in values for keys.

8. Save all files

If you provide invalid JSON format, application won't read this files and display error. If any key from map will be missing, warning in application log will be displayed if application try to use it.