Appendix E

Viz User Manual

This chapter contains Viz *user manual. The user manual is prepared using the latest version of* Viz. *However, many functions are applicable for older versions too.*

E.1 Using Viz for your OWN SLS on your OWN COP

Using Viz EW and Viz SIMRA to fine-tune or to analyze *your OWN* SLS algorithm on *your OWN* COP is relatively easy. You will need to modify your SLS a bit according to our simple rules. In summary, what you need to modify are as follows (see label 'A' to 'F' on the left hand side):

```
\-------------------------------------------------------------------------------/
\  1. SLS(command line parameters) { // "Instance" "Configuration" RunID ["no"]    /
\A  . set the SLS to read in COP instance data based on file "Instance";           /
\B  . configure SLS based on file "Configuration";                                 /
\  2. start from any solution of the COP instance;                                 /
\C  . create RunLog file with name: CurrentDir\Instance_Configuration_RunID.RunLog; /
\D  . record header information for RunLog (Row 1-9);                             /
\  3. while (the search is not finished) {                                         /
\  4.   search around the current solution with some heuristic/stochastic rule;    /
\  5.   if (the newly found solution is better than the best solution so far)      /
\  6.     update the best solution status;                                        /
\E  .   record solution and algorithm specific information (Row 10-17) into RunLog; /
\  7. }                                                                            /
\  8. return the best found solution during the search process;                    /
\F  . create smaller log: CurrentDir\Instance_Configuration_RunID.QuickRunLog;    /
\  9. }                                                                            /
\-------------------------------------------------------------------------------/
```

Figure E.1: Simple modifications to your SLS algorithm

> Note that we need to use "double quotes" to group long file name with spaces into one command line parameter! Otherwise your SLS program may tokenize the parameters incorrectly.

First, adjust your SLS algorithm to read in three (or four, the fourth one is optional) command line parameters and then process them according to instructions **(line A and B)**. The format of the command line parameters is as follows:

1. **"Instance"**, a file name that specify the COP instance that your SLS will attack.

   You just need one simple modification.

Please add one line of Best Known (BK) objective value of that particular instance *as the first line* of the instance file. Simply write 0 if BK is still unknown, then Viz will use Best Found BF value instead. However, please update the BK value as soon as you find better BK value for more *accurate* analysis.

Viz EW and SIMRA uses this information to compute important performance statistics. Thus more accurate BK is preferred. Of course, your SLS program must now skip this first additional line when you parse the modified instance file.

2. **"Configuration"**, a file name that specify how to externally configure your SLS algorithm.

   SLS algorithm usually have three group of things that can be configured:

   *. the parameter values

   *. decision variables to switch between the usage of certain SLS components

   *. decision variables to switch between the execution of certain search strategies

   To allow Viz EW to execute your SLS implementation with various configurations, you must allow both Viz EW and your SLS implementation to read external configuration file *.cfg. This configuration file is a simple text file with three columns separated by **tab** (`\t`) character (value, field-name, comment). Only your SLS algorithm understands the configuration file designed by yourself. Your SLS algorithm will read the configuration file line by line and configure itself. Viz EW will just pass the appropriate *.cfg file into your program according to the experiment design.

   However, Viz EW will actually read in *.fac file, and not *.cfg. This *.fac file describes the full factorial design of the configuration set that contains all possible combination of configurations. The *.fac file is basically the same *.cfg variant, but now you can have *several* values for a field-name. For example: PARAMETER-A can be either (0;1;2), and COMPONENT-B can be either (tabu-move;tabu-solution). This produces $3 * 2 = 6$ different configurations and each will likely perform differently. If you only want one configuration, set the domain of all configurable parts to be one single value only.

   Viz EW will use its built-in tuning algorithm in running these configurations.

   Notes: All the three columns must be present! Do not omit the third column (comment). Viz EW assumes the *.fac file contains 3 columns during parsing. Then, for quick start, you can simply create empty *.fac file (that is, your SLS will use its default configuration).

3. **RunID**, a number to indicate replications in Viz EW. It starts with "1".

4. [**"no"**] RunLog, an **optional** command line switch that Viz EW will issue when performing fine-tuning. This feature is used for quick 'true performance' test of your SLS algorithm (i.e. skipping part C, D, and E). It is known that recording RunLog actually slows down the actual SLS speed. With this, your SLS program can decide whether to log the information or not, without recompilation. Viz EW will only use the final information recorded in QuickRunLog (step F).

Next, to use FLST visualization, you need to modify your SLS implementation to log some simple information from the SLS run into a log file called RunLog. This RunLog is a simple log file that records some header information (line D in Figure E.1, row 1-9 in Table E.1) and important search information (line E in Figure E.1, row 10-17 in Table E.1) from start (iteration=0) to end (iteration=lastItr). The RunLog file must be precisely named as `Instance_Configuration_RunID.RunLog` and stored in the current working directory (the same directory where all your COP instances, SLS executables, configuration files reside) so that Viz EW can interpret it correctly. If existing RunLog file exists (you have run your SLS program before), simple overwrite it with the information from the most recent SLS run. The RunLog 2006b file format is described in Table E.1.

Viz EW uses the information from these RunLog files (of the same instance/fitness landscape) to build the FLST visualization. Viz SIMRA will then be used to display the visualization. With Viz, user does not need to compute complex computations (e.g. distance information, objective value average, fitness distance correlation ratio, etc) as Viz will do all the computation using the information provided in the RunLog files.

Please store all your files inside the current working directory. If your SLS executable requires external DLL files, copy those files into the working directory too. The current version of Viz EW will only process the files inside this working directory.

Note that when you write row 10-17 iteratively, please write in flushing mode rather than buffering mode, so that when the program is killed by Viz EW due to time/iteration limit exceeded, the RunLog for the past iterations before the program is killed is already written.

Row	Example	Field Name	Type	Remarks
1	2006b	FormatVersion	String	Current Viz EW will only use this version.
2	TSP	ProblemName	String	COP abbreviation: 'TSP', 'QAP', 'LABS'.
3	1	IsMinimizing	Boolean	0: False, 1: True.
4	eil51	InstanceName	String	Full path is not required.
5	51	InstanceSize	Integer	The instance size.
6	ILS	SolverName	String	SLS abbreviation: 'TS', 'ILS', 'SA'.
7	Tuned	RunDescription	String	You can write one line to describe your solver.
8	1000	NumberOfEntries	Integer	The row 10-17 below are repeated this many times.
9		Dummy1	Empty	Reserved for future version of Viz.
10	0, 1, ..., 50,	Solution	IntList	Comma separated list, last item inclusive!
11	1	IsFeasible	Boolean	0: Infeasible, 1: Feasible.
12	426.000000	ObjectiveValue	Double	
13	0.435000	TimeStamp	Double	Use timer inside SLS, default: 0.0.
14	"Tag"	Tag	String	Leave blank if not needed.
15	4	ProblemSpecific1	Double	For TS: TabuTenure, ILS: PerturbStr, SA: Temp
16	1	ProblemSpecific2	Double	For TS: -, ILS: Accept-1/Reject-0, SA: TossValue
17		ProblemSpecific3	Empty	Reserved for future version of Viz.

Table E.1: RunLog File Format (version 2006b)

Some notes on how to record the information (row 10-17) for several popular SLS algorithm:

1. For Tabu Search (TS) and Simulated Annealing (SA), record row 10-17 at the end of each iteration so that we can see the step by step progress taken by TS/SA. If we only record the local optima, we cannot see TS/SA search trajectory in details.

2. For Iterated Local Search (ILS), record row 10-17 **before** and **after** the acceptance criteria step, so that we can see the effect of the local search phase and whether that (new) local optima is accepted or rejected. Otherwise you will only see limited movement when many local optima after local search phase are rejected (you will be returned back to the old local optima ... that is, no movement seen at all).

3. For population based search (e.g. GA, ACO, PSO), you are working with population over generations. In order to make Viz display the most meaningful information for your population based search, please record the RunLog using this technique:

   (a) Write generation best solution (or best ant in ACO) according to row 10-17 format.

   (b) Repeat the process to the other member of the population in the current generation. Example: if you have 100 generations, and each generation has 20 members, then you will write 2.000 entries of row 10-17 into the RunLog and record the total number of entries in RunLog (row 8) to be 100*20 = 2.000 instead of 100!

   (c) In Viz SIMRA, set the playback trail to be 20 (show the 20 members of the population) and then advance the playback per 20 iterations (generation by generation). This will give you the visualization of the movement of best solution from generation to generation as well as the information of diversity of the other 19 members.

4. For other SLS, try to record row 10-17 in a place within your SLS.exe code so that the amount of information that you record is maximized.

RunLog files can be bloated and slow down the SLS runs. For the purpose of fine-tuning and displaying results, we use QuickRunLog format is used. A smaller log file that only contains basic information shown in Table E.2. This allows Viz EW to obtain the result of a particular run without having to parse through the usually big RunLog file.

E.2 Using Viz EW

There are three steps in Viz Experiment Wizard (EW):

Row	Example	Field Name	Remarks
1	1	IsMinimizing (0: No—1: Yes)	
2	1.0	The run time of this SLS run	
3	1000.0	Number of iterations of this SLS run	
4	426	Best Known (BK) Objective Value	
5	427	Best Found (BF) Objective Value	

Table E.2: QuickRunLog File Format (version 2008)

Step 1: Experiment Setup/Notes

User's tasks during this step (see Figure E.2, left) are as follows:

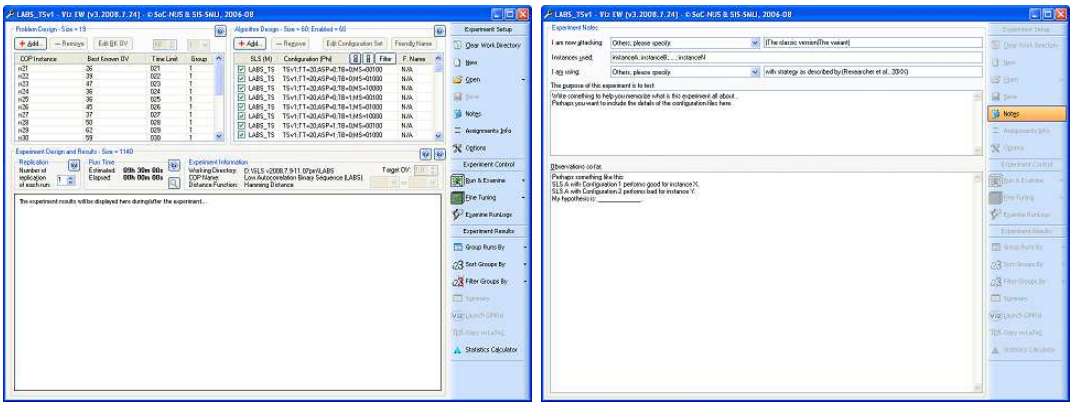

Figure E.2: Viz EW - Experiment Setup (left) - Experiment Notes (right)

1. User can choose to **'Clear Work Directory'** (start afresh), set **'New'** experiment setup (set all fields empty), **'Open'** previously saved experiment setup (*.experiment), or **'Save'** the current experiment setup (*.experiment).

2. Write down some **'Notes'** (see Figure E.2, right) to explain the rationale for current experiment. This is quite important user may want to run the experiment overnight, so that when he returns next morning, he will want to review again his experiment objective that he has setup in the previous night.

3. The default settings for each COP (instance file extension, time limit for each instance TL_i, natural distance function, etc) can be tweaked via COP **'Assignments Info'** (see Figure E.3, left).

   (a) TSP; *.tspx; TSPLIB EUC2D [3], use Bond/Permutation/Edge distance
   (b) QAP; *.datx; QAPLIB [1], use Hamming/Exact distance
   (c) LABS; *.labsx; CSPLIB [2], use Hamming distance
   (d) CFSP, *.cfspx; use Deviation distance
   (e) Other COP instances are not supported at the moment.

4. User can adjust Viz EW **'Options'** (see Figure E.3, right). Usually the defaults are okay:

   (a) Set minimum $|APset|$.
   (b) Set $|APset|$ as percentage of instance size.
   (c) Set maximum $|APset|$.
   (d) Set the default random seed for the AP layout algorithm.
   (e) Set the maximum number of iterations for the AP layout algorithm.
   A slightly better AP layout is expected for longer runs though not always guaranteed.
   (f) Set distance multiplier to make points layout in FLST visualization more reasonably spaced out.
   (g) Decide whether to delete RunLog files after conversion.

(h) Decide whether to load the last experiment setup file (if any) during the loading phase of Viz EW.

(i) Decide to show/hide tool tips (turn this off if you are already familiar with Viz EW).

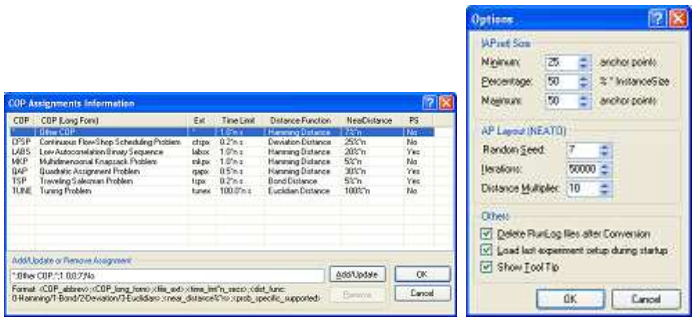

Figure E.3: Viz EW - COP Assignments Information (left), Options (right)

5. Setup **'Problem Design'**: add (or remove) set of COP training instances, **'Edit BK OV'** (if needed), adjust **'Time limit'**[1] or **'Instance Group'**.

6. Setup **'Algorithm Design'**: add (or remove) SLS.exe (M) + Configuration (Φ) pairs, **'Edit Configuration Set'**, or give **'Friendly Name'** to an $M + \Phi$ pair.

7. Specify **'Number of replications of each run'** (to test robustness).

8. Specify which $M + \Phi$ runs should be performed by checking or unchecking the corresponding $M + \Phi$. You can also **'Select All'** or **'Un-select All'** $M + \Phi$ or to **'Filter'** some $M + \Phi$ based on some simple constraint expression:

   CONFIGURABLE_PART$(= \mid < \mid >)$VALUE[*]

9. There are several options to execute the runs:

   (a) Click **'Run & Examine'** to run the SLS. Then, Viz EW will collect the RunLog files produced by the runs and examine them using FLST visualization techniques. Please note that preparing FLST visualization takes time. Thus ensure that the number of items in $M + \Phi$ list and the number of COP instances are not too many as full factorial design is adopted in this mode.

   (b) If you just want to see past results (if any) without re-running the experiment), click the drop down button and select **'View Last Examination Results'**. Viz EW will then parse the existing VDFs (if any). Note that the results displayed will be old results which may be inaccurate if the user has somehow changed the experiment setup or improve (/degrade) the SLS algorithm!

   (c) You can **'Fine-Tuning'** your SLS algorithm. Viz EW has built in tuning algorithm (LinearSampling, RandomSampling, GA, RCTS?, etc) that you can select in the drop down button. You can also adjust terminating criteria: Z and CTL via the drop down button. In this mode, you are encouraged to enlarge the $M + \Phi$ list with various logical entries. Viz EW will run them with 4th command line parameter "no" so that the SLS runs only record QuickRunLog file at the end of its run.

   (d) The final mode is **'Examine RunLogs'**. Basically, it is used to prepare FLST visualization from the RunLog files (presumably from your SLS that were running on non Windows platform). We encourage you to run your SLS via Viz EW interface.

Step 2: Human Guided Tuning

During the execution of the SLS runs, Viz EW will draw a visualization of fine-tuning process, update the elapsed time and predicted time left. The current $M + \Phi$ being processed is colored with LightPink and the current instances is mentioned. See Figure E.4.

[1]Each run on this COP instance will be killed if it takes more than the run time limit set for this instance. By doing this, we roughly know what is the maximum time to complete the entire experiment.

In ‘Run & Examine’ mode, the number of SLS runs are small and this part can be skipped. However, during the ‘Fine Tuning’ mode, user can decide to do the following actions based on the potential insights shown in the visualization:

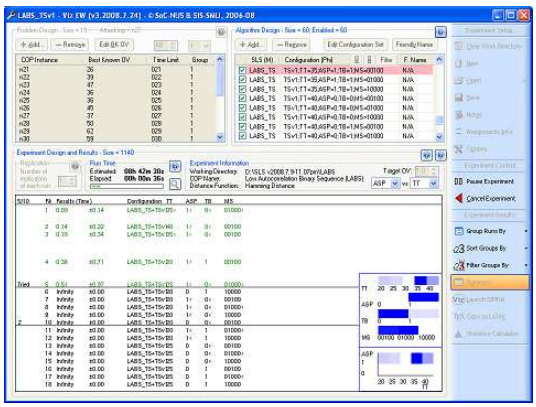

Figure E.4: Viz EW - Human Guided Tuning

1. User can choose to **‘Pause Experiment’**, select, un-select, filter, or remove some or all untried $M + \Phi$ in the list. Note that you can only adjust untried $M + \Phi$. This can help guiding the fine tuning process. Then, click **‘Resume Experiment’** to instruct the tuning algorithm to focus on the adjusted configuration space.

2. User can **‘Cancel Experiment’** and return to the setup mode anytime. Note that all experiment results obtained so far will be lost.

Step 3: Analyze Results

After the experiment is over, the results will be presented. The presentation of results depends on the run mode. In ‘Run & Examine’ mode, a list of runs is presented (Figure E.5, left). In ‘Fine Tuning’ mode, a visualization is presented (Figure E.5, right). User has the following options during this step:

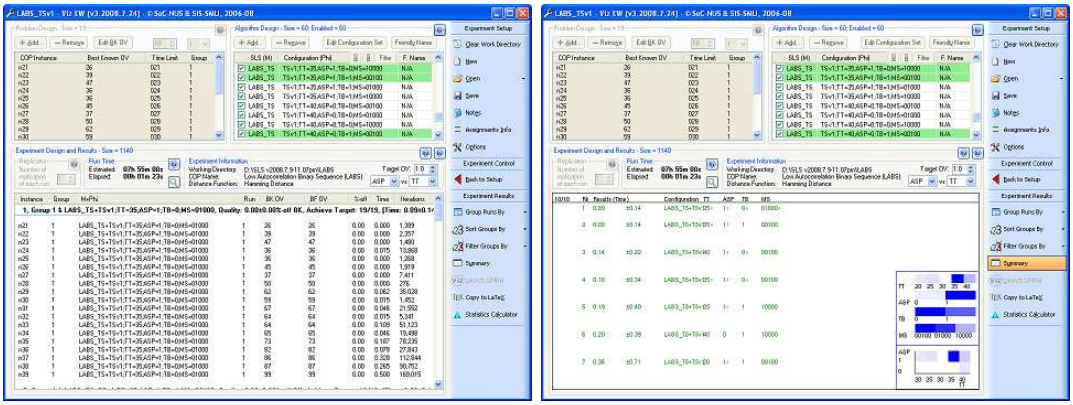

Figure E.5: Viz EW - Analyze Results

1. User can view the results from different angles. User can use the **‘Group Runs By’**, **‘Sort Groups By’**, and **‘Filter Groups By’** options to quickly classify, sort, and to filter runs/groups quickly.

2. The **‘Summary’** button toggles between the detailed table mode versus an information visualization mode.

3. By double clicking one (at most two) run(s) (or click **‘Launch SIMRA’**), the user will open Viz SIMRA to examine the selected runs in detail. Please refer to Chapter E.3 for details. If visual comparison is used, the two selected runs must be ‘compatible’: both must attack

the same COP instance even though the SLS or configuration can be different (this is what we want to compare).

4. There is a button **'Copy to LaTeX'** to save the current experiment results into LaTeX format. You can paste the information into your scientific paper directly.

5. We also have **'Statistics Calculator'** to test the significance of the results. Basically, the user can compare objective value/run time/number of iterations of the current groups (e.g. results from SLS A versus SLS B), then use this statistics calculator to verify whether the results from SLS A is significantly better (or worse) than the results from SLS B.

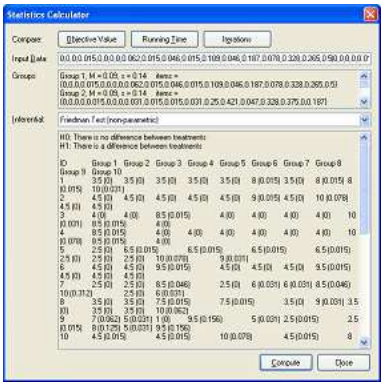

Figure E.6: Viz EW - Statistics Calculator

E.3 Using Viz SIMRA

This section will be rewritten soon!

Bibliography

[1] Rainer E. Burkard, Stefan E. Karisch, and Franz Rendl. A quadratic assignment problem library. *European Journal of Operational Research*, 55:115–119, 1991.

[2] CSPLIB. A Problem Library for Constraints.
http://www.csplib.org.

[3] Gerhard Reinelt. TSPLIB - A Traveling Salesman Problem Library. *ORSA Journal on Computing*, 3:376–384, 1991.