User Manual

The framework was designed to solve data instances for all three tracks of the ITC 2007 competition. The best way to run the solver is either running the executable JAR file via the command line, or through a Java IDE. The classes added or altered to the solver for this project are: ItcTest, ParamTune, exam.properties, ctt.properties, tim.properties, ItcNeighbourSelectionSeq, GroupAccept, HeuristicSequence, ItcGreatGelugeSeq, ItcHillClimbingSeq, ItcSimulatedAnnealingSeq, Random Restart and SelectNeighbour. The classes, GroupAccept, ParamTune and Random Restart are not used in the final hyper-heuristic and were added for either configuration purposes or to be a part of the hyper-heuristic but did not pass performance testing. The submitted solver is capable of running Muller's original 2007 code and the modified Sequence-Based hyper-heuristic. To change between the two modes and to change any other setting you want, you must navigate to the specific track's config file. For example, if I want to run data instances on the Examination track I will open the 'exam.properties' file under the 'src' folder. The Examination track properties file is shown in figures 1 and 2.

```
#Implementation
Model.Class=net.sf.cpsolver.itc.exam.model.ExModel
Model.Extension=sln

#Termination condition
Termination.Class=net.sf.cpsolver.ifs.termination.GeneralTerminationCondition
Termination.StopWhenComplete=false
Termination.TimeOut=247

#Neighbour selection criterion
# Remove # below for normal
#Neighbour.Class=net.sf.cpsolver.itc.heuristics.ItcNeighbourSelection
Neighbour.Class=net.sf.cpsolver.itc.heuristics.ItcNeighbourSelectionSeq
# modified one
#Neighbour.Class=net.sf.cpsolver.itc.heuristics.search.GroupAccept

#Construction
TabuSearch.MinSize=0
TabuSearch.MaxSize=0
TabuSearch.MaxIdle=100000
Value.ConflictWeight=5000
Itc.Construction.ValueWeight=1

#HC
HillClimber.Neighbours=net.sf.cpsolver.itc.heuristics.neighbour.selection.ItcSwapMove;net.sf.cpsolver.itc.exam.neighbour
HillClimber.Random=true
HillClimber.MaxIdle=25000

#GD
GreatDeluge.Neighbours=net.sf.cpsolver.itc.heuristics.neighbour.selection.ItcSwapMove;net.sf.cpsolver.itc.exam.neighbour
GreatDeluge.Random=true
GreatDeluge.CoolRateInv=9000000
# Exam 5 - 1.05 Exam 6 - 1.06
GreatDeluge.UpperBoundRate=1.07
GreatDeluge.LowerBoundRate=0.90

#SA
#Itc.NextHeuristicsOnReheat=true
#Itc.NextHeuristicsOnReheat.AlterBound=true
#Itc.Third=net.sf.cpsolver.itc.heuristics.search.ItcSimulatedAnnealingSeq
#SimulatedAnnealing.Neighbours=net.sf.cpsolver.itc.heuristics.neighbour.selection.ItcSwapMove;net.sf.cpsolver.itc.exam.n
#SimulatedAnnealing.InitialTemperature=1.5
#SimulatedAnnealing.CoolingRate=0.97
#SimulatedAnnealing.TempLengthCoef=5
#SimulatedAnnealing.ReheatLengthCoef=7

#Other
General.SaveBestUnassigned=0
General.CoolInAccept = true
```

Figure 1: The Examination track properties file.

```
#Other
General.SaveBestUnassigned=0
General.CoolInAccept = true

# End the sequence after x iterations it cant find a neighbour
Sequence.MaxSequenceResetIter = 10000
# 0-L 1-NL 2-DELTA
Sequence.LearningMethod = 2
Sequence.Sequence = true
# 0-RW 1-TORN
Sequence.SelectionType = 1
#Ordering of acceptance. -1 if not used. First MUST be 0.
Sequence.HC = -1
Sequence.GD = 0
Sequence.SA = -1

#Extensions
Exam.AllowDirectConflict=false
Exam.AllowBinaryViolations=false
Extensions.Classes=net.sf.cpsolver.ifs.extension.ConflictStatistics
```

Figure 2: The Examination track properties file.

The key configuration settings you need to know are, Termination.Timeout, Neighbour.Class, Itc.Third and all of the Sequence configurations.

Firstly, Termination.Timeout sets the length of time the solver is allowed to run for. For the runs performed in this project we set the timeout to 247 seconds as instructed by the ITC 2007 benchmarking program. Neighbour.Class configures if the neighbour selection uses Muller's original selection or the modified selection method. If Neighbour.Class is set to 'net.sf.cpsolver.itc.heuristics.ItcNeighbourSelection' Muller's original 2007 code will be executed. You must add 'Seq' to the end to execute the modified solver, that is, 'net.sf.cpsolver.itc.heuristics.ItcNeighbourSelectionSeq'.

The section bulk commented under the 'SA' comment is all for Simulated Annealing. If you wish to run Simulated Annealing for the track, then you must uncomment all the commented properties. It is important to note if Itc.Third is equals 'net.sf.cpsolver.itc.heuristics.search.ItcSimulatedAnnealing' Muller's original code will be ran. So, to run Muller's original code with Simulated Annealing both Neighbour.Class and Itc.Third need to be configured correctly. To run the modified Simulated Annealing, you must again add 'Seq' to the end. This then becomes 'net.sf.cpsolver.itc.heuristics.search.ItcSimulatedAnnealingSeq'.

The sequence properties will only be used when running the modified solver. 'Sequence.MaxSequenceResetIter' refers to the number of iterations in a row that the solver is allowed to unsuccessfully attempt to get a neighbour. After this the sequence will end and continue from the start. All the sequence scores remain the same.

'Sequence.LearningMethod' refers to the reward mechanism used. 0 is linear, 1 is non-linear and 2 is delta learning.

'Sequence.Sequence' should always be true but it allows you to disable to sequence building and have just the learning mechanism. If the learning mechanism is set to anything other than 0, 1 or 2 no learning will take place. Therefore, with 'Sequence.Sequence' set to false and with no learning the selection will be random.

'Sequence.SelectionType' is the heuristic selector used. There are only 2 implemented within the solver. 0 is roulette wheel and 1 is tournament selection.

'Sequence.HC', 'Sequence.GD' and 'Sequence.SA' refers to the ordering of the acceptance strategies. These must be configured together and correctly for the solver to work as intended. Any acceptance strategy you don't want to use can be set to any negative number, e.g. -1. Then the order of the Sequence must be assigned with the first strategy being configured to 0, then the next to 1. For example, in Figure 2 only Great Deluge will be run. If you wish to run Simulated Annealing, you must uncomment the Simulated Annealing properties. If you wish to run Hill Climbing, then Simulated Annealing and then Great Deluge you would set the configurations as displayed in Figure 3. These are all the key points to the config files. The config files for each track follow the same structure, the only differences being the neighbours used and parameter settings.

```
#Ordering of acceptance. -1 if not used. First MUST be 0.
Sequence.HC = 0
Sequence.GD = 2
Sequence.SA = 1
```

Figure 3: Config to set acceptance strategy to HC-SA-GD.

With the track configured you must then run the solver with the correct arguments. You can find the main function for the solver in the class 'ItcTest', in the package 'net.sf.cpsolver.itc.ItcTest'. The solver takes two mandatory arguments; the track you want to run and the path of the dataset you wish to solve. All the data instances are submitted with the code in the 'data' folder. The solver can also take three additional arguments: the output directory of the solution file, the timeout time and the seed number. For example, if you wish to run instance 5 of the Examination track for 247 seconds your arguments would be {exam data/exam/exam_comp_set5.exam exam_out/exam_comp_set5.sln 247}. This is displayed in an IDE in Figure 4 and the Mac terminal in Figure 5. If the timeout is not set, it'll use the timeout set in the tracks properties file and if a seed is not set, a random seed will be used.

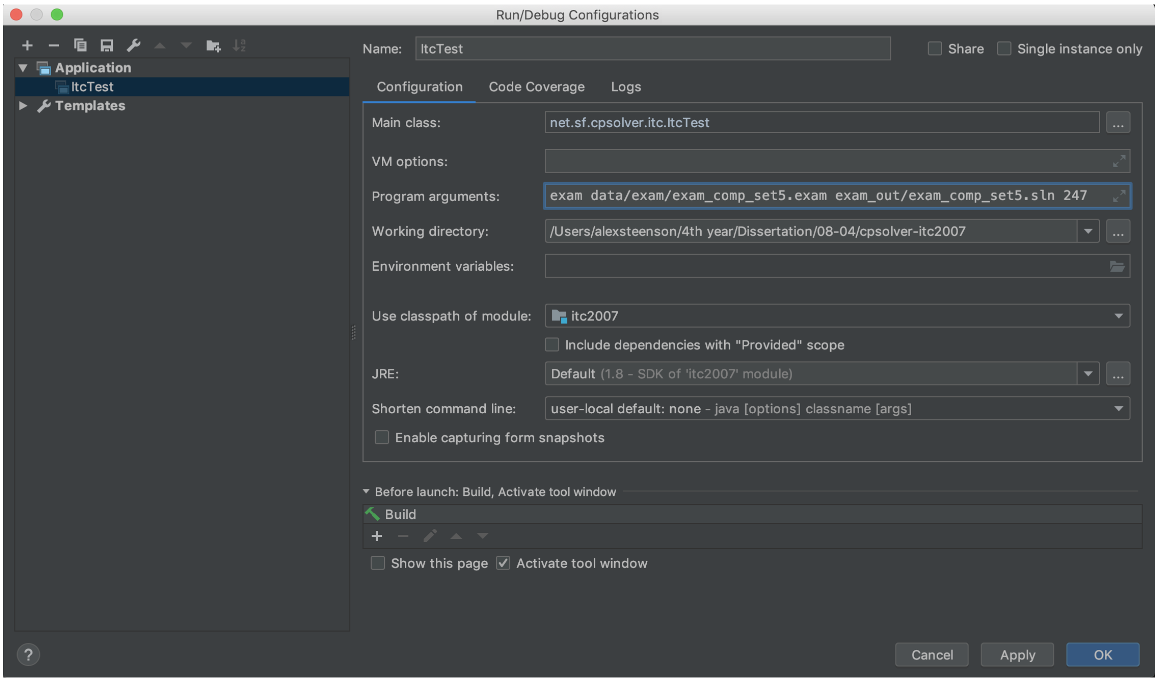

Figure 4: Program arguments for data instance 5 on the Examination track.

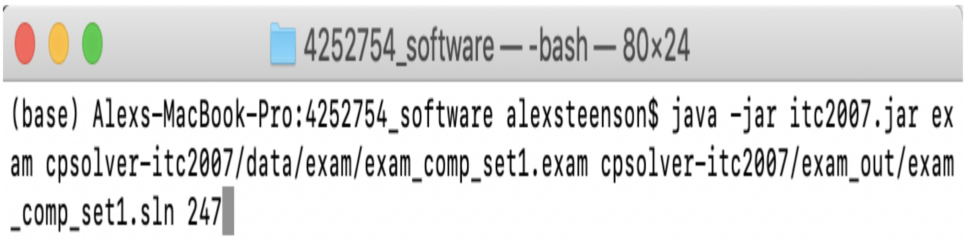

Figure 5: Program arguments for data instance 5 on the Examination track.

When you execute the solver it will then run using the properties set in 'exam.properties'. You would follow the same steps to run a data instance on a different track. You can also find your solution in the directory set in the third argument. With this '.sln' file you can use the validator issues by the ITC 2007 to validate the feasibility and solution value of the solution.

When the solver has finished it will write the best solution value, the iteration the best solution value was obtained on and the distance to feasibility in the file 'solutionValues.txt'. These values can then be used for future evaluation. Whilst the framework is solving a given problem instance, information will be output in the console about the progress of the solver. The current best solution value will be output during the solving process.

The benchmarking program to determine how long you are allowed to run the solver for on your computer can be located in the software folder, then under doc/benchmark.