Algorithm Configuration Survival Guide Part 2

SIGEVO Summer School

GECCO 2021

Leslie Pérez Cáceres

leslie.perez@pucv.cl

Lets configure…

What are we going to use?

- R https://cran.r-project.org
- The irace package https://iridia.ulb.ac.be/irace/

- ACOTSP https://iridia.ulb.ac.be/dorigo/ACO/aco-code/public-software.html

- Repository https://github.com/leslieperez/configuration_lab

Lets configure…

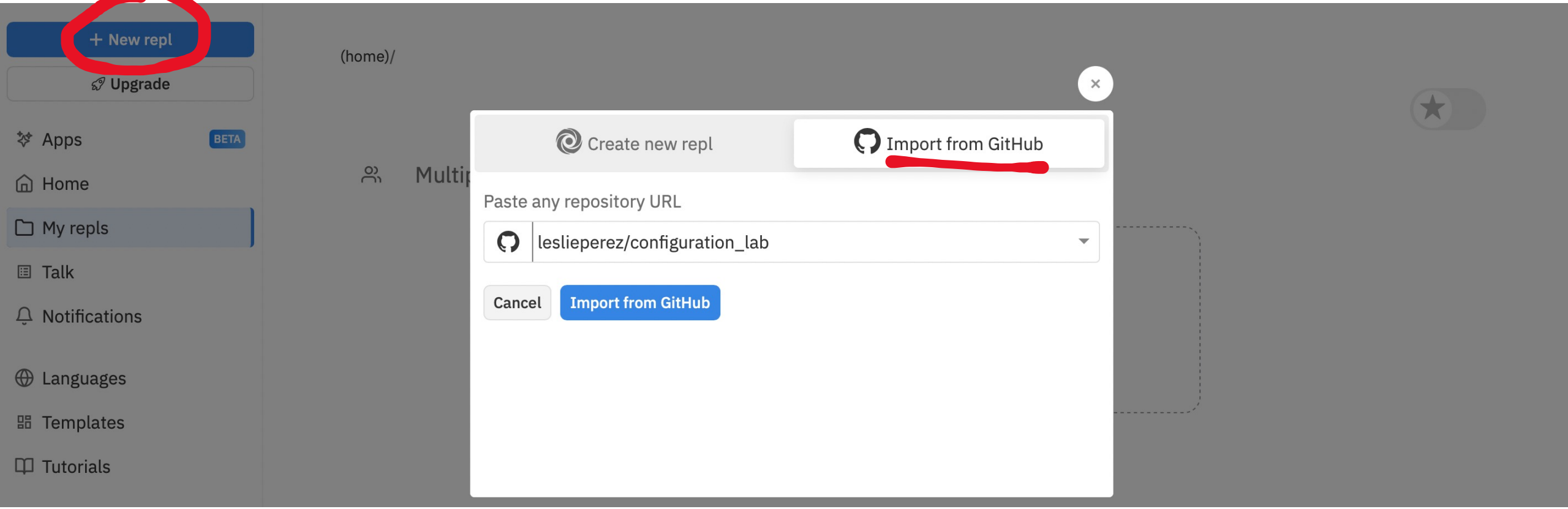
If you want to work in your computer:

- Rstudio https://www.rstudio.com

Don’t want to install anything?

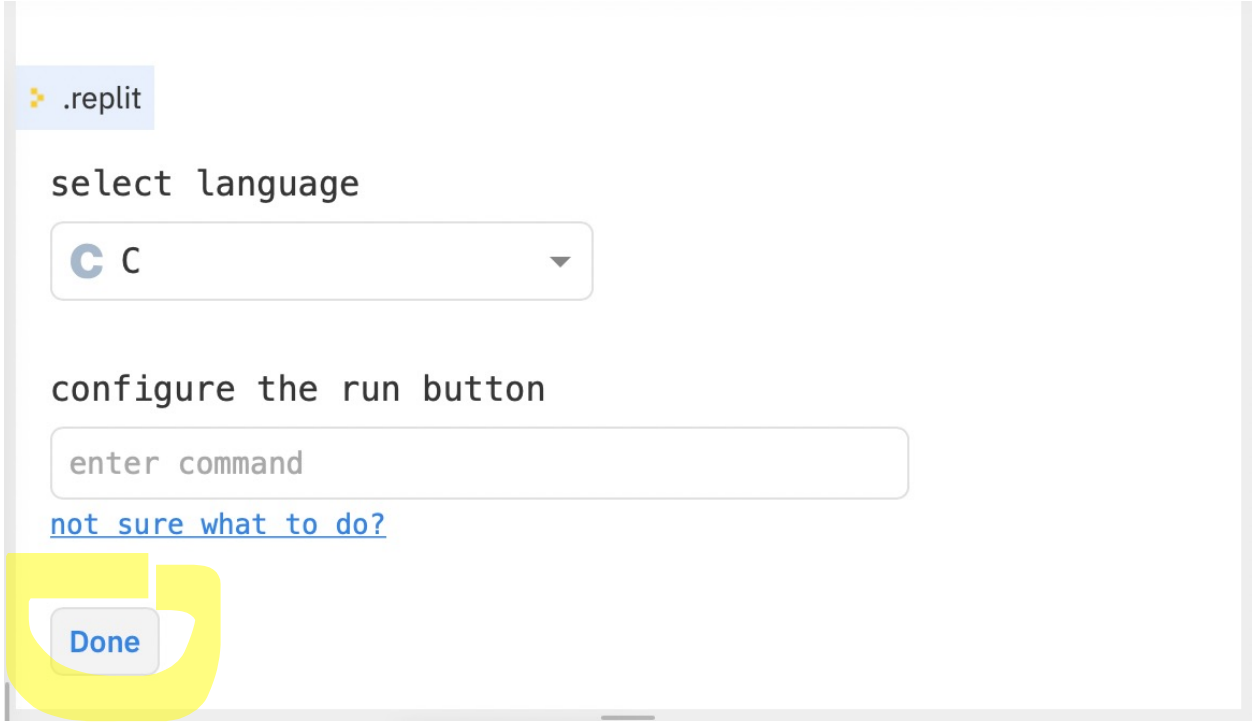
- http://www.replit.com

Using replit: http://www.replit.com



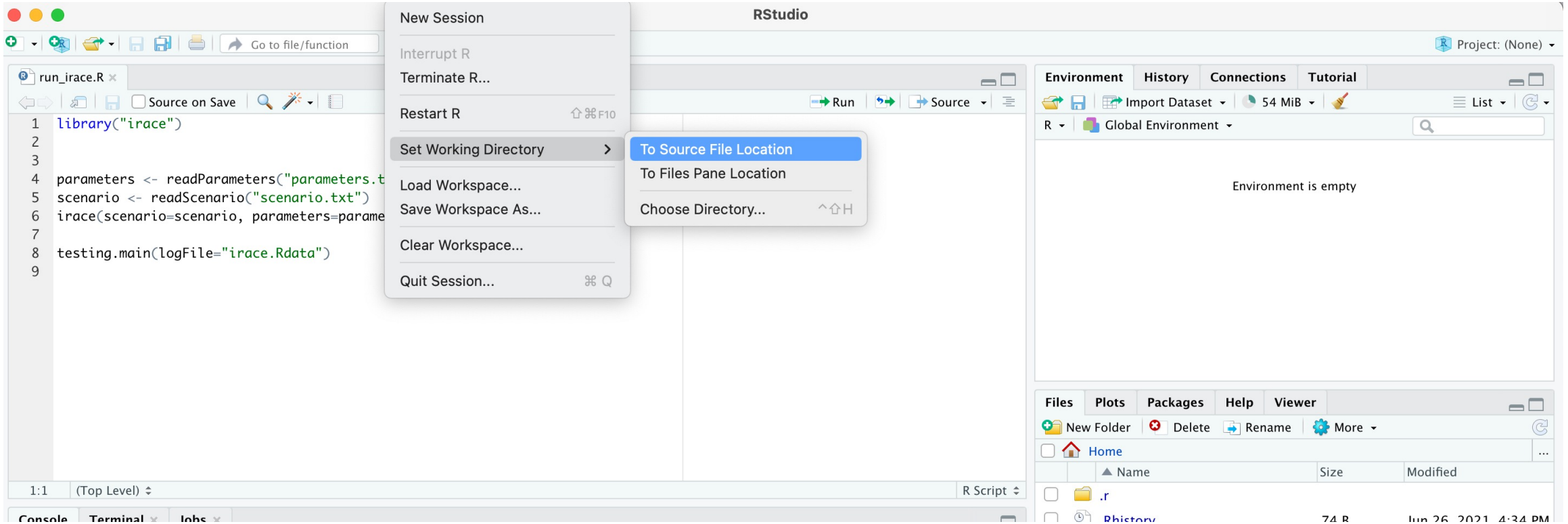
Using replit

- Set C as language (ACOTSP)



Using Rstudio

Open the run_irace.R file in Rstudio and set the working directory:



Prepare ACOTSP

Go to the shell:

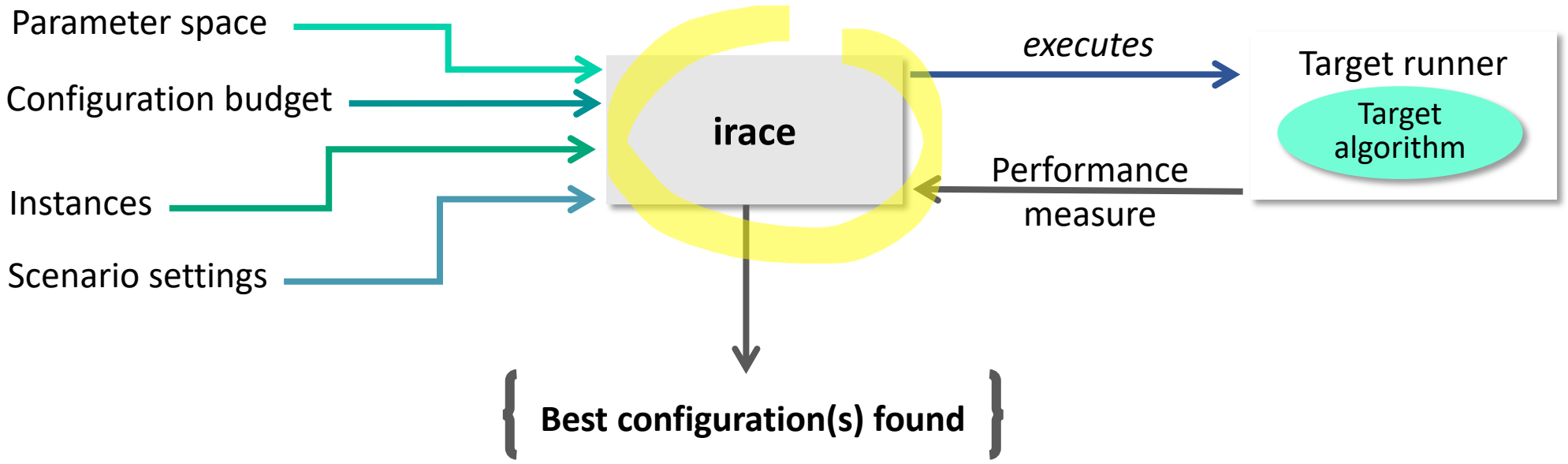
```
$ cd ACOTSP-1.03
$ make
```

- In your machine, compile as you normally do with C code

Before we can configure

Lets create the configuration setup

Configuration scenario setup

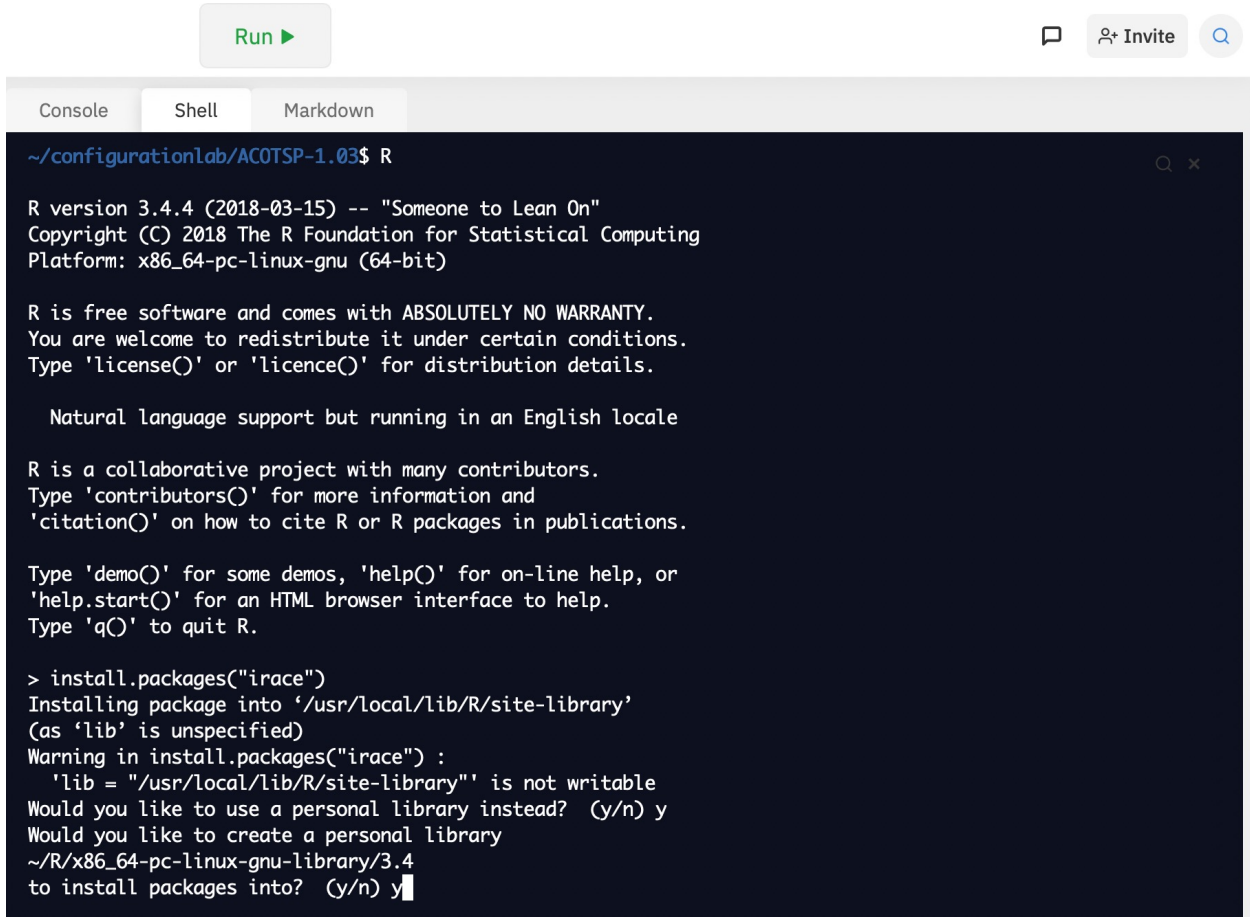


Installing irace

Open the R console in the shell and install irace

```
$ R
> install.packages("irace")
```

Install irace in the local library

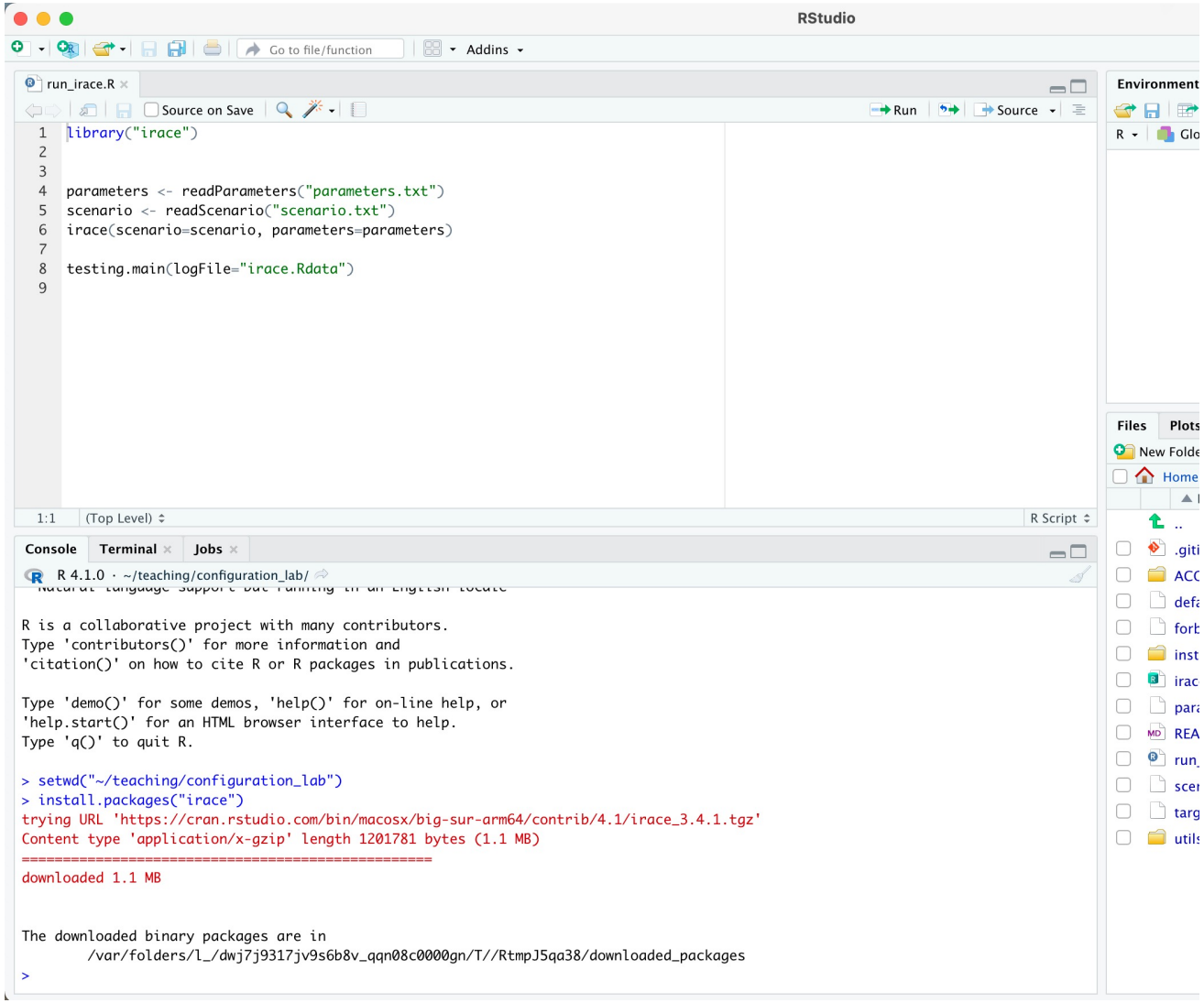


Installing irace

Open the R console in the shell and install irace

```
$ R
> install.packages("irace")
```

Install irace in the local library



Installing irace

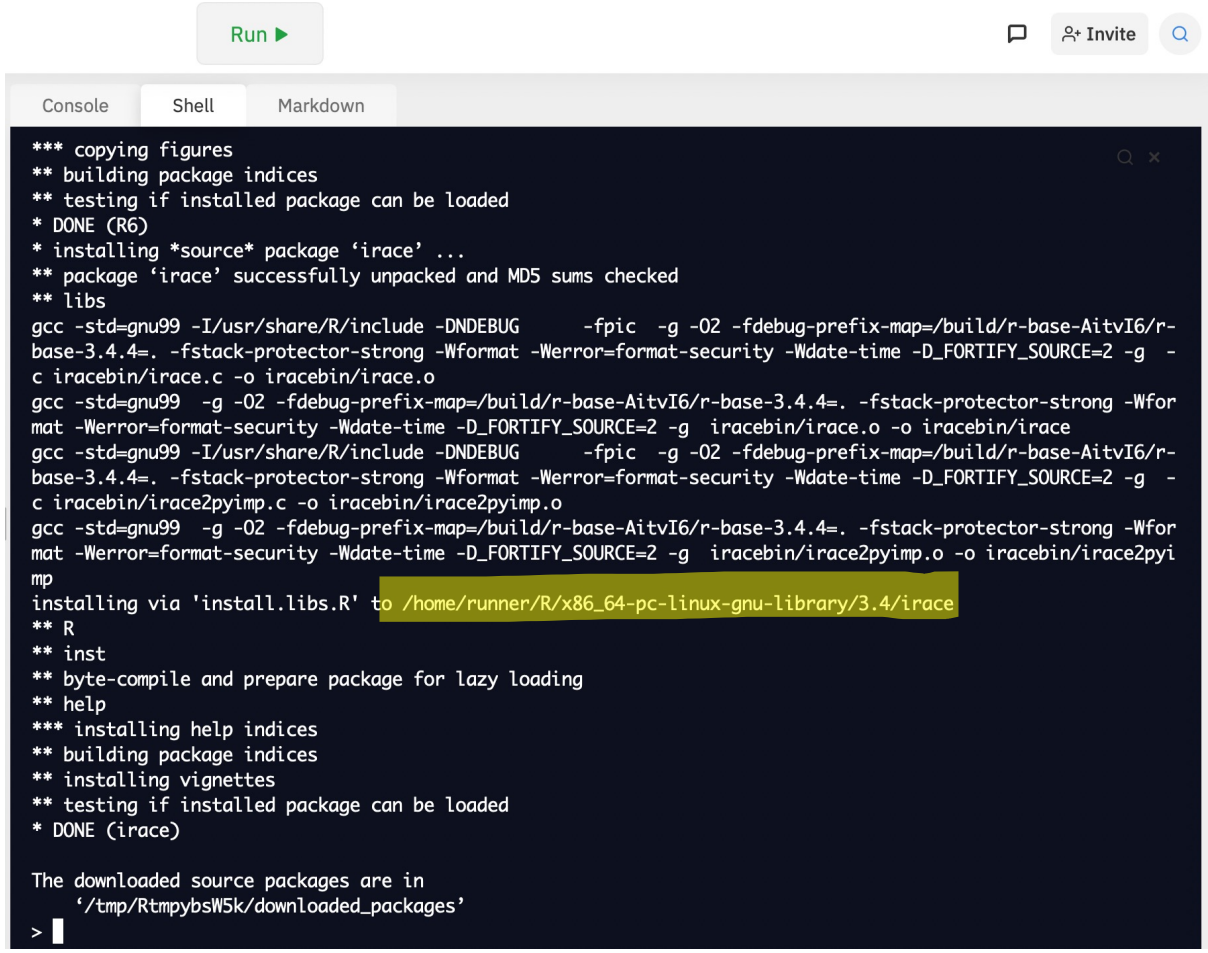
You can use irace from the **R console** or the **shell**.

- In the R console just load irace

```
> library("irace")
```

- In the shell use the executable

```
$ {irace_folder}/bin/irace
```



You can create a symbolic link:

```
$ ln -s /home/runner/R/x86_64-pc-linux-gnu-library/3.4/irace/bin/irace .
```

Options of irace

Options can be provided by:

- `scenario.txt`
- Arguments
- Direct setting

Options have a name and a flag

Ej. `parameterFile (--parameter-file)`

scenario.txt

```
########################################################## -*- mode: r -*- #####
## Scenario setup for Iterated Race (iRace).
############################################################################

## To use the default value of a parameter of iRace, simply do not set
## the parameter (comment it out in this file, and do not give any
## value on the command line).


## File that contains the description of the parameters.
parameterFile = "parameters.txt"
configurationsFile = "default.txt"
forbiddenFile = "forbidden.txt"



## Directory where the programs will be run.
execDir = "."


## Directory where tuning instances are located, either absolute path or
## relative to current directory.
trainInstancesDir = "../instances/small/training/"


## Set testing data
testInstancesDir = "../instances/small/testing/"
testNumElites = 5


## The maximum number of runs (invocations of targetRunner) that will performed. It
## determines the (maximum) budget of experiments for the tuning.
maxExperiments = 300


## Number of CPU cores to use
parallel = 3


## A value of 0 silences all debug messages. Higher values provide
## more verbose debug messages.
# debugLevel = 0
```

Options of irace

The irace package user guide:

https://cran.r-project.org/web/packages/irace/vignettes/irace-package.pdf

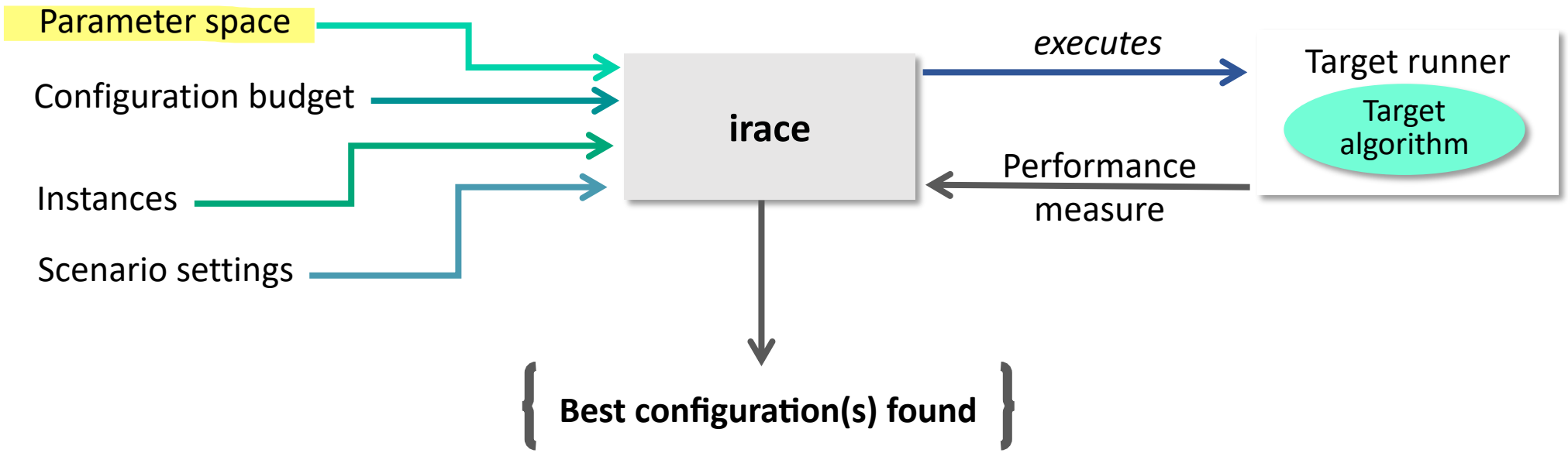
The irace package help:

```
> library("irace")
> irace.usage()
```

Or

```
$ ./irace --help
```

Configuration scenario setup



Parameters space

1. List all parameters involved in your algorithm
   - Include also hidden ones (magical constants)

2. Identify conditional parameters

3. Define an appropiate type:
   1. Caterorical
   2. Ordinal
   3. Numerical (real, integer, real log, integer log)

components

behaviour

4. Define a reasonable domain for each parameter

Parameters space

5. Check for configurations already known
   - We can provide them to irace

6. Check for “issues” already known
   - Parameter values combinations that are not desired
   - Parameter values combinations that could cause problems

Parameter space example

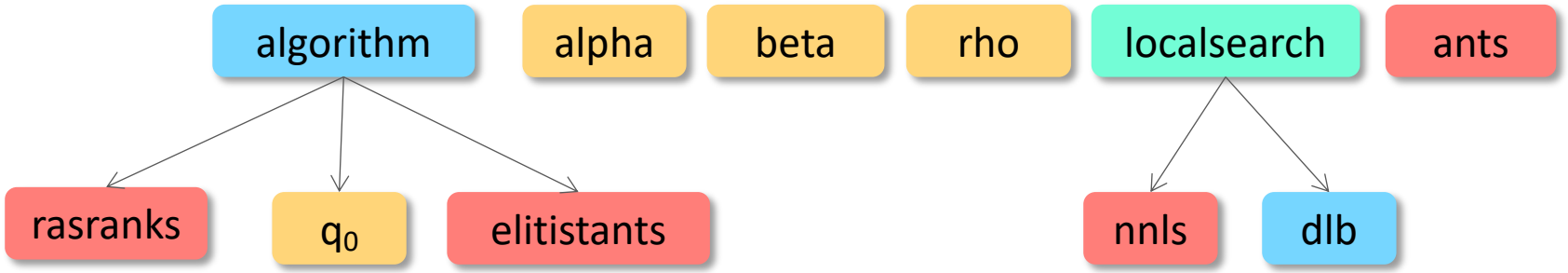
ACOTSP: Framework of Ant Colony Optimization algorithms for TSP (Thomas Stutzle)

http://www.aco-metaheuristic.org/aco-code/public-software.html

Which algorithms?

- Ant System
- Elitist Ant System
- Max-min Ant System
- Rank based Ant System
- Best-worst Ant System
- Ant Colony System

Parameter space example



Parameter space example

`parameters.txt`: describes the parameters to be configured

```
# name          switch            type   values          [conditions (using R syntax)]
algorithm     "--"              c  (as,mmas,eas,ras,acs)
localsearch   "--localsearch "  c  (0, 1, 2, 3)
alpha         "--alpha "        r  (0.00, 5.00)
beta          "--beta "         r  (0.00, 10.00)
rho           "--rho  "         r  (0.01, 1.00)
ants          "--ants "         i  (5, 100)
nnls          "--nnls "         i  (5, 50)             | localsearch %in% c(1, 2, 3)
q0            "--q0 "           r  (0.0, 1.0)          | algorithm %in% c("acs")
dlb           "--dlb "          c  (0, 1)              | localsearch %in% c(1,2,3)
rasrank       "--rasranks "     i  (1, 100)            | algorithm %in% c("ras")
elitistants   "--elitistants ”  i  (1, 750)            | algorithm %in% c("eas")
```

Parameter space example

`default.txt`: default parameter values or known configurations

```
algorithm localsearch alpha beta rho  ants nnls dlb q0 rasrank elitistants
as        0           1.0   1.0  0.95 10   NA   0   NA  NA      NA
```

Values must comply with conditionality conditions

`forbiddenFile (--forbidden-file)`

Parameter space example

forbidden.txt: forbidden combinations of parameter values

```
## Examples of valid logical operators are: ==  !=  >=  <=  >  <  &  |  !  %in%
(alpha > 2.0) && (beta > 3.0)
```

Warning

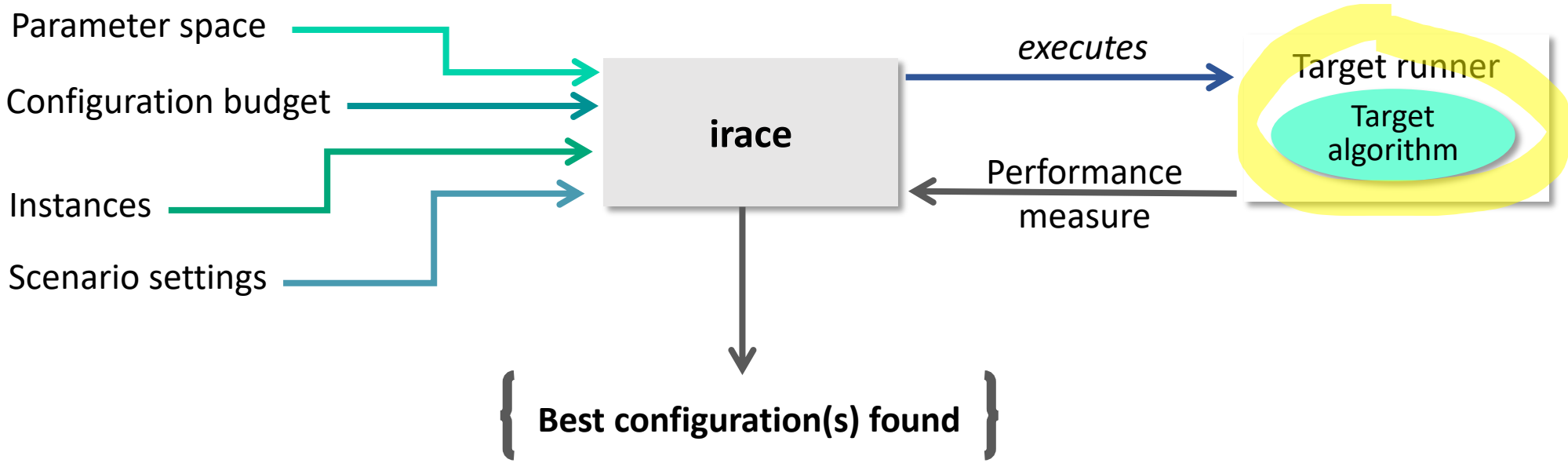
You might be suspecting that irace could be used like a…

… debugging tool

Yes!

For better or worse

Configuration scenario setup



Target algorithm execution

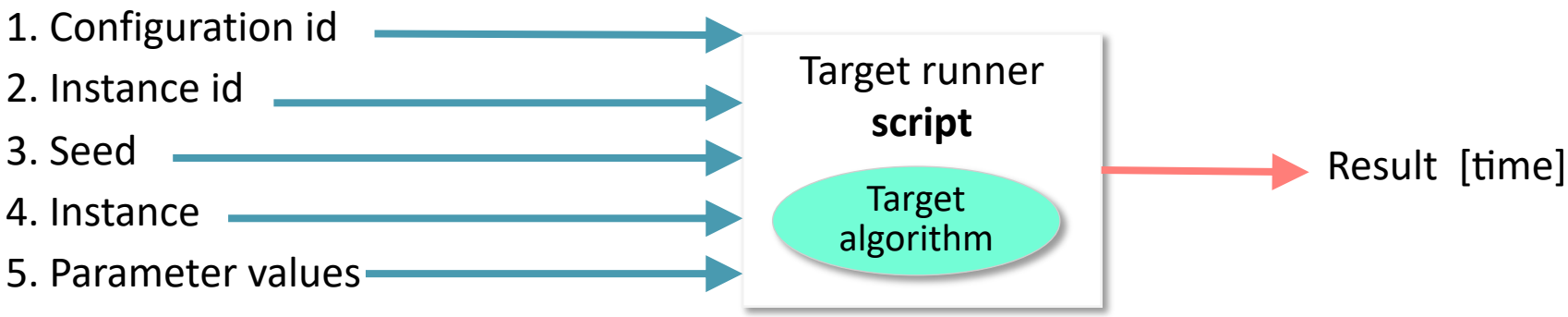
The target runner is a script that allows irace to execute the target algorithm

Options:

- Script file
  - Use file targetRunner in the folder
- R function
  - Use function provided in scenario.txt

Target algorithm execution: script file

The script can be provided in any programming language



Target algorithm: script file

`targetRunner (--target-runner)`

```
#!/bin/bash
# Path to the ACOTSP software
EXE=../ACOTSP-1.03/acotsp

# Read experiment information
CONFIG_ID="$1"
INSTANCE_ID="$2"
SEED="$3"
INSTANCE="$4"
shift 4 || error "Not enough parameters"
CONFIG_PARAMS=$*

# Fixed parameters that should be always passed to ACOTSP
FIXED_PARAMS=" --tries 1 --time 5 --quiet "
```

Target algorithm: script file

`targetRunner (--target-runner)`

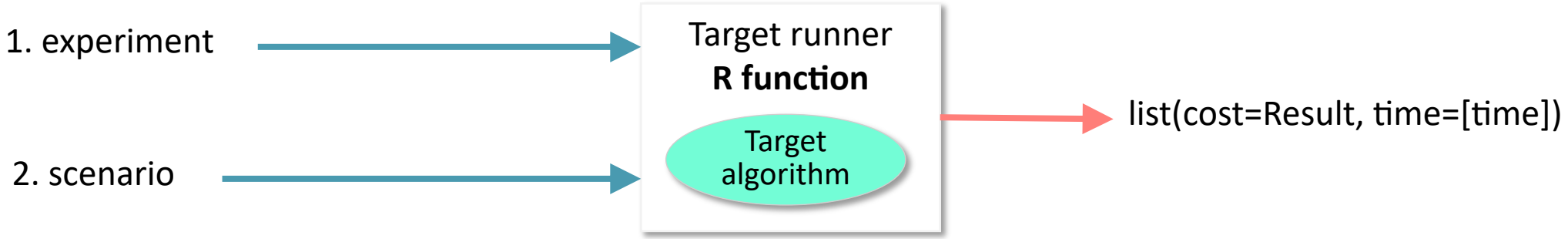
```
# File to write the execution output
STDOUT=c${CONFIG_ID}-${INSTANCE_ID}-${SEED}.stdout
STDERR=c${CONFIG_ID}-${INSTANCE_ID}-${SEED}.stderr

# Now we can call ACOTSP
$EXE ${FIXED_PARAMS} -i $INSTANCE --seed $SEED \
     ${CONFIG_PARAMS} 1> $STDOUT 2> $STDERR

# The following line is to extract the result
COST=$(cat ${STDOUT} | grep -o -E 'Best [-+0-9.e]+' | cut -d '
' -f2)

# Print it!
echo "$COST"
```

Target algorithm execution: R function



Target algorithm: R function

`targetRunner (--target-runner)`

```
## R function that executes the target algorithm
targetRunner = function (experiment, scenario) {
  exe = "../ACOTSP-1.03/acotsp"

  candidate_id  = experiment$id.configuration
  instance_id   = experiment$id.instance
  seed          = experiment$seed
  instance_name = experiment$instance

  # Define fixed parameters (budget or other options)
  fixed_params = "--tries 1 --time 5 --quiet"

  # Create parameter arguments for command line
  parameters    = buildCommandLine(experiment$configuration,
                                   experiment$switches)
```

Target algorithm: R function

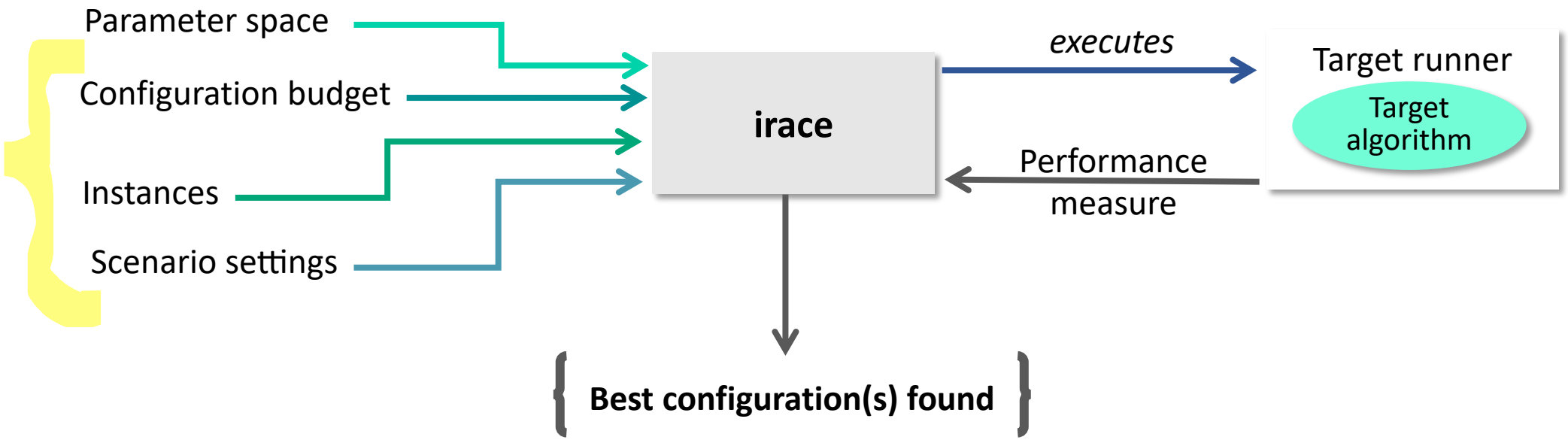
`targetRunner (--target-runner)`

```
# Create command line
 command_line = paste (fixed_params, "-i", instance_name,
                       "--seed", seed, parameters, sep=" ")
 # Execute command
 output = system2(command="exe", args=command_line,
                  stdout=TRUE, stderr=TRUE)

 # Parse output file
 sel = which(sapply(output, grep, pattern="try 0, Best",
             ignore.case = FALSE, value=FALSE) == 1)
 result = as.numeric(str_split(str_trim(str_split(
          output[sel],"," )[[1]][2]), "\\s")[[1]][2])

 # Return result
 return(result)
```

Configuration scenario setup



Configuration budget

Configuration bugdet can be specified as:

- Number of evaluations: `maxEvaluations (--max-evaluations)`
  - Number of target runner executions allowed

- Time: `maxTime (--max-time)`
  - Max time allowed (in seconds)
  - You must provide the time as output in target runner!

Instance set

Instances can be provided in:

- Folder: `trainInstancesDir (--train-instances-dir)`
  - Files inside the folder will be used as instances

- File: `trainInstancesFile (--train-instances-file)`
  - Lines in the file will be used as instances

- Test instances can be provided in same way

`testInstancesDir (--test-instances-dir)`

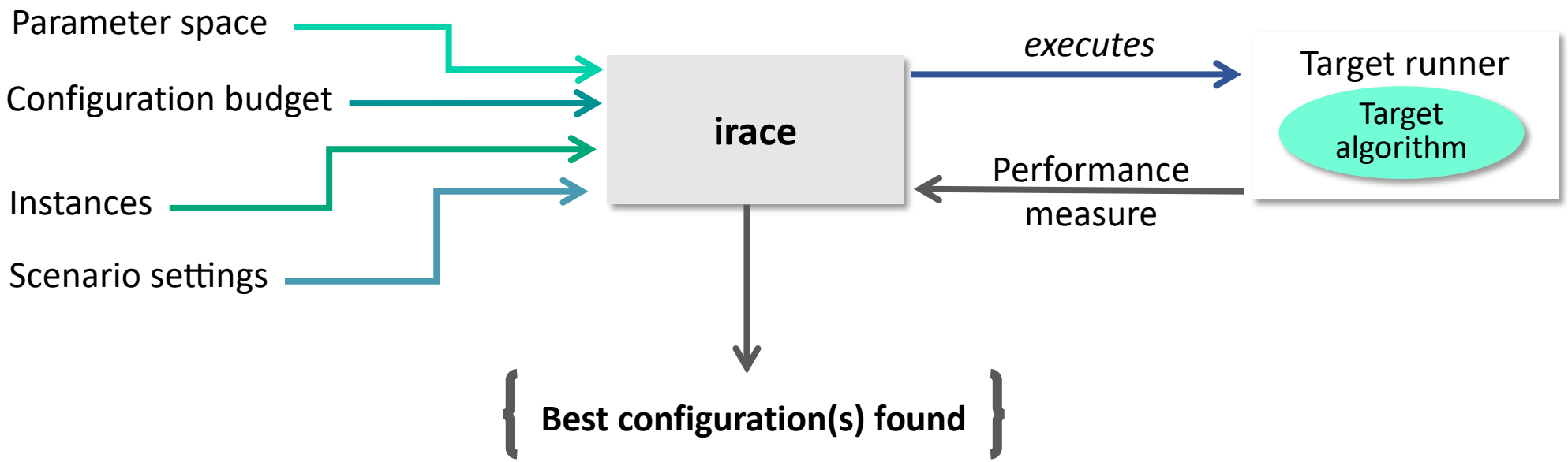
`testInstancesFile (--test-instances-file)`

Scenarios settings

There are many options inside irace!

- `capping (--capping)`: enables adaptive capping for runtime configuration
- `testType (--test-type)`: selects the statistical test used for elimination
- `deterministic (--deterministic)`: set the target algorithm as deterministic
- `parallel (--parallel)`: set the number of cores to be used in the evaluation

Configuration scenario setup



Check the `scenario.txt` file where the options are set

Executing irace

Go to `automatic` folder

In the R console:

```
> parameters = readParameters("parameters.txt")
> scenario = readScenario("scenario.txt")
> irace(scenario = scenario, parameters=parameters)
```

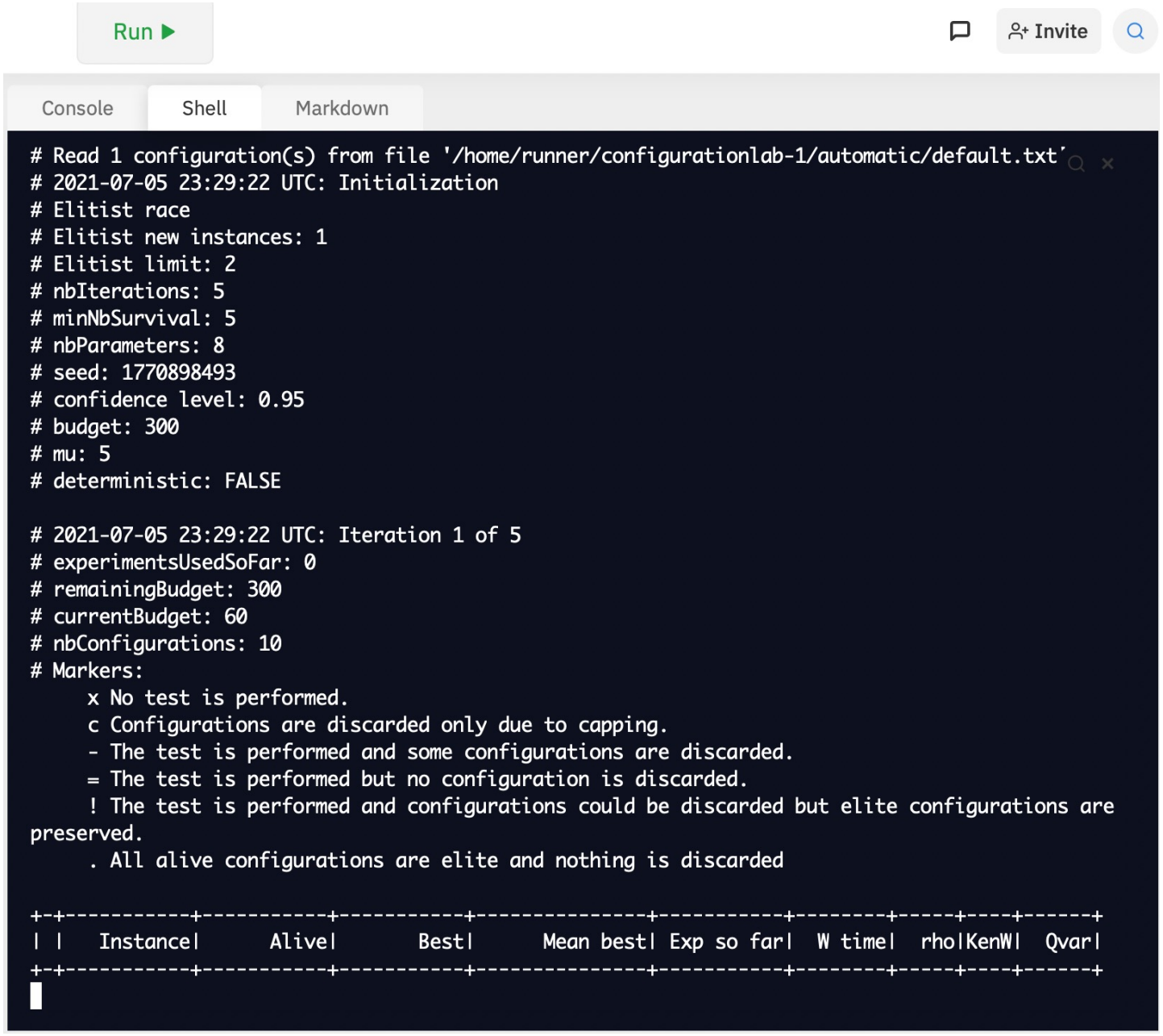
- From the shell:

```
$ irace --scenario scenario.txt
```

or

```
$ Rscript run-irace.R
```

Executing irace



During the execution

```
# 2021-07-05 22:36:50 -03: Iteration 1 of 5
# experimentsUsedSoFar: 0
# remainingBudget: 300
# currentBudget: 60
# nbConfigurations: 10
# Markers:
     x No test is performed.
     c Configurations are discarded only due to capping.
     - The test is performed and some configurations are discarded.
     = The test is performed but no configuration is discarded.
     ! The test is performed and configurations could be discarded but elite configurations are preserved.
     . All alive configurations are elite and nothing is discarded

+-+----------+----------+----------+--------------+----------+--------+-----+----+------+
| |   Instance|      Alive|       Best|      Mean best| Exp so far|  W time|  rho|KenW|  Qvar|
+-+----------+----------+----------+--------------+----------+--------+-----+----+------+
|x|          1|         10|         10|    2773.000000|         10|00:00:23|   NA|  NA|    NA|
|x|          2|         10|         10|    26539.00000|         20|00:00:24|+0.98|0.99|0.0109|
|x|          3|         10|         10|    36006.00000|         30|00:00:21|+0.98|0.98|0.0169|
|x|          4|         10|         10|    34559.00000|         40|00:00:20|+0.95|0.96|0.0219|
|-|          5|          1|         10|    36432.00000|         50|00:00:20|   NA|  NA|    NA|
+-+----------+----------+----------+--------------+----------+--------+-----+----+------+
Best-so-far configuration:          10    mean value:     36432.00000
Description of the best-so-far configuration:
   .ID. algorithm localsearch  alpha   beta    rho ants q0 rasrank elitistants
10   10      mmas           0 1.3508 2.2476 0.6061   58 NA      NA          NA
   .PARENT.
10       NA

# 2021-07-05 22:38:40 -03: Elite configurations (first number is the configuration ID; listed from best to worst according to the sum of ranks):
   algorithm localsearch  alpha   beta    rho ants q0 rasrank elitistants
10      mmas           0 1.3508 2.2476 0.6061   58 NA      NA          NA
```

Current best configuration

Once the configuration budget is over…

```
# 2021-07-05 23:48:48 -03: Elite configurations (first number is the configuration ID; listed from best to worst according to the su
# 2021-07-05 23:48:48 -03: Elite configurations (first number is the configuration ID; listed from best to worst according to the sum of ranks):
   algorithm localsearch  alpha   beta    rho ants     q0 rasrank elitistants
43       acs           0 1.7807 6.9714 0.6217   29 0.8073      NA          NA
36       acs           0 1.8288 6.5449 0.6078   34 0.7417      NA          NA
45       acs           0 1.3133 5.7576 0.5361    8 0.5153      NA          NA
35       acs           0 1.5458 7.5555 0.1707   24 0.4302      NA          NA
26       acs           0 1.7294 8.1875 0.4038   18 0.4167      NA          NA
# 2021-07-05 23:48:48 -03: Stopped because budget is exhausted
# Iteration: 7
# nbIterations: 6
# experimentsUsedSoFar: 300
# timeUsed: 0
# remainingBudget: 0
# currentBudget: 29
# number of elites: 5
# nbConfigurations: 7
   .ID. algorithm localsearch  alpha   beta    rho ants     q0 rasrank
43   43       acs           0 1.7807 6.9714 0.6217   29 0.8073      NA
36   36       acs           0 1.8288 6.5449 0.6078   34 0.7417      NA
45   45       acs           0 1.3133 5.7576 0.5361    8 0.5153      NA
35   35       acs           0 1.5458 7.5555 0.1707   24 0.4302      NA
26   26       acs           0 1.7294 8.1875 0.4038   18 0.4167      NA
   elitistants .PARENT. .ALIVE. .RANK.   .WEIGHT.
43          NA       31    TRUE     37 0.33333333
36          NA       31    TRUE     37 0.26666667
45          NA       35    TRUE     48 0.20000000
35          NA       14    TRUE     49 0.13333333
26          NA        2    TRUE     50 0.06666667
```

Final elite configurations

We also perform the testing

```
# 2021-07-06 00:00:27 -03: Testing configurations (in no particular order): 43 36 45 35 26
   algorithm localsearch  alpha   beta    rho ants     q0 rasrank elitistants
43       acs           0 1.7807 6.9714 0.6217   29 0.8073      NA          NA
36       acs           0 1.8288 6.5449 0.6078   34 0.7417      NA          NA
45       acs           0 1.3133 5.7576 0.5361    8 0.5153      NA          NA
35       acs           0 1.5458 7.5555 0.1707   24 0.4302      NA          NA
26       acs           0 1.7294 8.1875 0.4038   18 0.4167      NA          NA
# Testing of elite configurations: 5
# Testing iteration configurations: FALSE
# 2021-07-06 00:01:57 -03: Testing results (column number is configuration ID in no particular order):
       43     36     45     35     26
1t  12496  12352  12325  12359  12362
2t 313329 307856 313761 325329 312230
3t  49915  51216  51107  50300  51959
4t   9089   9148   9246   9257   9120
5t  39907  39134  39521  39349  41394
# 2021-07-06 00:01:57 -03: Finished testing
```

Testing results

Configuration data

- The configuration process produces a Rdata file (`irace.Rdata`)
  - It contains all performance data gathered by irace
- The data can be loaded in the R console

```
$ R
> load("irace-example.Rdata")
> iraceResults$parameters
> iraceResults$experiments
> iraceResults$allConfigurations
```

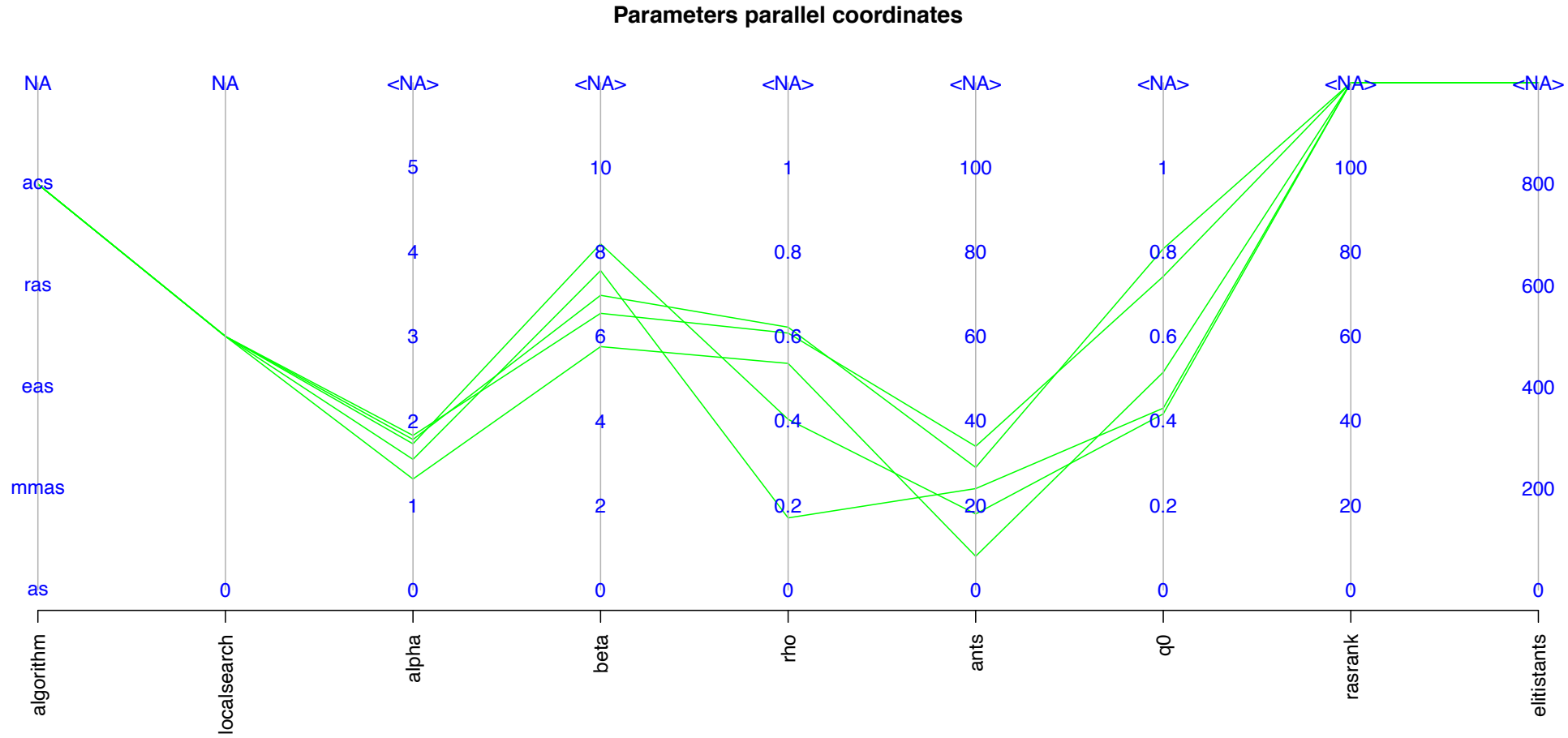
What can we do now?

How best configurations look like?

```
> library("irace")
> load("irace-example.R")
```

```
> parallelCoordinatesPlot(configurations=getFinalElites(iraceResults),
parameters=iraceResults$parameters, hierarchy=FALSE,
filename="pcoordinates")
```

Elite configurations



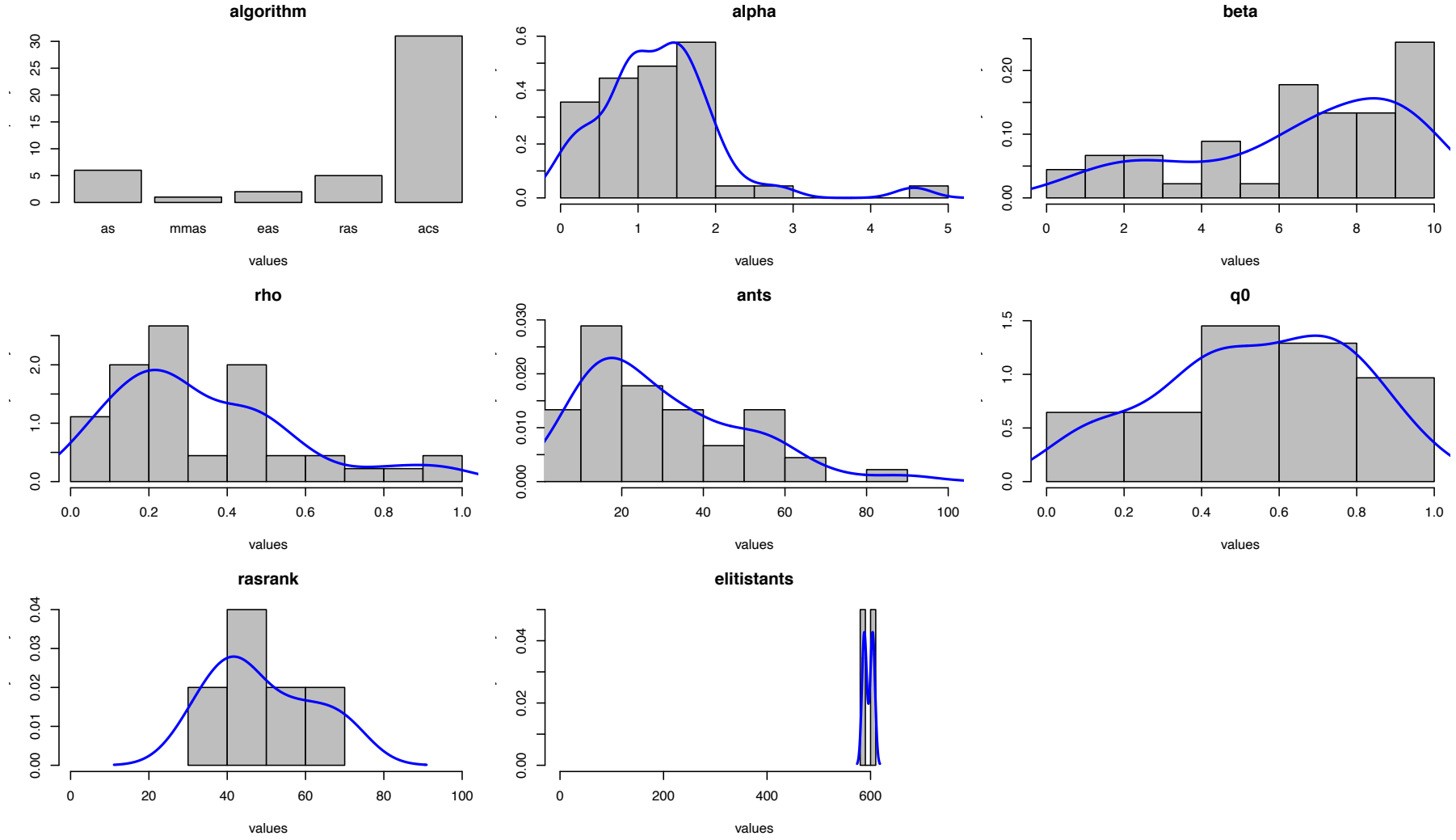
Parameter values

- What parameter values irace focused the search on?

```
> library("irace")
> load("irace-example.R")
```

```
> parameterFrequency(configurations=iraceResults$allConfigurations,
parameters=iraceResults$parameters, filename="frequency")
```

Parameter values



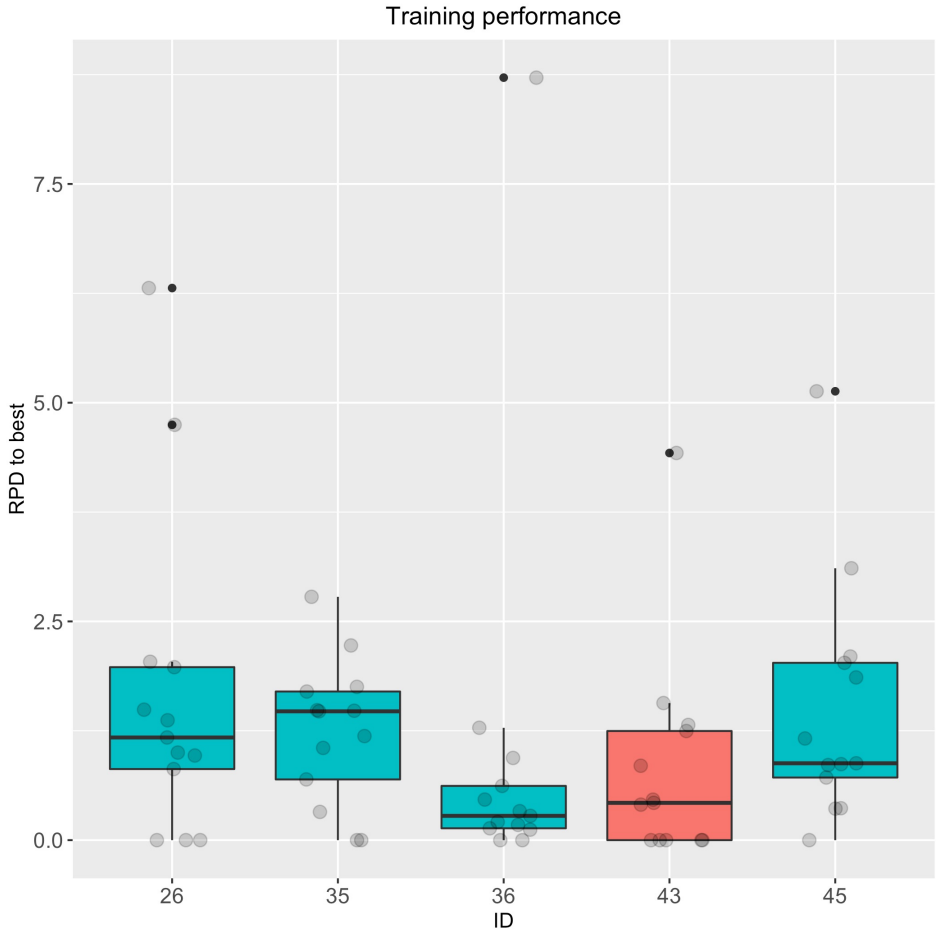
Testing / training performance

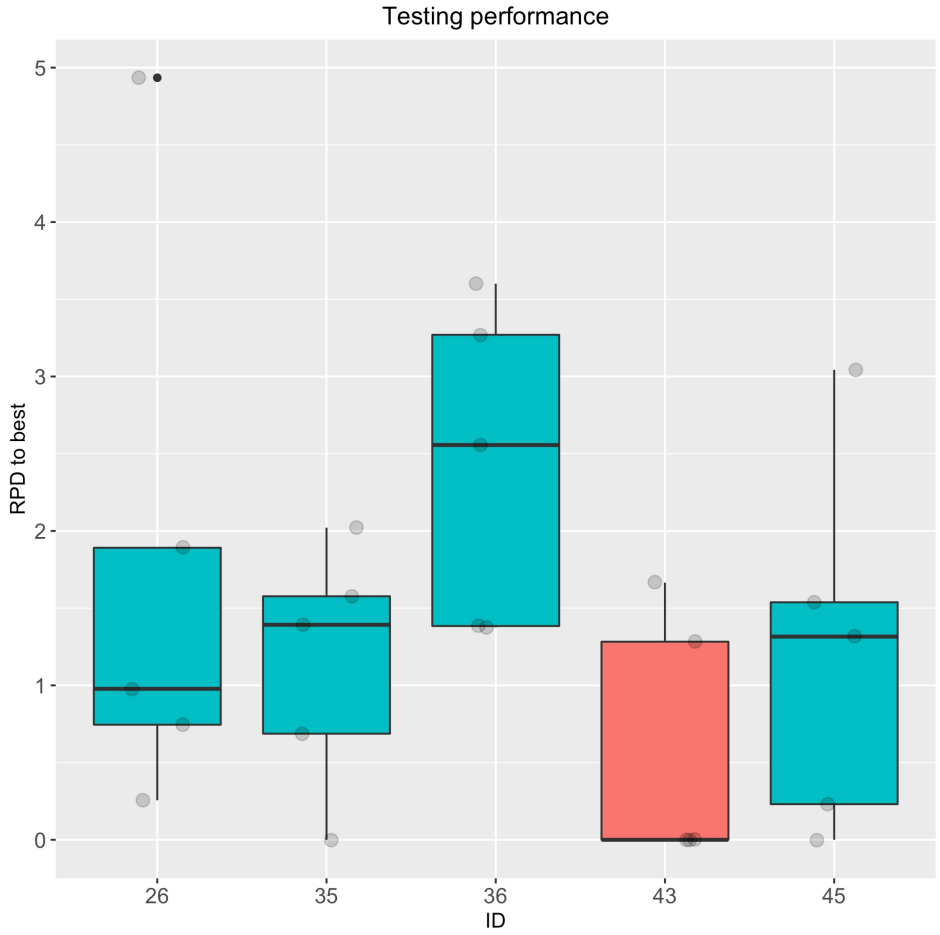
How the performance of the elite configurations looks like?

```
$ R
> install.packages("ggplot2")

> source("utils/boxplot.R")
> load("irace-example.R")
> boxplot_train(iraceResults, file_name="boxplot_train.png")
> boxplot_test(iraceResults, file_name="boxplot.png")
```

Performance of elite configurations





You can also checkout the interface

https://github.com/mrbarrientosg/iraceStudio

Do you want to know more?

Check the tutorial in GECCO 2021!

Automated Algorithm Configuration and Design

- Thomas Stützle *Université Libre de Bruxelles, Belgium*
- Manuel López-Ibáñez *University of Málaga, Spain*

How to contact us & keep updated

- The irace package website

  https://iridia.ulb.ac.be/irace/

- The github repository

  https://github.com/MLopez-Ibanez/irace

- The support group

  https://groups.google.com/g/irace-package?pli=1

Thanks for your attention !

Questions, comments?