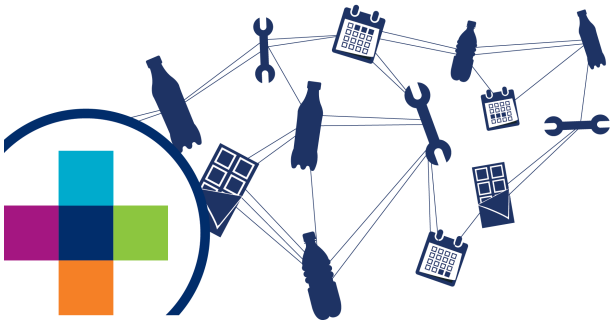
VEROLOG
VEHICLE ROUTING
AND LOGISTICS

ORTEC
OPTIMIZE YOUR WORLD

VeRoLog Solver Challenge 2019



Information for participants

This document contains important information for participants in the VeRoLog Solver Challenge 2019, including a description of the problem, rules and awards. Some useful tools, such as a solution validator and a benchmark program, are available at the challenge website under `https://verolog2019.ortec.com/tools`.

Contents

1 The challenge in a nutshell

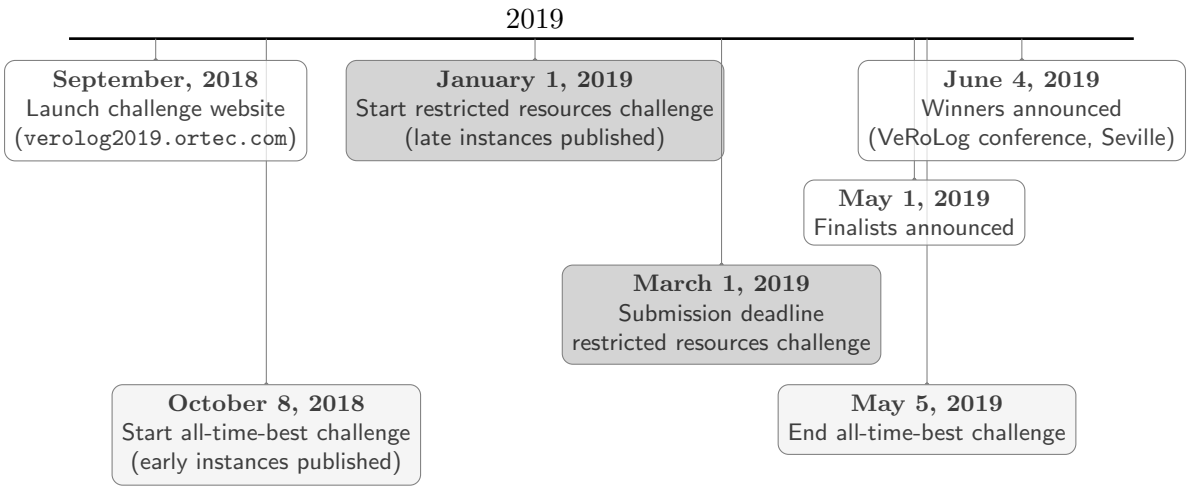
The VeRoLog Solver Challenge 2019 consists of two parts:

1. *An all-time-best challenge*
   The organizers disclose 25 instances (dataset "ORTEC-early") in October 2018. The main goal in this challenge is to find all-time-best solutions to those instances. Progress on the instances is shown on the challenge website (`https://verolog2019.ortec.com`). The all-time-best challenge runs until May 5, 2019. Solutions submitted in this part of the challenge may be obtained by any means available.

2. *A restricted resources challenge*
   This part is a challenge in the more traditional form. Resources are restricted, especially time. To participate you have to run your algorithm on a set of instances (dataset "ORTEC-late") and submit your solutions and implementation no later than March 1, 2019. Based on the submitted results, the organizers select the finalists. The algorithms of the finalists are then applied to a set of hidden instances. Per instance the participants are ranked and the participant with the lowest mean rank is the winner of the challenge.

Participants may compete in one part or in both parts of the challenge. A detailed description of the set-up of the challenge, including the official rules, is given in Section 3. The timeline below gives an overview of the important dates and deadlines.

1.1 Important dates



2 Problem description

The problem of the VeRoLog Solver Challenge 2019 is based on a problem that combines distribution and subsequent installation of equipment, such as vending machines. The machines must be delivered within a customer-dependent delivery window and must be installed by a technician as soon as possible after delivery.

The planning horizon consists of a period of consecutive days, numbered 1, 2, and so on. There are different kinds of machines, each having its own size, expressed in the same unit.

There are machine requests from customers that all have to be satisfied. A request consists of a number of machines of one kind, and a delivery window within which these machines have to be delivered. Each delivery window consists of a number of consecutive days within the planning horizon. If a customer needs more than one kind of machine, a separate request is defined for

each kind. For example, there can be a request for two machines of kind 5 that have to be delivered on day 8, 9 or 10.

Machine delivery. There is one depot location where all the machines are located at the start of the planning horizon. There are enough machines of each kind available to satisfy all the requests. Trucks are hired to transport the machines from the depot to the customers. There is no limit on the number of trucks that can be hired. All trucks are identical. A truck can accommodate any combination of machines whose total size does not exceed the truck's capacity, where the capacity is expressed in the same unit as the machine sizes. The delivery of a request cannot be split, i.e., all machines of a request must be delivered simultaneously by one truck. In the provided instances, none of the request sizes exceeds the vehicle capacity.

The daily route of a truck starts and ends at the depot, i.e., a truck that picks up a machine from the depot on a certain day must end its route at the depot that same day. A truck can return to the depot several times during a day to pick up machines. The total distance a truck can travel per day is limited to a given maximum. It does not take any time to load a machine at the depot or to unload a machine at a customer.

Machine installation. After delivery, each machine must be installed by a technician at the customer location. Installation of a machine cannot take place on the day the machine is delivered. For every full day a machine is 'idle', i.e., delivered at the customer but not yet installed, a fixed penalty is charged. This penalty is specified for each kind of machine. For example, suppose a request with three machines of type A is delivered to a customer on day 4 and installed on day 7. Then the penalty for machine type A is charged $3 \cdot 2 = 6$ times, as three machines were idle on days 5 and 6. If the machines are delivered on day 4 and installed on day 5, no penalty is charged.

Each technician has a skill set that determines which kinds of machines he/she can install. Installing a machine does not take any time. The maximum number of consecutive days a technician can work is 5. If a technician has worked for five consecutive days, he/she must have two days off. If a technician has worked for less than five consecutive days, a single day off suffices. All days outside the planning horizon are days off. The daily route of a technician must start and end at his/her home location. The total distance a technician can travel per day is limited to a given maximum. The same holds for the number of requests a technician can carry out per day, where carrying out a request means installing all the machines for that request.

Objective. In a solution all requested machines are delivered and installed within the planning horizon. The objective is to minimize the total cost. There are costs per unit of distance traveled by truck, for using a truck for a day, and for using a truck at all during the planning horizon. Additionally, there are costs per unit of distance traveled by a technician, for using a technician for a day, and for using a technician at all during the planning horizon. Finally, there are costs for every full day a machine is idle, specified for each kind of machine. In order to determine the traveled distances, integer coordinates are provided for the depot, the customer locations, and the technician's home locations. The distance between coordinates (x_1, y_1) and (x_2, y_2) is defined as $\lceil \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \rceil$, i.e., the ceiling of the Euclidean distance.

2.1 Format of the instance data

We now describe the format of the instance data, which is available in text format. Within each section, the entries are sorted by their (consecutive) IDs. We point out that all IDs in the input are positive integers, and that the depot has location ID 1 in every instance.

Example instance

An example of an instance file in text format follows below.

```
DATASET = VeRoLog solver challenge 2019
NAME = testInstance

DAYS = 7
TRUCK_CAPACITY = 6
TRUCK_MAX_DISTANCE = 250

TRUCK_DISTANCE_COST = 1
TRUCK_DAY_COST = 100
TRUCK_COST = 100000
TECHNICIAN_DISTANCE_COST = 1
TECHNICIAN_DAY_COST = 100000
TECHNICIAN_COST = 100

MACHINES = 4
1 2 200
2 1 200
3 3 500
4 2 100

LOCATIONS = 9
1 10 50
2 20 10
3 50  5
4 33  7
5 40 40
6 70 40
7  1 35
8 10  5
9 25 60

REQUESTS = 7
1 2 1 7 1 1
2 3 3 6 3 1
3 4 2 3 4 2
4 5 5 7 1 3
5 6 1 4 2 1
6 7 1 6 4 1
7 7 3 5 2 4

TECHNICIANS = 5
1 3 100 2 0 1 1 1
2 8 100 1 1 1 0 0
3 8 100 5 0 0 1 0
4 9 100 2 1 1 0 1
5 1  50 1 1 1 1 1
```

Explanation of example instance

Here we give an explanation of the input file. The different sections in the text file will always appear in this given order. However, additional line breaks or spaces may be present. Entries in a line are separated by spaces.

`DAYS` (integer) gives the number of days in the planning horizon. Recall that the days are numbered 1, 2, and so on.
`TRUCK_CAPACITY` (integer) denotes the capacity of one truck.
`TRUCK_MAX_DISTANCE` (integer) is the maximum distance that one truck can travel on a day.

The next section defines the weights for the costs related to the delivery trucks and the technicians.

`TRUCK_DISTANCE_COST` (integer) is the cost per unit of distance traveled by truck.
`TRUCK_DAY_COST` (integer) denotes the cost per route, i.e., the cost per used truck per day.
`TRUCK_COST` (integer) is the cost for using a truck during any day of the planning horizon. This cost should be multiplied by the maximum number of trucks needed on a day; recall that all trucks are identical.
`TECHNICIAN_DISTANCE_COST` (integer) is the cost per unit of distance traveled by a technician.
`TECHNICIAN_DAY_COST` (integer) is the cost per used technician per day.
`TECHNICIAN_COST` (integer) is the cost for using a technician during any day of the planning horizon. This cost should be multiplied by the number of different technicians used during the planning horizon.

The following four sections start with a name, followed by the number of entries in that section. For example, `MACHINES = 4` indicates that there are 4 machine kinds, the details of which are described in the following 4 lines.

Under `MACHINES`, the different machine kinds are listed. The first entry is the machine kind ID. The second entry gives the size of one machine of this machine kind (integer). The third entry denotes the penalty that is charged for every full day a machine of this kind is idle.

Under `LOCATIONS`, the coordinates of the depot location, the customer locations, and the technicians' home locations are given. The first entry is the location ID, the second and third entries denote the actual x- and y-coordinates, respectively, of the location (integers). Recall that the depot always has location ID 1. It is possible that a technician has location ID 1, too; in that case, the technician is based at the depot. Customers cannot have location ID 1.

Under `REQUESTS`, one can find all machine requests by the customers. The first entry is the request ID. The second entry denotes the location ID of the customer that created this request. The next two entries specify the first and last day of the delivery window for this request. The last two entries give the machine kind ID and the number of requested machines of this kind.

Under `TECHNICIANS`, the different technicians are listed. The first entry is the technician ID. The second entry is the location ID. Note that a technician's location ID can coincide with the location ID of the depot or a customer, and that different technicians can be based at the same location and thus have the same location ID. The third entry denotes the maximum distance the technician can travel per day, the fourth entry denotes the maximum number of requests the technician can carry out per day. The remaining entries, one for each machine kind, specify which machines the technician can (1) and cannot (0) install. For example, if there are 4 machine kinds and the last 4 entries of a technician are `1 1 0 1`, then this technician can install machine kinds 1, 2, and 4, but cannot install machine kind 3.

2.2 Format of the solution data

The solution file specifies the routes for the trucks and the technicians each day. Solution files should be submitted as text files. We will now explain how a solution file should be constructed.

Example solution

An example of a solution file in text format follows below.

```
DATASET = VeRoLog solver challenge 2019
NAME = testInstance

TRUCK_DISTANCE = 394
NUMBER_OF_TRUCK_DAYS = 3
NUMBER_OF_TRUCKS_USED = 1
TECHNICIAN_DISTANCE = 312
NUMBER_OF_TECHNICIAN_DAYS = 5
NUMBER_OF_TECHNICIANS_USED = 3
IDLE_MACHINE_COSTS = 400
TOTAL_COST = 601706

DAY = 1
NUMBER_OF_TRUCKS = 0
NUMBER_OF_TECHNICIANS = 0

DAY = 2
NUMBER_OF_TRUCKS = 1
1 5 3
NUMBER_OF_TECHNICIANS = 0

DAY = 3
NUMBER_OF_TRUCKS = 1
1 6 7 0 1 2
NUMBER_OF_TECHNICIANS = 1
1 5

DAY = 4
NUMBER_OF_TRUCKS = 0
NUMBER_OF_TECHNICIANS = 3
1 3 2
2 1
4 7

DAY = 5
NUMBER_OF_TRUCKS = 1
1 4
NUMBER_OF_TECHNICIANS = 0

DAY = 6
NUMBER_OF_TRUCKS = 0
NUMBER_OF_TECHNICIANS = 1
4 6 4

DAY = 7
NUMBER_OF_TRUCKS = 0
NUMBER_OF_TECHNICIANS = 0
```

Explanation of example solution

Here we give an explanation of the data file.

The first two lines (`DATASET` and `NAME`) refer to the instance that the solution file belongs to. The next section, listing some characteristics of the solution (starting with `TRUCK_DISTANCE` and ending with `TOTAL_COST`) is optional. When included, it should contain the following information:

`TRUCK_DISTANCE` (integer) is the total distance traveled by all trucks.
`NUMBER_OF_TRUCK_DAYS` (integer) is the total number of truck days used.
`NUMBER_OF_TRUCKS_USED` (integer) is the maximum number of trucks used on a single day.
`TECHNICIAN_DISTANCE` (integer) is the total distance traveled by all technicians.
`NUMBER_OF_TECHNICIAN_DAYS` (integer) is the total number of technician days used.
`NUMBER_OF_TECHNICIANS_USED` (integer) is the total number of technicians used.
`IDLE_MACHINE_COSTS` (integer) is the sum of the costs resulting from idle machines.
`TOTAL_COST` (integer) is the total cost of the solution.

This optional paragraph makes it easier to check the total cost of the solution. These results are also returned by the validator and hence can be checked by the user.

After these sections the actual solution is given, day by day. Every section starting with `DAY` first lists by the header `NUMBER_OF_TRUCKS` the number of trucks used this day, followed by the truck routes. Next, the header `NUMBER_OF_TECHNICIANS` indicates the number of technicians used, followed by the technician routes. All days of the planning horizon should be mentioned in the solution, even when no trucks or technicians are used during that day (such as days 1 and 7 in the example solution). Furthermore, for each day, the `NUMBER_OF_TRUCKS` and `NUMBER_OF_TECHNICIANS` parameters should always have a value, even when the value is 0 for that day.

To explain the format in which the routes should be given, let us consider day 3 in the example solution. On that day, one truck is used for delivering machines. The route for that truck is given as `1 6 7 0 1 2`.

- The first integer (`1`) refers to the ID of the truck. Recall that all trucks are identical, so we can simply number them 1, 2, 3, ...
- The remaining integers (`6 7 0 1 2`) form the sequence of request IDs the truck is delivering that day, where the integer `0` indicates a return to the depot. So in this example, truck `1` first delivers requests `6` and `7`, then returns to the depot to pickup new machines, and finally delivers requests `1` and `2`.

Note that all trucks depart from and must return to the depot at the end of the day, so this is not mentioned explicitly in the solution format.

The technician routes are given in a similar way: the first entry refers to the technician ID, the following entries refer to the sequence of the request IDs for the technician's route. All technician routes must begin and end at the technician's home location, which is therefore not mentioned explicitly in the solution format.

To conclude this section, we explain some of the parameter values in the example solution:

- `NUMBER_OF_TRUCK_DAYS` $= 0 + 1 + 1 + 0 + 1 + 0 + 0 = 3$, the result of summing up `NUMBER_OF_TRUCKS` over all days in the planning horizon.
- `NUMBER_OF_TRUCKS_USED` $= 1$, as this is the maximum number of trucks we use on any single day (or: the only truck we use to deliver all requests is the truck with ID 1).
- `NUMBER_OF_TECHNICIAN_DAYS` $= 0 + 0 + 1 + 3 + 0 + 1 + 0 = 5$.

- NUMBER_OF_TECHNICIANS_USED = 3, as the only technicians we use are those with IDs 1, 2 and 4.
- IDLE_MACHINE_COSTS = 400, which can be seen as follows. The only machines that are idle are those belonging to requests 3 (idle on day 3) and 6 (idle on days 4 and 5). As we see in the example instance, request 3 consists of two machines of type 4, for which the idle cost per day is 100. Hence, the total idle cost for request 3 is $1 \cdot 2 \cdot 100 = 200$. Request 6 consists of one machine of type 4, so the total idle cost for request 6 is $2 \cdot 1 \cdot 100 = 200$. This yields a total idle machine cost of 400.
- TOTAL_COST $= 394 \cdot 1 + 3 \cdot 100 + 1 \cdot 100.000 + 321 \cdot 1 + 5 \cdot 100.000 + 3 \cdot 100 + 400 = 610.706$, computed using the parameter values given in the second section of the example solution and the weights given in the example instance.

3 Rules

Please pay attention to the following rules.

General rules

Rule 1.1. This challenge seeks to encourage research into vehicle routing methods, and to offer prizes to the most successful methods. It is the spirit of these rules that is important, not the letter. With any set of rules for any challenge it is possible to work within the letter of the rules but outside the spirit. The organizers kindly ask the participants not to do this.

Rule 1.2. The organizers reserve the right to remove a participant from the challenge if the participant is determined by the organizers to have worked outside the spirit of the challenge rules. The organizers' decision is final in any matter. Decisions will be made by the organizers with the chair of VeRoLog having the casting vote in case of doubt.

Rule 1.3. The challenge consists of two parts: the first part is called the *all-time-best challenge* and the second part the *restricted resources challenge*. The rules for both parts are set out below. Participants may compete in one part or in both parts of the challenge.

Rule 1.4. Participants must submit their solutions —and, in the case of the restricted resources challenge, their programs— via the challenge website (https://verolog2019.ortec.com).

Rule 1.5. The rules set out in this document might be updated in due course. Any change of rules will be accompanied by a notification on the challenge website and a general email to all registered participants.

The all-time-best challenge

Rule 2.1. The main goal in this part of the challenge is to construct all-time-best solutions to the 25 instances in dataset "ORTEC-early". These instances are disclosed on the challenge website on Monday October 8, 2018. For this part of the challenge, there are no time or technology restrictions.

Rule 2.2. Solutions should be submitted via the challenge website (https://verolog2019.ortec.com). All submitted solutions are automatically validated, which may take a few seconds. For each instance, the website shows the cost of the current best solution for each participant.

Rule 2.3. The all-time-best challenge runs until 23:59 UTC on Sunday May 5, 2019.

Rule 2.4. For each instance, the team with the best solution at the deadline mentioned in Rule 2.3 is awarded €40. Additionally, for each instance and each of the 30 weeks between the start and the end of the all-time-best challenge, the front-runner for that instance at the end of the week (23:59 UTC on Sunday) is awarded €0,50. In case of ties the time that the solution was validated is decisive.

The restricted resources challenge

Rule 3.1. The instances for the restricted resources challenge are divided into two sets, "ORTEC-late" and "ORTEC-hidden". The hidden instances will be similar to the late instances. The late instances are made available on January 1, 2019, and the hidden instances after the upcoming VeRoLog conference in Seville, June 3-5, 2019.

Rule 3.2. In the restricted resources challenge the participants have to submit an implemented algorithm no later than 23:59 UTC on March 1, 2019. Any programming language can be used, as long as the submitted program can be run as described in the rules below.

Rule 3.3. The algorithm can be either deterministic or randomized. The participants that use a randomized algorithm should code their program in such a way that the exact run can be reproduced by specifying a random seed.

Rule 3.4. In the restricted resources challenge, the time limit T in seconds for an instance on your machine is given by

$$T = \lceil f \cdot (10 + R) \rceil$$

where f is the factor computed by your machine using the benchmark program on the challenge website (under `https://verolog2019.ortec.com/tools`), and R is the number of requests in the instance. For example, if the benchmark program returns the factor 0.8740 for your machine, then an instance with 750 requests should be solved by your program on your machine in $\lceil 0.8740 \cdot 760 \rceil = \lceil 664.24 \rceil = 665$ seconds.

Rule 3.5. The program should take as input an instance file in text format, and produce as output a solution in text format, all within the allowed time. The submitted program should accept four command-line arguments, in the following order: input file name, output file name, the calculation time in seconds as described in Rule 3.4 (integer), and a random seed (see Rule 3.6). For example:

```
>> my_solver.exe instance1.txt solution1.txt 665 18319894
>> python my_solver.py instance1.txt solution1.txt 665 18319894
```

Rule 3.6. Participants should run their program 9 times on each instance, each time with a different seed (see also Rule 3.10). The seeds are:

18319894
23390422
36197069
45945346
54500951
63196367
71110057
89578146
96527670

Note that the first digits of the seeds $(1, 2, \ldots, 9)$ can be used to differentiate between the different runs.

Rule 3.7. When calculating solutions to the late instances, the same version of the algorithm must be used for all instances. That is, the algorithm should not "know" which instance it is solving – while the algorithm may analyze the problem instance and set parameters accordingly, or may use the first digit of the seed to determine which strategy to use, it should not "recognize" the particular instance. See also Rule 1.1.

Rule 3.8. A submitted program should either be a Windows executable or a Python script, accompanied by any run-time dependencies required to run the program. Python packages that can be installed using Anaconda (e.g. `conda install <package>`) do not need to be included in the submission, as long as they are mentioned in the readme file mentioned in Rule 3.10.

Rule 3.9. Participants are allowed to make use of the solvers CPLEX, Gurobi and Xpress (see also Rule 3.14). Other third party software can only be used if it is freely available for all, even for commercial use. If any third party software is used, this should be specified in the readme file mentioned in Rule 3.10.

Rule 3.10. Each submission should consist of a single zip file. The zip file should contain:

- The program itself as described in Rule 3.8.
- A readme.txt file, with a brief description of the contents of the zip file and any usage of third party software (see Rule 3.9). In case of a Python script, the readme should contain the used Python version and a list of Python packages needed to run the program.
- Solutions to all the instances from "ORTEC-late" calculated with the program, for each of the 9 runs specified in Rule 3.6. So the total number of solutions in the zip file should be 9 times the number of instances in "ORTEC-late".

If a participant submit more than one zip file, only the last submitted zip file that was received before the deadline competes in the challenge.

Rule 3.11. Based on the submitted solutions, we calculate a rank per instance for each participant as follows. First, per instance, we remove the two best solutions and the two worst solutions. Hence, we are left with the middle 5 solutions on which we base the score for the participant. If these 5 solutions are all feasible, we take the average cost of those 5 solutions as score for the participant, and rank all participants accordingly. If the middle 5 solutions contain infeasible solutions, the corresponding participants are first ranked with respect to the *number* of feasible solutions, and then by the average cost of the feasible ones. Finally, we compute for each participant the mean rank, defined as the mean of all ranks.

Rule 3.12. Based on the mean ranks, the organizers select around 5 potential finalists for the second part of the restricted resources challenge. The results of the potential finalists are verified on the instances in "ORTEC-late" with the given random seeds on our machine (see Rule 3.14). The finalists are determined based on this verification.

Rule 3.13. The finalists' programs will be run by the organizers on the instances in "ORTEC-hidden", again with 9 random seeds whose first digits form the sequence $1, 2, \ldots, 9$ (not necessarily the same seeds as before) on our machine (see Rule 3.14). The same programs which were submitted for validation of the "ORTEC-late" instances will be used in the final. We compute the mean rank per finalist as described in Rule 3.11. The mean ranks will produce the final place list. The award for the winner of the restricted resources challenges is €2019. The runner-up is awarded €500, and the award for third place is €250.

Rule 3.14. The programs of the (potential) finalists will be run on a single core machine with a standard 64-bit Windows 10 operating system. Python versions 2.7 and 3.6 (for Windows) are pre-installed. Moreover, the solvers CPLEX, Gurobi and Xpress are pre-installed.

3.1 Ranking

In Rule 3.11 we explained how the participants are ranked in the restricted resources challenges. To illustrate this ranking method, suppose 7 participants solved 6 instances, and the table below contains for each participant and each instance the average cost of the middle 5 solutions (for the sake of simplicity, the average costs are assumed to be integer):

Instance	1	2	3	4	5	6
Participant A:	34	35	142	132	10	12
Participant B:	332	124	144	133	13	15
Participant C:	33	36	230	512	111	17
Participant D:	36	32	146	132	12	13
Participant E:	37	30	143	129	9	4
Participant F:	268	29	141	55	10	5
Participant G:	36	30	243	58	10	4

The ranks are as follows:

Instance	1	2	3	4	5	6
Participant A:	2	5	2	4.5	3	4
Participant B:	7	7	4	6	6	6
Participant C:	1	6	6	7	7	7
Participant D:	3.5	4	5	4.5	5	5
Participant E:	5	2.5	3	3	1	1.5
Participant F:	6	1	1	1	3	3
Participant G:	3.5	2.5	7	2	3	1.5

We now compute for each participant the mean of the ranks. The winner of the challenge is the participant with the lowest mean rank. In the example, the mean ranks are:

Mean rank
Participant A: 3.42
Participant B: 6
Participant C: 5.66
Participant D: 4.5
Participant E: 2.66
Participant F: 2.5
Participant G: 3.25

Hence participant F is the winner.

3.2 Awards

The awards are supplied by ORTEC. As mentioned in Rules 2.4 and 3.13, the awards are:

1. In the all-time-best challenge:
   - € 0,50 per instance per week for the instance's frontrunner at the end of the week;
   - € 40 per instance for the instance's frontrunner by the end of May 5, 2019.
2. In the restricted resources challenge the award for the winner is € 2019. The runner-up is awarded € 500, and the award for third place is € 250.

The awards will be presented at the upcoming VeRoLog conference in Seville, June 3-5, 2019 (https://verolog2019.sciencesconf.org/).