RoboCup Rescue Simulation League
2019 Agent Simulation Competition Rules

Luis G. Nardin, Sayed M. R. Modaresi, Dina Helal,
Kazunori Iwata, Okan Aşik and Farshid Faraji

Version 1.4

1 Introduction

Natural disasters are major adverse events that cause large-scale economic, human, and environmental losses. They are usually difficult to predict and even more challenging to prevent from happening. These characteristics demand disaster management strategies to be in place to reduce the negative impact of the disaster. Disaster management is usually divided in four phases:

- *Mitigation* includes the identification of hazards, the assessment of threats, and the implementation of measures to reduce potential losses and damages.
- *Preparedness* includes the elaboration of plans to deal with disasters and their consequences. This phase aims to have equipment and procedures ready for use when a disaster occurs.
- *Response* includes the search and rescue tasks executed during and just after a disaster.
- *Recovery* takes place after the occurrence of a disaster and involves the assistance of affected people and the restoration of the basic services.

Efficacious response plans are essential for reducing the negative impacts of natural disasters. Distributed coordination and planning are crucial for conducting the search and rescue tasks during this phase due to the usually non-existence of a centralized coordinator, involvement of multiple and heterogeneous entities, and lack of communication infrastructure.

Currently these hazardous tasks are performed by humans. However, the RoboCup Rescue Leagues aim to promote the use of artificial entities in performing these tasks.

The mission of the RoboCup Rescue Simulation League is to promote research and development in this socially significant domain at various levels. First, the league aims to promote the development of new computational artifacts that may improve the distributed coordination and planning among these heterogeneous artificial entities when performing search and rescue tasks on disaster scenarios. Second, it aims to provide a simulation software able to realistically represent natural disaster scenarios where these computational artifacts can be assessed. Finally, it aims to promote these research and development activities by conducting competitions to stimulate the exchange of ideas and experience among practitioners.

In particular, the RoboCup Rescue Agent Simulation competitions held during the RoboCup are mostly focused on the third objective. The competitions are

- Agent Simulation Competition (Section 2)
- Technical Challenge Competition (Section 3)

2 Agent Simulation Competition

The RoboCup Rescue Agent Simulation Competition involves primarily evaluating the performance of agent teams developed using the ADF Framework (`https://github.com/roborescue/rcrs-adf-core`, find a sample implementation at `https://github.com/roborescue/rcrs-adf-sample`) on different maps of the RoboCup Rescue Agent Simulation (RCRS) platform (`https://github.com/roborescue/`

rcrs-server). Specifically, it involves evaluating distributed coordination and planning algorithms enabling group of Ambulances, Police Forces, and Fire Brigades to rescue civilians and extinguish fires in cities where an earthquake has just occurred. Each team is evaluated on a set of scenarios that represent different disaster situations (e.g., civilians and fires, major fire in one corner of a city, blocked roads to refuges, damaged platoon agents, fire maps, civilian maps where only fires need to be extinguished or civilians need be saved).

This competition is divided in three rounds (i.e., preliminary, semi-final, and final rounds). At each round, the teams provide their code that will be executed on a set of scenarios and a score calculated to each scenario.

In addition to the teams performance on the scenarios, the teams are also evaluated based on a presentation of the strategy they implemented (see Section 2.2). The score obtained in this presentation counts as an additional scenario in all rounds.

At the end of each round, a set of teams with the highest sum of scores considering all the scenarios (including the presentation score) are selected to move to the next round of the competition.

The RoboCup Rescue Agent Simulation Technical Committee is responsible to elaborate the set of scenarios for the competition and all teams code are executed on the same set of scenarios.

2.1 Rules

(a) **Agents:** Teams shall implement all types of agents using the ADF Framework. Teams can only implement their own code to replace or extend the classes:

`adf.component.centralized.CommandExecutor`

`adf.component.centralized.CommandPicker`

`adf.component.communication.ChannelSubscriber`

`adf.component.communication.MessageCoordinator`

`adf.component.extraction.ExtAction`

`adf.component.module.algorithm.Clustering`

`adf.component.module.algorithm.PathPlanning`

`adf.component.module.complex.TargetAllocator`

`adf.component.module.complex.TargetDetector`

Teams are not allowed to change other classes, especially the Tactic classes. Teams should report any bug in these classes prior to the competition (see item (s) for comments on Bug Exploitation). It is the responsibility of the team to ensure that its code connects the correct number of agents to the server. It is the responsibility of the team to ensure that its code connects the correct number of agents to the server.

Teams must implement their code in a package named after their team's name (TEAM) and competition year (YEAR).

`$TEAM_YEAR.centralized.CommandExecutor`

`$TEAM_YEAR.centralized.CommandPicker`

`$TEAM_YEAR.communication.ChannelSubscriber`

`$TEAM_YEAR.communication.MessageCoordinator`

`$TEAM_YEAR.extraction.ExtAction`

`$TEAM_YEAR.algorithm.Clustering`

`$TEAM_YEAR.algorithm.PathPlanning`

`$TEAM_YEAR.complex.TargetAllocator`

`$TEAM_YEAR.complex.TargetDetector`

For instance, if the short name of the team is TEST and competition year is 2019, the team shall provide a package containing the methods:

`TEST_2019.centralized.CommandExecutor`

`TEST_2019.centralized.CommandPicker`

`TEST_2019.communication.ChannelSubscriber`

`TEST_2019.communication.MessageCoordinator`

`TEST_2019.extraction.ExtAction`

`TEST_2019.algorithm.Clustering`

`TEST_2019.algorithm.PathPlanning`

`TEST_2019.complex.TargetAllocator`

`TEST_2019.complex.TargetDetector`

Teams should provide a configuration file containing information of the classes they have changed from the original ADF Framework and a mapping between classes, package path, and file in their code.

(b) **Modularity:** Teams are forbidden to cast or make internal dependency between the classes listed in item (a). It is mandatory that these classes are usable independent of each other.

(c) **Code reuse:** The reuse of code of other teams from previous years is encouraged.

I. Teams are not allowed to have more than 50% of other team's code or logic for the classes
`adf.component.module.complex.TargetDetector`
`adf.component.module.complex.TargetAllocator`

II. Teams are allowed to reuse without any change the classes
`adf.component.centralized.CommandPicker`
`adf.component.centralized.CommandExecutor`
`adf.component.communication.ChannelSubscriber`
`adf.component.communication.MessageCoordinator`
`adf.component.extraction.ExtAction`
`adf.component.module.algorithm.Clustering`
`adf.component.module.algorithm.PathPlanning`

III. The reuse of any code or module from another team must be explicitly reported in a README document and released in the source-code and shared with the Organizing and Technical Committee members before the first day of competition. In the README document, the team must inform what motivated the reuse of the code or module and, if they made any changes (small or large) to the code or module, these changes must be described in detail.

IV. If the team used a class or module from another team without any change, the package name must not be changed.

The Organizing or Technical Committee members will check the teams' implementation with other teams' code from previous years to determine if the team complies with this rule. If the team does not comply, the team will be disqualified from the competition.

(d) **External libraries:** Teams are allowed to use external libraries, but these libraries must be open source and the libraries must not violate the competition rules. If any library is not open source and the team wants to use it in the competition, the team must submit its request for approval at least 1 (one) month before the beginning of the competition. The request must be submitted to the Technical and Organizing Committees.

(e) **Plagiarism:** If a team commits plagiarism, the team and its members will be banned participating on the current and next year's RoboCup Rescue Simulation competitions. The term plagiarism comprises any use of external knowledge without proper referencing, i.e., copying or using thoughts, ideas, texts or language in general and presenting them as their own. This applies for Team Description Papers as well as team code. All kinds of licenses and copyright have to be respected. This applies to the qualification process and the RoboCup Rescue tournaments. Please be aware that when a team is found guilty of committing plagiarism it is disqualified and banned at any time. This may also be in the middle of the tournament.

(f) **Shared memory:** Agents cannot use any form of shared memory, including static memory accessible to all agents, direct function calls between agents, or writing files for use by other agents during the scenario simulation. The exception is the Pre-Computation phase when agents are allowed to write files (see item (m) for details). The Organizing or Technical Committee may execute each agent of the team in a different virtual/physical machine if the team is suspected of violating this rule.

(g) **Remote participation:** Remote participation is not allowed in any circumstance. At least one team member must register and be locally present at the venue.

(h) **Rounds:** The competition is structured into three rounds: one preliminary round, one semifinal round, and one final round. The preliminary round will be executed in two consecutive days (first and second days of the competition), while the semifinal and final rounds are executed in one day each (third and fourth days of competition respectively).

(i) **Sessions:** Each round consists of several sessions. A session is comprised of a set of simulations in different scenarios. A member of the Organizing or Technical Committee will chair each session. The session chair is responsible for executing the simulations, collecting scores and logs, and handling any issues that arise during the session.

(j) **Code submission:** All teams must submit the team' source code (binary code will not be accepted) and adequate compiled scripts before the start of each round. The number and time of submissions and specific requirements will be explained during the competition setup time at the venue to the team leaders. The Organizing or Technical Committee has the authority to change the time of submissions and to audit all submitted teams' source-code.

(k) **Scenarios:** The scenarios will be provided by the Organizing or Technical Committee. Teams shall NOT know the disaster situation (i.e., map, random seeds, simulator configuration, parameter values, and phases of execution) of the session before it starts. All the conditions for a particular disaster situation will be identical for all the teams. A scenario is composed of a map, a set of rescue agents and civilians, and a set of configuration options for each of the simulator components.

(l) **Maps:** Each map is constrained to a maximum of $10,000$ roads and $10,000$ buildings. The building and road entrances are supposed to be fully connected. A validation tool will be used to check the full connectivity of roads and building entrances in each map. Teams do not have the right to complain in case road or building entrances are not fully connected if evidenced that this was not detected by the validation tool.

(m) **Phases:** The scenario simulation may be performed in two phases of execution of the team's code: the Pre-Computation phase (item (n)) and the Simulation phase (item (o)). The Pre-Computation phase, however, is not mandatory for all scenarios and is assumed a configuration parameter of the scenario. Thus, the execution of the Pre-Computation phase will be defined as a configuration parameter of the scenario (see item (k)).

(n) **Pre-computation phase:** The Pre-Computation phase allows an agent of each type to pre-process map-specific and scenario-specific data and store them into a file to use during the Simulation phase. Only one agent of each type can connect to the server and execute the Pre-Computation algorithm. This phase is limited to 2 minutes and after the time is elapsed the server will be terminated. Pre-Computation is allowed under the following conditions:

I. The data must be generated by a computer program with no human interaction or intervention.

II. Data for all maps must be generated by a single computer program.

III. The computer program should work for any new map.

IV. Agent must choose the file to store the pre-computing data.

V. Agents must be able to work if no Pre-Computation data is present for the map.

VI. The source-code of the Pre-Computation program must be released after the competition.

(o) **Simulation phase:** The Simulation phase corresponds to the actual team' simulation in the competition scenario. The team must connect in at most 3 minutes all agents to the kernel in order to perform the scenario simulation. The scenario simulation will begin no later than 3 minutes after the first agent begins its handshake with the kernel. All file permissions, except read permission for previously written files, will be removed.

(p) **Valid map:** The Organizing or Technical Committee is entitled to define whether a map results is valid or invalid in a session. The decision is based on the results of the map and it may be decided that a map is invalid when all the teams scores very close in the map.

(q) **Valid games:** Teams will NOT be entitled to rerun their agent team in most circumstances. It is expected that teams implement their agents so that they work correctly. In extreme circumstances teams may have the right of a single rerun. Circumstances that may result in a rerun are:

I. Power failure.
II. Accidental or deliberate termination of a kernel, simulator, or agent process.
III. Java Virtual machine crash.

In the case of rerun, the last score is used as the official score of the team on that map.

Examples of events that will NOT result in a rerun are:

I. Simulator crash.
II. Agents failing to fully connect before the simulation starts.
III. Agents crashing or failing to act during the run.
IV. Apparently incorrect behavior by a simulator or the viewer.
V. Simulator or ADF bug.

Teams that wish to request a rerun must do so in writing. The request must include the team's name, the scenario's name, the description of the problem, and the reasons why the team feels a rerun is appropriate. The request must also state whether the request is for a rerun of the team or a full session rerun. Only one Java Virtual Machine crash rerun request is accepted for each session.

(r) **Bugs:** It is the responsibility of each team to ensure that its code works correctly with the provided simulators. Although the Organizing or Technical Committee makes every effort to provide a reliable simulation environment, the Organizing or Technical Committee has no responsibility for any kind of software failure at competition time. Simulator and ADF Framework bugs are not sufficient grounds to request a rerun.

(s) **Bugs exploitation:** A team that exploit known bugs in the simulation package to gain advantage will be disqualified from the competition. Disqualifications will be made only after consultation with the RoboCup Trustees.

(t) **Team leader meetings:** Every day of the competition, there will be a team leader meeting before the beginning and after the end of the competition round to discuss issues or provide information about the competition. All team leaders of the teams participating in that day's round shall participate in these meetings, if the team leader fails to participate his/her issues and opinion will be disregarded.

(u) **Complains/Comments/Suggestions from teams: Only the team leader** of participating teams can complain, comment or make suggestions in writing to the Organizing or Technical Committee about the competition. Comments and suggestions of other team members will be disregarded. If these complains, comments or suggestions are deemed derogatory or abusive then the matter will be referred to the RoboCup Trustees and may result in penalties for the team concerned. Penalties may include points reduction or, in the worst case, disqualification.

(v) **Problem resolution:** If a problem arises during a session, **team leaders** may make a detailed written request for the session chair to resolve the problem. The session chair may make a decision on the spot or may refer it to the committee. Chair decisions are final, but if a team strongly disagrees, they may submit a written appeal to the committee. In order to allow the competition to continue, appeals will not be received during a round, but will be discussed by the committee at the end of each day. The Organizing or Technical Committee will make final decision at any condition.

(w) **Rule dispute resolution:** If there is an ambiguity in the rule or any unexpected situation happens, a temporary committee composed of the Technical, Organizing and Executive Committee members and the local chair have the power to take a decision regarding the issue. The temporary committee decision has the same effect as a rule.

(x) **Open source policy:** Source-code files must be released open-source immediately after the end of the competition to guarantee fair play and to encourage community activity after competition. Log files and related parameter files will be open access.

2.2 Presentation

The presentation aims to share the knowledge of the teams and improve the academic research aspects of the league. Each team will have 20 minutes to present their implementation and another 10 minutes for questions and answers. A presentation template is available at `https://github.com/roborescue/rcrs-templates/blob/master/presentation/presentation_template.pptx`.

The presentation will be evaluated by a panel of experts and the leader of the other teams. The final evaluation of the presentation will be incorporated into the score of the preliminary, semi-final and final rounds. The presentations will be evaluated according to a set of criteria:

- *Relevance* (5 points) Evaluates how relevant is the team's approach to the goals of the RoboCup Rescue. 0 means it is not relevant and 5 means it is very relevant.
- *Originality* (5 points) Evaluates how original is the proposed team's approach to RoboCup Rescue. 0 means it is not original and 5 means it is very original.
- *Significance* (5 points) Evaluates how significant to the league is the proposed team's approach. 0 means it is not significant and 5 means it is very significant.
- *Slide Content* (5 points) Evaluates the quality and completeness of the presentation material with respect to the team' strategy and the readability. 0 means that the presentation slides are of poor quality with respect to the content and 5 means that it is very informative and complete.
- *Slide Structure* (5 points) Evaluates the structure of the presentation material. 0 means that the presentation slides are poorly structured and 5 means that it is very well structured.
- *Talk* (5 points) Evaluates how much clear and easy to follow was the presentation and explanation, and whether the presenter had a positive attitude or not with respect to the presentation and the raised questions. 0 means the presentation is not clear or the presenter had a bad attitude and 5 means the presentation is clear and the presenter had good attitude.

Each team's presentation score will be calculated taking into account the sum of points given by the other teams excluding the X best and the X worst scores, where X is defined based on the number of competing teams during the first team leader's meeting, plus the points given by the 2 Committee members who are not member of any team participating on the RoboCup Rescue Simulation Agent Competition.

The score of each team will then be included in the ranking calculation in Section 2.3 as a scenario in all rounds that the team participates.

All teams will have the same number of evaluations and the same evaluators. In case some evaluator is not able to participate in the evaluation of all teams' presentation, his/her evaluation will be disregarded.

Team leaders or a representative assigned by the team leader must be present at all other teams presentation. The presentation session chair will check if the representative of each team is present at the beginning of each presentation, if there is no representative present the team's presentation score will be set to 0 (zero) and all the scores assigned by that team will be disregarded.

2.3 Ranking

Each round is composed of several sessions (S), and at each session the participating teams receive an identification ranging from t_1 to t_n, where n represents the number of teams participating on that session. Each session is comprised of a set of scenarios (M), and each scenario also receives an identification ranging from m_1 to m_p, where p represents the number of scenarios in that session. A score SC_{ji}^k is assigned to each team $i \in T$ ($T = \{t_1, \ldots, t_n\}$) at each session $k \in S$ ($S = \{s_1, \ldots, s_n\}$) for each scenario $j \in M$ ($M = \{m_1, \ldots, m_p\}$).

For each session k and scenario j, the Selective Minimum (SM_j^k) is calculated as

$$SM_j^k = max\left(SC_{ji}^k\right) - \left(\left(max\left(SC_{ji}^k\right) - mean\left(SC_{ji}^k\right)\right) \times 2\right), \tag{1}$$

and the Maximum Score (MS_j^k) is calculated as

$$MS_j^k = n \times SDC, \tag{2}$$

where n is the number of teams on session k, and SDC is the coefficient indicating the step between points among teams (we will use $SDC = 2$ in RoboCup Rescue 2019 competition).

The maximum value of each step is calculated as

$$MSS^k_{j/step\in\{1,\ldots,MS^k_j\}} = \frac{\left(\left(max\left(SC^k_j\right) - SM^k_j\right)\right)}{\left(MS^k_j \times \left(MS^k_j - step\right)\right)} \tag{3}$$

To each team is assigned the step value, whose $MSS^k_{j/step}$ value is lower than the team' score, but the $MSS^k_{j/step+1}$ value is greater than the team' score.

$$TP^k_{ji} = step \therefore MSS^k_{j/step} < SC^k_{ji} < MSS^k_{j/step+1} \tag{4}$$

The final team points for each scenario and team are calculated as

$$FTP^k_i = \sum_{j=m_1}^{m_p} TP^k_{ji} \tag{5}$$

The final team points is then used to generate a ranking of all the teams for that session. The team with the highest mean points is ranked as first, the second highest as second, and so on.

2.4 Hardware & Software Configuration

The competition will be run on a set of clusters composed of 4 computers each. One PC per cluster runs the simulator components (`Kernel Computer`); the remaining three runs the agent teams code (`Agent Computer`). The ideal hardware and software configuration for these computers are

Hardware

Each *Kernel Computer:* 1 x Intel processor (minimum 4 cores) and 8GB+ RAM
Each *Agent Computer:* 1 x Intel processor (minimum 4 cores) and 8GB+ RAM

Software

Operating System: Linux Ubuntu 16.04 LTS (64bit) or higher.

Java Virtual Machine: OpenJDK Java 8 or higher.

Simulator: The simulator package used in the Agent Simulation competitions are available at `https://github.com/roborescue/rcrs-server`.

Parameters: Possible range of parameter values and accessible parameters information are listed in Appendix A.

ADF: The ADF is available at `https://github.com/roborescue/rcrs-adf-code` and a sample code is available at `https://github.com/roborescue/rcrs-adf-sample`.

NOTE: These configurations are subject to change if the resources made available by the RoboCup Federation during the competition are different. If there is any change on these minimum requirements, the teams will be informed at the latest during the first leaders meeting prior to the competition.

3 Technical Challenge Competition

The Technical Challenge competition will explore the modularity of the teams implementation using the ADF Framework. The Technical Challenge will evaluate how easy it is to exchange the `TargetDetector` module for each type of entity among teams' code. The participation to the Technical Challenge will be compulsory. The teams do not need to send a separate Team Description Paper (TDP) to participate in this competition.

The Technical Challenge competition will be performed by exchanging the `TargetDetector` modules of the best three teams during the Preliminary round with all other teams. We will run each combination of teams' `TargetDetector` modules with the best three teams base code in some selected maps. The final score of the team on the specific map will be the worst score among the runs with the different base codes.

The RoboCup Rescue Agent Simulation chairs will check the modularity of the best three teams at the end of the Preliminary round using the Sample Agent. If the code is not modular, meaning that it depends on specific information not passed as reference, the team will be disqualified of the Agent Competition and the next best team in the Preliminary round will be evaluated.

The RoboCup Rescue Agent Simulation Organizing or Technical Committee is responsible to select the scenario for the competition and all teams code are executed on the same scenarios.

3.1 Rules

The Technical Challenges competition rules and the hardware configuration are the same of the Agent Simulation competition unless redefined below.

(a) **Rounds:** The competition is structured into one single round.

(b) **Number of base teams:** The number of best teams will be defined in the first day of competitions as this number depends on the number of participating teams.

3.2 Ranking

The competition is comprised of a set of scenarios (M). Each participating team receives an identification T ranging from t_1 to t_n, where n represents the number of teams participating on the competition. The best teams that provide the modules to replace in all teams receives an identification B ranging from b_1 to b_x, where x represents the number of teams which all other teams will use the modules from. Each scenario also receives an identification ranging from m_1 to m_p, where p represents the number of scenarios. A score SC^k_{ji} is assigned to each team $i \in T$ ($T = \{t_1, \ldots, t_n\}$) for each scenario $j \in M$ ($M = \{m_1, \ldots, m_p\}$) and $k \in B$ ($B = \{b_1, \ldots, b_x\}$).

The final team score is the mean among the worst scores in each scenario.

$$FTP_i = \frac{\sum_{j=1}^{p} arg\ min_{k \in B}(SC^k_{ji})}{p} \tag{6}$$

The final team scores are used to generate a ranking of all the teams. The team with the highest final score is ranked as first, the second highest as second, and so on.

4 Committees

The RoboCup Rescue Agent Competition is composed of three committees: Executive Committee (EC), Technical Committee (TC), and Organizing Committee (OC). Each committee is composed of 2 members and each member can participate in the same committee for at most 2 terms. Each term lasts for 3 consecutive years beginning at the RoboCup competition of the (re-)election year.

4.1 Current Committee Members

Role	Name	Country	Term	Period
EC	Farshid Faraji	Iran	2^o Term	2018–2021
EC	Luis G. Nardin	Germany	1^o Term	2017–2020
TC	Sayed M. R. Modaresi	Iran	2^o Term	2018–2021
TC	Dina Helal	Egypt	1^o Term	2017–2020
OC	Kazunori Iwata	Japan	1^o Term	2016–2019
OC	Okan Aşik	Turkey	1^o Term	2017-2020

4.2 Selection Process, Duties and Dismissal Rules

- **TC Selection** Selection to the TC will be held if there is a vacant position at the TC. A TC position becomes vacant if a TC member reaches the end of his/her term, a TC member is dismissed of his/her duties, or a TC member renounces his/her position.
- **OC Selection** Selection to the OC will be held if there is a vacant position at the OC. An OC position becomes vacant if an OC member reaches the end of his/her term, an OC member is dismissed of his/her duties, or an OC member renounces his/her position.
- **EC Selection** Selection to the EC will be held if there is a vacant position at the EC. An OC position becomes vacant if an EC member reaches the end of his/her term or an EC member renounces his/her position.

- **Nominating TC/OC candidates** Because there is a need to avoid concentration of TCs and OCs from the same region, we prioritize candidates of regions depending on the region where the current committee members reside. Currently, we prioritize candidates: 1^{st} North America, 2^{nd} Europe, and 3^{rd} other regions. To avoid such concentration, nomination of candidates from the same region of one current TC/OC member is not accepted. It is very important the TC/OC candidate are involved in the league and has experience on participating in RoboCup Rescue competitions.
- **Selection process for TC/OC** The selection process will be announced on the RoboCup competition venue in the year that there will be a vacant position in the TC or OC. The nomination deadlines will be published on the board, and personally and electronically to the team leaders. The selection will be done at the last official Team Leader meeting of the competition via a secret voting on the nominated candidates.
- **Selection process for EC** The selection process for a new EC member is conducted exclusively by the current EC members and it is conditional to the RoboCup Federation approval.
- **Duties** The TC/OC committee members have the responsibility of managing the league. Their specific tasks are
  - Elaborate RoboCup Rescue league road-map
  - Software improvement and maintenance
  - Update the web content
  - Update the rules of the competition
  - Organize the Qualification process
  - Negotiate with Local Organizing Committee
  - Manage competition schedule
  - Refereeing during the competition
- **Commitment** If elected to the TC/OC role, the nominated candidate shall commit to continue as member for at least 3 consecutive years. It is accepted that TC/OC members leave their role before the end of the term to assume another role in the RoboCup community, such as Executive or Technical Committee member.
- **Dismissed** TC/OC members that do not show commitment to the League and their responsibilities can be dismissed assuming that the majority of the other EC, TC, and OC members agree with the dismissal.

Acknowledgments

We thanks all the former RoboCup Rescue Simulation League members of the Technical, Organizing, and Executive Committees for their effort in promoting and organizing this competition. Special thanks to Nobuhiro Ito for his dedication to the RoboCup Rescue Simulation League.

A Parameters

The following tables describe the set of parameters that can change during the competition. Note that agents will not be given all the available parameters—see Table 1 for the set of parameters that agents are guaranteed to be able to query.

Table 1: Number of agents, refuges, and ignition points per scenario.

Entity	**Min**	**Max**
Fire brigade	0	50
Police force	0	50
Ambulance	0	50
Fire station	0	5
Police office	0	5
Ambulance centre	0	5
Civilians	0	1000
Refuges	0	Unlimited
Initial ignition points	0	50

The simulation parameters can vary between scenarios. Tables 2– 13 list all variable parameters and their possible ranges.

Table 2: Ranges for simulation parameters common to all components.

Parameter	**Description**	**Range**
random.seed	Seed for random number generator	Any range

Table 3: Ranges for kernel parameters.

Parameter	**Description**	**Range**
kernel.timesteps	Number of simulation timesteps	100—1000
kernel.agents.think-time	Number of milliseconds each agent has to send commands	500—3000

[1]Noise can be specified as input or output (or both). Input noise is applied as the agent sends a message to the server; output noise is applied as an agent receives a message. Thus, input noise is identical for all receivers but output noise is unique to each receiver. There are two types of noise: failure noise and dropout noise. Failure noise means a message disappears completely with no notification to either the sender or the receiver. Dropout noise removes the content of a message but the receiver still receives a zero-length communication from the sender, i.e., the sender knows a message was sent but the content is lost.

Table 4: Ranges for general communication channels configuration parameters.

Parameter comms.channels.*	**Description**	**Range**
count	Number of communication channels	1—20
max.platoon	Number of channels a platoon agent can subscribe to	0—10
max.centre	Number of channels a centre agent can subscribe to	0—20

Table 5: Ranges for voice channel parameters.

Parameter comms.channels.<x>.* <x> = channel number	**Description**	**Range**
type	Type of the channel	voice
range	Maximum range of a message in mm	0—300,000
message.size	Maximum size of a voice message in bytes	64—2,048
message.max	Maximum number of a voice message an agent can send per timestep	1—100

Table 6: Ranges for radio channel parameters.

Parameter comms.channels.<x>.* <x> = channel number	**Description**	**Range**
type	Type of the channel	radio
bandwidth	Maximum capacity of the channel in bytes per timestep	0—8,192

Table 7: Ranges for voice and radio channel parameters.[1]

Parameter comms.channels.<x>.* <x> = channel number	**Description**	**Range**
type	Type of the channel	radio
noise.[input | output].failure.use	Whether or not to use failure noise on channel <x> in input or output mode	yes or no
noise.[input | output].failure.p	Probability of message failure	0—1
noise.[input | output].dropout.use	Whether or not to use dropout noise on channel <x> in input or output mode	yes or no
noise.[input | output].dropout.p	Probability of message dropout	0—1

Table 8: Ranges for perception parameters.

Parameter	**Description**	**Range**
perception.los.max-distance	Maximum distance an agent can perceive world changes	30,000—150,000

Table 9: Ranges for perception parameters.

Parameter	Description	Range
collapse.wood.*	Proportion of wooden buildings with each degree of damage	0—1 Sum = 1
collapse.steel.*	Proportion of steel buildings with each degree of damage	0—1 Sum = 1
collapse.concrete.*	Proportion of concrete buildings with each degree of damage	0—1 Sum = 1

Table 10: Ranges for fire simulator parameters.

Parameter	Description	Range
fire.thank.maximum	Water tank capacity	5,000—150,000
fire.thank.refill-rate	Water tank refill rate	500—50,000
fire.extinguish.max-power	Maximum amount of water that can be used per timestep	500—150,000
fire.extinguish.max-distance	Maximal extinguish distance	10,000—150,000

Table 11: Ranges for misc simulator parameters.

Parameter **misc.*** **<type> = wood|steel|concrete**	**Description**	**Range**
buriedness.<type>.severity.rate	Probability that an agent in a collapse building of type <type> with a degree of collapse severity will be buried	0—1
buriedness.<type>.severity.value	Initial buriedness value for a buried agent in a collapsed building of type <type> with a degree of collapse severity	0—200
injury.collapse.<type>.severity.slight	Probability that an agent inside a collapsing building of type <type> with a degree of collapse severity will receive a slight injury	0—1
injury.collapse.<type>.severity.serious	Probability that an agent inside a collapsing building of type <type> with a degree of collapse severity will receive a serious injury	0—1
injury.collapse.<type>.severity.critical	Probability that an agent inside a collapsing building of type <type> with a degree of collapse severity will receive a critical injury	0—1
injury.collapse.slight	Amount of damage that a slight injury due to collapse causes	0—10,000
injury.collapse.serious	Amount of damage that a serious injury due to collapse causes	0—10,000
injury.collapse.critical	Amount of damage that a critical injury due to collapse causes	0—10,000
injury.collapse.multiplier.<type>	Damage multiplier for an agent of type <type> due to collapse	0—1

Table 12: Ranges for misc simulator parameters (continue).

Parameter **misc.*** **<type> = wood|steel|concrete**	**Description**	**Range**
injury.bury.<type>.severity.slight	Probability that an agent buried inside a building of type <type> with a degree of collapse severity will receive a slight injury	0—1
injury.bury.<type>.severity.serious	Probability that an agent buried inside a building of type <type> with a degree of collapse severity will receive a serious injury	0—1
injury.bury.<type>.severity.critical	Probability that an agent buried inside a building of type <type> with a degree of collapse severity will receive a critical injury	0—1
injury.bury.slight	Amount of damage that a slight injury due to buriedness causes	0—10,000
injury.bury.serious	Amount of damage that a serious injury due to buriedness causes	0—10,000
injury.bury.critical	Amount of damage that a critical injury due to buriedness causes	0—10,000
injury.bury.multiplier.<type>	Damage multiplier for an agent of type <type> due to buriedness	0—1
injury.fire.<type>.severity.slight	Probability that an agent inside a burning building of type <type> with a degree of collapse severity will receive a slight injury	0—1
injury.fire.<type>.severity.serious	Probability that an agent inside a burning building of type <type> with a degree of collapse severity will receive a serious injury	0—1

Table 13: Ranges for misc simulator parameters (continue).

Parameter **misc.*** **<type> = wood|steel|concrete**	**Description**	**Range**
injury.fire.<type>.severity.critical	Probability that an agent inside a burning building of type <type> with a degree of collapse severity will receive a critical injury	0—1
injury.fire.slight	Amount of damage that a slight injury due to fire causes	0—10,000
injury.fire.serious	Amount of damage that a serious injury due to fire causes	0—10,000
injury.fire.critical	Amount of damage that a critical injury due to fire causes	0—10,000
injury.fire.multiplier.<type>	Damage multiplier for an agent of type <type> due to fire	0—1
injury.<type>.k	k parameter for the damage progression function for injury type <type> (collapse, bury, fire)	0—1
injury.<type>.l	l parameter for the damage progression function for injury type <type> (collapse, bury, fire)	0—1
injury.<type>.noise.mean	Mean noise added to the damage progression function for injury type <type> (collapse, bury, fire)	0—1
injury.<type>.noise.sd	Standard deviation of noise added to the damage progression function for injury type <type> (collapse, bury, fire)	0—1

Table 14: Ranges for clear simulator parameters.

Parameter	**Description**	**Range**
clear.repair.rate	Rate of road clearing per police force agent in square m per timestep	0—50,000

Table 15: Ranges for ignition simulator parameters.

Parameter	**Description**	**Range**
ignition.random.lambda	Parameter of the Poisson distribution used to determine frequency of building ignition	0—1

Table 16: Parameters agents are guaranteed to be able to query.

Parameter	**Description**
kernel.agents.think-time	See Table 3
kernel.startup.connect-time	See Section 2.1—item (j)
comms.channels.count	See Table 4
comms.channels.<x>.type	See Table 5 and Table 6
comms.channels.<x>.range comms.channels.<x>.messages.size comms.channels.<x>.messages.max	See Table 5
comms.channels.<x>.bandwidth	See Table 6
fire.tank.maximum fire.tank.refill-rate fire.extinguish.max-power fire.extinguish.max-distance	See Table 10
clear.repair.rate	See Table 14
scenario.agents.fb	Number of Fire Brigades
scenario.agents.fs	Number of Fire Stations
scenario.agents.pf	Number of Police Forces
scenario.agents.po	Number of Police Offices
scenario.agents.at	Number of Ambulance Teams
scenario.agents.ac	Number of Ambulance Centres
kernel.communication-model	Communication model class name
kernel.perception	Perception model class name