RoboCup Rescue Agent Simulation League IranOpen 2018 Competition Rules and Setup

Farshid Faraji, Luis G. Nardin, Ali Modaresi, Dina Helal, Kazunori Iwata, and Nobuhiro Ito

Abstract

This document describes the rules and the ranking formula for the IranOpen 2018 RoboCup Rescue Agent Simulation League competitions.

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. Examples are evacuation and environmental impact assessment.

- *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 actions 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. These plans are usually elaborated by policy-makers based on previous experiences and with limited evaluation of efficacy. Assess response plans, however, is a challenging task as one cannot reproduce the real conditions of a natural disaster and its effects in a real environmental settings just for assessment purposes. In this case, simulation plays a key role in assisting to assess response plans by enabling the reproduction of the real conditions of a disaster in an artificial environmental setting.

The RoboCup Rescue Agent Simulation League mission is to promote research and development in this socially significant domain at various levels. The effective implementation of this mission is translated into three main league's goals. First, the league aims to provide a simulation software able to realistically represent natural disaster scenarios where response plans may be assessed. Second, it aims to define evaluation benchmarks for response plans that may be used by policy-makers. Finally, it aims to promote the research and development by conducting competitions to stimulate the exchange of ideas and experience among practitioners. This in turn may help in the development of more sophisticated and formalized plans to efficaciously respond to natural disasters.

The RoboCup Rescue Agent Simulation competition held during the RoboCup is focused mostly on the third objective. The competition is subdivided in the

- RoboCup Rescue Simulation Agent Competition (Section 2)
- Technical Challenge Competition (Section 3)
- Infrastructure Competition (Section **??**)

2 RoboCup Rescue Simulation Agent Competition

The RoboCup Rescue Simulation Agent Competition involves primarily evaluating the performance of agent teams developed using ADK Framework (`https://github.com/RCRS-ADF/RCRS-ADF`) on different maps of the RoboCup Rescue Agent Simulation (RCRS) platform (`https://sourceforge.net/projects/roborescue/files/roborescue/v1.2/`). Specifically, it involves evaluating coordination 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 respectively).

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.

A scenario is composed of a map, a set of initial rescue agent/civilian positions, and a set of configuration options for each of the simulator components. Each map is constrained to a maximum of 10,000 roads and 10,000 buildings. The buildings and roads 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. However, teams do not have the right to complain in case roads or building entrances are not fully connected if evidenced that this was not detected by the validation tool.

In addition to the agent's teams performance on the scenarios, the teams are also evaluated on a presentation of the strategy implemented. The score obtained in this presentation counts as an additional scenario for all the 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 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. The presentation will be evaluated by a panel of experts and the leader of the other teams. Best presentation award will be awarded.

2.2 Rules

(a) **Remote participation:** Remote participation is allowed for International teams. (Except for Iranian teams)

(b) **Rounds:** The competition is structured into three rounds: one preliminary round, one semifinal, and one final round. Each round will be executed in one day .

(c) **Sessions:** Each round consists of several sessions. A session is comprised of a set of simulations in different scenarios. A member of the

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.

(d) **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 in the brief presentation in the competition setup time at the venue. The Technical Committee has the authority to change the time of submissions and to audit all submitted teams' source-code.

(e) **Scenarios:** The scenarios will be provided by the Technical Committee. Teams shall NOT know the disaster situation (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 shall be identical for all the teams.

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

```
adf.component.extraction.ExtAction
adf.component.module.algorithm.Clustering
adf.component.module.algorithm.PathPlanning
adf.component.module.complex.TargetDetector
adf.component.module.complex.TargetAllocator
adf.component.centralized.CommandPicker
adf.component.centralized.CommandExecutor
```

Teams are not allowed to change other classes, especially the Tactic classes. 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 name.

```
$TEAM.extraction.ExtAction
$TEAM.algorithm.Clustering
$TEAM.algorithm.PathPlanning
$TEAM.complex.TargetDetector
$TEAM.complex.TargetAllocator
```

`$TEAM.centralized.CommandPicker`

`$TEAM.centralized.CommandExecutor`

For instance, if the short name of the team is TEST, the team should provide a package containing methods with

`TEST.extraction.ExtAction`

`TEST.algorithm.Clustering`

`TEST.algorithm.PathPlanning`

`TEST.complex.TargetDetector`

`TEST.complex.TargetAllocator`

`TEST.centralized.CommandPicker`

`TEST.centralized.CommandExecutor`

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

(g) **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 (h) for details). The Technical Committee may execute each agent of the team in a different virtual/physical machine if the team is suspected of violating this rule.

(h) **Phases:** The scenario simulation may be performed in two phases of execution of the team's code: the Pre-Computation phase (item (i)) and the Simulation phase (item (j)). 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 (e)).

(i) **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.

(j) **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.

(k) **Valid map:** The 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 score very close in the map.

(l) **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 name, the scenario name, a 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 just that team or for a full session rerun. Only one Java Virtual Machine crash rerun request is accepted for each session.

(m) **Bugs:** It is the responsibility of each team to ensure that its code works correctly with the provided simulators. Although the Technical Committee makes every effort to provide a reliable simulation environment, the 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.

(n) **Committee decisions:** If a problem arises during a session, teams may ask 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 heard during a round, but will be discussed by the committee at the end of each day. The Technical Committee will make final decision at any condition.

(o) **Comments from teams: Only the team leader** of participating teams can comment and make suggestions to the Technical Committee about the running of the competition. If these 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. Except tema leaders, other team members are not entitled to comment or make suggestions to the Technical Committee.

(p) **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.

(q) **Dispute resolution:** If there is an ambiguity in the rule and any unexpected situation happens, a temporary committee composed of the Technical, Organization 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.

(r) **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.

2.3 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^k_{ji} 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^k_j) is calculated as

$$SM^k_j = max\left(SC^k_{ji}\right) - \left(\left(max\left(SC^k_{ji}\right) - mean\left(SC^k_{ji}\right)\right) \times 2\right), \tag{1}$$

and the Maximum Score (MS^k_j) is calculated as

$$MS^k_j = n \times SDC, \tag{2}$$

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 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

$$FTP_i^k = \sum_{j=m_1}^{m_p} TP_{ji^k} \tag{5}$$

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

In the competition venue, there will be several clusters with 4 computers each. One PC per cluster will be reserved for the simulator components; the remaining three will be available to run the agent teams. The hardware and software configurations are

> *Hardware:* Intel i7 CPU processor 3.0GHz or higher, and 8.0 GB or higher RAM
>
> *Operating Systems:* Linux Ubuntu 16.04 LTS (64bit) or higher with Oracle Java 1.8 run-time or higher.
>
> *Simulation Packages:* The simulation packages used in the IranOpen 2018 RoboCup Rescue Simulation League competition 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 framework is available at `https://github.com/RCRS-ADF/RCRS-ADF`.

3 Technical Challenge Competition

The Technical Challenge competition will explore the modularity of the teams implementation using the ADK 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 from the best three teams during the Preliminary round with all other teams. We will run each combination of teams' code and the best three teams module in one randomly selected map. The final score of the team for the Technical Challenge will be the worst score among the runs with the different modules.

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 Technical Committee is responsible to select the scenario for the competition and all teams code are executed on the same scenario.

3.1 Rules

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

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

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

```
adf.component.extraction.ExtAction
adf.component.module.algorithm.Clustering
adf.component.module.algorithm.PathPlanning
adf.component.module.complex.TargetAllocator
adf.component.centralized.CommandPicker
adf.component.centralized.CommandExecutor
```

Teams are not allowed to change other classes, especially the Tactic classes. 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 name.

```
$TEAM.extraction.ExtAction
$TEAM.algorithm.Clustering
$TEAM.algorithm.PathPlanning
$TEAM.complex.TargetAllocator
```

```
$TEAM.centralized.CommandPicker
$TEAM.centralized.CommandExecutor
```

For instance, if the short name of the team is TEST, the team should provide a package containing methods with

```
TEST.extraction.ExtAction
TEST.algorithm.Clustering
TEST.algorithm.PathPlanning
TEST.complex.TargetAllocator
TEST.centralized.CommandPicker
TEST.centralized.CommandExecutor
```

Teams should be able to operate with other teams `TargetDetector` modules. Teams should provide a configuration file containing information of the classes they have changed from the original ADK Framework and a mapping between classes, package path, and file in their code.

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 teams will use the `TargetDectector` 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_{jik} 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.

$$FTP_i = \frac{\sum_{j=1}^{p} \min_{k \in B}(SC_{jik})}{p} \tag{6}$$

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.

3.3 Hardware & Software Configuration

One PC will be used to run the simulator components and the agent teams. The hardware and software configurations are

Hardware: Intel i7 CPU processor 3.0GHz or higher, and 8.0 GB or higher RAM

Operating Systems: Linux Ubuntu 16.04 LTS (64bit) or higher with Oracle Java 1.8 run-time or higher.

Simulation Packages: The simulation packages used in the IranOpen 2018 RoboCup Rescue Simulation League competition are available at `https://github.com/roborescue/rcrs-server`.

Parameters: The parameters and their possible values are the same as those in the 2014 Rules, which are listed in Section 8 of the document RoboCup 2014—RoboCup Rescue Simulation League Agent Competition (`https://sourceforge.net/projects/roborescue/files/competitions/2014/rsl14-rules.pdf`).

ADF: The framework is available at `https://github.com/RCRS-ADF/RCRS-ADF`.

Acknowledgments

We thanks all the former RoboCup Rescue Simulation League members of the Technical, Organization, and Executive Committees for their effort in promote and organize this competition.

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, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 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 2.2—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