AnyLogic Help > Experiment Framework > Optimization

Constraints and Requirements

AnyLogic supports *constraints* and *requirements* - additional restrictions imposed on the optimization parameters and solutions found by the optimization engine.

Constraints

A *constraint* is a condition defined upon optimization parameters. It defines a range for optimization parameters. Each time the optimization engine generates a new set of values for the optimization parameters, it checks whether these values satisfy the defined constraints. If they do, the engine runs the model with these values. If not, the engine generates another set of values, and so on. Thus the space of searching is reduced, and the optimization is performed faster.

A constraint is a well-formed arithmetic expression describing a relationship between the optimization parameters.

```
parameterA + parameterB + 2*parameterC = 10
parameterC - parameterA*parameterB >= 300
```

To define a constraint

1. Select the optimization experiment by clicking on it in the **Projects** view.
2. In the **Constraints** section of the **Properties** view, click the ➕ button to the right of the table **Constraints on simulation parameters (are tested before a simulation run)**.
3. Specify the constraint in the **Expression**, **Type** and **Bound** cells of the row. In the **Expression** field the top level agent is available as `root`.
4. Select the check box in the **Enabled** column. Otherwise the constraint is eliminated from optimization process and does not affect its results.

Requirements

A *requirement* is an additional restriction imposed on the solution found by the optimization engine. Requirements are checked at the end of each simulation, and if they are not met, the parameters used are rejected. Otherwise the parameters are accepted.

This means that if all variables meet their constraints at the end of the simulation, the corresponding set of parameter values is considered feasible, and the result of the simulation is accepted by the optimization engine. Otherwise, parameter values are considered infeasible, and the result is rejected.

A requirement can also be a restriction on a response that requires its value to fall within a specified range. Constraint may contain any variables of the agent, the model time symbol `t`, any arithmetic operations and method calls, such as, e.g., `sin()`, `cos()`, `sqrt()`, etc., or calls to your own methods.

```
0 <= 2*root.variable - root.statistics.max() <= 500
sqrt(variableC)>=49
```

To define a requirement

1. Select the optimization experiment by clicking it in the **Projects** view.
2. In the **Requirements** section of the **Properties** view, click the ➕ button to the right of the table **Requirements (are tested after a simulation run to determine whether the solution is feasible)**.
3. Specify the requirement in the **Expression**, **Type** and **Bound** cells of the row. In the **Expression** field the top level agent is available as `root`.
4. Select the check box in the **Enabled** column. Otherwise the requirement is eliminated from optimization process and does not affect its results.

Feasible and infeasible solutions

A feasible solution is one that satisfies all constraints and requirements.

The optimization engine makes finding a feasible solution its highest priority. Once it has found a feasible solution, it concentrates on finding better solutions.

The fact that a particular solution may be infeasible does not imply that the problem itself is infeasible. However, infeasible problems do exist. Here is an example:

```
parameterA + parameterB <= 4
parameterA + parameterB >= 5
```

Clearly, there is no combination that will satisfy both of these constraints.

If a model is constraint-feasible, the optimization engine will always find a feasible solution and search for the optimal solution (i.e., the best solution that satisfies all constraints).

If the optimization engine cannot find any feasible solutions, you should check the constraints for feasibility and fix the inconsistencies of the relationships modeled by the constraints.