Saving and Restoring AnyLogic 6 Model Snapshot

We are pleased to announce a new feature available in AnyLogic 6 Professional Edition: now you are able to save the full state of a model (the snapshot) during runtime to a file, restore it at a later time and continue running simulation from the same point.

This feature may be useful in several cases:

- Resilience: when a simulation takes very long time to complete, it may make sense to save its state periodically so that you do not have to start everything from scratch when e.g. the computer crashes.
- Skipping warm-up period: if you plan to run several scenarios with a simulation that differ only in what happens to the model after it warms up, you may run the model up to the end of its warm-up period only once, save the state, and then reload it for every scenario.
- Running distributed simulations: many parallel/distributed simulation frameworks require the ability to roll-back the model to some previous state (checkpoint). This may be needed to synchronize the clocks of concurrent simulations when one of them happens to "run too far".
- Any other case when you need to refer to a particular state of the model without running the simulation from the initial state.

AnyLogic model snapshot implementation is based on Java serialization mechanism.

Saving and restoring the snapshot manually using the toolbar

To save a snapshot of a running model manually using the toolbar buttons you first need to bring the model to the paused state (press *Pause*). In the *File* section of the running model toolbar you will find two new buttons: *Save Snapshot* and *Restore Saved Snapshot*. Press *Save Snapshot* and choose the file name and location. The model state is saved. You can now continue the simulation, stop, close it, or even shut down your computer.

To restore the saved model state you first need to run the same experiment (you do not need to actually run the simulation, it is sufficient just to launch the experiment). When the experiment is in the idle or paused state, press *Restore Saved Snapshot* and choose the snapshot file. The model state saved in that file will be restored and the model will be put in the paused state.

Saving and restoring the snapshot using the API

The same functionality is provided by two API methods of class `Presentation`:

- `saveSnapshot( String filename )` – pauses the experiment if it is currently running, saves the snapshot to a given file and resumes experiment (if it was running)
- `loadSnapshot( String filename )` – stops the experiment and loads the snapshot, but does not run it. If an error occurs rolls back to the current experiment and resumes it if it was running. When the snapshot is loaded, presentation completely forgets the model, which was running before (including the engine and active objects), therefore, it is recommended not to keep references to model objects after this method call

Both methods return immediately (operations are performed in a separate thread). When the method returns this doesn't mean that the operation is complete – it may be still in progress. In order to monitor events of snapshot actual loading/saving (and perform any actions), listeners can

be defined (see `addSnapshotListener` and `removeSnapshotListener` methods of `Presentation` class).

Starting experiments from a saved snapshot

Any of AnyLogic experiments can use a pre-saved model snapshot to start the simulation. Using this feature you can save a lot of time when performing the experiments with multiple runs (e.g. optimization, compare runs, or Monte Carlo) where *all runs have the same warm-up period*.

Consider, for example, the pedestrian dynamics model where people are exiting a large stadium. It make take a long time to wait for the people to full up the stadium exits, but the actual subject of simulation/optimization (e.g. the space layout between the stadium and the subway station) may come into play only after that. It makes a lot of sense to simulate once up to the point when the people start exiting the stadium, save the snapshot, and reuse that snapshot when trying out different layouts.

To tell the experiment to start each run from a saved snapshot, open the “Advanced” page of the Property View, check the checkbox “Load root object from snapshot” and specify the file name. The same functionality is available via the API: the method `setLoadRootFromSnapshot( String filename )` is included in all experiments.

Having loaded the snapshot, the experiment will set the parameters for each individual run using the `set_`... methods.

Please note that, for performance purposes the snapshot file is cached in the memory, so you may need to raise the Maximum available memory setting of the experiment.

AnyLogic snapshot file format

AnyLogic snapshot file has extension .als (AnyLogic Snapshot) and is a binary file. Its header contains the serialization version number that is used to mark compatible and incompatible changes.

Performance and resource requirements

For a (fairly large) Bass Diffusion Agent Based model with 1,000,000 of agents that occupies about 300M memory saving a snapshot takes approximately 2 minutes, same time is required to load a snapshot. The snapshot file size for that model is 38M.

Saving and restoring the model snapshot raises the memory requirements by 2 to 3 times, depending on the model size (this extra memory is needed only during saving and restoring).

How to ensure your model is serializable

For 99% of models you do not have to do anything to enable their serialization, i.e. saving their state at runtime: all AnyLogic objects are serializable, including all standard library objects. However, there are cases when the user needs to specify explicitly how the objects should be serialized and deserialized. This is described in AnyLogic documentation.

Restrictions

Open database connections and open files cannot save and restore their states. Therefore after restoring the snapshot the connections and files will be in closed state. This does not mean however you cannot continue accessing them. For example, if the model writes to a kind of a log file, it can continue writing there, provided the file is in “append” mode.

At the moment OptQuest optimization engine cannot save and restore its state, therefore you cannot save snapshots of Optimization experiments. In the future however OptTek will release a serializable version of OptQuest.