An OO Process Approach to ARGESIM Comparison C4 Dining Philosophers with AnyLogic

M. Gyimesi, F. Breitenecker, TU Vienna
mgyimesi@osiris.tuwien.ac.at

Simulator. AnyLogic (www.xjtek.com) is an object – orientated, general-purpose simulator for discrete but, continuous and hybrid applications. It supports modelling with UML – RT and the underlying modelling technology is based on Java so that building simulation models using AnyLogic should be easy for experienced programmers.

Model: The implementation of the Dining Philosopher model is made by defining three classes in a very easy way. The classes are: Philosopher, Chopstick and Table. The Table - class defines the global parameters and the interaction of the encapsulated Philosopher and Chopstick classes:

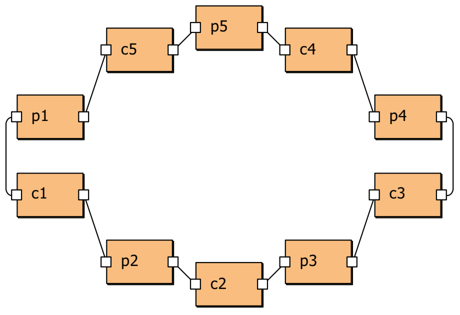

Fig. 1: Model layout of the Table class

The behaviour of the Philosopher and the Chopstick classes is defined by using state charts. Depending on the tasks (see below) the state charts differ slightly.

Experiments: The most interesting part of the Dining Philosopher problem is the occurrence and further avoidance of deadlocks. Therefore different strategies were used for seizing the chopstick:

1. First left then right chopstick
2. Asymmetry: One philosopher takes first right and then left chopstick
3. Monitoring the number of philosopher waiting for the right chopstick. If four philosophers are already waiting, the fifth philosopher can not seize the left chopstick

Remarks on implementation: Only the first (classical) strategy can cause a deadlock. AnyLogic handles simultaneous events by randomly choosing one of the scheduled events.

To get comparable results, a duration of 1.000.000 virtual time steps where taken for all experiments and the same distributions (thinking time: discrete uniform distribution [1,4], eating time: discrete uniform distribution [1,3]) where used for all strategies. The observation variables are number of deadlocks and number of time steps till a deadlock (for strategy a), mean thinking time, mean waiting time and mean eating time for one philosopher.

Results

Strategy a: This strategy is designed to generate deadlocks. AnyLogic has automatic deadlock detection. When a deadlock occurs, a new replication of the simulation will be started till the end of overall simulation time is reached. Figure 2 shows the distribution of performed simulation time per simulation run.

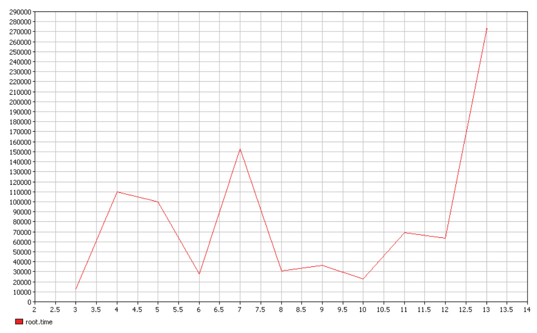

Fig. 2: Time steps per simulation run

Min:	11523.0
Max:	318134.0
Mean:	89124.21428571429
Variance:	9.450074562950548E9

Strategies b and c: Both strategies avoid deadlock situations. But as we see in the table below, the strategy operating with one asymmetric philosopher is more efficient (meaning, that the waiting time = starving time is the lowest - even more efficient than Strategy a.

	a)	b)	c)
thinking time	304684,49	413726,10	55579,58
waiting time	451490,62	255194,20	799993,12
eating time	243824,89	331081,78	35567,89

Table 1: Mean time values for one philosopher

C4 Classification: Object-oriented Process Approach

Simulator: AnyLogic 4.5