A C++ - based Object-oriented Solution of ARGESIM Comparison “C10 – Dining Philosophers II” using CSIM

Herb Schwetman, Mesquite Software Inc.;
Sabri Pllana, TU Vienna;
hds@mesquite.com / spllana@osiris.tuwien.ac.at

Simulator: CSIM (Mesquite Software) is a process-oriented, general-purpose simulation toolkit, which supports the development of process-oriented, discrete-event simulation models, by using the standard programming languages C and C++. Because of the nature of compiled C and C++ programs and CSIM's dynamic memory allocation, developed models are compact and efficient. CSIM supports the Object-Oriented Simulation (OOS).

Model: The OOS models the system by specifying the behaviour of objects during the time. Each philosopher in our model is an object of the class *Philosopher*, which is represented in Figure 1, by using the Unified Modelling Language (UML) notation.

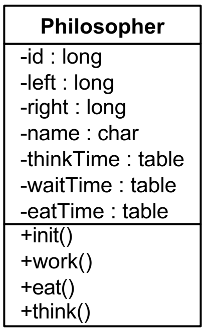
Philosopher
-id : long
-left : long
-right : long
-name : char
-thinkTime : table
-waitTime : table
-eatTime : table
+init()
+work()
+eat()
+think()

Figure 1, left: The UML representation of the class *Philosopher*

Figure 2 depicts the method *work()* of the class Philosopher, which implements the behaviour of the philosopher (its thinking, waiting and eating activities). Each chopstick is an object of the CSIM class *Facility*.

```
1   void Philosopher::work(void) {
2     create (name); // create the philosopher process
3     double waitBegin; // begin of the waiting time
4     double waitEnd; // end of the waiting time
5     while (1) {
6       this->think(); // thinking time
7       waitBegin = simtime(); // begin waiting
8       set_priority(2); // high priority
9       (*cs)[left].reserve(); // get the left chopstick
10      hold(1); // a pause after grabbing the left
11      set_priority(1); // low priority
12      (*cs)[right].reserve(); // get the right chopstick
13      waitEnd = simtime(); // end of the w. time
14      waitTime.record(waitEnd - waitBegin); // w.t.
15      this->eat(); // eating time
16      (*cs)[right].release(); // release the right
17      (*cs)[left].release(); // release the left
18    } // end while
19  };
```

Figure 2: The `work()` method of the class *Philosopher*

Task a: Simulation until deadlock. Table 1 shows simulation results for the case in which the deadlock is reached at simulation time 1116359. The objects of the CSIM class *Table* are used to collect explicit statistics on thinking, waiting, and eating times.

	Thinking time		Waiting time		Eating time	
	mean	s.d.	mean	s.d.	mean	s.d.
P1	5.50	2.87	11.46	7.74	5.49	2.87
P2	5.49	2.87	11.41	7.74	5.51	2.88
P3	5.52	2.88	11.47	7.77	5.49	2.87
P4	5.51	2.87	11.44	7.74	5.52	2.88
P5	5.51	2.86	11.46	7.73	5.51	2.87
All	5.51	2.87	11.45	7.74	5.50	2.87

Table 1: Average times (with standard deviation) of thinking, waiting, and eating periods for each philosopher

Statistics on usage of CSIM *facilities*, which are used to model chopsticks, are collected automatically by CSIM (see Table 2).

Chopstick utilization [%]					
C1	C2	C3	C4	C5	All
0.91	0.91	0.91	0.91	0.91	0.91

Table 2: Chopstick utilization

Task b: Simultaneous access. When a philosopher executes *(*cs)[right]reserve()* (see Fig. 2, line 9) it either gets the chopstick immediately (if the chopstick is free), or it is placed in the queue of philosophers waiting for the chopstick.

The default queue discipline is FCFS (First Come, First Served). If priorities differ, then the philosopher with a higher priority gets the chopstick first. In our model the philosopher on the right has a higher priority (see Fig. 2, line 8).

Task c: 50 simulation runs. Basically, our model allows an arbitrary number of philosophers. 50 simulation runs were executed by the command **`dph -np 5 -nr 50`**, where **`-np`** specifies the number of the philosophers, and **`-nr`** the number of the simulation runs.

The maximum and the minimum termination simulation times were 21915300 and 80177 units. The deadlock is detected by using the CSIM built-in event **`event_list_empty`**, which indicates that all philosophers are waiting. 50 simulation runs took 42 minutes of the wall clock time. Simulation is executed on Sun Ultra 10 workstation (CPU 440MHz, memory 256MB).

C10 Classification: Object/Process – oriented Approach

Simulator: CSIM Rel. of 2002