Chapter 8
Simulation Scenarios

Table 8.1 outlines the seventeen simulated scenarios run in our work. The goal of these simulations was to simulate the model proposed in Sect. 8.2. These simulations cover the four classes of job profiles discussed in Sect 1.2.

The first group of simulations covered the 2-user scenario, in which two users compete for access to system resources. The number of users in the simulation was then expanded to 6, 11, and 21. The total number of resources was held constant at 100 across the simulations (see Sect. 8.14 for a summary of assumptions) (Fig. 8.1).

8.1 Simulation Case 1–2 Users Base Case

This two-user, trivial case demonstrates the use of the simulator and its configuration. The total load (1000 tasks per user) is submitted at the beginning of the simulation.

$$L_1(x, PC) = \max\{1000 * L_0(x, PC) \cos(\pi * PC), 0\} \tag{8.1a}$$

$$L_2(x, PC) = \max\left\{1000 * L_0(x, PC) \sin\left(\pi * PC + \frac{3\pi}{2}\right), 0\right\} \tag{8.1b}$$

$$L_0(x, PC) = 1 \tag{8.2}$$

$$\sum_{PC=0}^{n} L_1(PC) = \sum_{PC=0}^{n} L_2(PC) = 1000 \tag{8.3}$$

A. Sedighi, M. Smith, *Fair Scheduling in High Performance Computing Environments*, https://doi.org/10.1007/978-3-030-14568-2_8

Table 8.1 Simulation scenarios covered by this research

Simulation number	Number of users	Job profile	Job profile description
1	1	$L_1(x, PC) = \max\{200^* L_0(x, PC)\cos(\pi^* PC), 0\},\ \sum_{PC=0}^{n} L_1(PC) = 1000$	8.1
		$L_2(x, PC) = \max\{200^* L_0(x, PC)\sin(\pi^* PC + 3\pi/2), 0\},\ \sum_{PC=0}^{n} L_1(PC) = 1000$	
2	2	$L_1(x, PC) = \max\{1000^* L_0(x, PC)\cos(\pi^* PC), 0\},\ \sum_{PC=0}^{n} L_1(PC) = 5000$	8.2
		$L_2(x, PC) = \max\{500^* L_0(x, PC)\sin(\pi^* PC + 3\pi/2), 0\},\ \sum_{PC=0}^{n} L_2(PC) = 5000$	
3	2	$L_1(x, PC) = 100^* L_0(x, PC),\ \sum_{PC=0}^{n} L_1(PC) = 5000$	8.3
		$L_2(x, PC) = \max\{200^* L_0(x, PC)\cos(\pi^* PC), 0\},\ \sum_{(PC=0)}^{n} L_2(PC) = 5000$	
4	2	$L_1(x, PC) = \max\{5000^* L_0(x, PC)\cos(\pi^* PC), 0\},\ \sum_{PC=0}^{n} L_1(PC) = 5000$	8.4
		$L_2(x, PC) = \max\{50^* L_0(x, PC)\sin(\pi^* PC + 3\pi/2), 0\},\ \sum_{PC=0}^{n} L_2(PC) = 50$	
5	2	$L_1(x, PC) = \max\{5000^* L_0(x, PC)\cos(\pi^* PC), 0\},\ \sum_{PC=0}^{n} L_1(PC) = 5000$	8.5
		$L_2(x, PC) = \max\{L_0(x, PC)\cos(\pi^* PC), 0\},\ \sum_{PC=0}^{n} L_2(PC) = 10$	
6	6	$L_1(x, PC) = \max\{200^* L_0(x, PC)\sin(\pi^* PC), 0\}, \sum_{PC=0}^{n} L_1(PC) = 1000 L_1(x, PC) = L_2(x, PC) = L_3(x, PC) = L_4(x, PC) = L_5(x, PC)$	8.6
		$L_6(x, PC) = \max\{500^* L_0(x, PC)\sin(\pi^* PC + 3\pi/2), 0\},\ \sum_{PC=0}^{n} L_2(PC) = 5000$	

7	6	$\mathrm{L}_1(\mathrm{x},\mathrm{PC}) = 20^*\ \mathrm{L}_0(\mathrm{x},\mathrm{PC}),\ \sum_{PC=0}^{n} L_1(PC) = 1000\ \mathrm{L}_1(\mathrm{x},\mathrm{PC}) = \mathrm{L}_2(\mathrm{x},\mathrm{PC}) = \mathrm{L}_3(\mathrm{x},\mathrm{PC}) = \mathrm{L}_4(\mathrm{x},\mathrm{PC}) = \mathrm{L}_5(\mathrm{x},\mathrm{PC})$	8.7
		$\mathrm{L}_6(\mathrm{x},\mathrm{PC}) = \max\{200^*\ \mathrm{L}_0(\mathrm{x},\mathrm{PC})\cos(\pi^*\ \mathrm{PC}), 0\},\ \sum_{PC=0}^{n} L_6(PC) = 5000$	
8	6	$\mathrm{L}_1(\mathrm{x},\mathrm{PC}) = \max\{5000^*\ \mathrm{L}_0(\mathrm{x},\mathrm{PC})\cos(\pi^*\ \mathrm{PC}), 0\}, \sum_{PC=0}^{n} L_1(PC) = 1000\mathrm{L}_1(\mathrm{x},\mathrm{PC}) = \mathrm{L}_2(\mathrm{x},\mathrm{PC}) = \mathrm{L}_3(\mathrm{x},\mathrm{PC}) = \mathrm{L}_4(\mathrm{x},\mathrm{PC}) = \mathrm{L}_5(\mathrm{x},\mathrm{PC})$	8.8
		$\mathrm{L}_6(\mathrm{x},\mathrm{PC}) = \max\{50^*\ \mathrm{L}_0(\mathrm{x},\mathrm{PC})\sin(\pi^*\ \mathrm{PC} + 3\pi/2), 0\},\ \sum_{PC=0}^{n} L_6(PC) = 50$	
9	6	$\mathrm{L}_1(\mathrm{x},\mathrm{PC}) = \max\{1000^*\ \mathrm{L}_0(\mathrm{x},\mathrm{PC})\cos(\pi^*\ \mathrm{PC}), 0\}, \sum_{PC=0}^{n} L_1(PC) = 1000\mathrm{L}_1(\mathrm{x},\mathrm{PC}) = \mathrm{L}_2(\mathrm{x},\mathrm{PC}) = \mathrm{L}_3(\mathrm{x},\mathrm{PC}) = \mathrm{L}_4(\mathrm{x},\mathrm{PC}) = \mathrm{L}_5(\mathrm{x},\mathrm{PC})$	8.9
		$\mathrm{L}_6(\mathrm{x},\mathrm{PC}) = \max\{\ \mathrm{L}_0(\mathrm{x},\mathrm{PC})\cos(\pi^*\ \mathrm{PC}), 0\},\ \sum_{PC=0}^{n} L_6(PC) = 10$	
10	11	$\mathrm{L}_1(\mathrm{x},\mathrm{PC}) = \max\{100^*\ \mathrm{L}_0(\mathrm{x},\mathrm{PC})\cos(\pi^*\ \mathrm{PC}), 0\},\ \sum_{\mathrm{PC}=0}^{\mathrm{n}} \mathrm{L}_1(\mathrm{PC}) = 500\mathrm{L}_1(\mathrm{x},\mathrm{PC}) = \ldots = \mathrm{L}_{10}(\mathrm{x},\mathrm{PC})$	8.10
		$\mathrm{L}_{11}(\mathrm{x},\mathrm{PC}) = \max\{500^*\ \mathrm{L}_0(\mathrm{x},\mathrm{PC})\sin(\pi^*\ \mathrm{PC} + 3\pi/2), 0\},\ \sum_{\mathrm{PC}=0}^{\mathrm{n}} \mathrm{L}_{11}(\mathrm{PC}) = 5000$	
11	11	$\mathrm{L}_1(\mathrm{x},\mathrm{PC}) = 10^*\ \mathrm{L}_0(\mathrm{x},\mathrm{PC}),\ \sum_{PC=0}^{n} L_1(PC) = 500\mathrm{L}_1(\mathrm{x},\mathrm{PC}) = \ldots = \mathrm{L}_{10}(\mathrm{x},\mathrm{PC})$	8.11
		$\mathrm{L}_{11}(\mathrm{x},\mathrm{PC}) = \max\{200^*\ \mathrm{L}_0(\mathrm{x},\mathrm{PC})\cos(\pi^*\ \mathrm{PC}), 0\},\ \sum_{PC=0}^{n} L_{11}(PC) = 5000$	
12	11	$\mathrm{L}_1(\mathrm{x},\mathrm{PC}) = \max\{500^*\ \mathrm{L}_0(\mathrm{x},\mathrm{PC})\cos(\pi^*\ \mathrm{PC}), 0\},\ \sum_{PC=0}^{n} L_1(PC) = 500\mathrm{L}_1(\mathrm{x},\mathrm{PC}) = \ldots = \mathrm{L}_{10}(\mathrm{x},\mathrm{PC})$	8.12
		$\mathrm{L}_{11}(\mathrm{x},\mathrm{PC}) = \max\{50^*\ \mathrm{L}_0(\mathrm{x},\mathrm{PC})\sin(\pi^*\ \mathrm{PC} + 3\pi/2), 0\},\ \sum_{PC=0}^{n} L_{11}(PC) = 50$	

(continued)

Table 8.1 (continued)

Simulation number	Number of users	Job profile	Job profile description
13	11	$L_1(x,PC) = \max\{500^* \; L_0(x,PC)\cos(\pi^* \; PC), 0\}, \sum_{PC=0}^{n} L_1(PC) = 500L_1(x,PC) = \ldots = L_{10}(x,PC)$	8.13
		$L_{11}(x,PC) = \max\{L_0(x,PC)\cos(\pi^* \; PC), 0\}, \sum_{PC=0}^{n} L_{11}(PC) = 10$	
14	21	$L_1(x,PC) = \max\{50^* \; L_0(x,PC)\cos(\pi^* \; PC), 0\}, \sum_{PC=0}^{n} L_1(PC) = 250L_1(x,PC) = \ldots = L_{20}(x,PC)$	8.10
		$L_{21}(x,PC) = \max\{500^* \; L_0(x,PC)\sin(\pi^* \; PC + 3\pi/2), 0\}, \sum_{PC=0}^{n} L_{21}(PC) = 5000$	
15	21	$L_1(x,PC) = 10^* \; L_0(x,PC), \sum_{PC=0}^{n} L_1(PC) = 250 \, L_1(x,PC) = \ldots = L_{20}(x,PC)$	8.11
		$L_{21}(x,PC) = \max\{200^* \; L_0(x,PC)\cos(\pi^* \; PC), 0\}, \sum_{PC=0}^{n} L_{21}(PC) = 5000$	
16	21	$L_1(x,PC) = \max\{250^* \; L_0(x,PC)\cos(\pi^* \; PC), 0\}, \sum_{PC=0}^{n} L_1(PC) = 250L_1(x,PC) = \ldots = L_{20}(x,PC)$	8.12
		$L_{21}(x,PC) = \max\{50^* \; L_0(x,PC)\sin(\pi^* \; PC + 3\pi/2), 0\}, \sum_{PC=0}^{n} L_{21}(PC) = 50$	
17	21	$L_1(x,PC) = \max\{250^* \; L_0(x,PC)\cos(\pi^* \; PC), 0\}, \sum_{PC=0}^{n} L_1(PC) = 250L_1(x,PC) = \ldots = L_{20}(x,PC)$	8.13
		$L_{21}(x,PC) = \max\{L_0(x,PC)\sin(\pi^* \; PC), 0\}, \sum_{PC=0}^{n} L_{21}(PC) = 10$	

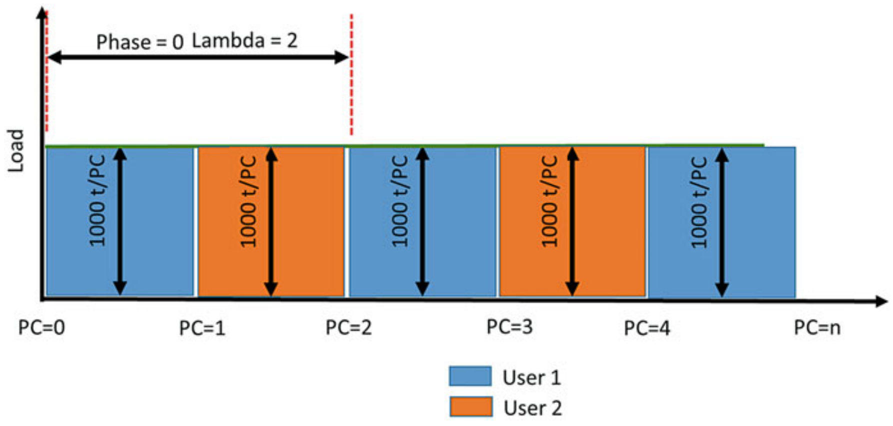

Fig. 8.1 Depiction of simulation case 1

8.2 Simulation Case 2–2 Users

In this two-user case, one user submits a sine load, and the other user submits a cosine load. The sine-load[1] user submits tasks in a start–stop fashion. The cosine-load user does so as well, but with a phase shift that complements the sine-load. In the first case, c_1 submits a workload that is complementary but has a higher rate than that of c_2 by 100%. The configuration for the simulator in this case is as follows:

```
(1) simulator.setNumberOfSubmitters(2);
(2) simulation.setWavelength("2:2");
(3) simulation.setPhase("0:1");
(4) simulation.setAmplitude("1000:500");
(5) simulation.setTaskCount("5000:5000");
```

We create a setup with two submitters (1), each of which has a wavelength of 2 (2). The sine-load user submits a load at PC zero and then at even PCs. The cosine-load user submits a load at PC 1 and then at odd PCs (3). The actual loads are 1000 tasks/PC for c_1 and 500 tasks/PC for c_2 (4). Both users send the same number of tasks (5000) over the duration of the simulation (5) (Fig. 8.2).

[1]An un-shifted sine wave can go negative, which does not make sense in this context. For this reason, we only consider the positive when we talk about sine and cosine: $\max\left\{L_1(x, PC)\sin\left(\pi^{*}PC + \frac{3\pi}{2}\right), 0\right\}$

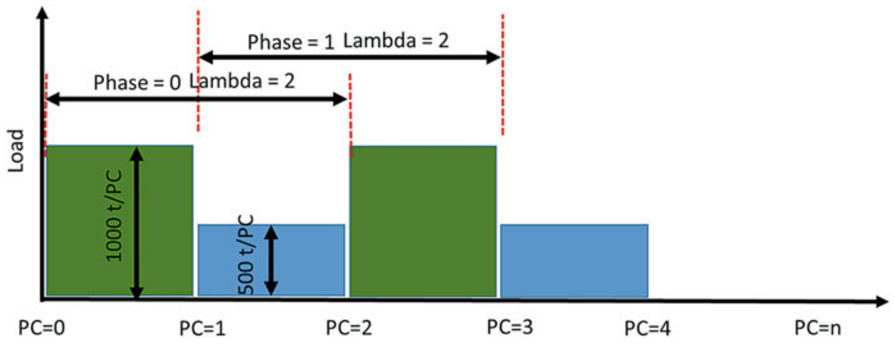

Fig. 8.2 Depiction of simulation case 2

8.3 Simulation Case 3–2 Users

In this two-user case, one user submits a sine load and the other user submits a constant load. For a given lambda, the total number of tasks submitted by each user is the same. Furthermore, the total number of tasks submitted by each user during the simulation is the same: 5000 each.

The API level configuration is as follows:

```
    :
(1) simulator.setNumberOfSubmitters(2);
(2) simulation.setWavelength("0:2");
(3) simulation.setPhase("0:0");
(4) simulation.setAmplitude("100:200");
(5) simulation.setTaskCount("5000:5000");
    :
```

8.4 Simulation Case 4–2 Users

In this two-user case, one user first submits a large, one-time workload of 5000 tasks. Then, a second user submits a one-time workload of 50 tasks, a mere fraction of the total number of tasks submitted by the first user. Interestingly, a small workload can be overwhelmed in this type of case. Configuring the simulator for this simulation is accomplished by using the API as follows:

```
    :
(1) simulator.setNumberOfSubmitters(2);
(2) simulation.setWavelength("2:2");
(3) simulation.setPhase("0:1");
(4) simulation.setAmplitude("5000:50");
(5) simulation.setTaskCount("5000:50");
    :
```

8.5 Simulation Case 5–2 Users

This case is similar to case 4, but the second user submits only 1 task/PC until each of their 10 tasks have been submitted. The API level configuration is as follows:

```
    :
(1) simulator.setNumberOfSubmitters(2);
(2) simulation.setWavelength("2:2");
(3) simulation.setPhase("0:1");
(4) simulation.setAmplitude("5000:1");
(5) simulation.setTaskCount("5000:10");
    :
```

8.6 Simulation Case 6–6 Users

This six-user case resembles case 2, but 5 users each submit a sine load in equal parts and one user submits a cosine load. All of the users submit their tasks in a start–stop fashion. $c_1 = c_2 = c_3 = c_4 = c_5$, but c_6 submits a complimentary workload at a higher rate. The configuration is as follows:

```
(1) simulator.setNumberOfSubmitters(6);
(2) simulation.setWavelength("2:2:2:2:2:2");
(3) simulation.setPhase("0:0:0:0:0:1");
(4) simulation.setAmplitude("200:200:200:200:200:500");
(5) simulation.setTaskCount("1000:1000:1000:1000:1000:5000");
```

The setup includes six submitters (1), each with a wavelength of 2 (2). The sine-load user submits a load at PC zero and then at even PCs. The cosine-load user submits a load at PC 1 and then at odd PCs (3). For $c_1 . . c_5$, the actual load is 200 tasks/PC up to 1000 tasks, and for c_6, the actual load is 500 tasks/PC up to 5000 tasks (4–5).

8.7 Simulation Case 7–6 Users

This six-user case resembles case 3, but 5 users each submit a constant load in equal parts and one user submits a sine load. For a given lambda, the total number of tasks submitted by the 5 constant-load users is the same as the total number of tasks submitted by the sine-load user.

The API level configuration is as follows:

```
     :
(1) simulator.setNumberOfSubmitters(6);
(2) simulation.setWavelength("0:0:0:0:0:2");
(3) simulation.setPhase("0:0:0:0:0:1");
(4) simulation.setAmplitude("20:20:20:20:20:200");
(5) simulation.setTaskCount("5000:5000:5000:5000:5000:5000");
     :
```

8.8 Simulation Case 8–6 Users

In this six-user case, 5 users submit large, one-time workloads of 1000 tasks. Then, one user submits a one-time workload of 50 tasks, a fraction of the total number of tasks submitted by the other users. This case resembles case 4, but with 6 users. The goal in this case is to analyze the multi-user case scenario of our workload. This was accomplished by using the API to configure the simulator as follows:

```
     :
(1) simulator.setNumberOfSubmitters(6);
(2) simulation.setWavelength("2:2:2:2:2:2:2");
(3) simulation.setPhase("0:0:0:0:0:0:1");
(4) simulation.setAmplitude("1000:1000:1000:1000:1000:50");
(5) simulation.setTaskCount("1000:1000:1000:1000:1000:50");
     :
```

8.9 Simulation Case 9–6 Users

This is similar to case 5, but with 6 users. 5 users each send 1000 tasks, while one user sends 1 task/PC. The API level configuration is as follows:

```
    :
(1) simulator.setNumberOfSubmitters(6);
(2) simulation.setWavelength("2:2:2:2:2:2");
(3) simulation.setPhase("0:0:0:0:0:1");
(4) simulation.setAmplitude("1000:1000:1000:1000:1000:1");
(5) simulation.setTaskCount("1000:1000:1000:1000:1000:10");
    :
```

8.10 Simulation Cases 10 and 14

Cases 10 and 14 are similar to case 6, but they feature 11 and 21 users, respectively.

8.11 Simulation Cases 11 and 15

Cases 11 and 15 are similar to case 7, but they feature 11 and 21 users, respectively.

8.12 Simulation Cases 12 and 16

Cases 12 and 16 are similar to case 8, but they feature 11 and 21 users, respectively.

8.13 Simulation Cases 13 and 17

Cases 13 and 17 are similar to case 9, but they feature 11 and 21 users, respectively.

8.14 Assumptions

While various assumptions are made in the aforementioned cases, each supports the overall goal of assessing the feasibility of a new scheduling model that uses three system parameters to reduce time-in-system. Due to the complexity of the task at hand, a number of assumptions are made. These assumptions are enumerated in this section.

8.14.1 No-Randomness Assumption

The simulation runs feature no randomness: a given input *always* led to the same output, and no random parameters or variables appeared anywhere in the simulation. This is required as the interest is in achieving predictable improvements in performance and not statistical significance across the various results. The simulation and results were predictable and repeatable in every case. The applicability of the Rawlsian Fair scheduler hypothesis is tested in each simulation. For each simulation, (a) the submission profiles are varied, (b) the number of users are varied, and (c) the method used to calculate seniority are varied. As the interest is to test the hypothesis under specific conditions, the specificity of the scenario will prove to be more applicable in a controlled environment and without any randomness as to the submission profile or number of users.

8.14.2 Task-Based-Workload Assumption

In the cases considered, a given job is composed of many requests or tasks, each of which is self-contained and can be executed independently. In addition, resources are allocated dynamically. Due to the nature of the interactive workloads that run in HPC environments, the resources allocated to a given user can change with every scheduling cycle. We also assume that the tasks are malleable (Feitelson et al. 1997): they are able to take advantage of new resources allocated during the scheduling cycle.

8.14.3 Resource-Constraint Assumption

The number of resources is held constant at 100 CPUs in all of the simulations. Although the number of resources is largely irrelevant, it had to be smaller than the workload and to remain constant. The number of resources had to be smaller than the workload because if it was not, users would not have been forced to compete for

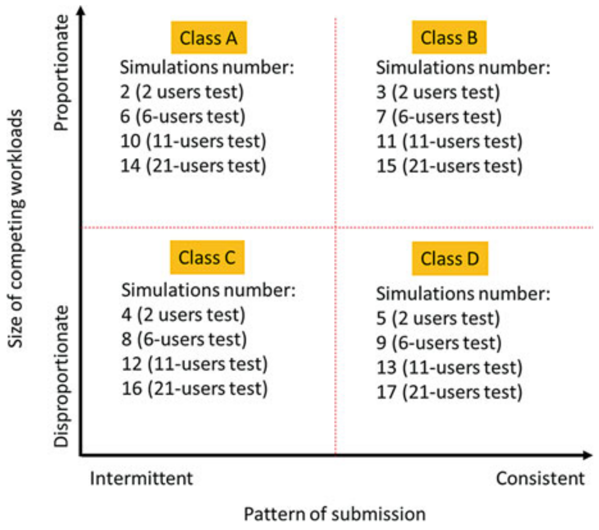

Fig. 8.3 Classification of the 16 simulation scenarios

resources and the simulation would not have been representative of real-world cases. The number of resources was held constant to reduce the number of variables.

8.14.4 Simulation Parameters and Configuration

17 simulation cases were run, each with seven scheduling algorithms: traditional fair-share and Rawlsian Fair with 6 different BH values. Simulation 1 was a "smoke-test" to assess the basic functionality of the simulator, but the other 16 tests were broken into four different classes of simulations, as is described in Sect. 1.2. Figure 8.3 maps the simulation runs to the four classes of job profiles. Below, an overview is provided of all of the simulations and the expectations for each class of simulation.

8.14.4.1 Simulations for Class-A: Simulations 2, 6, 10 and 14

Simulations 2, 6, 10, and 14 were conducted to determine the applicability of the Rawlsian Fair algorithm vis-à-vis Class A workloads. In these simulations, the overall load was held constant at 10,000. The parameters of the simulation keeps one user's submission load profile constant while distributing other loads among the rest of the users. The goal of these tests was to determine whether Rawlsian Fair is *as good as* fair-share scheduling. Rawlsian Fair was expected to introduce no addition delays, and the simulations validated this assumption. Since fair-share scheduling seeks long-term fairness (Kleban and Clearwater 2003), the primary objective of these tests was to make sure that Rawlsian Fair does not generate more delays than fair-share.

8.14.4.2 Simulation Class-B: Simulations 3, 7, 11, and 15

Simulations 3, 7, 11, and 15 were conducted to determine the applicability of the Rawlsian Fair algorithm vis-à-vis Class B workloads. In each of these simulations, one or more of the workloads submitted is a constant workload. In each case, the constant workload(s) competes with intermittent workloads. Since a constant workload submits a constant number of tasks, in 2-user simulations, the results achieved by the Rawlsian Fair algorithm were expected to be *as good as* those achieved by fair-share algorithm. As the number of users increased, the Rawlsian Fair algorithm was expected to yield results that were increasingly better than those of fair-share scheduling.

8.14.4.3 Simulation Class-C: Simulations 4, 8, 12, and 16

Simulations 4, 8, 12, and 16 were conducted to determine the applicability of the Rawlsian Fair algorithm vis-à-vis Class C workloads. In each simulation, Rawlsian Fair scheduling was expected to reduce the delay experienced by the smallest user(s).

8.14.4.4 Simulation Class-D: Simulations 5, 9, 13, and 17

Simulations 5, 9, 13, and 17 were conducted to determine the applicability of the Rawlsian Fair algorithm vis-à-vis Class D workloads. In each simulation, Rawlsian Fair scheduling was expected to reduce the delay experienced by the smallest user(s). As the number of users increased from 2 to 21, the delay experienced by the smallest user(s) was expected to remain constant.