Q1:

**Compute the solutions for simulations with 3 jobs and random seeds of 1, 2, and 3.**

Seed 1----

*ARG jlist*

*ARG jobs 3*

*ARG maxlen 10*

*ARG maxticket 100*

*ARG quantum 1*

*ARG seed 1*

*Here is the job list, with the run time of each job:*

*Job 0 ( length = 1, tickets = 84 )*

*Job 1 ( length = 7, tickets = 25 )*

*Job 2 ( length = 4, tickets = 44 )*

*Here is the set of random numbers you will need (at most):*

Random 651593

Random 788724

Random 93859

Random 28347

Random 835765

Random 432767

Random 762280

Random 2106

Random 445387

Random 721540

Random 228762

Random 945271

Answer:

Total tickets = 84 + 25 + 44 = 153

Random 651593 % 153 = 119

Count = 0 + 84(job0) = 84 < 119

Count = 84 + 25(job1) = 109 < 119

Count = 109 + 44(job2) = 153 > 119 (winner)

Run job 2

Job2(3/4)

Random 788724 % 153 = 9

Count = 0 + 84(job0) = 84 > 9 (winner)

Run job0

Job0(0/1) done

Random 93859 % (25+44) = 19

Count = 0 + 25(job1) = 25 > 19 (winner)

Run job1

Job1(6/7)

Random 28347 % 69 = 57

Count = 0 + 25(job1) = 25 < 57

Count = 25 + 44(job2) = 69 > 19(winner)

Run job2

Job2(2/3)

Random 835765 -> Winning ticket 37 (of 69) -> Run 2

Job2(1/4)

Random 432767 -> Winning ticket 68 (of 69) -> Run 2

Job2(0/4) done

Random 762280 -> Winning ticket 5 (of 25) -> Run 1

Job1(5/7)

Random 2106 -> Winning ticket 6 (of 25) -> Run 1

Job1(4/7)

Random 445387 -> Winning ticket 12 (of 25) -> Run 1

Job1(3/7)

721540 -> Winning ticket 15 (of 25) -> Run 1

Job1(2/7)

Random 228762 -> Winning ticket 12 (of 25) -> Run 1

Job1(1/7)

Random 945271 -> Winning ticket 21 (of 25) -> Run 1

Job1(0/7) done

All job done

Seed 2----

ARG jlist

ARG jobs 3

ARG maxlen 10

ARG maxticket 100

ARG quantum 1

ARG seed 2

Here is the job list, with the run time of each job:

Job 0 ( length = 9, tickets = 94 )

Job 1 ( length = 8, tickets = 73 )

Job 2 ( length = 6, tickets = 30 )

Here is the set of random numbers you will need (at most):

Random 605944

Random 606802

Random 581204

Random 158383

Random 430670

Random 393532

Random 723012

Random 994820

Random 949396

Random 544177

Random 444854

Random 268241

Random 35924

Random 27444

Random 464894

Random 318465

Random 380015

Random 891790

Random 525753

Random 560510

Random 236123

Random 23858

Random 325143

Random 605944 -> Winning ticket 169 (of 197) -> Run 2

Job0(9/9) job1(8/8) Job2(5/6)

Random 606802 -> Winning ticket 42 (of 197) -> Run 0

Job0(8/9) job1(8/8) Job2(5/6)

Random 581204 -> Winning ticket 54 (of 197) -> Run 0

Job0(7/9) job1(8/8) Job2(5/6)

Random 158383 -> Winning ticket 192 (of 197) -> Run 2

Job0(7/9) job1(8/8) Job2(4/6)

Random 430670 -> Winning ticket 28 (of 197) -> Run 0

Job0(6/9) job1(8/8) Job2(4/6)

Random 393532 -> Winning ticket 123 (of 197) -> Run 1

Job0(6/9) job1(7/8) Job2(4/6)

Random 723012 -> Winning ticket 22 (of 197) -> Run 0

Job0(5/9) job1(7/8) Job2(4/6)

Random 994820 -> Winning ticket 167 (of 197) -> Run 2

Random 949396 -> Winning ticket 53 (of 197) -> Run 0

Random 544177 -> Winning ticket 63 (of 197) -> Run 0

Random 444854 -> Winning ticket 28 (of 197) -> Run 0

Random 268241 -> Winning ticket 124 (of 197) -> Run 1

Random 35924 -> Winning ticket 70 (of 197) -> Run 0

Random 27444 -> Winning ticket 61 (of 197) -> Run 0

--> JOB 0 DONE at time 14

Random 464894 -> Winning ticket 55 (of 103) -> Run 1

Random 318465 -> Winning ticket 92 (of 103) -> Run 2

Random 380015 -> Winning ticket 48 (of 103) -> Run 1

Random 891790 -> Winning ticket 16 (of 103) -> Run 1

Random 525753 -> Winning ticket 41 (of 103) -> Run 1

Random 560510 -> Winning ticket 87 (of 103) -> Run 2

Random 236123 -> Winning ticket 47 (of 103) -> Run 1

Random 23858 -> Winning ticket 65 (of 103) -> Run 1

--> JOB 1 DONE at time 22

Random 325143 -> Winning ticket 3 (of 30) -> Run 2

--> JOB 2 DONE at time 23

Seed 2----

ARG jlist

ARG jobs 3

ARG maxlen 10

ARG maxticket 100

ARG quantum 1

ARG seed 3

Here is the job list, with the run time of each job:

Job 0 ( length = 2, tickets = 54 )

Job 1 ( length = 3, tickets = 60 )

Job 2 ( length = 6, tickets = 6 )

Here is the set of random numbers you will need (at most):

Random 13168

Random 837469

Random 259354

Random 234331

Random 995645

Random 470263

Random 836462

Random 476353

Random 639068

Random 150616

Random 634861

Random 13168 -> Winning ticket 88 (of 120) -> Run 1

Random 837469 -> Winning ticket 109 (of 120) -> Run 1

Random 259354 -> Winning ticket 34 (of 120) -> Run 0

Random 234331 -> Winning ticket 91 (of 120) -> Run 1

--> JOB 1 DONE at time 4

Random 995645 -> Winning ticket 5 (of 60) -> Run 0

--> JOB 0 DONE at time 5

Random 470263 -> Winning ticket 1 (of 6) -> Run 2

Random 836462 -> Winning ticket 2 (of 6) -> Run 2

Random 476353 -> Winning ticket 1 (of 6) -> Run 2

Random 639068 -> Winning ticket 2 (of 6) -> Run 2

Random 150616 -> Winning ticket 4 (of 6) -> Run 2

Random 634861 -> Winning ticket 1 (of 6) -> Run 2

--> JOB 2 DONE at time 11

Q2:

**Now run with two specific jobs: each of length 10, but one (job 0)with just 1 ticket and the other (job 1) with 100 (e.g., -l 10:1,10:100). What happens when the number of tickets is so imbalanced? Will job 0 ever run before job 1 completes? How often? In general, what does such a ticket imbalance do to the behavior of lottery scheduling?**

Command: **./lottery.py -l 10:1,10:100 -c**

The scheduler will always pick job 1 to execute the, after job 1 finish the job 0 will start. Highly unlikely the job 0 would every run, the chance 1/101 around 0.99% which is less than 1%.

This ticket imbalance will starve the less tickets job especially in short amount of time. For example, if the length of two job is around 1000, then the job 0 may 10 times to run in 1000 length.

Q3:

**When running with two jobs of length 100 and equal ticket allocations of 100 (-l 100:100,100:100), how unfair is the scheduler? Run with some different random seeds to determine the (probabilistic) answer; let unfairness be determined by how much earlier one job finishes than the other.**

Command: **./lottery.py -s 1 -l 100:100,100:100 -c**

Job 1 done at time 196, job 0 done at 200. Thus, job 1 is 4ms earlier

Command: **./lottery.py -s 2 -l 100:100,100:100 -c**

Job 1 done at time 190, job 0 done at 200. Thus, job 1 is 10ms earlier

Command: **./lottery.py -s 10 -l 100:100,100:100 -c**

Job 1 done at time 196, job 0 done at 200. Thus, job 1 is 10ms earlier

Command: **./lottery.py -s 100 -l 100:100,100:100 -c**

Job 1 done at time 185, job 0 done at 200. Thus, job 1 is 15ms earlier

Command: **./lottery.py -s 1000 -l 100:100,100:100 -c**

Job 1 done at time 200, job 0 done at 195. Thus, job 0 is 5ms earlier

Now lets sum up number of the all the errors then divide the number of he runes we got:

Base on above 5 simulations the average unfairness is 8.8 ms. But if we run 10000 times or we increase the job length by 10000 fold, the by Probability theory the average unfairness score will converge to 1.

**Q4:**

**How does your answer to the previous question change as the quantum size (-q) gets larger?**

**Command: ./lottery.py -s 1 -l 100:100,100:100 -q 10 -c**

Job 0 done at time 200, job1 done at 160. Notice the when we increase the length of time slice by 10 fold, the unfairness is also increase 10 fold. This make sense, previous our time slice only 1 thus, and if there are 4 instants of unfairness happen, the unfairness point only be 4. But when the time lice become 10, and four time unfair happen, the unfairness point will be 40.

**Q5**

**Can you make a version of the graph that is found in the chapter? What else would be worth exploring? How would the graph look with a stride scheduler?**

**Chart, line chart

Description automatically generated**

**Notice when the job length get larger and larger the fairness will extremely close to 1.**

**Stride graph below**

**Chart

Description automatically generated**

**Notice, the stride at begin will ascent shapely, then will become 1. Unlike lottery scheduler, the will close to 1 but never equal to 1.**