Lottery

jobs   seed

-j 2 -s 1

J0 ⇒ len = 1, tickets = 84

J1 ⇒ len = 7, tickets = 25

Random 495435 % total no. of tickets

= 495435 % 109 = 30 ⇒ J0 (0, 83)

* J0 (len = 1) J1 (len = 7)

1 J0 (len = 0) - finished J1 (len = 7) → remaining total tickets

449491 % 25 = 16 ⇒ J1 (0, 24)

* J1 (len = 7)

2 J1 (len = 6)

651593 % 25 = 18 → J1

* J1 (len = 6)

3 J1 (len = 5)

788724 % 25 = 24 ⇒ J1

* J1 (len = 5)

4 J1 (len = 4)

93859 % 25 = 9 ⇒ J1

5 J1 (len=3)

28347 % 25 = 22 ⇒ J1

6 J1 (len=2)

835765 % 25 = 15 ⇒ J1

7 J1 (len=1)

432767 % 25 = 17 ⇒ J1

8 _ J1 (len=0)

① -j 3 -s 1

J0 ⇒ len = 1, tickets = 84
J1 ⇒ len = 7, tickets = 25
J2 ⇒ len = 4, tickets = 44

651593 % 153 = 119 ⇒ *J2 (109, 152)

1 J2 (len = 3)

788724 % 153 = 9 ⇒ *J0 (0, 83)

2 J0 (len=0) finished
Updated remaining tickets total = 69

93859 % 69 = 19 ⇒ *J1 (0, 24)

3 ·J1 (len=6)

28347 % 69 = 57 ⇒ *J2 (24,68)

4 J2(len = 2)

835765 % 69 = 37 ⇒ *J2

5 J2 (len = 1)

432767 % 69 = 68 ⇒ *J2

6 J2 (len = 0) finished

Now just J1 remains, hence
will get CPU always

-j 3 -s 2 , -j 3 -s 1 similar

② -l 10:1, 10:100

When so imbalanced tickets, the

job with lower will mostly not get a chance to run

In this case that probability is

1/10

J0 doesn't run before J1 with the given random numbers (seed 0)

J0 is run after J1 finishes

With seed 2, J0 run once before J1 finished

3. -l 100, 100 : 100, 100

With seed 0

J0 finishes at 192

J1 " " 200

Unfairness = 200 - 192 = 8

With seed 1

J0 finishes at 200

J1 " " 196

Unfairness = 200 - 196 = 4

With seed 2

J0 finishes at 200

J1 " " 190

Unfairness = 200 − 190 = 10

With seed 3

J0 finishes at 196

J1 " " 200

Unfairness = 200 − 196 = 4

$$\text{Avg. Unfairness} = \frac{8+4+10+4}{4} = \frac{25}{4}$$

$$= 6.5$$

④ How does larger value of quantum size affect the unfairness?

With quantum 5, seed 3

J0 finishes at 185

J1 " " 200

Unfairness = 200 − 185 = 15

$185/200 = 0.925$

seed 3 quantum 25,

J0 finishes at 200

J1 '' '' 125

Unfairness = $200-125 = 75$

$125/200 = 0.625$

avg. unfairness = $\frac{15+75}{2} = 45$

Increase in quantum size increases the unfairness because the window of run is higher leading to more gap time wise.

5 -l 10:100, 10:100

With seed 3

J0 finishes at 20

J1 '' '' 19

Unfairness = $20-19 = 1$

With seed 2

J0 finishes at 19

J1 '' '' 20

Unfairness = $20-19 = 1$

with seed 1

J0 finishes at 20

J1 " " 16

Unfairness = 20 - 16 = 4

avg. unfairness = $\frac{1+1+4}{3} = 2$

-2 1000: 100 , 1000: 100

With seed 1

J0 finishes at 1903

J1 " " 2000

Unfairness = 2000 - 1903 = 97

With seed 2

J0 finishes at 1948

J1 " " 2000

Unfairness = 2000 - 1948 = 52

As in the graph of lotterry fairness

in chapter " it increases with job length.
Though the unfairness metric is defined as -

$$U = \frac{\text{time first job finishes}}{\text{time second job}} \text{ "}$$

Meaning $U = 1$ would be a perfectly fair scheduler, since jobs finish at same time.
I screwed up the metric since I calculated the difference which makes it slightly difficult to compare, I thought question defined it that way :shrug:

Stride scheduler

A, B, C
↓ ↓ ↓
100 50 250 tickets

10,000 - large number to compute "stride"

stride = $\underline{10000}$ = 100 => A

100

= 200 ⇒ B

= 50 ⇒ C

Pass - track the state per process
total executed, stride value added
Lowest "pass" is picked to be executed
next by CPU.

Job length should have no impact on stride scheduler because it will run exactly according to tickets count per process, instead of lottery which achieves the proportions probabilistically over time.
Stride scheduling gets them exactly right at the end of each scheduling cycle.
Therefore, the graph should be straight line just below 1 (almost perfectly fair)