4-tuple

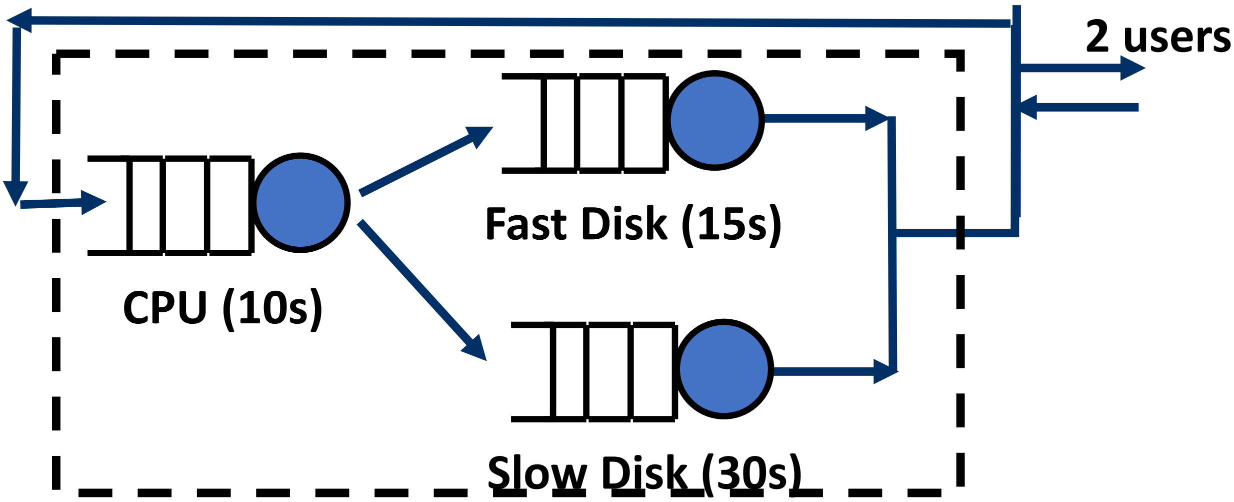
(CPU, CPU, CPU, CPU)
(FD, CPU, SD, FD) …

3 * 3 * 3 * 3 = 3^4

(2,0,0)
(1,0,1).
User 1@CPU, 2@FD or 1@FD & 2@CPU
1st choice (CPU,FD) (FD, CPU)
(0,2,0)
(0,0,2)
(1, 1, 0)
(0,1,1)

3 P_{(2,0,0)} + 3 P_{(2,0,0})
= 4 * P_{(1,1,0)} + 2 * P_{(1,0,1)}

6 P_{(2,0,0)}= 4 * P_{(1,1,0)} + 2 * P_{(1,0,1)}

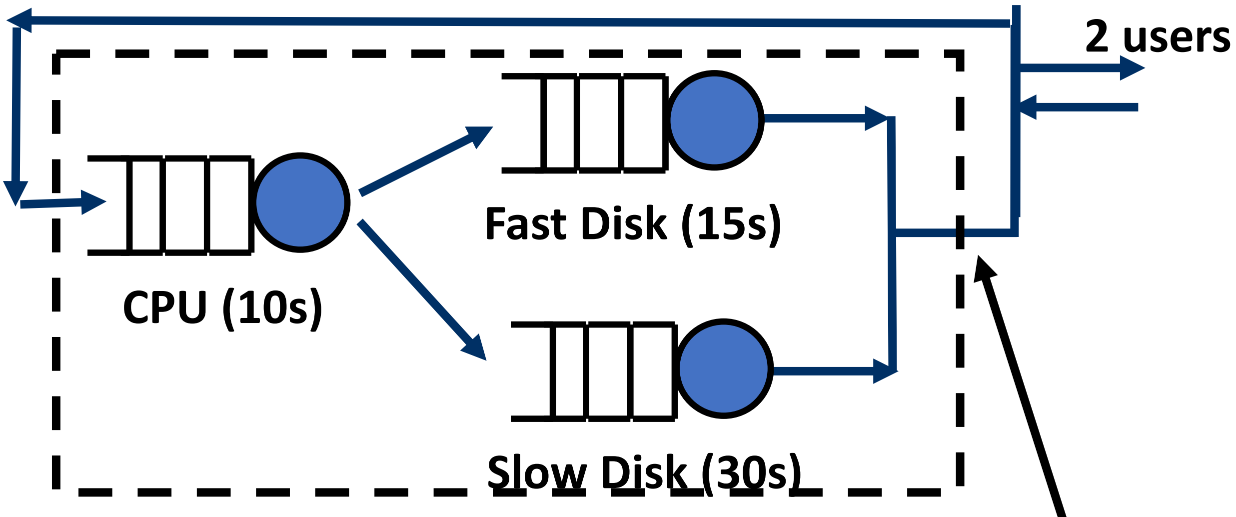



Choice 1:
(Location of User 1, Location of User 2)   3^n

Choice 2:
(#users at CPU, #users at fast disk, #users at slow disk)

Comparing the two choices of states
(CPU, FD) (FD, CPU)
(1,1,0)
(2,0,0) -> (0,1,1)  (3 delta) * (3 delta)

| | |
|---|---|
| Utilisation Law | **U = S X** |
| Forced Flow Law | V(j) = X(j)/X(0) |
| Service demand Law | D(j)=U(j)/X(0) |
| Little’s Law | N=X*R |
| Interactive response | M=X(0)*(Z+R) |




At steady state:
Throughput of the system
= Arrival rate of the CPU

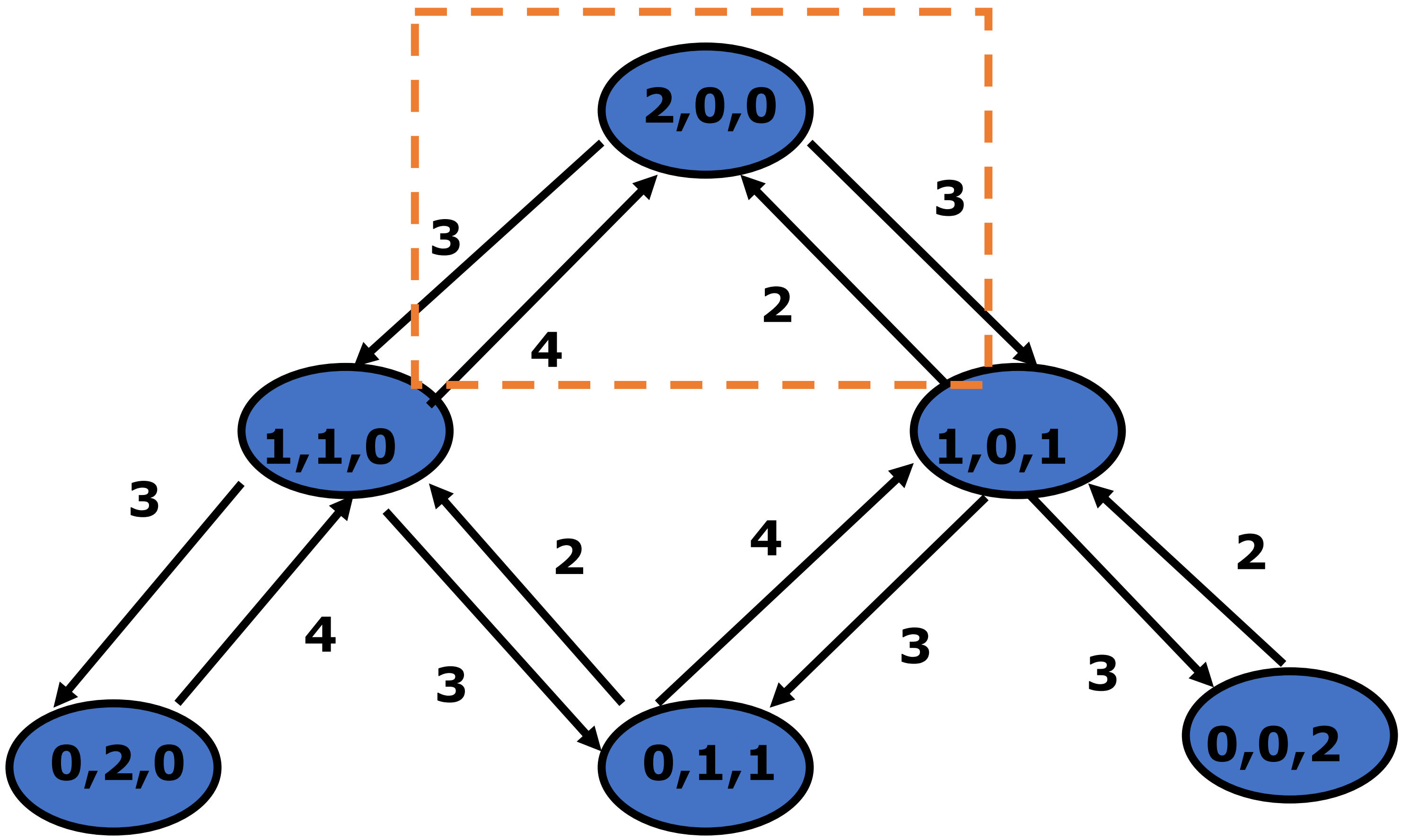
Arrival rate of CPU
= Throughput of the CPU

Water tank analogy

A pipe goes in, a pipe goes out

To have steady water level in the tank, you need

Rate of water into the tank = rate of water out of the tank




Rate of going into the state (2,0,0)
= 4 P110 + 2 P101

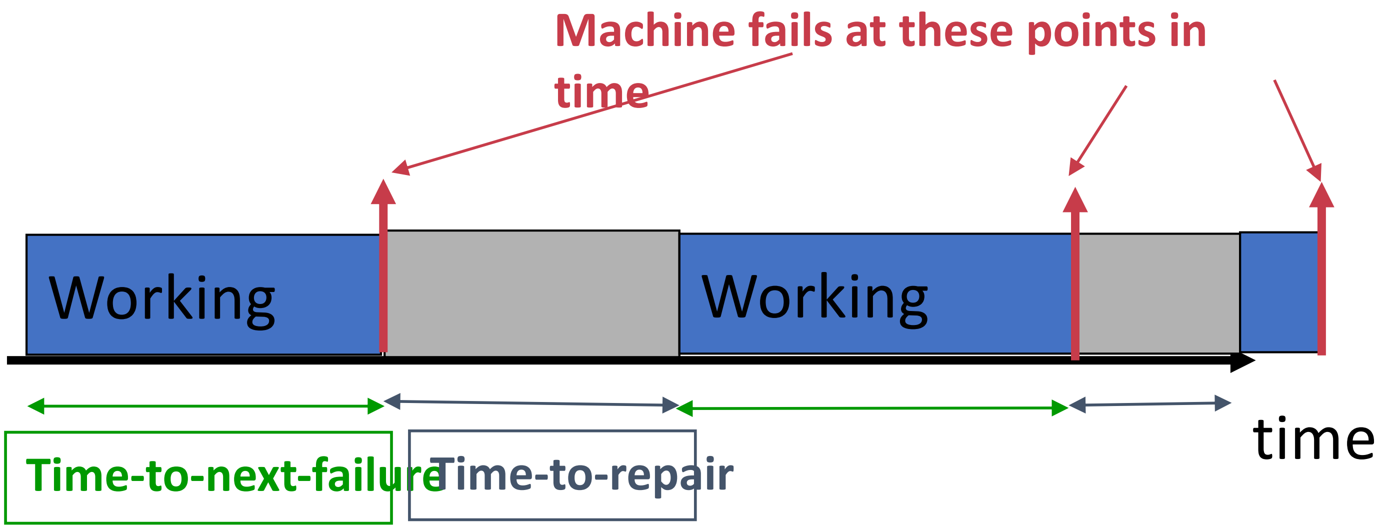
Rate of leaving the state (2,0,0)
= 3 P200 +3 P200
= 6 P200

4 P110 + 2 P101 = 6 P220

2 users
(2,0,0)
(1,1,0), (1,0,1)
(0,2,0), (0,1,1) (0,0,2)

4 users
(4,0,0)
(3,1,0), (3,0,1)
(2,2,0), (2,1,1) (2,0,2)
(1,3,0) (1,2,1)  (1,1,2)  (1,0,3)
(0,4,0) (0,3,1) (0,2,2)  (0,1,3)  (0,0,4)

| | **Meaning** |
|---|---|
| λ | 1 / (Mean-time-to-failure for a machine) |
| μ | 1 / (Mean service time to repair a machine) |
| M | Number of machines (M > N) |
| N | Number of repair staff |



| | Meaning |
|---|---|
| $\lambda$ | 1 / (Mean-time-to-failure for a machine) |
| $\mu$ | 1 / (Mean service time to repair a machine) |
| M | Number of machines |
| N | Number of repair staff |
| P(k) | Prob[k machines have failed] |