Samet Aytac

1819044

HW2 Report

In this report, i will analyze every method one by one which I implement. I will use best case and average case analysis and I will try to figure out both time and space complexity.

i : number of instructors in the department
s : number of students in the department
c : number of courses in the department
m : number of computers (machines) in the department
p : number of physical processors (cores) that will run your student threads concurrently

wc: number of students who are waiting for computer at that time(proportional with number of students)

wa: number of students who are waiting for advisor at that time(proportional with number of students)

sc: suitable courses for particular student(proportional with number of courses)

1)Instructor. isRegistrationOK()

For this function, It is clear that it is independent from all variables, It is just take some arguments and it returns it's value in linear time.

Time Complexity		Space Complexity	
Best Case	Average Case	Best Case	Worst Case
O(1)	O(1)	O(1)	O(1)

2)Barrier.await()

Again,this function is independent from all variables.

Time Complexity		Space Complexity	
Best Case	Average Case	Best Case	Average Case
O(1)	O(1)	O(1)	O(1)

3)Student.run()

This is the most complicated function we have. Let's start with best case. For best case, student will immedietly find computer,took his/her courses, got approval and join party. So, this function is O(1) for time and space complexity for best case.

For average case, it is more complicated. First, we need to figure out what is the probability that a student find computer. It is c/u. At that point, we need to find how much time this student waits. At

that point, an example would be good to understand concept. Let say we have 12 unregister student wait for computer and all computers are taken. So , for average time student need to wait for 6 students register. So, students should wait =(1-c/u) * u/2 * avarege time for students register who has computer. Let's check if students find computer,how much time for took the approval. Student take courses and he will wait for advisor. Let say,there is 8 students who is ready for approval and there is 2 advisor. So in change of (1-2/8), student will wait and there is 4 students per advisor,student will wait for 2 students in average. Then, let we check what is the probability of getting approval. It is sc/c+(sc-1)/(c-1)+(sc-2)/(c-2) . Now,let's formulize all of this by using expected value.

Let we introduce variables;

t_{total} -> Total time to joining party for student.
t_c->Time for joining party for student who has just got computer.
t_a -> Time for joining party for student who is in advisor's room(Advisor checking for this student)
t_r->Time for joining party for student who took advisor's approval.

P_{gc} ->Probability of finding empty computer= m/(wc)

P_a -> Probability of student's advisor is available for student = a/(wa)

P_{sc} -> Probability of student's all courses are suitable = $(\frac{sc}{c})$ * $(\frac{sc-1}{c-1})$ * $(\frac{sc-2}{c-2})$

$t_{total} = P_{gc} * t_c + (1 - P_{gc}) * (\frac{wc}{2}) * t_c$

$t_c = P_a * t_a + 1-(P_a) * (\frac{wa}{2a}) * t_a$

$t_a = P_{sc} * t_r + (1-P_{sc}) * t_{total}$

t_r=>linear

When we solve these 3 equation,we can figure out that

$$T_{total}=\frac{\left(\mathrm{Pa}+\frac{(1-Pa)*wa}{2a}\right)*\left(\mathrm{Pgc}+\frac{(1-Pgc)*wc}{2}\right)*Psc*tr}{1-\left(\left(\mathrm{Pa}+\frac{(1-Pa)*wa}{2a}\right)*\left(\mathrm{Pgc}+\frac{(1-Pgc)*wc}{2}\right)*(1-Psc)\right)}$$

For space complexity, if student waits, there is no additional space usage. So, we should focus on space usage when student is active. Only space usage is how many course student take. So

$S_{total}=P_{sc}*3*(\text{course size}) + (1-P_{sc}) * (S_{total}+3 * (\text{course size}))$

$$S_{total}= \frac{3*course\ size}{Psc}$$

(3*course size) is linear. We can omit it in Big O notation. When we use $P_{sc}=(\frac{sc}{c}) * (\frac{sc-1}{c-1}) * (\frac{sc-2}{c-2})$

$S_{total}= O((\frac{c}{sc}) * (\frac{c-1}{sc-1}) * (\frac{c-2}{sc-2}))$

Time Complexity		Space Complexity	
Best Case	Average Case	Best Case	Average Case
O(1)	$O(\frac{\left(Pa+\frac{(1-Pa)*wa}{2a}\right)*\left(Pgc+\frac{(1-Pgc)*wc}{2}\right)*Psc*tr}{1-\left(\left(Pa+\frac{(1-Pa)*wa}{2a}\right)*\left(Pgc+\frac{(1-Pgc)*wc}{2}\right)*(1-Psc)\right)})$	O(1)	$O((\frac{c}{sc}) * (\frac{c-1}{sc-1}) * (\frac{c-2}{sc-2}))$

2)Register.main()

For main,

```
for ( Student s : ceng.students )
  {
    s.start( ceng ) ;
  }
```

is the dominating part of code. So, this code will call student.run() method for s times. Since it is thread,time complexity will be divided to number of processors.

Time Complexity		Space Complexity	
Best Case	Average Case	Best Case	Average Case
O(s/p)	$O(\frac{s}{p} * \frac{\left(Pa+\frac{(1-Pa)*wa}{2a}\right)*\left(Pgc+\frac{(1-Pgc)*wc}{2}\right)*Psc*tr}{1-\left(\left(Pa+\frac{(1-Pa)*wa}{2a}\right)*\left(Pgc+\frac{(1-Pgc)*wc}{2}\right)*(1-Psc)\right)})$	O(s)	$O(s*(\frac{c}{sc}) * (\frac{c-1}{sc-1}) * (\frac{c-2}{sc-2}))$

Here is an example that how threads running paralelly.