Лабораторная работа №11

Модель системы массового обслуживания M |M| 1

Ибатулина Д.Э.

18 апреля 2025

Российский университет дружбы народов, Москва, Россия

Информация

Докладчик

- Ибатулина Дарья Эдуардовна
- студентка группы НФИбд-01-22
- Фундаментальная информатика и информационные технологии
- Российский университет дружбы народов
- · 1132226434@rudn.ru
- https://deibatulina.github.io



Вводная часть

Теоретическое введение

Модель системы массового обслуживания (СМО) M|M|1 — одна из базовых моделей теории массового обслуживания, широко применяемая для анализа процессов обслуживания заявок в различных системах (телекоммуникации, вычислительные сети, производственные процессы и др.).

Обозначение **M|M|1** расшифровывается следующим образом: - Первая буква **M** (Markovian) означает, что время между поступлениями заявок в систему подчиняется экспоненциальному распределению (процесс поступления заявок — пуассоновский). - Вторая буква **M** указывает, что время обслуживания каждой заявки также экспоненциально распределено. - Цифра **1** означает, что в системе имеется один обслуживающий канал (один сервер).

Цель и задачи работы

Цель работы

Реализовать модель $M \vert M \vert 1$ в CPN tools.

Задание

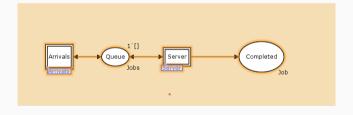
- Реализовать в CPN Tools модель системы массового обслуживания M|M|1;
- Настроить мониторинг параметров моделируемой системы и нарисовать графики очереди.

Выполнение лабораторной работы

Постановка задачи

В систему поступает поток заявок двух типов, распределённый по пуассоновскому закону. Заявки поступают в очередь сервера на обработку. Дисциплина очереди - FIFO. Если сервер находится в режиме ожидания (нет заявок на сервере), то заявка поступает на обработку сервером.

Граф сети системы обработки заявок в очереди

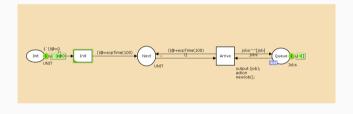


Декларации системы

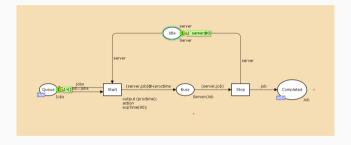
```
▼New net.cpn
   Step: 0
   Time: 0
 ▶ Options
 ▶ History
 ▼ Declarations
    ▼System
    ▼colset UNIT = unit timed;
    ▼colset INT = int:
    ▼colset Server = with server timed;
    ▼colset JobType = with A | B:
    ▼colset Job = record jobType : JobType *
     AT : INT:
    v colset Jobs = list Job;
    ▼colset ServerxJob = product Server * Job timed:
    ▼var proctime : INT:
    ▼var job : Job;
    var jobs : Jobs;

▼fun expTime (mean: int) =
     let
      val realMean Real.fromInt mean:
      val rv = exponential ((1.0 / realMean))
      floor (rv + 0.5)
     end:
    vfun intTime() = IntInf.toInt(time());
    ▼fun newJob() = {
     iobType = JobType.ran().
     AT = intTime() };
   ▶ Standard declarations
 ► Monitors
 System
     Arrivals
     Server
```

Граф генератора заявок системы



Граф процесса обработки заявок на сервере системы



Множества цветов системы (colorset) (1/2)

- · фишки типа UNIT определяют моменты времени;
- фишки типа INT определяют моменты поступления заявок в систему.
- фишки типа **JobType** определяют 2 типа заявок А и В;
- кортеж **Job** имеет 2 поля: jobType определяет тип работы (соответственно имеет тип **JobType**, поле **AT** имеет тип **INT** и используется для хранения времени нахождения заявки в системе);
- фишки Jobs список заявок;
- фишки типа **ServerxJob** определяют состояние сервера, занятого обработкой заявок.

Множества цветов системы (colorset) (2/2)

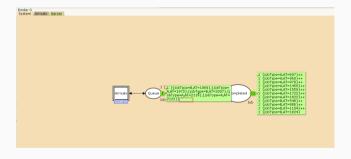
Переменные модели:

- · proctime определяет время обработки заявки;
- · job определяет тип заявки;
- · jobs определяет поступление заявок в очередь.

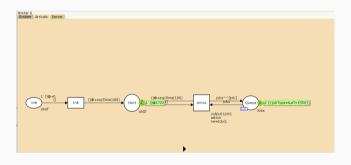
Определим функции системы:

- функция **expTime** описывает генерацию целочисленных значений через интервалы времени, распределённые по экспоненциальному закону;
- функция intTime преобразует текущее модельное время в целое число;
- функция **newJob** возвращает значение из набора **Job** случайный выбор типа заявки (А или В).

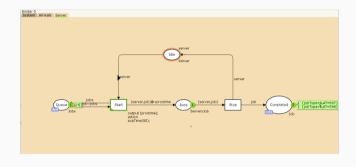
Граф сети системы обработки заявок в очереди



Граф генератора заявок системы



Граф процесса обработки заявок на сервере системы



Функция Predicate монитора Ostanovka

```
Binder 0
System Arrivals Server fun pred <Ostanovka>

fun pred (bindelem) = let fun predBindElem (Server'Start (1, {job.jobs,proctime})) = Queue_Delay.count() = 200 | predBindElem _ = false in predBindElem bindelem end
```

Функция Observer монитора Queue Delay

```
Binder 0

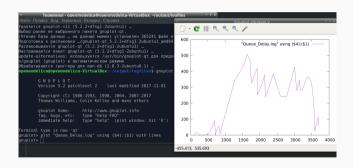
System Arrivals Server fun obs < Queue Delay >

fun obs (bindelem) = let  
fun obsBindElem (Server'Start (1, {job,jobs,proctime})) = (intTime() - (#AT job))  
| obsBindElem _ = ~1  
in  
obsBindElem bindelem  
end
```

Файл Queue_Delay.log

```
/home/openmodelica/output/logfiles/Queue_Delay.log - Mousepad
Файл Правка Поиск Вид Документ Справка
#data counter step time
0 1 3 101
322 2 11 429
215 3 13 433
315 4 18 561
281 5 20 589
442 6 24 778
381 7 26 785
436 8 29 870
458 9 31 923
422 10 33 939
414 11 36 1032
378 12 38 1056
308 13 40 1138
197 14 42 1219
83 15 45 1376
32 16 48 1436
0 17 51 1599
452 18 61 2131
412 19 63 2160
366 20 65 2194
369 21 67 2228
267 22 69 2230
298 23 71 2294
173 24 73 2301
```

График изменения задержки в очереди



Функция Observer монитора Queue Delay Real

```
Binder 0
System Arrivals Server fun obs < Queue Delay Real>
fun obs (bindelem) =
let
fun obsBindElem (Server'Start (1, {job,jobs,proctime})) =
Real.fromInt(intTime() - (#AT job))
| obsBindElem = ~1.0
in
obsBindElem bindelem |
end
```

Содержимое Queue_Delay_Real.log

```
/home/openmodelica/output/logfiles/Queue Delay Real.log - Mousepad
Файл Правка Поиск Вид Документ Справка
#data counter step time
0.000000 1 3 101
322.000000 2 11 429
215.000000 3 13 433
315.000000 4 18 561
281.000000 5 20 589
442.000000 6 24 778
381.000000 7 26 785
436.000000 8 29 870
458.000000 9 31 923
422.000000 10 33 939
414.000000 11 36 1032
378.000000 12 38 1056
308.000000 13 40 1138
197.000000 14 42 1219
83.000000 15 45 1376
32.000000 16 48 1436
```

Функция Observer монитора Long Delay Time

```
Binder 0
System Arrivals Server globref longdelaytime fun obs <Long Delay Time>
fun obs (bindelem) = 
if IntInf.toInt(Queue_Delay.last()) >= (llongdelaytime)
then 1
else 0
```

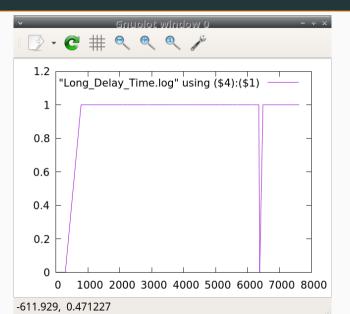
Определение longdelaytime в декларациях

```
▼ Declarations
 ▼ colset UNIT = unit timed:
 ▼colset INT = int;
 ▼colset Server = with server timed;
 ▼ colset JobType = with A | B:
 vcolset lob = record iobType : lobType *
  AT : INT:
 ▼colset Jobs = list Job;
 ▼colset Servendob = product Server * Job timed;
 ▼var proctime : INT:
 var job : Job:
 ▼ var jobs : Jobs;
 ▼fun expTime (mean; int) = let val realMean = Real.fromInt mean val ry = exponential ((1.0/realMean)) in floor (ry+0.5) end:
 vfun intTime() = IntInf.toInt (time ()):
 ▼fun newloh () = {
   jobType = JobType.ran (),
  AT = intTime () };
 ▼globref longdelaytime = 200;
                                       System Arrivals Server globref longdelaytime fun obs <Long Delay Time>
▼ Monitors
                                       globref longdelaytime = 200:
```

Содержимое Long_Delay_Time.log

```
/home/openmodelica/ou/out/florifiles/Long Delay Time log - Mousepar
Файл Правка Поиск Вид Документ Справка
#data counter step time
1 1 116 3490
0 2 118 3496
0 3 122 3604
1 4 124 3721
1 5 126 3767
0 6 130 3950
0 7 132 3954
0 8 135 4097
1 9 143 4551
1 10 145 4575
0 11 147 4592
1 12 149 4709
1 13 151 4720
1 14 153 4756
0 15 156 4817
0 16 159 4950
```

Периоды времени, когда значения задержки в очереди превышали 200



Заключительная часть

Выводы по работе

В процессе выполнения данной лабораторной работы я реализовала модель системы массового обслуживания M|M|1 в CPN Tools.