

# Презентация по лабораторной работе №11

Дисциплина: Имитационное моделирование

---

Лобанова П.И.

10 апреля 2025

Российский университет дружбы народов, Москва, Россия

## Информация

---

- Лобанова Полина Иннокентьевна
- Учащаяся на направлении “Фундаментальная информатика и информационные технологии”
- Студентка группы НФИбд-02-22
- [polla-2004@mail.ru](mailto:polla-2004@mail.ru)

Цель

---

Реализовать модель системы массового обслуживания  $M|M|1$ .

## Задание

---

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.

## Выполнение

---



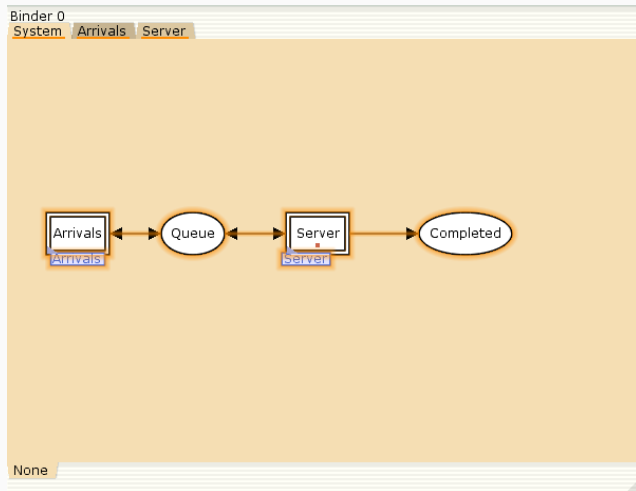


Рис. 1: Граф сети системы обработки заявок в очереди

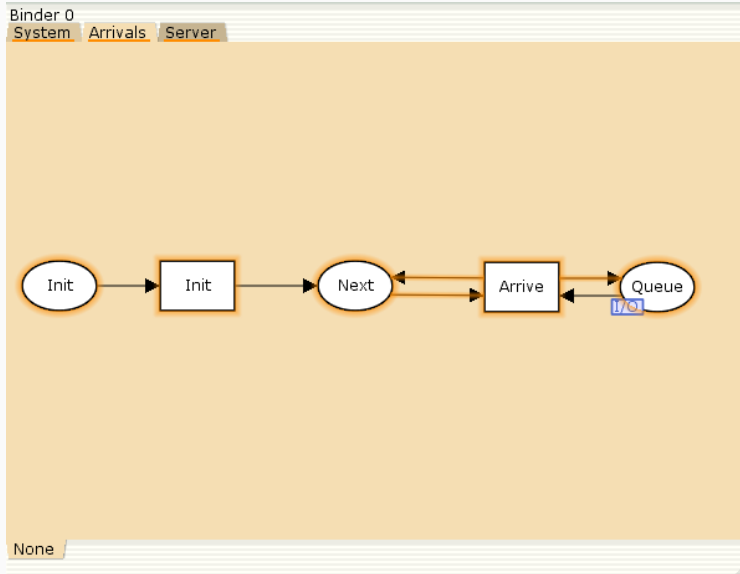


Рис. 2: Граф генератора заявок системы

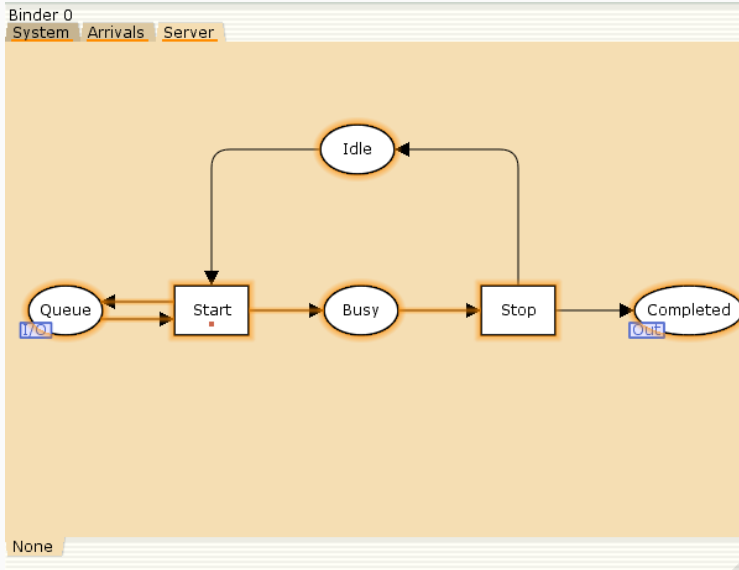


Рис. 3: Граф процесса обработки заявок на сервере системы

- ▼ Declarations
  - ▼ Standard declarations
    - ▼ colset BOOL = bool;
    - ▶ colset STRING
  - ▶ colset UNIT
  - ▼ colset INT = int;
  - ▼ colset Server = with server timed;
  - ▼ colset JobType = with A | B;
  - ▼ colset Job = record jobType : JobType \*
    - AT : INT;
  - ▼ colset Jobs = list Job;
  - ▼ colset SerxerxJob = 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))
    - in
      - floor (rv+0.5)
    - end;
    - fun intTime () = IntInf.toInt (time());
    - fun newJob () = {jobType = JobType.ran(),
      - AT = intTime ()}

Рис. 4: Задание деклараций

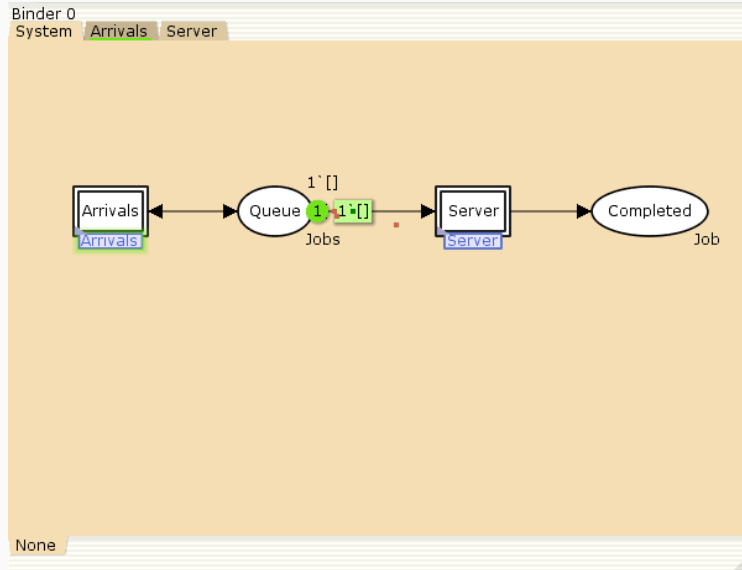


Рис. 5: Параметры элементов основного графа системы обработки заявок в очереди

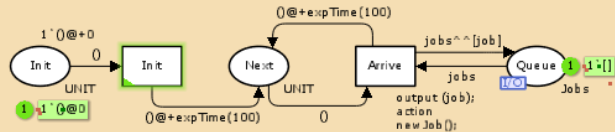


Рис. 6: Параметры элементов генератора заявок системы

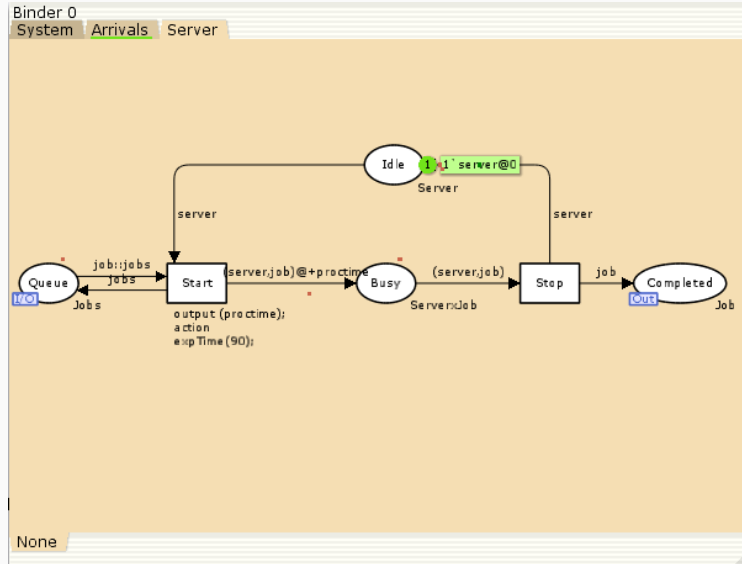


Рис. 7: Параметры элементов обработчика заявок системы

Для мониторинга параметров очереди системы  $M|M|1$  потребовалась палитра Monitoring. Выбрала Break Point (точка останова) и установила её на переход Start.

```
▼ Monitors
  ► Queue Delay
  ▼ Ostanovka
    Type: Break point
    ► Nodes ordered by pages
    ▼ Predicate
      fun pred (bindelem) =
      let
        fun predBindElem (Server'Start (1,
                                         {job,jobs,proctime}))
        = Queue_Delay.count()=200
          | predBindElem _ = false
      in
        predBindElem bindelem
      end
```

Рис. 8: Функция *Predicate* монитора *Ostanovka*



Далее необходимо было определить конструкцию Queue\_Delay.count(). С помощью палитры Monitoring выбрала Data Call и установила на переходе Start.

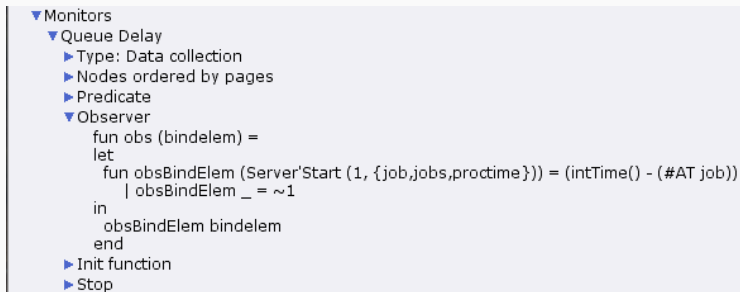


Рис. 9: Функция Observer монитора Queue Delay

После запуска программы на выполнение в каталоге с кодом программы появился файл Queue\_Delay.log, содержащий в первой колонке — значение задержки очереди, во второй — счётчик, в третьей — шаг, в четвёртой — время.



Рис. 10: График изменения задержки в очереди

Посчитала задержку в действительных значениях.

```
▼ Monitors
  ► Queue Delay
  ► Ostanovka
  ▼ Queue Delay Real
    ► Type: Data collection
    ► Nodes ordered by pages
    ► Predicate
    ▼ Observer
      fun obs (bindelem) =
      let
        fun obsBindElem (Server'Start {1, {job,jobs,proctime}}) = Real.fromInt(intTime()-(#AT job))
          | obsBindElem _ = ~1.0
      in
        obsBindElem bindelem
      end
```

Рис. 11: Функция *Observer* монитора *Queue Delay Real*

Посчитала, сколько раз задержка превысила заданное значение.

```
▼ globref longdelaytime = 200;
▼ Monitors
  ► Queue Delay
  ► Ostanovka
  ► Queue Delay Real
  ▼ Long Delay Time
    ► Type: Data collection
    ► Nodes ordered by pages
    ► Predicate
    ▼ Observer
      fun obs (bindelem) =
        if IntInf.toInt(Queue_Delay.last()) >= (!longdelaytime)
        then 1
        else 0
```

Рис. 12: Функция *Observer* монитора *Long Delay Time*

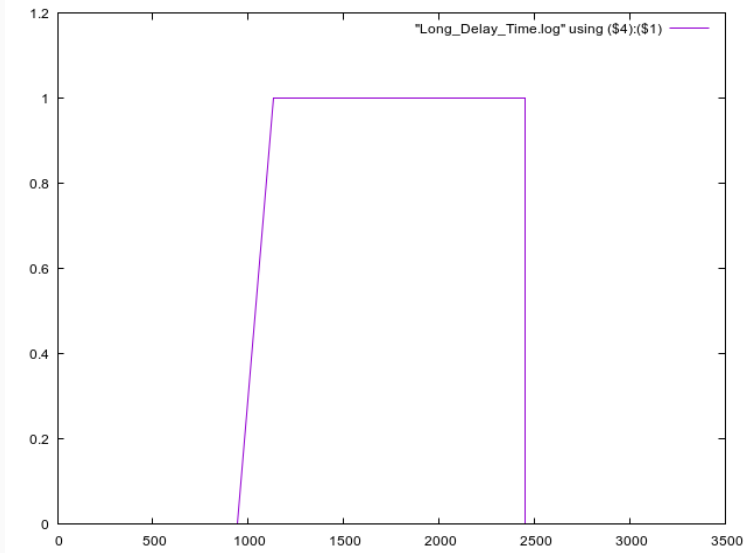


Рис. 13: Периоды времени, когда значения задержки в очереди превышали заданное значение

## Вывод

---

Я реализовала модель системы массового обслуживания  $M|M|1$ .