

# Лабораторная работа №11

Модель системы массового обслуживания  $M|M|1$

---

Дворкина Е. В.

19 апреля 2025

Российский университет дружбы народов, Москва, Россия

- Дворкина Ева Владимировна
- студентка
- группа НФИбд-01-22
- Российский университет дружбы народов
- 1132226447@rudn.ru
- <https://github.com/evdvorkina>



## Введение

---

### Цель работы

Реализовать в CPN Tools модель системы массового обслуживания  $M|M|1$ .

### Задание

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

## Выполнение лабораторной работы

---

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 очередь

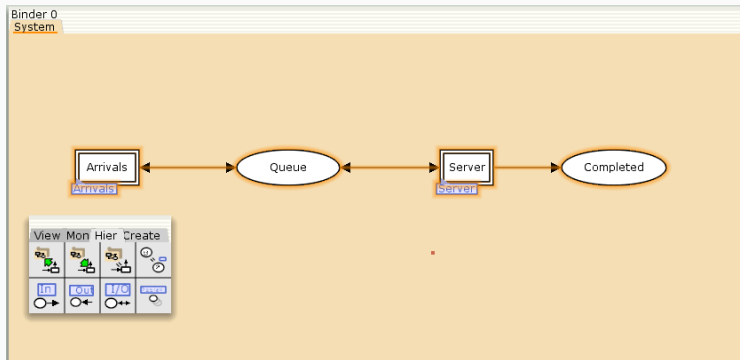


Рис. 1: Граф сети системы обработки заявок в очередь

# Граф генератора заявок системы

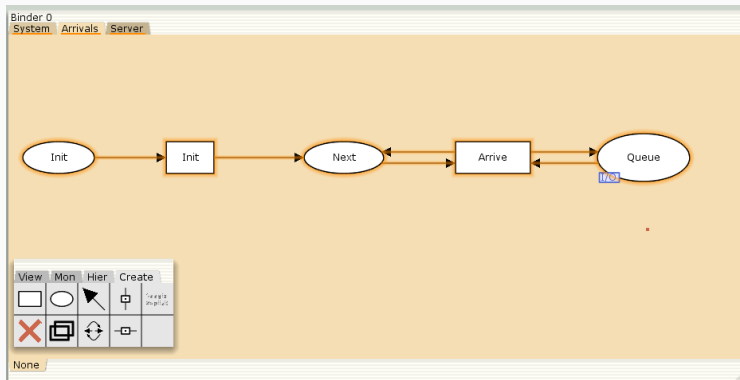


Рис. 2: Граф генератора заявок системы



# Граф процесса обработки заявок на сервере системы

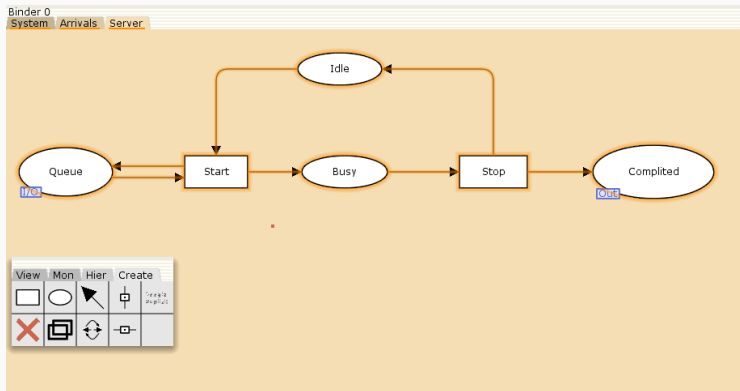
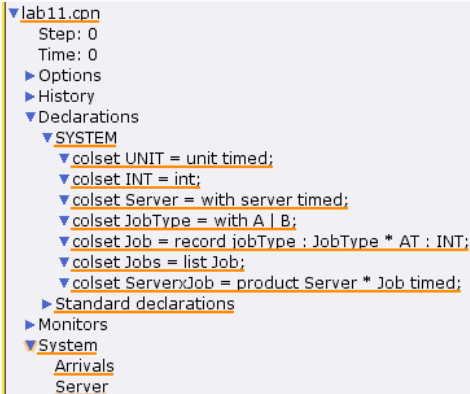


Рис. 3: Граф процесса обработки заявок на сервере системы



▼ lab11.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;  
▼ colset Jobs = list Job;  
▼ colset ServerxJob = product Server \* Job timed;  
► Standard declarations  
► Monitors  
▼ System  
    Arrivals  
    Server

Рис. 4: Определения множества цветов системы

```
▶ Options
▼ lab11.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;
      ▼ colset Jobs = list Job;
      ▼ colset ServersJob = product Server * Job timed;
      ▼ var proctime : INT;
      ▼ var job : Job;
      ▼ var jobs : Jobs;
      ▶ Standard declarations
    ▶ Monitors
    ▼ System
      Arrivals
      Server
```

Рис. 5: Определение переменных модели

```
▼ 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;
    ▼ 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))
      in
        floor (rv + 0.5)
      end;
    ▼ fun intTime () = IntInf.toInt (time());
    ▼ fun newJob () = {jobType = JobType.ran(), AT = intTime()};
```

Рис. 6: Определение функций системы

## Параметры модели основного графа

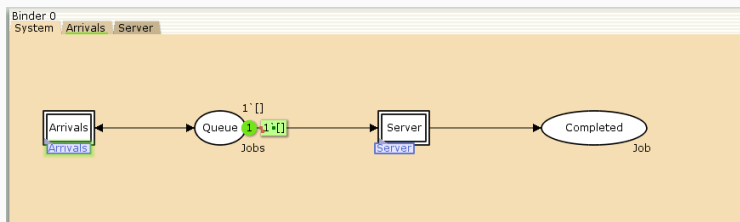


Рис. 7: Параметры элементов основного графа системы обработки заявок в очереди

# Параметры модели графа генератора заявок

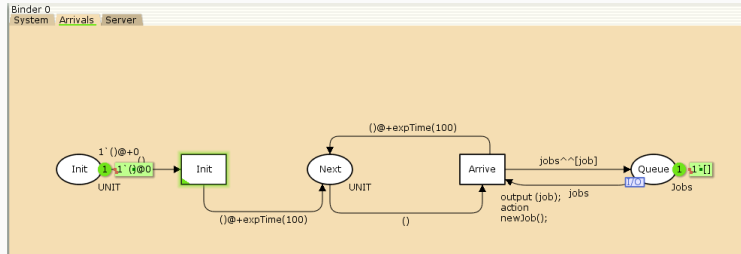


Рис. 8: Параметры элементов генератора заявок системы

# Параметры модели графа обработчика заявок системы

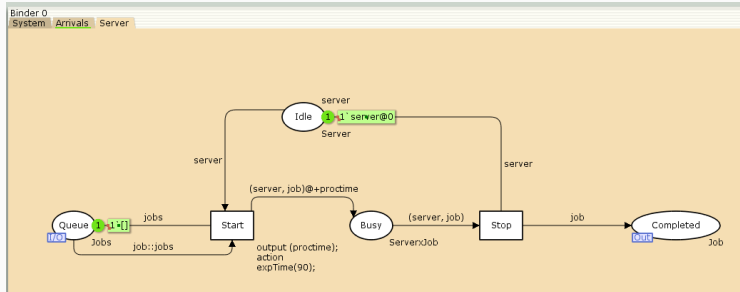


Рис. 9: Параметры элементов обработчика заявок системы

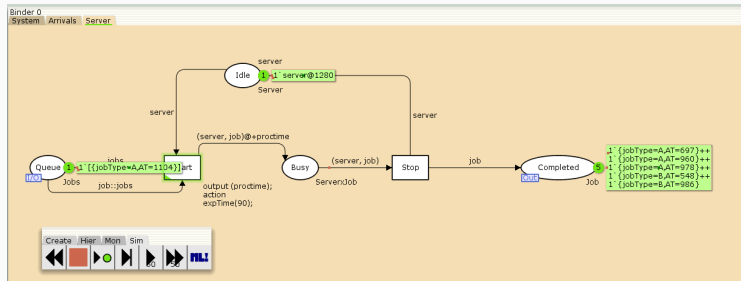


Рис. 10: Запуск системы обработки заявок в очереди



The screenshot displays the Ostanovka software interface. On the left, a tree view shows the configuration for a monitor named 'Ostanovka'. The 'Predicate' section is expanded, showing a function definition for 'pred' that checks for a specific condition in the simulation. On the right, a control panel with tabs 'Create', 'Hier', 'Mon', and 'Sim' is visible. The 'Mon' tab is active, showing a table of monitor parameters.

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

Create	Hier	Mon	Sim	
Data Coll	Mark Size	Break point	User def	Write in file
LL DC	Coun Tran	Place Cont	Tran Enab	

Рис. 11: Функция Predicate монитора Ostanovka

```
▼ Queue Delay
  ▶ Type: Data collection
  ▶ Nodes ordered by pages
  ▶ Predicate
  ▼ Observer
    fun obs (bindelem) =
      let
        fun obsBindElem (Server'Start (1, {job,jobs,proctime})) = (intTime() - (#AT job))
          | obsBindElem _ = ~1
      in
        obsBindElem bindelem
      end
  ▶ Init function
  ▶ Stop
```

Рис. 12: Функция Observer монитора Queue Delay

```
• global int queueDelayTime = 200;
```

### ▼ Monitors

#### ▶ Ostanovka

#### ▶ Queue Delay

### ▼ Real Queue Delay

#### ▶ 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
```

#### ▶ Init function

#### ▶ Stop

Рис. 13: Функция Observer монитора Queue Delay Real

# Мониторинг параметров моделируемой системы (Long Delay Time)

```
▼ globref longdelaytime=200;
▼ Monitors
  ▶ Ostanovka
  ▶ Queue Delay
  ▼ Real Queue Delay
    ▶ Type: Data collection
    ▶ Nodes ordered by pages
    ▶ Predicate
    ▶ Observer
    ▶ Init function
    ▶ Stop
  ▼ 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. 14: Функция Observer монитора Long Delay Time

```
#!/usr/bin/gnuplot -persist
# задаём текстовую кодировку,
# тип терминала, тип и размер шрифта
set encoding utf8
set term pdfcairo font "Arial,9"
# задаём выходной файл графика
set out 'qm.pdf'
# задаём стиль линии
set style line 2
plot "Queue_Delay.log" using ($4):($1) with lines
```

## График изменения задержки в очереди

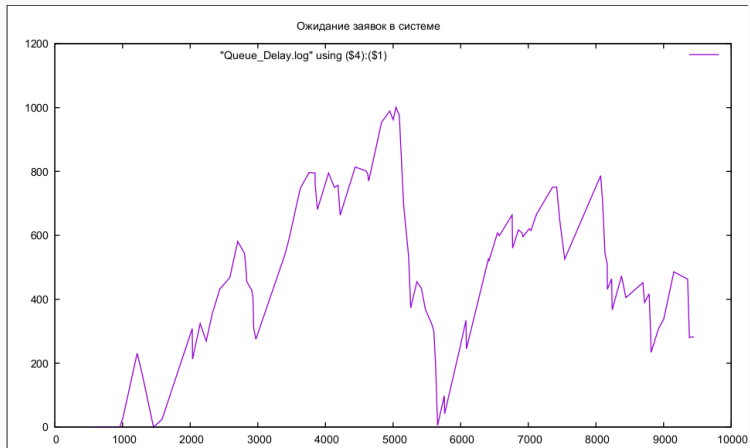


Рис. 15: График изменения задержки в очереди

```
#!/usr/bin/gnuplot -persist
# задаём текстовую кодировку,
# тип терминала, тип и размер шрифта
set encoding utf8
set term pdfcairo font "Arial,9"
# задаём выходной файл графика
set out 'qm.pdf'
# задаём стиль линии
set style line 2
plot [0:] [0:1.2] "Long_Delay_Time.log" using ($4):($1) with lines
```

## Периоды времени, когда значения задержки в очереди превышали заданное значение

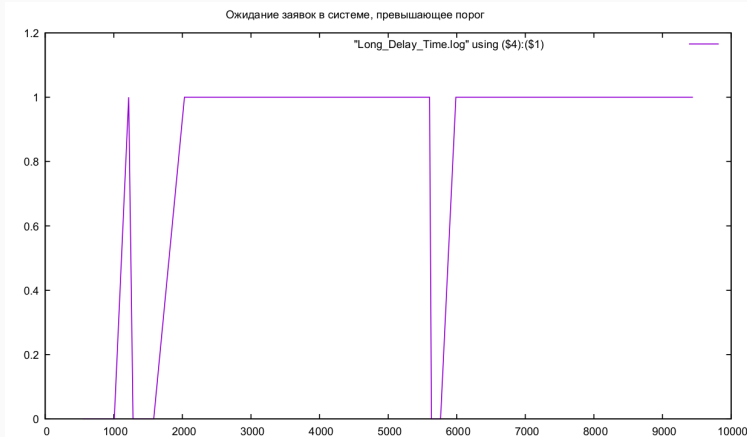


Рис. 16: Периоды времени, когда значения задержки в очереди превышали заданное значение



В результате выполнения работы была реализована в CPN Tools модель системы массового обслуживания  $M|M|1$ .