Лабораторная работа №11

Дисциплина - имитационное моделирование

Пронякова О.М.

03 апреля 2025

Российский университет дружбы народов, Москва, Россия

Информация

Докладчик

- Пронякова Ольга Максимовна
- студент НКАбд-02-22
- факультет физико-математических и естественных наук
- Российский университет дружбы народов

Создание презентации

Цель работы

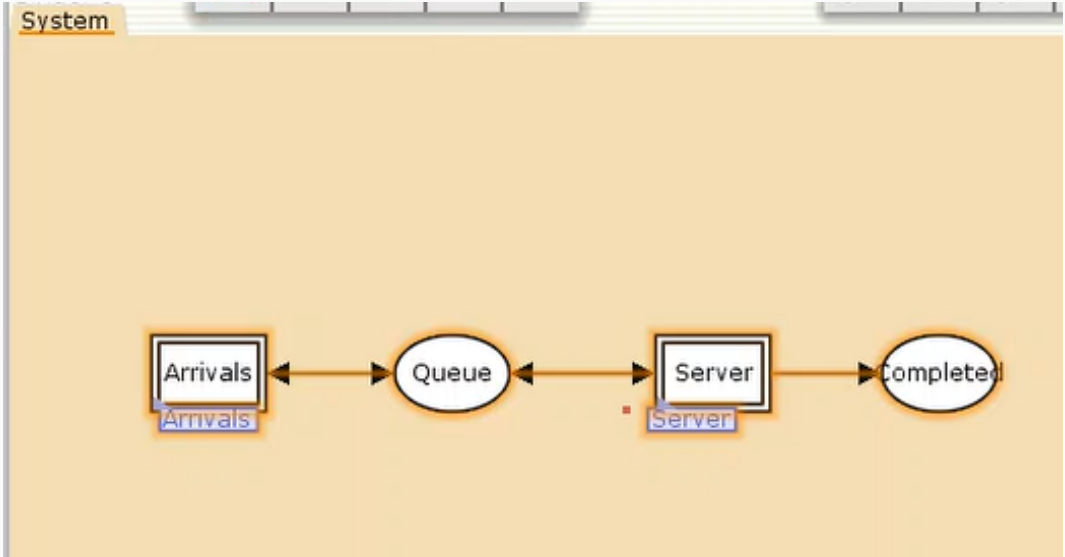
Научиться работать с Моделью системы массового обслуживания M |M |1

Этапы выполнения работы

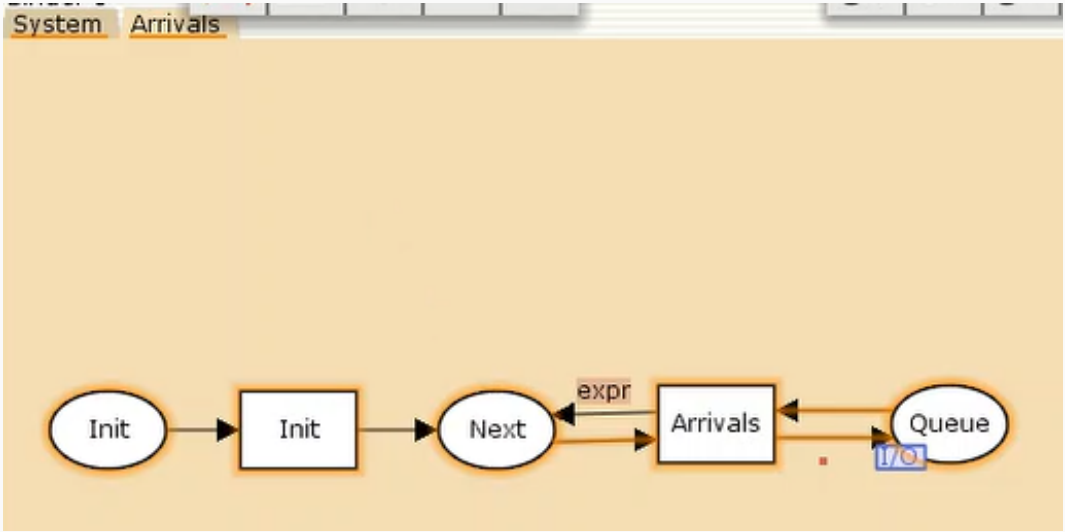
В систему поступает поток заявок двух типов, распределённый по пуассоновскому закону. Заявки поступают в очередь сервера на обработку. Дисциплина очереди - FIFO. Если сервер находится в режиме ожидания (нет заявок на сервере), то заявка поступает на обработку сервером.

Этапы выполнения работы

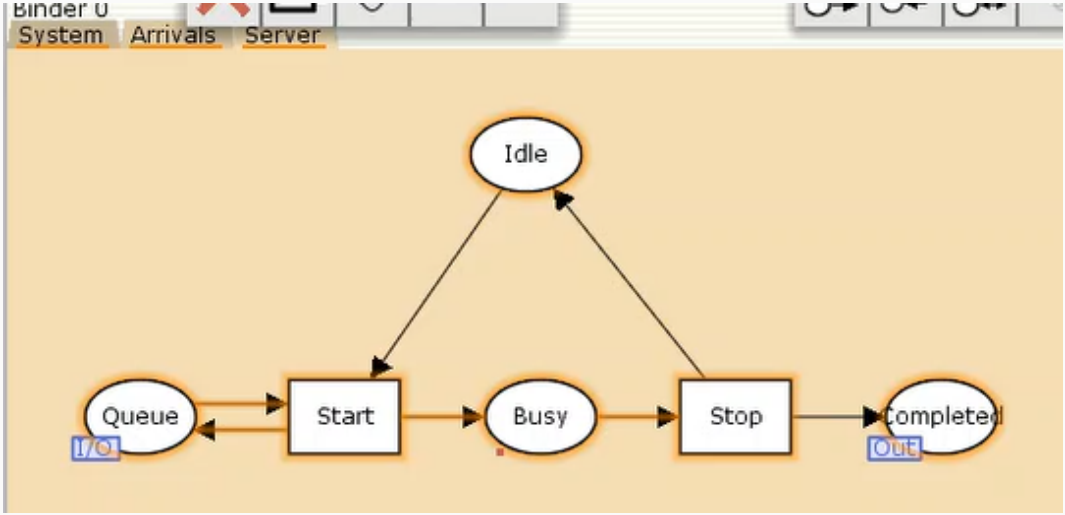
Будем использовать три отдельных листа: на первом листе опишем граф системы (рис. 11.1), на втором — генератор заявок (рис. 11.2), на третьем — сервер обработки заявок (рис. 11.3). 1.1. Сеть имеет 2 позиции (очередь — Queue, обслуженные заявки — Complited) и два перехода (генерировать заявку — Arrivals, передать заявку на обработку сер- веру — Server). Переходы имеют сложную иерархическую структуру, задаваемую на отдельных листах модели (с помощью соответствующего инструмента меню — Hierarchy)(рис.1).


System
Arrivals
Queue
Server
Completed
Arrivals
Server

раф генератора заявок имеет 3 позиции (текущая заявка — Init, следующая заявка — Next, очередь — Queue из листа System) и 2 перехода (Init — определяет распределение поступления заявок по экспоненциальному закону с интенсивностью 100 заявок в единицу времени, Arrive — определяет поступление заявок в очередь)(рис.2).


System
Arrivals
Init
Init
Next
expr
Arrivals
Queue
I/O

Граф процесса обработки заявок на сервере имеет 4 позиции (Busy — сервер занят, Idle — сервер в режиме ожидания, Queue и Complited из листа System) и 2 перехода (Start — начать обработку заявки, Stop — закончить обработку заявки)(рис.3).


Binder 0
System
Arrivals
Server
Idle
Queue
Start
Busy
Stop
Completed
I/O
Out

Зададим декларации системы. Определим множества цветов системы (colorset): – фишки типа UNIT определяют моменты времени; – фишки типа INT определяют моменты поступления заявок в систему. – фишки типа JobType определяют 2 типа заявок — A и B; – кортеж Job имеет 2 поля: jobType определяет тип работы (соответственно име- ет 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(рис.4).

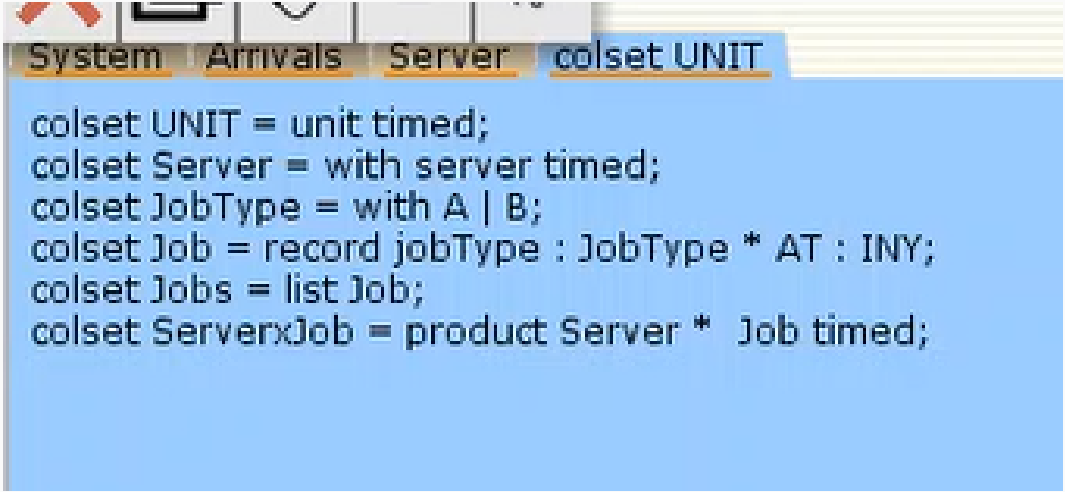


Рис. 4: Декларации системы

Переменные модели: – proctime — определяет время обработки заявки; – job — определяет тип заявки; – jobs — определяет поступление заявок в очередь(рис.5).

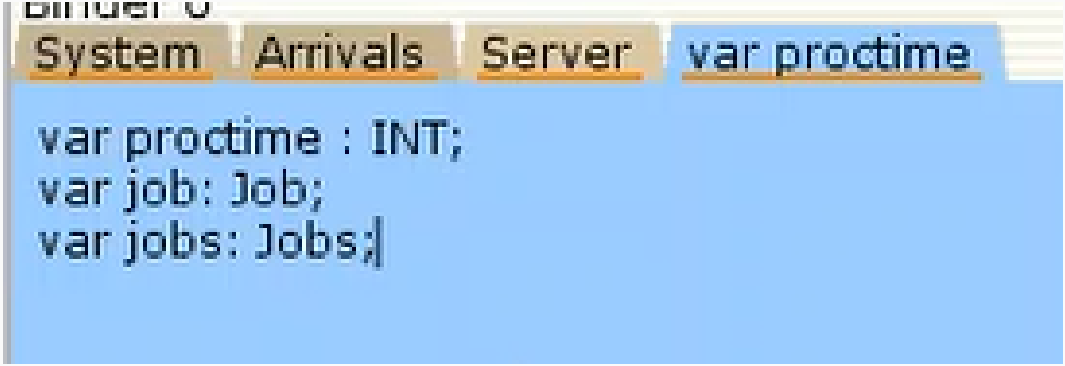


Рис. 5: Декларации системы

Определим функции системы: – функция expTime описывает генерацию целочисленных значений через интерва- лы 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 (A или B)(рис.6), (рис.7), (рис.8), (рис.9).

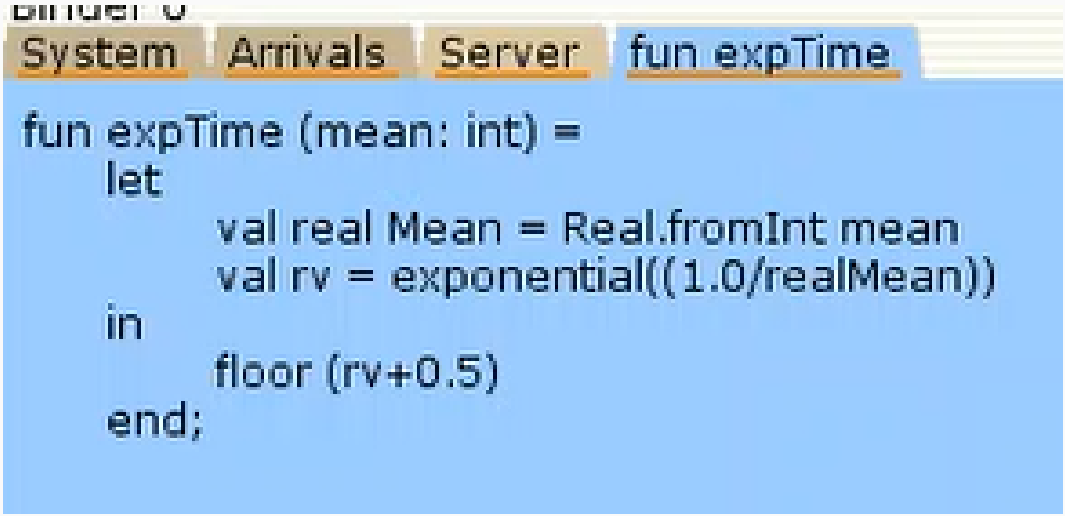


Рис. 6: Декларации системы

Рис. 7: Декларации системы

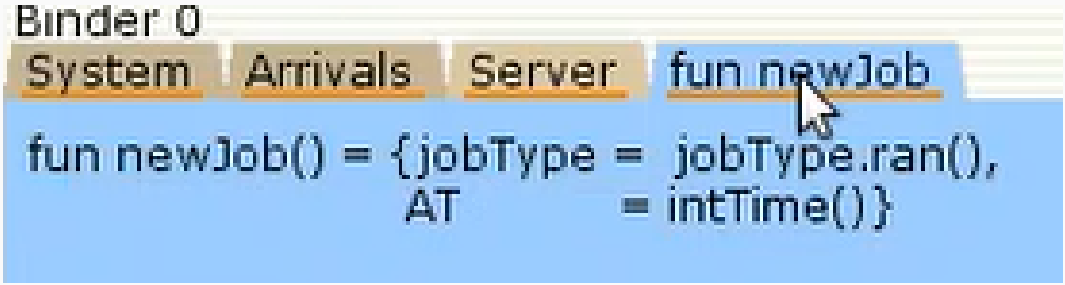
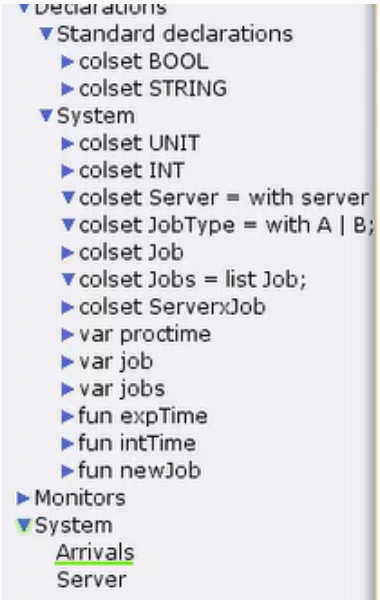


Рис. 8: Декларации системы


Declarations
Standard declarations
colset BOOL
colset STRING
System
colset UNIT
colset INT
colset Server = with server
colset JobType = with A | B;
colset Job
colset Jobs = list Job;
colset ServerxJob
var proctime
var job
var jobs
fun expTime
fun intTime
fun newJob
Monitors
System
Arrivals
Server

Зададим параметры модели на графах сети. – у позиции Queue множество цветов фишек — Jobs; начальная маркировка 1‘[] определяет, что изначально очередь пуста. – у позиции Completed множество цветов фишек — Job(рис.10).

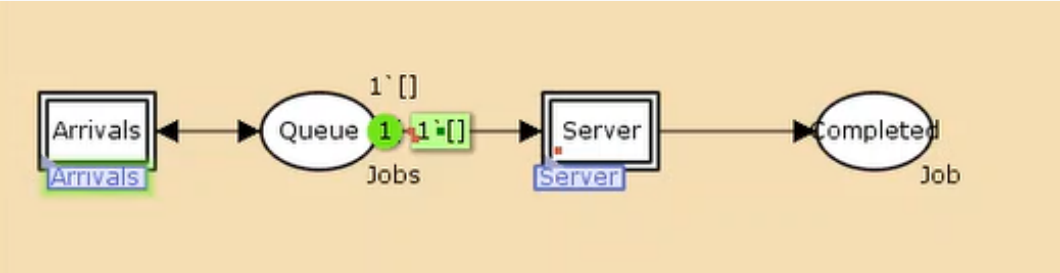


Рис. 10: Параметры элементов основного графа системы обработки заявок в очереди

На листе Arrivals: – у позиции Init: множество цветов фишек — UNIT; начальная маркировка 1‘()(**0?**) определяет, что поступление заявок в систему начинается с нулевого момента времени; – у позиции Next: множество цветов фишек — UNIT; – на дуге от позиции Init к переходу Init выражение () задаёт генерацию заявок; – на дуге от переходов Init и Arrive к позиции Next выражение ()@+expTime(100) задаёт экспоненциальное распределение времени между поступлениями заявок; – на дуге от позиции Next к переходу Arrive выражение () задаёт перемещение фишки; – на дуге от перехода Arrive к позиции Queue выражение jobs^[1] задает поступление заявки в очередь; – на дуге от позиции Queue к переходу Arrive выражение jobs задаёт обратную связь(рис.11).

[1] job

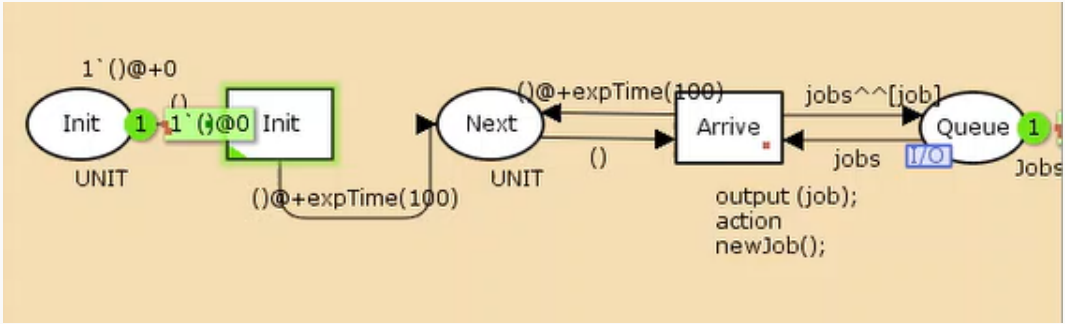
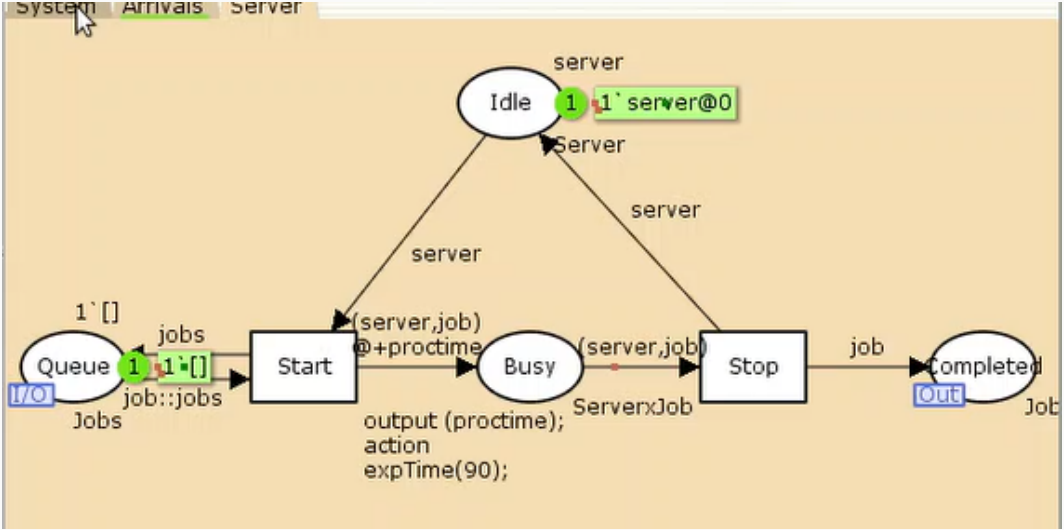


Рис. 11: Параметры элементов генератора заявок системы

На листе Server: – у позиции Busy: множество цветов фишек — Server, начальное значение мар- кировки — 1‘server@0 определяет, что изначально на сервере нет заявок на обслуживание; – у позиции Idle: множество цветов фишек — ServerxJob; – переход Start имеет сегмент кода output (proctime); action expTime(90); определяющий, что время об- служивания заявки распределено по экспоненциальному закону со средним временем обработки в 90 единиц времени; – на дуге от позиции Queue к переходу Start выражение job::jobs определяет, что сервер может начать обработку заявки, если в очереди есть хотя бы одна заявка;

– на дуге от перехода Start к позиции Busy выражение (server,job)@+proctime запускает функцию расчёта времени обработки заяв- ки на сервере; – на дуге от позиции Busy к переходу Stop выражение (server,job) говорит о завершении обработки заявки на сервере; – на дуге от перехода Stop к позиции Completed выражение job показывает, что заявка считается обслуженной; – выражение server на дугах от и к позиции Idle определяет изменение состояние сервера (обрабатывает заявки или ожидает); – на дуге от перехода Start к позиции Queue выражение jobs задаёт обратную связь(рис.12).


System
Arrivals
Server
server
Idle
1
1`server@0
Server
server
server
1`[]
jobs
(server,job)
@+proctime
(server,job)
job
Queue
1
1`[]
Start
Busy
Stop
Completed
I/O
job::jobs
Out
Jobs
ServerxJob
Job
output (proctime);
action
expTime(90);

Запуск модели(рис.13),(рис.14).

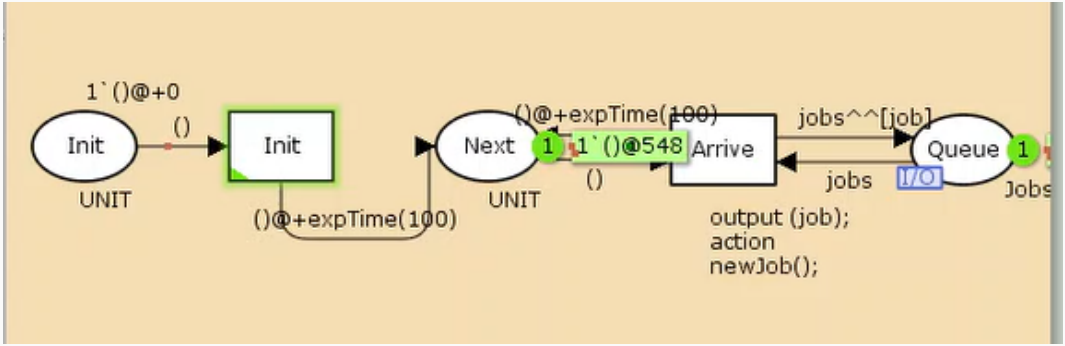


Рис. 13: Запуск модели

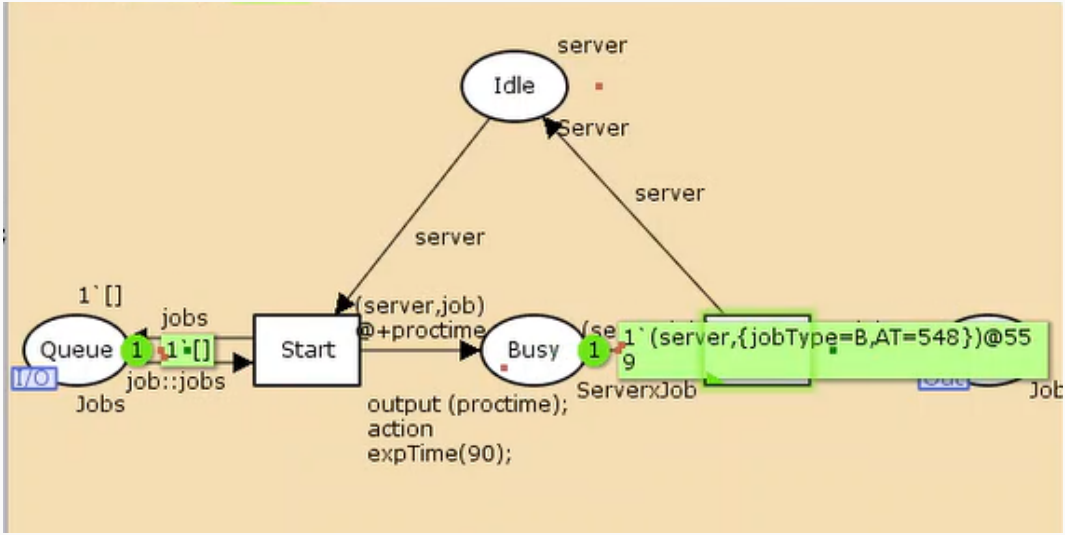


Рис. 14: Запуск модели

Выводы

Научилась работать с Моделью системы массового обслуживания M |M |1