Лабораторная работа №11

Имитационное моделирование

Екатерина Канева, НФИбд-02-22

Содержание

Список иллюстраций

Список таблиц

# 1 Цель работы

Построить модель M|M|1 в CPN Tools.

# 2 Задание

1. Построить модель M|M|1.
2. Выполнить мониторинг параметров модели, построить графики.

# 3 Выполнение лабораторной работы

Модель состояла из 3 листов. Сначала я построила лист System (рис. 1):

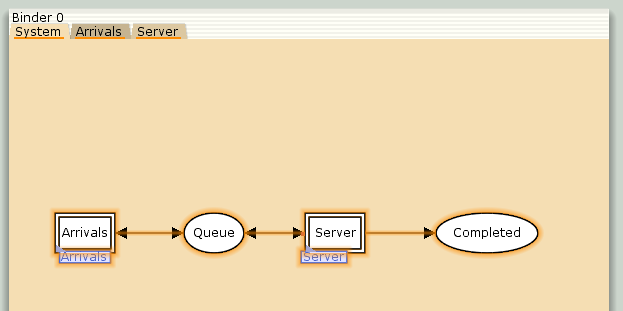


Рис. 1: Лист System.

Далее я задала декларации системы: множества цветов (рис. 2), переменные (рис. 3), функции (рис. 4) модели.

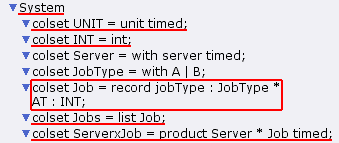


Рис. 2: Множества цветов.

Рис. 3: Переменные.

Рис. 3: Переменные.

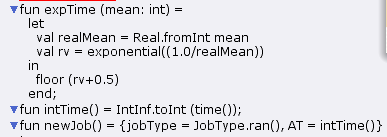


Рис. 4: Фнукции.

Далее я построила листы Arrivals (рис. 5) и Server (рис. 6):

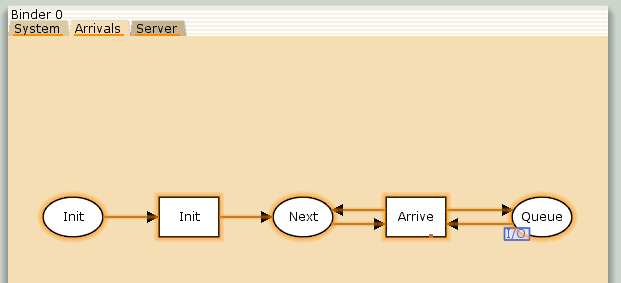


Рис. 5: Лист Arrivals.

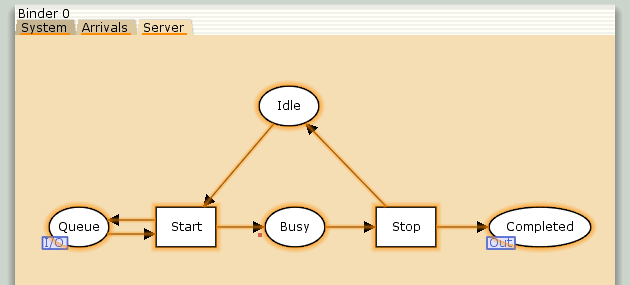


Рис. 6: Лист Server.

Потом я задала параметры модели на графах сети. Там было много параметров, поэтому вся информация будет на картинках (рис. 7, 8 и 9):

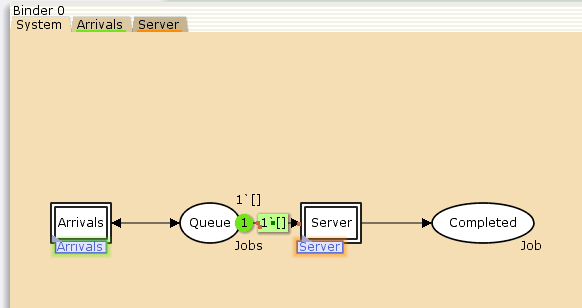


Рис. 7: Лист System после задания параметров на графе.

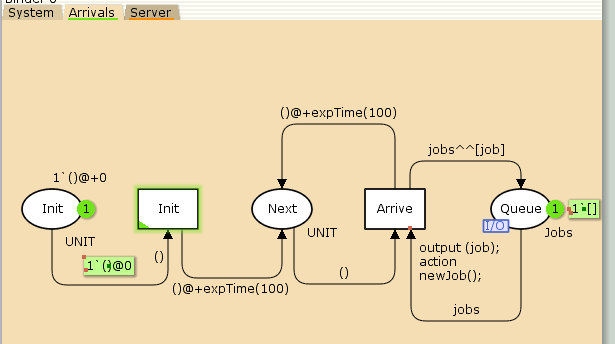


Рис. 8: Лист Arrivals после задания параметров на графе.

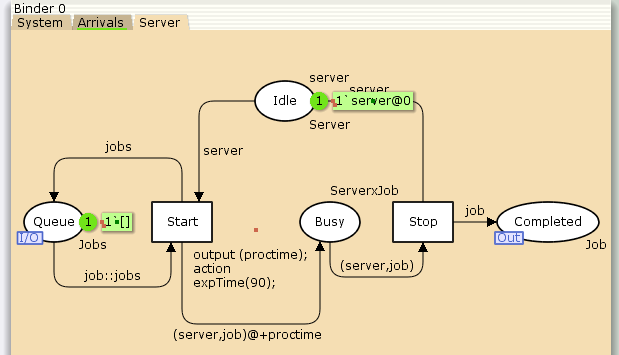


Рис. 9: Лист Server после задания параметров на графе.

Далее я приступила к мониторингу сети. Для этого с помощью палитры Monitoring установила точку останова на переход Start, новый монитор назвала Ostanovka, функцию Predicate изменила на следующую:

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

Потом выбрала Data Coll, установила снова на Start, назвала Queue Delay и изменила функцию Observer на следующую:

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

Далее я запустила моделирование, выполнила более 300 шагов, получила на выводе файл Queue\_Delay.log (рис. 10):

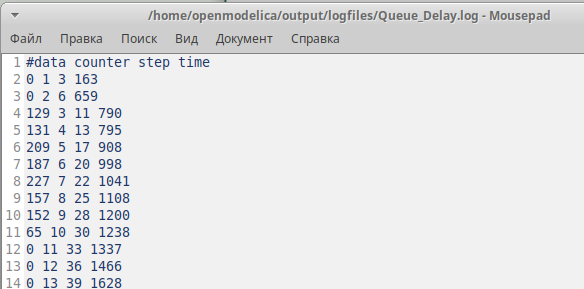


Рис. 10: Содержимое Queue\_Delay.log.

Далее я построила график значений задержки в очереди, для этого я написала следующий код — он также подходит и для графика, который надо будет построить позже (рис. 11):

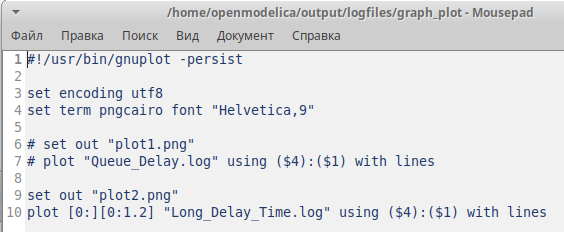


Рис. 11: Скрипт для построения графиков.

Получился следующий график (рис. 12):

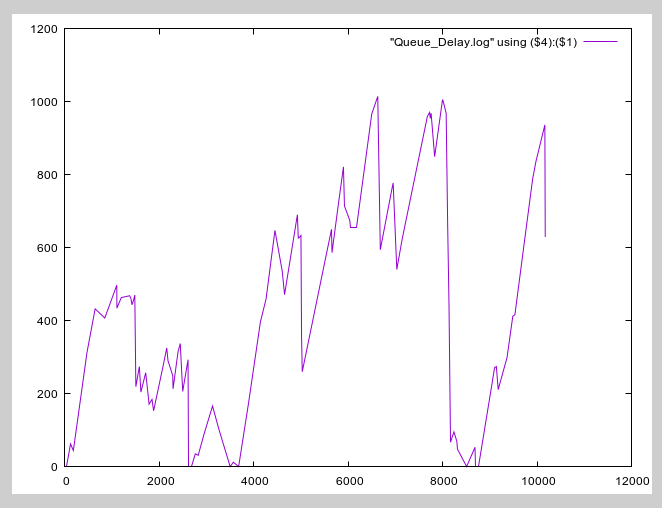


Рис. 12: График значений задержки в очереди.

Далее я с помощью палитры Monitoring снова установила точку останова на переход Start, новый монитор назвала Queue Delay Real, функцию Observer изменила на следующую:

fun obs (bindelem) =  
 let  
 fun obsBindElem (Server'Start (1, {job, jobs, proctime}))  
 = Real.fromInt(intTime() - (#AT job))  
 | obsBindElem \_ = ~1.0  
 in  
 obsBindElem bindelem  
 end

После этого я получила файл Queue\_Delay\_Real.log, похожий на Queue\_Delay.log, только здесь значения имеют действительный тип. После этого я с помощью палитры Monitoring я снова установила Data Coll на Start. Теперь новый монитор я назвала Long Delay Time и сделала следующйю функцию Observer:

fun obs (bindelem) =  
 if IntInf.tiInt(Queue\_Delay.last())>=(!longdelaytime)  
 then 1  
 else 0

Далее было необходимо определить глобальную переменную longdelaytime, которая бы была границей для большой задержки (рис. 13):

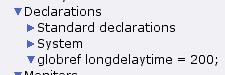


Рис. 13: Глобальная переменная longdelaytime.

После этого я немного изменила скрипт для построения графика в GNU Plot (рис. 11) и построила график (рис. 14), демонстрирующий, в какие периоды времени значения задержки в очереди превышали заданное значение 200:

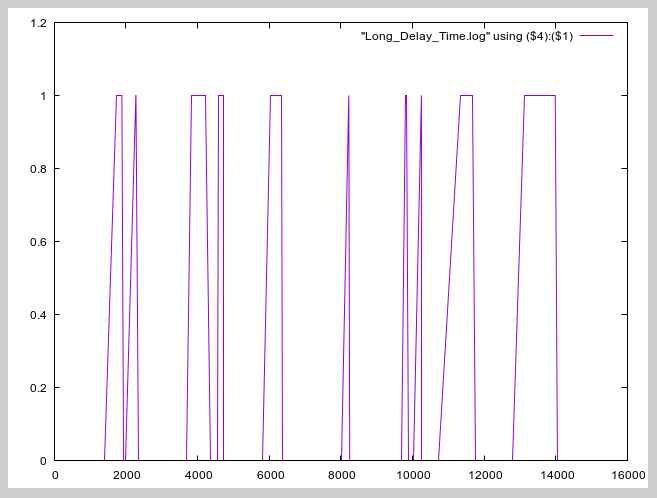


Рис. 14: График периодов превышения задержки.

Когда значение графика равно 1, была задержка больше допустимой, когда 0 — превышения не было. Видим, что если ставить переменную 200, то задержка редко превышала допустимую.

# 4 Выводы

Построили модель M|M|1 в CPN Tools.

# Список литературы