

Лабораторная работа №12

Пример моделирования простого протокола передачи данных

Шияпова Дарина Илдаровна

Содержание

1	Введение	4
1.1	Цели и задачи	4
2	Выполнение лабораторной работы	5
2.1	Упражнение	11
3	Выводы	17

Список иллюстраций

2.1	Задание деклараций	5
2.2	Начальный граф	7
2.3	Добавление промежуточных состояний	9
2.4	Задание деклараций	9
2.5	Модель простого протокола передачи данных	10
2.6	Запуск модели простого протокола передачи данных	10
2.7	Пространство состояний для модели простого протокола передачи данных	16

1 Введение

1.1 Цели и задачи

Цель работы

Реализовать простой протокол передачи данных в CPN Tools.

Задание

- Реализовать простой протокол передачи данных в CPN Tools.
- Вычислить пространство состояний, сформировать отчет о нем и построить граф.

2 Выполнение лабораторной работы

Основные состояния: источник (Send), получатель (Receiver). Действия (переходы): отправить пакет (Send Packet), отправить подтверждение (Send ACK). Промежуточное состояние: следующий посылаемый пакет (NextSend). Зададим декларации модели (рис. 2.1).

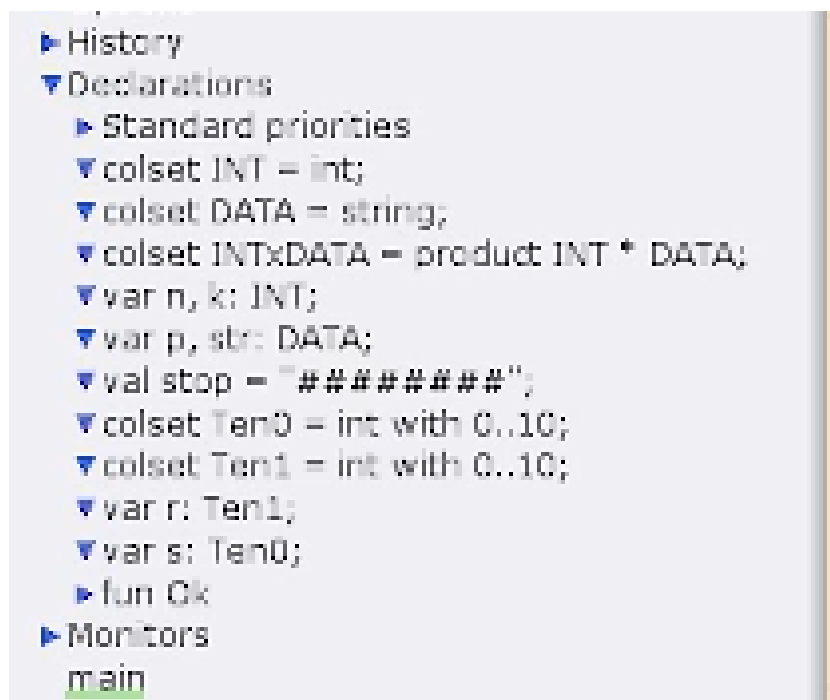


Рис. 2.1: Задание деклараций

Состояние Send имеет тип INTxDATA и следующую начальную маркировку (в соответствии с передаваемой фразой).

Стоповый байт ("#####") определяет, что сообщение закончилось. Состояние Receiver имеет тип DATA и начальное значение 1"" (т.е. пустая строка,

поскольку состояние собирает данные и номер пакета его не интересует). Состояние NextSend имеет тип INT и начальное значение 1'1. Поскольку пакеты представляют собой кортеж, состоящий из номера пакета и строки, то выражение у двусторонней дуги будет иметь значение (n,p). Кроме того, необходимо взаимодействовать с состоянием, которое будет сообщать номер следующего посылаемого пакета данных. Поэтому переход Send Packet соединяем с состоянием NextSend двумя дугами с выражениями n (рис. 12.1). Также необходимо получать информацию с подтверждениями о получении данных. От перехода Send Packet к состоянию NextSend дуга с выражением n, обратно – k.

Построим начальный граф(рис. 2.2):

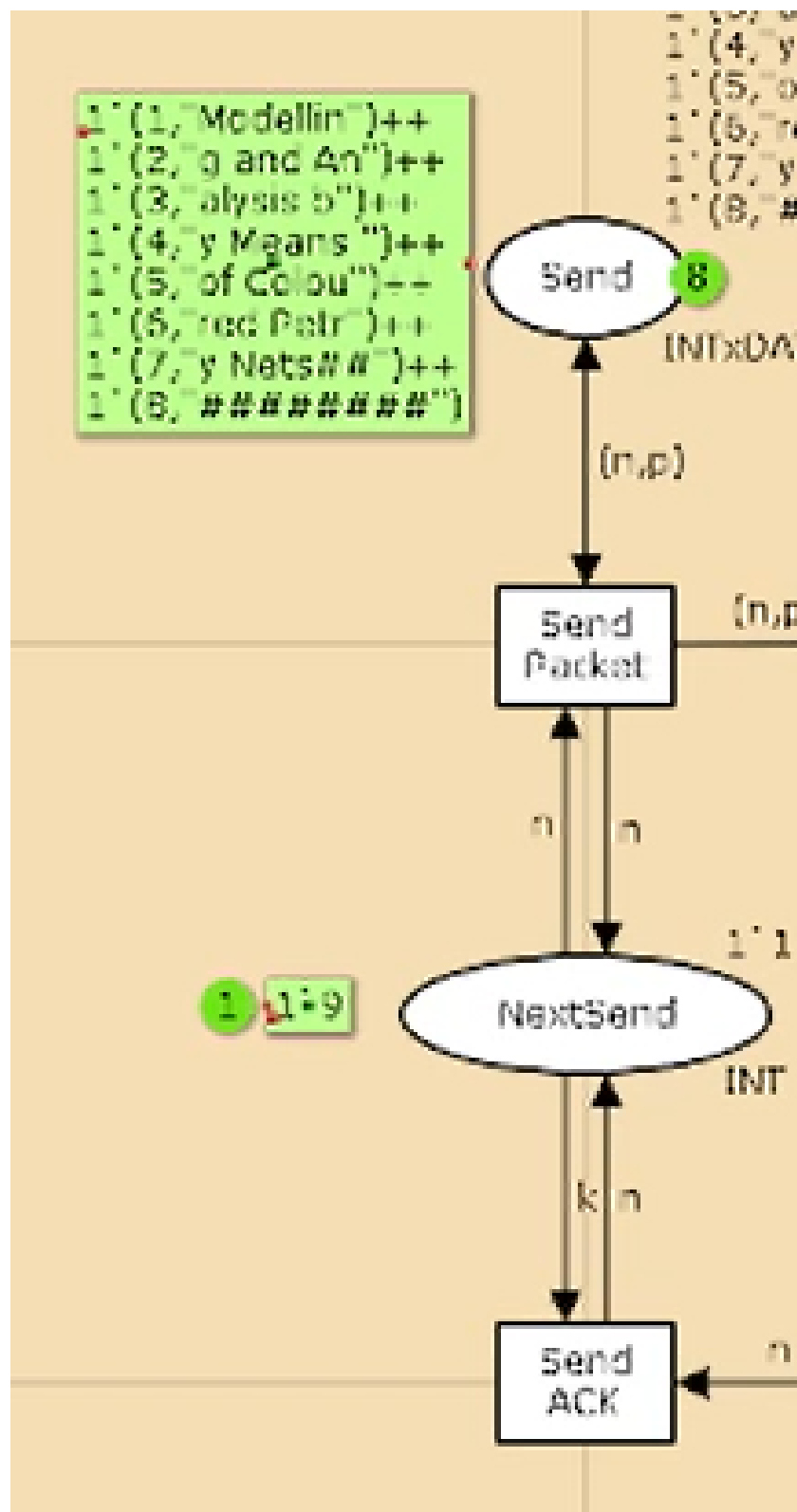


Рис. 2.2: Начальный граф

Зададим промежуточные состояния (А, В с типом INTxDATA, С, D с типом INTxDATA) для переходов (рис. 12.2): передать пакет Transmit Packet (передаём (n,p)), передать подтверждение Transmit ACK (передаём целое число k). Добавляем переход получения пакета (Receive Packet). От состояния Receiver идёт дуга к переходу Receive Packet со значением той строки (str), которая находится в состоянии Receiver. Обратно: проверяем, что номер пакета новый и строка не равна стоп-биту. Если это так, то строку добавляем к полученным данным. Кроме того, необходимо знать, каким будет номер следующего пакета. Для этого добавляем состояние NextRec с типом INT и начальным значением 1'1 (один пакет), связываем его дугами с переходом Receive Packet. Причём к переходу идёт дуга с выражением k, от перехода — if n=k then k+1 else k. Связываем состояния В и С с переходом Receive Packet. От состояния В к переходу Receive Packet — выражение (n,p), от перехода Receive Packet к состоянию С — выражение if n=k then k+1 else k. От перехода Receive Packet к состоянию Receiver: if n=k and also p<>stop then str^p else str. (если n=k и мы не получили стоп-байт, то направляем в состояние строку и к ней прикрепляем p, в противном случае посылаем только строку). На переходах Transmit Packet и Transmit ACK зададим потерю пакетов. Для этого на интервале от 0 до 10 зададим пороговое значение и, если передаваемое значение превысит этот порог, то считаем, что произошла потеря пакета, если нет, то передаём пакет дальше. Для этого задаём вспомогательные состояния SP и SA с типом Ten0 и начальным значением 1'8, соединяем с соответствующими переходами (рис. 2.3):

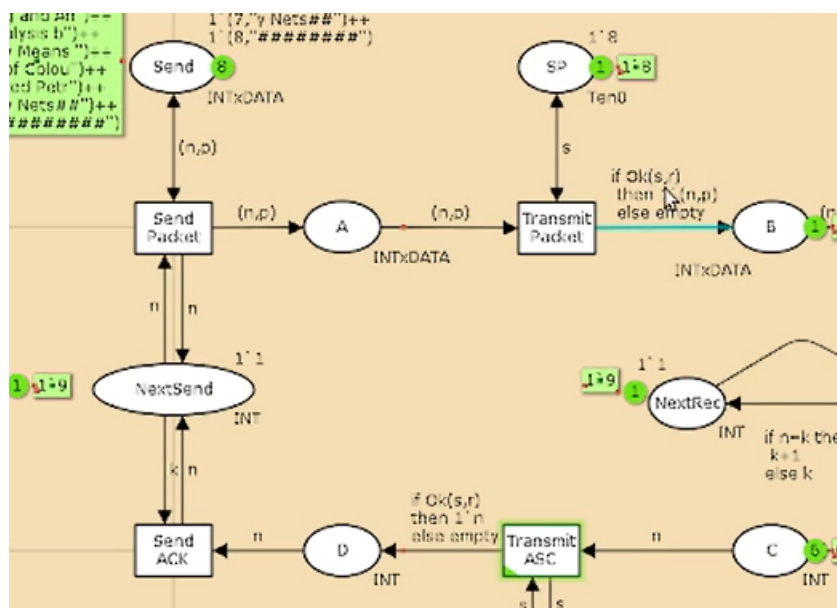


Рис. 2.3: Добавление промежуточных состояний

В декларациях задаём(рис. 2.4):

```

var s: Ten0 = 0;
colset Ten0 = int with 0..10;
colset Ten1 = int with 0..10;
var r: Ten1;
var s: Ten0;
fun Ok(s: Ten0, r: Ten1) = (r <= s)
Monitors

```

Рис. 2.4: Задание деклараций

Таким образом, получим модель простого протокола передачи данных (рис. 12.3). Пакет последовательно проходит: состояние Send, переход Send Packet, состояние A, с некоторой вероятностью переход Transmit Packet, состояние B, попадает на переход Receive Packet, где проверяется номер пакета и если нет совпадения, то пакет направляется в состояние Received, а номер пакета передаётся последовательно в состояние C, с некоторой вероятностью в переход Transmit ACK, далее в состояние D, переход Receive ACK, состояние NextSend (увеличивая

на 1 номер следующего пакета), переход Send Packet. Так продолжается до тех пор, пока не будут переданы все части сообщения. Последней будет передана стоп-последовательность(рис. 2.5):

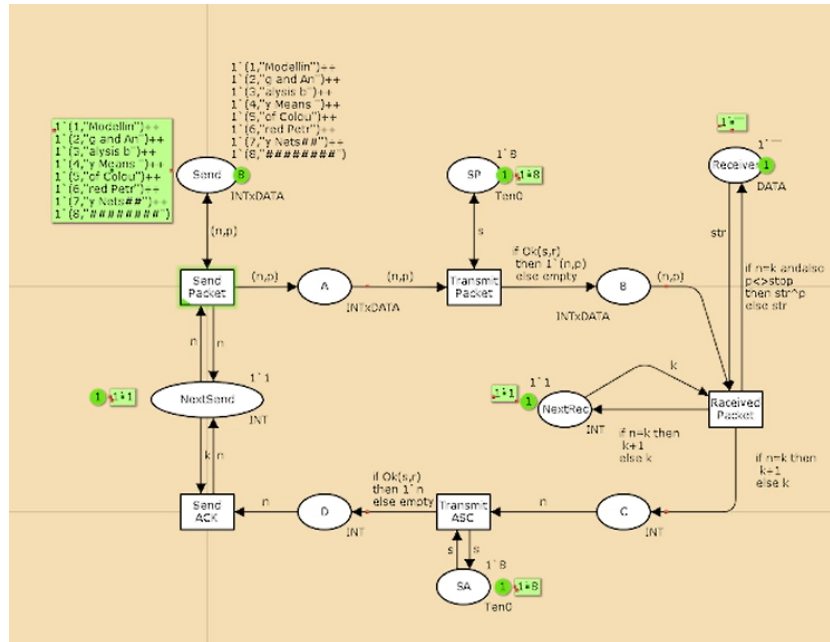


Рис. 2.5: Модель простого протокола передачи данных

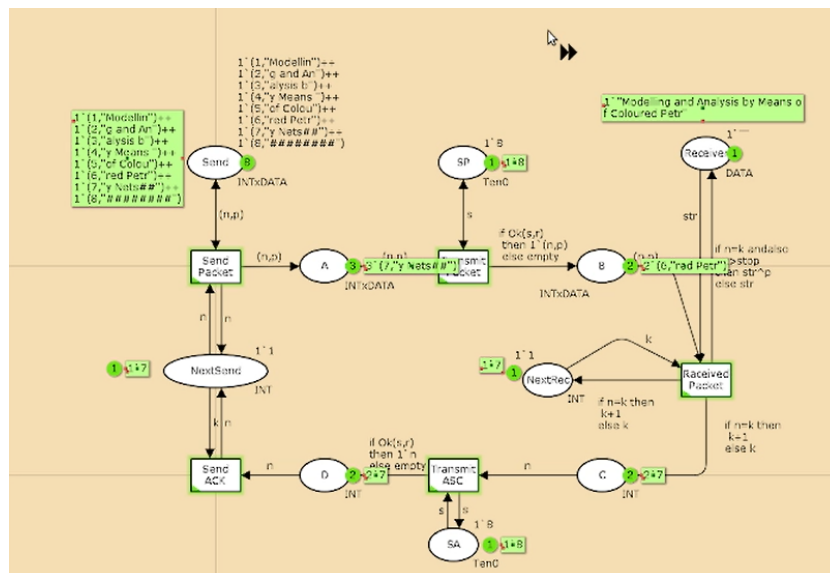


Рис. 2.6: Запуск модели простого протокола передачи данных

2.1 Упражнение

Вычислим пространство состояний. Прежде, чем пространство состояний может быть вычислено и проанализировано, необходимо сформировать код пространства состояний. Этот код создается, когда используется инструмент Войти в пространство состояний. Вход в пространство состояний занимает некоторое время. Затем, если ожидается, что пространство состояний будет небольшим, можно просто применить инструмент Вычислить пространство состояний к листу, содержащему страницу сети. Сформируем отчет о пространстве состояний и проанализируем его. Чтобы сохранить отчет, необходимо применить инструмент Сохранить отчет о пространстве состояний к листу, содержащему страницу сети и ввести имя файла отчета.

Из него можно увидеть:

- 13341 состояний и 206461 переходов между ними.
- Указаны границы значений для каждого элемента: промежуточные состояния A, B, C(наибольшая верхняя граница у A, так как после него пакеты отбрасываются. Так как мы установили максимум 10, то у следующего состояния B верхняя граница – 10), вспомогательные состояния SP, SA, NextRec, NextSend, Receiver(в них может находиться только один пакет) и состояние Send(в нем хранится только 8 элементов, так как мы задали их в начале и с ними никаких изменений не происходит).
- Указаны границы в виде мультимножеств.
- Маркировка home для всех состояний (в любую позицию можно попасть из любой другой маркировки).
- Маркировка dead равная 4675 [9999,9998,9997,9996,9995,...] – это состояния, в которых нет включенных переходов.

Statistics

State Space

Nodes: 19641
Arcs: 312965
Secs: 300
Status: Partial

Scc Graph

Nodes: 10323
Arcs: 269857
Secs: 20

Boundedness Properties

Best Integer Bounds

	Upper	Lower
Main'A 1	20	0
Main'B 1	10	0
Main'C 1	6	0
Main'D 1	5	0
Main'NextRec 1	1	1
Main'NextSend 1	1	1
Main'Reciever 1	1	1
Main'SA 1	1	1
Main'SP 1	1	1
Main'Send 1	8	8

Best Upper Multi-set Bounds

Main'A 1	20`(1,"Modellin")++
15`(2,"g and An")++	
9`(3,"alysis b")++	
4`(4,"y Means ")	
Main'B 1	10`(1,"Modellin")++
7`(2,"g and An")++	
4`(3,"alysis b")++	
2`(4,"y Means ")	
Main'C 1	6`2++
5`3++	
3`4++	
1`5	
Main'D 1	5`2++
3`3++	
2`4++	
1`5	
Main'NextRec 1	1`1++
1`2++	
1`3++	
1`4++	
1`5	
Main'NextSend 1	1`1++
1`2++	
1`3++	
1`4	
Main'Reciever 1	1`""++
1`"Modellin"++	

```

1`"Modelling and An"++
1`"Modelling and Analysis b"++
1`"Modelling and Analysis by Means "
    Main'SA 1          1`8
    Main'SP 1          1`8
    Main'Send 1        1`(1,"Modellin")++
1`(2,"g and An")++
1`(3,"alysis b")++
1`(4,"y Means ")++
1`(5,"of Colou")++
1`(6,"red Petr")++
1`(7,"y Nets##")++
1`(8,"#####")

```

Best Lower Multi-set Bounds

```

    Main'A 1          empty
    Main'B 1          empty
    Main'C 1          empty
    Main'D 1          empty
    Main'NextRec 1     empty
    Main'NextSend 1    empty
    Main'Reciever 1    empty
    Main'SA 1          1`8
    Main'SP 1          1`8
    Main'Send 1        1`(1,"Modellin")++
1`(2,"g and An")++
1`(3,"alysis b")++
1`(4,"y Means ")++
1`(5,"of Colou")++

```

```
1`(6,"red Petr")++  
1`(7,"y Nets##")++  
1`(8,"#####")
```

Home Properties

Home Markings

None

Liveness Properties

Dead Markings

4675 [9999,9998,9997,9996,9995,...]

Dead Transition Instances

None

Live Transition Instances

None

Fairness Properties

Main'Recieved_Packet 1 No Fairness

Main'Send_ACK 1 No Fairness

Main'Send_Packet 1 Impartial
Main'Transmit_ACK 1 No Fairness
Main'Transmit_Packet 1 Impartial

Сформируем начало графа пространства состояний, так как их много(рис. 2.7):

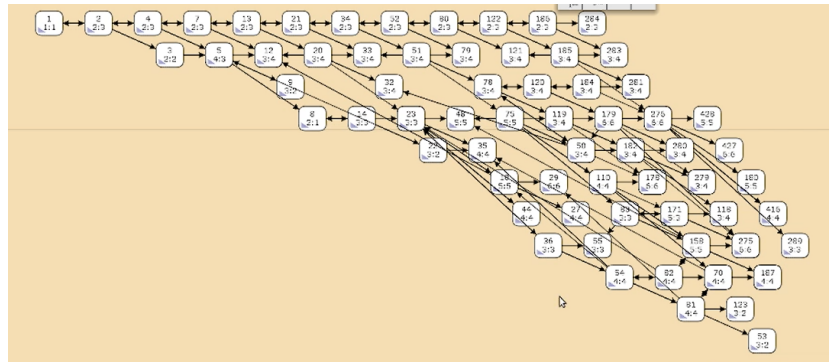


Рис. 2.7: Пространство состояний для модели простого протокола передачи данных

3 Выводы

В процессе выполнения данной лабораторной работы я реализовала простой протокол передачи данных в CPN Tools и проведен анализ его пространства состояний.