Лабораторная работа № 12

Пример моделирования простого протокола передачи данных

Демидова Екатерина Алексеевна

Содержание

# 1 Введение

## 1.1 Цели и задачи

**Цель работы**

Реализовать в CPN Tools простой протокол передачи данных и провести анализ.

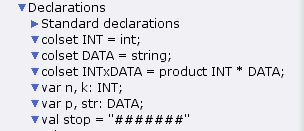
**Задание**

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

# 2 Выполнение лабораторной работы

## 2.1 Реализация задачи в CPN Tools

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. [??]).

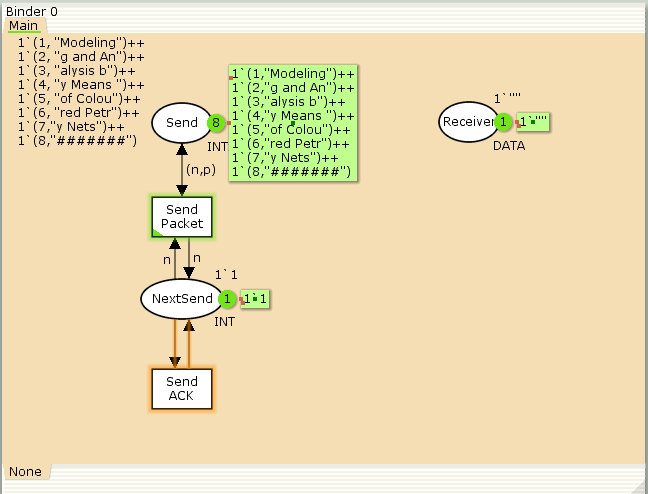


Задание деклараций

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

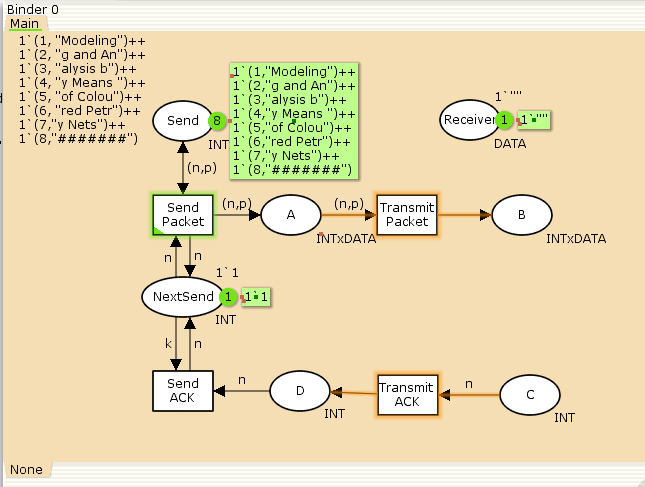
Стоповый байт (“########”) определяет, что сообщение закончилось. Состояние Receiver имеет тип DATA и начальное значение 1’“” (т.е. пустая строка, поскольку состояние собирает данные и номер пакета его не интересует). Состояние NextSend имеет тип INT и начальное значение 1’1. Поскольку пакеты представляют 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. [??]):



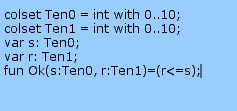
Начальный граф

Зададим промежуточные состояния (A, B с типом INTxDATA, C, 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 B и C с переходом Receive Packet. От состояния B к переходу Receive Packet — выражение (n,p), от перехода Receive Packet к состоянию C — выражение if n=k then k+1 else k. От перехода Receive Packet к состоянию Receiver: if n=k andalso 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 толко 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. [??]):



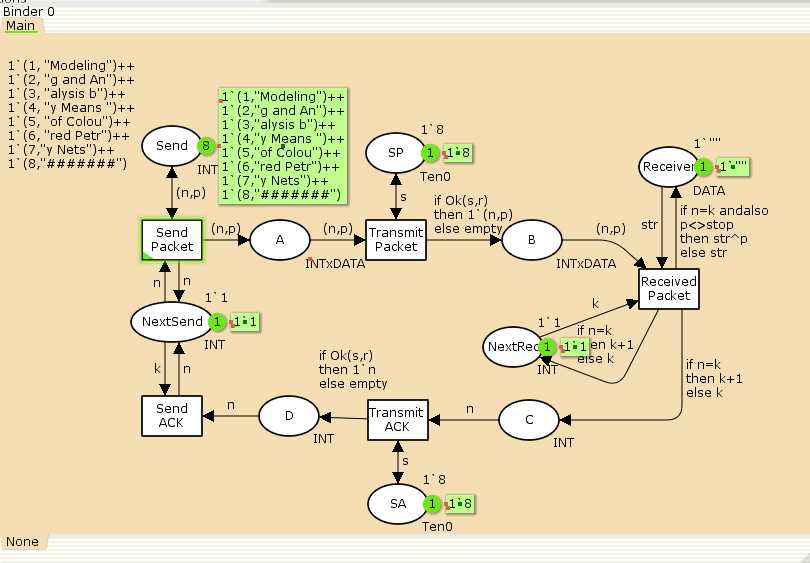
Добавление промежуточных состояний

В декларациях задаём(рис. [??]):



Задание деклараций

Таким образом, получим модель простого протокола передачи данных (рис. 12.3). Пакет последовательно проходит: состояние Send, переход Send Packet, состоя- ние 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 после- довательно 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. [??]):



Модель простого протокола передачи данных

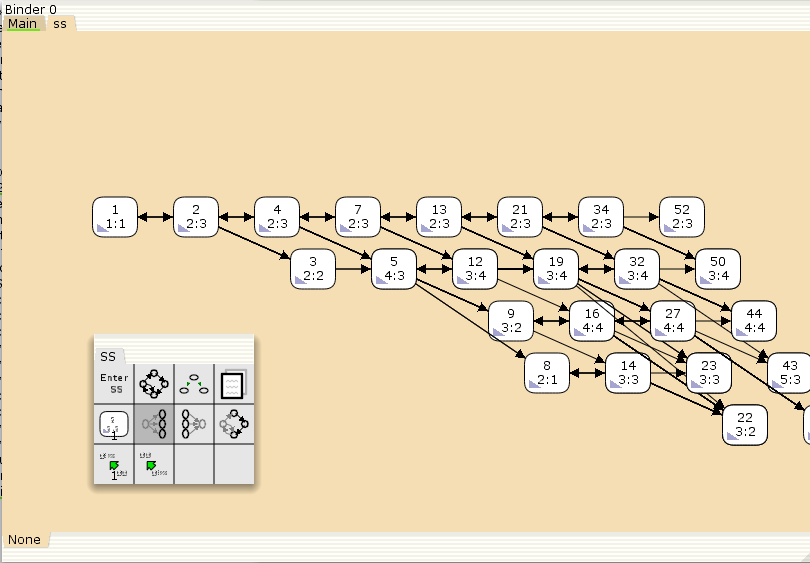
## 2.2 Пространство состояний

Затем сформируем отчет пространства состояний. Из него может увидеть:

* есть 31174 состояний и 520806 переходов между ними, в графе строго соединенных компонент 16375 узлов и 438014 дуг.
* Затем 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).
* Также указаны границы в виде мультимножеств.
* Маркировка home для всех состояний, так как в любую позицию мы можем попасть из любой другой маркировки.
* Маркировка dead равная 10991 [31174,31173,31172,31171,31170,…] – это состояния, в которых нет включенных переходов(связано с условными конструкциями).
* В конце указано, что бесконечно часто могут происходит(Impartial Transition Instances)ь события Send\_Packet и Transmit\_Packet(они позволяют сети всегда передавать данные). Также указаны Transition Instances with No Fairness: Send\_ACK, Transmit\_ACK, Received\_Packet. В них возможны бесконечные последовательности, но не срабатывают.

CPN Tools state space report for:  
/home/openmodelica/lab12.cpn  
Report generated: Fri May 17 01:59:51 2024  
  
  
 Statistics  
------------------------------------------------------------------------  
  
 State Space  
 Nodes: 31174  
 Arcs: 520806  
 Secs: 300  
 Status: Partial  
  
 Scc Graph  
 Nodes: 16375  
 Arcs: 438014  
 Secs: 12  
  
  
 Boundedness Properties  
------------------------------------------------------------------------  
  
 Best Integer Bounds  
 Upper Lower  
 Main'A 1 21 0  
 Main'B 1 10 0  
 Main'C 1 7 0  
 Main'D 1 5 0  
 Main'NextRec 1 1 1  
 Main'NextSend 1 1 1  
 Main'Receiver 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 21`(1,"Modeling")++  
17`(2,"g and An")++  
12`(3,"alysis b")++  
7`(4,"y Means ")++  
2`(5,"of Colou")  
 Main'B 1 10`(1,"Modeling")++  
8`(2,"g and An")++  
6`(3,"alysis b")++  
3`(4,"y Means ")++  
1`(5,"of Colou")  
 Main'C 1 7`2++  
5`3++  
4`4++  
2`5  
 Main'D 1 5`2++  
4`3++  
3`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++  
1`5  
 Main'Receiver 1 1`""++  
1`"Modeling"++  
1`"Modelingg and An"++  
1`"Modelingg and Analysis b"++  
1`"Modelingg and Analysis by Means "  
 Main'SA 1 1`8  
 Main'SP 1 1`8  
 Main'Send 1 1`(1,"Modeling")++  
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'Receiver 1 empty  
 Main'SA 1 1`8  
 Main'SP 1 1`8  
 Main'Send 1 1`(1,"Modeling")++  
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  
 10991 [31174,31173,31172,31171,31170,...]  
  
 Dead Transition Instances  
 None  
  
 Live Transition Instances  
 None  
  
  
 Fairness Properties  
------------------------------------------------------------------------  
 Main'Received\_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. [??]):



Пространство состояний для модели простого протокола передачи данных

# 3 Выводы

В результате выполнения работы был реализован в CPN Tools простой протокол передачи данных и проведен анализ его пространства состояний.