Лабораторная работа №12

Пример моделирования простого протокола передачи данных

Кадров Виктор Максимович

Содержание

# 1 Введение

## 1.1 Цели и задачи

**Цель работы**

Реализовать в *CPN Tools* простой протокол передачи данных и провести анализ[1].

**Задание**

* Реализовать в *CPN Tools* простой протокол передачи данных[2].
* 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. 1).

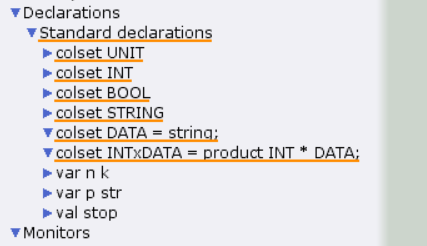


Рис. 1: Задание деклараций

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.

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):

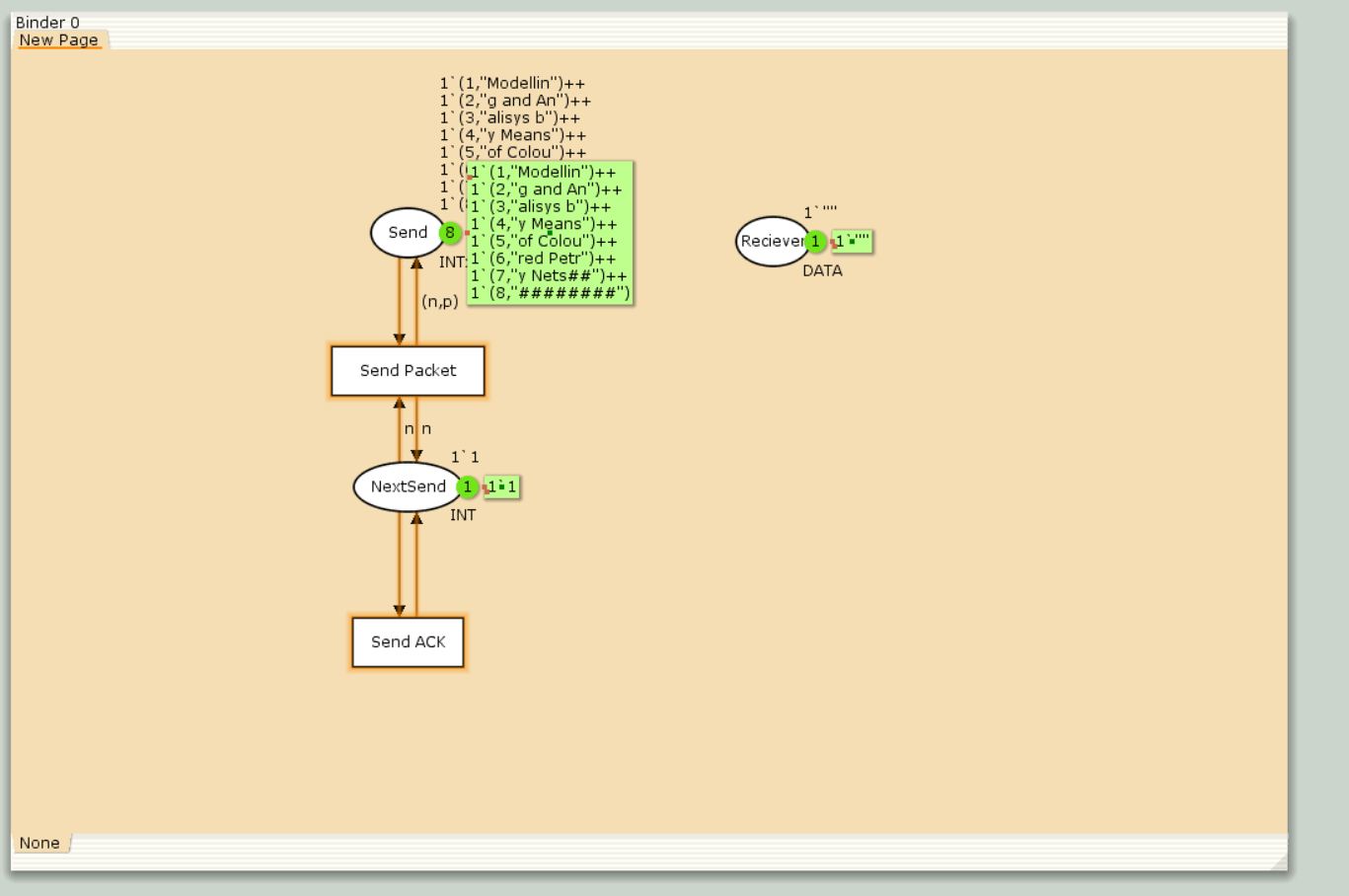


Рис. 2: Начальный граф

Зададим промежуточные состояния (A, B с типом INTxDATA, C, D с типом INTxDATA) для переходов: 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. 3):

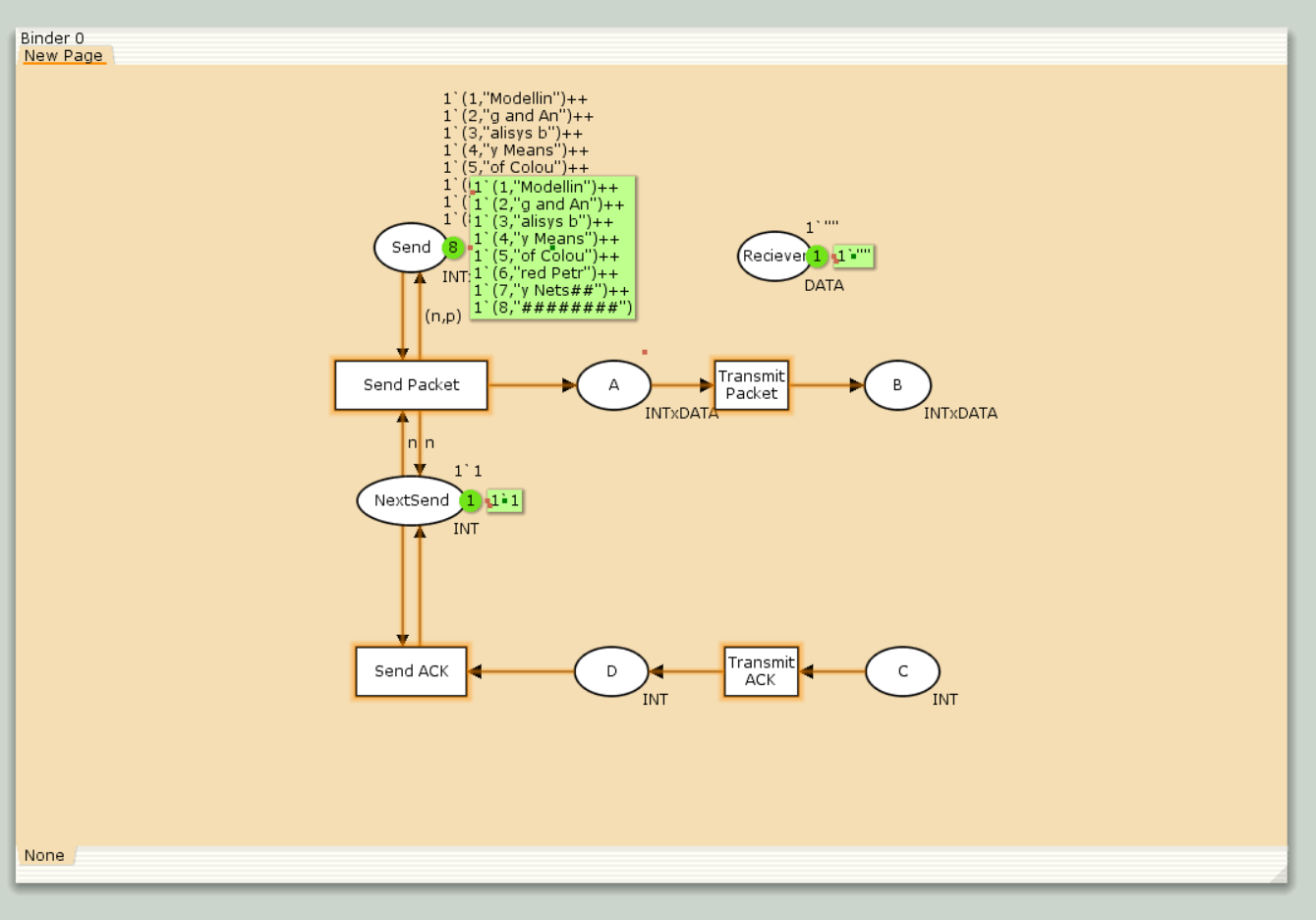


Рис. 3: Добавление промежуточных состояний

В декларациях задаём(рис. 4):

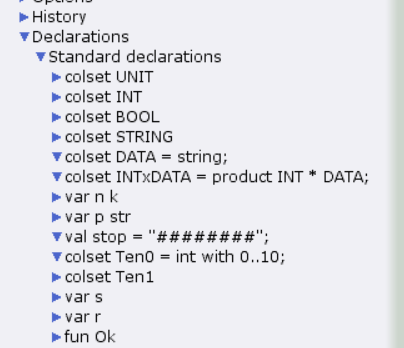


Рис. 4: Задание деклараций

Таким образом, получим модель простого протокола передачи данных. 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. 5):

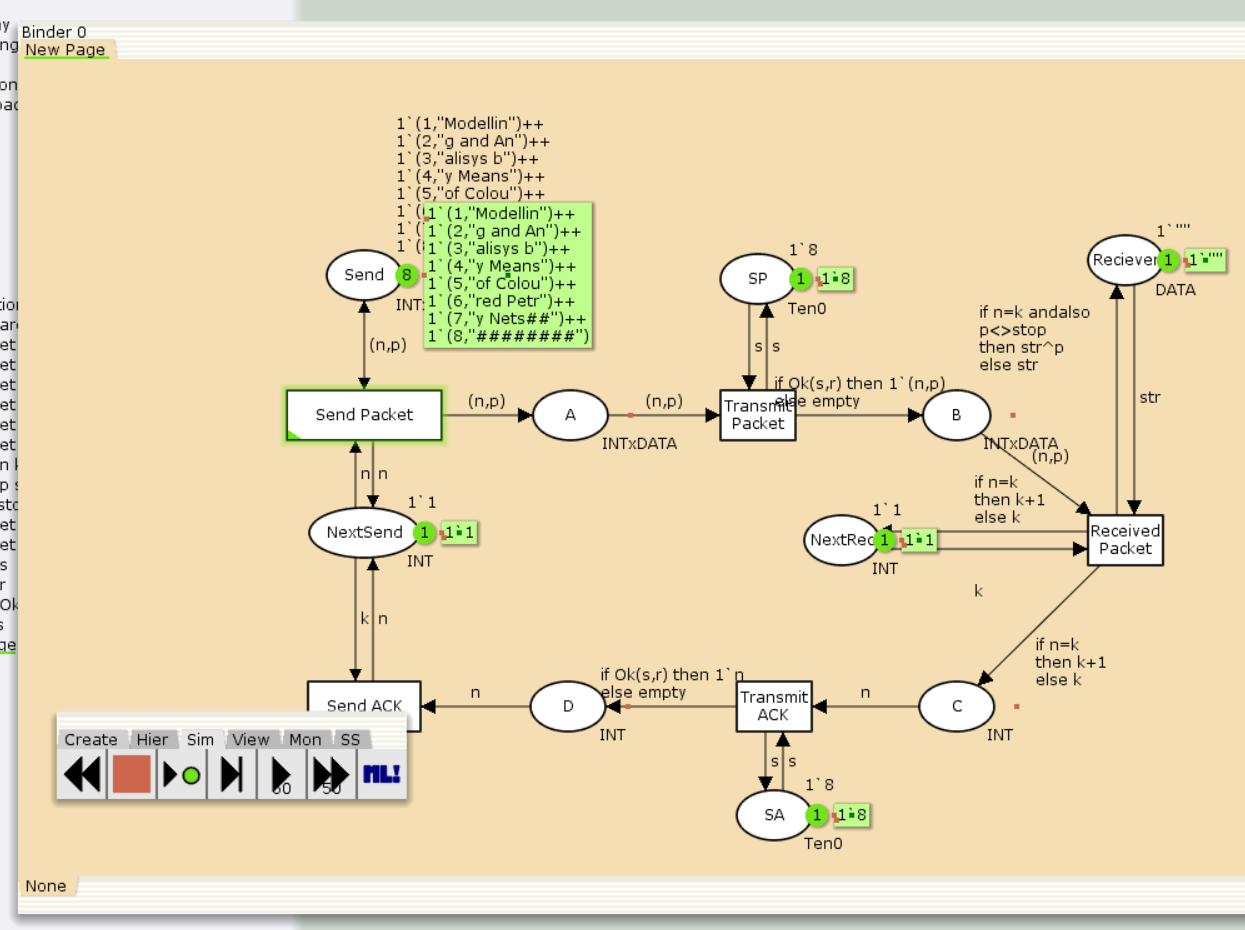


Рис. 5: Модель простого протокола передачи данных

Запустим модель(рис. 6).

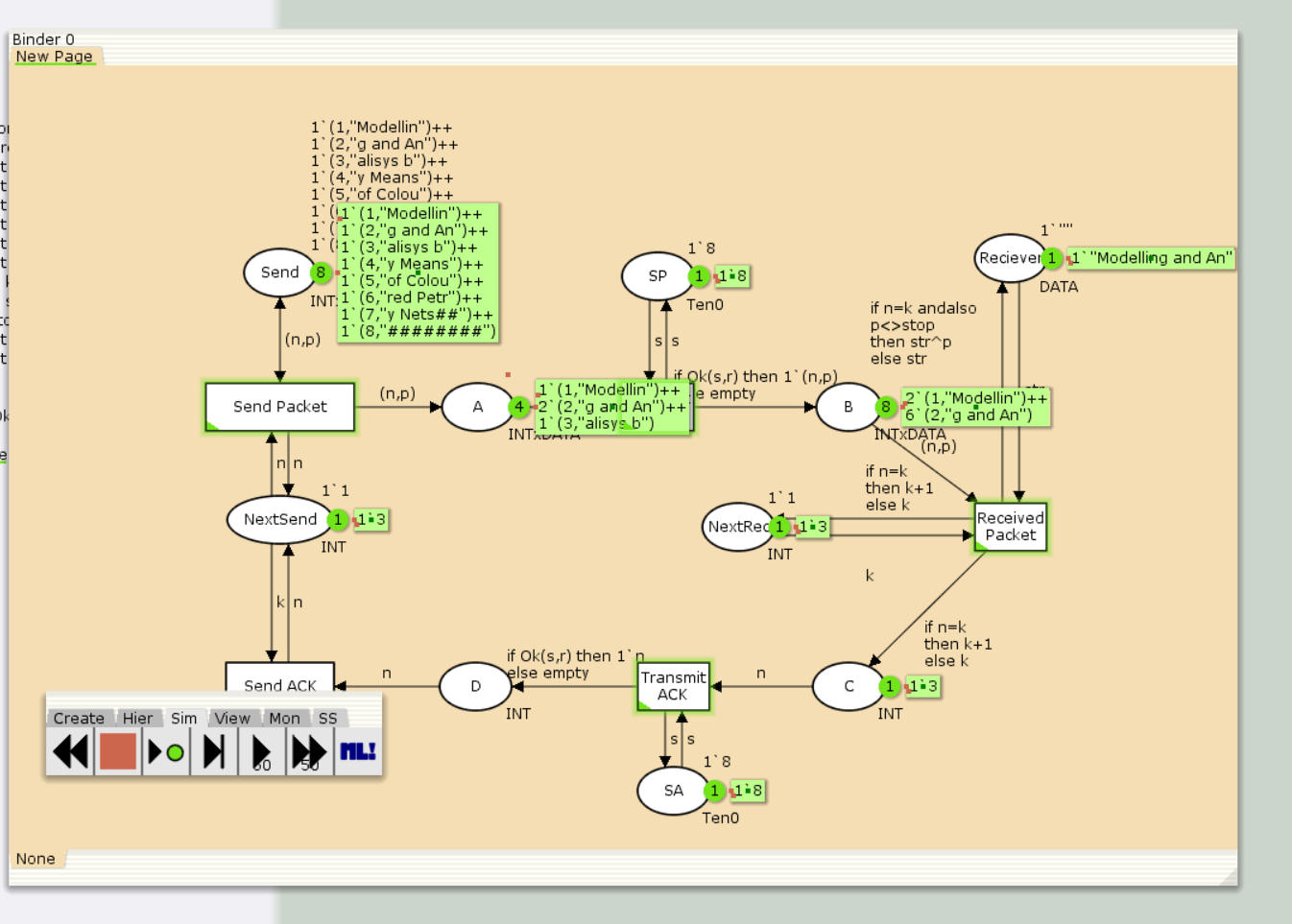


Рис. 6: Запуск модели простого протокола передачи данных

## 2.2 Пространство состояний

Затем сформируем отчет пространства состояний. Из него может увидеть:

* есть 15804 состояния и 247803 перехода между ними, в графе строго соединенных компонент 9348 узлов и 211289 дуг.
* Затем указаны границы значений для каждого элемента: промежуточные состояния A, B, C, D(наибольшая верхняя граница у A, так как после него пакеты отбрасываются), 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 Markings равная None для всех состояний, так как модель где-то завершается и не входит в бесконечный цикл, при любых обстоятельствах во время выполнения модели процесса всегда можно достичь маркировки, где успешно передаётся вся информация и принимается подходящее решение.
* Маркировка dead равная 6555 [9999,9998,9997,9996,9995,…] - это состояния, в которых не активированы никакие переходы.
* В конце указано, что бесконечно часто могут происходить(Impartial Transition Instances) 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. Это означает, что существует последовательность бесконечных срабатываний, в которой переход непрерывно включён с определённого момента, но больше не срабатывает. .

CPN Tools state space report for:  
/home/openmodelica/Desktop/lab12.cpn  
Report generated: Sat Apr 26 22:52:29 2025  
  
  
 Statistics  
------------------------------------------------------------------------  
  
 State Space  
 Nodes: 15804  
 Arcs: 247803  
 Secs: 300  
 Status: Partial  
  
 Scc Graph  
 Nodes: 9348  
 Arcs: 211289  
 Secs: 13  
  
  
 Boundedness Properties  
------------------------------------------------------------------------  
  
 Best Integer Bounds  
 Upper Lower  
 lab\_12'A 1 12 0  
 lab\_12'B 1 7 0  
 lab\_12'C 1 5 0  
 lab\_12'D 1 7 0  
 lab\_12'NextRec 1 1 1  
 lab\_12'NextSend 1 1 1  
 lab\_12'Receiver 1 1 1  
 lab\_12'SA 1 1 1  
 lab\_12'SP 1 1 1  
 lab\_12'Send 1 8 8  
  
 Best Upper Multi-set Bounds  
 lab\_12'A 1 12`(2,"g and An")++  
9`(3,"alysis b")++  
4`(4,"y Means ")  
 lab\_12'B 1 7`(2,"g and An")++  
4`(3,"alysis b")++  
2`(4,"y Means ")  
 lab\_12'C 1 5`3++  
4`4++  
2`5  
 lab\_12'D 1 1`2++  
6`3++  
3`4++  
1`5  
 lab\_12'NextRec 1 1`3++  
1`4++  
1`5  
 lab\_12'NextSend 1 1`2++  
1`3++  
1`4  
 lab\_12'Receiver 1 1`"Modelling and An"++  
1`"Modelling and Analysis b"++  
1`"Modelling and Analysis by Means "  
 lab\_12'SA 1 1`8  
 lab\_12'SP 1 1`8  
 lab\_12'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  
 lab\_12'A 1 empty  
 lab\_12'B 1 empty  
 lab\_12'C 1 empty  
 lab\_12'D 1 empty  
 lab\_12'NextRec 1 empty  
 lab\_12'NextSend 1 empty  
 lab\_12'Receiver 1 empty  
 lab\_12'SA 1 1`8  
 lab\_12'SP 1 1`8  
 lab\_12'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  
 6555 [9999,9998,9997,9996,9995,...]  
  
 Dead Transition Instances  
 None  
  
 Live Transition Instances  
 None  
  
  
 Fairness Properties  
------------------------------------------------------------------------  
 lab\_12'Received\_Packet 1  
 No Fairness  
 lab\_12'Send\_ACK 1 No Fairness  
 lab\_12'Send\_Packet 1 Impartial  
 lab\_12'Transmit\_ACK 1 No Fairness  
 lab\_12'Transmit\_Packet 1  
 Impartial

Сформируем начало графа пространства состояний, так как их много(рис. 7):

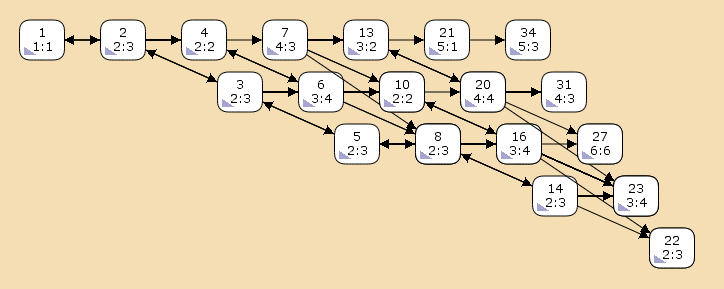


Рис. 7: Пространство состояний для модели простого протокола передачи данных

# 3 Выводы

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

# Список литературы

1. Королькова А.В., Кулябов Д.С. Лабораторная работа 12. Пример моделирования простого протокола передачи данных [Электронный ресурс].

2. Королькова А.В., Кулябов Д.С. Сети Петри. Моделирование в CPN Tools [Электронный ресурс].