

# **Лабораторная работа № 12**

**Пример моделирования простого протокола передачи данных**

Шуплецов Александр Андреевич

# Содержание

<b>1</b>	<b>Введение</b>	<b>4</b>
1.1	Цели и задачи . . . . .	4
<b>2</b>	<b>Выполнение лабораторной работы</b>	<b>5</b>
2.1	Реализация задачи в CPN Tools . . . . .	5
<b>3</b>	<b>Выводы</b>	<b>10</b>

## Список иллюстраций

2.1	Задание деклараций . . . . .	5
2.2	Начальный граф . . . . .	6
2.3	Добавление промежуточных состояний . . . . .	7
2.4	Задание деклараций . . . . .	7
2.5	Модель простого протокола передачи данных . . . . .	8
2.6	Пространство состояний для модели простого протокола передачи данных . . . . .	9

# 1 Введение

## 1.1 Цели и задачи

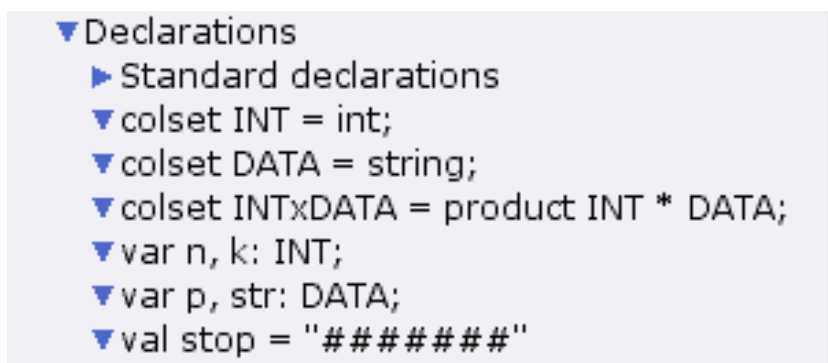
### Цель работы

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

## 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.

A screenshot of the 'Declarations' window in CPN Tools. The window has a light blue background and a tree view on the left. The tree view shows 'Declarations' expanded, with 'Standard declarations' selected. The main area displays the following declarations:

```
▼ Declarations
  ► Standard declarations
    ▼ colset INT = int;
    ▼ colset DATA = string;
    ▼ colset INTxDATA = product INT * DATA;
    ▼ var n, k: INT;
    ▼ var p, str: DATA;
    ▼ val stop = "#####"
```

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 выраже-

ние 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:

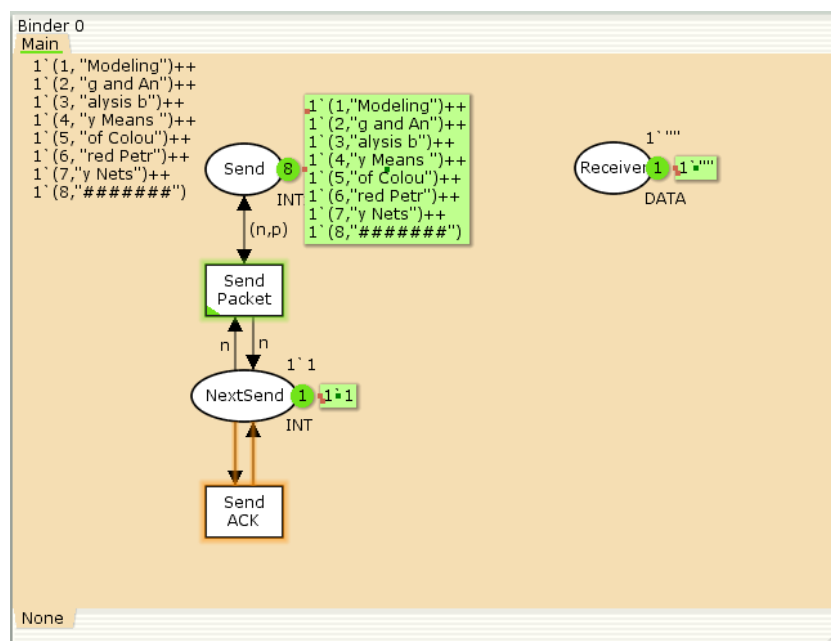


Рис. 2.2: Начальный граф

Зададим промежуточные состояния (A, B с типом INTxDATA, C, D с типом INTxDATA) для переходов. На переходах Transmit Packet и Transmit ACK зададим потерю пакетов. Для этого на интервале от 0 до 10 зададим пороговое значение  $i$ , 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:







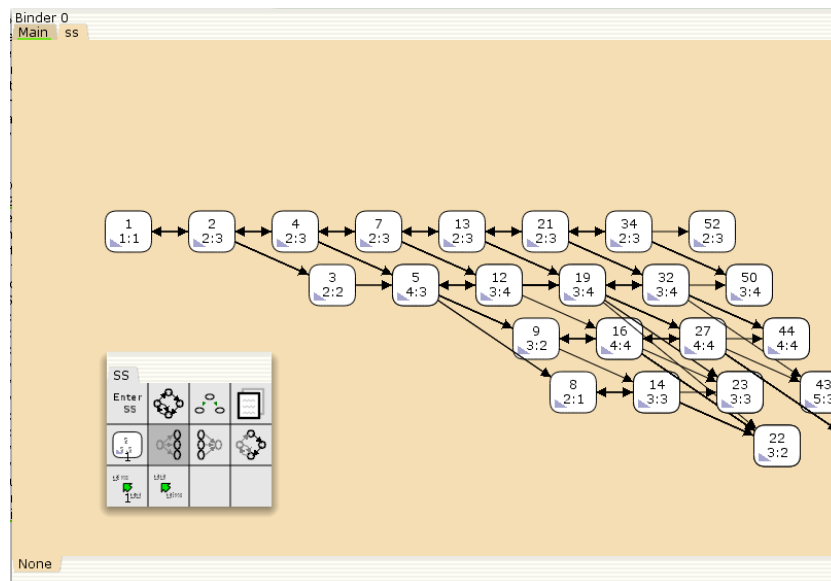


Рис. 2.6: Пространство состояний для модели простого протокола передачи данных

## **3 Выводы**

В результате выполнения работы я реализовал в CPN Tools простой протокол передачи данных и провел анализ его пространства состояний.