Отчет по лабораторной работе №1. Реализация протоколов автоматического запроса повторной передачи Go-Back-N и Selective Repeat

Мануилов Георгий, студент гр. 3640102/80201

1 Постановка задачи

Требуется разработать систему из двух агентов, способных обмениваться данными друг с другом. Требования к системе:

- Должна моделироваться ненадежность канала связи: с заданной вероятностью пакеты при передаче должны теряться.
- Должна обеспечиваться доставка получателю всех отправленных данных посредством протоколов автоматического запроса повторной передачи Go-Back-N и Selective Repeat.

2 Технические подробности реализации

Программа-агент реализована на языке программирования C++. Программа может функционировать в двух режимах: прием и передача. Система представляет из себя два экземпляра программы, запущенных в разных режимах. Взаимодействие между процессами осуществляется с помощью очередей сообщений посредством библиотеки Boost.Interprocess.

Программа разделена на следующие сущности:

- **OneWayTransducer** - примитивный приемник/передатчик. Не контролирует доставку, работает только в одну сторону, утилизирует одну очередь сообщений.
- **SendingTransducer** - передатчик, реализующий протокол автоматического запроса повторной передачи данных, утилизирует две очереди сообщений (два экземпляра OneWayTransducer): одну для отправки, одну для получения подтверждения доставки.
- **ReceivingTransducer** - приемник, реализующий протокол автоматического запроса повторной передачи данных, утилизирует две очереди сообщений (два экземпляра OneWayTransducer): одну для получения, одну для отправки подтверждения доставки.

Каждый пакет (сообщение) для передачи содержит порядковый номер и 1 байт данных.
Программа-агент получает на вход путь к конфигурационному файлу, который содержит в себе следующие поля:

- **mode** - режим функционирования агента (SEND/RECEIVE)
- **mq_id** - ID канала передачи данных (должен совпадать у приемника и передатчика)
- **protocol** - ARQ протокол (GBN/SR)
- **file** - путь к файлу с данными для передачи (только для передатчика)
- **window_size** - размер скользящего окна (только для передатчика)
- **timeout** - время в милисекундах, после которого сообщение считается потерянным, если не пришло подтверждение доставки (только для передатчика)
- **loss_probability** - вероятность (от 0 до 1) потери сообщения (только для приемника)

Исходный код решения, а также инструкции по сборке и запуску представлены в git репозитории по ссылке `https://github.com/dev0x13/networks_labs/tree/master/lab1`.

3 Пример работы программы

Конфигурация агента-передатчика:

```
mode = SEND
mq_id = 1
protocol = GBN
file = /home/george/networks_labs/lab1/cfg/test_file
window_size = 2
timeout = 500
```

Конфигурация агента-приемника:

```
mode = RECEIVE
mq_id = 1
protocol = GBN
loss_probability = 0.3
```

Вывод:

```
$ ./agent ../cfg/sender.cfg
Attached to sending queue ‘1_receive‘
Attached to receiving queue ‘1_ack‘
Sent: Message[b: h] to ‘1_receive‘
Sent: Message[b: e] to ‘1_receive‘
Sent: Message[b: h] to ‘1_receive‘
Sent: Message[b: e] to ‘1_receive‘
Ack: Message[b: h] from ‘1_ack‘
Sent: Message[b: e] to ‘1_receive‘
Sent: Message[b: l] to ‘1_receive‘
Ack: Message[b: e] from ‘1_ack‘
Ack: Message[b: l] from ‘1_ack‘
Sent: Message[b: l] to ‘1_receive‘
Sent: Message[b: o] to ‘1_receive‘
Ack: Message[b: l] from ‘1_ack‘
Sent: Message[b: o] to ‘1_receive‘
Sent: Message[b:
] to ‘1_receive‘
Ack: Message[b: o] from ‘1_ack‘
Ack: Message[b:
] from ‘1_ack‘
Transmission finished!
Sent: hello
```

```
$ ./agent ../cfg/receiver.cfg
Attached to sending queue ‘1_ack‘
Attached to receiving queue ‘1_receive‘
Lost: Message[b: h] from ‘1_receive‘
Received: Message[b: h] from ‘1_receive‘
Lost: Message[b: e] from ‘1_receive‘
Received: Message[b: e] from ‘1_receive‘
Received: Message[b: l] from ‘1_receive‘
Received: Message[b: l] from ‘1_receive‘
Lost: Message[b: o] from ‘1_receive‘
Received: Message[b: o] from ‘1_receive‘
Received: Message[b:
] from ‘1_receive‘
Transmission finished!
Received: hello
```

4 Оценка и сравнение эффективности протоколов

Эффективность протоколов оценивалась по двум параметрам:

- по коэффициенту эффективности $k = \frac{\text{количество пакетов для передачи}}{\text{количество переданных пакетов}}$
- по времени от начала до конца передачи в милисекундах t.

Для оценки проводилась серия сеансов передачи с разными размерами скользящего окна w и с разными вероятностями потери пакетов p. Во всех сеансах использовался один и тот же набор данных (85 пакетов). Полученные зависимости представлены в таблицах 1 и 2, и на рис. 1-4.

p	t_{GBN}, мс	k_{GBN}	t_{SR}, мс	k_{SR}
0	293	1.00	292	1.00
0.1	354	0.83	274	0.89
0.2	393	0.75	456	0.73
0.3	570	0.53	524	0.71
0.4	708	0.42	607	0.63
0.5	1081	0.27	768	0.51
0.6	1525	0.19	1234	0.34
0.7	1690	0.18	1337	0.32
0.8	3203	0.09	2052	0.21
0.9	7372	0.04	4922	0.09

Таблица 1: Зависимость эффективности протоколов от вероятности потери пакета p при фиксированном размере скользящего окна $w = 3$

w	t_{GBN}, мс	k_{GBN}	t_{SR}, мс	k_{SR}
1	1718	0.5	1676	0.51
2	992	0.44	979	0.52
3	1097	0.27	697	0.53
4	913	0.25	739	0.44
5	1124	0.16	533	0.54
6	945	0.16	428	0.54
7	853	0.15	394	0.54
8	846	0.14	460	0.47
9	631	0.17	425	0.47
10	630	0.15	375	0.5

Таблица 2: Зависимость эффективности протоколов от размера скользящего окна w при фиксированной вероятности потери пакета $p = 0.5$

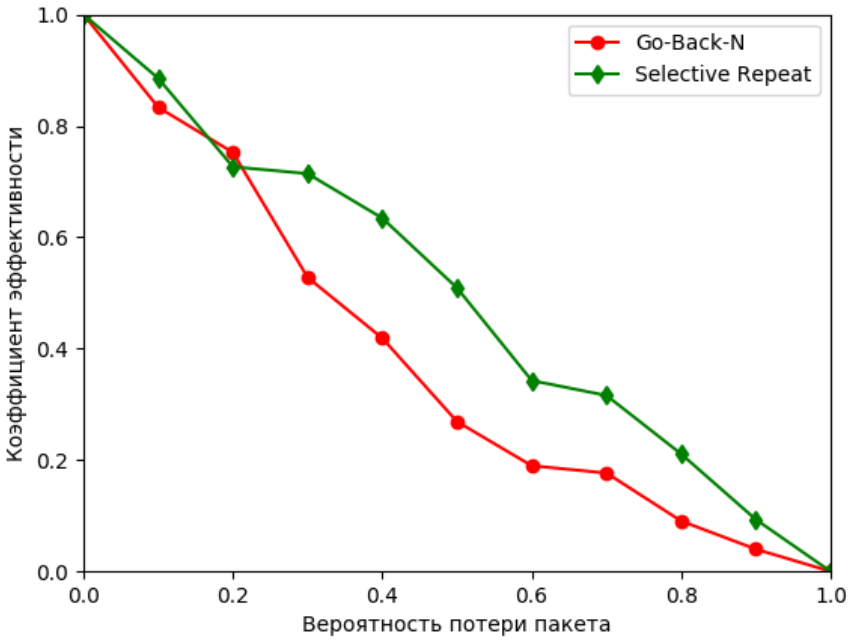

Рис. 1: Зависимость коэффициента эффективности от вероятности потери пакета

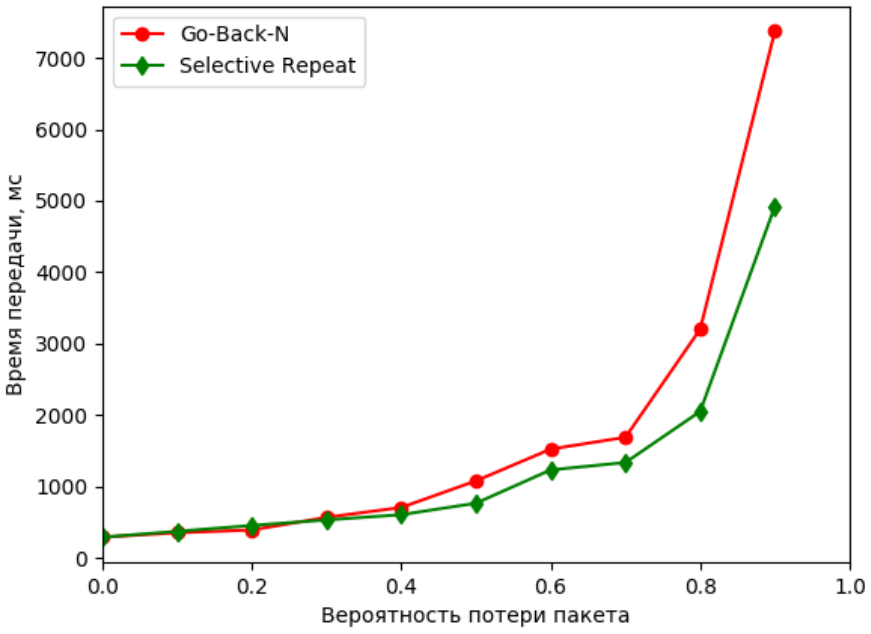

Рис. 2: Зависимость времени передачи от вероятности потери пакета

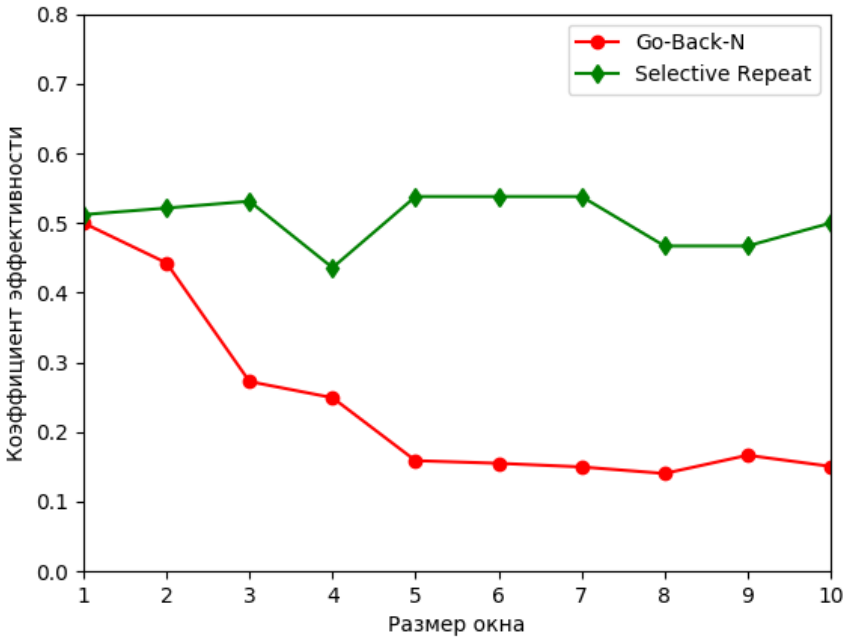

Рис. 3: Зависимость коэффициента эффективности от размера скользящего окна

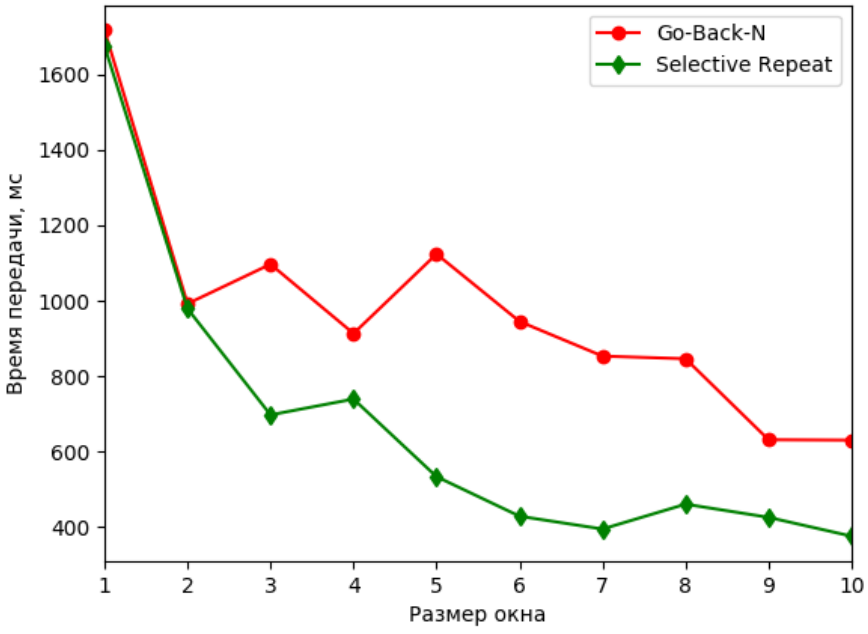

Рис. 4: Зависимость времени передачи от размера скользящего окна

По представленным зависимостям можно сделать следующие выводы:

- При малых (< 0.3) вероятностях потери пакета и фиксированном размере окна протоклы показывают одинковую эффективность.
- При больших (> 0.3) вероятностях потери пакета и фиксированном размере окна Selective Repeat показывает большую эффективность, чем Go-Back-N.
- Размер окна не оказывает влияние на коэффициент эффективности для Selective Repeat при фиксированной вероятности потери пакета, но его увеличение ведет к уменьшению времени передачи, то есть чем больше размер окна, тем больше эффективность протокола.
- Увеличение размера окна при фиксированной вероятности потери пакета ведет к уменьшению эффективности Go-Back-N.