Лабораторная работа 1

Простые модели компьютерной сети

Ланцова Яна Игоревна

Содержание

# 1 Цель работы

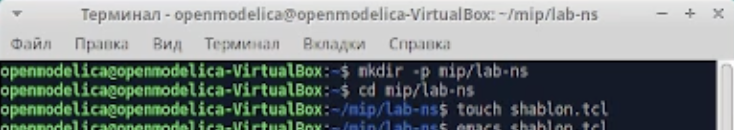
Приобрести навыки моделирования сетей передачи данных с помощью средства имитационного моделирования NS-2, а также проанализировать полученные результаты моделирования.

# 2 Задание

1. Создать шаблон сценария для NS-2;
2. Выполнить простой пример описания топологии сети, состоящей из двух узлов и одного соединения;
3. Выполнить пример с усложнённой топологией сети;
4. Выполнить пример с кольцевой топологией сети;
5. Выполнить упражнение.

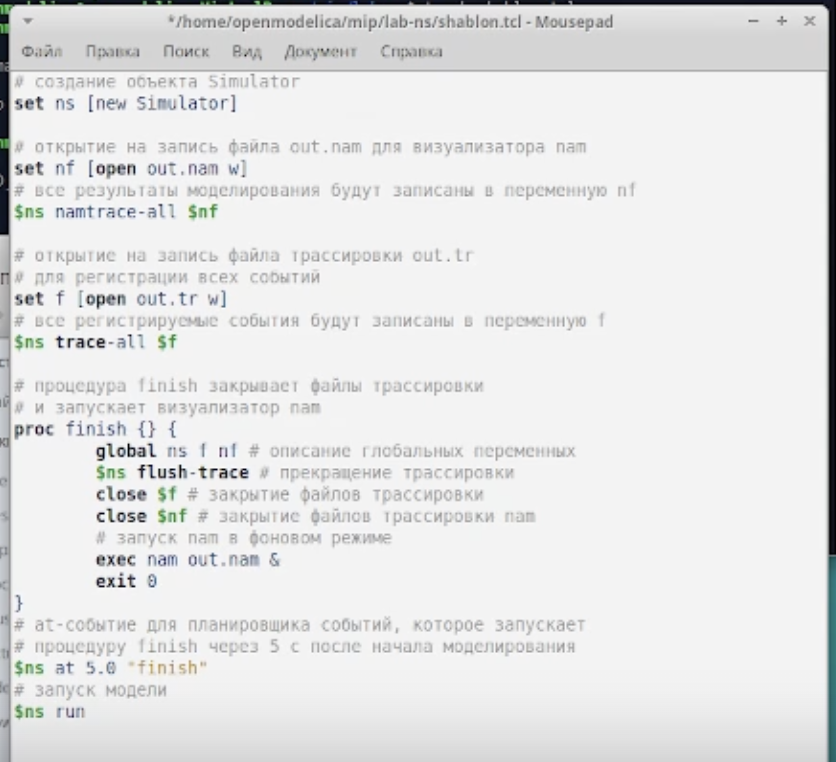
# 3 Выполнение лабораторной работы

В своём рабочем каталоге создадим директорию mip, в которой будут выполняться лабораторные работы. Внутри mip создадим директорию lab-ns, а в ней файл shablon.tcl (рис. [??]).



создание директории

Сначала создадим объект типа Simulator. Затем создадим переменную nf и укажем, что требуется открыть на запись nam-файл для регистрации выходных результатов моделирования. Вторая строка даёт команду симулятору записывать все данные о динамике модели в файл out.nam. Далее создадим переменную f и откроем на запись файл трассировки для регистрации всех событий модели. После этого добавим процедуру finish, которая закрывает файлы трассировки и запускает nam. С помощью команды at указываем планировщику событий, что процедуру finish запустим через 5 с после начала моделирования, после чего запустим симулятор ns(рис. [??]).



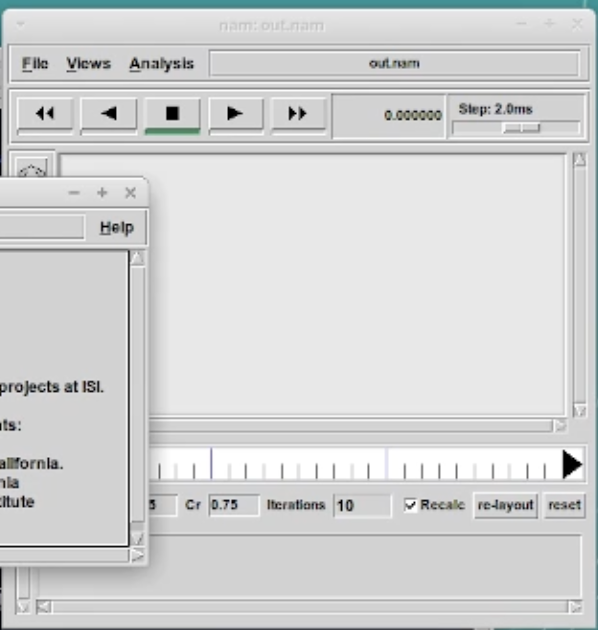
написание кода

Сохранив изменения в отредактированном файле shablon.tcl и закрыв его, можно запустить симулятор командой (рис. [??]):

команда для запуска симулятора

команда для запуска симулятора

Увидим пустую область моделирования, поскольку ещё не определены никакие объекты и действия (рис. [??]).



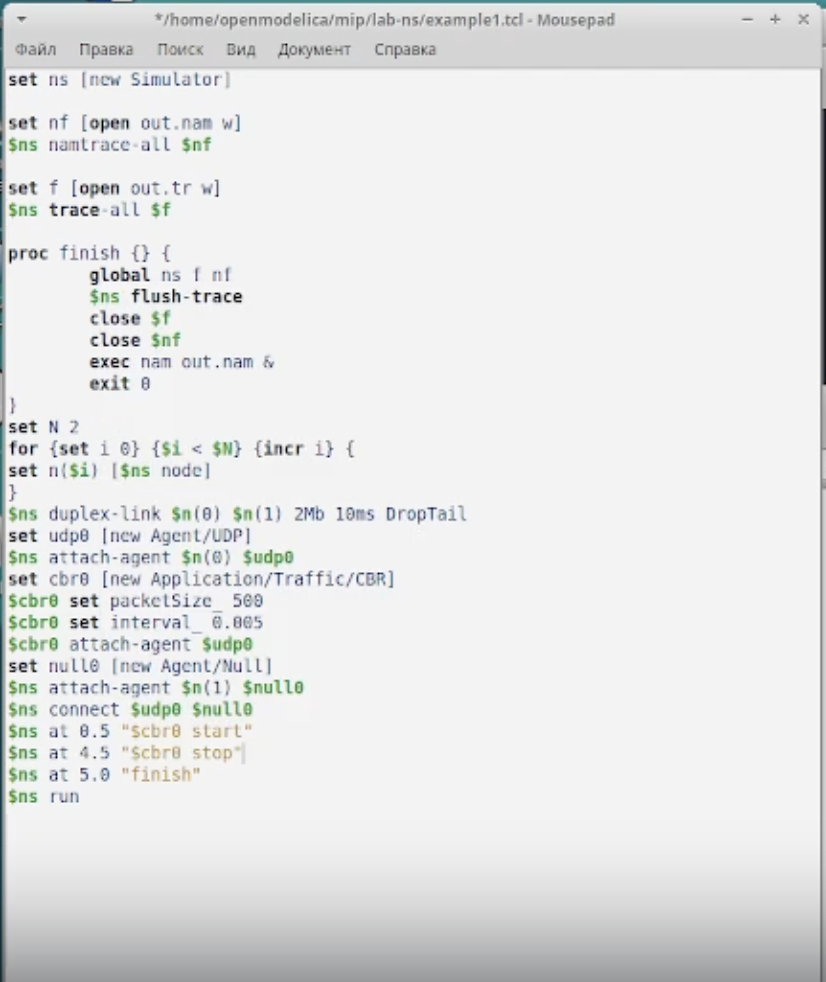
окно симулятора

Скопируем содержимое созданного шаблона в новый файл: cp shablon.tcl example1.tcl (рис. [??]).

копирование шаблона

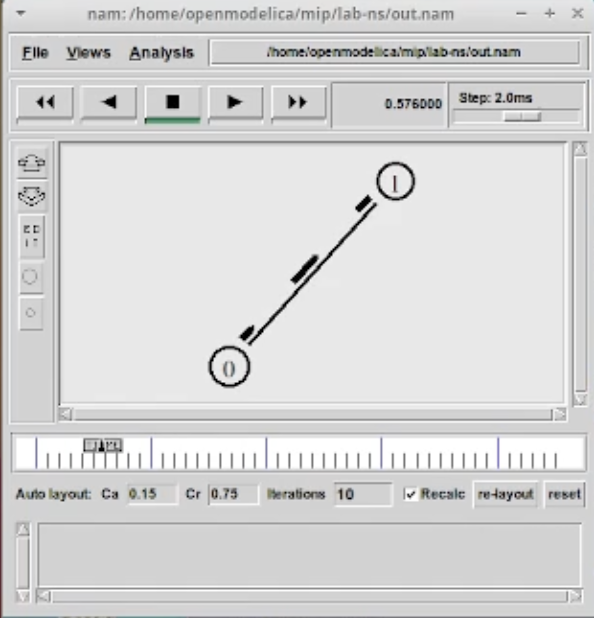
копирование шаблона

Откроем example1.tcl на редактирование. Добавим в него до строки $ns at 5.0 "finish" описание топологии сети. Создадим агенты для генерации и приёма трафика. Создается агент UDP и присоединяется к узлу n0. В узле агент сам не может генерировать трафик, он лишь реализует протоколы и алгоритмы транспортного уровня. Поэтому к агенту присоединяется приложение. В данном случае — это источник с постоянной скоростью (Constant Bit Rate, CBR), который каждые 5 мс посылает пакет R = 500 байт. Таким образом, скорость источника: . Далее создадим Null-агент, который работает как приёмник трафика, и прикрепим его к узлу n1. Соединим агенты между собой. Для запуска и остановки приложения CBR добавляются at-события в планировщик событий (перед командой $ns at 5.0 “finish”) (рис. [??]).



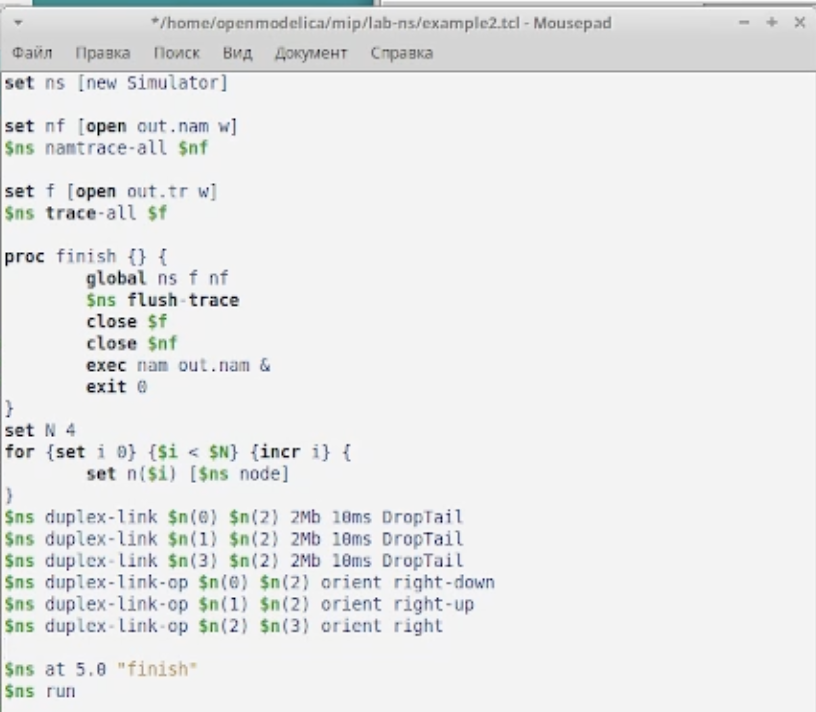
редактирование файла example1.tcl

Сохранив изменения в отредактированном файле и запустив симулятор, получим в качестве результата запуск аниматора nam в фоновом режиме (рис. [??]).



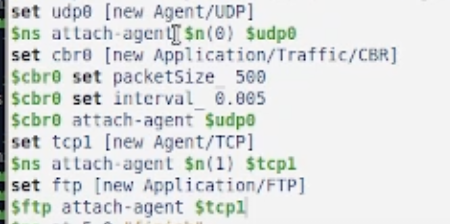
окно симулятора

Скопируем содержимое созданного шаблона в новый файл: cp shablon.tcl example2.tcl и откроем example2.tcl на редактирование. Создадим 4 узла и 3 дуплексных соединения с указанием направления (рис. [??]).



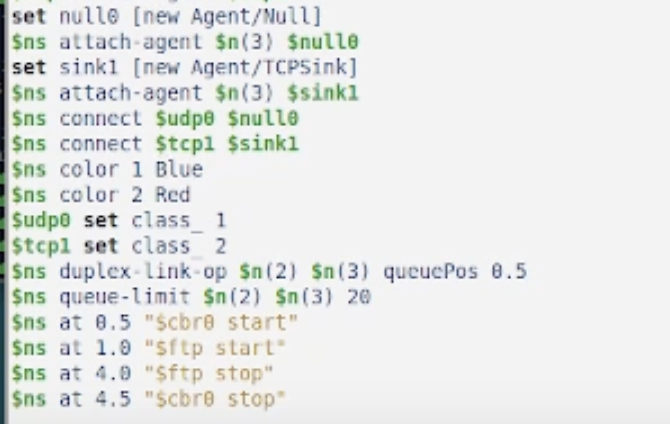
редактирование файла example2.tcl

Создадим агент UDP с прикреплённым к нему источником CBR и агент TCP с прикреплённым к нему приложением FTP (рис. [??]).



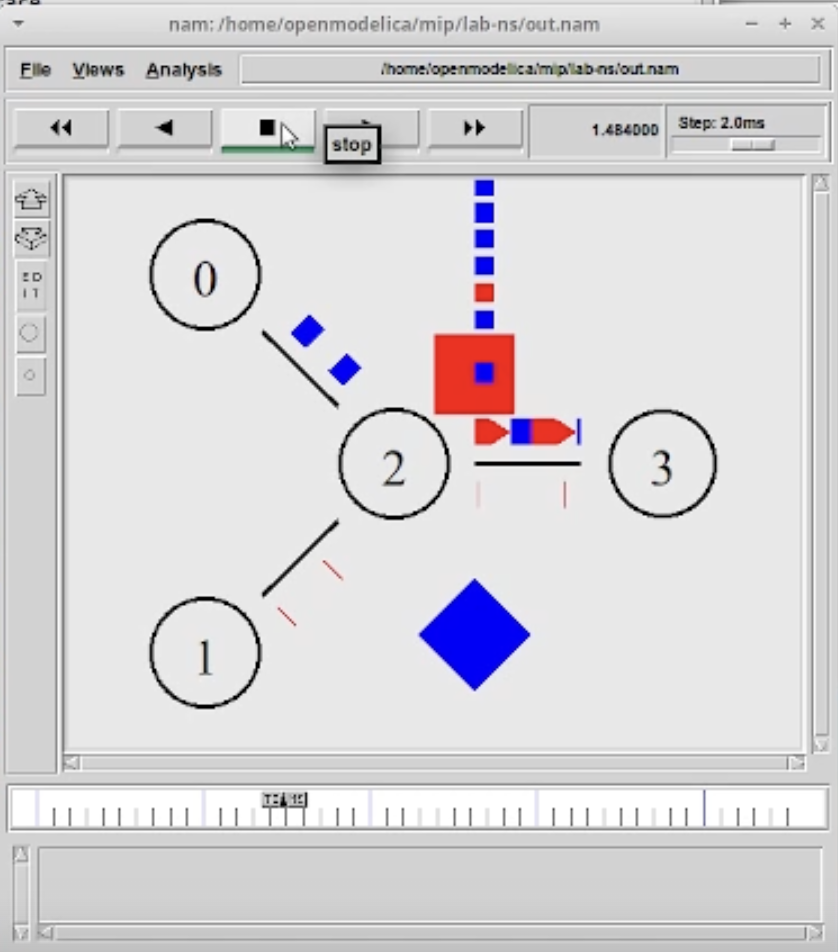
редактирование файла example2.tcl

Создадим агенты-получатели. Соединим агенты udp0 и tcp1 и их получателей. Зададим описание цвета каждого потока. Выполним отслеживание событий в очереди и наложение ограничения на размер очереди. Добавим at-события (рис. [??]).



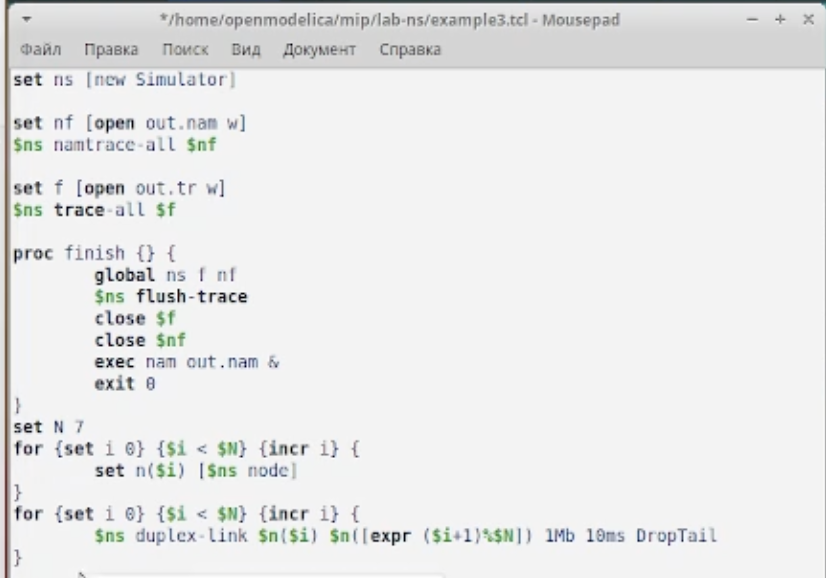
редактирование файла example2.tcl

Сохранив изменения в отредактированном файле и запустив симулятор, получим анимированный результат моделирования (рис. [??]).



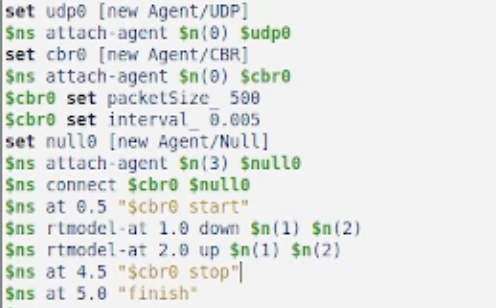
окно симулятора

Скопируем содержимое созданного шаблона в новый файл: cp shablon.tcl example3.tcl и откроем example3.tcl на редактирование. Опишем топологию моделируемой сети. Далее соединим узлы так, чтобы создать круговую топологию. Каждый узел, за исключением последнего, соединяется со следующим, последний соединяется с первым. Для этого в цикле использован оператор %, означающий остаток от деления нацело (рис. [??]).



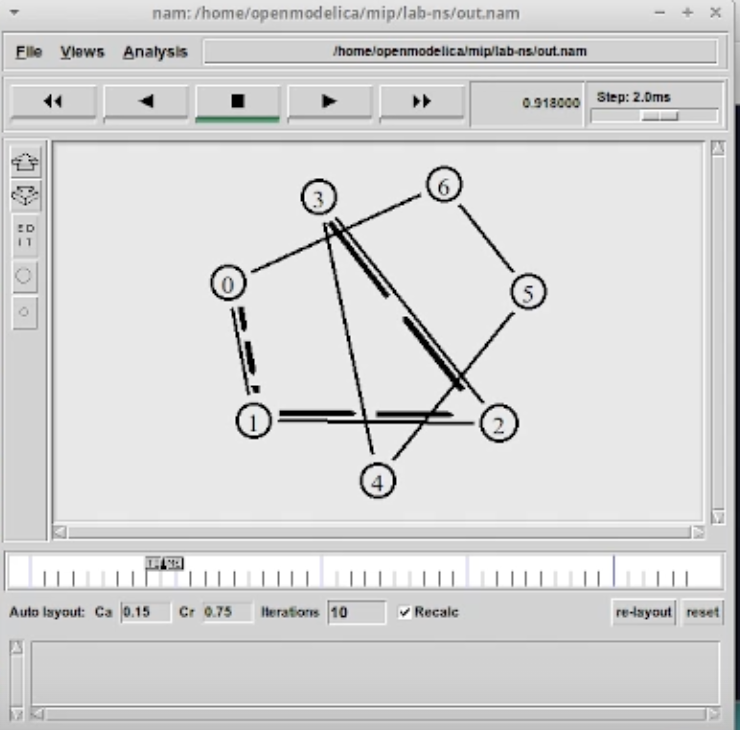
редактирование файла example3.tcl

Зададим передачу данных от узла n(0) к узлу n(3). Данные передаются по кратчайшему маршруту от узла n(0) к узлу n(3), через узлы n(1) и n(2). Добавим команду разрыва соединения между узлами n(1) и n(2) на время в одну секунду, а также время начала и окончания передачи данных (рис. [??]).



редактирование файла example3.tcl

Добавив в начало скрипта после команды создания объекта Simulator: $ns rtproto DV увидим, что сразу после запуска в сети отправляется небольшое количество маленьких пакетов, используемых для обмена информацией, необходимой для маршрутизации между узлами. Когда соединение будет разорвано, информация о топологии будет обновлена, и пакеты будут отсылаться по новому маршруту через узлы n(6), n(5) и n(4) (рис. [??]).



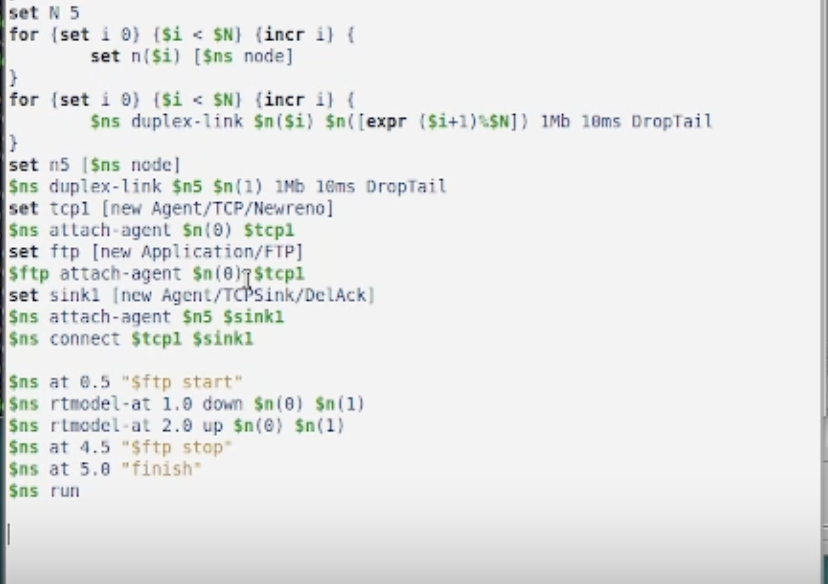
окно симулятора

**Упражнение**

Внесем следующие изменения в реализацию примера с кольцевой топологией сети:

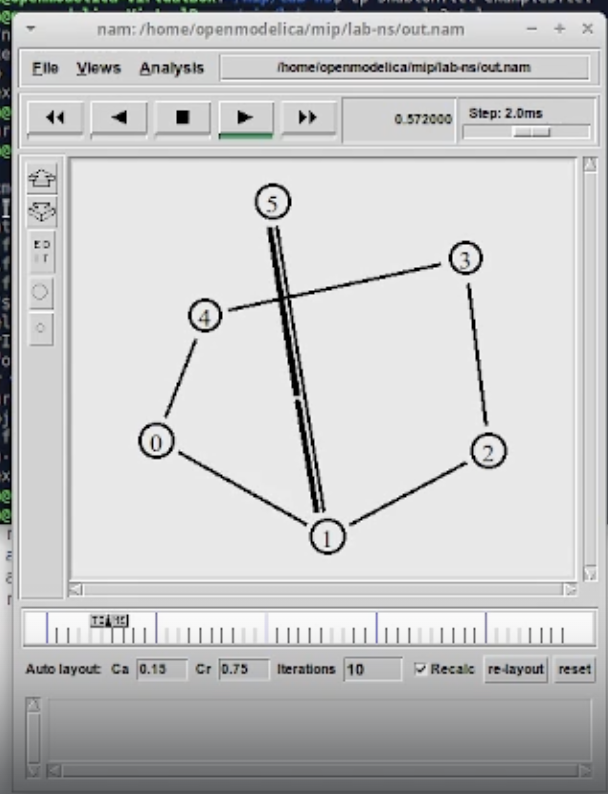
* передача данных должна осуществляться от узла n(0) до узла n(5) по кратчайшему пути в течение 5 секунд модельного времени;
* передача данных должна идти по протоколу TCP (тип Newreno), на принимающей стороне используется TCPSink-объект типа DelAck; поверх TCP работает протокол FTP с 0,5 до 4,5 секунд модельного времени;
* с 1 по 2 секунду модельного времени происходит разрыв соединения между узлами n(0) и n(1);
* при разрыве соединения маршрут передачи данных должен измениться на резервный, после восстановления соединения пакеты снова должны пойти по кратчайшему пути.

Изменим количество узлов в кольце на 5, а 6 узел n(5) отдельно присоединим к узлу n(1). Вместо агента UDP создадим агента TCP (типа Newreno), а на принимающей стороне используем TCPSink-объект типа DelAck; поверх TCP работает протокол FTP с 0,5 до 4,5 секунд модельного времени Также зададим с 1 по 2 секунду модельного времени разрыв соединения между узлами n(0) и n(1)(рис. [??]).



редактирование файла example3.tcl

Запустим программу и увидим, что пакеты идут по кратчайшему пути через узел n(1) ([??]).



окно симулятора

При разрыве соединения часть пакетов теряется, но поскольку данные обновляются пакеты начинают идти по другому пути. После восстановления соединения пакеты снова идут по кратчайшему пути.

# 4 Выводы

В процессе выполнения данной лабораторной работы я приобрела навыки моделирования сетей передачи данных с помощью средства имитационного моделирования NS-2, а также проанализировала полученные результаты моделирования.