Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра информационных и управляющих систем

Работа допущена к защите

Зав. кафедрой ИУС

\_\_\_\_\_\_\_\_И.Г. Черноруцкий

«\_\_\_\_\_»\_\_\_\_\_\_\_\_ г.

**ВЫПУСКНАЯ РАБОТА БАКАЛАВРА**

Тема: ***Сравнение быстродействия алгоритмов маршрутизации в беспроводных самоорганизующихся сетях***

**Направление:** 09.03.04 – Программная инженерия

Выполнил студент гр. 43504/1-2 \_\_\_\_\_\_\_\_\_\_\_\_\_Я. С. Мамонтов

Руководитель д.т.н., проф. \_\_\_\_\_\_\_\_\_\_\_\_\_Т. В. Коликова

Санкт-Петербург

2015 г

# Реферат

с. 50 рис. 15

В данной работе проведено сравнение быстродействия алгоритмов маршрутизации в беспроводных Ad-Hoc сетях при некоторых сценариях применения.

Данное исследование призвано определить некоторые характеристики алгоритмов маршрутизации, сравнить выбранные образцы по этим характеристикам и определить сильные и слабые стороны, выявить возможные закономерности между такими параметрами сети как количество узлов и их взаимное расположение и быстродействием алгоритмов.

Оглавление

[Введение 7](#_Toc421577206)

[Актуальность 8](#_Toc421577207)

[Цели и задачи 9](#_Toc421577208)

[Краткое содержание 10](#_Toc421577209)

[1. Обзор предметной области 11](#_Toc421577210)

[1.1. Беспроводные Ad Hoc сети 11](#_Toc421577211)

[1.2. Протокол маршрутизации AODV 14](#_Toc421577212)

[1.3. Mesh-сети стандарта IEEE 802.11s 16](#_Toc421577213)

[1.4. Mesh-сети на основе продукта Motorola Mesh 20](#_Toc421577214)

[2. Перенос ПО Motorola Mesh в систему симуляции 21](#_Toc421577215)

[2.1. Выбор симулятора сетей 21](#_Toc421577216)

[2.2. Внутренне устройство симулятора NS-3 23](#_Toc421577217)

[2.3. Создание интерфейса связи симулятора и протокола маршрутизации 25](#_Toc421577218)

[3. Разработка тестовых сценариев 27](#_Toc421577219)

[3.1. Способы измерения производительности протоколов маршрутизации в беспроводных самоорганизующихся сетях 27](#_Toc421577220)

[3.2. Общие для всех сценариев положения 30](#_Toc421577221)

[3.3. Описание первого тестового сценария 31](#_Toc421577222)

[3.4. Описание второго тестового сценария 32](#_Toc421577223)

[3.5. Описание третьего тестового сценария 33](#_Toc421577224)

[3.6. Описание четвертого тестового сценария 35](#_Toc421577225)

[4. Анализ экспериментальных данных 37](#_Toc421577226)

[4.1. Анализ результатов первого сценария 37](#_Toc421577227)

[4.2. Анализ результатов второго сценария 41](#_Toc421577228)

[4.3. Анализ результатов третьего сценария 43](#_Toc421577229)

[4.4. Анализ результатов четвертого сценария 45](#_Toc421577230)

[4.5. Общие выводы 47](#_Toc421577231)

[Заключение 49](#_Toc421577232)

[Библиографический список 50](#_Toc421577233)

# Введение

В современном мире практически все межличностные, деловые, торговые и другие коммуникации плавно перешли из физического мира в мир виртуальный. Практически все, что мы можем наблюдать вокруг себя так или иначе связано с информационными технологиями. Каждую минуту в мире генерируется огромное количество уникальной информации и для того чтобы она смогла беспрепятственно передвигаться через границы и сквозь расстояния по всей планете существуют сети передачи данных.

Существует много разнообразных сетей. Они делятся на глобальные и локальные, военные и гражданские, закрытые и публичные, бесплатные и платные, проводные и беспроводные. Постоянно разрабатываются новые виды цифровой связи и улучшаются старые.

В данной работе рассматриваются беспроводные сети построенные по стандарту WiFi. Это целый сегмент оборудования и технологий применяемых и в промышленности и в быту. Широкому кругу людей они известны как средства мобильной связи и доступа к интернету. Одним из направлений данной области как раз и являются так называемые Ad-Hoc сети – это одноранговые мобильные самоорганизующиеся сети.

## Актуальность

В последнее время наблюдается повышенный интерес к мобильным Ad-Hoc сетям (Mobile Ad-Hoc Networks, MANET). MANET сети обладают целым рядом свойств в положительную сторону отличающих их от традиционных иерархических сетей: децентрализованное управление, позволяющее сети адаптироваться к изменяющимся условиям, возможность увеличения дальности передачи информации без увеличения мощности передатчиков. Сети данного типа могут применяться в различных ситуациях: во время военных действий, в структурах МЧС где важна скорость развертывания, в системах транспорта, в сенсорных сетях где необходимы большой радиус действия и малое энергопотребление.

Однако данные сети имеют свои трудности и нетривиальные задачи . Например, поиск наилучшего пути от источника к приемнику, своевременное обнаружение разрыва связи, поддержание актуальности информации о состоянии путей и др. Решать все эти проблемы призваны алгоритмы маршрутизации.

На данный момент уже существует большое количество алгоритмов маршрутизации, а также постоянно ведется разработка новых. В подобной ситуации мы должны иметь четкое представление о каждом алгоритме его сильных и слабых сторонах. В этой работе мы рассмотрим несколько алгоритмов маршрутизации и сравним их по некоторым параметрам.

## Цели и задачи

Целью данной работы является, используя симулятор компьютерных сетей, сравнить быстродействие алгоритмов маршрутизации в беспроводных самоорганизующихся сетях в некоторых сценариях применения.

Для достижения поставленной цели был сформулирован ряд задач:

* Анализ существующих алгоритмов;
* Выбор симулятора компьютерных сетей;
* Интегрирование ПО Motorola Mesh с выбранным симулятором;
* Создание тестовых сценариев;
* Проведение тестов с целью получить сравнительные данные;
* Анализ результатов.

## Краткое содержание

Работа состоит из четырех главных частей.

В первой части раскрывается предметная область, даются основные сведения необходимые для понимания дальнейшей работы. Описываются тестируемы в работе алгоритмы маршрутизации их теоретические характеристики и достоинства.

Во второй части обосновывается выбор конкретного симулятора компьютерных сетей, описываются его возможности и внутренне устройство. Описывается процесс переноса ПО Motorola Mesh в симулятор.

В третью часть помещено детальное описание и обоснование разработанных тестовых сценариев. Используемые метрики и подходи тестирования.

В четвертой части находится полное описание полученных результатов и их анализ.

# 1. Обзор предметной области

## 1.1. Беспроводные Ad Hoc сети

Беспроводная мобильная Ad Hoc сеть или MANET сеть состоит из мобильных платформ, способных свободно перемещаться в пространстве, которые могут быть установлены на самолетах, кораблях, машинах, на людях и на любых маленьких предметах способных нести на себе небольшой микроконтроллер. MANET – это автономная система мобильных устройств, она может быть как изолированной, так и иметь доступ к другим сетям, например, к Интернету.

Особенность MENET сетей состоит в том, что они не обладают сформированной топологией, она постоянно изменяется. Узлы сети могут передвигаться в пространстве относительно друг друга и на большие расстояния. Среда распространения также непрерывно изменяется, возникают различные радио помехи, на пути сигнала появляются и исчезают препятствия. Помимо этого, для всех беспроводных систем характерны ограниченная полоса пропускания и зона радиовидимости. В следствии, мы имеем невозможность применения сетевых протоколов и алгоритмов маршрутизации, применяемых в классических проводных и беспроводных иерархических сетях, т.к. они оказываются абсолютно неэффективными и не обеспечивают нужную производительность в данных обстоятельствах.

Для применения в Ad Hoc сетях протоколы маршрутизации должны обладать следующими качествами [1]:

* Быть распределенными. Все узлы сети должны принимать участие в работе протокола.
* Обеспечивать стабильную работу в условиях быстро меняющейся топологии.
* Обеспечивать небольшое время построения маршрутов.
* Обладать механизмами определения разрыва связей и подсчета метрик.
* Гарантировать отсутствие маршрутных петель и проблемы счета до бесконечности.
* Иметь высокую степень масштабируемости.
* Использовать как можно меньше ресурсов сети.
* Поддерживать механизм качества обслуживания (Quality of Service).

Все существующие на данные момент протоколы маршрутизации можно классифицировать по нескольким признакам.

**По использованию данных геолокации:**

1. Топологические. Используют только данные о сетевых соединениях между узлами.
2. Географические. Кроме данных о сетевых соединениях используют информацию о географическом положении для выбора оптимальных путей и прогнозирования появления либо исчезновения связи.

**По принципу работы:**

1. Проактивные. Состояние обо всех соединениях и маршрутах постоянно хранятся на всех или на нескольких корневых узлах. Для поддержания актуальности информации периодически посылаются служебные сообщения со всей необходимой информацией о конфигурации сети и ее изменениях. (Пример OLSR, FSR).
2. Реактивные. Путь до конкретного узла устанавливается только при необходимости. Для этого узел отправитель рассылает запрос на все соседние узлы, которые в свою очередь также отправляет его своим соседям. При достижении этого пакета узла получателя, устанавливается соединение и посылается ответ узлу отправителю. При потере соединения данный механизм запускается заново. (Пример AODV, DSR).
3. Гибридные. Комбинируют в себе реактивный и проактивный режимы работы. Они хорошо подходят для больших сетей, где все узлы разбиваются на несколько частей, внутри которых организовывается проактивная сеть, а между собой группы связываются реактивной сетью. (Пример HWMP).

**По способу определения оптимальности маршрута:**

1. Протоколы вектора расстояния. Оптимальным считают маршрут, содержащий наименьшее число промежуточных узлов.
2. Протоколы со сложной метрикой. Используют различные метрики и комбинации из них, например, здержка при доставке пакета, пропускная способность канала, уровень сигнала и др.

Все классы протоколов имеют свои достоинства и недостатки. Каждый протокол хорошо справляется со своей задачей только в определенных условиях. Дальше мы рассмотрим несколько протоколов более подробно и раскроем их теоретические достоинства и недостатки.

## 1.2. Протокол маршрутизации AODV

AODV (Ad hoc On-Demand Distance Vector) – реактивный протокол маршрутизации, разработанный для мобильных Ad Hoc сетей и призванный понизить нагрузку на процессор и оперативную память устройств и уменьшить использование сети для передачи служебной информации. Он разрабатывался на основе двух предшествующих протоколов: реактивного DSR и проактивного DSDV. Данный протокол является динамическим, самостартующим и многоскачковым.

Когда необходимо отправить информацию другому узлу сети, узел отправитель рассылает всем своим соседям пакет запроса пути (Route Request, RREQ), которые в свою очередь рассылают его своим соседям. Когда пакет RREQ достигает узел назначения, инициируется механизм посылки пакета ответа (Route Reply, RREP), который двигаясь, точно тем же путем что пришел RREQ, достигает узел отправитель. После чего путь считается успешно установленным и начинается передача данных.

С помощью технологии порядковых номеров назначения (Destination Sequence Number) алгоритм избавляется от таких проблем самоорганизующихся сетей, как путевые циклы и «счета до бесконечности» Белмана-Форда. Каждый узел сети имеет свой собственный порядковый номер назначения, который увеличивается на единицу после каждого нового установления пути с каким-либо другим узлом. При отправке сообщения RREQ в него вместе с адресами отправителя и назначения помещается порядковый номер назначения, который в дальнейшем при прохождении пакета по сети, будет играть роль маркера, с помощью которого определяется свежесть информации хранящейся в таблицах маршрутизации.

В таблицах маршрутизации же хранятся только по одной записи для каждого узла назначения. Эта записи состоит из адреса узла назначения, адреса соседа, на который следует посылать информацию и порядкового номера назначения. Когда узел получает новый пакет RREQ он, если у него есть запись для соответствующего узла назначения, сравнивает порядковый номер, хранящийся у него в таблице с номером, полученным из пакета. Если имеющийся у него номер больше или равен пришедшему, то считается, что хранящаяся у него информация является достаточно достоверной и он, если данная функция включена, не рассылает дальше пакет RREQ, а посылает обратно пакет RREP в котором указывает путь до необходимого узла назначения и новый порядковый номер. В противном случае пакет RREQ отправляется дальше, а в таблице обновляется информация об узле отправителе.

Когда узел назначения принимает RREQ, он не только отправляет обратно ответ, но и обновляет свой порядковый номер. Когда промежуточный узел принимает пакет RREP, он также обновляет свою таблицу маршрутизации и добавляет запись об узле назначения, от которого пришел пакет.

В процессе передачи данный все узлы постоянно контролируют связи между соседями. Если обнаруживается разрыв, то узлы между которыми это произошло рассылают конечным узлам пути сообщения ошибки пути (Route Error, RERR). После чего вновь начинается процесс поиска пути.

## 1.3. Mesh-сети стандарта IEEE 802.11s

IEEE 802.11s – это стандарт, который описывает новый перспективный класс широкополосных беспроводных сетей, которые носят название «mesh-сети». Главными принципами построения mesh-сетей является самоорганизация архитектуры, масштабируемость, динамическое распределение трафика и непрерывный мониторинг состояния сети. Mesh-сети могут быть стационарными, мобильными и смешанными.

В обычных сетях стандарта 802.11 терминальные станции (STA) связаны с точками доступа (AP) и могут взаимодействовать только с ними. AP обычно связаны с проводной локальной сетью и часто имеют выход в Интернет, но между собой они не могут взаимодействовать. Mesh-сети же предлагают иной подход: существует одно устройство, называемое mesh-узел, которое может совмещать в себе несколько функций. Во-первых, mesh-узлы могут взаимодействовать между собой посредством одноранговых соединений. Во-вторых, mesh-узел по-прежнему может выступать в роли точки доступа, давая таким образом возможность терминальным станциям иметь доступ к сети. В-третьих, mesh-узел может исполнять роль портала в любую другу сеть, например, в локальную сеть Ethernet.

Стандарт не фиксирует какой-либо один алгоритм маршрутизации, а только описывает некоторые характеристики, которыми он должен обладать. Однако в качестве стандартного в тексте документа описан протокол **HWMP (Hybrid Wireless Mesh Protocol) –** гибридный протокол маршрутизации, основанный на протоколе AODV и адаптированный для работы на MAC-уровне. Также стандарт описывает одну обязательную к исполнению всеми устройствами метрику – Airtime Link Metric. Данная метрика представляет собой оценку времени передачи по каналу связи с учетом процента ошибок и скорости передачи данных. Кроме того протоколы могу использовать и любые другие метрики позволяющие повысить точность определения оптимальности пути, например, задержка при передачи от отправителя к получателю, вероятность потери пакета, расстояние между узлами, пропускная способность и др.

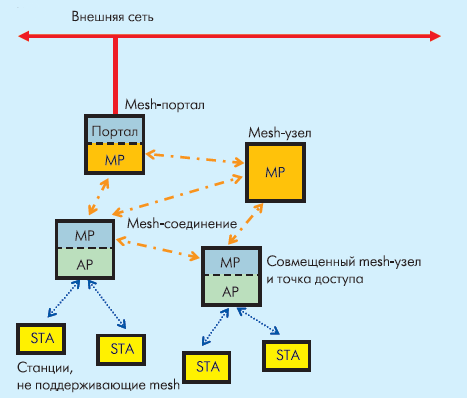


Рис 1.1. Архитектура mesh-сети.

Протокол HWMP объединяет в себе сразу два режима работы, реактивный и проактивный. В реактивном режиме протокол работает схожим образом с алгоритмом AODV. Перед передачей данный узел отправитель рассылает широковещательный пакет PREQ (Path Request), однако пути уже выбираются не только на основании порядкового номера назначения, а и на основании метрики полученной заранее с помощью специального механизма. Этот пакет распространяется по сети, пока не достигнет адресата. По мере продвижения от узла к узлу, поле метрики в пакете постоянно обновляется. В итоге узел адресат получит пакет с полной метрикой пути, если он получит несколько таких пакетов, то соответственно выберет путь с наилучшей метрикой. Затем узел адресат отправляет по полученному пути пакет ответ PREP (Path Reply) в котором указывает рассчитанную на предыдущем этапе метрику. Приняв его, узел отправитель считает путь установившемся и начинает передачу данных. Механизм порядковых номеров в данном случае тоже используется, и работает схожим с AODV образом.

Помимо упомянутых ранее, в пакетах данного протокола имеется еще одно поле называемое «время жизни» или TTL (Time to Live). С помощью данного поля можно ограничить число узлов через которые проходит пакет. При создании любого пакета, отправитель определяет значение TTL, и на каждом промежуточном узле оно будет уменьшаться на единицу. При достижении нуля трансляция пакета прекращается.

Проактивный режим отличается от реактивного только тем, что в сети назначается один или несколько корневых узлов. Данные узлы периодически рассылают PREQ пакеты. Остальные узлы принявшие этот пакет записывают к себе в таблицу путь до корневого узла, транслируют пакет PREQ следующим узлам сети и отправляют пакет PREP корневому узлу. В результате данного процесса все узлы знают путь до корневого узла, а корневой узел знает все пути до всех остальных узлов. Это помогает быстрее находить нужные пути, узлу отправителю не нужно запускать механизм поиска пути, а достаточно обратиться к корневому узлу, который уже знает этот путь. Однако данный режим несколько увеличивает загруженность сети, т.к. корневые узлы должны постоянно следить за состоянием сети и посылать служебные пакеты.

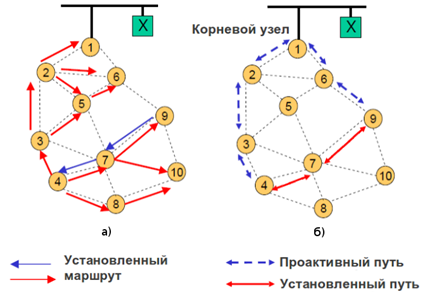


Рис 1.2. Пример построения маршрута от узла 4 до узла 9 в реактивном (а) и проактивном (б) режимах

## 1.4. Mesh-сети на основе продукта Motorola Mesh

Сердцем программного обеспечения Motorola Mesh является протокол MSRTM (Mesh Scalable Routing) – это высокопроизводительный протокол маршрутизации, разработанный специально для использования в беспроводных mesh-сетях. MSR технология позволяется динамически создавать самоформирующиеся, самовостанавливаемые, многоскачковые маршруты в Ad Hoc сетях.

Данный протокол является гибридным, т.е. совмещает в себе реактивные и проактивные технологии. Это закрытая коммерческая разработка, однако, из открытых источников, известно, что она включает в себя множество запатентованных алгоритмов и технологий, например, такие как методы определения метрик путей, выбор наилучшего пути, различные алгоритмы и технологии связанные с маршрутизацией и самоорганизацией сети.

# 2. Перенос ПО Motorola Mesh в систему симуляции

## 2.1. Выбор симулятора сетей

Для тестирования новых протоколов существует два глобальных подхода: тестирование на реально железе и имитационное моделирование. У тестирования на реальных устройствах существует много достоинств и недостатков. К самому серьезному достоинству, пожалуй, можно отнести то, что мы проверяем алгоритмы практически в тех условиях, в которых они и будут применяться и именно на том оборудовании, на котором будет, при дальнейшей эксплуатации, исполняться код алгоритма. К недостаткам можно отнести то, что к самому тестированию и отлаживанию работы алгоритма маршрутизации добавляется большая работа по тестирования и отладке самого оборудования и в существующих реалиях получается так, что последним приходится заниматься гораздо больше. Также данный вид тестирования является очень дорогостоящим, требующим долгой и кропотливой настройке и проверке. А в случаях большого количества узлов в сети, такое тестирования становится просто не выполнимым.

Имитационное моделирование или симуляция позволяет решить все проблемы тестирования на реальном оборудовании. С помощью специального программного обеспечения можно настроить практически любые параметры среды передачи данных, конфигурации сети, поведения отдельных узлов, вид передаваемого по сети трафика и многое другое. Используя симулятор сетей, можно настроить тестовый стенд и описать сценарий тестирования всего за несколько часов.

На данный момент симуляторы сетей нашли очень широкое применение в научной сфере. Они используются для исследований поведения различных сетевых технологий, протоколов маршрутизации и коммуникации и даже приложений использующих сеть передачи данных.

Существует большое количество различных симуляторов, некоторые из них являются коммерческими, некоторые являются узконаправленными, например, симуляторы персональных сетей, симуляторы конкретного оборудования. Встречаются симуляторы под разные платформы и написанные на разных языках, таких как C/C++, Java, Python.

Нами был сделан выбор в пользу симулятора NS-3 (Network Simulator 3) - свободно распространяемый фреймворк позволяющий описывать сценарии тестирования на языке C++. Первая версия проекта была разработана во французском институте INRIA и вышла в свет в июле 2006 года. В настоящее время это бурно развивающийся проект, над которым работают сотни людей по всему миру. На данный момент поддерживается не очень большое количество протоколов и сетевых технологий, в отличие от некоторых конкурентов, например, ns-2. Однако для наших целей там оказались все необходимые модели, какие как, Wi-Fi, технологии стандарта IEEE 802.11s и HWMP, AODV и такие высокоуровневые протоколы как ARP, ICMP, UDP и TCP. К тому же симулятор обладает большой документацией и гибкой архитектурой, которая позволяет при необходимости за небольшое время перенести в систему любой новый протокол.

## 2.2. Внутренне устройство симулятора NS-3

NS-3 – это дискретно-событийный симулятор компьютерных сетей. Это означает, что в симуляторе не эмулируется непрерывное течение времени, а весь процесс симуляции происходит скачками от события к событию.

В симуляторе, полностью повторяется структура обычного стека протоколов и все модули можно четко распределить по уровням модели OSI (см. рис 2.1).



Рис 2.1. Архитектура стека протоколов NS-3

В нашем случае (см. рис. 2.2) в роли среды передачи данный выступает класс WifiChannel, который эмулирует задержку передачи и уровень сигнала. Физический уровень имитирует класс WifiPhy, здесь задаются различные параметры физического оборудования, такие как частота передачи сигнала, стандарт, способ модуляции и др. Роль канального уровня представляют сразу несколько моделей: MacLow, DcfManager, MacHigh и WifiNetDevice. Класс WifiNetDevice является как бы оболочкой для остальных классов, он их хранит в себе и вызывает в нужный момент. В роли MacHigh могут выступать несколько различных классов в зависимости от которых, наше устройство может играть разную роль в сети: точка доступа, терминальное устройство или узел Ad Hoc сети.

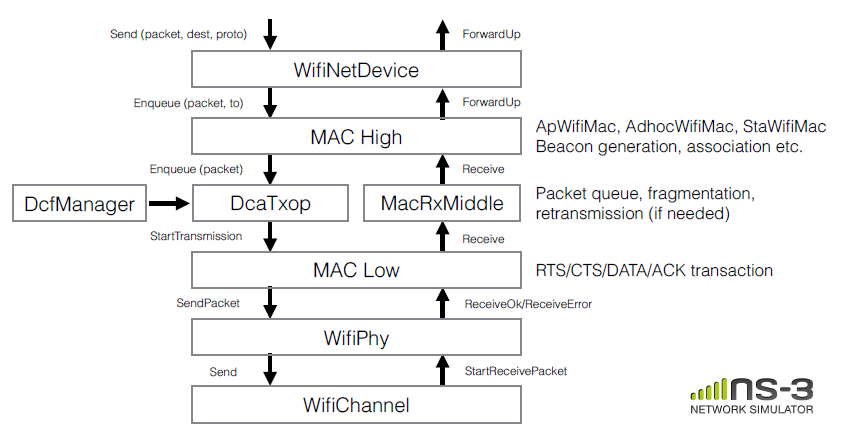


Рис 2.2. Архитектура модуля Wi-Fi

Каждый класс содержит некоторое количество параметров, которые в дальнейшем влияют на параметры симуляции. Например, в классе DcfManager мы можем настроить различные параметры фрагментации и ретрансляции пакетов, в классе MacLow мы задаем различные значения алгоритмов рассылки служебных Wi-Fi пакетов и алгоритма регулировки пропускной способности канала (WifiRemoteStationManager).

## 2.3. Создание интерфейса связи симулятора и протокола маршрутизации

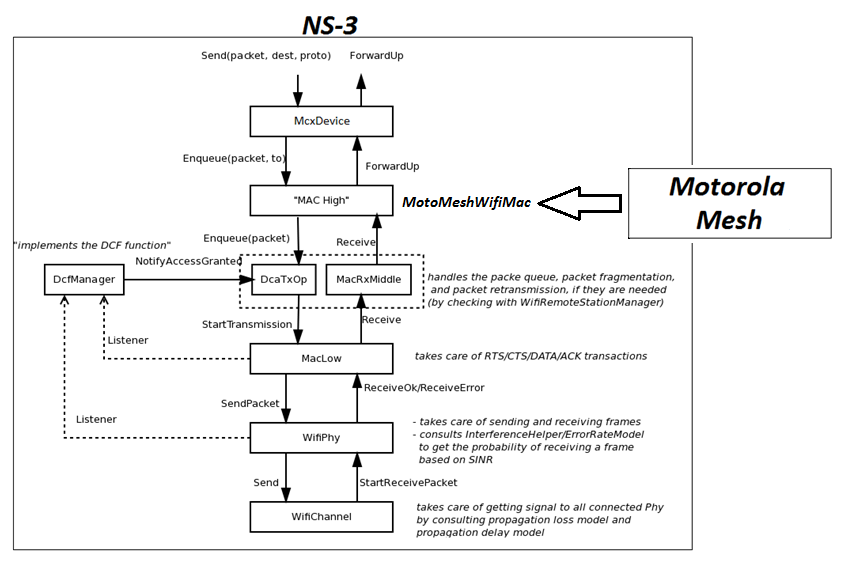
Для того чтобы реализовать полную функциональность ПО Motorola Mesh в симуляторе NS-3, нам потребовалось разработать новый класс на канальном уровне модели OSI или на уровне MAC High модели NS-3 (см. рис 2.3).

Рис 2.3. Схема внедрения ПО Motorola Mesh в симулятор NS-3

Наш класс MotoMeshWifiMac унаследовал от родительского класса RegularWifiMac все необходимые методы для полноценной работы с нижними уровнями Wi-Fi.

Были переопределены следующие методы:

* Enqueue (Ptr<const Packet> packet, Mac48Address to) – метод вызываемый верхними уровнями для отсылки пакета ниже по стеку. Код здесь был переписан для работы с ПО Motorola Mesh.
* Receive (Ptr<Packet> packet, const WifiMacHeader \*hdr) – метод вызываемый нижними уровнями, в данном случае кодом Motorola Mesh.

Также были добавлены новые методы расширяющие функциональность класса:

* MotoMeshInit () – метод вызываемый перед началом работы, для проведения инициализации Motorola Mesh.
* StartBeaconing () – метод реализующий механизм отправки и приема «Hello» сообщений (Beacons).
* SetRoot (bool root) – метод включает проактивный режим в протоколе Motorola Mesh и назначает конкретные заданные при конфигурации корневые узлы.

# 3. Разработка тестовых сценариев

## 3.1. Способы измерения производительности протоколов маршрутизации в беспроводных самоорганизующихся сетях

Для того чтобы сравнивать между собой протоколы, необходимы как количественные так и качественные метрики, которые бы отображали их пригодность и производительность. Данные метрики должны быть независимыми от используемого протокола.

С помощью следующих метрик можно полностью описать качественные характеристики протоколов маршрутизации для MANET сетей [5]:

1. Распределенность операций. Это один из главных показателей, показывающий насколько равномерно операции поддержания сети распределяются между всеми ее участниками.
2. Отсутствие маршрутных петель. Не критический показатель, однако, механизмы предотвращения маршрутных петель повышают стабильность и надежность сети.
3. Адаптируемость к уровню трафика. Показывает, присутствуют ли в протоколе механизмы адаптации к различным уровням трафика, которые достаточно важны в постоянно изменяющих свое состояние MANET сетях.
4. Проактивные операции. Несколько обратный к предыдущему показатель, который указывает на наличие проактивных алгоритмов, суть которых подробно описана в предыдущих разделах.
5. Безопасность. Ввиду своих особенностей MANET сети обладают меньшей безопасностью перед остальными видами сетей. Поэтому наличие алгоритмов безопасности играет немаловажную роль.
6. Поддержка однонаправленных соединений. Многие протоколы не поддерживают данные соединения. Однако однонаправленные связи очень часть возникают в беспроводных сетях, в частности из-за разности в мощности приемника и передатчика на двух устройствах. Поддержка подобных связей также может улучшить производительность сети.

Описанные выше метрики можно без труда вычислить просто из документов описывающих протоколы. Гораздо более сложно вычислимыми, но более полно описывающими производительность MANET сети, являются количественные метрики. Рассмотрим некоторые из них [5]:

1. Задержка. Среднее время задержки между отправление и приемом пакета. Чаще всего измеряют двустороннюю задержку (RTT, Round Trip Time) складывающуюся из задержек, полученных при передачи пакета в обе стороны.
2. Пропускная способность. Количество информации, которая может быть передана в единицу времени.
3. Процент поврежденных или не доставленных пакетов. Имеет большое значение для таких протоколов передачи данных как TCP.
4. Отношение среднего числа всех переданных бит информации к числу переданных бит данных. Данная метрика показывает, насколько много служебной информации предает протокол.
5. Отношение количества всех отправленных и принятых пакетов в сети.

Также мы должны быть уверены в том, что все протоколы тестировались при одинаковых внешних условиях. Ниже представлен список параметров, которые важно учесть при тестировании [5]:

1. Размер сети.
2. Связанность сети. Среднее количество соседей для каждого узла.
3. Скорость изменения сети.
4. Полоса пропускания соединений.
5. Частота появления однонаправленных связей.
6. Модель передвижения узлов сети. Узлы сети могут передвигаться совершенно по-разному. Будь то пешеход, автомобиль, спасатель в экстренной ситуации или микроволновая печь являющаяся частью умного дома.

## 3.2. Общие для всех сценариев положения

В данной работе сравнению подвергаются следующие протоколы маршрутизации: Motorola Mesh (реактивный и проактивный режимы), HWMP (реактивный и проактивный режимы) в составе стандарта IEEE 802.11s, AODV. Также в некоторых сценариях подразумевается использовать режим работы WiFi Ad Hoc без алгоритмов маршрутизатии как эталон, для сравнения.

Во всех тестовых сценариях используются одинаковые настройки переменных симулятора:

* Стандарт WiFi: 802.11g.
* Модель распространения радиоволн: логарифмическая.
* Показатель ослабления: 3.
* Относительное расстояние: 1м.
* Относительная потеря мощности: 46,677 дБ.
* Механизм CTS/RTS не используется.
* Алгоритм адаптации скорости передачи: ARF.

В наших экспериментах было решено остановиться только на двух метриках: двусторонняя задержка и пропускная способность. Данный выбор обусловлен тем, что выбранные метрики являются наиболее часто применяемыми, их легко получить на реальном оборудовании, что при дальнейших исследованиях позволит подтвердить либо опровергнуть полученные в данной работе результаты. Также не больше количество метрик обусловлено экономическими и практическими соображениями.

## 3.3. Описание первого тестового сценария

**Начальное состояние:**

Линейная топология, расстояние между узлами равно 0.55X, где X – максимальное расстояние для выбранного стандарта WiFi (в случаях использования проактивных протоколов, главным узлом назначается узел с номером равным [S/2]).

**Действия:**

Первый узел сети начинает передачу данных последнему. В ходе эксперимента количество узлов варьируется (2, 3, 4, 5, 6, 8, 10, 15, 20, 25, 30, 35, 40). Также варьируется время начала посылки данных (сразу после формирования сети, и через 100с. после). Каждую сборку необходимо запустить 30 раз с разными зернами генератора случайных чисел (1..30)

**Регистрируемы параметры:**

Двусторонняя задержка (RTT), пропускная способность (throughput) между первым и последним узлами.

**Комментарии:**

После проведения всех тестовых процедур и получения результатов, необходимо построить график зависимости двусторонней задержки от количества узлов, который покажет, насколько быстро работают алгоритмы относительно друг друга и контрольного режима работы Wi-Fi Ad Hoc. Так же необходимо построить график зависимости пропускной способности от количества узлов, по которому можно будет судить, как много служебной информации посылает каждый из протоколов.

## 3.4. Описание второго тестового сценария

**Начальное состояние:**

Линейная топология состоящая из 4 узлов, расстояние между узлами равно 0.55X, где X – максимальное расстояние для выбранного стандарта WiFi (в случаях использования проактивных протоколов, главным узлом назначается узел с номером равным 3).

**Действия:**

Первый узел сети начинает передачу данных последнему. Варьируется время начала посылки данных (сразу после формирования сети, и через 100с. после). Каждую сборку необходимо запустить 1000 раз с разными зернами генератора псевдослучайных чисел (1..1000)

**Регистрируемы параметры:**

Двусторонняя задержка (RTT) между первым и последним узлами

**Комментарии:**

После прохождения всех тестовых циклов и сбора всех данных, необходимо построить график зависимости двусторонней задержки от времени для каждого протокола и для эталонного Ad Hoc. Данный график поможет качественно оценить время нахождения пути и стабильность работы в целом.

## 3.5. Описание третьего тестового сценария

**Начальное состояние:**

Топология, состоящая из 4 узлов, схема изображена на рис. 3.1 (в случаях использования проактивных протоколов, главным узлом назначается узел с номером 4).

**Действия:**

Спустя 100 с. после формирования сети узел 1 начинает посылку данных узлу 4. Путь должен сформироваться через узел 2, т.к. путь через узел 3обладает худшей метрикой. Спустя 150 с. после формирования сети узел 2 пропадает. Тест необходимо запустить 1000 раз с разными зернами генератора псевдослучайных чисел (1..1000)

**Регистрируемы параметры:**

Двусторонняя задержка (RTT) между узлами 1 и 4. Время, потраченное алгоритмами на перестроение пути.

**Комментарии:**

После прохождения всех тестовых циклов и сбора всех данных, необходимо вычислить время, потраченное каждым алгоритмом на переключение пути.

В силу того, что двусторонняя задержка измеряется с помощью протокола ICMP с периодом измерения 1с., для вычисления времени нахождения нового пути предполагается сделать следующие шаги:

* Для каждой выборки необходимо найти первое после 150с. значение двусторонней задержки и его порядковый номер.
* Из порядкового номера найденной задержки вычесть 50 (момент пропадания узла 2 относительно начала измерения) и к полученному числу прибавить значение задержки деленное на 1000. Полученное число будет представлять собой время нахождения нового пути в секундах.
* Усреднить результаты по всем выборкам.

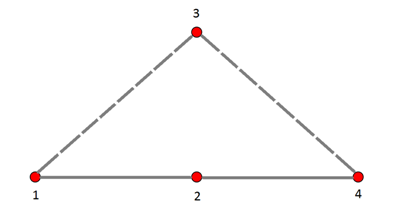


Рис 3.1. Топология сети для сценария №3 (сплошной линией показан путь, имеющий лучший показатель качества связи).

## 3.6. Описание четвертого тестового сценария

**Начальное состояние:**

Топология, состоящая из 4 узлов, схема изображена на рис.3.2 и рис.3.3 (в случаях использования проактивных протоколов, главным узлом назначается узел с номером 4).

**Действия:**

Спустя 100 с. после формирования сети узел 1 начинает посылку данных узлу 4. Путь должен сформироваться через узел 2, т.к. путь через узел 3обладает худшей метрикой. Через 150с. после формирования сети узел 2 начинает смещаться в сторону. Тест необходимо запустить 1000 раз с разными зернами генератора псевдослучайных чисел (1..1000)

**Регистрируемы параметры:**

Двусторонняя задержка (RTT) между узлами 1 и 4.

**Комментарии:**

После прохождения всех тестовых циклов и сбора всех данных, необходимо построить график зависимости двусторонней задержки от времени для каждого протокола.

Данный сценарий основан на патенте США № 7412241, на основе которого построены алгоритмы вычисления метрик путей в Motorola Mesh. Описанный в патенте метод основывается на таких показателях связи, как сила принимаемого сигнала и соотношение сигнал-шум. Также утверждается, что данный метод является эффективным и способен предсказывать разрывы соединения. С помощью описанного выше сценария мы проверим, насколько эффективен данный алгоритм, реализованный в ПО Motorola Mesh, и сравним его с другими протоколами такого же класса.

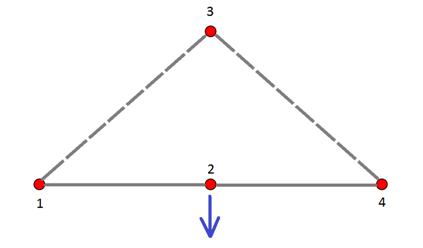


Рис 3.2. Топология сети для сценария №4 (сплошной линией показан путь, имеющий лучший показатель качества связи).

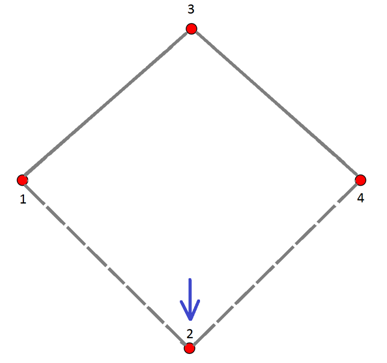


Рис 3.3. Топология сети для сценария №4 (сплошной линией показан путь, имеющий лучший показатель качества связи).

# 4. Анализ экспериментальных данных

## 4.1. Анализ результатов первого сценария

Проведение эксперимента было разделено на три стадии: измерение двусторонней задержки при старте измерений в момент начала симуляции, измерение двусторонней задержки спустя 100с. после начала симуляции и измерение пропускной способности спустя 100с. после начала симуляции. На каждой стадии было проведено 30 независимых измерений по 100с. в каждом, таким образом, размер выборки для каждой точки составил 3000.

Рис 4.1. График зависимости двусторонней задержки от количества узлов при старте измерений в момент начала теста

Результаты измерений первой стадии представлены на рисунке 4.1. На графике видно, что контрольные измерения Ad Hoc имеют ровную линейную зависимость и превзойти эти показатели не смог не один из протоколов. Ближе всех к контрольным измерениям оказался AODV, т.к. он не рассылал “Hello” сообщений, в то время как остальные алгоритмы оказались заметно выше на графике вследствие того, что в начальный момент сразу после начала симуляции, узлам требуется какое-то время на формирование сети, заполнение таблиц соседей с помощью “Hello” сообщений. Хуже всех в данном случае оказался MotoMesh с реактивным режимом, при увеличении количества узлов задержка резко возрастает. Это происходит потому, что в начальные моменты формирования сетей, кроме “Hello” сообщений протокол рассылает специальные служебные пакеты для измерения метрики путей.

Рис 4.2. График зависимости двусторонней задержки от количества узлов при старте измерений спустя 100с. после момента начала теста

Результаты измерений второй стадии представлены на рисунке 4.2. В данном случае картина кардинально изменилась, все результаты стали гораздо ближе друг к другу.

После анализа двух стадий первого сценария можно сделать следующие выводы: AODV является самым стабильным, его показатели не зависят от момента начал посылки данных; остальным алгоритмам требуется некоторое время на формирование сети для хорошей и быстрой работы; лучше всего себя показал MotoMesh в проактивном режиме, он единственный, кому удалось улучшить показатель задержки относительно контрольных измерений, однако на еще несформированной сети он показывает худшие результаты.

Рис 4.3. График зависимости относительной пропускной способности от количества узлов

Результаты измерения пропускной способности представлены на рисунке 4.3. На графике представлено отношение пропускной способности полученной для каждого протокола к пропускной способности, полученной в контрольных измерениях. Из полученных данных видно, что меньше всего на пропускную способность влияет AODV, MotoMesh потребляет примерно 10% полосы пропускания, а HWMP примерно 20%. Можно сделать выводы о том, что меньше всего служебного трафика по сети посылает AODV, больше всего HWMP.

Также по результатам первого сценария выяснилось, что у протокола HWMP число скачков ограничено 33. А в протоколе MotoMesh выяснилась следующая особенность работы: в реактивном режиме число скачков ограничено 21, при несформированной сети 5, в проактивном режиме при несформированной сети также наблюдается ограничение на 15 скачков. В отличие от HWMP, где данное ограничение установлено программно, в MotoMesh ограничение является особенностью работы алгоритма и программно не изменяется. Данное ограничение протокола является достаточно критическим и может сильно сказаться на применении данного протокола.

## 4.2. Анализ результатов второго сценария

В ходе эксперимента используя линейную топологию сети, состоящую из 4 узлов, для каждого протокола было проведено 1000 измерений по 100 секунд. Измерения проводились на сформировавшейся сети спустя 100с. после начала симуляции. Полученные результаты представлены на рисунке 4.4. в виде графиков зависимости средней двусторонней зависимости для каждого протокола от времени.

Из результатов можно сделать выводы, что реактивный режим протокола MotoMesh не отличается от других тестируемых реактивных протоколов (AODV, HWMP реактивный режим). А вот проактивный режим работы MotoMesh в выгодную сторону отличается от всех остальных протоколов. Он имеет максимально короткий переходной период и стабильную работу на протяжении всего времени измерения. Самую нестабильную работу показал проактивный режим HWMP, на протяжении всего времени измерения, наблюдаются достаточно большие скачи задержки от 4 до 16мс.

**Двусторонняя задержка, мс**

Рис 4.4. Графики зависимости двусторонней задержки от времени для протоколов сверху вниз: MotoMesh reactive, MotoMesh proactive, HWMP reactive, HWMP proactive, AODV

## 4.3. Анализ результатов третьего сценария

После проведения всех экспериментов третьего сценария были посчитаны среднеарифметические задержки на поиск нового пути, полученные результаты представлены на рисунке 4.5.

Рис 4.5. Диаграмма времени поиска нового маршрута для каждого алгоритма

Из полученных данных видно, что существенно более плохие результаты показывает протокол MotoMesh. Самой минимальной задержкой обладает проактивный режим протокола HWMP. Также можно заметить, что у обоих протоколов при прочих равных, проактивный режим показал лучшие результаты. Данный факт на практике показывает описанные ранее достоинства проактивного режима.

В данном эксперименте протокол MotoMesh работал со стандартными настройками, которые в данном случае оказались не эффективными, поэтому мы произвели новую настройку протокола и провели повторно тесты, полученный результаты представлены на рисунке 4.6.

Рис 4.6. Диаграмма времени поиска нового маршрута для каждого алгоритма

В ходе настройки были выставлены новые значения параметра GARBAGE\_TIMEUOT и NEIGBOUR\_TIMEOT, которые отвечают за время реагирования на изменение состояния связи между двумя узлами. Данные изменения помогли существенно улучшить исследуемы показатель.

Из полученных результатов можно сделать вывод, что HWMP лучше справляется с данной задачей, однако MotoMesh имеет преимуществом возможность тонкой настройки, что может быть полезно при его применении в самых различных ситуациях.

## 4.4. Анализ результатов четвертого сценария

Из результатов данного сценария были исключенные данные, полученные от протокола HWMP в силу его нестабильной работы. Результаты остальных протоколов представлены на рисунке 4.7.

Рис 4.7. График зависимости двусторонней задержки от времени для четвертого сценария

Из графика видно, что в момент 20с. от начала измерений, когда начал двигаться узел №2 параметры связи еще улучшались. Приблизительно в момент 70с. когда узел №2 и №4 оказались на одинаковом расстоянии, качество связи начало постепенно ухудшаться и значение задержки возросло до 8 секунд.

В момент 120с. наблюдается всплеск у всех алгоритмов, это вызвано тем, что этот момент срабатывает таймер у протокола верхнего уровня ARP и начинается пересылка дополнительных пакетов, что естественно влияет на общую задержку.

Ожидаемых результатов данный сценарий не дал, из графика не видно ни существенной разницы между протоколами, ни момента переключения пути, за исключением реактивного режима MotoMesh.

На графике реактивного режима протокола MotoMesh отчетливо виден всплеск в районе 139с., данный всплеск как раз соответствует моменту переключения на альтернативный маршрут.

На графике проактивного режима протокола MotoMesh в райное между 75 и 115 секундами видно повышение задержки, отличающее его от остальных алгоритмов. Данное повышение возникает вследствие того, что в тот момент года уровень сигнала до какого-нибудь узла падает, корневой узел начинает рассылать служебные пакеты, для более точного измерения метрики пути, вследствие чего и наблюдается повышение общей двусторонней задержки.

## 4.5. Общие выводы

После проведение всех тестовых сценариев можно сделать следующие выводы:

* Протокол AODV оказался самым стабильным и предсказуемым. В условиях проведенных исследований он показал все свои лучшие стороны. Его работа не зависит от сформированности сети, он одинаково хорошо работает и в первые моменты образования топологии и спустя время. Он показал лучшие значения при измерениях пропускной способности. Требуемое на поиск нового маршрута время, которое было измерено во втором сценарии, также оказалось лучше других реактивных протоколов участвовавших в сравнении. Данные показатели хорошо согласуются с его теоретическими характеристиками, так как при его разработке главными факторами стояли простота реализации и минимальные объемы служебной информации передаваемой по сети.
* Протокол HWMP, в моделируемых в данной работе сценариях, неожиданно оказался очень нестабильным в плане задержек при передачи данных. Также он больше всех влияет на пропускную способность канала связи, при его использовании скорость передачи данный падает на 20% относительно контрольных измерений. Данные результаты объясняются тем, что реактивный режим HWMP основан на протоколе AODV с добавлением новых технологий, которые влекут за собой больший объем предаваемой служебной информации, и проактивный режим вследствие своих алгоритмических особенностей передает больше служебных данных. Однако в тесте на скорость поиска нового маршрута, проактивный режим показал себя лучше всех, что также подтверждает его теоретические характеристики, именно снижение задержек при поиске путей и является главной задачей проактивных протоколов.
* Протокол Motorola Mesh показал себя хорошо в первом сценарии, где в проактивном режиме ему удалось уменьшить задержку даже относительно контрольных измерений, однако при несформированной сети данный протокол испытывает существенные трудности. Также проактивный режим оказался самым стабильным по результатам второго сценария. Реактивный режим мало чем отличился от других тестируемых в данной работе реактивных протоколов, что приводит нас к мысли о том, что данный класс алгоритмов маршрутизации достиг своих пределов, и улучшать дальнейшую работу протоколов можно только комбинируя несколько подходов к маршрутизации. Также в ходе исследований был выявлен ряд существенный недостатков описанных выше, что говорит нам о том, что работу над данным протоколом необходимо продолжить для устранения найденных дефектов.

# Заключение

В ходе выполнения работы нами были изучены общие подходы к маршрутизации в беспроводных самоорганизующихся сетях и построению протоколов маршрутизации. Изучены подходы к тестированию и симуляции сетевых протоколов. Проведена большая работа по переносу ПО Motorola Mesh в среду системы симуляции компьютерных сетей NS-3. Разработаны и промоделированы тестовые сценарии для сравнения протоколов маршрутизации.

По результатам проведенных исследований были выявлены некоторые характеристики изучаемых протоколов, которые могут быть полезны при дальнейших исследованиях в данной области или при применении подобных технологий на практике.

Также в ходе исследования был обнаружен ряд дефектов в ПО Motorola Mesh, исправление которых в дальнейшем может существенно повысить качество работы протокола.

# Библиографический список

1. Маршрутизация в беспроводных Ad hoc-сетях / В. М. Винокуров и др. Доклады ТУСУРа. – 2010, №2 (22), часть 1. – с. 288-292
2. Маршрутизация в широкополосных беспроводных mesh-сетях стандарта IEEE 802.11s / В. Вишневский и др. – ЭЛЕКТРОНИКА: НТБ, 2008, №6, с. 64-68
3. Ad hoc On-Demand Distance Vector (AODV) Routing. / C. Percins, E. Belding-Royer, S. Das – IETF RFC 3561, July 2003.
4. IEEE Std 802.11-2012. IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area network – Specific requirements – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications / IEEE, March 2012
5. Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations / S. Corson, J. Macker – IETF RFC 2501, January 1999
6. NS-3 Website: <https://www.nsnam.org/>
7. Unatet States Patent No.: US 7412241 B2 Method to provide a measure of link reliability to a routing protocol in an Ad Hoc wireless network / T. Strutt, Date of Patent: Aug. 12, 2008