ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ
Ордена Трудового Красного Знамени федеральное государственное бюджетное
образовательное учреждение высшего образования
«Московский технический университет связи и информатики»

На правах рукописи

Кулаков Михаил Сергеевич

Разработка принципов организации мобильных сетевых структур в авионике

Специальность 05.12.13 – Системы, сети и устройства телекоммуникаций

Диссертация
на соискание ученой степени кандидата
технических наук

Научный руководитель:
доктор технических наук,
доцент Шаврин Сергей Сергеевич

Москва - 2017

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность работы. Традиционными средствами управления воздушным движением (УВД) является радиолокация, радиопеленгация и голосовая радиосвязь. В дополнение к ним, в настоящее время, внедряются технологии цифровой передачи данных, призванные повысить эффективность УВД для современного авиационного трафика и активного развития беспилотной авиации [1]. Одной из таких технологий является «радиовещательное автоматическое зависимое наблюдение» - АЗН-В [2]. Её концепция заключается в периодической передаче данных о местоположении и намерениях участников воздушного движения. На территории Российской Федерации осуществляется внедрение двух стандартов, реализующих технологию АЗН-В, предназначенных для обеспечения *ситуационной осведомлённости* экипажа воздушного судна (ВС) и авиадиспетчеров - VDL Mode 4 и 1090ES [3].

Проблема ситуационной осведомленности существует, в первую очередь, в отдаленных и океанических регионах из-за низкой плотности покрытия наземными средствами контроля и управления воздушным движением. Построение телекоммуникационной сети между участниками воздушного движения сможет обеспечить решение этой проблемы путём передачи данных АЗН-В от воздушных судов, находящихся за пределами прямого приёма пунктами УВД или *базовых станций (БС)*, реализующих технологию АЗН-В, в связи с этим тема является актуальной.

Перспективный подход при построении такого рода сети - применение протоколов мобильных самоорганизующихся сетей, т.к. они обладают рядом свойств, необходимых для функционирования динамических сетевых структур: автоконфигурация, самооптимизация, самовосстановление и при этом не требуют наличия жёсткой иерархии сетевых узлов [4].

Стандарты 1090ES и VDL Mode 4 ориентированы на передачу данных от различных систем авионики, поэтому они могут быть использованы для построения на их основе самоорганизующейся сети передачи данных АЗН-В. Однако, в связи с тем, что пригодность указанных стандартов для реализации на их основе мобильной самоорганизующейся сети до сих пор не была подтверждена теоретическими исследованиями и практическими реализациями, то существует задача проверки такой возможности.

Степень разработанности темы. Среди ученых, систематизировавших вопросы функционирования самоорганизующихся телекоммуникационных сетей, хотелось бы особо отметить: А.Е. Кучерявого, А.И. Парамонова, А. Букерша, Ш. Раджива, Х. Лабиода [5, 6, 7, 8, 9, 10, 11]. Вопросы производительности методов доступа к среде для мобильных

самоорганизующихся сетей рассмотрены в публикациях С. Эйхлера, К. Билструпа, Е. Улемана, Г. Штрёма и других [12, 13]. Применение протоколов мобильных самоорганизующихся сетей для транспорта и последние разработки в данной области представлены в публикациях А.В. Абилова, А.В. Рослякова, Й. Лиу, К.Е. Перкинса, М. Фога, Е. Паломара, В. Наумова, Б. Карпа, Х.Т. Кунга и иных исследователей [14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26].

К теме данной работы наиболее близки публикации исследовательской группы Ф. Хоффмана, Д. Медины, А. Волица, С. Аяза, К.Х. Рокитански [27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38]. В работах группы рассматривается построение мобильной самоорганизующейся сети между участниками воздушного движения с целью обеспечения пассажирам на борту доступа в сеть Интернет. В качестве стандарта связи предлагается стандарт LDASC1 [39], находящийся на стадии разработки и не принятый к внедрению на территории РФ. Общий подход к осуществлению маршрутизации сообщений схож, а в качестве одного из алгоритмов маршрутизации рассматривается «жадный» алгоритм, но при этом отсутствует математическое доказательство применимости алгоритма.

Целью диссертационной работы является повышение ситуационной осведомленности пунктов УВД в отдаленных и океанических регионах путём создания мобильной самоорганизующейся сети между участниками воздушного движения для передачи данных АЗН-В. Для достижения цели решены следующие задачи:

1. Проведены аналитические исследования связности сетевых узлов в отдаленных и океанических регионах по реальным полётным данным с учётом особенностей физического уровня стандарта VDL Mode 4: частота канала, мощность передатчика, чувствительность приёмника, а также с учётом ослабления сигнала при распространении и воздействия помех;
2. Разработан протокол маршрутизации, позволяющий передавать сообщения АЗН-В наземным пунктам УВД в условиях низких связности сети и пропускной способности канала;
3. Разработана дискретно-временная модель, в которой совмещены функциональная модель стандарта VDL Mode 4 и разработанный протокол маршрутизации. При реализации модели разработан комплекс алгоритмов решающего устройства, доступа к среде, генерирования трафика АЗН-В и взаимодействия уровней стандарта VDL Mode 4;
4. Проведено имитационное моделирование работы сети для различных сценариев движения сетевых узлов – участников воздушного движения. По

результатам произведена оценка производительности сети по числу отправленных и полученных сообщений, задержкам передачи сообщений и общему количеству узлов, от которых были получены сообщения АЗН-В.

Методы исследований. При выполнении исследований были использованы методы математического моделирования, теории телекоммуникационных сетей, теории вероятностей и имитационного моделирования.

Научная новизна работы.

1. Предложен метод повышения ситуационной осведомленности систем УВД в отдаленных и океанических регионах, основывающийся на применении алгоритмов самоорганизующихся сетей для стандарта авиационной связи VDL Mode 4.
2. Разработан протокол маршрутизации самоорганизующейся телекоммуникационной сети для авиационного стандарта связи, обеспечивающий передачу данных в условиях низкой связности сети и низкой пропускной способности каналов связи. Достоинствами протокола являются: отсутствие необходимости получения данных обо всех узлах сети и необходимости использования дополнительных методов обхода сетевого графа, простота реализации, а также функционирование на любом транспортном средстве, оборудованном приёмопередатчиком VDL Mode 4.
3. Разработана дискретно-временная имитационная модель самоорганизующейся телекоммуникационной сети, построенной между участниками воздушного движения, а также пунктами УВД, учитывающая характер движения узлов, распространение сигнала и функциональную модель стандарта VDL Mode 4.

Теоретическая и практическая значимость работы. Теоретическая значимость работы заключается в разработке и исследовании модели мобильной самоорганизующейся сети, функционирующей на основе авиационного стандарта связи и построенной между участниками воздушного движения и пунктами наблюдения в отдаленных и океанических регионах.

Практическая значимость работы заключается в разработке алгоритмов и структур данных, необходимых для функционирования канального уровня приёмопередатчиков стандарта VDL Mode 4, которые могут быть использованы при их технической реализации.

Использование результатов имитационного моделирования позволит сократить количество БС стандарта VDL Mode 4 в отдаленных и океанических регионах, а также

планировать маршруты полётов с учётом связности сети для обеспечения приёма сообщений АЗН-В от участников движения, находящихся за пределами прямой видимости.

Результаты диссертационной работы использованы в ФГУП «ГосНИИАС» (г. Москва): 1) при выполнении НИР «Модем» (отчет № 1601/13 от 28.07.2015 "Разработка технологий создания авиационных информационно-управляющих систем на основе транспондера АЗН-В, работающего в режиме VDL 4"); 2) в НИР «Айсберг» (отчёт № 147(16649)2015 от 01.08.2015); 3) в НИР «Исследование-Норма-Транспорт-2» (гос. контракт № 107131030010 от 25.07.2013, ФГУП «Морсвязьспутник», г. Москва). Акт об использовании представлен в Приложении В к диссертационной работе.

Положения, выносимые на защиту

1. Применение алгоритмов маршрутизации самоорганизующихся сетей на основе стандарта VDL Mode 4 способствует повышению ситуационной осведомленности систем УВД за счёт передачи данных АЗН-В от участников воздушного движения, находящихся за пределами прямой видимости систем УВД. Периоды получения данных через сеть могут составлять от нескольких минут до нескольких часов, и зависят от плотности воздушного движения и маршрутов полёта.
2. VDL Mode 4 является самым перспективным стандартом, внедряемым на территории РФ, для создания на его основе самоорганизующихся телекоммуникационных сетей передачи данных о местоположении и намерениях участников воздушного движения. В отличие от стандарта 1090ES он предоставляет возможность связи «борт-борт», поддерживает процедуры управления соединением, процедуры передачи пользовательской информации и использует метод доступа к среде, позволяющий каждому приёмопередатчику резервировать собственные слоты для передачи сообщений с низкой вероятностью коллизий.
3. Разработанная имитационная модель мобильной самоорганизующейся телекоммуникационной сети, построенной между участниками воздушного движения и пунктами УВД на основе стандарта VDL Mode 4, позволяет моделировать мобильность узлов по реальным полетным данным, учитывает изменение сигнала при распространении, вероятность возникновения битовых ошибок и особенности функциональной модели стандарта VDL Mode 4. С помощью модели можно оценить такие показатели производительности сети, как количество отправленных и полученных сообщений, задержки при передаче сообщений по сети и число узлов, от которых поступили сетевые сообщения с данными АЗН-В.

4. Применение адаптивных значений, предложенных численных параметров - диапазона выбора временного слота, периода вещания сетевых сообщений БС и периода хранения записей в таблице маршрутизации, в зависимости от количества сетевых узлов, позволяют повысить эффективность работы сети, выражаемую в количестве отправленных и полученных сообщений, задержках при передаче сообщений по сети и число узлов, от которых поступили сетевые сообщения с данными АЗН-В.

Достоверность полученных результатов обеспечена соответствующим применением используемых математических методов, правильностью постановки решаемых задач, а также используемых допущений и ограничений. Подтверждена представлением и обсуждением полученных научных результатов на научно-технических конференциях, публикацией основных результатов работы в рецензируемых журналах, соответствием применяемых моделей физическим процессам в самоорганизующихся телекоммуникационных сетях, средствами имитационного моделирования.

Апробация результатов работы. Основные результаты диссертационного исследования были доложены: на 7-й, 8-й, 9-й, 10-й международной отраслевой научно-технической конференции «Технологии информационного общества» (МТУСИ, г. Москва, 2013, 2014, 2015, 2016) [40, 41, 42, 43], на международной научно-технической конференции «Фундаментальные проблемы радиоэлектронного приборостроения INTERMATIC» (МИРЭА, г. Москва, 2012, 2013) [44, 45], на всероссийской научно-практической конференции «Моделирование авиационных систем» (ФГУП «ГосНИИАС», г. Москва, 2013) [46], на международной молодежной научно-практической конференции «ИНФОКОМ» (СКФ МТУСИ, г. Ростов-на-Дону, 2013) [47], а также на 70й региональной научно-технической конференции студентов, аспирантов и молодых ученых «СТУДЕНЧЕСКАЯ ВЕСНА» (СПбГУТ, г. Санкт-Петербург, 2016) [48].

По теме диссертационного исследования опубликовано 13 печатных работ [46, 47, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58], из них 4 в рецензируемых периодических изданиях, входящих в перечень ВАК Минобрнауки России. Основные результаты по теме диссертации получены автором лично.

Структура и объем работы. Диссертация состоит из введения, 5 глав, заключения, списка сокращений, библиографического списка, включающего 105 наименований, и трёх приложений. Работа содержит 186 страниц текста , 55 рисунков и 24 таблицы.

РАЗДЕЛ 1. АНАЛИЗ ОСОБЕННОСТЕЙ РАБОТЫ СИСТЕМ УПРАВЛЕНИЯ ВОЗДУШНЫМ ДВИЖЕНИЕМ И СТАНДАРТА VDL MODE 4. МОБИЛЬНЫЕ САМООРГАНИЗУЮЩИЕСЯ AD HOC СЕТИ

1.1. Управление воздушным движением. АЗН-В

До конца 20х годов прошлого века, полеты осуществлялись только в условиях хорошей видимости земли с самолёта. Если погодные условия значительно ухудшались и пилот не мог чётко видеть землю, полет откладывался. В результате развития радионавигационных средств и различных приборов на борту самолета, появилась возможность совершать так называемые "полеты по приборам". Не смотря на это, до конца 60х годов, считалось необходимым, чтобы пилот мог наблюдать землю при совершении маневров при посадке и на последних ста метрах высоты. При пилотировании как пассажирских, так и военных самолетов, пилоты совершают свои действия с учётом инструкций, получаемых от авиадиспетчеров из пунктов управления воздушным движением. В основном указывается высота полёта для того, чтобы исключить вероятность авиакатастрофы [59].

Управление полетом самолета, основано на трёх основных средствах аэронавигации: радиолокации, радиопеленгации и аналоговой голосовой радиосвязи.

1. Радиолокация. Определение местоположения, направления и габаритов летательного аппарата (ВС) средствами радиолокации вблизи аэропортов и особых мест, требующих контроля воздушного пространства. Данные используются диспетчерами для осуществления корректировки движения ВС.
2. Радиопеленгация. Для систем управления полетом и заходом на посадку используется система стационарных радиомаяков, которые размещены в аэропортах, а также в важных точках вдоль маршрутов воздушного движения. Бортовое оборудование ВС принимает посылаемые, на соответствующей частоте, сигналы радиомаяка. Для обеспечения надежной идентификации определенного радиомаяка радионавигационные сигналы периодически сменяются опознавательными сигналами, при этом характеристики радиосигналов зависят от отрезка маршрута, по которому следует ВС. Например, интенсивность передачи радиосигналов может варьироваться в зависимости от расстояния ВС до радиомаяка. Это позволяет пилоту точнее направлять свой самолет, по нужному маршруту. Лётчик получает данные о направлении, по которому располагается радиостанция, передающая сигнал от

всенаправленного пеленгатора. Соответствующий азимут, относительно радиостанции, указывается индикатором на приборной панели самолёта. Дальномерная аппаратура, измеряющая расстояние от ВС до радиостанции, также является важным элементом всенаправленного пеленгатора. В совокупности с азимутом, получаемым с помощью радиостанции, это позволяет летчику точно определить свое положение.

3. Аналоговая голосовая радиосвязь используется для осуществления переговоров между авиадиспетчерами и лётчиками. При этом используется полоса радиочастот 108 – 137 МГц, а ширина канала составляет 25 кГц. Для передачи голосовых сигналов используется аналоговая амплитудная модуляция.

Управление воздушным движением (УВД) в РФ — это координирование, планирование и организация движения воздушных судов, выполняющих полёты или движущихся по земле (аэродрому) с целью совершения взлётно-посадочных операций [60]. Цель УВД — обеспечение эффективности, регулярности и безопасности полётов. В СССР, согласно Воздушному кодексу, УВД возлагалось на органы Единой Системы УВД (ЕС УВД), а также на различные ведомственные органы УВД, в пределах, установленных для них районов и зон. Созданная вначале 70-х гг. ЕС УВД, занимает ведущее место и в действующей системе управления. К тому времени, управление полётами гражданских и военных воздушных судов, которые выполняются практически в одном и том же воздушном пространстве, принадлежали различным ведомствам. Однако, постепенно возрастая, плотность и интенсивность воздушного движения достигли такого уровня, что, их согласование и координация с различными пунктами управления, стали затруднительными. Ради удовлетворения интересов безопасности, в рамках ЕС УВД, произошло объединение гражданских и военных органов УВД.

На сегодняшний день, заметная доля исследований и разработок посвящена вопросам создания и обеспечения эффективного функционирования интеллектуальных коммуникационных систем контроля и управления движением различных транспортных средств. Согласно данным компании exxonmobil число личных легковых автомобилей к 2040 составит 1600 млн. Это число больше, чем в 2 раза, по сравнению с 2010 годом[61]. По прогнозу компании Boeing, в период с 2007г. до 2026 г., авиакомпании разных стран мира приобретут 28600 новых самолетов. Это также соответствует приросту числа самолетов примерно в 2 раза [62]. Кроме этого, ожидается прирост числа ВС малой (частной) авиации и прирост числа беспилотных летательных аппаратов (БПЛА), в процессе их развития и расширения сфер применения. В совокупности это привело к необходимости внедрения

новых цифровых технологий передачи данных, которые должны дополнять традиционные средства УВД. Одной из таких технологий является «автоматическое зависимое наблюдение», широко представленное двумя видами систем: 1090ES (extended squitter) [63] и VDL (Very High Frequency Data Link). VDL, в русскоязычной терминологии ICAO - ОВЧ ЛПД (очень высокой частоты линия передачи данных), имеющая несколько редакций – Режим (Mode) 2[64], Режим 3 [65], Режим 4 [66, 67, 68, 69, 70, 71, 72, 73, 74].

Концепция автоматического зависимого наблюдения подразумевает, что каждый участник движения периодически сообщает своё местоположение и намерения. По методу передачи данных о местоположении стандарты автоматического зависимого наблюдения разделяются на два вида:

1. Уже отмеченное радиовещательное автоматическое зависимое наблюдение (АЗН-В) – участник движения передает навигационные данные в широковещательном режиме.
2. Автоматическое зависимое наблюдение контрактное (АЗН-К) – участник движение арендует спутниковые каналы и использует их для передачи навигационных данных наземным станциям.

Так как одновременно в одном воздушном пространстве могут использоваться несколько стандартов АЗН-В, существует дополнительная концепция автоматического зависимого наблюдения ретрансляционного (АЗН-Р), т.е. обмена данными между системами разных стандартов.

1.2 VDL Mode 2, Mode 3, Mode 4

1.2.1 VDL Mode 2

Является стандартом цифровой передачи данных в УКВ канале, основанным на эталонной модели взаимодействия открытых систем (ЭМВОС).

В стандарте VDL Mode 2 определен функционал трех первых уровней ЭМВОС.

1) Физический уровень обеспечивает переключение приёмопередатчика на частоты каналов, передачу и приём сигнала, а также функцию уведомления об успешности совершения этих действий. В качестве модуляции, техническая документация определяет для стандарта VDL Mode 2 дифференциальную восьмипозиционную манипуляцию фазовым сдвигом (D8PSK) со скоростью модуляции 10,5 кБод или скоростью передачи 31,5 Кбит/с.

2) На канальном уровне обеспечивается надежная передача сообщений с данными АЗН-В, а также доступ к среде передачи. Канальный уровень разделён на два подуровня и имеет объект управления. Подуровень доступа к среде передачи (MAC) использует метод многостанционного доступа с контролем несущей и избеганием коллизий (CSMA/CA). Подуровень сервисов канала данных (DLS) использует протокол управления авиационным УКВ каналом (AVLC) и является разновидностью высокоуровневого протокола управления каналом передачи данных (HDLC). Основные функции данного протокола – обнаружение и исправление ошибок, повторная передача искаженных сообщений, сборка и разборка сообщений.
3) Сетевой уровень реализован частично и предназначен для обеспечения доступа пользователей к подсети Aeronautical Telecommunication Network (ATN). Этот уровень отвечает за передачу сетевых сообщений в сеть, контроль ошибок передачи, управление потоком данных, фрагментацию данных и сборку сообщений, а также управление соединениями. Основными протоколами подуровня являются режим связи точка-точка и адресная широковещательная передача данных от наземной станции к нескольким мобильным (бортовым) станциям. Подробное описание уровня стандарта VDL Mode 2 находится в документе ISO 8028.

В своё время, VDL Mode 2 должен был заменить некоторые существующие аэронавигационные системы или улучшить их работу благодаря использованию его физического канала.

На рисунке 1 изображено соответствие уровней VDL Mode 2 уровням модели ЭМВОС (OSI).

<table>
<tr><td rowspan="5">Верхние уровни ISO</td><td colspan="2">Приложение</td><td>Уровень 7</td></tr>
<tr><td colspan="2">Представление</td><td>Уровень 6</td></tr>
<tr><td colspan="2">Сеанс</td><td>Уровень 5</td></tr>
<tr><td colspan="2">Транспортировка</td><td>Уровень 4</td></tr>
<tr><td colspan="2">Межсетевой обмен</td><td rowspan="2">Уровень 3</td></tr>
<tr><td rowspan="4">VDL-2</td><td colspan="2">ISO 8208</td></tr>
<tr><td>LME</td><td>DLS (AVLC)</td><td rowspan="2">Уровень 2</td></tr>
<tr><td colspan="2">MAC (CSMA)</td></tr>
<tr><td colspan="2">D8PSK 31,5 Кбит/с</td><td>Уровень 1</td></tr>
</table>

Рисунок 1. Соответствие уровней VDL Mode 2 уровням модели OSI

1.2.2 VDL Mode 3

VDL Mode 3 является стандартом цифровой передачи данных в авиационном УКВ диапазоне. Стандарт обеспечивает цифровую связь между мобильными станциями (ВС и автомобилями в аэропорте), также между мобильными станциями и стационарными наземными станциями. Технические характеристики схожи со стандартом VDL Mode 2, т.е.:

- тип модуляции D8PSK;
- метод многостанционного доступа с временным разделением канала (Time Division Multiple Access - TDMA);
- скорость передачи информации 31,5 Кбит/с;
- определены физический и канальный (с внутренним разделением) уровни модели ЭМВОС.

Основные отличия от стандарта VDL Mode 2: предусмотрена передача голосовых сообщений в цифровом виде; ширина доступного для использования частотного диапазона увеличена на 1 МГц. Также в документах VDL Mode 3 описан принцип взаимодействия между ВС и наземными (базовыми) станциями, когда наземные станции назначают временные слоты для приёмопередатчиков летательных аппаратов. Стандарт VDL Mode 3 полностью совместим со стандартом VDL Mode 2.

1.2.3 VDL Mode 4

VDL Mode 4 является стандартом цифровой передачи данных в авиационном УКВ диапазоне. Обеспечивает информационный обмен между мобильными станциями (ВС и автомобилями в аэропорте) и между мобильными станциями и стационарными наземными (базовыми) станциями. VDL Mode 4 даёт возможность участникам воздушного движения эффективно обмениваться краткими периодическими сообщениями с низкой вероятностью коллизии и способен поддерживать критические по времени прикладные задачи.

VDL Mode 4 передает цифровые данные, используя стандартные авиационные УКВ-каналы с разнесением 25 кГц и метод самоорганизующегося многостанционного доступа с временным разделением (Self-organized Time Division Multiple Access - SoTDMA). Метод TDMA делит канал связи на временные сегменты, выделяя сначала фрейм, который затем уже делится на интервалы времени (слоты). Начало каждого слота представляет собой возможность станции вести передачу. VDL Mode 4 способен управлять ситуацией при

перегрузке (то есть, когда требуется больше слотов, чем доступно в данное время) и приспосабливаться к существующему движению в управляемом и безопасном режиме.

Минутный фрейм VDL Mode 4 делится на большое число (4500) кратких временных слотов длительностью 13,33 мс, синхронизированных на основе глобального времени UTC. Каждый слот может быть использован приёмопередатчиком, установленным на борту ВС, на автомобиле или на стационарной наземной станции, для передачи различных данных. В каждом сообщении передается информация о резервировании слота, таким образом, запланированное время использования слотов для передачи известны всем пользователям, которые находятся в зоне прямого приёма друг от друга**.** Данный поход делает эффективным использование канала передачи данных, т.к. пользователи не ведут передач одновременно. Подобный самоорганизующийся протокол доступа к среде для канального уровня VDL Mode 4, не требует участия наземной инфраструктуры, поэтому стандарт может обеспечивать обмен данными широковещательно и с адресацией, как в режиме “борт-борт”, так и в режиме “земля-борт”.

Каждый участник воздушного движения (ВС, автомобили в аэропорте и наземные станции) снабжен приёмопередатчиком для определения местоположения и времени, который управляет информационным обменом по каналу связи, передает и получает данные.

На рисунке 2 показана упрощенная структура базового приёмопередатчика VDL Mode 4. Эта структура одинакова для бортового и наземного исполнения.

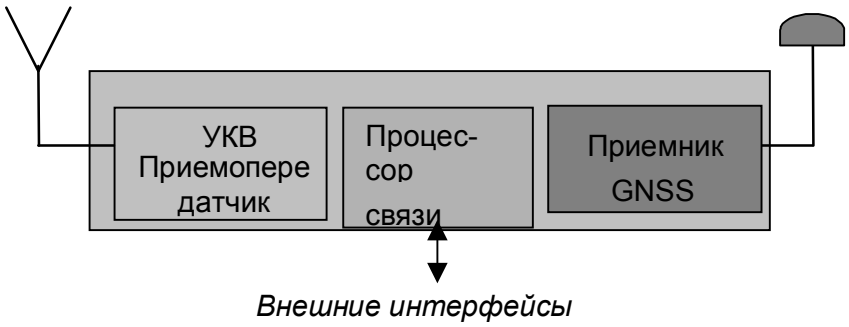


Рисунок 2. Упрощенная структурная схема транспондера VDL Mode 4

Реально функционирующая архитектура может отличаться от той, что показана на рисунке. Например, на ВС АОН (авиации общего назначения) может быть установлен один интегрированный блок транспондера, что соответствует приведенной структуре, а в установке на транспортном ВС может применяться внешний приемник GNSS или же могут применяться данные о навигации и времени, поступающие от других навигационных систем. На транспортных ВС практически всегда используется дублированное оборудование для обеспечения резервирования, а также могут использоваться несколько УКВ-установок и

антенны для обеспечения расширения коммуникационных способностей. На стационарной наземной станции может включать эталонный приемник GNSS (GNSS reference receiver) для организации передачи поправок с земли на борт.

Транспондер может быть сопряжен с различными внешними устройствами, такими как дисплей, компьютеры и базы данных.

Приемник GNSS обеспечивает данные о положении и времени по всему земному шару. Оба эти компонента имеют большое значение для работы приёмопередатчика. Обычно данные о времени поступают от системы GNSS, но они могут также поступать и от другого источника, например, от бортовых атомных часов.

УКВ приемопередатчик в мобильной установке используется как для передачи собственных координат и другой соответствующей информации к другим пользователям (включая наземные станции), так и для получения данных от других пользователей. Приемопередатчик может работать на каналах с разнесением 25 кГц. Являясь стационарной, наземная станция также будет передавать свои координаты в режиме вещания с регулярными интервалами.

В минимальной конфигурации, УКВ-приемопередатчик состоит из одного передатчика и двух приемников, способных вести мониторинг разных каналов одновременно, обычно каналов типа GSC (Global Signalling Channels). Способность вести прием и передачу по разным каналам одновременно зависит от разнесения частоты между каналами, количества и размещения антенн на ВС и типа радиоустановки на борту. Точные требования к одновременному приему/передаче данных по разным каналам будут также зависеть от необходимости поддерживать различные прикладные функции стандарта связи. Транспортные наземные средства аэропорта обычно работают на одном канале. Этот может быть тот же канал, что используется для связи с ВС на земле. Однако по оперативным причинам может применяться и отдельный канал.

Как упоминалось, VDL Mode 4 функционирует в авиационном УКВ-диапазоне, то есть на частотах 108-136,975 МГц. Приёмопередатчик использует алгоритм избирательности для УКВ-частот, позволяющий станции выбирать наиболее мощный сигнал из двух перекрывающих друг на друга сигналов, что обеспечивает эффективное повторное использование временных слотов и спектра.

Для использования в глобальном масштабе предусматривается пара глобальных каналов сигнализации. Этих каналов достаточно для обеспечения функций управления и организации воздушного движения ATM (Air Traffic Management) в большинстве регионов. Однако может возникнуть необходимость в дополнительных локальных каналах (Local

Signalling Channels - LSC) для перегруженных зон терминалов, а также в аэропортах с высокой плотностью движения. При этом возможно, что могут потребоваться дополнительные УКВ-каналы для передачи восходящих и нисходящих сообщений с прикладными данными.

Устройством, координирующим использование каналов связи, является процессор связи, или иными словами компьютер. Процессор связи соединен с УКВ-приемопередатчиком и приемником GNSS. В своей памяти он хранит виртуальное изображение фрейма временных слотов. Он осуществляет передачу полученных от приемника GNSS данных о местонахождении с учетом информации по синхронизации, исходя из секундного импульса UTC.

Процессор связи следит за распределением слотов в интересах выполнения собственных передач станции. Он постоянно обновляет собственную карту слотов и выбирает слоты из свободных слотов на карте, или повторно использует слоты удаленных станций.

Работа стандарта VDL Mode 4 строится на основе следующих функциональных базовых элементах:

1) Физического уровня, предназначенного для обеспечения процесса передачи цифровых данных по каналам связи авиационного диапазона.
2) Структуры фрейма на основе принципа TDMA (коллективный доступ с временным разнесением каналов).
3) Эталонной синхронизации времени, позволяющей установить единый маркер начала каждого слота связи.
4) Данных о местонахождении используемых для организации доступа к слотам.
5) Гибкой структуры сообщения, которая способна поддерживать широкий диапазон протоколов передачи данных и вещательных передач.
6) Функции выбора слота, определяющей, когда станция получит доступ к каналу, и хранящая информацию по текущему и планируемому выделению слотов.
7) Различных режимов работы, обеспечивающих автономный или управляемый доступ к слотам.
8) Функции управления доступом к слотам, контролирующей использование каждого слота. Стандарта VDL Mode 4 поддерживает:

- автономное управление доступом, позволяющее станциям получать доступ к кванту без обязательного контроля со стороны ведущей станции;

- несколько схем управляемого доступа, позволяющих станциям выделять слоты другим станциям и наземным станциям контролировать общий доступ к слотам.

Документация VDL Mode 4 основывается на эталонной модели взаимосвязи открытых систем OSI, разработанной Международной организацией по стандартизации ISO. Уровневая структура, применяемая для режима VDL Mode 4, показана на рисунке 4.

Режим VDL Mode 4 поддерживает манипуляцию GFSK с частотой модуляции 19200 бит/с.

В VDL Mode 4 канальное время делится на слоты времени определенной длины. «Суперфрейм», который является важным термином в управлении каналами VDL Mode 4, состоит из группы слотов, охватывающих период в 60 с. «Суперфрейм» содержит 4500 слотов (что равно 75 слотами в секунду). Это показано на рисунке 3.

Каждый слот доступен для ведения приема и передачи любой станцией, работающей на канале передачи данных. Одно сообщение о положении может занимать один слот времени на линии связи. Другие передачи могут занимать больше одного слота в зависимости от прикладной задачи. Максимально допустимая длина передачи равна 1 секунде. Передача с такой максимальной длиной будет занимать 75 слотов.

VDL Mode 4 требует синхронизации времени для обеспечения доступа базовым станциям без взаимных помех. Стандартом времени для VDL Mode 4 является универсальное скоординированное время (UTC). Это время основано изначально на отсчете системы GNSS, но возможны и другие источники, если они соотносимы с UTC. Концепция синхронизации времени для VDL Mode 4 должна удовлетворять самым строгим требованиям к точности, непрерывности и целостности работы для авиации.

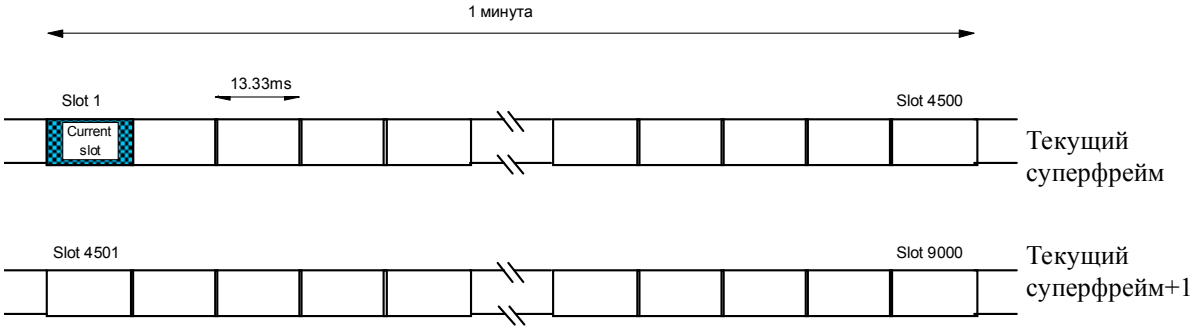


Рисунок 3. Структура фрейма TDMA

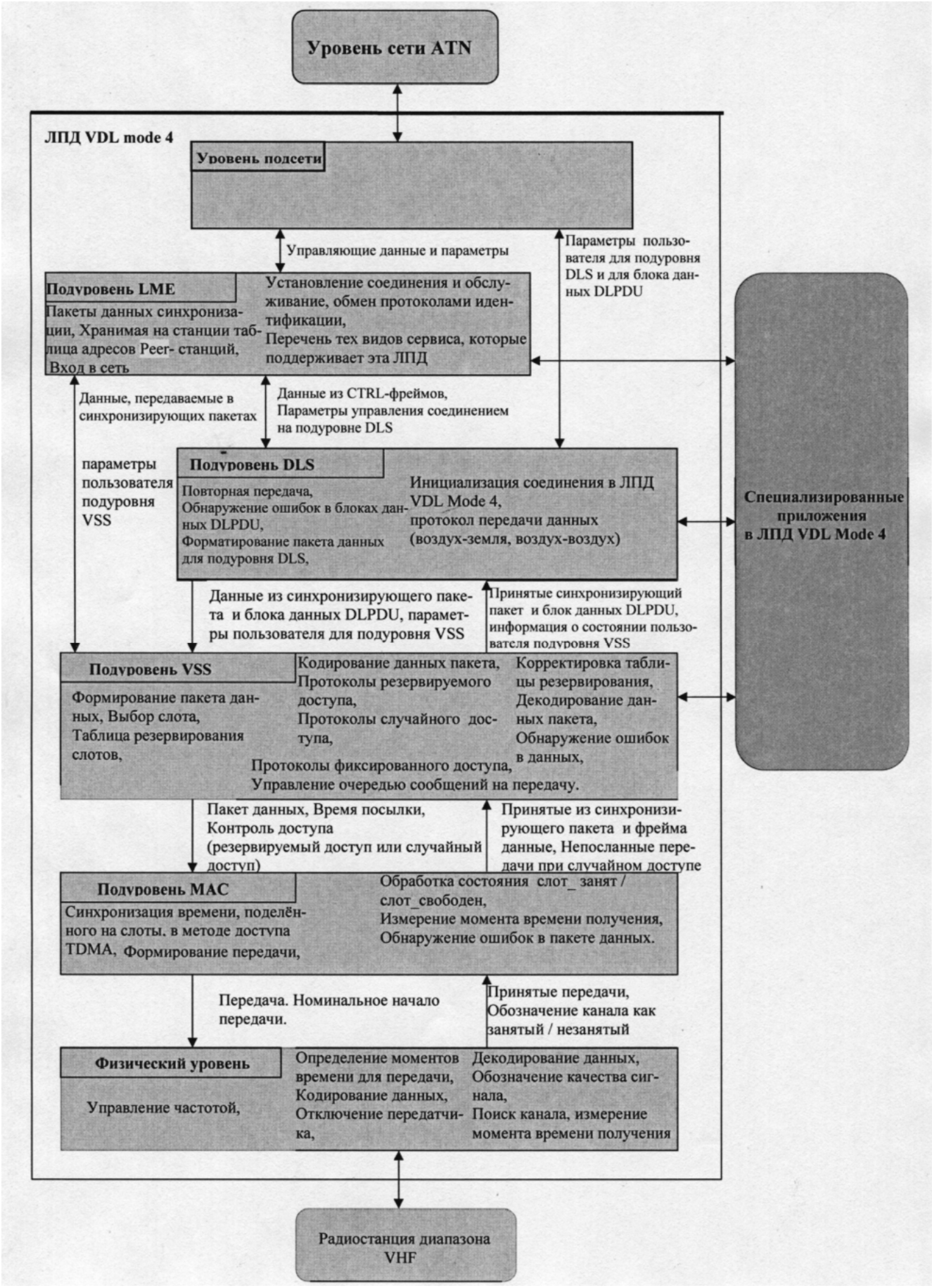


Рисунок 4. Уровневая организация VDL Mode 4 в соответствии с OSI

Для оценки времени по UTC применяются различные методы, например:

1. GNSS: Пользователь, снабженный приемником GNSS, может определить время UTC с точностью до 400 нс (2 сигма). Это первичный источник времени для VDL Mode 4, обеспечивающий независимость от наземных станций.
2. Сеть наземных станций: наземные станции VDL Mode 4 могут применяться для повышения охвата зоны наблюдения в режиме “борт-земля”. Эти наземные станции синхронизированы по UTC с помощью таймера времени GNSS, атомных часов или другими способами и передают в режиме вещания сообщения о синхронизации относительно UTC**.** Сообщения о временной синхронизации предоставляют пользователям возможности определения времени с точностью до микросекунды. Наземные станции режима VDL Mode 4 передают сообщения о синхронизации на регулярной основе для обеспечения гарантии доступности правильного времени.
3. Другие бортовые источники времени, включая атомные часы, также могут стать доступными для применения на борту.
4. Синхронизация от других пользователей: Пользователь, который не имеет доступа к наземной станции GNSS или VDL Mode 4 (в любом сочетании) может опираться на данные других синхронизированных пользователей в воздушном пространстве. Это может привести к повышению погрешностей при определении времени, но синхронизация системы времени сохраняется, а моделирование показало, что таким влиянием на работу системы можно пренебречь. Один синхронизированный пользователь способен обеспечивать синхронизацию в большой зоне охвата.
5. Плавающая сеть: Эта функциональность аналогична варианту (4) с тем лишь отличием, что все пользователи потеряли синхронизацию времени UTC, полученную с помощью GNSS или от наземной станции. В этом случае пользователи будут по-прежнему вести вещательные передачи сообщений о местонахождении, и предпринимать попытки синхронизации на основе данных других пользователей. В отсутствии синхронизации часы каждого пользователя будут отклоняться от правильных показателей. Пользователи пытаться корректировать свои часы с учетом “среднего показателя отклонения”. Это сохранит определенную степень синхронизации сети, пока не восстановятся источники времени по GNSS или по данным с земли. Все это должно рассматриваться как режим нейтрализации отказа, который сохраняет основную функцию связи в течение значительно более длительного периода времени, чем это возможно без такого координирования.

В VDL Mode 4 управление доступом к каналам основано на данных о положении, которые содержатся в сообщениях синхронизации от мобильных и наземных станций. Данные о координатах поступают от GNSS, или, в случае бортовой системы, они могут поступать от других бортовых источников данных о местонахождении.

1.3 Мобильные самоорганизующиеся Ad Hoc сети

На данный момент реальная картина такова, что все предписания, условия и принципы сетевого взаимодействия авиационной телекоммуникационной сети функционируют лишь в наземном сегменте, так называемой AFTN (aeronautical fixed telecommunication network). В сегменте AFTN увязаны наземные станции АЗН-В, метеосерверы, серверы аэронавигационной информации и серверы, хранящие информацию со спутниковых терминалов. Сетевой обмен в рассматриваемом сегменте организован на базе стека протоколов IP. На данный момент внедряется современная версия протокола IP – IPv6. Тем не менее, отсутствие сетевого взаимодействия «борт-борт» не является, в общем смысле, нереализованной частью (даже на уровне концепции). Предпосылки в сложившейся ситуации следующие:

- стек протоколов IP нацелен на выполнение сетевых задач в системах с фиксированной топологией, а также по большей части увязанных по кабельным линиям связи;
- для осуществления маршрутизации в таких сетях используется IP-адрес, генерируемый (и чаще всего строго определяемый) провайдером для каждого ключевого узла - маршрутизатора или коммутатора, которые являются отдельными устройствами, функцией которых является перенаправление информации по сети либо к оконечным узлам, либо далее к подобным устройствам;
- получающиеся подсети «возглавляемые» коммутаторами имеют свой список адресов, характерный только для конкретной подсети.

В связи с вышесказанным – организация информационной сети обмена данными, в условиях высокой интенсивности движения абонентов (ВС) и отсутствия возможности использования маршрутизаторов и коммутаторов (в классическом исполнении), является трудновыполнимой задачей, что требует рассмотрения иных концепций создания сетей и осуществления маршрутизации в них.

Также стоит отметить, что концепция построения сетей по стеку протоколов TCP/IP подразумевает наличие в пакете информации о более высоких уровнях модели OSI. Стандарт VDL Mode 4 поддерживает функционирование на первых трех уровнях этой модели, причем 3й уровень – сетевой, имеет ограниченный функционал, что в нашем случае это относится к соединению «борт-земля». Описан лишь механизм возможного входа в сеть ATN, что имеет название «подсетевой» уровень. Таким образом, организация обмена информацией по стеку протоколов TCP/IP может еще оказаться и затруднительной из-за того, что часть информации (например, о высших уровнях) будет просто отсутствовать, но занимать объем в общем пакете. В конечном итоге, концепция построения сети по стеку интернет протоколов нечётко укладывается в рассматриваемую модель OSI. Например: физический и канальный уровни, описываемые в документах VDL Mode 4 и несущие раздельные функции по модели OSI, будут являться одним уровнем в представлении модели TCP/IP.

По ряду рассмотренных проблем: ВС и иные технические средства оборудованные транспондером VDL Mode 4 не могут осуществлять передачу своей информации, а также прием её от тех участников движения, которые не находятся в зоне устойчивой радиовидимости транспондера, т.к. лишены возможности маршрутизации своей информации.

Для подобных случаев одним из перспективных подходов является применение протоколов беспроводных самоорганизующихся (или динамических, или Ad Hoc) сетей. Данный вид сетей, как динамическая структура, производит собственное формирование всякий раз, когда сетевые узлы, использующие специальные алгоритмы, оказываются в пределах прямой видимости друг от друга. Каждый из них выполняет в динамической сети помимо собственных функций (сбор, генерирование, передача и прием информации), также, что немаловажно, функции ретрансляционного пункта для всех ближайших узлов. Таким образом, узлы, расстояние между которыми больше дальности прямой видимости, могут поддерживать обмен данными между собой, если им готовы помочь другие сетевые узлы, находящиеся между ними, передавая сообщения по цепочке ретрансляторов (маршрутизаторов). Иными словами, каждый узел в сети служит элементом инфраструктуры для передачи сообщений других узлов и является источником собственных сообщений.

На сегодняшний день, самоорганизующиеся сети внедряются в малых объёмах, для экспериментальных целей. Причин этому несколько, например, в мобильной сети, при соединении, задействованы лишь 2 телефона и одна вышка. Передача информации осуществляется от одного телефона к другому, а коммуникационная функция возложена на базовую станцию. Стационарная инфраструктура даёт возможность создавать большие и

очень надежные инфокоммуникационные ресурсы. Подобная инфраструктура получила большое распространение в областях, где потребность в ней наиболее велика.

Однако в случае масштабных сбоев, например, массового отключения электропитания, фиксированная инфраструктура становится уязвимой - её работа нарушается из-за выхода из строя (или временной неработоспособности) главных узлов, даже не смотря на то, что отдельные узлы в зоне действия сети остаются исправными.

Надежность динамических сетей в случае применения их для авиационной связи намного выше. Если одно мобильное устройство отключается, соединения в сети видоизменяются таким образом, чтобы в наибольшей степени компенсировать выбывший элемент. Таким образом, сеть подстраивается или самовосстанавливается с подключением и отключением устройств. Но в этом существует ряд трудностей. Сообщения должны передаваться узлами в сети таким образом, чтобы оно могло быть передано заново даже в том случае, если в ходе ретрансляции послания какие-то звенья цепи связи между отправителем и адресатом прекратят работу. Оптимальный путь доставки сообщения адресату должен определяться узлами в сети даже при условии, что отправляющие устройства не имеют возможности определить точного местоположения узла-адресата. Кроме того, сетевые узлы должны поддерживать механизмы, позволяющие справляться с возможным возникновением повторных сообщений исходящих от других узлов.

Вышеуказанные затруднения и препятствия в авионике до сих пор активно исследуются, а стандарты по созданию динамических самоорганизующихся сетей средствами авионики находятся на начальной стадии разработки, поэтому также существует задача определения критериев для оценки эффективности и качества самоорганизующейся авиационной сети.

Обычно для целей создания сети с высокой интенсивностью топологии рассматривается такой вид самоорганизующихся сетей как мобильная Ad Hoc сеть. Зачастую mesh-сети обладают иерархичной инфраструктурой, в чем и состоит основное отличие. Например, в каждой ячейке, состоящей из нескольких узлов, используется главный узел – наподобие базовой станции. В свою очередь, Ad Hoc сеть не обладает некоей жёсткой иерархией, но требует комплексных подходов для обеспечения процесса маршрутизации, учитывающих динамическое изменение нагрузки на сетевых узлах и характер их мобильности.

Основными базовыми стандартами для построения, как Ad Hoc, так и mesh самоорганизующихся сетей являются:

- IEEE 802.15.1;

- IEEE 802.11;
- IEEE 802.15.4.

Стандарты семейства IEEE 802 достаточно распространены и нашли широкое применение в публичных сетях. Подробного рассмотрения этих стандартов не требуется, т.к. научно-исследовательская работа нацелена на использование алгоритмов маршрутизации самоорганизующихся сетей для создания сетевого протокола, базовым стандартом, поддерживающим необходимые уровни модели OSI: этим стандартом будет являться VDL Mode 4.

1.3.1 Основные алгоритмы маршрутизации мобильных самоорганизующихся Ad Hoc сетей

Мобильные Ad Hoc сети(mobile Ad Hoc networks(MANET)), являясь одним из видов самоорганизующейся сети передачи данных, обладают переменной топологией и не имеют жёсткой структуры. Они выполняют задачу информационного обмена между подвижными объектами. В общем случае, каждое устройство, в подобной сети, может двигаться в любых направлениях. Соответственно соединения между узлами будут меняться достаточно часто. Как было упомянуто, каждый узел должен ретранслировать трафик в независимости от собственного назначения, что в свою очередь ставит перед разработчиками MANET, в качестве основной, задачу об обеспечении наличия на каждом сетевом узле информации для правильной маршрутизации.

Беспроводные самоорганизующиеся сети обладают минимальным конфигурированием и возможностью быстрого развертывания, а отсутствие жесткой структуры и возможность ретрансляции сообщений по всей сети предполагает различные способы применения.

Возможность практического применения сетей MANET в различных сферах деятельности и структурах, таких как: силовые структуры, безопасность (МЧС), транспорт, медицина, спорт, сельское хозяйство и многое другое, делает их популярной темой для исследований и разработок.

Основным направлением исследований мобильных Ad hoc сетей является разработка алгоритмов маршрутизации. Такие алгоритмы должны быстро реагировать на изменение положения узлов, стремиться сократить объем служебной информации, обладать функцией своевременного поиска новых маршрутов и если требуется, то способствовать уменьшению энергопотребления узлов.

В настоящее время было разработано большое количество алгоритмов маршрутизации MANET. Классификация по принципу функционирования базового алгоритма для обеспечения процесса маршрутизации изображена на рисунке 5 [9].

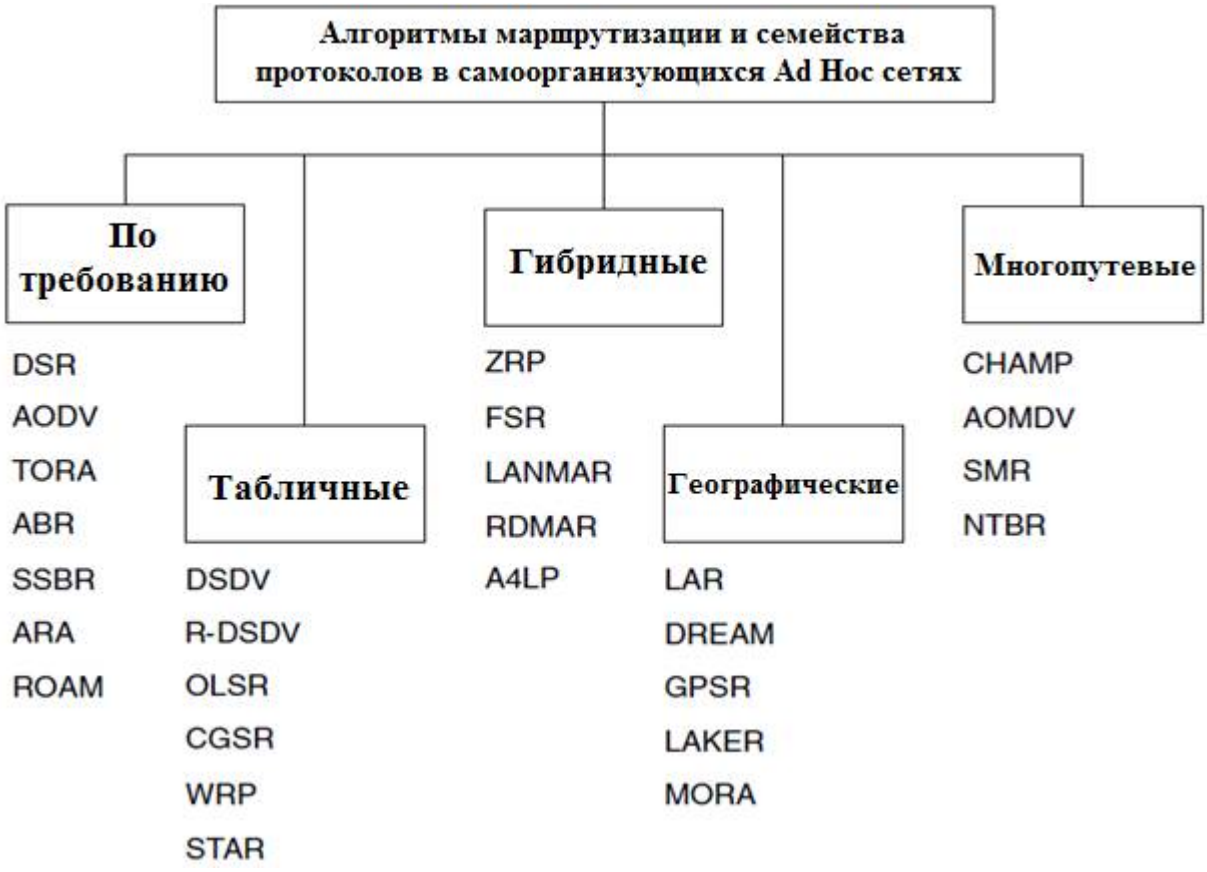


Рисунок 5. Классификация протоколов маршрутизации мобильных Ad hoc сетей по используемым алгоритмам маршрутизации

Source-initiated - семейство протоколов, где маршрут устанавливается только по требованию источника. Обычно процесс поиска маршрута осуществляется с помощью рассылки служебной информации ближайшим соседним узлам, которые в свою очередь осуществляют их ретрансляцию. Когда маршрут до адресата найден, процесс поиска заканчивается. Для удержания установившегося маршрута протоколы обычно содержат специальные служебные процедуры для продления жизни соединений.

Table-driven – это «табличные» протоколы маршрутизации. По используемому подходу, узлы передают и записывают своевременную информацию о маршрутах от одного узла к другому по всей сети. Для этого могут использоваться различные подходы, но, в общем, эти протоколы ориентированы на уменьшение служебной информации, рассылаемой по сети, и стараются поддерживать обновление информации о маршрутах.

Location-aware: данная разновидность протоколов предполагает, что каждый узел сети знает о координатах всех узлов в сети. Самой простейшей системой для получения

координат может являться глобальная система позиционирования GPS. Информация о географическом расположении узлов в дальнейшем используется для установки маршрутов.

Multipath или «многопутевая» маршрутизация заключается в том, что создается несколько путей от отправителя к получателю. Основное преимущество установления нескольких путей в том, что пропускная способность между узлами используется более эффективно и увеличивается гарантия доставки сообщения. Так же это может быть полезно при перегрузке сети. Несколько путей генерируются «по требованию», или с использованием проактивного подхода. И это имеет большое значение, т.к. установленные соединения быстро рушатся при увеличении мобильности сети.

Гибридные протоколы маршрутизации являются комбинацией проактивных, реактивных и location-aware протоколов.

Для транспортных сетей перспективным семейством является гибридные протоколы location-aware, т.к. подразумевается, что все участники движения знают своё местоположение и намерения и обмениваются этими данными с другими узлами [75, 76, 77].

1.4 Выводы

1. При рассмотрении процесса развития системы управления воздушным движением в РФ, согласно приведённым данным касательно авиационного трафика, можно отметить, что современная воздушная обстановка потребовала применения новых перспективных технологий цифровой передачи информации для различных систем авионики. Как было указано, одной из таких технологий является «радиовещательное автоматическое зависимое наблюдение», которая внедряется на территории РФ по программе Минтранса «Внедрение средств вещательного автоматического зависимого наблюдения».

2. Среди стандартов, реализующих технологию АЗН-В, в разделе рассмотрен стандарт VDL Mode 4 и его предыдущие версии, т.к. по сравнению со стандартом 1090ES он предоставляет возможность связи «борт-борт», поддерживает процедуры управления соединением, процедуры передачи пользовательской информации и использует метод доступа к среде, позволяющий каждому приёмопередатчику резервировать собственные слоты для передачи сообщений с низкой вероятностью коллизий. Также, такие функции, как переиспользование слотов, удержания синхронизации сети и использование нескольких каналов для передачи данных свидетельствует о широком наборе возможностей и целесообразности использования данного стандарта для развития авиационной связи.

3. Так как целью диссертационной работы является повышение ситуационной осведомленности пунктов УВД в отдаленных и океанических регионах путём создания мобильной самоорганизующейся сети между участниками воздушного движения для передачи данных АЗН-В, то в разделе приведён анализ технологии мобильных самоорганизующихся сетей.

Из свойств мобильных самоорганизующихся сетей выделены наиболее важные для динамических сетевых структур:

1. Самоконфигурация – узлы в сети находят и строят маршруты самостоятельно, без наличия какой-либо жёсткой иерархии;
2. Самооптимизация – построение маршрутов и их поддерживание производится согласно оптимальным требованиям (пропускная способность, коэффициент потерянных сообщений, энергоресурс и т.п.);
3. Самовосстановление – протоколы маршрутизации мобильной самоорганизующейся сети поддерживают механизмы восстановления старых и построения новых маршрутов.

4. Среди рассмотренных протоколов маршрутизации мобильных самоорганизующихся сетей, в качестве основы для дальнейшей разработки, выделено семейство гибридных протоколов, использующих данные о местоположении сетевых узлов. Принципы функционирования этих протоколов и технологии АЗН-В содержат одинаковый базовый алгоритм: сетевые узлы периодически обмениваются информацией о своём местоположении и намерениях.

РАЗДЕЛ 2. АНАЛИЗ ВОЗМОЖНОСТИ РАЗВЕРТЫВАНИЯ МОБИЛЬНОЙ САМООРГАНИЗУЮЩЕЙСЯ СЕТИ ДЛЯ ПЕРЕДАЧИ ДАННЫХ АЗН-В НАЗЕМНЫМ СИСТЕМАМ УВД. СВЯЗНОСТЬ СЕТИ

2.1 Современная ситуация

Возможность создания эффективной и надежной сети в условиях высокой мобильности и относительно ограниченной пропускной способности сетевых узлов является одной из сложнейших задач в области авионики и телекоммуникаций.

Создание сети передачи данных, в свою очередь, позволит в полном объеме получать информацию с ВС, увеличит возможности контроля воздушного пространства, а значит, повысит безопасность движения летательных аппаратов в критических, с крайне низкой ситуационной осведомленностью, областях (отдаленные малонаселенные регионы, местности с «трудным» ландшафтом, океанические регионы, приполярные регионы).

В современных аэропортах ежедневно происходит несколько десятков вылетов и совершается не меньшее количество посадок. Постоянный рост количества воздушных судов различных типов, изменение характера их движения и дальности перевозок, а также увеличение парка малой авиации и дистанционно пилотируемых (беспилотных) ВС, используемых различными службами (государственными силовыми структурами, лесоохраной и т.п.) создал повышенную нагрузку на системы контроля их местоположения и определения параметров движения, особенно на низких высотах полёта. Таким образом, традиционные средства контроля и управления воздушным движением - радиолокация и пеленгация выполняют возложенные на них функции уже не в полной мере. Причины сложившейся ситуации таковы:

1. Обширные регионы и отдельные территории страны, не охвачены федеральными службами радиолокационного и радионавигационного контроля;
2. Плохой уровень наблюдаемости объектов, движущихся по земле, и легких, малоразмерных ВС с помощью средств радиолокации, что обусловлено свойствами материалов, используемых в их конструкциях, их малыми геометрическими размерами, а также сложностью приема сигналов при наличии высоких препятствий (гор, возвышенностей) из-за множественных переотражений радиоволн от местных предметов;

3. Оснащение, в обозримом будущем, всех регионов традиционными средствами управления воздушным движением является невыполнимой задачей из-за их дороговизны.

2.2 Сеть авиационной электросвязи

Для решения сетевых задач в авионике, ICAO были разработаны рекомендации по созданию авиационной сети связи - aeronautical telecommunication network (ATN) [78]. На данный момент сеть ATN строится на базе двух моделей:

- ATN/OSI соответственно на базе модели ЭМВОС (OSI);
- ATN/IPS, основанная на стеке протоколов Интернет (IPS).

Базовая эталонная модель взаимодействия открытых систем или сетевая модель OSI (англ. open systems interconnection basic reference model) имеет 7 уровней: физический, канальный, сетевой, транспортный, сеансовый, представительский и прикладной.

Разработка требований к процессу внедрения сети ATN осуществляется согласно региональным аэронавигационным соглашениям. Конкретные районы применение стандартов связи в отношении ATN/OSI или ATN/IPS, также совершаются согласно этим соглашениям.

Общие требования к сети ATN:

- на основе заранее установленных принципов маршрутизации определяется(ются) санкционированный(е) тракт(ы);
- Передача, ретрансляция и доставка сообщений в ATN совершается в соответствии со значениями приоритетов, при этом сообщения не дискриминируются, а также не задерживаются необоснованно;
- ATN даёт возможность идентифицировать виды обмена данными, передача которых, по санкционированным для этих типов и категорий трафика видам, устанавливается пользователем сети;
- Обеспечение процессов связи в ATN происходит в соответствии с предписанными требуемыми характеристиками связи;
- Когда существует один или несколько санкционированных трактов, ATN должна обеспечивать обмен прикладной информацией;
- В тех случаях, когда санкционированные тракты отсутствуют, сеть ATN производит уведомление соответствующих прикладных процессов;

- В сети ATN должно обеспечиваться эффективное использование подсетей с ограниченной шириной полосы;
- Для различных прикладных процессов ATN обеспечивает обмен адресной информацией;
- Абсолютное время дня, в сети ATN указывается с точностью до 1 с всемирного координированного времени (UTC).

На рисунке 6 показана упрощенная схема ATN, которая не отображает всех ее функциональных возможностей (например, обеспечиваемые службой обработки сообщений УВД, функции пересылки и хранения информации). Для установления отдельных требований к характеристикам сквозной связи в рамках ATN определены различные точки сквозных соединений. Однако, в целях упрощения оценки реализации указанных функциональных требований, может потребоваться определить иные точки сквозных соединений. Точки сквозных соединений, в подобных случаях, следует четко определить и увязать с точками, указанными на рисунке.

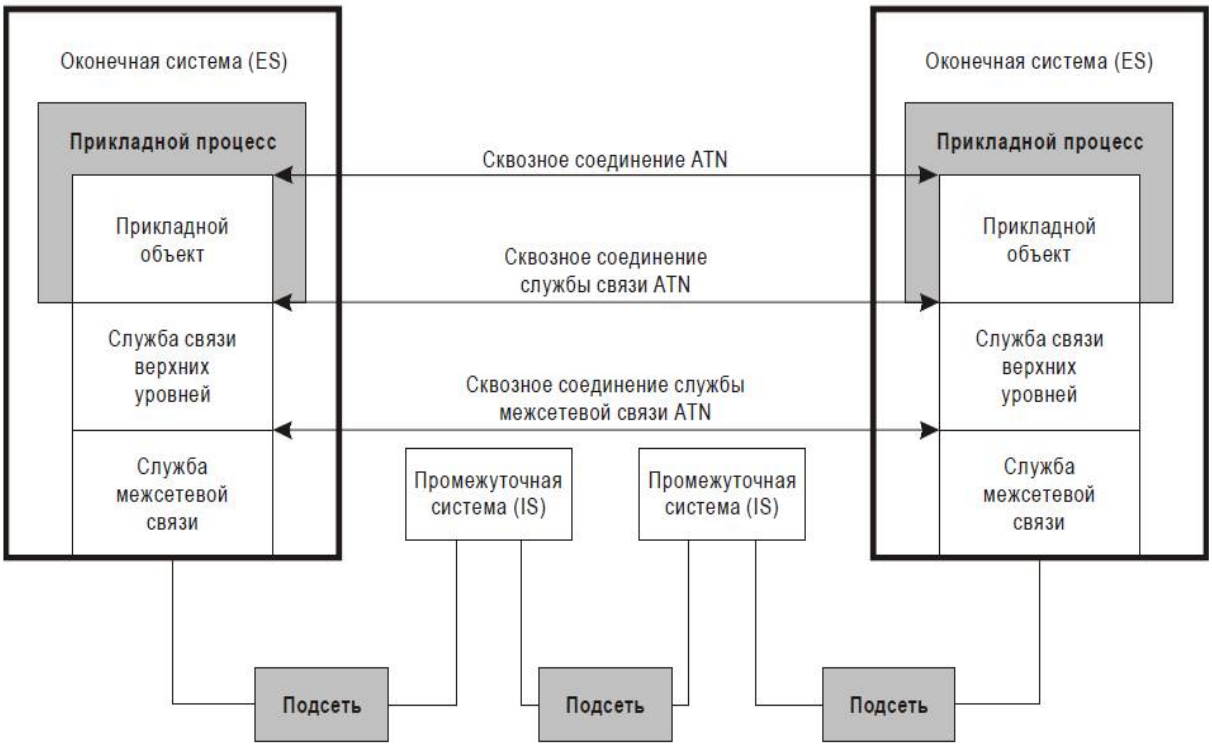


Рисунок 6. Упрощенная схема сети ATN

IS – концептуальное представление функциональных возможностей, которое не полностью соответствует роутеру. Для роутера, который реализует управление системами, требуются протоколы оконечной системы; в этом случае он также выполняет функции оконечной системы.

Выходящие за сферу применения документов SARPS элементы обозначены затенением. Интерфейсы между объектом прикладного уровня и пользователем устанавливаются требованиями пользователей. Интерфейсы определяют и обеспечивают функциональные возможности и взаимную совместимость сетей ATN. Основные уровни и основные условия взаимодействия определяются документацией по ATN.

2.3 Самоорганизующаяся сеть, как ненаправленный граф

Важной оценкой самоорганизующихся, и в общем случае, любых сетей с ретрансляторами является параметр связности. Для оценки связности радиосети существуют два подхода: расчет по графам состояний сети (общий для всех сетей) и вероятностный подход [79].

Говоря о теории графов, стоит отметить, что многие территориально распространённые объекты имеют сетевую структуру. К ним относятся и такие древние коммуникационные системы, как сети дорог, и такие современные информационные системы, как вычислительные сети. К тому же классу систем относятся железнодорожные коммуникации, сети авиационных линий, системы связи и сети передачи данных, электроэнергетические сети и др. Для них характерным является то, что все включенные в их состав объекты могут быть разделены на два типа: терминальные и транзитные пункты (места отправления материальных, энергетических или информационных потоков, а также места, где они обрабатываются и пересылаются) и коммуникационные каналы – линии связи между различными терминальными и транзитными пунктами. В зависимости от физической природы коммуникации могут иметь вполне определенную структуру, а могут быть вполне умозрительными: авиалинии, морские пути, направления радиолиний. Однако все они имеют общие черты. Поэтому для математического описания столь разных по физической природе, типу функционирования и назначения систем оказывается весьма удобным использование математических моделей, основанных на теории графов.

Представление структуры сети в виде графа является первым шагом для составления математической модели (например, графы переходов служат основой для составления соответствующих систем уравнений). Формулировка задачи в терминах графов позволяет использовать для её решения уже имеющийся математический аппарат теории графов.

Графом называется совокупность вершин (узлов) и ребер связанных между собой. Т.е. граф – это абстрактная математическая система, состоящая из двух множеств – вершин и

ребер и отображения множества вершин и множества ребер (отображение инцидентности), таким образом, вершины графа (не обязательно все) связаны между собой ребрами [80].

Поскольку в большинстве случаев в схемах расчета надежности используется двухполюсный граф (у них есть два полюса – входной и выходной), то во многих прикладных задачах, в особенности в задачах связанных с нахождением различных показателей надежности технических систем со сложной структурой, возникает необходимость определения связности двухполюсных графов или сетей.

Понятие минимального пути является одним из основных в анализе связности двухполюсных графов. Подмножество ребер двухполюсного графа, которое позволяет, переходя от одного ребра к другому через инцидентную (смежную) им обоим вершину, пройти путь от входного полюса к выходному, при этом удаление хотя бы одного ребра из этого подмножества ведёт к тому, что уже невозможно пройти путь по оставшемуся подмножеству, является минимальным путем двухполюсного графа. Таким образом, минимальным путем является путь, в котором отсутствуют петли (циклы) и «висячие», не соединяющие две вершины, ребра. Иначе подобный путь носит название «простой путь». Точнее было бы называть такой путь не минимальным, а простым, но следует придерживаться той терминологии, которая используется в теории надежности [81].

Из этого определения следует, что минимальный путь представляет собой последовательное соединение некоторых ребер графа, причем «крайние» ребра этого соединения инцидентны соответственно входному и выходному полюсам.

Следует отметить, что оптимальный алгоритм маршрутизации самоорганизующейся сети должен иметь внедренные механизмы избегания петель и нахождения оптимального (не всегда минимального) пути. Наличие петель при маршрутизации может стать серьезной проблемой, т.к. самоорганизующиеся сети обладают синергетическим эффектом. В данном случае – информационный пакет, попадая в петлю на маршруте, может быть размножен большим количеством своих копий, что приведет к большой нагрузке или даже отказу сети.

Вычисление различных показателей связности, например вероятности связности при условии случайного существования ребер графа, на практике сталкивается со значительными вычислительными трудностями, поскольку: решение указанной задачи сводится фактически к прямому перебору. В то же время, используя пути и разрезы графа, удается получать достаточно простые (по сравнению с точными методами нахождения соответствующих характеристик) граничные — верхнюю и нижнюю — оценки требуемого показателя. Количественными оценками связности графа (сети) могут служить оценки Эзари-Прошана и Литвака-Ушакова [82].

Находимые с помощью графов и расчета граничных оценок Эзари-Прошана и Литвака-Ушакова значения связности эффективны для случаев медленного изменения топологии в самоорганизующихся сетях или, еще в большей мере, для беспроводных самоорганизующихся сенсорных сетей, где узлы стационарны или меняют своё положение относительно редко. Так как в случае мобильных сетей топология изменяется постоянно и относительно часто, такие оценки могут не представлять пользы, их значения информативны только в конкретных случаях (в определенные временные моменты). Для нахождения общих показателей связности используется вероятностный подход.

2.4 Вероятностный подход для нахождения общих показателей связности

Сеть является связной тогда, когда между любой парой узлов существует маршрут, который, в общем случае, может включать одну или несколько ретрансляций. Для оценки связности использованы следующие допущения:

- связь между узлами существует, если расстояние между ними не более чем R (значение R для всех радиостанций одинаково);
- территориальное распределение радиостанций равномерное, при этом существование связи между узлами определяется по методу Ваксмана с использованием пуассоновского процесса;
- при распространении радиоволн учитываются средние потери распространения, а также медленные и быстрые замирания сигнала;
- приёмопередатчики используют антенны с круговой диаграммой направленности.

В данной методике под связностью сетевых узлов понимается ситуация, когда с определенной вероятностью P_{ISO} исключено наличие изолированных узлов в сети. Для оценки связности используется значение дальности радиосвязи (прямой видимости) R, при котором, в зависимости от заданной плотности территориального размещения сетевых узлов – λ_S, обеспечивается связность узлов с вероятностью $P_{CON} \geq P_{CON\_TPEБ}$, где $P_{CON\_TPEБ}$ означает требуемую вероятность связности узлов. Таким образом, P_{ISO} – это вероятность того, что некоторый сетевой узел окажется вне зоны прямого приёма другими узлами. Зона общей площади покрытия радиосети определяется дальностью радиосвязи R.

Рассмотренный подход для оценки связности сети учитывает территориальное распределение сетевых узлов, а также возможные условия распространения радиоволн, в частности медленные и быстрые замирания и средние потери распространения. Представленные ниже соотношения позволяют в явном виде определить такие параметры,

как необходимая дальность радиосвязи в зависимости от числа узлов в сети и требуемой площади сетевого покрытия.

Значение требуемой дальности радиосвязи R, при использовании радиостанциями антенн с круговой диаграммой направленности, когда известно число радиостанций k и площадь территории системы радиосвязи S, определяется по формуле [83]:

$$R[S,k] = \sqrt{\frac{\ln\left(-k/\ln P_{CON}\right)\cdot S}{k\cdot\pi\mu(\eta,\sigma)}}, \qquad (2.4.1)$$

где $\mu(\eta,\sigma) = \frac{2}{\eta}\Gamma\left(\frac{2}{\eta}\right)e^{\left(\frac{\sqrt{2}\sigma}{\eta}\right)^2}$ – коэффициент влияния медленных и быстрых замираний на связность радиостанций; $\eta = 2 \div 4$ – параметр средних потерь распространения, $\sigma = (\ln 10)/10\sigma_{dB}$, $\sigma_{dB} = 6 \div 10$дБ – параметр медленных замираний; $\Gamma(\cdot)$ – гамма-функция.

На рисунках 7, 8, 9 и 10 представлены результаты проведенных расчётов зависимости R(S,k) при различных параметрах.

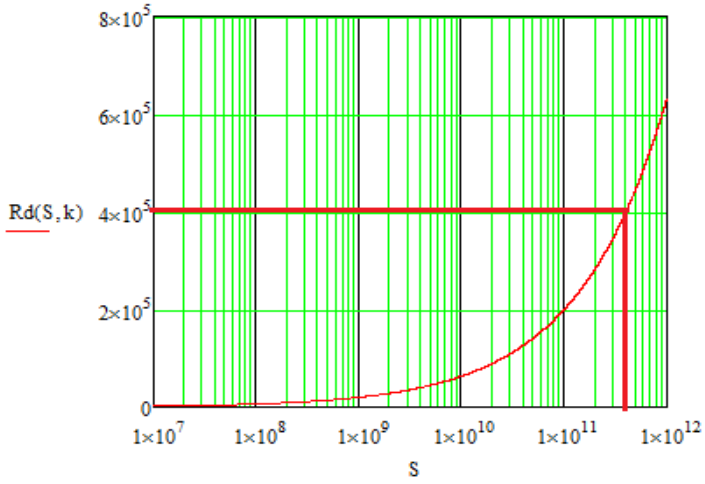


Рисунок 7. График зависимости R(S,k) при 3х соседних узлах, Pcon = 90%, η = 2

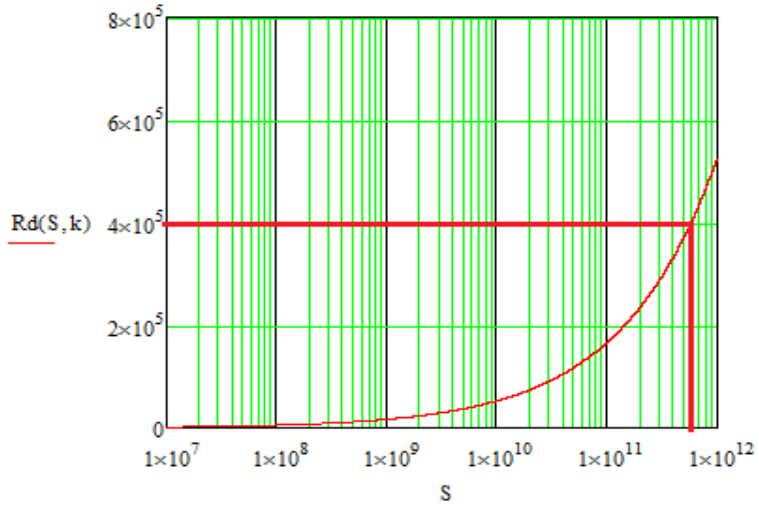


Рисунок 8. График зависимости R(S,k) при 5 соседних узлах, Pcon = 90%, η = 2

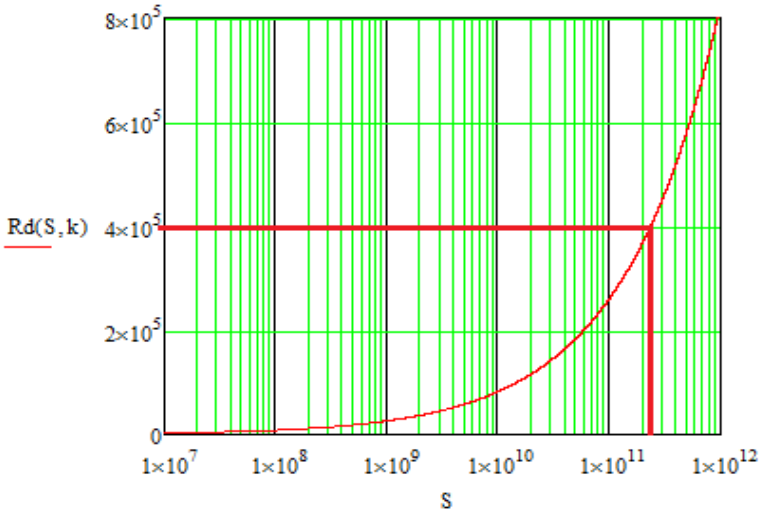


Рисунок 9. График зависимости R(S,k) при 3 соседних узлах, Pcon = 99%, η = 4

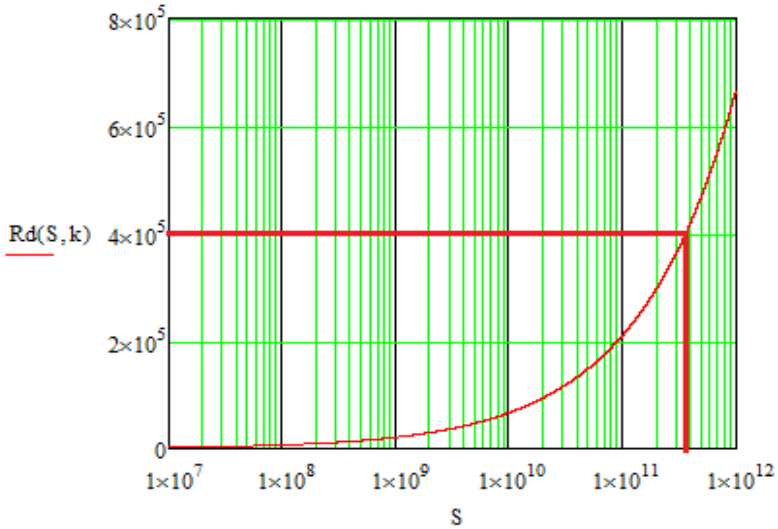


Рисунок 10. График зависимости R(S, k) при 5 соседних узлах, Pcon = 99%, η =4

По полученным графикам можно сделать следующие выводы:

- с увеличением показателя средних потерь распространения η, требуемая дальность радиосвязи R увеличивается;
- с увеличением числа узлов сети k, требуемая дальность радиосвязи R уменьшается;
- при увеличении требуемой вероятности связности Pcon, требуемая дальность радиосвязи увеличивается.

По полученным данным, при дальности радиосвязи около 400 км (что примерно соответствует дальности радиосвязи по документам VDL Mode 4), требуемой вероятности связности Pcon = 90% и параметра средних потерь η =2, для случая с тремя соседними узлами зона покрытия составляет около 500 000 км^2, а для случая с 5-ю соседними узлами – 700 000 км^2.

При требуемой вероятности связности Pcon = 99% и параметра средних потерь η = 4 для случая с тремя соседними узлами зона покрытия составляет около 300 000 км^2, а для случая с 5ю соседними узлами – 500 000 км^2. Параметр медленных замираний для обоих случаев одинаков и составляет 6 дБ.

В таблице 1, согласно рисункам 7 – 10, приведены значения площади покрытия сети в зависимости от количества узлов, полученные для следующих условий: R = 370,4 км (согласно документации VDL Mode 4); P_{CON} = 0.9; η = 2; $\sigma_{дБ}$ = 10 дБ.

Таблица 1. Площадь покрытия сети в зависимости от количества узлов

Площадь покрытия сети, млн. км^2	**0.5**	**0.7**	**1**	**3**	**5**	**10**
Количество узлов	**3**	**5**	**9**	**35**	**75**	**150**

Большинство отдаленных и океанических регионов занимают площади в несколько миллионов квадратных километров, при этом в течение дня количество ВС, пролетающих в них одновременно, может не достигать необходимых теоретических значений, что означает низкую, местами стремящуюся к нулю, связность развертываемой сети. Однако характер движения ВС в рассматриваемых регионах имеет свои особенности: ВС следуют по маршрутам с эшелонированием, скорость изменения их положения друг относительно друга невелика, расположение БС фиксировано, дальность радиосвязи в несколько раз превышает величину эшелонирования, воздушное движение имеет периоды минимума и максимума. С учётом этих особенностей с помощью программного инструментария дискретно-событийного моделирования OMNET++ [84] был смоделирован реальный полетный трафик

ВС в некоторых отдаленных и океанических регионах: на Дальнем Востоке, Ямале и в Северо-Атлантическом коридоре.

2.5 Моделирование сети с использованием реальных полетных данных

На рисунке 11 отображена картина реальных полетных данных в определенный момент времени. Данные по этой картинке получены с интернет ресурса flightradar24.com [85], где отображаются данные АЗН-В (по «сквиттеру», режим S) и АЗН-К. Отсутствие большой интенсивности в том или ином месте не говорит об отсутствии там ВС, это может быть следствием отсутствия наземной инфраструктуры УВД. Такие места являются зонами процедурных полетов, где самолеты следуют по строго установленным маршрутам с большим разнесением ВС друг от друга. В таких местах очень эффективным может являться организация сетевых структур по передаче данных АЗН-В и другой полетной информации. Для проверки возможности развертывания самоорганизующейся авиационной сети в подобных регионах нужно рассмотреть реальные сценарии полетов ВС.

При моделировании учтены такие параметры, как: расстояние между объектами (по трем координатам), мощность передатчика, чувствительность приемника, затухание при распространении, диаграмма направленности антенн, уровень шума и др. Также, кроме данных с электронного ресурса, были использованы реальные полетные данные, полученные по расписаниям и маршрутам полетов вертолетов на Ямале.

Рисунок 11. Реальная картина плотности движения гражданских ВС, использующих АЗН-В

На рисунке 12 показан пример из сценария моделирования трафика в Ямале. Зоны покрытия наземными станциями отмечены окружностями. Когда ВС и/или наземные станции находятся в зоне устойчивого приема, возможность связаться отмечается двунаправленной линией. По результатам моделирования сценариев развертывания самоорганизующейся сети выигрыш во времени обозревания ВС, т.е. период, когда данные АЗН-В могут дойти до наземной станции, составляет от 10 до 30% в зависимости от плотности движения и удачности расположения маршрутов ВС в различные моменты времени.

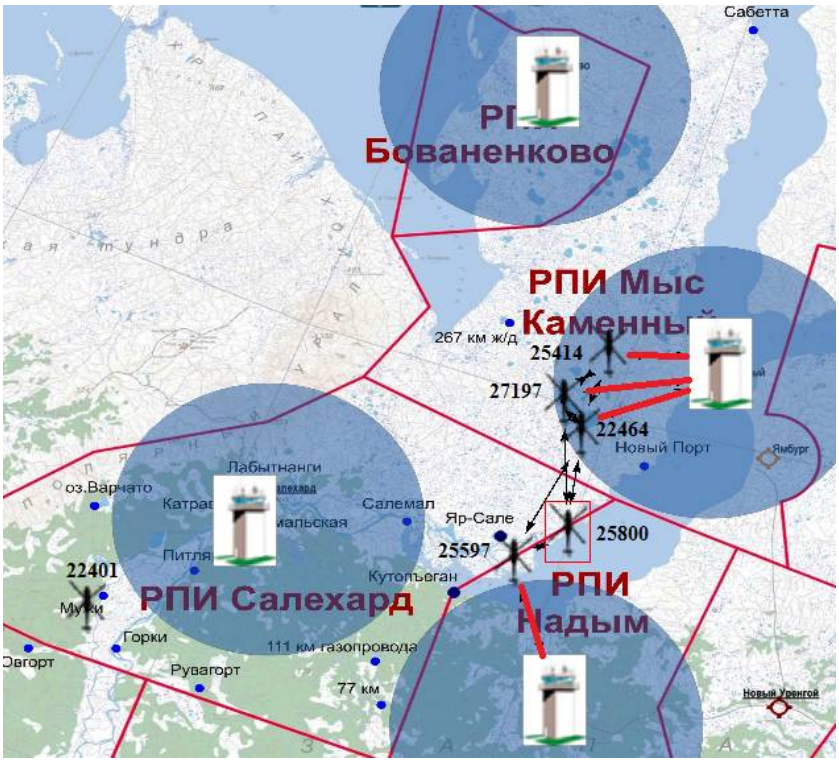


Рисунок 12. Пример моделирования возможности развёртывания самоорганизующейся сети на Ямале

Для разъяснения полученных результатов в таблице 2 и таблице 3 приведены матрицы связности для сценария без сети и с сетью соответственно.

В таблицах: красным цветом обозначено отсутствие связи, темно-зеленым наличие радиовидимости, светло-зеленым теоретическая возможность связи через ретрансляцию. Сравнивая изображение и матрицы можно сделать вывод о том, что при наличии радиовидимости хотя бы на двух «концах» сетевой структуры (т.е. хотя бы одного полного пути от верхней вершины графа до нижней вершины), то она будет полносвязной.

Таблица 2. Матрица связности при отсутствии сети

	Надым	25414	27197	22464	25800	25597	Мыс Каменный
Надым	-						
25414		-					
27197			-				
22464				-			
25800					-		
25597						-	
Мыс Каменный							-

Таблица 3. Матрица связности при наличии сети

	Надым	25414	27197	22464	25800	25597	Мыс Каменный
Надым	-						
25414		-					
27197			-				
22464				-			
25800					-		
25597						-	
Мыс Каменный							-

Для более наглядного исследования преимуществ внедрения самоорганизующихся сетей для целей передачи данных АЗН-В был рассмотрен сценарий с исключением наземной станции на Мысе Каменном. По полученным данным была проведена оценка увеличения времени обзора наземной станцией вертолетов, в сравнении с наличием сети и её отсутствием, что отражено на рисунках 13, 14 и 15. По оси абсцисс отложено время по UTC в минутах, по оси ординат: «1» – наличие связи с наземной станцией, «0» – отсутствие. Красной линией отмечены данные при отсутствии сети, штрихпунктирной – при наличии.

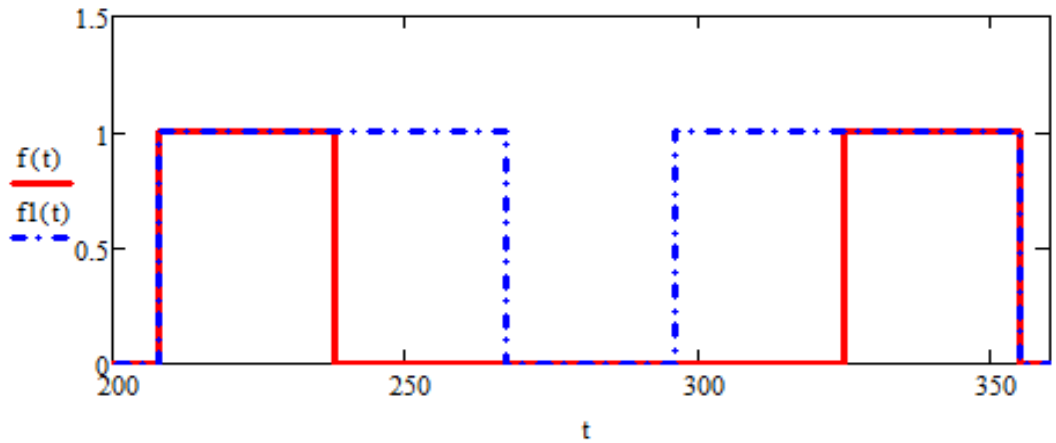


Рисунок 13. Оценка времени обзора для маршрута 27197 Надым-Бованенково (выигрыш около 50%)

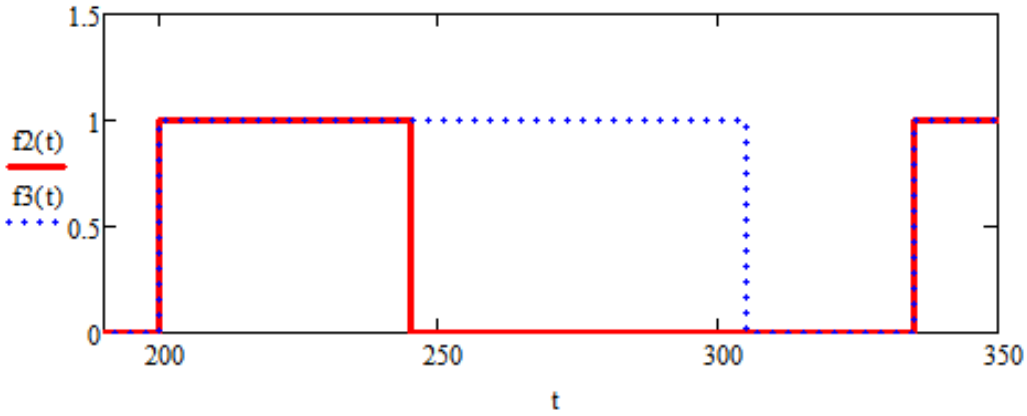


Рисунок 14. Оценка времени обзора для маршрута 25414 Салехард – Сабетта (выигрыш около 70%)

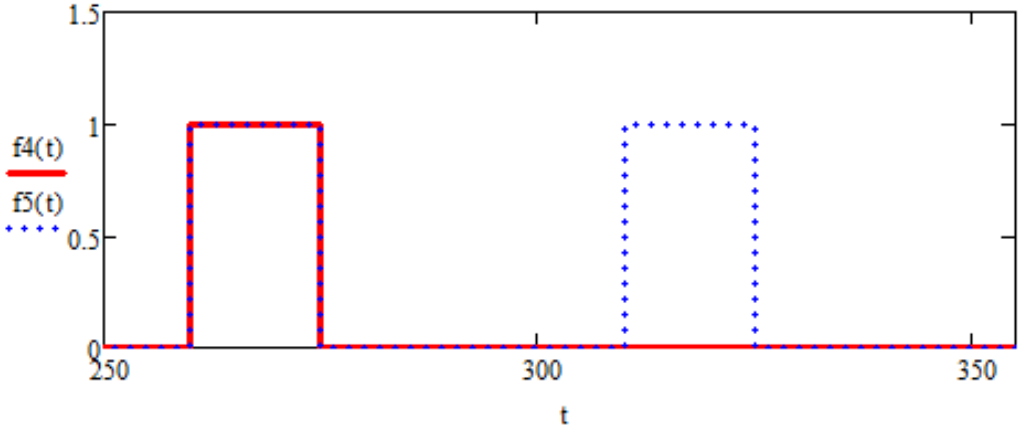


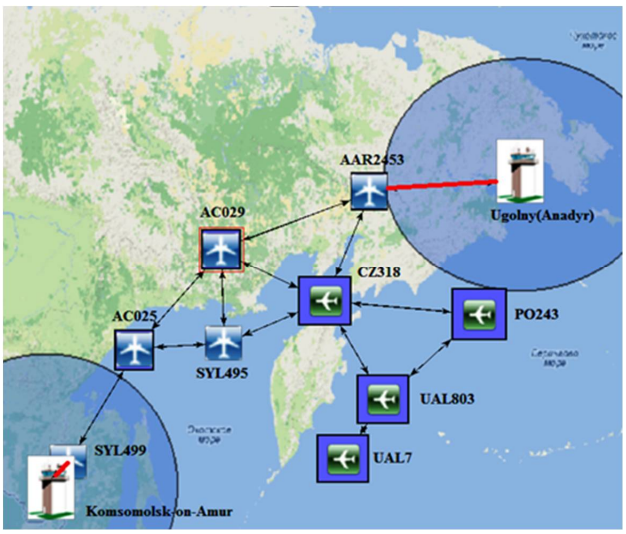
Рисунок 15. Оценка времени обзора для маршрута 25597 Яр-Сале – Мыс Каменный (выигрыш около 30%)

В таблице 4 приведены результаты моделирования 160 мин. полетных данных вертолётов в районе Ямала.

Таблица 4. Периоды получения сообщений от ВС для Ямала

Маршрут	Без сети, мин.	С сетью, мин.
Салехард – Овгорт (22401)	30	30
Салехард – Сабетта (25414)	50	140
Надым – Яр Сале (27197)	40	160
Яр Сале – Мыс Каменный (25597)	90	120
Надым – Бованенково (22464)	110	160
Надым – Бованенково (25800)	130	160

Сходные по принципу моделирования сценарии были рассмотрены для Дальнего Востока и Северо-Атлантического коридора (рисунок 16 и 17).



	Ugolny (Anadyr)	AAR2453 Seattle - Seoul	AC029 Vancouver – Beijing	CZ318 LA – Guangzh	PO243 Cincinnati - Austin	AC025 Vancouver – Shanghai	SYL495 Yakutsk – Ugolny	UAL803 Washing. - Tokyo	UAL7 Houston - Tokyo.	SYL 499 Yakutsk – Ugolny	Komsomol sk-on-Amur
Ugolny (Anadyr)	-										
AAR2453 Seattle - Seoul		-									
AC029 Vancouver – Beijing			-								
CZ318 LA – Guangzh				-							
PO243 Cincinnati - Austin					-						
AC025 Vancouver – Shanghai						-					
SYL495 Yakutsk – Ugolny							-				
UAL803 Washing. - Tokyo								-			
UAL7 Houston - Tokyo									-		
SYL 499 Yakutsk – Ugolny										-	
Komsomol sk-on-Amur											-

Рисунок 16. Пример моделирования возможности развертывания самоорганизующейся сети на Дальнем Востоке

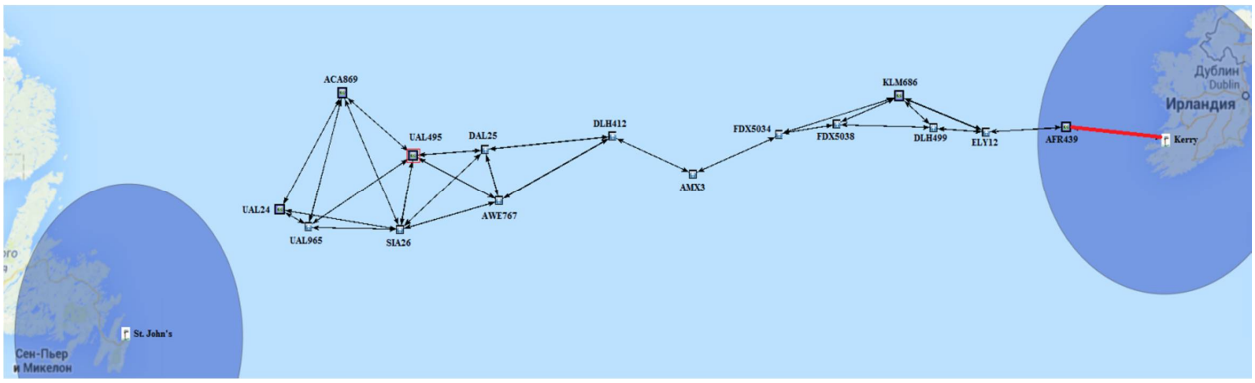


Рисунок 17. Пример моделирования возможности развертывания самоорганизующейся сети в Северо-Атлантическом коридоре

Выигрыш во времени обозревания ВС – это доля периода в процентах, когда ВС, находящееся за пределами прямой видимости пунктов УВД, может передать этим пунктам данные АЗН-В через других участников воздушного по самоорганизующейся сети.

По результатам моделирования, выигрыш во времени обозревания ВС в двух сценариях достигал 100%. Таким образом, в зависимости от плотности движения и расположения маршрутов ВС в различные моменты времени, величина периода получения сообщений через сеть варьировалась от нескольких минут до 2-х час. В часы минимальной загруженности Северной Атлантике число узлов достигает 50.

Также были рассмотрены сценарии полёта ВС малазийских авиалиний маршрута MH370 (с возможными отклонениями), пропавшего в небе над Южно-Китайским морем 8 марта 2014 г. С учётом возможных изменений в пути следования маршрута, было установлено, что через сеть маршрут можно было бы наблюдать в среднем 20 мин.

Полученные результаты позволили сделать выводы о возможности применения протоколов мобильных самоорганизующихся сетей для повышения ситуационной осведомленности пунктов УВД в отдаленных и океанических регионах.

2.6 Основные элементы и особенности функционирования воздушно-космической сети

Спутник связи является искусственным спутником Земли, который используется для обеспечения ретрансляции или информационного обмена с помощью радиосигнала между точками на поверхности земли, находящимися вне зоны прямого приёма друг от друга. Зона спутникового покрытия определяется его техническими характеристиками, положением на орбите, и ориентацией приёмопередающего оборудования.

Концепция передачи данных АЗН по спутниковому сегменту подразумевает собой использование спутниковых линий передач данных для доставки информации о местоположении и намерениях ВС наземным системам УВД, в ведомстве которого данное ВС находится [86]. Представленная концепция может быть реализована с помощью технологии автоматического зависимого наблюдения контрактного (АЗН-К) [87]. Информация о местоположении, формируемая на борту ВС, передается по каналу передачи данных вверх. Затем на спутнике происходит обработка этой информации, коммутация и передача по каналу вниз на наземный спутниковый терминал.

Применение спутниковых группировок на геостационарных орбитах оптимально для систем телерадиовещания, где задержки сигнала величиной в 250 мс, для каждого направления передачи, не оказывают значительного влияния на качественные характеристики сигналов [88]. Однако, голосовая связь более чувствительна к задержкам, а поскольку задержка в геостационарных спутниковых системах суммарно составляет около 600 мс, с учетом времени обработки и коммутации на земле по технологии IP, такие системы менее эффективны при передаче голоса и чувствительного к задержкам трафика. Т.к. передаваемая информация АЗН – это данные, которые требовательны ко времени передачи и их качество зависит от времени доставки до пунктов УВД, то спутниковые системы связи на геостационарных спутниках представляют также низкую эффективность для обеспечения информацией систем УВД. К тому же, в обслуживаемые регионы геостационарных космических аппаратов не входят районы с большой географической широтой, т. е. реальное глобальное покрытие не обеспечивается.

Спутники, запускаемые на земную орбиту, на высоту до 1000 км, с периодом вращения вокруг земли порядка 1,5-2 часа называются низкоорбитальными спутниками.

Группировки низкоорбитальных спутников, в которых с абонентским оборудованием взаимодействует цепочка следующих один за другим спутников, используются в качестве систем передачи данных. Для обеспечения хорошего покрытия земли низкоорбитальных спутников должно быть не менее 50. Так как низкоорбитальные спутники располагаются значительно ниже геостационарных, то и задержки при распространении и обработке сигнала на сегментах вверх и вниз значительно ниже, а также меньше и мощность передатчика, и чувствительность приемника [89, 90]. Одно из основных преимуществ систем подвижной связи – они могут быть использованы для организации информационного обмена между разными типами терминального оборудования, которое размещено на воздушных, морских и наземных транспортных средствах. Связь с космическими объектами на орбите Земли производится преимущественно в диапазонах L, S, C, Ku и Ka. Изначально низкоорбитальные группировки спутников создавались для узкой области применения, связанной, с передачей небольших и относительно редких сообщений. Однако они могут эффективно использоваться для передачи пользовательских данных и телефонной связи. На сегодняшний день, заявлено около 40 различных проектов по созданию и разработке низкоорбитальных спутниковых группировок, как отечественными, так и зарубежными фирмами, которые оцениваются как вполне реализуемые [91].

На рисунке 18 отображена схема организации обмена данными АЗН-В с использованием низкоорбитальных спутников [92]. Общая схема разделена на 3 части:

- воздушно-транспортная сеть;
- космическая сеть;
- наземная сеть.

Воздушно-транспортная сеть подразумевает собой сегмент, связанный с определенной наземной станцией (НС) или группой наземных станций АЗН-В. Участниками воздушно-транспортной сети могут являться различные ВС (гражданские самолёты, вертолеты, беспилотные летательные аппараты, наземные транспортные средства), имеющие на борту средства приема и передачи данных АЗН-В (например, транспондер VDL Mode 4). По сети также может передаваться информация от дополнительных технологий АЗН-В:

1. Traffic Information Services-Broadcast (TIS-B) - сервис обозревания авиационного трафика для ВС, позволяющий ВС, не оборудованными транспондерами ADS-B, передавать свое местоположение (получаемое с помощью радаров) воздушным судам, оборудованным ADS-B;
2. Flight Information Services-Broadcast (FIS-B) – приложение технологии АЗН-В, обеспечивающее передачу в широковещательном режиме бесплатной информации о

погоде, временные ограничения полётов (temporary flight restrictions (TFRs)) и передачу некоторой специальной информации о текущей воздушной обстановке;

3. Automatic Dependent Surveillance Re-broadcast (ADS-R) – ретрансляция информации с помощью наземной станции АЗН-В с одного соединения на другое, ВС, оборудованному приемником АЗН-В;
4. CPDLC - позволяет устанавливать диалоговые режимы CPDLC для обеспечения обмена сообщениями между диспетчером и пилотом и пересылки сообщений на земле, а также управлять этими режимами и прекращать их;
5. Differential GNSS(DGNSS) – средства передачи дифференциальных поправок локальных навигационных систем;
6. A-SMGB – система управления наземным движением.

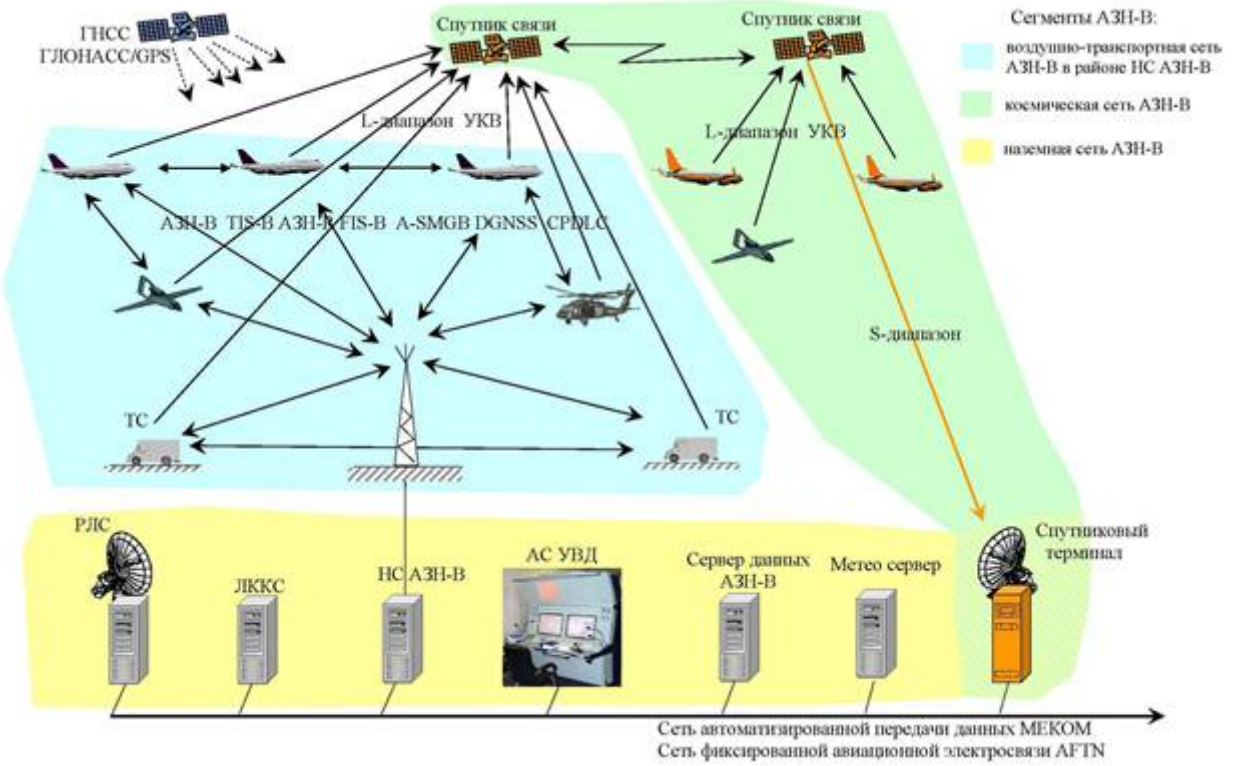


Рисунок 18. Схема организации обмена данными АЗН-В (при использовании спутниковой связи)

Все участники сети, оборудованные приемниками GNSS (ГЛОНАСС/GPS), будут получать информацию о своем местоположении, а также время UTC для синхронизации передачи и приема данных.

Говоря о воздушно-транспортной сети, мы подразумеваем, что информация АЗН-В от любого из её участников может быть принята, обработана и передана спутником на спутниковый терминал и далее системам УВД.

Теоретически предполагается, что может существовать спутниковая связь «борт-борт», что обозначено двунаправленной стрелкой между спутниками. Наличие такой связи может положительно сказаться на общей работе сети.

Космическая сеть состоит из следующих элементов:

- ВС, использующие приемо-передатчики ЛПД VDL-4 и ЛПД 1090ES;
- спутники связи;
- наземные спутниковые терминалы.

Для связи по каналу вверх, используется L – диапазон и УКВ-диапазон частот. Для связи по каналу вниз, используется диапазон S. Спутник связи должен агрегировать все потоки данных АЗН-В, поступающие на него, при этом «отсеивая», те данные, которые не нуждаются в том, чтобы быть переданными по спутниковому сегменту. Для эффективной работы сети спутник связи должен быть оборудованным приемником VDL Mode 4.

Спутниковый терминал должен быть увязан с наземной авиационной сетью AFTN для обеспечения доставки полной информации о всех участниках сети, чья информация была передана по спутниковому сегменту.

Стоит отметить, что максимальная нагрузка будет приходиться именно на космический сегмент сети, т.к. выше было указано, что спутник должен будет агрегировать потоки со всех систем АЗН-В. Поэтому спутниковый сегмент должен обеспечивать относительно большую пропускную способность для удовлетворения современного объема информации авиационного трафика и иметь запас для возможного увеличения потоков данных в сети в будущем. Альтернативой является уменьшение потока за счет учета признака ВС «вне сети» - только с таких ВС информация АЗН-В должна транслироваться со спутника на землю.

В состав наземной сети АЗН-В входят следующие элементы:

- РЛС – радиолокационная станция, целью которой является обнаружение воздушных объектов в зоне действия местного органа УВД, используется для обеспечения данных TIS-B;
- локальная контрольно-корректирующая станция (ЛККС) – система, генерирующая корректирующие поправки для ВС;
- НС АЗН-В – наземная станция, на которой установлены различные системы, поддерживающие ЛПД АЗН-В (в данном случае VDL Mode 4);
- АС УВД – автоматизированная система УВД, обычно авиадиспетчерская;
- сервер данных АЗН-В - хранилище данных АЗН-В;
- метеосервер – хранилище метеоданных;

- спутниковый терминал – приемное и коммутационное устройство спутникового сегмента.

Все элементы наземной сети должны быть увязаны по общему сетевому интерфейсу. На данный момент, фиксированная сеть ATN строится на базе сетевого протокола IPv6. Данные на каждом элементе наземной сети должны быть доступны для всех систем УВД. Данный сегмент также подвержен большим нагрузкам потоками данных, однако, фиксированная сеть строится на основе кабельных систем, чья пропускная способность достаточно высока, а современный протокол IPv6 и протоколы, строящиеся на его основе, обеспечат надежный обмен данными.

2.6.1 Функционирование самоорганизующейся авиационной Ad Hoc сети при наличии спутникового сегмента. Признак отсутствия сети

Как указывалось выше, концепция передачи данных АЗН-В по спутниковому сегменту имеет несколько недостатков:

1. Высокая задержка передачи данных из-за распространения сигнала вверх и вниз, его обработки, коммутации и передачи по фиксированной сети;
2. Повышенная нагрузка на спутниковый сегмент при больших потоках данных;
3. Из-за агрегирования потоков от систем, основанных на разных стандартах АЗН-В, могут возникнуть трудности с корректным разделением информации АЗН-К от АЗН-В и др.

Так как подразумевается, что передача данных АЗН-В по спутниковому сегменту будет осуществлять только тогда, когда самолет находится не в сети (либо в группе изолированных от сети ВС) и его обзор наземной станцией также не осуществляется, то ко всему этому добавляется тот факт, что такая передача данных АЗН-В будет являться односторонней. Т.е. будучи изолированным от сети, ВС сможет лишь пересылать информацию о своем местоположении и намерениях, но не сможет получить какой-либо информации от наземной станции. Однако такой вид передачи данных АЗН-В может нести собой большую пользу, например, при поиске и спасании. Кроме того, информация о полетных данных, полученная системами УВД может стать эффективной при планировании и прокладке новых маршрутов движения ВС.

В связи с тем, что ВС будут передавать данные АЗН-В по спутниковому сегменту, будучи изолированными от сети и наземной станции, то требуется выделить чёткий признак отсутствия сети. Как указано во второй разделе, одним из возможных алгоритмов

маршрутизации может являться географический алгоритм. Рассмотренный подход подразумевает проактивный метод передачи информации о наземной станции всем участникам сети. Каждый ВС должен хранить эту информацию в течение некоторого периода времени, затем дополнять или обновлять её. Исходя из этого, алгоритм генерации признака отсутствия сети может быть следующим: ВС получает информацию о наземной станции и записывает её в таблицу наземных станций, при этом запускается счетчик, который будет отсчитывать момент от времени прибытия последнего сообщения с данными о наземной станции. Если в течение этого времени новый пакет с данными от какой-либо наземной станции не приходит, то ВС ждёт еще небольшой промежуток времени. Этот промежуток, который можно назвать защитным, используется для того, чтобы учесть случаи кратковременного отсутствия связи. Если проходит промежуток ожидания информации о наземной станции, а также защитный промежуток, то ВС, генерируя данные АЗН-В, добавляет в пакет информацию о том, чтобы быть обработанным спутниковым сегментом. Данная концепция рассматривается с тем условием, что спутник будет оснащён приемником VDL Mode 4.

Проведенное моделирование сценариев полетов, с учетом реальных полетных данных, теоретически доказывает возможность применения самоорганизующихся Ad Hoc сетей для их развертывания в отдаленных и океанических регионах. Также по этим данным можно судить о возможности организации более гибких маршрутов полетов ВС для уменьшения количества наземных станций, что даст положительный экономический эффект.

2.7 Выводы

1. Построение воздушной самоорганизующейся сети на основе стека протоколов TCP/IP потребовало бы доработки стандарта VDL Mode 4, т.к. стандартом определяются только физический и канальный уровни модели OSI. В свою очередь, использование протоколов TCP/IP для мобильных самоорганизующихся сетей является избыточным, т.к. MANET предназначены для выполнения специальных задач и обычно не осуществляют межсетевого взаимодействия (либо делают это через интернет-шлюзы). Таким образом, использования уникальных адресов второго уровня достаточно для организации сети, а комплексная структура сообщения стандарта VDL Mode 4 позволяет дифференцировать пользовательскую и служебную информацию.

2. По проведенным в разделе аналитическим исследованиям связности сети и моделировании реальных полётных данных установлено, что именно связность является

определяющим фактором эффективности работы сети – это следует из расчётов необходимого кол-ва узлов для покрытия определенной территории и большого разброса в потенциальных периодах получения сообщений АЗН-В по сети. Однако выигрыш во времени наблюдения ВС по сообщениям АЗН-В через сеть в нескольких случаях составила 100%, т.е. полный интервал времени, когда между ВС и пунктами УВД не было прямой видимости, имелась возможность передачи сообщений через другие ВС, что в свою очередь говорит о целесообразности построения мобильной самоорганизующейся сети между участниками воздушного движения в отдаленных и океанических регионах.

3. В связи с возможностью использования низкоорбитальных спутниковых аппаратов для наблюдения участников воздушного движения, в разделе также рассмотрено использование мобильной самоорганизующейся сети в составе воздушно-космической сети. По результатам проведенного исследования выявлено, что ВС, которые не имеют возможности передавать данные по сети (например, в отсутствии соседних узлов и маршрута до БС), могут вещать сообщения АЗН-В с меткой об отсутствии сети. Подобные сообщения уже могут быть ретранслированы по спутниковому сегменту или переданы на землю.

4. Наблюдение ВС с помощью спутниковых аппаратов затруднено из-за большой зоны приёма спутниками, которая будет превышать стандартный размер УКВ-соты стандарта VDL Mode 4 (порядка 400км). Таким образом, в зону приёма спутника попадёт несколько «неорганизованных» по алгоритму STDMA УКВ-сот, что вызовет резкое увеличение количества коллизий при приёме сообщений.

РАЗДЕЛ 3. РАЗРАБОТКА КОМПЬЮТЕРНОЙ МОДЕЛИ VDL MODE 4

3.1 Общее описание

Модель и моделирование - это универсальные понятия, являющие собой основу одного из наиболее современных и мощных методов познания явлений, систем или процессов в любой профессиональной области.

Построение модели, в качестве системной задачи, требует комплексного подхода: анализа и синтеза исходных данных, проверки гипотез и теорий. Это требует глубоких знаний специалистов в предметной области. Однако столь сложный подход позволяет строить точные модели реальных систем и использовать модели для оценки показателей эффективности работы системы в различных ситуациях.

Таким образом, модель используется для изучения объекта, который он описывает, или воспроизведения каких-либо свойств и процессов, происходящих в нём, т.е. модель сама может являться объектом. Однако, чаще всего модель – это описание объекта или целой системы, при определенных условиях и предположениях замещающая оригинал.

Из вышесказанного следует, что моделирование – это особая форма эксперимента, при котором происходит процесс изучения не оригинала, что называется простым или обычным экспериментом, а над объектом, замещающим оригинал. Для обеспечения правильности процесса и результатов моделирования важен изоморфизм оригинала и модели, т.е. определенная схожесть копии и применимость знаний, с помощью которых она была предложена.

На данный момент, технология радиовещательного автоматического зависимого наблюдения в РФ находится на стадии внедрения и возможности стандарта VDL Mode 4 не изучены в полной мере, поэтому для обеспечения эффективности процесса внедрения, требуется создание компьютерной модели стандарта VDL Mode 4, которая наиболее полно отражает его структурные особенности и функционал. Разработанная компьютерная модель будет являться основой для новых целевых применений стандарта VDL Mode 4, и способствовать ускорению внедрения новых систем авионики.

Одним из будущих приложений для стандарта VDL Mode 4 будет являться передача данных АЗН-В до пунктов УВД по сети. Это позволит в некоторой степени решить проблему ситуационной осведомленности авиадиспетчеров и экипажей ВС в отдаленных и океанических регионах. Многоинтервальная передача данных АЗН-В от ВС до пунктов УВД будет обеспечиваться с помощью разработанного протокола маршрутизации мобильных

самоорганизующихся сетей. Перспективность подобного подхода была утверждена на 12ой аэронавигационной конференции международного комитета гражданской авиации - ICAO[93].

Кроме проблемы с ситуационной осведомленностью, также существует опасность наведения имитационных помех злоумышленниками на транспондеры участников движения. Идентификация злоумышленника может быть осуществлена с помощью мультилатерации. Данное приложение может быть дополнено с помощью самоорганизующейся сети, если не удается найти несколько приемников сигнала от злоумышленника в зоне радиовидимости.

Применение протоколов самоорганизующихся сетей также планируется для целей коммутации аналоговых каналов. Так как аналоговая голосовая радиосвязь до сих пор является одним из методов УВД, рассматривается возможность выстраивать сеть аналоговых ретрансляторов для обеспечения голосового сообщения между пилотами и диспетчерами за переделами зоны прямой радиовидимости.

Компьютерное моделирование это современный инструмент разработки и тестирования, как отдельных протоколов, так и целых телекоммуникационных систем. В данном случае компьютерное моделирование позволяет с некоторой степенью достоверности решать следующие задачи:

1. Проверить работоспособность системы при участии большого количества сетевых узлов, что актуально для транспортных систем, в особенности в авиации.
2. Протестировать работоспособность системы при изменении уровня или типа влияния различных факторов, таких как: помехи, мобильность узлов, взаимодействие между стеками протоколов и т.п.
3. Проанализировать эффективность новых расширений для системы.

Создание таких тестовых испытаний в реальных условиях может занять большое количество времени и материальных средств. Применение же правильных аналитических и математических моделей в рамках компьютерного моделирования может дать удовлетворительную точность без больших материальных затрат.

На сегодняшний день существует множество объектно-ориентированных программных инструментариев дискретно-событийного моделирования, самые популярные из них: NS-2, OPNET и OMNeT++.

В рамках данной работы используется пакет сетевого моделирования OMNeT++, т.к. он обладает рядом следующих преимуществ [94]: открытый код (любой желающий может писать код под собственные модели), удобный интерфейс с большим количеством специализированных субинструментариев и библиотек, модульная структура,

высокоэффективное программное ядро, направленное на оптимизацию дискретно-событийного моделировании, поддержка в интернет-сообществе и кроссплатформенность (может быть установлен на машины с различными операционными системами).

Структурно-функциональная модель VDL Mode 4, описанная в стандарте ICAO 9816, разделена на 3 уровня модели OSI: физический, канальный и сетевой.

Канальный уровень имеет гибкую структуру и разделен на следующие подуровни:

- Подуровень *доступа к среде* (Media Access Control, MAC), который использует множественный доступ к среде распространения сигнала с временным разделением между станциями (TDMA);
- Подуровень *специальных сервисов линии передачи данных VDL Mode 4* (VDL Mode 4 Specific Services, VSS), который обеспечивает связь путём использования гибкого формата пакета данных и связываемых с ним протоколов передачи и резервирования над подуровнем MAC;
- Подуровень *установления соединения* (Data Link Service, DLS), который обеспечивает сервисы, ориентированные на установление соединения типа точка-точка и широковещательные сервисы над подуровнем VSS;
- Подуровень *управления связью* (Link Management Entity, LME), который устанавливает и поддерживает связь внутри УКВ-соты.

Для создания компьютерной модели VDL Mode 4 в пакете моделирования OMNeT++ был выбран субинструментарий MiXiM, т.к. он имеет множество модулей и библиотек для разработки моделей физического, канального и сетевого уровней.

В рамках компьютерной модели средствами языка C++ и внутреннего языка описания сети в OMNeT++ (NED – network description) создан комплексный модуль, состоящий из следующих элементов: модуль мобильности, модуль физического уровня и модуль канального уровня (с внутренним разделением). Кроме этого создан так называемый модуль контроля сетевого интерфейса, т.к. структура MiXiM подразумевает, что именно данные модули взаимодействуют с модулем управления соединения, который создает, отслеживает и удаляет соединения во время моделирования. Разработанные модули представлены на рисунке 19.

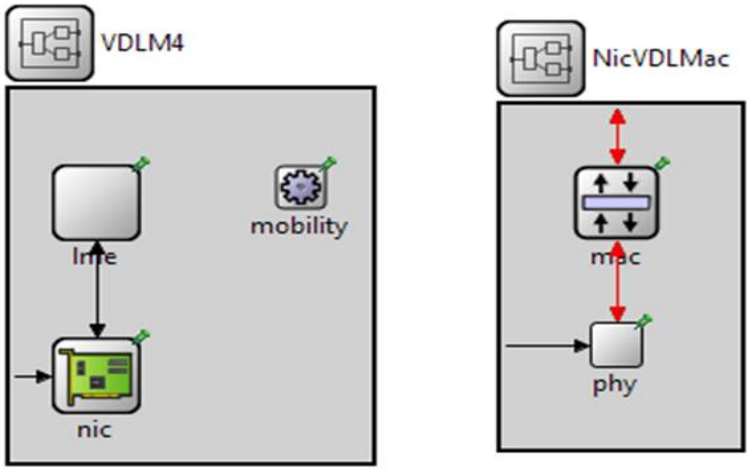


Рисунок 19. Вид модуля, представляющего сетевой узел (слева) и модуля контроля сетевого интерфейса

Модуль мобильности позволяет использовать различные модели движения сетевых узлов во время моделирования: случайное, стационарное, по координатам, по кругу, по прямой и т.д. Требуется лишь указать тот тип, который нужен и соответствующие этому типу параметры.

Модуль физического уровня разделяется на две составляющих – модель распространения радиоволн и решающее устройство. При моделировании могут быть использованы различные модели распространения сигнала, которые учитывают изменения сигнала, как во временном, так и в частотном домене. Решающее устройство делает вывод о том, верно ли принят сигнал, таким образом, здесь учитывается уровень принимаемого сигнала, ОСШ, наличие битовых ошибок в принятом пакете и др. В разработанной модели решающего устройства также учитывается возможность перекрытия двух сигналов при конфликте слотов. Из технической документации следует, что отношение между полученными сигналами должно составлять более 12 дБ, в ином случае оба (или более) пакетов отбрасываются. Входными данными для модуля физического уровня являются: мощность передатчика, чувствительность приемника, уровень шума и другие параметры, зависящие непосредственно от используемой модели распространения волн и решающего устройства. Кроме указанных функций, решающее устройство осуществляет обработку запросов о состоянии канала, исходящих от уровня доступа к среде и отправляет на верхний уровень сообщение со сведениями об отброшенном пакете, если он был принят с ошибками. Модуль физического уровня соответственно включает в себя физический уровень структурно-функциональной модели VDL Mode 4.

Программный модуль доступа к среде (MAC) содержит в себе уровни MAC и VSS структурно-функциональной модели. В данном модуле реализован метод STDMA, т.е. здесь

содержится и обрабатывается карта временных слотов, которая складывается индивидуально для каждого узла. Модуль общается как с верхним, так и с нижним уровнями при помощи служебных сообщений. Для нижнего уровня предусмотрены следующие служебные сообщения: прослушивание канала, переключение на прием и переключение на передачу (по окончании переключения физический уровень отправляет соответствующее сообщение). Также, модуль формирует значение длительности передачи, для физического уровня, исходя из скорости передачи и результирующей длины пакета в битах. Для верхнего модуля (LME), на данном этапе разработки, предусмотрены следующие служебные сообщения: уведомление об окончании этапа настройки и запрос синхронизирующего пакета данных (synchronization burst). В терминологии документа 9816 этот пакет не содержит синхронизирующей информации в обычном смысле, здесь – это пакет подуровня LME, содержащий информацию, позволяющую узлу согласовывать карту временных слотов с другими узлами в области собственной УКВ-соты. Уведомление об окончании этапа настройки нужно для информирования верхних подуровней о том, что закончен требуемый период формирования карты временных слотов, либо потребовался срочный вход в сеть. Когда с верхних подуровней приходит пакет, он инкапсулируется в пакет VSS и полученный пакет, вместе с данными о длительности передачи отправляется на физический уровень.

Модуль LME отвечает за формирование синхронизирующих пакетов данных, обработку таблицы соседних узлов и может осуществлять управление подуровнем DLS, если того требует приложение, использующее VDL Mode 4. Также как и остальные модули, модуль LME общается с модулем MAC при помощи служебных сообщений. По приходу сообщения об окончании этапа настройки на данном модуле запускается приложение A3H-B, которое с помощью служебного сообщения, отправляемого на модуль MAC, размещает резервирование. Количество резервируемых слотов зависит от требуемой частоты вещания синхронизирующих пакетов в минуту (по умолчанию 12 раз в минуту). По приходу запроса на синхронизирующий пакет, модуль собирает требуемые данные, устанавливает битовую длину пакета и отправляет его на подуровень MAC. Все данные о местоположении генерируются при помощи прямого доступа к модулю мобильности. В свою очередь, обработка таблицы соседних узлов осуществляется с помощью прямого доступа к модулю MAC.

В среде моделирования с дискретными событиями каждое событие происходит в конкретный момент времени. Для имитации временного разделения канала моделирование происходит от слота к слоту. Граф состояний разработанного модуля MAC представлен на рисунке 20.

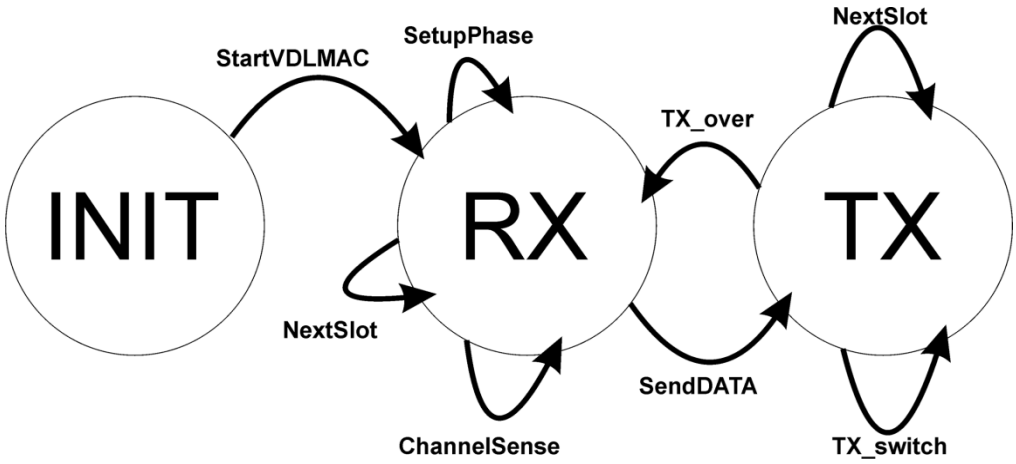


Рисунок 20. Граф состояний модуля MAC

В начальном состоянии INIT устанавливаются события окончания фазы настройки и перехода к следующему слоту, также состояние модуля MAC устанавливается на прием, а текущий слот приравнивается нулю. Далее (*StartVDLMAC*) происходит переход в состояние RX (прием), в котором может быть несколько событий: следующий слот (*NextSlot*), окончание фазы настройки (*SetupPhase*), прослушивание канала (*ChannelSense*) и отправка данных (*SendDATA*). Событие «отправка данных» переводит MAC из состояния RX в TX (передача), в котором также могут происходить несколько событий: следующий слот, переключение на передачу (*TX_switch*) и переключение на прием (*TX_over*). Событие «переключение на прием» меняет состояние TX на RX.

3.2 Файлы модели и проекта в среде Eclipse

Основная часть кода модели внедрена непосредственно в C++ проект MiXiM и распределена по его файлам. Т.к. базовых изменений в коде библиотеки не проводилось, код модели помещен в раздел модулей (далее будут описаны изменения, произведенные в этом разделе).

В MiXiM были использованы следующие разделы:

- «analogueModel» - содержит файлы, описывающие эффекты, налагаемые на распространяемый сигнал согласно различным математическим моделям. Функционал MiXiM позволяет производить расчёт изменений, как во временной, так и в частотной области;
- «application» - директория уровня приложений ЭМВОС, в рамках моделирования, по сути, являющимся генератором информационного (пользовательского) трафика;

- «connectionManager» - раздел для файлов, описывающих функционал специальных модулей управляющих созданием и удалением связей между узлами;
- «mac» - директория, носящая название подуровня доступа к среде канального уровня ЭМВОС. Является разделом для реализации различных протоколов канального уровня;
- «messages» - содержит специфичные файлы OMNeT++ формата .msg описания различного рода сообщений, а также их заголовочные файлы и файлы источника кода;
- «netw» - раздел реализаций протоколов сетевого уровня ЭМВОС;
- «nic» - директория для сложных модулей OMNeT++ специфичных для библиотеки MiXiM – network interface control (контроллер сетевого интерфейса). Они состоят из модулей физического и MAC уровней и связей между ними. Только с этим типом программных модулей взаимодействуют модули управления соединением (раздел connectionManager). Прямое объяснение создания подобных модулей и выделения отдельного раздела отсутствует. Предположительно, это отсылка к реальным устройствам типа «сетевая плата», реализующим в себе функционал двух указанных уровней;
- «node» - директория хранящая описания в формате «.ned» скомпонованных сетевых узлов, т.е. собранных из различных модулей с указанной иерархией между ними и полностью готовых для использования их в модели;
- «phy» - раздел хранящий в себе файлы моделей решающих устройств и описание работы модуля физического уровня. Модуль физического уровня производит инициализацию модели распространения радиоволн и решающего устройства в едином модуле;
- «power» - содержит файлы описания модулей типа «батарея», используемых для реализации моделей с автономными узлами;
- «transport» - директория для различных протоколов транспортного уровня ЭМВОС, агрегирующих потоки с верхних уровней;
- «utility» - раздел, который содержит в себе дополнительные файлы и инструменты для модулей из других директорий. Например, заголовочные файлы с константами и общими внешними функциями или элементы протоколов межмодульного взаимодействия, обеспечивающие взаимный доступ к служебной информации.

На рисунке 21 показаны используемые директории в MiXiM.

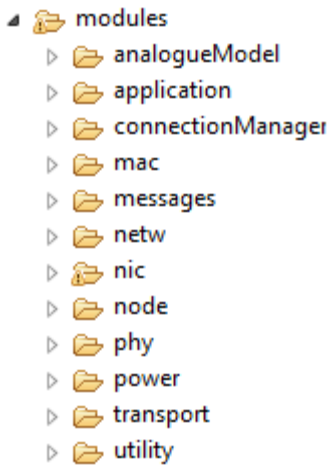


Рисунок 21. Используемые директории в MiXiM

В директорию моделей распространения радиоволн были помещены заголовочный файл и файл источника кода (FPL.h и FPL.cc), описывающие характер ослабления уровня сигнала при распространении в свободном пространстве для УКВ диапазона[95].

В разделе «mac» был помещён функционал подуровней MAC, LME, DLS и сетевого уровня, т.к. все они являются частью канального уровня VDL Mode 4,

В раздел «phy» был помещён код решающего устройства для VDL Mode 4, а также был изменён (дополнен) программный код физического уровня для генерации решающего устройства VDL Mode 4 совместно с указанной ранее моделью ослабления волн.

В директорию «utility» добавлен файл, содержащий некоторые константы и перечисления, а также программный код позволяющий обмениваться подуровню MAC и сетевому подуровню адресами второго уровня, прикрепляемыми к сообщениям, передающимся между этими подуровнями в качестве служебной информации.

В разделы «nic» и «node» помещены файлы в формате «.ned», описывающие комплексный модуль узла, функционирующего по стандарту VDL Mode 4 с дополнительной функцией многоинтервальной передачи данных АЗН-В.

Также в директорию сообщений было помещено несколько описаний различных типов сообщений, используемых в модели.

3.3 Описание программного кода и используемых алгоритмов

3.3.1 Физический уровень

Реализация физического уровня состоит из трёх элементов: модель распространения радиоволн, решающее устройство и непосредственно сам модуль физического уровня.

В рамках работы была использована модель распространения сигнала в свободном пространстве, реализованная в комбинации двух классов «FreePropagLossModel» и «FreePropagLossMapping».

Класс «FreePropagLossMapping» наследован от класса «SimpleConstMapping», который создаёт и хранит набор отсчетов и предоставляет к ним доступ с помощью константного итератора за постоянное время работы. Под отсчетами подразумеваются значения в каком-либо домене, по которым будет производиться отображение согласно модели распространения радиоволн. В рамках разработанной модели по данным отсчетам во времени будет определяться расстояние, пройденное сигналом, а согласно полученному расстоянию будет рассчитано ослабление сигнала. В качестве входных параметров для функции создания набора отсчетов задаются диапазон, шаг и домен. Атрибутами класса являются: расстояние, которое прошёл сигнал, указатель на модель распространения радиоволн (в данном случае на FreePropagLossModel) и домен (временной и/или частотный) в котором требуется проводить отображение. В классе кроме конструктора и конструктора копирования определены две функции:

- getValue – рассчитывает ослабление сигнала в зависимости от расстояния на несущей частоте, определенной для физического уровня;
- constClone – для создания константной копии объекта класса, для последующей работы с ним константного итератора.

Класс «FreePropagLossModel» наследован от класса «AnalogueModel», который предоставляет интерфейс для работы с моделью распространения физическому уровню. Переменными для класса являются: несущая частота сигнала, тип используемого пространства в модели (плоское, трёхмерное, тороидальное), размер пространства в граничных координатах, класс «FreePropagLossMapping» в качестве дружеского.

В классе кроме конструктора объявлены функции:

- initFromMap – функция для инициализации модулем физического уровня;
- filterSignal – для обработки сообщения типа «AirFrame», путём добавления ослабления сигнала в зависимости от времени.

«AirFrame» является представлением сообщения распространяющегося по каналу связи. Все физические уровни узлов, которые «слышат» «AirFrame», по информации хранящейся в нем, могут узнать уровень шума, определить наличие интерференции между несколькими сообщениями и т.д.

Класс решающего устройства наследован от класса «BaseDecider», который выступает в качестве основного класса для создания конкретных алгоритмов решающих устройств.

Основным назначением решающих устройств в MiXiM является проверка параметров получаемых сигналов и вынесения решения о возможности их корректного приёма, а также сбор статистики. Класс «VDLDecider» выполняет следующие основные функции:

- Проверка уровня сигнала относительно порогового значения;
- Если в канале несколько перекрывающихся сигналов, то решающее устройство проверяет уровень наибольшего из сигналов: он должен быть на 12 $дБ_м$ выше относительно других;
- Проверяет величину задержки сигнала при распространении канала (выполнение величины защитного интервала);
- Генерирует событие возникновения битовой ошибки в сообщении.

Входными значениями являются: порог отношения сигнал шум и чувствительность приёмника. Кроме конструктора и функции инициализации используемой физическим уровнем, в классе описана работа функций:

- «processNewSignal»: В первую очередь производится проверка наличия в решающем устройстве уже обрабатываемого сигнала. Если на данный момент решающее устройство не занято, то производится вычисление его мощности в текущий момент, и если уровень сигнала превышает порог, то сигнал отправляется на последующую обработку. Если же решающее устройство занято, то производится сравнение по мощности обрабатываемого и поступившего сигнала, если какой-то из них больше на 12 $дБ_м$, то решающее устройство продолжает обработку большего, иначе принимается решение о невозможности получения обоих сообщений;
- «processSignalHeader»: производит проверку по мощности длительностью равной длине заголовка. Решающее устройство работает в упрощенном режиме поэтому, если заголовок принят верно, то последующая проверка сигнала не проводится;
- «isSNRAboveThreshold»: функция рассчитывающая величину ОСШ (производится расчет уровня шума и уровня сигнала на заданном функции отрезке), в качестве результата выдаёт решение о результирующем значении относительно минимального значения (порога);
- «packetOk» - функция генерирующая событие появления битовой ошибки, вызывается при создании результата;
- «createResult» - генерирует результат работы решающего устройства, который затем направляется на уровень MAC в качестве служебной информации.

Для внедрения описанных классов в общую модель, был доработан код физического уровня, уже представленного в виде отдельного простого модуля в библиотеке MiXiM. В функцию getAnalogueModelFromName( ) была добавлена возможность инициализации реализованной модели распространения сигнала в свободном пространстве, а в функцию getDeciderFromName( ) – инициализация разработанного решающего устройства. Подробное описание принципов функционирования физического уровня в MiXiM приведено в публикации “MiXiM – The Physical Layer An Architecture Overview” [96].

3.3.2 Канальный уровень

Канальный уровень представлен тремя отдельными простыми модулями: доступа к среде, генератор сообщений “synchronization burst”, также выступающий промежуточным модулем для сетевого уровня и, непосредственно, подуровень маршрутизации (сетевой).

Сетевой подуровень был определен в составе канального уровня, т.к. он обеспечивает маршрутизацию на локальном уровне, по сути, между равными станциями (Ad Hoc) и делает это с помощью адресов второго уровня или как ещё обозначается – физических адресов. Несомненно, в этом подуровне присутствует множество признаков сетевого уровня модели ЭМВОС или сетевого уровня стека TCP/IP. Однако присутствуют и признаки кросс-уровневых систем: в предлагаемой модели подуровень маршрутизации имеет возможность непосредственного доступа к некоторым данным на MAC или LME подуровне, вместо того, чтобы просто отправлять запросы на обслуживание. На основе данных с обозначенных подуровней выбираются маршрутизаторы и определяются таймеры, т.е. функционирование подуровня неразрывно с другими.

3.3.3 Подуровень MAC

В разработанной модели подуровень доступа к среде совмещен с подуровнем VSS системы VDL Mode 4, т.к. этот подуровень определяет основную логику взаимодействия с картой слотов. Модуль представлен классом “VDLMacLayer”, который наследован от базового для данных типов модулей – “BaseMacLayer”. “BaseMacLayer”, в свою очередь, наследован от “BaseLayer”, который определяется в качестве базового класса для создания модулей отвечающих за конкретные уровни и подуровни. Базовый модуль содержит в себе идентификаторы соединений с верхним и нижним уровнями: служебное и межуровневое. Служебные соединения служат для передачи между модулями (уровнями) служебной

информации, например запрос состояния канала от уровня доступа к физическому и сформированный ответ в обратном направлении. Межуровневое соединение служит для передачи от уровня к уровню протокольных единиц (PDU – protocol data unit) для дальнейшей инкапсуляции/декапсуляции. Соответственно, если уровень находится наверху или внизу стека, то те или иные соединения будут отсутствовать. Также в базовом уровне определены функции, переопределение которых в наследуемых классах не подразумеваются – это функции отправки вверх и вниз сообщений по соответствующим соединениям и функции записи пакетов (записывает тип сообщения – собственное, входящее исходящее). Для исполнения роли базового класса определены также и чисто виртуальные функции, в которых должны быть описаны конкретные действия с сообщениями: “handleUpperMsg”, “handleLowerMsg”, “handleUpperControl”, “handleLowerControl”, “handleSelfMsg”.

Класс “BaseMacLayer” дополнен необходимыми атрибутами канального уровня: длина заголовка в битах и адрес второго уровня (MAC-адрес). Кроме этого в классе хранится длина заголовка физического уровня – длина обучающей последовательности, т.к. именно этот уровень передаёт параметры для передачи сообщения на физический уровень. Для этого используется функция “createSignal”. Принимающая на вход начало передачи сообщения, длительности передачи, мощность и скорость передачи. Исходя из этих параметров, вычисляется масштабирование по мощности и по скорости передачи (функции также определены), т.е. определяется массив точек для будущей обработки на приёме, о чём было упомянуто в описании физического уровня. Также среди параметров определен указатель на физический уровень: в данном случае он используется для управления переключения физического уровня на приём и передачу, таким образом, имитируется реальная форма радиоимпульса с нарастанием и затуханием по мощности (эти скорости можно определить для физического уровня через файл инициализации «.ini»).

Как было упомянуто, сообщения в MiXiM могут быть инкапсулированы и декапсулированы, поэтому в “BaseMacLayer” определены функции для данных операций с сообщениями. Полезным дополнением являются функции присоединения и считывания контрольной информации, прикрепляемой к сообщениям. Это сделано для того, чтобы можно было обрабатывать сообщения согласно какой-нибудь контрольной информации и не задействовать для этого отдельные сообщения, передаваемые между уровнями. Так, например, MAC-уровень может передавать адреса второго уровня сетевому уровню, который будет соотносить физические адреса с логическими, а затем, при отправке данных, специфицировать их для MAC-уровня при обмене с использованием соединения точка-точка.

Как упоминалось, совмещенный модуль, представляющий подуровни MAC и VSS системы VDL Mode 4 описан в классе “VDLMacLayer”. В классе объявлены два указателя на сообщение типа “VDLMacPkt”. Они используются для обработки данных synchronization burst, помещаемых в фиксированную часть сообщения VDL Mode 4 (данные с подуровня LME). Два указателя требуются из-за того, что данные запрашиваются за некоторое время до отправки, как это и должно быть в реальном транспондере. В модели запрос происходит непосредственно перед слотом отправки, т.е. в предшествующем слоте, но т.к. в последующем слоте от зарезервированного также может оказаться размещенное собственное резервирование, требуется второй указатель, чтобы не потерять запрошенные данные или не удалить существующие.

Для обработки сообщений, в которых передаются пользовательские данные (к ним относятся и сетевые сообщения) объявлена структура для создания и обработки очереди этих сообщений на уровне доступа – она включает себе указатель на сообщение и его приоритет. Таким образом, работа с пользовательскими данными происходит согласно очереди с приоритетами.

Работа MAC уровня основана на различных состояниях: инициализация, приём, передача. Для этого класс содержит перечисление (именованные константы) согласно этим состояниям и переменную для хранения текущего состояния.

Кроме некоторых перечислений используемых для заполнения сообщения VDL Mode 4 определено перечисление типов внутренних (собственных) сообщений, которые используются на MAC уровне:

- Сообщение для начала процесса инициализации, используемое для задержки начала работы модуля. Это нужно для имитации разного время начала работы транспондеров;
- Сообщение о конце фазы настройки транспондера VDL Mode 4 длительностью 4628 ≈ 61,7 с. Это время нужно для постройки карты слотов. Однако фаза может закончиться и раньше, если потребуется ускорить процесс доступа к среде, например, если по включении транспондера рядом с ним окажется большое число ВС;
- Сообщение для запуска процесса проверки статуса слота – прослушивания канала;
- Сообщение о необходимости переключения транспондера на передачу и отправки сформированного сообщения;

- Сообщение об успешном приёме сигнала из эфира и необходимости его дальнейшей обработки;
- Сообщение о наступление окончания какого-либо события без результата;
- Переход на следующий слот;
- Синхронизация – имитация синхронизации по секундной временной метке, принимаемой со спутников ГНСС (выравнивание временной шкалы);
- Сообщение о необходимости обработки очереди пользовательских сообщений;
- Сообщение об истечении времени записи в таблице соседей узла.

Класс также содержит переменные для идентификации окончания фазы настройки: является ли узел базовой станцией, номер текущего слота, длительность слота (для возможности моделирования систем с иным кол-вом слотов и скоростью передачи), мощность передатчика, скорость передачи и индивидуальное время «включения» приёмопередатчика.

В модели определены несколько массивов, отражающих следующие данные, обрабатываемые на уровне MAC: таблица соседних узлов и соответствующие таймеры, фрагменты обрабатываемых сетевых сообщений, слоты, используемые для инкрементированного широковещательного доступа. Кроме этого, определены типы данных, отражающие текущие используемые параметры QoS и параметры выбора слотов.

Переменная “LastSlot” имеющая тип “Slot” используется для проверки изменений в последнем «целевом» слоте. Целевым слотом является слот, зарезервированный ранее другим узлом для вещания в нём. Дело в том, что обновление данных о резервировании происходит в момент приёма сообщения в нём, однако, может возникнуть ситуация, когда сообщение получено с ошибкой или узел вышел из зоны радиовидимости. Это выливается в то, что изменения в переменные для резервирования не будут внесены. Переменная “LastSlot” путём манипуляций с ней позволяет избежать ошибок в резервировании.

Самой ключевой переменной определяемой на уровне MAC и используемой им является пользовательский тип данных (класс), описывающий структуру временного слота. Дело в том, что для реализации механизмов подуровня VSS слоту присуждаются различные значения и состояния.

Два адреса второго уровня используются для определения узла, зарезервировавшего слот. Владелец может быть как нынешним, так и будущим, т.к. в информации, получаемой в резервировании (а также при собственной процедуре резервирования), имеются данные о смещении резервирования в другой слот в последующих «суперфреймах».

Уровень занятости слота используется при процедуре резервирования, он может иметь значения: свободен, занят и особенно занят. Эти состояния определяются документацией ICAO [78].

Кроме этого в структуре слота содержатся следующие переменные:

- уровень мощности принятого сигнала в слоте;
- текущее состояние резервирования (да или нет);
- тип резервирования;
- число «суперфреймов» (минут) на которые слот зарезервирован;
- номер потока слотов, уровень слота при резервировании (от 0 до 4);
- последнее сообщение, полученное в слоте;
- является ли резервирование широковещательным;
- кол-во минут, через которое слот будет зарезервирован другим пользователем;
- позиция будущего слота (в который сместится нынешний владелец);
- было ли получено периодическое смещение для нынешнего пользователя слота;
- позиция номинального слота, относительно которого он был выбран;
- для какого приложения был зарезервирован слот.

Для класса слота определены: конструктор по умолчанию, конструктор с параметрами, конструктор копирования, оператор присваивания, функция перестановки между двумя переменными типа “Slot”.

Следуя стандарту VDL Mode 4, каждый MAC модуль модели имеет массив из 4500 слотов.

Функции, определенные в классе, будут описаны ниже по мере описания функционирования модуля MAC.

На рисунке 22 изображен класс-граф канального уровня.

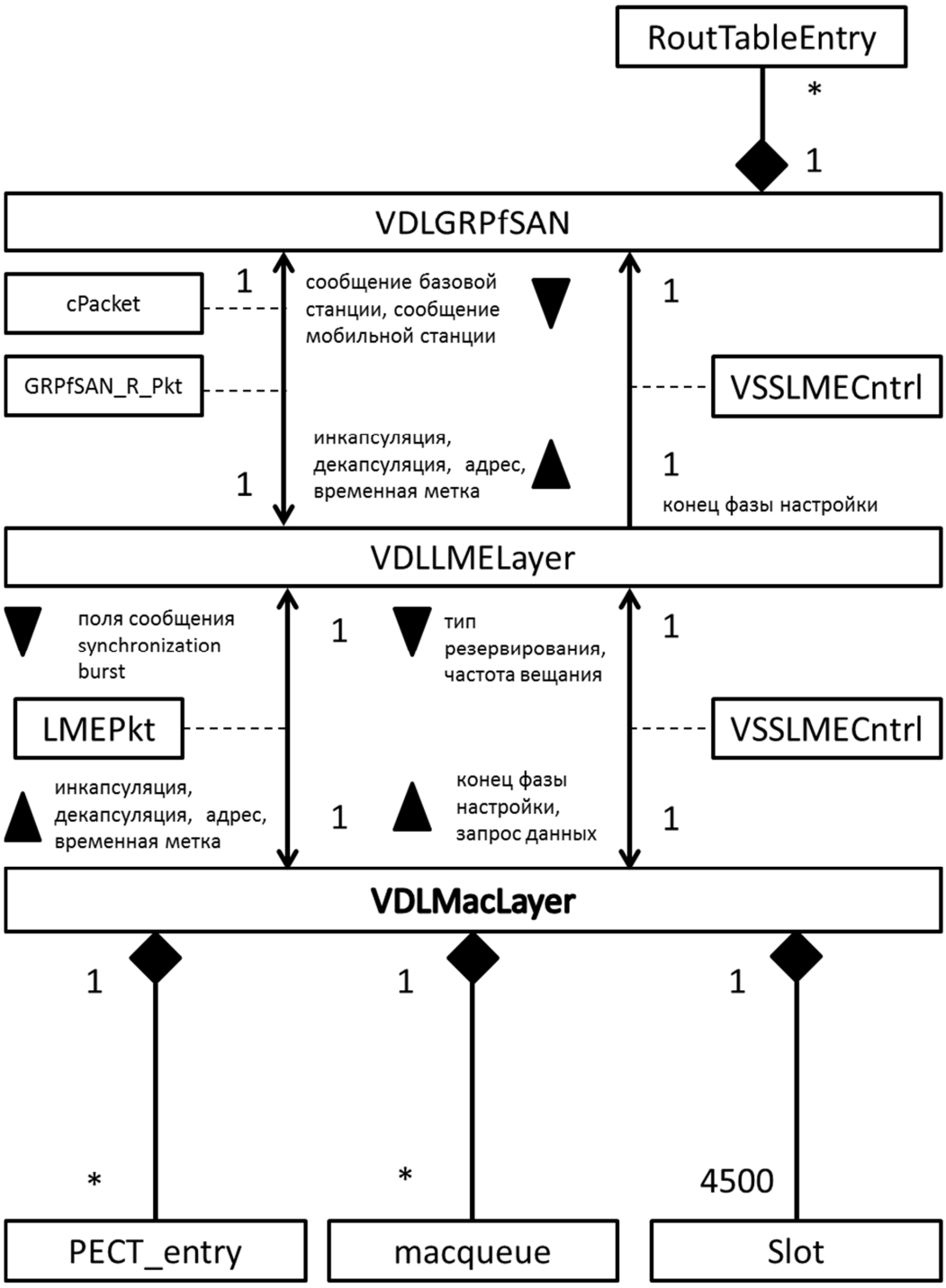


Рисунок 22. Класс-граф канального уровня

Непосредственно после инициализации объекта с помощью конструктора в общей программе симуляции вызывается функция initialize(), т.к. значения некоторых свойств могут быть получены только после конструирования других (или всех) объектов в модели. Таким образом, например, объект получает значения для переменных-членов из файла инициализации и создаёт в куче объекты внутренних таймеров, представленных в классе сообщений OMNeT++. Также, могут быть определены те атрибуты, значения которых требуется отслеживать с помощью графического интерфейса в процессе отладки модели –

функция WATCH(). В качестве входных параметров функция initialize() принимает целочисленную переменную, характеризующую стадии инициализации. В MiXiM обычно используются две стадии: на первой производится сама вышеописанная инициализация, а на второй производится проверка полученных атрибутами класса значений с установленными ограничениями, при этом значения могут быть выведены с помощью объекта “ev” выходного потока для графического интерфейса Tkenv.

Tkenv представляет собой графический интерфейс пользователя для проведения моделирования. Он поддерживает интерактивное выполнение моделирования, анимацию, инспекцию, трассировку и отладку. В дополнение к процессу разработки модели и её отладки, Tkenv полезен на этапе изучения принципов моделирования в OMNeT++, а также для графической презентации работы модели, так как он позволяет пользователю получить подробное представление о состоянии и истории модели в любой момент её исполнения. При комбинировании вместе с отладчиком, Tkenv может значительно ускорить разработку модели. Наиболее важные возможности Tkenv:

- визуализация сети;
- отображение потока информации исходящей от модулей (объект “ev”);
- поддержка различных режимов исполнения: от события к событию, нормальное, быстрое, экспресс;
- выполнение модели до определенной точки (в моделируемом времени или до конкретного события);
- перезапуск моделирования;
- анимация процессов обмена сообщениями;
- просмотр содержимого объектов и переменных в модели;
- визуализация статистики (гистограмм и т.д.) во время выполнения моделирования;
- запись событий для последующего анализа.

Возвращаясь к функции initialize(), на первом этапе происходит регистрация сигналов для записи статистики отправленных и полученных сообщений между MAC-уровнями различных узлов. Далее с помощью синтаксического анализа («парсинга»), производится присваивание переменным-членам значений из файла инициализации: мощность передатчика, флаг базовой станции, длина очереди сообщений, скорость передачи (бит/с), время начала работы модуля. Подуровень устанавливается в состояние: инициализация. В куче создаются объекты всех необходимых внутренних сообщений. В сообщении для проверки состояния канала устанавливается режим прослушивания до указанного времени, а

само значение времени на 0.9 длительности слота, т.к. за это время слот ещё должен находиться в состоянии максимума мощности, что поможет явно определить его занятость. Далее запускается таймер старта модуля. Если модель находится в режиме отладки, то включается наблюдение за представляющими интерес атрибутами, в подавляющем большинстве – это конкретные элементы массива слотов и их переменные-члены. Для каждого модуля в отладочных и презентационных целях создается файл, куда записываются время получения сообщения, идентификатор источника и его местоположение.

На втором этапе в функции initialize() происходит сравнение максимальной мощности передачи установленной в модуле управления соединениями и значения мощности передатчика на подуровне MAC, при этом мощность передатчика не может быть выше, т.к. по максимальной мощности рассчитывается возможная дальность радиосвязи. Также происходит проверка состояния физического уровня модели, т.к. подразумевается, что работа начинается с состояния приёма и при попытке переключения на передачу ничего не произойдёт.

В конечном итоге, в качестве информации для отладки выводятся результирующие значения переменных-членов класса.

Стоит отметить, что для запуска собственных сообщений используется встроенная функция scheduleAt(), которая принимает указатель на сообщение и время его прихода. Таким образом, модуль посылает сообщение самому себе, а установленное время прибытия заносится в набор будущих событий модели.

Функция handleUpperMsg() является перегруженной версией функции из класса BaseMacLayer. Она принимает на вход указатель на тип данных cMessage, а далее проверяет категорию этого сообщения. В модели, от вышележащих модулей может приходить только две категории сообщений – АЗН-В и сетевые: InterSub_ADSBmsg и InterSub_NetwMsg соответственно. Приставка InterSub указывает на то, что данные категории сообщений «распознаются» на всех подуровнях, начиная с MAC и выше, и объявлены в качестве перечислений в заголовочном файле, содержащим константы – VDL.h.

Если обрабатываемое сообщение из категории АЗН-В, то производится проверка на равенство нулю двух указателей на тип VDLMacPkt поочередно. Дело в том, что может возникнуть ситуация, когда указатель ещё не освобожден, т.е. сообщение не отправлено, однако, запрос на данные сверху уже также произведен и ответ получен. Это возникает в момент собственного резервирования узлом двух и более слотов подряд для отправки разных сообщений, т.к. данные запрашиваются перед слотом, в котором они должны быть отправлены – это упрощенная реализация реальной системы. Если оба указателя заняты, то

модуль сигнализирует об ошибке и моделирование останавливается, что свидетельствует о лишних или повторяющихся запросах данных на верхние уровни. Если же один из указателей свободен, то с помощью функции инкапсуляции encapsMsg() и стандартных манипуляций приведения типов они получают адрес нового сообщения в области динамической памяти.

Для обработки сообщений из сетевой категории используется очередь типа «первым пришёл – первым вышел» - FIFO. Изначально, так же как и в первом случае, динамически выделяется память под новое сообщение, инкапсулирующее данные сверху. Далее создается объект элемента очереди сообщений – он содержит копию результирующего сообщения и его приоритет, т.к. предполагается, что очередь в будущем будет функционировать и с приоритетами. Далее производится проверка существования объекта собственного сообщения включающего процессы обработки очереди. Если до принятия данных сверху в очереди не было сообщений, то момент обработки очереди включается в набор будущих событий (тип собственного сообщения VDLMAC_MacQueue).

Функция обработки контрольных сообщений от верхних модулей handleUpperControl() лишь передаёт указатель на сообщение функции handleSelfMsg(), т.к. именно в этой функции сосредоточена большая часть процессов по обработке сообщений в зависимости от состояния модуля MAC.

Итак, функция по обработке собственных сообщений отслеживает и изменяет текущее состояние подуровня доступа к среде и в зависимости от этого производит обработку поступившего сообщения.

Модуль начинает свою работу с состояния инициализации, самое первое сообщение которое будет обрабатывать функция – это сообщение о старте работы модуля. Как было сказано, событие, представленное этим сообщением, планируется ещё на этапе инициализации модуля как части общей модели функцией initialize(), там же получается само значение времени старта из файла инициализации. В условии связанном с типом стартового сообщения проводится проверка на кратность времени старта секунде. В положительном случае в этот же момент работы модели генерируется событие «синхронизация», иначе модуль дожидается времени кратной секунде. Этот механизм также является отражением реальной системы, синхронизирующейся по секундной метке ГНСС.

При возникновении события «синхронизация», представленного собственным сообщением «Sync» с типом «VDLMAC_Sync» запускается фаза настройки модуля доступа к среде. Фаза настройки относится и ко всем верхним модулям, т.к. состояние «передача» становится доступным лишь по окончании фазы настройки. Процедуры ускоренного доступа

к среде и необходимость их применения затрагиваются в части 2 документа ICAO 9816 (например, если станция при включении обнаруживает большое количество соседних станций), в обычно же режиме фаза настройки длится минуту и 128 слотов, т.е. 4628 слотов суммарно. Соответственно, на этот момент и определяется событие окончания фазы настройки. Далее согласно текущему времени модели рассчитывается временной слот, в котором находится подуровень доступа к среде, при этом состояние подуровня устанавливается в режим приёма. Также запускаются события наступления следующего слота и синхронизации, на физический уровень отправляется запрос прослушивания канала.

Следующим состоянием подуровня доступа к среде является «приём», при этом присутствует два дополнительных состояния: период фазы настройки и период после её окончания, на предмет чего и производится проверка в самом начале обработки сообщения. Далее рассматривается тип переданного в функцию сообщения.

В первую очередь проверяется событие наступления следующего слота. Данное событие является ключевым в модели, при его наступлении обновляется большинство счётчиков, состояний и переменных, как в самом слоте, так и в модуле непосредственно, т.е. можно сказать, что моделирование происходит «послотого».

Изначально инкрементируется значения текущего слота, в котором находится подуровень MAC, и происходит проверка выхода значения за величину в 4500 слотов. Производится проверка наличия будущего источника для резервирования слота и момент наступления резервирования в следующем «суперфрейме» по специально отведенному для этого счетчику, устанавливаемому согласно информации резервирования получаемой из информационных сообщений от других узлов. В положительном случае, осуществляется обновление всех необходимых переменных-членов объекта согласно новому резервированию слота:

- новым источником, зарезервировавшим слот, объявляется будущий источник;
- величина длительности по количеству «суперфреймов» резервирования выбирается случайной в пределе от 4 до 8 и будет уточнена согласно полученному в будущем сообщению;
- поля, связанные с будущим резервированием, обнуляются: позиция будущего слота для нового источника, факт получения периодического смещения;
- тип резервирования устанавливается в значение «широковещательный». На данный момент поддерживается только этот тип резервирования и его разновидность - комбинированная с инкрементированным резервированием,

однако в будущем, планируется резервирование слота в зависимости от требуемого источником типа;

- слот объявляется в качестве зарезервированного;
- предполагается, что слот будет зарезервирован для нужд АЗН-В, однако, эта информация также уточняется во время получения сообщения от нового источника;
- очищается информация о последнем, полученном в этом слоте, сообщении от прошлого источника, если она присутствовала.

Если резервирование должно наступить более чем через один «суперфрейм», то просто производится декрементирование соответствующего счётчика. В любом из случаев производится планирование события наступления следующего слота и запуск прослушивания состояния канала.

На следующем шаге производится проверка переменной обозначенной, как LastSlot. Она рассматривается на каждом новом переходе слота и заполняется, только если у слота есть текущий источник отличный от самого узла. Этому предшествует проверка на изменение переменных и содержимого полученного сообщения относительно LastSlot и предыдущего слота. Эта мера нужна на случай, если в зарезервированном слоте ничего не будет получено, что может привести к пропуску обновления некоторых счётчиков в слоте.

Также, в режиме фазы настройки возможен приём сообщений от других узлов для составления карты слотов, он одинаков и после окончания фазы настройки. Сообщения от других узлов приходят после их обработки на физическом уровне, и для них вызывается функция handleLowerMsg( ), которая точно так же, как и функция handleUpperControl( ) перенаправляет обработку в функцию handleSelfMsg( ) по описанным ранее соображениям. Для удобства работы все информационные сообщения помечаются при отправке типом VDLMAC_DATA, на предмет соответствия которому и происходит проверка в определённых функцией случаях.

В первую очередь, после приёма, с помощью глобальной функции emit() производится запись в общую статистику модуля информации о полученных на подуровне MAC сообщениях (процесс записи статистики будет описан ниже). Далее вызывается функция ReceiveReserv( ) для записи необходимой информации в карту слотов. После этого проверяется тип резервирования и идентификатор сообщения, при этом требуется ещё и узнать тип инкапсулированного сообщения. Если рассматривать этот процесс с логики разделения уровней в модели, то все модули входят в состав канального уровня и как такового вышестоящего уровня (например, сетевого) у него не имеется, т.е. модули

разделены исключительно логически. В различных случаях могут быть задействованы, как и все вышележащие модули, так и только один – для этого и проводится проверка идентификатора инкапсулированного сообщения. Если проверяемый тип не является одним из сетевых (от базовой или мобильной станции), то он может быть обработан на подуровнях MAC и VSS. Если же полученный тип – сетевой, то он отправляется на подуровень LME, при этом предварительно, с помощью функции decapsPkt( ), декапсулируется информационная единица протокола, содержащееся в обрабатываемом сообщении (следующий этап в цепочке if else).

При дальнейшей обработке VDLMAC_DATA, производятся процедуры записи данных в текстовый файл для отладки, создание которого осуществлялось ещё в функции initialize( ). Затем производится проверка наличия записей в таблице соседних узлов, которая в документе ICAO 9816 носит название Peer Entity Contact Table – PECT. Если запись отсутствует, то создается новая, в неё помещаются копии PDU с подуровней MAC и LME, счётчик доступности устанавливается в 0, а сам узел помечается в качестве доступного. Счётчик доступности срабатывает по возникновению соответствующего события, для этого создается объект cMessage, которому присваивается название соответствующее адресу узла, от которого было получено информационное сообщение, а также задается тип VDLMAC_PECT. При внесении события в набор будущих событий (future event set) модели происходит расчет момента времени возникновения события. Это происходит согласно параметру TL3, обозначенному в части 2й документа ICAO 9816. Согласно этому параметру ищется слот, который зарезервирован целевым узлом в будущем, если такого слота нет, то таймер откладывается на «суперфрейм». Если подобная полученной записи уже существует, то производится её поиск в общем массиве и очистка, соответствующие процедуры производится и для счётчика доступности.

На следующем этапе ветвления происходит обработка события окончания фазы настройки – VDLMAC_EndOfSetupPhase. Соответственно производится отмена состояния фазы настройки модуля, создается объект служебного сообщения, используемого между подуровнями и происходит его отправка с пометкой об окончании фазы настройки. Также происходит проверка на наличие собственного резервирования в следующем слоте, т.к. данный случай не исключен.

Следуя далее по операторам ветвления для обработки событий, очередной идёт обработка события синхронизации. Этот этап отличается от подобного, происходящего в состоянии инициализации. Здесь может существовать два случая: по возникновении события слот либо уже перешёл в состояние кратное 75 слотам или 1 секунде (т.к. синхронизация

происходит по секундной метке) или нет. Во всех случаях надо произвести одинаковые действия: отменить текущее событие следующего слота в наборе будущих событий и запустить его снова, тем самым «замедлив» или «ускорив» течение карты слотов.

Далее, в процессе ветвления состояния RX, рассматривается обработка события изменения счётчика доступности узла. На этапе создания, в качестве имени счётчика (собственного сообщения) присваивается адрес узла, к которому он привязан. Соответственно, при обработке происходит сравнение адреса в таблице соседей и имени собственного сообщения. Если соответствие найдено, то счётчик увеличивается на единицу, при этом проверяется его результирующее значение. Если оно достигло 3х, то запись удаляется из таблицы (принимается решение о том, что узел вышел из зоны радиовидимости), счётчик так же удаляется из общего массива, иначе, счётчик перезапускается.

Также, для отладочных целей существует случай обработки неизвестного сообщения, тогда модуль сообщает в информационной консоли о неизвестном событии.

Работа модуля в состоянии RX при окончании фазы настройки имеет небольшие дополнения и изменения в обработке событий в операторах ветвления.

При обработке события перехода в следующий слот появляются процедуры связанные с передачей данных: если в текущий слот зарезервирован самим узлом, то состояние модуля переключается в TX (передача), и запускается событие отправки данных. Также, производится проверка необходимости запроса данных с верхних уровней, с помощью служебного сообщения класса VSSLMECntrl, для приложения АЗН-В или для отправки нулевого резервирования в слоте, который был зарезервирован под передачу сетевых сообщений, но при этом очередь сообщений оказалась пуста.

В операторы ветвления был добавлен случай обработки запроса от подуровня LME на резервирование слотов для приложения АЗН-В, для этого вызывается функция CreateReserv( ), а также событие для обработки очереди сообщений, упомянутого в функции обработки сообщений от верхних уровней.

Итак, при возникновении события типа VDLMAC_MacQueue, проверяется тип первого сообщения, т.к. по принципу очереди FIFO из связного списка при отправке должен будет извлечён первый элемент. Рассмотрение типа требуется для дальнейшей модификации модели для работы с очередью, в которой находятся разные типы сообщений, т.к. при этом могут понадобиться особые процедуры резервирования слотов и пр. Основным элементом в обработке события, касающегося очереди, является процесс выбора слота. Т.к. разработанный алгоритм маршрутизации позволяет использовать «однослотовые»

сообщения, то было принято решение использовать процедуры фиксированного доступа к среде, а в частности: комбинированные широковещательное и инкрементированное резервирование, не смотря на то, что стандартом предусмотрено использование случайного доступа для передачи пользовательской информации. Всё дело в том, что приложение многоинтервальной передачи данных АЗН-В реализуемое с помощью протокола маршрутизации мобильной самоорганизующейся Ad Hoc сети, по своей сути схоже с самим приложением передачи данных АЗН-В. Конкретнее, так же создаётся поток сообщений, несущий информацию о местоположении и намерениях узлов, в котором сообщения следуют друг за другом с фиксированной частотой, например, раз в минуту. Основное отличие: каждое сообщение будет испытывать индивидуальные задержки, о величине которых будет сказано в разделе, посвященной результатам моделирования. При использовании случайного доступа, с увеличением количества узлов в сети, особенно в моменты вещания базовой станцией сетевого сообщения, которое должно быть ретранслировано всеми, ситуация на подуровне доступа к среде станет схожей с методом множественного доступа типа «слотированная алоха» (slotted aloha), относительная пропускная способность которой не достигает и значения величиной 0.4 [97]. Возвращаясь к процессу выбора слотов, изначально устанавливается диапазон поиска подходящего слота. Начало устанавливается на позицию, следующую за текущим слотом. Для определения позиции конца используется специально внедренный параметр, в коде носящий имя NetwReserv_QoS_Par. Данный параметр имеет размерность в слотах, т.е., если параметр равен 400, то именно на это количество слотов вперёд производится поиск подходящего для комбинированного широковещательного и инкрементированного резервирования слота. Подходящим для этого типа резервирования является слот, зарезервированный станцией для собственных нужд на число «суперфреймов» более трёх. Если при поиске слот соответствует всем условиям, то поиск заканчивается, а номер слота вносится в массив, который будет прочитан при отправке данных АЗН-В, если номера слотов будут совпадать, то вместо широковещательного резервирования будет отправлено комбинированное. Однако в процессе поиска также может встретиться слот уже зарезервированный для сетевых нужд, следственно поиск останавливается и далее никаких манипуляций не производится. В том или ином случае, если подходящего слота найдено не было, вызывается функция CreateReserv( ) и для сетевых нужд создается простое широковещательное резервирование.

Идеология использования параметра NetwReserv_QoS_Par заимствована из подхода к резервированию используемого непосредственно в системе VDL Mode 4. В ней имеется набор параметров на подуровне VSS для резервирования собственных слотов и

перерезервирования чужих слотов приёмопередатчиком. Некоторые из них позволяют менять диапазоны и условия выбора слотов, меняя тем самым соотношение «скорость доступа к среде / вероятность коллизии». При этом в документах 9816 упоминается, что приложения могут выступать инициаторами в процессах резервирования слотов. В общей совокупности назначения указанных параметров и возможностей приложений, в системе VDL Mode 4 реализован принцип качества обслуживания трафика приложений – QoS. Результаты вариаций параметра NetwReserv_QoS_Par приведены в разделе с результатами моделирования. На этом рассмотрение работы модуля MAC в состоянии RX окончено, далее будет рассмотрено состояние TX.

Случай работы модуля в состоянии TX является «коротким» относительно INIT и RX. В нём присутствуют:

- Событие синхронизации;
- Обработка очереди сообщений;
- Включение передатчика на физическом уровне (тип собственного сообщения VDLMAC_SendData). На физическом уровне через файл инициализации указывается время переключения состояния с приёма на передачу (RXtoTX). Таким образом, в разработанной модели имитируется период нарастания мощности в реальном радиосигнале;
- В дальнейшем также должно существовать событие перехода в следующий слот, т.к. в полноценной реализации модуль должен отправлять «многослотовые» сообщения.

В порядке упоминания далее будет дано описание функций encapsMsg( ), ReceiveReserv(), CreateReserv( ) и decapsPkt( ).

Как видно из названия функции encapsMsg( ) её предназначение заключается в инкапсуляции данных. Входными данными является указатель на объект типа cPacket*, встроенный в OMNeT++. Принимаемые с верхних подуровней информационные единицы соответственно инкапсулируются в информационную единицу подуровня MAC. Именно здесь начинается формирование конечного облика любого сообщения VDL Mode 4, т.е. в модели создается объект, который содержит все поля подуровня. Значения этих полей по большей части будут определены перед непосредственной отправкой сообщения, чтобы наиболее точно отразить текущее состояние приёмопередатчика и информации о резервировании. Обработка информации с верхних уровней начинается с определения идентификатора данных. Т.к. подуровню MAC и VSS предшествует подуровень LME, то проверка производится именно по совокупности идентификаторов данных, содержащихся в

информационной единице этого подуровня. Также, определяется служебная информация (в данном случае это адрес второго уровня), которая также принимается во внимание при инкапсуляции. На данный момент в модели для подуровня LME используется три типа данных: synchronization burst типа «базовое сообщение», сетевое сообщение от базовой станции и сетевое сообщение от мобильной станции. Если сверху был получено базовое сообщение АЗН-В, то в куче создается новый объект типа VDLMacPkt и адрес источника устанавливается в собственный адрес узла.

Случай с сетевыми данными требуется рассмотреть немного отдельно, т.к. требуется описание механизмов работы функции декапсуляции. Как уже упоминалось, при обработке сообщения могут быть задействованы как один, так и несколько поудровней. Например, при обработке сообщений от мобильной станции адрес следующего ретранслятора указывается в части обрабатываемой подуровнем маршрутизации, фиксированная же часть сообщения, относящаяся к подуровням MAC и VSS, должна при этом остаться неизменной, т.к. хранит адрес источника сообщения, с которым, в конечном итоге, на базовой станции будут соотнесены данные о местоположении, хранящиеся в PDU LME и они также должны остаться неизменными. В этих целях, при декапсуляции на подуровне MAC, в качестве служебной информации блоку данных назначается временная метка, эта метка будет уникальной, так как уникален временной момент начала обработки каждого сообщения. Переходя от одного верхнего уровня к другому при декапсуляции сетевых сообщений процесс повторяется, таким образом, на всех подуровнях существуют данные связанные одинаковой меткой. Если на каком-то подуровне удаляются такие данные, то по указанной метке находятся связанные с ними на других подуровнях и также удаляются. Учитывая рассмотренный механизм, на каждом подуровне имеется массив для хранения частей сообщений.

Возвращаясь к процессу инкапсуляции сетевых сообщений, можно отметить два варианта: с верхнего уровня поступила новая информация, или информация, которая была обработана и связана временной меткой на этапе декапсуляции. При этом всегда используется поиск в массиве обрабатываемых частей сообщений, который использует служебную информацию об адресе и временной метке. В любом случае, при поступлении PDU с идентификатором сетевой информации от базовой станции (BSRetrMsg), создается новый экземпляр фиксированной части сообщения VDL Mode 4, т.к. протокол маршрутизации подразумевает обновление информации о местоположении и адресе излучающего узла на каждом этапе ретрансляции этого сообщения. При положительном результате поиска производится удаление связанной с этим PDU части на MAC уровне.

Процесс обработки информации с идентификатором сетевого сообщения мобильной станции (MSADSBData) по большей части схож. Используется тот же механизм поиска, однако, при его положительном результате связанная часть сохраняется, как того требует протокол маршрутизации. Отрицательный результат сигнализирует о том, что с верхних уровней получена собственная, сгенерированная самим узлом, информация, в связи с чем, требуется создание нового экземпляра фиксированной части – информационной единицы протокола подуровней MAC и VSS.

В конце функции encapsMsg( ) к общей длине сообщения в битах добавляется длина фиксированной части – 58 бит, а далее вызывается функция encapsulate( ), которая отдаёт переданный в функцию объект во владение возвращаемому объекту.

Функция CreateReserv( ) принимает на вход указатель на тип cMessage, который, является базовым для всех типов сообщений. Изначально определяются переменные характеризующие тип резервирования, приложение, которое размещает резервирование и кол-во слотов в минуту, которое нужно зарезервировать. На данный момент функция рассматривает два типа приложений и создаёт для них периодическое широковещательное резервирование. При этом для определения переменных, в случае резервирования для АЗН-В приложения, используется информация, переданная через междууровневое служебное сообщение, а в случае с сетевым достаточно соответствие типа сообщения, находящегося в очереди. Для поиска подходящих слотов используются параметры и методы, определяемые документом 9816; параметры имеют значения по умолчанию. Согласно значению кол-ва слотов необходимых для резервирования создается массив номинальных слотов, т.е. тех слотов вокруг которых будет размещён диапазон выбора (т.е. сама переменная равна размеру одного «крыла» диапазона выбора). На следующем этапе производится определение диапазона выбора слотов. Если резервирование комбинированное широковещательное и инкрементированное, то позиция первого номинального слота (и его диапазон выбора) размещается относительно текущего слота, иначе размещение происходит относительно начала «суперфрейма». Согласно первой позиции номинального слота заполняются остальные элементы массива номинальных слотов. При осуществлении процедуры поиска определяются начало и конец диапазона выбора для каждого номинального слота, таким образом, чтобы не выходить за пределы «суперфрейма».

В зависимости от значения переменных связанных с состоянием слота, слоты разделяются на 5 уровней. Каждый слот в диапазоне помещается в массив соответствующий его уровню. Далее проводится восходящая проверка на условие, что хотя бы в одном из этих массивов находится требуемое для выбора кол-во слотов. Если условие выполнено, то

массив передается в функцию FindBestSlot( ) для окончательного определения слота для резервирования, т.к. в массиве нужно ещё определить слот с наименьшим уровнем шума, причём таких слотов может быть несколько, тогда среди них слот выбирается случайно (с использованием равномерного распределения). Далее, у результирующего слота заполняются все необходимые поля для дальнейшего обслуживания резервирования, при этом, если слот удалось зарезервировать на период менее 4х «суперфреймов», то с помощью функции FindFtSlot( ) для него находится слот для будущего смещения. На рисунке 23 проиллюстрирован процесс выбора слота для периодического широковещательного резервирования, где: НИ- номинальный интервал, НСС – номинальный стартовый слот, НС – номинальный слот, НСП – номинальный слот передачи, ИВ – интервал выбора.

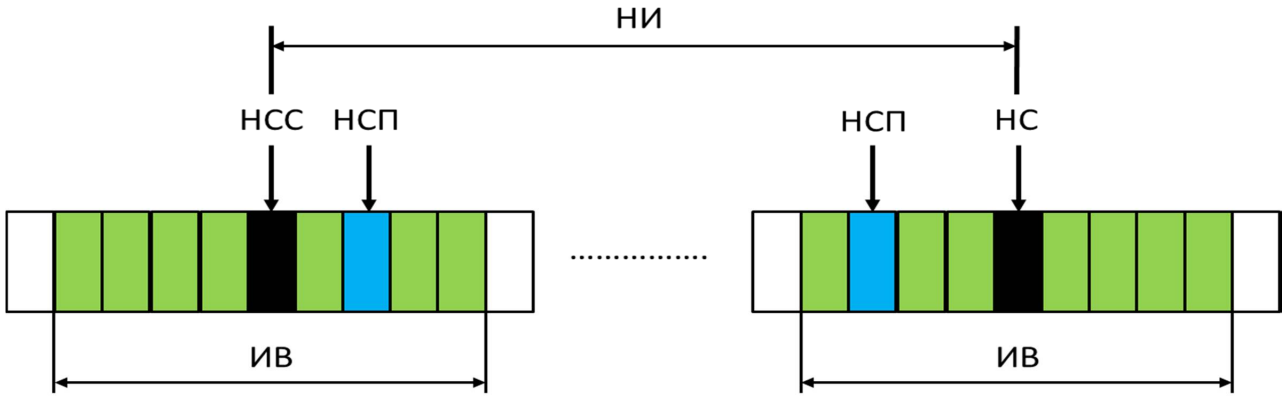


Рисунок 23. Иллюстрация процесс выбора слота для периодического широковещательного резервирования

При поиске слотов может случиться ситуация, когда ни одного слота не найдено, тогда разрешается сдвигать группу номинальных слотов, чтобы рассмотреть слоты, изначально не вошедшие в диапазон поиска. Т.к. лучше всего занимать слоты, начиная с наименьшего уровня, сдвиг производится каждый раз, когда таких слотов не найдено вообще. В самом первом случае ветвления, если слотов нулевого уровня не найдено, то слоты более высоких уровней не ищутся, при этом сразу следует переход к сдвигу. Суммируя, можно сказать, что процесс поиска разбит на группы сдвигов, которые происходят в восходящем согласно уровню слотов режиме. Сдвиг начинается вправо на полный диапазон поиска (двойная величина переменной), пока крайняя правая часть этого диапазона не «упрется» в последний слот «суперфрейма», затем в обратном порядке. Соответственно, если в процессе сдвига в обе стороны не было найдено ни одного слота нулевого уровня, то процесс повторяется, но уже относительно слотов первого уровня и т.д. В конечном итоге, если поиск не увенчался успехом, то на верхний уровень отправляется уведомление об этом. После процесса поиска слота, зарезервированным может оказаться следующий слот после текущего, поэтому производится проверка на это условие и запрос

данных на верхний уровень, при положительном результате. Вопросы оптимальности принципов сдвига не рассматривались.

В функции CreateReserv( ) часто используются функции FindDistance( ) и isCCIProtect( ). Первая использует передаваемый ей адрес соседнего узла для поиска данных о его местоположении, а также через доступ к модулю мобильности, получает собственные координаты и далее рассчитывает расстояние до соседнего узла. Вторая применяет широкий набор правил для проверки слота на условия помехи, излучаемой узлами, вещающими в одном канале и в одинаковом слоте.

Также в CreateReserv( ) используется функция SetSlotForReserv( ), которая принимает номер слота и тип резервирования. Исходя из полученных данных, заполняет некоторые поля структуры слота связанные с резервированием, при этом, если у слота нет будущих владельцев, то срок резервирования устанавливается в случайный период от 4-х до 8-ми «суперфреймов».

Функция ReceiveReserv( ) принимает на вход указатель на сообщение и указатель на слот. Так же, как и в функции CreateReserv( ) определяются тип резервирования и приложение, к которому оно принадлежит, из полученного сообщения считываются значения периодического таймаута и периодического смещения. Далее используется оператор ветвления, выбирающий случай в зависимости от типа резервирования. Для случая комбинированного резервирования, отдельной ветви не требуется, т.к. оно определяется при значении периодического таймаута равного трём и ненулевой величины периодического смещения. В общей сложности рассматривается 4 случая разных значений периодического таймаута и смещения, при этом учитываются условия перерезервирования (конфликта слотов).

Функция FindFtSlot( ) принимает на вход номер слота и во многом схожа с функцией CreateReserv( ), однако:

- имеет фиксированный диапазон поиска слотов;
- в качестве номинального слота использует тот же номинальный слот, что и резервирование, которое будет перемещено в этот будущий слот;
- отсутствует сдвиг слотов.

Весь остальной функционал, такой, как деление слотов на уровни с применением к ним параметров качества обслуживания, в общем и целом сохраняется.

Также схожа с функцией CreateReserv( ) и функция FindIncremOffset( ), которая используется при необходимости создать комбинированное широковещательное и инкрементированное (так и просто инкрементированное) резервирование. В функции

используются параметры и условия определения диапазона поиска слотов для инкрементированного резервирования, причём значение смещения относительно слота, для которого требуется создать резервирование и сам диапазон поиска вокруг номинально слота разнится. В массивы слотов разных уровней включаются только те слоты, номер которых кратен четырём – это требуется в дальнейшем для преобразования значения в информацию о периодическом смещении, поэтому функция возвращает тип данных целого типа.

Важной частью модуля MAC, реализующей свойства системы VDL Mode 4, является функция handleLowerControl( ), которая обрабатывает служебную информацию с нижнего уровня, которым является физический уровень. Для обработки поступающей информации и событий, как и большинство других функций, handleLowerControl( ) использует один из операторов ветвления языка C++ - switch. Срабатывание различных случаев оператора зависит от типа сообщения, на которое указывает передаваемый в функцию указатель.

В случае обработки типа сообщения соответствующего ответу на запрос прослушивания состояния канала, из этого сообщения считывается результат прослушивания и принимается решение о занятости канала в текущем слоте, а также записывается полученная мощность сигнала.

В процессе обработки типа сообщения об окончании переключения приемопередатчика возникает два случая: физический уровень переключен на состояние передачи или на состояние приема.

Если приемопередатчик находится в состоянии приёма, то состояние всего модуля так же переводится в состояние приёма.

Если приемопередатчик находится в состоянии передачи, то сообщение может быть отправлено, для этого надо произвести различные манипуляции. Производится проверка типа и приложения резервирования в текущем слоте. В случае широковещательного резервирования для приложения АЗН-В работа происходит с сообщением, хранящимся вне очереди (см. работу функции handleUpperMsg( )). Предварительно заполняются поля фиксированной части сообщения VDL Mode 4, а затем начинается процесс определения данных о резервировании. Если слот зарезервирован на число «суперфреймов» более трёх (строго), то поле периодического таймаута устанавливается в значение 3 и производится проверка на предмет того не входит ли рассматриваемый слот в массив кандидатов для размещения комбинированного периодического и инкрементированного резервирования. В положительном случае, вызывается упомянутая выше функция для инкрементированного резервирования FindIncremOffset( ) и результат её работы деленный на 4 (для того чтобы уместиться в биты поля) заносится в поле периодического смещения. В противном случае

поле периодического смещения устанавливается в 0. После рассмотренных процедур переменная в структуре слота, отвечающая за кол-во «суперфреймов», на которые слот зарезервирован, декрементируется.

Если текущий слот зарезервирован менее чем на 3 «суперфрейма», то периодический таймаут выставляется в значение на 1 меньше, чем кол-во «суперфреймов», на которые слот зарезервирован и производится проверка на предмет того, что в этом слоте уже была определена и отправлена величина периодического смещения. В положительном случае, используя значение будущего слота в который переместится текущий, рассчитывается и заполняется поле периодического смещения. В ином случае это значение получается с помощью функции FindFtSlot( ), т.е. определяется и резервируется новый слот.

После всех процедур определения информации для резервирования, соответствующая структура вносится в объект сообщения, с помощью функции AttachSignal( ), образно говоря, создается облик сигнала и результирующее сообщение отправляется на физический уровень. Указатель на сообщение АЗН-В очищается и производится проверка на ненулевое значение дополнительного указателя, затем происходит проверка на необходимость запроса информации с верхнего уровня и проверка результирующих значений полей в структуре слота после процедур отправки. На этом обработка собственного широковещательного резервирования для приложения АЗН-В в текущем слоте заканчивается, а далее рассматривается случай резервирования для сетевых приложений.

При обработке собственного резервирования для отправки данных приложения многоинтервальной передачи данных АЗН-В может возникнуть несколько случаев.

Т.к. резервирование для данного приложения по возможности размещается на несколько «суперфреймов», то может возникнуть ситуация, когда никаких сообщений в очереди нет, т.е. она пуста, поэтому следует освободить резервирование для других станций, воспользовавшись нулевым резервированием. Для этого используется обычное сообщение synchronization burst, таким образом, все процедуры схожи с предыдущим случаем, однако, поля периодического таймаута и смещения всегда устанавливаются в 0 и очищаются соответствующие поля структуры слота.

В случае если очередь непустая, то изначально слот также проверяется на предназначенность для размещения комбинированного периодического и инкрементированного резервирования в отправляемом в нём сообщении, т.к. для данного приложения слот также может быть зарезервирован более чем на 3 «суперфрейма». Соответственно, происходят те же процедуры определения значений полей резервирования. Основные отличия от процедур для приложения АЗН-В отличаются в том, что:

- для обработки используется очередь сообщений, из которой извлекается первое по порядку сообщение;
- принцип заполнения полей сообщения так же зависит от того, собственное это сообщение или ретранслируемое;
- запускается процесс обработки очереди сообщений, в случае, если требуется разместить дополнительное резервирование.

В случае обработки типа сообщения соответствующего концу процесса передачи, то на физический уровень отправляется запрос на переключение в режим приёма

Если же получен тип сообщения, что на физическом уровне приём закончился неудачей, то состояние слота устанавливается в значение «особенно занят», т.к. предполагается, что кроме возможной интерференции в одном слоте нескольких сигналов, в этом слоте также было принято сообщение с низким уровнем сигнала и через некоторое время может произойти сближение с приёмопередатчиком и приём станет возможным. Таким образом, существует некоторая неопределенность, и правила перерезервирования не могут быть применены.

Функция AttachSignal( ) используется при отправке сообщения на физический уровень. Согласно результирующей длине сообщения, которая определяется суммой битовой длины всех информационных единиц протоколов, содержащейся в этом сообщении и скорости передачи, рассчитывается длительность сообщения в секундах. Далее, для отправки на нижний уровень, к сообщению прикрепляется служебная информация в виде объекта характеризующего сигнал (функция CreateSingnal( )), создающая отображение сигнала по мощности и скорости передачи), при этом используются такие данные, как: текущий момент времени, длительность сообщения, мощность передатчика и скорость передачи.

Деструктор модуля MAC выполняет обычную для него задачу очистки динамически выделенных переменных и контейнеров их содержащих.

Также, в описании класса модуля MAC присутствуют несколько функций, которые позволяют обеспечить доступ к некоторым переменным и контейнерам класса:

- getBSstatus( ) – устанавливает является ли узел базовой станцией;
- getAddrCoord( ) – возвращает координаты узла по переданному в функцию адресу;
- getPECT_table( ) – предоставляет доступ к таблице соседних узлов;
- getNetwHandleMsgs( ) – предоставляет доступ к массиву частей сообщений связанных временной меткой;

- IsNeighbour( ) – устанавливает: является ли переданный в функцию адрес адресом соседнего узла.

По документу 9816 все транспондеры (приемопередатчики) должны иметь в качестве сетевого адреса уникальный 27-битный адрес ICAO, он является аналогом MAC адреса. Задача распределения адресов решается достаточно просто. В OMNeT++ набор узлов в моделируемой сети представляется в качестве массива, соответственно, каждый модуль, с помощью функций доступа к другим модулям, узнаёт свой номер в этом массиве.

Для записи статистики во всех модулях используется подход, комбинирующий механизм сигнализирования и объявления переменных статистики. Это позволяет разделить процесс генерирования результатов от процесса их записи, что обеспечивает большую гибкость для того что записывать и в какой форме. Все переменные статистики объявляются в файлах NED с пометкой @statistic, а сами модули излучают значения, используемые для записи в статистику с помощью сигналов. Инструментарий моделирования записывает данные в результирующие файлы с помощью добавления специальных слушателей для этих сигналов. Путём возможности выбора, какой тип слушателей добавлять, пользователь может контролировать то, что записывать в файлы с результатами и какие вычисления применять перед записью. Подход с использованием механизма сигнализирования позволяет рассчитывать агрегированные статистические данные, такие, например, как общее количество потерь пакетов в сети, а также позволяет реализовывать период «разогрева» сети (временной промежуток без записи статистики), без реализации со стороны программного кода модуля. Используя этот подход можно создавать отдельные модули, записывающие определенный тип статистических данных без затрагивания других модулей.

3.3.4 Подуровень LME

Функционал модуля, реализующего подуровень LME, описывается двумя файлами – LME.h и LME.cc. Соответственно, в заголовочном файле находится общее описание класса подуровня LME с объявлением всех атрибутов и функций-членов, а в файле источника находится непосредственная реализация работы функций-членов.

Если обратиться к заголовочному файлу, то можно увидеть, что модуль наследован от класса BaseLayer, т.к., например, нет нужды в таких атрибутах, как адрес узла или функциях отображения информационной единицы протокола для реализации объекта сигнала. Атрибутами класса являются:

- «Length» - длина PDU подуровня LME;

- «LMEPacket» - указатель на собственное исходящее сообщение, с которым на текущий момент работает подуровень;
- «StreamNumber» - номер потока слотов. По рабочим документам, подуровень LME должен создавать отдельный поток слотов для каждого приложения, которое обращается к этому уровню;
- «NetwHandleMsgs» - массив обрабатываемых частей сообщений, относящихся к подуровню LME, связанных временной меткой:
- «BroadcastFrequency» - частота вещания АЗН-В отчётов в минуту.

Функции-члены класса LME будут описаны далее в порядке реализации их в программном коде.

Конструктор класса задаёт нулевые значения или просто инициализирует его атрибуты, а также вызывает конструктор базового класса. Деструктор так же, как и во всех остальных модулях удаляет динамически созданные объекты и очищает массивы их содержащие.

В функции инициализации модуля initialize( ) из общего файла инициализации (обычно omnetpp.ini) считываются значение для атрибутов «Length» и «BroadcastFrequency», а также вызывается функция initialize( ) базового класса.

Обработки собственных сообщений в модуле LME нет, поэтому для отладочных целей функция handleSelfMsg( ) лишь выводит сообщение об ошибке.

Функция handleUpperMsg( ) для обработки PDU исходящих от верхних уровней во много схожа с такой же для модуля MAC. На данном подуровне не предусмотрена очередь сообщений, т.к. все очереди должны находиться на подуровне VSS, соответственно, обрабатываемые PDU инкапсулируются (с помощью функции encapsMsg( )) и отправляются нижележащему модулю. Для отладочных целей производится проверка на ненулевой указатель на обрабатываемое сообщение.

Функция инкапсуляции encapsMsg( ) принимает на вход указатель на тип данных cPacket, являющимся одним из базовых для библиотеки OMNeT++. Далее производится проверка на наличие какой-либо служебной информации, прикреплённой к обрабатываемому сообщению. Предполагается, что этой информацией является сетевой адрес, и он не должен иметь нулевого значения. Согласно значению полученного адреса, с помощью оператора ветвления производятся основные процедуры обработки сообщения.

В случае если адрес является широковещательным и узел является базовой станцией, то выносится решение, что инкапсулируются данные базовой станции для периодического вещания своего местоположения, обеспечивающие необходимую информацию для

маршрутизации. В информационных полях PDU подуровня LME заполняются некоторые типовые значения, а также местоположение узла, а к результирующей структуре добавляется служебная информация в виде адреса для модуля MAC.

В случае если адрес в служебной информации является адресом самого узла и узел не является базовой станцией, то принимается решение о том, что производится отправка собственных данных приложения многоинтервальной передачи данных АЗН-В. Все процедуры заполнения полей PDU идентичны предыдущему случаю, за исключением того, что тип PDU устанавливается в значение «сетевые данные мобильной станции», вместо «сетевые данные базовой станции».

В любом другом случае принимается решение о том, что согласно полученному из служебной информации адресу требуется найти в массиве обрабатываемых фрагментов сообщений, связанных временной меткой, соответствующий фрагмент. Таким образом, собирается сообщение для маршрутизации данных от другой мобильной станции, т.к. оно должно сохраняться по мере прохождения по сети.

В конечном итоге, для отладочных целей производится проверка на ненулевой указатель на сообщение типа LMEPkt, полученный функцией указатель отдается во владение результирующему объекту LMEPkt, который затем возвращается.

Также, в модуле присутствует функция decapsMsg( ), которая задействуется в случае обработки сетевых сообщений. Согласно идентификатору типа PDU LME из него декапсулируется (отменяется владение объектом) PDU верхнего уровня. В свою очередь, извлеченному PDU присваивается такая же временная метка, а в качестве сопровождающей контрольной информации присоединяется адрес источника, обозначенный в контрольной информации на нижнем уровне. Затем создается запись в таблице обрабатываемых фрагментов сообщений, которая представляется в виде пары «адрес-фрагмент». В конечном итоге, декапсулированный объект возвращается функцией. Таким образом, функция handleLowerMsg( ) для обработки сообщений от нижнего уровня, проверяет идентификатор PDU LME и если он соответствует одному из сетевого типа, то декапсилирует полученное сообщение с помощью функции decapsMsg( ) и отправляет его вышележащему модулю.

Функцией, частично реализующей приложение вещания данных АЗН-В, является функция handleLowerControl( ). Она обрабатывает одно сообщение, которое реализует механизм обмена служебной информацией между подуровнями модели – VSSLMECntrl (программный код сообщения также написан для модели). Непосредственно для обработки используется оператор ветвления, проверяющий тип информации.

Одним из рассматриваемых типов является окончание фазы настройки на подуровне MAC/VSS. Он означает, что приложение АЗН-В может быть запущено, а также, в принципе, могут быть запущены остальные приложения, требующие передачи данных, поэтому эта информация направляется далее, на верхние модули. Затем поля сообщения со служебной информацией заполняются необходимыми значениями для размещения резервирования:

- тип информации устанавливается в «старт АЗН-В»;
- тип резервирования устанавливается в «периодическое широковещательное»;
- частота вещания АЗН-В отчётов устанавливается согласно значению из файла инициализации;
- номеру потока присваивается новое значение;
- значение поля «конец потока» устанавливается в отрицательное значение;

Заполненное сообщение отправляется по служебной межмодульной связи вниз.

Следующим рассматриваемым типом является «запрос данных АЗН-В». Он сигнализирует о том, что на уровне VSS подходит момент начала слота с собственным резервированием для передачи данных АЗН-В. Согласно этому запросу заполняются поля PDU LME, в том числе данные о местоположении узла. Заполненное сообщение отправляется нижнему модулю, а указатель на текущее обрабатываемое сообщение очищается. На этом общее описание принципов работы модуля LME окончено.

3.3.5 Подуровень маршрутизации

Подуровень маршрутизации на данном этапе разработки осуществляет основную часть функционала для реализации приложения многоинтервальной передачи данных АЗН-В. Этот функционал обеспечивается классом VDLGRPfSAN, который наследован от класса BaseLayer по тем же соображениям, что и VDLLMELayer. Соответственно, программный код класса VDLGRPfSAN является основой функционала модуля GRPfSAN, представляющего собой подуровень маршрутизации и находящийся на вершине структурной схемы модели.

Класс имеет большое число различных атрибутов (заголовочный файл GRPfSAN.h). Атрибуты типа simsignal_t - Send, BSsend, Rcvd, PropagationTime, NumACs, соответственно для записи статистики по кол-ву отправленных сообщений мобильными станциями, по кол-ву отправленных сообщений базовыми станциями, по кол-ву принятых сообщений базовой станцией, по величине времени задержки и по кол-ву отслеженных бортов (приёмопередатчиков). SN – значение текущего номера последовательности для отправки в сообщении от базовой станции. BSPkt и MSPkt указатели на текущее обрабатываемое

сообщение от базовой или мобильной станции (зависит от статуса узла). IsBaseStation – индикатор является ли модуль базовой станцией. Указатели pMAC и pLME для доступа к модулям MAC и LME – требуются для обеспечения операции по удалению фрагментов сетевых сообщений связанных временной меткой. BS_TrafficGenPeriod – величина периода вещания сетевых сообщений базовой станцией в секундах. Собственные сообщения модуля типа cMessage BS_TrafficGen, PostInit и MS_TrafficGen. BSs – таблица базовых станций, запись в которой состоит из пары «адрес-координаты». Таблица узлов, которые могут быть выбраны в качестве маршрутизаторов – RoutingTable, а также её запись – RoutTableEntry, состоящая из адреса узла и массива пар «сообщение от базовой станции – целое число». Массивы таймеров для записей в таблице. SPEnd – метка окончания фазы настройки. RTentry_time – период хранения записи в таблице маршрутизаторов. BSavailable_time – период хранения записи в таблице базовых станций. NetBroadFreq и PerMinutes – комбинация переменных, означающая кол-во раз в минуту, которые устанавливаются в сообщении отправляемой базовой станцией для указания мобильным станциям частоты вещания АЗН-В отчётов для многоинтервальной передачи. RcvdAdrss – массив для индивидуальных полученных адресов базовой станцией. Реализация функций-членов будет описана далее в порядке размещения их файле источника кода – GRPfSAN.cc.

Конструктор объектов класса инициализирует атрибуты нулевыми значениями, т.к. конкретные значения будут получены в процессе инициализации модуля с помощью функции initialize( ). В свою очередь, деструктор удаляет динамически созданные атрибуты и очищает хранящие их контейнеры.

Функция initialize( ) так же, как и в остальных модулях производит присваивание значений атрибутам класса значениями, взятыми из файла инициализации. Изначально регистрируются сигналы для записи статистики, т.е. соотносятся с названиями статистик в NED-файле. Далее инициализируются величины периодов вещания сетевых сообщений базовой станцией и хранения записей в таблицах, предназначенных для маршрутизации. В конечном итоге, в куче инициализируются собственные сообщения, а сообщение, относящееся к событию постинициализации, вносится в набор будущих сообщений.

В функции handleSelfMsg( ) используется оператор ветвления «if-else» для обработки событий, представленных собственными сообщениями.

При возникновении события генерирования сетевого сообщения от базовой станции, соответствующего типу GRPfSAN_BSTrafGen, в куче создается новый объект типа GRPfSAN_R_Pkt (отдельный тип сообщения, написанный при разработке). Далее происходит заполнение следующих информационных полей:

- адрес базовой станции устанавливается в значение собственного адреса модуля, т.к. только базовая станция может генерировать данную информацию, а мобильные узлы лишь её ретранслируют;
- устанавливается новое значение уникальной последовательности;
- время жизни устанавливается в значение равное 7;
- метка тупика устанавливается в значение «ложь» (на данный момент не используется);
- поля значений частоты передачи АЗН-В отчётов по сети выставляются согласно значениям, взятым из файла инициализации;
- битовая длина устанавливается в 0, т.к. эта часть информации входит в информационное поле PDU LME и его величина учитывается и добавляется на подуровне LME.

Дополнительно к сообщению добавляется служебная информация в виде широковещательного адреса, которая используется на нижележащих модулях. Заполненное PDU отправляется вниз. Далее производится сигнализирование статистики об отправке сетевого сообщения базовой станцией, а собственное сообщение с типом события GRPfSAN_BSTrafGen снова запускается на указанное в атрибуте класса BS_TrafficGenPeriod время.

Некоторые действия по окончательной инициализации модуля маршрутизации можно произвести лишь по завершении инициализации других модулей, т.е. когда для всех простых модулей была вызвана функция initialize( ). Для этого и существует событие постинициализации модуля (GRPfSAN_PostInit). При его возникновении определяются указатели на другие модули для осуществления к ним доступа, и определяется статус узла в качестве базовой станции.

При окончании времени существования записи в таблице базовых станций возникает соответствующее событие, представленное собственным сообщением с типом GRPfSAN_BSavailability. Для каждого такого таймера в качестве служебной информации в имени сообщения указывается адрес базовой станции, к которой он относится. Следуя этой информации, выполняются процедуры поиска и очистки записей в таблице базовых станций и массиве таймеров для них.

При окончании времени существования записи в таблице узлов для маршрутизации возникает соответствующее событие, представленное собственным сообщением с типом GRPfSAN_RTtimer. Т.к. запись в таблице включает в себя несколько адресов: один непосредственно, как переменная в записи, а другой в указателе на сообщение от базовой

станции, то таймер также сопровождается двумя адресами в качестве служебной информации. Адрес потенциального узла-маршрутизатора находится в имени собственного сообщения, а адрес базовой станции прикрепляется в виде служебной информации, извлечение которой и происходит в начале обработки события. По этой же причине процедуры поиска соответствующей записи в таблице маршрутизаторов и массиве таймеров производится по двум адресам. При этом если после удаления сообщения от базовой станции массив пар «сообщение от базовой станции – целое число» для обозначенного адреса маршрутизатора оказывается пустым, то запись также удаляется и из таблицы.

Если узел не является базовой станцией, то в функции handleSelfMsg( ) появляется возможность возникновения события отправки АЗН-В отчёта, т.е. создания необходимой информации для его маршрутизации. Вначале производится проверка условия того, что таблица базовых станций и таблица маршрутизаторов не являются пустыми, т.к. для записей существуют таймеры хранения и к этому моменту они могут быть удалены. При отправке данных необходимо установить, какая из базовых станций является ближайшей к узлу, поэтому производятся процедуры её поиска. Поиск осуществляется по таблице соседей, по каждой её записи находится сообщение от базовой станции. Если в другой записи будет найдено такое же сообщение от БС, но с более «новым» значением индивидуальной последовательности, то текущее рассматриваемое сообщение будет заменено, в противном случае останется прежним, т.к. они полностью идентичны. При нахождении отличного сообщения будет произведена проверка уже в таблице базовых станций на предмет того, какая из них ближе и соответствующее значение указателя на сообщение в поиске либо будет заменено, либо останется прежним. В итоге, по окончании поиска будем иметь ближайшую базовую станцию. Эти процедуры нужны для «синхронизации» записей в таблице маршрутизаторов с записями в таблице базовых станций, чтобы не оказалось так, что ближайшая базовая станция найдена, а маршрутизаторов для ретрансляции к ней данных нет – в сценариях с плохой связностью такая ситуация вполне вероятна.

После определения ближайшей базовой станции, согласно используемому жадному алгоритму маршрутизации, должен быть найден ближайший к этой базовой станции маршрутизатор. Изначально создаётся массив, в который добавляются все подходящие для маршрутизации к выбранной базовой станции узлы: проверяется не только наличие базовой станции в массиве, относящемся к определенному адресу узла-маршрутизатора, но и наличие самого этого узла в таблице соседних узлов, для чего используется доступ к модулю MAC, т.к. может сложиться ситуация, когда узел уже вышел из зоны радиовидимости в момент его выбора. В качестве информационной единицы протокола подуровня маршрутизации,

представляющей данные от мобильной станции, используется класс из библиотеки OMNeT++ - cPacket, являющийся базовым. В поле адресата устанавливается адрес выбранной базовой станции, а в поле источника – выбранный маршрутизатор. Именно поле источника будет каждый раз изменяться при попадании на новый маршрутизатор. В качестве служебной информации для нижележащих модулей, прикрепляемой к заполненному сообщению, обозначается собственный адрес узла. Полученный объект cPacket (или PDU) отправляется вниз, новое событие отправки данных АЗН-В по сети назначается согласно значениям, указанным в сообщении от выбранной базовой станции, и в конечном итоге, производится сигнализирование для статистики по отправленным сообщениям от мобильной станции.

Как и в остальных модулях модели, функции handleLowerMsg( ) используется для обработки сообщений с нижележащих уровней, а точнее фрагментов сетевых сообщений, связанных временной меткой, которые также будут являться информационными единицами протокола данного подуровня. Для обработки принятой модулем единицы, используются следующая служебная информация: адрес, прикрепленный к сообщению нижним уровнем и временная метка. Используя эту информацию, с помощью доступа к массиву обрабатываемых фрагментов сетевых сообщений модуля подуровня LME, определяется тип сообщения для дальнейшей обработки. Этот тип будет сетевым сообщением либо от базовой станции, либо от мобильной.

При обработке сообщения от базовой станции в первую очередь проверяется, не является ли сам узел базовой станцией. В положительном случае сообщение игнорируется, т.к. это либо сообщение, посланное этим узлом, либо сообщение от другой базовой станции, которая не несёт полезной информации для маршрутизации или статистики. Здесь лишь производится процедура удаления фрагментов комплексного сообщения на всех подуровнях с помощью доступа к ним.

Если адрес, обозначенный в служебной информации, равен адресу, хранящемуся в обрабатываемом сообщении, то это значит, что было принято сообщение от базовой станции, находящейся в зоне радиовидимости узла. Изначально производится проверка записи о базовой станции в таблице базовых станций. В зависимости от результата, вносится новая запись или обновляется старая, при этом схожие процедуры выполняются и для таймера этой записи – производится поиск в массиве таймеров и если запись отсутствует, то создается и запускается новый таймер, иначе таймер перезапускается. Также происходит проверка наличия записи в таблице маршрутизаторов, при её отсутствии выносится решение о необходимости ретрансляции этого сообщения, при этом также проверяется условие фазы

настройки узла и ненулевое значение времени жизни пакеты. Если данные условия не выполнены, то производятся процедуры очистки фрагментов сообщения на всех подуровнях.

Следуя далее по оператору ветвления, возникает лишь одна ситуация, когда принято сообщение от базовой станции ретранслируемое мобильной станцией, тогда, в первую очередь, нужно проверить, находится ли узел в таблице соседей, т.е. было ли от него получено хотя бы одно АЗН-В-сообщение. Эта проверка необходима, т.к. при отсутствии данных о местоположении узла к нему не могут быть применены расчеты для внесения записи в таблицу маршрутизации. Снова стоит отметить, что данная ситуация вероятна при плохой связности сети – узел на короткое время возникает в зоне радиовидимости, а затем выходит из неё, соответственно выбора таких узлов в качестве маршрутизаторов стоит избегать.

Итак, если узел не находится в таблице соседей, то осуществляются процедуры удаления фрагментов сообщения на всех подуровнях. В ином случае, снова происходит обращение к таблице соседних узлов, но на этот раз проверяется наличие базовой станции в зоне радиовидимости узла. В положительном случае так же производится очистка фрагментов сообщения, т.к. оно не несёт полезной информации – узлу не требуется маршрутизация данных АЗН-В на данный момент. Если же в зоне радиовидимости нет базовых станций, то принятое сообщение может быть использовано узлом для создания и обновления таблиц для маршрутизации. В первую очередь, проверяются записи массива базовых станций и либо добавляется новая, либо обновляется существующая запись. Для заполнения записи используется доступ к подуровню LME, где обрабатываемые фрагменты сообщения хранятся в массиве в виде пары «адрес-PDU LME». Соответственно, используются координаты, хранящиеся в найденном по адресу из служебной информации PDU LME для заполнения записи, а в переменную адреса записи таблицы базовых станций помещается адрес, указанный в обрабатываемом сообщении. Если до этого момента таблица базовых станций была пуста и при этом фаза настройки узла окончена, то запускается собственное сообщение для начала процедур отправки и АЗН-В отчётов по сети. Также, процедуры обновления используются для массива таймеров для записей в таблице базовых станций.

В качестве следующего шага производится проверка на наличие сообщения базовой станции с таким же индивидуальным номером последовательности в таблице маршрутизаторов. Если оно отсутствует, то должно быть ретранслировано узлом, поэтому далее производится декрементирование поля времени жизни сообщения и если в результате его значение не равно нулю, то сообщение с прикрепленной служебной информацией в виде

адреса направляется нижележащему модулю. Стоит отметить, что адрес в служебной информации, отправляемой нижнему модулю, идентичен адресу в служебной информации с него пришедшему. Он будет использован на подуровне LME и MAC для сборки сообщения воедино, однако, на подуровне MAC адрес источника будет заменён на собственный адрес узла.

Далее начинается процесс заполнения таблицы маршрутизаторов. Если в таблице маршрутизаторов уже существует запись, соответствующая адресу в служебной информации с нижнего модуля, а также запись с сообщением идентичным полученному сообщению от базовой станции, то фрагменты сообщений очищаются на всех подуровнях, т.к. это сообщение уже было получено. Если такой записи нет, то производится поиск аналогичного сообщения от базовой станции ретранслированного другими узлами, при этом происходит проверка на тот факт, что расстояние от узла до базовой станции меньше расстояния от маршрутизатора до базовой станции – это нужно для работы жадного метода. В положительном случае существующая запись обновляется, а таймер с ней связанный перезапускается. В противном случае все фрагменты сообщения удаляются на всех подуровнях и связанный с ним таймер также. Если в таблице маршрутизаторов записи с такой базовой станцией не существует, то создается новая, а её таймер перезапускается.

Если в таблице маршрутизаторов отсутствует запись, соответствующая адресу в служебной информации с нижнего модуля, то производится проверка на расстояние до базовой станции от самого узла и от узла ретранслятора. Если узел отправитель ближе к базовой станции, то в таблице маршрутизаторов создается новая запись и запускается соответствующий таймер, в противном случае фрагменты сообщения удаляются.

При обработке сообщения от мобильной станции, с помощью доступа к модулю MAC, устанавливается источник целого сообщения. Далее, в первую очередь проверяется, не является ли сам узел базовой станцией. В данном случае, если адреса, указанные в этой информационной единице протокола равны между собой и равны адресу узла-получателя, то выносится решение, что базовой станцией принято сетевое сообщение АЗН-В. В процедуре производится вычисление общего времени прохождения сообщения от маршрутизатора к маршрутизатору с учётом пребывания в очереди, а затем полученное значение записывается в статистику. Также, производится запись статистики полученных сообщений от мобильной станции и номера борта, если сообщение от него ещё не было получено. Далее фрагменты сообщения на всех подуровнях очищаются, что происходит и в случае, если условие равенства адресов не выполняется (этого случая не должно возникать, но для отладочных целей он рассматривается).

В случае если адрес ретранслятора в полученном PDU равен адресу самого узла и при этом он не является базовой станцией, то это означает, что узел был выбран в качестве маршрутизатора для сообщения. Изначально производится проверка по таблице соседей подуровня MAC, что в зоне радиовидимости уже находится нужная базовая станция. В положительном случае, адрес маршрутизатора в PDU устанавливается в адрес назначения и данные направляются на нижележащий уровень вместе со служебной информацией в виде того же адреса, что и в принятой вместе с PDU служебной информацией (для восстановления целого сообщения на всех подуровнях). Если базовой станции нет среди соседних узлов, то для выбора следующего маршрутизатора для сообщения подуровень использует схожие процедуры, что и при отправке собственного сетевого сообщения АЗН-В (функция handleSelfMsg( ), событие GRPfSAN_MSTrafGen).

Во всех иных случаях узел принял сообщение, предназначающееся для другого маршрутизатора – тогда запускаются процедуры удаления фрагментов сообщений на всех подуровнях.

Следующей основной функцией в модуле является функция handleLowerControl( ), обрабатывающая служебные сообщения, приходящие от нижележащих уровней. Функция обрабатывает наступление события окончания фазы настройки узла, о чём и сигнализируют нижние уровни. В случае если узел является базовой станцией, то модуль отправляет собственное сообщение о необходимости начала вещания сетевых сообщений. Если же узел является мобильной станцией, то изначально проверяется наличие записей в таблице базовых станций и при их наличии запускается собственное сообщение о начале передачи данных АЗН-В для маршрутизации.

Функция FindDistance( ) всего лишь определяет расстояние по переданным ей координатам.

Функции handleUpperControl( ) и handleUpperMsg( ) для обработки сообщений от верхних уровней не задействованы, т.к. на данный момент модуль подуровня маршрутизации является самым верхним. Программный код некоторых функций приведен в Приложении А.

Разработанная модель подробно учитывает особенности функциональной модели VDL Mode 4, а также такие факторы, как: мобильность узлов сети, условия распространения волн, физический уровень, канальный уровень с несколькими подуровнями и взаимодействие между ними, что позволяет проводить исследования рассматриваемой сети с высокой степенью достоверности. Также предусмотрена возможность варьирования параметров модулей доступа к среде и маршрутизации. Модель позволяет получать

следующие числовые показатели производительности сети: количество отправленных и полученных сообщений, задержки при передаче сообщений по сети, а также число узлов, от которых были получены сетевые сообщения с данными о местоположении и намерениях.

3.4 Выводы

1. В разделе представлено комплексное описание разработанной дискретно-временной модели мобильной самоорганизующейся сети, работающей на основе стандарта VDL Mode 4 и среды моделирования. При разработке модели была выбрана модульная структура – каждый сетевой узел имеет модуль мобильности, модуль контроля сетевого интерфейса (с модулями физического и канального уровня), модуль подуровня LME и модуль подуровня маршрутизации. Использованный подход позволяет в большинстве случаев менять программный код любого модуля, не затрагивая при этом другие.

2. Так как стандарт VDL Mode 4 носит описательный характер и не предлагает конкретных реализаций, то в рамках модели реализованы и протестированы алгоритмы резервирования слотов, алгоритмы внутреннего и внешнего взаимодействия подуровней стандарта VDL Mode 4 и алгоритмы обработки информационных сообщений. Программная среда моделирования OMNeT++ даёт возможность создавать модели с большой глубиной детализации, поэтому можно утверждать, что разработанные алгоритмы, с незначительными изменениями могут быть применены в реальном приёмопередающем оборудовании.

3. В модели предусмотрена возможность варьирования параметров модулей доступа к среде и маршрутизации, при этом суммарное кол-во входных параметров модели составляет около 50 шт., что свидетельствует о широких возможностях для исследования модели приёмопередатчиков стандарта VDL Mode 4, а также построенной между ними воздушной мобильной самоорганизующейся сети.

РАЗДЕЛ 4. РАЗРАБОТКА ПРОТОКОЛА МАРШРУТИЗАЦИИ ДЛЯ УЗЛОВ САМООРГАНИЗУЮЩЕЙСЯ МОБИЛЬНОЙ СЕТИ, ФУНКЦИОНИРУЮЩЕЙ НА БАЗЕ СТАНДАРТА VDL MODE 4. ЧИСЛЕННЫЕ ПАРАМЕТРЫ, ВЛИЯЮЩИЕ НА ПРОИЗВОДИТЕЛЬНОСТЬ СЕТИ

4.1 Анализ возможных подходов по созданию «образа» сети на каждом узле

Первоочередная задача, которую нужно решить при разработке протокола маршрутизации – это обеспечение получения информации об участниках сети другими узлами. К достоинству проактивной маршрутизации относится её возможность более надежной (относительно реактивной маршрутизации) поддержки установленных маршрутов, т.к. либо все узлы, либо те узлы, которые входят в «зону» маршрутизации, получают необходимую информацию о маршрутах. Однако, чем выше интенсивность изменения топологии сети, тем чаще приходится обновлять информацию в таблице маршрутизации и информацию о соседних узлах. Иначе, велика вероятность возникновения большого числа отказов при маршрутизации, а значит и потери пакетов. В поддержку данного подхода может выступать тот факт, что обычно самолеты и вертолеты гражданских авиалиний движутся по строго установленным маршрутам, проложенным заранее. При этом дальность радиосвязи достаточно велика, но и расстояния между узлами могут быть значительными. Функционал VDL Mode 4 отчасти поддерживает такой подход, т.к. ВС производят широковещательную рассылку о своих полетных данных и намерениях. Эта информация может быть занесена в отдельную таблицу, представляющую внутреннюю область памяти в транспондере, либо вынесенную внешне, к которой транспондер будет обращаться. Затем информация о соседних узлах может быть инкапсулирована с помощью заголовков в передаваемом пакете, таким образом, ВС за пределами радиовидимости будут получать информацию об узлах, являющихся соседними другим узлам. Т.к. рассылка ведется широковещательно, на кол-во ретрансляций может быть наложено ограничение, а также подразумевается удалять одинаковые пакеты, приходящие в разное время.

Итак, mesh-сеть является некоторым обобщением различных сетевых структур, т.е. структура может быть совершенно различной: звезда, кольцо, объединение колец, ячеистая топология, каждый с каждым. Главным отличием mesh-сетей от других информационных сетей является использование в принципе взаимодействия узлов специальных алгоритмов обмена данными. Выше приводились различные алгоритмы маршрутизации, используемые для мобильных Ad Hoc сетей, однако, не конкретизировались способы обеспечения

информации об участниках сети. Для создания «образа» сети (информации об участниках сети) предлагается алгоритм, основывающийся на проактивном принципе, т.е. использующий таблицы маршрутизации.

В рассматриваемом случае был использован алгоритм, блок-схема которого представлена на рисунке 24. Каждый летательный аппарат имеет свой личный идентификационный номер (ИН), а так же содержит массив со своими координатами. Работа алгоритма состоит из следующих этапов:

1) Каждый ВС совершает широковещательную рассылку своих координат и ИН;
2) Все ВС получают эти сообщения от тех узлов, которые находятся в пределах устойчивой радиосвязи и заносят их в свои таблицы;
3) Каждый ВС вновь совершает широковещательную рассылку, но в сообщениях уже содержится записанная ими таблица видимости;
4) Все ВС получают таблицы видимости от тех узлов, которые находятся в пределах устойчивой радиосвязи и дополняют свои таблицы видимости.

Таким образом, каждый ВС в процессе работы данного алгоритма будет содержать и обновлять две таблицы: таблицу соседей (узлы в зоне устойчивой радиовидимости) и полную таблицу видимости (таблицу, где отражены ВС и наземные станции доступные через ретрансляцию). Если требуется начать сеанс связи между бортами, каждый ВС может использовать две полученные таблицы и инициализировать один из существующих алгоритмов нахождения кратчайшего пути (например, алгоритм Дейкстры) для определения маршрута и узлов ретрансляторов.

Опираясь на выработанный алгоритм, была создана имитационная модель создания образа сети на языке программирования Fortran 90. Входными параметрами модели являются: радиус радиовидимости каждого узла (одинаковая), координаты каждого узла на плоскости (координатная сетка размером 100x100), личный идентификатор каждого ВС. Выходными результатами является таблица видимости и таблица соседних узлов. Рассмотрены пять узлов, положение которых генерируется случайным образом.

Перейдем к исследованию параметров сети полученной с помощью компьютерной модели. Для начала обозначим входные параметры для получения результатов работы модели: координатная сетка размером 100x100, радиус устойчивой радиосвязи – 50, личный идентификатор каждого ВС и координаты представлены в таблице 5.

Выходными результатами работы программы являются две таблицы: таблица соседей (таблица 6) и таблица видимости (таблица 7).

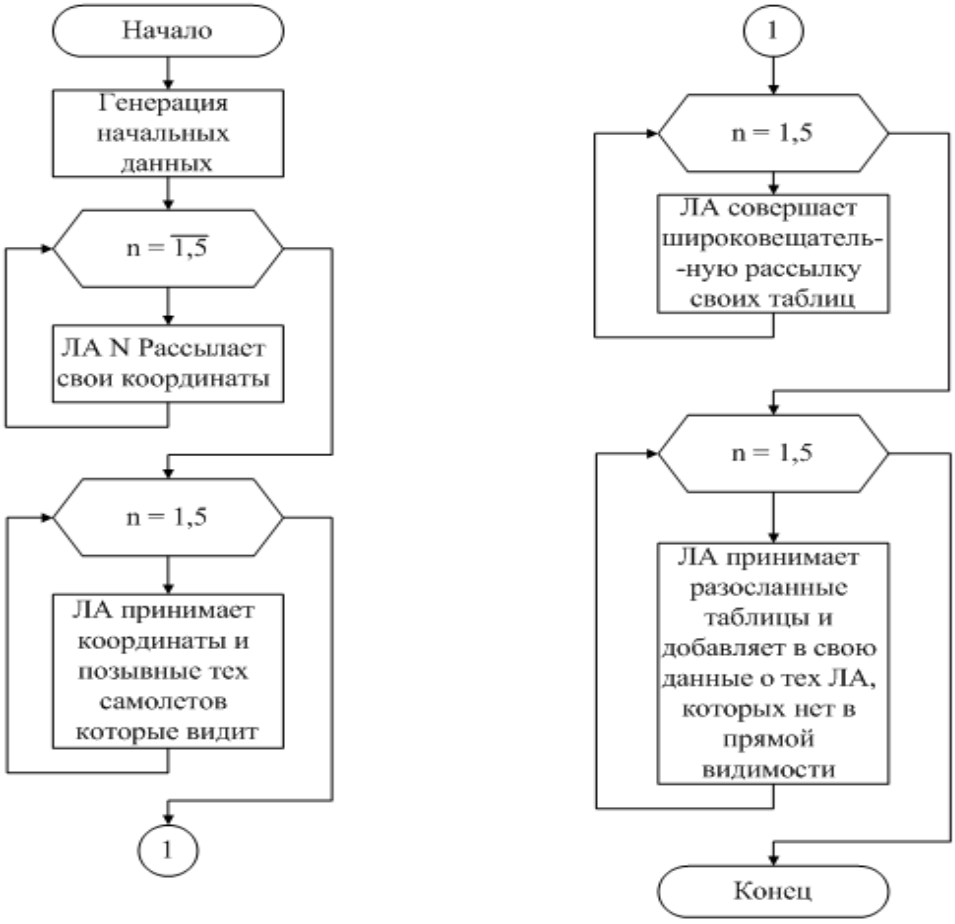


Рисунок 24. Блок-схема алгоритма создания «образа» сети

Таблица 5. Список координат и ИН каждого ВС

Личный ИН ВС	X	Y
first	2	76
second	25	72
third	10	18
fourth	49	84
fifth	15	42

Таблица 6. Полученные таблицы узлов находящихся в зоне устойчивого радиоприема каждого узла (соседних узлов)

First			
Соседний узел	X	Y	Расстояние
second	25	72	23.35
fourth	49	84	47.68
fifth	15	42	36.4

Продолжение Таблицы 6

Second			
Соседний узел	X	Y	Расстояние
first	2	76	23.35
fourth	49	84	26.83
fifth	15	42	31.62
Third			
Соседний узел	X	Y	Расстояние
fifth	15	42	24.52
Fourth			
Соседний узел	X	Y	Расстояние
first	2	76	47.68
second	25	72	26.83
fifth			
Соседний узел	X	Y	Расстояние
first	2	76	36.4
second	25	72	31.62
third	10	18	24.52

Таблица 7. Таблица видимости каждого узла

First	
ИН узла	Расстояние
Second	23.35
Fourth	47.68
Fifth	36.4
Third	58.55
Second	
ИН узла	Расстояние
First	23.35
Fourth	26.83
Fifth	31.62
Third	56.04

Продолжение Таблицы 7

Third	
ИН узла	Расстояние
fifth	24.52
first	58.55
second	56.04
fourth	76.66
fourth	
ИН узла	Расстояние
first	47.67
second	26.83
fifth	54.04
third	76.66
fifth	
ИН узла	Расстояние
first	36.4
second	31.62
third	24.51
fourth	54.03

По результатам выполнения программы можно сделать вывод о том, что при работе предложенного алгоритма все узлы в сети получают информацию о других узлах, т.е. в таблицы заносятся те узлы, которые находятся вне зоны устойчивого радиоприема другими узлами. Расстояние определено по двум полученным координатам. Совершенно тот же принцип работает и для трехмерных координат. Информация о расстоянии, скорости и курсе может служить в дальнейшем для осуществления маршрутизации для пакетов с данными АЗН-В.

Однако данный подход имеет один большой недостаток. Размер передаваемого сетевого сообщения будет расти прямо пропорционально количеству пройденных интервалов совместно с количеством соседних узлов добавляемых на этих интервалах. Это приведет к тому, что в сетях уже с несколькими десятками узлов результирующий размер может увеличиться до нескольких сотен слотов (даже более), при этом ценность данных о местоположении и намерении узлов, находящихся за сотни и тысячи километров друг от друга, для них весьма сомнительна. Цель работы обеспечить передачу данных АЗН-В до

наземных систем УВД, поэтому для обеспечения узлов сети необходимой информацией для маршрутизации может быть использован иной подход, который рассматривается в следующем пункте.

4.2 Выбор алгоритма маршрутизации

Работа стандарта VDL Mode 4 уже подразумевает обмен узлами структурированной информацией, которая может быть использована для маршрутизации. Этой информацией является таблица соседних узлов, которая содержится и обновляется на каждом приёмопередатчике. Запись содержит 27-битный адрес ICAO и информацию о местоположении каждого соседнего узла - эти данные берутся из периодических сообщений вещаемых участниками движения. Узлам остаётся лишь узнать местоположение адресата, которыми являются базовые станции. Сообщения с местоположением базовой станции (БС), которые также вещаются ими периодически, могут быть ретранслированы узлами по сети для создания записей в таблицах маршрутизации. Данный метод построения таблиц маршрутизации является проактивным и нацелен на уменьшение количества служебных сообщений, передаваемых в сети. Минимизация количества передаваемой информации была выбрана в качестве основной цели при разработке протокола маршрутизации, т.к. 1 канал VDL Mode 4 обладает относительно небольшой пропускной способностью, большая часть которой обязательно должна быть выделена под сообщения АЗН-В.

Согласно рассмотренным особенностям работы сети, в качестве алгоритма маршрутизации был выбран «жадный» алгоритм (greedy algorithm). Каждый узел передаёт сообщение одному из соседних узлов - наиболее подходящему с локальной точки зрения (локально оптимальный выбор), таким образом, используется таблица соседних узлов, создаваемая на каждом узле, согласно принципам работы VDL Mode 4. Локально оптимальный выбор совершается согласно метрике «Most Forward Radius» (MFR), выбранной для разработанного протокола, также в угоду минимизации передаваемой информации. При использовании данной метрики в качестве маршрутизатора в зоне прямой видимости выбирается узел, который находится ближе всех к адресату, с целью достичь минимального числа интервалов между узлами, которое должно пройти сообщение.

Географической маршрутизацией называют методы маршрутизации, основанные на местоположении; они получили большое внимание с тех пор, как были впервые предложены в 1980х. По сравнению с методами, основывающимися на топологии, географические методы имеют 2 уникальных преимущества для крупномасштабных беспроводных сетей с

высокой динамикой узлов, имеющих ограничения по электропитанию (автономные узлы). Во-первых, в географической маршрутизации, узел передаёт сообщение, основываясь только на местоположении узла назначения и узлов, находящихся в зоне прямой радиовидимости. Тем самым минимизируется размер таблицы маршрутизации и часть нагрузки информационного трафика, состоящая из служебных сообщений. Во-вторых, время сходимости, т.е. период времени, когда узел получает информацию для маршрутизации, также может быть меньше относительно «топологических» методов. Всё это представляется весьма привлекательным в качестве применения в беспроводных сетях с высокой динамикой узлов, т.к. разработка «топологического» алгоритма маршрутизации с быстрой сходимостью для рассматриваемых условий является сложной задачей. Рассмотренные выше преимущества в паре с развитием ГНСС и локальных механизмов самоконфигурации способствуют успешной применимости географической маршрутизации для множества беспроводных сетей.

Обычно, географические протоколы маршрутизации состоят из двух основных частей: «жадная» маршрутизации и маршрутизация по грани. «Жадный» алгоритм пытается доставить сообщение как можно ближе к месту назначения, на каждом шаге используя лишь локальную информацию. Таким образом, каждый узел передаёт сообщение соседнему, что представляется для него самым подходящим с локальной точки зрения (локально оптимальный выбор). Алгоритм маршрутизации по грани помогает, когда сообщение попало в «тупик», т.е. в том месте, где «жадный» алгоритм не даёт решения и не может найти подходящего соседнего узла для передачи. Несколько методов может быть использовано, чтобы определить наиболее подходящий соседний узел для «жадной» маршрутизации. Эти методы основываются на «жадных» метриках. Кроме локальной информации, в настоящее время, множество параметров, влияющих на производительность маршрутизации, также принимаются во внимание при разработке метрик для маршрутизации, вдобавок к жадным метрикам. Более сложные метрики могут значительно повысить производительность беспроводных сетей. Как бы то ни было, выбор таких метрик маршрутизации не может производиться в произвольном порядке из-за потенциальной несовместимости с «жадной» маршрутизацией, что может привести к снижению производительности сети и возникновению петель.

Итак, «жадный» алгоритм маршрутизации – это алгоритм, совершающий локально оптимальный выбор на каждом шаге с целью достичь глобального оптимума. Для географической маршрутизации «жадный» алгоритм выбирает самую легкую исходящую связь на каждом ретранслирующем узле, а вес каждой связи определяется выбранной

метрикой маршрутизации, которая принимает во внимание информацию о местоположении и другие различные факторы, влияющие на производительность. Т.к. важно установить факторы, влияющие на производительность в жадной маршрутизации, то совместимость между «жадным» алгоритмом и метриками маршрутизации должна быть хорошо проанализирована, чтобы избежать потенциальных проблем. В качестве двух базовых аспектов совместимости при анализе выбирается отсутствие петель и согласованность.

Отсутствие петель (зацикливаний) подразумевает, что протокол маршрутизации никогда не должен создавать петель при передаче сообщений в любой сетевой топологии. Это одно из самых важных условий для алгоритма маршрутизации, которое накладывает минимальные ограничения на его работоспособность. Очевидно, что подобное свойство протокола (создавать петли) не несёт ничего полезного для любой сети.

Согласованность, как условие, подразумевает собой требование того, что решения о маршрутизации сообщений, совершаемые всеми узлами по пути передачи, должны иметь некоторое соответствие. Например, если узел 1 решает передавать свой трафик узлу n по пути p(1, n) = <1, 2, …, n>, то и остальные узлы на пути p должны совершать соответствующее решение: узел 2 узлу 3, узел 3 узлу 4 и т.д. Согласованность гарантирует доступность знания и легкость контроля путей доставки сообщений, что очень важно для управления сетью и её безопасностью.

Совместимость между «жадным» алгоритмом в маршрутизации и метриками маршрутизации зависит от того, каким образом в этой системе взаимосвязей используется географическая информация. Существует два принципиально разных подхода в географической маршрутизации [98, 99].

Первый подход заключается в том, чтобы рассматривать географическую маршрутизацию в качестве основного метода, работающего таким образом, что для каждого пребывающего сообщения, передающий узел выбирает один из соседних узлов локально оптимальным способом и соответственно пересылает сообщение ему. Преимущество такого подхода в том, что узлам не нужно хранить какие-либо состояния топологии сети, кроме местоположения узла назначения. С другой стороны, этот метод требует много расчетов приходящихся на каждое сообщение, т.е. для каждого приходящего сообщения узел должен рассчитать наиболее подходящий узел-маршрутизатор, исходя из информации, содержащейся в передаваемом сообщении. Что, соответственно, может выразиться в необходимости в лишней (служебной) информации, помещаемой в каждое сообщение. Нагрузка на вычислительные ресурсы потенциально может замедлить процесс маршрутизации сообщений, а также увеличить расход батареи, если узел является

автономным. Как было отмечено, недостаток в отсутствии информации на узлах в сети о сетевой топологии выливается в необходимость передавать в каждом сообщении дополнительную информацию для выбора следующего маршрутизатора на пути передачи, включающую в себя местоположения узла отправителя и его адресата, а также, возможно, иную, влияющую на производительность сети, информацию. При этом если учесть тот факт, что в динамических беспроводных сетях размер сообщения выбирается небольшим относительно сетей с фиксированной топологией (для того, чтобы избежать проблем с нестабильными соединениями), то соответствующее увеличение размера сообщения может оказать весьма негативное влияние на производительность сети. В конечном итоге, т.к. следующий узел для передачи сообщения выбирается каждым узлом-маршрутизатором в условиях постоянно меняющейся топологии, но при этом никакой информации об этих изменениях не хранится на узлах, то возникают сложности в отслеживании путей, которые проходят сообщения, несущие в себе единый поток информации, что негативно может сказаться на безопасности сети.

Во втором подходе, географическая маршрутизация может быть использована в качестве механизма поиска маршрутов по запросу (проактивный подход), а также может быть совмещена с традиционными механизмами маршрутизации. Две широко используемых схемы маршрутизации могут быть совмещены с географической маршрутизацией. Первая – это маршрутизация «от источника», в которой узел-источник записывает пути, найденные географической маршрутизацией, и соответственно, включает эту информацию в служебную часть своих сообщений. Затем, промежуточный узел маршрутизирует сообщения согласно этой служебной информации. Вторая схема – это маршрутизация от интервала к интервалу, где информация о следующем интервале передачи устанавливается по пути, открытому географической маршрутизацией. Узел-источник лишь включает в сообщения в качестве информации для маршрутизации адрес назначения, а промежуточные узлы маршрутизируют эти сообщения, основываясь на их собственных знаниях о следующем узле для передачи. Чем проще (короче) найденный маршрут от узла источника для первой схемы и меньше число кандидатов для выбора в качестве маршрутизатора для второй схемы, тем больше преимущество выражаемое в уменьшении затрат на вычисление маршрутов и служебную нагрузку в сообщениях. Для многих комбинаций методов нахождения путей и методов маршрутизации сообщений, будь то «флуд», алгоритм Дейкстра или Белмана-Форда отмечается необходимость в сложном компромиссе между количеством информации для поддержания актуальности связей в сети и объемами вычислений для выбора маршрутизаторов.

Представим модель беспроводной сети в качестве связного направленного графа G(V,E) с количеством элементов |V| и |E|. V – это множество вершин, каждая из которых представляет узел в сети. Каждая связь в сети представлена направленным ребром между вершинами – множество E.

Если узлы u и v могут связываться напрямую, т.е. один находится в зоне радиовидимости другого, то существуют две встречные связи между узлами u и v. Связь $\overrightarrow{uv}$ относится к передаче сообщений от u к v, а связь $\overrightarrow{vu}$ от узла v к u. Соответственно связь $\overrightarrow{uv}$ называется исходящей для узла u, а $\overrightarrow{vu}$ входящей. В модели рассматриваются двунаправленные связи, т.е. подразумевается, что существует ребро $\overrightarrow{uv}$ между вершинами u и v, если существует ребро $\overrightarrow{vu}$ между вершинами v и u.

Вершина v называется соседней вершине u, если существует ребро $\overrightarrow{uv}$ между вершинами u и v. Набор N(u) вершины u – это набор её соседних вершин, который может быть определён, как:

$$N(u) = \{v|\overrightarrow{uv} \in E\} \qquad (4.2.1)$$

Путь от вершины-источника u_1 до назначения u_n может быть представлен в качестве последовательности вершин u_1, $u_2 \ldots u_n$, где существуют направленное ребро между каждой вершиной u_k и u_{k+1} для $1 \leq k \leq n\text{-}1$. Путь является простым, если вершины в нём не совпадают. Циклом является путь, в котором не совпадают все вершины, кроме начальной и конечной.

Алгебраическая система, основанная на связном направленном графе, имеет четыре кортежа $\langle L, F, w, \preceq \rangle$, где:

- L: множество сигнатур;
- F: множество потоков трафика;
- w: весовая функция связи;
- $\preceq$: порядок отношения.

Сигнатура $l_{\overrightarrow{uv}}$ множества L описывает характеристики связи $\overrightarrow{uv}$, такие как: местоположение узлов *u* и *v,* потери сообщений, энергопотребление на бит передачи, частоту канала и т.п. Каждая связь в рассматриваемой модели помечена сигнатурой. Иными словами, множество L описывает все рассматриваемые характеристики связи в сети.

Поток $f_{\langle s,d \rangle}$ множества F представляет собой информационный трафик, который должен быть доставлен из *s* в *d*. Мощность множества $F \leq |V| \times (|V-1|)$.

w(.) – это функция, рассчитывающая вес связи для различных потоков. *w*(.) функция двух переменных, соответственно одна из них является сигнатурой, а другая потоком.

Например, для потока $f_{\langle s,d\rangle}$, вес исходящего соединения $\overrightarrow{uv}$ узла u это $w(l_{\overrightarrow{uv}}, f_{\langle s,d\rangle})$, где $l_{\overrightarrow{uv}} \in L$, а $f_{\langle s,d\rangle} \in F$.

$\preceq$ используется для сравнения всех исходящих связей для передающего узла, основываясь на функции $w(.)$, таким образом, чтобы он мог передавать сообщения по связи с наименьшим весом. Если $w\left(l_{\overrightarrow{uv}}, f_{\langle s,d\rangle}\right) \preceq w\left(l_{\overrightarrow{ux}}, f_{\langle s,d\rangle}\right)$, то связь $\overrightarrow{uv}$ не хуже (легче или равна) связи $\overrightarrow{ux}$ для потока $f_{\langle s,d\rangle}$.

Стоит отметить, что для «жадного» алгоритма не все исходящие связи могут быть рассмотрены в качестве кандидатов для передачи сообщения – могут быть рассмотрены лишь те связи, чей вес строго меньше, чем ϕ, где ϕ порог для принятия решения «жадным» алгоритмом.

Используя рассмотренную алгебраическую систему, можно исследовать применимость «жадной» маршрутизации, основываясь на следующих свойствах:

Мьютекс: $\forall\, l_{\overrightarrow{uv}}, l_{\overrightarrow{vu}} \in L, f_{\langle s,d\rangle} \in F,\ w\left(l_{\overrightarrow{uv}}, f_{\langle s,d\rangle}\right) \prec \phi$ следует $w\left(l_{\overrightarrow{vu}}, f_{\langle s,d\rangle}\right) \succeq \phi$

Переход:

$\forall\, l_{\overrightarrow{u_1u_2}}, l_{\overrightarrow{u_2u_3}}, \dots, l_{\overrightarrow{u_{k-1}u_k}}, l_{\overrightarrow{u_1u_k}} \in L, f_{\langle s,d\rangle} \in F, w\left(l_{\overrightarrow{u_1u_2}}, f_{\langle s,d\rangle}\right) \prec \phi, w\left(l_{\overrightarrow{u_2u_3}}, f_{\langle s,d\rangle}\right) \prec \phi, \dots,$ и $w\left(l_{\overrightarrow{u_{k-1}u_k}}, f_{\langle s,d\rangle}\right) \prec \phi$ следует $w\left(l_{\overrightarrow{u_1u_k}}, f_{\langle s,d\rangle}\right) \prec \phi$

Независимость источников:

$\forall\, f_{\langle s_1,d\rangle}, f_{\langle s_2,d\rangle} \in F, l_{\overrightarrow{uv}} \in L,$ всегда верно $w\left(l_{\overrightarrow{uv}}, f_{\langle s_1,d\rangle}\right) = w\left(l_{\overrightarrow{uv}}, f_{\langle s_2,d\rangle}\right)$

Строгое предпочтение: $\forall\ f_{\langle s,d\rangle} \in F,$ не существует случаев, когда для двух связей $\overrightarrow{uv}$ и $\overrightarrow{uw}$ из узла u выполняется $w\left(l_{\overrightarrow{uv}}, f_{\langle s,d\rangle}\right) = w\left(l_{\overrightarrow{uw}}, f_{\langle s,d\rangle}\right)$.

В географической маршрутизации «жадный» алгоритм пытается доставить трафик сообщений как можно «ближе» к месту назначения шаг за шагом, при этом значение «ближе» представляется согласно выбранным метрикам маршрутизации. Не всегда необходимо рассматривать именно евклидовое расстояние, а можно применять иные метрики, также основанные на информации о местоположении. Физический смысл «мьютекса» заключается в следующем: из двух узлов, один «ближе» к адресату, чем другой, для определенного потока сообщений. Переход означает, что «расстояние» между узлами-маршрутизаторами и узлом назначения строго уменьшается в процессе передачи определенного потока. Независимость источников означает, что метрики маршрутизации, т.е. «расстояние», между узлами-маршрутизаторами и узлом назначения, не должны изменять информацию узла источника для определенного потока. Строгое предпочтение означает, что существует единственный узел среди остальных, который «ближе» всего к месту назначения, т.е. не возникает ничьей.

Рассматриваемая алгебраическая система отличается от алгебраической системы, используемой для дистанционно-векторных протоколов и протоколов, учитывающих состояние соединений, т.к. в последней не учитывается вес связей в зависимости от потока сообщений. В «жадной» маршрутизации местоположение узла-отправителя и узла-адресата напрямую влияет на выбор путей передачи. Более подробное описание и доказательство теорем алгебраической системы приведено в Приложении Б.

При применении «жадного» алгоритма в качестве основы для передачи сообщений, каждый узел-маршрутизатор рассчитывает следующий узел для передачи для каждого входящего сообщения. Т.к. на узлах отсутствует информация о полном пути сообщения, то и последовательной передачи (свойство согласованности) не может быть соответственно. Из этого следует, что в данном случае необходимо рассмотреть лишь условие отсутствия петель.

Когда «жадный» алгоритм используется в качестве алгоритма нахождения путей по запросу (реактивный метод), узлам-маршрутизаторам нет необходимости хранить текущее состояние потока. Когда новый поток поступает на узел-источник, то он инициализирует фазу поиска маршрутов, используя «жадный» метод для пересылки сообщения с запросом маршрута. Это сообщение записывает все узлы, через которое оно прошло. Адресат возвращает сообщение с ответом на запрос маршрута источнику. В ответе содержится весь путь целиком, найденный сообщением запроса. Соответственно, когда узел-источник получает этот ответ, он может начать процесс передачи сообщений, включая при этом в служебную часть сообщения весь найденный путь. Узлы-маршрутизаторы пересылают сообщение согласно записанному в нём пути. Это автоматически гарантирует согласованность процесса маршрутизации, что также гарантирует отсутствие в нём петель.

«Жадный» алгоритм также может быть использован в качестве проактивного метода нахождения маршрутов и быть совмещен с маршрутизацией от интервала к интервалу (hop-by-hop). Когда узлу требуется отправить поток сообщений, он в первую очередь инициирует фазу нахождения маршрута, а соответственно и процесс заполнения таблиц маршрутизации на промежуточных узлах. Далее, при передаче информационных сообщений они следуют по маршруту «проложенному» во время фазы поиска, т.е. на каждом шаге используются таблицы маршрутизации промежуточных узлов, таким образом, информация о полном пути в поля сообщения не добавляется.

Существует два способа построения таблиц маршрутизации во время фазы поиска маршрута. В первом случае, поля таблицы заполняются с помощью ответного сообщения на

запрос поиска маршрута. Во втором случае, таблицы маршрутизации строятся только в зависимости от прохождения самого запроса поиска маршрута.

При использовании метода с ответным сообщением узел-источник отправляет сообщение для поиска маршрута, используя «жадный» алгоритм маршрутизации. Сообщение записывает все узлы, которое оно проходит, а адресат при его получении возвращает ответное сообщение, содержащее полный найденный путь. Промежуточные узлы, заполняют свои таблицы маршрутизации при получении сообщения с ответом на запрос маршрута.

При прямом построении таблиц маршрутизации узел, которому необходимо передать поток сообщений, сначала отправляет сообщение с запросом на построение маршрута, используя «жадный» метод маршрутизации. Таблицы маршрутизации на промежуточных узлах заполняются согласно прохождению этого сообщения. Ответное сообщение также, возможно, необходимо для подтверждения существования пути.

Тем не менее, несмотря на похожесть сочетания «жадного» алгоритма с методом передачи от интервала к интервалу и сочетания «жадного» алгоритма с проактивным методом маршрутизации, требования для обеспечения отсутствия петель различны и при этом зависят от способа построения таблиц маршрутизации.

В предложенном в работе протоколе маршрутизации для передачи сообщений используется метрика *Most Forward within Radius* (*MFR*). При использовании данной метрики в качестве маршрутизатора в зоне прямой видимости выбирается узел, который находится ближе всех к адресату, с целью достичь минимального числа интервалов между узлами, которые должно пройти сообщение. При этом для построения таблиц маршрутизации используется проактивный метод прямого построения. Согласно рассмотренной алгебраической системе, предложенный алгоритм не образует петель и сходится в сетях, где алгоритм не получает пустого множества значений выбора. Также, протокол является согласованным.

4.3 Обработка сетевых сообщений

Информационные единицы подуровня VSS стандарта VDL Mode 4 должны соответствовать структуре кадра стандарта ISO3309 [100], описанной в таблице 8, за исключением случаев, указанных далее в описании.

Сообщение, занимающее один слот, содержит 24 октета данных (n=24). Таким образом, при назначении 8 битов (или одного октета) в качестве неинформационных битов и двух октетов для флагов, в «однослотовом» сообщении остающееся значение октетов n = 21.

Таблица 8. Формат данных

Октет	*Номер бита*							
	8	*7*	*6*	*5*	*4*	*3*	*2*	*1*
–	0	1	1	1	1	1	1	0
1	s_{27}	s_{26}	s_{25}	ver_3	ver_2	ver_1	rid	a/d
2	s_{24}	s_{23}	s_{22}	s_{21}	s_{20}	s_{19}	s_{18}	s_{17}
3	s_{16}	s_{15}	s_{14}	s_{13}	s_{12}	s_{11}	s_{10}	s_9
4	s_8	s_7	s_6	s_5	s_4	s_3	s_2	s_1
5	mi_k	. . . . .			mi_4	mi_3	mi_2	mi_1
6	in_k							
7 ÷ (n-5)			. . . . .					
(n-4)								
(n-3)		in_1	rd_k	. . . . .				
(n-2)	$erid_k$				$erid_1$			rd_1
(n-1)	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}	c_{16}
(n)	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8
–	0	1	1	1	1	1	1	0

Подполе «автономный/адресуемый» (*a/d*) имеет только два значения: 0 – при передаче сообщения в слоте с помощью случайного алгоритма доступа, или в случае, если слот был зарезервирован приёмопередатчиком (узлом) самим для себя; 1 – при передаче сообщения в слоте по «задержанному» алгоритму доступа, или в слоте, который был зарезервирован для приёмопередатчика другим приёмопередатчиком. Таким образом, подполе «автономный/адресуемый» показывает, зарезервировал ли приёмопередатчик слот для самого себя или этот слот был зарезервирован для него другим приёмопередатчиком.

Если узел передаёт сообщение и при этом устанавливает бит a/d в нуль, то это подразумевает, что он сформировал правильную таблицу резервирования. Передача, выполняемая мобильным узлом, которая содержит периодическое широковещательное резервирование, и в которой бит a/d установлен в единицу, будет вызывать процедуры изолирования (информация от узла, передавшего такое сообщение должна игнорироваться).

Согласно разработанному алгоритму, бит a/d, в сетевых сообщениях, устанавливается в значение равное нулю.

Подполе номера версии (*ver*) должно обозначать версию VDL Mode 4, которая поддерживается узлом. В текущей редакции стандарта, при передаче сообщений, оно должно быть установлено в 000_2. Если узел принимает сообщение, в котором номер версии не равен нулю ($ver \neq 0$), она должна проинформировать пользователя подуровня VSS, что был принят ненулевой номер версии, и игнорировать остаток этого сообщения.

Адрес отправителя (*s*) на передающей станции должен кодироваться в 27-битном поле, как это описано в таблице 9.

Таблица 9. Кодирование адреса отправителя

Побитное кодирование			*Тип описания*	*Биты с 1 по 24*
27	26	25		
0	0	0	Мобильная станция	Неуникальная идентичность
0	0	1	Воздушное судно	24-битный адрес ICAO
0	1	0	Наземное транспортное средство	Адресное пространство, администрируемое в национальном домене
0	1	1	Зарезервировано для будущего применения	
1	0	0	Наземная станция	Адресное пространство, администрируемое через организацию ICAO
1	0	1	Наземная станция	Адресное пространство, делегируемое организацией ICAO
1	1	0	Зарезервировано для будущего применения	
1	1	1	Все станции при широковещательной передаче	Все станции

Исходя из таблицы 9 и технической документации VDL Mode 4 (ICAO DOC 9816), 27-битный адрес приёмопередатчика должен состоять из 24-битного уникального бортового адреса ICAO и трёх бит идентификации типа места или транспортного средства, где установлен приёмопередатчик. Идентификатор сообщения (*mi*) в сообщении должен кодироваться в поле переменной длины, как это определяется в таблице 10.

Таблица 10. Назначение битов идентификатора (*mi*)

Поле идентификатора (mi)								*Назначенный тип данных*	*Пользователь подуровня VSS*
mi_8	mi_7	mi_6	mi_5	mi_4	mi_3	mi_2	mi_1		
x	x	x	x	x	x	x	0	Сообщение типа “synchronization burst”	LME
x	x	x	x	1	1	0	1	Данные подуровня DLS	DLS
x	x	x	x	x	0	1	1		
x	x	x	x	x	1	1	1		
1	0	0	1	0	1	0	1		
x	x	1	0	0	1	0	1		
x	x	x	x	x	0	0	1	Сообщение с данными общего запроса	Определяется полем *r-mi*
x	0	0	0	0	1	0	1	Нет операции (данные отсутствуют)	VSS
x	0	1	1	0	1	0	1	Зарезервировано	
x	1	0	1	0	1	0	1	Зарезервировано	
x	1	1	1	0	1	0	1	Сообщение с данными общего запроса	Определяется полем *r-mi*
0	0	0	1	0	1	0	1	Расширение идентификатора ID до следующих четырёх битов	
x	1	0	0	0	1	0	1	Сообщение входа в сеть	VSS

Биты, обозначенные как "x", доступны для использования в рамках информационного поля. Подполе "r-mi" является запрашиваемым подполем идентификатора сообщения.

Согласно таблице 10, существует два свободных идентификатора сообщения x0110101 и x1010101. Для сетевых сообщений использован идентификатор x0110101, а установкой “x” в 0 или 1 определяются типы сетевых сообщений – сообщение от базовой станции или мобильного узла соответственно.

Информационное поле (*in*) определяется документацией, как необязательное и должно содержать данные подуровня VSS, определяемые пользователем. В свою очередь, пользователем подуровня VSS являются различные информационные и управляющие приложения авионики. В качестве одного из таких приложений и позиционируется сеть передачи данных АЗН-В, поэтому в случае с сетевыми сообщениями, в поле *in* передаются данные, используемые для построения таблиц маршрутизации, а также для осуществления выбора маршрутизаторов.

Идентификатор резервирования (*rid*) в сообщении должен кодироваться в 1-битном поле. Если *rid* равняется единице, то это означает, что тип резервирования является либо «нулевым» резервированием, либо периодическим широковещательным резервированием, либо комбинированным периодическим широковещательным и инкрементированным широковещательным резервированием, и что в сообщении не задействовано подполе расширенного идентификатора резервирования (*erid*). В противном случае, поле *erid* должно обозначать другие типы резервирования: ответное сообщение; резервирование для протокола с большим отрицательным смещением слотов (BND); резервирование блока слотов базовой станцией в «суперфрейме» или в секунде; резервирование для запроса на одну адресуемую станцию; резервирование для запроса на передачу информации; инкрементированное широковещательное резервирование.

Подполе *rid* (и *erid*, если оно присутствует) определяет интерпретацию поля данных резервирования (*rd*). Биты, обозначенные как "x", являются доступными для использования внутри поля данных резервирования (*rd*).

В сумме, при передаче «однослотового» сообщения, для использования доступно следующее кол-во бит:

БИТЫ = 21*8 – 1(a/d) – 1 (rid) – 3(ver) – 3(идентификатор станции) – 24 (ИКАО-адрес) – 8 (идентификатор сообщения) – 16 (CRC) – 10 (резервирование) ≈ 102 бита (примерно, т.к. кол-во бит может варьироваться в подполе резервирования и из-за битстаффинга [101]).

Если вести передачу в двух слотах, то между флагами, разделяющими блоки информации, может уместиться 54 октета, с учётом нарастания мощности в начале

сообщения, спада с защитным интервалом в конце сообщения и 66 бит на фиксированную часть. В «трёхслотовом» сообщении, с учётом тех же условий, может быть задействовано до 86 октетов для передачи пользовательской информации. При дальнейшем прибавлении каждого слота доступное информационное поле будет увеличиваться на 32 октета.

Если у узла существует необходимость запроса таблицы соседних узлов у другого узла, то данная операция должна производиться по запросу экипажа ВС. Это связано с тем, что таблицы соседних узлов могут занимать такое кол-во бит, передача которого может привести к существенным нагрузкам на канал передачи данных.

Согласно технической документации, чтобы один узел передал другому свою таблицу соседей, запрашивающий должен инициализировать соединение, затем запросить эту информацию и после ряда процедур между подуровнями соединения принять её. Сообщения с данными пользователя и данные субъектов управления на подуровне LME должны передаваться в информационных полях INFO, UDATA и CTRL, т.е. в информационных единицах протокола передачи данных (*datalink protocol data unit - DLPDU*), которые все вместе определяются, как блоки данных DATA DLPDU. Данные субъектов управления на подуровне LME должны содержаться только в полях CTRL и UCTRL. Уровень установления соединения должен обрабатывать размеры блоков данных с фрагментированием или без, в зависимости от его размера и использовать при этом соответствующие процедуры. В сообщении с пользовательскими данными должен содержаться только один блок данных DATA DLPDU.

Для осуществления запроса должны быть использованы полу подуровня VSS, указанные в таблице 11.

Таблица 11. Блок запроса информации

Описание	*Октет*	*Номер бита*							
		8	*7*	*6*	*5*	*4*	*3*	*2*	*1*
	5	r-mi$_5$	……				0	0	1
Запрашиваемый идентификатор сообщения (*r-mi*)	6	x	r-mi$_n$	……..					r-mi$_2$
Специфические параметры пользователя VSS (*prm*)	7 ÷ (n-3)								Prm$_{11}$
	(n-2)	Prm$_{k8}$							

Для получения таблицы соседей, узел, в сообщении с запросом, должен указать специфичный *r-mi* идентификатор. Из таблицы существующих типов сообщений можно воспользоваться одним из зарезервированных под будущее использование, например: 11010101.

В качестве передаваемых параметров VSS могут использоваться параметры по умолчанию, т.к. не существует конкретных значений для рассматриваемого приложения. Может ли уточнение, каких либо из этих параметров, действительно повысить эффективность информационного обмена – вопрос моделирования. Указанное подполе является необязательным.

В таблице 12 приведены используемые и передаваемые в сообщениях VDL Mode 4 параметры QoS.

Таблица 12. Передаваемые параметры QoS VDL Mode 4

Симв-ол	*Наименование параметра*		*Минимальное значение*	*Максимальное значение*	*По умолчанию*	*Пошаговое приращение*
Q1	Приоритет		0	15	11	1
Q2a	Ограничения на дальность при выборе слота для уровня 1		0	1000 NM	150 NM	1 NM
Q2b	Ограничения на дальность при выборе слота для уровня 2		0	1000 NM	150 NM	1 NM
Q2c	Ограничения на дальность при выборе слота для уровня 3		0	1000 NM	0	1 NM
Q2d	Ограничения на дальность при выборе слота для уровня 4		0	1000 NM	300 NM	1 NM
Q3	Замещение данных, стоящих в очереди		FALSE	TRUE	FALSE	–
Q4	Количество доступных слотов		1	20	3	1
$Q5_{min}$	параметры повторной передачи в подуровне VSS	минимум	0 сек.	20 сек.	0 сек.	13.33 мс
$Q5_{max}$		максимум	1 сек.	20 сек.	5 сек.	13.33 мс
$Q5_{mult}$		множитель	1 сек.	2.5 сек.	1 сек.	0.01 сек.
$Q5_{exp}$		показатель степени	1	2.5	1.5	0.01
$Q5_{num}$		количество попыток	1	15	4	1
$Q5_{wait}$		максимальное время ожидания ответа	1 сек.	120 сек.	60 сек.	1 сек.

В таблице 13 приведён результирующий формат сообщения для запроса таблицы соседних узлов, учитывающих вышеизложенные условия и утверждения. Передаваемое сообщение также дожни включать соответствующие подполя фиксированной части (до 5 октета и после октета n-2).

Как видно из таблицы 13, значение lg ограничено числом 7. Для того чтобы создать некоторую картину местоположения и намерений требуется: идентификатор узла, его местоположение и вектор направления, что в упрощенном варианте составляет:

27 (Адрес ICAO) + 12 (Широта) + 12 (Высота) + 14 (Долгота) + 11 (наземная скорость) + 11 (наземная траектория) = 77 бит

По проведенным расчётам, для одного слота размер информационного поля составит примерно 102 бита, т.е. существует возможность разместить в сообщении информацию только об одном узле. В двух слотах доступно уже 366 бит, т.е. уместится информация о 4-ёх узлах. Если использовать 5 слотов, то размера поля информации будет достаточно для 14 узлов, при этом 5 слотов составляет 1/15 от секундного кадра. Такой объём информации при единичных и неодновременных запросах займёт малую часть пропускной способности канала, поэтому поле lg для данного запроса предлагается по умолчанию ставить в значение 5.

Таблица 13. Результирующий формат сообщения

Описание	*Октет*	*Номер бита*							
		8	*7*	*6*	*5*	*4*	*3*	*2*	*1*
	5	1	0	1	0	1	0	0	1
Запрашиваемый идентификатор сообщения (*r-mi*)	6	x	x	x	x	x	1	1	0
Специфические параметры пользователя VSS (*prm*)	7 ÷ (n-10)	…….							$prm_{1,1}$
	(n-9)								$prm_{n,n}$
Адрес получателя (*d*)	(n-8)	d_{24}	d_{23}	d_{22}	d_{21}	d_{20}	d_{19}	d_{18}	d_{17}
	(n-7)	d_{16}	d_{15}	d_{14}	d_{13}	d_{12}	d_{11}	d_{10}	d_9
	(n-6)	d_8	d_7	d_6	d_5	d_4	d_3	d_2	d_1
Адрес получателя (*d*) - старшие биты, Флаг отправитель/получатель (*sdf*), Смещение слота для ответа (*ro*) - старшие биты	(n-5)	ro_{12}	ro_{11}	ro_{10}	ro_9	sdf	d_{27}	d_{26}	d_{25}
Смещение слота для ответа (*ro*)	(n-4)	ro_8	ro_7	ro_6	ro_5	ro_4	ro_3	ro_2	ro_1
Длина (*lg*)	(n-3)	res	res	res	res	res	lg_3	lg_2	lg_1
Приоритет (*pr*), Расширенный идентификатор резервирования = 0010_2	(n-2)	0	0	1	0	pr_4	pr_3	pr_2	pr_1

Определения и значения использованных полей указаны в таблице 14.

Таблица 14. Параметры, устанавливаемые при передаче сообщения запроса таблицы соседних узлов

Подполе	*Диапазон значений*	*Способ кодирования / предпринимаемое действие*	*Определения*
Смещение слота для ответа (*ro*)	0 ÷ 4095		*ro* определяет слот относительно первого слота передачи
Адрес получателя (*d*)	0 ÷ (2^{27} – 1)	Таблица 10	*d* является 27-битным адресом станции назначения
Флажок отправитель / получатель (*sdf*)	Булевый	Если *sdf = 0*, то резервировать слот ответа для станции-получателя, чтобы она в нём осуществила передачу. Если *sdf = 1*, то резервировать слот ответа для станции-отправителя, чтобы она в нём осуществила передачу.	*sdf* показывает, которая из станций будет отвечать в зарезервированном ответном слоте. Отметьте, что станцией-отправителем является станция, разместившая резервирование.
Длина (*lg*)	0 ÷ 7		*lg* является числом, на единицу меньшим числа слотов, которое резервируется для ответа
Приоритет (*pr*)	0 ÷ 15	Таблица 16	Используется для сортировки очереди отправляемых сообщений на подуровне VSS

Поле приоритета может иметь несколько значений, указанных в таблице 15.

Таблица 15. Приоритеты сообщений

Категории сообщений	Q1
Управление сетью/системами	14
Связь на случай бедствия	13
Связь на случай крайней необходимости	12
Высокоприоритетные сообщения, связанные с безопасностью полёта	11
Сообщения, связанные с безопасностью полёта, имеющие нормальный приоритет	10
Связь на случай метеорологических осложнений	9
Связь на случай регулирования условий проведения полёта	8
Сообщения, приходящие как воздушное обслуживание	7
Администрирование сети/системы	6
Сообщения воздушного администрирования	5
Зарезервировано для будущего использования	4
Связь на случай настоятельного приоритетного администрирования и для чартерных рейсов ООН	3
Связь на случай высокоприоритетного администрирования и с госорганами / правительством	2
Администрирование с нормальным приоритетом	1
Администрирование с низким приоритетом	0

Конкретная величина приоритета будет зависеть от поправок, выносимых на обсуждение в ICAO. Теоретически значения могут быть: 12, 11, 10, 8, 4, 1 и 0. Т.к. неиспользуемым на данный момент является значение приоритета равное 4, то именно оно используется в модели для сетевых сообщений.

Ответное сообщение на запрос таблицы маршрутизации будет иметь структуру, представленную в таблице 16:

Таблица 16. Структура ответного сообщения

Описание	*Октет*	*Номер бита*							
		8	*7*	*6*	*5*	*4*	*3*	*2*	*1*
Флаг	–	0	1	1	1	1	1	1	0
(*a/d*), *rid*, *ver*, Адрес отправителя (три старших бита)	1	s_{27}	s_{26}	s_{25}	0	0	0	0	1
адрес отправителя (*s*)	2	s_{24}	s_{23}	s_{22}	s_{21}	s_{20}	s_{19}	s_{18}	s_{17}
	3	s_{16}	s_{15}	s_{14}	s_{13}	s_{12}	s_{11}	s_{10}	s_9
	4	s_8	s_7	s_6	s_5	s_4	s_3	s_2	s_1
Идентификатор сообщения (*mi*)	5	1	1	0	1	0	1	0	1
Информационное поле (*in*)		N8	N7	N6	N5	N4	N3	N2	N1
	6	$S1_8$	$S1_7$	$S1_6$	$S1_5$	$S1_4$	$S1_3$	$S1_2$	$S1_1$
	7	$S1_{17}$	$S1_{16}$	$S1_{15}$	$S1_{14}$	$S1_{13}$	$S1_{12}$	$S1_{11}$	$S1_9$
	8	$S1_{25}$	$S1_{24}$	$S1_{23}$	$S1_{22}$	$S1_{21}$	$S1_{20}$	$S1_{19}$	$S1_{18}$
	9	Lat6	Lat5	Lat4	Lat3	Lat2	Lat1	$S1_{27}$	$S1_{26}$
	10	Alt2	Alt1	Lat12	Lat11	Lat10	Lat9	Lat8	Lat7
	11	Alt10	Alt9	Alt8	Alt7	Alt6	Alt5	Alt4	Alt3
	12	Lon6	Lon5	Lon4	Lon3	Lon2	Lon1	Alt12	Alt11
	13	Lon14	Lon13	Lon12	Lon11	Lon10	Lon9	Lon8	Lon7
	14	Gs8	Gs7	Gs6	Gs5	Gs4	Gs3	Gs2	Gs1
	15	Gt5	Gt4	Gt3	Gt2	Gt1	Gs11	Gs10	Gs9
	16	$S2_2$	$S2_1$	Gt11	Gt10	Gt9	Gt8	Gt7	Gt6
	17	И так далее от 2 до n-го пользователя							
	18 ÷ (n-3)	x	x	x	x	x	x	x	x
Расширенный идентификатор резервирования (*erid*)	(n-2)	0	0	0	0	0	x	x	x
Код циклического контроля CRC (c)	(n-1)	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}	c_{16}
	(n)	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8
Флаг	–	0	1	1	1	1	1	1	0

В таблице 16 представлены следующие значения и определения (кроме обозначенных в самой таблице) подполей сообщения с ответом на запрос таблицы соседних узлов:

1. «Автономный/адресуемый» установлено в 1_2, т.к. сообщение будет передаваться в слотах, зарезервированных другим приёмопередатчиком;
2. Версия VDL Mode 4 и идентификатор резервирования – 0_2;
3. В качестве идентификатора сообщения *mi* использовано незарезервированное значение из возможных сообщений подуровня DLS;
4. N_n – означает кол-во узлов, о которых передана информация;
5. SN_n – адрес узла из таблицы соседних узлов;
6. Lat_n, Alt_n, Lon_n – соответственно широта, высота и долгота в кодировке по алгоритму CPR (compact position reporting);
7. *Gs* – наземная скорость;
8. *Gt* – наземная траектория;
9. Расширенный идентификатор резервирования – 0_2.

В параметре N указывается кол-во записанных в сообщении узлов из передаваемой таблицы соседних узлов. При считывании 77 бит - получается информация об одном узле, далее уменьшая N на единицу и считывая ещё 77 бит, получается информация о втором узле. Соответственно, N по достижении им 0, при этом остаток сообщения (кроме информации о резервировании и CRC) игнорируется.

Таким образом, не только источник, запрашивающий таблицу соседних узлов, но и остальные узлы, «слышащие» это сообщение, получают данные об узлах, которые могут находиться за пределами прямого приёма от них.

Во многих, рассматриваемых на данный момент случаях применения самоорганизующейся сети, в конечном итоге, связь должна осуществляться именно с наземной (базовой) станцией. Этими случаями являются: многоинтервальная передача сообщений АЗН-В от участников воздушного движения, находящихся за пределами прямого приёма пунктами УВД; идентификация террористической деятельности (например, фантомные сетевые узлы или спам); ретрансляция голоса от участников воздушного движения, находящихся за пределами прямого приёма пунктами УВД или от операторов БПЛА до авиадиспетчера.

Идеология, заключающаяся в том, что координаты наземных станций будут заложены в память приёмопередатчика или бортового компьютера имеет особенности, которые могут негативно повлиять на автономность сети - кто-то должен будет заниматься зашиванием этих данных в аппаратуру. Существует вероятность, что любой участник воздушного

движения может изменить маршрут следования из-за различных обстоятельств, поэтому возникает большое количество вопросов. Будут ли данные о БС закладываться каждый раз при смене маршрута или в бортовом компьютере будут храниться все возможные наземные станции? Что делать, если появляется новая наземная станция или борт, на котором установлен приёмопередатчик, полностью меняет место дислокации, где нет ни одной известной ему наземной станции?

В противоположность рассматриваемому подходу, в работе применён подход, подразумевающий в себе периодическую передачу базовыми станциями в широковещательном режиме специальных сообщений, которые затем ретранслируются сетевыми узлами. Данный подход позволяет избежать необходимости в предварительной записи информации о БС в приёмопередатчики, что имеет следующие преимущества:

1. Отсутствует необходимость временных и финансовых затрат для обслуживания приёмопередатчиков на предмет наличия актуальной информации о БС;
2. Мобильные узлы шлют сетевые сообщения АЗН-В только в том случае, когда было получено сообщение от БС, что говорит о работоспособности БС и существовании маршрута для передачи данных;
3. Использование передвижных БС (например, для развёртывания сети БЛА).

Предположим, что информация о наземных станциях заложена в приёмопередатчик тем или иным способом. При этом стоит отметить, что некоторые рассматриваемые величины и таймеры – предполагаемые, точное же их значение определяется в разделе, посвященной моделированию.

Для целей обеспечения процесса передачи данных АЗН-В по сети от узлов, вышедших из зоны прямого приёма пунктами УВД, было разработано сетевое сообщение типа «synchronization burst». Данное сообщение маршрутизируется мобильными узлами до БС, если это возможно. Формат сообщения типа «synchronization burst» приведен в таблице 17, в которой присутствуют следующие подполя:

- «tqc» - двоичный флаг, указывающий на то, передаётся ли в сообщении точка изменения маршрута или текущий вектор состояния (0 и 1 соответственно);
- «b/g» - двоичный флаг, указывающий на то, передаётся ли в сообщении барометрическая или геометрическая высота (0 и 1 соответственно);
- «cprf» - двоичный флаг, указывающий формат местоположения по алгоритму CPR: четный или нечетный (0 и 1 соответственно);

- «nic» - категория навигационной точности (от 20 морских миль до 7.5 м), принимает значения от 0000_2 до 1011_2;
- «lat», «lon» - широта и долгота, 12 и 14 бит по алгоритму CPR соответственно;
- «balt» - высота, разделенная по диапазонам – 12 бит;
- «tfom» - тип источника для внутренних часов, принимает значения от 000_2 до 111_2;
- «*da*» - задержка выдачи навигационных данных, разбитая на 16 диапазонов, от 100 мс до 4 с (значения от 0000_2 до 1111_2);
- «*id*» и «*ID*» - идентификатор переменного информационного поля «*in*» и его расширение.

Таблица 17. Формат сообщения типа «synchronization burst»

Октет	*Номер бита*							
	8	*7*	*6*	*5*	*4*	*3*	*2*	*1*
5	nic_4	nic_3	nic_2	nic_1	cprf	b/g	tqc	0
6	lat_8	lat_7	lat_6	lat_5	lat_4	lat_3	lat_2	lat_1
7	$balt_{12}$	$balt_{11}$	$balt_{10}$	$balt_9$	lat_{12}	lat_{11}	lat_{10}	lat_9
8	$balt_8$	$balt_7$	$balt_6$	$balt_5$	$balt_4$	$balt_3$	$balt_2$	$balt_1$
9	lon_8	lon_7	lon_6	lon_5	lon_4	lon_3	lon_2	lon_1
10	$tfom_2$	$tfom_1$	lon_{14}	lon_{13}	lon_{12}	lon_{11}	lon_{10}	lon_9
11	da_4	da_3	da_2	da_1	ID_4	ID_3	ID_2	ID_1
12	$id1_4$	$id1_3$	$id1_2$	$id1_1$	$id2_4$	$id2_3$	$id2_2$	$id2_1$
13	$id3_4$	$id3_3$	$id3_2$	$id3_1$	in_k			
14								
15			.....					
16								
17	in_{14}	in_{13}	in_{12}	in_{11}	in_{10}	in_9	in_8	in_7
18	in_6	in_5	in_4	in_3	in_2	in_1		

В таблице 18 приводится битовое наполнение переменного информационного поля для сетевого сообщения типа «synchronization burst».

Таблица 18. Информационное поле сетевого пакета синхронизации с идентификатором

Описание	*Октет*	*Номер бита*							
		8	*7*	*6*	*5*	*4*	*3*	*2*	*1*
Идентификатор ID информационного поля	11	x	x	x	x	1	0	1	1
Биты ретранслятора sd 27 по 4	12	sd27	sd26	sd25	sd24	sd23	sd22	sd21	sd20
	13	sd19	sd18	sd17	sd16	sd15	sd14	sd13	sd12
	14	sd11	sd10	sd9	sd8	sd7	sd6	sd5	sd4
Биты ретранслятора sd с 3 по 1 и биты адреса наземной станции ss с 27 по 23	15	sd3	sd2	sd1	ss27	ss26	ss25	ss24	ss23
Биты адреса наземной станции с 22 по 1	16	ss22	ss21	ss20	ss19	ss18	ss17	ss16	ss15
	17	ss14	ss13	ss12	ss11	ss10	ss9	ss8	ss7
	18	ss6	ss5	ss4	ss3	ss2	ss1		

Используемый идентификатор не имеет дополнительных уточнений «id1», «id2» и «id3», поэтому эти подполя доступны в качестве информационных битов. Данный подход позволяет передавать в переменном информационном поле два 27-битных ICAO-адреса.

Протокол маршрутизации предусматривает, что при каждой ретрансляции биты ретранслятора «sd» перезаписываются на биты следующего ретранслятора. Биты «ss» не меняются, т.к. это конечный пункт (адресат) и по этому адресу происходит поиск соответствующей ему БС и маршрутизаторов.

Используя данный подход, нет необходимости копить по мере прохождения сообщения набор меток или адресов ретрансляторов: алгоритм маршрутизации оперирует местоположением, а не какой-то конкретной цепочкой ретрансляторов. Данный подход частично схож с технологией MPLS [102], где на каждом маршрутизаторе лишь хранятся записи, по какому направлению дальше передавать пакет, т.е. полный путь (знание о сетевой топологии) на каждом маршрутизаторе не содержится. Идентификатор информационного поля равный B_{16} (1011_2) выбран согласно таблице 19 и находится в строке 18 (не зарезервирован).

Таблица 19. Значения идентификаторов информационного поля для сообщения «synchronization burst»

Идентификатор информационного поля (ID)	*Расширение 1 поля ID (id1)*	*Расширение 2 поля ID (id2)*	*Наименование информационного поля*
0_{16}	отсутствует	отсутствует	Базовое сообщение
1_{16}	отсутствует	отсутствует	Высокая динамика
2_{16}	отсутствует	отсутствует	Полные данные о местоположении
3_{16}	отсутствует	отсутствует	Базовое наземное
4_{16}	отсутствует	отсутствует	Время UTC
5_{16}	отсутствует	отсутствует	SVQ в одном слоте
$6_{16} \div 7_{16}$	отсутствует	отсутствует	Доступно для будущего использования
8_{16}	отсутствует	отсутствует	TCP/SVQ в двух слотах
9_{16}	отсутствует	отсутствует	TCP в одном слоте
A_{16}	0_{16}	отсутствует	Доступно для будущего применения
A_{16}	1_{16}	отсутствует	Данные, характеризующие воздушное судно (позывной, категория, статус)
A_{16}	$2_{16} \div 9_{16}$	отсутствует	Доступно для будущего применения
A_{16}	A_{16}	0_{16}	Высокое разрешение координат
A_{16}	A_{16}	$1_{16} \div 9_{16}$	Доступно для будущего применения
A_{16}	A_{16}	A_{16}	Признак применения последующих расширений идентификатора ID (поле доступно для будущего применения)
A_{16}	A_{16}	$B_{16} \div F_{16}$	Доступно для будущего применения
A_{16}	$B_{16} \div F_{16}$	отсутствует	Доступно для будущего применения
$B_{16} \div E_{16}$	отсутствует	отсутствует	Доступно для будущего применения
F_{16}	отсутствует	отсутствует	Нет обеспечиваемых информационных полей

Документацией 9816 предусмотрена возможность использования дополнительных локальных каналов связи авиационного диапазона для обеспечения функционирования различных приложений стандарта VDL Mode 4. Так как существует вероятность возникновения высокой информационной нагрузки сетевыми сообщениями на узлах (особенно расположенных вблизи БС): предложена концепция, в которой для создания мобильной самоорганизующейся сети используются локальные каналы связи.

Согласно концепции, для того, чтобы идентифицировать себя как участника сети, узел должен передавать в широковещательном режиме обычные сообщения типа «synchronization burst» на специальном сетевом канале. Узлы, которые находятся в зоне радиовидимости, но не передают данные на этом специализированном канале, не могут выступать в качестве участников сети. Как было описано выше сетевое сообщение «synchronization burst» имеет специальный тип, в поле данных хранит адрес наземной станции и станции ретранслятора, к которой он хочет отправить это сообщения.

Стандартом предусматривается возможность передачи базовыми станциями сообщений типа «DoS» (*Directy of Service*), поэтому частота передачи обычных и сетевых сообщений на специализированных локальных каналах может указываться именно в этих сообщениях.

Разработанный протокол позволяет использовать «однослотовые» сообщения и избежать задействования процедур «короткой» передачи, которые подразумевают отправку сообщений с подтверждением получения, что позволяет уменьшить число служебных сообщений в сети. Для подтверждения отправления сетевого сообщения с данными АЗН-В на узле-источнике и промежуточных узлах используется метод пассивного прослушивания, т.е. отправленное сообщение сохраняется узлом на некоторое время, при вещании этого сообщения промежуточным узлом и его получении (т.к. используются всенаправленные антенны) производится проверка на схожесть. В случае отсутствия на приёме схожего сообщения в некоторый период времени, производится новая попытка отправки сообщения. Для адресации используются 27-битные уникальные адреса ICAO, по сути, являющиеся аналогами MAC-адресов для приёмопередатчиков VDL Mode 4, поэтому маршрутизация осуществляется на канальном уровне.

На рисунках 25, 26а и 26б представлены упрощенные блок-схемы алгоритмов обработки сетевых сообщений на подуровне маршрутизации для МС. Алгоритм работы БС заключается лишь в формировании необходимой информации для периодического вещания и приёма сообщений, адресованных ей. На блок-схемах обозначение «PDU» относится к

термину – protocol data unit (информационная единица протокола), таким образом, на подуровне маршрутизации подобных единиц две – блок информации БС и блок информации МС. Первый состоит из адреса БС, времени жизни сообщения и уникального номера, второй состоит из адреса БС, которой адресуется сообщение и адрес промежуточного узла. Оба блока занимают всего 54 бита в переменном информационном поле «однослотового» сообщения АЗН-В.

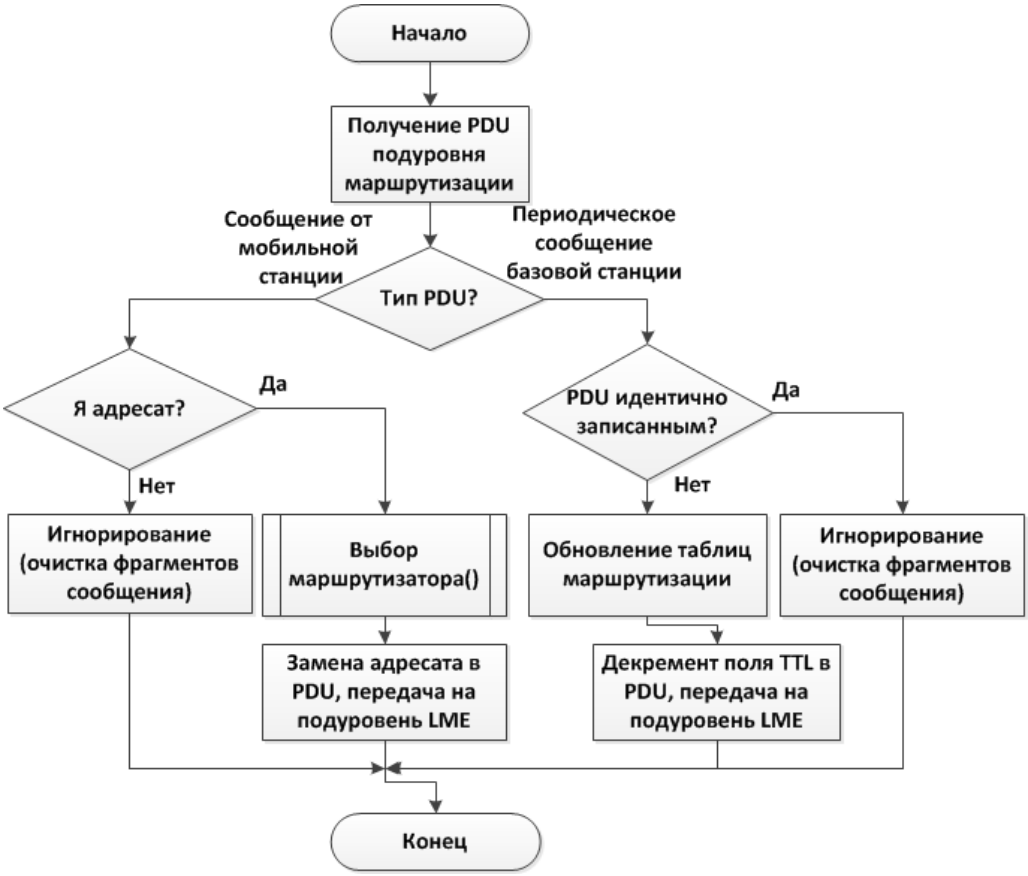
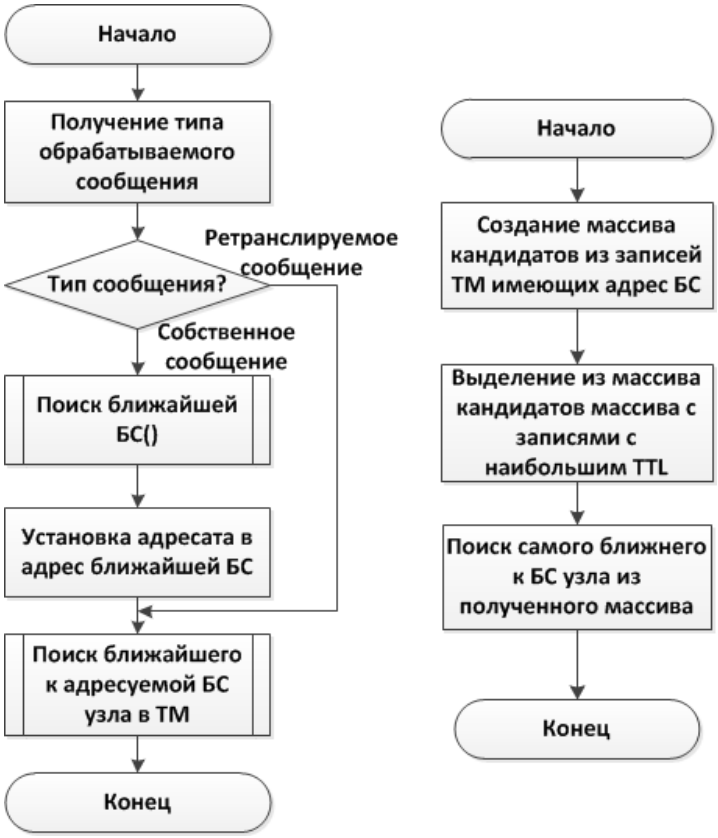


Рисунок 25. Упрощенная блок-схема обработки сетевого сообщения на подуровне маршрутизации МС



а) б)

Рисунок 26. Упрощенные блок-схемы: а) процесс выбора маршрутизатора; б) поиск ближайшего узла

Реализация всех этапов маршрутизации при помощи одних лишь «однослотовых» сообщений потребовала особых подходов для размещения и извлечения служебной информации. На рисунке 27 изображены блоки информации, относящиеся к различным подуровням VDL Mode 4, обрабатываемые мобильными станциями и значения, подвергающиеся изменениям. По мере ретрансляции сообщения от БС, на каждом промежуточном узле производится уменьшение времени жизни сообщения на единицу. Также, каждый из промежуточных узлов меняет адрес источника сообщения на собственный, таким образом, следующий промежуточный узел записывает соотношение «узел-базовая станция» и определяет из значения времени жизни, сколько интервалов прошло сообщение. Сама же информация о БС записывается согласно данным, указанным в части сообщения с координатами. С этими координатами связывается адрес БС на подуровне маршрутизации.

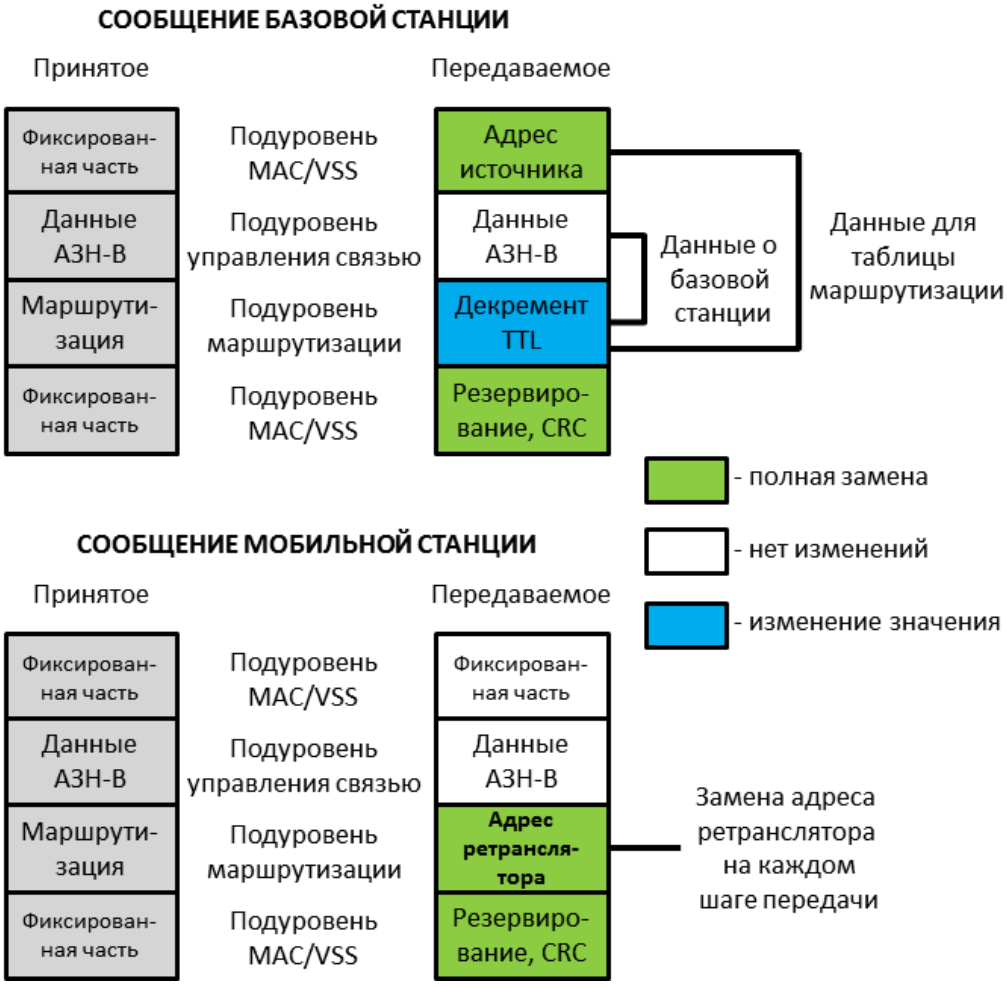


Рисунок 27. Изменение полей сообщения при его обработке на узлах

При передаче по сети сообщения от МС, каждый промежуточный узел устанавливает лишь адрес следующего ретранслятора, выбранного им согласно информации в таблице маршрутизации. Во всех случаях, при каждой ретрансляции узел устанавливает собственную информацию о резервировании слотов, т.к. использует собственную карту слотов и процедуры резервирования.

Сетевые сообщения на узлах должны быть обработаны на всех рассмотренных подуровнях, при этом некоторые части должны оставаться неизменными, поэтому возникла задача увязывания фрагментов сообщения на подуровнях. Для её решения была использована единая временная метка, которая назначается фрагментам, что проиллюстрировано на рисунке 28. Уникальность временной метки обеспечивается уникальностью времени приёма сообщения.

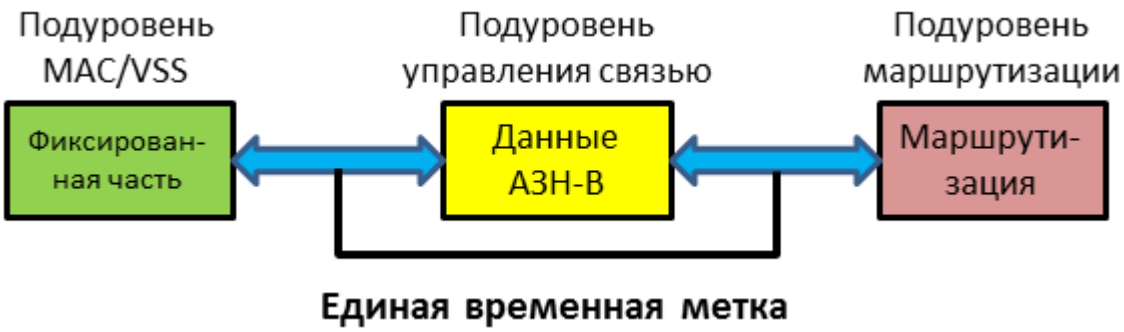


Рисунок 28. Связывание фрагментов сетевых сообщений на различных подуровнях единой временной меткой

Для управления производительностью сети в работе предложены несколько изменяемых числовых параметров, значения которых исследованы в процессе моделирования. Параметр резервирования позволяет приложению выбирать один из типов фиксированного доступа к среде, балансируя между вероятностью коллизии и временем доступа. Период вещания сетевых сообщений и хранения записей в таблице маршрутизации БС, необходим для поддержания актуальной информации на узлах с учётом связности сети и информационной нагрузки.

4.4 Выводы

1. В разделе предложен метод построения таблицы маршрутизации с помощью периодического вещания и ретрансляции по сети данных с идентификаторами и местоположением сетевых узлов. Данный подход может вызывать передачу больших объёмов служебной информации и резкое увеличение кол-ва необходимых слотов для её передачи, поэтому предложен метод с периодическим вещанием сообщений только от базовых станций. Каждый узел записывает информацию о соседних узлах, от которых он получил информацию о БС, тем самым формируя таблицы маршрутизации. Для выбора маршрутизаторов использован «жадный» алгоритм, использующий в качестве метрики расстояние от узла до БС.

2. По результатам анализа алгебраической системы, основанной на связном направленном графе и имеющей четыре кортежа (множество сигнатур, множество потоков трафика, весовая функция связи и порядок отношения) установлено, что «жадный» алгоритм, совместно с проактивным методом построения таблиц маршрутизации, даёт протокол, который не образует петель при передаче сообщений и даёт решение, когда существует непустое множество выбора. Для разрешения проблемы пустого множества выбора

используется число интервалов, которое прошло сообщение БС, также хранящееся в таблице маршрутизации.

3. В целях минимизации трафика при передаче сетевых сообщений были использованы «однослотовые» сообщения типа «synchronization burst» с информацией о маршрутизации, хранящейся в переменной информационной части сообщения.

4. Для обработки фрагментов сетевых сообщений производится их связывание по уникальному адресу ICAO и временной метке. Разработанный протокол маршрутизации позволяет минимизировать кол-во служебной информации передаваемой по сети в 3 раза по сравнению с методами передачи, предусмотренными в стандарте VDL Mode 4, а значит уменьшить влияние на пропускную способность канала и обеспечить сетевой обмен с малыми энергозатратами, что актуально для беспилотных летательных аппаратов.

РАЗДЕЛ 5. ИССЛЕДОВАНИЕ ПОКАЗАТЕЛЕЙ ПРОИЗВОДИТЕЛЬНОСТИ СЕТИ. ВОЗМОЖНЫЕ ТЕХНИЧЕСКИЕ РЕАЛИЗАЦИИ

5.1 Моделирование сети и анализ показателей

Изначально в работе идёт речь о развертывании самоорганизующейся Ad Hoc сети в отдаленных и океанических регионах, где плотность пунктов управления воздушным движением (БС) низкая, а ВС входят на некоторое время в зоны процедурных полетов, поэтому стоит уточнить сетевые топологии именно для этого случая. В отдаленных и океанических регионах движение ВС можно охарактеризовать как «коридороподобное», т.е. обычно ВС движутся из одного конца маршрута в другой, друг за другом, либо навстречу друг друга, с большим разнесением. Над удаленными регионами для самолетов гражданской авиации продольное эшелонирование может доходить до значения около 100 км, а над океаническими регионами и до 200 км. Обычно высота полета в таких регионах может колебаться от 10 км до 12 км. Скорость движения ВС, в зависимости от их маршрута и возможностей двигателей, обычно варьируется в районе 800 км/ч до 1000 км/ч. На таких высотах ландшафт земли обычно не играет особой роли, поэтому в обычном случае дальность радиовидимости, с небольшими допущениями, ограничивается радиогоризонтом, мощностью передатчика и чувствительностью приемника. По документам VDL Mode 4 дальность радиовидимости должна составлять около 200 морских миль, что приблизительно равно 370 км.

Для того чтобы изучить возможности полученного протокола был рассмотрен сценарий, где 50 узлов двигаются с рассмотренным разбросом по разнесению и скорости. Общий поток движения имеет форму креста, в центре которого располагается базовая станция. Также, присутствует и 20 % узлов, двигающихся вне креста. Территория имеет размеры 3200 км x 3200 км, т.е. квадрат площадью порядка 10 млн. км^2. Согласно исследованиям, проведенным в разделе 2, при равномерном распределении узлов связность стремилась бы к нулю, однако, благодаря выбранному сценарию движения она намного выше.

На рисунках 29, 30, 31, 32 изображены показатели производительности сети по рассмотренному сценарию, при варьировании параметра диапазона выбора временного слота: отношение полученных к переданным сообщениям, общее число переданных сообщений, усредненное время задержек при передаче сообщений по сети, максимальные задержки.

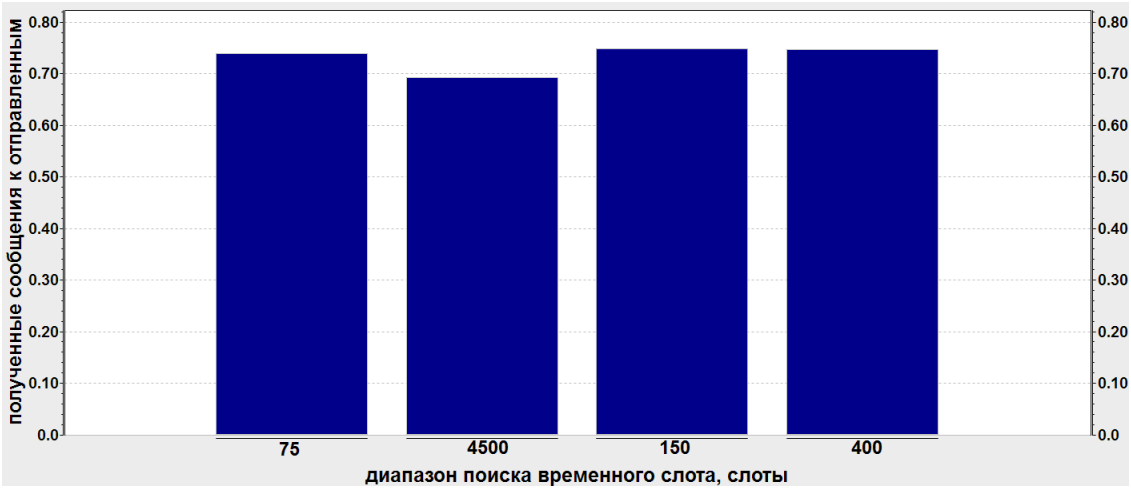


Рисунок 29. Отношение полученных к переданным сообщениям при варьировании параметра «диапазон выбора временного слота»

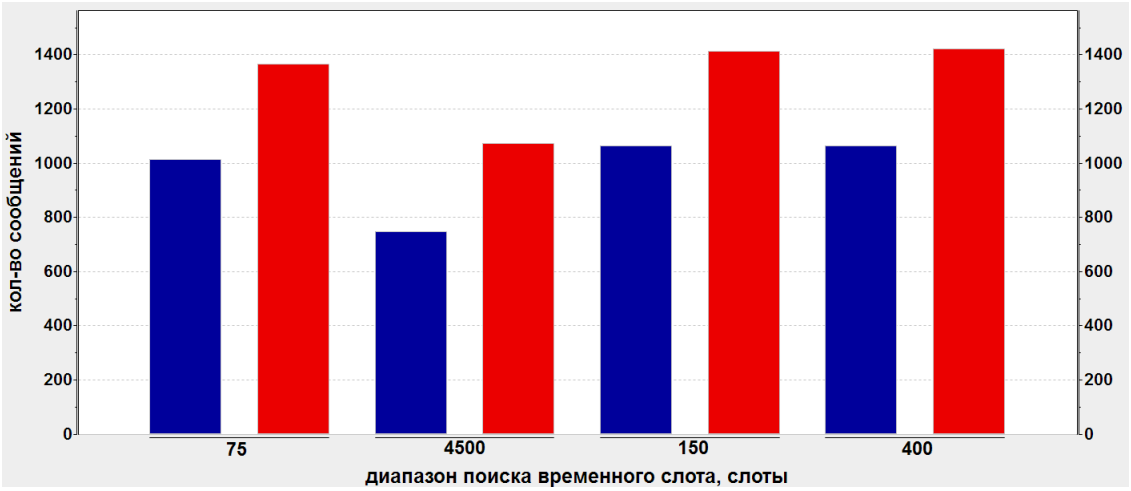


Рисунок 30. Кол-во полученных (синим) и переданных (красным) узлами сетевых АЗН-В-сообщений при варьировании параметра «диапазон выбора временного слота»

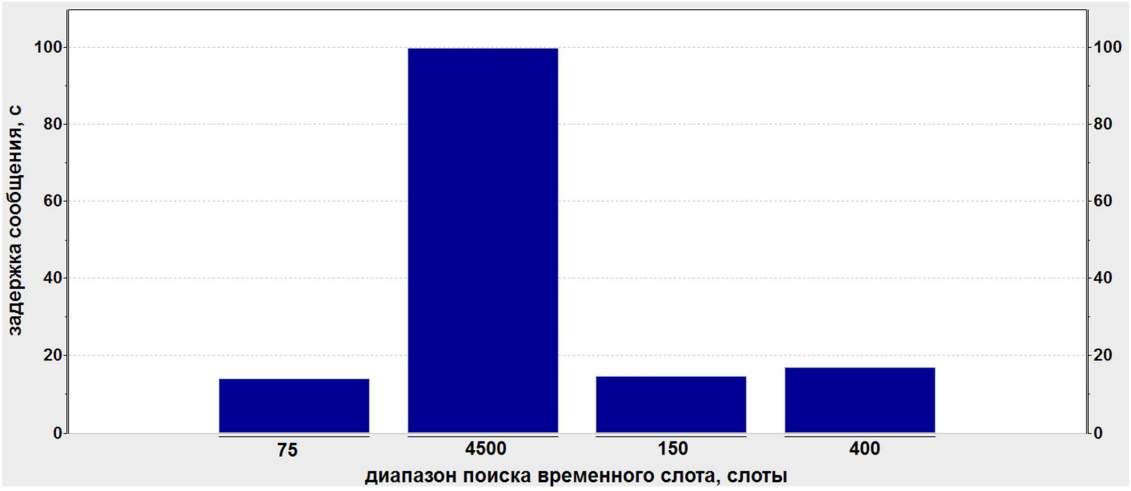


Рисунок 31. Усредненное время задержки при передаче сообщения от узла до БС при варьировании параметра «диапазон выбора временного слота»

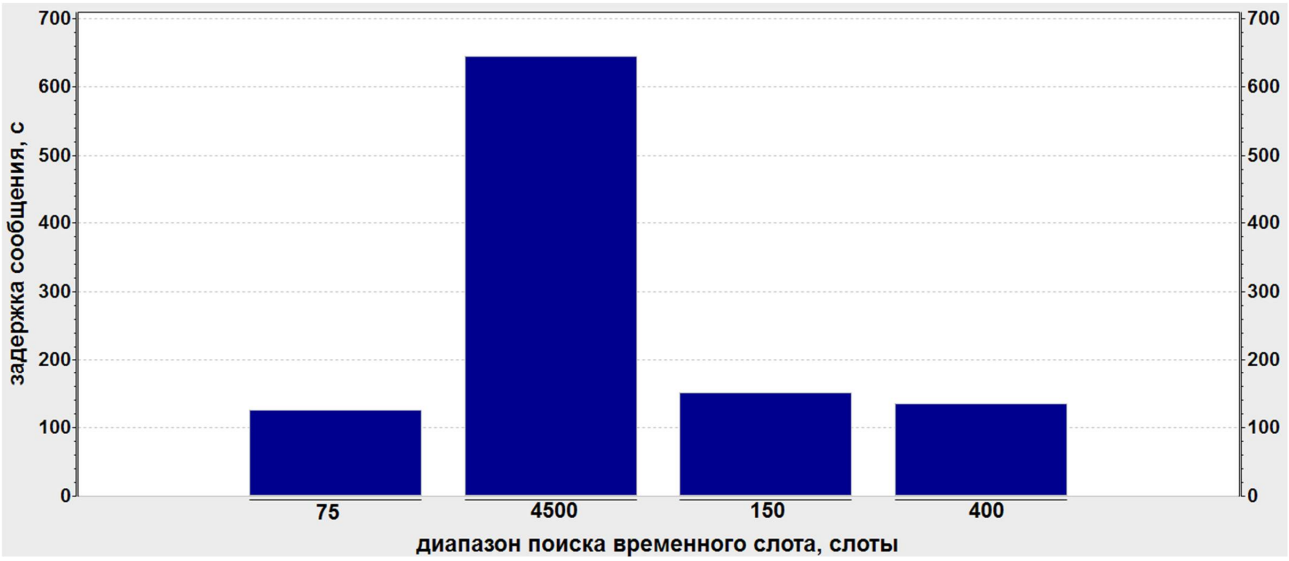


Рисунок 32. Усредненное максимальное время задержки при передаче сообщения от узла до БС при варьировании параметра «диапазон выбора временного слота»

В таблице 20 приведены основные результаты моделирования рассмотренного сценария при варьровании диапазона выбора временного слота.

Таблица 20. Показатели производительности сети с 50ю узлами при варьировании диапазона выбора временного слота

Диапазон выбора временного слота, слоты	Отношение кол-ва полученных сообщений к отправленным, %	Задрежка, с	Максимальная задержка, с	Получено сообщений
75	74%	**14,1 с**	**125,9 с**	1012
4500	70%	99,8 с	645,2 с	748
150	**75%**	14,7 с	150,5 с	1063
400	**75%**	17 с	135,3 с	**1064**

При значении диапазона выбора временного слота равного 75 слотам было достигнуто наилучшее значение по задержкам передаваемых сообщений. При значении параметра равным 400 слотам получено наилучшее значение по количеству полученных сообщений и их отношению к отправленным. Это объясняется тем, что при меньших значениях диапазона выбора слота узлы чаще размещают резервирования, т.е. происходит более быстрый доступ к среде передачи. Однако подобные резервирования размещаются без предупреждения для других узлов, что приводит к увеличению коллизий, результатом чего является меньшее количество принятых сообщений относительно больших значений параметра. Самый худшие показатели были получены для параметра равного 4500 слотам, означающего поиск слота для резервирования по всей карте слотов, т.е. в карте слотов

выбирается подходящий для инкрементированной передачи слот, находящийся далеко относительно текущей позиции слота, что резко увеличивает период доступа к среде, а, следовательно, задержки. Также это сказывается на вероятности устаревания информации для маршрутизации – уменьшение кол-ва принятых сообщений относительно других исследованных значений параметра.

На рисунках 33, 34, 35, 36 изображены показатели производительности сети по рассмотренному сценарию, при варьировании параметра «период вещания сетевых сообщений базовой станцией».

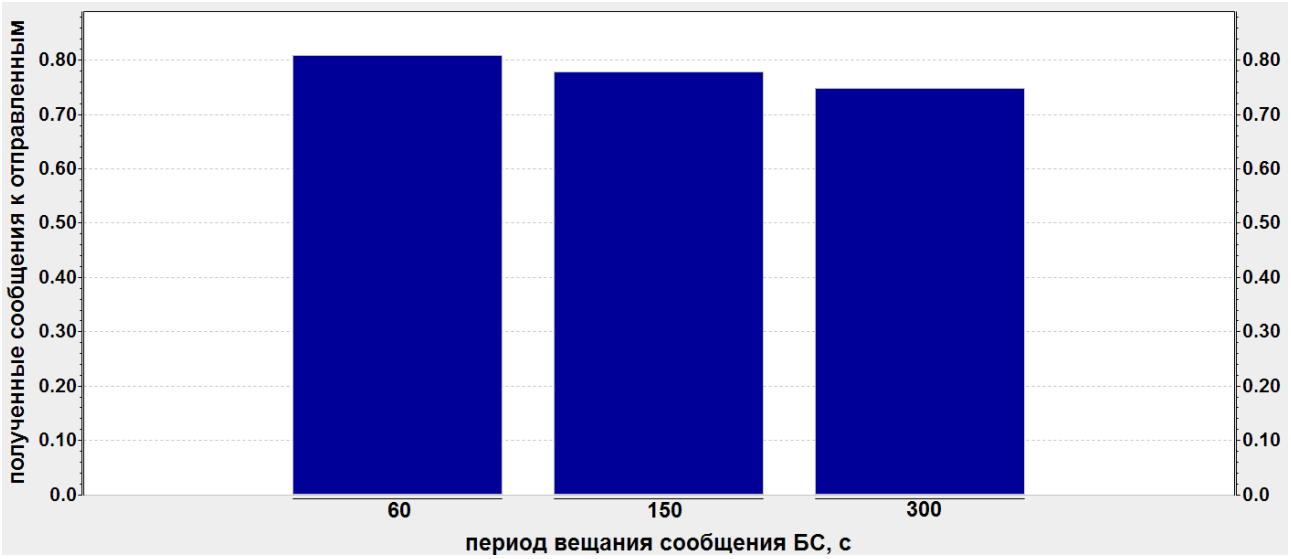


Рисунок 33. Отношение полученных к переданным сообщениям при варьировании параметра «период вещания сетевых сообщений базовой станцией»

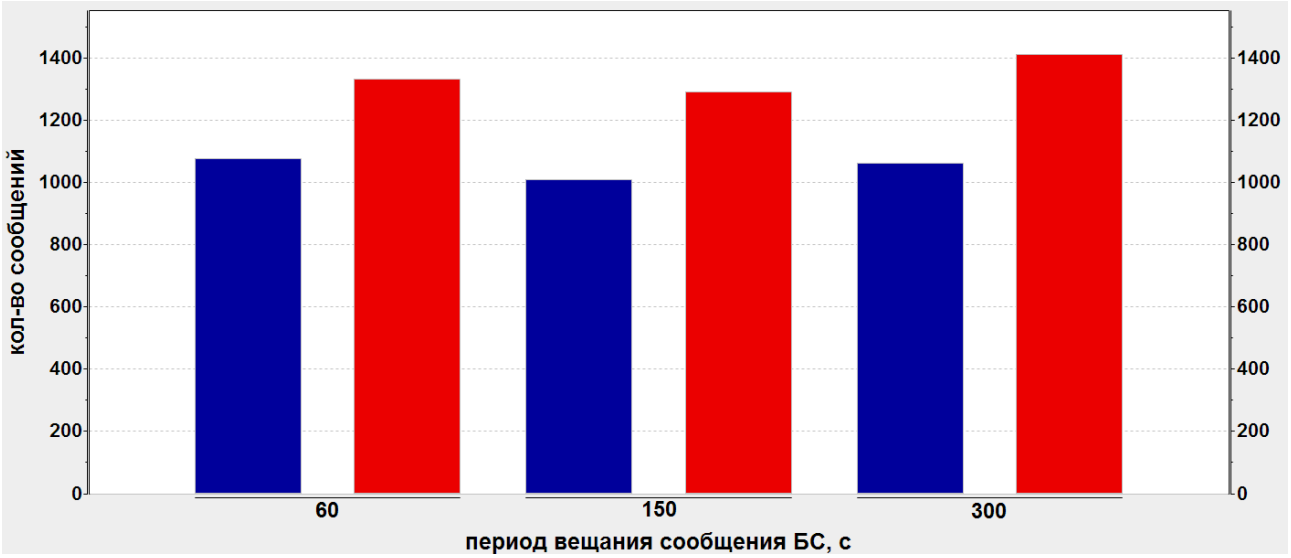


Рисунок 34. Всего получено (синим) и отправлено (красным) сообщений при варьировании параметра «период вещания сетевых сообщений базовой станцией»

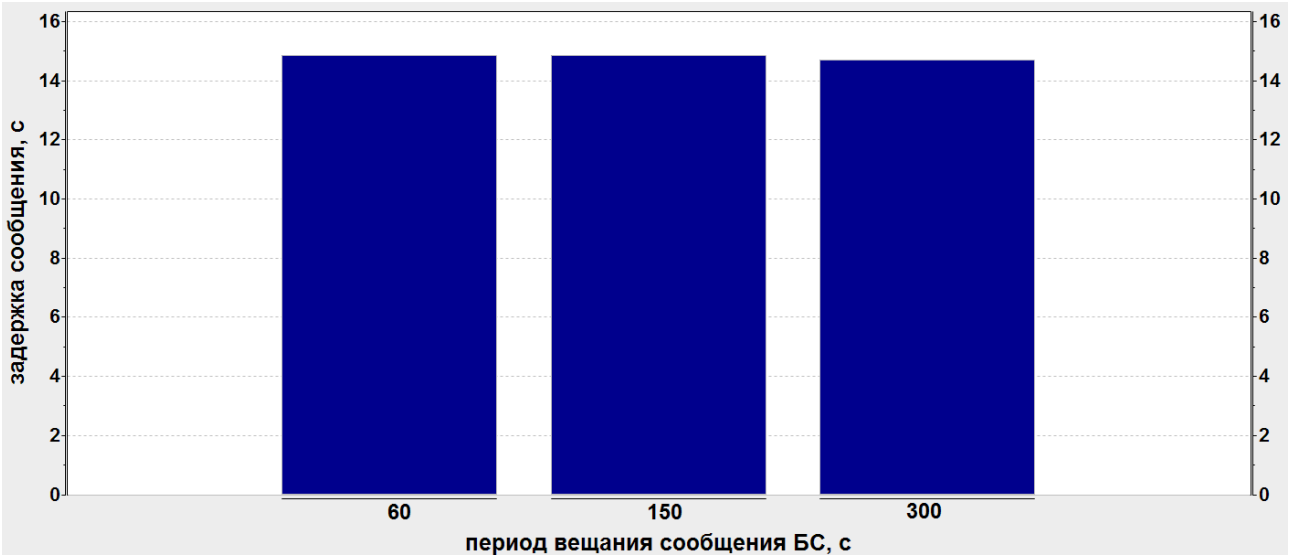


Рисунок 35. Усредненное время задержки при передаче сообщения от узла до БС при варьировании параметра «период вещания сетевых сообщений базовой станцией»

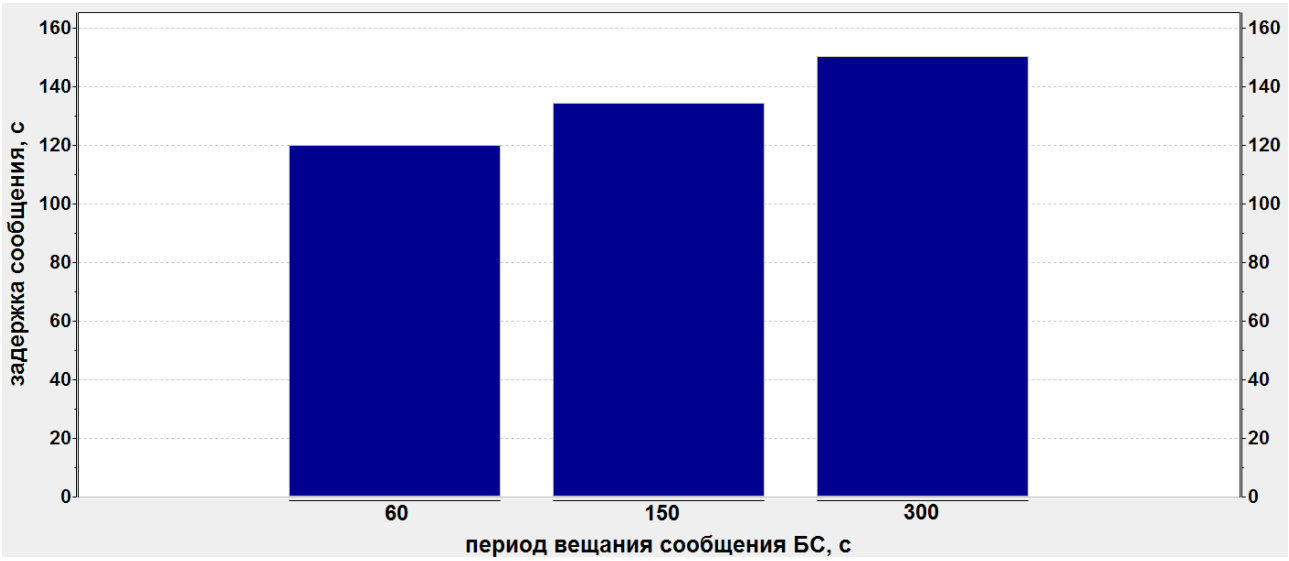


Рисунок 36. Усредненное максимальное время задержки при передаче сообщения от узла до БС при варьировании параметра «период вещания сетевых сообщений базовой станцией»

В таблице 21 приведены результаты моделирования рассмотренного сценария при варьировании периода вещания сетевых сообщений базовой станцией.

Таблица 21. Показатели производительности сети с 50ю узлами при варьировании периода вещания сетевых сообщений базовой станцией

Период вещания БС, с	Отношение кол-ва полученных сообщений к отправленным, %	Задрежка, с	Максимальная задержка, с	Получено сообщений
60 с	**81%**	14,9 с	**119,9 с**	**1078**
150 с	78%	14,9 с	134,5 с	1008
300 с	75%	**14,7 с**	150,5 с	1062

При периоде вещания сетевых сообщений БС равным 60 с достигаются наилучшие показатели производительности по пиковым задержкам, кол-ву полученных сообщений и их отношению к кол-ву отправленных сообщений. Это связано с тем, что при более частом вещании сетевых сообщений базовой станцией узлы чаще обновляют таблицы маршрутизации, т.е. дальние узлы отправляют меньше сообщений по «неактуальным» маршрутам. При значении параметра величиной в 300 с снижается актуальность информации для маршрутизации на узлах, что выражается в снижении показателей производительности сети. При этом стоит отметить, что величина средних задержек во всех случаях почти не отличается.

На рисунках 37, 38, 39, 40 изображены показатели производительности сети по рассмотренному сценарию, при варьировании периода хранения записей в таблице маршрутизации и таблице базовых станций.

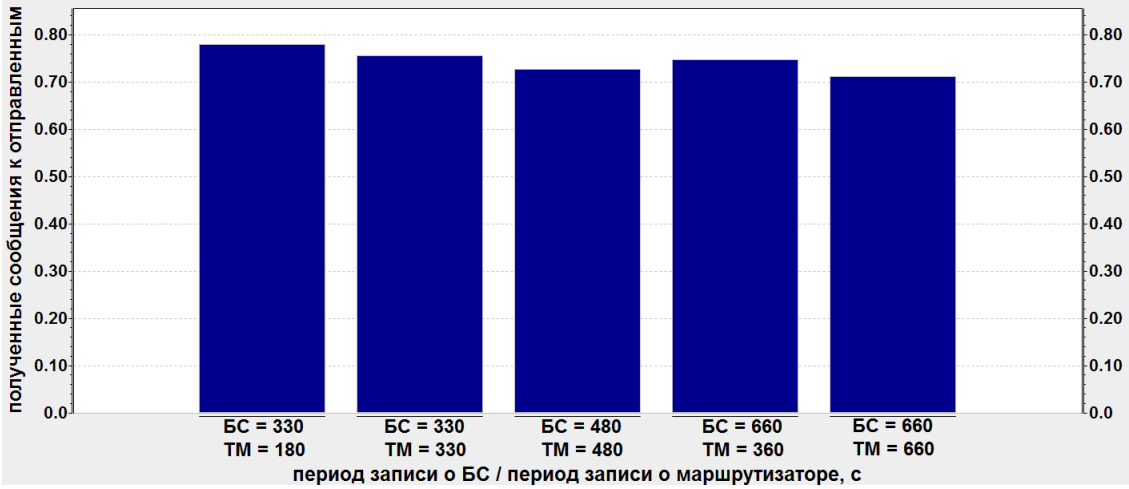


Рисунок 37. Отношение полученных к переданным сообщениям при варьировании периода хранения записей в таблице маршрутизации и таблице базовых станций

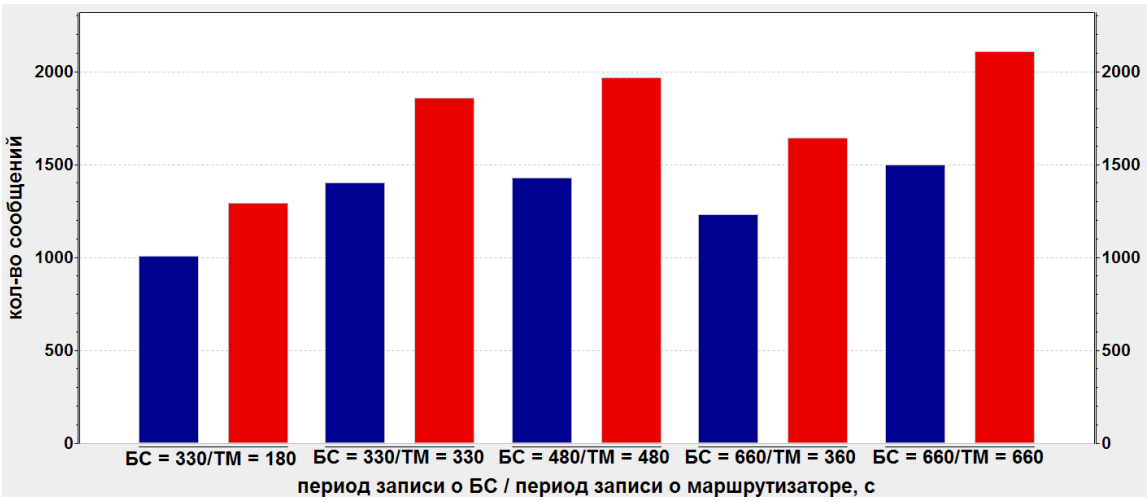


Рисунок 38. Всего получено (синим) и отправлено (красным) сообщений при варьировании периода хранения записей в таблице маршрутизации и таблице базовых станций

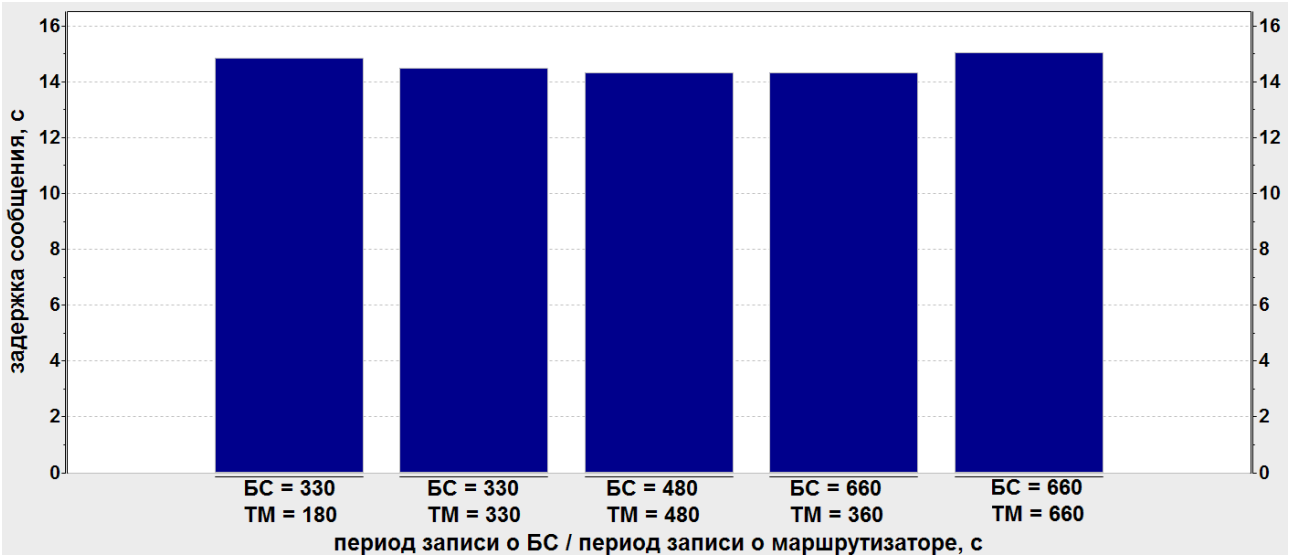


Рисунок 39. Усредненное время задержки при передаче сообщения от узла до БС при варьировании периода хранения записей в таблице маршрутизации и таблице базовых станций

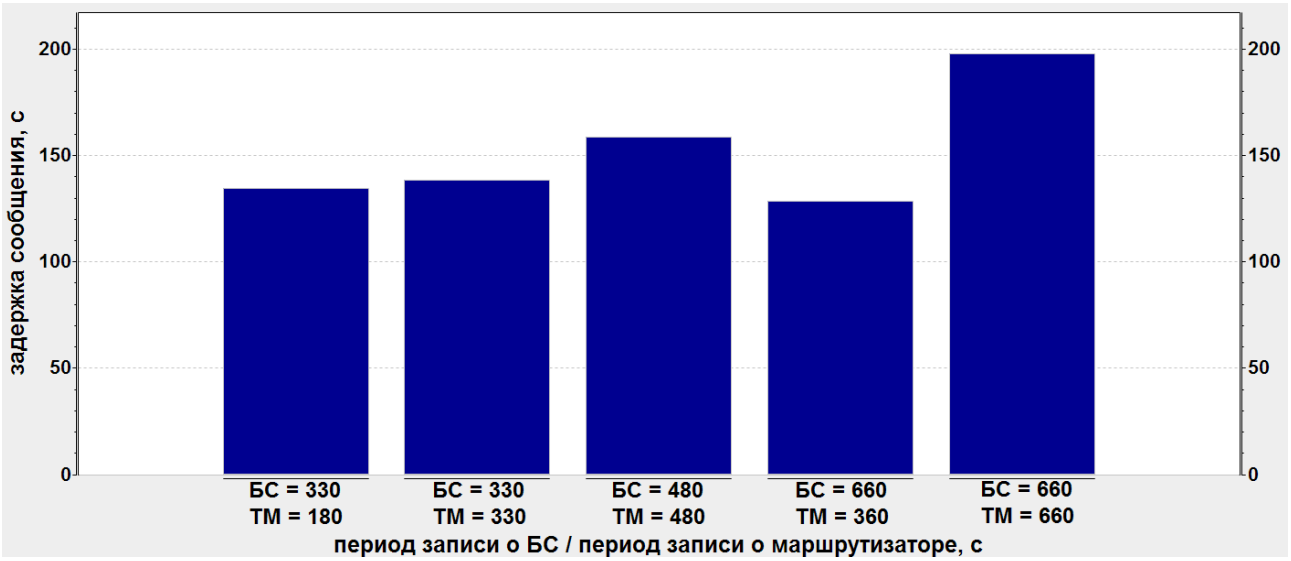


Рисунок 40. Усредненное максимальное время задержки при передаче сообщения от узла до БС при варьировании периода хранения записей в таблице маршрутизации и таблице базовых станций

В таблице 22 приведены результаты моделирования рассмотренного сценария при варьировании периода вещания сетевых сообщений базовой станцией.

Таблица 22. Показатели производительности сети с 50ю узлами при варьировании периода вещания сетевых сообщений базовой станцией

Период хранения БС/ТМ, с	Отношение кол-ва полученных сообщений к отправленным, %	Задрежка, с	Максимальная задержка, с	Получено сообщений
330 с / 180 с	**78%**	14,9 с	134,5 с	1009
330 с / 330 с	76%	14,5 с	138,4 с	1403
480 с / 480 с	73%	**14,3 с**	158,6 с	1428
660 с / 360 с	75%	**14,3 с**	**128,6 с**	1232
660 с / 660 с	71%	15 с	197,6 с	**1498**

При коротком периоде хранения информации на промежуточных узлах, в условиях плохой связности, может возникать ситуация с преждевременным удалением всё ещё актуальной информации о маршрутизаторах, что сказывается на общем числе отправляемых сообщений. Это уменьшило информационную нагрузку в случае значения параметра равного 330 с / 180 с и повысило отношение полученных сообщений к отправленным относительно других исследованных значений. Однако, при длительном периоде узлы продолжают отправлять сообщения, когда актуальность информации о маршрутизации была утеряна. Например, для периода 660с для двух записей, общее кол-во полученных сообщений было максимальным при самом низком соотношении полученных к отправленным сообщениям. Полученные задержки были меньше для промежуточных значений, т.е. можно сделать вывод о необходимости применения адаптивных значений периодов хранения информации о маршрутах для сети.

На рисунках 41, 42, 43 и 44 изображены показатели производительности сети по рассмотренному сценарию, при варьировании кол-ва сообщений АЗН-В в минуту, передаваемых широковещательно (RR – report rate) и передаваемых по сети (NRR – network report rate).

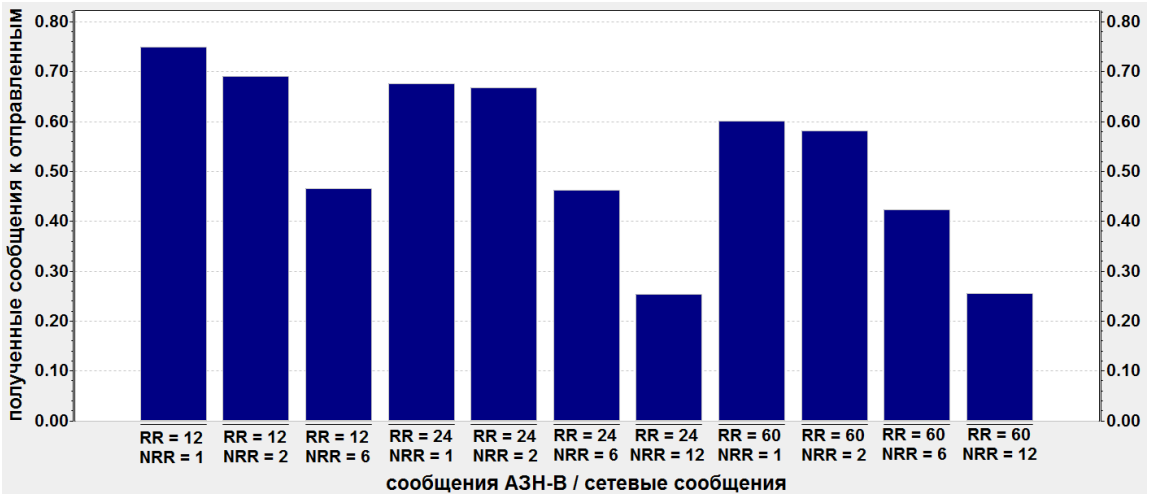


Рисунок 41. Отношение полученных к переданным сообщениям при варьировании кол-ва сообщений АЗН-В в минуту, передаваемых широковещательно и передаваемых по сети

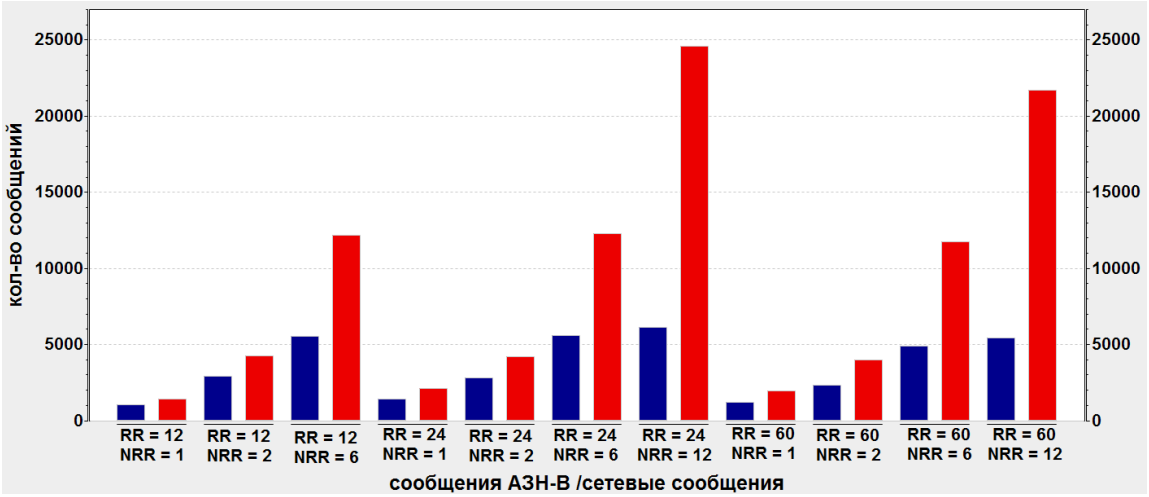


Рисунок 42. Всего получено (синим) и отправлено (красным) сообщений при варьировании кол-ва сообщений АЗН-В в минуту, передаваемых широковещательно и передаваемых по сети

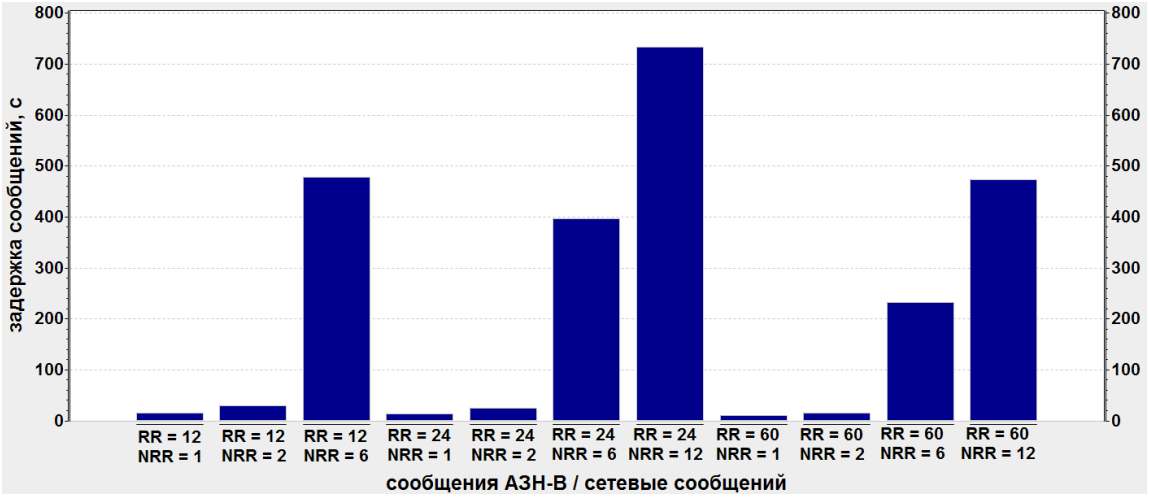


Рисунок 43. Усредненное время задержки при передаче сообщения от узла до БС при варьировании кол-ва сообщений АЗН-В в минуту, передаваемых широковещательно и передаваемых по сети

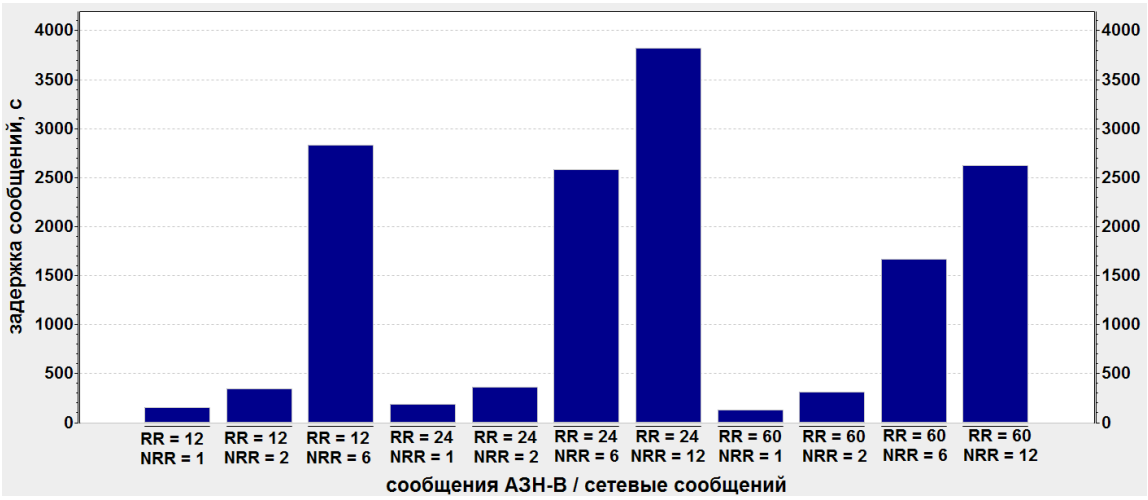


Рисунок 44. Усредненное максимальное время задержки при варьировании кол-ва сообщений АЗН-В в минуту, передаваемых широковещательно и передаваемых по сети

В таблице 23 приведены показатели производительности сети по рассмотренному сценарию при варьировании кол-ва сообщений АЗН-В в минуту, передаваемых широковещательно и передаваемых по сети.

Таблица 23. Показатели производительности сети с 50ю узлами при варьировании кол-ва сообщений АЗН-В в минуту, передаваемых широковещательно и передаваемых по сети

RR/NRR, сообщений в минуту	Отношение кол-ва полученных сообщений к отправленным, %	Задрежка, с	Максимальная задержка, с	Получено сообщений
12/1	**75%**	14,7 с	150,5 с	1063
12/2	69%	29,5 с	341,9 с	2931
12/6	47%	478,1 с	2834,8 с	5545
24/1	68%	13,3 с	186,9 с	1431
24/2	67%	24,4 с	358,2 с	2801
24/6	46%	396,4 с	2582,1 с	5586
24/12	25%	732,8 с	3817,8 с	**6131**
60/1	60%	**10,5 с**	**131,7 с**	1181
60/2	58%	14,8 с	312,5 с	2307
60/6	42%	231,6 с	1665,7 с	4919
60/12	25%	473,4 с	2624,2 с	5430

Для рассмотренного параметра отношение полученных сообщений к отправленным снижается, как при увеличении кол-ва сообщений АЗН-В, передаваемых в прямой видимости, так и при увеличении кол-ва сообщений, передаваемых по сети. В данном случае сказывается увеличение информационной нагрузки на канал передачи данных,

увеличиваются потери сообщений. Стоит отметить важный момент, в сценарии с установленным значением параметра 60/1 (RR/NRR) задержки были минимальными – это связано с особенностями выбранного доступа к среде. При высокой частоте периодических широковещательных резервирований (24 и 60), размещаемых узлами, увеличивается и количество комбинированных инкрементированных резервирований, используемых для сетевых сообщений – это положительно сказывается на кол-ве полученных сообщений и их задержках. Для значений кол-ва сетевых сообщений 6 и 12 максимальные задержки составили десятки минут, увеличилось и количество полученных сообщений, однако, не пропорционально эксперименту с 1 сетевым сообщением в минуту.

На рисунках 45, 46, 47 и 48 изображены показатели производительности сети по рассмотренному сценарию, при варьировании параметра периодического смещения на подуровне доступа к среде.

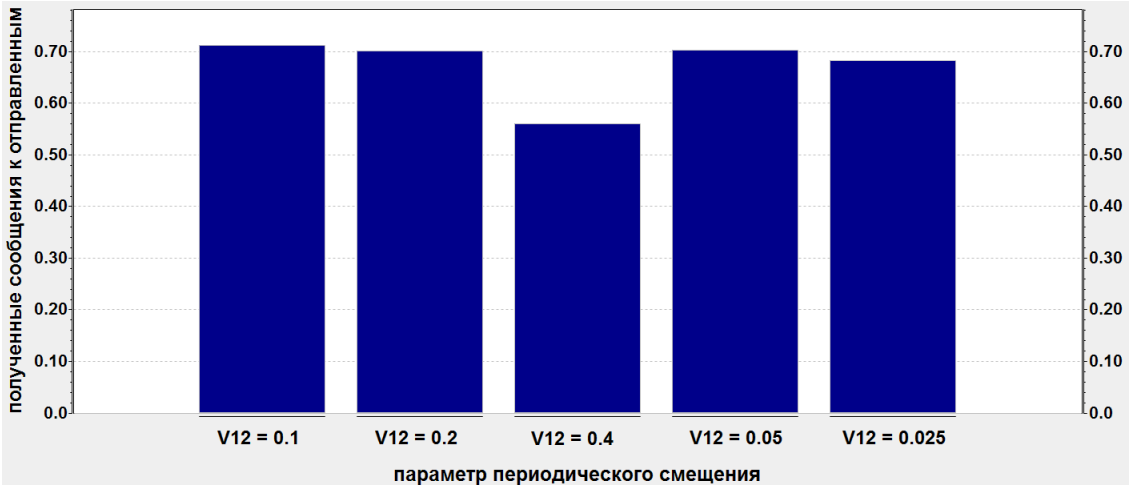


Рисунок 45. Отношение полученных к переданным сообщениям при варьировании параметра периодического смещения на подуровне доступа к среде

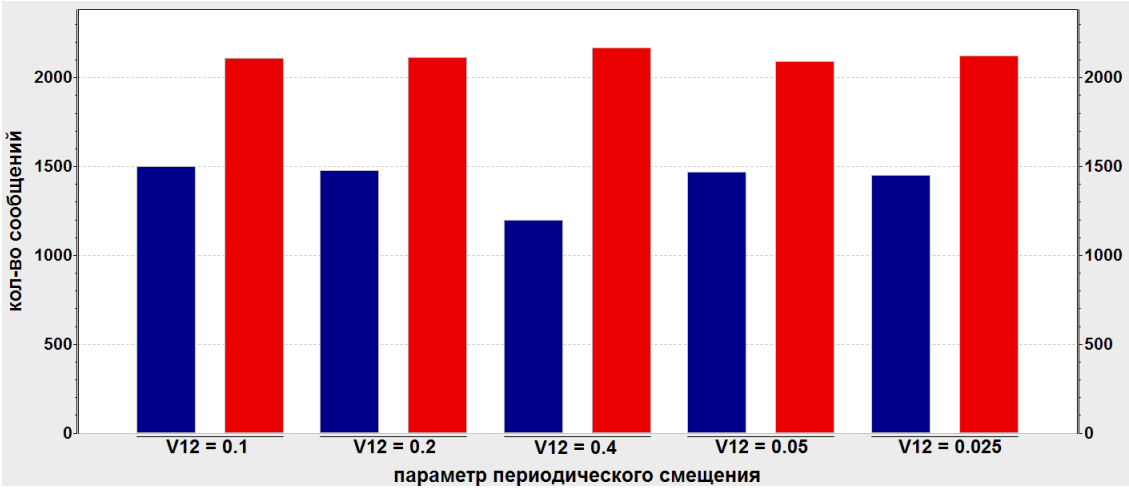


Рисунок 46. Всего получено (синим) и отправлено (красным) сообщений при варьировании параметра периодического смещения на подуровне доступа к среде

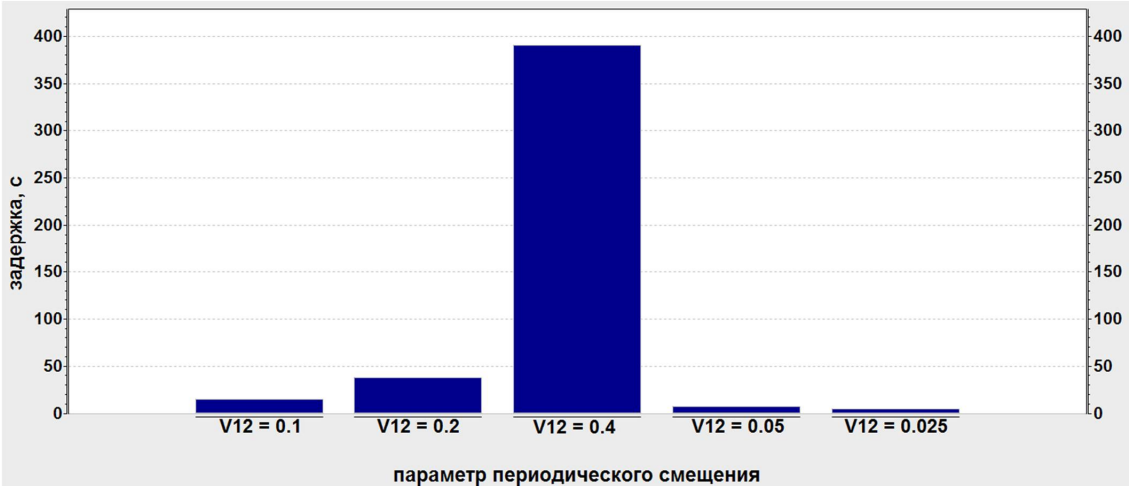


Рисунок 47. Усредненное время задержки при передаче сообщения от узла до БС при варьировании параметра периодического смещения на подуровне доступа к среде

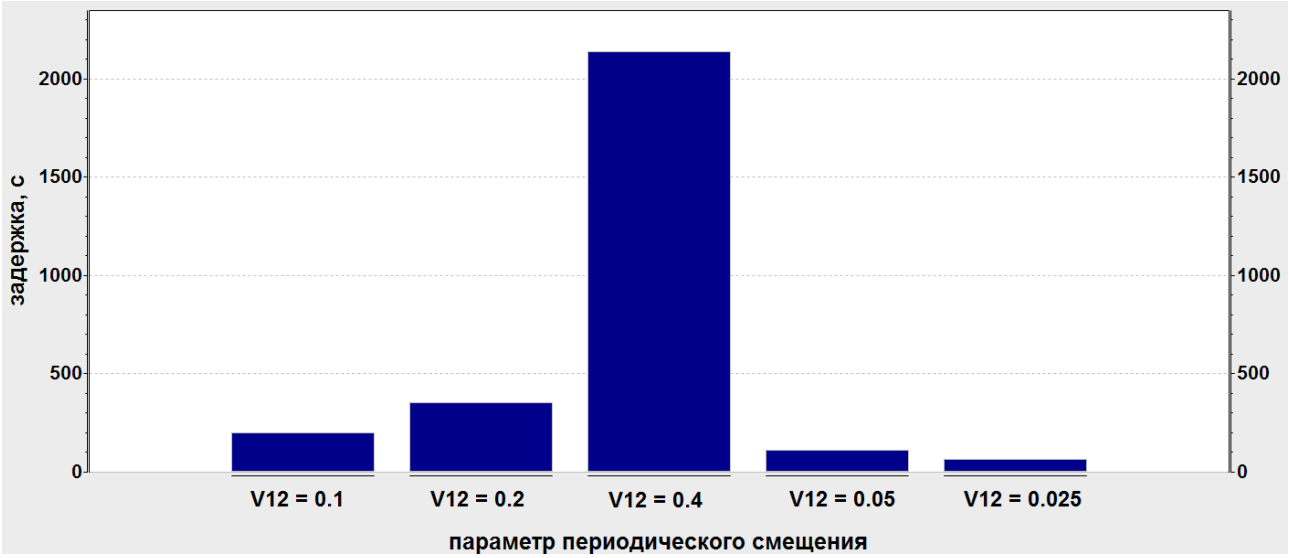


Рисунок 48. Усредненное максимальное время задержки при варьировании параметра периодического смещения на подуровне доступа к среде

В таблице 24 приведены показатели производительности сети с 50ю узлами при варьировании параметра периодического смещения на подуровне доступа к среде.

Таблица 24. Показатели производительности сети с 50ю узлами при варьировании параметра периодического смещения на подуровне доступа к среде

Параметр периодичес-кого смещения	Отношение кол-ва полученных сообщений к отправленным, %	Задрежка, с	Максимальная задержка, с	Получено сообщений
0,1	**71%**	15 с	197,6 с	1500
0,2	70%	38 с	352,5 с	1476
0,4	56%	390,1 с	2137 с	1197
0,05	70%	7,6 с	110,5 с	1467
0,025	68%	**4,4 с**	**63,4 с**	1448

Параметр периодического смещения определяет диапазон слотов-кандидатов по обе стороны от номинального слота для размещения широковещательного периодического резервирования. При значении параметра равным 0,025 (наименьшем из рассмотренных) достигнуты наименьшие величины задержек сетевых сообщений. Это связано с предложенным механизмом резервирования слотов для передачи сетевых сообщений – когда в процессе размещения резервирования узел не находит подходящих слотов для использования комбинированного инкрементированного и широковещательного резервирования, он использует процедуры размещения нового широковещательного резервирования. При больших значениях рассматриваемого параметра увеличивается и диапазон слотов-кандидатов, в результате, для передачи сетевого сообщения может быть выбран слот, отстоящий далеко от текущего, что негативно отражается на задержках.

Стоит также отметить, что кол-во узлов, от которых базовой станцией были получены сообщения АЗН-В по сети, для всех исследованных параметров составило 44±1 узла, т.е. от большей части узлов удалось получить хотя бы одно сообщение, не смотря на низкую связность сети.

Для сценария с двумя базовыми станциями, для каждой базовой станции, в среднем были получены следующие значения: отношение кол-ва принятых сообщений к отправленным – 80%, пиковое время задержки – 118,1 с, усредненное время задержки – 7,2 с, получено сообщений каждой БС в среднем – 1487. Таким образом, суммарно для всей сети было получено сообщений около 3000 сообщений от 48 ВС, что соответственно в 2 раза больше, чем в сценарии с одной базовой станцией, однако, остальные показатели остались прежними т.к. связность сети практически не изменилась.

В сценарии со 150ю узлами, при параметрах модели одинаковых для конфигурации с периодом хранения записей в таблице БС и таблице маршрутизации по 660 с соответственно, наблюдалось: уменьшение отношения полученных сетевых сообщений к отправленным до значения равного 66%, увеличение времени задержек до 66,8 с, максимальных задержек до 1509,3 с. Ухудшение указанных показателей связано с возрастанием информационной нагрузки в 3 раза. Однако кол-во полученных сообщений составило 7298, что в 4,87 раз больше, чем в сценарии с 50ю узлами. Данные результаты можно объяснить увеличением связности сети, т.к. кол-во узлов было больше в 3 раза на одной и той же территории.

Для исследования производительности сети также был рассмотрен сценарий с территорией Дальнего Востока площадью 4,8 млн. кв. км, который охватывает 4 аэропорта, в которых потенциально могут быть расположены БС: Петропавловск-Камчатский, Анадырь, Магадан и Оха. Полетные данные были взяты с сайта flightradar24.com, при этом в течение

дня рассматривалось 13 промежутков по 50 мин., в которых суммарно было зафиксировано 58 бортов. Пиковое число узлов в один рассматриваемый период составило 13 шт. В результате, лучшие показатели были получены для меньших, относительно предыдущего сценария, значений периодов хранения информации для маршрутизации величиной в 480с (вместо 660с). Таким образом, для лучшего случая, были получены следующие значения: отношение кол-ва принятых сообщений к отправленным – 61%, пиковое время задержки – 14,4 с, усредненное время задержки – 6,2 с. При этом суммарно было отслежено 48 бортов, а в одном из рассматриваемых промежутков отношение числа принятых сообщений к отправленным составило 0,99, т.е. в течение 50 мин. существовала полностью связная структура, а некоторые сообщения были потеряны только из-за битовых ошибок, вероятность возникновения которых была установлена в значение 10^{-4} согласно технической документации VDL Mode 4.

Также в работе был исследован характер нагрузки на пропускную способность сети на основе VDL Mode 4. При варьировании числа узлов в сети и частоты вещания сообщений АЗН-В устанавливались предельные значения, в результате которых в какой-либо момент времени на любом узле все слоты оказывались занятыми, т.е. не оставалось незарезервированных слотов. На рисунке 49 изображен случай для 30 сообщений АЗН-В в минуту, т.к. это число является наибольшим для одного глобального канала.

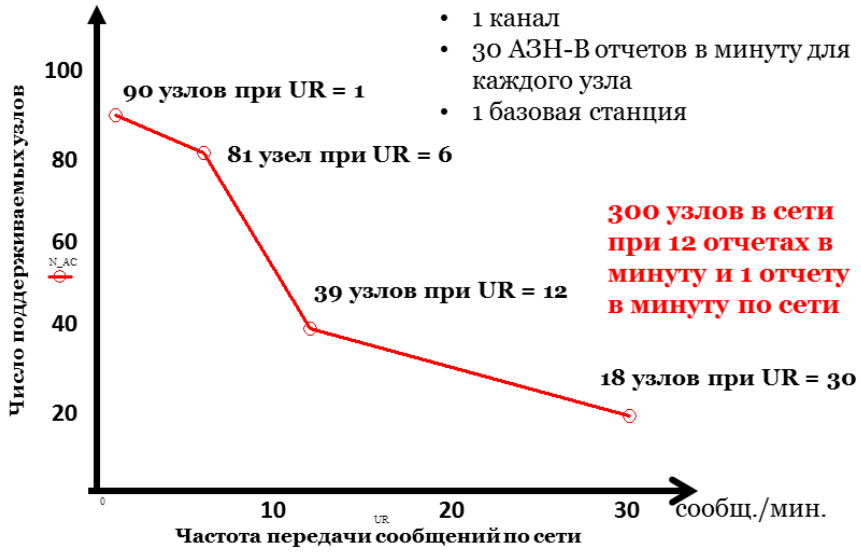


Рисунок 49. Число поддерживаемых узлов в сети в зависимости от частоты вещания сообщений АЗН-В

Максимально поддерживаемое число узлов равное 300м было получено при 12ти сообщениях АЗН-В в минуту и 1м сообщении АЗН-В в минуту отправляемом по сети.

Однако, такой сценарий нереализуем на территории РФ – в отдаленных и океанических регионах не скапливается такого кол-ва узлов при условии существования одной (и даже двух) БС. При этом можно сделать вывод о существовании запаса по пропускной способности для будущего увеличения числа ВС.

В результате можно заключить, что внедрение самоорганизующихся сетей на основе авиационного стандарта связи несёт огромный потенциал для безопасности воздушного движения. Это даст возможность отслеживать десятки воздушных бортов в реальном времени, следующих по маршрутам в зонах без наблюдения с земли. Основным влияющим фактором является связность сети. Построение сети связи также позволит внедрять новые авиационные приложения, основанные на многоинтервальной передаче данных. ICAO не выставляет требований к производительности приложений на основе самоорганизующихся сетей, поэтому проведенные исследования могут быть их основой.

5.2 Комплекс полунатурного моделирования

Стенд разрабатывается для целей моделирования авиационных самоорганизующихся телекоммуникационных сетей с применением реальных транспондеров. Система позволит производить оценку эффективности работы сетей и VDL Mode 4. Создание стенда также нацелено на реализацию учебного класса для будущих пользователей усовершенствованной системы ЛПД режима 4 с сетевой надстройкой.

Стенд, а также учебный класс создается в интересах компаний и государственных служб, имеющих парк летательных аппаратов (в том числе беспилотных) и компаний, занимающихся авиаперевозками грузов и пассажиров (авиакомпаний).

Стенд полунатурного моделирования авиационных сетей и учебный класс позволят повысить эффективность внедрения новых алгоритмов функционирования систем АЗН-В и предоставить комплекс для обучения пилотов и операторов, которые будут работать с подобными системами.

Нормативные документы. Реализация АЗН-В в Российской Федерации осуществляется в соответствии с Программой «Внедрение средств вещательного автоматического зависимого наблюдения (2011 – 2020 годы)», утвержденной Минтрансом России 19 мая 2011г.

Принципы функционирования VDL Mode 4 описаны в документах ICAO 9816 part 1 и ICAO 9816 part 2, 2004.

Принципы организации и требования к сетевой структуре описаны в документе ICAO 9896 «Руководство по сети авиационной электросвязи (ATN), использующий стандарты и протоколы пакета протоколов Интернет (IPS)», 2010.

При проектировании использовался отчет о НИР на тему «Предложения по совершенствованию нормативно технической базы, регулирующей использование глобальной навигационной спутниковой системы ГЛОНАСС и систем на ее основе в интересах навигационно-информационного обеспечения транспортного комплекса Российской Федерации».

Общая структурная схема стенда полунатурного моделирования показана на рисунке 50.

Состав стенда полунатурного моделирования:

1. Управляющая ЭВМ;
2. Интерфейсы для подключения пользовательских ЭВМ;
3. Модифицированные транспондеры с интерфейсами для внешнего управления;
4. Блоки аттенюаторов;
5. Источник питания транспондеров;
6. Соединительные кабели.

Состав учебного класса:

1. Стенд полунатурного моделирования;
2. Пользовательские ЭВМ;
3. Сетевой коммутатор;
4. Интерактивные доски.

Управляющая ЭВМ. Модель и характеристики управляющей ЭВМ выбраны исходя из потребности к высокой производительности вычислительного и графического процессоров.

Управляющая ЭВМ должна обеспечивать следующие функции:

- Имитация сигналов ГНСС;
- Генерация временных сдвигов;
- Сбор статистических данных;
- Обмен данными и управление пользовательскими ЭВМ.

Имитация сигналов ГНСС выполняется с помощью программы. Вся информация, получаемая транспондером по приемнику ГНСС, моделируется для каждого транспондера на управляющей ЭВМ и передается по внешним управляющим интерфейсам.

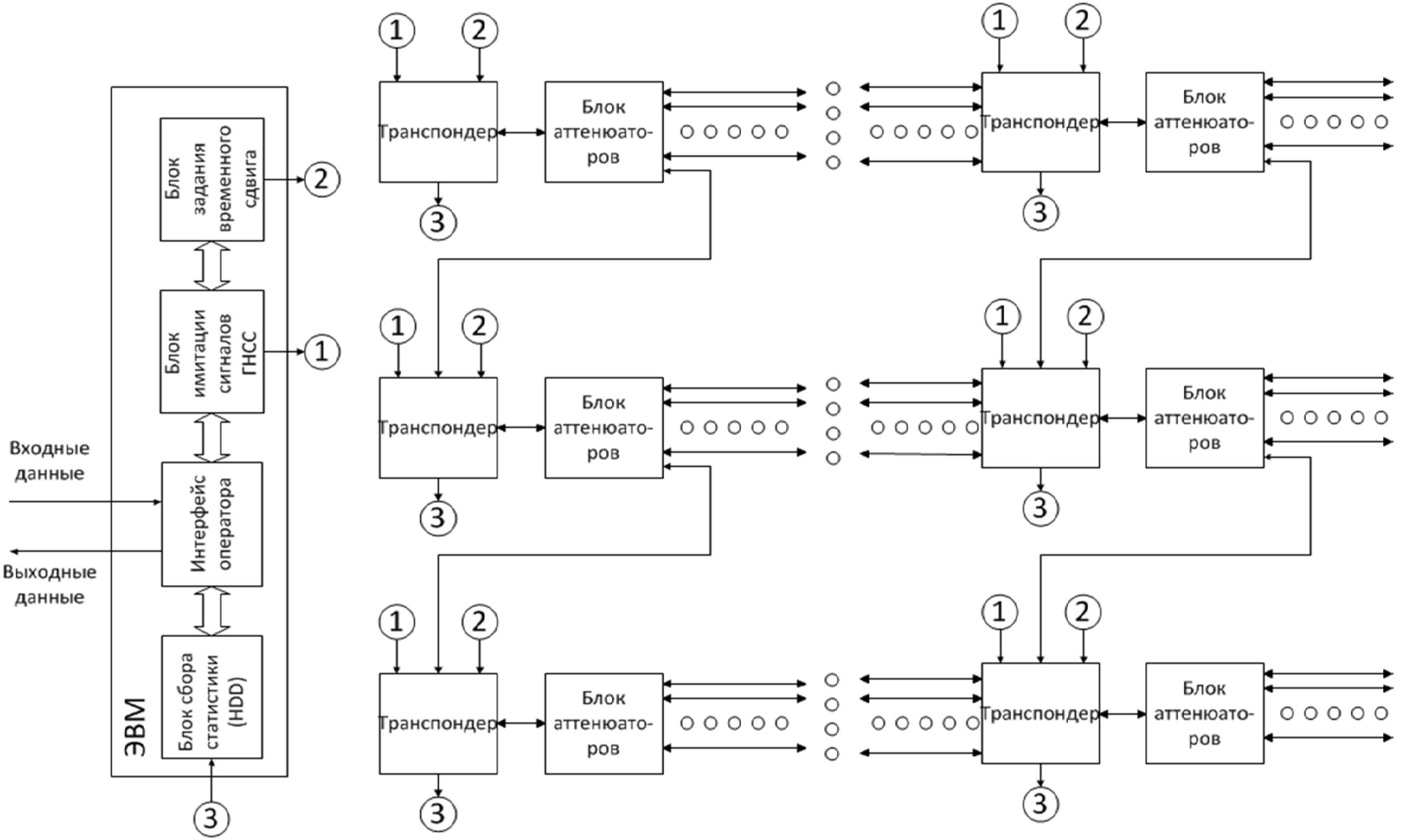


Рисунок 50. Общая структурная схема стенда полунатурного моделирования

Генерация временных сдвигов также выполняется с помощью программы. Программа генерации временных сдвигов ориентирована на имитацию задержки распространения сигналов между летательными аппаратами.

Сбор статистических данных производится на внешний жесткий диск, подключенный как к управляющей ЭВМ, так и к транспондерам. На внешний жесткий диск должна поступать статистическая информация обо всех ключевых событиях сети – передача и прием/потеря пакетов данных, выбираемые слоты, переключение режимов вещания и т.д. Собранные данные классифицируются и анализируются в дальнейшем управляющей ЭВМ.

Обмен данными и управление пользовательскими ЭВМ происходит на программном уровне. Для осуществления эффективного взаимодействия управляющей и пользовательских ЭВМ требуется создать локальную вычислительную сеть (ЛВС) на базе технологии Ethernet. Управляющая ЭВМ должна осуществлять контроль доступа и отображение информации только для того пользовательского ЭВМ, который соотноситься с определенным транспондером. Определения соответствия пользовательской ЭВМ и определенного транспондера генерируется на этапе создания сценария работы комплекса полунатурного моделирования.

Интерфейсы для подключения пользовательских ЭВМ. Подключение пользовательских ЭВМ должно быть осуществлено по ЛВС. Пользовательские и управляющая ЭВМ объединяются в ЛВС с помощью сетевого коммутатора.

Модифицированные транспондеры с интерфейсами для внешнего управления. В комплексе полунатурного моделирования должны применяться малогабаритные транспондеры стандарта VDL Mode 4 с реализованным физическим уровнем и механизмами выбора и резервирования временных слотов. Для получения информации ГНСС и моделирования времени распространения сигнала с помощью управляющей ЭВМ должны использоваться специальные дополнительные интерфейсы. Тип и количество интерфейсов будут определены на завершающем этапе разработки транспондера. ,

Блоки аттенюаторов используются для имитации затухания при распространении сигналов между транспондерами. Возможность использования управляемых транспондеров будет рассмотрена в процессе проектирования транспондеров.

Аттенюаторы должны ослаблять сигнал в полосе от 108 до 137 МГц и иметь величину ослабления от 20 до 120 дБ.

Состав учебного класса. На рисунке 51 представлена общая схема учебного класса для отработки навыков использования системы VDL 4 с самоорганизующейся надстройкой.

- Управляющая ЭВМ обеспечивает контроль и управление всей общей системой.

- Пользовательские ЭВМ соединены в общую ЛВС.
- Транспондеры соединены друг с другом с помощью блоков аттенюаторов и управляются непосредственно ЭВМ.
- На интерактивные доски, расположенных на двух стенах, выводится информация общего процесса моделирования, а также учебная и справочная информация.

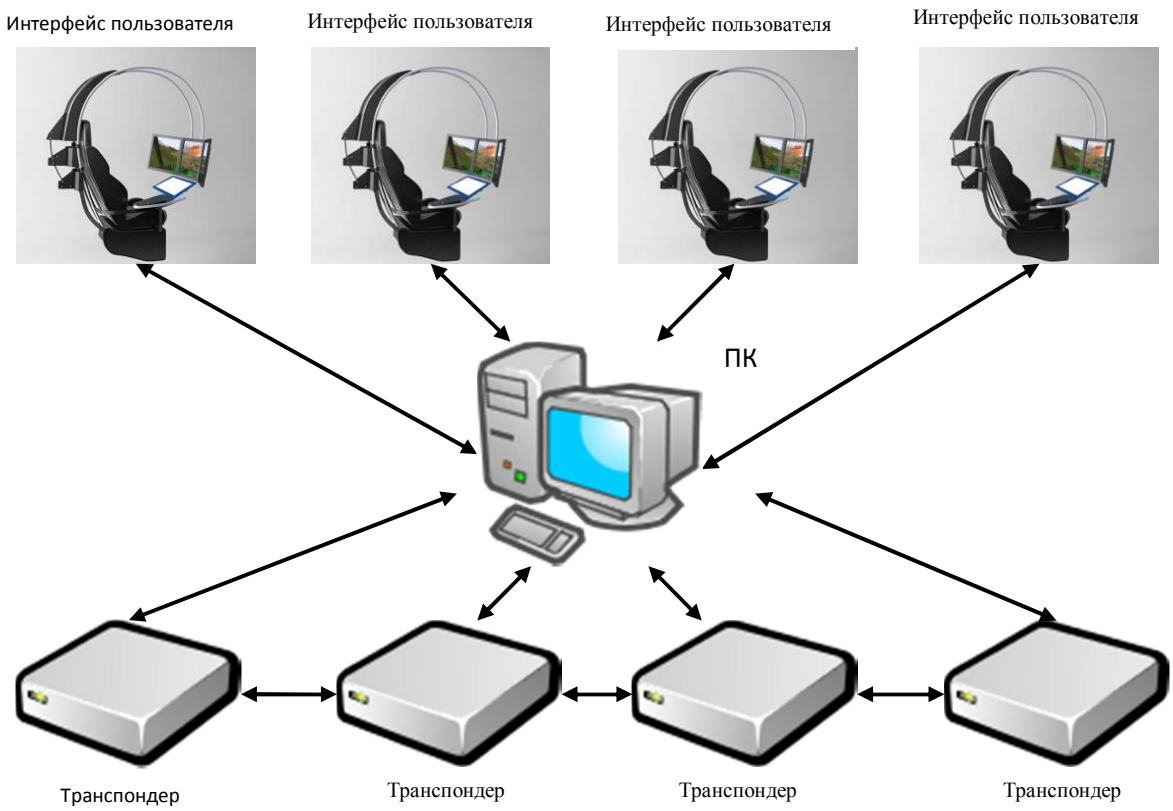


Рисунок 51. Общая схема учебного класса

Сетевой коммутатор (network switch) является устройством, которое используется в пакетных сетях передачи данных. Его основная функция - объединение нескольких сетевых сегментов, на канальном уровне модели стека протоколов IP. Коммутатор передает данные (коммутирует пакеты) от одного порта к другому, основываясь на содержащейся в кадре информации. Для выбора выходного порта коммутатор использует MAC-адрес устройства, к которому передаются данные из таблицы коммутации.

Интерактивная доска, представляющая собой сенсорный экран с большой диагональю, работает как часть системы совместно с компьютером и проектором. Проектор, в свою очередь, используется для отображения происходящего на экране компьютера на поверхность интерактивной доски.

Локальная вычислительная сеть комплекса полунатурного моделирования (ЛВС КПМ) самоорганизующихся авиационных сетей обеспечивает информационный обмен в реальном масштабе времени между КПМ и пользовательскими ЭВМ, а также доступ в глобальную сеть Интернет.

Сообщения информационного обмена могут содержать любую информацию – как и управляющую, так и персональную. Доступ к сети Интернет может быть использован для организации удаленного доступа и управления КПМ и учебного класса. На рисунке 52 показана общая схема ЛВС в учебном классе. Выход в сеть Интернет может быть осуществлен при помощи любого сетевого маршрутизатора дост. Маршрутизаторы работают на сетевом (третьем) уровне модели OSI.

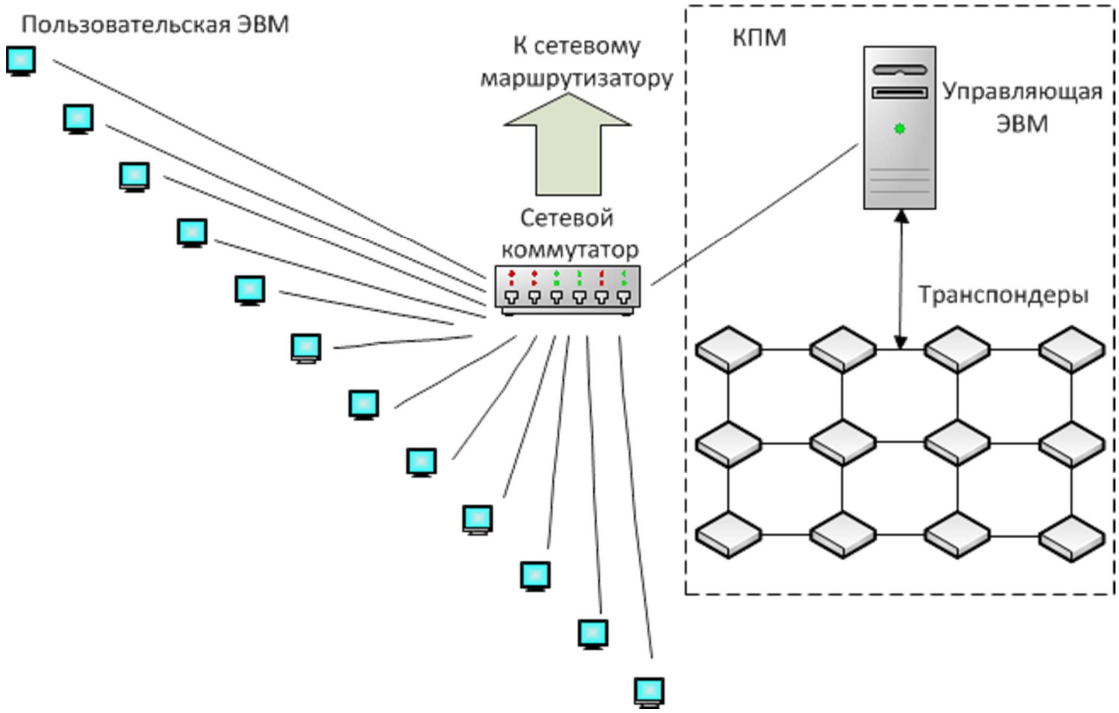


Рисунок 52. Общая схема ЛВС учебного класса

Полунатурное моделирование является одним из видов исследования различных систем на их моделях. В отличие от компьютерного и математического моделирования, где объекты или системы представлены в виде формул или программных блоков, что является лишь приближением к реальным составляющим системы, полунатурное моделирование нацелено на использование реальных объектов. Таким образом, учитываются те физические особенности процессов, которые не могут быть учтены в абстрактных моделях.

Воздушные самоорганизующиеся сети являются сложными системами, представляющими совокупность аппаратных и программных модулей. Так как специфика применения подобных сетей требует высокой надежности их работы, необходимо создание

многофункциональной системы моделирования, максимально приближенной к реальной системе.

Стенд полунатурного моделирования самоорганизующихся авиационных сетей предназначен для линий передачи данных АЗН-В режима VDL 4. Целью создания стенда является моделирование различных сетевых протоколов и оценка эффективности их работы, а также имитация сценариев работы сети.

Интерфейс пользователя имеет вид тренажера и предоставляет оператору возможность отслеживать данные об изменениях в сети, а также влиять на выполнение сценария работы сети, например, инициализируя передачу данных или запрос на ретрансляцию. На мониторе должны отображаться моделируемые параметры, общая картина работы сети и варианты выбора действий.

ЭВМ является главным средством контроля и управления стендом. Программными средствами обеспечивается:

- обмен данными с пользовательским интерфейсом;

- имитация сигналов ГНСС (например, протокола NMEA);

- управление временным сдвигом слота (имитация распространения электромагнитных волн);

- сбор статистических данных.

Стенд имитационного моделирования воздушных самоорганизующихся сетей должен выполнять, как минимум, следующие задачи:

- Моделирование ретрансляция данных АЗН-В при наличии препятствий для распространения сигнала (рисунок 53);

- Возможность оператору осуществлять запрос данных о соседних ВС от любого ВС в зоне радиовидимости транспондера (рисунок 54);

- Моделирование обмена данными с наземной станцией с помощью установления сетевого соединения (рисунок 55);

- Сбор и отображение, как общей статистики, так и событий происходящих во время моделирования;

- Моделирование различных сценариев развертывания сети.

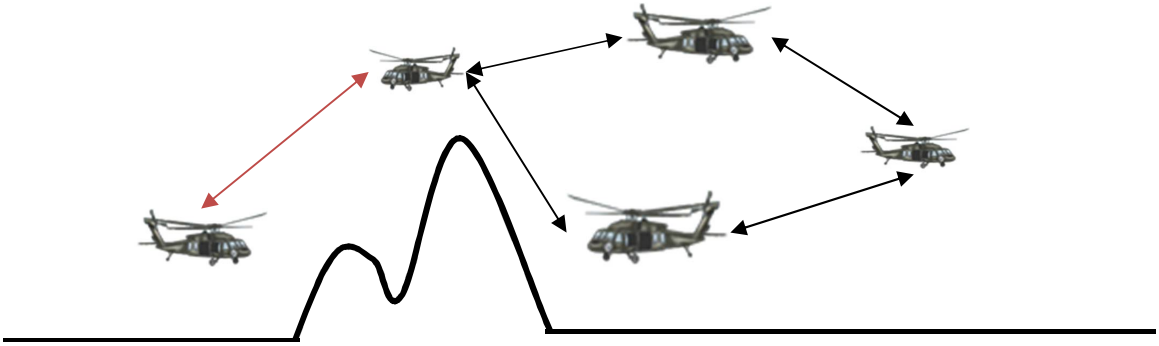


Рисунок 53. Ретрансляция данных АЗН-В при наличии препятствий для распространения сигнала

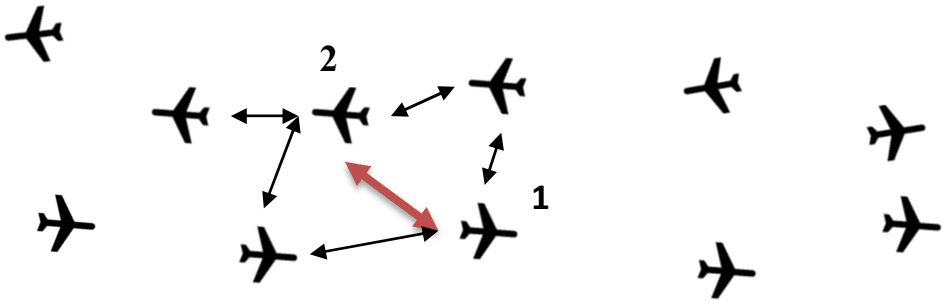


Рисунок 54. Осуществление запроса данных о соседних ВС одним участником сети у другого (например, 1 и 2)

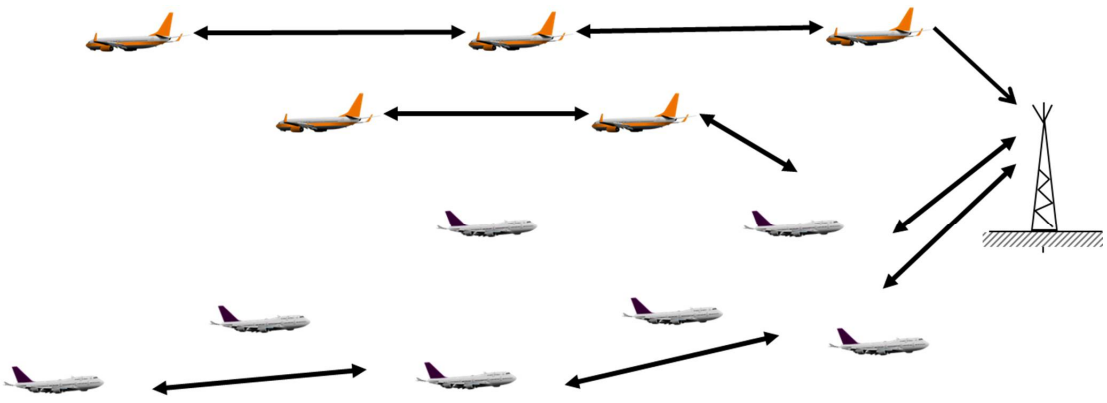


Рисунок 55. Обмен данными с наземной станцией с помощью установления сетевого соединения

Создание стенда полунатурного моделирования самоорганизующихся воздушных сетей позволит существенно повысить эффективность разработки протоколов маршрутизации самоорганизующихся сетей и аппаратной реализации транспондеров, а также позволит создать демонстрационную платформу для наглядного представления возможностей разрабатываемой системы. Это также позволит осуществлять эффективную

отработку различных сценариев функционирования протоколов маршрутизации и общей системы в целом. Получаемые данные могут быть использованы как для подтверждения или опровержения теоретических выкладок, так и для дальнейшей разработок и модификаций системы. На основе стенда будет иметься возможность проводить испытания без натурного (летного) исполнения, что существенно снизит стоимость разработок и повысить скорость исследований. Комплекс полунатурного моделирования вместе с учебным классом позволит осуществлять подготовку специалистов для работы с новыми системами АЗН-В, в частности VDL 4 с сетевой модификацией.

5.3 Возможная техническая реализация

На данный момент в качестве основы для технической реализации системы VDL Mode 4 и приложений на её основе рассматривается разработанная ФГУП «ГосНИИАС» совместно с НИИМА «Прогресс» система на кристалле, включающая две микросхемы: К5200МХ014 (РППУ-ЛСН) – аналоговая часть и К1917ВС014 (ЦПП-ЛСН) – цифровая часть.

5.4 Безопасность сети

Основной задачей защиты информации в сети является организация закрытого канала передачи данных, обеспечение достоверности, целостности, конфиденциальности, а также подтверждения авторства передаваемых сообщений.

Ниже представлены основные угрозы для функционирования распределённой самоорганизующейся сети ВС:

- Радиоперехват передаваемой информации с целью получения координат воздушного судна, реализация атаки "человек посередине";
- Организация имитационных помех, подобных реальным сигналам ВС, с целью сбить с курса;
- Завал спамом с целью перегрузки управляющей аппаратуры и дезориентации – DoS-атака;

Для защиты от перечисленных угроз предлагается использовать криптографические методы защиты информации.

Шифрование - процесс преобразования исходной информации в некоторый набор символов с целью её сокрытия от неавторизованных лиц. Шифрование - обратимый процесс,

т.е. легальные участники информационного обмена, обладающие ключом, применив обратную операцию (расшифровывание), восстанавливают исходное сообщение.

Существует два вида шифрования: симметричное (или одноключевое) и асимметричное (двухключевое) [103].

Симметричное или одноключевое шифрование представляется, как функция:

$$E_k(m) = c \tag{5.4.1}$$

где E - функция шифрования;

k - ключ;

m - исходное сообщение;

c - зашифрованное сообщение.

Функция расшифровывания D имеет вид:

$$D_k(c) = m \tag{5.4.2}$$

Основные операции симметричного шифрования - многократные перестановка и замена, управляемые ключом. Симметричное шифрование бывает блочным и потоковым. Отличие заключается в том, что в потоковом шифровании операции производятся над текущим битом, а в блочном - над блоком информации фиксированного размера.

Главной проблемой симметричного шифрования является безопасное распространение ключа между абонентами.

Для асимметричного шифрования характерно наличие двух различных ключей для шифрования и расшифровывания. Таким образом, формулы (5.1) и (5.2) принимают вид:

$$E_{k1}(m) = c$$

$$D_{k2}(c) = m$$

$$k1 \neq k2$$

Причём, $k2$ невозможно за разумное время вычислить по известному $k1$. Асимметричные криптосистемы называют также криптосистемами с открытым ключом, т.к. зашифровать сообщение ключом $k1$ может любой пользователь, а расшифровать данное сообщение сможет лишь владелец секретного ключа $k2$.

Концепция асимметричной криптографии основывается на т.н. односторонних функциях с лазейкой. Вычисление значения такой функции не представляет сложности, однако, значение обратной функции не вычисляемо без ключа-"лазейки". Примерами таких функций и обратных к ним в конечных полях являются: произведение – факторизация, возведение в степень – дискретное логарифмирование.

Для обеспечения криптостойкости асимметричных систем, необходимая длина пакета должна составлять 2048 бит, в то же время, взаимодействие между ВС осуществляется

короткими посылками по 256 бит. В асимметричных криптосистемах обработка данных происходит дольше, чем в симметричных, в связи с большим количеством вычислений.

Таким образом, закрытый канал целесообразно обеспечивать методом симметричного потокового шифрования. Потоковое шифрование обладает высокой криптостойкостью, не снижает скорость передачи. Для криптосистемы потокового шифрования характерна простая аппаратная реализация.

Для формирования симметричного ключа, а также подтверждения авторства передаваемых сообщений предлагается использовать асимметричную криптосистему на эллиптических кривых. Применение эллиптической криптографии позволяет обеспечить высокую криптостойкость при длине ключа 256 бит [104].

В отличие от классической криптографии, где все операции осуществляются в конечных полях, в эллиптической криптографии вычисления производятся в поле точек эллиптической кривой, вида

$$y^2 \equiv x^3 + a \cdot x + b (mod\ p) \tag{5.4.3}$$

Алгоритмы RSA, Эль-Гамаля, Диффи-Хеллмана могут быть выполнены на эллиптических кривых.

Каждому абоненту присвоен уникальный номер борта и пара $\langle$открытый ключ, закрытый ключ$\rangle$. Открытые ключи хранятся централизованно в базе данных, доступ к которой разрешён только легальным участникам информационного обмена. Закрытые ключи известны только их владельцам.

Далее в общем виде представлены формулы шифрования/дешифрования на эллиптических кривых:

$$E_{\{pub\_K, pr\_K\}}(DATA) = DATA + pr\_K1 \times pub\_K2 = DATA + pr\_K1 \cdot pr\_K2 \times G \tag{5.4.4}$$

$$D_{\{pub\_K, pr\_K\}}(E\_DATA) = E\_DATA - pr\_K2 \times pub\_K1 = E\_DATA - pr\_K2 \cdot pr\_K1 \times G \tag{5.4.5}$$

где:

DATA – открытый текст;

E_DATA – зашифрованные данные;

pub_K1, pub_K2 – открытые ключи абонентов 1 и 2;

G – генерирующая точка на кривой;

pr_K1, pr_K2 – закрытые ключи абонентов 1 и 2;

+ - операция сложения точек;

× - операция умножения точки на число.

DATA, E_DATA, pub_K1, pub_K2 и G – точки эллиптической кривой.

Каждый абонент с определённой периодичностью осуществляет вещательную передачу номера своего борта (в открытом виде). Для установления соединения по полученному номеру из базы данных извлекается открытый ключ абонента и производится передача зашифрованного сообщения, где в качестве открытого текста DATA в формуле (5.4) используется симметричный ключ.

Безопасность системы основана на проблеме дискретного логарифмирования эллиптической кривой в конечных полях. Злоумышленник не сможет ни прочитать, ни отправить сообщение, не зная закрытый ключ. Таким образом, осуществляется авторизация абонентов, в то же время, симметричный ключ защищен от компрометации и подмены. Для защиты от повторов ранее переданных сообщений используется метка времени в зашифрованном сообщении.

5.5 Выводы

1. Исследование результатов моделирования разработанной дискретно-временной модели мобильной самоорганизующейся сети, построенной на основе стандарта VDL Mode 4 между участниками воздушного движения и пунктами УВД, показало, что развёртывание сети в отдаленных и океанических регионах сможет обеспечить наблюдение ВС, находящихся за пределами прямой видимости БС. Полученные значения периодов наблюдения ВС через сеть, вне зоны прямой видимости, достигали 2 часов и покрывали 100% длительности этих периодов. Также для одного из сценариев было получено значение коэффициента потерянных сообщений равное 10^{-4}, что свидетельствовало о наличии полностью связной сетевой структуры и корректности разработанного протокола маршрутизации. Предположения, выдвинутые в разделе, посвященном аналитическим исследованиям связности сети о том, что связность рассматриваемой сети является основным фактором, влияющим на её производительность, были подтверждены моделированием.

2. Внедрение новых стандартов и технологий в гражданской авиации является длительным процессом [105]. Поэтому в разделе был рассмотрен комплекс полунатурного моделирования с обучающим классом, в котором используются реальные приёмопередатчики стандарта VDL Mode 4. Кроме проведения полунатурных испытаний, целью КПМ является ознакомление лиц, заинтересованных в развитии авионики в РФ, с современными технологиями цифровой передачи данных. В разделе также рассмотрена основа для технической реализации приёмопередатчиков стандарта VDL Mode 4.

3. Важной частью функционирования сети является информационная безопасность и в рассматриваемом случае она напрямую влияет на безопасность полётов и воздушного движения в целом. Одним из эффективных методов защиты информации является шифрование. В разделе рассмотрены некоторые виды возможных информационных атак и методы криптографии. Так как все ВС обладают уникальным бортовым идентификатором и периодически в широковещательном режиме сообщают их, то был сделан вывод о возможности применения двухключевых криптоалгоритмов в целях защиты информации.

ЗАКЛЮЧЕНИЕ

1. Предложен метод повышения ситуационной осведомленности пунктов УВД в отдаленных и океанических регионах путём передачи сообщений АЗН-В от ВС, находящихся за пределами прямой видимости, с помощью мобильной самоорганизующейся сети.

2. Теоретическими исследованиями связности мобильной самоорганизующейся сети в отдаленных и океанических регионах и моделирования реальных полётных данных доказано, что существует возможность передачи данных АЗН-В по сети периодом до 2-х часов, в течение всего времени отсутствия прямого приёма между ВС и пунктами УВД.

3. Разработана компьютерная модель, учитывающая особенности функциональной модели VDL Mode 4, а также такие факторы как мобильность узлов сети и условия распространения радиоволн. Путём варьирования параметров программных модулей, модель позволяет получать следующие числовые показатели производительности сети: количество отправленных и полученных сообщений, задержки при передаче сообщений по сети, а также число узлов, от которых были получены сетевые сообщения с данными о местоположении и намереньях.

4. Разработан протокол маршрутизации сообщений АЗН-В от ВС до пунктов УВД, использующий «жадный» алгоритм выбора маршрутизаторов и реактивный метод построения таблиц маршрутизации. Разработанный алгоритм позволяет использовать «однослотовые» сообщения и минимизирует количество служебной информации, передаваемой по сети. Доказана применимость выбранной метрики маршрутизации для рассматриваемой сети.

5. Предложены числовые параметры: диапазон выбора временного слота, период вещания сетевых сообщений БС и хранения записей в таблице маршрутизации. По результатам моделирования установлено, что наилучшими значениями является 150 слотов, 150с и 660с для предложенных параметров соответственно, при этом усредненные и максимальные задержки составили 7,4с и 19,8 с, число полученных сообщений – 256 и было отслежено 22 борта в сценарии с 50 узлами. Максимальное число узлов в сети составило 300 шт. для 12 АЗН-В отчётов в минуту в зоне прямой видимости и 1 отчёту в минуту по сети.

СПИСОК СОКРАЩЕНИЙ

АЗН-В - радиовещательное автоматическое зависимое наблюдение

АЗН-К - автоматическое зависимое наблюдение контрактное

АЗН-Р - автоматическое зависимое наблюдение ретрансляционное

БПЛА – беспилотный летательный аппарат

БС – базовая станция

ВС – воздушное судно

ВС АОН – воздушное судно авиации общего назначения

ГНСС - глобальная навигационная спутниковая система

ЕС УВД – единая система управления воздушным движением

ЛПД - линия передачи данных

МЧС - министерство Российской Федерации по делам гражданской обороны, чрезвычайным ситуациям и ликвидации последствий стихийных бедствий

НС – наземная станция

ОСШ – отношение сигнал/шум

УВД - управление воздушным движением

УКВ - ультракороткие волны

ЭМВОС – эталонная модель взаимодействия открытых систем

AFTN - Aeronautical Fixed Telecommunication Network, фиксированная (наземная) авиационная сеть электросвязи

ATM – Air Traffic Management, управление воздушным движением

ATN - Aeronautical Telecommunication Network, авиационная сеть электросвязи

AVLC – Aviation VHF Link Control, протокол управления авиационным УКВ каналом

CSMA/CA - Carrier Sense Multiple Access / Collision Avoidance, множественный доступ с контролем несущей и избеганием коллизий

DLS – Data-Link Service, подуровень сервисов канала данных

D8PSK - Differential 8-Phase Shift Keying, восьмипозиционная фазовая манипуляция

ES – Extended squitter, расширенный «сквиттер»

FIFO – First In First Out, тип очереди «первый вошёл, первый вышел»

GFSK – Gaussian Frequency Shift Keying, частотная манипуляция с гауссовой предмодуляционной фильтрацией

GNSS - Global Navigation Satellite System, глобальная навигационная спутниковая система

GPS - Global Positioning System, глобальная система позиционирования

GSC - Global Signaling Channels, глобальные сигнализационные каналы

HDLC - High-Level Data Link Control, высокоуровневый протокол управления каналом связи

IEEE - Institute of Electrical and Electronics Engineers, институт инженеров по электротехнике и электронике

ICAO - International Civil Aviation Organization, международная организация гражданской авиации

IP – Internet Protocol, интернет протокол

IPS - Internet Protocol Stack, стек интернет протоколов

IS – Intermediate System, промежуточная система

ISO - International Organization for Standardization, международная организация по стандартизации

LDASC1 - L-band Digital Aeronautical Communication System, цифровая авиационная система связи L-диапазона

LME - Link Management Entity, подуровень управления связью

LSC - Local Signaling Chanel, локальный канал сигнализации

MAC - Media Access Control, подуровень контроля доступа к физической среде передачи данных

MANET – Mobile Ad Hoc Network, мобильная Ad Hoc сеть

NED – Network Description, язык описания модели сети

OSI - Open Systems Interconnection Basic Reference Model, базовая эталонная модель взаимодействия открытых систем

PDU – Protocol Data Unit, информационная единица протокола

PECT – Peer Entity Contact Table, таблица узлов в прямой зоне видимости

SARP - Standard and Recommended Practice, отдел ICAO по выработке стандартов и рекомендаций

SOTDMA (STDMA) - Self-organized Time division Multiple Access, самоорганизующийся множественный доступ с временным разделением канала

TCP/IP – Transmission Control Protocol / Internet Protocol, сетевая модель передачи данных протокола интернет

TDMA - Time-division multiple access, множественный доступ с временным разделением канала

UTC - Universal Time Coordinated, универсальное координированное время

VDL Mode 2, Mode 3, Mode 4 - Very high frequency Data Link Mode 2 (очень высокой частоты линия передачи данных режима 2, режима 3, режима 4)

VSS - VDL Mode 4 Specific Services, подуровень специальных сервисов ОВЧ ЛПД Режима 4

СПИСОК ЛИТЕРАТУРЫ

1. Фальков, Э. Экспериментальные полеты БЛА в общем воздушном пространстве [Электронный ресурс] / Э. Фальков, В. Воронов // UAV.RU: интернет-издание, посвященное беспилотной авиации. — 2012. — Режим доступа: http://uav.ru/articles/adsb.pdf (дата обращения: 22.06.2017).
2. ICAO Doc. ADS-B implementation and operations guidance document – 2014. – Edition 7.0. – P. 85.
3. Программа внедрения средств вещательного автоматического зависимого наблюдения (2011 - 2020 годы): утв. Минтрансом РФ 19 мая 2011г. // Совместное заседание секций НТС Минтранса. 2010. Протокол № ВО-57 от 10.11.2010.
4. Self-configuring and self-optimizing network (SON) use cases and solutions. 3GPP Technical Report. 3GPP TR36.902 version 9.1.0. March 2010.
5. Кучерявый, А.Е. Самоорганизующиеся сети / А.Е. Кучерявый, Е.А. Кучерявый, А.В. Прокофьев. – СПб.: Изд-во «Любавич», 2011. – 310с.
6. Сбор данных с наземного сегмента летающей сенсорной сети как система массового обслуживания / А.В. Шкляева, Р.В. Киричек, А.И. Парамонов, А.Е. Кучерявый // Интернет вещей и 5G: сборник трудов 2-ой международной научно-технической конференции студентов, аспирантов и молодых ученых. – СПб.: СПбГУТ, 2016. – С. 12 – 16.
7. Самоорганизующиеся сети связи мультиагентных робототехнических систем / Е.Г. Борисов, А.Г. Владыко, А.И. Парамонов, Р.В. Киричек // Актуальные проблемы защиты и безопасности: сборник трудов XIX всероссийской научно-практической конференции. – СПб.: РАРАН, 2016. – С. 210 – 217.
8. Дао, Ч.Н. Анализ структуры сетей связи на базе беспилотных летательных аппаратов / Ч.Н. Дао, А.И. Парамонов // Распределенные компьютерные и телекоммуникационные сети: управление, вычисление, связь (DCCN-2016): сборник материалов XIX международной научной конференции. – М.: РУДН, 2016. – С. 92 – 100.
9. Boukerche, A. Algorithms and protocols for wireless and mobile Ad Hoc networks / A. Boukerche – Canada: University of Ottawa, 2009 – 497 p.
10. Rajeev, S. Mobile, wireless, and sensor networks: technology, applications, and future directions / S. Rajeev – Wiley, 2006. – 430 p.
11. Labiod, H. Wireless Ad Hoc and sensor networks / H. Labiod – Wiley, 2008. – 317 p.

12. On the ability of the 802.11p MAC method and STDMA to support real-time vehicle-to-vehicle communication / K. Bilstrup, E. Uhlemann, E. G. Ström, U. Bilstrup // EURASIP Journal on Wireless Communications and Networking. – 2009. – 13 p.
13. Bilstrup, K. Scalability issues of the MAC methods STDMA and CSMA of IEEE 802.11p when used in VANETs / K. Bilstrup, E. Uhlemann, E. Ström // in Proc. of the ICC'10 Workshop on Vehicular Connectivity. – Cape Town – 2010.
14. Васильев, Д.С. Протоколы маршрутизации в MANET / Д.С. Васильев, А.В. Абилов // Электросвязь. – 2014. – № 11. – С. 52 – 54.
15. Vasilev, D.S. Peer selection algorithm in flying Ad Hoc networks / D.S. Vasilev, A.V. Abilov, V.V. Khvorenkov // International Siberian Conference on Control and Communications (SIBCON–2016) resources of the Internet. – 2016. – С. 382 – 386.
16. Шамонов, М.Ю. Мобильные самоорганизующиеся сети беспилотных летательных аппаратов flying Ad Hoc networks (FANETs) / М.Ю. Шамонов, А.В. Абилов // Приборостроение в XXI веке - 2016. Интеграция науки, образования и производства: сборник материалов XII Международной научно-технической конференции. – 2017. – С. 542 – 550.
17. Кайсина, И.А. Анализ эффективности протоколов маршрутизации OLSR и AODV в летающей сети FANET / И.А. Кайсина, Д.С. Васильев, А.В. Абилов // Вестник ИжГТУ им. М.Т. Калашникова. – 2017. – Т. 20. – № 1. – С. 87 – 90.
18. Кайсина, И.А. Модель в среде NS-3 для передачи видеоданных в сети БПЛА / И.А. Кайсина, Д.С. Васильев, А.В. Абилов // Выставка инноваций - 2017 (весенняя сессия): сборник материалов XXIII республиканской выставки-сессии студенческих инновационных проектов. – Ижевск: ИжГТУ имени М.Т. Калашникова, 2017. – С. 69 – 74.
19. Стандартизация интернета вещей / М.Ю. Самсонов, А.Ю. Гребешков, А.В. Росляков, С.В. Ваняшин // Электросвязь. – 2013. – № 8. – С. 10 – 13.
20. Дроздова, Е.А. Исследование граничных значений задержек в беспроводных сенсорных сетях / Е.А. Дроздова, А.В. Росляков // Интернет вещей и 5G: сборник трудов 2-ой международной научно-технической конференции студентов, аспирантов и молодых ученых. – СПб.: СПбГУТ, 2016. – С. 7 – 11.
21. The insight of message delivery delay in VANETs with an bidirectional model / Y. Liu, J. Niu, J. Ma et al. // J.Netw.Comput.Appl. – 2013. – vol. 36. – № 5. – P.1287–1294.
22. Perkins, C.E. Highly dynamic destination sequence distance-vector routing (DSDV) for mobile computers / C.E. Perkins, P. Bhagwat. // In proceedings of Conference on

Communications Architectures «Protocols and Applications» (SIGCOMM'94). – New York, 1994. – October.

23. Evaluating the impact of a novel message dissemination scheme for vehicular networks using real maps / M. Fogue, P. Garrido, F.J. Martinez et al. // Transp.Res. Part C. Emerg.Technol. – 2012. – № 25. – P. 61–80.
24. Hindering false event dissemination in VANETs with proof-of-work mechanism / E. Palomar, J.M. Fuentes, A.I.G. Tablas, A. Alcaide // Transp.Res. Part C. Emerg.Technol. – 2012. – № 23. – P. 85–97.
25. Naumov, V. Connectivity Aware Routing (CAR) in vehicular ad hoc networks / V.Naumov, T.R. Gross // Proceedings of 26th IEEE International Conference on Computer Communications «INFOCOM 2007». – Anchorage, 2007. – May.
26. Karp, B. GPSR: greedy perimeter stateless routing for wireless networks / B. Karp, H.T. Kung // Mobile Computer Networks. – 2000. – P. 243–254.
27. Feasibility of an Aeronautical Mobile Ad Hoc Network Over the North Atlantic Corridor / D. Medina, F. Hoffmann, S. Ayaz, C. H. Rokitansky // IEEE SECON. – 2008. – P. 109 – 116.
28. Hoffmann, F. Optimization of Routing and Gateway Allocation in Aeronautical Ad Hoc Networks Using Genetic Algorithms / F. Hoffmann, D. Medina, A. Wolisz. // Proceedings of IWCMC'2011. – Istanbul, 2011. – July.
29. A Geographic Routing Strategy for North Atlantic In Flight Internet Access Via Airborne Mesh Networking / D. Medina, F. Hoffmann, F. Rossetto, C.-H. Rokitansky // IEEE/ACM Transactions on Networking. – 2012, August. – Vol. 20. – № 4 – P. 1231–1244.
30. Hoffmann, F. Joint Routing and Scheduling in Mobile Aeronautical Ad Hoc Networks / F. Hoffmann, D. Medina, A. Wolisz // IEEE Transactions on Vehicular Technology. – 2013, July. – Vol. 62. – № 6. – P. 2700 – 2712.
31. Topology characterization of high density airspace aeronautical ad hoc networks / D. Medina, F. Hoffmann, S. Ayaz, C. H. Rokitansky // 5th IEEE International Conference on Mobile Ad Hoc and Sensor Systems. – 2008. – P. 295 – 304.
32. Hoffmann, F. Two-step delay based Internet gateway selection scheme for aeronautical ad hoc networks / F. Hoffmann, D. Medina, A. Wolisz // IEEE 20th International Symposium on Personal, Indoor and Mobile Radio Communications. – 2009, September. – P. 2638 – 2642.

33. Hoffmann, F. Protocol architecture analysis for Internet connectivity in aeronautical ad hoc networks / F. Hoffmann, D. Medina // 29th Digital Avionics Systems Conference. – 2010, October. – P. 3.C.4-1 - 3.C.4-12.
34. .North Atlantic Inflight Internet Connectivity via Airborne Mesh Networking / D. Medina, F. Hoffmann, F. Rossetto, C. H. Rokitansky // IEEE Vehicular Technology Conference (VTC Fall). – 2011, September. – P. 1 – 5.
35. A Crosslayer Geographic Routing Algorithm for the Airborne Internet / D. Medina, F. Hoffmann, F. Rossetto, C. H. Rokitansky // IEEE International Conference on Communications (ICC). – 2010, May. – P. 1 – 6.
36. NEWSKY - Selected simulation results / M. Ehammer, T. Graupl, C.-H. Rokitansky et al. // Integrated Communications, Navigation and Surveillance Conference ICNS '09. – 2009, May. – P. 1 – 9.
37. Routing in the Airborne Internet / D. Medina, F. Hoffmann, F. Rossetto, C. H. Rokitansky // Integrated Communications Navigation and Surveillance Conference (ICNS). – 2010, May. – P. A7-1 - A7-10.
38. Performance Evaluation of Network Mobility Handover over Future Aeronautical Data Link / S. Ayaz, F. Hoffmann, C. Sommer et al. // IEEE Global Telecommunications Conference (GLOBECOM 2010). – 2010, December. – P. 1 – 6.
39. LDACS1 System Definition Proposal: Deliverable D2 / M. Sajatovic, B. Haindl, M. Ehammer et al. // Eurocontrol Study Report. – 2009, February. – Edition 1.0.
40. Кулаков, М.С. Исследование авиационной сети связи на базе режима VDL-4 с поддержкой алгоритмов самоорганизации / М.С. Кулаков // Технологии информационного общества: программа VII международной отраслевой научно-технической конференции. – М.: МТУСИ, 2013. – С. 7.
41. Кулаков, М.С. Анализ особенностей функционирования мобильных самоорганизующихся сетей MANET на уровне доступа к среде MAC / М.С. Кулаков // Технологии информационного общества: сборник тезисов VIII международной отраслевой научно-технической конференции. – М.: МТУСИ, 2014. – С. 49.
42. Кулаков, М.С. Компьютерное моделирование авиационной УКВ ЛПД Режима 4 / М.С. Кулаков // Технологии информационного общества: сборник тезисов IX международной отраслевой научно-технической конференции. – М.: МТУСИ, 2015. – С. 63.
43. Кулаков, М.С. Моделирование мобильной самоорганизующейся сети на базе ОВЧ ЛПД Режима 4 / М.С. Кулаков // Технологии информационного общества: сборник

тезисов X международной отраслевой научно-технической конференции. – М.: МТУСИ, 2016. – С. 50.

44. Кулаков, М.С. Применение алгоритмов самоорганизации для режима VDL 2 / М.С. Кулаков // Фундаментальные проблемы радиоэлектронного приборостроения INTERMATIC – 2012: программа международной научно-практической конференции. – М.: МИРЭА, 2012. – С. 28.
45. Кулаков, М.С. Анализ сценариев развертки мобильных Ad Hoc сетей на базе режима VDL Mode 4 / М.С. Кулаков // Фундаментальные проблемы радиоэлектронного приборостроения NTERMATIC – 2013: программа международной научно-практической конференции. – М.: МИРЭА, 2013. – С. 33.
46. Кулаков, М.С. Воздушные самоорганизующиеся сети / М.С. Кулаков // Моделирование авиационных систем: материалы всероссийской научно-практической конференции. – М.: ФГУП «ГосНИИАС», 2013. – С. 172 – 173.
47. Кулаков, М.С. Применение протоколов Location - Aware MANET для сети авиационного стандарта, функционирующей в режиме VDL Mode 4 / М.С. Кулаков // ИНФОКОМ – 2013: труды международной молодежной научно-практической конференции. – Ростов - на - Дону.: СКФ МТУСИ, 2013. – С. 127 – 130.
48. Шаврин, С.С. Разработка протокола маршрутизации самоорганизующейся Ad Hoc сети для систем АЗН-В / С.С. Шаврин, М.С. Кулаков // Труды учебных заведений связи. – 2016. – Т. 2. – № 2. – С. 88-93.
49. Кулаков, М.С. Применение алгоритмов самоорганизации для режима VDL 2 / М.С. Кулаков // INTERMATIC – 2012: материалы международной научно-практической конференции. – М.: МИРЭА, 2012. – Часть 5. – С. 58 – 62.
50. Кулаков, М.С. Анализ сценариев развертки мобильных Ad Hoc сетей на базе режима VDL Mode 4 / М.С. Кулаков // INTERMATIC – 2013: материалы международной научно-практической конференции. – М.: МИРЭА, 2013. – Часть 4. – С. 49 – 53.
51. Григорьев, И.Д. Роль использования технологии самоорганизующихся сетей в концепции "интернет вещей" / И.Д. Григорьев, М.С. Кулаков // Телекоммуникации и информационные технологии. – М.: МТУСИ, 2015. – Т. 2. – № 1. – С. 49-52.
52. Фальков, Э.Я. Имитационное моделирование самоорганизующейся сети АЗН-В в режиме VDL 4 / Э.Я. Фальков, М.С. Кулаков, В.В. Егоров // Авиационные системы в XXI веке: сборник докладов юбилейной всероссийской научно-технической конференции. – М.: ФГУП «ГосНИИАС», 2017. – С. 323 – 329.

53. Кулаков, М.С. Моделирование мобильной самоорганизующейся сети на базе ОВЧ ЛПД Режима 4 / М.С. Кулаков // Технологии информационного общества: сборник трудов X международной отраслевой научно-технической конференции. – М.: МТУСИ, 2016. – С. 136.
54. Кулаков, М.С. Моделирование авиационной ОВЧ ЛПД Режима 4 / М.С. Кулаков // Перспективные технологии в средствах передачи информации - ПТСПИ'2015: материалы конференции. – Суздаль, 2015. – С. 129 – 133.
55. Кулаков, М.С. Анализ особенностей функционирования мобильных самоорганизующихся сетей MANET на уровне доступа к среде MAC / М.С. Кулаков // Т•Comm. Телекоммуникации и транспорт. – 2014. – Т.8. – N 10. – С. 39 – 42.
56. Клёсова, Ю.В. Перспективные технологии в авиации на базе ОВЧ ЛПД Режима 4 / Ю.В. Клёсова, И.А. Татарчук, М.С. Кулаков // Т•Comm: Телекоммуникации и транспорт. – 2015. – Том 9. – No8. – С. 63 – 67.
57. Кулаков, М.С. Организация мобильных сетевых структур с коридороподобным типом движения сетевых узлов / М.С. Кулаков, С.С. Шаврин // Проектирование и технология электронных средств. – 2015. – No2. – С. 2. – 8.
58. Шаврин, С. Сети Ad Hoc для транспорта на базе авиационной системы связи VDL Mode 4 / С. Шаврин, М. Кулаков // Первая миля. – 2016. – № 7 (60). – С. 19 – 24.
59. ICAO Doc. 4444. Правила аэронавигационного обслуживания. Организация воздушного движения. – 2016. – Издание шестнадцатое. – 508 с.
60. Бордунов, В. Д. Правовое регулирование международных полетов гражданских воздушных судов / В. Д. Бордунов, А. И. Котов, Ю. Н. Малеев. – М.: Наука, 1988. – 209 с.
61. Exxonmobil [Офиц. сайт]. URL: http://corporate.exxonmobil.com/en/energy/energy-outlook/charts/light-duty-fleet-by-type-chart?parentId=8b22b63c-0329-4e0f-a820-9845ea41be7b (дата обращения: 28.11.14).
62. АвиаПорт.Ru – авиация и бизнес. Новостной ресурс. URL: http://www.aviaport.ru/digest/2007/08/22/126752.html (дата обращения: 29.11.16).
63. Technical Provisions for Mode S Services and Extended Squitter – ICAO Doc 9871 AN/460, Second Edition, 2012. – 352 p.
64. DO-281. Minimum operational performance standards for aircraft VDL Mode 2 physical, link, and network layer.
65. Prospective VDL Mode 3. Режим доступа URL: http://www.scribd.com/doc/73723696/72/PROSPECTIVE-VDL-MODE-3 Загл. с экрана.

66. ICAO Doc 9816 AN/448. Manual on VHF Digital Link (VDL) Mode 4. – 2004. – First Edition. – 406 p.
67. ETSI EN 301 842-1 V1.3.2 (2010-12). VHF air-ground Digital Link (VDL) Mode 4 radio equipment; Technical characteristics and methods of measurement for ground-based equipment; Part 1: EN for ground equipment.
68. ETSI EN 301 842-2 V1.5.1 (2006-11) Electromagnetic compatibility and Radio spectrum Matters (ERM); VHF air-ground Digital Link (VDL) Mode 4 radio equipment; Technical characteristics and methods of measurement for ground-based equipment; Part 2: General description and data link layer.
69. ETSI EN 301 842-3 V1.2.1 (2006-11). Electromagnetic compatibility and Radio spectrum Matters (ERM); VHF air-ground Digital Link (VDL) Mode 4 radio equipment; Technical characteristics and methods of measurement for ground-based equipment; Part 3: Additional broadcast aspects.
70. ETSI EN 301 842-4 V1.2.1 (2006-11). Electromagnetic compatibility and Radio spectrum Matters (ERM); VHF air-ground Digital Link (VDL) Mode 4 radio equipment; Technical characteristics and methods of measurement for ground-based equipment; Part 4: Point-to-point functions.
71. ETSI EN 302 842-1 V1.2.2 (2010-12). VHF air-ground and air-air Digital Link (VDL) Mode 4 radio equipment; Technical characteristics and methods of measurement for aeronautical mobile (airborne) equipment; Part 1: Physical layer.
72. ETSI EN 302 842-2 V1.2.1 (2006-12). Electromagnetic compatibility and Radio spectrum Matters (ERM); VHF air-ground and air-air Digital Link (VDL) Mode 4 radio equipment; Technical characteristics and methods of measurement for aeronautical mobile (airborne) equipment; Part 2: General description and data link layer.
73. ETSI EN 302 842-3 V1.2.1 (2006-12). Electromagnetic compatibility and Radio spectrum Matters (ERM); VHF air-ground and air-air Digital Link (VDL) Mode 4 radio equipment; Technical characteristics and methods of measurement for aeronautical mobile (airborne) equipment; Part 3: Additional broadcast aspects.
74. ETSI EN 302 842-4 V1.2.1 (2006-12). Electromagnetic compatibility and Radio spectrum Matters (ERM); VHF air-ground and air-air Digital Link (VDL) Mode 4 radio equipment; Technical characteristics and methods of measurement for aeronautical mobile (airborne) equipment; Part 4: Point-to-point functions.
75. Maihofer, C. A survey of geocast routing protocols / C. Maihofer // IEEE Communications Surveys & Tutorials. – 2004 – 6:32–42.

76. Mauve, M. A survey on position-based routing in mobile Ad Hoc networks / M. Mauve, J. Widmer, H. Hartenstein // IEEE Network. – 2001. – 15(6):30–39.
77. Geographic routing made practical / Y.J. Kim, R. Govindan, B. Karp, S. Shenker // in: Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation (NSDI'05). – Berkeley, 2005.
78. Приложение 10 к Конвенции о международной гражданской авиации. Авиационная электросвязь. Системы связи. – 2007. – Издание второе. – Т. 3. – 277 с.
79. Птицын, Г.А. Живучесть динамических сетей связи: учебное пособие / Г.А. Птицын; под реакцией А.В. Петракова – М.: МТУСИ, 2008. – 96 с.
80. Харари, Ф. Теория графов / Ф. Харари. – М.: Изд-во "Мир", 1973. – 302 с.
81. Основы теории надёжности электронных средств / Н. К. Юрков, А. В. Затылкин, С. Н. Полесский и др. — Пенза: Изд-во ПГУ, 2012. — 100 с.
82. Райншке, К. Оценка надежности систем с использованием графов / К. Райншке, И.А. Ушаков; под ред. И.А. Ушакова. – М.: Радио и связь, 1988. – 208 с.
83. Фокин, Г.А. Управление самоорганизующимися пакетными радиосетями на основе радиостанций с направленными антеннами: автореф. дис. … канд. техн. наук: 05.12.13 / Фокин Григорий Алексеевич – СПб., 2009. –17 с.
84. Varga, A. The ONMeT++ discrete event simulation system / A. Varga // in Proceeding of European Simulation Multiconference. – 2001. – P. 1 – 7.
85. FlightRadar24 [Информационный ресурс]// Режим доступа URL: http://www.flightradar24.com, свободный. –Загл. с экрана.
86. ICAO DOC 9925 AN/475. Руководство по авиационной подвижной спутниковой (маршрутной) службе. – 2010. – Издание первое. – 192 с.
87. ICAO Global Operational Data Link Document (GOLD). – 2013. – 2nd edition.
88. Невзоров, Ю.В. Факторы, влияющие на скорость передачи информации по космической радиолинии: развитие систем СВЧ радиосвязи / Ю.В Невзоров, О.И. Козак, О.В. Васильев // Электросвязь. – 2012. – №8. – С. 29–31.
89. Концепция построения разновысотной многоспутниковой системы связи с мобильными абонентами / Г.Н. Мальцев, К.Ю. Цветков, А.В. Родионов и др. // Труды Военно-космической академии имени А.Ф. Можайского; под ред. М.М. Пенькова. – СПб.: ВКА им. А.Ф. Можайского, 2011. – Выпуск № 630. – С. 5-10.
90. Концепция построения разновысотной многоспутниковой системы связи с мобильными абонентами: варианты реализации бортового коммуникационного оборудования спутника-ретранслятора / К.Ю. Цветков, А.В. Родионов, А.Ф. Акмолов

и др. // Труды Военно-космической академии имени А.Ф. Можайского; под ред. М.М. Пенькова. – СПб.: ВКА им. А.Ф. Можайского, 2012. – Выпуск № 635. – С. 5-13.

91. Фатеев, В.Ф. Инфраструктура малых космических аппаратов / В.Ф. Фатеев. – М.: Радиотехника, 2011. – 432 с.
92. Концепция построения разновысотной многоспутниковой системы связи с мобильными абонентами: канал управления кластера / К.Ю. Цветков, А.В. Родионов, А.Ф. Акмолов и др. // Труды Военно-космической академии имени А.Ф. Можайского; под ред. М.М. Пенькова. – СПб.: ВКА им. А.Ф. Можайского, 2011. – Выпуск № 632. – С. 5-10.
93. ИКАО Doc 10007 AN-Conf/12. Двенадцатая аэронавигационная конференция. – Монреаль, 2012. – 596 с.
94. Ariza-Quintana, A. Implementation of MANET routing protocols on OMNeT++ / A. Ariza-Quintana, E. Casilari, A. Triviño Cabrera // in proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops “Simutools’08”. – 2008.
95. Рекомендация МСЭ-R P.525-2 (1994 г.), *Расчет ослабления в свободном пространстве*.
96. MiXiM – The physical layer. An architecture overview / K. Wessel, M. Swigulski, A. Köpke, D. Willkomm //in Proceeding of the 2nd International Workshop on OMNeT++. – Rome, 2009. – P. 1 – 8.
97. Таненбаум, Э., Компьютерные сети / Э. Таненбаум, Д. Уэзеролл. – 5-е изд. — СПб.: Питер, 2012. — 960 с.
98. Li, Y. Rules of designing routing metrics for greedy, face, and combined greedy-face routing / Y. Li, Y. Yang, X. Lu // IEEE Transactions on Mobile Computing. – 2010. – vol. 9. – Issue No. 04. – P. 1-9.
99. Yang, Y. Design Guidelines for Routing Metrics in Multihop Wireless Networks / Y. Yang, J. Wang // IEEE INFOCOM The 27th Conference on Computer Communications. – 2008. – 10.1109/INFOCOM.2008.222.
100. ГОСТ Р ИСО/МЭК 3309-98 (1998 г.), Информационная технология. Передача данных и обмен информацией между системами. Процедуры управления звеном данных верхнего уровня. Структура кадра.
101. Галкин, В.А. Телекоммуникации и Сети / В.А. Галкин, Ю.А. Григорьев. — М.: МГТУ им. Н. Э. Баумана, 2003. – 608 с.

102. Олвейн, В. Структура и реализация современной технологии MPLS. Руководство Cisco / В. Олвейн. — М.: Вильямс, 2004. — 480 с.

103. Шнайер, Б. Прикладная криптография. Протоколы, алгоритмы и исходные тексты на языке С / Б. Шнайер. – 2-е изд. – М.: Триумф, 2002. – 816 с.

104. Рябко, Б.Я. Криптографические методы защиты информации: учебное пособие для вузов / Б.Я.Рябко, А.Н.Фионов. – М.: Горячая линия – Телеком, 2005. – С. 98 – 103.

105. Благодарный, Г. Национальная забава – догонять. Почему в России система автоматического зависимого наблюдения внедряется со скрипом? [Электронный ресурс] / Г. Благодарный // Транспорт России. Всероссийская транспортная еженедельная информационно-аналитическая газета. – 2009. URL: http://transportrussia.ru/item/18-natsionalnaya-zabava-dogonyat.html (дата обращения: 03.08.2017).

ПРИЛОЖЕНИЕ А. ПРОГРАММНЫЙ КОД ОСНОВНЫХ ФУНКЦИЙ КОМПЬЮТЕРНОЙ МОДЕЛИ

А.1 Код программной реализации выполняющий резервирование слота в структуре STDMA кадра для последующей передачи данных:

```
void VDLMacLayer::CreateReserv(cMessage* msg) { int ResType; int ResFor; int NR; if (msg == MacQueueReserv) { debugEV << "Starting procedure for MAC queue reservation. \n"; if(MacQueue.front().pkt->getKind() == InterSub_NetwMsg) { debugEV << "A reservation for SO network" << endl; ResType = IncremPeriodBroadRes; NR = 1; ResFor = InterSub_NetwMsg; } } else { VSSLMECntrl* vlcntrl = static_cast<VSSLMECntrl*>(msg); ResType = vlcntrl->getResRequestType(); debugEV << "Reservation for ADS-B function. \n"; NR = vlcntrl->getBroadcastFrequency();ResFor = InterSub_ADSBmsg; } switch(ResType) { case IncrementBroadReserv: break; case IncremPeriodBroadRes: case PeriodicBroadReserv: { debugEV << "Creating reservation for periodical broadcast application\n"; PerBroadParams.V11 = NR; int *pNomSltPos = new int[NR]; //array for nominal slots numbers int SelectRange = ((PerBroadParams.V12)/2.0)*(M1/PerBroadParams.V11); debugEV << "Selection range = " << SelectRange << " \n"; if(ResType == IncremPeriodBroadRes) pNomSltPos[0] = CurrSlot + 1 + 1 + SelectRange; else pNomSltPos[0] = int((M1/(2.0*NR)) + 0.5); for (int i = 1; i < NR; i++) { pNomSltPos[i] = pNomSltPos[i-1] + (M1/NR); } bool SearchFailed = false; for(int i = 0; (i < NR) && !SearchFailed; i++) { std::vector<int> SltLvl0; int begin = pNomSltPos[i] - SelectRange; if (begin < 0) begin = 0; int end = pNomSltPos[i] + SelectRange; if (end > 4500) end = 4500; for(int j = begin; j <= end; j { if(SlotMap[j].SlotState == EspeciallyBusy) { continue; } else if(SlotMap[j].reserved == false) { SltLvl0.push_back(j); } } if(SltLvl0.empty()) { SearchFailed = true; } } if(!SearchFailed) { for(int i = 0; i < NR; i++) { std::vector<int> SltLvl0; std::vector<int> SltLvl1; std::vector<int> SltLvl2; std::vector<int> SltLvl3; std::vector<int> SltLvl4; int begin = pNomSltPos[i] - SelectRange; if (begin < 0) begin = 0; int end = pNomSltPos[i] + SelectRange; if (end > 4500) end = 4500; for(int j = begin; j <= end; j++) { if(SlotMap[j].SlotState == EspeciallyBusy) { continue; } else if(SlotMap[j].reserved == false) { SltLvl0.push_back(j); } else if(isCCIProtect(SlotMap + j) && (FindDistance(SlotMap[j].Src) >= (380*NM))) { SltLvl1.push_back(j); } else if((SlotMap[j].ResIsBroadcast == true) && (FindDistance(SlotMap[j].Src) >= (380*NM { SltLvl2.push_back(j); } else if((FindDistance(SlotMap[j].Src) >= (0*NM)) && (FindDistance(SlotMap[j].Src) <= (380*NM))) { //SlotMap[j].level = level3; SltLvl3.push_back(j); } else if((FindDistance(SlotMap[j].Src)) > (380*NM)) { SltLvl4.push_back(j); } } debugEV << "Slots of level 0 founded for nominal slot " << pNomSltPos[i] << ": " << SltLvl0.size() << " \n"; debugEV << "Slots of level 1 founded for nominal slot " << pNomSltPos[i] << ": " << SltLvl1.size() << " \n"; debugEV << "Slots of level 2 founded for nominal slot " << pNomSltPos[i] << ": " << SltLvl2.size() << " \n"; debugEV << "Slots of level 3 founded for nominal slot " << pNomSltPos[i] << ": " << SltLvl3.size() << " \n"; debugEV << "Slots of level 4 founded for nominal slot " << pNomSltPos[i] << ": " << SltLvl4.size() << " \n"; if(SltLvl0.size() >= QoSParDefault.Q4) { int SmallestRSSI = FindBestSlot(SltLvl0); SlotMap[SmallestRSSI].NomSltPos = pNomSltPos[i]; SetSlotForReserv(SmallestRSSI, PeriodicBroadReserv); if((SlotMap[SmallestRSSI].ResByNumOfFrames < 4) && (ResFor == InterSub_ADSBmsg))
```

```
SlotMap[SmallestRSSI].FtSltPos = FindFtSlot(SmallestRSSI);
SlotMap[SmallestRSSI].ReservedFor = ResFor;  SearchFailed = false;  debugEV << "Slot for
reservation of NominalSlot " << pNomSltPos[i] << " is " << SmallestRSSI << " \n";  } else
if(((SltLvl0.size() + SltLvl1.size() + SltLvl2.size() + SltLvl3.size() + SltLvl4.size()) >=
QoSParDefault.Q4) && !(SltLvl0.empty()))  {  {  int SmallestRSSI = FindBestSlot(SltLvl0);
SlotMap[SmallestRSSI].NomSltPos = pNomSltPos[i];  SetSlotForReserv(SmallestRSSI,
PeriodicBroadReserv);  if((SlotMap[SmallestRSSI].ResByNumOfFrames < 4) && (ResFor ==
InterSub_ADSBmsg))  SlotMap[SmallestRSSI].FtSltPos = FindFtSlot(SmallestRSSI);
SlotMap[SmallestRSSI].ReservedFor = ResFor;  SearchFailed = false;  debugEV << "Slot for
reservation of NominalSlot " << pNomSltPos[i] << " is " << SmallestRSSI << " \n";  } }  }  }
debugEV << SearchFailed << endl; if(SearchFailed) //shift NSs to the right  {  bool SearchFailed1
= true;  {  bool MaxReached = false; //var to determine if max right position was reached by SI
while(SearchFailed1 && (!MaxReached))  {  int CorrShift = 0; //corrected shift value if selection
range occupies the area father than 4999 (SlotMap max position)  for(int i = 0; i < NR; i++)  {
pNomSltPos[i] = pNomSltPos[i] + ((2*SelectRange) + 1); //shift NS position to double SR plus 1,
to get to the undiscovered range  if((i == (NR-1)) && ((pNomSltPos[i] + SelectRange) > 4999)) //if
SR gets to the area father than 4999  {  MaxReached = true;  CorrShift = (SelectRange +
pNomSltPos[i]) - 4999; //assign a value to corrected shift for other NS  pNomSltPos[i] = 4999 -
SelectRange;  }  if(CorrShift) //if CorShift != 0, then there appears a situation, when SR reached
negative area  {  for(int j = 0; j < (NR-1); j++)  {  pNomSltPos[j] = pNomSltPos[j] - CorrShift;  }  }
}  SearchFailed1 = false;  for(int i = 0; (i < NR) && !SearchFailed1; i++)  {  std::vector<int>
SltLvl0;  int begin = pNomSltPos[i] - SelectRange;  if (begin < 0)  begin = 0;  int end =
pNomSltPos[i] + SelectRange;  if (end > 4500)  end = 4500;  for(int j = begin; j <= end; j {
if(SlotMap[j].SlotState == EspeciallyBusy)  {  continue;//skip this slot  }  else
if(SlotMap[j].reserved == false)//if slot is unreserved  {  //SlotMap[j].level = level0;
SltLvl0.push_back(j); }  }  if(SltLvl0.empty())  {  SearchFailed1 = true;  }  }  }  }
if(SearchFailed1)  if(ResType == IncremPeriodBroadRes)  pNomSltPos[0] = CurrSlot + 1 + 1 +
SelectRange;  else  pNomSltPos[0] = int((M1/(2.0*NR)) + 0.5); //first position  for (int i = 1; i <
NR; i++) //next positions  {  pNomSltPos[i] = pNomSltPos[i-1] + (M1/NR);  }  //SearchFailed =
false;  {  bool ZeroReached = false;  while(SearchFailed1 && (!ZeroReached))//if we didn't found
any slots of level 0, we will shift nominal slots to the left at the double selection range  {  int
CorrShift = 0; for(int i = 0; i < NR; i++)  {  if(CorrShift && (i != 0)) {  pNomSltPos[i] =
pNomSltPos[i] - CorrShift;  }  else {  pNomSltPos[i] = pNomSltPos[i] - ((2*SelectRange) + 1);  }
if((i == 0) && ((pNomSltPos[i] - SelectRange) < 0))  {  ZeroReached = true;  CorrShift =
((2*SelectRange) + 1) - (SelectRange - pNomSltPos[i]); pNomSltPos[i] = SelectRange;  }  }
SearchFailed1 = false;  for(int i = 0; (i < NR) && !SearchFailed1; i++)  {  std::vector<int>
SltLvl0;  int begin = pNomSltPos[i] - SelectRange;  if (begin < 0)  begin = 0;  int end =
pNomSltPos[i] + SelectRange;  if (end > 4500)  end = 4500;  for(int j = begin; j <= end; j++)  {
if(SlotMap[j].SlotState == EspeciallyBusy)  {  continue;  }  else if(SlotMap[j].reserved == false)  {
SltLvl0.push_back(j); }  }  if(SltLvl0.empty())  {  SearchFailed1 = true;  }  }  }  }
if(!SearchFailed1)  {  for(int i = 0; i < NR; i++)  {  std::vector<int> SltLvl0;  std::vector<int>
SltLvl1;  std::vector<int> SltLvl2;  std::vector<int> SltLvl3;  std::vector<int> SltLvl4; int begin =
pNomSltPos[i] - SelectRange;  if (begin < 0)  begin = 0;  //setting the end of select range  int end =
pNomSltPos[i] + SelectRange;  if (end > 4500)  end = 4500;  for(int j = begin; j <= end; j++)  {
if(SlotMap[j].SlotState == EspeciallyBusy)  {  Continue;  }  else if(SlotMap[j].reserved == false)  {
```

```
SltLvl0.push_back(j); } else if(isCCIProtect(SlotMap + j) && (FindDistance(SlotMap[j].Src) >=
(380*NM))) { SltLvl1.push_back(j); } else if((SlotMap[j].ResIsBroadcast == true) &&
(FindDistance(SlotMap[j].Src) >= (380*NM))) { SltLvl2.push_back(j); } else
if((FindDistance(SlotMap[j].Src) >= (0*NM)) && (FindDistance(SlotMap[j].Src) <= (380*NM)))
{ SltLvl3.push_back(j); } else if((FindDistance(SlotMap[j].Src)) > (380*NM)) {
SltLvl4.push_back(j); } } debugEV << "Slots of level 0 founded for nominal slot " <<
pNomSltPos[i] << ": " << SltLvl0.size() << " \n"; debugEV << "Slots of level 1 founded for
nominal slot " << pNomSltPos[i] << ": " << SltLvl1.size() << " \n"; debugEV << "Slots of level 2
founded for nominal slot " << pNomSltPos[i] << ": " << SltLvl2.size() << " \n"; debugEV <<
"Slots of level 3 founded for nominal slot " << pNomSltPos[i] << ": " << SltLvl3.size() << " \n";
debugEV << "Slots of level 4 founded for nominal slot " << pNomSltPos[i] << ": " <<
SltLvl4.size() << " \n"; //now we must check the Q4 parameter if(SltLvl0.size() >=
QoSParDefault.Q4) { int SmallestRSSI = FindBestSlot(SltLvl0);
SlotMap[SmallestRSSI].NomSltPos = pNomSltPos[i]; SetSlotForReserv(SmallestRSSI,
PeriodicBroadReserv); if((SlotMap[SmallestRSSI].ResByNumOfFrames < 4) && (ResFor ==
InterSub_ADSBmsg)) SlotMap[SmallestRSSI].FtSltPos = FindFtSlot(SmallestRSSI);
SlotMap[SmallestRSSI].ReservedFor = ResFor; debugEV << "Slot for reservation of NominalSlot "
<< pNomSltPos[i] << " is " << SmallestRSSI << " \n"; } else if(((SltLvl0.size() + SltLvl1.size() +
SltLvl2.size() + SltLvl3.size() + SltLvl4.size()) >= QoSParDefault.Q4) && !(SltLvl0.empty())) {
int SmallestRSSI = FindBestSlot(SltLvl0); SlotMap[SmallestRSSI].NomSltPos = pNomSltPos[i];
SetSlotForReserv(SmallestRSSI, PeriodicBroadReserv);
if((SlotMap[SmallestRSSI].ResByNumOfFrames < 4) && (ResFor == InterSub_ADSBmsg))
SlotMap[SmallestRSSI].FtSltPos = FindFtSlot(SmallestRSSI);
SlotMap[SmallestRSSI].ReservedFor = ResFor; //SearchFailed = false; debugEV << "Slot for
reservation of NominalSlot " << pNomSltPos[i] << " is " << SmallestRSSI << " \n"; } } } else
//shift NSs to the right { bool SearchFailed2 = true; if(ResType == IncremPeriodBroadRes)
pNomSltPos[0] = CurrSlot + 1 + 1 + SelectRange; else pNomSltPos[0] = int((M1/(2.0*NR)) +
0.5); //first position for (int i = 1; i < NR; i++) //next positions { pNomSltPos[i] = pNomSltPos[i-
1] + (M1/NR); } { bool MaxReached = false; while(SearchFailed2 && (!MaxReached)) { int
CorrShift = 0; for(int i = 0; i < NR; i++) { pNomSltPos[i] = pNomSltPos[i] + ((2*SelectRange) +
1); if((i == (NR-1)) && ((pNomSltPos[i] + SelectRange) > 4999)) { MaxReached = true;
CorrShift = (SelectRange + pNomSltPos[i]) - 4999; pNomSltPos[i] = 4999 - SelectRange; }
if(CorrShift) //if CorShift != 0, then there appears a situation, when SR reached negative area {
for(int j = 0; j < (NR-1); j++) { pNomSltPos[j] = pNomSltPos[j] - CorrShift; } } }
SearchFailed2 = false; for(int i = 0; (i < NR) && !SearchFailed2; i++) { std::vector<int>
SltLvl1; int begin = pNomSltPos[i] - SelectRange; if (begin < 0) begin = 0; int end =
pNomSltPos[i] + SelectRange; if (end > 4500) end = 4500; for(int j = begin; j <= end; j++) {
if(SlotMap[j].SlotState == EspeciallyBusy) { continue;//skip this slot } else
if(isCCIProtect(SlotMap + j) && (FindDistance(SlotMap[j].Src) >= (380*NM))) {
//SlotMap[j].level = level1; SltLvl1.push_back(j); } } if(SltLvl1.empty()) { SearchFailed2 =
true; } } } } if(SearchFailed2) //moving NS to the left, if it was failed at right { if(ResType ==
IncremPeriodBroadRes) pNomSltPos[0] = CurrSlot + 1 + 1 + SelectRange; else pNomSltPos[0] =
int((M1/(2.0*NR)) + 0.5); for (int i = 1; i < NR; i++) //next positions { pNomSltPos[i] =
pNomSltPos[i-1] + (M1/NR); } { bool ZeroReached = false; while(SearchFailed2 &&
(!ZeroReached)) { int CorrShift = 0; for(int i = 0; i < NR; i++) { if(CorrShift && (i != 0)) {
```

```
pNomSltPos[i] = pNomSltPos[i] - CorrShift; } else { pNomSltPos[i] = pNomSltPos[i] -
((2*SelectRange) + 1); } if((i == 0) && ((pNomSltPos[i] - SelectRange) < 0)) { ZeroReached =
true; CorrShift = ((2*SelectRange) + 1) - (SelectRange - pNomSltPos[i]); pNomSltPos[i] =
SelectRange; } } SearchFailed2 = false; for(int i = 0; (i < NR) && !SearchFailed2; i++) {
std::vector<int> SltLvl1; int begin = pNomSltPos[i] - SelectRange; if (begin < 0) begin = 0; int
end = pNomSltPos[i] + SelectRange; if (end > 4500) end = 4500; for(int j = begin; j <= end; j++)
{ if(SlotMap[j].SlotState == EspeciallyBusy) { continue; } else if(isCCIProtect(SlotMap + j)
&& (FindDistance(SlotMap[j].Src) >= (380*NM))) { //SlotMap[j].level = level1;
SltLvl1.push_back(j); } } if(SltLvl1.empty()) { SearchFailed2 = true; } } } }
if(!SearchFailed2) { for(int i = 0; i < NR; i++) { std::vector<int> SltLvl0; std::vector<int>
SltLvl1; std::vector<int> SltLvl2; std::vector<int> SltLvl3; std::vector<int> SltLvl4; int begin =
pNomSltPos[i] - SelectRange; if (begin < 0) begin = 0; //setting the end of select range int end =
pNomSltPos[i] + SelectRange; if (end > 4500) end = 4500; for(int j = begin; j <= end; j++) {
if(SlotMap[j].SlotState == EspeciallyBusy) { continue;//skip this slot } else
if(SlotMap[j].reserved == false) { //SlotMap[j].level = level0; SltLvl0.push_back(j); } else
if(isCCIProtect(SlotMap + j) && (FindDistance(SlotMap[j].Src) >= (380*NM))) {
SltLvl1.push_back(j); } else if((SlotMap[j].ResIsBroadcast == true) &&
(FindDistance(SlotMap[j].Src) >= (380*NM))) { SltLvl2.push_back(j); } else
if((FindDistance(SlotMap[j].Src) >= (0*NM)) && (FindDistance(SlotMap[j].Src) <= (380*NM)))
{ SltLvl3.push_back(j); } else if((FindDistance(SlotMap[j].Src)) > (380*NM)) {
SltLvl4.push_back(j); } } debugEV << "Slots of level 0 founded for nominal slot " <<
pNomSltPos[i] << ": " << SltLvl0.size() << " \n"; debugEV << "Slots of level 1 founded for
nominal slot " << pNomSltPos[i] << ": " << SltLvl1.size() << " \n"; debugEV << "Slots of level 2
founded for nominal slot " << pNomSltPos[i] << ": " << SltLvl2.size() << " \n"; debugEV <<
"Slots of level 3 founded for nominal slot " << pNomSltPos[i] << ": " << SltLvl3.size() << " \n";
debugEV << "Slots of level 4 founded for nominal slot " << pNomSltPos[i] << ": " <<
SltLvl4.size() << " \n"; if(SltLvl1.size() >= QoSParDefault.Q4)//in every branch we must find the
best slot for transmission { int SmallestRSSI = FindBestSlot(SltLvl1);
SlotMap[SmallestRSSI].NomSltPos = pNomSltPos[i]; SetSlotForReserv(SmallestRSSI,
PeriodicBroadReserv); if((SlotMap[SmallestRSSI].ResByNumOfFrames < 4) && (ResFor ==
InterSub_ADSBmsg)) SlotMap[SmallestRSSI].FtSltPos = FindFtSlot(SmallestRSSI);
SlotMap[SmallestRSSI].ReservedFor = ResFor; //SearchFailed = false; debugEV << "Slot for
reservation of NominalSlot " << pNomSltPos[i] << " is " << SmallestRSSI << " \n"; } the source of
that signal else if(((SltLvl0.size() + SltLvl1.size() + SltLvl2.size() + SltLvl3.size() + SltLvl4.size())
>= QoSParDefault.Q4) && !(SltLvl1.empty())) { int SmallestRSSI = FindBestSlot(SltLvl1);
SlotMap[SmallestRSSI].NomSltPos = pNomSltPos[i]; SetSlotForReserv(SmallestRSSI,
PeriodicBroadReserv); if((SlotMap[SmallestRSSI].ResByNumOfFrames < 4) && (ResFor ==
InterSub_ADSBmsg)) SlotMap[SmallestRSSI].FtSltPos = FindFtSlot(SmallestRSSI);
SlotMap[SmallestRSSI].ReservedFor = ResFor; //SearchFailed = false; debugEV << "Slot for
reservation of NominalSlot " << pNomSltPos[i] << " is " << SmallestRSSI << " \n"; } } } else {
bool SearchFailed3 = true; if(ResType == IncremPeriodBroadRes) pNomSltPos[0] = CurrSlot +
1 + 1 + SelectRange; else pNomSltPos[0] = int((M1/(2.0*NR)) + 0.5); for (int i = 1; i < NR; i++)
//next positions { pNomSltPos[i] = pNomSltPos[i-1] + (M1/NR); } { bool MaxReached = false;
while(SearchFailed3 && (!MaxReached)) { int CorrShift = 0; for(int i = 0; i < NR; i++) {
pNomSltPos[i] = pNomSltPos[i] + ((2*SelectRange) + 1); if((i == (NR-1)) && ((pNomSltPos[i] +
```

```
SelectRange) > 4999)) { MaxReached = true; CorrShift = (SelectRange + pNomSltPos[i]) -
4999; pNomSltPos[i] = 4999 - SelectRange; } if(CorrShift) { for(int j = 0; j < (NR-1); j++) {
pNomSltPos[j] = pNomSltPos[j] - CorrShift; } } } SearchFailed3 = false; for(int i = 0; (i < NR)
&& !SearchFailed3; i++) { std::vector<int> SltLvl2; int begin = pNomSltPos[i] - SelectRange; if
(begin < 0) begin = 0; int end = pNomSltPos[i] + SelectRange; if (end > 4500) end = 4500;
for(int j = begin; j <= end; j++) { if(SlotMap[j].SlotState == EspeciallyBusy) { continue;//skip
this slot } else if((SlotMap[j].ResIsBroadcast == true) && (FindDistance(SlotMap[j].Src) >=
(380*NM))) { SltLvl2.push_back(j); } } if(SltLvl2.empty()) { SearchFailed3 = true; } } } }
if(SearchFailed3) { if(ResType == IncremPeriodBroadRes) pNomSltPos[0] = CurrSlot + 1 + 1 +
SelectRange; else pNomSltPos[0] = int((M1/(2.0*NR)) + 0.5); for (int i = 1; i < NR; i++) {
pNomSltPos[i] = pNomSltPos[i-1] + (M1/NR); } //SearchFailed = false; { bool ZeroReached =
false; while(SearchFailed3 && (!ZeroReached)) { int CorrShift = 0; //corrected shift value if
selection range occupies the area of negative numbers for(int i = 0; i < NR; i++) { if(CorrShift
&& (i != 0)) //if CorShift != 0, then there appears a situation, when SR reached negative area {
pNomSltPos[i] = pNomSltPos[i] - CorrShift; } else //else continue to shift NSs left to 2*SR + 1 {
pNomSltPos[i] = pNomSltPos[i] - ((2*SelectRange) + 1); } if((i == 0) && ((pNomSltPos[i] -
SelectRange) < 0)) { ZeroReached = true; CorrShift = ((2*SelectRange) + 1) - (SelectRange -
pNomSltPos[i]); pNomSltPos[i] = SelectRange; } } SearchFailed3 = false; for(int i = 0; (i <
NR) && !SearchFailed3; i++) { std::vector<int> SltLvl2; int begin = pNomSltPos[i] -
SelectRange; if (begin < 0) begin = 0; int end = pNomSltPos[i] + SelectRange; if (end > 4500)
end = 4500; for(int j = begin; j <= end; j++)//marker slots and put them in level arrays in selection
range of every nominal slot { if(SlotMap[j].SlotState == EspeciallyBusy) { continue;//skip this
slot } else if((SlotMap[j].ResIsBroadcast == true) && (FindDistance(SlotMap[j].Src) >=
(380*NM)))//broadcast reservation and the distance >= Q2b { //SlotMap[j].level = level2;
SltLvl2.push_back(j); } } if(SltLvl2.empty()) { SearchFailed3 = true; } } } }
if(!SearchFailed3) { for(int i = 0; i < NR; i++) { std::vector<int> SltLvl0; std::vector<int>
SltLvl1; std::vector<int> SltLvl2; std::vector<int> SltLvl3; std::vector<int> SltLvl4; int begin =
pNomSltPos[i] - SelectRange; if (begin < 0) begin = 0; int end = pNomSltPos[i] + SelectRange; if
(end > 4500) end = 4500; for(int j = begin; j <= end; j++) { if(SlotMap[j].SlotState ==
EspeciallyBusy) { continue; } else if(SlotMap[j].reserved == false) { SltLvl0.push_back(j); }
else if(isCCIProtect(SlotMap + j) && (FindDistance(SlotMap[j].Src) >= (380*NM))) {
//SlotMap[j].level = level1; SltLvl1.push_back(j); } else if((SlotMap[j].ResIsBroadcast == true)
&& (FindDistance(SlotMap[j].Src) >= (380*NM)))//broadcast reservation and the distance >= Q2b
{ //SlotMap[j].level = level2; SltLvl2.push_back(j); } else if((FindDistance(SlotMap[j].Src) >=
(0*NM)) && (FindDistance(SlotMap[j].Src) <= (380*NM))) { SltLvl3.push_back(j); } else
if((FindDistance(SlotMap[j].Src)) > (380*NM))/ { SltLvl4.push_back(j); } } debugEV << "Slots
of level 0 founded for nominal slot " << pNomSltPos[i] << ": " << SltLvl0.size() << " \n";
debugEV << "Slots of level 1 founded for nominal slot " << pNomSltPos[i] << ": " <<
SltLvl1.size() << " \n"; debugEV << "Slots of level 2 founded for nominal slot " << pNomSltPos[i]
<< ": " << SltLvl2.size() << " \n"; debugEV << "Slots of level 3 founded for nominal slot " <<
pNomSltPos[i] << ": " << SltLvl3.size() << " \n"; debugEV << "Slots of level 4 founded for
nominal slot " << pNomSltPos[i] << ": " << SltLvl4.size() << " \n"; if(SltLvl2.size() >=
QoSParDefault.Q4)//in every branch we must find the best slot for transmission { int
SmallestRSSI = FindBestSlot(SltLvl2); SlotMap[SmallestRSSI].NomSltPos = pNomSltPos[i];
SetSlotForReserv(SmallestRSSI, PeriodicBroadReserv);
```

```
if((SlotMap[SmallestRSSI].ResByNumOfFrames < 4) && (ResFor == InterSub_ADSBmsg))
SlotMap[SmallestRSSI].FtSltPos = FindFtSlot(SmallestRSSI);
SlotMap[SmallestRSSI].ReservedFor = ResFor;  //SearchFailed = false;  debugEV << "Slot for
reservation of NominalSlot " << pNomSltPos[i] << " is " << SmallestRSSI << " \n";  } else
if(((SltLvl0.size() + SltLvl1.size() + SltLvl2.size() + SltLvl3.size() + SltLvl4.size()) >=
QoSParDefault.Q4) && !(SltLvl2.empty()))  {  int SmallestRSSI = FindBestSlot(SltLvl2);
SlotMap[SmallestRSSI].NomSltPos = pNomSltPos[i];  SetSlotForReserv(SmallestRSSI,
PeriodicBroadReserv); if((SlotMap[SmallestRSSI].ResByNumOfFrames < 4) && (ResFor ==
InterSub_ADSBmsg))  SlotMap[SmallestRSSI].FtSltPos = FindFtSlot(SmallestRSSI);
SlotMap[SmallestRSSI].ReservedFor = ResFor;  //SearchFailed = false;  debugEV << "Slot for
reservation of NominalSlot " << pNomSltPos[i] << " is " << SmallestRSSI << " \n";  }  }  }  else
//shift NSs to the right  {  bool SearchFailed4 = true;  if(ResType == IncremPeriodBroadRes)
pNomSltPos[0] = CurrSlot + 1 + 1 + SelectRange;  else  pNomSltPos[0] = int((M1/(2.0*NR)) +
0.5); //first position  for (int i = 1; i < NR; i++) //next positions  {  pNomSltPos[i] = pNomSltPos[i-
1] + (M1/NR);  }  {  bool MaxReached = false;  while(SearchFailed4 && (!MaxReached))  {  int
CorrShift = 0;  for(int i = 0; i < NR; i++)  {  pNomSltPos[i] = pNomSltPos[i] + ((2*SelectRange)
+ 1); if((i == (NR-1)) && ((pNomSltPos[i] + SelectRange) > 4999)) {  MaxReached = true;
CorrShift = (SelectRange + pNomSltPos[i]) - 4999;  pNomSltPos[i] = 4999 - SelectRange;  }
if(CorrShift) {  for(int j = 0; j < (NR-1); j++)  {  pNomSltPos[j] = pNomSltPos[j] - CorrShift;  }  }
}  SearchFailed4 = false;  for(int i = 0; (i < NR) && !SearchFailed4; i++)  {  std::vector<int>
SltLvl3;  int begin = pNomSltPos[i] - SelectRange;  if (begin < 0)  begin = 0;  int end =
pNomSltPos[i] + SelectRange;  if (end > 4500)  end = 4500;  for(int j = begin; j <= end; j++)  {
if(SlotMap[j].SlotState == EspeciallyBusy)  {  continue;//skip this slot  }  else
if((FindDistance(SlotMap[j].Src) >= (0*NM)) && (FindDistance(SlotMap[j].Src) <= (380*NM)))
{  //SlotMap[j].level = level3;  SltLvl3.push_back(j);  }  }  if(SltLvl3.empty())  {  SearchFailed4 =
true;  }  }  }  }  if(SearchFailed4)  {  if(ResType == IncremPeriodBroadRes)  pNomSltPos[0] =
CurrSlot + 1 + 1 + SelectRange;  else  pNomSltPos[0] = int((M1/(2.0*NR)) + 0.5); //first position
for (int i = 1; i < NR; i++) //next positions  {  pNomSltPos[i] = pNomSltPos[i-1] + (M1/NR);  }  {
bool ZeroReached = false;  while(SearchFailed4 && (!ZeroReached))  {  int CorrShift = 0;
for(int i = 0; i < NR; i++)  {  if(CorrShift && (i != 0))  {  pNomSltPos[i] = pNomSltPos[i] -
CorrShift;  }  else //else continue to shift NSs left to 2*SR + 1  {  pNomSltPos[i] = pNomSltPos[i] -
((2*SelectRange) + 1);  }  if((i == 0) && ((pNomSltPos[i] - SelectRange) < 0)) //if SR gets to the
negative area  {  ZeroReached = true;  CorrShift = ((2*SelectRange) + 1) - (SelectRange -
pNomSltPos[i]); //assign a value to corrected shift for other NS  pNomSltPos[i] = SelectRange;  }
}  SearchFailed4 = false;  for(int i = 0; (i < NR) && !SearchFailed4; i++)  {  std::vector<int>
SltLvl3;  int begin = pNomSltPos[i] - SelectRange;  if (begin < 0)  begin = 0;  int end =
pNomSltPos[i] + SelectRange;  if (end > 4500)  end = 4500;  for(int j = begin; j <= end; j++)  {
if(SlotMap[j].SlotState == EspeciallyBusy)  {  continue;//skip this slot  }  else
if((FindDistance(SlotMap[j].Src) >= (0*NM)) && (FindDistance(SlotMap[j].Src) <= (380*NM)))
{  //SlotMap[j].level = level3;  SltLvl3.push_back(j);  }  }  if(SltLvl3.empty())  {  SearchFailed4 =
true;  }  }  }  }  if(!SearchFailed4)  {  for(int i = 0; i < NR; i++)//beginning of slot lvl marker  {
std::vector<int> SltLvl0; //vectors for positions of slots of different levels  std::vector<int> SltLvl1;
std::vector<int> SltLvl2;  std::vector<int> SltLvl3;  std::vector<int> SltLvl4; int begin =
pNomSltPos[i] - SelectRange;  if (begin < 0)  begin = 0;  //setting the end of select range  int end =
pNomSltPos[i] + SelectRange;  if (end > 4500)  end = 4500;  for(int j = begin; j <= end; j++)  {
```

```
if(SlotMap[j].SlotState == EspeciallyBusy) { continue;//skip this slot } else
if(SlotMap[j].reserved == false)//if slot is unreserved { //SlotMap[j].level = level0;
SltLvl0.push_back(j); } else if(isCCIProtect(SlotMap + j) && (FindDistance(SlotMap[j].Src) >=
(380*NM))) { SltLvl1.push_back(j); } else if((SlotMap[j].ResIsBroadcast == true) &&
(FindDistance(SlotMap[j].Src) >= (380*NM))) { SltLvl2.push_back(j); } else
if((FindDistance(SlotMap[j].Src) >= (0*NM)) && (FindDistance(SlotMap[j].Src) <= (380*NM)))
{ SltLvl3.push_back(j); } else if((FindDistance(SlotMap[j].Src)) > (380*NM)) {
SltLvl4.push_back(j); } } debugEV << "Slots of level 0 founded for nominal slot " <<
pNomSltPos[i] << ": " << SltLvl0.size() << " \n"; debugEV << "Slots of level 1 founded for
nominal slot " << pNomSltPos[i] << ": " << SltLvl1.size() << " \n"; debugEV << "Slots of level 2
founded for nominal slot " << pNomSltPos[i] << ": " << SltLvl2.size() << " \n"; debugEV <<
"Slots of level 3 founded for nominal slot " << pNomSltPos[i] << ": " << SltLvl3.size() << " \n";
debugEV << "Slots of level 4 founded for nominal slot " << pNomSltPos[i] << ": " <<
SltLvl4.size() << " \n"; //now we must check the Q4 parameter if(SltLvl3.size() >=
QoSParDefault.Q4) { int SmallestRSSI = FindBestSlot(SltLvl3);
SlotMap[SmallestRSSI].NomSltPos = pNomSltPos[i]; SetSlotForReserv(SmallestRSSI,
PeriodicBroadReserv); if((SlotMap[SmallestRSSI].ResByNumOfFrames < 4) && (ResFor ==
InterSub_ADSBmsg)) SlotMap[SmallestRSSI].FtSltPos = FindFtSlot(SmallestRSSI);
SlotMap[SmallestRSSI].ReservedFor = ResFor; //SearchFailed = false; debugEV << "Slot for
reservation of NominalSlot " << pNomSltPos[i] << " is " << SmallestRSSI << " \n"; } else
if(((SltLvl0.size() + SltLvl1.size() + SltLvl2.size() + SltLvl3.size() + SltLvl4.size()) >=
QoSParDefault.Q4) && !(SltLvl3.empty())) { int SmallestRSSI = FindBestSlot(SltLvl3);
SlotMap[SmallestRSSI].NomSltPos = pNomSltPos[i]; SetSlotForReserv(SmallestRSSI,
PeriodicBroadReserv); if((SlotMap[SmallestRSSI].ResByNumOfFrames < 4) && (ResFor ==
InterSub_ADSBmsg)) SlotMap[SmallestRSSI].FtSltPos = FindFtSlot(SmallestRSSI);
SlotMap[SmallestRSSI].ReservedFor = ResFor; //SearchFailed = false; debugEV << "Slot for
reservation of NominalSlot " << pNomSltPos[i] << " is " << SmallestRSSI << " \n"; } } } else {
bool SearchFailed5 = true; if(ResType == IncremPeriodBroadRes) pNomSltPos[0] = CurrSlot +
1 + 1 + SelectRange; else pNomSltPos[0] = int((M1/(2.0*NR)) + 0.5); //first position for (int i =
1; i < NR; i++) //next positions { pNomSltPos[i] = pNomSltPos[i-1] + (M1/NR); } { bool
MaxReached = false; while(SearchFailed5 && (!MaxReached)) { int CorrShift = 0; for(int i =
0; i < NR; i++) { pNomSltPos[i] = pNomSltPos[i] + ((2*SelectRange) + 1); if((i == (NR-1)) &&
((pNomSltPos[i] + SelectRange) > 4999)) { MaxReached = true; CorrShift = (SelectRange +
pNomSltPos[i]) - 4999; //assign a value to corrected shift for other NS pNomSltPos[i] = 4999 -
SelectRange; } if(CorrShift) { for(int j = 0; j < (NR-1); j++) { pNomSltPos[j] = pNomSltPos[j]
- CorrShift; } } } SearchFailed5 = false; for(int i = 0; (i < NR) && !SearchFailed5; i++) {
std::vector<int> SltLvl4; int begin = pNomSltPos[i] - SelectRange; if (begin < 0) begin = 0; int
end = pNomSltPos[i] + SelectRange; if (end > 4500) end = 4500; for(int j = begin; j <= end; j++)
{ if(SlotMap[j].SlotState == EspeciallyBusy) { continue;//skip this slot } else
if((FindDistance(SlotMap[j].Src)) > (380*NM)) { //SlotMap[j].level = level4;
SltLvl4.push_back(j); } } if(SltLvl4.empty()) { SearchFailed5 = true; } } } }
if(SearchFailed5) //moving NS to the left, if it was failed at right {//so at first we must shift NS to
their previous positions if(ResType == IncremPeriodBroadRes) pNomSltPos[0] = CurrSlot + 1 +
1 + SelectRange; else pNomSltPos[0] = int((M1/(2.0*NR)) + 0.5); //first position for (int i = 1; i
< NR; i++) //next positions { pNomSltPos[i] = pNomSltPos[i-1] + (M1/NR); } { bool
```

```
ZeroReached = false;  while(SearchFailed5 && (!ZeroReached)) { int CorrShift = 0; for(int i = 0; i < NR; i++) { if(CorrShift && (i != 0)) { pNomSltPos[i] = pNomSltPos[i] - CorrShift; } else //else continue to shift NSs left to 2*SR + 1 { pNomSltPos[i] = pNomSltPos[i] - ((2*SelectRange) + 1); } if((i == 0) && ((pNomSltPos[i] - SelectRange) < 0)) { ZeroReached = true; CorrShift = ((2*SelectRange) + 1) - (SelectRange - pNomSltPos[i]); //assign a value to corrected shift for other NS pNomSltPos[i] = SelectRange; } } SearchFailed5 = false; for(int i = 0; (i < NR) && !SearchFailed5; i++) { std::vector<int> SltLvl4; int begin = pNomSltPos[i] - SelectRange; if (begin < 0) begin = 0; int end = pNomSltPos[i] + SelectRange; if (end > 4500) end = 4500; for(int j = begin; j <= end; j++) { if(SlotMap[j].SlotState == EspeciallyBusy) { continue;//skip this slot } else if((FindDistance(SlotMap[j].Src)) > (380*NM)) { //SlotMap[j].level = level4; SltLvl4.push_back(j); } } if(SltLvl4.empty()) { SearchFailed5 = true; } } } } if(!SearchFailed5) { for(int i = 0; i < NR; i++)//beginning of slot lvl marker { std::vector<int> SltLvl0; //vectors for positions of slots of different levels std::vector<int> SltLvl1; std::vector<int> SltLvl2; std::vector<int> SltLvl3; std::vector<int> SltLvl4; int begin = pNomSltPos[i] - SelectRange; if (begin < 0) begin = 0; //setting the end of select range int end = pNomSltPos[i] + SelectRange; if (end > 4500) end = 4500; for(int j = begin; j <= end; j++) { if(SlotMap[j].SlotState == EspeciallyBusy) { continue;//skip this slot } else if(SlotMap[j].reserved == false)//if slot is unreserved { SltLvl0.push_back(j); } else if(isCCIProtect(SlotMap + j) && (FindDistance(SlotMap[j].Src) >= (380*NM))) { SltLvl1.push_back(j); } else if((SlotMap[j].ResIsBroadcast == true) && (FindDistance(SlotMap[j].Src) >= (380*NM))) { SltLvl2.push_back(j); } else if((FindDistance(SlotMap[j].Src) >= (0*NM)) && (FindDistance(SlotMap[j].Src) <= (380*NM))) { SltLvl3.push_back(j); } else if((FindDistance(SlotMap[j].Src)) > (380*NM)) { SltLvl4.push_back(j); } } debugEV << "Slots of level 0 founded for nominal slot " << pNomSltPos[i] << ": " << SltLvl0.size() << " \n"; debugEV << "Slots of level 1 founded for nominal slot " << pNomSltPos[i] << ": " << SltLvl1.size() << " \n"; debugEV << "Slots of level 2 founded for nominal slot " << pNomSltPos[i] << ": " << SltLvl2.size() << " \n"; debugEV << "Slots of level 3 founded for nominal slot " << pNomSltPos[i] << ": " << SltLvl3.size() << " \n"; debugEV << "Slots of level 4 founded for nominal slot " << pNomSltPos[i] << ": " << SltLvl4.size() << " \n"; //now we must check the Q4 parameter if(SltLvl4.size() >= QoSParDefault.Q4) { int SmallestRSSI = FindBestSlot(SltLvl4); SlotMap[SmallestRSSI].NomSltPos = pNomSltPos[i]; SetSlotForReserv(SmallestRSSI, PeriodicBroadReserv); if((SlotMap[SmallestRSSI].ResByNumOfFrames < 4) && (ResFor == InterSub_ADSBmsg)) SlotMap[SmallestRSSI].FtSltPos = FindFtSlot(SmallestRSSI); SlotMap[SmallestRSSI].ReservedFor = ResFor; //SearchFailed = false; debugEV << "Slot for reservation of NominalSlot " << pNomSltPos[i] << " is " << SmallestRSSI << " \n"; } else if(((SltLvl0.size() + SltLvl1.size() + SltLvl2.size() + SltLvl3.size() + SltLvl4.size()) >= QoSParDefault.Q4) && !(SltLvl4.empty())) {//if there is any slot of lvl0 //if(!(SltLvl0.empty())) int SmallestRSSI = FindBestSlot(SltLvl4); SlotMap[SmallestRSSI].NomSltPos = pNomSltPos[i]; SetSlotForReserv(SmallestRSSI, PeriodicBroadReserv); if((SlotMap[SmallestRSSI].ResByNumOfFrames < 4) && (ResFor == InterSub_ADSBmsg)) SlotMap[SmallestRSSI].FtSltPos = FindFtSlot(SmallestRSSI); SlotMap[SmallestRSSI].ReservedFor = ResFor; //SearchFailed = false; debugEV << "Slot for reservation of NominalSlot " << pNomSltPos[i] << " is " << SmallestRSSI << " \n"; } } } else {
```

```
debugEV << "Cannot find a slot for transmission!\n"; } } } } } } } } } } } if (msg != MacQueueReserv) delete msg; delete [] pNomSltPos; } break; } }
```

А. 2 Код программной реализации описывающий процедуру обработку сетевых сообщений на подуровне маршрутизации:

```
void VDLGRPfSAN::handleLowerMsg(cMessage *msg) { cObject* cInfo = msg->removeControlInfo(); LAddress::L2Type addr = MacToNetwControlInfo::getAddressFromControlInfo(cInfo); simtime_t TS = msg->getTimestamp(); std::vector<std::pair<LAddress::L2Type, LMEPkt*>>* pNHMsgs = pLME->getNetwHandleMsgs(); std::vector<VDLMacPkt*>* pMacNHMsgs = pMac->getNetwHandleMsgs(); int MsgType; for(std::vector<std::pair<LAddress::L2Type, LMEPkt*>>::iterator it = pNHMsgs->begin(); it != pNHMsgs->end(); it++) { if(((it->first) == addr) && (((it->second)->getTimestamp()) == TS)) MsgType = (it->second)->getID(); } if(MsgType == BSRetrMsg) { GRPfSAN_R_Pkt* BaseStatnPkt = static_cast<GRPfSAN_R_Pkt*>(msg); if(IsBaseStation) { debugEV << "Message from me or other BS!\n"; std::vector<std::pair<LAddress::L2Type, LMEPkt*>>::iterator DelIt = pNHMsgs->end(); for(std::vector<std::pair<LAddress::L2Type, LMEPkt*>>::iterator it = pNHMsgs->begin(); it != pNHMsgs->end(); it++) { if(((it->first) == addr) && (it->second->getTimestamp() == TS)) DelIt = it; } if(DelIt != pNHMsgs->end()) { debugEV << "Delete corresponding message part on LME layer. \n"; delete (DelIt->second); pNHMsgs->erase(DelIt); } std::vector<VDLMacPkt*>::iterator ErsIt = pMacNHMsgs->end(); for(std::vector<VDLMacPkt*>::iterator it = pMacNHMsgs->begin(); it != pMacNHMsgs->end(); it++) { if(((*it)->getSrcAddr() == addr) && ((*it)->getTimestamp() == TS)) ErsIt = it; } if(ErsIt != pMacNHMsgs->end()) { debugEV << "Delete corresponding message part in MAC layer. \n"; delete (*ErsIt); pMacNHMsgs->erase(ErsIt); } delete msg; } else if(addr == (BaseStatnPkt->getBSaddr())) { debugEV << "Message from BS in my radiorange. \n"; bool found = false; //searching for entry of that BS, if it's not - remember for(std::vector<std::pair<LAddress::L2Type, Coord>>::iterator it = BSs.begin(); it != BSs.end(); it++) { if(((*it).first) == BaseStatnPkt->getBSaddr()) { found = true; Coord BS_coord; for(std::vector<std::pair<LAddress::L2Type, LMEPkt*>>::iterator i = pNHMsgs->begin(); i != pNHMsgs->end(); i++) { if((i->first) == addr) { BS_coord = (i->second)->getCoord(); } } it->second = BS_coord; } } if(!found) { std::pair<LAddress::L2Type, Coord> p; p.first = BaseStatnPkt->getBSaddr(); Coord BS_coord; for(std::vector<std::pair<LAddress::L2Type, LMEPkt*>>::iterator i = pNHMsgs->begin(); i != pNHMsgs->end(); i++) { if((i->first) == addr) { BS_coord = (i->second)->getCoord(); } } p.second = BS_coord; BSs.push_back(p); } { bool fnd = false; for(std::vector<cMessage*>::iterator it = BSavailability.begin(); it != BSavailability.end(); it++) { if(strcmp((*it)->getName() , ((BaseStatnPkt->getBSaddr()).str().c_str())) == 0) { fnd = true; if((*it)->isScheduled()) cancelEvent((*it)); scheduleAt(simTime() + BSavailable_time, (*it)); } if (fnd) break; } if(!fnd) { BSavailability.push_back(new cMessage((BaseStatnPkt->getBSaddr()).str().c_str(), GRPfSAN_BSavailability)); scheduleAt(simTime() + BSavailable_time, BSavailability.back()); } } bool MsgFnd = false; for(std::vector<RoutTableEntry>::iterator it = RoutingTable.begin(); it != RoutingTable.end(); it++) { bool found = false; for(std::vector<std::pair<GRPfSAN_R_Pkt*, int>>::iterator iter = (it->BStations).begin(); iter != (it->BStations).end(); iter++) { if(((iter->first)->getBSaddr() ==
```

```
(BaseStatnPkt->getBSaddr())) && (((iter->first)->getSeqNumber()) == (BaseStatnPkt->getSeqNumber()))) { found = true; MsgFnd = true; } if(found) break; } if(MsgFnd) break; } if(!MsgFnd) { int TTL = BaseStatnPkt->getLifeTime(); BaseStatnPkt->setLifeTime(--TTL); if((TTL > 0) && SPEnd) { debugEV << "There is no such BS message in routing table - rebroadcast!\n"; NetwControlInfoL2::setControlInfo(BaseStatnPkt, addr); sendDown(BaseStatnPkt); } else { if (TTL == 0) debugEV << "There is no such BS message in routing table, but TTL = 0. \n"; std::vector<std::pair<LAddress::L2Type, LMEPkt*>>::iterator DelIt = pNHMsgs->end(); for(std::vector<std::pair<LAddress::L2Type, LMEPkt*>>::iterator it = pNHMsgs->begin(); it != pNHMsgs->end(); it++) { if(((it->first) == addr) && (it->second->getTimestamp() == TS)) DelIt = it; } if(DelIt != pNHMsgs->end()) { debugEV << "Delete corresponding message part on LME layer. \n"; delete (DelIt->second); pNHMsgs->erase(DelIt); } std::vector<VDLMacPkt*>::iterator ErsIt = pMacNHMsgs->end(); for(std::vector<VDLMacPkt*>::iterator it = pMacNHMsgs->begin(); it != pMacNHMsgs->end(); it++) { if(((*it)->getSrcAddr() == addr) && ((*it)->getTimestamp() == TS)) ErsIt = it; } if(ErsIt != pMacNHMsgs->end()) { debugEV << "Delete corresponding message part in MAC layer. \n"; delete (*ErsIt); pMacNHMsgs->erase(ErsIt); } delete msg; } } } else { bool BSf = false; std::vector<PECT_entry> neighb = pMac->getPECT_table(); for(std::vector<PECT_entry>::iterator it = neighb.begin(); it != neighb.end(); it++) { if(((*it).pkt->getVehicleType() == BaseStation) && ((*it).reachable)) { debugEV << "There is some BS in my neighbours -> ignore this message. \n"; BSf = true; std::vector<std::pair<LAddress::L2Type, LMEPkt*>>::iterator DelIter = pNHMsgs->end(); for(std::vector<std::pair<LAddress::L2Type, LMEPkt*>>::iterator iter = pNHMsgs->begin(); iter != pNHMsgs->end(); iter++) { if(((iter->first) == addr) && (iter->second->getTimestamp() == TS)) DelIter = iter; } if(DelIter != pNHMsgs->end()) { debugEV << "Delete corresponding message part on LME layer. \n"; delete (DelIter->second); pNHMsgs->erase(DelIter); } std::vector<VDLMacPkt*>::iterator ErsIter = pMacNHMsgs->end(); for(std::vector<VDLMacPkt*>::iterator iter = pMacNHMsgs->begin(); iter != pMacNHMsgs->end(); iter++) { if(((*iter)->getSrcAddr() == addr) && ((*iter)->getTimestamp() == TS)) ErsIter = iter; } if(ErsIter != pMacNHMsgs->end()) { debugEV << "Delete corresponding message part in MAC layer. \n"; delete (*ErsIter); pMacNHMsgs->erase(ErsIter); } delete msg; } if(BSf) break; } if(!BSf) { bool found = false; //searching for entry of that BS, if it's not - remember for(std::vector<std::pair<LAddress::L2Type, Coord>>::iterator it = BSs.begin(); it != BSs.end(); it++) { if(((*it).first) == BaseStatnPkt->getBSaddr()) { debugEV << "Entry about such BS: " << (BaseStatnPkt->getBSaddr()) << " already exists. Refreshing. \n"; found = true; Coord BS_coord; for(std::vector<std::pair<LAddress::L2Type, LMEPkt*>>::iterator i = pNHMsgs->begin(); i != pNHMsgs->end(); i++) { if((i->first) == addr) { BS_coord = (i->second)->getCoord(); } } it->second = BS_coord; } } if(!found) { debugEV << "New entry, BS address: " << (BaseStatnPkt->getBSaddr()) << endl; if(BSs.empty() && SPEnd) { debugEV << "Base station's table is empty -> starting traffic generating process. \n"; MS_TrafficGen = new cMessage("MS_Traffic", GRPfSAN_MSTrafGen); scheduleAt(simTime(), MS_TrafficGen); } std::pair<LAddress::L2Type, Coord> p; p.first = BaseStatnPkt->getBSaddr(); Coord BS_coord; for(std::vector<std::pair<LAddress::L2Type, LMEPkt*>>::iterator i = pNHMsgs->begin(); i != pNHMsgs->end(); i++) { if((i->first) == addr) { BS_coord = (i->second)->getCoord(); } } p.second = BS_coord; BSs.push_back(p); } { bool fnd = false; for(std::vector<cMessage*>::iterator it = BSavailability.begin(); it != BSavailability.end(); it++) {
```

```
if(strcmp((*it)->getName() , ((BaseStatnPkt->getBSaddr()).str().c_str())) == 0) { fnd = true;
if((*it)->isScheduled()) cancelEvent((*it)); scheduleAt(simTime() + BSavailable_time, (*it)); }
if (fnd) break; } if(!fnd) { BSavailability.push_back(new cMessage((BaseStatnPkt-
>getBSaddr()).str().c_str(), GRPfSAN_BSavailability)); scheduleAt(simTime() +
BSavailable_time, BSavailability.back()); } } { bool MsgFnd = false; debugEV <<
RoutingTable.empty() << endl; for(std::vector<RoutTableEntry>::iterator it =
RoutingTable.begin(); it != RoutingTable.end(); it++) { bool found = false; debugEV << (it-
>BStations).empty() << endl; for(std::vector<std::pair<GRPfSAN_R_Pkt*, int>>::iterator iter =
(it->BStations).begin(); iter != (it->BStations).end(); iter++) { debugEV << ((iter->first)-
>getBSaddr()) << " " << ((iter->first)->getSeqNumber()) << endl; if(((iter->first)->getBSaddr() ==
(BaseStatnPkt->getBSaddr())) && (((iter->first)->getSeqNumber()) == (BaseStatnPkt-
>getSeqNumber()))) { found = true; MsgFnd = true; } if(found) break; } if(MsgFnd) break;
} bool Rebroad = false; if(!MsgFnd) { int TTL = BaseStatnPkt->getLifeTime(); debugEV <<
"Set TTL to: " << (TTL-1) << endl; BaseStatnPkt->setLifeTime(--TTL); if((TTL > 0) &&
SPEnd) { Rebroad = true; debugEV << "There is no such BS message in routing table -
rebroadcast!\n"; NetwControlInfoL2::setControlInfo(BaseStatnPkt, addr);
sendDown(BaseStatnPkt); } else if(TTL == 0) { debugEV << "There is no such BS message in
routing table, but TTL = 0. \n"; std::vector<std::pair<LAddress::L2Type, LMEPkt*>>::iterator
DelIt = pNHMsgs->end(); for(std::vector<std::pair<LAddress::L2Type, LMEPkt*>>::iterator it =
pNHMsgs->begin(); it != pNHMsgs->end(); it++) { if(((it->first) == addr) && (it->second-
>getTimestamp() == TS)) DelIt = it; } if(DelIt != pNHMsgs->end()) { debugEV << "Delete
corresponding message part on LME layer. \n"; delete (DelIt->second); pNHMsgs->erase(DelIt);
} std::vector<VDLMacPkt*>::iterator ErsIt = pMacNHMsgs->end();
for(std::vector<VDLMacPkt*>::iterator it = pMacNHMsgs->begin(); it != pMacNHMsgs->end();
it++) { if(((*it)->getSrcAddr() == addr) && ((*it)->getTimestamp() == TS)) ErsIt = it; } if(ErsIt
!= pMacNHMsgs->end()) { debugEV << "Delete corresponding message part in MAC layer. \n";
delete (*ErsIt); pMacNHMsgs->erase(ErsIt); } delete msg; } } //now there is a need to update
routing table or create new entry in it bool fnd = false; for(std::vector<RoutTableEntry>::iterator
it = RoutingTable.begin(); it != RoutingTable.end(); it++)//searching for entry in RT { if(it-
>RouterAddr == addr)//if there is already an entry with such address { fnd = true;
//std::vector<std::pair<GRPfSAN_R_Pkt*, int>> NodeBSvec = it->BStations; //getting node's BS
list bool fd = false; std::vector<std::pair<GRPfSAN_R_Pkt*,int>>::iterator EraseIt = (it-
>BStations).end(); for(std::vector<std::pair<GRPfSAN_R_Pkt*,int>>::iterator iter = (it-
>BStations).begin(); iter != (it->BStations).end(); iter++) { if(((iter->first)->getBSaddr()) ==
(BaseStatnPkt->getBSaddr())) //if there is an entry with such BS { fd = true; if(((iter->first)-
>getSeqNumber()) == (BaseStatnPkt->getSeqNumber())) { debugEV << "I've already received
base station's pkt with such seq num from that node: " << addr << endl;
std::vector<std::pair<LAddress::L2Type, LMEPkt*>>::iterator DelIt = pNHMsgs->end();
for(std::vector<std::pair<LAddress::L2Type, LMEPkt*>>::iterator hmit = pNHMsgs->begin();
hmit != pNHMsgs->end(); hmit++) { if(((hmit->first) == addr) && (hmit->second-
>getTimestamp() == TS)) DelIt = hmit; } if(DelIt != pNHMsgs->end()) { debugEV << "Delete
corresponding message part on LME layer. \n"; delete (DelIt->second); pNHMsgs->erase(DelIt);
} std::vector<VDLMacPkt*>::iterator ErsIt = pMacNHMsgs->end();
for(std::vector<VDLMacPkt*>::iterator hmit = pMacNHMsgs->begin(); hmit != pMacNHMsgs-
>end(); hmit++) { if(((*hmit)->getSrcAddr() == addr) && ((*hmit)->getTimestamp() == TS))
```

```
ErsIt = hmit; } if(ErsIt != pMacNHMsgs->end()) { debugEV << "Delete corresponding message part in MAC layer. \n"; delete (*ErsIt); pMacNHMsgs->erase(ErsIt); } delete msg; } else { debugEV << "I've received BS pkt from node: " << addr << ", with info of BS: " << BaseStatnPkt->getBSaddr() << ", with TTL: " << BaseStatnPkt->getLifeTime() << endl; bool NotSingle = false;//searching for other entry in table for(std::vector<RoutTableEntry>::iterator iterat = RoutingTable.begin(); iterat != RoutingTable.end(); iterat++) { if(iterat->RouterAddr != addr) { std::vector<std::pair<GRPfSAN_R_Pkt*, int>> NodeBSvec = it->BStations; for(std::vector<std::pair<GRPfSAN_R_Pkt*, int>>::iterator BSit = NodeBSvec.begin(); BSit != NodeBSvec.end(); BSit++) { if((BSit->first)->getBSaddr() == BaseStatnPkt->getBSaddr()) NotSingle = true; } } if(NotSingle) break; } if(NotSingle) //if there is already an entry with such BS address { Coord BS_coord; for(std::vector<std::pair<LAddress::L2Type, Coord>>::iterator itr = BSs.begin(); itr != BSs.end(); itr++) { if((itr->first) == BaseStatnPkt->getBSaddr()) BS_coord = itr->second; } Coord node_coord = pMac->getAddrCoord(addr); ChannelMobilityPtrType ptrMobility = ChannelMobilityAccessType().get(); Coord MyPos = ptrMobility->getCurrentPosition(); if(FindDistance(node_coord, BS_coord) <= FindDistance(MyPos, BS_coord)) { delete (iter->first); iter->first = BaseStatnPkt->dup(); iter->second = BaseStatnPkt->getLifeTime(); for(std::vector<cMessage*>::iterator MsgIt = RTtimers.begin(); MsgIt != RTtimers.end(); MsgIt++) { if(strcmp((*MsgIt)->getName() , (addr.str().c_str())) == 0) { cObject* pCntrlInfo = (*MsgIt)->getControlInfo(); if((NetwControlInfoL2::getAddressFromControlInfo(pCntrlInfo)) == (BaseStatnPkt->getBSaddr())) { if((*MsgIt)->isScheduled()) cancelEvent((*MsgIt)); scheduleAt(simTime() + RTentry_time, (*MsgIt)); } } } } else { EraseIt = iter; bool fm = false; std::vector<cMessage*>::iterator ErsIt = RTtimers.end(); for(std::vector<cMessage*>::iterator MsgIt = RTtimers.begin(); MsgIt != RTtimers.end(); MsgIt++) { if(strcmp((*MsgIt)->getName() , (addr.str().c_str())) == 0) { cObject* pCntrlInfo = (*MsgIt)->getControlInfo(); if((NetwControlInfoL2::getAddressFromControlInfo(pCntrlInfo)) == (BaseStatnPkt->getBSaddr())) { fm = true; ErsIt = MsgIt; } } if(fm) break; } if(ErsIt != RTtimers.end()) { delete (*ErsIt); RTtimers.erase(ErsIt); } } } else { delete (iter->first); iter->first = BaseStatnPkt->dup(); iter->second = BaseStatnPkt->getLifeTime(); for(std::vector<cMessage*>::iterator MsgIt = RTtimers.begin(); MsgIt != RTtimers.end(); MsgIt++) { if(strcmp((*MsgIt)->getName() , (addr.str().c_str())) == 0) { cObject* pCntrlInfo = (*MsgIt)->getControlInfo(); if((NetwControlInfoL2::getAddressFromControlInfo(pCntrlInfo)) == (BaseStatnPkt->getBSaddr())) { if((*MsgIt)->isScheduled()) cancelEvent((*MsgIt)); scheduleAt(simTime() + RTentry_time, (*MsgIt)); } } } } } } if(fd) break; } if(EraseIt != (it->BStations).end()) { delete (EraseIt->first); it->BStations.erase(EraseIt); } } if (fnd) break; } if(!fnd) //if there is no entry with such address {// i'm must decide either to add such entry or not debugEV << "I've received BS pkt from node: " << addr << ", with info of BS: " << BaseStatnPkt->getBSaddr() << ", with TTL: " << BaseStatnPkt->getLifeTime() << endl; bool f = false; for(std::vector<RoutTableEntry>::iterator it = RoutingTable.begin(); it != RoutingTable.end(); it++) { std::vector<std::pair<GRPfSAN_R_Pkt*, int>> NodeBSvec = it->BStations; for(std::vector<std::pair<GRPfSAN_R_Pkt*,int>>::iterator iter = NodeBSvec.begin(); iter != NodeBSvec.end(); iter++) { if(((iter->first)->getBSaddr()) == (BaseStatnPkt->getBSaddr())) { debugEV << "There is some entry about such BS in routing table...\n"; f = true; Coord BS_coord; for(std::vector<std::pair<LAddress::L2Type, Coord>>::iterator iterat = BSs.begin(); iterat != BSs.end(); iterat++) { if((iterat->first) == BaseStatnPkt->getBSaddr()) BS_coord = iterat->second;
```

```
} Coord node_coord = pMac->getAddrCoord(addr); ChannelMobilityPtrType ptrMobility =
ChannelMobilityAccessType().get(); Coord MyPos = ptrMobility->getCurrentPosition();
debugEV << "Distance from node to BS: " << FindDistance(node_coord, BS_coord) << endl;
debugEV << "Distance from me to BS: " << FindDistance(MyPos, BS_coord) << endl;
if(FindDistance(node_coord, BS_coord) <= FindDistance(MyPos, BS_coord)) { debugEV <<
"The node I received info from is closer to BS, than me, adding to table. \n"; debugEV << "Adding
an entry into routing table!\n"; RoutTableEntry entry; entry.RouterAddr = addr;
std::pair<GRPfSAN_R_Pkt*, int> p; p.first = BaseStatnPkt->dup(); p.second = BaseStatnPkt-
>getLifeTime(); entry.BStations.push_back(p); RoutingTable.push_back(entry);
RTtimers.push_back(new cMessage(addr.str().c_str(), GRPfSAN_RTtimer));
NetwControlInfoL2::setControlInfo(RTtimers.back(), BaseStatnPkt->getBSaddr());
scheduleAt(simTime() + RTentry_time, RTtimers.back()); } else { debugEV << "The node I
received info from is farther to BS, than me -> discarding. \n"; if(!Rebroad) {
std::vector<std::pair<LAddress::L2Type, LMEPkt*>>::iterator DelIter = pNHMsgs->end();
for(std::vector<std::pair<LAddress::L2Type, LMEPkt*>>::iterator LMEiter = pNHMsgs->begin();
LMEiter != pNHMsgs->end(); LMEiter++) { if(((LMEiter->first) == addr) && (LMEiter-
>second->getTimestamp() == TS)) DelIter = LMEiter; } if(DelIter != pNHMsgs->end()) {
debugEV << "Delete corresponding message part on LME layer. \n"; delete (DelIter->second);
pNHMsgs->erase(DelIter); } std::vector<VDLMacPkt*>::iterator ErsIter = pMacNHMsgs->end();
for(std::vector<VDLMacPkt*>::iterator MACiter = pMacNHMsgs->begin(); MACiter !=
pMacNHMsgs->end(); MACiter++) { if(((*MACiter)->getSrcAddr() == addr) && ((*MACiter)-
>getTimestamp() == TS)) ErsIter = MACiter; } if(ErsIter != pMacNHMsgs->end()) { debugEV
<< "Delete corresponding message part in MAC layer. \n"; delete (*ErsIter); pMacNHMsgs-
>erase(ErsIter); } delete msg; } } } } if (f) break; } if(!f) { debugEV << "There is no any
route entry for that BS, adding an entry into routing table!\n"; RoutTableEntry entry;
entry.RouterAddr = addr; std::pair<GRPfSAN_R_Pkt*, int> p; p.first = BaseStatnPkt->dup();
p.second = BaseStatnPkt->getLifeTime(); entry.BStations.push_back(p);
RoutingTable.push_back(entry); RTtimers.push_back(new cMessage(addr.str().c_str(),
GRPfSAN_RTtimer)); NetwControlInfoL2::setControlInfo(RTtimers.back(), BaseStatnPkt-
>getBSaddr()); scheduleAt(simTime() + RTentry_time, RTtimers.back()); } } } } } } else
if(MsgType == MSADSBData) { MacPkt* MSpkt = static_cast<MacPkt*>(msg);
LAddress::L2Type source; for(std::vector<VDLMacPkt*>::iterator it = pMacNHMsgs->begin(); it
!= pMacNHMsgs->end(); it++) { if((((*it)->getSrcAddr()) == addr) && (((*it)->getTimestamp())
== (TS))) source = (*it)->getSrcAddr(); } if(IsBaseStation) { if((MSpkt->getSrcAddr() ==
MSpkt->getDestAddr()) && (MSpkt->getDestAddr() == pMac->getMACAddress())) { debugEV
<< "I received a message from: " << source << " ,addressed to me! \n"; debugEV << "Propagation
time: " << (simTime() - MSpkt->getCreationTime()) << endl; double PropagTime = (simTime() -
MSpkt->getCreationTime()).dbl(); emit(Rcvd, 1); emit(PropagationTime, PropagTime);
std::vector<std::pair<LAddress::L2Type, LMEPkt*>>::iterator DelIter = pNHMsgs->end();
for(std::vector<std::pair<LAddress::L2Type, LMEPkt*>>::iterator iter = pNHMsgs->begin(); iter
!= pNHMsgs->end(); iter++) { if(((iter->first) == addr) && (iter->second->getTimestamp() ==
TS)) DelIter = iter; } if(DelIter != pNHMsgs->end()) { debugEV << "Delete corresponding
message part on LME layer. \n"; delete (DelIter->second); pNHMsgs->erase(DelIter); }
std::vector<VDLMacPkt*>::iterator ErsIter = pMacNHMsgs->end();
for(std::vector<VDLMacPkt*>::iterator iter = pMacNHMsgs->begin(); iter != pMacNHMsgs-
```

```
>end(); iter++) { if(((*iter)->getSrcAddr() == addr) && ((*iter)->getTimestamp() == TS)) ErsIter
= iter; } if(ErsIter != pMacNHMsgs->end()) { debugEV << "Delete corresponding message part
in MAC layer. \n"; delete (*ErsIter); pMacNHMsgs->erase(ErsIter); } delete msg; } else {
debugEV << "Received some MS message, and I'm a BS -> ignore. \n";
std::vector<std::pair<LAddress::L2Type, LMEPkt*>>::iterator DelIter = pNHMsgs->end();
for(std::vector<std::pair<LAddress::L2Type, LMEPkt*>>::iterator iter = pNHMsgs->begin(); iter
!= pNHMsgs->end(); iter++) { if(((iter->first) == addr) && (iter->second->getTimestamp() ==
TS)) DelIter = iter; } if(DelIter != pNHMsgs->end()) { debugEV << "Delete corresponding
message part on LME layer. \n"; delete (DelIter->second); pNHMsgs->erase(DelIter); }
std::vector<VDLMacPkt*>::iterator ErsIter = pMacNHMsgs->end();
for(std::vector<VDLMacPkt*>::iterator iter = pMacNHMsgs->begin(); iter != pMacNHMsgs-
>end(); iter++) { if(((*iter)->getSrcAddr() == addr) && ((*iter)->getTimestamp() == TS)) ErsIter
= iter; } if(ErsIter != pMacNHMsgs->end()) { debugEV << "Delete corresponding message part
in MAC layer. \n"; delete (*ErsIter); pMacNHMsgs->erase(ErsIter); } delete msg; } } else
if(MSpkt->getSrcAddr() == pMac->getMACAddress()) { debugEV << "I was selected as a router
for message from node: " << source << " ,sending it to BS: " << MSpkt->getDestAddr() << endl;
std::vector<PECT_entry> neighb = pMac->getPECT_table(); bool BSisNeighb = false;
for(std::vector<PECT_entry>::iterator it = neighb.begin(); it != neighb.end(); it++) //searching for
BS in my neighbours { if((it->pkt->getSrcAddr()) == (MSpkt->getDestAddr()) && (it-
>reachable)) BSisNeighb = true; if(BSisNeighb) break; } if(BSisNeighb) { debugEV << "BS
for which message is routing is my neighbour -> send. \n"; LAddress::L2Type dest = MSpkt-
>getDestAddr(); MSpkt->setSrcAddr(dest); NetwControlInfoL2::setControlInfo(MSpkt, addr);
sendDown(MSpkt); } else { debugEV << "Searching for router for BS: " << MSpkt-
>getDestAddr() << endl; LAddress::L2Type router = NO_RECEIVER;
std::vector<LAddress::L2Type> Routers; for(std::vector<RoutTableEntry>::iterator it =
RoutingTable.begin(); it != RoutingTable.end(); it++) {
for(std::vector<std::pair<GRPfSAN_R_Pkt*, int>>::iterator iter = (it->BStations).begin(); iter !=
(it->BStations).end(); iter++) { if((iter->first)->getBSaddr() == MSpkt->getDestAddr())
Routers.push_back(it->RouterAddr); } } for(std::vector<LAddress::L2Type>::iterator it =
Routers.begin(); it != Routers.end(); it++) { if(router == NO_RECEIVER) router = *it; else {
Coord router_coord = pMac->getAddrCoord(router); Coord it_coord = pMac->getAddrCoord(*it);
Coord bs_coord; for(std::vector<std::pair<LAddress::L2Type, Coord>>::iterator BSit =
BSs.begin(); BSit != BSs.end(); BSit++) { if((BSit->first) == (MSpkt->getDestAddr())) bs_coord
= BSit->second; } if(FindDistance(it_coord, bs_coord) < FindDistance(router_coord, bs_coord))
{ router = *it; } } } debugEV << "Chosen router addr is: " << router << endl; MSpkt-
>setSrcAddr(router); NetwControlInfoL2::setControlInfo(MSpkt, addr); sendDown(MSpkt); } }
else { debugEV << "MS ADS-B data received, but another node was selected as a router. \n";
std::vector<std::pair<LAddress::L2Type, LMEPkt*>>::iterator DelIter = pNHMsgs->end();
for(std::vector<std::pair<LAddress::L2Type, LMEPkt*>>::iterator iter = pNHMsgs->begin(); iter
!= pNHMsgs->end(); iter++) { if(((iter->first) == addr) && (iter->second->getTimestamp() ==
TS)) DelIter = iter; } if(DelIter != pNHMsgs->end()) { debugEV << "Delete corresponding
message part on LME layer. \n"; delete (DelIter->second); pNHMsgs->erase(DelIter); }
std::vector<VDLMacPkt*>::iterator ErsIter = pMacNHMsgs->end();
for(std::vector<VDLMacPkt*>::iterator iter = pMacNHMsgs->begin(); iter != pMacNHMsgs-
>end(); iter++) { if(((*iter)->getSrcAddr() == addr) && ((*iter)->getTimestamp() == TS)) ErsIter
```

```
= iter; } if(ErsIter != pMacNHMsgs->end()) { debugEV << "Delete corresponding message part
in MAC layer. \n"; delete (*ErsIter); pMacNHMsgs->erase(ErsIter); } delete msg; } } }
```

ПРИЛОЖЕНИЕ Б. АЛГЕБРА «ЖАДНОЙ» МАРШРУТИЗАЦИИ

Рассмотрим алгебраическую систему, основанную на связном направленном графе, которая имеет четыре кортежа $\langle L, F,\ w, \preceq \rangle$, где:

- L: множество сигнатур;
- F: множество потоков трафика;
- w: весовая функция связи;
- $\preceq$: порядок отношения.

Сигнатура $l_{\overrightarrow{uv}}$ множества L описывает характеристики связи $\overrightarrow{uv}$, такие как: местоположение узлов *u* и *v,* потери сообщений, энергопотребление на бит передачи, частоту канала и т.п. Каждая связь в рассматриваемой модели помечена сигнатурой. Иными словами, множество L описывает все рассматриваемые характеристики связи в сети.

Поток $f_{\langle s,d \rangle}$ множества F представляет собой информационный трафик, который должен быть доставлен из *s* в *d*. Мощность множества $F\ \leq |V| \times (|V-1|)$.

$w(.)$ – это функция, рассчитывающая вес связи для различных потоков. $w(.)$ является функцией двух переменных, соответственно одна из них является сигнатурой, а другая потоком. Например, для потока $f_{\langle s,d \rangle}$, вес исходящего соединения $\overrightarrow{uv}$ узла *u* это $w(l_{\overrightarrow{uv}}, f_{\langle s,d \rangle})$, где $l_{\overrightarrow{uv}} \in L$, а $f_{\langle s,d \rangle} \in F$.

$\preceq$ используется для сравнения всех исходящих связей для передающего узла, основываясь на функции $w(.)$, таким образом, чтобы он мог передавать сообщения по связи с наименьшим весом. Если $w\left(l_{\overrightarrow{uv}}, f_{\langle s,d \rangle}\right) \preceq w\left(l_{\overrightarrow{ux}}, f_{\langle s,d \rangle}\right)$, то связь $\overrightarrow{uv}$ не хуже (легче или равна) связи $\overrightarrow{ux}$ для потока $f_{\langle s,d \rangle}$.

Стоит отметить, что для «жадного» алгоритма не все исходящие связи могут быть рассмотрены в качестве кандидатов для передачи сообщения – могут быть рассмотрены лишь те связи, чей вес строго меньше, чем ϕ, где ϕ порог для принятия решения «жадным» алгоритмом.

Используя рассмотренную алгебраическую систему, можно исследовать применимость «жадной» маршрутизации, основываясь на следующих свойствах:

Мьютекс: $\forall\ l_{\overrightarrow{uv}}, l_{\overrightarrow{vu}} \in L, f_{\langle s,d \rangle} \in F,\ w\left(l_{\overrightarrow{uv}}, f_{\langle s,d \rangle}\right) \prec \phi$ следует $w\left(l_{\overrightarrow{vu}}, f_{\langle s,d \rangle}\right) \succeq \phi$

Переход:

$\forall\ l_{\overrightarrow{u_1u_2}}, l_{\overrightarrow{u_2u_3}}, \dots, l_{\overrightarrow{u_{k-1}u_k}}, l_{\overrightarrow{u_1u_k}} \in L, f_{\langle s,d \rangle} \in F, w\left(l_{\overrightarrow{u_1u_2}}, f_{\langle s,d \rangle}\right) \prec \phi, w\left(l_{\overrightarrow{u_2u_3}}, f_{\langle s,d \rangle}\right) \prec \phi, \dots,$ и $w\left(l_{\overrightarrow{u_{k-1}u_k}}, f_{\langle s,d \rangle}\right) \prec \phi$ следует $w\left(l_{\overrightarrow{u_1u_k}}, f_{\langle s,d \rangle}\right) \prec \phi$

Независимость источников:

$\forall\ f_{\langle s_1,d \rangle},\ f_{\langle s_2,d \rangle} \in F, l_{\overrightarrow{uv}} \in L,$ всегда верно $w\left(l_{\overrightarrow{uv}},\ f_{\langle s_1,d \rangle}\right) =\ w\left(l_{\overrightarrow{uv}},\ f_{\langle s_2,d \rangle}\right)$

Строгое предпочтение: $\forall\ f_{\langle s,d\rangle} \in F$, не существует случаев, когда для двух связей $\overrightarrow{uv}$ и $\overrightarrow{uw}$ из узла *u* выполняется $w\left(l_{\overrightarrow{uv}},\ f_{\langle s,d\rangle}\right) = w\left(l_{\overrightarrow{uw}},\ f_{\langle s,d\rangle}\right)$.

Теорема 1: В качестве метода маршрутизации сообщений, жадный алгоритм *R* не имеет циклов тогда и только тогда, когда метрики маршрутизации подразумевают выполнения свойств «мьютекса» и перехода.

Доказательство:

Докажем достаточное условие от противного. Т.к. поток информации $f_{\langle s,d\rangle}$ инициированный узлом *s* маршрутизируется узлу назначения *d* алгоритмом *R* в беспроводной сети G(V,E), то петля возникает в процессе ретрансляции сообщения.

В зависимости от количества узлов в петле, можно рассмотреть два типа: петля, состоящая из двух узлов и петля, состоящая из трёх узлов и более.

Если петля из двух узлов $c(m,n)$ создана алгоритмом маршрутизации *R* в процессе передачи потока $f_{\langle s,d\rangle}$,то все сообщения этого потока передаются туда и обратно между узлом *m* и *n*, например, до тех пор, пока не будет отброшен по достижении 0 значения поля времени жизни. Очевидно, что существуют две связи $l_{\overrightarrow{mn}}$ и $l_{\overrightarrow{nm}} \in L$ между узлом *m* и *n* в этом случае. Основываясь на принципе «жадной» маршрутизации, сообщение из потока $f_{\langle s,d\rangle}$ может быть передано узлом *u* узлу *v* тогда и только тогда, когда условие $w\left(l_{\overrightarrow{uv}}, f_{\langle s,d\rangle}\right) \prec \phi$ удовлетворяется. Поэтому, если петля $c(m,n)$ возникает, то также должны выполняться оба условия $w\left(l_{\overrightarrow{mn}}, f_{\langle s,d\rangle}\right) \prec \phi$ и $w\left(l_{\overrightarrow{nm}}, f_{\langle s,d\rangle}\right) \prec \phi$. Соответственно возникает противоречие с тем, что метрика маршрутизации имеет свойство «мьютекса», таким образом, петля между двумя узлами не может возникнуть в процессе передачи $f_{\langle s,d\rangle}$.

Предположим, что «жадный» алгоритм в процессе маршрутизации потока $f_{\langle s,d\rangle}$ создаёт петлю, состоящую более чем из двух узлов: $c(u_1,u_2,\dots,u_k)$, где $k \geq 3$. Все сообщения из потока $f_{\langle s,d\rangle}$ будут доставлены обратно узлу u_1 через связи $\overrightarrow{u_1u_2}, \overrightarrow{u_2u_3}, \dots, \overrightarrow{u_{k-1}u_k}, \overrightarrow{u_ku_1}$. Более того, это также справедливо для связи $l_{\overrightarrow{u_1u_k}} \in L$ из-за симметричности связей. Следуя принципу «жадной» маршрутизации, для потока $f_{\langle s,d\rangle}$ вес каждой связи в цикле $c(u_1,u_2,\dots,u_k)$, где $k \geq 3$ строго меньше ϕ, т.е. $w\left(l_{\overrightarrow{u_1u_2}}, f_{\langle s,d\rangle}\right) \prec \phi, w\left(l_{\overrightarrow{u_2u_3}}, f_{\langle s,d\rangle}\right) \prec \phi, \dots, w\left(l_{\overrightarrow{u_{k-1}u_k}}, f_{\langle s,d\rangle}\right) \prec \phi$ и $w\left(l_{\overrightarrow{u_ku_1}}, f_{\langle s,d\rangle}\right) \prec \phi$. Исходя из свойства перехода, можно сделать вывод, что условие $w\left(l_{\overrightarrow{u_1u_k}}, f_{\langle s,d\rangle}\right) \prec \phi$ удовлетворяется. Итак, оба условия $w\left(l_{\overrightarrow{u_1u_k}}, f_{\langle s,d\rangle}\right) \prec \phi$ и $w\left(l_{\overrightarrow{u_ku_1}}, f_{\langle s,d\rangle}\right) \prec \phi$ выполняются, что противоречит факту наличия «мьютекса» в метрике маршрутизации. Следовательно, петли, состоящие более чем из двух узлов, не могут быть созданы «жадной» маршрутизацией.

Необходимые условия доказываются исходя из простых примеров. Если метрика маршрутизации не имеет «мьютекса», то при использовании «жадного» алгоритма может возникнуть петля между двумя узлами-маршрутизаторами. Рассмотрим сеть на рис. 1, где для потока $f_{\langle s,d\rangle}$, $w\left(l_{\overrightarrow{su}}, f_{\langle s,d\rangle}\right) \prec w\left(l_{\overrightarrow{sv}}, f_{\langle s,d\rangle}\right) \prec \phi$, $w\left(l_{\overrightarrow{uv}}, f_{\langle s,d\rangle}\right) \prec w\left(l_{\overrightarrow{uw}}, f_{\langle s,d\rangle}\right) \prec \phi \prec w\left(l_{\overrightarrow{us}}, f_{\langle s,d\rangle}\right)$ и $w\left(l_{\overrightarrow{vu}}, f_{\langle s,d\rangle}\right) \prec w\left(l_{\overrightarrow{vw}}, f_{\langle s,d\rangle}\right) \prec w\left(l_{\overrightarrow{vs}}, f_{\langle s,d\rangle}\right)$. Из-за отсутствия свойства «мьютекса», также выполняется $w\left(l_{\overrightarrow{vu}}, f_{\langle s,d\rangle}\right) \prec \phi$. Согласно «жадному» алгоритму, все сообщения потока $f_{\langle s,d\rangle}$ будут передаваться узлу *u* узлом *s*, а затем направлены узлу *v*. Т.к. связь $\overrightarrow{vu}$ локально оптимальная и $w\left(l_{\overrightarrow{vu}}, f_{\langle s,d\rangle}\right) \prec \phi$, то все сообщения потока $f_{\langle s,d\rangle}$, приходящие от узла *u* будут снова переданы обратно узлу *u*, формируя тем самым петлю маршрутизации между узлами *u* и *v*.

Если же метрика маршрутизации имеет свойство «мьютекса», но не имеет свойства перехода, то могут возникать петли, содержащие множество узлов. Снова обратимся к рис. 1 и предположим, что связи удовлетворяют следующим неравенствам: $w\left(l_{\overrightarrow{su}}, f_{\langle s,d\rangle}\right) \prec \phi \prec w\left(l_{\overrightarrow{sv}}, f_{\langle s,d\rangle}\right)$, $w\left(l_{\overrightarrow{uv}}, f_{\langle s,d\rangle}\right) \prec w\left(l_{\overrightarrow{uw}}, f_{\langle s,d\rangle}\right) \prec \phi \prec w\left(l_{\overrightarrow{us}}, f_{\langle s,d\rangle}\right)$ и $w\left(l_{\overrightarrow{vs}}, f_{\langle s,d\rangle}\right) \prec w\left(l_{\overrightarrow{vw}}, f_{\langle s,d\rangle}\right) \prec \phi \prec w\left(l_{\overrightarrow{vu}}, f_{\langle s,d\rangle}\right)$. Очевидно, что свойство «мьютекса» удовлетворяется, но отсутствует соответствие условию перехода, поэтому для потока $f_{\langle s,d\rangle}$, «жадный» алгоритм создает петлю маршрутизации, в которую входят узлы *s, u* и *v*.

Согласно свойствам реактивного метода маршрутизации и Теореме 1 имеем:

Теорема 2: Для реактивного метода маршрутизации, используемый «жадный» алгоритм *R* не имеет петель и обеспечивает согласованную маршрутизацию тогда и только тогда, когда метрика маршрутизации имеет два свойства: «мьютекс» и переход.

Теорема 3: Протокол маршрутизации *R*, основанный на «жадном» методе построения таблиц маршрутизации по ответному сообщению на запрос поиска маршрута, не имеет петель тогда и только тогда, когда метрики маршрутизации имеют три свойства: «мьютекс», переход и независимость источников.

Докажем необходимые условия от обратного. Пусть «жадный» алгоритм маршрутизации создаёт петлю $c(u_1, u_2, \ldots, u_k)$ в процессе передачи потока $f_{\langle s_x,d\rangle}$, через узлы, построившие свои таблицы маршрутизации также по «жадному» алгоритму. Все сообщения потока $f_{\langle s_x,d\rangle}$, которые были ретранслированы u_1, придут на него обратно через связи $\overrightarrow{u_1u_2}, \overrightarrow{u_2u_3}, \ldots, \overrightarrow{u_{k-1}u_k}, \overrightarrow{u_ku_1}$. Также это справедливо и для связи $\overrightarrow{u_1u_k} \in E$ из-за свойства симметричности связей в рассматриваемой сети. Основываясь на принципе «жадного» алгоритма для каждой связи в петле должен существовать такой поток $f_{\langle s_y,d\rangle}$, для которого выполняется соотношение $w(l_{\overrightarrow{u_iu_{i+1}}}, f_{\langle s_y,d\rangle}) \prec \phi$, где $1 \leq i < k\text{-}1$. Согласно свойству

независимости источников всегда выполняется условие: $w\left(l_{\overrightarrow{u_i u_{i+1}}}, f_{\langle *,d\rangle}\right) = w(l_{\overrightarrow{u_i u_{i+1}}}, f_{\langle s_y,d\rangle})$, где $f_{\langle *,d\rangle}$ означает любой информационный трафик направленный узлу d в F. Поэтому неравенства $w\left(l_{\overrightarrow{u_1 u_2}}, f_{\langle *,d\rangle}\right) \prec \phi$, $w\left(l_{\overrightarrow{u_2 u_3}}, f_{\langle *,d\rangle}\right) \prec \phi$, …, $w\left(l_{\overrightarrow{u_{k-1} u_k}}, f_{\langle *,d\rangle}\right) \prec \phi$ и $w\left(l_{\overrightarrow{u_k u_1}}, f_{\langle *,d\rangle}\right) \prec \phi$ также выполняются. Согласно условию перехода, можно прийти к выводу, что $w\left(l_{\overrightarrow{u_1 u_k}}, f_{\langle *,d\rangle}\right) \prec \phi$ также справедливо. В результате и $w\left(l_{\overrightarrow{u_1 u_k}}, f_{\langle *,d\rangle}\right) \prec \phi$ и $w\left(l_{\overrightarrow{u_k u_1}}, f_{\langle *,d\rangle}\right) \prec \phi$ выполняются, что противоречит факту наличия свойства «мьютекса» в метрике маршрутизации, а, следовательно, отсутствие петель гарантируется.

Докажем необходимые условия. Очевидно, если метрика не несет в себе свойства «мьютекса» или перехода, то в процессе передачи сообщения может возникнуть петля, что показано в доказательстве Теоремы 1. Поэтому нужно лишь доказать необходимость свойства независимости источников. Представим, что из-за отсутствия свойства независимости источников для потока $f_{\langle s_1,d\rangle}$ связи удовлетворяют следующим неравенствам: $w\left(l_{\overrightarrow{s_1 u}}, f_{\langle s_1,d\rangle}\right) \prec \phi$, $w\left(l_{\overrightarrow{u v_1}}, f_{\langle s_1,d\rangle}\right) \prec w\left(l_{\overrightarrow{u v_2}}, f_{\langle s_1,d\rangle}\right) \prec \phi \prec w\left(l_{\overrightarrow{u s_1}}, f_{\langle s_1,d\rangle}\right), w\left(l_{\overrightarrow{v_1 v_2}}, f_{\langle s_1,d\rangle}\right) \prec w\left(l_{\overrightarrow{v_1 w}}, f_{\langle s_1,d\rangle}\right) \prec \phi \prec w\left(l_{\overrightarrow{v_1 u}}, f_{\langle s_1,d\rangle}\right)$, $w\left(l_{\overrightarrow{v_2 w}}, f_{\langle s_1,d\rangle}\right) \prec \phi \prec w\left(l_{\overrightarrow{v_2 v_1}}, f_{\langle s_1,d\rangle}\right) \prec w\left(l_{\overrightarrow{v_2 u}}, f_{\langle s_1,d\rangle}\right)$, $w\left(l_{\overrightarrow{w d}}, f_{\langle s_1,d\rangle}\right) \prec \phi \prec w\left(l_{\overrightarrow{w v_1}}, f_{\langle s_1,d\rangle}\right) \prec w\left(l_{\overrightarrow{w v_2}}, f_{\langle s_1,d\rangle}\right)$. Согласно этому, «жадный» алгоритм находит для потока $f_{\langle s_1,d\rangle}$ путь $s_1uv_1v_2wd$, в то время, как путь $s_2uv_2v_1wd$ определяется тем же «жадным» алгоритмом для потока $f_{\langle s_2,d\rangle}$ из-за отсутствия свойства независимости источников. В этом случае наличие «мьютекса» и перехода не меняет ситуации.

Теперь представим, что потоки $f_{\langle s_1,d\rangle}$ и $f_{\langle s_2,d\rangle}$ возникают одновременно. Примем, что сообщения MP_{s1d} и MP_{s2d} – ответ на запрос поиска маршрута для этих потоков соответственно. Основываясь на данных в сообщении MP_{s2d}, узел v_1 заполняет поля в своей таблице маршрутизации, при этом поле будет иметь вид [адресат = d, следующий узел = w]. Далее сообщение отправляется на узел v_2. В это же время, схожий процесс возникает на узле v_2, согласно сообщению MP_{s1d}, и в результате поля его таблицы маршрутизации будут иметь вид [адресат = d, следующий узел = w], а сообщение будет отправлено на узел v_1. После приёма сообщения MP_{s2d}, таблица маршрутизации на узле v_2 будет обновлена в вид [адресат = d, следующий узел = v_1], что также произойдёт и на узле v_1, чья запись в таблице маршрутизации примет вид [адресат = d, следующий узел = v_2]. По завершении фазы поиска маршрута для потоков $f_{\langle s_1,d\rangle}$ и $f_{\langle s_2,d\rangle}$ возникнет петля между узлами v_1 и v_2. Учитывая, что в сетях возникает множество потоков сообщений, возникновение такой ситуации возможно.

Требования в случае построения маршрута в «прямом» направлении отличаются.

Теорема 4: Протокол маршрутизации *R,* использующий метод построения маршрутов при ретрансляции сообщения с запросом на поиск маршрута, не создаёт петель тогда и

только тогда, когда метрика маршрутизации имеет свойство «мьютекса» и свойство перехода.

В данном случае нужно лишь доказать достаточные условия, т.к. необходимые условия могут быть доказаны схожим образом, что и в Теореме 1.

Докажем достаточные условия от противного. Пусть «жадный» алгоритм маршрутизации создаёт петлю $c(u_1, u_2, \dots, u_k)$ в процессе передачи потока, основывающемся на таблицах маршрутизации, построенных согласно продвижению сообщения с запросом поиска маршрута. $f_{\langle s_x,d\rangle}$

Дабы не терять общий смысл, положим, что таблица маршрутизации на узле u_i построена благодаря сообщению поиска маршрута MPs_id для потока $f_{\langle s_i,d\rangle}$, и время построения равно $t(u_i, MPs_id)$, где $1 \le i \le k$. Если $s_1 = s_2 = \dots = s_k$, т.е. все таблицы маршрутизации на узлах-ретрансляторах построены одним потоком, то петля $c(u_1, u_2, \dots, u_k)$ не может быть создана согласно Теореме 1. Далее рассмотрим случай, в котором один поток имеет отличный от других источник.

Для простоты примем $k + 1 = 1$. Учитывая описанные особенности метода нахождения пути должно выполняться следующее условие: $t(u_i, MP_{s_id}) < t(u_{i+1}, MP_{s_{i+1}d})$, где $1 \le i \le k$, т.е. связь $\overrightarrow{u_iu_{i+1}}$ не может быть компонентой пути от узла u_x до d. Поэтому оба неравенства $t(u_1, MP_{s_1d}) < t(u_k, MP_{s_kd})$ и $t(u_k, MP_{s_kd}) < t(u_1, MP_{s_1d})$ удовлетворяются, что является противоречием. Тем самым обеспечивается отсутствие петель.

В то время как для двух рассмотренных методов создания таблиц маршрутизации, использующих для этого продвижение сообщения с запросом о поиске пути, отличаются требования для обеспечения отсутствия петель - для обеспечения согласованности требования одинаковы.

Достаточные условия. По свойству строгого предпочтения, существует лишь одна связь, являющаяся самой «лёгкой» среди других исходящих из узла связей для определенного потока трафика. Поэтому, также учитывая свойства «мьютекса» и перехода, существует только один путь без петель, который может проложить «жадный» алгоритм. Кроме того, записи в таблицах маршрутизации, относящиеся к определенному адресату на промежуточных узлах, не могут быть изменены другими потоками сообщений из-за свойства независимости источников. Всё это не имеет никакого отношения к методам построения таблицы маршрутизации – соответственно условие согласованности выполняется.

Необходимые условия:

- а) Для «обратного» метода построения таблицы маршрутизации, очевидно, что если в метрике отсутствует свойство «мьютекса» или свойство перехода или

свойство независимости источников, то протокол *R* может создавать петли, о выполнении условия согласованности уже не может быть и речи. Поэтому, в данном случае стоит лишь рассмотреть необходимость выполнения свойства строгого предпочтения. Если взглянуть на рис. 2, то с одной стороны для потока сообщений $f_{\langle *,d\rangle}$ связи удовлетворяют условию $w\left(l_{\overrightarrow{s_1u}}, f_{\langle *,d\rangle}\right) \prec \phi, w\left(l_{\overrightarrow{s_2u}}, f_{\langle *,d\rangle}\right) \prec \phi$ и условию $w\left(l_{\overrightarrow{uv_1}}, f_{\langle *,d\rangle}\right) = w\left(l_{\overrightarrow{uv_2}}, f_{\langle *,d\rangle}\right) \prec \phi$, где «*» соответствует s_1 или s_2. Поэтому, когда поток сообщений $f_{\langle s_1,d\rangle}$ возникает, таблица маршрутизации на узле *u* может быть настроена как [адресат = *d,* следующий узел = v_1]. Однако, во время передачи потока $f_{\langle s_1,d\rangle}$, возникает поток $f_{\langle s_2,d\rangle}$, а таблица маршрутизации на узле *u* может быть перестроена как [адресат = *d,* следующий узел = v_2] «жадным» алгоритмом нахождения пути для потока $f_{\langle s_2,d\rangle}$, таким образом, согласованность разрушается.

- б) В случае с «прямым» методом построения маршрутов требуется доказать необходимость свойства независимости источников. Для этого можно обратиться к примеру доказательства необходимых условий Теоремы 3. Если таблицы маршрутизации строятся в порядке соответствующем первичности возникновения потока $f_{\langle s_1,d\rangle}$ и вторичности возникновения потока $f_{\langle s_2,d\rangle}$, то записи в таблицах созданные сообщением поиска пути для прежнего потока, будут перезаписаны сообщением поиска последующего потока. Итак, при возникновении потока $f_{\langle s_1,d\rangle}$, записи на узлах *u,* v_1 и v_2 будут иметь следующие значения соответственно: [адресат = *d,*следующий узел = v_1], [адресат = *d,*следующий узел = v_2] и [адресат = *d,*следующий узел = *w*], а поток сообщений $f_{\langle s_1,d\rangle}$ будет передаваться по пути $s_1uv_1v_2wd$. Далее, в процессе передачи потока $f_{\langle s_1,d\rangle}$, при возникновении потока $f_{\langle s_2,d\rangle}$ и окончании его фазы поиска пути, записи в таблицах маршрутизации для узлов *u,* v_1 и v_2, обновляются следующим образом [адресат = *d,*следующий узел = v_2], [адресат = *d,*следующий узел = *w*] и [адресат = *d,*следующий узел = v_1]. В этом случае оставшиеся сообщения потока $f_{\langle s_1,d\rangle}$ будут переданы по новому пути $s_1uv_2v_1wd$ – согласованность нарушена.

Далее будут рассмотрены существующие метрики маршрутизации.

Обращаясь к рассмотренной алгебраической системе, для *MFR* также могут быть рассмотрены четыре кортежа < *L, F, w,* $\preceq$ >.

L описывает множество связей в сети, т.е. рассматривается граф *G(V, E)* и $L = \{\langle(x_u, y_u), (x_v, y_v)\rangle | \overrightarrow{uv} \in E\}$, где (x_u, y_u) и (x_v, y_v) географическое местоположение соответственно узлов *u, v* $\in V$. $F = \{f_{\langle s,d\rangle} | s, d \in V\}$ - представляет потоки в сети. *w* – весовая функция, определяемая следующим образом: $w\left(l_{\overrightarrow{uv}}, f_{\langle s,d\rangle}\right) = ||v, d||_2 - \left||u, d|\right|_2 = \sqrt{(x_v - x_d)^2 + (y_v - y_d)^2} - \sqrt{(x_u - x_d)^2 + (y_u - y_d)^2}$, где $||v, d||_2$и$\left||u, d|\right|_2$ – Евклидовы расстояния до узла *d* от узлов *v* и *u* соответственно.

Оператор предпочтения $\preceq$ определяется как: $w\left(l_{\overrightarrow{uv}}, f_{\langle s,d\rangle}\right) \preceq w(l_{\overrightarrow{uw}}, f_{\langle s,d\rangle}) \doteq w\left(l_{\overrightarrow{uv}}, f_{\langle s,d\rangle}\right) \geq w(l_{\overrightarrow{uw}}, f_{\langle s,d\rangle})$, где ≥ значит больше или равно по числовому значению.

Пороговое значение ϕ является нулём, т.е. только тогда, когда $w\left(l_{\overrightarrow{uv}}, f_{\langle s,d\rangle}\right) > 0$ сообщение может быть передано по связи $\overrightarrow{uv}$.

Легко доказать, что метрика маршрутизации имеет свойства «мьютекса», перехода и независимости источников, но не имеет строгого предпочтения. Из этого можно заключить следующее:

1. Метрика *MFR* может быть совмещена с проактивными и реактивными методами маршрутизации, что позволит получить протокол маршрутизации, который не образует петель и который сходится в сетях, где алгоритм не получает пустого множества значений выбора;
2. Если метрика комбинируется с проактивным методом маршрутизации, то протокол согласованный, однако, для реактивного метода, результат противоположный.

Эти заключения также справедливы и для схожих метрик маршрутизации, таких как: NFR, GRS, GEAR, the best PRR×distance and NADV.

При использовании метрики *Random Progress Forwarding (RPF)* в беспроводной сети *G(V, E),* сообщение передаётся тому соседнему узлу, который ближе всего к узлу-адресату, однако не по Евклидову расстоянию, а по проекции.

Значения для *L, F,* $\preceq$ и ϕ те же, что и для *MFR*, ключевая разница только в определении для весовой функции: $w\left(l_{\overrightarrow{uv}}, f_{\langle s,d\rangle}\right) = ||u, v||_2 \cos\alpha = \sqrt{(x_v - x_u)^2 + (y_v - y_u)^2} \cos\alpha$, где α – угол между рёбрами $\overrightarrow{ud}$ и $\overrightarrow{uv}$.

Для данной метрики могут выполняться условия: $w\left(l_{\overrightarrow{uv}}, f_{\langle s,d\rangle}\right) \prec \phi$ и $w\left(l_{\overrightarrow{vu}}, f_{\langle s,d\rangle}\right) \prec \phi$, что говорит об отсутствии свойства «мьютекса» в метрике, из чего можно сделать вывод, что при комбинации с проактивным или реактивным методом *RPF* не обеспечивает отсутствия петель.

Метрика *Line Progress Forwarding (LPF)* отличается от метрики *RPF* тем, что весовая функция определяется следующим образом $w\left(l_{\overrightarrow{uv}}, f_{\langle s,d\rangle}\right) = ||u,v||_2 \cos\alpha = \sqrt{(x_v - x_u)^2 + (y_v - y_u)^2}\cos\alpha$, где α угол между рёбрами $\overrightarrow{sd}$ и $\overrightarrow{uv}$, т.е. при маршрутизации учитывается местоположение и промежуточных узлов и источника с адресатом.

Данная метрика содержит свойства «мьютекса» и перехода, но не имеет свойств независимости источника и строгого предпочтения, поэтому можно сделать следующие выводы:

1. Метрика *LPF* при комбинировании с проактивной методикой маршрутизации сообщений даёт протокол маршрутизации без петель и сходящийся в сети, где алгоритм не поучает пустого множества значений выбора, также протокол будет являться согласованным;
2. При комбинировании с реактивным методом, условие отсутствия петель не может быть выполнено.

Отсутствие свойства строгого предпочтения в метрике маршрутизации также называется в некоторых работах феноменом «ничьей», поэтому ситуации с возникновением «ничьей» при выборе маршрутизатора должна быть исключена.

В географической маршрутизации, «ничья» может быть разрешена с помощью детерминистского правила касательно местоположения соседних узлов. Подробнее, если $l_{\overrightarrow{uv}}, l_{\overrightarrow{uw}} \in L$ и $w\left(l_{\overrightarrow{uv}}, f_{\langle s,d\rangle}\right) = w\left(l_{\overrightarrow{uw}}, f_{\langle s,d\rangle}\right)$ для потока сообщений от узла *s* к узлу *d*, то порядок оказания предпочтения определяется следующим образом:

$$w'\left(l_{\overrightarrow{uv}}, f_{\langle s,d\rangle}\right) \prec w'\left(l_{\overrightarrow{uw}}, f_{\langle s,d\rangle}\right) \Leftrightarrow w\left(l_{\overrightarrow{uv}}, f_{\langle s,d\rangle}\right) \prec w\left(l_{\overrightarrow{uw}}, f_{\langle s,d\rangle}\right) \| \left(l_{\overrightarrow{uv}}, f_{\langle s,d\rangle}\right) = w\left(l_{\overrightarrow{uw}}, f_{\langle s,d\rangle}\right)$$
$$\&\& \; x_v < x_w \parallel w\left(l_{\overrightarrow{uv}}, f_{\langle s,d\rangle}\right) = w\left(l_{\overrightarrow{uw}}, f_{\langle s,d\rangle}\right) \;\&\&\; x_v = x_w \;\&\&\; y_v < y_w.$$

ПРИЛОЖЕНИЕ В. АКТ ОБ ИСПОЛЬЗОВАНИИ НАУЧНЫХ РЕЗУЛЬТАТОВ ДИССЕРТАЦИОННОЙ РАБОТЫ

ГОСУДАРСТВЕННЫЙ НАУЧНЫЙ ЦЕНТР РФ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ УНИТАРНОЕ ПРЕДПРИЯТИЕ
«ГОСУДАРСТВЕННЫЙ НАУЧНО-ИССЛЕДОВАТЕЛЬСКИЙ ИНСТИТУТ АВИАЦИОННЫХ СИСТЕМ»

Россия, 125319, Москва, ул. Викторенко, 7
Тел.: (499) 157-70-47
Факс: (499) 943-86-05

Дата 05.09.2017 г. Исх. № 1900/09-233

УТВЕРЖДАЮ
Генеральный директор
ФГУП «ГосНИИАС»
С.Ю. Желтов

АКТ
об использовании научных результатов диссертационной работы Кулакова Михаила Сергеевича «Разработка принципов организации мобильных сетевых структур в авионике», представленной на соискание ученой степени кандидата технических наук

Комиссия в составе начальника подразделения 1900 ФГУП «ГосНИИАС» Фалькова Э.Я., начальника сектора 1903 ФГУП «ГосНИИАС» Егорова В.В. и ведущего инженера сектора 1943 ФГУП «ГосНИИАС» настоящим подтверждает, что:

1. Результаты аналитического исследования связности авиационной самоорганизующейся сети в отдаленных и океанических регионах по реальным полётным данным, представленные в работе Кулакова М.С., использованы при выполнении НИР «Модем» (отчет № 1601/13 от 28.07.2015 "Разработка технологий создания авиационных информационно-управляющих систем на основе транспондера АЗН-В, работающего в режиме VDL 4") и в НИР «Исследование-Норма-Транспорт-2» (гос. контракт № 107131030010 от 25.07.2013, ФГУП «Морсвязьспутник», г. Москва). В том числе, исследования позволили определить длительность периодов получения сообщений АЗН-В по сети. Это, в свою очередь, дало возможность утверждать о перспективности применения технологии мобильных самоорганизующихся сетей в авионике.
2. Результаты эффективности работы авиационной мобильной самоорганизующейся сети для различных сценариев движения сетевых узлов, полученные при исследовании имитационной модели, представленной в работе Кулакова М.С., использованы при выполнении НИР «Лист» (отчёт № 1601/2015 от 11.03.2015 «Разработка авиационных интегрированных связных технологий. Пояснительная записка») и в НИР «Айсберг» (отчёт № 147(16649)2015 от 01.08.2015). Исследования позволили определить значения параметров качества обслуживания, которые необходимо использовать при реализации приёмопередатчика стандарта VDL Mode 4 с сетевыми функциями.

Начальник подразделения 1900 ФГУП «ГосНИИАС» Фальков Э.Я.

Начальник сектора 1903 ФГУП «ГосНИИАС» Егоров В.В.

Ведущий инженер сектора 1943 ФГУП «ГосНИИАС» Дробенюк Г.Н.