

**ФЕДЕРАЛЬНОЕ АГЕНСТВО СВЯЗИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М. А. БОНЧ-БРУЕВИЧА» (СПбГУТ)
ФАКУЛЬТЕТ ИНСТИТУТ НЕПРЕРЫВНОГО ОБРАЗОВАНИЯ (ИНО)**

АНАЛИТИЧЕСКИЙ ПРАКТИКУМ №1

по дисциплине:

«Разработка имитационных моделей инфокоммуникационных сетей и систем»

тема: Ethernet и управление доступом к среде передачи

Выполнил студент:

Рыжкова Дарья Анатольевна

IV курса группы ПИБ-11з

(09.03.04 - Программная инженерия)

студенческий билет № 1905218

Санкт-Петербург, 2025

Цель практикума

Исследование механизма управления доступом к среде передачи (CSMA/CD) в сетях Ethernet и анализ влияния сетевой нагрузки на производительность шинной топологии.

Порядок выполнения практикума

Перед созданием новой модели сети необходимо добавить новый проект (**project**) и сценарий (**scenario**). Проект – это группа зависимых сценариев, каждый из которых описывает различные детали сети. Проект может содержать множество сценариев.

Создание нового проекта:

1. Запуск **Riverbed Modeler Academic Edition**, в меню **File** выбрать **New...**
2. Выбрать **Project** \Rightarrow **OK** \Rightarrow озаглавить проект как *<номер_студенческого_билета>_Ethernet* \Rightarrow *1905218_Ethernet*, а сценарий – как *Coax_a* \Rightarrow **OK**.

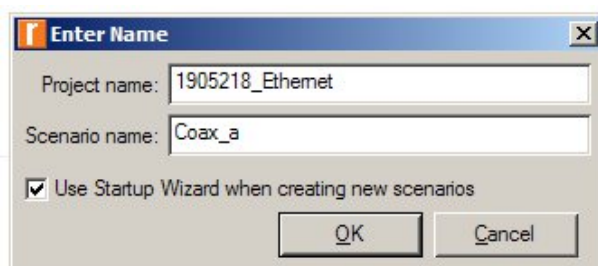


Рисунок 1. Создание проекта *1905218_Ethernet* и сценария *Coax_a*

После создания нового сценария для добавления нужно использовать начальный **<мастер запуска> (Startup Wizard)**. Параметры мастера позволяют: определить начальную топологию сети, масштаб и размер сети; выбрать карту для фона сети (план здания, города); сопоставить палитру компонентов (базу ресурсов сети) сценарию.

3. В окне **Startup Wizard: Initial Topology** выбрать **Create empty scenario** \Rightarrow нажать **Next** \Rightarrow в списке **Network Scale** выбрать **Office** \Rightarrow **Next**.

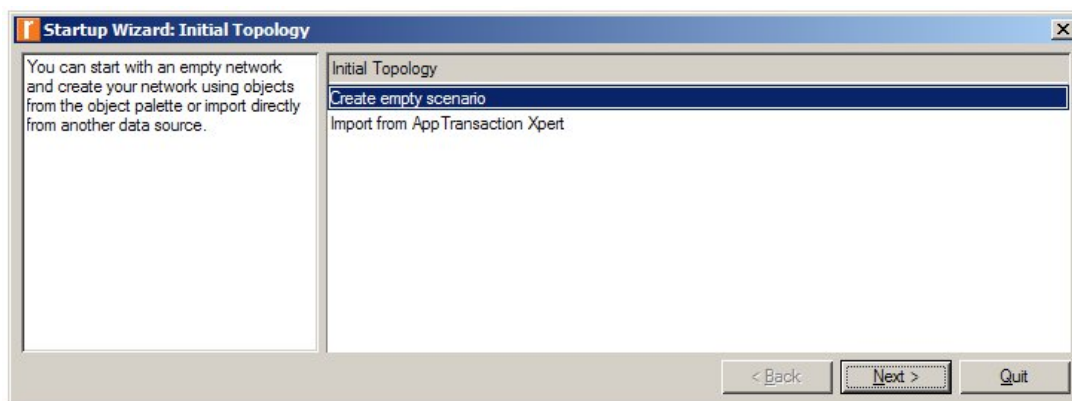


Рисунок 2. Создание пустого сценария *Empty scenario* в диалоговом окне начальной топологии *Initial Topology*

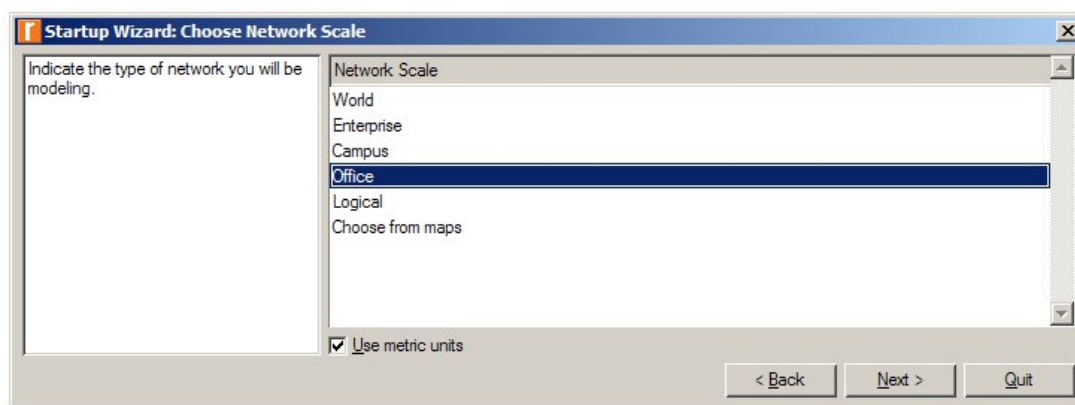


Рисунок 3. Выбор масштаба сети (диалоговое окно *Choose Network Scale*). Выбираем *Office* – сеть масштаба офиса. Выбрана опция *Use metric units* – использование метрической системы

4. В окне **Startup Wizard: Specify Size** в поле **X Span** ввести **200**, в поле **Y Span** ввести **100** \Rightarrow дважды на **Next** \Rightarrow **Finish**.

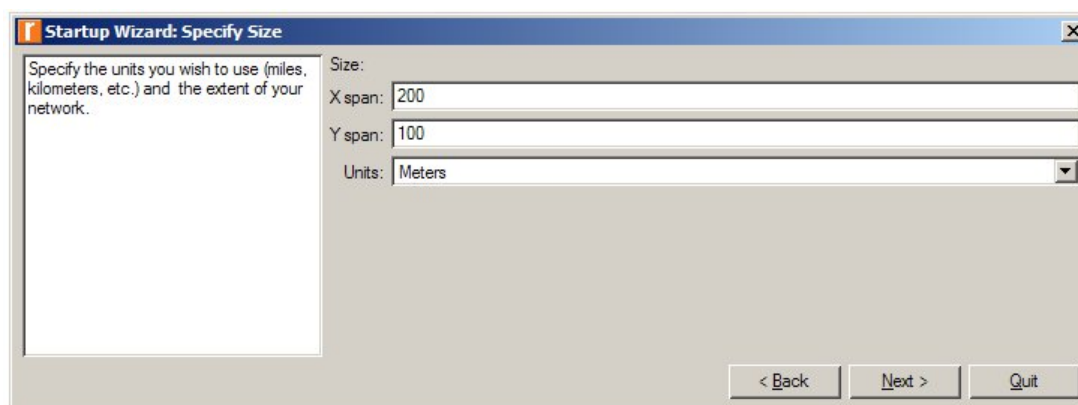


Рисунок 4. Определение размера рабочей области 200x100м. Диалоговое окно *Select Technologies* для выбора используемых технологий в данной лабораторной работе пропускаем со значениями, выставленными по умолчанию

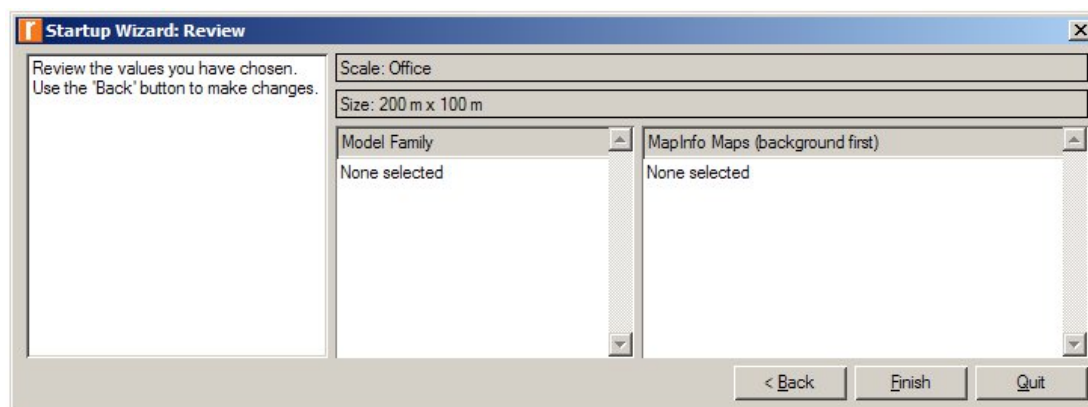


Рисунок 5. В окне обзора Review производится проверка выбранных ранее значений создаваемой сети

5. Закрывать открывшееся окно базы ресурсов **Object Palette**.

Модель сети создается с помощью редактора, используя узлы (nodes) и каналы связи (links) из базы ресурсов (object palette). Узел сети – это реальный объект сети, способный передавать и принимать информацию. Канал связи представляет собой среду передачи данных, между узлами, которая может быть электрическим или оптоволоконным кабелем. Все эти объекты доступны в базе ресурсов (окне с изображениями узлов и связей).

Для создания топологии сети можно использовать один из трех методов или их комбинацию. Первый метод предполагает импорт существующей топологии. Второй — ручная расстановка узлов и связей в рабочей области. Третий метод, который мы будем использовать, — это метод «быстрого создания топологии» (**Rapid Configuration**). Этот метод позволяет быстро создать сеть, выбрав тип топологии, узлов и каналов связи, что упрощает процесс проектирования сети.

Метод Rapid Configuration создает сеть за одно действие, после выбора топологии сети, типов узлов и типов связей между узлами.

Создание сети Ethernet с топологией 'шина' (Bus)

1. В меню **Topology** выбрать **Rapid Configuration...**
2. В выпадающем меню **Configuration** выбрать *Bus* ⇒ **Next**.

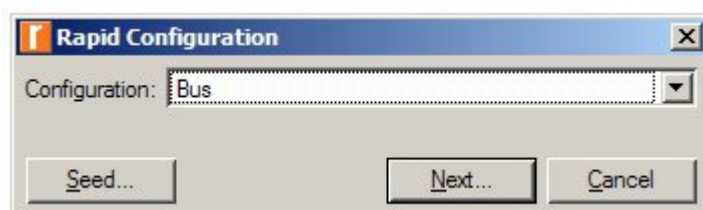


Рисунок 6. Выбор типа топологии сети

3. В открывшемся окне **Rapid Configuration: Bus** \Rightarrow **Select Models...**; в выпадающем меню **Model list** выбрать *ethcoax* \Rightarrow **OK**.
4. В окне **Rapid Configuration: Bus** установить следующие 8 значений (см. на рисунке ниже) \Rightarrow **OK**.

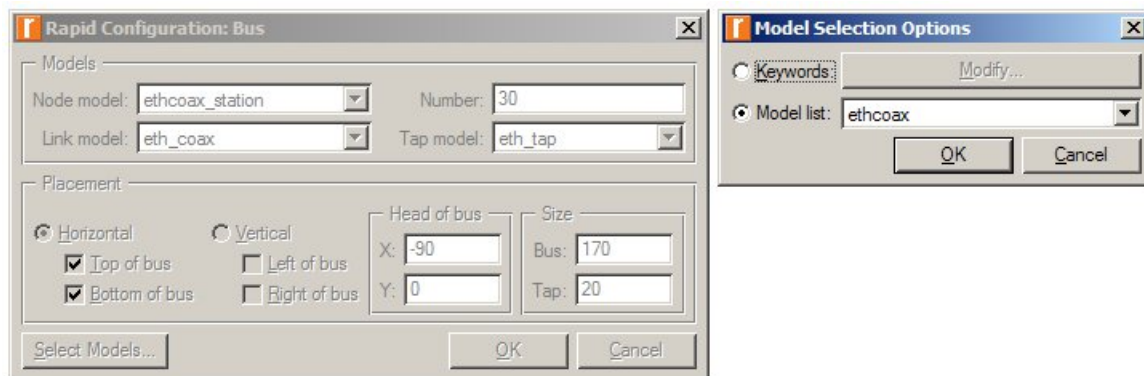


Рисунок 7. Создание сети Ethernet шинной топологии с 30 узлами, ее позиционирование в рабочей области

Модель **eth_coax** для создания коаксиальной сети Ethernet – это шина Ethernet, которая позволяет соединять узлы с приёмниками и передатчиками шины через отводы (тапы).

По состоянию на 2011 год использование коаксиального кабеля для Ethernet было признано устаревшим стандартом по решению Института инженеров электротехники и электроники (IEEE).

5. Чтобы сконфигурировать общую шину: ПКМ на горизонтальной линии (Ethernet Bus) \Rightarrow в открывшемся меню выбрать **Edit Attributes (Advanced)**.
6. ЛКМ на параметре *eth_coax* \Rightarrow в выпадающем меню выбрать **Edit...** \Rightarrow в списке выбрать *eth_coax_adv* \Rightarrow **OK**.

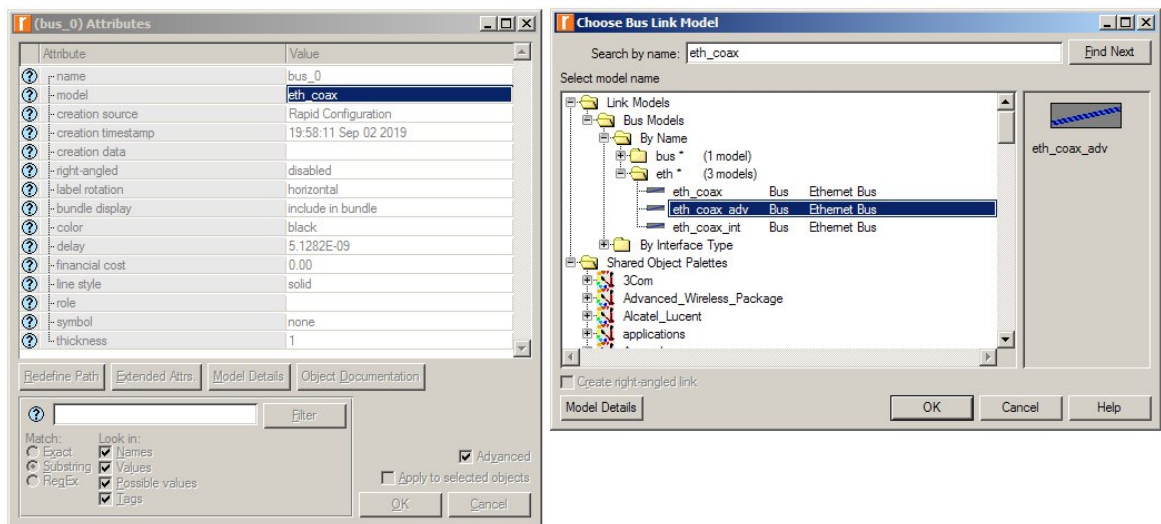


Рисунок 8. Настройка коаксиальной шины

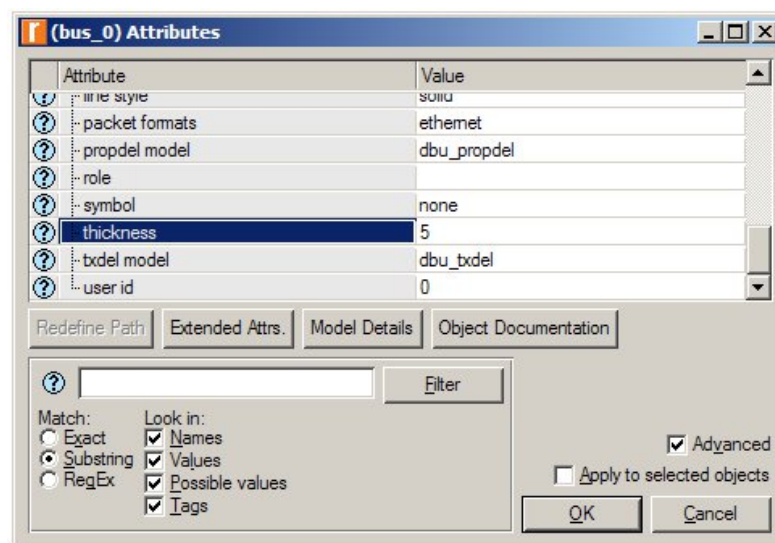


Рисунок 9. Установка параметра *thickness* для шины

7. Установить значение параметра *thickness* равным 5 \Rightarrow OK.

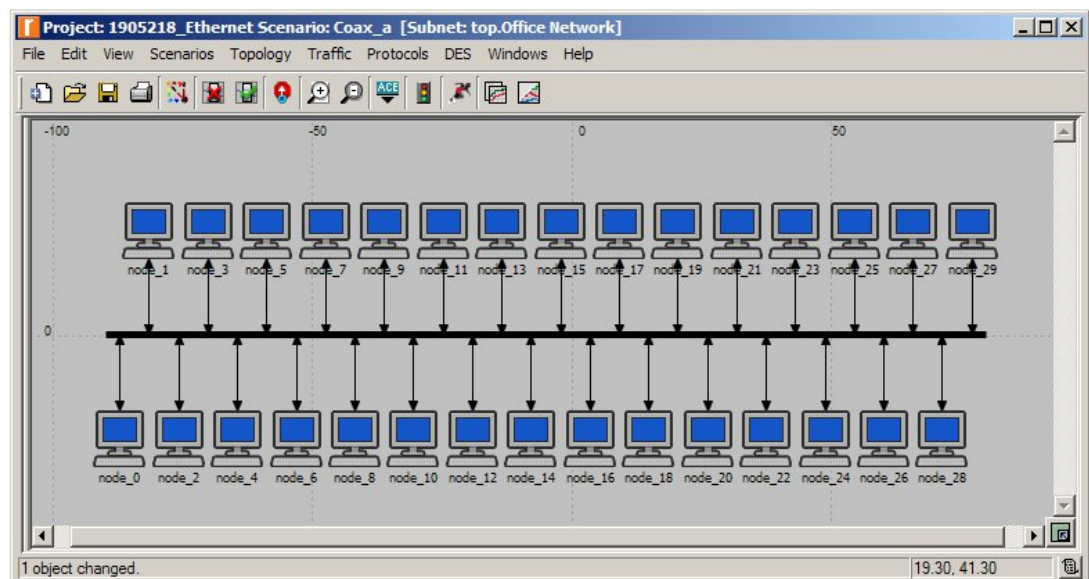


Рисунок 10. Получившаяся сеть после настройки конфигурации Rapid Configuration

Генерация трафика

1. ПКМ на любом из 30 узлов \Rightarrow выбрать **Select Similar Nodes**, чтобы выделить разом все узлы в сети.

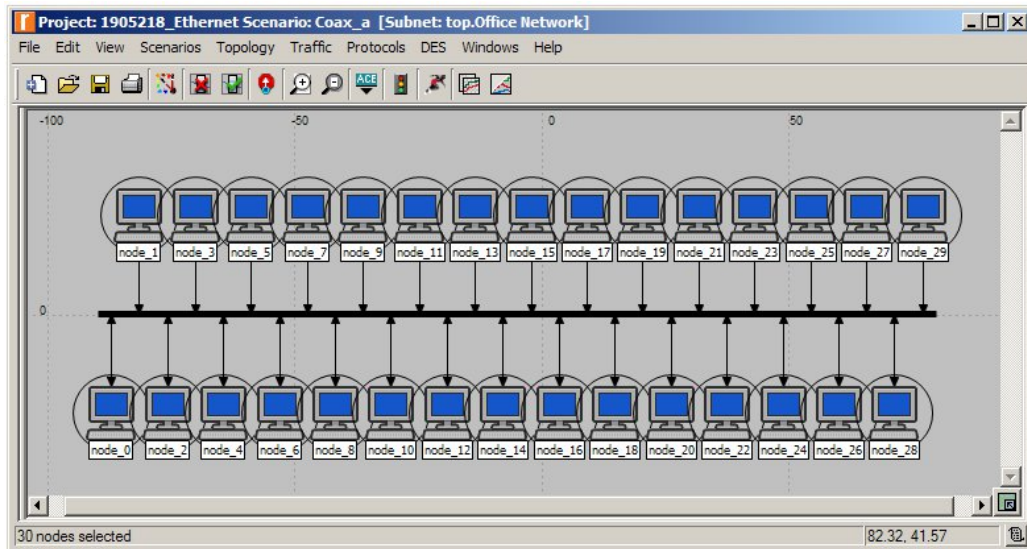


Рисунок 11. Выделение узлов сети для их последующей настройки

2. ПКМ на любом из выделенных 30 узлов \Rightarrow в появившемся меню выбрать **Edit Attributes**.
3. В открывшемся окне поставить галочку в поле **Apply to selected objects**, чтобы избежать необходимости конфигурировать каждый узел в отдельности.

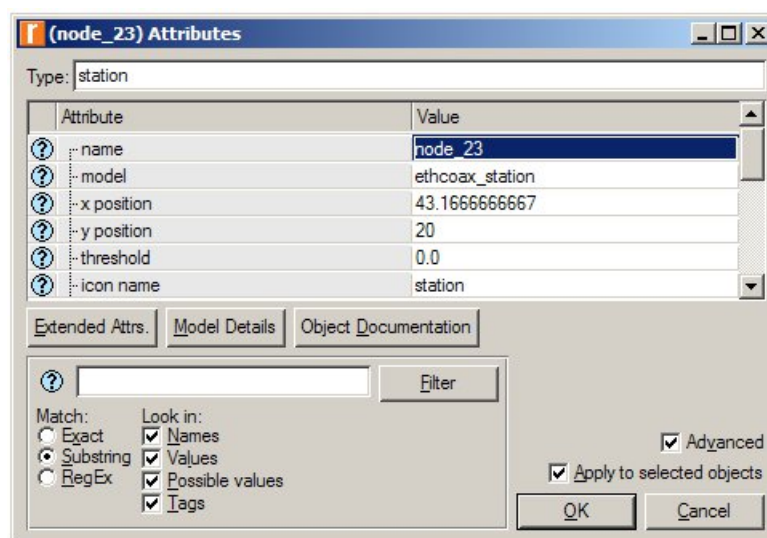


Рисунок 12. Окно настройки атрибутов узлов сети

4. Открыть иерархическое меню **Traffic Generation Parameters** \Rightarrow изменить значение параметра *ON State Time* на *exponential(100)*, а параметра *OFF State Time* – на *constant(0)*.

Тип распределения для ON State Time с параметром *exponential(100)*. Это означает, что время, в течение которого узел находится в состоянии передачи данных, будет следовать экспоненциальному распределению со средним значением 100 секунд. Это позволяет моделировать случайные интервалы активности узла.

Тип распределения для OFF State Time со значением 0. Это означает, что узел не будет находиться в состоянии простоя, всегда будучи готовым к передаче данных.

5. Открыть иерархическое меню **Packet Generation Arguments** \Rightarrow изменить значение параметра *Packet Size (bytes)* на *constant(1024)*.

Это означает, что все пакеты, генерируемые узлами, будут иметь одинаковый размер — 1024 байта. Это позволяет моделировать трафик с постоянным размером пакетов, что может быть полезно для анализа производительности сети при определенных условиях.

6. Установить значение параметра *Interarrival Time (seconds)* как *exponential(0.1)* \Rightarrow **ОК**.

Interarrival Time (временной интервал между пакетами) — ключевой параметр, определяющий частоту генерации трафика в Riverbed Modeler.

Interarrival Time задает временные промежутки между отправкой пакетов узлом. Экспоненциальное распределение со средним значением 0.1 секунды имитирует случайный характер трафика, характерный для реальных сетей (например, веб-запросы или VoIP).

Нагрузка на сеть: Чем меньше среднее значение (например, 0.05 вместо 0.1), тем чаще генерируются пакеты, увеличивая нагрузку. Нагрузка рассчитывается по формуле:

$$Load = \frac{Packet\ Size \times 8}{Interarrival\ Time} \cdot Number\ of\ Nod$$

В нашем случае, *exponential(0.1)* — умеренная нагрузка, для 1024-байтовых пакетов и 30 узлов нагрузка равна:

$$Load = \frac{1024 \times 8}{0,1} \cdot 30 = 81,920 \text{бит/с} \times 30 = 2\,457,6 \text{Кбит/с} = 2,4 \text{Мбит/с}$$

Также более частые пакеты (малые значения Interarrival Time) увеличивают вероятность коллизий в Ethernet-сетях с шинной топологией.

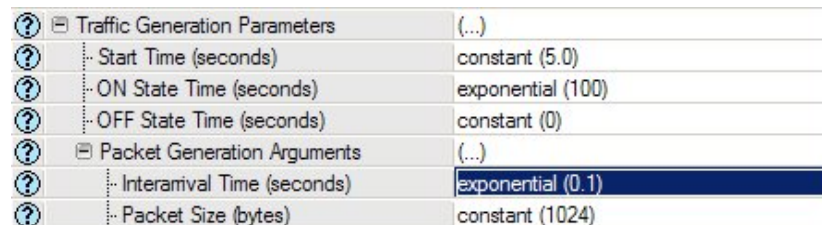


Рисунок 13. Настройка параметров Traffic Generation Parameters

7. В открывшемся окне **Warning** нажать **Yes** \Rightarrow сохранить проект.

Сбор статистики

1. ПКМ где-либо на рабочей области (но не на одном из элементов сети) \Rightarrow в появившемся меню выбрать **Choose Individual DES Statistics** \Rightarrow открыть иерархическое меню **Global Statistics**.
2. Открыть иерархическое меню **Ethcoax** \Rightarrow выбрать *Delay (sec)*.
3. Открыть иерархическое меню **Traffic Sink** \Rightarrow выбрать *Traffic Received (bits/sec)* и *Traffic Received (packets/sec)*.
4. Открыть иерархическое меню **Traffic Source** \Rightarrow выбрать *Traffic Sent (bits/sec)* и *Traffic Sent (packets/sec)* \Rightarrow **OK**.

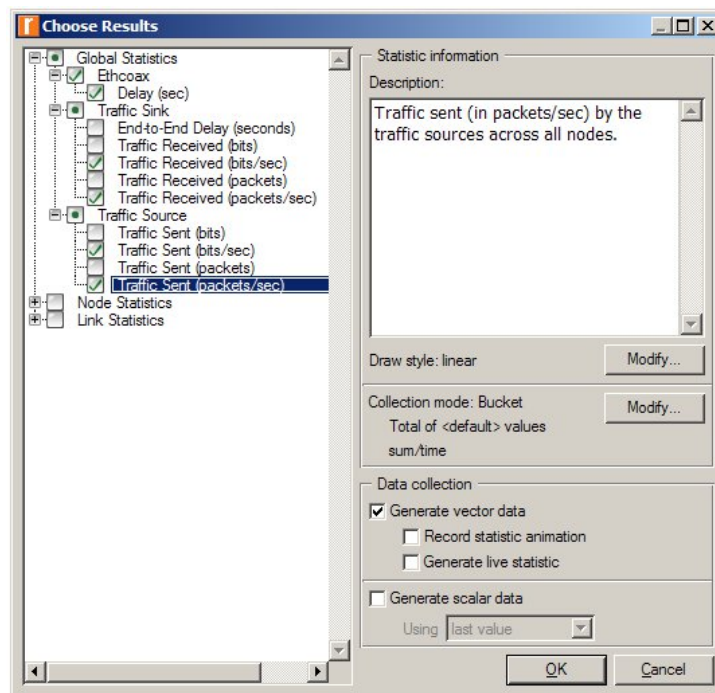


Рисунок 14. Выбор глобальных статистик

5. ПКМ на узле *node_0* \Rightarrow в появившемся меню выбрать **Choose Individual DES Statistics** \Rightarrow открыть иерархическое меню **Ethcoax** \Rightarrow выбрать *Collision Count* \Rightarrow **OK**.

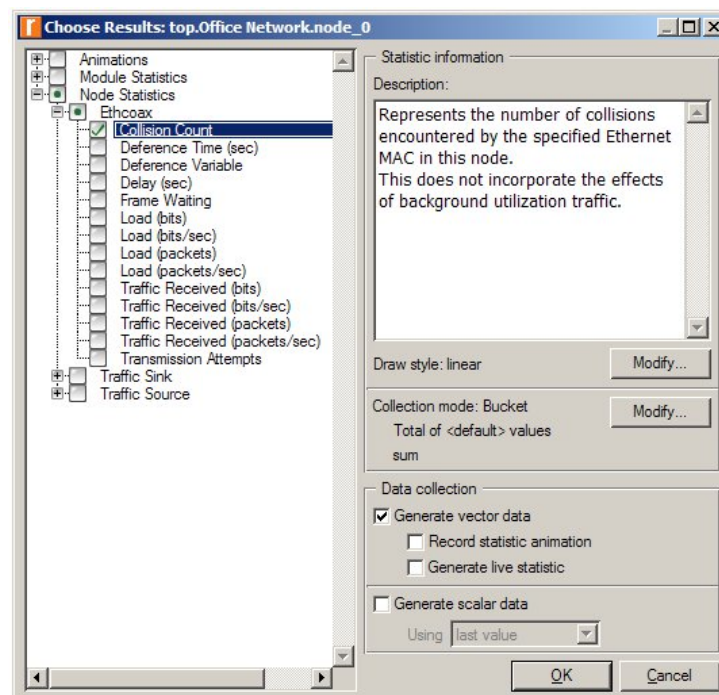


Рисунок 15. Выбор параметров для сбора статистики из древа статистик для узла 0

Настройка и запуск имитационного моделирования

1. На панели инструментов нажать кнопку **Configure/Run Discrete Event Simulation (DES)**.

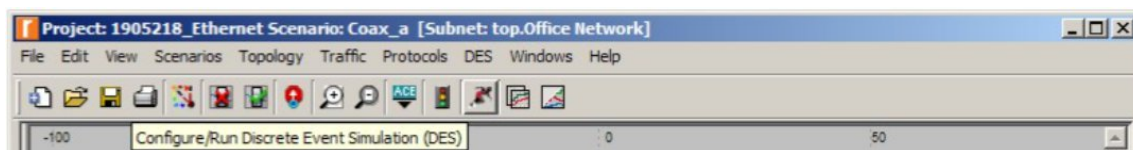


Рисунок 16. Панель инструментов

2. В открывшемся окне **Configure/Run DES** установить значение параметра *Duration* равным *15 second(s)* – продолжительность моделирования: 15 секунд сетевой активности (модельное время).

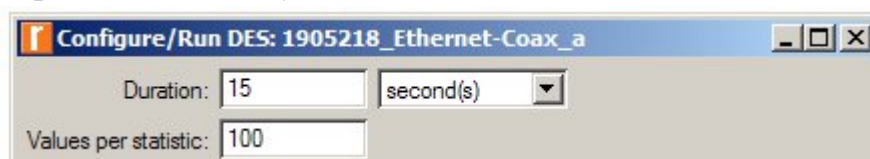


Рисунок 17. Задание длительности прогона

3. Чтобы запустить моделирование, нужно нажать **Run**.

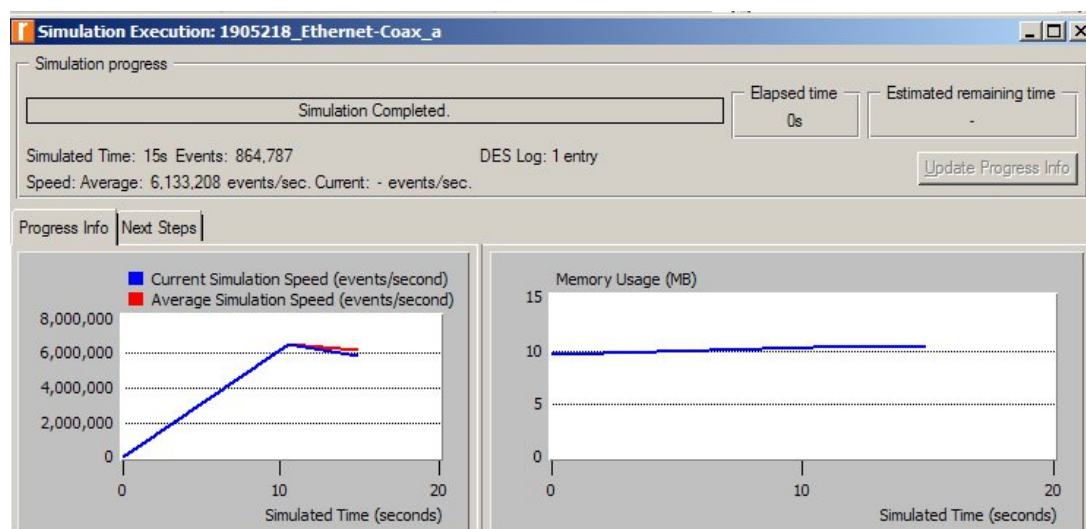


Рисунок 18. Окно *Simulation Execution* выполнения прогона, вкладка *Progress Info*

- ☐ Elapsed time – длительность прогона в секундах (реальное время).
 - ☐ Simulated time – модельное время.
4. По окончании нажать **Close** \Rightarrow сохранить проект.

Создание копий сценария

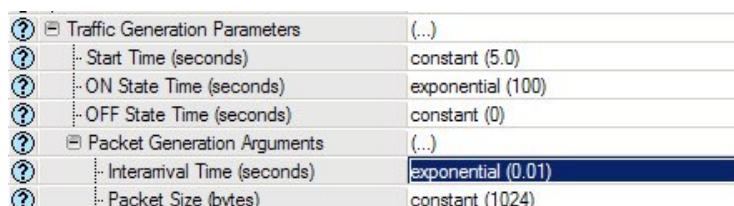
Во время сравнения сетей необходимо сохранить базовую сеть как один сценарий, а экспериментальную сеть создать в другом сценарии. Необходимо

скопировать существующий сценарий и уже в копии можно делать изменения топологии.

Анализ влияния интервала между поступлениями пакетов на работу сети

1. Создать 2 копии сценария **Coax_a** и озаглавить их как **Coax_b** и **Coax_c**.
 - ☐ Чтобы создать копию сценария, в меню **Scenarios** следует выбрать **Duplicate Scenario...** \Rightarrow присвоить имя копии \Rightarrow **OK**.
 - ☐ Для переключения между сценариями в меню **Scenarios** выбрать **Switch To Scenario** \Rightarrow выбрать нужный сценарий.
2. Для всех узлов установить значение параметра **Interarrival Time (seconds)** как:

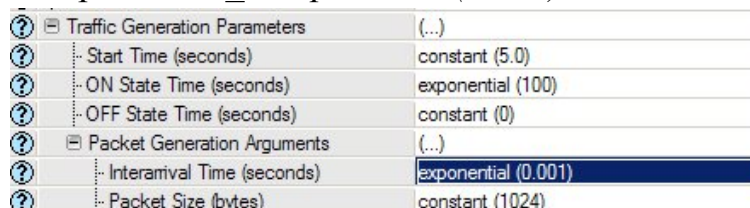
- ☐ В сценарии **Coax_b**: *exponential(0.01)*



?	[-] Traffic Generation Parameters	(...)
?	Start Time (seconds)	constant (5.0)
?	ON State Time (seconds)	exponential (100)
?	OFF State Time (seconds)	constant (0)
?	[-] Packet Generation Arguments	(...)
?	Interarrival Time (seconds)	exponential (0.01)
?	Packet Size (bytes)	constant (1024)

Рисунок 19. Настройка параметра *Interarrival Time*

- ☐ В сценарии **Coax_c**: *exponential(0.001)*



?	[-] Traffic Generation Parameters	(...)
?	Start Time (seconds)	constant (5.0)
?	ON State Time (seconds)	exponential (100)
?	OFF State Time (seconds)	constant (0)
?	[-] Packet Generation Arguments	(...)
?	Interarrival Time (seconds)	exponential (0.001)
?	Packet Size (bytes)	constant (1024)

Рисунок 20. Настройка параметра *Interarrival Time*

- ☐ Чтобы сконфигурировать все узлы разом, следует использовать опции **Select Similar Nodes** и **Apply to selected objects**.
3. Произвести запуск имитационного моделирования для созданных копий.

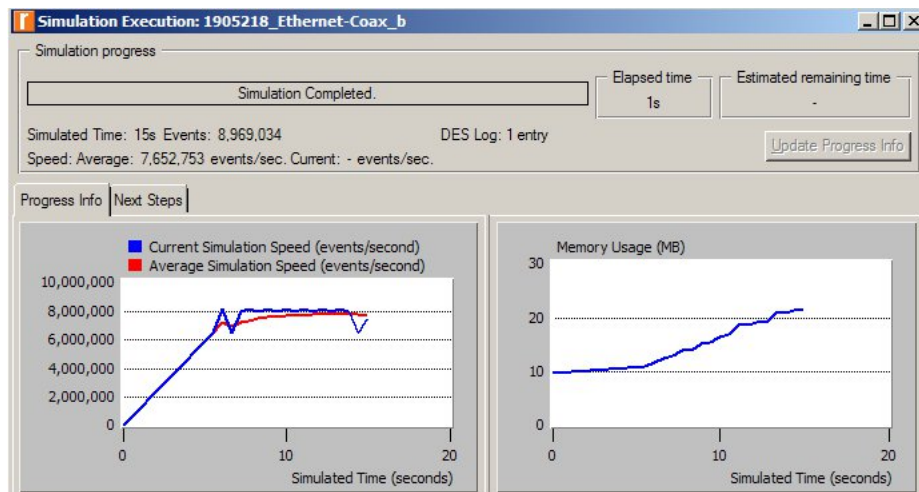


Рисунок 21. Информация выполнения прогона сценария Coax_b

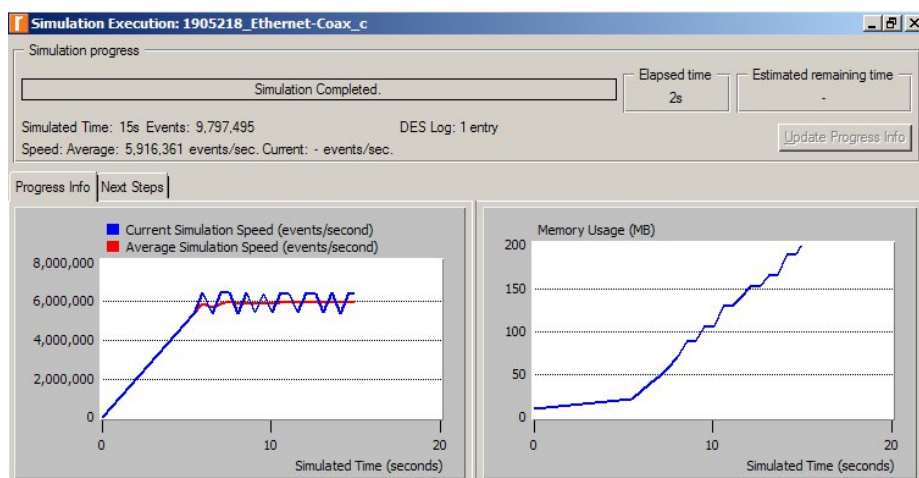


Рисунок 22. Информация выполнения прогона сценария Coax_c

Просмотр результатов моделирования

1. В меню **DES** выбрать **Results** \Rightarrow выбрать **Compare Results...**
2. В открывшемся окне **Results Browser** выбрать *все 3 сценария*.
3. Открыть иерархическое меню **Global Statistics** \Rightarrow выбрать метрику *Delay (sec)* \Rightarrow в выпадающем меню выбрать **Overlaid Statistics** и **As Is** \Rightarrow нажать **Show**.

Overlaid Statistics позволяет накладывать несколько графиков на один, что полезно для сравнения разных сценариев или параметров.

As Is означает, что статистика будет отображаться в исходном виде, без дополнительных преобразований.

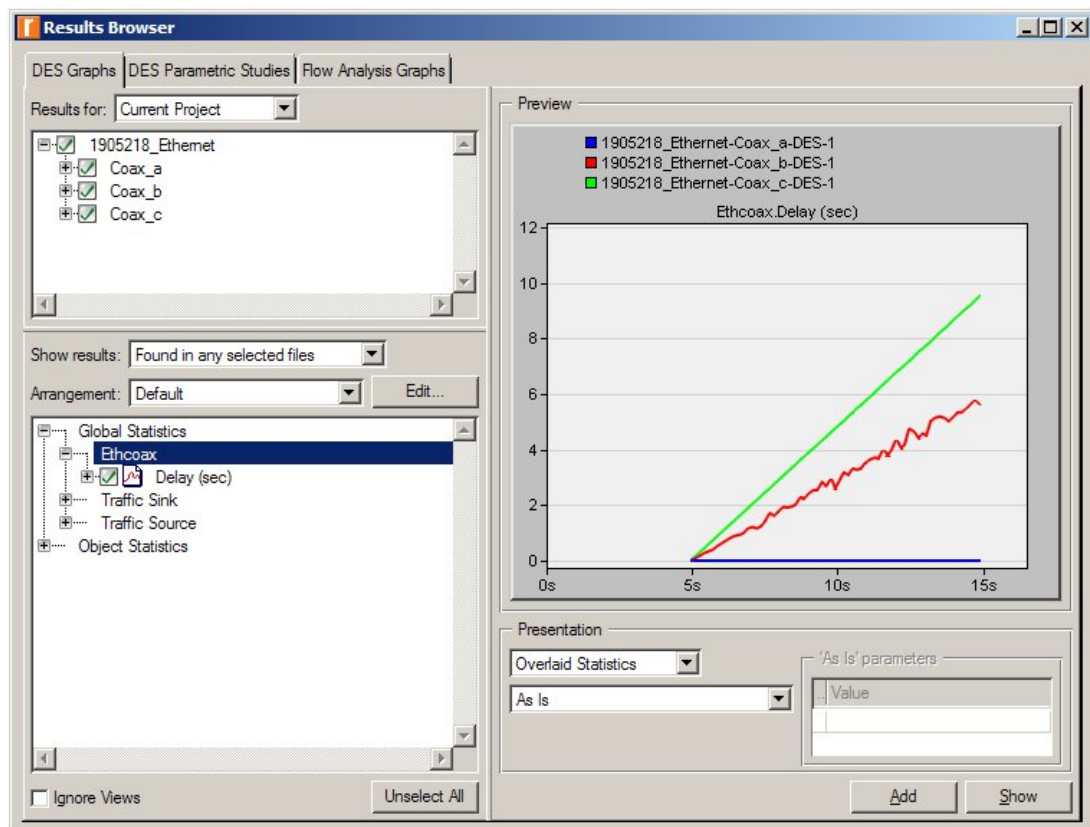


Рисунок 23. Окно выбора статистики для сравнения сценариев

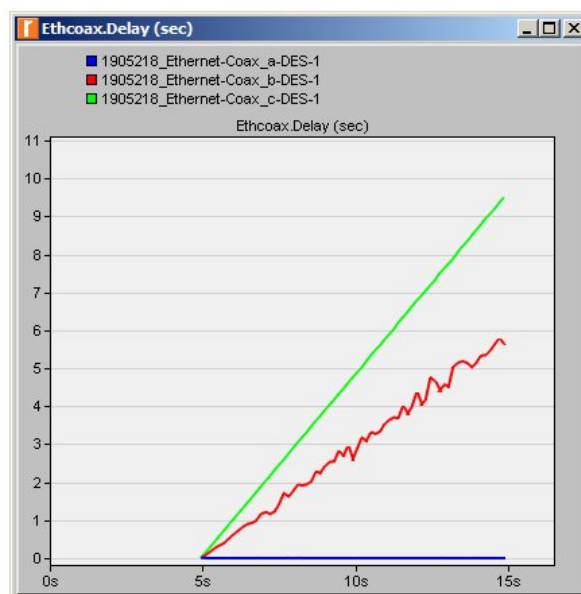


Рисунок 24. График для метрики Delay (sec) для сценариев Coax_a, Coax_b, Coax_c

Параметр Interarrival Time определяет частоту генерации пакетов. Чем меньше значение, тем чаще узлы отправляют данные:

- ☐ Coax_a: exponential(0.1) → Средний интервал 0.1 сек (10 пакетов/сек на узел).
- ☐ Coax_b: exponential(0.01) → 0.01 сек (100 пакетов/сек на узел).
- ☐ Coax_c: exponential(0.001) → 0.001 сек (1000 пакетов/сек на узел).

Задержка в сценарии Соах_а остается минимальной и стабильной на протяжении всего времени моделирования. Это соответствует ожиданиям, поскольку Interarrival Time = 0.1 сек обеспечивает относительно низкую нагрузку на сеть.

В сценарии Соах_b линия на графике демонстрирует нестабильный рост, с заметными скачками, и достигает значения близко к 6 секундам. Задержка значительно выше и нестабильна. Перепады на графике могут быть вызваны коллизиями и очередями пакетов, что характерно для сетей с умеренной нагрузкой.

В сценарии Соах_с задержка растет линейно и достигает максимальных значений. Параметр Interarrival Time = 0.001 сек создает чрезвычайно высокую нагрузку на сеть.

Вывод: Сеть в Соах_а не перегружена, задержка практически отсутствует. Сеть в Соах_b испытывает значительные коллизии и перегрузки, что требует оптимизации. Сеть в Соах_с сильно перегружена, задержка становится неприемлемо высокой из-за огромного количества пакетов.

Соах_а подходит для низкой нагрузки, задержка минимальна. Соах_b требует оптимизации для снижения коллизий и задержки. Соах_с требует значительных изменений в сети для снижения перегрузки и задержки.

4. Аналогично построить графики для метрик:

- ❑ Collision Count для узла node_0

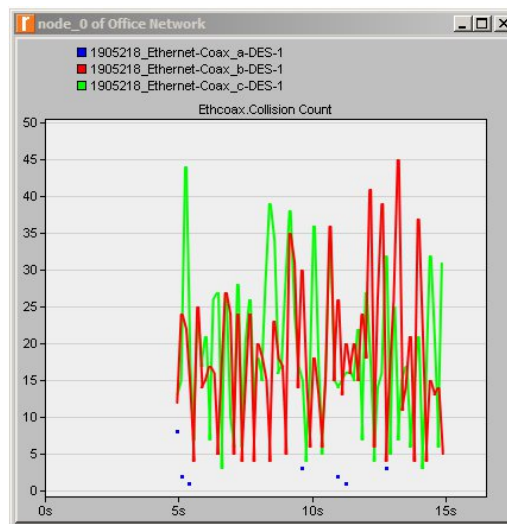


Рисунок 25. График для метрики Collision Count для узла node_0

В сценарии Соах_а количество коллизий на узле node_0 очень мало. Значения коллизий стабильны и не превышают отметку 5, что соответствует ожиданиям для сети с низкой нагрузкой.

Графики Соах_b и Соах_c: оба графика демонстрируют значительные колебания в количестве коллизий, что указывает на нестабильность сети и частые столкновения пакетов. Хотя оба графика имеют сильные перепады, их общий уровень и динамика схожи. Это говорит о том, что оба сценария испытывают высокую нагрузку, но различия в частоте коллизий между ними не существенны.

Вывод: Сеть в Соах_a стабильна, коллизии минимальны. Сети в Соах_b и Соах_c сильно перегружены, что приводит к частым коллизиям.

□ Traffic Sent (bits/sec) & Traffic Received (bits/sec)

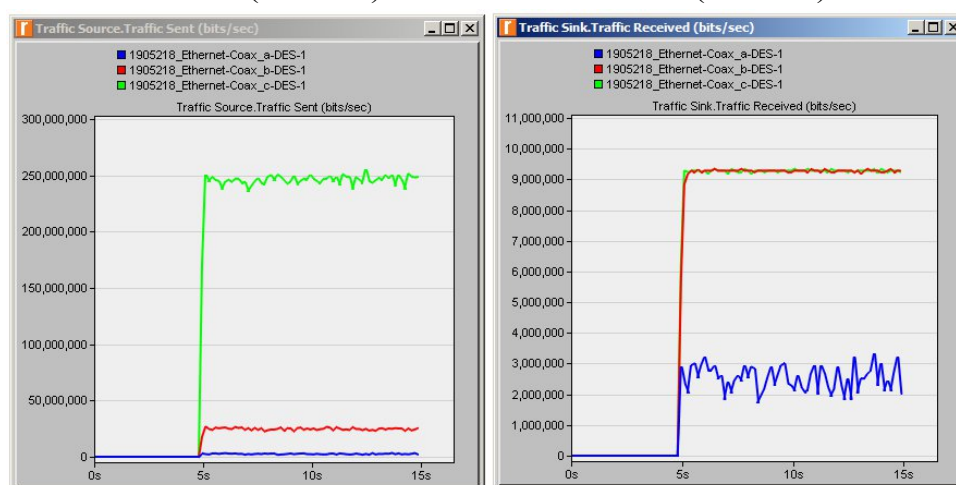


Рисунок 26. Графики для метрик Traffic Sent (bits/sec) и Traffic Received (bits/sec)

Анализ графиков по метрике Traffic Sent (bits/sec). Все три графика до 5 секунд модельного времени находятся на нуле: это указывает на то, что в начальный период моделирования (до 5 секунд) нет передачи трафика в любой из сетей. Это может быть связано с настройками моделирования или задержкой в инициализации узлов.

После 5 секунд график Соах_a остается близким к нулю, что означает очень низкий уровень передачи трафика. Это соответствует ожиданиям для сети с низкой нагрузкой и большим Interarrival Time (0.1 сек).

После 5 секунд график Соах_b начинает расти и достигает стабильного значения около 25 Мбит/сек. Колебания на графике незначительны, что указывает на более стабильную передачу трафика относительно сценария Соах_c.

График Соах_c демонстрирует более резкий рост и достигает значений около 250 Мбит/сек. Колебания на этом графике более выражены. Высокая частота коллизий и перегрузки сети объясняют значительные колебания в передаче трафика.

Анализ графиков по метрике Traffic Received (bits/sec). График Соах_a по истечении 5 секунд начинает расти, но с заметными перепадами, со

значениями 2-3Мбит/с. Это может быть связано с низкой частотой отправки пакетов и отсутствием перегрузок в сети.

Графики Соах_b и Соах_c демонстрируют более высокие значения принятого трафика около 9Мбит/с и стабильную передачу с минимальными колебаниями. Это указывает на то, что обе сети активно используют канал и эффективно обрабатывают трафик.

Вывод: Сеть в Соах_a принимает трафик с относительно низкой скоростью, что соответствует низкой нагрузке и большим Interarrival Time (0.1 сек). Сети в Соах_b и Соах_c имеют высокую пропускную способность и относительно стабильную передачу трафика, несмотря на высокую нагрузку. Однако в Соах_c могут быть проблемы с перегрузками и коллизиями, что требует дополнительной оптимизации.

□ Traffic Sent (packets/sec) & Traffic Received (packets/sec)

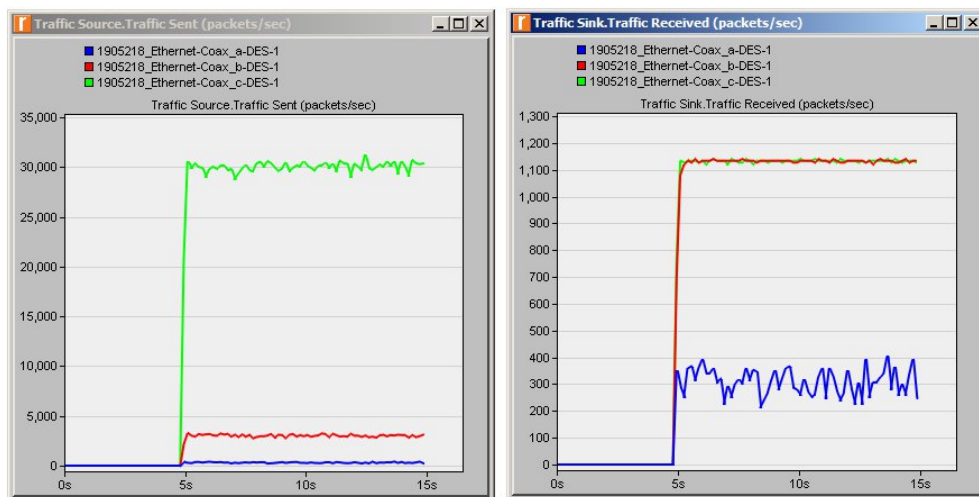


Рисунок 27. Графики для метрик Traffic Sent (packets/sec) и Traffic Received (packets/sec)

После 5 секунд график Соах_a остается близким к нулю, что означает очень низкий уровень отправки пакетов. График Соах_b возрастает до значения около 3200 пакетов/сек, с небольшими колебаниями, что указывает на относительно стабильную отправку пакетов. График Соах_c с более резким ростом, достигает значений около 30,000 пакетов в секунду, в сети высокая нагрузка и значительные колебания в отправке пакетов из-за перегрузок.

После 5 секунд график Соах_a начинает расти, но с заметными перепадами со значениями 200-400 пакетов/сек. Графики Соах_b и Соах_c возрастают до значения около 1150 пакетов/сек с очень маленькими колебаниями.

Вывод: в Соах_a низкая скорость приема пакетов с минимальной задержкой, что говорит о низкой эффективности передачи трафика, следует

увеличить частоту отправки пакетов для более эффективного использования канала. В Соax_b и Соax_c, наоборот, высокая скорость приема пакетов и стабильная передача указывают на эффективное использование канала. Однако, в Соax_c могут быть проблемы с перегрузками и коллизиями, что требует дополнительной оптимизации.

Получившиеся в результате графики аналогичны приведенным в методическом пособии по выполнению лабораторной работы.

Анализ влияния количества станций на работу сети

- ☐ Создать копию сценария *Coax_b* и озаглавить её как *Coax_d*
- ☐ В этом новом сценарии удалить все узлы с нечетными номерами (*node_1, node_3, ..., node_29*)

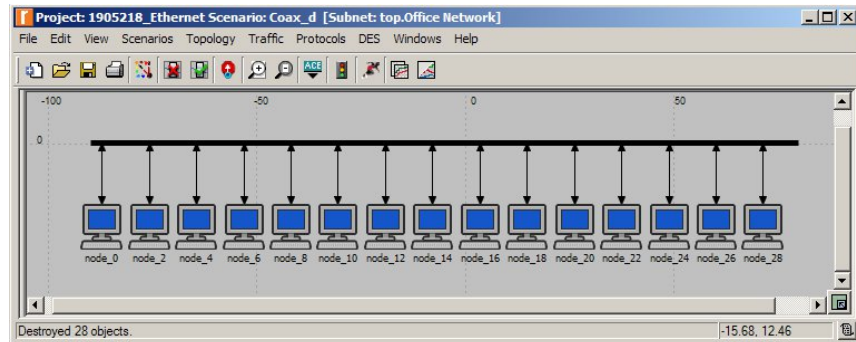


Рисунок 28. Измененная сеть сценария *Coax_d*

- ☐ Запустить имитационное моделирование для сценария *Coax_d*

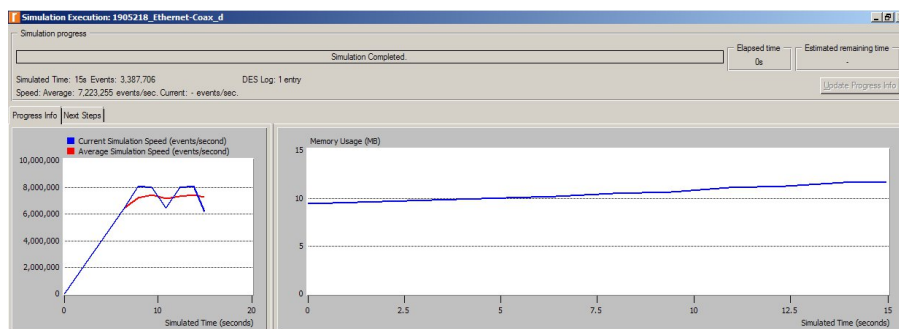


Рисунок 29. Результат запуска имитационного моделирования *Coax_d*

- ☐ По окончании сохранить проект
- ☐ Для сценариев *Coax_b* и *Coax_d* построить сравнительные графики:
 - ☐ Traffic Sent (packets/sec)

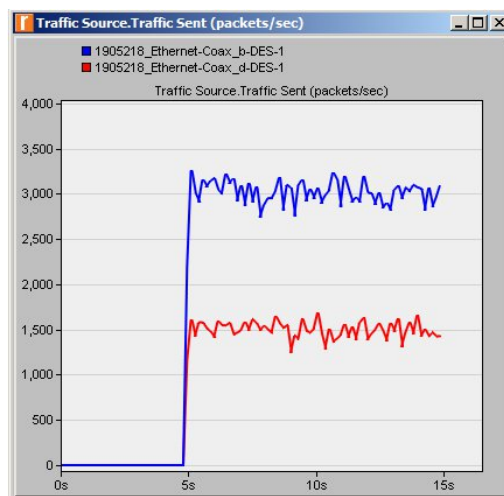


Рисунок 30. Графики для метрики *Traffic Sent (packets/sec)* для сценариев *Coax_b* и *Coax_d*

В начальный период моделирования (до 5 секунд) нет отправки пакетов в любой из сетей. График Соах_b демонстрирует рост, но с заметными колебаниями. Это может быть связано с коллизиями и перегрузками в сети.

График Соах_d показывает более высокую скорость отправки пакетов, чем Соах_b, но также с заметными колебаниями. Это может быть связано с тем, что удаление узлов изменило распределение нагрузки в сети.

Вывод: Сеть в Соах_b испытывает проблемы со стабильностью передачи. Сеть в Соах_d имеет более высокую скорость отправки пакетов, но стабильность передачи снижена. Высокая скорость отправки пакетов и заметные колебания указывают на то, что сеть испытывает перегрузки или проблемы с топологией. Необходимо уменьшить нагрузку или оптимизировать сеть для стабильной передачи.

□ Delay (sec)

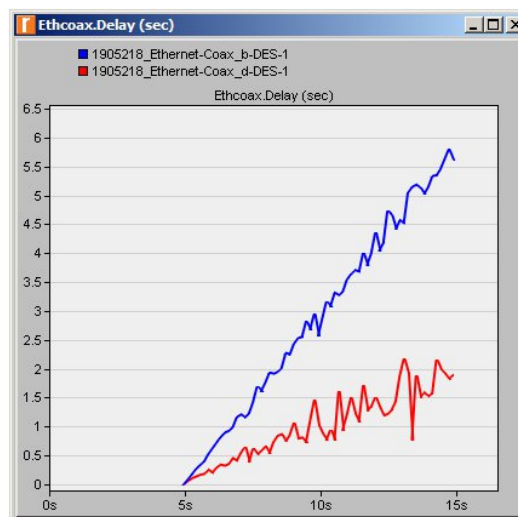


Рисунок 31. Графики для метрики Delay (sec) для сценариев Соах_b и Соах_d

График Соах_b демонстрирует рост задержки, что может быть связано с коллизиями и перегрузками в сети. Беспорядочные перепады указывают на нестабильность сети. График Соах_d показывает более низкую задержку, чем Соах_b, но с более резкими колебаниями. Это может быть связано с тем, что удаление узлов уменьшило общую нагрузку на сеть, но изменило распределение трафика.

Вывод: Сеть в Соах_b испытывает значительную задержку из-за высокой нагрузки и коллизий. Сеть в Соах_d имеет более низкую задержку, чем Соах_b, но стабильность передачи снижена из-за резких перепадов.

Анализ влияния размера пакета на работу сети

1. Создать копию сценария *Coax_b* и озаглавить её как *Coax_e*
2. В этом новом сценарии для всех узлов установить значение параметра **Packet Size (bytes)** как *constant(512)*

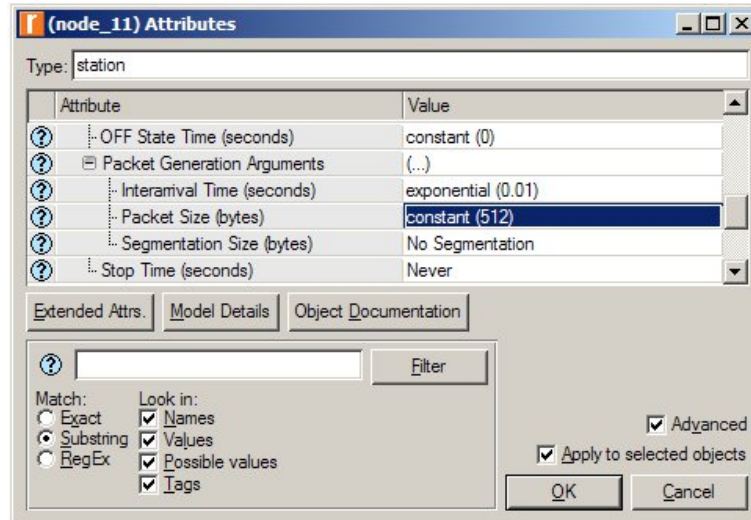


Рисунок 32. Изменение размера пакета узлов сценария *Coax_e*

- ☐ Обратите внимание, что в предыдущих сценариях размер пакета был равен 1024 байт
 - ☐ Чтобы сконфигурировать все узлы разом, используйте опции **Select Similar Nodes** и **Apply to selected objects**
3. Запустить имитационное моделирование для сценария *Coax_e*

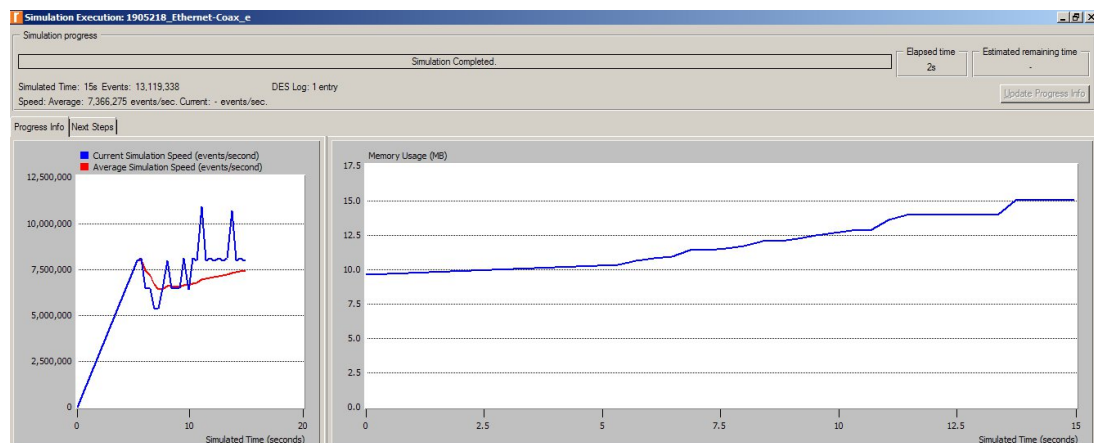


Рисунок 33. Результат запуска имитационного моделирования *Coax_e*

4. По окончании сохранить проект
5. Для сценариев *Coax_b* и *Coax_e* построить сравнительные графики:
 - ☐ Delay (sec)

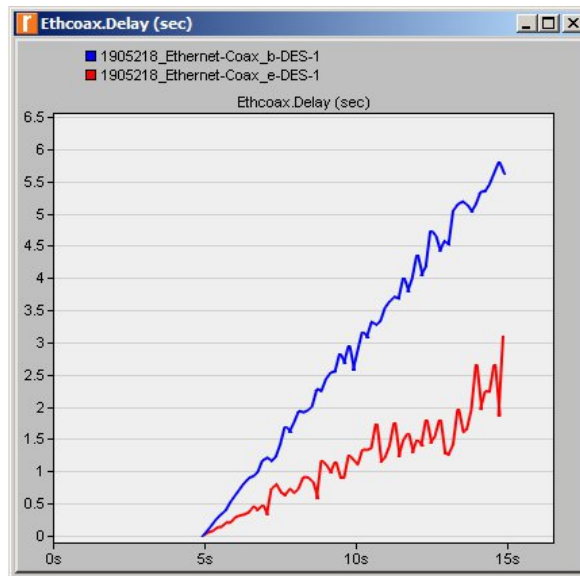


Рисунок 34. Графики для метрики Delay (sec) для сценариев Coax_b и Coax_e

График Coax_b аналогичен предыдущему в сравнении. График Coax_e показывает более низкую задержку, чем Coax_b, но с более выраженными колебаниями. Это может быть связано с тем, что уменьшение размера пакета до 512 байт увеличило количество пакетов, передаваемых по сети, что привело к более частым коллизиям и перепадам.

Вывод: Сеть в Coax_e имеет более низкую задержку, чем Coax_b, но стабильность передачи снижена из-за более частых коллизий.

□ Traffic Sent (bits/sec) & Traffic Received (bits/sec)

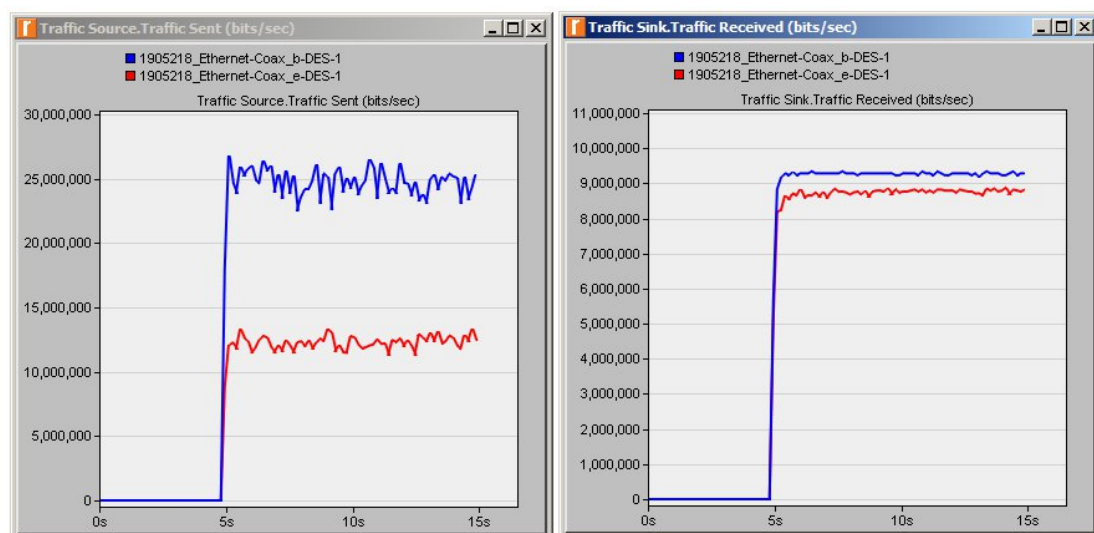


Рисунок 35. Графики для метрик Traffic Sent (bits/sec) и Traffic Received (bits/sec)

Анализ графиков по метрике Traffic Sent (bits/sec). График Coax_b демонстрирует быстрый рост передачи трафика, что может быть связано с высокой частотой отправки пакетов и размером пакета 1024 байта. Беспорядочные перепады указывают на нестабильность сети.

График Соах_е показывает более низкую скорость передачи трафика, чем Соах_б, но с менее выраженными колебаниями. Это может быть связано с тем, что уменьшение размера пакета до 512 байт уменьшило нагрузку на сеть и частоту коллизий.

Сеть в Соах_б активно использует канал, но испытывает перегрузки и коллизии. Сеть в Соах_е имеет более низкую скорость передачи трафика, но стабильность передачи улучшена. Более низкая скорость передачи трафика и менее резкие колебания указывают на то, что сеть испытывает меньшие проблемы с перегрузками.

Анализ графиков по метрике Traffic Received (bits/sec). График Соах_б демонстрирует быстрый рост приема трафика, стабильность графика указывает на эффективное использование канала. График Соах_е показывает чуть более низкую скорость приема трафика, чем Соах_б, но также стабильную. В Соах_е, где пакет уменьшен до 512 байт, скорость приема трафика снижается до примерно 8,600,000 бит/сек по сравнению с Соах_б, где она составляет около 9,300,000 бит/сек.

Это связано с тем, что меньший размер пакета приводит к увеличению количества пакетов, передаваемых по сети, но общий объем переданных данных уменьшается из-за меньшего размера каждого пакета.

Оба графика демонстрируют стабильную передачу трафика, но Соах_е имеет немного меньшую скорость приема из-за уменьшенного размера пакета.

Уменьшение размера пакета может привести к более эффективному использованию канала в плане количества пакетов, но общая пропускная способность снижается.

□ Traffic Sent (packets/sec) & Traffic Received (packets/sec)



Рисунок 36. Графики для метрик Traffic Sent (packets/sec) и Traffic Received (packets/sec)

Анализ графиков по метрике Traffic Sent (packets/sec). Оба графика резко возрастают и примерно одинаковы. Это может показаться неожиданным, поскольку уменьшение размера пакета до 512 байт в Coax_e, что логично, должно было бы привести к увеличению количества пакетов, отправляемых по сети. А теперь вспомним, что сценарий Coax_e есть копия сценария Coax_b без изменения параметра частоты отправки пакетов ($\text{Interarrival Time} = \text{exponential}(0.01)$), поэтому и графики оказались примерно одинаковы.

Вывод: Изменение размера пакета не сильно отразилось на графиках метрики Traffic Sent (packets/sec), различия в графиках небольшие.

Анализ графиков по метрике Traffic Received (packets/sec). График Coax_e показывает более высокую скорость приема пакетов, чем Coax_b. Это ожидаемо, поскольку уменьшение размера пакета до 512 байт должно увеличить частоту приема пакетов, передаваемых по сети.

Вывод: Сеть в Coax_e имеет более высокую частоту приема пакетов из-за меньшего размера пакета.

Уменьшение размера пакета в Coax_e привело к увеличению частоты приема пакетов, что может быть полезно для приложений, требующих высокой частоты обновления данных. Однако это также может увеличить накладные расходы на заголовки пакетов и обработку, что может повлиять на общую эффективность сети.

Выводы

Метод CSMA/CD является основным методом доступа к среде передачи в сетях Ethernet, Fast Ethernet и Gigabit Ethernet. Он обеспечивает обнаружение коллизий и позволяет нескольким устройствам делиться одной средой передачи данных. Принцип работы следующий: Перед отправкой данных узел прослушивает канал. Если канал свободен, узел начинает передачу. Если обнаруживается коллизия, узел отправляет сигнал jam и ждет случайное время перед повторной попыткой.

В ходе работы были смоделированы различные сценарии сети Ethernet (Coax_a, Coax_b, Coax_c, Coax_d, Coax_e) для анализа влияния нагрузки и размера пакета на работу сети.

Были получены навыки в настройке и анализе сетевых моделей, включая изменение размера пакета и удаление узлов для оценки их влияния на сетевую производительность.

Сценарии Coax_a, Coax_b и Coax_c показали, как увеличение нагрузки (уменьшение Interarrival Time) влияет на задержку и коллизии в сети: чем меньше значение Interarrival Time, тем чаще генерируются пакеты, что приводит к росту коллизий (статистика Collision Count) и увеличению потерь пакетов (статистика Traffic Received). Coax_b и Coax_c с меньшим значением интервала Interarrival Time демонстрировали высокую задержку и частые коллизии из-за высокой нагрузки.

Удаление узлов в Coax_d привело к снижению задержки до значения около 2 секунд, но с более резкими колебаниями. Это может быть связано с уменьшением общей нагрузки на сеть и изменениями в распределении трафика.

Сравнение Coax_b и Coax_e показало, что уменьшение размера пакета до 512 байт может снизить задержку и улучшить стабильность сети, но также может увеличить количество пакетов, что может привести к большему количеству коллизий, но конкретно в нашем случае число пакетов не увеличилось из-за заданного Interarrival Time (exponential(0.01) в обоих сценариях).