

Network coding

Сегментация

к.ф.-м.н., м.н.с. Степанов Евгений Павлович

Программа курса

Подходы:

- 1. Управление перегрузкой**
 - Современные протоколы управления перегрузкой TCP
- 2. Демультимплексирование/мультиплексирование**
 - Многопоточные транспортные протоколы
 - Маршрутизация на уровне интернет провайдеров
 - Network Coding
- 3. Сегментация**
 - TCP Proxy
- 4. Балансировка**
 - Балансировка нагрузки и управление трафиком
- 5. Преобразование сообщений**
 - FEC
 - Сжатие

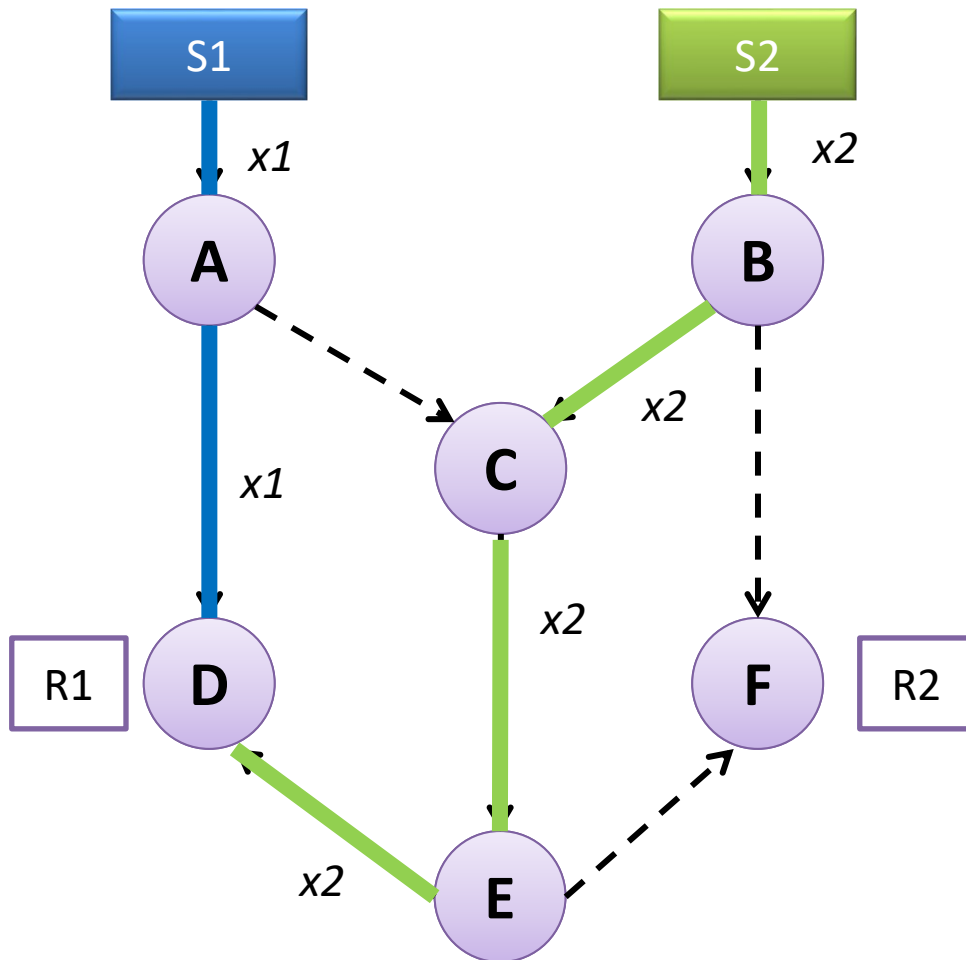
Модели оценки качества сервиса:

- Активные и пассивные измерения
- NS3: моделирование поведения сети с высокой точностью
- Сетевое исчисление: математический подход к качеству сервиса

Примеры:

- Управление сетевыми ресурсами в Центрах Обработки Данных
- Обеспечение качества сервиса в сетях доставки контента
- Пропускная способность по требованию

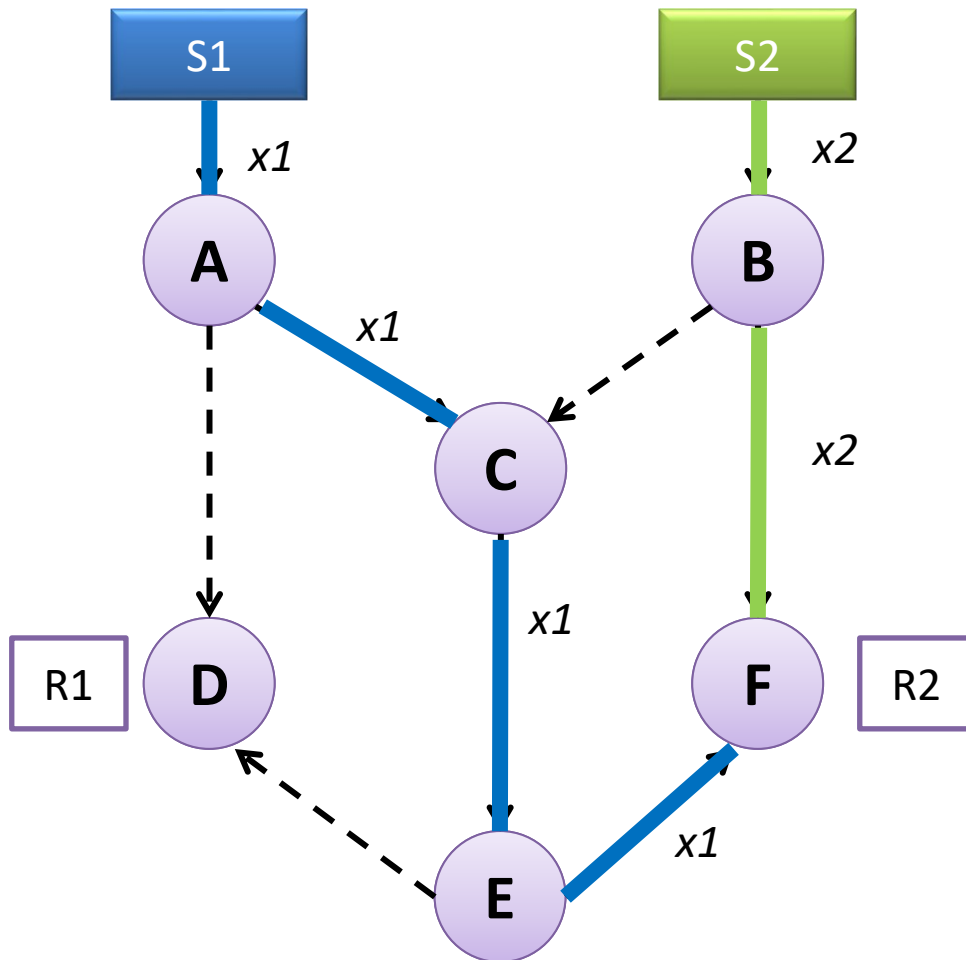
Примеры: butterfly network



- Ориентированный граф
- Источники: $S1$ и $S2$
- Получатели: $R1$ и $R2$
- Слотированное время:
1 бит в единицу времени

Передача до $R1$ –
– без затруднений

Примеры: butterfly network

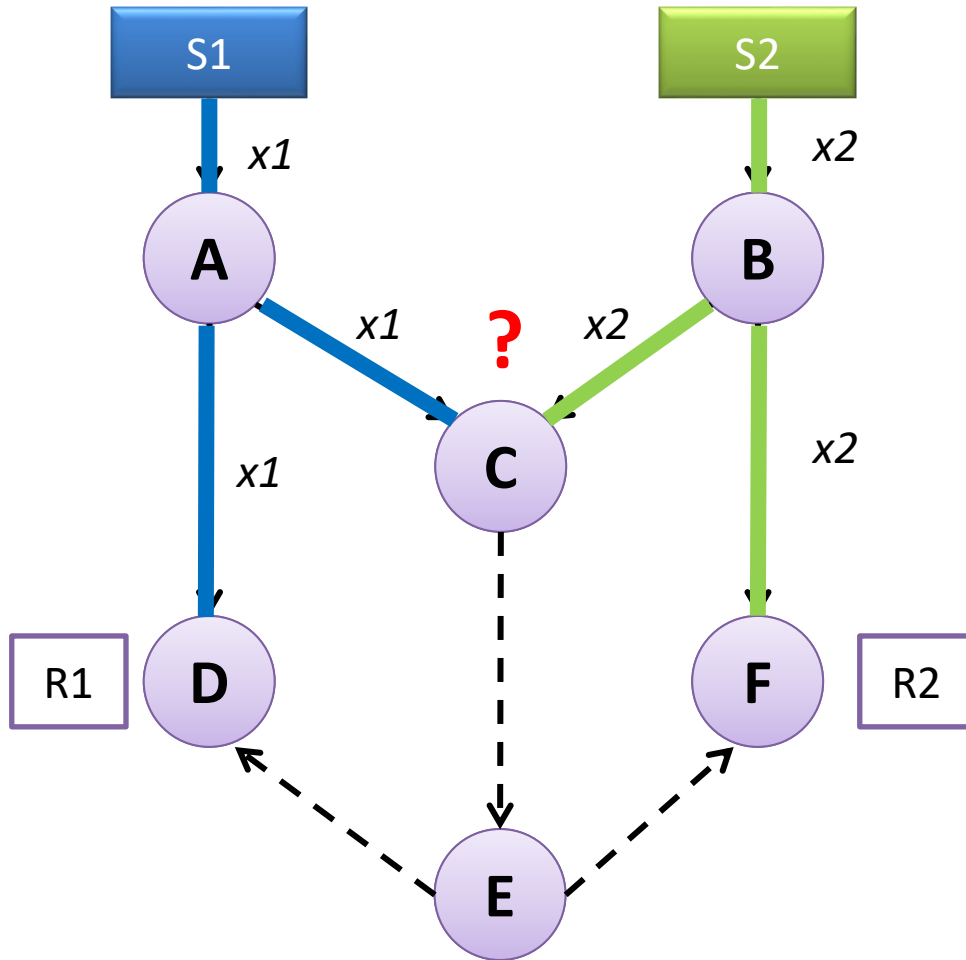


- Ориентированный граф
- Источники: S_1 и S_2
- Получатели: R_1 и R_2
- Слотированное время:
1 бит в единицу времени

Передача до R_1 –
– без затруднений

Передача до R_2 –
– без затруднений

Примеры: butterfly network

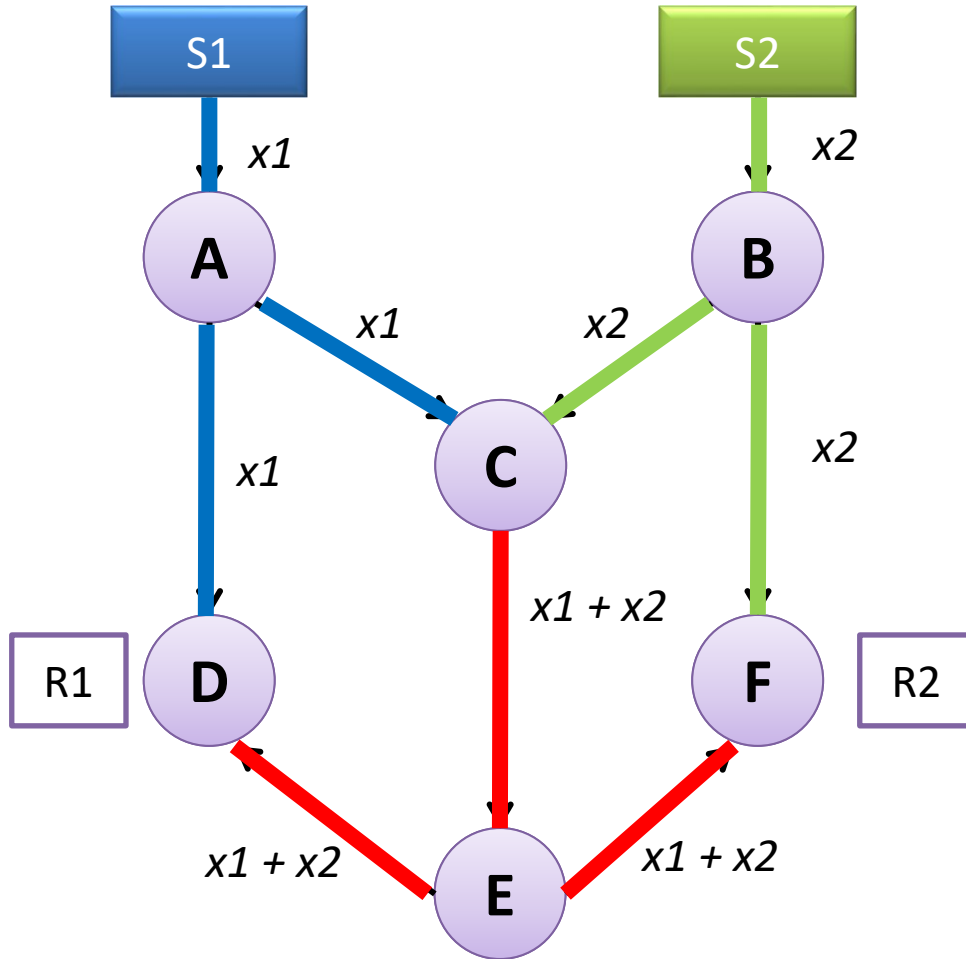


- Ориентированный граф
- Источники: S1 и S2
- Получатели: R1 и R2
- Слотированное время:
1 бит в единицу времени

Multicast - ?

*Традиционный подход:
выбираем x1 или x2 и
отправляем дальше*

Примеры: butterfly network



Network coding:

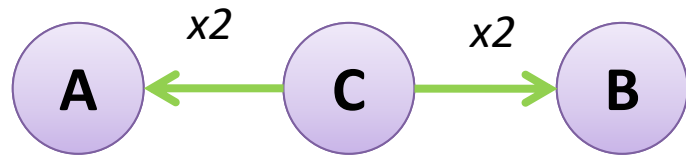
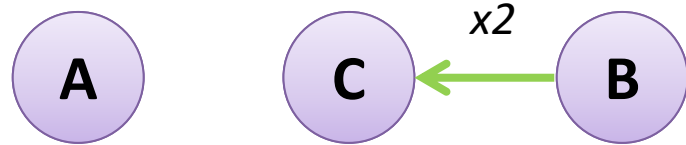
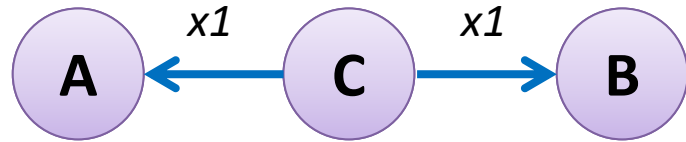
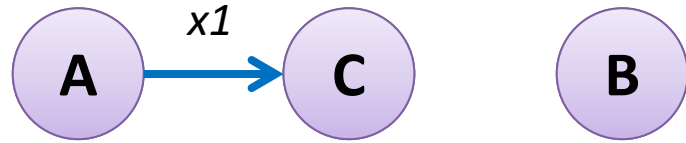
*вычислительная мощность
перестает быть узким местом –
добавим дополнительную
обработку в сеть*

У получателя R_1 : $\{x_1, x_1+x_2\}$

У получателя R_2 : $\{x_2, x_1+x_2\}$

***Увеличили пропускную
способность!***

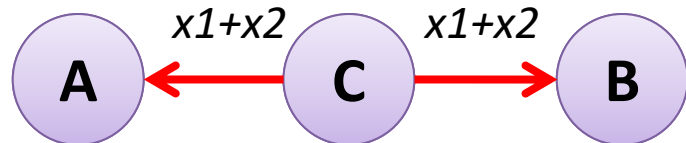
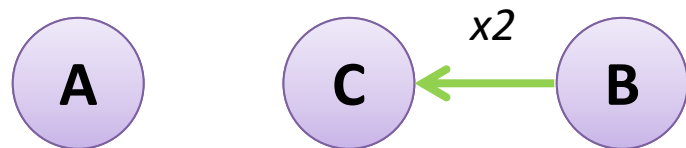
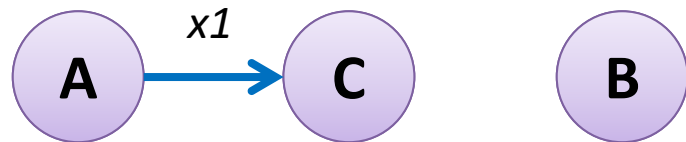
Примеры: беспроводная сеть



Устройство находится либо в режиме передачи данных, либо в режиме приема данных.

Можем ускорить процесс?

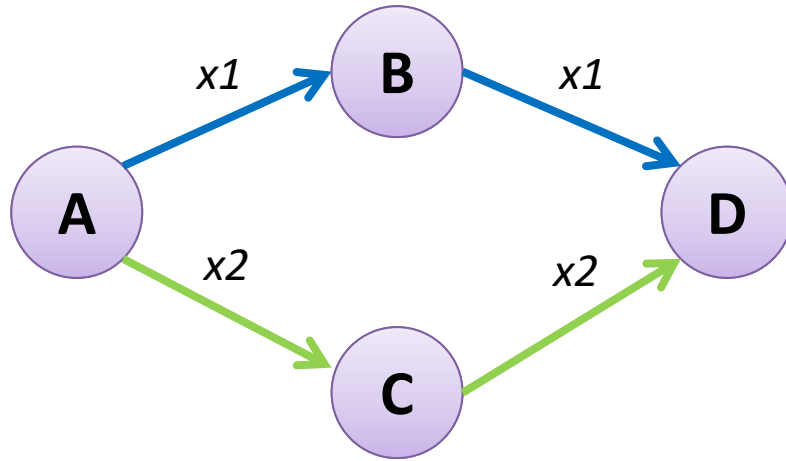
Примеры: беспроводная сеть



Network coding: Да!

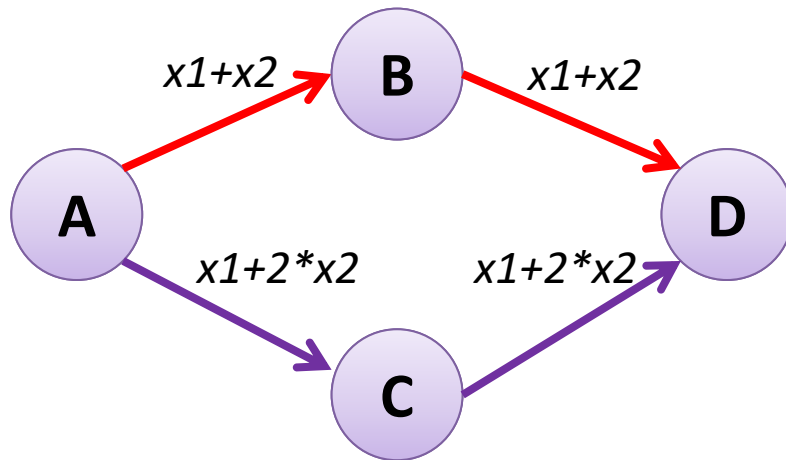
*Увеличили пропускную
способность, повысили
энергоэффективность!*

Примеры: безопасность



*Нарушитель может
подключиться к одной линии
связи*

Повысили безопасность!



Определения

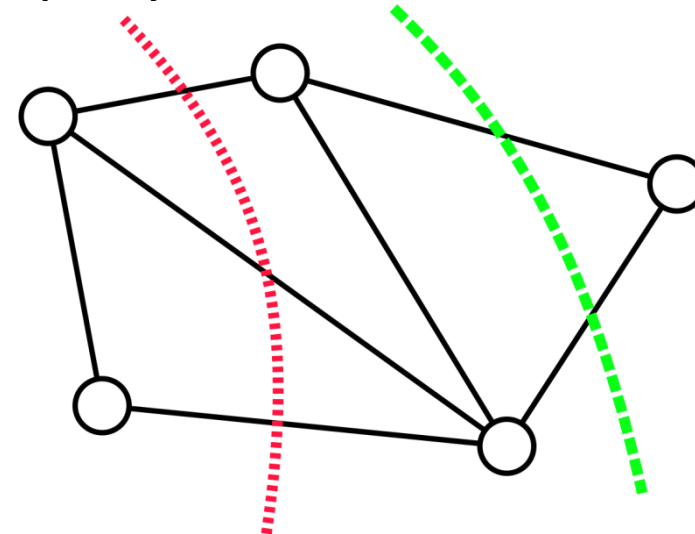
Пусть $G = (V, E)$ – ориентированный граф.

- у каждого ребра графа единичная пропускная способность
- разрешены параллельные рёбра.

Разрезом между S и R ($S, R \subset V$) называется множество рёбер графа, чьё удаление из графа разъединяет S от R .

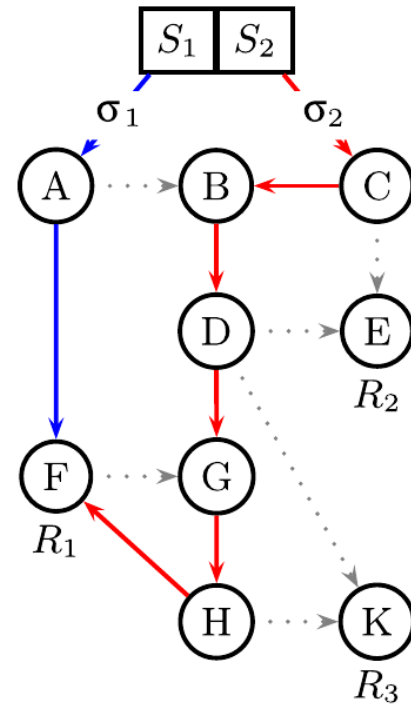
Значением разреза является сумма пропускных способностей рёбер разреза.

Минимальный разрез – разрез с минимальным значением.



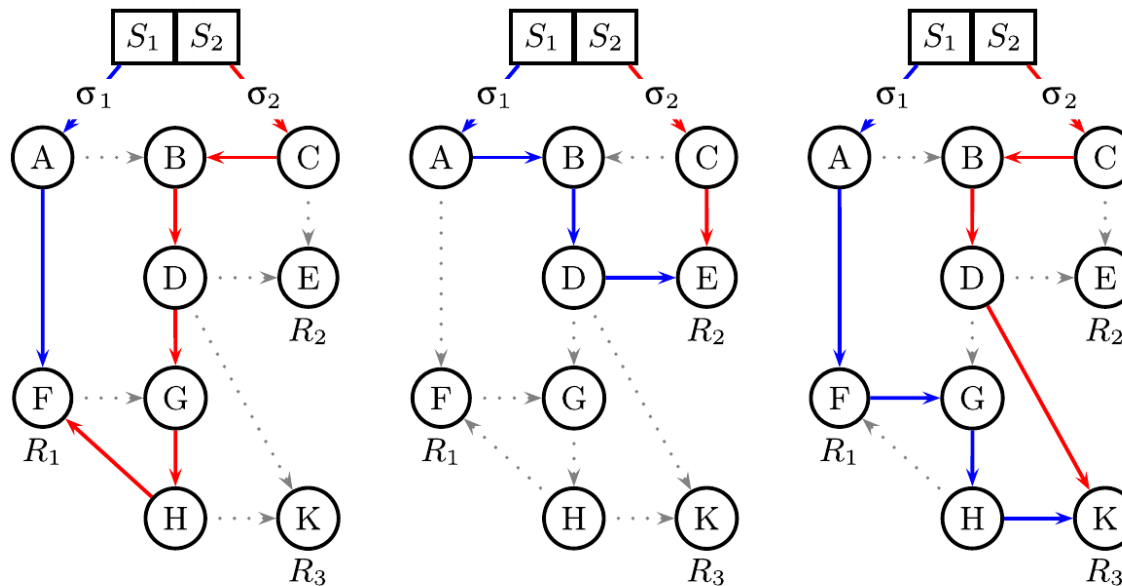
Основная теорема мультикаста

Пусть $G = (V, E)$ – ориентированный ациклический граф с рёбрами единичной пропускной способности, h источниками, расположенными на одной вершине графа и N получателями.



Основная теорема мультикаста

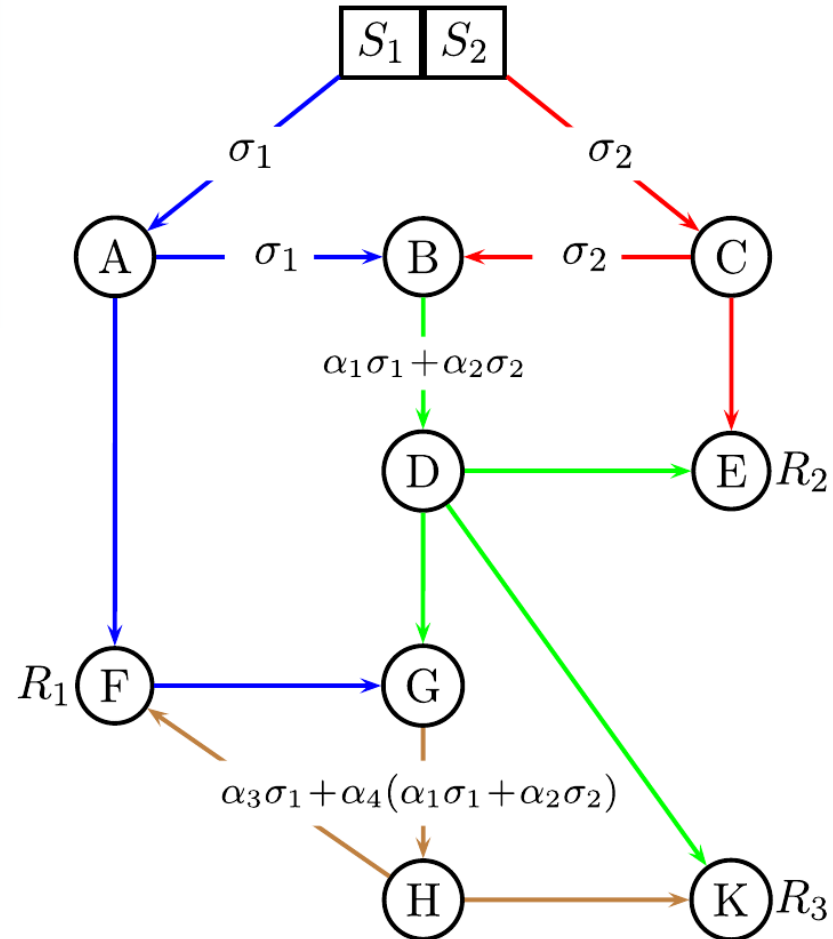
Пусть $G = (V, E)$ – ориентированный ациклический граф с рёбрами единичной пропускной способности, h источниками, расположенными на одной вершине графа и N получателями. Предположим, что значение минимального разреза до каждого получателя равно h .



Основная теорема мультикаста

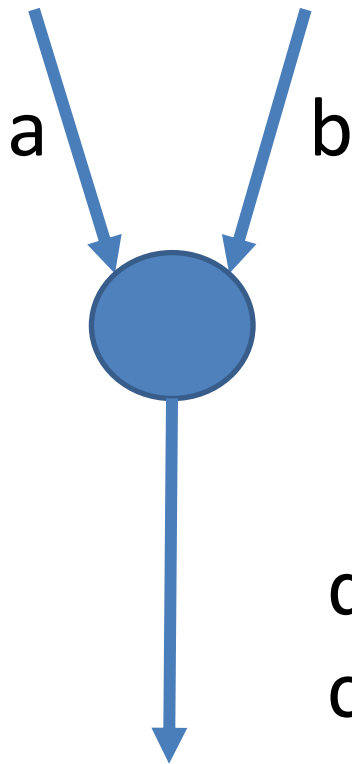
Пусть $G = (V, E)$ – ориентированный ациклический граф с рёбрами единичной пропускной способности, h источниками, расположенными на одной вершине графа и N получателями. Предположим, что значение минимального разреза до каждого получателя равно h . Тогда существует схема передачи над достаточно большим конечным полем F_q (в которой промежуточные сетевые узлы линейно комбинируют приходящие символы над F_q), которая доставляет информацию от источников одновременно всем получателям со скоростью h .

Основная теорема мультикаста



Локальный кодирующий вектор

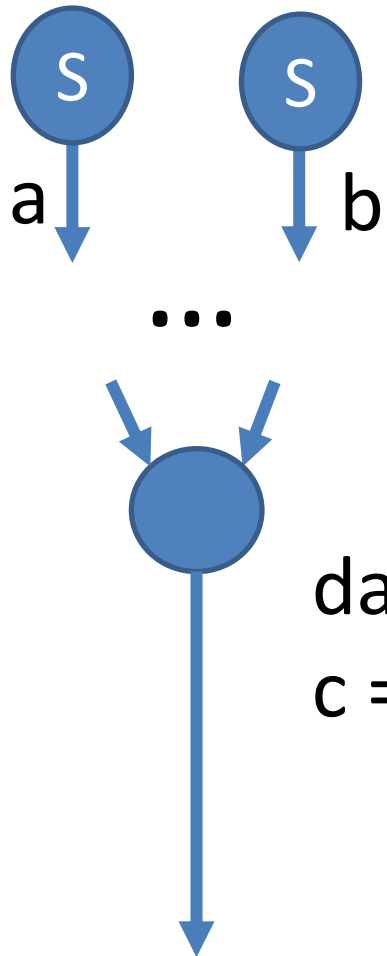
Локальным кодирующим вектором $c_{local}(e)$ ребра e является вектор коэффициентов над полем F_q которые используются для умножения входящих в ребро e СИМВОЛОВ.



$$\text{data} = \alpha_1 a + \alpha_2 b$$

$$c_{local} = (\alpha_1, \alpha_2)$$

Глобальный кодирующий вектор



(Глобальным) кодирующим вектором $c(e)$ ребра e является вектор коэффициентов, с которыми символы источников передаются (в линейной комбинации) через ребро e .

$$\text{data} = \alpha_1 a + \alpha_2 b$$

$$c = (\alpha_1, \alpha_2)$$

Матрица A_j на i -ой строке содержит кодирующий вектор последней дуги в пути (S_i, R_j) .

Система уравнений получателя

Пусть ρ_i^j – символ, проходящий через последнее ребро пути (S_i, R_j) . Тогда для получения исходных символов получатель R_j должен решить уравнение следующего вида:

$$\begin{bmatrix} \rho_1^j \\ \dots \\ \rho_h^j \end{bmatrix} = A_j \begin{bmatrix} \sigma_1 \\ \dots \\ \sigma_h \end{bmatrix}$$

Чтобы все получатели могли решить свои системы уравнений, у всех матриц A_j должен быть полный ранг.

Основная теорема мультикаста

Алгебраическая формулировка: В случае линейного сетевого кодирования существуют значения для компонентов $\{\alpha_k\}$ локальных кодирующих векторов над достаточно большим конечным полем F_q такие что все матрицы \mathbf{A}_j , $1 \leq j \leq N$, определяющие информацию, которая приходит получателю, имеют полный ранг.

Ограничения сетевого кодирования

Необязательные:

- Рёбра единичной пропускной способности;
- Расположение источников в одном узле;
- Нулевая задержка;
- Ацикличность графа;

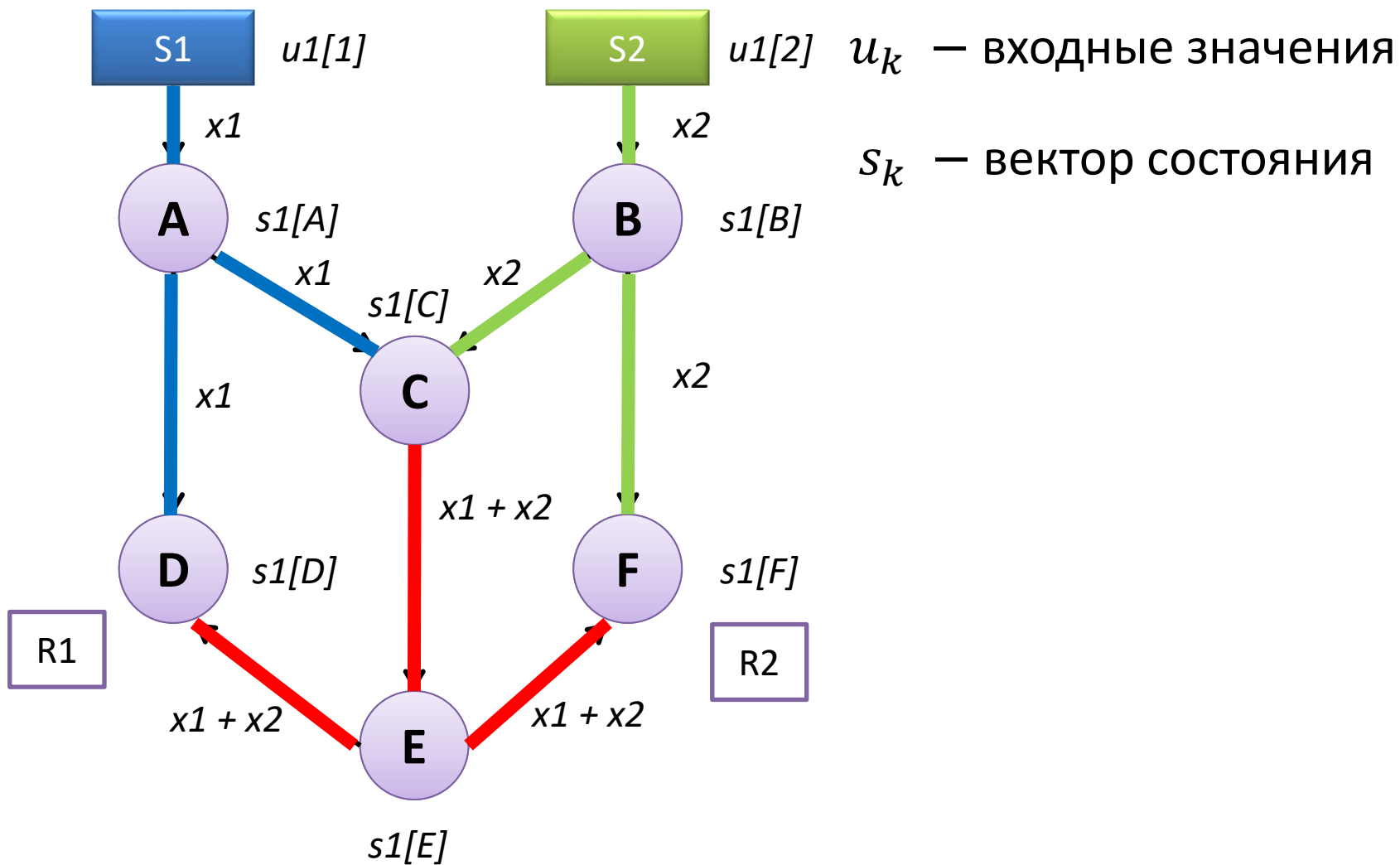
Обязательные:

- Ориентированность графа;
- Одинаковые значения минимальных разрезов.

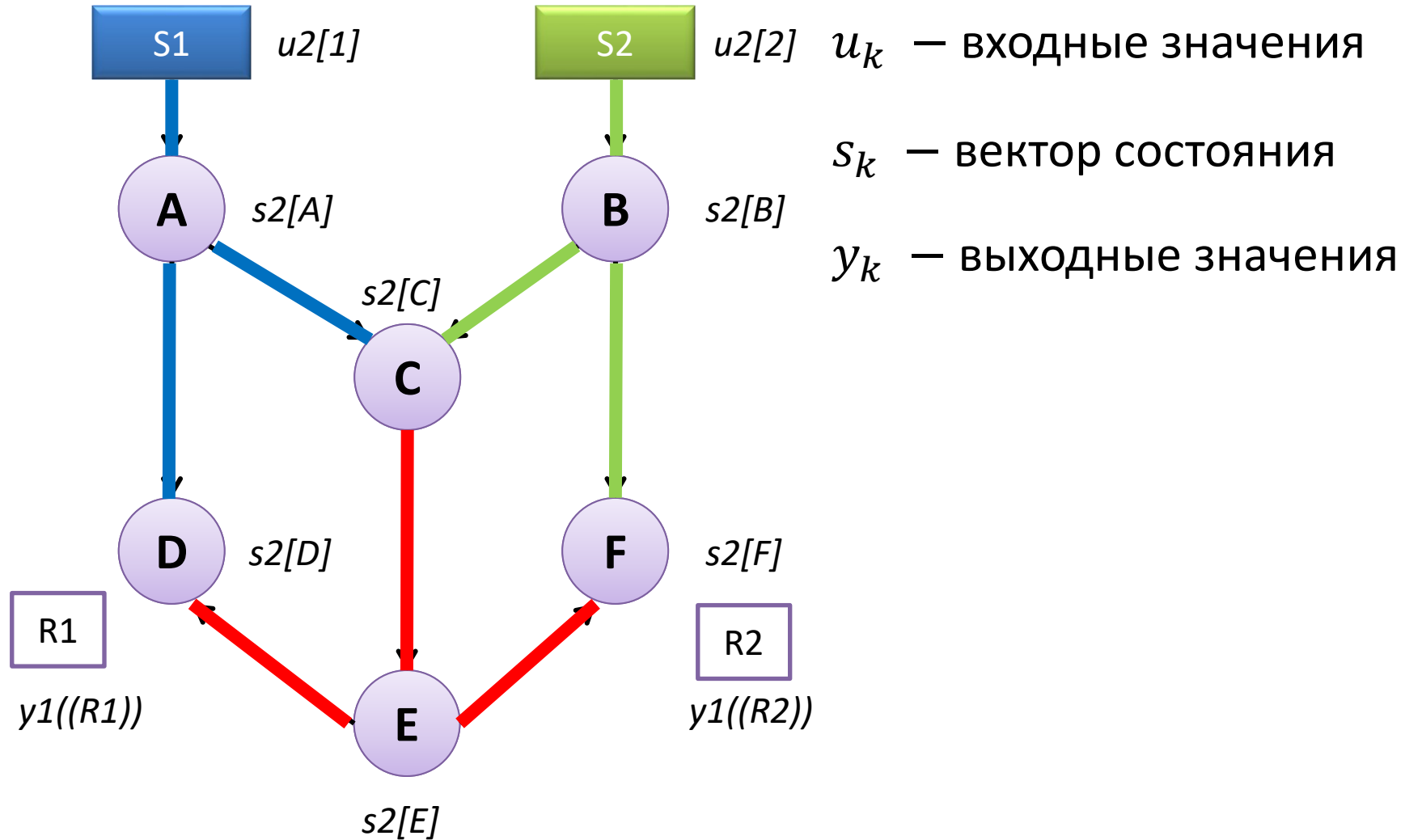
Основные вопросы

- Какой размер алфавита необходим для реализации network coding?
- В каком месте проводить операции network coding?
- С какой скоростью отправители могут генерировать данные для передачи?
- Какое преимущество у сетевого кодирования?

Алгебраическая модель



Алгебраическая модель



Алгебраическая модель

Пространство состояний

$$\begin{cases} s_{k+1} = As_k + Bu_k, \\ y_k = C_j s_k + D_j u_k \end{cases}$$

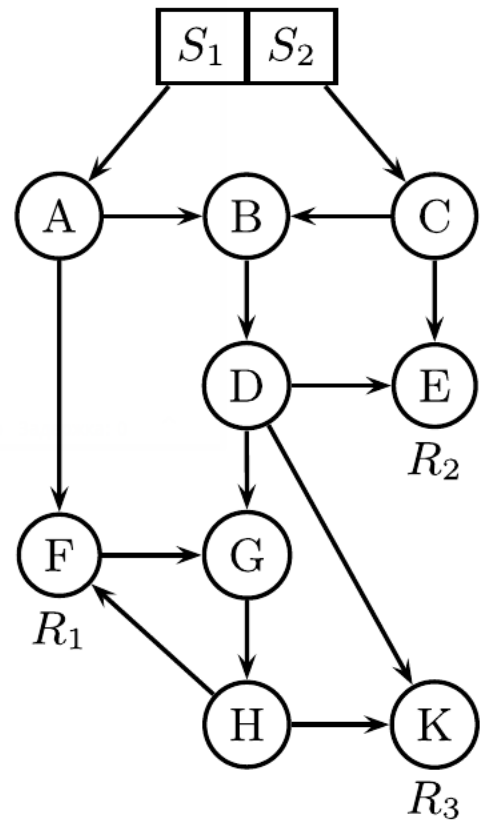


$$A_j = D_j + C_j(I - A)^{-1}B$$

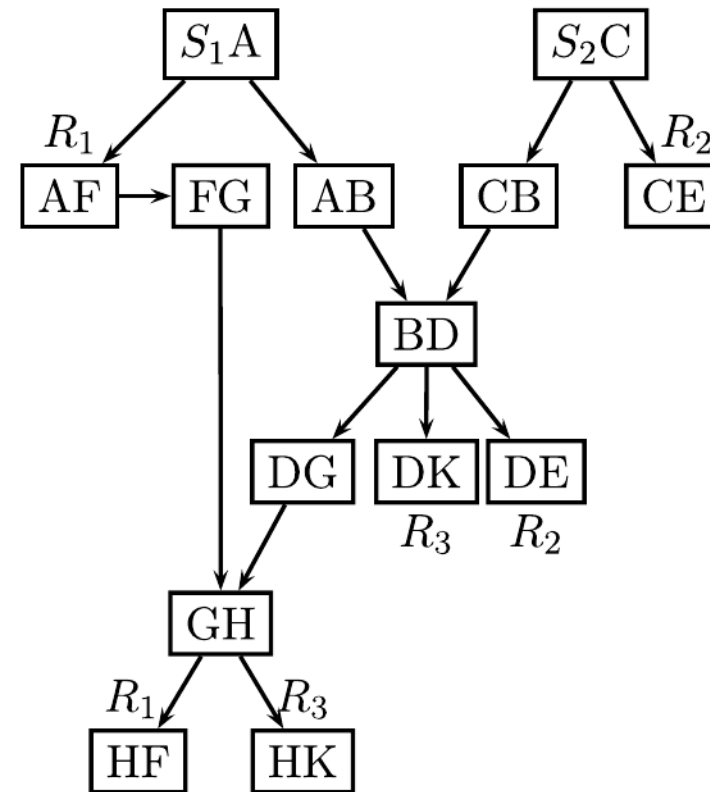
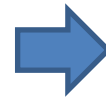
- u_k — входные значения
- y_k — выходные значения
- s_k — вектор состояния
- A, B, C_j, D_j — матрицы связей (топология сети, связи со входами и выходами)
- $A = \{\alpha\}_{ij}$

Теорема: для мультикаст-сети алфавит размера $q > N$ всегда является достаточным

Комбинаторная модель

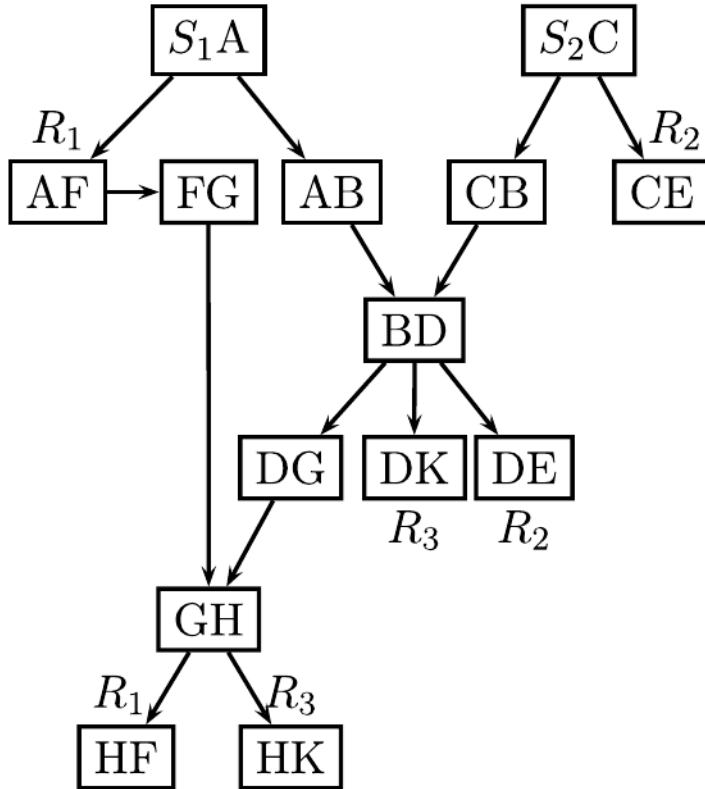


Граф сети

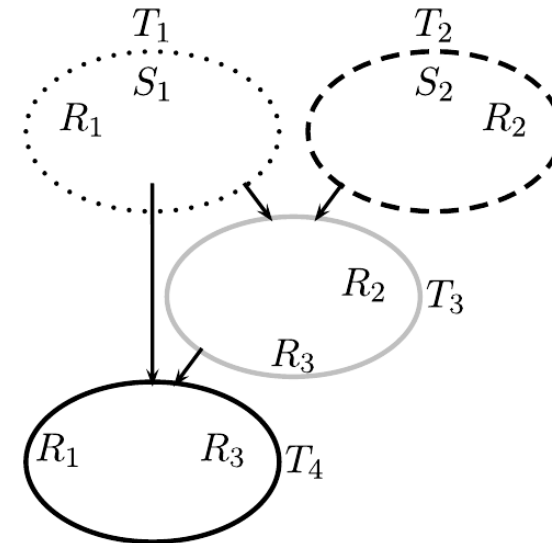


Реберный граф сети

Комбинаторная модель



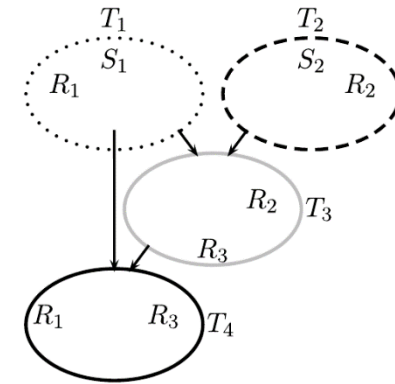
Реберный граф сети



Граф поддеревьев

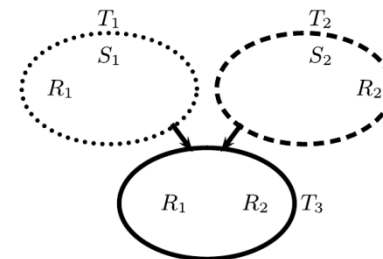
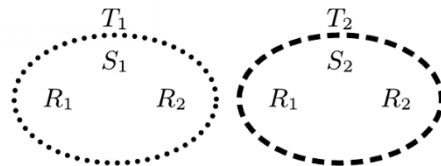
Комбинаторная модель

- Граф поддеревьев с мультикаст св-вом называется *минимальным*, если удаление любого ребра приводит к нарушению мультикаст св-ва
- Минимальные графы поддеревьев обладают рядом свойств
- Перечисление минимальных графов



Граф поддеревьев

Лемма: для сети с двумя источниками и двумя получателями существует ровно два минимальных графа поддеревьев



Информационно-теоретическая модель

- Отправитель генерирует B сообщений со скоростью ω_S
- Дискретное время
- На отправителе $|Out(S)|$ функций $\{f_i^S\}$
- На вершине v $|Out(v)|$ функций f_i^v

$$f_i^S: 2^{n\omega_S} \rightarrow 2^n$$

$$f_i^v: 2^{n|In(v)|} \rightarrow 2^n$$



Получатель принимает со скоростью $\frac{n\omega_S B}{n(B+|V|)} \rightarrow \omega_S$ при $B \gg |V|$

- Результат верен, если можно различить получаемые сообщения
- Рассчитываем вероятность потери сообщения

$$P = 2^{-n(|V| - \omega_S)}$$

$$\omega_S < m = \min_v |vS|$$

Задача линейного программирования

Max-Flow LP:

maximize f_{RS}

subject to

$$\sum_{(v,u) \in E} f_{vu} = \sum_{(u,w) \in E} f_{uw}, \quad \forall u \in V \quad (\text{flow conservation})$$

$$f_{vu} \leq c_{vu}, \quad \forall (v,u) \in E \quad (\text{capacity constraints})$$

$$f_{vu} \geq 0, \quad \forall (v,u) \in E$$

Задача линейного программирования

Network Coding LP:

maximize χ

subject to

$$f_{R_i S}^i \geq \chi, \quad \forall i$$

$$\sum_{(v,u) \in E} f_{vu}^i = \sum_{(u,w) \in E} f_{uw}^i, \quad \forall u \in V, \quad \forall i \quad (\text{flow conservation})$$

$$f_{vu}^i \leq f_{vu}, \quad \forall (v,u) \in E \quad (\text{conceptual flow constraints})$$

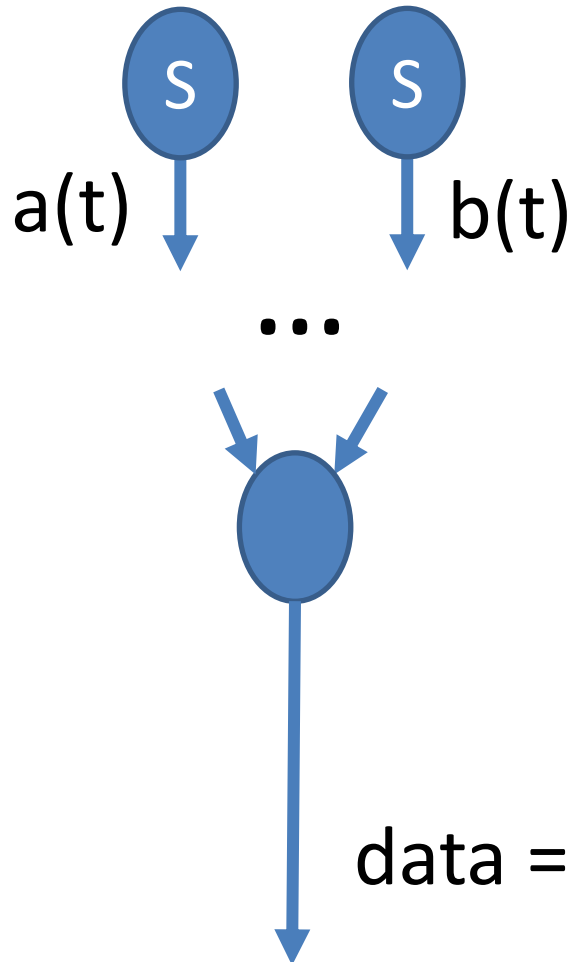
$$f_{vu} \leq c_{vu}, \quad \forall (v,u) \in E \quad (\text{capacity constraints})$$

$$f_{vu}^i \geq 0 \text{ and } f_{vu} \geq 0, \quad \forall (v,u) \in E, \quad \forall i$$

Построение сетевого кода

- Найти h непересекающихся маршрутов до каждого получателя, найти множество кодирующих точек и построить минимальную конфигурацию.
- Жадным алгоритмом ищем кодирующие вектора

Сети с задержками: проблема



В узел сети приходят символы, посланные источниками в разные моменты времени.

Узлы составляют линейную комбинацию и посылают далее.

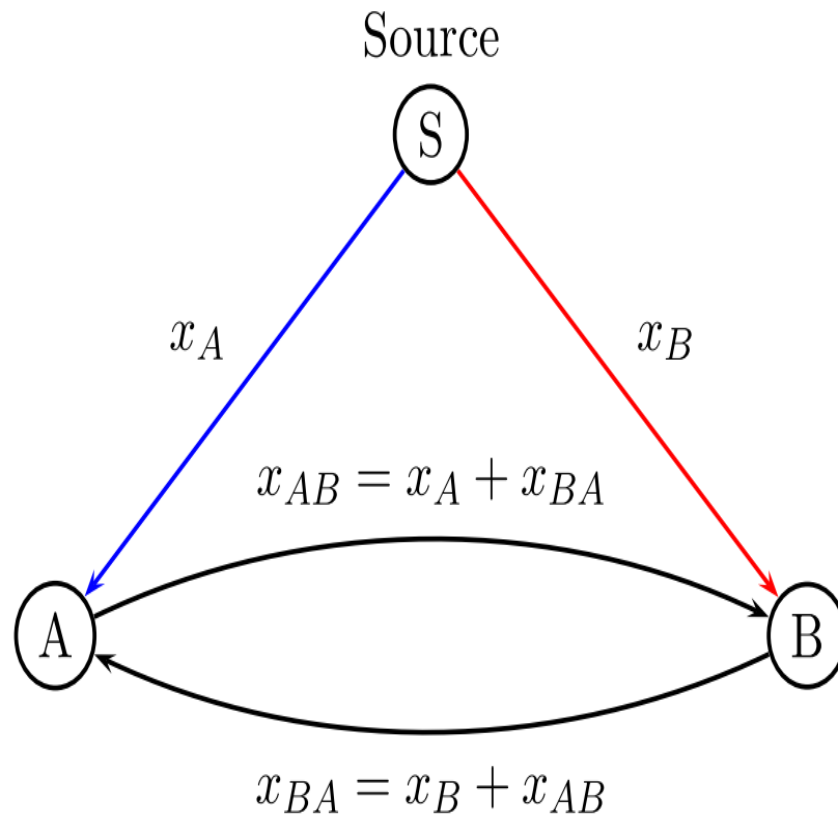
Получателю приходят линейные комбинации из данных, посланных в разное время.

$$\text{data} = \alpha_1 a(t_i) + \alpha_2 b(t_j)$$

Сети с задержками: подходы

1. Добавление к каждому пакету метки поколения.
Промежуточный узел производит кодирование только при получении всех пакетов поколения. Хорошо подходит для сетей со значительными различиями в задержке между путями.
2. Разбиение времени на слоты и использование оператора задержки D. Можно кодировать все приходящие символы вне зависимости от их поколения, а затем решать на стороне получателя систему уравнений с операторами задержки.

Сети с циклами: проблема



$$x_{AB} = x_A + x_{BA} = x_A + x_B + x_{AB} \Rightarrow$$
$$x_A + x_B = 0$$

Сети с циклами: подходы

Необходимо ввести задержки, чтобы не было циклов с нулевой задержкой. Затем используются следующие подходы:

1. Утверждается, что сети с циклами можно рассматривать в качестве свёрточных кодов, для которых применимы специальные алгоритмы декодирования (например, алгоритм Витерби).
2. Можно попытаться удалить циклы. Возможно только в случае простых циклов (которые не имеют общих рёбер).

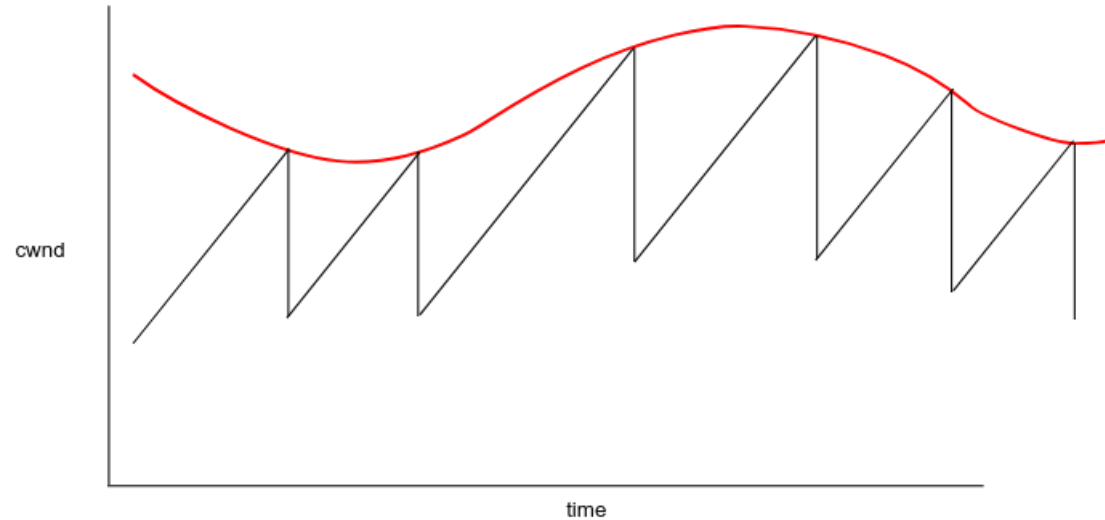
Проблемы сетевого кодирования

- Много теории, практически нет случаев успешного применения;
- Во многих работах предполагается, что узлы сети кооперируются, а не пытаются послать как можно больше;
- Испытания в реальных сетях часто не показывали преимуществ кодирования.

Проксирование как метод улучшения качества сервиса

Управление перегрузкой

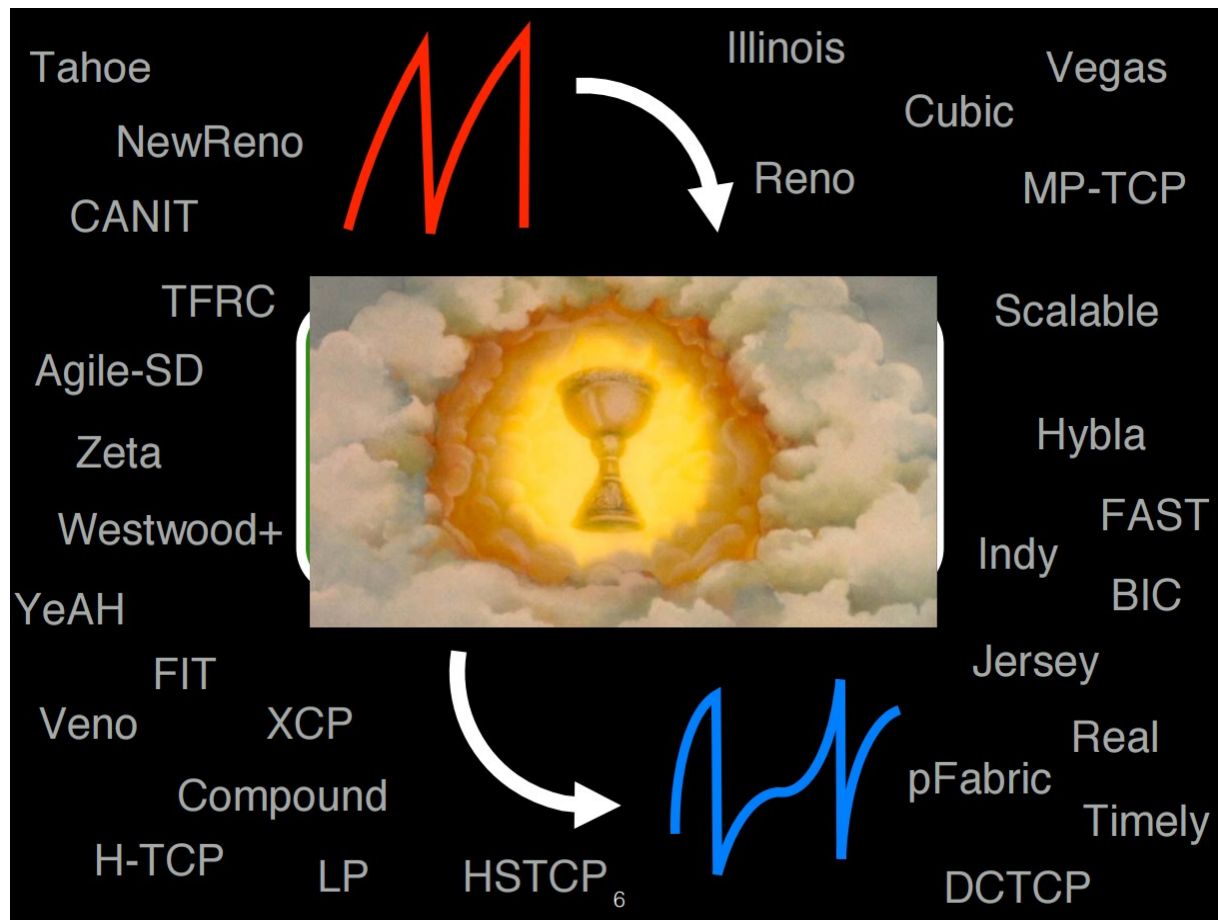
- Определяем интенсивность отправки пакетов в сеть
- Пытаемся достичь максимальной доступной пропускной способности, не создавая при этом перегрузок



TCP Sawtooth, red curve represents the network capacity

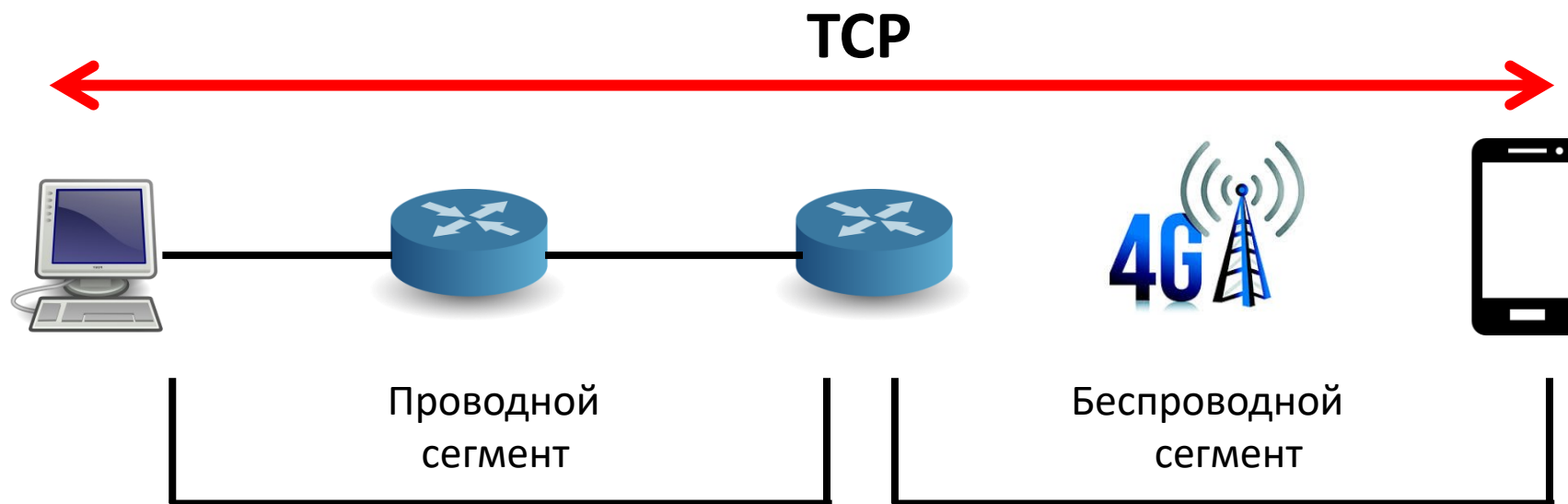
Многообразиие алгоритмов

Работа алгоритма управления перегрузкой зависит от:



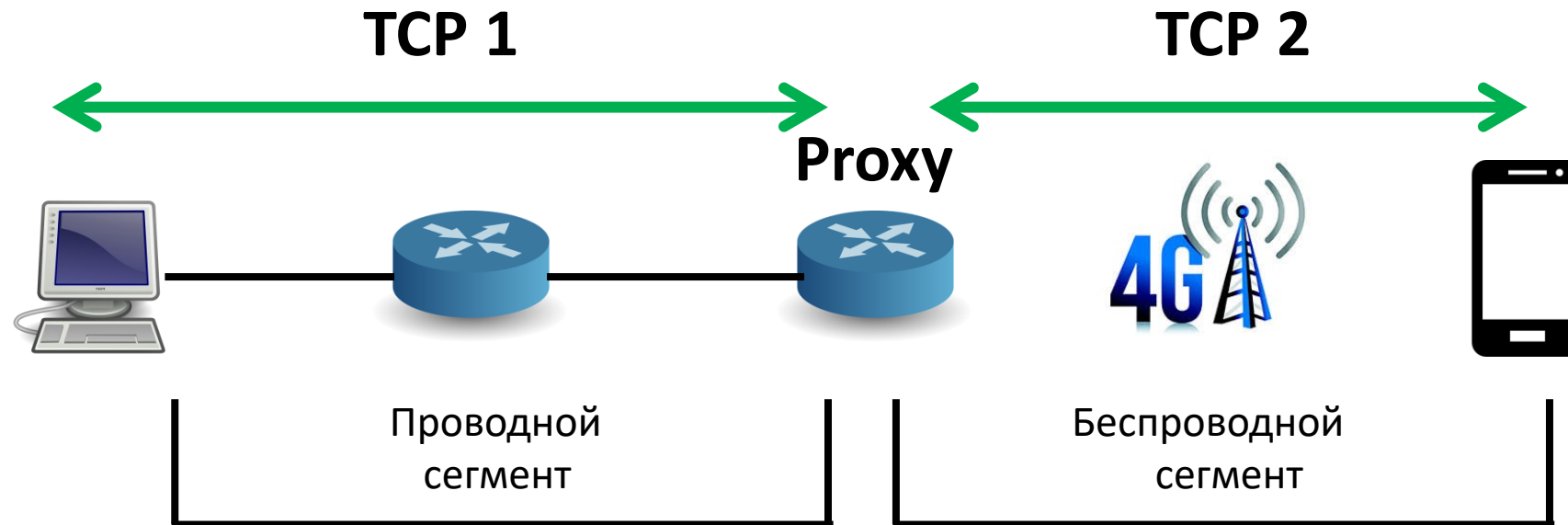
- Задержки
- Надежности канала
- Нагрузки в сети
- Возможностей оборудования
- Политики очередизации

Разнородная среда передачи данных



Универсальный алгоритм будет проигрывать
специализированным

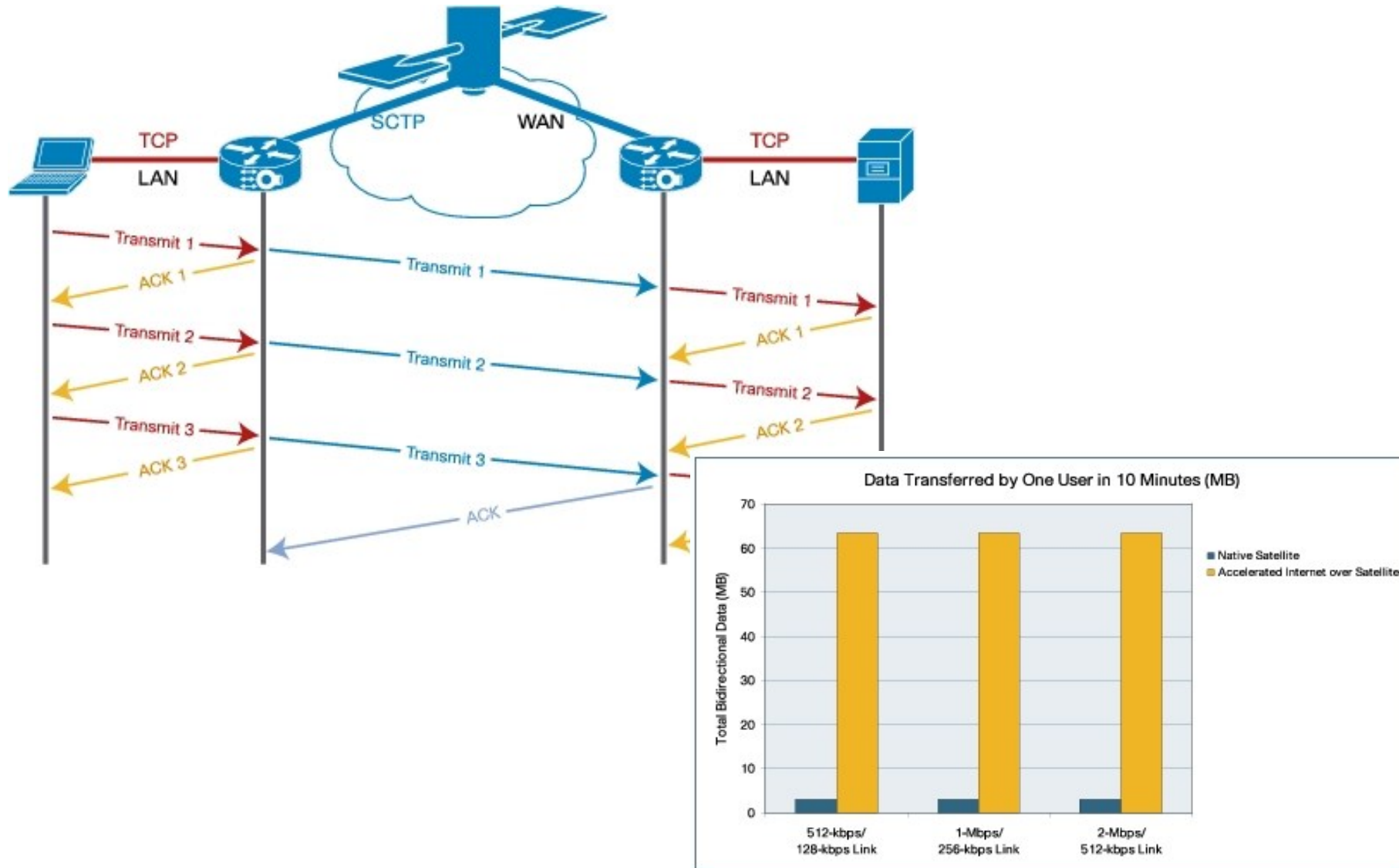
Split TCP



- Увеличение скорости реакции на изменение ситуации в сети
- Уменьшение задержки
- Увеличение скорости передачи

- Proxy (POP)

Cisco NCE



Split TCP efficiency model

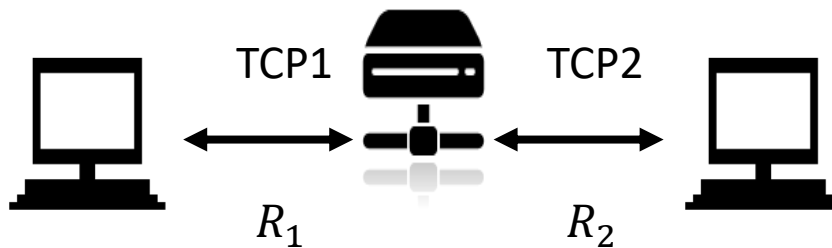
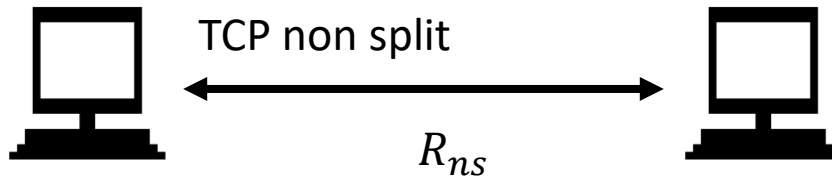
- R_{ns} — non-split connection rate
- R_{sp} — split connection rate
- R_1 — TCP1 rate
- R_2 — TCP2 rate
- T — non-split connection RTT
- q — non-split connection packet loss probability
- T_2 — TCP2 RTT
- q_2 — TCP2 packet loss probability

$$R_{ns} = \frac{1}{T} \sqrt{\frac{3}{2q}} \quad (1)$$

$$\frac{R_{sp}}{R_{ns}} = \frac{T}{T_2} > 1 \quad (4)$$

$$R_2 = \frac{1}{T_2} \sqrt{\frac{3}{2q_2}} \quad (2)$$

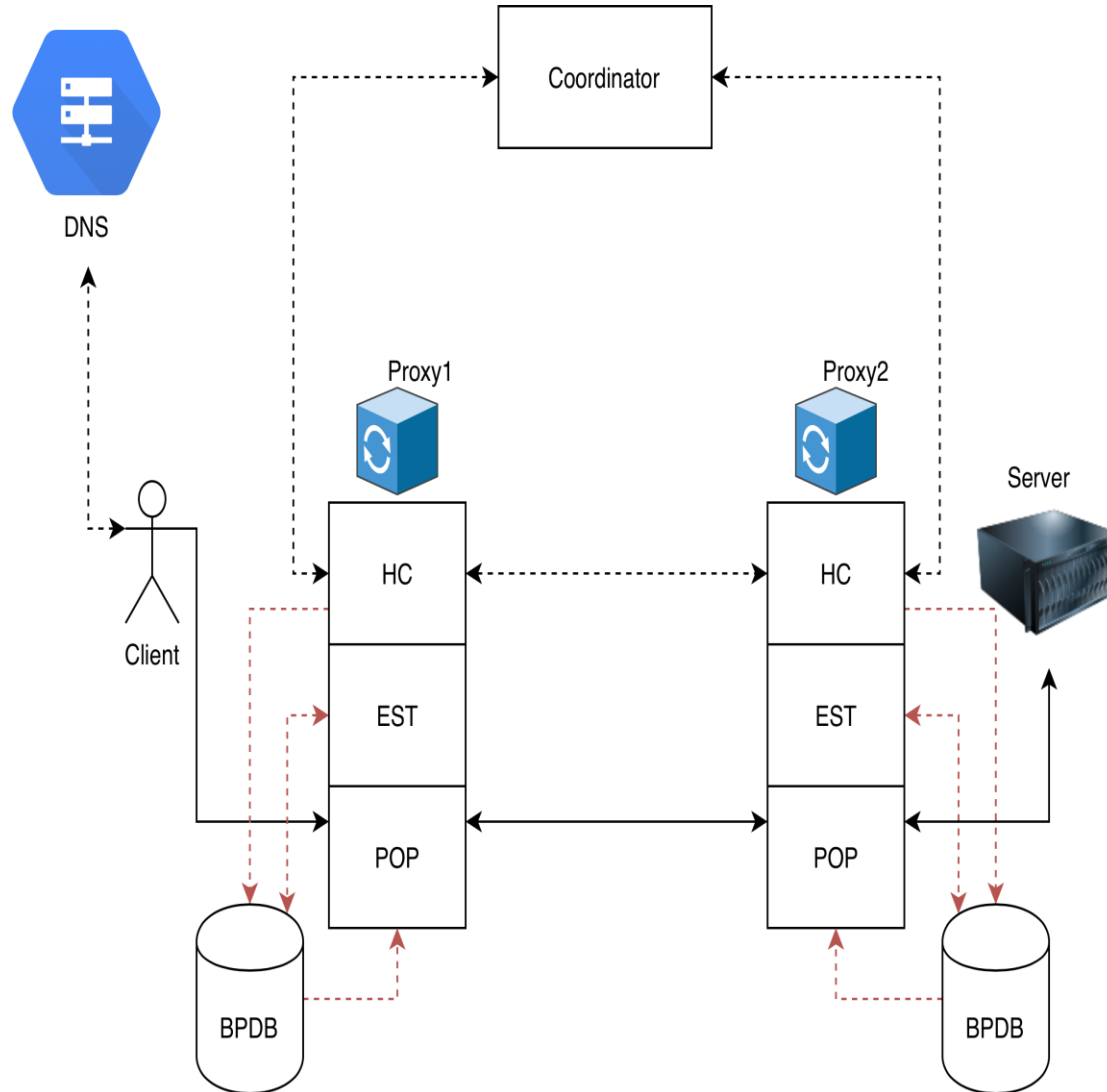
$$\frac{R_{sp}}{R_{ns}} = \frac{T}{T_2} \sqrt{\frac{q}{q_2}} \quad (3)$$



Model restrictions

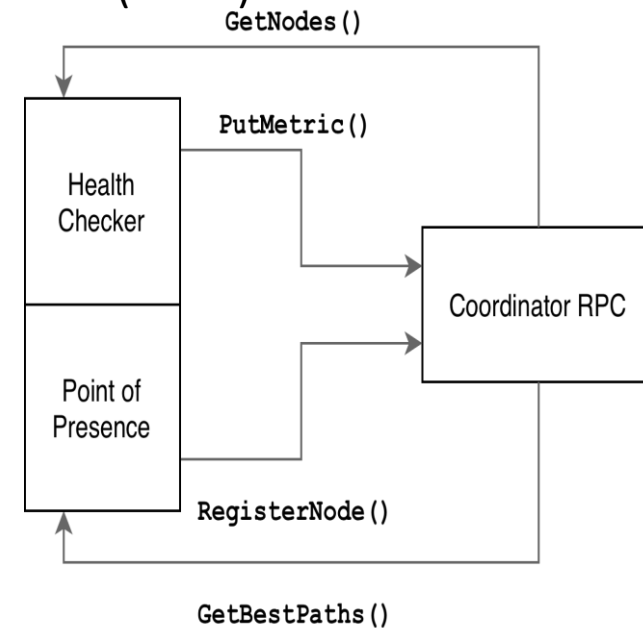
1. The average rate estimation is derived for TCP Reno.
2. The connection is network limited.
3. The obtained estimations are not suitable for short-lived connections.
4. The model doesn't take into account proxy overhead.

Dynamic Split TCP



Components:

1. DNS server (DNS)
2. Coordinator
3. Health-Checker (HC)
4. Point of presence (POP)
5. Local path estimator (EST)
6. Best Paths DataBase (BPDB)



Нерешенные вопросы

- подбор оптимальной конфигурации в параметрах агентов сегментированного соединения;
- количество и размещение прокси-серверов, реализующих сегментированный подход.

Программа курса

Подходы:

- 1. Управление перегрузкой**
 - Современные протоколы управления перегрузкой TCP
- 2. Демультимплексирование/мультиплексирование**
 - Многопоточные транспортные протоколы
 - Маршрутизация на уровне интернет провайдеров
 - Network Coding
- 3. Сегментация**
 - TCP Proxy
- 4. Балансировка**
 - Балансировка нагрузки и управление трафиком
- 5. Преобразование сообщений**
 - FEC
 - Сжатие

Модели оценки качества сервиса:

- Активные и пассивные измерения
- NS3: моделирование поведения сети с высокой точностью
- Сетевое исчисление: математический подход к качеству сервиса

Примеры:

- Управление сетевыми ресурсами в Центрах Обработки Данных
- Обеспечение качества сервиса в сетях доставки контента
- Пропускная способность по требованию