Интернет: перегрузка

Введение в компьютерные сети

чл.-корр. РАН Смелянский Р.Л.

Кафедра АСВК

План

- Что такое перегрузка?
- Необходимость обратной связи у отправителя
- Почему возникает перегрузка и как ею можно управлять
- Где располагать управление перегрузкой: Основные подходы
  - В сети
  - Со стороны получателя
- Управление перегрузкой в TCP
  - TCP Tahoe
  - TCP Reno
  - TCP RTT измерение
  - Управление производительностью на практике

Где и как управлять перегрузкой?

Введение в компьютерные сети

чл.-корр. РАН Смелянский Р.Л.

Кафедра АСВК ф-т ВМК МГУ

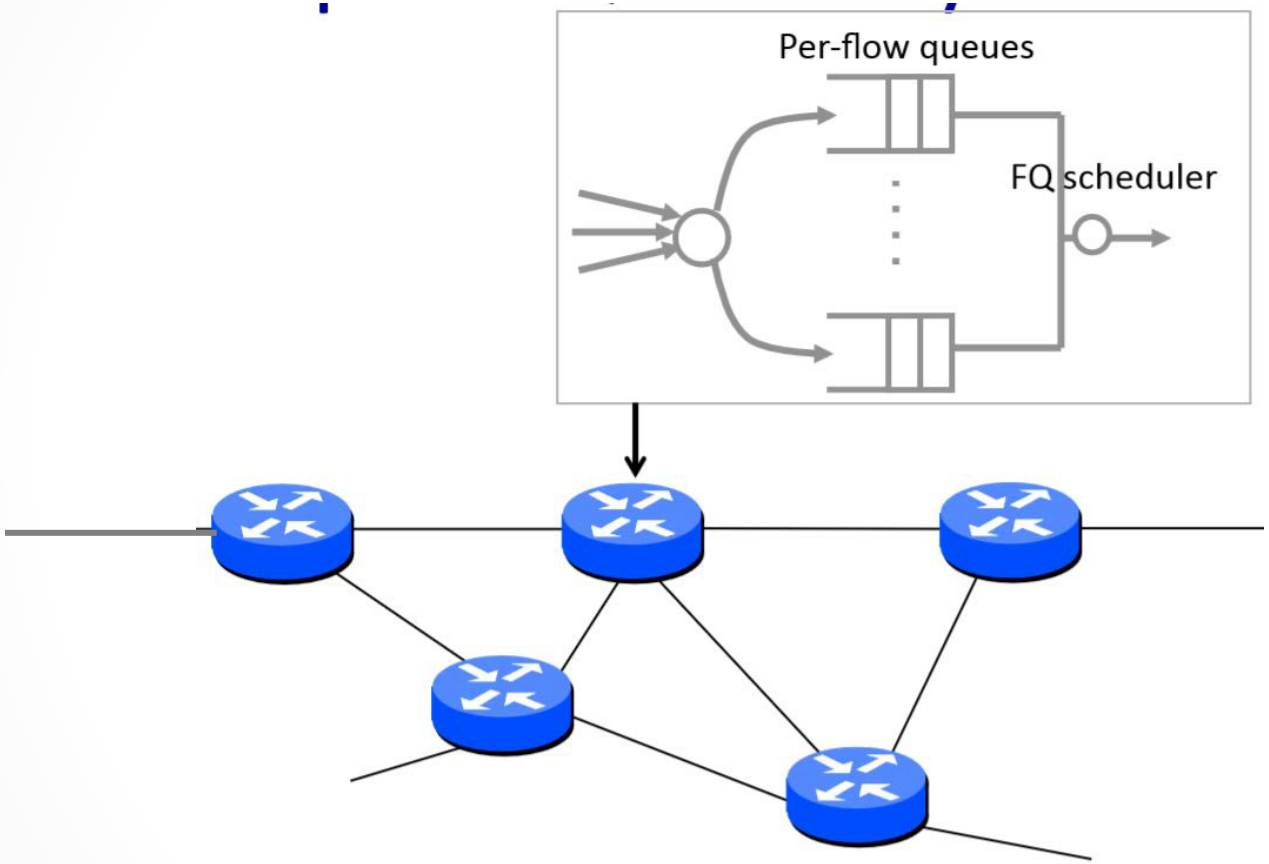
План

Мы хотим построить алгоритм управления перегрузками такой, чтобы:

1. Высокая пропускная способность: каналы загружены, скорость потоков высокая
2. Распределение пропускной способности каналов в соответствии с max-min справедливостью
3. Быстрая реакция на изменения состояния сети
4. Распределенное управление

- Где надо размещать управление перегрузкой:
  - В сети каждом маршрутизаторе (WFQ)
  - На каждом хосте
- Скользящее окно и AIMD

Пример: FQ на каждом маршрутизаторе



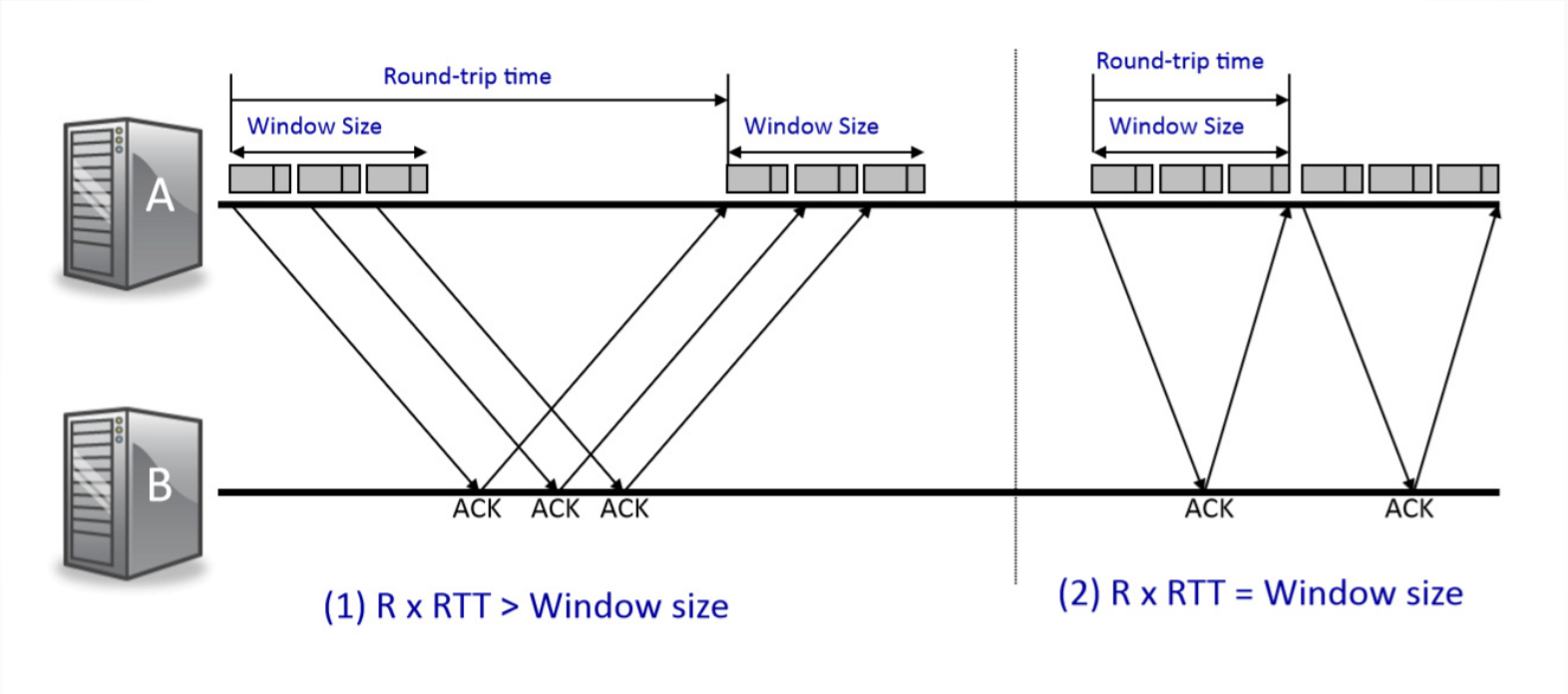
Управление перегрузками в TCP

(управление на хостах)

В TCP управление перегрузкой размещается на конечном хосте

- Реакция на события, наблюдаемые на конечном хосте (например, потеря пакета).
- Использование скользящего окна, предназначенного для управления потоком в TCP
- Постараться оценить сколько пакетов можно безопасно отправить в сеть одновременно.

Скользящее окно в TCP



Скользящее окно перегрузки в TCP

TCP варьирует число пакетов отправляемых в сеть, изменяя размер скользящего окна:

размер окна=min{ Объявленное окно, Окно перегрузки }

получатель — отправитель (cwnd)

Как определить размер cwnd?

AIMD управление перегрузкой для одного потока

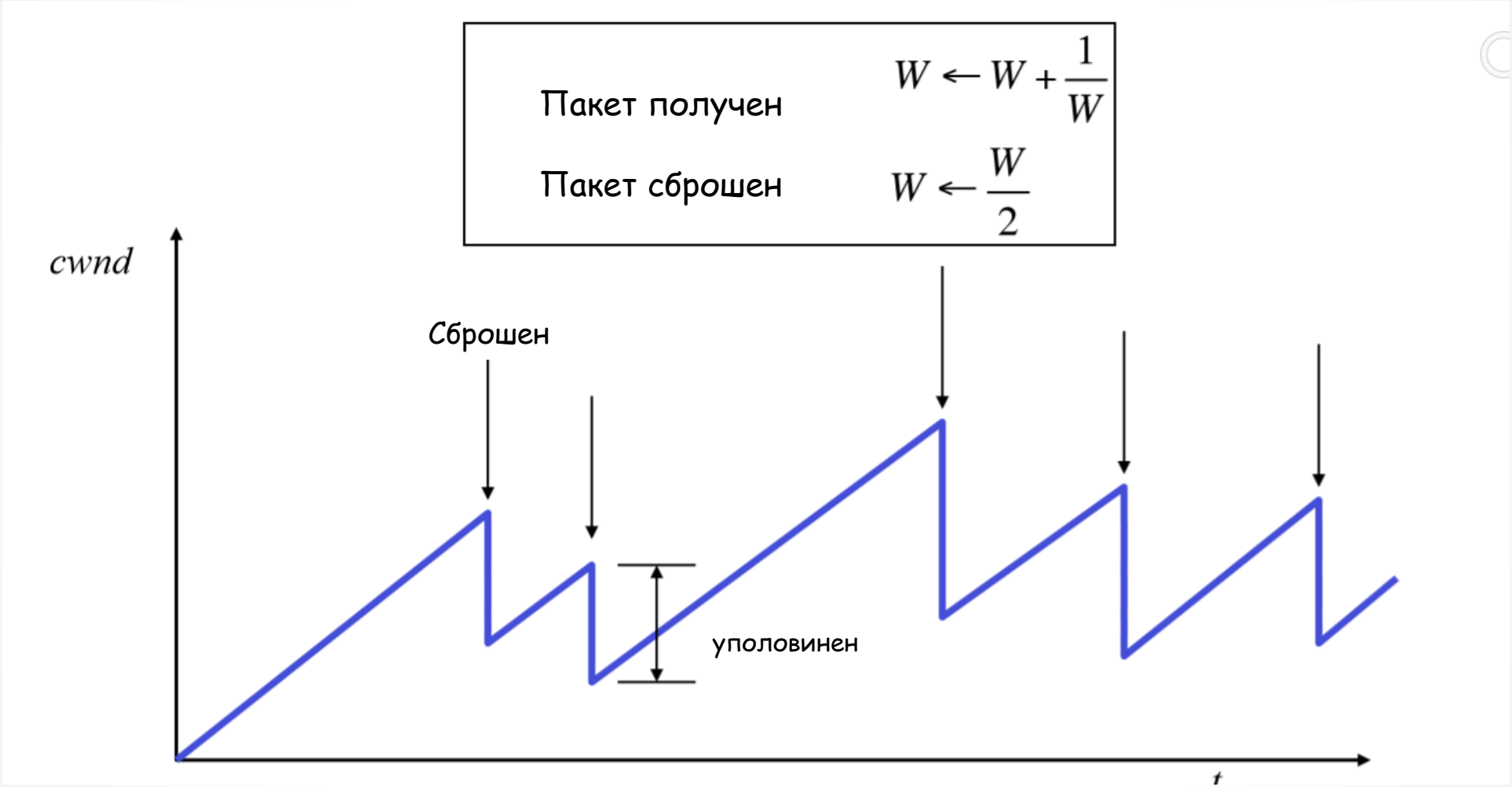
Введение в компьютерные сети

проф. Смелянский Р.Л.
Лаборатория Вычислительных комплексов
ф-т ВМК МГУ

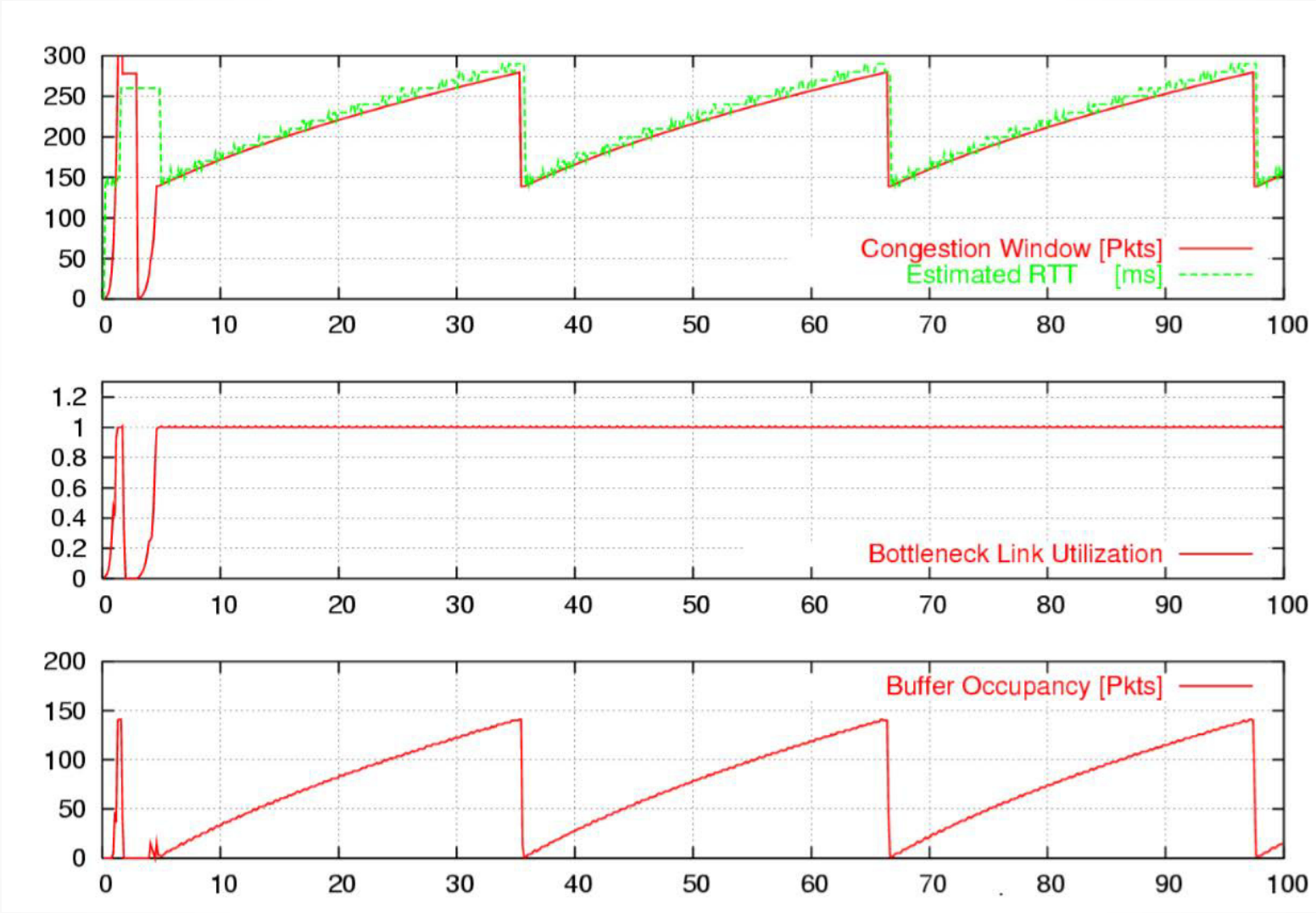
AIMD
(Additive Increase Multiple Decrease)

- Если пакет получен успешно: $W \leftarrow W + \frac{1}{W}$
- Если пакет был сброшен: $W \leftarrow \frac{W}{2}$

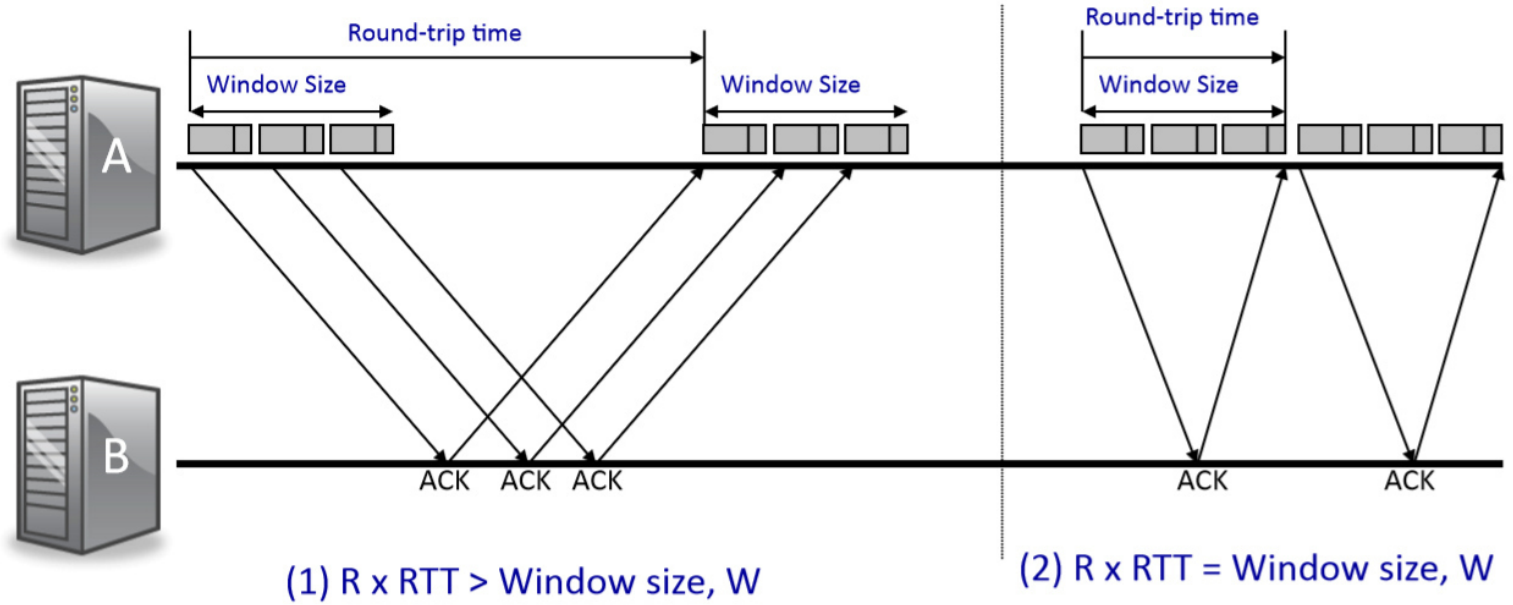
Пила AIMD


Пакет получен
$W \leftarrow W + \frac{1}{W}$
Пакет сброшен
$W \leftarrow \frac{W}{2}$
cwnd
Сброшен
уполовинен
t

Примеры динамики одиночного потока

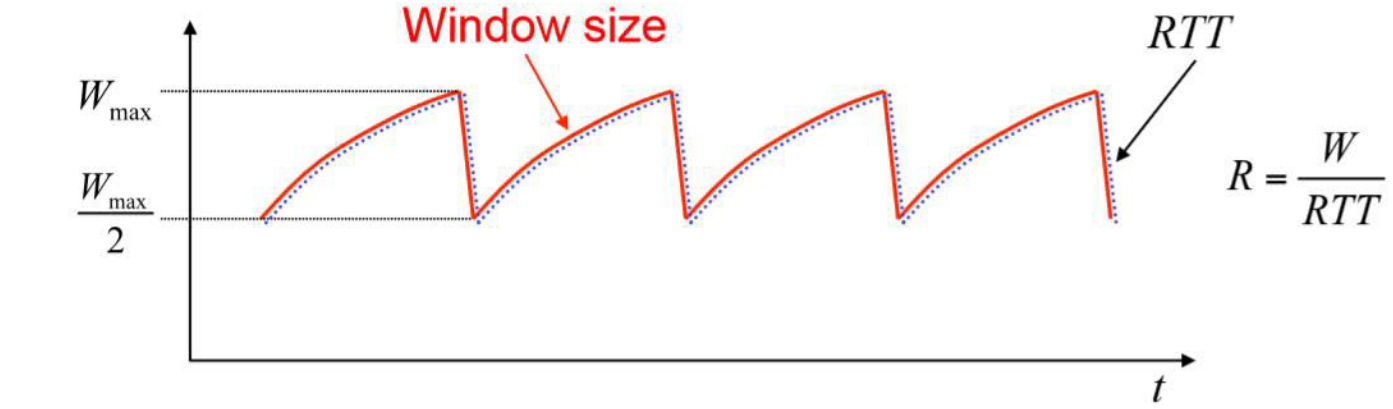


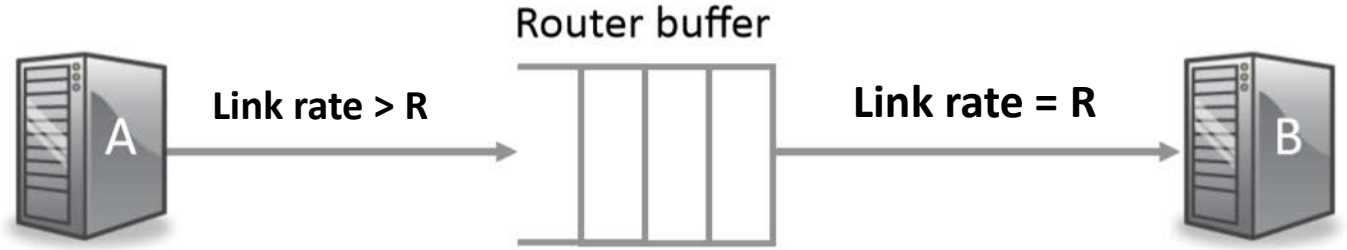
Скорость отправки для одиночного потока



$$R = \frac{W}{RTT}$$

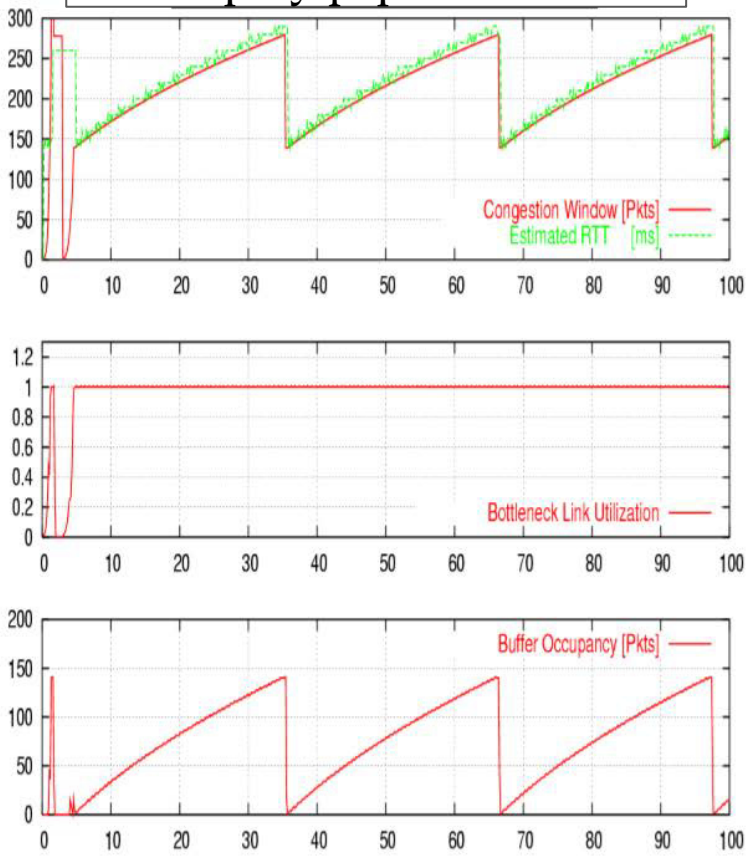
Скорость отправки для одиночного потока



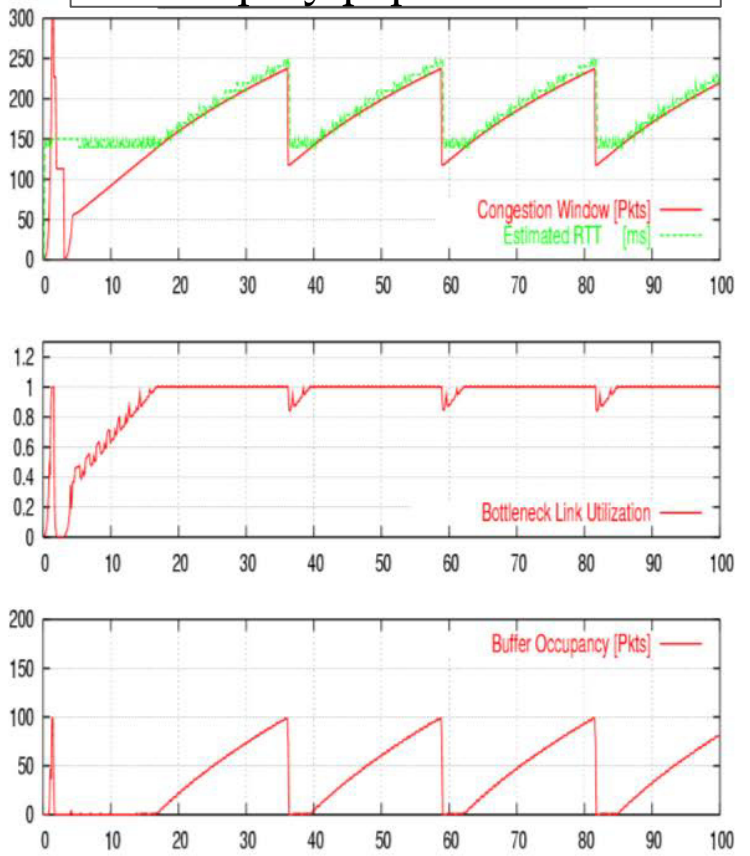


Насколько большим должен быть буфер?

Размер буфера $W = RTT*R$



Размер буфера $W < RTT*R$



Промежуточные выводы

1. В TCP управление перегрузкой располагается на хосте

- Реакция на события, наблюдаемые на конечном хосте (например, потеря пакета).
- Использование скользящего окна, предназначенного для управления потоком в TCP
- Стараться как можно быстрее оценить сколько пакетов можно безопасно отправить в сеть одновременно.
- Изменять размер окна в соответствии с алгоритмом AIMD

Комментарии для одиночного потока

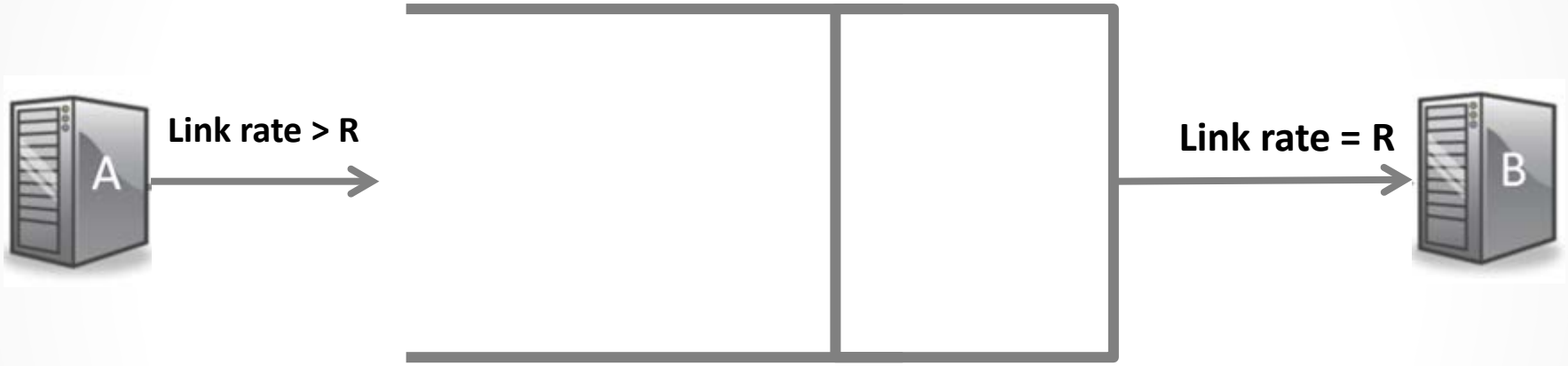
1. Окно увеличивают, сокращают в соответствии с AIMD
2. … пробировать как много байт канал еще может вместить
3. Пилообразное поведение – нормальная форма динамики
4. Скорость отправки постоянная
5. Размер буферного пространства определяет соотношение – RTT x R

Управление перегрузкой: AIMD с несколькими потоками

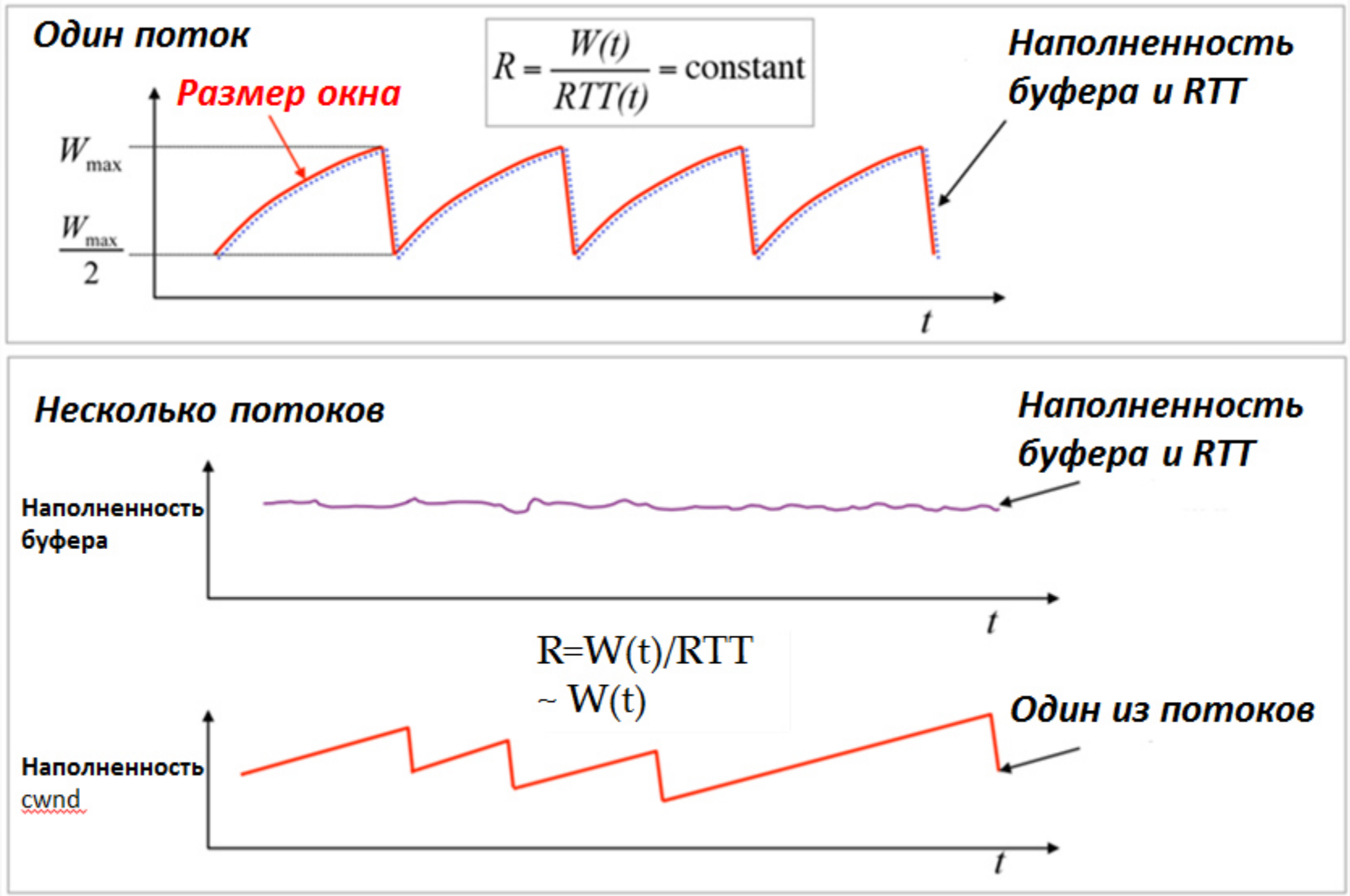
Введение в компьютерные сети

чл.-корр. РАН Смелянский Р.Л.
Кафедра АСВК ф-т ВМК МГУ

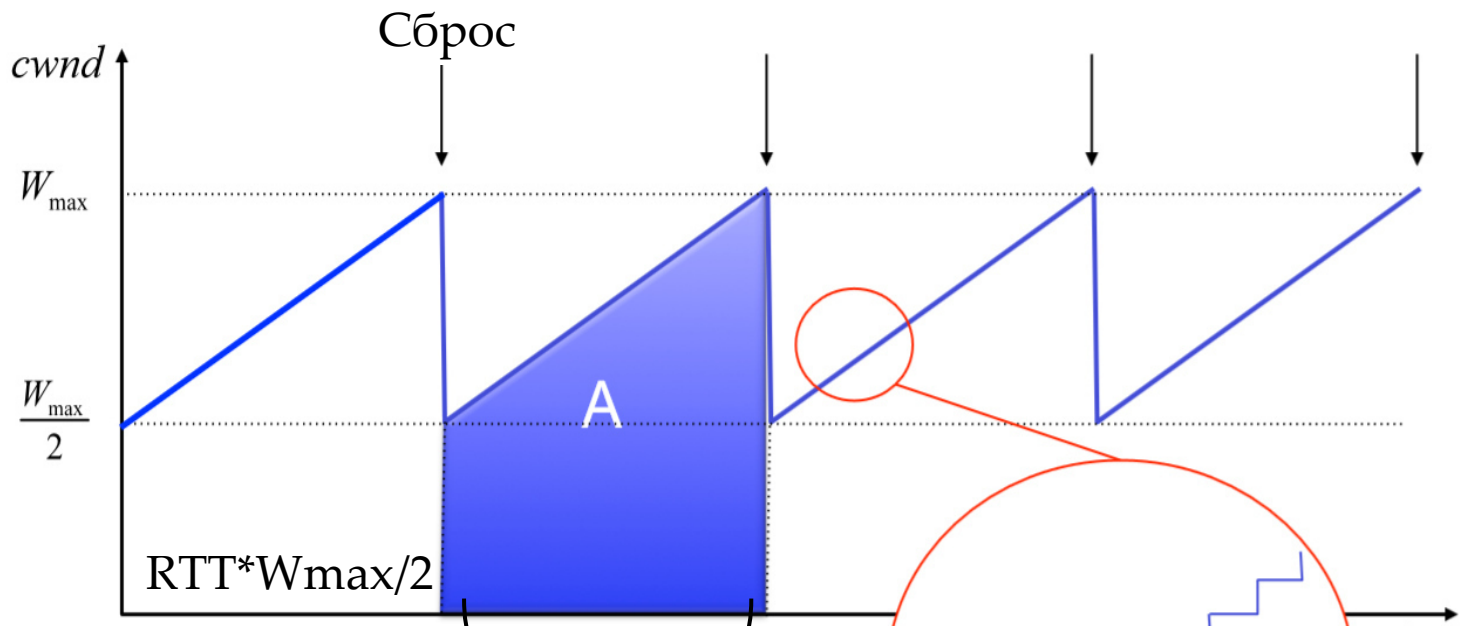
Буфер маршрутизатора



Один поток vs много потоков



Интуитивная геометрическая интерпретация



Вероятность сброса пакета

$p \approx 1/A$, где $A = \frac{3}{8} W^2$

Пропускная способность

$$R = \frac{A}{\left(\frac{W_{max}}{2}\right) RTT} = \sqrt{\frac{3}{2}} \frac{1}{RTT\sqrt{p}}$$

Интерпретация уравнения скорости

$$R = \sqrt{\frac{3}{2}} \frac{1}{RTT\sqrt{p}}$$

$$RTT \to 0 \Rightarrow R \to \infty$$

$$p \to 0 \Rightarrow R \to \infty$$

$$\frac{R1}{R2} = \sqrt[2]{\frac{p_2}{p_1}}$$

Комментарии для нескольких потоков

1. Окно увеличивают/сокращают в соответствии с AIMD
2. … пробировать как много байт канал еще может вместить
3. В «узком месте» будут скапливаться пакеты разных потоков
4. Скорость отправки меняется в зависимости от размера окна
5. AIMD очень чувствителен к вероятности потери пакетов
6. AIMD ущемляет потоки с большим RTT

Заключение:

Управление перегрузками в TCP

В TCP управление перегрузкой размещается на конечном хосте

- Реакция на события, наблюдаемые на конечном хосте (например, потеря пакета).
- Использование скользящего окна, предназначенного для управления потоком в TCP
- Механизм AIMD позволяет оперативно оценить сколько пакетов можно безопасно отправить в сеть одновременно.

Реализация управления перегрузкой на практике
(базовые алгоритмы)

Введение в компьютерные сети
чл.-корр. РАН Смелянский Р.Л.
Кафедра АСВК

Три основных вопроса

1. *Когда следует посылать новые данные?*
2. *Когда следует посылать данные повторно?*
3. *Когда надо отправлять подтверждения?*

TCP Pre-Tahoe

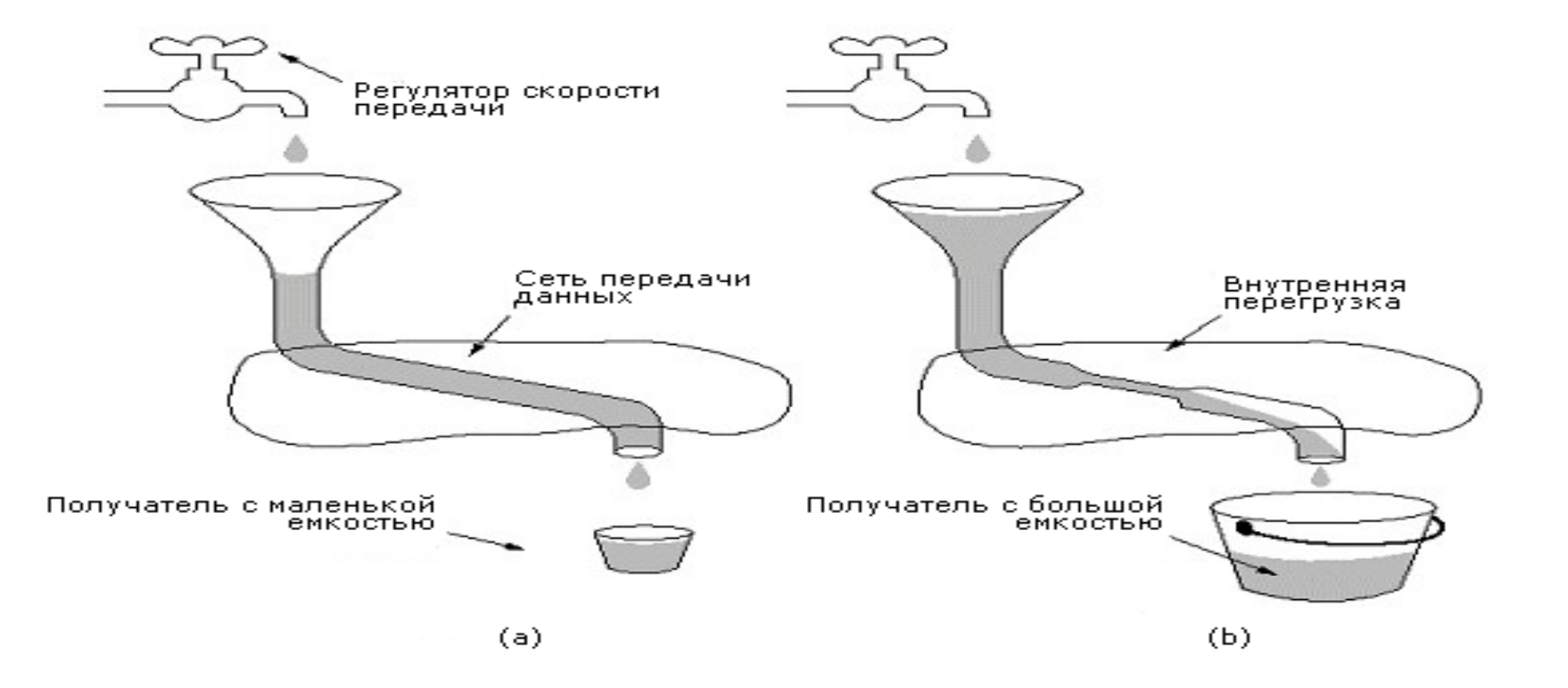
Немного истории

- *1983 – ARPA переходит на TCP/IP*
- *1986 - Интернет страдает от перегрузок*
- *1987 - Ван Якобсон предлагает TCP Tahoe*
- *1990 – Добавляются режимы быстрого восстановления и быстрой повторной передачи (Reno)*

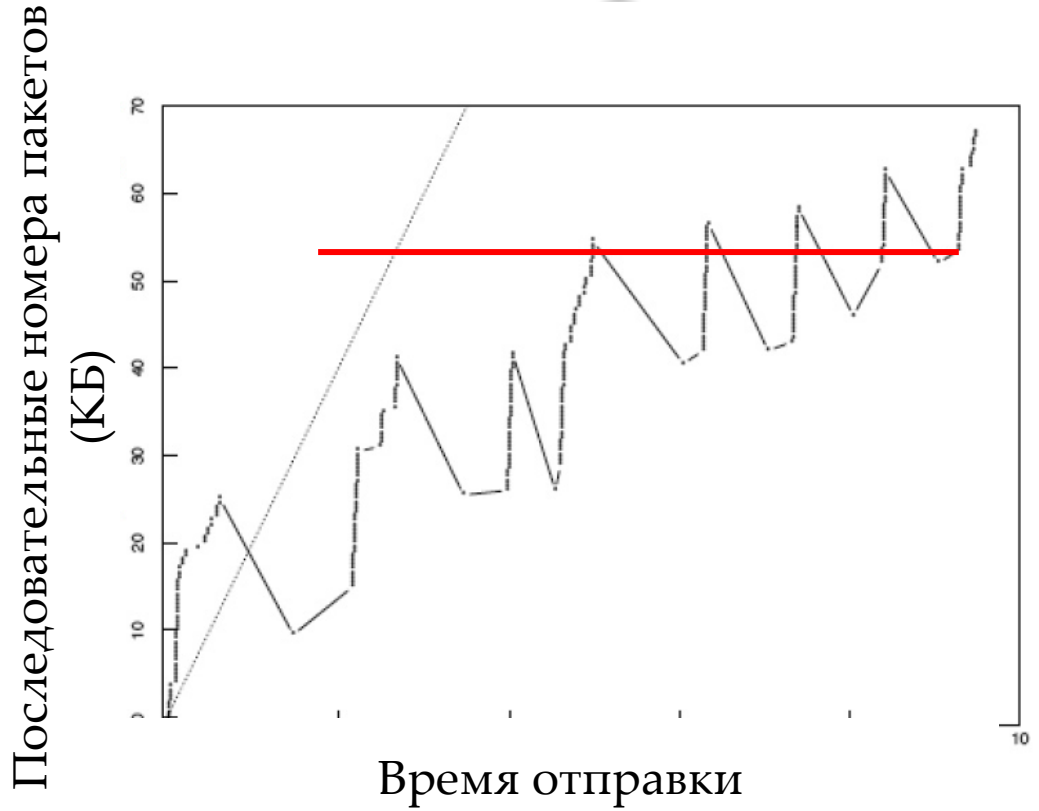
TCP Pre-Tahoe

- *Получатель устанавливает размер окна управления потоком (размер скользящего окна)*
- *Отправитель шлет пакеты, число которых полностью соответствует размеру скользящего окна*
- *На каждый пакет устанавливают таймер*
- *Проблема: что будет если размер окна превышает пропускную способность сети?*

Управление потоком и управление перегрузкой



TCP образца 1986



Переотправка одного пакета 4 раза!

Рисунок из статья ван Якобсона и Карела

TCP Tahoe

Три усовершенствования TCP

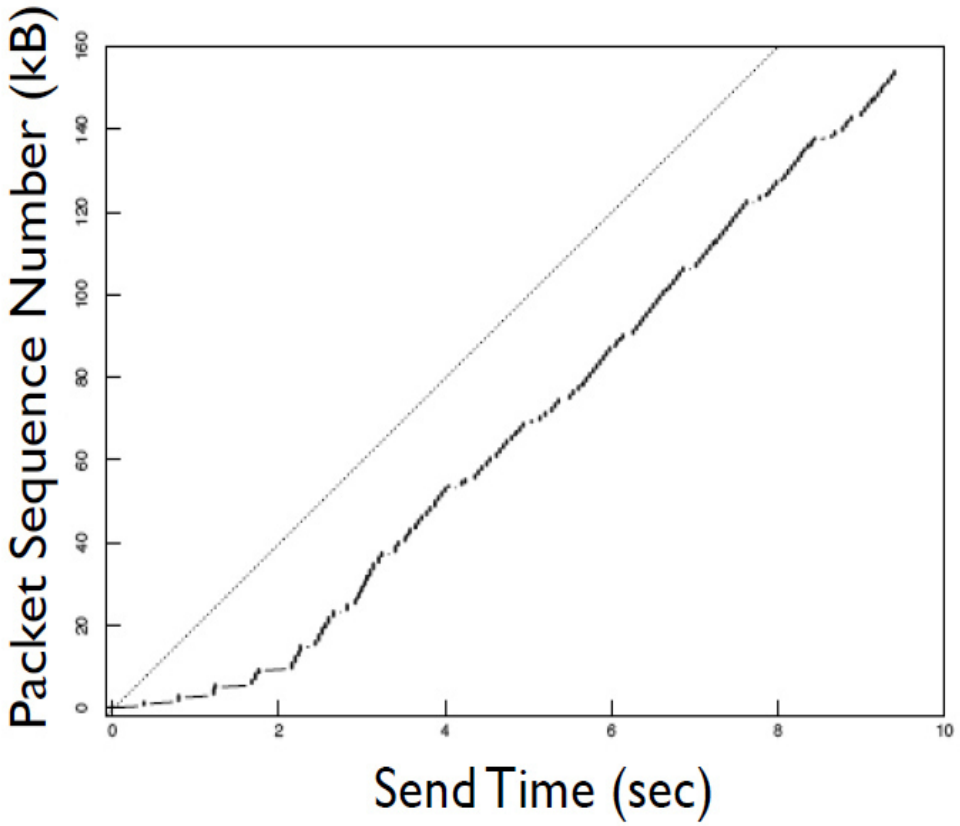
- *Окно перегрузки (CWND)*
- *Оценка Time_out*
- *Саморегулирование (Self-clocking)*

Окно перегрузки (TCP Tahoe)

- *Отправитель*
- *узнает от получателя FCWND (Flow Control WND – окно управления потоком)*
- *оценивает размер CWND*
- *Окно отправителя = min (FCWND,CWND)*
- *Разделение фазы управления перегрузкой на две*
- *Медленный старт*
- *Предотвращение перегрузки - стабилизация*

Медленный старт

- *Медленный старт*
- *CWND = MSS*
- *На каждый ACK увеличиваем окно на MSS*
- *Экспоненциально увеличиваем (удваиваем) размер окна перегрузки, прощупывая возможность сети*
- *«медленный» по сравнению с изначальным алгоритмом*

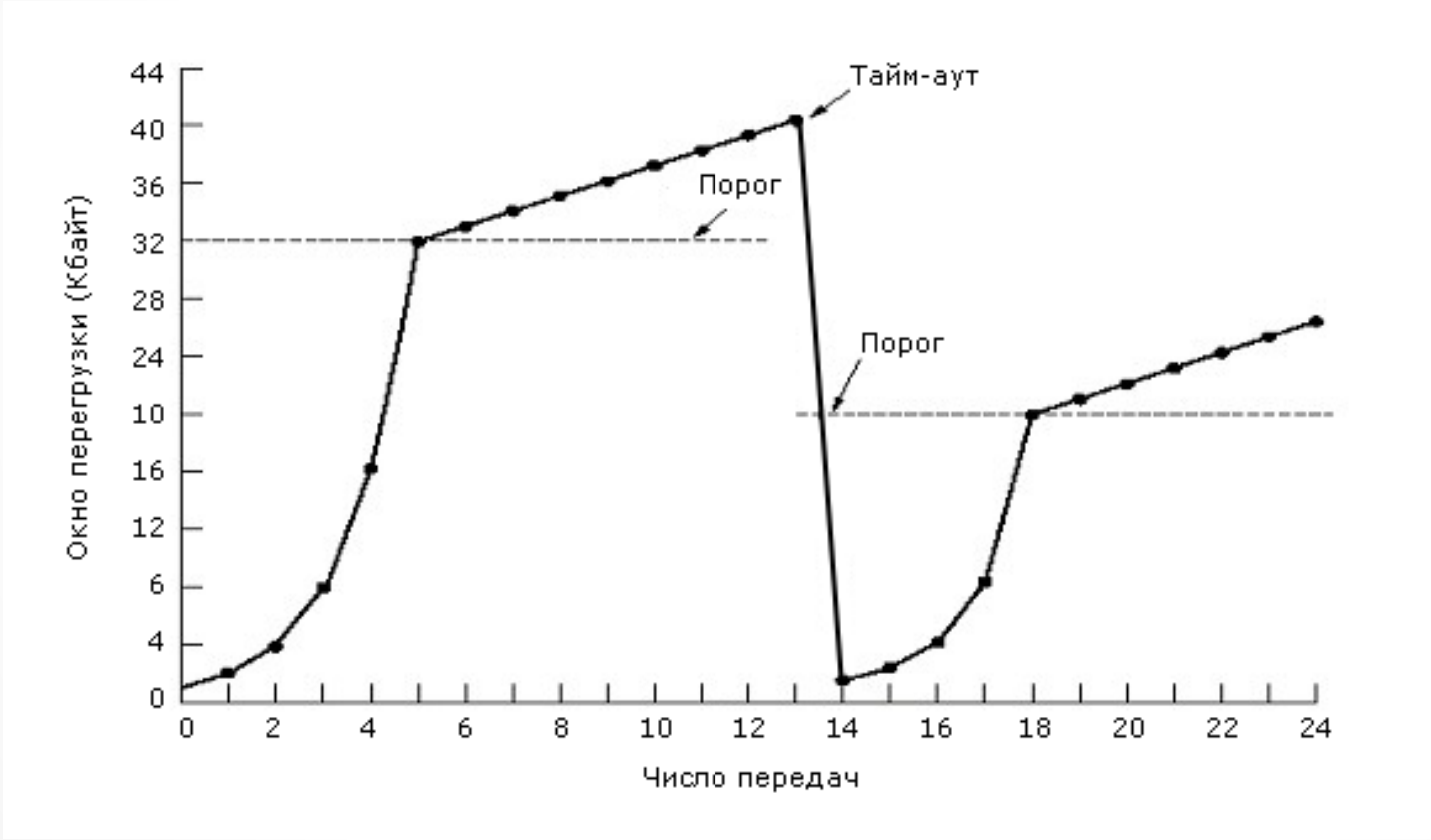


Предотвращение перегрузки

- *Медленный старт (slow start)*
  - *Увеличиваем CWND на MSS на каждое подтверждение*
  - *Экспоненциальный рост (за RTT удваиваем cwnd) пока не достигнем порога (cwnd/2 в предыдущей фазе предотвращения перегрузки) или не обнаружим перегрузку*

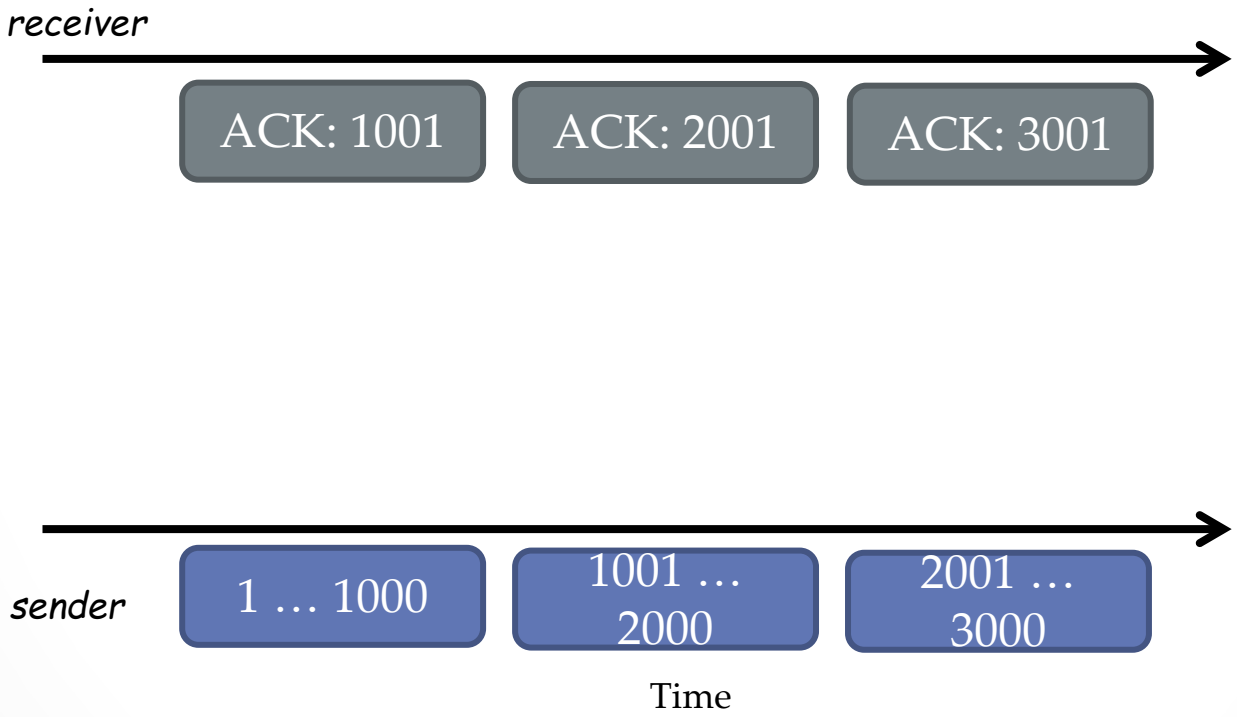
- *Предотвращение перегрузки (congestion avoidance)*
  - *Увеличиваем окно перегрузки только на MSS^2/CWND при каждом подтверждении*
  - *За каждый RTT (cwnd/MSS) увеличиваем cwnd на MSS*
  - *Линейный рост*

Пример управления перегрузками



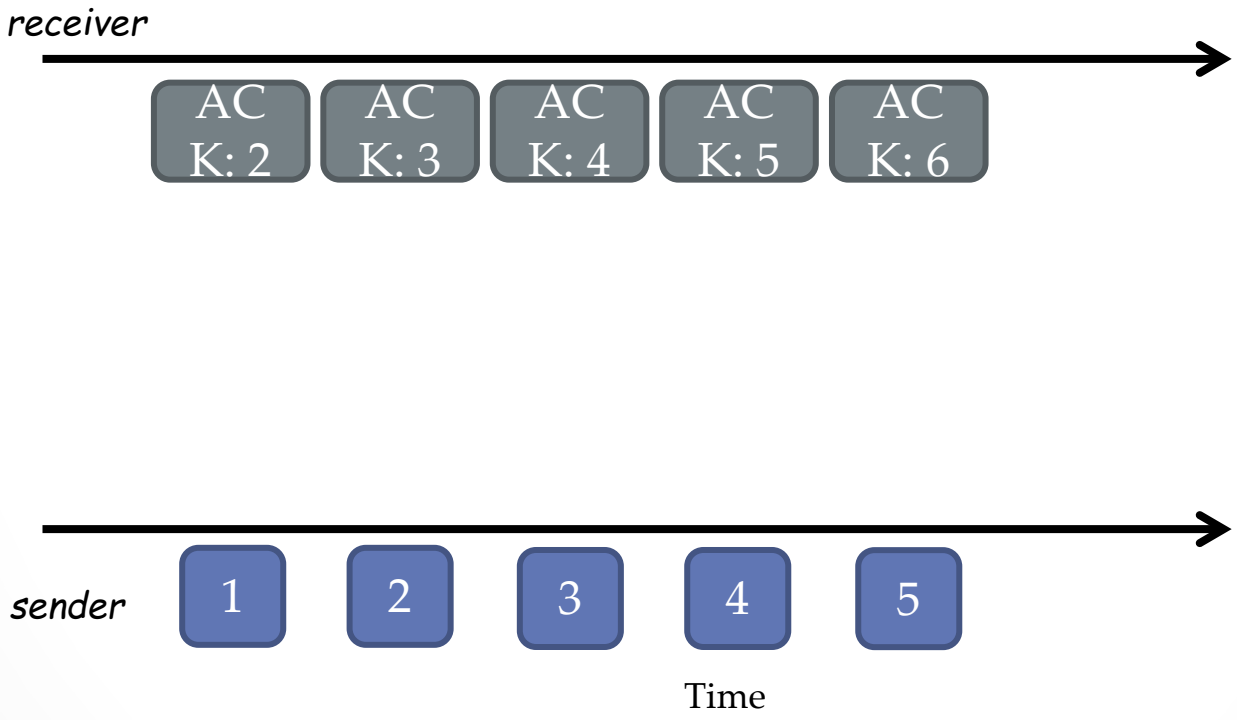
Механизм подтверждений

- *В подтверждении содержится номер следующего ожидаемого байта*



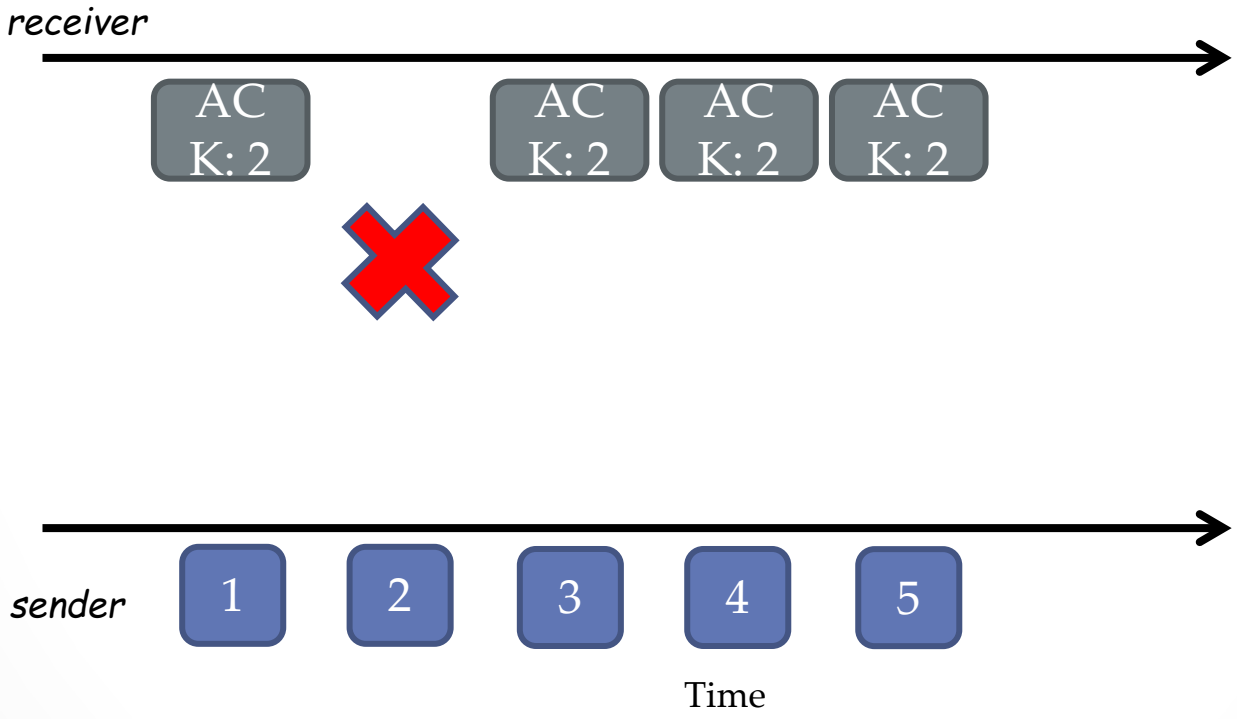
Механизм подтверждений

- *Для упрощения будем говорить не о байтах, а о пакетах.*



Механизм подтверждений

- *При потере пакета появляются повторные подтверждения (duplicate ACK)*



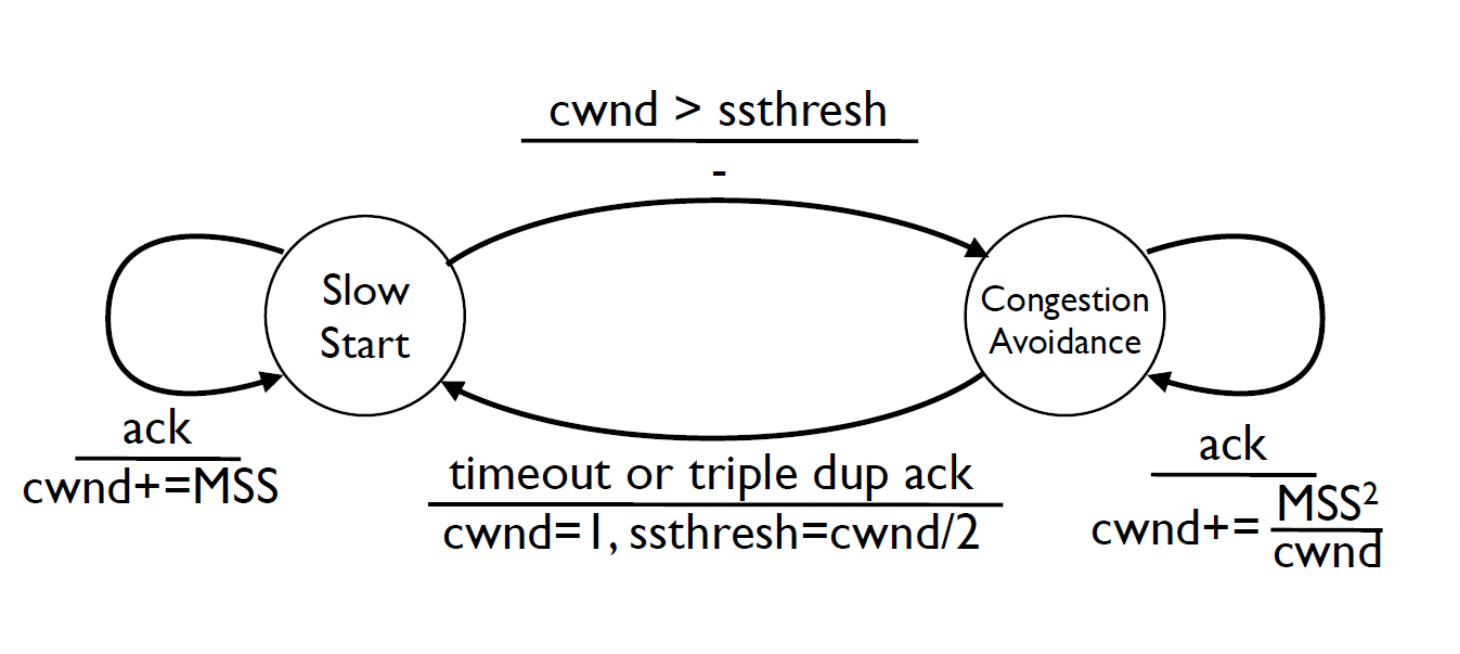
Стратегия Tahoe

- *Стратегия:*
  - *Используя медленный старт, быстро нащупать доступную пропускную способность сети*
  - *Приблизившись к насыщению, перейти в режим предотвращения перегрузки, очень осторожно пробируя возможность роста*

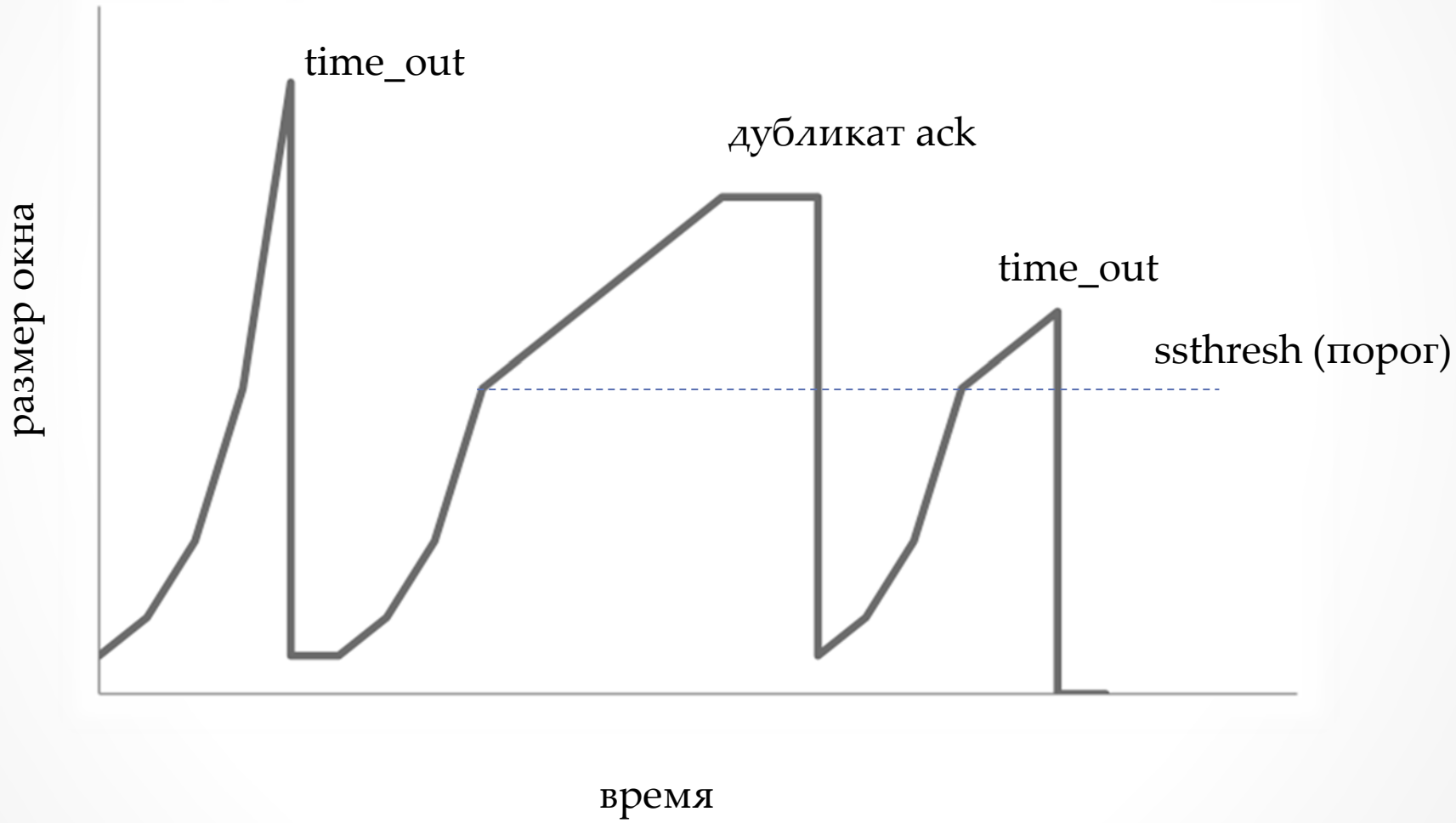
- *Признак перегрузки - Потеря пакета*
  - *по таймеру;*
  - *3 повторных подтверждения (всего 4 подтверждения с одинаковым номером).*

- *Три сигнала:*
  - *Рост номеров уведомлений – передача данных идет хорошо*
  - *Повторные уведомления – где-то произошла задержка/потеря данных, прекратить увеличение окна перегрузки*
  - *Если наступил Time_out или три раза получили dup ACK с одним и тем же номером, то устанавливаем порог = cwnd/2, cwnd = 1 и переходим в фазу медленного страта*
  - *Если пришло ACK с нужным номером, то продолжаем в фазе предотвращения перегрузки*

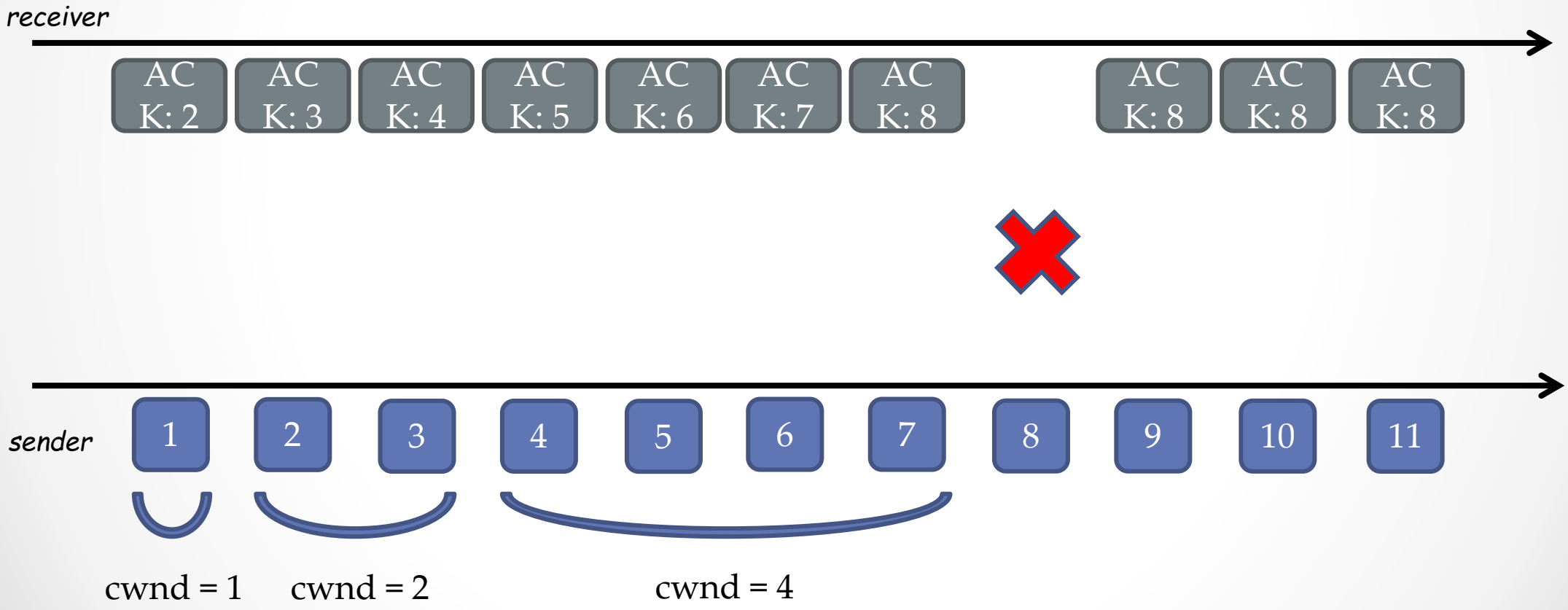
Диаграмма состояний TCP Tahoe



Динамика TCP Tahoe



Пример работы TCP Tahoe

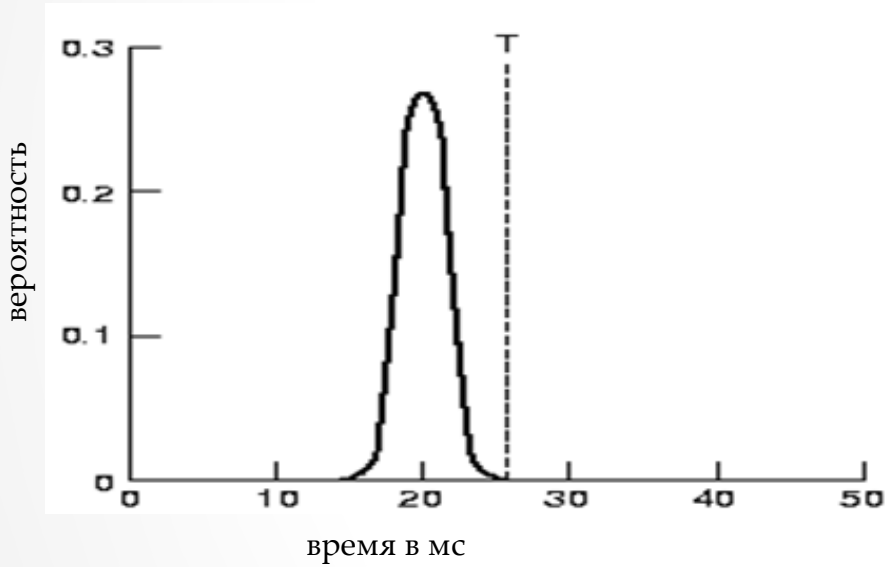


Оценка time-out

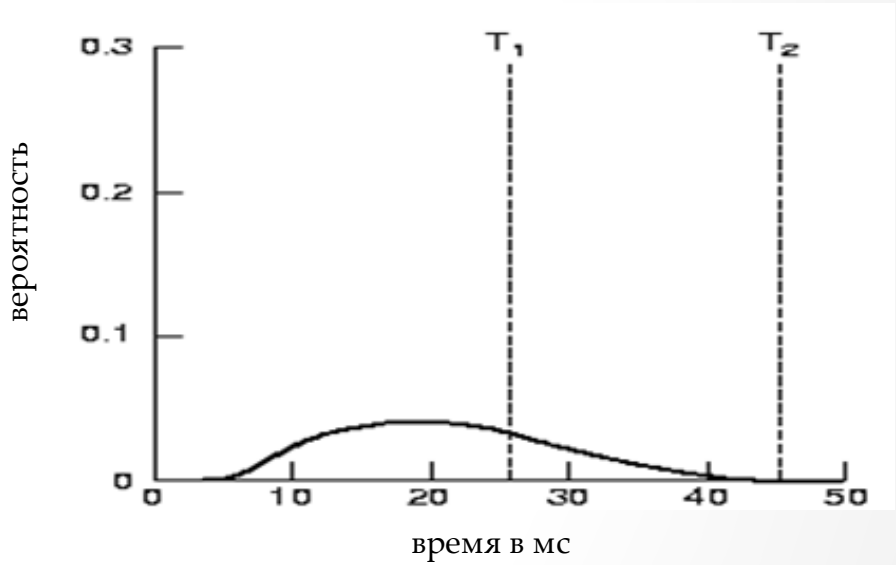
- *RTT измерение критично для оценки time-out*
  - *Если слишком коротко – впустую тратим ресурсы сети на повторные передачи, «ломаем» медленный старт*
  - *Если слишком длинный – зря тратим ресурсы на ожидание*

- *Трудности –*
  - *RTT меняется очень динамично*
  - *RTT сильно зависит от загрузки (load) сети*

Распределение задержки на канальном и транспортном уровнях



На канальном уровне



На транспортном уровне

Pre Tahoe time_out

- r – переменная, начальная оценка RTT
- m – измерение RTT для последнего подтвержденного пакета
- Вычисляем взвешенное среднее –

  $r := \alpha * r + (1-\alpha) * m$, где $0 < \alpha < 1$ (обычно 7/8)
- Time_out = $\beta * r$, где $\beta = 2$

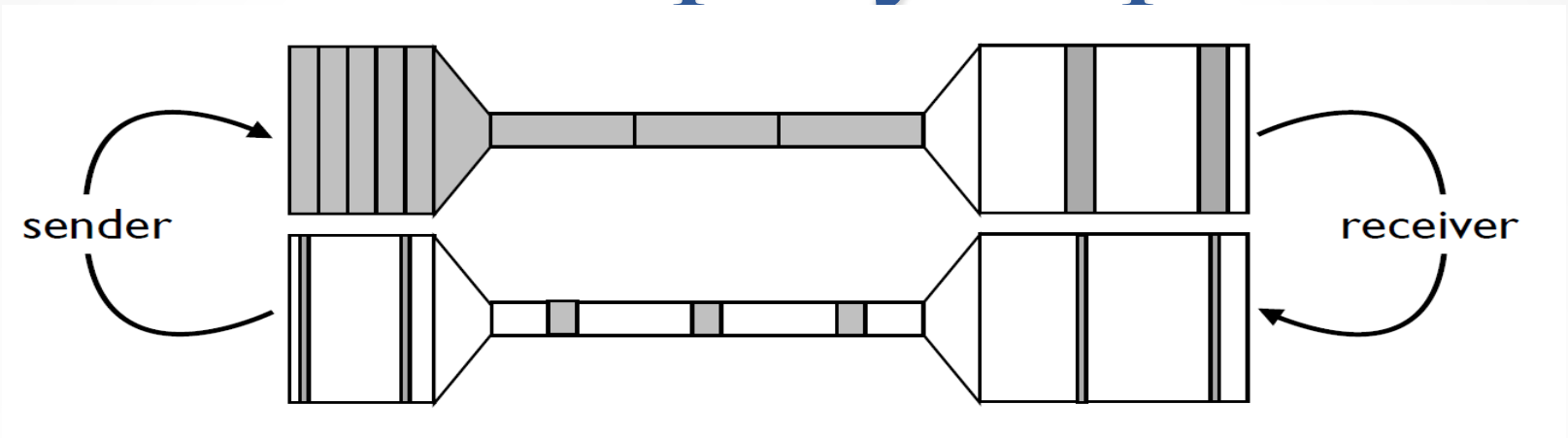
- В чем проблема?

  β - постоянная величина и не учитывает разброс значений m

TCP Tahoe time-out (т.2 стр.132-134)

- *r – переменная, начальная оценка RTT*
- *m – измерение RTT для последнего подтвержденного пакета*
- *Ошибка – e:=m-r.*
- *Вычисляем r:= ar + (1-a)m, где 1-a~ 0.125*
- *Измеряем вариацию – v := av + (1-a)|e| (средне линейное отклонение)*
- *time-out = r + β*v, где β=4*
- *В случае повторной передачи RTT = β * time-out*

Саморегулировка



- *Отправлять данные только после того, как предыдущие покинули сеть*
- *Посылать данные только при получении уведомления*
- *Отправлять уведомления как можно быстрее – это важно!*

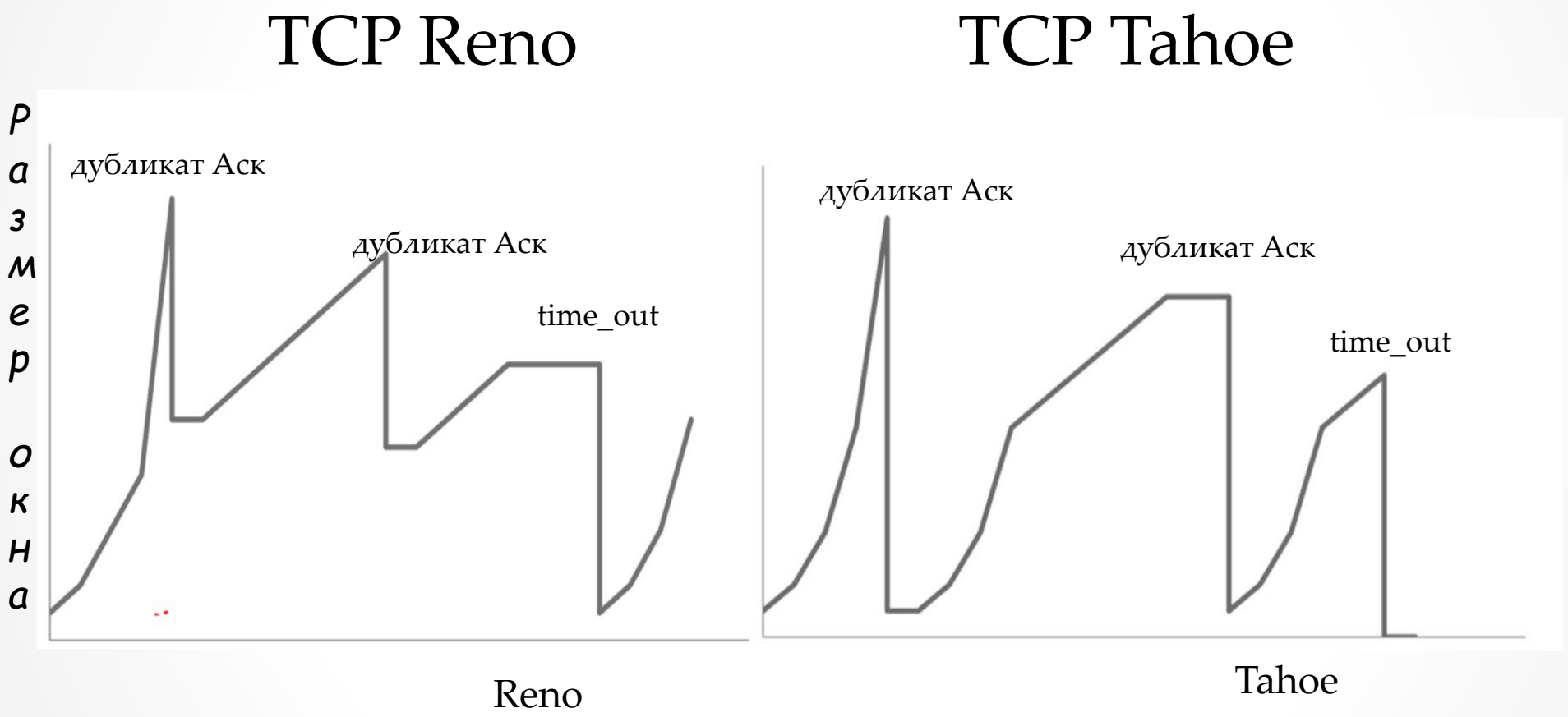
Прочесть «Congestion Avoidance and Control» van Jacobson and Karels

TCP Reno

TCP Tahoe vs TCP Reno

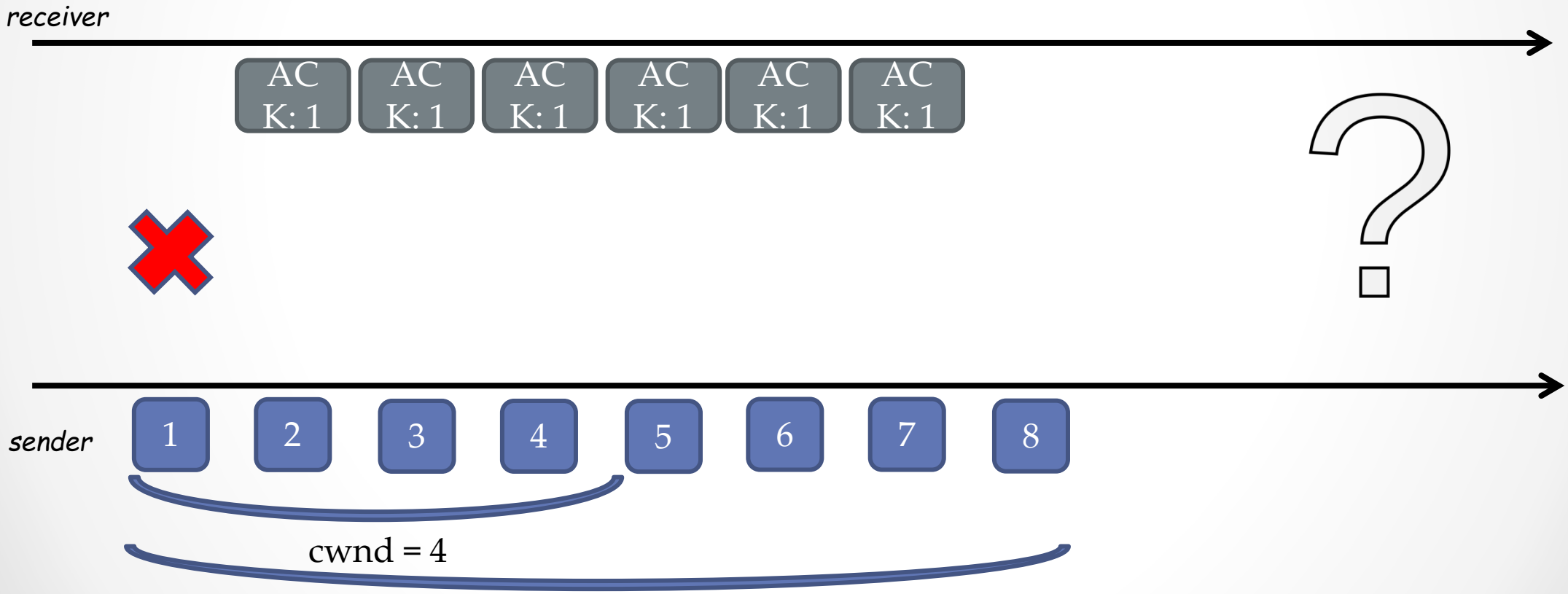
- *TCP Tahoe - Не ждать time-out, а уже по тройному уведомлению (подразумевается потеря пакета) начинать повторную пересылку и переход в фазу медленного старта*
  - *установить порог = CWDN/2*
  - *сбросить CWDN в 1*
  - *начать медленный старт*
- *TCP Reno – Добавлены фазы быстрой переотправки (fast retransmit) и быстрого восстановления (fast recovery):*
  - *по тройному ACK*
    - *cwnd = cwnd/2, а не 1*
    - *шлем запрашиваемый сегмент (fast retransmit)*
    - *Переходим в фазу быстрого восстановления (fast recovery)*
  - *По time-out схема Tahoe*

TCP Tahoe vs TCP Reno



Быстрое восстановление

Какая должна быть реакция на большое количество повторных подтверждений ?



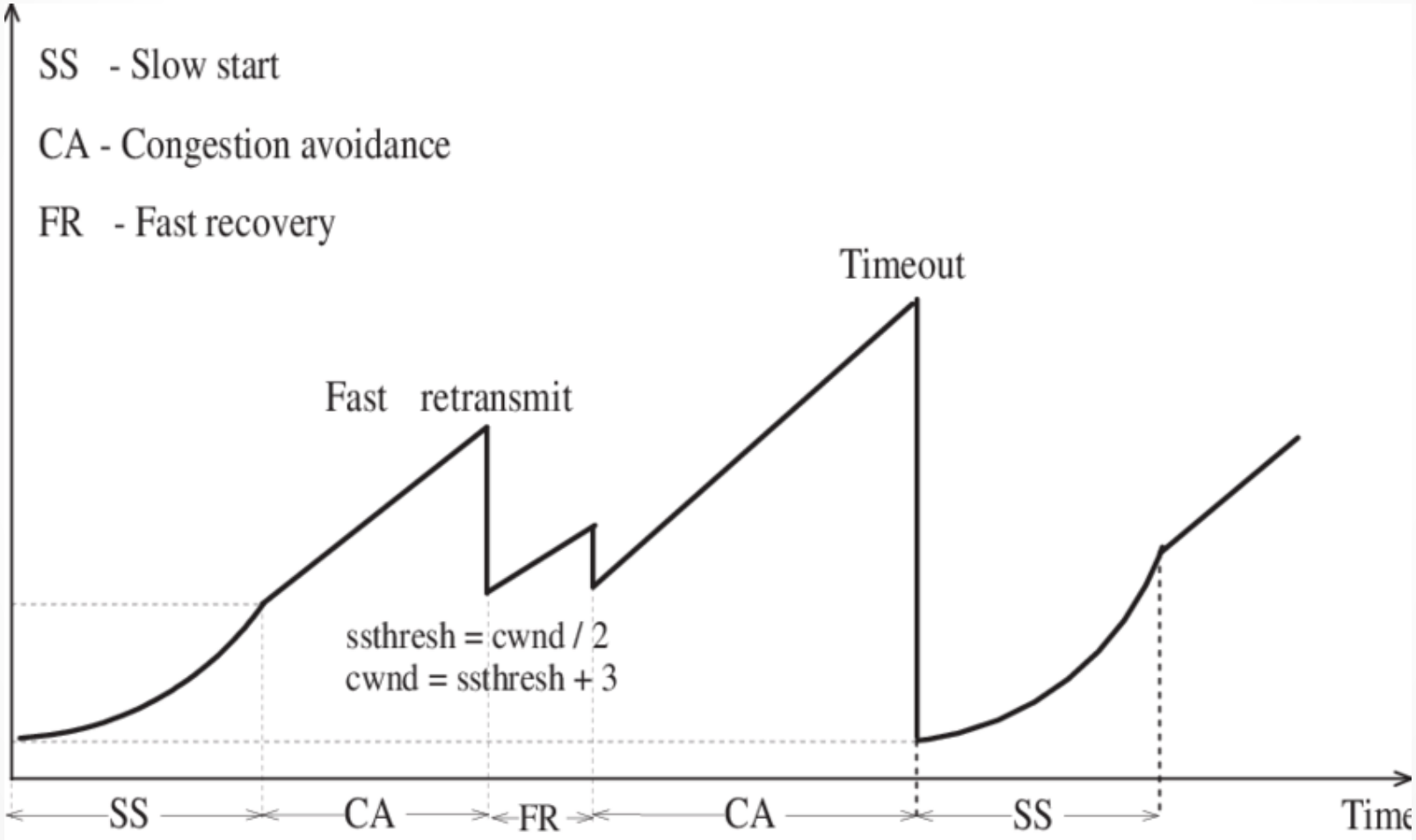
Быстрое восстановление

- *Окно перегрузки определяет количество пакетов, которое может одновременно находиться в сети*
- *Повторное подтверждение означает, что некоторый пакет был доставлен не по порядку -> в сети стало меньше пакетов, чем cwnd*
- *3 повторных подтверждения означают, что 3 пакета покинуло сеть*
- *Проблема: мы не можем сдвинуть sliding window вправо на 3 позиции из-за неподтвержденного пакета*
- *Решение: т.к sliding window > congestion window, то мы можем продолжить отправку пакетов, не боясь перегрузить получателя.*

Быстрое восстановление

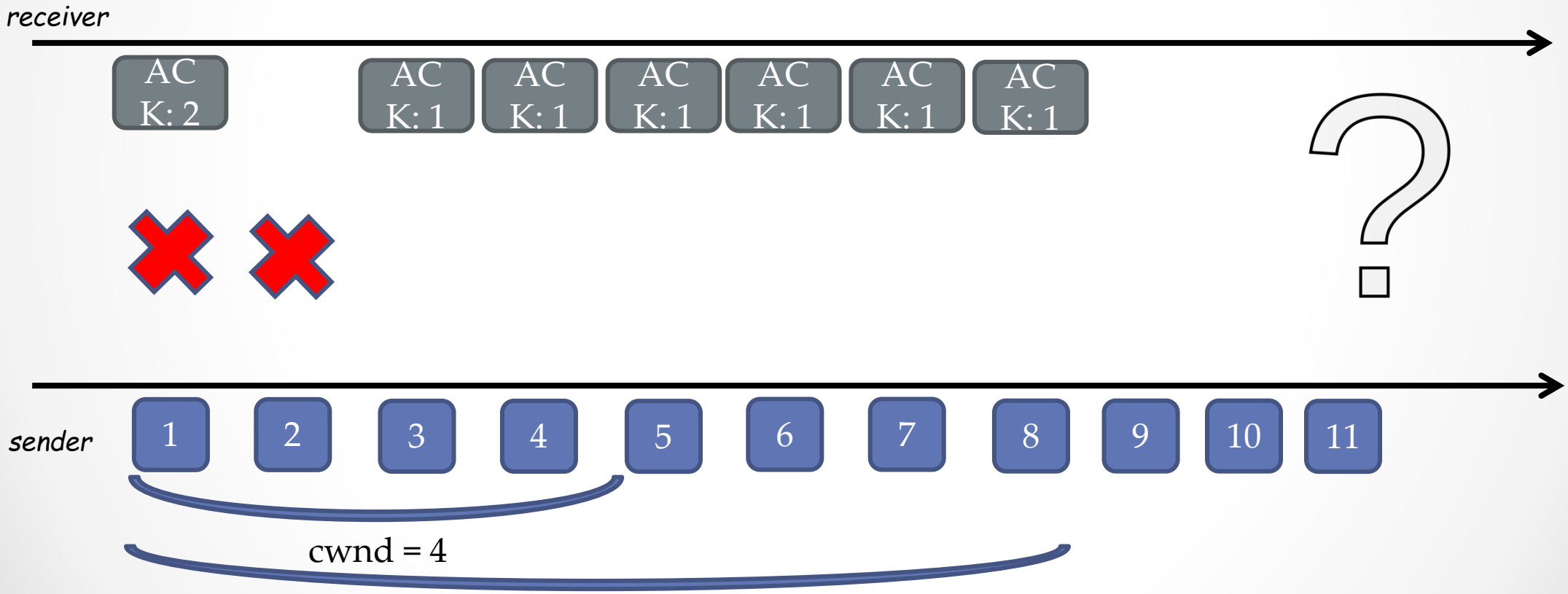
- *В начале фазы быстрого восстановления устанавливаем окно в cwnd/2 + 3*
- *Каждое последующее повторное подтверждение означает выход еще одного пакета из сети, поэтому на каждый dup ack увеличиваем cwnd на 1: cwnd += 1*
- *При получении подтверждения на переотправленный пакет мы можем двигать левую границу cwnd, поэтому возвращаем значение cwnd в cwnd/2.*

Быстрое восстановление



Быстрое восстановление: проблема нескольких потерь

Представим, что теряется несколько пакетов



TCP New Reno

- Решает проблему нескольких потерянных пакетов
- По time-out поведение такое же как и у Tahoe/Reno
- В фазе быстрого восстановления:
  - при входе в эту фазу - запомнить последний не подтвержденный пакет
  - при каждом повторном уведомлении – увеличить CWND на 1
- Начать отправку новых пакетов пока находимся в фазе быстрого восстановления
- Когда последний пакет подтвержден:
  - вернуться в фазу предотвращения перегрузки
  - восстановить размер CWND до того размера, который оно имело до входа в фазу быстрого восстановления

Определение перегрузки

- Потеря пакета
  - по таймеру;
  - 3 повторных подтверждения (всего 4 подтверждения с одинаковым номером).
- По изменению RTT (при перегрузке увеличивается задержка буферизации)
- По уведомлениям из сети (ECN)
- Гибридные методы

Управление перегрузками

- *Одна из сложнейших проблем в компьютерных сетях (особенно на неоднородных, протяженных, с ошибками соединениях)*
- *Основной подход: AIMD (additive increase, multiple decrease)*
- *Дома построить диаграмму состояний для TCP Reno*