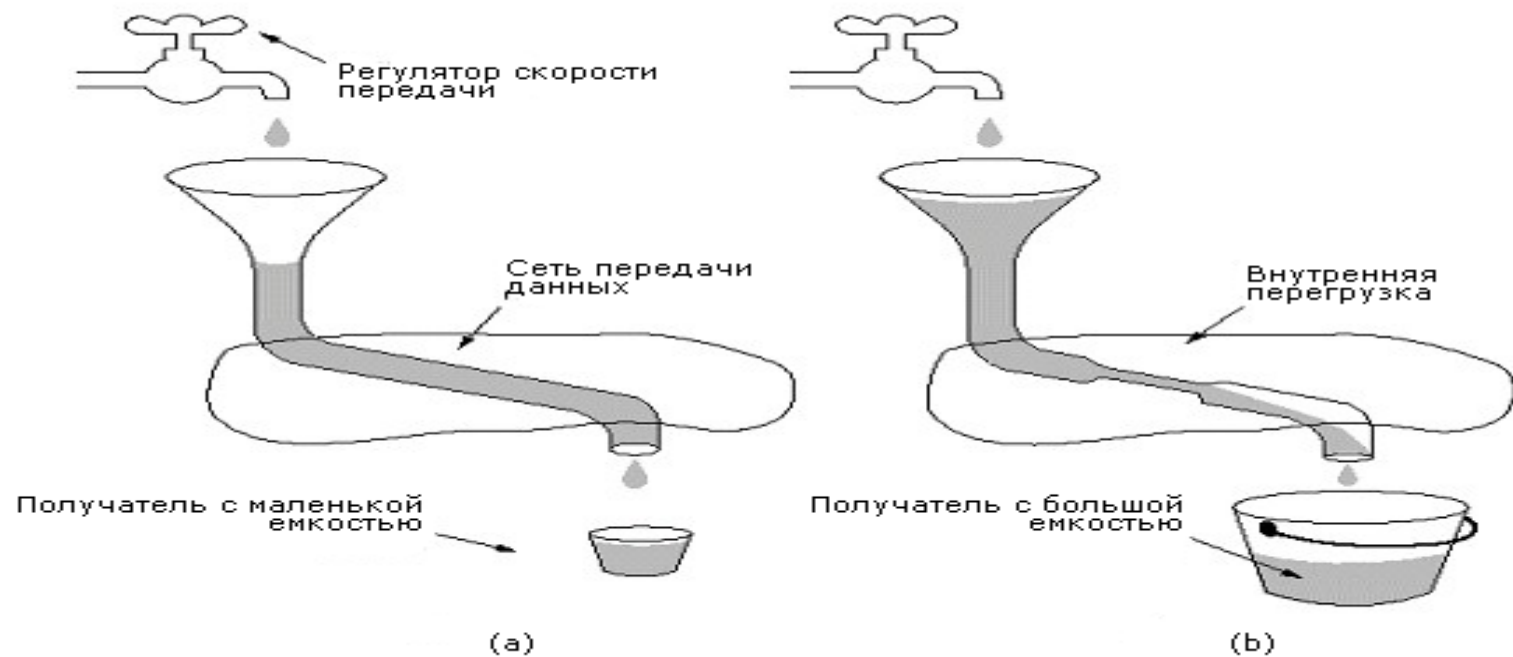




Управление перегрузкой: ТСР Tahoe

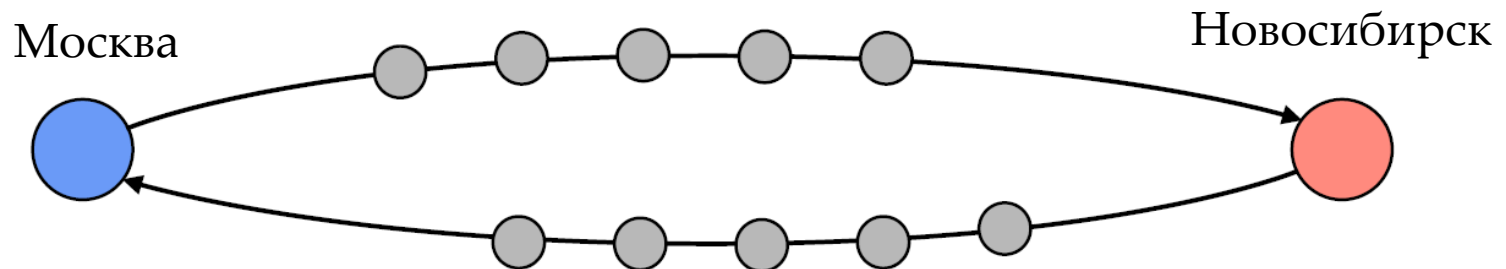
Введение в компьютерные сети
проф. Смелянский Р.А.
Лаборатория Вычислительных комплексов
ф-т ВМК МГУ

Проблема





Проблема



Управление перегрузкой: поддержка скорости передачи данных так, чтобы не перегружать сеть (число посланных, но не подтвержденных пакетов) и не терять производительность

Нужны два окна: для контроля получателя, второе – для контроля сети



Три основных вопроса

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



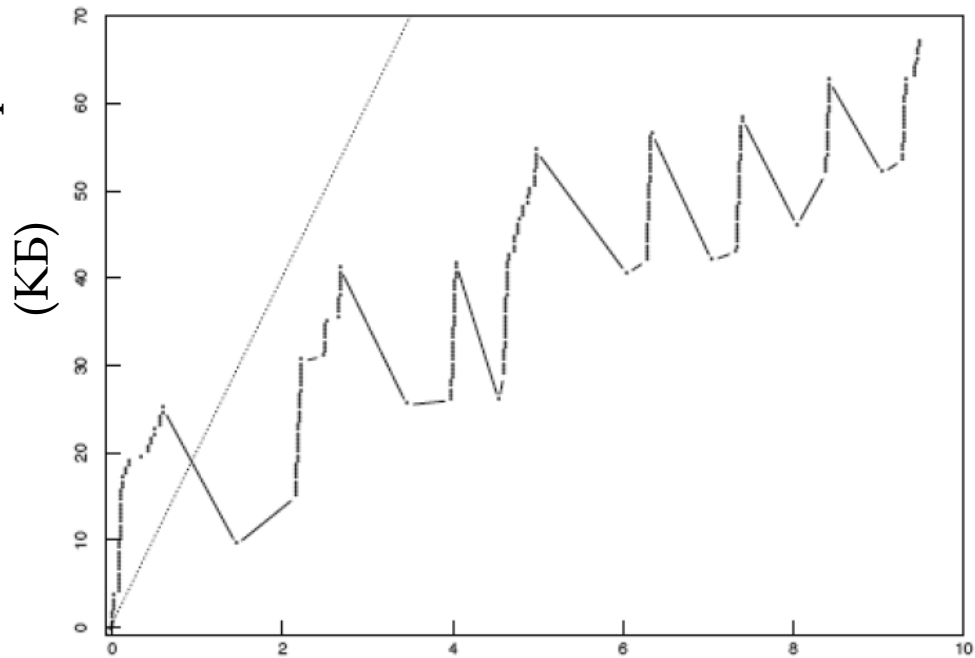
TCP Pre-Tahoe

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



ТСР образца 1986

Последовательные номера пакетов
(КБ)



Время отправки

Рисунок из статьи ван Якобсона и Карела



Три усовершенствования ТСР

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



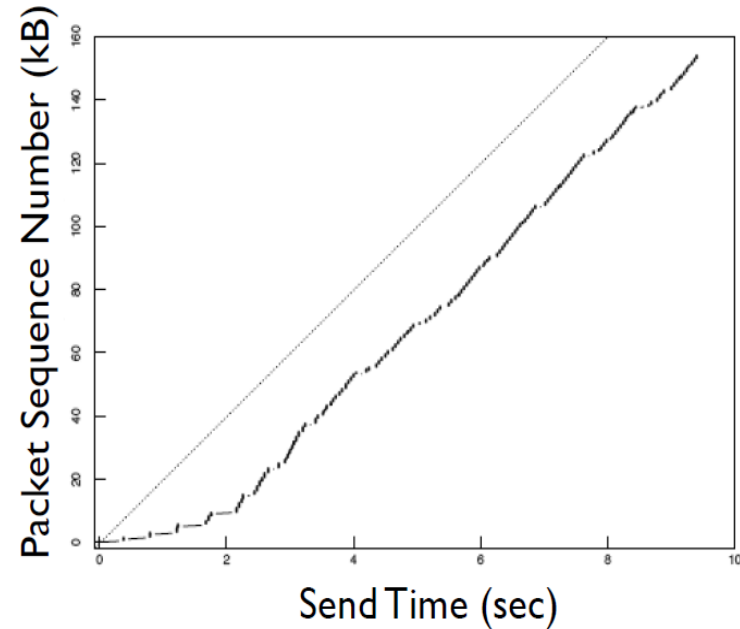
Окно перегрузки (TCP Tahoe)

- **Отправитель**
 - узнает от получателя *FCWND* (*Flow Control WND*)
 - 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) окно перегрузки, прощупывая возможность сети
- «медленный» по сравнению с изначальным алгоритмом





Предотвращение перегрузки

- Медленный старт
 - Увеличиваем $CWND$ на MSS на каждое подтверждение
 - Экспоненциальный рост (за RTT удваиваем $cwnd$) пока не достигнем порога ($cwnd/2$ в предыдущей фазе предотвращения перегрузки)
- Предотвращение перегрузки
 - Увеличиваем окно перегрузки только на $MSS^2/CWND$ при каждом подтверждении
 - За каждый RTT ($cwnd/MSS$) увеличиваем $cwnd$ на MSS
 - Линейный рост

Пример управление перегрузками

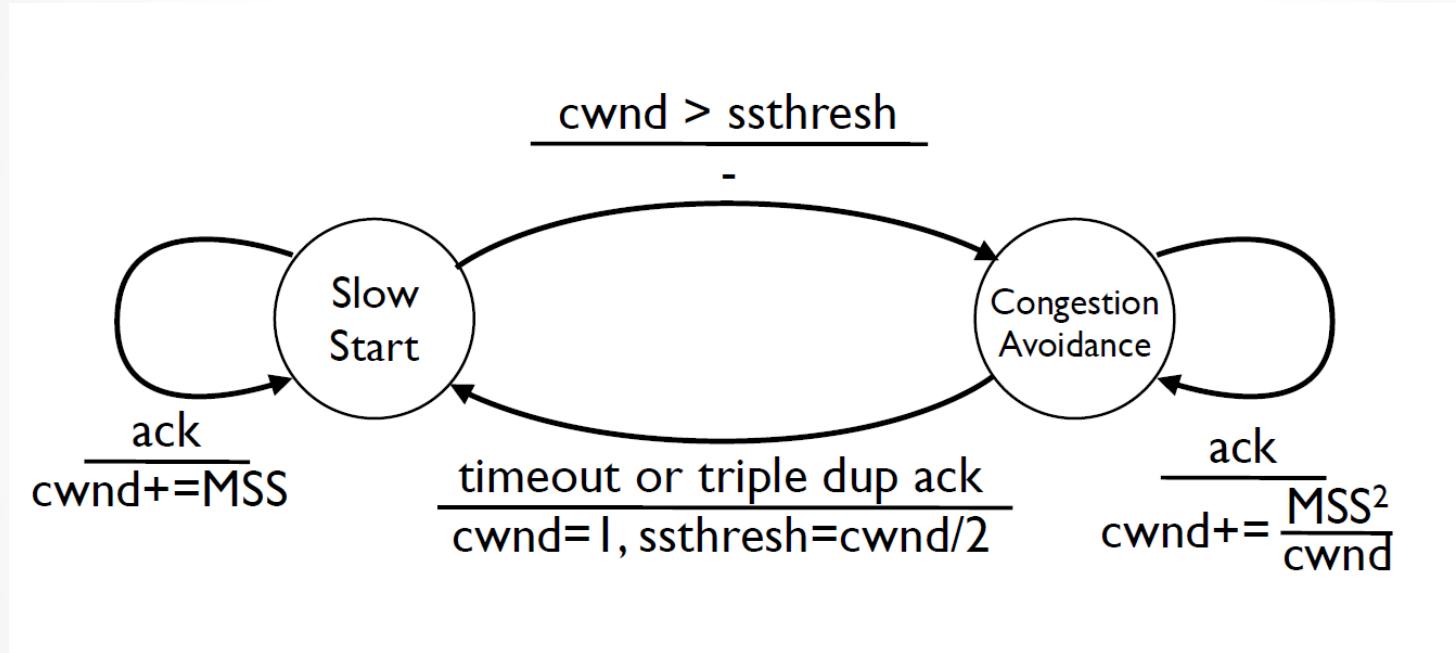




Стратегия Tahoe

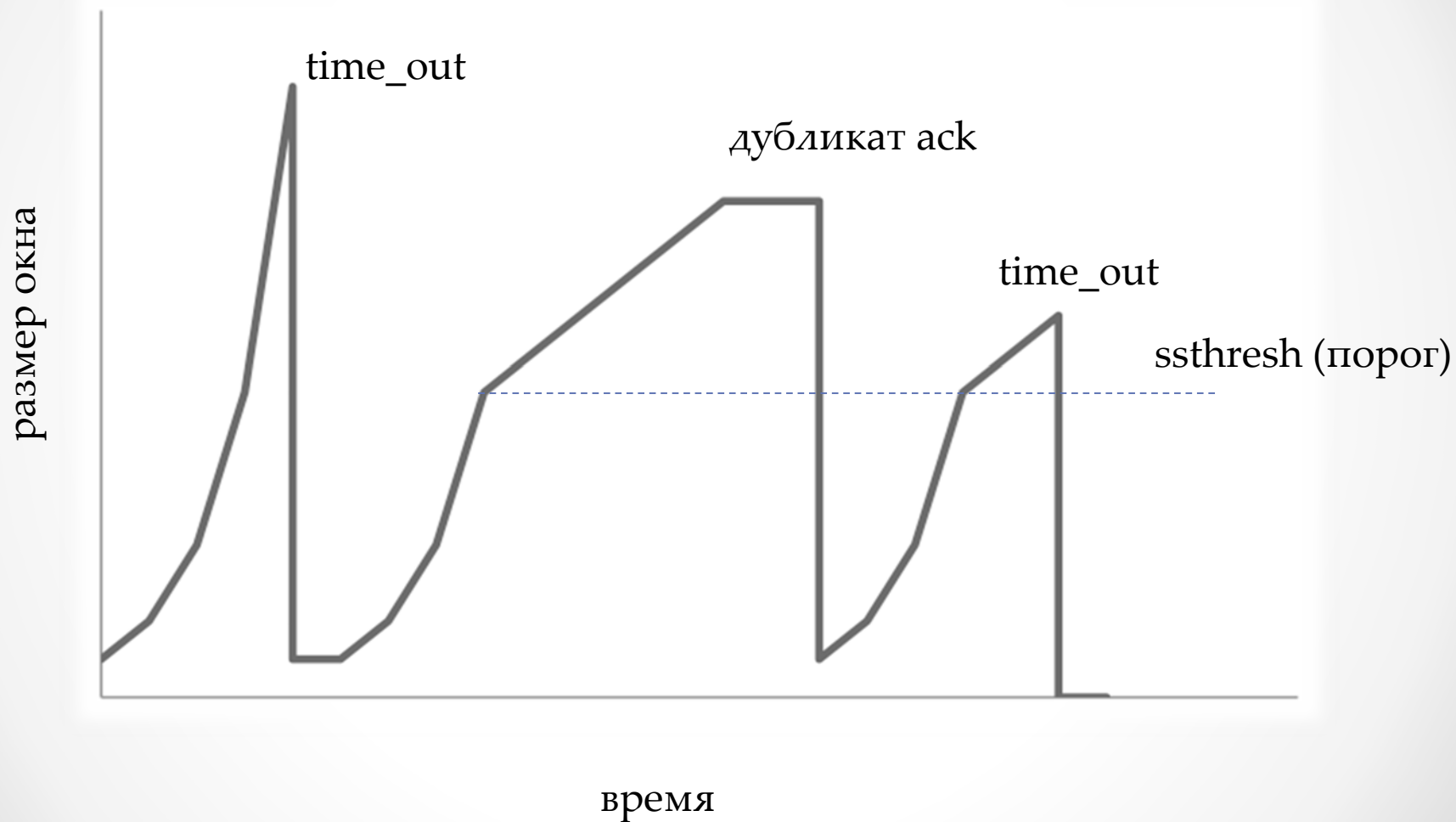
- **Стратегия:**
 - Используя медленный старт, быстро нащупать доступную пропускную способность сети
 - Приблизившись к насыщению, перейти в режим предотвращения перегрузки, очень осторожно пробируя возможность роста
- **Три сигнала:**
 - Рост номеров уведомлений - передача данных идет хорошо
 - Повторные уведомления - где-то произошла задержка/потеря данных
 - `Time_out` - что-то работает не так как надо

Диаграмма состояний TCP Tahoe



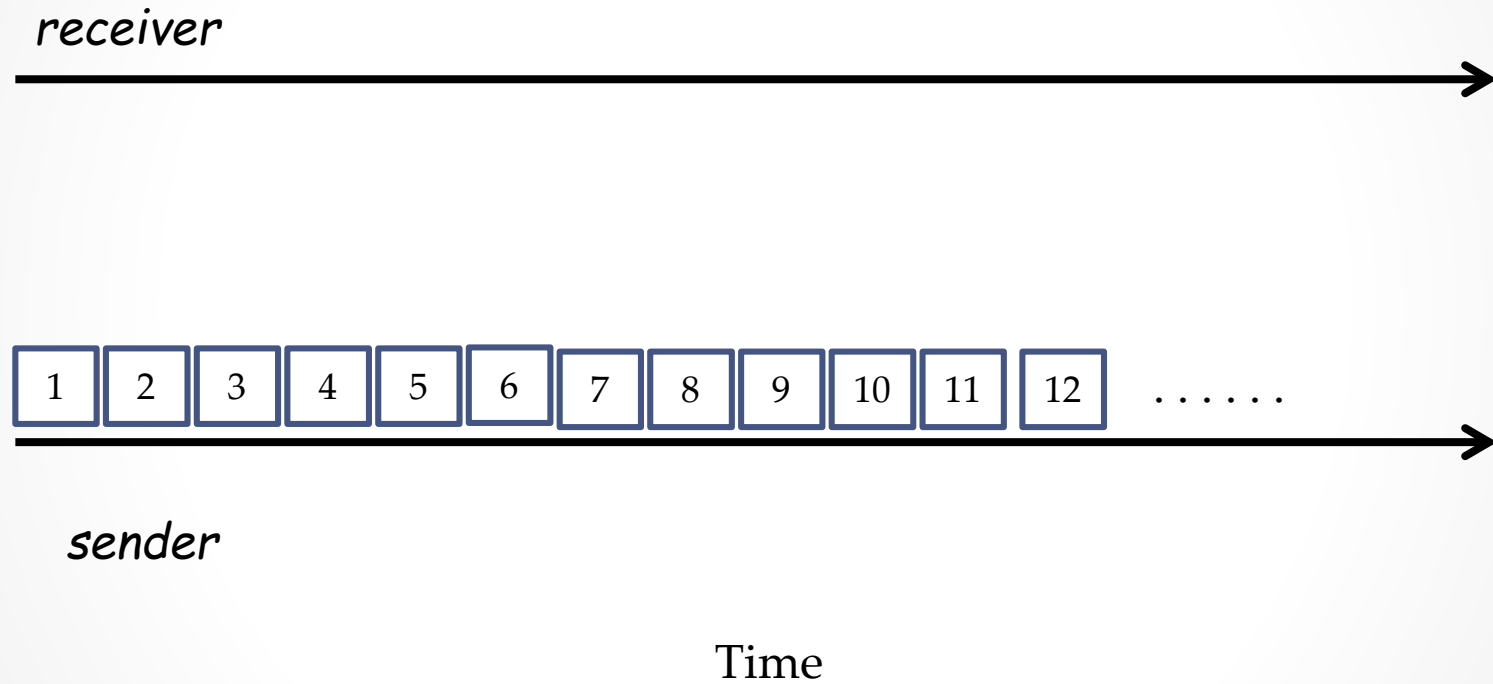


Динамика ТСР Tahoe





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*



Pre Tahoe time_out

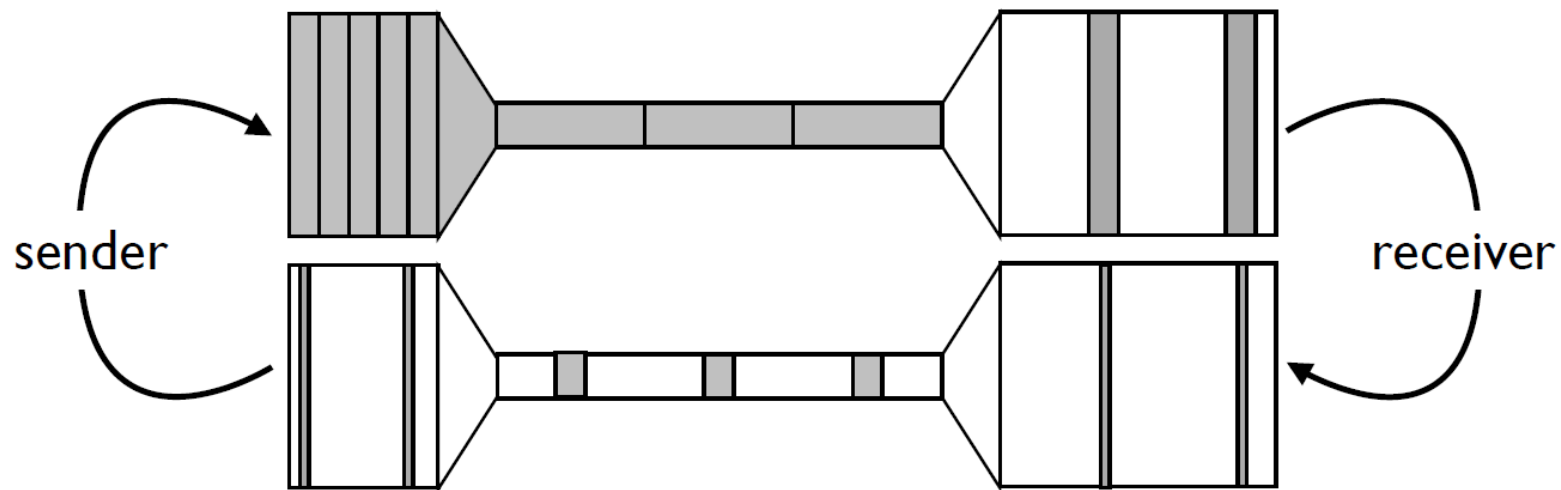
- r - начальная оценка RTT из соображений здравого смысла
- t - измерение RTT для последнего подтвержденного пакета
- Вычисляем взвешенное среднее - $r = \alpha r + (1 - \alpha)t$,
где $0 < \alpha < 1$
- $\text{Time-out} = \beta r$, где $\beta = 2$
- В чем проблема?

TCP Tahoe time-out (т.2 стр.132-134)

- r - начальная оценка RTT из соображений здравого смысла
- t - измерение RTT для последнего подтвержденного пакета
- Ошибка - $e = t - r$, где t измерение для последнего аск.
- Вычисляем $r = \alpha r + (1 - \alpha)t$, где $\alpha \sim 0.25$
- Измеряем вариацию - $v = \alpha v + (1 - \alpha)|e|$
- $\text{time-out} = r + \beta \times v$, где $\beta = 4$.
- В случае повторной передачи $\text{RTO} = \beta * \text{time-out}$



Самонастройка





Принципы самонастройки

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

Прочсть «Congestion Avoidance and Control» van Jacobson and Karels