第六章 传输层

TCP拥塞控制

TCP 拥塞控制

- 虽然网络层也试图管理拥塞，但是，大多数繁重的任务是由TCP来完成的，因为针对拥塞的真正解决方案是减慢数据率
- 分组守恒：当有一个老的分组离开之后才允许新的分组注入网络
- TCP希望通过动态维护窗口大小来实现这个目标

TCP 拥塞控制

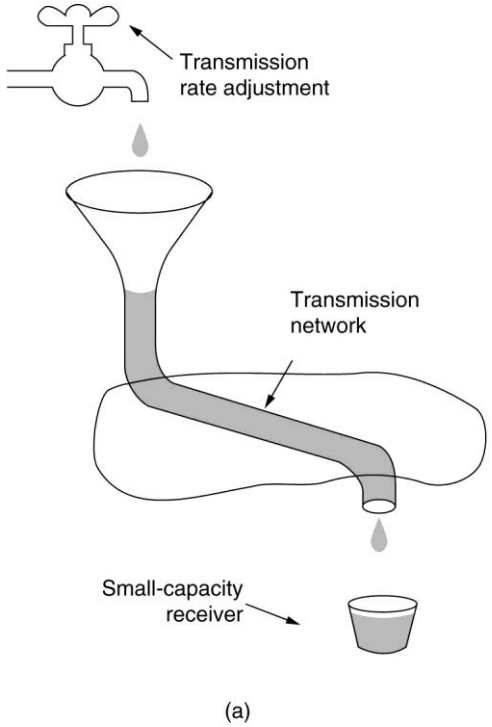
拥塞检测 （Congestion detection）

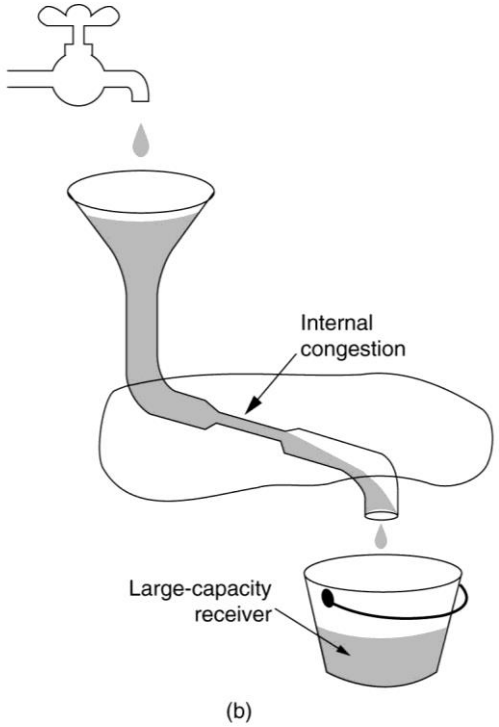
- 所有的互联网TCP算法都假定超时是由拥塞引起的，并且通过监视超时的情况来判断是否出现问题

拥塞控制 （Congestion prevention）

- 当一个连接建立的时候，双方选择一个合适的窗口大小，接收方根据自己的缓冲区大小来指定窗口的大小。
- 如果发送者遵守此窗口大小的限制，则接收端不会出现缓冲区溢出的问题，但可能由于网络内部的拥塞而发生问题

TCP 拥塞控制





(a) 快速的网络向小容量的接收方传输数据

(b) 慢速的网络向大容量的接受方传输数据

TCP 拥塞控制

- 互联网解决方案应该是认识到两个潜在的问题的：网络容量，接收者容量，然后单独地处理这两个问题
- 为此，每个发送者维护两个窗口：

接收者窗口	大小反映了目前窗口的容量 （容易控制）
拥塞窗口	大小反映了网络目前的容量（难于控制）

发送者发送的数据字节数是两个窗口中小的那个窗口数

决定拥塞窗口的大小

- 慢启动算法（Slow Start） (尝试的过程)：
  - 当连接建立的时候，发送者用当前使用的最大数据段长度初始化拥塞窗口，然后发送一个最大的数据段
  - 如果在定时器超期之前收到确认，则将拥塞窗口翻倍，然后发送两个数据段……直至超时（或达到接收方窗口的大小）
  - 确定出拥塞窗口的大小
    - 如：如果试图发送 4096 字节没有问题，但是发送8192字节的时候，超时没有收到应答，则拥塞窗口设为4096个字节

慢速启动算法

按指数增长趋势定义拥塞窗口大小cwnd

- 初始：cwnd0 = MaxSegL（当前数据段长度）
- 增长：cwnd1 = 2 cwnd0

  cwnd2 = 2 cwnd1

  • • •
- 截止：达到接收窗口大小或超时

慢速启动算法图例

拥塞窗口二进制指数增长至接收窗口大小或超时

TCP 拥塞控制

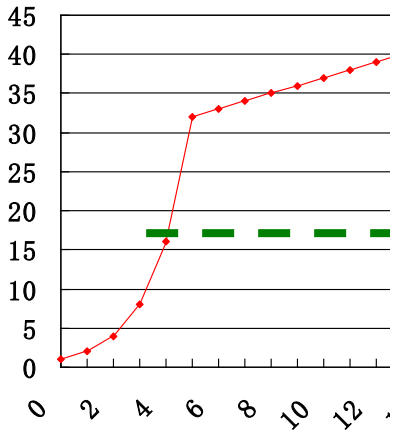
- 除了使用接收者窗口和拥塞窗口，TCP拥塞控制还是用了第三个参数，阈值（threshold），初始化为64K
- 当一个超时发生的时候，阈值降为当前拥塞窗口的一半，同时将拥塞窗口设为一个最大数据段的长度
- 使用慢启动算法来决定网络的容量，拥塞窗口增长到阈值时停止指数增长
- 从这个点开始，每次成功的传输都会让拥塞窗口线性增长（即每次仅增长一个最大的数据段长度）

拥塞控制算法-CWin指数增长

$Threshold_0$=64K

$Threshold_1$=32K

超时



指数增长

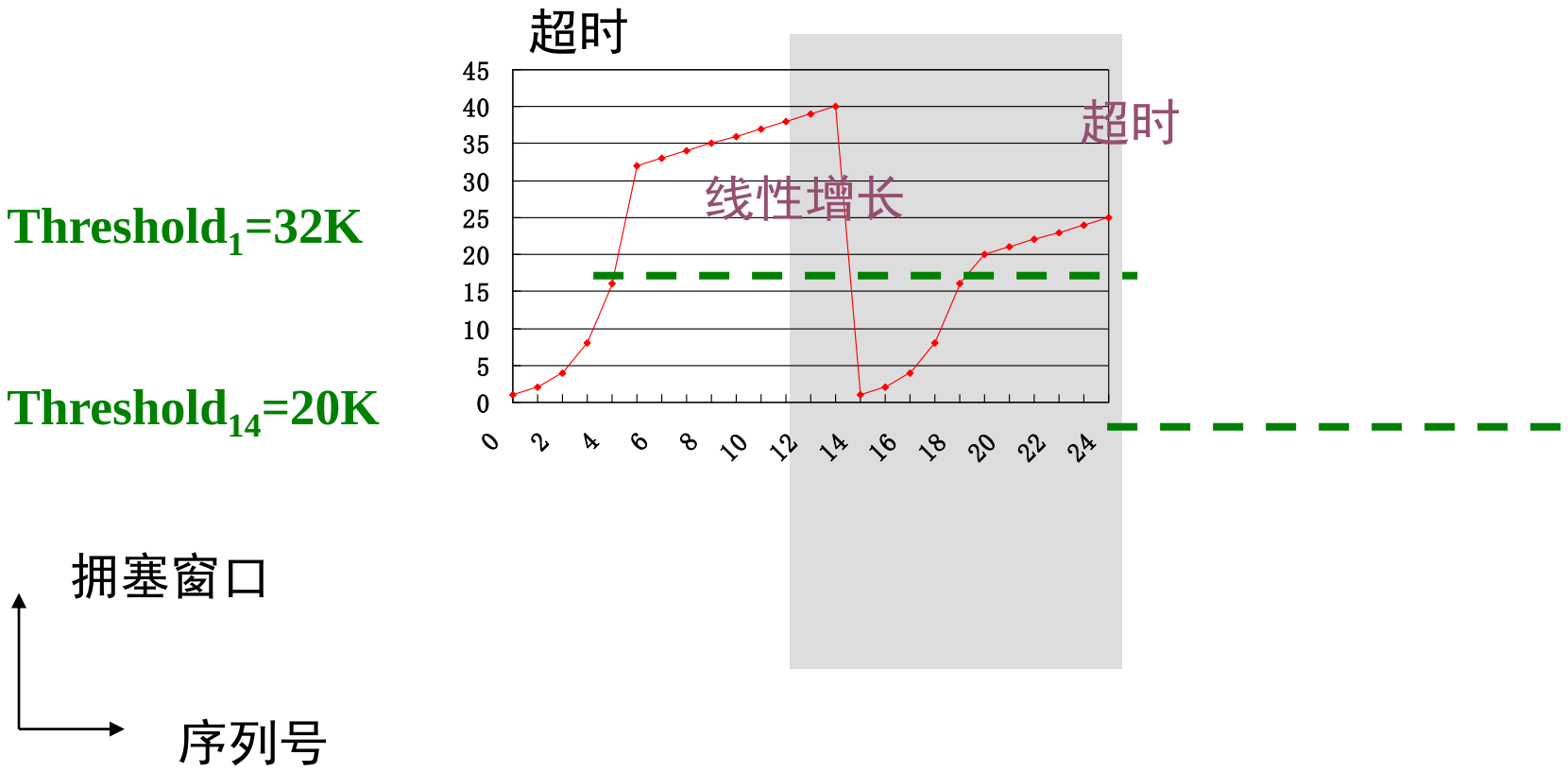
MaxSegL=1024

拥塞窗口

序列号

关键参数（临界值，接收窗口，拥塞窗口）

拥塞控制算法- cwnd线性增长



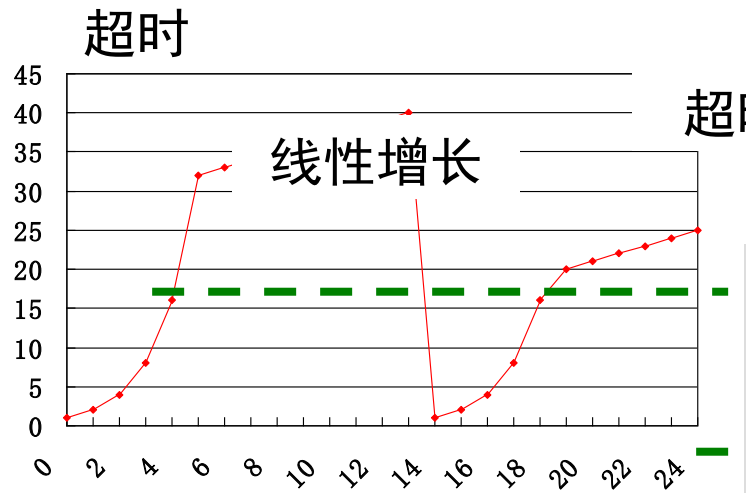
关键参数（临界值，接收窗口，拥塞窗口）

拥塞控制算法-重新慢速启动

$Threshold_0$=64K

$Threshold_1$=32K

$Threshold_{14}$=20K



MaxSegL=1024

线性增长

指数增长

常数？

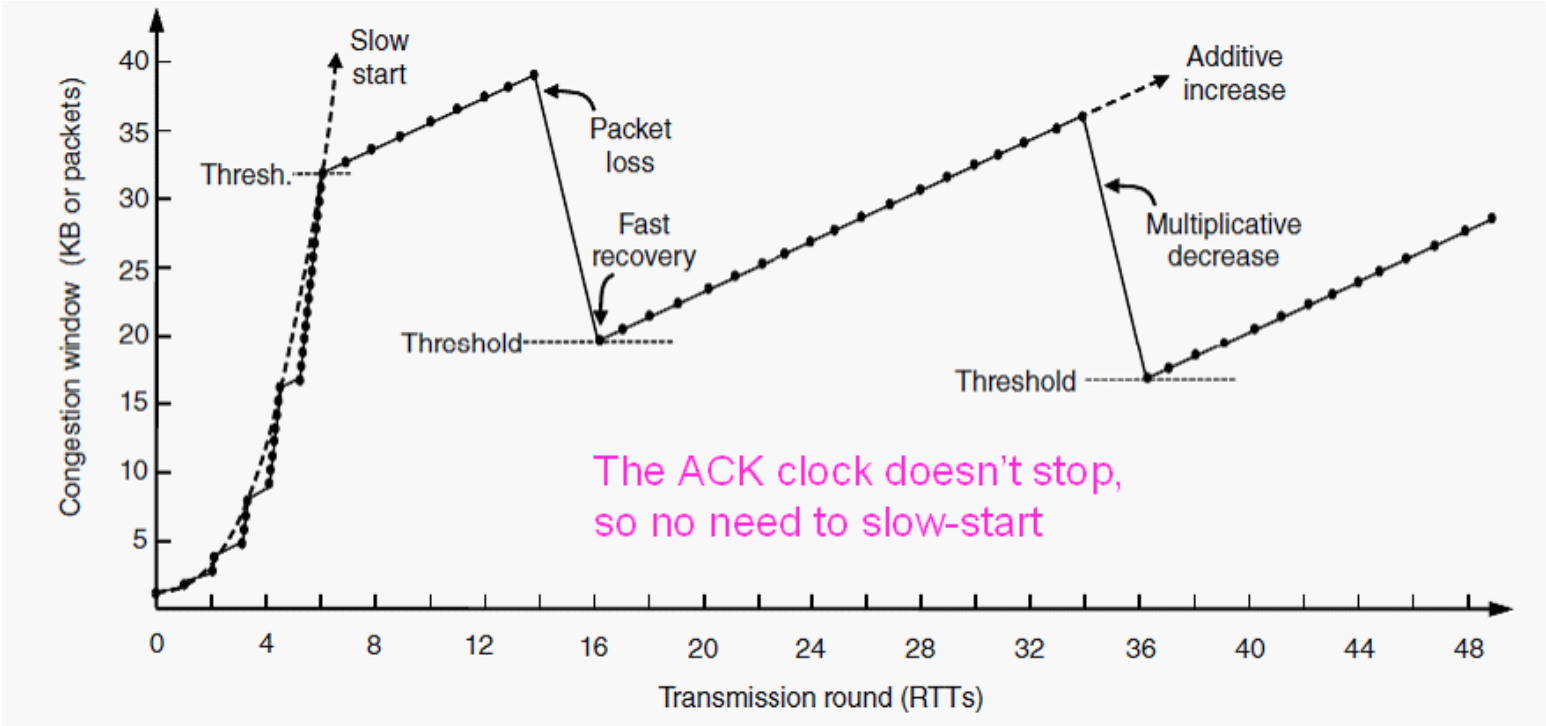
拥塞窗口

序列号

关键参数（临界值，接收窗口，拥塞窗口）

注意

快速恢复


Slow start
Packet loss
Fast recovery
Additive increase
Multiplicative decrease
Thresh.
Threshold
Threshold
The ACK clock doesn't stop, so no need to slow-start
Congestion window (KB or packets)
Transmission round (RTTs)
0
4
8
12
16
20
24
28
32
36
40
44
48
5
10
15
20
25
30
35
40

拥塞控制算法

- 定义初始拥塞窗口阈值和窗口大小

  $Threshold_0$ 和 $cwnd_0$

- 初始超时

  拥塞窗口阈值减半：Threshold1 = CWND / 2

- cwnd二进制指数增长至确认超时

拥塞控制算法

- cwnd线性增长至确认超时
  - 拥塞窗口值减半：$Threshold_n = CWND_n / 2$
  - 定义窗口大小：$cwnd = cwnd_0$
- 重新开始慢速启动过程

注意

- 如果收到一个ICMP抑制分组（ ICMP source quench）并被送给TCP传输实体，则这个事件被当作超时对待

小结

- TCP拥塞控制遵循分组守恒定律
- 两种因素引起拥塞警报
  - 接收方处理不过来
  - 通信子网中出现拥塞
- 处理拥塞的具体方法
  - Window size
  - Congestion window
- CWND通过慢起动方法尝试而来
- 通过阈值调节CWND尝试的精度

思考题

- TCP拥塞控制的原则是什么？
- 引起TCP拥塞的两种因素是什么？
- TCP怎么进行拥塞控制？
- 拥塞窗口怎么获得？
- 阈值有什么作用？
- 多次慢启动尝试，CWND是否会达到一个不变的常数？

谢谢观看

致谢

本课程课件中的部分素材来自于：（1）清华大学出版社出版的翻译教材《计算机网络》（原著作者：Andrew S. Tanenbaum，David J. Wetherall）；（2）思科网络技术学院教程；（3）网络上搜到的其他资料。在此，对清华大学出版社、思科网络技术学院、人民邮电出版社、以及其它提供本课程引用资料的个人表示衷心的感谢！

对于本课程引用的素材，仅用于课程学习，如有任何问题，请与我们联系！