案例学习之
TCP的速率调整算法

基于拥塞的TCP速率调整算法

TCP拥塞控制基本思想

- 当发生丢包事件时，降低发送速率（减少拥塞窗口CongWin的大小）
- 所有通过拥塞区域的发送端都将降低发送速率

  →注入拥塞路径的流量减少

  →由此缓解壅塞状况

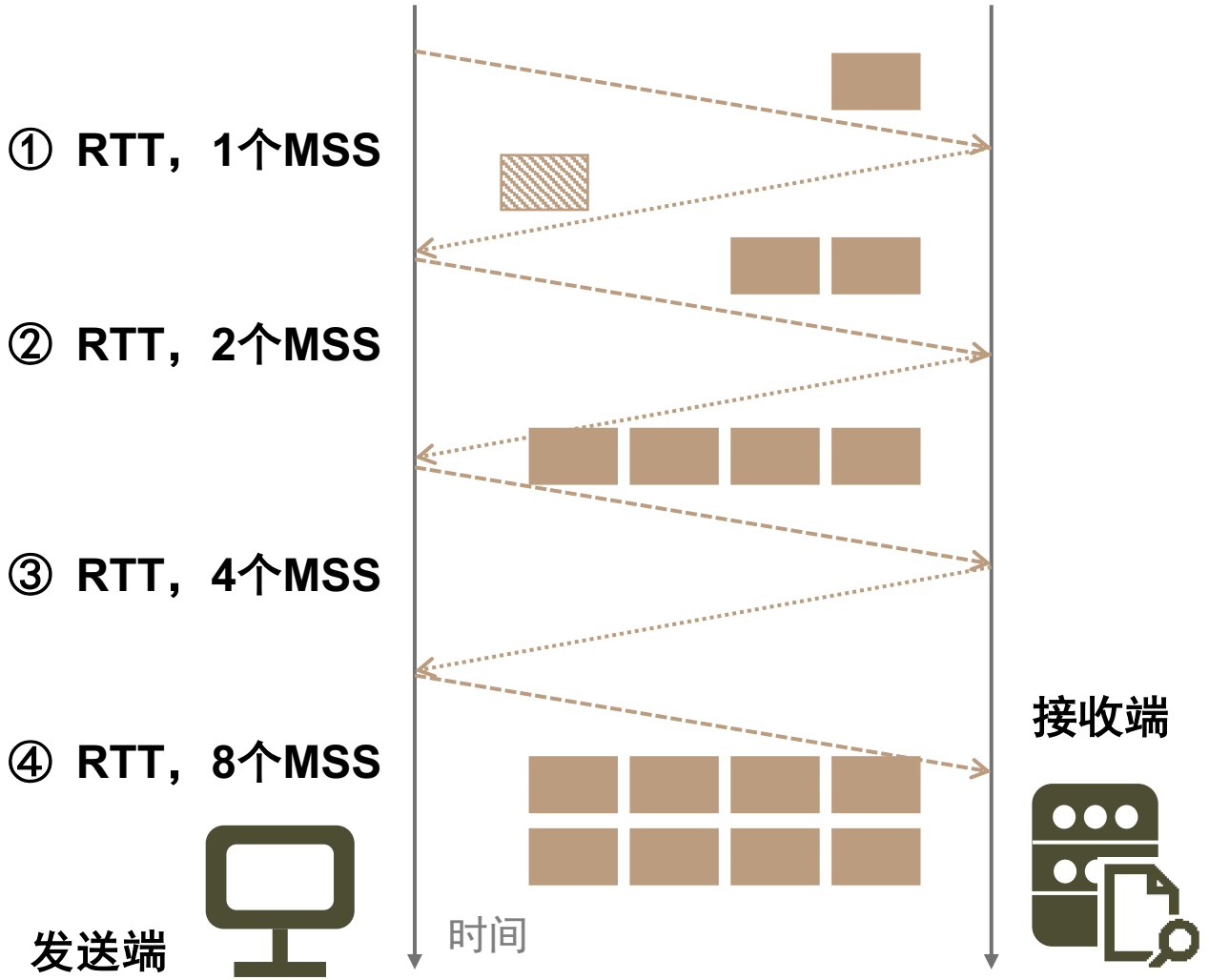
Q3 TCP发送端采用何种算法改变数据的发送速率？

慢速启动 (指数增长) + 拥塞控制 (线性增长) + 响应超时 (事件处理)

慢速启动过程

慢速启动过程

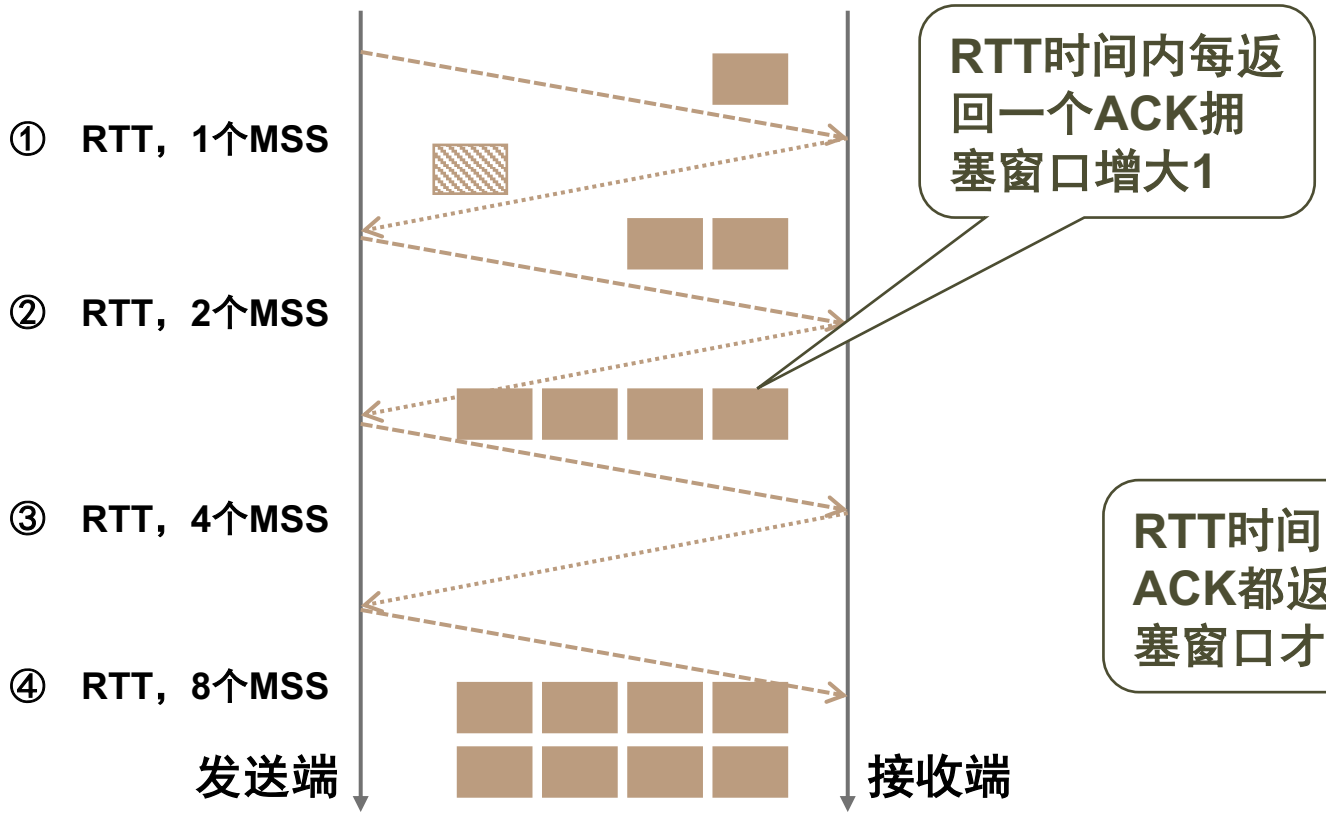
- TCP连接建立时，拥塞窗口CongWin初始为一个MSS
- 在初始阶段按指数增长速度加大发送速率直到发生“丢包事件”或拥塞窗口超过某个阈值
- 发生“丢包事件”后将CongWin窗口减半并按线性速度增大

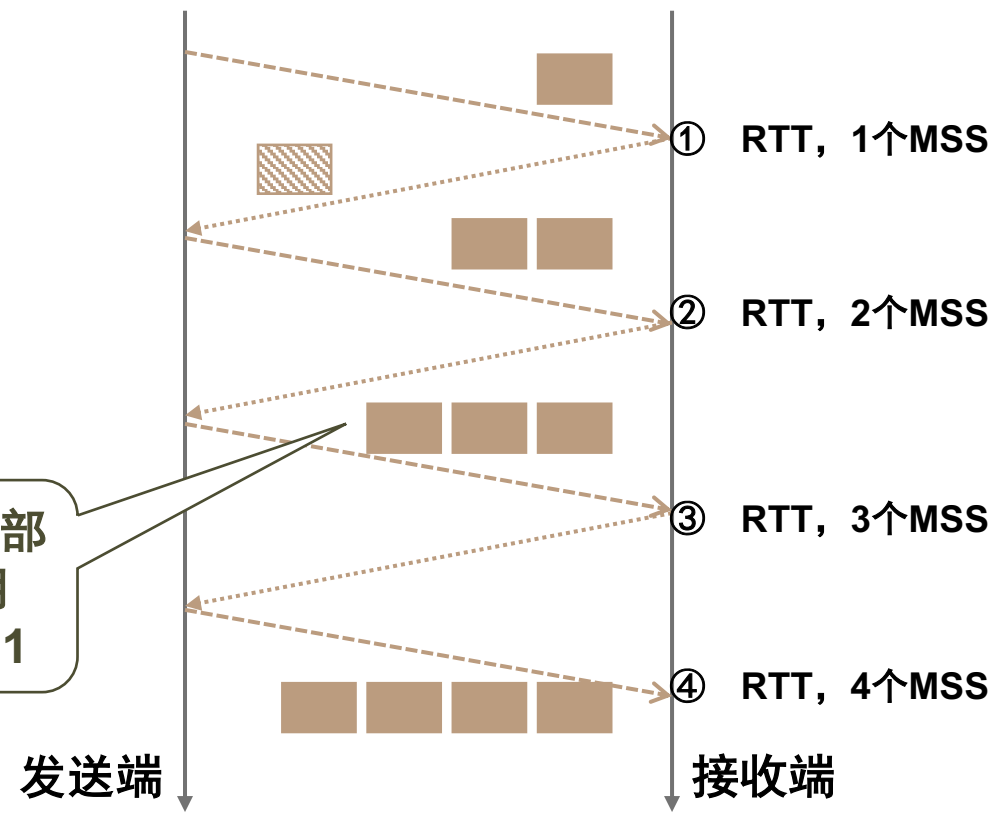


慢速启动vs.线性增长

慢速启动：初始CongWin=1，每次RTT内所有确认都返回后CongWin大小加倍

线性增长：初始CongWin=1，每次RTT内所有确认都返回后CongWin大小加大一个MSS





确认 TCP段

拥塞避免（AIMD算法）

拥塞避免：TCP拥塞控制协议的线性增加阶段。

- 原发送速率 = CongWin/RTT
- 新发送速率 = (CongWin+MSS) / RTT

TCP把“丢包”作为拥塞出现的主要依据，因此要增大未发送的TCP段的计时器，以免不必要的重传。

逐步递增

- 每当经过一个RTT就将CongWin窗口增大一个MSS

加倍递减

- 一旦发现丢失段立即将CongWin窗口减半
- 对于保留在发送窗口中的段将重传计时器的值加倍

如何响应“丢包事件”

慢速启动阈值：用来统一管理拥塞窗口

- 慢速启动阈值标志着慢速启动的结束和线性增长的开始
- 慢速启动阈值的初始值较大（例如64K，或流量控制窗口RcvWin大小）

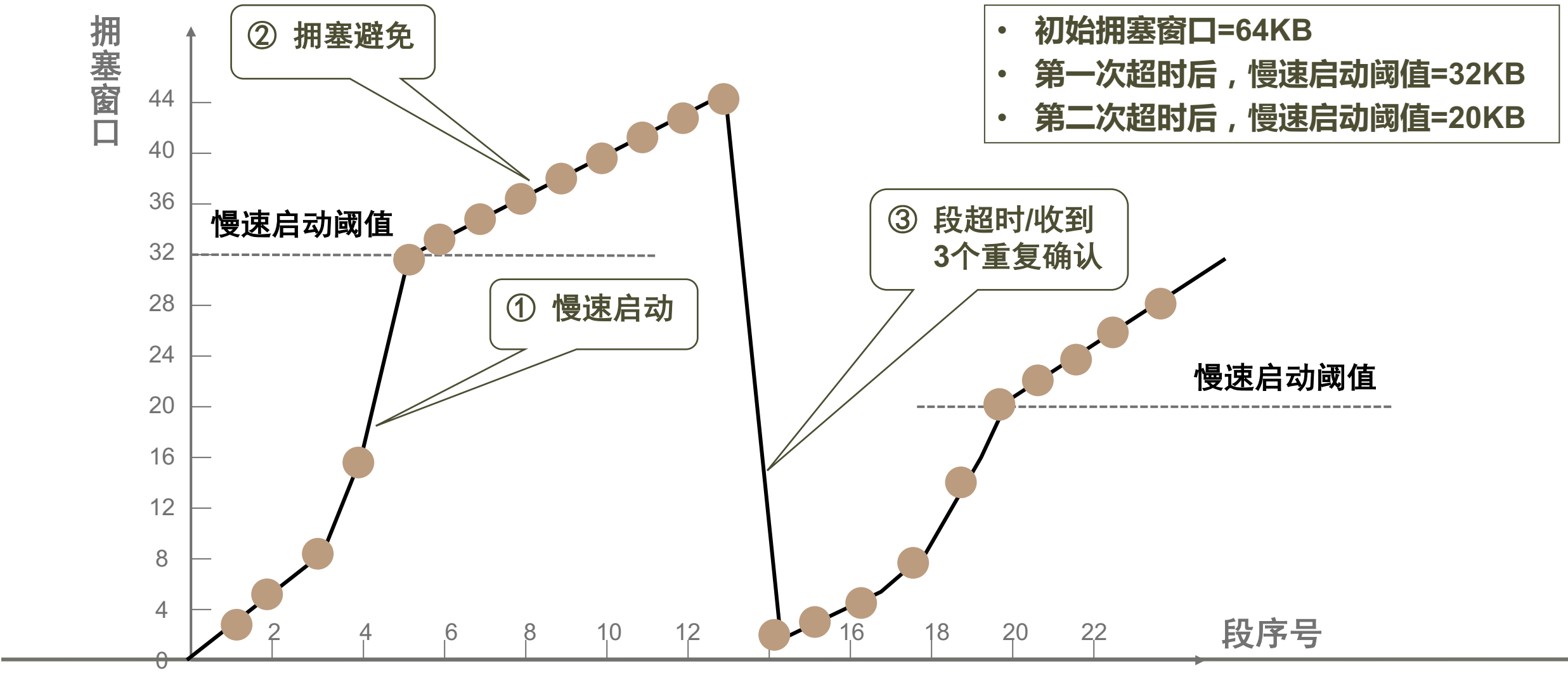
某个段超时

- 发送端进入“慢速启动”阶段，拥塞发送窗口CongWin置为1个MSS
- CongWin大小按指数增长直到达到慢速启动阈值
- CongWin大小按线性增长，进入拥塞避免阶段

三个重复ACK

- 拥塞窗口CongWin大小减半
- CongWin大小按线性增长

TCP拥塞窗口变化（TCP Tahoe版本）

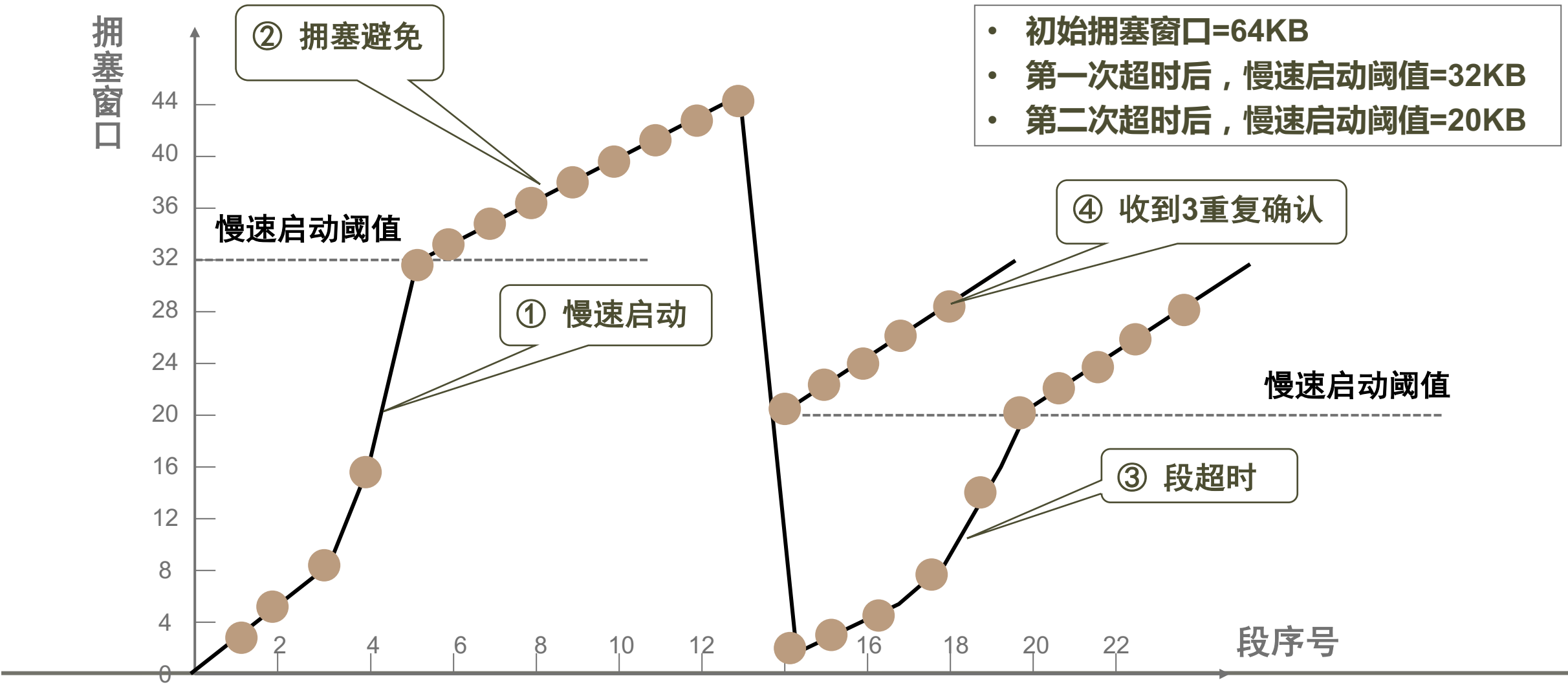


- 初始拥塞窗口=64KB
- 第一次超时后，慢速启动阈值=32KB
- 第二次超时后，慢速启动阈值=20KB

北京大学

TCP Tahoe是TCP的一个实现版本

TCP拥塞窗口变化（TCP Reno版本）



北京大学

TCP Reno是TCP的一个实现版本

TCP的快速恢复机制

快速恢复：收到重复ACK后，每收到一个ACK发送一个段，直到丢失的段得到确认，然后进入拥塞避免阶段。

- 如果TCP段传输路径发生拥塞，通常被丢弃的包不止一个
- 发送端连续收到3个重复确认，可以推测发生了传输错误而不是拥塞

示例1：段1所在的包被丢弃

- 接收端每收到一个段给发送端反馈一个重复确认
- 发送端每收到一个重复ACK就发送一个段

TCP拥塞控制算法概述

- 初始的发送速率 = CongWin/RTT
- 慢速启动：每个RTT后CongWin大小加倍
- 线性增长：每个RTT后CongWin加大一个MSS

拥塞窗口递增阶段

- 当拥塞窗口低于慢速启动阈值时，发送端进入慢速启动阶段，拥塞窗口CongWin大小按指数快速增大
- 当拥塞窗口CongWin大于慢速启动阈值时，发送端进入拥塞避免阶段，窗口CongWin大小按线性速度增大

- 拥塞丢包时急速减小发送速率
- 偶尔丢包时缓慢减小发送速率

“丢包”事件的响应

- 发生“ 3次重复ACK ”丢包事件时，慢速启动阈值置为当前拥塞窗口CongWin大小的一半，并把拥塞窗口CongWin设置成慢速启动阈值，进入拥塞避免阶段（窗口大小线性增长）
- 发生“超时”丢包事件时，慢速启动阈值设为当前拥塞窗口CongWin大小的一半，并把拥塞窗口CongWin设置成1个MSS，开始慢速启动过程

注：本算法基于Reno版本的TCP实现