计算机网络第5章作业参考答案

1、端口的作用是什么？为什么端口要划分为三种？

2、TCP如何实现端到端可靠性传输？

3、描述TCP连接建立的三握手过程和连接释放的四报文握手？

4、如何计算RTTs、RTT_D、RTO？

7、在TCP的拥塞控制中，什么是慢开始、拥塞避免、快重传、快恢复算法？

8、拥塞控制和流量控制的作用和区别？

5、第8版课后习题：5-33

分析：

根据RFC 2988建议，$RTO = RTTs + 4 \times RTT_D$

初次测量时，$RTTs(1) = RTT(1)$

$RTT_D(1) = RTT(1)/2$

后续测量时，

$$RTT_S(i) = (1-\alpha) \times RTT_S(i-1) + \alpha \times RTT(i) \qquad \alpha = 1/8$$

$$RTT_D(i) = (1-\beta) \times RTT_D(i-1) + \beta \times |RTTs(i) - RTT(i)| \qquad \beta = 1/4$$

5、第8版课后习题：5-33

解答(1):

RTO=6秒是人为初始设定的

测试出RTT样本值为 1.5秒，比6秒少，没有超时，所以这个RTT样本值是可用的，可以用来计算新的RTO，否则就得用修正算法了

$RTT(1) = 1.5\ s$

$RTTs(1) = 1.5\ s$

$RTT_D(1) = RTT(1)/2 = 0.75\ s$

$RTO(1) = RTTs(1) + 4 \times RTT_D(1)$

$= 1.5 + 4 \times 0.75$

$= 4.5\ s$

5、第8版课后习题：5-33

解答(2):

$RTTs(1) = 1.5\ s$

$RTT_D(1) = 0.75\ s$

$RTT(2) = 2.5\ s$

$RTTs(2) = (1 - \alpha) \times RTT_S(1) + \alpha \times RTT(2)$

$= (1 - 1/8) \times 1.5 + 1/8 \times 2.5 = 1.625\ s$

$RTT_D(2) = (1- \beta) \times RTT_D(1) + \beta \times |RTTs(2) - RTT(2)|$

$= (1 - 1/4) \times 0.75 + 1/4 \times |1.625 - 2.5| = 0.78125\ s$

$RTO(2) = RTTs(2) + 4 \times RTT_D(2)$

$= 1.625 + 4 \times 0.78125 = 4.75\ s$

5、第8版课后习题：5-34

解答：

$\alpha = 0.1$　　$RTT(1) = 30$ ms　　$RTTs(1) = 30$ ms

$RTT(2) = 26$ ms　　$RTT(3) = 32$ ms　　$RTT(4) = 24$ ms

$$RTTs(2) = (1-\alpha) \times RTT_S(1) + \alpha \times RTT(2)$$
$$= (1-0.1) \times 30 + 0.1 \times 26 = 29.6 \text{ ms}$$

$$RTTs(3) = (1-\alpha) \times RTT_S(2) + \alpha \times RTT(3)$$
$$= (1-0.1) \times 29.6 + 0.1 \times 32 = 29.84 \text{ ms}$$

$$RTTs(4) = (1-\alpha) \times RTT_S(3) + \alpha \times RTT(4)$$
$$= (1-0.1) \times 29.84 + 0.1 \times 24 = 29.256 \text{ ms}$$

RTT测量样本值的变化幅度可以超过20%，但是加权平均后平滑的往返时间RTTs的变化幅度最多只有0.1%

6、在建立TCP连接时，发送方设定超时重传时间是RTO =2.2s，且上一次测量计算的RTTs值为1.4s。当发送方接到对方的连接确认报文段时，测量出RTT样本值为1.5s，试计算现在的RTO值

根据RFC 2988建议，$RTO = RTTs + 4 \times RTT_D$

初次测量时，$RTTs(1) = RTT(1)$

$RTT_D(1) = RTT(1)/2$

后续测量时，

$$RTT_S(i) = (1-\alpha) \times RTT_S(i-1) + \alpha \times RTT(i) \qquad \alpha = 1/8$$

$$RTT_D(i) = (1-\beta) \times RTT_D(i-1) + \beta \times |RTTs(i) - RTT(i)| \qquad \beta = 1/4$$

解答：

RTO(1) = 2.2 s　　RTTs(1) = 1.4 s　　RTT(2) = 1.5 s

RTTs(2) = (1 – α) × RTT_S(1) + α × RTT(2)

= (1 – 1/8) ×1.4 + 1/8 × 1.5 = 1.4125 s

RTO(1) = RTTs (1) + 4×RTT_D(1)

= 1.4 + 4×RTT_D(1) = 2.2 s

=> RTT_D(1) = 0.2 s

RTT_D(2)= (1- β) × RTT_D(1) + β × | RTTs(2) –RTT(2)|

= (1 – 1/4)× 0.2 + 1/4 ×| 1.4125 – 1.5| = 0.171875 s

RTO(2) = RTTs(2) + 4×RTT_D(2)

= 1.4125 + 4×0.171875 = 2.1 s

9、若采用滑动窗口机制对于两个相邻接点A（发送方）和B（接收方）的通信过程进行流量控制。假定帧的序号长度为3个二进制位，发送窗口和接受窗口的大小都是7，当A发送了编号为0、1、2、3这4个帧后，而B接受了这4个帧，但仅应答了0、1两个帧。请问：此时，A的发送窗口将要发送的帧序号为哪些？此时，B的接收窗口内可能的最大帧序号为多少？

解答：帧序号长度为3位，则帧的序号为0，1，…，7。发送窗口大小为7，则初始窗口内的帧序号为0，1，2，3，4，5，6。窗口中的帧是在接收到B确认前，可以连续发送的帧序号。

A的初始窗口：　0 1 2 3 4 5 6

A连续发送序号为0、1、2、3的4个帧后，窗口状态如下：

连续发送4个帧后的A窗口：　0 1 2 3 4 5 6

B收到了4个帧后，确认了0、1两个帧，则A的发送窗口将移出0、1两个帧序号，移入序号7和新的序号0，即此时发送窗口的序号为2、3、4、5、6、7、0（新）等。其中，序号为2、3的帧是已经发送并等待确认的，序号为4、5、6、7、0的帧是可以发送还没有发送的。

收到0、1帧确认后的A窗口：　0 1 2 3 4 5 6 7 0

此时，将要发送的帧分两种情况：

（1） 如果未发生超时，4、5、6、7、0是将要发送的帧。

（2）如果发生了超时，则2、3将会重传，那么将要发送的帧序号为 2、3、4、5、6、7、0

此时，3是A已经发送出去的最大帧序号，因此，B的接收窗口内可能的最大帧序号为3（上一轮确认前可能发出的最大序号），但此后B的接收窗口内可能的最大帧序号为7。

10、设TCP 的ssthresh的初始值为8(单位为报文段)。当拥塞窗口上升到12时该网络发生了超时，TCP 使用慢开始和拥塞避免。

试分别求出第1 次到第15 次传输的各拥塞窗口大小并说明拥塞控制窗口每一次变化的原因。

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
cwnd															

解答：

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
cwnd	1	2	4	8	9	10	11	12	1	2	4	6	7	8	9

- TCP使用慢开始，第一次发送1段，窗口大小为1；
- 随后按2的指数增长，增长到ssthresh的初始值8，需要经过$\log_2 8=3$次，即第4次；
- 随后进入第一轮拥塞避免算法，按线性增长到12，需要12-8=4次，即第8次；
- 此时，发生超时将开始新一轮的慢开始，窗口重新设置为1，同时新的ssthresh值更新为12/2=6；
- 新一轮慢开始阶段由1按指数增长到大于6，需要3次（$2^3=8>6$），即发生超时后的第4次，总第8+4=12次。
- 进入第二轮拥塞避免，窗口值由新的ssthresh值6开始线性增长，传输到第15次时，线性增长了15-12=3次，此时窗口值为6+3=9。

11、设TCP 的ssthresh的初始值为8(单位为报文段)。采用Reno TCP拥塞控制方法，当拥塞窗口上升到12时收到 3 个重复的确认，在第13次传输后发生了超时，试分别求出第1 次到第15 次传输的各拥塞窗口大小，给出不同算法的执行阶段，说明不同阶段的ssthresh门限值的大小。

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
cwnd															

解答：

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
cwnd	1	2	4	8	9	10	11	12	6	7	8	9	10	1	2

- TCP使用慢开始，第一次发送1段，窗口大小为1；
- 随后按2的指数增长，增长到ssthresh的初始值8，需要经过$\log_2 8=3$次，即第4次；
- 随后进入第一轮拥塞避免算法，按线性增长到12，需要12-8=4次，即第8次；
- 此时，收到 3 个重复的确认，窗口cwnd和ssthresh值更新为原来cwnd的一半，即12/2=6，开始快重传和快恢复算法；
- 进入新一轮拥塞避免阶段，按线性增长到10；
- 在第13次传输后发生超时，开始新一轮的慢启动，窗口重新设置为1，同时新的ssthresh值更新为10/2=5；
- 新一轮慢启动阶段，在第15次传输时由1按指数增长到2。

12、通信信道带宽为1Gb/s，端到端时延为10ms。TCP的发送窗口为65535字节。试问：网络可能达到的最大吞吐量是多少?信道的利用率是多少?

（1）不考虑协议首部的封装，直接把数据发送到信道上

信道上发送的数据长度L为

$L = 65,535 \text{ Byte} = 65,535 \times 8 \text{ bit} = 524,280 \text{ bit}$

发送时间 $= 524,280 \text{ bit} \div 10^9 \text{ bit/s}$

$\approx 5.24 \times 10^{-4} \text{ s}$

$= 0.524 \text{ ms}$

往返时延 $= 2 \times 10 \text{ ms} = 20 \text{ ms}$

- TCP采用确认重传机制，成功的数据发送过程包括TCP报文段的发送和确认两个部分，缺一不可。
- 在连续AQR协议和累积确认机制下，在可能达到的最大吞吐量情况下，网络时延最小可以按照(发送时延+往返时延)来计算。

网络可达最大吞吐量=发送的数据 / 网络时延

= 发送的数据 / (发送时延+往返时延)

= 524,280 b/ (0.524+20) ms

= 0.524280 Mb / 20.524 ms

≈ 25.544 Mb/s

信道利用率 = 数据发送时延/占用信道时间
= 发送时延/ (发送时延+往返时延)
= 0.524/20.524
≈ 2.55%

或　信道利用率 = 吞吐量/信道带宽
= 25.544(Mb/s) / 1 (Gb/s)
≈ 2.55%

通信信道带宽为1Gb/s，端到端时延为10ms。TCP的发送窗口为65535字节。试问：网络可能达到的最大吞吐量是多少?信道的利用率是多少?

（2）考虑下层协议封装，但是不考虑数据链路层

数据长度 $L = (65535+20+20)\times 8 = 524600$ bit

带宽 $C = 10^9$ b/s

发送时间 $T_s = L/C = 0.0005246$ s

端到端时延 $T_d = 10\times 10^{-3}$ s

Throught $= L/(T_s + 2\times T_d) = 524600/0.0205246$

≈ 25.5 Mb/s

Efficiency $= T_s/(T_s + 2\times T_d) \approx 2.55\%$

通信信道带宽为1Gb/s，端到端时延为10ms。TCP的发送窗口为65535字节。试问：网络可能达到的最大吞吐量是多少?信道的利用率是多少?

进一步讨论：

IP数据报的最大总长度为65535字节

那么TCP报文段的数据部分最大为65495字节

当TCP的发送窗口为65535字节时，至少需要分成两个TCP报文段进行发送，那么在IP层上向信道发送的数据长度L应该增加两个IP首部和两个TCP首部，也就是L=65535+2×(20+20)=65615字节。

按照流水发送和累积确认方式，则下一步的计算方法同前。

通信信道带宽为1Gb/s，端到端时延为10ms。TCP的发送窗口为65535字节。试问：网络可能达到的最大吞吐量是多少?信道的利用率是多少?

（3）考虑所有的下层协议封装，数据链路层为802.3MAC帧

MAC帧的最大长度为 6+6+2+1500+4 =1518 字节。

再加7个字节的前同步码，1个字节的帧开始定界符，信道上允许发送的数据长度L为1526 字节。

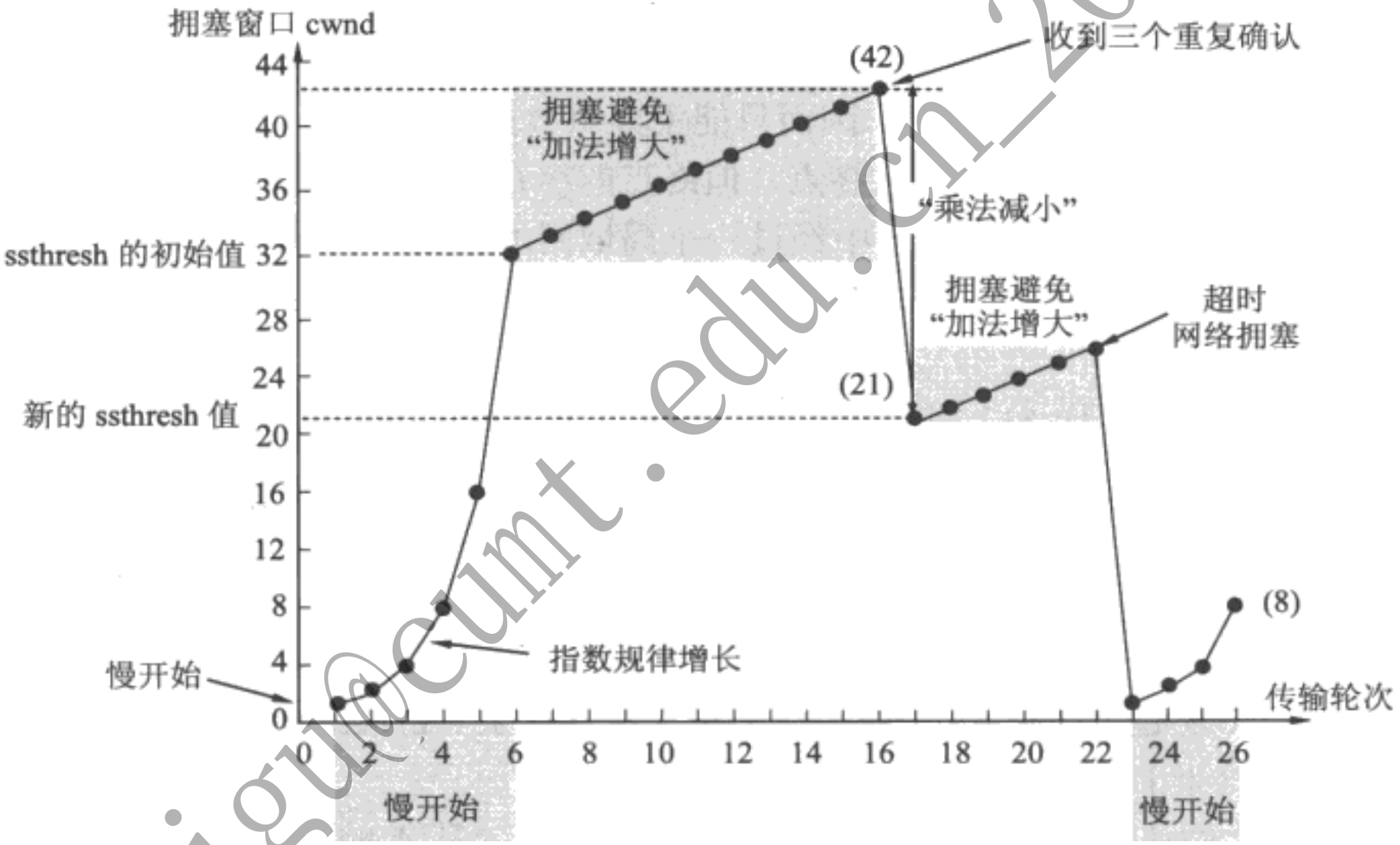
如果再考虑曼切斯特编码会导致带宽效率下降一半，那吞吐量的估算就更繁琐了。

显然，如此复杂的情况与题目关注的网络吞吐量的初衷是不符的，可以不考虑。

第8版课后习题 5-38，15个轮次拥塞窗口大小及变化原因

轮次	拥塞窗口	拥塞窗口变化的原因
1	1	网络发生了超时，TCP 使用慢开始算法
2	2	拥塞窗口值加倍
3	4	拥塞窗口值加倍
4	8	拥塞窗口值加倍，这是 ssthresh 的初始值
5	9	TCP 使用拥塞避免算法，拥塞窗口值加 1
6	10	TCP 使用拥塞避免算法，拥塞窗口值加 1
7	11	TCP 使用拥塞避免算法，拥塞窗口值加 1
8	12	TCP 使用拥塞避免算法，拥塞窗口值加 1
9	1	网络发生了超时，TCP 使用慢开始算法
10	2	拥塞窗口值加倍
11	4	拥塞窗口值加倍
12	6	拥塞窗口值加倍，但到达 12 的一半时，改为拥塞避免算法
13	7	TCP 使用拥塞避免算法，拥塞窗口值加 1
14	8	TCP 使用拥塞避免算法，拥塞窗口值加 1
15	9	TCP 使用拥塞避免算法，拥塞窗口值加 1

第8版课后习题 5-39，拥塞窗口与传输轮次的关系曲线



第8版课后习题 5-39

(2) 慢开始时间间隔：[1, 6]和[23, 26]。

(3) 拥塞避免时间间隔：[6, 16]和[17, 22]。

(4) 在第 16 轮次之后发送方通过收到三个重复的确认，检测到丢失了报文段，因为题目给出，下一个轮次的拥塞窗口减半了。

在第 22 轮次之后发送方是通过超时检测到丢失了报文段，因为题目给出，下一个轮次的拥塞窗口下降到 1 了。

(5) 在第 1 轮次发送时，门限 ssthresh 被设置为 32，因为从第 6 轮次起就进入了拥塞避免状态，拥塞窗口每个轮次加 1。

在第 18 轮次发送时，门限 ssthresh 被设置为发生拥塞时拥塞窗口 42 的一半，即 21。

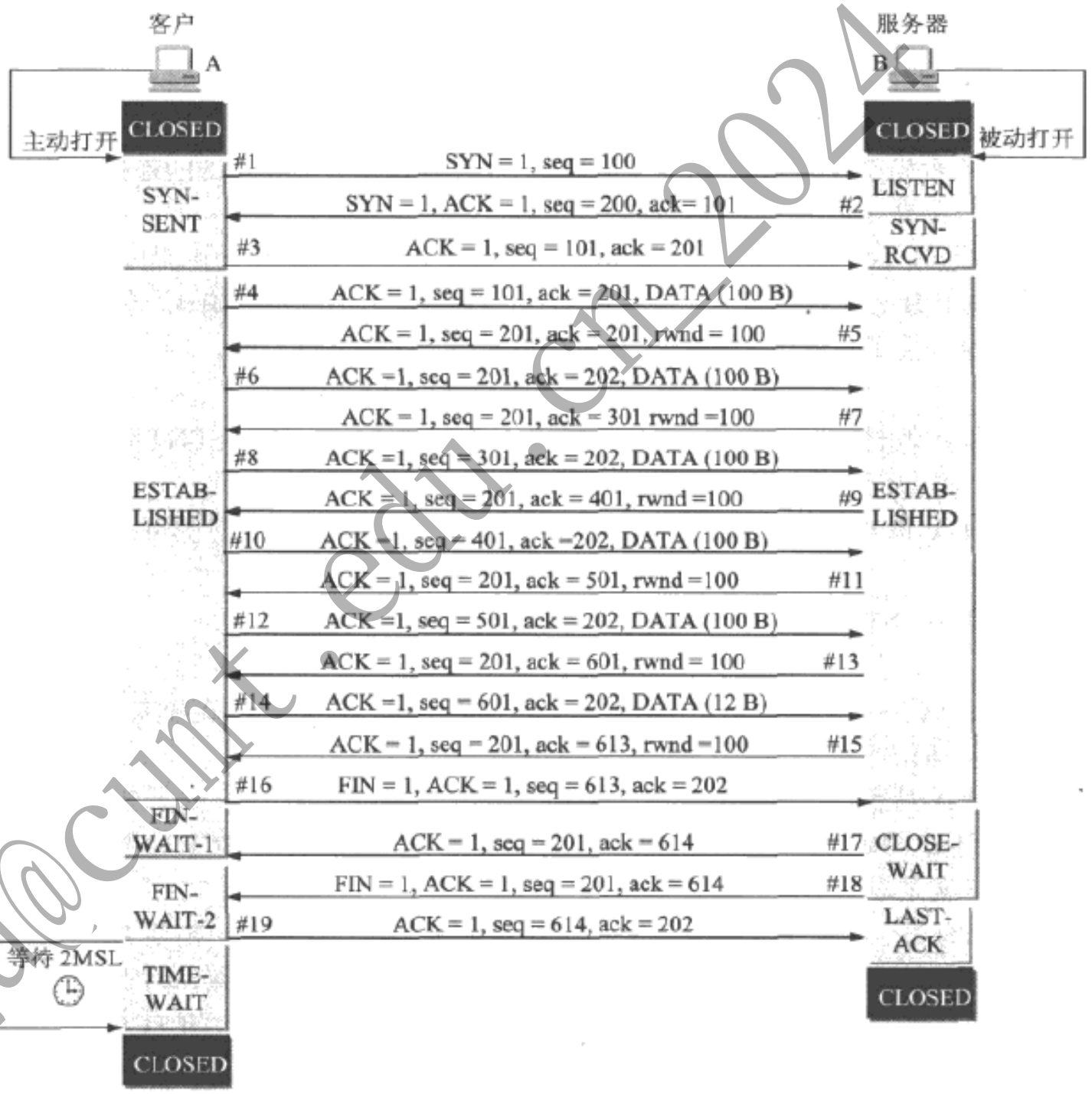
在第 24 轮次发送时，门限 ssthresh 被设置为发生拥塞时拥塞窗口 26 的一半，即 13。

第8版课后习题 5-39

(6) 第 1 轮次发送报文段 1。(cwnd = 1)
第 2 轮次发送报文段 2, 3。(cwnd = 2)
第 3 轮次发送报文段 4 ~ 7。(cwnd = 4)
第 4 轮次发送报文段 8 ~ 15。(cwnd = 8)
第 5 轮次发送报文段 16 ~ 31。(cwnd = 16)
第 6 轮次发送报文段 32 ~ 63。(cwnd = 32)
第 7 轮次发送报文段 64 ~ 94。(cwnd = 33)
因此第 70 报文段在第 7 轮次发送出。

(7) 检测出了报文段的丢失时拥塞窗口 cwnd 是 8，因此拥塞窗口 cwnd 的数值应当减半，等于 4，而门限 ssthresh 应设置为检测出报文段丢失时拥塞窗口 8 的一半，即 4。

第8版课后习题 5-41



说明：

- 左边是客户端，右边是服务器
- 因为服务器端是独立确认，所以不消耗序号。
- 客户端对服务器端数据的捎带确认，所以确认序号也保持不变
- #4的 ack=201，是客户端对 #2 的服务器端seq=200 的确认，所以是 201
- 在三握手过程中，服务器会消耗一个序号，所以#5 中的服务器的seq会变成201
- 因为，#5 的服务器端发送的是独立确认，不消耗序号
- 所以，#7 的服务器端的序号seq 保持为 201
- #6和#8分别是对#5和#7的捎带确认
- 因为#5和#7的 seq是 201，所以 #6和#8的 ack 就是 202
- 后续依次如此