A

姓名： wissen 班级： w24 学号： 921080

《计算机网络》2006年期末试题、参考答案及评阅建议

以下参考解答包含比较详细的分析和说明，并非所有信息都是答卷所要求的。

（2006年12月31日 19:20 – 21:20）

一. (对协议的认识和理解)［共40分］以下是在两个主机之间用TCP/IP通信时在其中一方，即202.38.118.4上使用tcpdump得到的一部分结果，共计有19个顺序听到的分组，其中的序列号都已经处理成绝对数值，并去掉了tcpdump加上的本地时刻信息。根据这些结果，回答后面的问题（问题从试卷第3页开始）。

(1) IP 221.222.79.11.62159 > **202.38.118.4.80:** S 3752965856:3752965856(0) win 65535 <mss 1440,nop,wscale 0,nop,nop,timestamp 507829845 0>

(2) IP **202.38.118.4.80** > 221.222.79.11.62159: S 1932173887:1932173887(0) ack 3752965857 win 5792 <mss 1460,nop,nop,timestamp 184285706 507829845,nop,wscale 2>

(3) IP 221.222.79.11.62159 > **202.38.118.4.80:** . ack 1932173888 win 65535 <nop,nop,timestamp 507829846 184285706>

(4) IP 221.222.79.11.62159 > **202.38.118.4.80:** P 3752965857:3752966277(420) ack 1932173888 win 65535 <nop,nop,timestamp 507829846 184285706>

(5) IP **202.38.118.4.80** > 221.222.79.11.62159: . ack 3752966277 win 1716 <nop,nop,timestamp 184285748 507829846>

(6) IP **202.38.118.4.80** > 221.222.79.11.62159: . 1932173888:1932175316(1428) ack 3752966277 win 1716 <nop,nop,timestamp 184285918 507829846>

(7) IP **202.38.118.4.80** > 221.222.79.11.62159: . 1932175316:1932176744(1428) ack 3752966277 win 1716 <nop,nop,timestamp 184285918 507829846>

(8) IP **202.38.118.4.80** > 221.222.79.11.62159:
. 1932173888:1932175316(1428) ack 3752966277 win
1716 <nop,nop,timestamp 184286044 507829846>

(9) IP **202.38.118.4.80** > 221.222.79.11.62159:
. 1932173888:1932175316(1428) ack 3752966277 win
1716 <nop,nop,timestamp 184286296 507829846>

(10) IP 221.222.79.11.62159 > **202.38.118.4.80:**
. ack 1932176744 win 65535 <nop,nop,timestamp
507829850 184286296>

(11) IP **202.38.118.4.80** > 221.222.79.11.62159:
. 1932176744:1932178172(1428) ack 3752966277 win
1716 <nop,nop,timestamp 184286348 507829850>

(12) IP 221.222.79.11.62159 > **202.38.118.4.80:**
. ack 1932178172 win 65535 <nop,nop,timestamp
507829851 184286348>

(13) IP **202.38.118.4.80** > 221.222.79.11.62159:
. 1932178172:1932179600(1428) ack 3752966277 win
1716 <nop,nop,timestamp 184286443 507829851>

(14) IP 221.222.79.11.62159 > **202.38.118.4.80:**
. ack 1932179600 win 65535 <nop,nop,timestamp
507829852 184286443>

(15) IP **202.38.118.4.80** > 221.222.79.11.62159:
. 1932179600:1932181028(1428) ack 3752966277 win
1716 <nop,nop,timestamp 184286534 507829852>

(16) IP **202.38.118.4.80** > 221.222.79.11.62159:
FP 1932181028:1932181888(860) ack 3752966277 win
1716 <nop,nop,timestamp 184286534 507829852>

(17) IP 221.222.79.11.62159 > **202.38.118.4.80:**
. ack 1932181889 win 64828 <nop,nop,timestamp
507829852 184286534>

(18) IP 221.222.79.11.62159 > **202.38.118.4.80:**
F 3752966277:3752966277(0) ack 1932181889 win 65535
<nop,nop,timestamp 507829852 184286534>

(19) IP **202.38.118.4.80** > 221.222.79.11.62159:
. ack 3752966278 win 1716 <nop,nop,timestamp
184286581 507829852>

1. 对于上述摘要式的表达，试将上面摘要 (1) 翻译成用十六进制表示的 IP 分组。为简单起见，有的信息已经给出，已知IP头不包含选项，而阴影格子表示整个分组已经结束。请填写出其它的部分，每个格子填写一个十六进制数（请填写清晰，明显模棱两可的字迹不算对）。(10分)

表1

4	5	0	0	0	0	3	c	0	0	0	0	4	0	0	0
3	0	0	6	d	d	a	7	d	d	d	e	4	f	0	b
c	a	2	6	7	6	0	4	f	2	c	f	0	0	5	0
d	f	b	1	b	6	e	0	0	0	0	0	0	0	0	0
a	0	0	2	f	f	f	f	5	7	b	b	0	0	0	0
0	2	0	4	0	5	a	0	0	1	0	3	0	3	0	0
0	1	0	1	0	8	0	a	1	e	4	4	d	e	5	5
0	0	0	0	0	0	0	0								

说明：本题考查对tcpdump及TCP和IP两个协议报文头结构的熟悉程度。

(1) 要求能够在tcpdump中正确定位IP头及TCP头的关键字段及其包含的信息；

(2) 要求正确认识IP地址和端口的表示方法，能够在十六进制和十进制表示法之间互换；

(3) 能够根据tcpdump给定信息确定头长度，对于本题来说，由于dump下来的数据长度为0，所以IP头以后的全部内容都是TCP头，共10个32-bit字，因此TCP头长度字段应为0xa；

(4) 能够识别标志位，本题dump结果带有S（SYN）标志，而ACK无效，故标志位部分为0x02；

(5) 对于序列号的转换成十六进制问题，不少同学抱怨计算器无力处理这么大的整数，实在很抱歉；但原则上这不是个大问题，手工计算并不复杂，使用短除法取每步余数是转换进制的基本功，不应该忘记，注意最后的结果倒过来写：

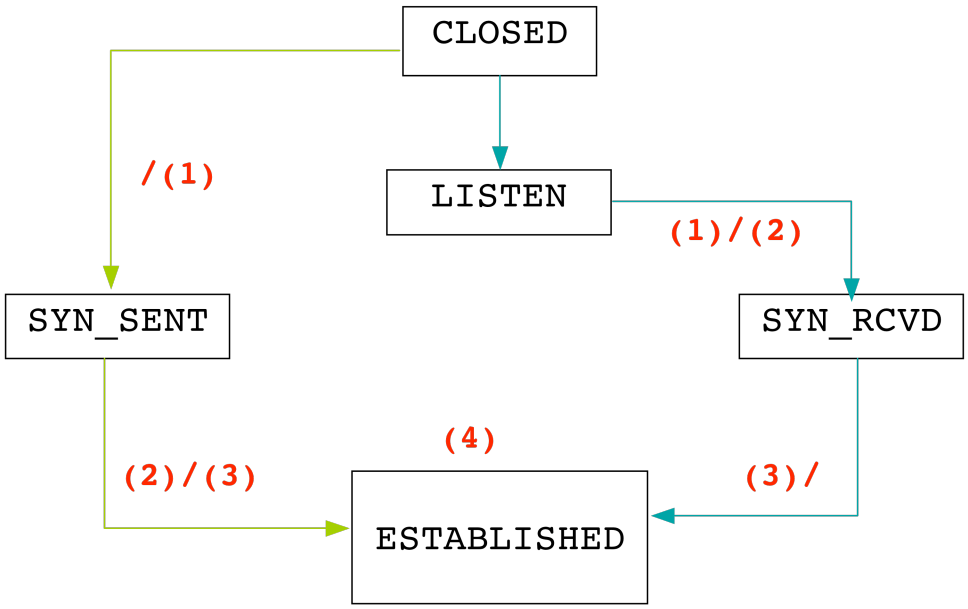
```
16 |3752985856
   |234560366   ... 0
    |14660022   ... 14(e)
     |916251    ... 6
      |57265    ... 11(b)
       |3579    ... 1
        |223    ... 11(b)
         |13    ... 15(f)
           0    ... 13(d)
```

2. 一般来说，IP头中的TTL的初始值总是设置为2的整数次幂，具体设置为多大，跟不同的操作系统有关。根据以上分组 (1) 的细节，试猜测从221.222.79.11到202.38.118.4之间经过了多少个路由器？(4分)

TTL变成0x30 (48)，比较大的可能是由64经过16跳；除非走过很长的环路，由128经过80跳或由255经过207跳的可能性微乎其微。

说明：本题考查对TTL的理解。

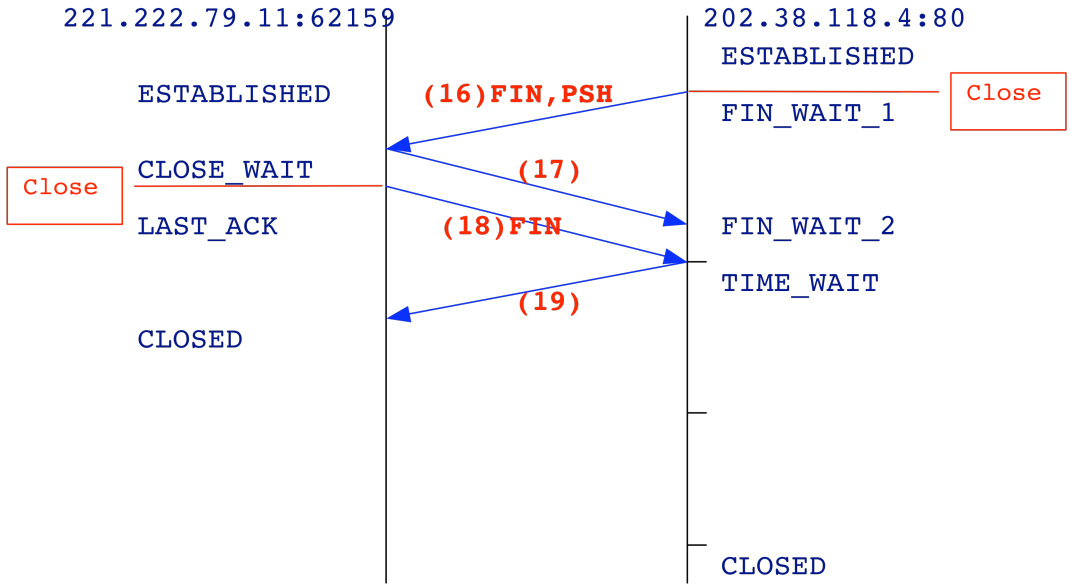
3. 画出TCP建立连接的状态转移示意图，并在图中标示出分组 (1) – (4) 所处的位置。(6分)



说明：本题考查对TCP建立连接过程中状态变化的理解。严格的说，各分组在标注中应体现是作为输入还是输出出现的。在状态转移图上，(input)/(output) 是个关于状态转移过程的很标准的写法。评阅试卷时对考生是否写对输入 / 输出的次序不作要求；状态转移过程中只写了输入没写输出

的，也不作追究；但只写了输出没写输入的，则不认为正确。如果把TCP状态图中其它与本题无关的状态转移途径也画了出来，并不能加分。

4. 画出TCP关闭连接的时序图，在图中标明有关分组的编号（不需要标明TCP协议头里的序列号），并注明有关分组到达时通信两端状态的变化。(6分)



说明：本题考查对TCP拆除连接过程中状态变化的理解。

5. 根据以上的记录，从应用层来看，你猜测谁是客户端、谁是服务器；客户发给服务器的请求消息和服务器发给客户的应答消息分别有多少个字节？写出你的理由和计算式子。(7分)

221.222.79.11是客户，202.38.118.4是服务器，因为一般主动发起连接的一方是客户。客户发给服务器的请求消息共计3752966277 – 3752975857 = 420 bytes；服务器发给客户的应答消息共计 1932181888 – 1932173888 = 8000 bytes。

计算分别使用双方第一字节的序列号及关闭时最后一字节之后一个序列号做减法。

如果采用TCP payload长度加和的方法，则应注意不把重传报文做重复的计算。

说明：本题考查对TCP序列号与应用层数据之间关系的理解，特别考查对SYN及FIN要占一个序列号的了解。

6. (7分) TCP的时间戳 (timestamp) 选项用来表示一个分组发出的时间，接收方收到这个分组，在对应的应答分组里面就把这个时间戳照原样返回回去，这样发送方就能根据这个信息获得一个精确的关于RTT的采样。因此，尽管我们已经抹掉了tcpdump记录的时间信息，你仍然可以通过timestamp得到一些关于时间的暗示。注意通信两端的时间戳的选择是相对独立的，也不需要同步。结合这些暗示，并根据你对自适应重传及拥塞控制的理解，

   (a) 说明上述分组中有哪几个属于重传，它们对应的timeout分别大约是RTT的多少倍？

属于重传的分组是第(8)和第(9)号分组。因为它们的序列号与先前已经发出的第(6)号分组相同。

观察第(4)号分组，它是客户对服务器先前发出一个分组的确认，其中带有服务器先前发出分组的timestamp是184285706；再注意到第(5)号分组，其中的服务器方timestamp是184285748，其差值为42，是这一次对RTT进行采样的结果。

第(6)、(7)号分组于timestamp = 184285918 时先后发出，此后没有及时收到ACK。而在timestamp = 184286044 时发出第(8)号分组，实际是对(6)的第一次重传，其间的间隔是 126；又在 timestamp = 184286296 时发出第(9)号分组，即对(6)的第二次重传，和前次重发的间隔是 252。

因此，两次重发对应的 timeout 分别是 RTT 的大约3倍和6倍，恰好体现了TCP指数后退的timeout设置原理。

说明：本题考查考生根据新知识对现有原始分组数据进行进一步信息挖掘的能力。

   (b) 说明202.38.118.4主机80端口上的拥塞窗口在上述哪些分组被tcpdump侦听到的时刻发生了变化，是什么样的变化。

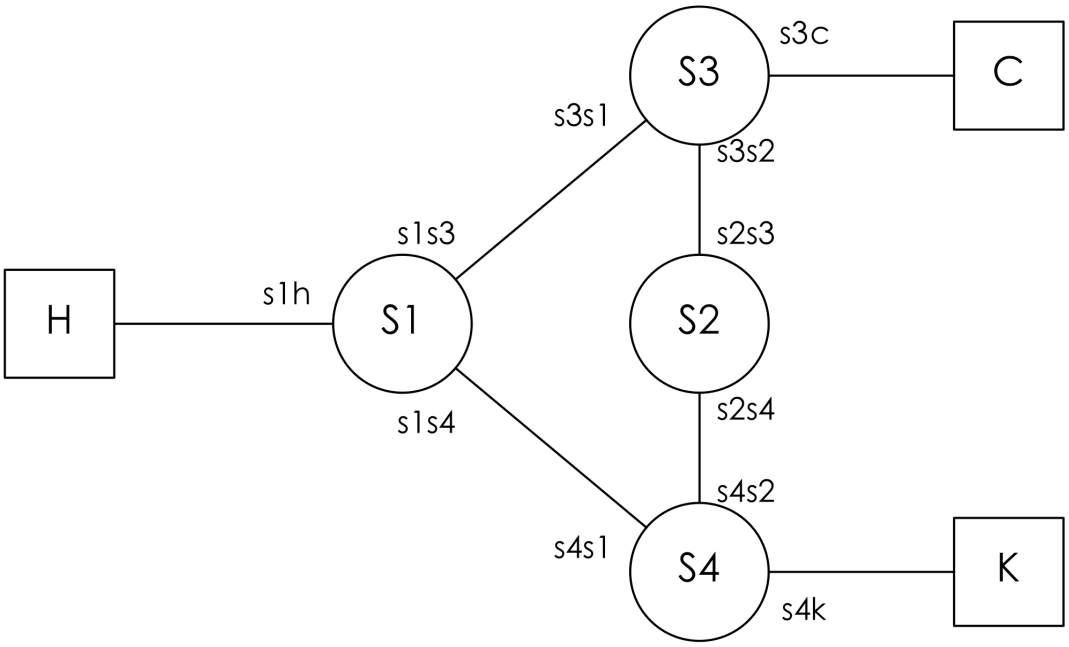
分组(2)是服务器接受客户端连接请求并发出自己的SYN报文，将拥塞窗口设置为1 MSS (1460 byte)，即进入慢启动。在分组(3)到达时，获得第一次ACK，拥塞窗口增长为2 MSS；分组(4)所捎带的确认号与(3)相同，不起进一步的作用。因此分组(6)、(7)共计2 MSS长的数据（含TCP头）在同一时刻先后发出。

分组(8)发生时，因timeout进入慢启动，同时慢自动门限（*ssthres*）降低到1 MSS；分组(9)是再次timeout的重传，继续慢启动；从现象上看，拥塞窗口直到分组(14)到达时才（线性的）增加到2 MSS以上。因为(6)、(7)、(11)、(13)都填满了MSS，有理由认为这期间服务器拥有足够多的数据要发送给客户，并不存在数据暂时发完之后的等待，因此实际发送的数据才能大致反映拥塞窗口的大小。（那么给出的信息是否足以判断具体的算法呢？）

说明：本题考查考生观察并客观描述网络行为现象的能力。

二． (交换和路由：存储转发的数据结构及其维护) [共40分] H和C之间的网络包含若干个转发设备，同时还连接有主机K，其拓扑结构如图二，其中每个设备的端口不妨用它自己和它邻近的节点标记。

7. 假定这些转发设备都是网桥（交换机），即H和C处于同一个局域网中。假设初始状态下所有网桥上的散列表都是空的，然后依次发生以下事件。其中每一步传送过程中，每个网桥是否能接收到相应的数据；每个网桥对相应数据帧的处理是**扩散**还是**转发**；这一步完成后B1 ~ B4各网桥上的散列表情况如何？请对表7-1～表7-3所列每一步填空（表格中给出了一个例子）。(15分)



图二

说明：本题的考查要点是后向学习法，所以生成树不作要求；但作为参考答案，必须按照生成树的算法严格考虑。对于评阅而言，在不考虑生成树情况下，可能存在同一目标有若干端口都可到达的情况，考生能够自圆其说即可。

从技术角度，首先应确定生成树。注意，每个网络都是点到点的网络。假定各网桥的标号S1<S2<S3<S4，则根据生成树算法得到各端口的状态如下图7。其中，每个网桥（除根以外）必有且仅有一个root port指向根的方向；每个网络有且仅有一个designated port持续传递配置消息；一般情况下候选端口（alternative port）或备用端口（backup port）不参加后向学习也不参与转发，该端口上收到的分组则应抛弃。注意，S2和S3及S4都有联系，且通

过S3和S4到根S1的距离是相等的，但S3标号更小，所以S2的root port是s2s3，而s2s4处于候选状态。

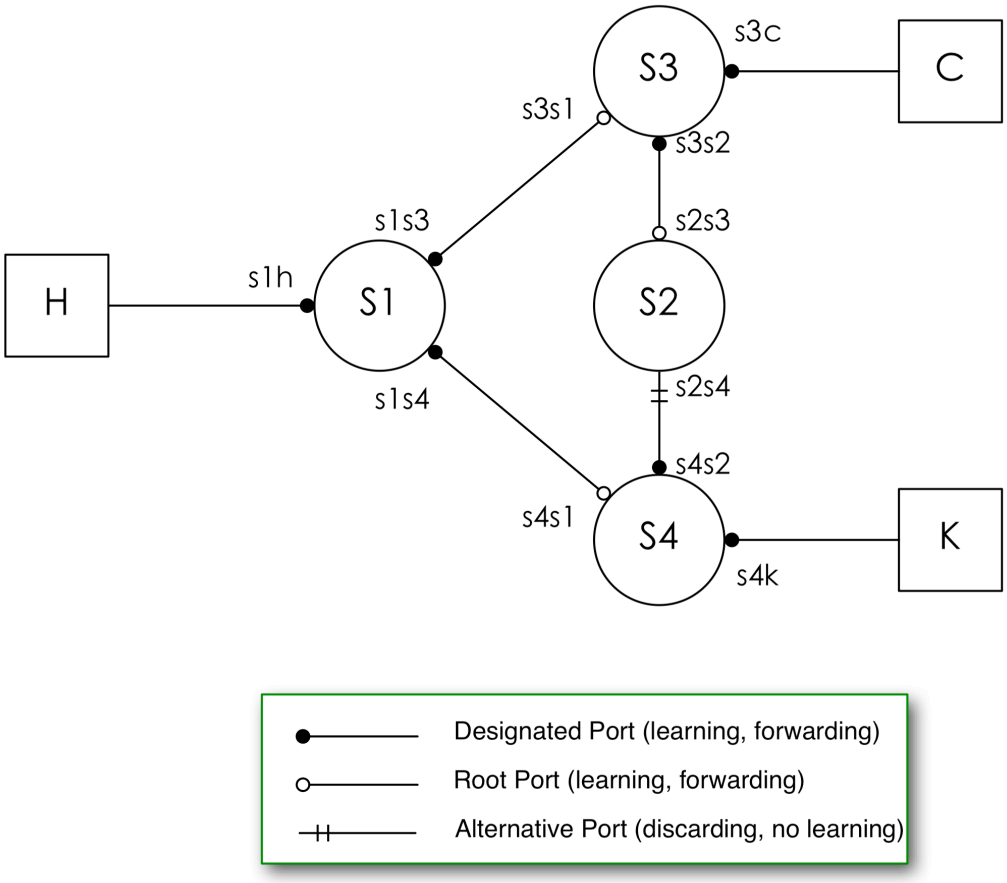


图7

a. K发送报文给H：表7-1

网桥		S1	S2	S3	S4
对该帧的处理	接收与否	接收	接收	接收	接收
	扩散或转发	扩散	—	扩散	扩散
处理完成后的散列表	主机/端口	K/s1s4	K/s2s3	K/s3s1	K/s4k
	主机/端口				—
	主机/端口				—

b. H发送报文给K：表7-2

网桥		S1	S2	S3	S4
对该帧的处理	接收与否	接收	收不到	收不到	接收
	扩散或转发	转发	—	—	转发
处理完成后的散列表	主机/端口	K/s1s4	K/s2s3	K/s3s1	K/s4k
	主机/端口	H/s1h			H/s4s1
	主机/端口				

c. C发送报文给H：表7-3

网桥		S1	S2	S3	S4
对该帧的处理	接收与否	接收	接收	接收	收不到
	扩散或转发	转发	—	扩散	—
处理完成后的散列表	主机/端口	K/s1s4	K/s2s3	K/s3s1	K/s4k
	主机/端口	H/s1h	C/s2s3	C/s3c	H/s4s1
	主机/端口	C/s1s3			

8. 假定所有这些转发设备都是运行同一链路状态路由协议的路由器，并且假定开始时C到K选择了S3-S2-S4路径。如果在C到K发送数据的过程当中，S2-S4之间的链路突然故障，之后又在10分钟后得以恢复，会发生什么情况？(5分)

链路故障的瞬间，S2会发现该故障并更新自身的LSP，经过重新计算得到去往节点K的next-hop是S3；但这时S3还没有收到S2更新后的LSP并完成新的前向搜索计算，因此保持到节点K下一跳是S2。这时C发往K的分组会在S3和S2之间形成环路；但这一过程会十分短暂，因为S2将链路变化后的LSP立即扩散给其它路由器，不会等到LSP发送的周期。

10分钟后，s2s4链路恢复，在S2更新后的LSP扩散到S3之前，C到K的分组会继续使用S3-S1-S4路径；S3收到S2的新LSP并更新转发表之后，将直接改用S3-S2-S4路径，链路恢复后不会发生任何环路。

说明：本题考查对链路状态算法在链路变化过程中路由行为的理解。

9. 假定所有这些转发设备都是运行同一距离向量路由协议的路由器，并且假定开始时C到K选择了S3-S2-S4路径。考虑有“水平分割”(Split Horizon) 和“毒性逆转”(Poison Reverse) 的情况。如果在C到K发送数据的过程当中，S4-K之间的链路突然故障，试在表9中分析这之后S1~S4各路由器上到K的路由变化，如果达到了收敛或者下面表格填满，则分析中止。表项以“距离/下一跳”形式表达；备注栏用于填写是否产生了环路、是否达到了收敛。(15分)

表9

路由器 / 时刻	S1	S2	S3	S4	备注
初始状态	2/S4	2/S4	3/S2	1/K	
1个周期后	2/S4	2/S4	3/S2	∞/K	s4k链路故障
2个周期后	4/S3	∞/S4	3/S2	∞/K	S4 → S1: ∞, S3 → S1: 3; S4 → S2: ∞, S3 → S2: ∞; S2 → S3: 2, S1 → S3: 2; S1 → S4: ∞, S2 → S4: ∞
3个周期后	4/S3	∞/S4	∞/S2	5/S1	S4 → S1: ∞, S3 → S1: 3; S4 → S2: ∞, S3 → S2: ∞; S2 → S3: ∞, S1 → S3: ∞; S1 → S4: 4, S2 → S4: ∞
4个周期后	∞/S3	6/S4	∞/S2	5/S1	S4 → S1: ∞, S3 → S1: ∞; S4 → S2: 5, S3 → S2: ∞; S2 → S3: ∞, S1 → S3: ∞; S1 → S4: 4, S2 → S4: ∞
5个周期后	∞/S3	6/S4	7/S2	∞/S1	…
6个周期后	8/S3	∞/S4	7/S2	∞/S1	…
7个周期后	8/S3	∞/S4	∞/S2	9/S1	不收敛，但也不产生环路

说明：本题考查逻辑推理能力及对距离向量算法的理解。题目选择的拓扑结构是四个路由器组成的环路。课堂讲过，如果多于两个点之间发生环路或者慢收敛，则水平分割和毒性逆转都不奏效。这个题目给出了一个相当生动的例子。如果头脑清醒的进行运用距离向量算法及毒性逆转条件，是不难得到上述结果的。

10. 假定这些节点和路由器都运行IPv6。其中节点S4的转发表如下表10，试说明去往某目的地址2001:da8:ffca:2676::4应该选择哪一条路由。 (5分)

表10

Destination	Next-Hop	Iface
0::/0	2001:da8:200:9002::1	s4s1
3ffe::/16	2001:da8:200:9001::27	s4s2
2001:da8:ffca:2674::/63	2001:da8:200:9002::1	s4s1
2001:da8:ffca:26::/64	2001:da8:200:9001::27	s4s2
2001:da8:ffca:2674::/64	#link3	s4k
2001:da8:ffca:2676:4::/80	2001:da8:200:9001::27	s4s2

运用最长匹配的原则，应选择第一条，即
0::/0，下一跳2001:da8:200:9002::1，接口s4s1。

说明：本题考查对CIDR最长匹配（或more-specific win）准则的理解。本题与去年某题雷同。有的同学对去年试题答案未加认真体会，以为打字有误；乃至今年覆辙，实在遗憾。

三． (时序图：点到点和端到端的滑动窗口协议) [共20分] 设SWS = RWS = 4，用时序图研究滑动窗口协议的行为。约定：(a) 接收方使用请求下一分组形式的累计确认，例如ACK[5]表示PACKET[1] ~ PACKET[4]均已收妥；(b) 接收方在收到分组但发现并非当前期望的下一分组时发出重复确认（DUPACK），例如在PACKET[3]到来但PACKET[2]尚未收到时发出DUPACK[2]。设发送方的重传超时为 2×RTT，其中RTT是双方之间延迟的长期统计平均值。假定发送方需要发给对方7个分组的消息，而在发送过程中PACKET[2]和PACKET[5]在首次发送过程中丢失了。

11. (12分)如果发送方在超时或收到一个重复确认时都认为是发生了分组丢失，从而进行重传。

(a) 对于上述整个交易的完成，会比仅仅在超时的情况下才重传分组节约大约多少时间？试通过时序图来说明问题。

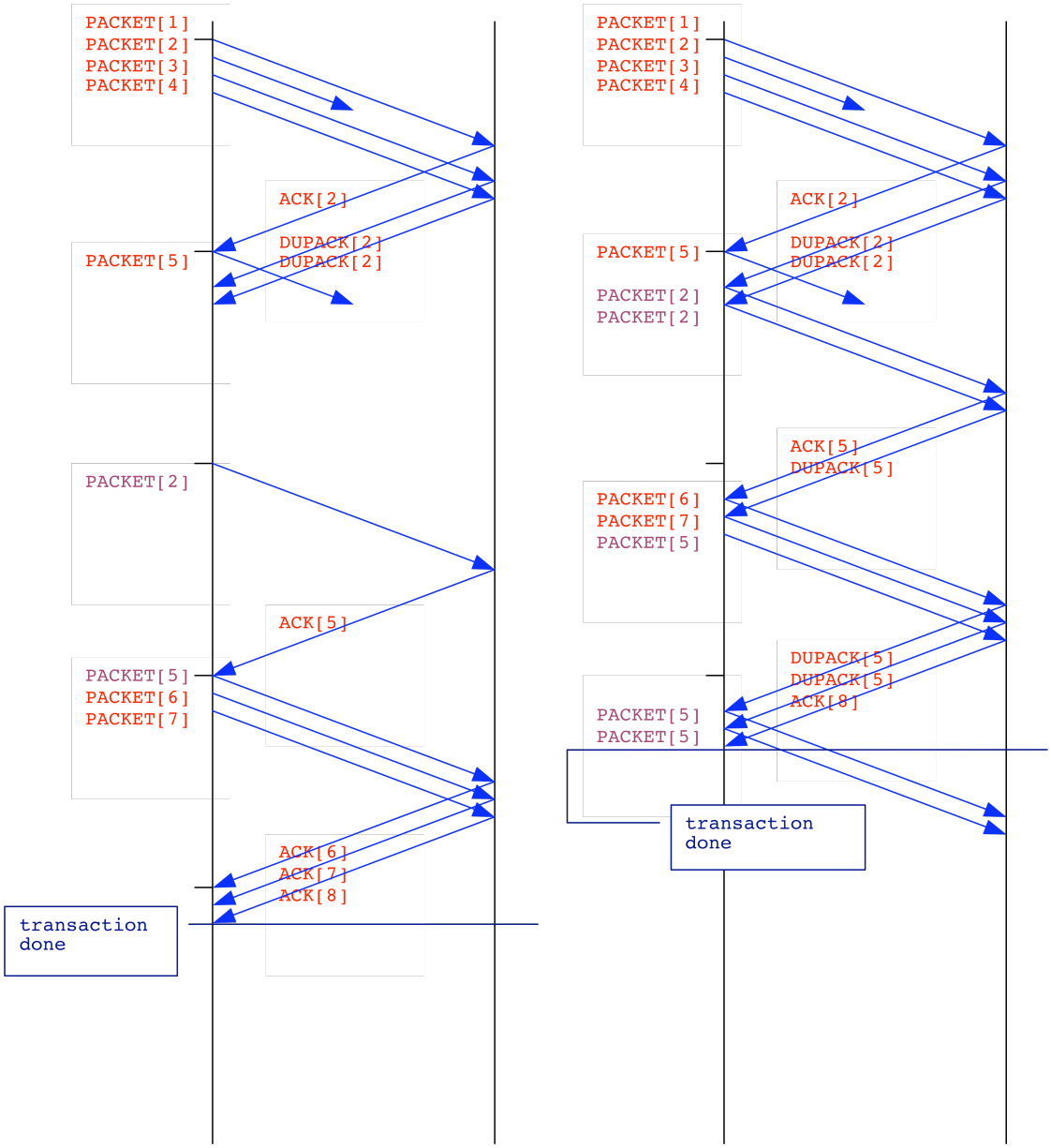


图11

画出时序图对比只有超时重传和遇到一个DUPACK就重传的情况如图11。不难看出，使用DUPACK加速重传完成7个分组的交易，在丢失2、5两个分组的情况下，大约节省1 RTT的时间。（其中，重传发生时，timer也应复位，这是个比较自然的考虑。）

需要说明的是，在PACKET[5]发出后但未到timeout前，收到ACK[5]，将发出PACKET[6]和PACKET[7]，是发送方窗口移动到{6,7,8,9}的结果。而按照题意，发送方还没有PACKET[8]和PACKET[9]可以发送。

说明：本题考查利用时序图研究问题的能力和逻辑思维能力。

(b) 设要传送的分组大小都是P，通信双方之间的带宽是B，则整个通信的有效平均吞吐率 (throughput) Th是多少？

交易完成以全部7个分组得到确认为标志。如图11右边的时序图分析，总共耗费的时间是 3 RTT + 5 P/B, 所以

$$Th = \frac{7P}{3RTT + \frac{5P}{B}}$$

说明：本题考查吞吐率的概念。考生如果定义了别的交易完成的准则，只要计算合理，也都是可以接受的解答。

讨论：这里的RTT必须足够大，从而能够保证一个RTT时间尺度上足够完成SWS大小的数据的传送。这时如果带宽B或者RTT非常大，则吞吐率主要由RTT决定，接近于 7P/3RTT。

但如果RTT非常小，小到有的ACK返回的时候还有窗口中的分组没有发出去，以上的时序图和吞吐率的计算都会发生变化。过于细节的讨论，本题不做要求。

12. (8分)上述收到一个重复确认即重传分组的做法，

(a) 是否适合一条物理链路上的点到点滑动窗口协议？简要说明理由。

对于RTT非常小的点到点链路，该协议是不合适的。试考虑有的ACK返回的时候还有窗口中的分组没有发出去的情况，这时候DUPACK有可能反而发生在超时之后，从而造成过多的重传。

对于RTT足够大的点到点链路，重复确认能够作为链路偶然突然故障或者发生冲突的一个标志，从而加快重传。对于偶然故障来说，这意味着这样的故障必须要在分组传输的时间尺度上得以恢复，否则就没有意义；对于冲突而导致的发送失败，则并不鼓励过于积极的实施重传。所以是否合适，取决于具体的物理链路特征。

(b) 是否适合端到端的滑动窗口协议？用时序图举例说明你的结论。

对于RTT变化范围大并且中间节点具有存储功能的端到端环境，仅仅凭一次重复确认作为分组丢失的标志，是不可取的。如图，由于PACKET[2]延迟比PACKET[3]更长而导致DUPACK[2]的发生，这时PACKET[2]的重传实

属多余。而且，如果网络这时候处于排队的状态，则这种过于积极的重传反而加重网络朝拥塞状态转移。

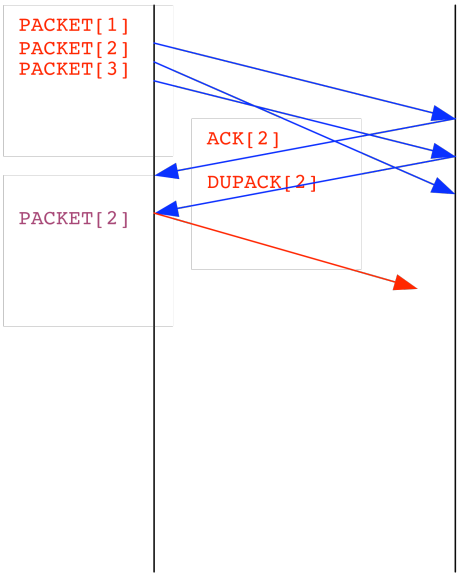


图12

说明：本题考查对端到端通信信道与点到点物理链路的区别与联系的理解。上述答案只具有参考意义。本题评阅建议以开放式问题对待。要求考生逻辑清晰、言之有理。