

# 1 复习题

1. 没有不同。主机和端系统可以互换。端系统包括 PC，工作站，WEB 服务器，邮件服务器，网络连接的 PDA，网络电视等等。
2. 假设爱丽丝是国家 A 的大使，想邀请国家 B 的大使鲍勃吃晚餐。爱丽丝没有简单的打个电话说“现在我没一起吃晚餐吧”。而是她先打电话给鲍勃建议吃饭的日期与时间。鲍勃可能会回复说那天不行，另外一天可以。爱丽丝与鲍勃不停的互发讯息直到他们确定一致的日期与时间。鲍勃会在约定时间（提前或迟到不超过 15 分钟）出现再大使馆。外交协议也允许爱丽丝或者鲍勃以合理的理由礼貌的退出约会。
3. 联网（通过网络互联）的程序通常包括 2 个，每一个运行在不同的主机上，互相通信。发起通信的程序是客户机程序。一般是客户机请求和接收来自服务器程序的服务。
4. 互联网向其应用提供面向连接服务（TCP）和无连接服务（UDP）2 种服务。每一个互联网应用采取其中的一种。
 

面向连接服务的原理特征是：

  - ① 在都没有发送应用数据之前 2 个端系统先进行“握手”。
  - ② 提供可靠的数据传送。也就是说，连接的一方将所有应用数据有序且无差错的传送到连接的另一方。
  - ③ 提供流控制。也就是，确保连接的任何一方都不会过快的发送过量的分组而淹没另一方。
  - ④ 提供拥塞控制。即管理应用发送进网络的数据总量，帮助防止互联网进入迟滞状态。

无连接服务的原理特征：

  - ① 没有握手
  - ② 没有可靠数据传送的保证
  - ③ 没有流控制或者拥塞控制
5. 流控制和拥塞控制的两个面向不同的对象的不同的控制机理。流控制保证连接的任何一方不会因为过快的发送过多分组而淹没另一方。拥塞控制是管理应用发送进网络的数据总量，帮助防止互联网核心（即网络路由器的缓冲区里面）发生拥塞。
6. 互联网面向连接服务通过使用确认，重传提供可靠的数据传送。当连接的一方没有收到它发送的分组的确认（从连接的另一方）时，它会重发这个分组。
7. 电路交换可以为呼叫的持续时间保证提供一定量的端到端的带宽。今天的大多数分组交换网（包括互联网）不能保证任何端到端带宽。当发生拥塞等网络问题时，TDM 中的数据丢失可能只会是一部分，而 FDM 中就可能是大部分或全部。
8. 在一个分组交换网中，在链路上流动的来自不同来源的分组不会跟随任何固定的，预定义的模式。在 TDM 电路交换中，每个主机从循环的 TDM 帧中获得相同的时隙。
9.  $t_0$  时刻，发送主机开始传输。在  $t_1=L/R_1$  时刻，发送主机完成发送并且整个分组被交换机接收（无传输时延）。因为交换机在  $t_1$  时刻接收到了整个分组，它可以在  $t_1$  时刻开始向接收主机发送分组。在  $t_2=t_1+L/R_2$  时刻，交换机完成传输且接收主机收到了整个分组（同样，无传输时延）。所以，端到端时延是  $L/R_1+L/R_2$ 。
10. 在一个虚电路网络中，每个网络核心中的分组交换机都对经过它传输的虚电路的连接状态信息进行维护。有的连接状态信息是维护在一个虚电路数字传输表格中。
11. 面向连接的 VC 电路的特点包括：a.建立和拆除 VC 电路是需要一个信令协议；b.需要在分组交换中维持连接状态。有点方面，一些研究者和工程人员争论到：使用 VC 电路可以更容易提供 QoS 业务，如：保证最小传输率的业务，以及保证端到端的最大分组延时的业务。
12. a.电话线拨号上网：住宅接入；b.电话线 DSL 上网：住宅接入或小型办公；c.混合光纤同轴电缆：住宅接入；d.100M 交换机以太网接入：公司；e.无线局域网：移动接入；f.蜂窝移动电话（如 WAP）：移动。
13. 一个第一层 ISP 与所有其它的第一层 ISP 相连；而一个第二层 ISP 只与部分第一层 ISP 相连。而且，一个第二层 ISP 是一个或多个第一层 ISP 的客户。
14. POP 是 ISP 网络中一个或多个路由器构成的一个组，其它 ISP 中的路由器也可以能连接到这个 POP。NAP 是一个很多 ISP（第一层，第二层，以及其它下层 ISP）可以互联的局部网络。
15. HFC 的带宽是用户间共享的。在下行信道，所有的分组从头到尾由同一个源发出，因此在下行信道不会发生冲突。
16. 以太网的传输速率有：10Mbps，100Mbps，1Gbps 和 10Gbps。对于一个给定的传输速率，如果用户单独在线路上传输数据，则可以一直保持这个速率；但是如果有多用户同时传输，则每个都不能达到所给定的速率（带

宽共享)。

17. 以太网通常以双绞线或者细的同轴电缆为物理媒体，也可以运行在光纤链路和粗同轴电缆上。
18. 拨号调制解调器：最高 56Kbps，带宽专用；ISDN：最高 128Kbps，带宽专用；ADSL：下行信道 5—8Mbps，上行信道最高 1Mbps，带宽专用；HFC：下行信道 10—30Mbps，上行信道一般只有几 Mbps，带宽共享。
19. 时延由处理时延、传输时延、传播时延和排队时延组成。所有这些时延除了排队时延都是固定的。
20. 5 种任务为：错误控制，流量控制，分段与重组，复用，以及连接建立。是的，这些任务可以由两层（或更多层）来执行，比如：经常在多于一个层次上执行错误控制。
21. 英特网协议栈的 5 个层次从上倒下分别为：应用层，传输层，网络层，链路层，和物理层。每一层的主要任务见 1.7.1 节。应用层是网络应用程序及其应用层协议存留的地方；运输层提供了一个在应用程序的客户机和服务器之间传输应用层报文的服务；网络层负责将称为数据报的网络层分组从一台主机移动到另一台主机；链路层：通过一系列分组交换机（Internet 中的路由器）在源和目的地之间发送分组；物理层：将该帧中的一个一个比特从一个节点移动到下一个节点。
22. 应用层报文：应用程序要发出的在传输层上传递的数据；传输层报文段：将应用层报文加上传输层包头，由传输层管理和封装的信息；网络层数据报：将传输层报文段加上网络层包头之后封装；链路层帧：将网络层数据报加上链路层包头之后封装。
23. 路由器处理第一层到第三层（这是一个善意的谎话？本人理解为“这样说不确切”因为现代路由器常常还要扮演防火墙或者高速缓存器的角色，一次也处理第四层）；链路交换机处理第一层和第二层；主机处理所有的 5 层。

## 习题

1. 本题不止一个答案，很多协议都能解决这个问题，下面是一个简单的例子：

### Messages from ATM machine to Server

Msg name	purpose
HELO <userid>	Let server know that there is a card in the ATM machine
PASSWD <passwd>	ATM card transmits user ID to Server
BALANCE	User enters PIN, which is sent to server
WITHDRAWL <amount>	User requests balance
BYE	User asks to withdraw money
	user all done

### Messages from Server to ATM machine (display)

Msg name	purpose
PASSWD	Ask user for PIN (password)
OK	last requested operation (PASSWD, WITHDRAWL) OK
ERR	last requested operation (PASSWD, WITHDRAWL) in ERROR
AMOUNT <amt>	sent in response to BALANCE request
BYE	user done, display welcome screen at ATM

**Correct operation:**

```

client                                server
HELO (userid)  -----> (check if valid userid)
               <----- PASSWD
PASSWD <passwd> -----> (check password)
               <----- OK (password is OK)
BALANCE        ----->
               <----- AMOUNT <amt>
WITHDRAWAL <amt> -----> check if enough $ to cover
               <----- withdrawl
               <----- OK
ATM dispenses $
BYE            ----->
               <----- BYE

```

**In situation when there's not enough money:**

```

HELO (userid)  -----> (check if valid userid)
               <----- PASSWD
PASSWD <passwd> -----> (check password)
               <----- OK (password is OK)
BALANCE        ----->
               <----- AMOUNT <amt>
WITHDRAWAL <amt> -----> check if enough $ to cover
               <----- withdrawl
               <----- ERR (not enough funds)
error msg displayed
no $ given out
BYE            ----->
               <----- BYE

```

2. a. 电路交换网更适合所描述的应用，因为这个应用要求在可预测的平滑带宽上进行长期的会话。由于传输速率是已知，且波动不大，因此可以给各应用会话预留带宽而不会有太多的浪费。另外，我们不需要太过担心由长时间典型会话应用积累起来的，建立和拆除电路时耗费的开销时间。  
b. 由于所给的带宽足够大，因此该网络中不需要拥塞控制机制。最坏的情况下（几乎可能拥塞），所有的应用分别从一条或多条特定的网络链路传输。而由于每条链路的带宽足够处理所有的应用数据，因此不会发生拥塞现象（只会有非常小的队列）。
3. a. 因为这 4 对相邻交换机，每对之间可以建立  $n$  条连接，；因此最多可以建立  $4n$  条连接。  
b. 可以通过右上角的交换机建立  $n$  条连接，并且可以通过左下角交换机建立  $n$  条连接，因此最多可以建立  $2n$  条连接。
4. 由于收费站间隔 100km，车速 100km/h，收费站以每 12m 通过一辆汽车的速度提供服务。a) 10 辆车，第一个收费站要花费 120s，即 2 分钟来处理。每一辆车要达到第二个收费站都会有 60 分钟的传输延时，因此每辆车要花费 62 分钟才能达到第二个收费站，从第二个收费站到第三个收费站重复这一过程。因此，（端到端）总延时为 124 分钟。  
b) 每两个收费站之间的延时为  $7 \times 12 \text{ 秒} + 60 \text{ 分} = 61 \text{ 分 } 24 \text{ 秒}$ ，（端到端）总延时  $= 3624 \times 2 = 7,248\text{s}$ ，即 112 分 48 秒。
5. a) 传输一个分组到一个链路层的时间是  $(L+h)/R$ 。Q 段链路的总时间为： $Q(L+h)/R$ 。所以发送文件所需要总的时间为： $t_s + (L+h)/Q$ 。  
b)  $Q(L+2h)/R$   
c) 由于链路上没有存储转发延时，因此，总延时为： $t_s + (L+h)/R$ 。
6. a) 传播时延  $d_{\text{prop}} = m/s$  秒  
b) 传输时延  $d_{\text{trans}} = L/R$  秒  
c) 端到端时延  $d_{\text{end-to-end}} = (m/s + L/R)$  秒  
d) 该分组的最后一个 bit 刚刚离开主机 A。  
e) 第一个比特在链路中，还没有到达 B。

f) 第一个比特已经到达 B。

g)  $m = LS/R = 100 \times 2.5e8 / 28e3 = 893 \text{ km}$

7. 考虑分组中的第一个 bit。在这个 bit 被传输以前，先要收集这个分组中的其它 bit，这个需要： $48 \times 8 / 64e3 = 6e-3 \text{ s} = 6 \text{ ms}$

分组的传输延时： $48 \times 8 / 1e6 = 384e-6 \text{ s} = 0.384 \text{ ms}$

传播延时：2ms

到该 bit 被解码的延时为： $6 + 0.384 + 2 = 8.384 \text{ ms}$ （英文答案中的那个“.”表示乘）

8. a) 由于每个用户需要十分之一的带宽，因此可以支持 10 个用户。

b)  $p = 0.1$

c) 
$$\binom{40}{n} p^n (1-p)^{40-n}$$

d) 
$$1 - \sum_{n=0}^9 \binom{40}{n} p^n (1-p)^{40-n}$$

我们用中心极限定理来求这个概率的近似解。令  $X_j$  表示 J 个用户同时传输的概率，如  $P(X_j=1) = p$ ，则：

$$\begin{aligned} P(\text{"11 or more users"}) &= 1 - P\left(\sum_{j=1}^{40} X_j \leq 10\right) \\ P\left(\sum_{j=1}^{40} X_j \leq 10\right) &= P\left(\frac{\sum_{j=1}^{40} X_j - 4}{\sqrt{40 \cdot 0.1 \cdot 0.9}} \leq \frac{6}{\sqrt{40 \cdot 0.1 \cdot 0.9}}\right) \\ &\approx P\left(Z \leq \frac{6}{\sqrt{3.6}}\right) = P(Z \leq 3.16) \\ &= 0.999 \end{aligned}$$

所以所求概率约为：0.001

9. a) 10,000

b) 
$$\sum_{n=N+1}^M \binom{M}{n} p^n (1-p)^{M-n}$$

10. 传输这 N 个分组需要  $LN/R$  秒。当一批 N 个分组到达时，缓存器内是空的。

第一个分组没有排队延时，第二个分组的排队延时为  $L/R$  秒.....第 N 个分组的排队延时为： $(N-1) L/R$  秒，所以平均排队延时为：

$$\frac{1}{N} \sum_{n=1}^N (n-1) L/R = \frac{L}{R} \frac{1}{N} \sum_{n=0}^{N-1} n = \frac{L}{R} \frac{1}{N} \frac{(N-1)N}{2} = \frac{L}{R} \frac{(N-1)}{2}$$

11. a) 传输延时为  $L/R$ ，总延时为

$$\frac{IL}{R(1-I)} + \frac{L}{R} = \frac{L/R}{1-I}$$

b) 令  $x = L/R$ ，则总延时与 x 的函数为：总延时 =  $x / (1 - ax)$ 。

12. a) 一共有 Q 个节点（源主机喝 N-1 个路由器）用  $d_{proc}^q$  表示第 q 个节点的处理延时。用  $R^q$  表示第 q 个节点处的传

输速率，令  $d_{trans}^q = L / R^q$ 。用  $d_{prop}^q$  表示通过第 q 条链路的传播延时。则：

$$d_{end-to-end} = \sum_{q=1}^Q [d_{proc}^q + d_{trans}^q + d_{prop}^q]$$

b)用  $d_{queue}^q$  表示节点 q 处的平均排队延时，则：

$$d_{end-to-end} = \sum_{q=1}^Q [d_{proc}^q + d_{trans}^q + d_{prop}^q + d_{queue}^q].$$

13. 实验题？不会考吧。。。。。

14. a) “带宽时延”积 =  $(1e7/2.5e8) * 1e6 = 40,000\text{bit}$

b) 40000bit

c) 一条链路的带宽时延积就是这条链路上具有的比特数的最大值。

d)  $1e7/4e4 = 250\text{m}$ , 比一个足球场的长度还长。

e)  $s/R$

15.  $2.5e8/1e6 = 25\text{bps}$

16. a)  $(1e7/2.5e8) * 1e9 = 40,000,000\text{bit}$

b) 400,000bit (包长度)

c)  $1e7/4e5 = 25\text{m}$

17. a) 传播时延 =  $1e7/2.5e8 = 40\text{ms}$ ；传输时延 =  $4e5 \times 250/2.5e8 = 400\text{ms}$

因此总延时为：440ms

b) 传播时延 =  $2 \times 40 = 80\text{ms}$  (发送及返回确认)；传输时延 =  $4e4 \times 250/2.5e8 = 40\text{ms}$ ，传送 10 个分组，总时延 =  $10 \times (80 + 40) = 1200\text{ms} = 1.2\text{s}$

18. a) 地球同步卫星距离地面 3600km，因此该链路的传播时延 =  $3600e3/2.4e8 = 150\text{ms}$

b)  $150e-3 \times 10e6 = 1,500,000$

c)  $60 \times 10e6 = 6e8\text{bit}$

19. 我们假设旅客和行李对应到达协议栈顶部的数据单元，当旅客检票的时候，他的行李也被检查了，行李和机票被加上标记。这些信息是在包裹层被添加的 (if Figure 1.20 that 不知道怎么翻译.....) 允许在包裹层使服务生效或者在发送侧将旅客和行李分离，然后在目标侧 (如果可能的话) 重新组合他们。当旅客稍后通过安检，通常会另外添加一个标记，指明该旅客已经通过了安检。这个信息被用于保证旅客的安全运输。(答非所问?)

20. a) 将报文从源主机发送到第一个分组交换机的时间 =  $7.5e6/1.5e6 = 5\text{s}$ 。由于使用存储转发机制，报文从源主机到目标主机的总时间 =  $5 \times 3$  (跳) = 15s。

b) 将第一个分组从源主机发送到第一个分组交换机的时间 =  $1.5e3/1.5e6 = 1\text{ms}$ 。

第一个分组交换机完成接收第二个分组所需的时间 = 第二个分组交换机完成接收第一个分组所需的时间 =  $2 \times 1\text{ms} = 2\text{ms}$ 。

c) 目标主机收到第一个分组所需的时间 =  $1\text{ms} \times 3$  (跳) = 3ms，此后每 1ms 接收一个分组，因此完成接收 5000 个分组所需的时间 =  $3 + 4999 \times 1 = 5002\text{ms} = 5.002\text{s}$ 。可以看出采用分组传输所用的时间要少的多 (几乎少 1/3)。

d) 缺点：

1) 分组在目标侧必须按顺序排放；

2) 报文分组产生了很多分组，由于不论包的大小如何，包头大小都是不变的，报文分组中包头子节的损耗会高于其它方式。

21. JAVA 程序试验。。。。。略

22. 目标侧接受到第一个分组所需的时间 =  $\frac{S+40}{R} \times 2 \text{ sec}$ 。之后，每  $(S+40)/R$  秒，目标侧接受到一个分组。所以发送所有文件所需的时间：

$$\text{delay} = \frac{S+40}{R} \times 2 + \left(\frac{F}{S} - 1\right) \times \left(\frac{S+40}{R}\right) = \frac{S+40}{R} \times \left(\frac{F}{S} + 1\right)$$

为了计算最小时延对应的 S，对 delay 进行求导，则：

$$\frac{d}{dS} \text{delay} = 0 \Rightarrow \frac{F}{R} \left(\frac{1}{S} - \frac{40+S}{S^2}\right) + \frac{1}{R} = 0 \Rightarrow S = \sqrt{40F}$$

## 2 复习题

1. The Web: HTTP; file transfer: FTP; remote login: Telnet; Network News: NNTP; email: SMTP.
2. P51 网络体系结构是指以分层的方式来描述通信过程的组织体系。（例如五层网络结构）另一方面，应用体系结构是由应用程序的研发者设计，并规定应用程序的主要结构（例如 客户机/服务器或 P2P）从应用程序研发者的角度看，网络体系结构是固定的，并为应用程序提供了特定的服务集合。
3. P52 在即时讯息中，发起人连接到中心服务器，查找接收方的 IP 地址是典型的客户机/服务器模式。在这之后，即时信息可以在相互通信的双方进行直接的端到端通信。不需要总是打开的中间服务器。
4. P53 发起通信的进程为客户机，等待联系的进程是服务器。
5. No. As stated in the text, all communication sessions have a client side and a server side. In a P2P file-sharing application, the peer that is receiving a file is typically the client and the peer that is sending the file is typically the server.
6. P54 目的主机的 IP 地址和目的套接字的端口号。
7. 在日常生活中你或许会使用 Web 浏览器和邮件阅读器。你或许还会用到 FTP 用户代理，Telnet 用户代理，音频/视频播放器用户代理（比如 Real Networks player），即时信息代理，P2P 文件共享代理。
8. There are no good examples of an application that requires no data loss and timing. If you know of one, send an e-mail to the authors.
9. 当两个通信实体在相互发送数据前第一次交换控制分组信息时使用握手协议。SMTP 在应用层使用握手协议。然而 HTTP 不是这样。
10. P56、57 因为与这些协议相联系的应用都要求应用数据能够被无差错的有序的接收。TCP 提供这种服务，而 UDP 不提供。TCP 提供可靠的数据传输服务，而 UDP 提供的是不可靠数据传输。
11. P66 当用户第一次访问一个站点。这个站点返回一个 cookie 号码。这个 cookie 码被存储在用户主机上并由浏览器管理。在随后的每次访问（和购买）中，浏览器将这个 cookie 码回送该站点。这样当用户访问该站点时，都会被该站点所知道。
12. P62 在非流水线的 HTTP 持久连接中，客户机只能在接收到服务器发来的前一个响应后才能发出新的请求。在流水线的 HTTP 持久连接中，浏览器只要有需要就会发出请求，不需要等待服务器的响应信息。HTTP/1.1 的默认模式使用了流水线方式的持久连接
13. P67 Web 缓存能够使用户所希望的内容距离用户更近，或许就在用户主机所连接的局域网内。Web 缓存能够减小用户请求的所有对象的时延，即使是该对象没有被缓存，因为缓存能够减少链路上的流量。因此改善了所有应用的性能。因为一般情况下客户机与 Web 缓存器的瓶颈带宽要比客户机与起始服务器之间的瓶颈带宽大的多。如果用户所请求的对象在 Web 缓存器上，则该 Web 缓存器可以迅速将该对象交付给用户。
14. 实验题，应该不考吧。。。。。
15. P70、71 FTP 使用两个并行的 TCP 连接，一个连接用来传送控制信息（例如一个传送文件的请求），另一个连接用于准确地传输文件。因为控制信息不是在文件传输地连接上传送，所以 FTP 的控制信息是带外传送的。
16. P81 信息从 Alice 的主机发送到她的邮件服务器，使用 HTTP 协议。然后邮件从 Alice 的邮件服务器发送到 Bob 的邮件服务器，使用 SMTP 协议。最后 Bob 将邮件从他的邮件服务器接收到他的主机，使用 POP3 协议。
17. 无。
18. P80 在下载并删除方式下，在用户从 POP 服务器取回他的邮件后，信息就被删除调。这就为移动的用户带来一个问题。因为该用户有可能想从不同的机器上访问邮件。（办公 PC，家用 PC 等）。在下载并保留方式下，在用户取回邮件后，邮件不会被删除。这同样也会带来一些不便。因为每次当用户在一台新的机器上取回存储的邮件时，所有的没有被删除的信息都将会被传送的新的机器上（包括非常老的邮件）。
19. P88 是的，一个机构的邮件服务器和 Web 服务器可以有完全相同的主机名别名。MX 记录被用来映射邮件服务器的主机名到它的 IP 地址。如果 Type=MX，则 Value 是别名为 Name 的邮件服务器的规范主机名。RR: resource record. 为了获得邮件服务器的规范主机名，DNS 客户机应当请求一条 MX 记录；而为了获得其他服务器的规范主机名，DNS 客户机应当请求 CNAME 记录。Type=CNAME
20. P93 P2P 文件共享系统的覆盖网络包括参与到文件共享系统中的节点和节点间的逻辑连接。如果 A 和 B 之间有一条非永久性的 TCP 连接，那么我们就在 A 和 B 之间有一条逻辑连接（在图论领域被称为一条“边”）。一个覆盖网络不包括路由器。在 Gnutella 网络中，当一个节点想要加入到 Gnutella 网络，它首先发现已经在网络中

的一个或多个节点的 IP 地址。然后它向这些节点发送加入请求信息。当这个节点接收到确认信息时，它就成为了 Gnutella 网络的一员。节点通过周期性的更新信息保持它们的逻辑连接。（在 Gnutella 中，对等方形成了一个抽象的逻辑网络，该网络被称为覆盖网络。用图论的术语来说，如果对等方 A 与另一个对等方 B 维护了一条 TCP 连接，那么我们说在 A 和 B 之间有一条边。该图由所有活跃的对等方和连接的边（持续的 TCP 连接）组成，该图定义了当前的 Gnutella 覆盖网络。

## 21. Three companies as of this writing (August 2004) are KaZaA, eDonkey, Bit Torrent.

Napster 提供集中式目录来跟踪位于对等方中的内容。Gnutella 使用全分布方法定位内容。KaZaA 结合了前二者的思想，通过指派少量更有力度的对等方作为组长，利用了对等方的不均匀性，形成了一个层次覆盖网络的顶层。

## 22. P99、104 对于 UDP 服务器，没有欢迎套接字，所有来自不同客户机的数据通过同一个套接字进入服务器。对于 TCP 服务器，有欢迎套接字，每次一个客户机建立一个到服务器的连接，就会建立一个新的套接字。因此，为了同时支持 n 个连接，服务器需要 n+1 个套接字。

## 23. 对于 TCP 应用，一旦客户机开始执行，它就试图建立一个到服务器的 TCP 连接。如果 TCP 服务器没有运行，那么客户机就会建立连接失败。对于 UDP 应用，客户机不需要在其执行的时候立即建立连接（或试图与 UDP 服务器通信）。

## 习题

1. a) F P62 b) T P62 c) F P61 d) F P64 Data 首部行表示服务器产生并发送响应报文的日期和时间。
2. Access control commands: USER, PASS, ACT, CWD, CDUP, SMNT, REIN, QUIT.  
Transfer parameter commands: PORT, PASV, TYPE, STRU, MODE. Service commands: RETR, STOR, STOU, APPE, ALLO, REST, RNFR, RNT0, ABOR, DELE, RMD, MRD, PWD, LIST, NLST, SITE, SYST, STAT, HELP, NOOP.
3. SFTP: 115, NNTP: 119.
4. Application layer protocols: DNS and HTTP Transport layer protocols: UDP for DNS; TCP for HTTP
5. Persistent connections are discussed in section 8 of RFC 2616 (the real goal of this question was to get you to retrieve and read an RFC). Sections 8.1.2 and 8.1.2.1 of the RFC indicate that either the client or the server can indicate to the other that it is going to close the persistent connection. It does so by including the connection-token "close" in the Connection-header field of the http request/reply.  
客户机和服务器都可以向对方声明它准备关闭持久连接。通过在 HTTP 请求/响应中的 Connection 首部行中包含 Connection: close 来完成此项操作。加密服务???
6. The total amount of time to get the IP address is  $RTT_1 + RTT_2 + \Lambda + RTT_n$ . Once the IP address is known,  $RTT_0$  elapses to set up the TCP connection and another  $RTT_0$  elapses to request and receive the small object. The total response time is  $2RTT_0 + RTT_1 + RTT_2 + \Lambda + RTT_n$
7. a)  $RTT_1 + \Lambda + RTT_n + 2RTT_0 + 3 \times 2RTT_0 = 8RTT_0 + RTT_1 + \Lambda + RTT_n$   
b)  $RTT_1 + \Lambda + RTT_n + 2RTT_0 + 2RTT_0 = 4RTT_0 + RTT_1 + \Lambda + RTT_n$   
c)  $RTT_1 + \Lambda + RTT_n + 2RTT_0 + RTT_0 = 3RTT_0 + RTT_1 + \Lambda + RTT_n$
8. HTTP/1.0: GET, POST, HEAD. P63 当浏览器请求一个对象时，使用 GET 方法。HTTP 客户机常常在用户提交表单时使用 POST 方法，例如用户向搜索引擎提供搜索关键词。实体中包含的就是用户在表单字段中的输入值。当服务器收到 HEAD 方法的请求时，会用一个 HTTP 报文进行响应，但是并不返回请求对象。应用程序开发者常用 HEAD 方法进行故障跟踪。HTTP/1.1: GET, POST, HEAD, OPTIONS, PUT, DELETE, TRACE, CONNECT. See RFCs for explanations. PUT 方法常与 Web 发布工具联合使用，它允许用户把对象上传到指定 Web 服务器的指定路径下。PUT 方法也被那些需要向 Web 服务器上传对象的应用程序使用。DELETE 方法允许用户或者应用程序删除 Web 服务器上的对象。
9. a) 通过一个传输速率为 R 的链路传输长度为 L 的对象需要的时间是 L/R。平均时间是对象的平均大小除以 R:  $\Delta = (900,000 \text{ bits}) / (1,500,000 \text{ bits/sec}) = 0.6 \text{ sec}$  链路的流量强度是:  $\beta \Delta = (1.5 \text{ requests/sec})(0.6 \text{ sec/request}) = 0.9$ . 因此，平均访问时延是:  $\Delta / (1 - \beta \Delta) = (0.6 \text{ sec}) / (1 - 0.9) = 6 \text{ seconds}$ . 因此，总的平均响应时间是:  $6 \text{ sec} + 2 \text{ sec} = 8 \text{ sec}$ .  
b) 因为有 40% 的请求有机构的网络满足，所以访问链路的流量强度减少了 40%。因此平均访问时延是:  $(0.6 \text{ sec}) / [1 - (0.6)(0.9)] = 1.2 \text{ seconds}$  如果请求由缓存器满足的话，其响应时间近似为 0。当缓存器未命中时，平均响应时间是  $1.2 \text{ sec} + 2 \text{ sec} = 3.2 \text{ sec}$  因此平均响应时间是:  $(0.4)(0 \text{ sec}) + (0.6)(3.2 \text{ sec}) = 1.92 \text{ seconds}$  因此平均响应

时间由 8sec 减少到 1.92sec。

10. 无。

11. UIDL 是唯一识别码列表的缩写。当一个 POP3 客户端发出一个 UIDL 命令，服务器返回储存在用户邮箱里的所有邮件的唯一邮件识别码。这个命令对下载并保留方式有用。通过保留上次收取的邮件的列表信息，客户能够使用 UIDL 命令来确定在服务器上的哪些邮件是已经被阅读过的。

12. a) C: dele 1

C: retr 2

S: (blah blah ...

S: .....blah)

S: .

C: dele 2

C: quit

S: +OK POP3 server signing off

b) C: retr 2

S: blah blah ...

S: .....blah

S: .

C: quit

S: +OK POP3 server signing off

c) C: list

S: 1 498

S: 2 912

S: .

C: retr 1

S: blah .....

S: ....blah

S: .

C: retr 2

S: blah blah ...

S: .....blah

S: .

C: quit

S: +OK POP3 server signing off

13. a) 对于一个给定的域名，IP 地址或网络管理员名的输入，whois 数据库能被用来定位相应的登记人，whois 服务器，DNS 服务器等。

f) 一个入侵者能使用 whois 数据库和 nslookup 工具来检测目标机构的 IP 地址范围，DNS 服务器地址等。

g) 通过分析攻击包的源地址信息，受害者能够使用 whois 来掌握有关于攻击来源的域的信息，并能够通知来源域的管理员。

14. 因为是全双工链路，你在每个方向都有 128kbps，上载不会影响下载。然而，对于不对称链路，由于 metered acks 上载能够显著的减少下载速率。

15. 在覆盖网络中有 N 个节点和  $N(N-1)/2$  条边。

16. a) 在这种情况下，这五个 Gnutella 客户都立即知道它们少了一个邻居。考虑五个客户中的一个，比如 Bob。假设当 X 离开后，Bob 只有三个邻居。这是 Bob 需要同另一个对等点建立 TCP 连接。Bob 要有一个活跃对等点的最新列表；他不断地连接列表中地对等点知道其中一个接收它的 TCP 连接请求。

b) 在这种情况下，Bob 不能立即知道 X 已经离开了。只有当他尝试向 X 发送信息（query 或 ping）时，Bob 才会知道 X 已经离开。当 Bob 尝试发送信息时，Bob 的 TCP 将会产生数个不成功的连接信息。这时 Bob 的 TCP 将会通知 Gnutella 客户机 X 已经离开。然后 Bob 将会尝试与一个新的对等点建立 TCP 连接，以此重建第五个连接。

17. a) 在直接连接 Bob 和 Alice 的 TCP 上传送 QueryHit 信息的优点是 QueryHit 信息在因特网的基本路由上传送，没



- 有经过中间对等点，因此，从 Bob 到 Alice 的传送信息的时延要短。缺点是每个有匹配信息的对等点都要求 Alice 打开一个 TCP 连接；因此 Alice 或许不得不为一个查询打开数十或数百个 TCP 连接。并且，当 Alice 在 NAT 后面时情况会更复杂。
- b) 当一个 Query 信息到达一个对等点时，该对等点就将 MessageID 和与信息到达相关的 TCP 套接字记录在列表中。当该对等点接收到带有同样 MessageID 的 QueryHit 信息时，它就检索列表来查询到它应该将信息发往哪个套接字。
- c) 当 Query 信息到达 Bob 时，它将包含信息从 Alice 到达 Bob 所经过的所有对等点的 IP 地址的顺序列表。当 Bob 回传一个 QueryHit 信息时，它将把这个顺序列表拷贝到信息中，当一个对等点接收到 QueryHit 信息时，他就能用这个列表来决定它要发送的下一个对等点。
18. 对于 Ping/Pong 信息的状况答案没有改变，只需把 Query 信息变为 Ping 信息，QueryHit 信息变为 Pong 信息。
19. a) 每一个超级组长大约负责  $200^2 = 40,000$  个对等点。因此，我们将需要大约 100 个超级组长来支持 4 百万的对等点。
- b) 每个组长将储存其子对等方共享的所有文件的元数据；一个超级组长将储存其子组长所储存的所有元数据。一个普通对等方首先将发送一个 query 到它的组长。这个组长将以一个匹配回应，并有可能将这个 query 发送给它的超级组长。超级组长将回应一个匹配信息（通过覆盖网络）。超级组长还有可能进一步将这个 query 发送给其他的超级组长。
20. Alice 发送她的 query 到至多 N 个邻居。每个邻居又发送这个 query 到至多  $M=N-1$  个邻居。这些邻居中的每一个又将这个 query 发送到至多 M 个邻居，因此最大的查询报文数为：
- $$N + NM + NM^2 + \dots + NM^{(K-1)} = N(1 + M + M^2 + \dots + M^{(K-1)})$$
- $$= N(1-M^K)/(1-M) = N[(N-1)^K - 1]/(N-2).$$
21. a) 如果先运行 TCP 客户机，那么客户机将试图与不存在的服务器进程建立 TCP 连接。TCP 连接将无法完成。
- b) UDP 客户机不和服务器建立 TCP 连接。因此，先运行 UDP 客户机，再运行 UDP 服务器是可以的，不会出错。当客户机和服务器运行起来后，你可以使用该应用程序再客户机上输入一行。
- d) 如果使用了不同的端口号，那么客户机将会试图和一个错误的进程或一个不存在的进程建立 TCP 连接，将会出错。
22. See Web-server programming assignment for this chapter for guidance.
23. 在原来的行中，UDP 客户机在创建一个套接字时没有指定端口号，在这种情况下，编码让下面的操作系统选择一个端口号。在替换行中，当 UDP 客户机执行时，一个 UDP 套接字以端口号 5432 建立。UDP 服务器需要知道客户机端口号以便于它能够将分组回送给正确的客户机套接字。查看 UDP 服务器的编码我们就会看到客户机的端口号不是固定在服务器编码中的；相反，UDP 服务器通过拆开它从客户机接收到的数据报（使用 `getPort()`）来确定客户机的端口号。因此 UDP 服务器能与客户机的任何端口号协同工作，包括 5432。所以 UDP 服务器不需要修改。
- Before:
- Client socket = x (chosen by OS)
- Server socket = 9876
- After:
- Client socket = 5432
- Server socket = 9876

### 3 复习题

1. P127 源端口号为 y, 目的端口号为 x。
2. P131 应用程序开发者可能不想其应用程序使用 TCP 的拥塞控制, 因为这会在出现拥塞时降低应用程序的传输速率。通常, IP 电话和 IP 视频会议应用程序的设计者选择让他们的应用程序运行在 UDP 上, 因为他们想要避免 TCP 的拥塞控制。还有, 一些应用不需要 TCP 提供的可靠数据传输。
3. P131 是的, 应用程序开发者可以将可靠数据传输放到应用层协议中完成。但是这需要相当大的工作量和进行调试。
4. a) false b) false c) true d) false e) true f) false g) false
5. a) 20 bytes (110-90=20bytes) b) ack number = 90 P155 第一个包丢失, 发送第一个包之前的一个包的 ACK
6. P155 3 个报文段, 第一个报文段, 客户机到服务器, seq=43, ack=80; 第二个报文段, 服务器到客户机, seq=80, ack=44; 第三个报文段, 客户机到服务器, seq=44, ack=81。
7. R/2 P180 R/2
8. P176 错误, 其阈值将被设置为拥塞窗口目前值的一半 (乘性减)。

### 习题

1. A → S 源端口号: 467 目的端口号: 23  
 b) B → S 源端口号: 513 目的端口号: 23  
 c) S → A 源端口号: 23 目的端口号: 467  
 d) S → B 源端口号: 23 目的端口号: 513  
 e) Yes.  
 f) No.
2. P128 假设主机 A,B,C 的 IP 地址为 a,b,c. (a,b,c 各不相同)  
 到主机 A: 源端口=80, 源 IP 地址=b, 目的端口=26145, 目的 IP 地址=a;  
 到主机 C: 左边进程: 源端口=80, 源 IP 地址=b, 目的端口=7532, 目的 IP 地址=c;  
 到主机 C: 右边进程: 源端口=80, 源 IP 地址=b, 目的端口=26145, 目的 IP 地址=c;
3. P132 UDP 检查和

```

01010101
+01110000
11000101
11000101
+01001100
00010001

```

1 的补码=11101110

为了检测错误, 接收方将四个字相加 (三个原始字和一个检测字)。如果结果包含 0, 那么接收方就知道分组中存在错误。所有的 1bit 错误都将被检测出来, 但是年年个比特的错误有可能被忽略 (例如, 如果第一个字的最后一个数变为 0, 并且第二个字的最后最后一个数变为 1)。

4. P138 假设发送方处于“等待来自上层的调用 1”状态, 接收方处于“等待来自下层的 1”。发送方发送一个带有序列号 1 的分组, 然后转到“等待 ACK 或 NAK1”的状态等待 ACK 或 NAK。假设现在接收方正确接收到带有序列号 1 的分组, 发送一个 ACK, 然后转入“等待来自下层的 0”状态, 等待带有序列号 0 的分组。然而, ACK 出错了。当 rdt.2.1 发送方接收到出错的 ACK, 它就重发带有序列号 1 的分组。然而, 接收方在等待带有序列号 0 的分组并在它没有收到带有序列号 0 的分组时一直发送 NAK。因此发送方会一直发送带有序列号 1 的分组, 这时接收方会一直发送这个分组的 NAK 信息。两边都不会从这个状态中跳出, 进入了死循环。
5. P140 为了回答这个问题, 首先考虑为什么我们需要序列号。我们看到发送方需要序列号以便于接收方能够区分出一个分组是不是已经接收到的分组的重复。考虑 ACK 信息, 发送方不需要这个信息 (也就是一个 ACK 的序列号) 来告诉发送方检测到一个重复的 ACK。因为当他接收到原始 ACK 信息后它就转入下一个状态, 所以一个重复的 ACK 信息对 rdt3.0 的发送方是很明显的。重复的 ACK 信息不是发送方需要的 ACK 信息, 因此被 red3.0

发送方忽略了。

6. P139 rdt3.0 协议的发送方与 rdt2.2 协议的发送方的不同之处在于引入了超时机制.我们已经看到超时机制地引入增加了从发送方到接收方数据流中出现重复分组地可能性.然而,rdt2.2 协议地接收方已经能够处理重复分组.(在 rdt2.2 中当接收方发送地 ACK 丢失时,发送方就会重传旧的数据.这时接收方就会接收到重复的分组.) 因此 rdt3.0 中的接收方同 rdt2.2 中的接收方相同.
7. Suppose the protocol has been in operation for some time. The sender is in state “Wait for call from above” (top left hand corner) and the receiver is in state “Wait for 0 from below”. The scenarios for corrupted data and corrupted ACK are shown in Figure 1.

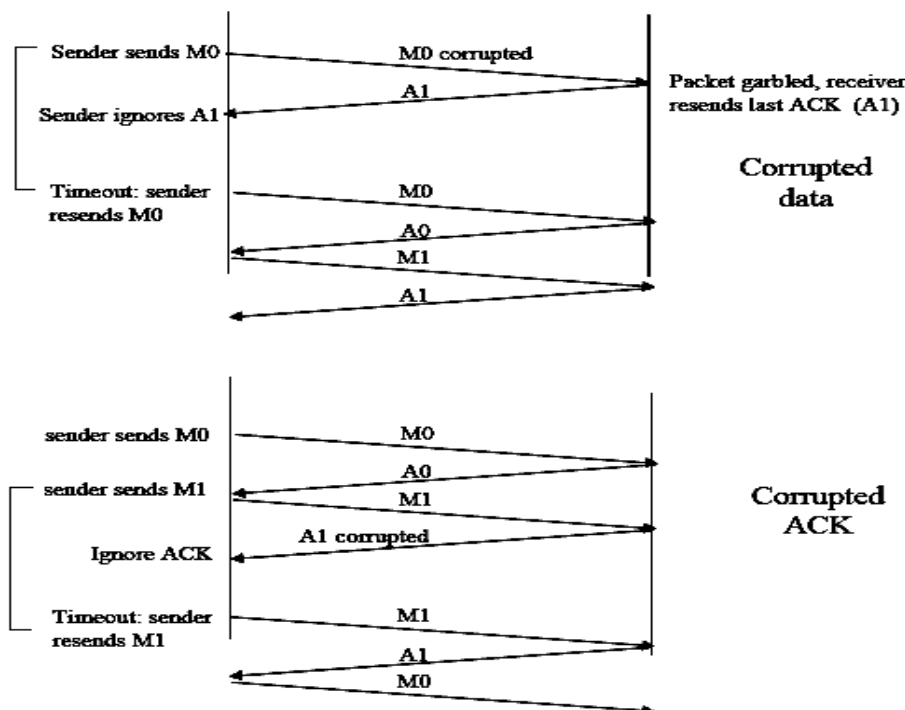
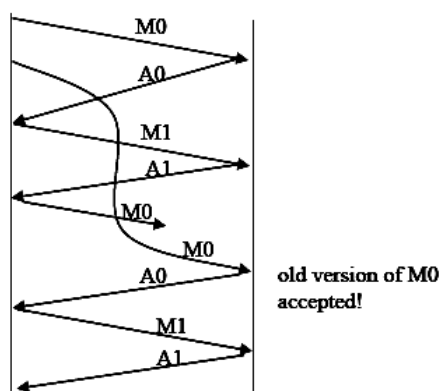
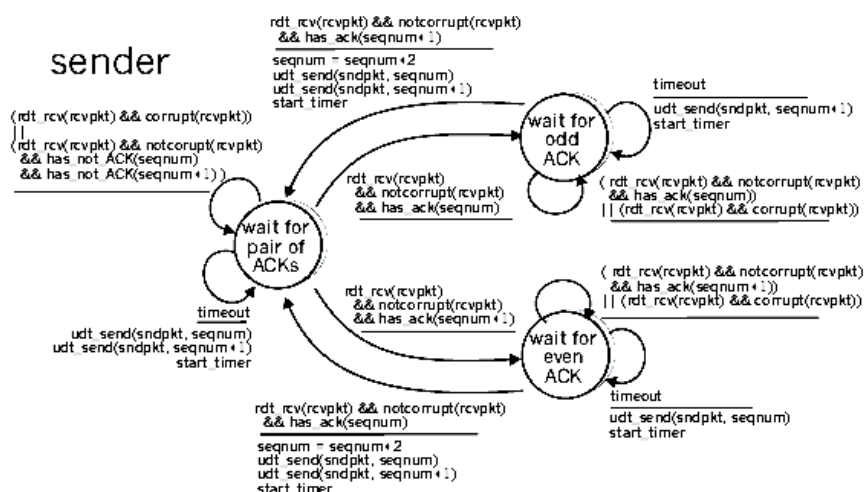


Figure 1: rdt 3.0 scenarios: corrupted data, corrupted ACK

8. P138 这里,我们加入一个定时器,它的值比我们已知的往返传播时延大.我们在”等待 ACK 或 NAK0”和”等待 ACK 或 NAK1”状态各加入一个超时事件.如果超时事件出现,那么最后传输的分组将被重传.让我们看看为什么这个协议仍然能和 rdt2.1 的接收方协同工作.假设超时是由数据分组的丢失引起的,比如,一个从发送方到接收方的信道上的分组丢失.在这种情况下,接收方从没有接收过之前传送的分组,从接收方的角度看,如果超时重传的分组被接收到,它看起来就和最初传输的分组被接收是一样的.现在假设一个 ACK 丢失.接收方最终将由于超时重传分组.但是这个重传动作是和当一个错误 ACK 出现时的重传动作是完全相同的.因此当出现 ACK 丢失或出现错误 ACK 时,发送方的重传动作是一样的.rdt2.1 的接收方已经能够处理出现错误 ACK 的状况.
9. 协议仍将工作,因为如果接收到的带有错误的分组实际上被丢掉的话,重传就会发生(从接收方的观点看,这两种情况哪一个会发生,或者同时发生是不可知的).要对此问题进行更进一步的讨论,就必须考虑到定时器超时过早发生的情况.在这种情况下,如果每个超大分组被确认,并且每个接收的超大分组确认信息导致另一个超大分组被发送,当  $n$  趋近于无穷时,分组  $n$  被发送的次数将无限增加.
- 10.



11. 在仅使用 NAK 的协议中,只有当接收到分组  $x+1$  时才能检测到分组  $x$  的丢失.也就是说接收方接收到  $x-1$  然后接收到  $x+1$ ,只有当接收方接收到  $x+1$  时才发现  $x$  的丢失.如果在传输  $x$  和传输  $x+1$  之间有很长时间的延时,那么在只有 NAK 的协议中, $x$  的修复要花费很长的时间.另一方面,如果要发送大量的数据,那么在只有 NAK 的协议中修复的速度将很快.并且,如果错误很少,那么 NAK 只是偶尔发送,并且从不发送 ACK.与只有 ACK 的情况相比,只有 NAK 的情况将明显减少反馈时间.
12. It takes 8 microseconds (or 0.008 milliseconds) to send a packet. in order for the sender to be busy 90 percent of the time, we must have  $util = 0.9 = (0.008n) / 30.016$  or  $n$  approximately 3377 packets.
13. 在 GBN 可靠数据传输协议中,发送方持续发送分组直到它接收到一个 NAK.如果到  $n-1$  之前的分组已经被正确的接收,这个 NAK 只是为分组  $n$  产生.也就是说, $n$  总是未被接收的分组的最小序号.当发送方接收到分组  $n$  的 NAK,它从分组  $n$  开始重传.这和书上讲的 GBN 协议相类似,除了在流水线上没有未被确认分组的最大数.注意发送方不能确切的直到有多少分组未被确认.如果当前的序号是  $k$ ,最后一个 NAK 是分组  $n$  的,那么在流水线上或许就有  $k-(n-1)$  的分组未被确认.还要注意接收方只有在接收到更高序列号的分组时才能确认分组  $n$  的丢失.(接收分组的序列号的空缺告诉接收方位于孔雀位置的分组丢失).因此,对于接收方,当数据速率低时,(比如,两个分组之间的时间比较长),将会比数据速率高时花费更长的时间来确认分组的丢失.
14. 在我们的解决方案中,发送方在接收到一对报文的 ACK(seqnum 和 seqnum+1)后才开始发送下一对报文.数据分组携带有两 bit 的序列码.也就是说,游泳的序列号是 0,1,2,3.ACK 信息携带已经确认的数据分组的序列号.接收方和发送方的 FSM 由下图所示.注意发送状态记录:(1)当前对没有收到 ACKs;(2)只收到 seqnum 的 ACK 或只收到 seqnum+1 的 ACK.在本图中,我们假设 seqnum 由 0 起始,发送方已经发送第一对数据.



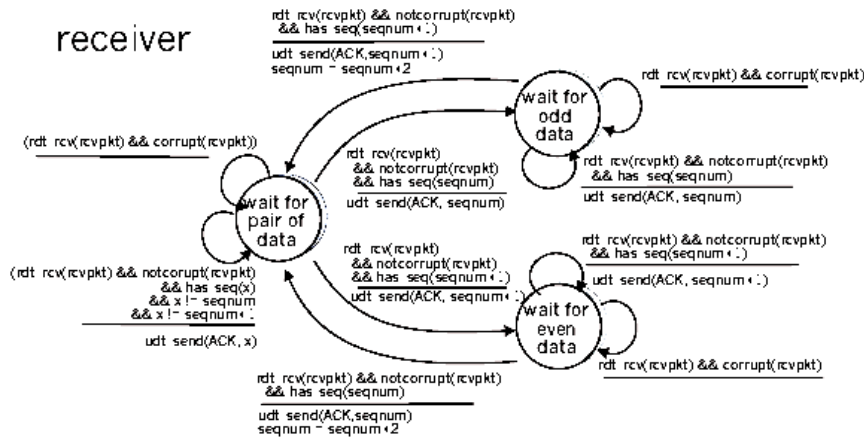


Figure 2: Sender and receiver for Problem 3.11

Sender

Receiver

```
make pair (0,1)
send packet 0
```

Packet 0 drops

```
send packet 1
```

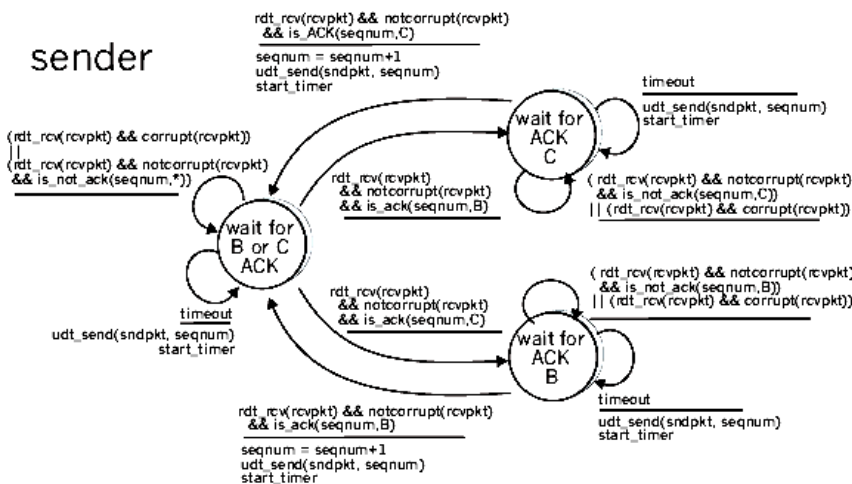
```
receive packet 1
buffer packet 1
send ACK 1
```

```
receive ACK 1
(timeout)
resend packet 0
```

```
receive packet 0
deliver pair (0,1)
send ACK 0
```

```
receive ACK 0
```

15. 这个问题是简单停等协议 rdt3.0 的变种.因为信道有可能丢失数据,还因为在其中一个接收方已经接收到数据的情况下发送方也有可能重传数据.(要么是因为定时器超时过早发生,要么是因为另一个接收方还没有正确的接收数据),所以序列号是必须的.就像是在 rdt3.0 中一样,在这里一个 0-bit 序列号是必须的.发送方和接收方的 FSM 如下图所示.在这个问题中,发送方状态显示发送方是否只从 B 接收到 ACK,只从 C 接收到 ACK,还是从 B 和 C 都没有接收到 ACK.接收方状态显示是否接收到的序列号是接收方等待的序列号.



## receiver B

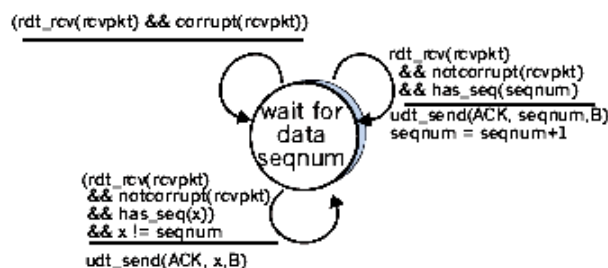


Figure 3. Sender and receiver for Problem 3.12

16. a) 在这里的窗口长度为  $N=3$ , 假设接收方已经接收到分组  $k-1$ , 并且已经对该分组以及之前的所有分组进行了确认。如果所有这些确认信息都已经被发送方接收, 那么发送方的窗口是  $[k, k+N-1]$ 。接下来假设所有这些 ACK 都没有被发送方接收。在这种情况下, 发送方的窗口包含  $k-1$  和直到包括  $k-1$  之前的  $N$  的分组。因此发送方的窗口是  $[k-N, k-1]$ 。因此, 发送方的窗口大小是 3, 并且从  $[k-N, k]$  中的某个数开始。
- b) 如果接收方在等待分组  $k$ , 那么它已经接收(并确认)了分组  $k-1$  以及在这之前的  $N-1$  个分组。如果所有这  $N$  的 ACK 都没有被发送方接收到, 那么值为  $[k-N, k-1]$  的 ACK 信息可能仍在回传。因为发送方已经发送了分组  $[k-N, k-1]$ , 那么发送方肯定已经接收到了分组  $k-N-1$  的 ACK。一旦接收方发送了分组  $k-N-1$  的 ACK, 它就不会在发送低于  $k-N-1$  的分组的 ACK。因此发送方接收到的所有报文的 ACK 字段的可能值为从  $k-N-1$  到  $k-1$ 。
17. 因为 A-to-B 信道会丢失请求报文, A 将需要超时 timeout 和重传请求信息。(为了能丢失中恢复过来)。因为信道时延是未知且变化的, A 有可能发送重复的请求(比如, 重传一个已经被 B 接收到的请求)。为了能够检测出重复的请求报文, 协议将使用序列号。1-bit 序列号就能够满足停/等类型的请求/响应协议的需求。
- “Wait for Request 0 from above” 这里请求方在等待一个来自上层的请求一个单元数据的命令。当它接收到来自上层的请求是, 它就向 B 发送一个请求报文 R0, 启动一个计时器并将状态转移到“Wait for D0”等待 D0 状态。当处于“Wait for Request 0 from above”状态是, A 忽略它从 B 接收到的所有信息。
  - “Wait for D0” 在这里请求方等待来自 B 的 D0 数据报文。A 的计时器在这个状态是一直运行的。如果计时器超时, A 重新发送一个 R0 报文, 重启计时器并保持这个状态。如果接收到来自 B 的 D0 报文, A 停止计时器, 并将状态转移到“Wait for Request 1 from above”等待来自上层的调用 1。如果 A 在这个状态接收到一个 D1 数据报文, 它将把此忽略。
  - “Wait for Request 1 from above” 在这里请求方再次等待来自上层的请求一个单元数据的命令。当它从上层接收到一个请求时, 它向 B 发送一个请求报文 R1, 启动计时器并将状态转移到“Wait for D1”。当处于 Wait for Request 1 from above 状态时, A 忽略从 B 接收到的任何数据。
  - “Wait for D1” 在这里请求方等待来自 B 的 D1 数据报文。在这个状态 A 的计时器一直在运行。如果计时器超时, A 将发送另一个 R1 报文, 重启计时器并保持这个状态。如果接收到来自 B 的 D1 报文, A 停止计时器, 并将状态转移到“Wait for Request 0 from above”在这个状态 A 将忽略重 B 接收到的信息。
- 数据提供方 B 只有两个状态:
- “Send D0” 在这种状态, B 持续通过发送 D0 来回应接收到的 R0 报文, 并保持这个状态。如果 B 接收到一个 R1 报文, 它就直到 D0 报文已经被正确接收。因此就将这个 D0 数据丢失(因为它已经被另一端接收)。并转移到“Send D1”状态。在这种状态它将用 D1 回应下一个请求报文。
  - “Send D1” 在这种状态, B 持续通过发送 D1 来回应接收到的 R1 报文, 并保持在这个状态。如果 B 接收到一个 R0 报文, 它就由此直到 D1 报文已经被正确接收, 并转移到“Send D1”状态。
18. 为了避免图 3.27 出现的情况, 我们要避免让接受者窗口的最前端(也就是具有最高序列号的那个)与发送窗口的最尾端(发送窗口中的具有最低序列号的那个)交迭在同一个序列号空间中。也就是说, 序列号空间必须足够大到让整个接收窗口和整个发送窗口在此序列号空间中不会出现交迭。因此, 我们需要测定在任何给定时刻由接收方和发送方覆盖的序列号有多大。假设接收方等待的最低序列号是分组  $m$  的序列号。在这种情况下, 接收方窗口是  $[m, m+w-1]$ , 并且它已经接收(并确认)了分组  $m-1$  和此前的  $w-1$  个分组, 这里  $w$  是窗口的尺寸。如果所有这  $w$  个 ACK 都没有被发送方接收到, 那么值为  $[m-w, m-1]$  的 ACK 报文将仍被传回。如果带有这些 ACK 号码的 ACK 都没有被发送方接收, 那么发送方的窗口将是  $[m-w, m-1]$ 。因此, 发送窗口的最低边界是  $m-w$ , 接收窗口的最大边界是  $m+w-1$ 。为了使接收窗口的前沿和发送窗口的后沿不出现交迭, 因此序列号空间必须大到能过容纳  $2w$  长度的序列号。也

就是说,序列号空间长度必须至少使窗口长度的两倍  $k \geq 2w$ .

19. a) 正确.假设发送方窗口大小为 3,在  $t_0$  时刻发送分组 1,2,3.在  $t_1(t_1 > t_0)$  时刻接收方确认 1,2,3.在  $t_2(t_2 > t_1)$  时刻发送方计时器超时,重发 1,2,3.在  $t_3$  时刻接收到重复的分组并重新确认 1,2,3.在  $t_4$  时刻发送方接收到接收方在  $t_1$  时刻发送的 ACK,并将其窗口前移到 4,5,6.在  $t_5$  时刻发送方接收到接收方在  $t_2$  发送的 ACK 1,2,3.这些 ACK 是在当前窗口之外的报文的 ACK.

b) 是的,本质上同 a 中是一样的.

c) True.

d) 正确.当窗口尺寸为 1 时,SR,GBN,的比特交替协议在功能上相同.窗口尺寸 1 排除了失序分组的可能性.在这种情况下,一个累积的 ACK 就是一个普通的 ACK.因为在窗口内它只能与一个分组有关.

20. 有  $2^{32} = 4,294,967,296$  个可能的序列号.

a) 序列号不会因报文数增加而有增量,而是随发送数据比特数量的增加而有增量.所以 MSS(最大报文长度)的大小与问题无关.能够从 A 发送到 B 的文件的最大尺寸能被  $2^{32} \approx 4.19$  Gbytes 所描述.

b) 报文数是:  $\lceil 2^{32}/1460 \rceil = 2,941,758$ .每个报文增加 66bytes 的首部,所以总共增加了  $2,941,758 \times 66 = 194,156,028$  bytes 的首部.要传输的总比特数为  $2^{32} + 194,156,028 = 3,591 \times 10^7$  bits  
因此使用 10Mbps 链路需要  $10 \times 10^6 = 3,591$  秒 = 59 分钟来传输文件.

21. Denote  $\text{EstimatedRTT}^{(n)}$  for the estimate after the  $n$ th sample.

$$\text{EstimatedRTT}^{(1)} = \text{SampleRTT}_1$$

$$\text{EstimatedRTT}^{(2)} = x\text{SampleRTT}_1 + (1-x)\text{SampleRTT}_2$$

$$\text{EstimatedRTT}^{(3)} = x\text{SampleRTT}_1 + (1-x)[x\text{SampleRTT}_2 + (1-x)\text{SampleRTT}_3]$$

$$= x\text{SampleRTT}_1 + (1-x)x\text{SampleRTT}_2 + (1-x)^2\text{SampleRTT}_3$$

$$\text{EstimatedRTT}^{(4)} = x\text{SampleRTT}_1 + (1-x)\text{EstimatedRTT}^{(3)}$$

$$= x\text{SampleRTT}_1 + (1-x)x\text{SampleRTT}_2 + (1-x)^2x\text{SampleRTT}_3 + (1-x)^3\text{SampleRTT}_4$$

b)

$$\text{EstimatedRTT}^{(n)} = x \sum_{j=1}^{n-1} (1-x)^j \text{SampleRTT}_j$$

$$+ (1-x)^n \text{SampleRTT}_n$$

感觉应该是  $j-1$

c)

$$\text{EstimatedRTT}^{(\infty)} = \frac{x}{1-x} \sum_{j=1}^{\infty} (1-x)^j \text{SampleRTT}_j$$

$$= \frac{1}{9} \sum_{j=1}^{\infty} 9^j \text{SampleRTT}_j$$

22. 让我们看看如果 TCP 测量重传报文的 SampleRTT 会出现什么情况.假设源发送分组 P1,P1 的定时器超时,源接着发送 P2---同一个分组的一个新的拷贝.进一步假设源测量 P2 的 SampleRTT(重传的分组).最后假设在传输 P2 后很快 P1 的 ACK 到达.源将错误的把这个 ACK 当作 P2 的 ACK,并计算出错误 SampleRTT 值.

23. SendBase:是最近未被确认的字节的序号.SendBase-1 是接收方已正确按序接收到数据的最后一个字节的序号.

LastByteRcvd:从网络中到达的并且已经放入主机 B 接收缓存中的数据流最后一个字节的编号.

在任一给定时刻  $t$ ,SendBase-1 是发送方知道的已经被接收方正确的按序接收的最后一个比特的序列号.在  $t$  时刻被接收方(正确的和按序的)接收到的真正的最后一个 byte 要比在链路上传输的 ACK 要大所以

$$\text{SendBase}-1 \leq \text{LastByteRcvd}$$

24. y:发送方接收到的最新 ACK 的值

在  $t$  时刻,发送方接收到的 ACK 的值为  $y$ ,据此发送方可以确认接收方已经接收了序号到  $y-1$  的数据.如果  $y \leq \text{SendBase}$  或在线路上有其他的 ACK,在  $t$  时刻接收方(正确的和按序的)接收到的真正的最后一个 byte 要比  $y-1$  大.所以  $y-1 \leq \text{LastByteRcvd}$

25. 假设发送了分组  $n, n+1, n+2$ ,并且分组  $n$  被接收并被确认.如果分组  $n+1$  和  $n+2$  在端到端的路径上被重新排序)也就是说,被以  $n+2, n+1$  的顺序接收),那么在收到第一个冗余 ACK 就重传的策略下,在接收到分组  $n+2$  时将产生一个



分组  $n$  的冗余 ACK, 这将引起重传发生。在接收到 3 个冗余 ACK 才执行重传的策略下, 只有在分组  $n$  后面的两个分组被正确接收, 然而分组  $n+1$  没有被接收的情况下, 重传才会发生。设计三个冗余 ACK 策略的设计者可能感觉等待两个分组(而不是一个)是在当需要时触发快速重传和当分组重新排序时不进行过早的重传之间权衡的结果。

26. P171 如果在图 3.45b 中到达速率的增加超过了  $R/2$ , 那么到达队列的总到达速率将超过队列的容量, 这将导致随着到达速率的增加包的丢失也随之增加。当到达速率  $= R/2$  时, 每 3 个离开队列的分组中就有 1 个是重传分组。随着丢失的增加, 甚至是离开队列的分组较大分片都将是重传数据。即使是将离开队列的最大离去速率给定为  $R/2$ , 并给定随到达速率的增加  $1/3$  或更多的数据将被传输。成功递交数据的吞吐量也不会超过  $\lambda_{out}$ 。同样的原因, 如果离开队列的分组中有一般的分组为重传分组, 并且输出分组的最大速率是  $R/2$ , 那么  $\lambda_{out}$  的最大值是  $(R/2)/2$  or  $R/4$ 。
27. P178 收到 3 个冗余 ACK 后, TCP 将拥塞窗口减小一般, 然后线性地增长。但是超时事件发生时, TCP 发送方进入一个慢启动阶段, 即它将拥塞窗口设置为  $1MSS$ , 然后窗口长度以指数速度增长。拥塞窗口持续以指数速度增长, 知道 CongWin 达到超时事件前窗口值地一半为止。此后, CongWin 以线型速率增长, 就像收到 3 个冗余 ACK 一样。
- 运行 TCP 慢启动的时间间隔是  $[1,6]$  和  $[23,26]$
  - 运行 TCP 避免拥塞时的时间间隔是  $[1,6]$  和  $[17,22]$
  - 在第 16 个传输周期后, 通过 3 个冗余 ACK 能够检测到一个报文段丢失。如果有一个超时, 拥塞窗口尺寸将减小为 1。
  - 在第 22 个传输周期后, 因为超时能够检测到一个报文段丢失, 因此拥塞窗口的尺寸被设置为 1。
  - Threshold 的初始值设置为 32, 因为在这个窗口尺寸是慢启动停止, 避免拥塞开始。
  - 当检测到报文段丢失时, threshold 被设置为拥塞窗口值的一半。当在第 16 个周期检测到丢失时, 拥塞窗口的大小是 42, 因此在第 18 个传输周期时 threshold 值为 21。
  - 当检测到报文段丢失时, threshold 被设置为拥塞窗口值的一半。当在第 22 个周期检测到丢失时, 拥塞窗口的大小是 26, 因此在第 24 个传输周期时 threshold 值为 13。
  - 在第一个传输周期内, 报文段 1 被传送; 在第二个传输周期发送报文段 2-3; 在第三个传输周期发送报文段 4-7, 在第四个传输周期发送 8-15; 在第五个传输周期发送 16-31; 在第六个传输周期发送 32-63; 在第七个传输周期发送 64-96。因此, 报文段 70 在第 7 个传输周期内发送。
  - 当丢失出现时拥塞窗口和 threshold 的值被设置为目前拥塞窗口长度 8 的一般。因此新的拥塞窗口和 threshold 的值为 4。

28.

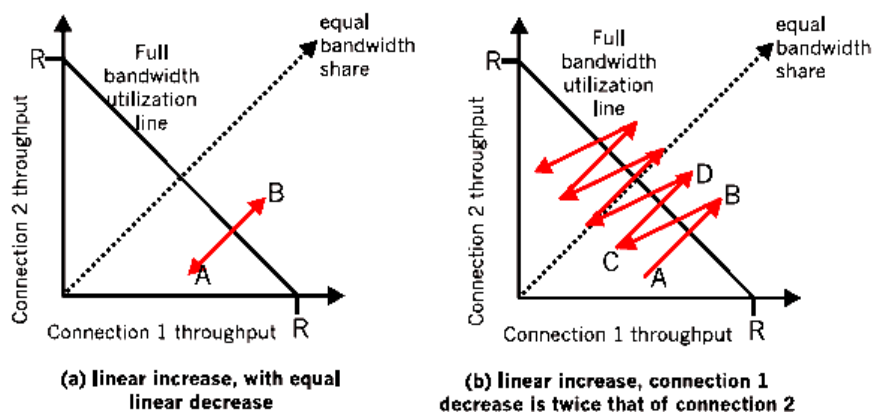


Figure 4: Lack of TCP convergence with linear increase, linear decrease

在图 4 (a) 中, 连接 1 和连接 2 因为丢包导致的线性减小的速率和他们线性增加的速率相同并相等。在这种情况下, 吞吐量不会从 AB 间相连的线段上移开。在图 4 (b) 中, 因丢包导致的连接 1 和连接 2 线性减小的速率之比为 2:1。也就是说, 不论何时出现丢包时, 连接 1 窗口减小量是连接 2 的两倍。我们看到最终, 在经过足够多的丢包和随后的增加后, 连接 1 的吞吐量将趋向 0, 全部链路带宽将被分配给连接 2。

29. 如果 TCP 是停等协议, 那么将超时间隔加倍作为拥塞控制机制已经足够。然而, TCP 使用流水线 (因此不是停等协议), 这允许发送方有数倍的未被确认的报文段。当端到端路径高度拥塞时, 将超时间隔加倍不会阻止 TCP 发送方在第一次发送时发送大量报文段。因此就需要一种拥塞控制机制, 当出现网络拥塞的迹象时, 阻止“接收来自上层应用的数据”



30. 在这个问题中，因为接收方的接收缓存能够容纳整个文件，因此不会出现接收方溢出的危险。并且，因为不会出现分组丢失和定时器超时，TCP 的拥塞控制不会抑制发送方，所以不需要拥塞控制。然而，主机 A 的进程不会持续的向套接字发送数据，因为发送方的缓存将很快被填满。一旦发送方缓存被填满，进程就会以平均速率  $R \ll S$  传送数据。所以需要 TCP 流量控制。
31. a) 丢包率  $L$ ，是丢失的包数量与发送包数量的比率。在一个循环中，有 1 个包丢失，在这个循环中发送的包数是：

$$\begin{aligned}
 \frac{W}{2} + \left(\frac{W}{2} + 1\right) + \Lambda + W &= \sum_{n=0}^{W/2} \left(\frac{W}{2} + n\right) \\
 &= \left(\frac{W}{2} + 1\right) \frac{W}{2} + \sum_{n=0}^{W/2} n \\
 &= \left(\frac{W}{2} + 1\right) \frac{W}{2} + \frac{W/2(W/2 + 1)}{2} \\
 &= \frac{W^2}{4} + \frac{W}{2} + \frac{W^2}{8} + \frac{W}{4} \\
 &= \frac{3}{8}W^2 + \frac{3}{4}W
 \end{aligned}$$

因此，丢包率为：

$$L = \frac{1}{\frac{3}{8}W^2 + \frac{3}{4}W}$$

- b) 因为  $W$  很大， $3/8W^2 \gg 3/4W$ 。所以  $L \approx 8/3 W^2$  or  $W \approx (8/3L)^{1/2}$ 。因此，我们得到  
平均吞吐量  $= 3/4 (8/3L)^{1/2} \cdot MSS/RTT = 1.22 MSS/RTT(L)^{1/2}$
32. P180 考虑一条具有 1500 字节报文段和 100ms RTT。从 P180 的 TCP 吞吐量等式，我们得到：  
 $10 \text{ Gbps} = 1.22 \times (1500 \times 8 \text{ bits}) / (0.1 \text{ sec} \times \text{sqrt}(L))$   
 Or  $\text{sqrt}(L) = 14640 \text{ bits} / (10^9 \text{ bits}) = 0.00001464$   
 Or  $L = 2.14 \times 10^{-10}$
33. 将在  $t_1$  时刻的 CongWin 和 Threshold 值用在  $t_2$  时刻的优点是 TCP 不需要经过慢启动的避免拥塞阶段即可直接跳到在  $t_1$  时刻得到的吞吐量值。使用这些值的缺点是它们可能已经不正确了。比如，如果路径在  $t_1$  到  $t_2$  的时间内变得更拥塞了，发送方会将大量的窗口内的有用报文段发送到已经更加拥塞的路径上去。
34. P183 最小传输时延是  $2RTT + O/R$ 。取得该时延的最小窗口长度  $W$  为：

$$\begin{aligned}
 W &= \min \left\{ w : w \geq \frac{RTT + S/R}{S/R} \right\} \\
 &= \left\lceil \frac{RTT}{S/R} \right\rceil + 1.
 \end{aligned}$$

R	min latency	W
28 Kbps	28.77 sec	2
100 Kbps	8.2 sec	4
1 Mbps	1 sec	25
10 Mbps	0.28 sec	235

35. a)  $K$  为涵盖对象的窗口的数量。(P186)  
 $K = \text{number of windows that cover the object}$   
 $= \min \{k : 3^0 + 3^1 + \Lambda + 3^{k-1} \geq O / S\}$   
 $= \min \{k : (1-3^k)/(1-3) \geq O / S\}$

$$= \min \{k : 3^k \geq 1 + 2O/S\}$$

$$= \lceil \log_3(1+2O/S) \rceil$$

b) Q is the number of times the server would idle for an object of infinite size.

Q 为当对象包含无数个报文段时服务器可能停滞的次数。

$$Q = \max \{k : RTT + S/R - S/R \cdot 3^{k-1} \geq 0\} = 1 + \lceil \log_3(1 + RTT/(S/R)) \rceil$$

c)

$$\begin{aligned} \text{latency} &= \frac{O}{R} + 2RTT + \sum_{k=1}^P \text{stall}_k \\ &= \frac{O}{R} + 2RTT + \sum_{k=1}^P \left( RTT + \frac{S}{R} - \frac{S}{R} 3^{k-1} \right) \\ &= \frac{O}{R} + 2RTT + P(RTT + S/R) - \frac{(3^P - 1)S}{2R} \end{aligned}$$

36.

R	O/R	P	Min latency	Latency with slow start
28 Kbps	29.25 s	3	31.25 sec	33.18 sec
100 Kbps	8.19 s	5	10.19 sec	13.86 sec
1 Mbps	819 msec	7	2.81 sec	9.26 sec
10 Mbps	82 msec	7	2 sec	9 sec

设 S=536 字节

$$P = \min \{Q, K-1\}$$

$$Q = 1 + \lceil \log_2(1 + RTT/(S/R)) \rceil$$

$$K = \lceil \log_2(1 + 2O/S) \rceil$$

$$\text{Min latency} = O/R + 2RTT$$

$$\text{Latency with slow start} = 2RTT + O/R + P[RTT + S/R] - (2^P - 1)S/R \quad \text{答案计算有误}$$

R	O/R	P	Min latency	latency with slow start
28Kbps	28.6s	3	30.6	32.99
100Kbps				
1Mbps				
10Mbps				

37. a) T

b) T

c) F

d) F

38. a)

$$Q = \max \left\{ k : RTT + \frac{S}{R} - \frac{S}{R} 2^{k-1} \geq 0 \right\}$$

$$\begin{aligned}
&= \max \left\{ k : 2^{k-1} \leq 1 + \frac{RTT}{S/R} \right\} \\
&= \max \left\{ k : k \leq \log_2 \left( 1 + \frac{RTT}{S/R} \right) + 1 \right\} \\
&= \left\lfloor \log_2 \left( 1 + \frac{RTT}{S/R} \right) \right\rfloor + 1
\end{aligned}$$

b)

$$\begin{aligned}
\text{latency} &= \frac{O}{R} + 2RTT + \sum_{k=1}^P \text{stall}_k \\
&= \frac{O}{R} + 2RTT + \sum_{k=1}^P \left( RTT + \frac{S}{R} - \frac{S}{R} 3^{k-1} \right) \\
&= \frac{O}{R} + 2RTT + P(RTT + S/R) - \frac{(3^P - 1) S}{2 R}
\end{aligned}$$

左式中 3 改为 2

P 为实际停滞次数。

$$[X]^+ = \max(x, 0)$$

$$\text{时延} = 2RTT + O/R + \sum_{k=1}^{K-1} [S/R + RTT - 2^{k-1} S/R]^+$$

由 P185 页的时延等式，我们注意到只有在  $k \leq Q$  时，方括号中的值才不等于 0。因此我们可以用 P 代替 K-1，并移除[...] 的限制。将恒等式应用到等式中得到希望的结果。

39. P186 当服务器发送一个报文段，它接收到 ACK 需要等  $TS/R + RTT$ 。第 k 个窗口的传输时间为  $(S/R)2^{k-1}$ 。第 k 个窗口的停滞时间为：

$$\left[ \frac{TS}{R} + RTT - 2^{k-1} \frac{S}{R} \right]^+$$

可能的停滞次数 Q 为：

$$\begin{aligned}
Q &= \max \left\{ k : RTT + \frac{TS}{R} - \frac{S}{R} 2^{k-1} \geq 0 \right\} \\
&= \max \left\{ k : 2^{k-1} \leq T + \frac{RTT}{S/R} \right\} \\
&= \max \left\{ k : k \leq \log_2 \left( T + \frac{RTT}{S/R} \right) + 1 \right\} \\
&= \left\lfloor \log_2 \left( T + \frac{RTT}{S/R} \right) \right\rfloor + 1
\end{aligned}$$

服务器停滞的实际次数  $P = \min(Q, K-1)$ ，时延为：

$$\text{latency} = 2RTT + \frac{O}{R} + \sum_{k=1}^P \left( RTT + \frac{TS}{R} - \frac{S}{R} 2^{k-1} \right)$$

简化为:

$$\text{latency} = 2RTT + \frac{O}{R} + P\left(RTT + \frac{TS}{R}\right) - (2^P - 1)\frac{S}{R} + (T - 1)\frac{S}{R}$$

40. The fraction of the response time that is due to slow start is (P188)

$y/(x+y)$  where  $x = 2 RTT + O/R$  and  $y = P (RTT + S/R) - (2^P - 1) S/R$

## 4 复习题

1. 网络层的分组名称是数据报. 路由器是根据包的 IP 地址转发包; 而链路层是根据包的 MAC 地址来转发包.
2. 数据报网络中网络层两个最重要的功能是: 转发, 选路. 虚电路网络层最重要的三个功能是: 转发, 选路, 和呼叫建立.
3. P200 转发是当一个分组到达路由器的一条输入链路时, 该路由器将该分组移动到适当的输出链路. 选路是当分组从发送方流向接收方时, 网络层必须决定这些分组所采用的路由或路径.
4. 是, 都使用转发表, 要描述转发表, 请参考 4.2 节. 在虚电路网络中, 该网络的路由器必须为进行中的连接维持连接状态信息. 每当跨越一台路由器则创建一个新连接, 一个新的连接项必须加到该路由器转发表中; 每当释放一个连接, 必须从该表中删除该项. 注意到即使没有 VC 号转换, 仍有必要维持连接状态信息, 该信息将 VC 号与输出接口号联系起来. 每当一个端系统要发送分组时, 它就为该分组加上目的地端系统的地址, 然后将该分组推进网络中. 完成这些无需建立任何虚电路. 在数据报网络中的路由器不维护任何有关虚电路的状态信息. 每个路由器有一个将目的地址影射到链路接口的转发表; 当分组到达路由器时, 该路由器使用该分组的地址在该转发表中查找适当的输出链路接口. 然后路由将其该分组项该输出链路接口转发. 虽然在数据报网络中不维持连接状态信息, 它们无论如何在其转发表中维持了转发状态信息. 在数据报网络中的转发表是由选录算法修改的, 通常每 1 到 5 分钟左右更新转发表. 在虚电路网络中, 无论何时通过路由器拆除一条现有的连接, 路由器中的转发表就更新.
5. P202 单个分组: 确保交付; 具有延时上界的确保交付. 分组流: 有序分组交付; 确保最小带宽; 确保最大时延抖动. 因特网的网络层不提供这些服务. ATM 的 CBR (恒定比特率) 服务同时提供确保交付和计时. ABR (可用比特率) 不提供该假想服务.
6. 交互式实时多媒体应用, 如: IP 电话和视频会议. 这些应用都得益于 ATM 的 CBR 服务的实时性.
7. 正是由于有影子拷贝, 在每个输入端口的转发都由本地决定, 而不用调用中心选路处理器. 这种分散的转发方式避免了在路由器的某一个节点出现转发处理的瓶颈.
8. P211 (1) 经内存交换: 在输入和输出端口之间的交换是在 CPU 控制下完成的. 输入与输出端口的作用就像在传统操作系统中的 I/O 设备一样. 一个分组到达一个输入端口, 该端口会先通过中断方式向选路处理器发出信号. 于是, 该分组就被拷贝到处理器内存中. 选路处理器从分组首部中取出目的地址, 在转发表中找出适当的输出端口, 并将该分组拷贝到输出端口的缓存中. (2) 经一根总线交换: 输入端口经一根总线将分组直接传送到输出端口, 不需要选路处理器的干预. 由于总线是共享的, 故一次只能有一个分组通过总线传送. (3) 经一个互联网络交换: 使用一个纵横的网络, 是一个由  $2n$  条总线组成的互联网络, 它将  $n$  个输出端口和  $n$  个输入端口连接, 一个到达某个端口的分组沿着连到输出端口的水平总线穿行, 直至该水平总线与连到所希望的输出端口的垂直总线之交点.
9. P213 如果在输入端口因为交换结构速率慢而引起队列长度的加大, 最终将路由器的缓存空间耗尽, 就会出现“分组丢失”. 如果交换结构速率大于线路速率的  $n$  倍 ( $n$  是输入端口的数量) 就可以消除分组丢失的问题.
10. 因为输出线速率慢而导致输出端队列长度加大, 最终将耗尽输出端口的存储空间, 在这样的情况下, 分组就被丢弃了.
11. HOL 阻塞是在一个输入队列中的一个分组因为被位于线头的另一个分组阻塞, 即使输出端口是空闲的, 也必须等待线头分组发送完了才能通过交换结构发送. (中文版 P215 图 4-11) 它发生在输入端口.
12. 有, 每个接口都有一个 IP 地址.
13. 11011111.00000001.00000011.00011011
14. 电脑上试验? 不会考吧, 略……
15. 通过 8 个接口, 要检索 3 次转发表.
16. P218 首部占 50% (一般数据报承载共 40 字节首部 (20 字节 IP 首部加上 20 字节 TCP 首部)).
17. IP 数据报内的 8 个 bit 的协议字段包含了目标主机应该将报文段交给传输层哪个协议的信息.
18. P225 典型的无线路由器都包含一个 DHCP (动态主机配置协议) 服务, DHCP 可以用来为这 5 台 PC 机自动分配地址以及路由器接口. 是的, 无线路由器也用 NAT (网络地址转换) 来从 ISP 获得唯一的 IP 地址. 因为处于无线路由器覆盖下的主机的移动性比较大, 使用了 NAT 后使局域网作为一个封闭的网络, 其出口 IP 地址只有一个, 在局域网内部 NAT 路由器在使用 DHCP 为局域网中的主机分配 IP 地址. 这样处于一个路由器下的主机不会因为主机数的增减而频繁的向 ISP 申请 IP 地址. 增加了其适用性和扩展性.

19. P230 流量类型: 该 8 比特字段与我们在 IPv4 中看到的 TOS(服务类型)字段的含义相似。下一个首部: 该字段标识该数据报中内容(数据字段)需要交付给那个协议(如 TCP 或 UDP)。该字段使用与 IPv4 首部中协议字段相同的值。
20. 同意, 因为整个 IPv6 数据报(包括首部字段)在通过 IPv4 隧道时都是被一个 IPv4 数据报包裹着的。
21. 链路状态选路算法: 是用完整的、全局性的网络信息来计算从源到目的地直接的最低费用路径。距离向量选路算法: 以迭代的、分布式的方式计算出最低费用路径, 每个节点只能算出到达它要发给分组报文的相邻节点的最低费用路径, 然后通过迭代计算出到达目的节点的最低费用路径。
22. 每个自治系统由一组在相同管理者控制下的路由器组成, 在相同的 AS 内所有路由器运行同样的自治系统内部选路算法。不同 AS 的网关路由器运行自治系统间选路协议, 以决定不同 AS 之间的选路路由, 而 AS 内部路由器只需要知道到达其它内部路由器以及网关路由器的路由路径, 因此网络的规模扩展问题得以解决。
23. 不必要, 每个 AS 系统都有路由管理自治权(内部运行同样的算法即可)。不同 AS 的网关路由器运行自治系统间选路协议以决定不同 AS 之间的选路路由。各个 AS 内运行不同的自治系统内选路算法不会影响不同 AS 网关路由器的选路路由。
24. 注: 此题的图应该是 P248 图 4-32、33、34。不会改变, 因为来自 A 的通告告知 D 如果通过路由器 A 到达 Z 需要 11 跳, 而 D 通过 B 到达 Z 只需要 7 跳, 因此没有必要修改转发表中到达 Z 的相应条目。而如果通告指出 A 通过 C 只需要 4 跳就可以到达 Z, 那么 D 的转发表就应该做相应的修改。
25. 使用 OSPF(开放最短路径优先)时, 一个路由器周期性向自治系统内所有的其它路由器广播选路信息, 而不仅仅是向其相邻路由器广播。这个由路由器发出的路由信息中, 该路由器到每个邻近路由器的距离信息都显示为一个相应的条目。使用 RIP(选路信息协议)时, 一个路由器只向邻近的路由器发送通告, 通告中包括该路由器到 AS 内所有目的子网全部网络的信息(经过哪个路由器, 需要多少跳到达目的子网)。
26. 路径上 ASs 的顺序。
27. 因为 AS 内部选路和 AS 间选路存在选路目标上的差别(1)策略: 在 AS 之间, 策略问题时至关重要的, 而 AS 内部, 一切都是以相同的管理控制名义进行的, 因此策略问题在 AS 内部不太重要; (2)规模: 一个选路算法及其数据结构在处理大量网络的选路或大量网络之间的选路时的适应能力是 AS 间选路的一个关键问题, 而在 AS 内部, 可伸缩性是第二关心的问题; (3)性能: 由于 AS 间选路是面向策略的, 因此所用路由质量(如性能)通常是次要关心的问题, 而在 AS 内部, 选路要考虑的问题更多的集中在一条路由实现的性能级别上。
28. ISP C 可以通过 BGP 的 MED 建议 ISP B 从东岸对等点路径到达 ISP D。例如, ISP C 的东岸 BGP 路由器可以提供一个到 ISP D 的 MED 值为 5 的路由, 而 ISP C 的西岸可以提供一个到 ISP D 的 MED 值为 10 的路由, 由于是基于低代价选择路由, ISP B 知道 ISP C 选择从东岸接收通信。实际上, 路由器可以忽略 MED 值, 因此 ISP B 也可以用热土豆选路法从 ISP C 的西岸对等点向 ISP D 发送通信。(以上纯属直译, 本人不知所云, 晕 ing)。
29. 子网: 是一个大网络中的一部分, 一个子网内是不含路由器的, 它的边界由路由器和主机端口决定。前缀: 是一个 CDIR 化的地址的网络部分, 地址写成 a. b. c. d/x 的形式, 一个前缀覆盖一个或多个子网。BGP 路由: 当一个路由器通过 BGP 会话通告一个前缀时, 它随着前缀包括一些 BGP 属性。用 BGP 的术语来说, 带有属性的前缀被称为一个 BGP 路由。
30. NEXT-HOP 属性标明一条给定前缀的通告路径(在 AS 外接收通告)上的第一个路由器的 IP 地址。路由器在配置转发表时使用 NEXT-HOP 属性。AS-PATH: 路由器使用 AS-PATH 属性来检测和防止循环通告, 路由器也使用 AS-PATH 属性对相同的前缀在多条路径中进行选择。
31. 一个第一层的 ISP B 不会去传送其它两个第一层 ISP (A 和 C)之间的通信流, 这是由于 B 有同等的协议。要实现这一策略, ISP B 不会经过 C 发送到 A 路由的通告, 也不会经过 A 发送到 C 路由的通告。
32. N 次单播有几个缺点: (1)效率: 同一个分组的多份拷贝需经过同一条链路发送到可能的多个链路上。这样, 源节点必须生成同一分组的多份拷贝。(2)寻址: 源节点必须找到所有接收方的地址。
33. (a) 无控制洪泛: 真; 控制洪泛: 真; 生成树广播: 假。(b) 无控制洪泛: 真; 控制洪泛: 假; 生成树广播: 假。
34. 不用。
35. IGMP(互联网组管理协议)是只运行在一台主机和与其直接相连的多播路由器(第一跳多播路由器)之间的协议。IGMP 允许一个主机指定(到第一条多播路由器)一个想加入的多播组。然后, 该多播路由器与其它多播路由器一起工作(也就是运行一个多播路由协议), 以确保主机与所加入组相应的最后一跳路由器之间正常的数据通信。
36. 一棵组共享树, 所有的发送者用同一个路由选择树来发送它们的多播通信。而一棵基于源的树, 来自给定源的

1. a. 对于一个面相连接的网络来说, 路由器故障会影响到这个连接的所有路由. 至少需要出现故障的路由器上游的路由器重新建立一个新的到目标节点的下行部分的路径, 同时, 发出一个信息, 该信息包含所有与建立一个新的路径有关要求. 而且, 先前路径中出现故障的路由器下游的所有路由器都必须拆除这个故障连接, 同时发出另一个信息, 包含了与所要做的事情有关的所有要求.

b. 为了让路由器能够确定一条输出链路的延时(或延时的界限),就需要知道通过这条链路传输的所有会话通信的特性. 也就是说, 路由器必须知道内部每一个会话的状态, 这对于一个面相连接的网络是有可以能的, 但是对于一个面向无连接的网络则不可能. 此时面向连接的网络更加可取.

2.
  - a. 一个链路能够承载的最大虚电路数量= $2^{16}=65536$
  - b. 中心节点可以从 0 到 65535 中任取一个 VC 号. 这种情况下, 进行中的虚电路数量小于 65536 而没有相同的未用 VC 号是不可能的.
  - c. 每一个链路可以自由的从 0 到 65535 中分配一个 VC 号. 因此, 很可能一个虚电路每一个链路在它的路径上都有各不相同的 VC 号. 而虚电路路径上的每一个路由器都需要为到达的分组更换一个与输出链路有关的 VC 号.
3. 虚电路转发表中, 分别为: 入接口, 入 VC 码, 出接口, 出 VC 码. 数据报网络转发表中, 分别为: 目标地址, 出接口.
4.
  - a. 不能为新的虚电路分配 VC 号.
  - b. 每个链路有 2 个可用的 VC 号, 共 4 条链路, 因此共有  $2^4=16$  种不同的组合.
5. 在一个虚电路网络中, 存在一条端到端的连接, 而这条路径上的每个路由器都必须保持这个连接状态, 因此术语称作连接服务. 在一个基于无连接网络层的面向连接运输服务中 (如基于 IP 的 TCP 传输) 由终端系统保持连接状态, 而路由器并不清楚是怎样连接的, 因此, 术语中称作面相连接服务.
6. 为解释为什么会这样, 让我们来看一个实际的设计实例. 为简要起见, 假设所有的分组都是相同大小的. 交换系统是基于时分复用的: 时间被分为帧, 而每一帧又分为  $n$  个时隙, 而每个时隙结构中需要交换一个分组, 每一条输入线路对应一帧中的一个时隙, 由于输入线路中的每一帧最多只有一个分组到达, 因此交换结构可以处理每一帧的所有分组.

7. 前綴匹配	接口
11100000	0
11100001 00000000	1
11100001	2
其它	3

目标地址范围	接口
00000000 到 00111111	0
01000000 到 01111111	1
10000000 到 10111111	2
11000000 到 11111111	3

目标地址范围	接口
10000000 到 10111111 (64)	0
11000000 到 11011111 (32)	1
11100000 到 11111111 (32)	2
00000000 到 01111111 (128)	3





## NAT 转换表

WAN 侧	LAN 侧
126.13.89.67, 4000	192.168.0.1, 3345
128.119.40.86, 4000	192.168.0.1, 3345
128.119.40.86, 4001	192.168.0.1, 3346
128.119.40.86, 4002	192.168.0.2, 3445
128.119.40.86, 4003	192.168.0.2, 3446
128.119.40.86, 4004	192.168.0.3, 3545
128.119.40.86, 4005	192.168.0.3, 3546

18. a. 我们来分析一下当 Arnold 试图与 Bernard 建立一个 TCP 连接的时会发生什么. Arnold 发出一个 TCP SYN 分组, 带有目标地址 138.76.29.7 以及一些目标端口号, 如 x. 当 NAT 网收到这个 TCP SYN 分组时, 由于没有条目指明如何从 WAN 侧建立连接, NAT 不知道应该将这个分组送到内网的哪个主机. 因此, NAT 会丢弃这个 SYN 分组.
- b. 在 Arnold 和 Candy, Candy 和 Bernard 之间已经存在了 TCP 连接, 通过这两个 TCP 连接, Arnold 可以给 Bernard 发送一条信息, 更确切的说, Arnold 可以要求 Bernard 申请建立一条直接从 Bernard 到 Arnold 的 TCP 连接. 由于是 Bernard 要求发起连接, 因此, 可以经过 Barnard 的 NAT 建立这个连接. 只要在 Arnold 和 Barnard 之间建立了这条直接的 TCP 连接, Arnold 就可以要求 Bernard 通过这个直接的 TCP 连接传送文件了.
19. 不可能设计出这种技术. 为了直接在 Arnold 与 Barnard 之间建立 TCP 连接, 必须由 Arnold 和 Bernard 之一申请建立他们之间的 TCP 连接, 而覆盖 Arnold 和 Bernard 的 NAT 会将来自 WAN 侧到达的 SYN 分组丢弃, 因此如果 Arnold 和 Bernard 都在 NAT 之后, 则他们都不能申请建立一个 TCP 连接.

20.

u-v-w-z, u-v-x-y-z, u-v-x-w-z, u-v-x-w-y-z,  
u-x-y-z, u-x-w-z, u-x-y-w-z, u-x-w-y-z, u-x- v-w-z, u-x- v-x-y-z, u-x-v-x-w-z, u-x- v-x-  
w-y-z

21.

Step	N'	D(s),p(s)	D(t),p(t)	D(u),p(u)	D(v),p(v)	D(w),p(w)	D(y),p(y)	D(z),p(z)
0	x	$\infty$	$\infty$	$\infty$	3,x	1,x	6,x	$\infty$
1	xw	$\infty$	$\infty$	4,w	2,w		6,x	$\infty$
2	xwv	$\infty$	11,v	3,v			3,v	$\infty$
3	xwvu	7,u	5,u				3,v	$\infty$
4	xwvuy	7,u	5,u					17,y
5	xwvuyt	6,t						7,t
6	xwvuyts							7,t

22.

a) from node s to all nodes (note: ties broken in favor of leftmost column)

Step	N'	D(t),p(t)	D(u),p(u)	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	s	1,s	4,s	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$
1	st		3,t	t,10	$\infty$	$\infty$	5,t	3,t
2	stu			4,u	6,u	$\infty$	5,t	3,t
3	stuz			4,u	6,u	$\infty$	5,t	
4	stuzv				5,v	7,v	5,t	
5	stuzvw					6,w	5,t	
6	stuzvwy					6,w		

b) from node t to all nodes (note: ties broken in favor of leftmost column)

Step	N'	D(s),p(s)	D(u),p(u)	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	t	1,t	2,t	9,t	$\infty$	$\infty$	4,t	2,t
1	ts		2,t	9,t	$\infty$	$\infty$	4,t	2,t
2	tsu			3,u	5,u	$\infty$	4,t	2,t
3	tsuz			3,u	5,u	$\infty$	4,t	
4	tsuzv				4,v	6,v	4,t	
5	tsuzvw					5,w	4,t	
6	tsuzvwy					5,w		

c) from node u to all nodes (note: ties broken in favor of leftmost column)

Step	$N'$	$D(s),p(s)$	$D(t),p(t)$	$D(v),p(v)$	$D(w),p(w)$	$D(x),p(x)$	$D(y),p(y)$	$D(z),p(z)$
0	u	4,u	2,u	1,u	3,u	$\infty$	$\infty$	$\infty$
1	uv	4,u	2,u		2,v	4,v	2,v	$\infty$
2	ut	3,t			2,v	4,v	2,v	4,t
3	utw	3,t				3,w	2,v	4,t
4	utwy	3,t				3,w		4,t
5	utwys					3,w		4,t
6	utwysx							4,t

d) from node v to all nodes (note: ties broken in favor of leftmost column)

Step	$N'$	$D(s),p(s)$	$D(t),p(t)$	$D(u),p(u)$	$D(w),p(w)$	$D(x),p(x)$	$D(y),p(y)$	$D(z),p(z)$
0	v	$\infty$	9,v	1,v	1,w	3,v	1,v	$\infty$
1	vu	5,u	3,u		1,w	3,v	1,v	$\infty$
2	vuw	5,u	3,u			2,w	1,v	$\infty$
3	vumy	5,u	3,u			2,w		15,y
4	vumyx	5,u	3,u					15,y
5	vumyxt	4,t						6,t
6	vumyxts							6,t

e) from node w to all nodes (note: ties broken in favor of leftmost column)

Step	$N'$	$D(s),p(s)$	$D(t),p(t)$	$D(u),p(u)$	$D(v),p(v)$	$D(x),p(x)$	$D(y),p(y)$	$D(z),p(z)$
0	w	$\infty$	$\infty$	3,w	1,w	1,x	$\infty$	$\infty$
1	ww	$\infty$	10,v	2,v		1,x	2,v	$\infty$
2	wvx	$\infty$	10,v	2,v			2,v	$\infty$
3	wvxu	6,u	4,u				2,v	$\infty$
4	wvxuy	6,u	4,u					16,y
5	wvxuyt	5,t						6,t
6	wvxuyts							6,t

f) from node y to all nodes (note: ties broken in favor of leftmost column)

Step	$N'$	$D(s),p(s)$	$D(t),p(t)$	$D(u),p(u)$	$D(v),p(v)$	$D(w),p(w)$	$D(x),p(x)$	$D(z),p(z)$
0	y	$\infty$	4,y	$\infty$	1,y	$\infty$	6,y	14,y
1	yv	$\infty$	4,y	2,v		2,v	4,v	14,y
2	yvu	6,u	4,u			2,v	4,v	14,y
3	yvuw	6,u	4,u				3,w	14,y
4	yvuwx	6,u	4,u					14,y
5	yvuwxxt	5,t						6,t
6	yvuwxxts							6,t

g) from node z to all nodes (note: ties broken in favor of leftmost column)

Step	$N'$	$D(s),p(s)$	$D(t),p(t)$	$D(u),p(u)$	$D(v),p(v)$	$D(w),p(w)$	$D(x),p(x)$	$D(y),p(y)$
0	z	$\infty$	2,z	$\infty$	$\infty$	$\infty$	$\infty$	14,z
1	zt	3,t		4,t	11,t	$\infty$	$\infty$	6,t
2	zts			4,t	11,t	$\infty$	$\infty$	6,t
3	ztsu				4,u	7,u	$\infty$	6,t
4	ztsuv					6,v	8,v	6,t
5	ztsuvw						7,w	6,t
6	ztsuvwxy						7,w	

		Cost to				
		u	v	x	y	z
From	v	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$
	x	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$
	y	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$
	z	$\infty$	5	2	10	0

		Cost to				
		u	v	x	y	z
From	v	1	0	$\infty$	15	5
	x	2	$\infty$	0	1	2
	y	$\infty$	15	1	0	10
	z	4	5	2	3	0

		Cost to				
		u	v	x	y	z
From	v	1	0	3	15	5
	x	2	3	0	1	2
	y	3	15	1	0	3
	z	4	5	2	3	0

		Cost to				
		u	v	x	y	z
From	v	1	0	3	4	5
	x	2	3	0	1	2
	y	3	4	1	0	3
	z	4	5	2	3	0

24. 这个问题的用词是模棱两可的. 我们对它的理解是“从算法第一次进行计算开始的迭代次数”(也就是说, 假设某个节点最初只知道到达它的邻节点的费用). 我们假设算法是同步进行的(也就是说, 从同一时刻起, 所有节点同时计算出它们的距离表, 并交换表).

每一次迭代, 节点与它的邻节点交换距离表. 因此, 假设你是节点 A, 你的邻节点是 B, 在一次迭代之后, 所有 B 的邻节点(都是距离你 1 到 2 跳的节点)都会知道通向你的 1 跳或 2 跳的最少费用路径(在 B 告诉它的所有邻节点要到达你所需的费用之后).

用  $d$  表示网络的直径——网络中任意两个节点之间非环路的最长路径的长度. 经过  $d-1$  次迭代后, 所有的节点都会知道一个到达所有其它节点需要  $d$  跳或更少跳的最短路径. 由于任何多于  $d$  跳的路径必定含有环路(含有环路的路径比不含环路的路径费用更大), 因此, 该算法最多经过  $d-1$  次迭代后结束.

另外, 除非特别指定了链路费用的限制, 否则因为链路费用变化而调用 DV 算法, 不会影响到完成迭代计算所需迭代次数的限制.

25. a.  $D_x(y)=4$ ;  $D_x(w)=1$ ;  $D_x(u)=6$ .  
 b. 首先考虑如果  $c(x, y)$  变化. 不论  $c(x, y)$  变大或者变小(只要  $c(x, y)$  大于 0), 从  $x$  到  $u$  的最小费用路径仍然为经过  $w$ , 所需费用为 6. 所以,  $c(x, y)$  发生变化  $x$  不用通知任何邻节点发生了变化. 现在考虑  $c(x, w)$  变化时: 如果  $c(x, w)=m \leq 5$ , 那么到  $u$  的最小费用路径仍然为经过  $w$ , 而费用变为  $5+m$ .  $x$  将通知它的邻节点这一费用变化. 如果  $c(x, y)=n > 5$ , 那么最小费用路径变为经过  $y$ , 费用为 10;  $x$  将通知它的邻节点这一费用变化.  
 c.  $c(x, y)$  发生任何变化, 执行距离向量算法后,  $x$  将不通知其邻节点有一条通向  $u$  的新的最低费用路径.

26.

Node x table

		Cost to		
		x	y	z
From	x	0	5	2
	y	$\infty$	$\infty$	$\infty$
	z	$\infty$	$\infty$	$\infty$

Node y table

		Cost to		
		x	y	z
From	x	$\infty$	$\infty$	$\infty$
	y	5	0	6
	z	$\infty$	$\infty$	$\infty$

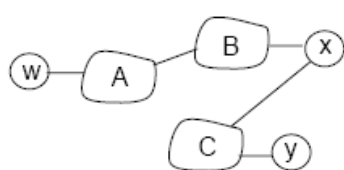
		Cost to		
		x	y	z
From	x	0	5	2
	y	5	0	6
	z	2	6	0

Node z table

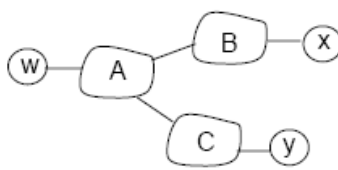
		Cost to		
		x	y	z
From	x	$\infty$	$\infty$	$\infty$
	y	$\infty$	$\infty$	$\infty$
	z	2	6	0

		Cost to		
		x	y	z
From	x	0	5	2
	y	5	0	6
	z	2	6	0

27. 由于在 BGP 中从 AS 到目标点的路径信息是可以得到的, 路径环路的检测就很简单了-如果一个 BGP 对等点收到一个路径中包含它自己 AS 号的路由, 那么使用该路由就会产生环路.
28. C 要使 B 处理所有东岸的 B 到 D 流量的一个方法是, C 只从它的东岸对等点公布到 D 的路由.
- 29.



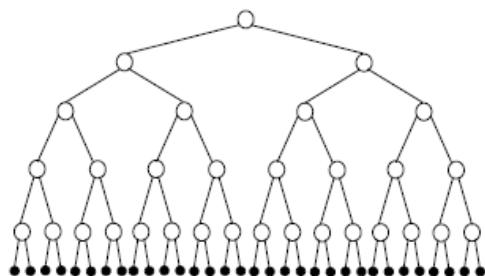
X's view of the topology



W's view of the topology

在上图解答中, 由于 x 收到的到达 y 和到达 w 的路由中都不包含 AC 之间的链接, 因此 x 看不见 AC 之间的连接 (也就是说, x 收到的到达目的节点路由公告中没有同时出现 AS A 和 AS C 的). w 的同理.

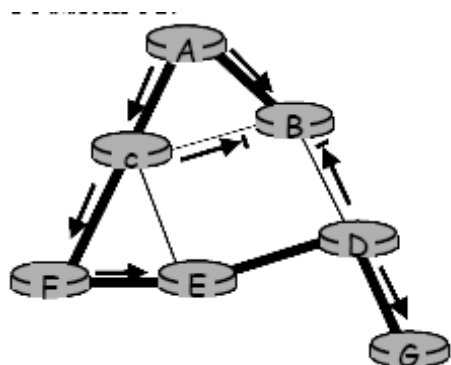
30. 最低费用树为: s 到 t (费用 1), u 到 t (费用 2), v 到 u (费用 1), w 到 v (费用 1), y 到 v (费用 1).  
我们可以对该树为最小费用的原因做如下的非形式化的讨论: u, v, w, 和 y 之间相互连接所需的最小费用为 3, 要将 s 连接到 u, v, w, y 中最小费用为 3 (通过 t).
31. 如图所示:



32 个用户通过路由器二叉树连接到发送方. 采用网络层广播的话, 消息拷贝在每个链路中只传播以一次. 一共有  $62(2+4+8+16+32)$  个链路交叉(所以费用为 64). 采用单播模拟的话, 发送方将信息拷贝发送到任意一个接收方都要经过一条 5 跳的路径. 一共有  $160(5 \times 32)$  个链路交叉(费用 160).

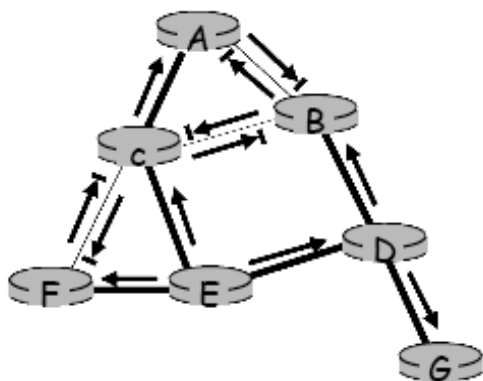
所有接收方连成一线, 发送方在线的一侧, 这种网络拓扑使单播模拟与真正的网络层广播产生的费用相差最大.

32.



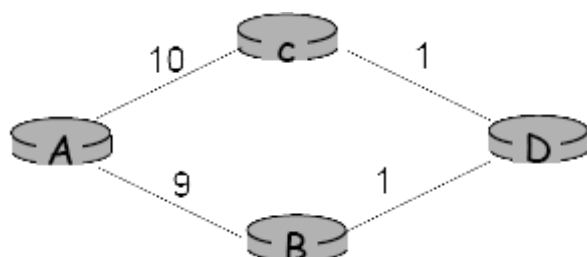
图中粗线指出了从 A 到所有目标节点的最短路径树. 还可能其它的解决方法. 在这一树中, B 没有到达 C 或 D 的路由.

33.



34. 所给网络拓扑的基于中心的树为: 将 A 连到 D, B 连到 C, E 连到 C, F 连到 C(都是直接连接). 这个基于中心的树与最小费用树不相同.

35. 如图:



运用 Dijkstra 算法, 将 A 作为源, 则最低单播费用路径树为连接 AC, AB, 和 BD, 总费用为 20. 而最低费用路径树为: 连接 AB, BD, DC, 费用只有 11.

- 
36. 1 个时间步后 3 个拷贝被发送, 2 个时间步后 6 个, 3 个时间步后 12 个…… $t$  个时间步后  $3 \cdot 2^{t-1}$  拷贝在那个时间步被发送.
37. 协议必须在应用层建立. 例如: 一个应用程序将向所有的其它组员以应用层信息的方式循环广播它的标识.
38. 一个简单的应用层协议将使所有的所有组员知道其它组员的标识, 应用在任何情况下都会发出一个包括对于其它所有节点标识的多播信息. 由于多播信道在多播应用本身的数据同时分发标识信息, 因此这个协议是带内传输的. 用现存的多播分发机制完成信号的带内传输使得这个设计变的十分简单.
39.  $32-4=28$  bits 可以用于多播地址, 所以, 多播地址的数量为:  $N=2$  的 28 次方个. 所以两个多播组地址冲突的几率为:  $1/N=2$  的 -28 次方  $=3.73 \cdot 10^{-9}$   
1000 个多播组的就直接看英语答案吧, 最后那句话的大概的意思是: “忽略 cross-product term (不知道是什么东西……) 的话, 近似结果为: ……”

(终于翻完了, 哈哈, 由于时间仓促, 有错误还请见谅)

## 5 复习题

1. 虽然每条链路都能保证数据包在端到端的传输中不发生差错，但它不能保证 IP 数据包是按照正确的顺序到达最终的目的地。IP 数据包可以使用不同的路由通过网络，到达接收端的顺序会不一致，因此，TCP 需要用来使字节流按正确的序号到达接收端。
2. 链路层能够向网络层提供的服务有：成帧，链路接入，可靠传送，流量控制，纠错，检错，全双工传输等。其中，在 IP 中有的服务是：成帧，检错。  
在 TCP 有的服务是：成帧，可靠传送，流量控制，检错以及全双工传输。
3. 会出现冲突。因为当一个节点在传输数据的同时，又开始接受数据，这种情况下必然会发生冲突。
4. 时隙 ALOHA：1，2 和 4（时隙 ALOHA 只是部分分散，因为它要求所有节点的时钟同步）。令牌传输：1，2，3 和 4。
5. 略
6. 当一个节点传送一个帧时，该节点只有在此帧在整个环网中传播一遍后才释放令牌，这样，如果  $L/R$  比传播延时小，令牌环协议的效率将是很低的。
7.  $2^{48}$  个 MAC addresses;  $2^{32}$  个 IPv4 addresses;  $2^{128}$  个 IPv6 addresses
8. c 的适配器会处理这些帧，但是不会将这些帧中的 IP 数据包传递给 c。  
如果 A 使用的是广播地址，则 c 不仅会处理而且会传递这些数据包。
9. ARP 查询要在广播帧中发送是因为查询主机不知道哪个适配器的地址对应于要查询的 IP 地址。而 ARP 响应时，由于发送节点知道要给哪个适配器发送响应，所以该响应在包含具体目的 MAC 地址的帧中发送，而不必发送广播帧。
10. 不可能。每个 ARP 模块管理该局域网内的适配器，并且每个适配器（MAC）拥有唯一的 LAN 地址。
11. 这三种以太网技术具有相同的帧结构。
12. 每个比特发生一次跳变，由于是全 1 码，因此每两个比特之间也会发生跳变。 $2 \times 10^6$  次，即每秒 2 千万次跳变。
13. 第 5 次冲突后，适配器从 {0, 1, 2..., 31} 中选择 K，故 K 为 4 的概率为  $1/32$ ，它对应于 204.8ms 的时延。

第五章习题：

```

1 0 1 0 0
1 0 1 0 0
1 0 1 0 0
1 0 1 1 1
0 0 0 1 1

```

1. 最右面的一列和最下面的一行是校验比特。
2. 看懂例 5-6 后，可以举出很多类似的例子。

例如，对于正确二维偶检验矩阵：

```

0 0 0 0
1 1 1 1
0 1 0 1
1 0 1 0

```

检测和纠正单比特的例子是：

```

0 0 0 0
1 1 0 1
0 1 0 1
1 0 1 0

```

双比特差错不能检测的例子：

```

0 0 0 0
1 0 0 1
0 1 0 1
1 0 1 0

```

3. 解：计算因特网校验和，我们把 16 比特的值全部加起来：

```

00000000 00000001
00000010 00000011
00000100 00000101
00000110 00000111
00001000 00001001

```

00010100 00011001

他的和是 11101011 11100110

4. 解：如果我们用 10101010000 除以 1001，我们可以得到 10010111，余数是 001，即 R=001。

5. 解：(a)

$$\begin{aligned} E(p) &= Np(1-p)^{N-1} \\ E'(p) &= N(1-p)^{N-1} - Np(N-1)(1-p)^{N-2} \\ &= N(1-p)^{N-2}((1-p) - p(N-1)) \end{aligned}$$

$$E'(p) = 0 \Rightarrow p^* = \frac{1}{N}$$

(b)

$$E(p^*) = N \frac{1}{N} \left(1 - \frac{1}{N}\right)^{N-1} = \left(1 - \frac{1}{N}\right)^{N-1} = \frac{\left(1 - \frac{1}{N}\right)^N}{1 - \frac{1}{N}}$$

$$\lim_{N \rightarrow \infty} \left(1 - \frac{1}{N}\right) = 1 \quad \lim_{N \rightarrow \infty} \left(1 - \frac{1}{N}\right)^N = \frac{1}{e}$$

$$\lim_{N \rightarrow \infty} E(p^*) = \frac{1}{e}$$

6. 解：

$$\begin{aligned} E(p) &= Np(1-p)^{2(N-1)} \\ E'(p) &= N(1-p)^{2(N-2)} - Np2(N-1)(1-p)^{2(N-3)} \\ &= N(1-p)^{2(N-3)}((1-p) - p2(N-1)) \end{aligned}$$

$$E'(p) = 0 \Rightarrow p^* = \frac{1}{2N-1}$$

$$E(p^*) = \frac{N}{2N-1} \left(1 - \frac{1}{2N-1}\right)^{2(N-1)}$$

$$\lim_{N \rightarrow \infty} E(p^*) = \frac{1}{2} \cdot \frac{1}{e} = \frac{1}{2e}$$

7. 解：纯 ALOHA，对于几乎所有的 p 值其效率均为零，如下图：



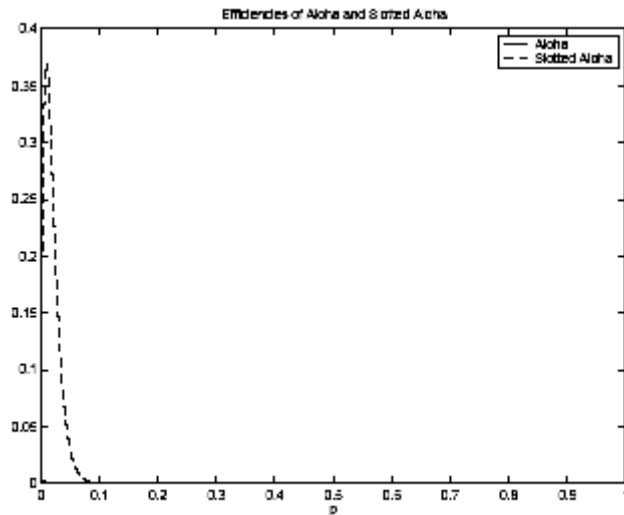


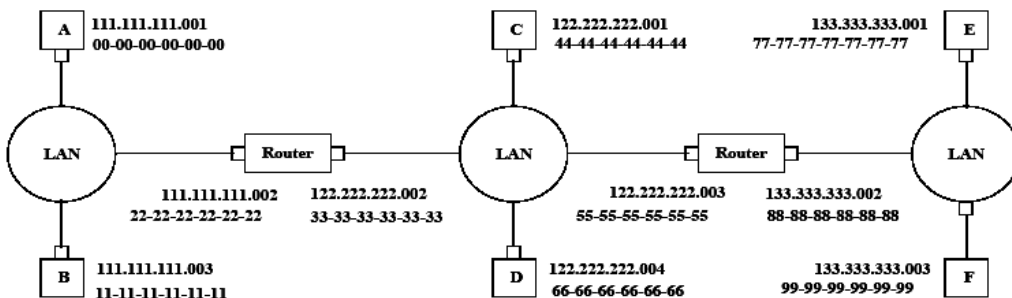
Figure 6: Efficiency of Aloha and Slotted Aloha

8.解：轮询的长度是： $N(Q/R + t_{poll})$ 。

在一个轮询中传输的比特数是  $NQ$ ，最大的吞吐量是：

$$\frac{NQ}{N(Q/R + t_{poll})} = \frac{R}{1 + \frac{t_{poll}R}{Q}}$$

9.解：(a)，(b)，(c) 如下图：



- d) 1.在 A 表格中确定数据，可以路由到节点 111.111.111.002
- 2.主机 A 用 ARP 来确定 LAN 的地址是 111.111.111.002，也就是，22-22-22-22-22-22.
- 3.A 中的适配器和以太网包的以太网的目的地址是：22-22-22-22-22-22.
- 4.第一个路由器接收到分组并解包，该路由器的转发表指示数据包发到 IP 为 122.222.003 的主机。
- 5.然后第一个路由器使用 ARP 来获取相关的以太网地址，为 55.55.55.55.55.55。
- 6.继续以上过程直到分组到达主机 F。
- e) A 的 ARP 必须知道 IP 为 111.111.111.002 的主机的局域网地址。主机 A 发送在一个广播帧里发送 ARP 请求，第一个路由器收到请求包，并给主机 A 发送一个 ARP 响应包。该 ARP 响应包由一个目的地址为 00.00.00.00.00.00 的以太网帧来承载。

10. 等待的时间为 51200 个比特的时间，对于 10Mbps 的以太网来说，等待的时间是：

$$\frac{51.2 \times 10^3 \text{ bits}}{10 \times 10^6 \text{ bps}} = 5.12 \text{ msec}$$

而对于 100Mbps，则为 512μs。

11.  $t = 0$  时，A 开始传输数据， $t = 576$ ，A 完成传输。在最坏的情况下，B 在  $t = 224$  时开始发送数据， $t = 224 + 225 = 449$ ，B 的第一个比特到达 A。因为  $449 < 576$ ，故 A 在完成传输前会中止。

12.

Time, $t$	Event
0	A and B begin transmission
225	A and B detect collision
273	A and B finish transmitting jam signal
$273+225=498$	B's last bit arrives at A; A detects an idle channel
$498+96=594$	A starts transmitting
$273+512=785$	B returns to Step2 B must sense idle channel for 96 bit times before it transmits
$594+225=819$	A's transmission reaches B

因为在 B 安排重传时间前，A 的重传信号就已经到达了 B，所以在 A 重传数据的时候 B 暂停传输。这样 A 和 B 就不会冲突。

13. 要使  $1/(1+5a)=0.5$ ，等价于  $a=0.2=t_{\text{prop}}/t_{\text{trans}}$ ， $t_{\text{prop}}=d/(1.8*10^8)\text{m/sec}$ ，以及  $t_{\text{trans}}=(576\text{bits})/(10^8\text{bits/sec})$ ，解出  $d$  等于 265 米。对于 100Mbps 以太网标准，两台主机最短距离是 200m。

因为 A 在传输数据时要检测是否有其他的主机也在传输数据，所以  $t_{\text{trans}}$  必须大于  $2t_{\text{prop}}=2.265\text{m}/1.8*10^8\text{m/s}=2.94\mu\text{s}$ 。因为  $2.94<5.76$ ，故 A 在完成传输前将检测到 B 的信号。

14. a. 设  $y$  为一个随机变量，表示成功传输需要的时隙：

$$P(Y = m) = \beta(1 - \beta)^{m-1},$$

其中  $\beta$  是成功的概率。

这是个几何分布，均值为  $1/\beta$ 。连续浪费的时隙是  $x=y-1$ ，则

$$x = E[X] = E[Y] - 1 = \frac{1 - \beta}{\beta}$$

$$\beta = Np(1 - p)^{N-1}$$

$$x = \frac{1 - Np(1 - p)^{N-1}}{Np(1 - p)^{N-1}}$$

$$\text{efficiency} = \frac{k}{k + x} = \frac{k}{k + \frac{1 - Np(1 - p)^{N-1}}{Np(1 - p)^{N-1}}}$$

b. 最大效率等于  $x$  的最小值，也等于  $\beta$  的最大值。由题意可知， $\beta$  的最大值是  $p=1/N$ 。

$$\text{efficiency} = \frac{k}{k + \frac{1 - (1 - \frac{1}{N})^{N-1}}{(1 - \frac{1}{N})^{N-1}}}$$

$$\lim_{N \rightarrow \infty} \text{efficiency} = \frac{k}{k + \frac{1 - 1/e}{1/e}} = \frac{k}{k + e - 1}$$

c.

d. 很明显，当 k 趋近于无穷大时， $k/(k+e+1)$  接近 1。

15. A.

$$\begin{aligned} & \frac{900m}{2 \cdot 10^8 m/sec} + 4 \cdot \frac{20bits}{10 \times 10^6 bps} \\ &= (4.5 \times 10^{-6} + 8 \times 10^{-6}) sec \\ &= 12.5 \mu sec \end{aligned}$$

B.

- T=0, A 和 B 都传输。
- T=12.5μs, A 检测到冲突。
- T=25μs, B 的最后一个比特终止传输。
- T=37.5μs, A 重传第一个比特到 b

$$t = 37.5 \mu sec + \frac{1000bits}{10 \times 10^6 bps} = 137.5 \mu sec$$

- A 的包完全到达 B。

C.

$$12.5 \mu sec + 5 \cdot 100 \mu sec = 512.5 \mu sec$$

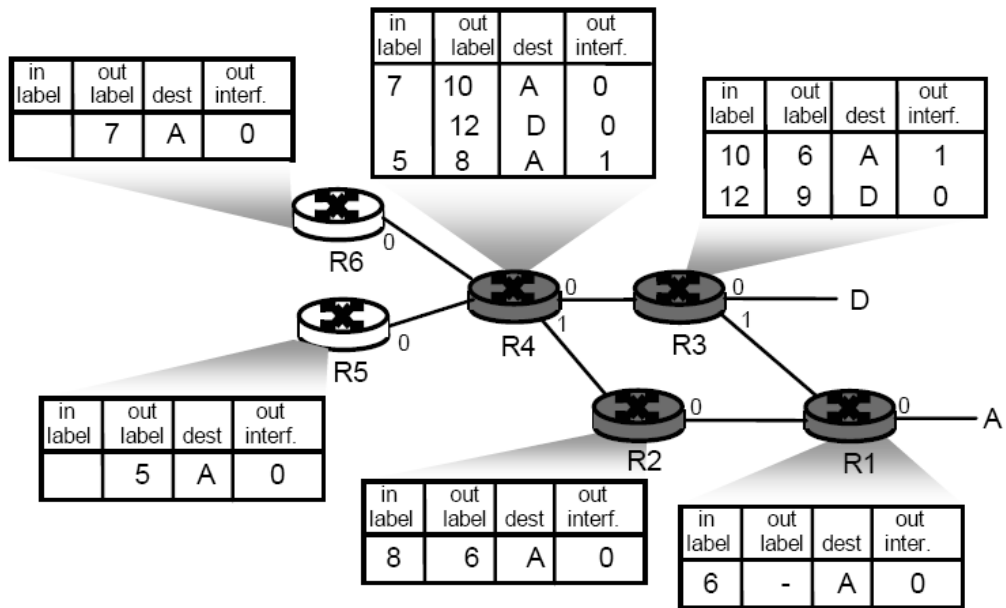
16. 填充  $8 \cdot L$  个比特需要的时间:  $\frac{L \cdot 8}{64 \times 10^3} sec = \frac{L}{8} msec.$

B.  $L=1500$  时, 包延时为:  $\frac{1500}{8} msec = 187.5 msec.$ ,  $L=48$  时, 包延时为:  $\frac{48}{8} msec = 6 msec.$

C. 存储转发延时:  $\frac{L \cdot 8 + 40}{R}$ ,  $L=1500$  时, 延时为:  $\frac{1500 \cdot 8 + 40}{155 \times 10^6} sec \approx \frac{12}{155} msec \approx 77 \mu sec$ ,

当  $L=48$  时, 延时小于  $1 \mu s$ 。

D. 对于典型的 ATM 链路速度, 存储转发延时很小。但是当  $L=1500$  时, 包延时对于实时语音应用而言非常大。



17.

## 6 复习题

1. APs 周期性的发送信标帧，AP 的一个信标帧通过 11 个信道中的一个发送。信标帧允许附近的无线基站发现和识别 AP。
2. 1) 基于无线主机的 MAC 地址；2) 用户名和密码的结合。在这 2 中情况中，AP 把信息传送给认证服务器。
3. 不对
4. 2 个原因：1) 无线信道中误码率比较高；2) 在有线的以太网中，发送站点能够检测到是否有碰撞发生，然而在 802.11 中站点不能检测到碰撞。
5. 不对
6. 每一个无线基站都可以设置一个 RTS 阈值，因此只有当将要传送的数据帧长度长于这个阈值时，RTS/CTS 序列才会被用到。
7. 没有好处。假设有 2 个站点同时想发送数据，并且他们都使用 RTS/CTS。如果 RTS/CTS 的帧长和数据帧长一样时，信道就会被浪费，因为发送 RTS/CTS 的时间和发送数据的时间一样。因此 RTS/CTS 交换只有当 RTS/CTS 帧长远小于数据帧长时才有用。
8. 开始时，交换机在其转发表中有一个表项标识了无线站点和前一个 AP 的联系。当无线基站和新的 AP 联系时，新的 AP 将创建一个包括无线基站 MAC 地址以及以太网广播帧的帧。当交换机收到该帧时，更新其转发表，使得无线站点可以通过新的 AP 到达。
9. UMTS 源于 GSM，CDMA200 源于 IS-95。

## 习题

1. 输出  $d_1 = [-1, 1, -1, 1, -1, 1, -1, 1]$ ;  $d_0 = [1, -1, 1, -1, 1, -1, 1, -1]$
2. 发送方 2 的输出 =  $[1, -1, 1, 1, 1, -1, 1, 1]$ ;  $[1, -1, 1, 1, 1, -1, 1, 1]$
- 3.

$$d_1^1 = \frac{1 \times 1 + (-1) \times (-1) + 1 \times 1 + 1 \times 1 + 1 \times 1 + (-1) \times (-1) + 1 \times 1 + 1 \times 1}{8} = 1$$

$$d_1^2 = \frac{1 \times 1 + (-1) \times (-1) + 1 \times 1 + 1 \times 1 + 1 \times 1 + (-1) \times (-1) + 1 \times 1 + 1 \times 1}{8} = 1$$

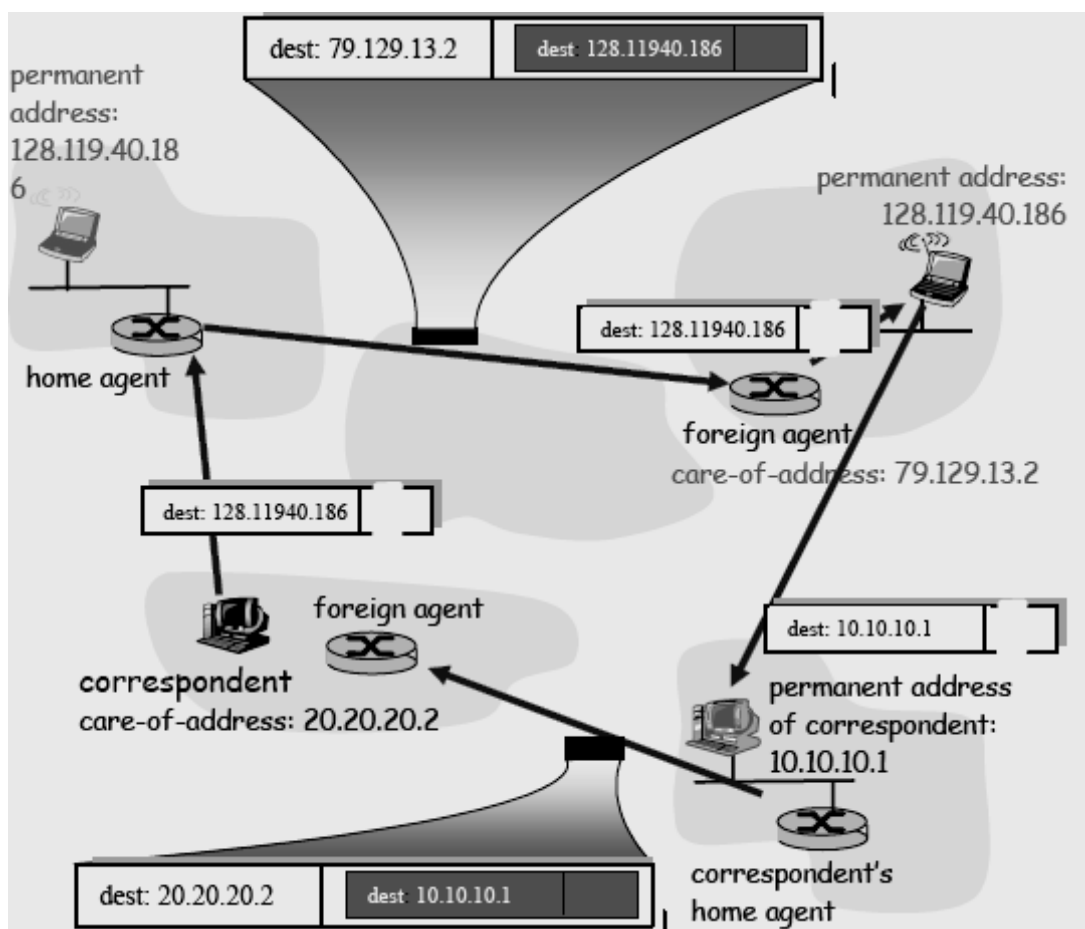
4. a) 两个 AP 有不同的 SSID 和 MAC 地址。一个到达咖啡馆的无线站点将会和其中一个 AP 的联系。发生联系后，在新的站点和 AP 之间会建立一条虚链路。把两个 ISP 的 AP 标识为 AP1 和 AP2。假设新的站点和 AP1 相关联。当它发送一个帧的时候，它将会到达 AP1。虽然 AP2 也会收到这个帧，但是它不会处理这个帧，因为这个帧发送给它的。因此这两个 ISP 能在相同的信道上平行地工作。尽管如此，这两个 ISP 将共享相同的无线带宽。如果不同的 ISP 中的无线站点同时发送数据，将会产生碰撞。对 802.11b 来说，两个 ISP 的最大合计传输速率为 Mbps。
- b) 现在如果不同的 ISP 中的 2 个无线终端同时发送数据，就不会产生碰撞。因此这 2 个 ISP 的最大合计传输速率为 22 Mbps 对 802.11b 来说。
5. 这样设计是为了公平。我们假设开始只有 H1 这一个无线站点发送数据，当时当 H1 发送到一半时，另一个站点 H2 也要发送一个帧，为了简单起见，我们还是假设没有隐藏的终端。在发送之前，H2 检测到信道忙，因此它要选择一个随机的回退值。现在我们假设 H1 发送完第一个帧以后，如果退回到第一步，即它等待一个 DIFS 然后发送第二个帧。那么 H2 仍将被阻塞并再次等待信道的空闲时刻。因此，如果 H1 有 1000 个帧要传的话，那么 H2 只有等 H1 传完这 1000 个帧后才能有机会接入这个信道。但是如果 H1 传完第一个帧后退回到第二步，那么它也将选取一个随机的回退值，这样的话就给了 H2 发送数据的机会。
6. 不含有数据的帧长为 32 字节。假设传输速率是 11 Mbps，传输一个控制帧的时间为  $32 \times 8 / 11 \text{ Mbps} = 23 \mu\text{sec}$ 。传输数据帧所需的时间为  $(8256 \text{ bits}) / (11 \text{ Mbps}) = 751 \mu\text{sec}$ 。总时间为：  
 $\text{DIFS} + \text{RTS} + \text{SIFS} + \text{CTS} + \text{SIFS} + \text{FRAME} + \text{SIFS} + \text{ACK} = \text{DIFS} + 3\text{SIFS} + (3 \times 23 + 751) \mu\text{sec} = \text{DIFS} + 3\text{SIFS} + 820 \mu\text{sec}$ 。
7. a) 不会。因为在距离矢量算法中，目的点的改变信息只会在相邻的节点间传输（这不同于链路状态路由，在这个算法中，信息的改变将会通过广播发送到所有的路由器，因此在链路状态广播后，所有的路由器将会知道网

络中的变化)。

b) 在距离矢量算法中,不同的路由器对移动节点访问的网络有不同的认识。路由器不会知道被访问网络的变化,除非信息通过发生在到移动节点的路由器间的一对路由信息的交换传给它。

c) 时间范围和网络的直径(源到目的地的最长距离)的概念相似。因为路由信息只会通过路径上的相邻路由器信息的成对的交换传播。因此在最坏的情况下,从网络中一点到另一点传播信息所需要的时间决定与网络的直径。

8. 如果通信者是移动的,那么任何到达它的数据报都会经过它的归属代理。网络中的正在被访问的外部代理也和这个过程有关,因为外部代理向通信者的归属代理通报了通信者的位置。通信者的归属代理接收到的数据报将会被封装并在通信者的归属代理与外部代理之间传送。



9. 因为数据报必须先传到归属代理,然后从归属代理再传到移动站点,这样的话时延就会比直接路由时的时延大。注意到从通信者到移动站点的直接时延实际上可能比从通信者到归属代理再到移动站点这个过程的总时延要小。它还决定于不同路径段的时延。注意到间接路由也会引入一个归属代理处理时延。
- 10.
11. 可以有相同的转交地址。如果转交地址就是外部代理的地址的话,这个地址就是相同的。一旦外部代理拆封了隧道数据报(tunneled datagram)并确定了移动节点的地址,那么分离的地址就要单独传送数据报到它们在这个被访问网络中不同的目的地。
12. 如果 MSRN 被提供给 HLR,那么,一旦 MSRN 发生改变,它的值在 HLR 中必须更新。在 HLR 中提供 MSRN 的好处是它的值能够被迅速地提供,而不用去询问 VLR。

## 7 复习题

1. 流式存储音频/视频：暂停/恢复，重新定位，快进，实时交互的音频视频：人们进行实时的通信和响应。
2. 第一阵营：TCP/IP 协议中的基本原理没有变化，按照需求增加带宽，仍然使用有超高速缓存，内容分布，多点覆盖的网络。  
第二阵营：提供一个可以使应用在网络中节省带宽的网络服务。  
第三阵营：有区别的服务：提出了在网络边缘的简单分类和维护方案。并且根据他们在路由队列中的级别给出了不同的数据包和级别。
3. 6.1：简单，不需要 meta file 和流媒体服务器  
6.2：允许媒体播放器通过 web 服务器直接进行交互，不需要流媒体服务器。  
6.3：媒体播放器直接与流媒体服务器进行交互，以满足一些特殊的流媒体应用。
4. 端到端时延是分组从源经过网络到目的地所需要的时间。  
分组时延抖动是这个分组与下个分组的端到端时延的波动。
5. 在预定的播放时间之后收到的分组不能被播放，所以，从应用的观点来看，这个分组可以认为是丢失了。
6. 第一种方案：在每 n 个数据块之后发送一个冗余的数据块，这个冗余的被。。。【the redundant chunk is obtained by exclusive OR-ing the n original chunks】  
第二种方案：随起始的数据流发送一个低分辨率，低比特率的方案，这种交错不会增加数据流的带宽需求。
7. 不同会话中的 RTP 流：不同的多播地址  
同一会话中不同流：SSRC field  
RTP 和 RTCP 分组通过端口号区别。
8. 传达报告分组：包括分组丢失部分的信息，最后序列号，两次到达时间间隔的抖动。发送报告分组：大部分目前产生的 RTP 分组的时间戳和 wall clock time，发送分组的数量，发送字节的数量，源描述分组：发送方的 e-mail 地址，发送方的姓名，产生 RTP 流的应用。
9. 在非抢占式优先级排队中，一旦开始分组的发送，它就不会被打断。在抢占式优先级排队中，当一个比目前分组有着更高优先级的分组到达时，即使目前的分组没有传送完毕，也会被打断，来传送更高优先级的分组。这意味着分组的一部分将在网络中分成分离的块来传送，而这些块并不都有着合适的头字段，由于这个原因，并没有采用抢占式优先级排队。
10. 一个非保持工作的调度规则是时分复用的。为什么一个循环帧被分割进 slot，而这些 slot 对于一些特定的级别是独享的。
11. 可量测性：每流资源预留是指用路由器处理资源预留和保存经过这个路由器的每一个流的状态。  
灵活的服务：为少量的预先给定的服务级别提供的 Intserv framework

## 8 复习题

1. 机密性是仅有发送方和预定的接收者能够理解传输的报文内容。窃听者可以截取到报文，但是报文在一定程度上进行了加密，就使得截获者不能解密所截获的报文。报文的完整性是接收方无论原始报文加密与否都可以检测到所接受的报文是否在传输中发生改变。一个加密的报文在传输的过程中被改变它仍然是被加密的（窃听者仍然不能够获取原始的明文）但是如果这个错误未被检测到则报文信息完整性将不存在。同样，一个报文在传输中被改变将使得明文没有加密特性。
2. 被动入侵只是监听（嗅探、截获）报文。主动入侵不仅监听报文，还主动的发送报文到网络中。
3. 最重要的区别就是在对称密钥系统中发送者和接收者都要知道同样的密钥。在公钥系统中，加密和解密密钥截然不同的。加密密钥是公开的，但是解密密钥只有接收者知道。
4. 在这种情况下，可以发动已知明文攻击。如果入侵者通过某种手段得到了发送者加密的报文，则可以发动选择明文攻击。
5. 如果每个用户想要和其他的  $N$  个用户通信的话，那么每两个用户必须要有一个公用的对称密钥。因为有  $N*(N-1)/2$  对用户，所以就需要  $N*(N-1)/2$  把密钥。使用公钥密码系统需要  $2N$  把密钥。
6. 确定用户是否处于“活动”状态，用于对付回放攻击。
7. 所谓在生存周期只使用一次是指一旦一个协议使用了一个不重复数，就永远不会再使用这个数字。生存周期指的是不重数的生存周期。
8. 中间人攻击的攻击者将自己插到发送者和接收者之间，改变发送者和接收者之间传输的数据（如：再加密）。中间人攻击是及其有害的，因为（如图 8-13 所示）发送者和接受者会认为他们之间是进行的机密传输，而安心的发送和接收文件。使用对称密钥是会有中间人攻击。
9. 假设 Bob 给 Alice 发送了一个加密的文档。可鉴别：Alice 能够确认是 Bob 发送加密文档。不可伪造：Alice 能够确认只有 Bob 可以发送这个加密文档（即：其他人不可能得到密钥，然后加密发送文档）。不可否认：Alice 能够确认除了 Bob 没有人能够发送这个文档。为了说明更进一步的差别，假设 Bob 和 Alice 共有一个全世界只有他们两个人知道的密钥。如果 Alice 收到一个使用这一密钥加密的文档，而且确信不是她自己对该文档进行的加密，那么这个文档就是可鉴别的，不可伪造的（假设使用了一个足够强大的加密系统）。然而 Alice 不能保证 Bob 一定发送了这个文档，因为实际上她自己也知道这个密钥，可能这个文档是她自己加密后发送的。
10. 报文摘要在很多方面类似于检查和。报文摘要算法对一个任意长的报文  $m$  进行处理后，计算生成一个固定长度的数据“指纹”，称之为报文摘要  $H(m)$ 。报文摘要对于数据的保护体现在：如果  $m$  改变为  $m'$ ，则由原始数据计算而得的  $m$  的摘要  $H(m)$  不会和数据已改变的  $m'$  的摘要  $H(m')$  相同。
11. 在用私钥对一个短的报文摘要而不是对整个报文进行加密时，公钥加密报文摘要更好。因为使用像 RSA 这样的加密技术是很昂贵的，对少部分数据进行加密比对大部分数据进行加密是更可取的。
12. 与被加密的报文摘要相关的报文没有必要加密。对报文进行的是机密性的加密，而对报文摘要进行的是整体性的加密，这两个（加密的）目的不同。
13. 一个密钥分发中心被用来为通信双方创建并分配一对对称的密钥，要求只有通信双方才有他们自己的对称的密钥，利用密钥，它们可以将加密/解密后的通信信息发送/从...接收（到）密钥分发中心。每个公钥都被分别绑定了权力认证的信息，CA（密钥分发中心）用它自己的（CA 中心的）私钥对这些公钥进行签名。因此，得到某 CA 的公钥后，实体可以从中提取出 CA 签名的公钥，对 CA 签名进行检验后，该实体就可以拥有 CA 验证公钥。
14. ESP 提供鉴权、完整性和加密性，AH 提供鉴权和完整性。