TCP : Transmission Control Protocol  
 reliable, in-order byte-stream data transfer

loss: acknowledgements and retransmissions

 flow control :sender won't overwhelm receiver

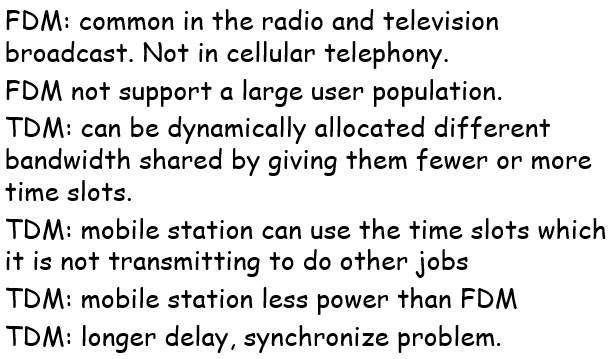
congestion control: senders " slow down sending rate ” when network congested.

UDP- User Datagram Protocol

connectionless ,unreliable data transfer, no flow control no congestion control

App’s using TCP: HTTP(Web), FTP (file transfer), Telnet (remote login), SMTP (email)

App’s using UDP: streamingmedia, teleconferencing, DNS, Internet telephony



packet-switching:data sent thru net in discrete “chunks”

**滑动窗口**：

若从滑动窗口的观点来统一看待1比特滑动窗口、后退n及选择重传三种协议，它们的差别仅在于各自窗口尺寸的大小不同而已。1比特[滑动窗口协议](https://baike.baidu.com/item/%E6%BB%91%E5%8A%A8%E7%AA%97%E5%8F%A3%E5%8D%8F%E8%AE%AE)：发送窗口=1，接收窗口=1；后退n协议：发送窗口>1，接收窗口=1；选择重传协议：发送窗口>1，接收窗口>1。

**1比特滑动窗口协议**

当[发送窗口](https://baike.baidu.com/item/%E5%8F%91%E9%80%81%E7%AA%97%E5%8F%A3)和接收窗口的大小固定为1时，[滑动窗口协议](https://baike.baidu.com/item/%E6%BB%91%E5%8A%A8%E7%AA%97%E5%8F%A3%E5%8D%8F%E8%AE%AE)退化为[停等协议](https://baike.baidu.com/item/%E5%81%9C%E7%AD%89%E5%8D%8F%E8%AE%AE)（stop-and-wait）。该协议规定发送方每发送一帧后就要停下来，等待接收方已正确接收的确认（acknowledgement）返回后才能继续发送下一帧。由于接收方需要判断接收到的帧是新发的帧还是重新发送的帧，因此发送方要为每一个帧加一个序号。由于停等协议规定只有一帧完全发送成功后才能发送新的帧，因而只用一比特来编号就够了。

**后退n协议**

由于[停等协议](https://baike.baidu.com/item/%E5%81%9C%E7%AD%89%E5%8D%8F%E8%AE%AE)要为每一个帧进行确认后才继续发送下一帧，大大降低了信道利用率，因此又提出了后退n协议。后退n协议中，发送方在发完一个[数据帧](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%B8%A7)后，不停下来等待应答帧，而是连续发送若干个数据帧，即使在连续发送过程中收到了接收方发来的应答帧，也可以继续发送。且发送方在每发送完一个数据帧时都要设置超时定时器。只要在所设置的超时时间内仍未收到确认帧，就要重发相应的数据帧。如：当发送方发送了N个帧后，若发现该N帧的前一个帧在计时器超时后仍未返回其确认信息，则该帧被判为出错或丢失，此时发送方就不得不重新发送出错帧及其后的N帧。

从这里不难看出，后退n协议一方面因连续发送[数据帧](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%B8%A7)而提高了效率，但另一方面，在重传时又必须把原来已正确传送过的数据帧进行重传（仅因这些数据帧之前有一个数据帧出了错），这种做法又使传送效率降低。由此可见，若传输信道的传输质量很差因而[误码率](https://baike.baidu.com/item/%E8%AF%AF%E7%A0%81%E7%8E%87)较大时，连续测协议不一定优于[停止等待协议](https://baike.baidu.com/item/%E5%81%9C%E6%AD%A2%E7%AD%89%E5%BE%85%E5%8D%8F%E8%AE%AE)。此协议中的[发送窗口](https://baike.baidu.com/item/%E5%8F%91%E9%80%81%E7%AA%97%E5%8F%A3)的大小为k，接收窗口仍是1。MAX\_SEQ=2^n-1, n bits sequences number.

**选择重传协议**

在后退n协议中，接收方若发现错误帧就不再接收后续的帧，即使是正确到达的帧，这显然是一种浪费。另一种效率更高的策略是当接收方发现某帧出错后，其后继续送来的正确的帧虽然不能立即递交给接收方的高层，但接收方仍可收下来，存放在一个缓冲区中，同时要求发送方重新传送出错的那一帧。一旦收到重新传来的帧后，就可以原已存于缓冲区中的其余帧一并按正确的顺序递交高层。这种方法称为选择重发（SELECTICE REPEAT），其工作过程如图所示。显然，选择重发减少了浪费，但要求接收方有足够大的缓冲区空间。k=.size of sending window Ws, size of receiving window Wr, Ws+Wr<=2^n.

纯ALOHA协议(Pure ALOHA). 当传输点有数据需要传送的时候，它会向立即向通讯频道传送。接收点在收到数据后，会ACK传输点。如果接收的数据有错误，接收点会向传输点发送NACK。当网络上的两个传输点同时向频道传输数据的时候，会发生冲突，这种情况下，两个点都停止一段时间后，再次尝试传送。

分段（或时隙）ALOHA协议(Slotted ALOHA) 这是对纯ALOHA协议的一个改进，思想是用时钟来统一用户的数据发送。改进之处在于，**它把频道在时间上分段，每个传输点只能在一个分段的开始处进行传送。用户每次必须等到下一个时间片才能开始发送数据，每次传送的数据必须少于或者等于一个频道的一个时间分段。**这样很大的减少了传输频道的冲突。从而避免了用户发送数据的随意性，减少了数据产生冲突的可能性，提高了信道的利用率。

如果假设传输点对频道的使用是符合泊松分布的话，我们可以得以下公式：

纯ALOHA协议：

https://gss1.bdstatic.com/9vo3dSag_xI4khGkpoWK1HF6hhy/baike/s%3D71/sign=14573d2a3a01213fcb334cdd55e72dba/1ad5ad6eddc451da188c0518b5fd5266d11632ce.jpg

分段ALOHA协议：

https://gss0.bdstatic.com/94o3dSag_xI4khGkpoWK1HF6hhy/baike/s%3D65/sign=106619eed72a283447a6350e5ab5f7b6/18d8bc3eb13533faf2b49addabd3fd1f40345bc6.jpg

S是通过量，G是提供的流量(每单位时间通过的数据包数量)。

从这个公式我们可以看出，对纯ALOHA来说，当G = 0.5 的时候，S 达到最大值 18.4%。 只有当的时候，纯ALOHA系统才处于稳定状态。对分段ALOHA来说， 当G = 1 的时候，S 达到最大值 36%。只有当的时候，分段ALOHA系统才处于稳定状态。

**Pure ALOHA**: 想发就发，碰撞了结束当前传输等待一段随机时间。多用户共享通道，导致碰撞。帧传输在一个完全任意的时间。

PureAloha吞吐量：N是在传输一完整帧时间内提交的平均数量的新帧。服从破松分布。

G是在提交一完整帧时间内的平均数量的新旧帧，也服从破松分布。

P0为不经历碰撞的概率，吞吐量S=G\*P0

P0=e^(-2G), S = G\*e^(-2G)



**Slotted ALOHA**:时间被分段，包必须在一个区间内传输。data are always sent at the beginning of a time slot . If a collision occurs, finish the current transmission and retry after a random amount of time slots. Slotted ALOHA cuts the vulnerable period for packets from 2t to t. This doubles the best possible throughput from 18.4% to 36.8%

The Slotted ALOHA algorithm :

1. If a host has a packet to transmit, it waits until the beginning of the next slot before sending

2. Listen to the broadcast and check if the packet was destroyed

3. If there was a collision, wait a random number of slots and try to send again

Contention period

In Pure ALOHA：two packet transmission times. In Slotted ALOHA: one packet transmission time. In CSMA, up to 2\*end-to-end delay.

性能：CSMA > Slotted ALOHA > Pure ALOHA

**1-persistent CSMA**: 遇忙持续监听，当信道空闲立即传输。如果碰撞发生，等待一段随机时间再重新开始。**Non-persistent CSMA:** 遇忙等待一个随机时间然后再重新监听，遇闲立即传输。如果碰撞发生，等待一个随机时间再重新开始。**P-Persistent CSMA**：遇闲，以概率p传输包，如果不传输就等待一个slot再重新监听，如果忙也等待一个slot再重新监听。如果碰撞发生，等待一个随机时间再重新开始。对于p坚持，p=0.5的话，如果发现信道空闲，以50%的概率发送包，如果不发送等待一个slot再重新尝试监听。尝试次数的平均数量为：

**CSMA/CD**:带冲突监测的载波监听多路访问。如果两个站同时传输数据，会发生碰撞。停止传输。空闲就立即传输，忙的话等待直到空闲。监测到碰撞就立即中止传输。等待一个随机时间后再重新开始。载波监听减少碰撞次数，碰撞监测减少碰撞影响，使得信道能够尽早使用。How long does it take to realize there has been a collision?

Worst case: 2 x end-to-end prop. delay

**RTT**(Round-Trip Time): 往返时延。在计算机网络中它是一个重要的性能指标，表示从发送端发送数据开始，到发送端收到来自接收端的确认（接收端收到数据后便立即发送确认），总共经历的时延。

一般认为单向时延=传输时延t1+传播时延t2+排队时延t3

t1是数据从进入节点到传输媒体所需要时间，通常等于数据块长度/信道带宽

t2是信号在信道中需要传播一定距离而花费的时间，等于信道长度/传播速率

t3可笼统归纳为随机噪声，由途径的每一跳设备及收发两端负荷情况及吞吐排队情况决定

讨论数据链路层的最小帧长问题时，发送时延大于RTT才能检测到冲突.

最小帧长=RTT\*传输速率。

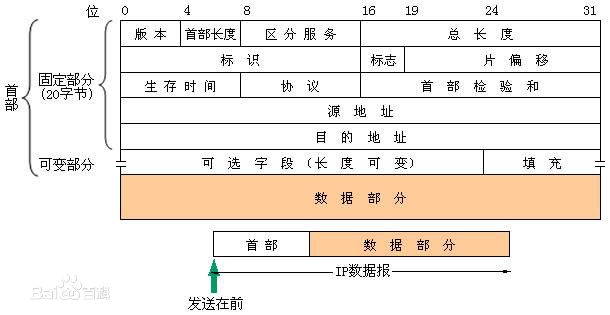
**forwarding**: move packets from router’s input to appropriate router output

**routing**: determine route taken by packets from source to dest.

forwarding: process of getting through single interchange

routing: process of planning trip from source to dest

**IP**: Internet Protocol 数据报格式



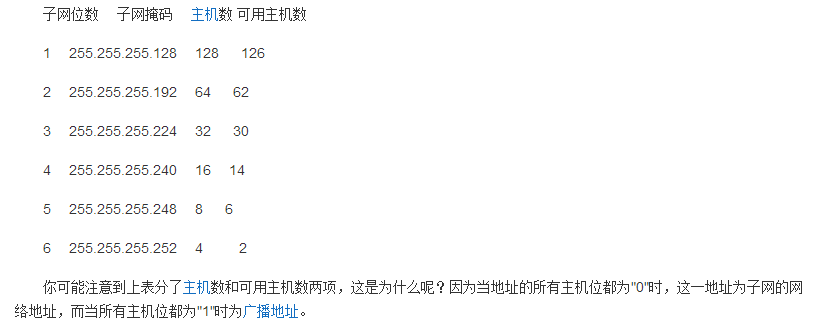
首部的前一部分是固定长度，共20字节，是所有IP数据报必须具有的。在首部的固定部分的后面是一些可选字段，其长度是可变的。首部中的源地址和目的地址都是IP协议地址。**首部长度**　占4位，可表示的最大十进制数值是15。请注意，这个字段所表示数的单位是32位字长（1个32位字长是4字节），因此，当IP的首部长度为1111时（即十进制的15），首部长度就达到60字节。当IP分组的首部长度不是4字节的整数倍时，必须利用最后的填充字段加以填充。因此数据部分永远在4字节的整数倍开始，这样在实现IP协议时较为方便。首部长度限制为60字节的缺点是有时可能不够用。但这样做是希望用户尽量减少开销。最常用的首部长度就是20字节（即首部长度为0101），这时不使用任何选项。**总长度**　总长度指首部和数据之和的长度，单位为字节。总长度字段为16位，因此数据报的最大长度为2^16-1=65535字节。在IP层下面的每一种数据链路层都有自己的帧格式，其中包括帧格式中的数据字段的最大长度，这称为最大传送单元MTU(Maximum Transfer Unit)。当一个数据报封装成链路层的帧时，此数据报的总长度（即首部加上数据部分）一定不能超过下面的数据链路层的MTU值。**标志(flag)** MF=1即表示后面“还有分片”的数据报。MF=0表示这已是若干数据报片中的最后一个。**片偏移**：较长的分组在分片后，某片在原分组中的相对位置。也就是说，相对用户数据字段的起点，该片从何处开始。片偏移以8个字节为偏移单位。这就是说，除了最后一个分片，每个分片的长度一定是8字节（64位）的整数倍。

IP地址是一个32位的二进制数，通常被分割为4个“8位二进制数”（也就是4个字节）。

一个**A类IP地址**由1字节的网络地址和3字节主机地址组成，网络地址的**最高位必须是“0”**， 地址范围从1.0.0.0 到126.0.0.0。可用的A类网络有126个，每个网络能容纳1亿多个主机。 一个**B类IP地址**由2个字节的网络地址和2个字节的主机地址组成，网络地址的**最高位必须是“10”**，地址范围从128.0.0.0到191.255.255.255。可用的B类网络有16382个，每个网络能容纳6万多个主机 。 一个**C类IP地址**由3字节的网络地址和1字节的主机地址组成，网络地址的**最高位必须是“110”**。范围从192.0.0.0到223.255.255.255。C类网络可达209万余个，每个网络能容纳254个主机。 IP地址由两部分组成，即网络号（Network ID）和主机号（Host ID）。网络号标识的是Internet上的一个子网，而主机号标识的是子网中的某台主机。网络地址分解成两个域后，带来了一个重要的优点：IP数据包从网际上的一个网络到达另一个网络时，选择路径可以基于网络而不是主机。只有在一个网络号下的计算机之间才能“直接”互通，不同网络号的计算机要通过[网关](https://baike.baidu.com/item/%E7%BD%91%E5%85%B3)（Gateway）才能互通。

IP Fragmentation and Reassembly

network links have **MTU (max.transfer size)** - largest possible link-level frame. large IP datagram divided (“fragmented”) within net. one datagram becomes several datagrams. “reassembled” only at **final destination.** IP header bits used to identify, order related fragments.

子网。IP网络还允许划分成更小的网络，称为子网（Subnet），这样就产生了子网掩码。**子网掩码的作用就是用来判断任意两个IP地址是否属于同一子网络**，这时只有在同一子网的计算机才能"直接"互通。要将一个网络划分为多个子网，因此网络号将要占用原来的主机位。那计算机是怎样才知道这一网络是否划分了子网呢？这就可以从子网掩码中看出。子网掩码和IP地址一样有32bit，确定子网掩码的方法是其与IP地址中标识网络号的所有对应位都用"1"，而与主机号对应的位都是"0"。

A Link-State Routing Algorithm

Dijkstra’s algorithm:

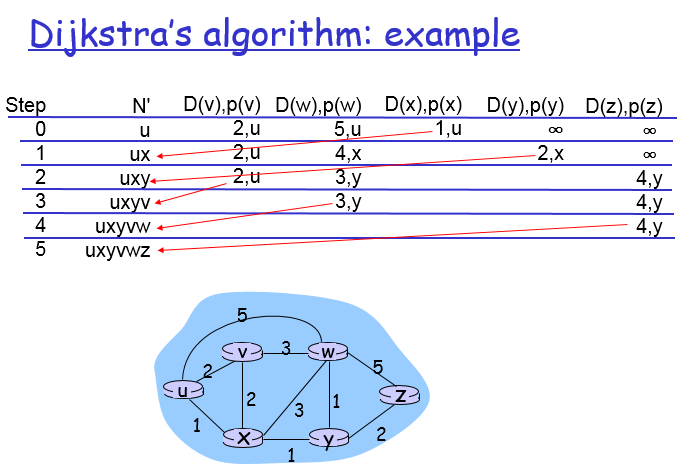
C (x,y) : 从结点x到y的连接开销，如果不可达为无穷大。

D(v): 目前的开销从源到目标点v。

P(v): 从源到目标点v的前驱结点。

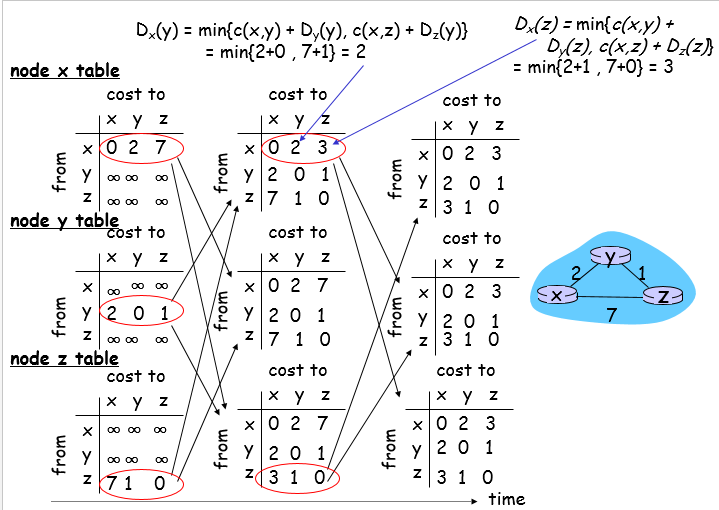
N’：最低代价已知的点集合。

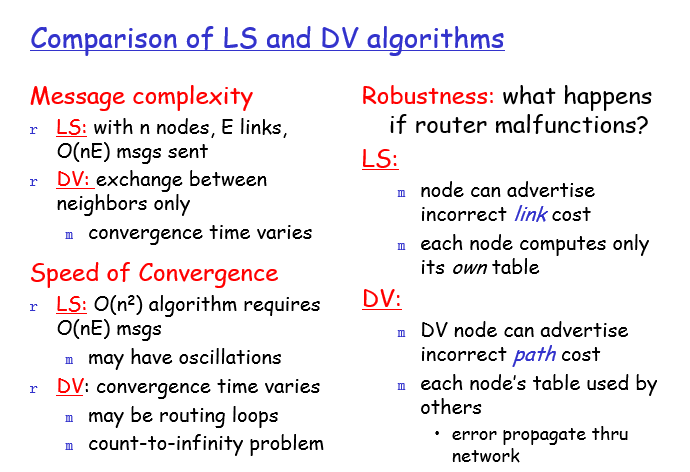
以起始点为中心向外层层扩展，直到扩展到终点为止。



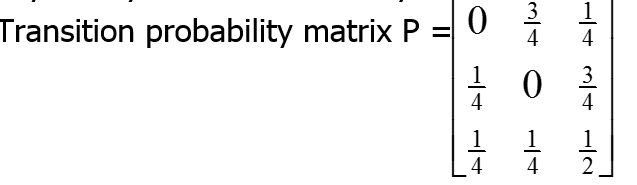
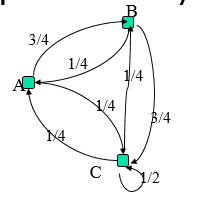
Distance Vector Algorithm 距离向量算法

* Dx(y) = estimate of least cost from x to y
* Node x knows cost to each neighbor v: c(x,v)
* Node x maintains distance vector **D**x = [Dx(y): y є N ]
* Node x also maintains its neighbors’ distance vectors
* For each neighbor v, x maintains : **D**v = [Dv(y): y є N ]

基本思想：每个结点周期性的发送自己的距离向量给自己的邻居，当一个结点x接收到邻居的新的距离向量时，更新自己的距离向量通过：**Dx(y) ← minv{c(x,v) + Dv(y)} for each node y ∊ N 。**如果到任意终点的向量值改变了的话，就通知邻居。



**马尔科夫过程**



We further define the probability vector

We can solve the equation and get the solution:  

**排队论**

λ**平均达到率** u**平均服务率**

**服务强度** p = λ/u Pn = (1-p)\*p^n (当有k个包在队列，n=k+1)

mean number of packets L=p/(1-p) mean number in the queue Lq=p^2/(1-p)

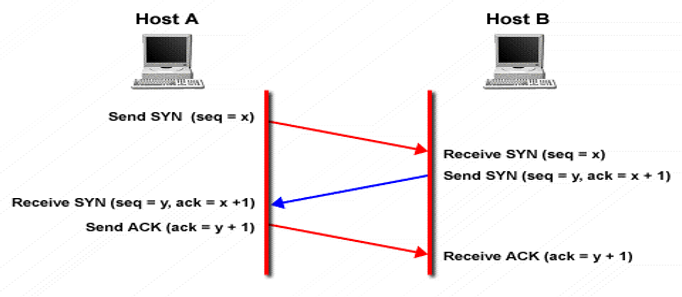
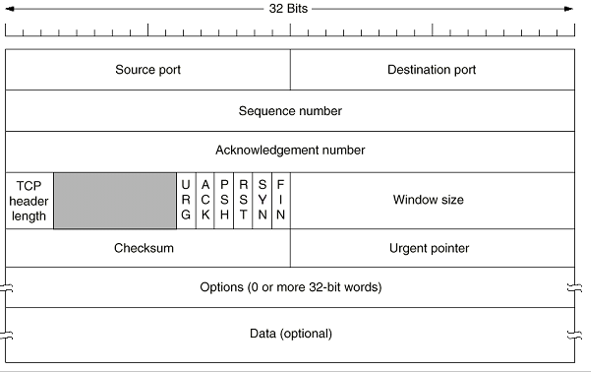
mean waiting time W=1/(u\*(1-p)) mean waiting time in the queue Wq=p/(u\*(1-p))

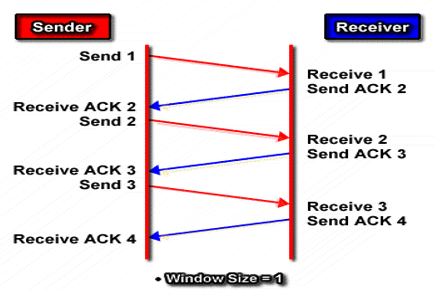
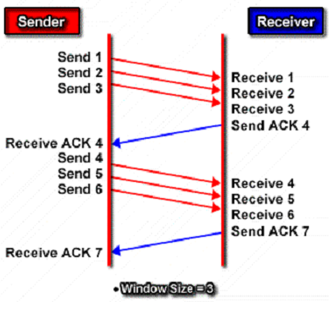
Elements of Transport Protocols: Addressing, Connection Establishment, Connection Release, Flow Control and Buffering, Multiplexing, Crash Recovery

**IP is a Layer 3 protocol and TCP is a Layer 4 protocol. TCP is connection-oriented and ensures reliability but IP is connection-less with best effort attempts at delivery.**

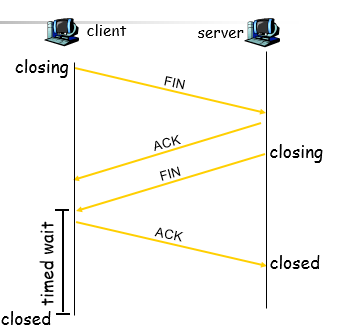
TCP: connection-oriented, reliable, divides outgoing message into segments, reassembles message at the destination station, re-sends anything not received.

UDP: connectionless, unreliable, transmit messages, provides no software checking for segment delivery(unreliable), does not reassemble incoming messages, use no acknowledgments, provides no flow control.



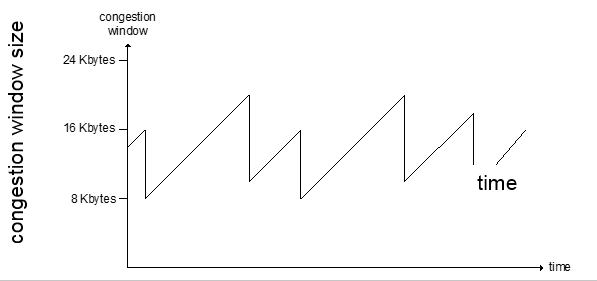
 

窗口大小指在接收到ACK之前能够传输的信息数量。

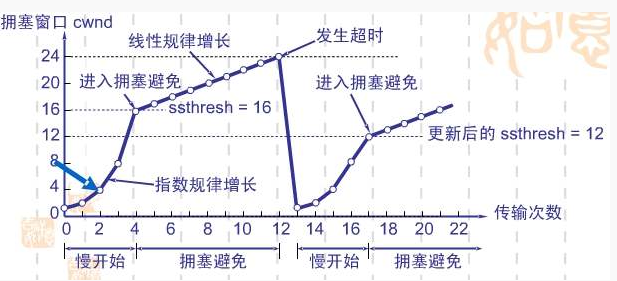


**Flow Control:** Sender won’t overflow receiver’s buffer by transmitting too much, too fast. **Congestion** **Control:** too many sources sending too much data too fast for network to handle. Manifestations: lost packets (buffer overflow at routers) long delays (queueing in router buffers).

**TCP congestion control**: additive increase (increase CongWin by 1 MSS every RTT until loss detected), multiplicative decrease (cut CongWin in half after loss). (Avoid Congestion)



**TCP Slow Start:**

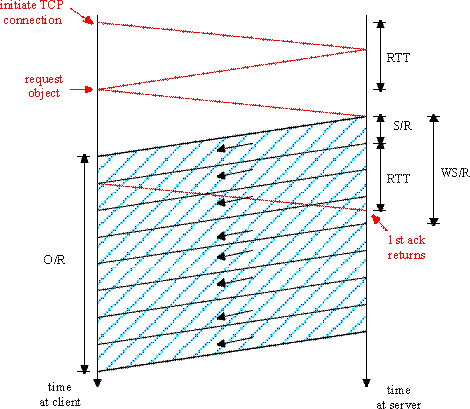
**控制过程**：   
  -[1]. TCP连接初始化，将拥塞窗口cwind设置为1个报文段，即cwind=1；   
  -[2]. 执行慢开始算法，cwind按指数规律增长，直到cwind == ssthresh时，开始执行拥塞避免算法，cwind按线性规律增长；   
  -[3]. 当网络发生拥塞，把ssthresh值更新为拥塞前ssthresh值的一半，cwind重新设置为1，再按照 [2] 执行。When a triple duplicate ACK occurs, Threshold set to CongWin/2 and CongWin set to threshold.

TCP throughput: When window size is W, throughput=W(window size)/RTT, after loss, window drops to W/2, throughput to W/(2\*RTT).So, the average throughput=0.75\*W/RTT.

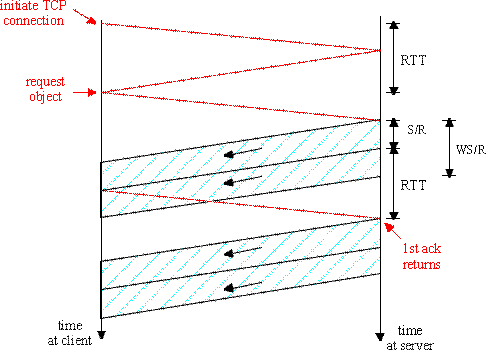
Ignore congestion, delay is influenced by TCP connection establishment, data transmission delay and slow start.

Fixed congestion window:

1.WS/R > RTT+S/R.W(window size),S:MSS(bits) R(one link between client and server of rate). ACK for first segment in window returns before window’s worth of data sent. Delay=2\*RTT+O/R. O(object size)bits



2.W\*S/R<RTT+S/R.wait for ACK after sending window’s worth of data sent. K=O/(WS) . delay = 2RTT + O/R +(k-1)\*[S/R+RTT-WS/R]



3.with slow start





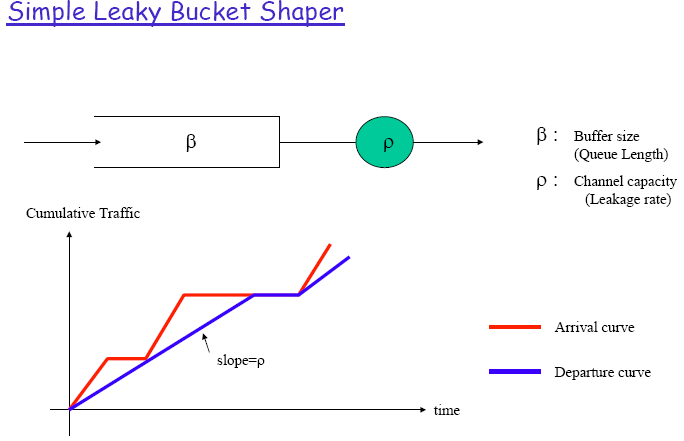


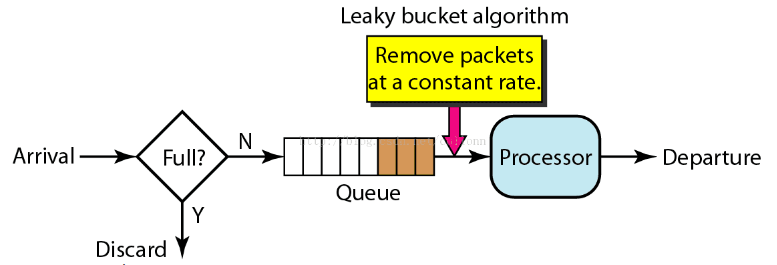


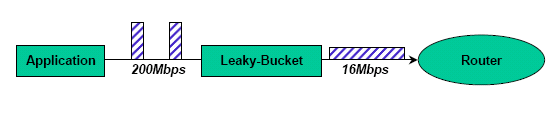


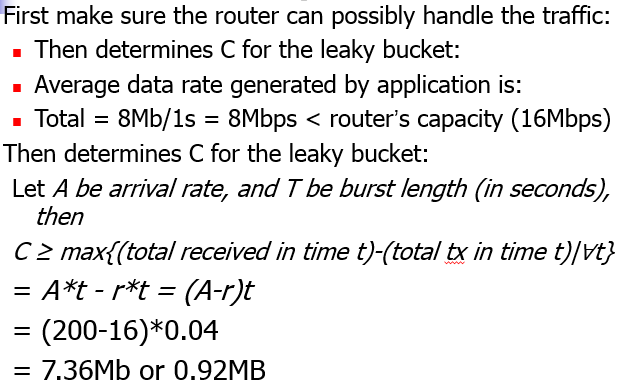
漏桶算法（Leaky Bucket）

 到达的数据包（网络层的PDU）被放置在底部具有漏孔的桶中（数据包缓存）；漏桶最多可以排队b个字节，漏桶的这个尺寸受限于有效的系统内存。如果数据包到达的时候漏桶已经满了，那么数据包应被丢弃；数据包从漏桶中漏出，以常量速率（r字节/秒）注入网络，因此平滑了突发流量。

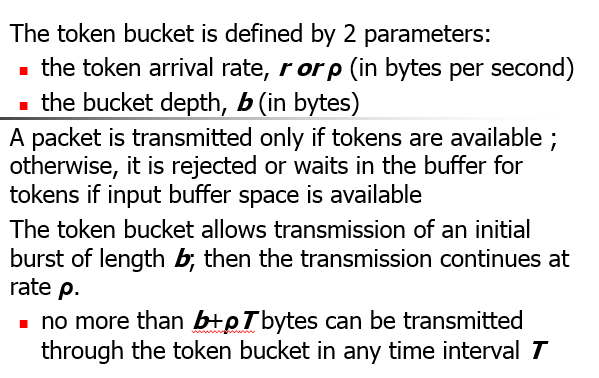


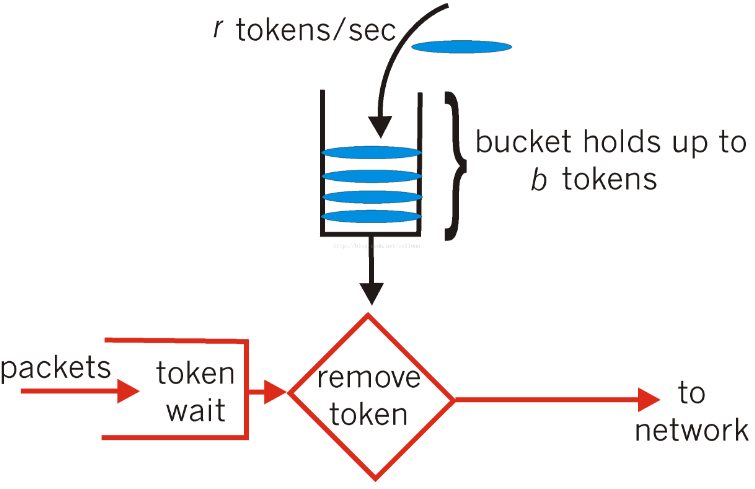


Sender max. transmission rate of 200Mbps Router capacity is 16MbpsApplication generates data in burst of 8Mb, with a burst length of 40ms, once every second. 



令牌桶算法：露桶算法当buffer不为空时，输出率是常数。令牌桶有两个桶：一个是数据缓冲，一个是令牌缓冲（令牌以一个常数率生成）。

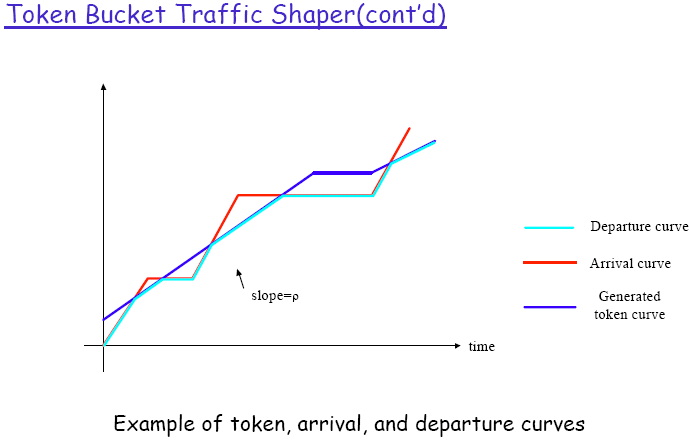




令牌桶算法的基本过程如下:1.每秒会有 r 个令牌放入桶中，或者说，每过 1/r 秒桶中增加一个令牌2.桶中最多存放 b 个令牌，如果桶满了，新放入的令牌会被丢弃3.当一个 n 字节的数据包到达时，消耗 n 个令牌，然后发送该数据包4.如果桶中可用令牌小于 n，则该数据包将被缓存或丢弃

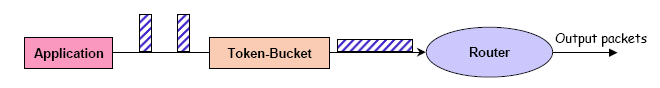
“漏桶算法”能够强行限制数据的传输速率，而“令牌桶算法”在能够限制数据的平均传输数据外，还允许某种程度的突发传输。在“令牌桶算法”中，只要令牌桶中存在令牌，那么就允许突发地传输数据直到达到用户配置的上限，因此它适合于具有突发特性的流量。

Let: *S be burst length; C be token capacity;* ρ be token arrival rate; *M be max output rate.* *Since: (a) Max size of output burst = C + ρS bytes, and (b) but it must also equals MS bytes.* *Hence: C + ρS = MS or S=C/(M−ρ)*

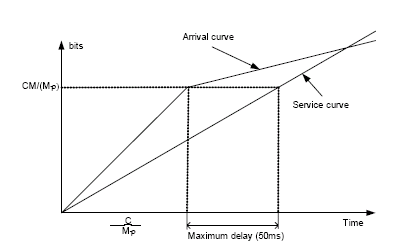


Example:

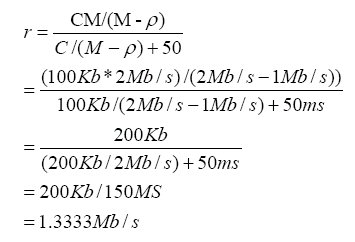
Consider the arrival traffic characterized by a token bucket with parameters ρ (average rate) = 1 Mbps, M (maximum rate) = 2 Mbps, and C (token depth) = 100Kb. What is the minimum rate r that needs to be allocated by a router in order to guarantee a delay no larger than 50 ms?



We consider the worst case where the maximum burst length is given by **S=C/(M-ρ)**. Then the maximum accumulative amount of arrival traffic to the router is **MS=MC/(M-ρ)**. We can draw the arrival and service curve as follows



The minimum rate needs to be allocated by the router in order to guarantee a delay no larger than 50 ms is just the slope of the service curve. From the above figure, we can see the slope is given by:



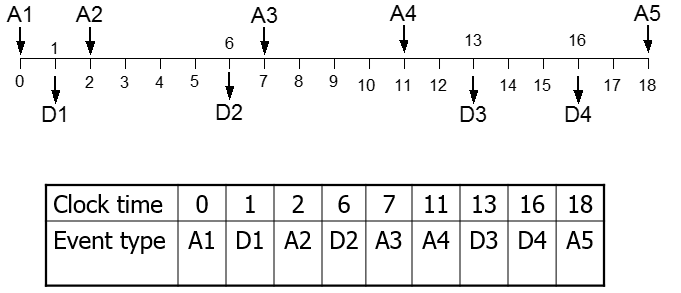
Single-server queuing system

System state uniquely determined by status of server and the number of customers waiting.

System state remains constant unless there is an arrival or departure.

Build model to advance simulation clock at arrival and departure times

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Inter-arrival time | 2 | 5 | 4 | 7 |
| Service time | 1 | 4 | 6 | 3 |



Random Number Generator

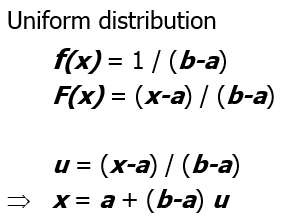
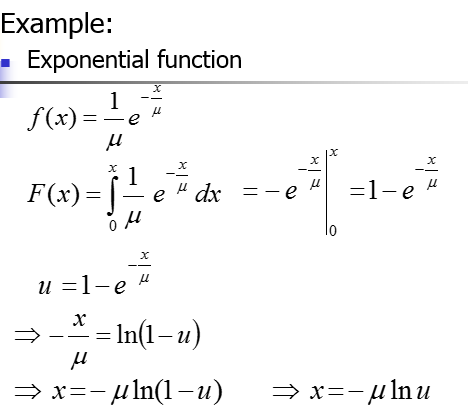
Basic Requirements: Sequence of numbers generated are independent from each other; Numbers are uniformly distributed between 0 and 1.

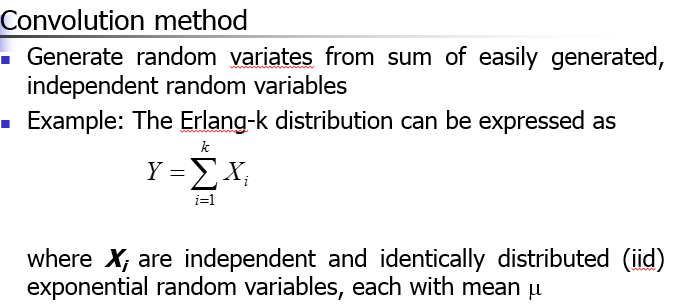
***Yi*** = (***a Yi-1*** + ***C***) mod ***M Y0*** is the ***seed. After Yi is generated, it can be normalized over the interval (0, 1) by Ui = Yi / M.***

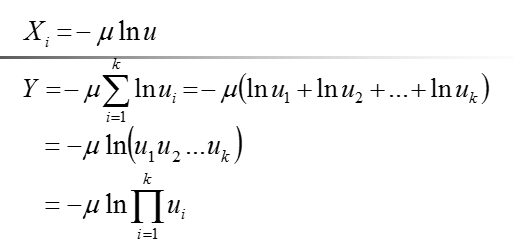
Basic Method:

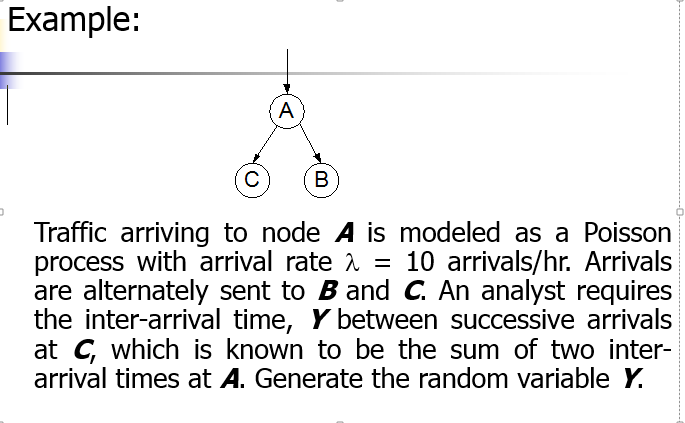
a) Generate ***u***

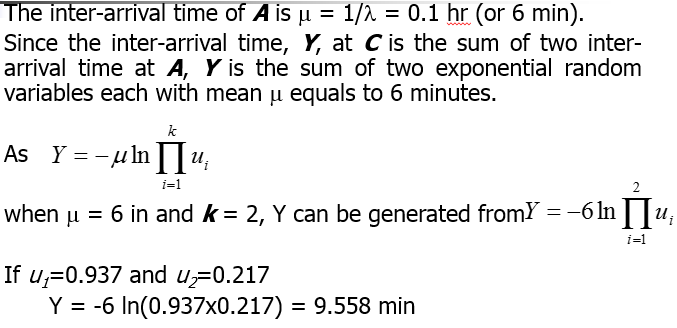
b) Select variety ***x*** such that ***F(x) = u*** i.e. ***x*** = ***F-1(u)***

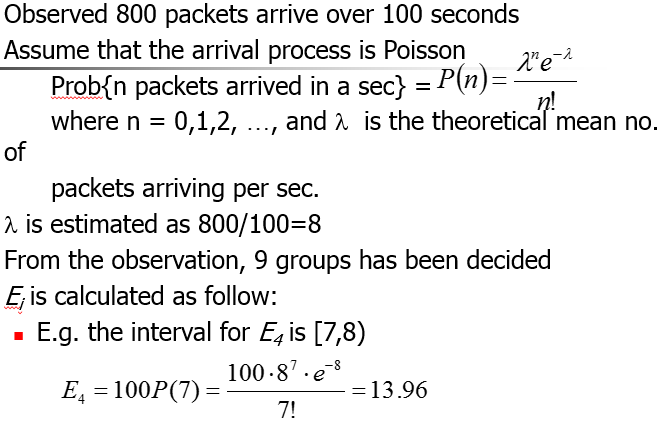


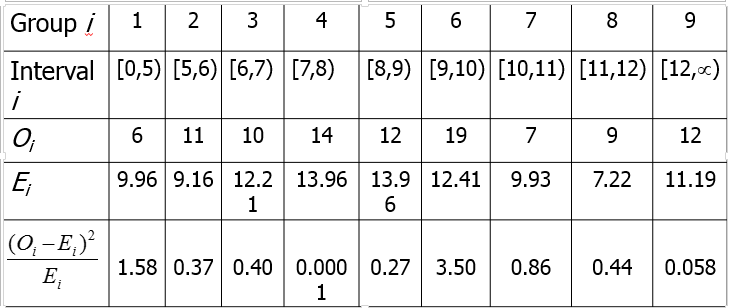


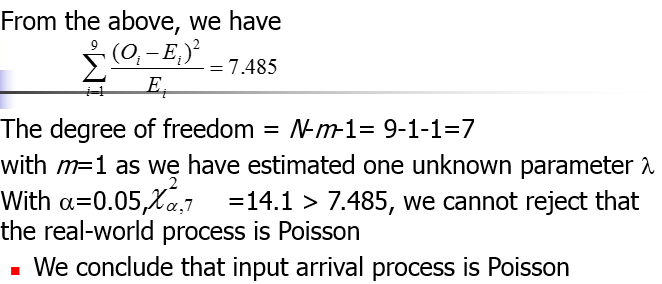




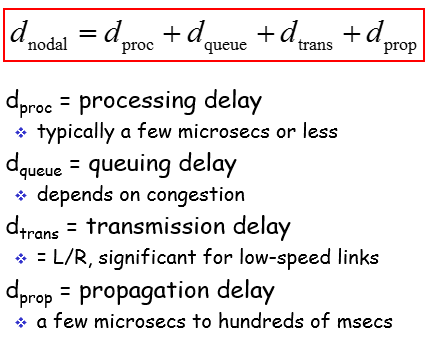


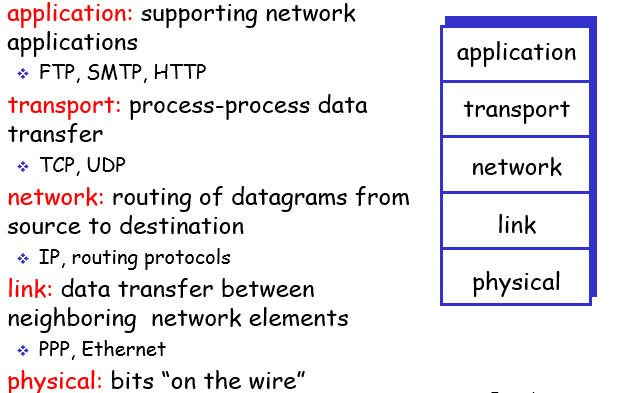






**Circuit Switching**: dedicated circuit per call: telephone net. End-end resources reserved for “call”. Link bandwidth, switch capacity, dedicated resources: no sharing. Guaranteed performance. Call setup required. Network resource divided into “pieces”. Resource piece idle if not used by owning call.

**Packet Switching**: data sent thru net in discrete “chunks”. End-end data stream divided into packets. Each packet uses full link bandwidth. Resource used as needed. Store and forward: packets move one hop at a time. Great for bursty data, resource sharing, simpler, no call setup. Excessive congestion: packet loss and loss. 



5层：application、transport、network、data link、physical

Functions of Data Link Layer: **Provide service interface to the network layer, dealing with transmission errors, regulating data flow. Framing, Error Control, Flow control.**

Data Link Layer: Logical Link Control Sublayer (**LLC**), Medium Access Control Sublayer (**MAC**).

**Network Layer**: transport segment from sending to receiving host; on sending side encapsulates segments into datagrams; on receiving side, delivers segments to transport layer;

Elements of **Transport Layer**: Addressing, Connection Establishment, Connection Release, Flow Control and Buffering, Multiplexing, Crash Recovery.

封装成帧(framing): 封装成帧（framing）就是在一段数据的前后分别添加首部和尾部，这样就构成了一个帧。

错误校验(error detection): **Parity Checking**（奇偶校验）single bit parity, detect one bit error; two dimensional bit parity, detect and correct single bit error. Cyclic Redundancy Check