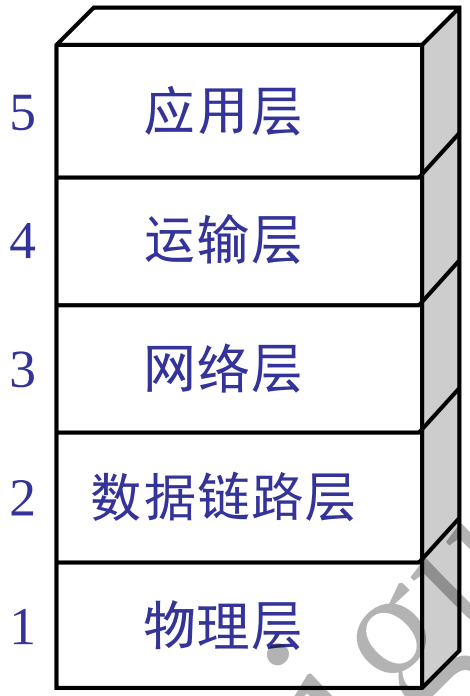
计算机网络

顾 军

计算机学院

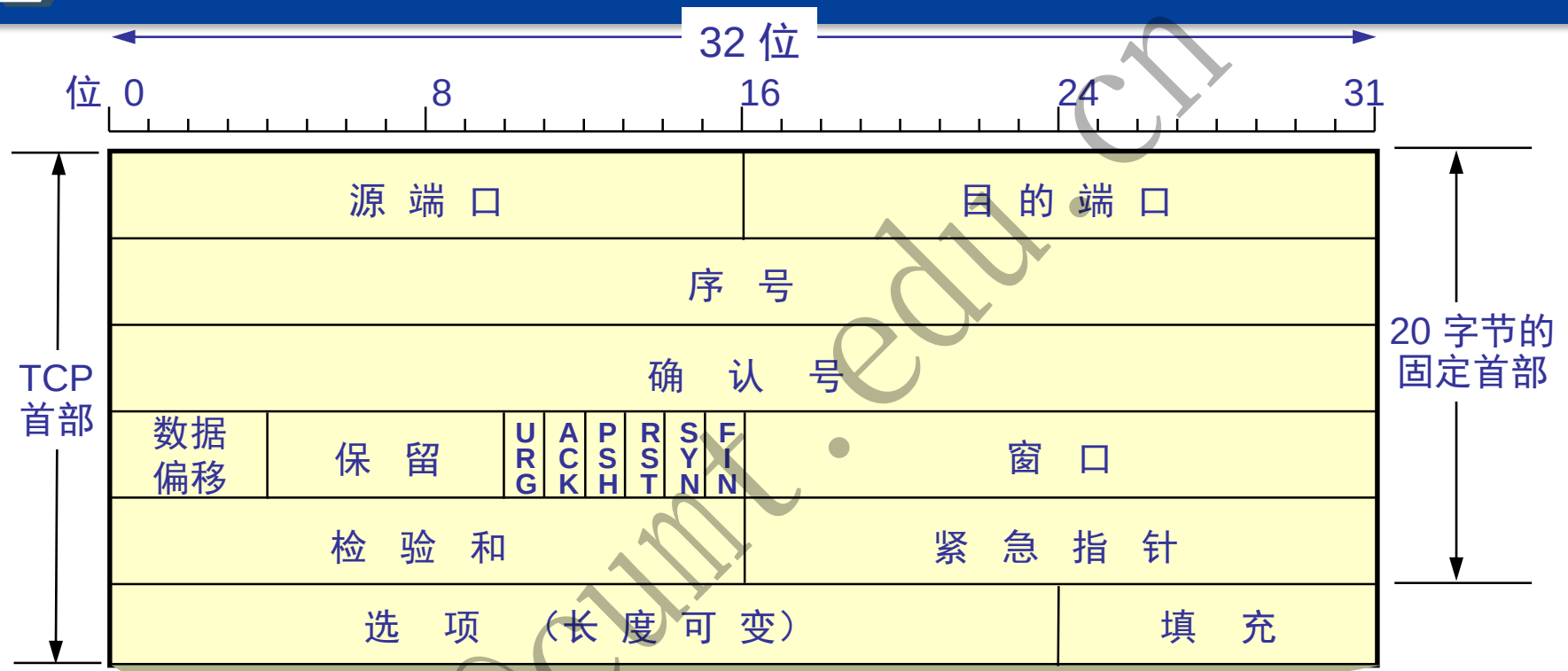
jgu@cumt.edu.cn

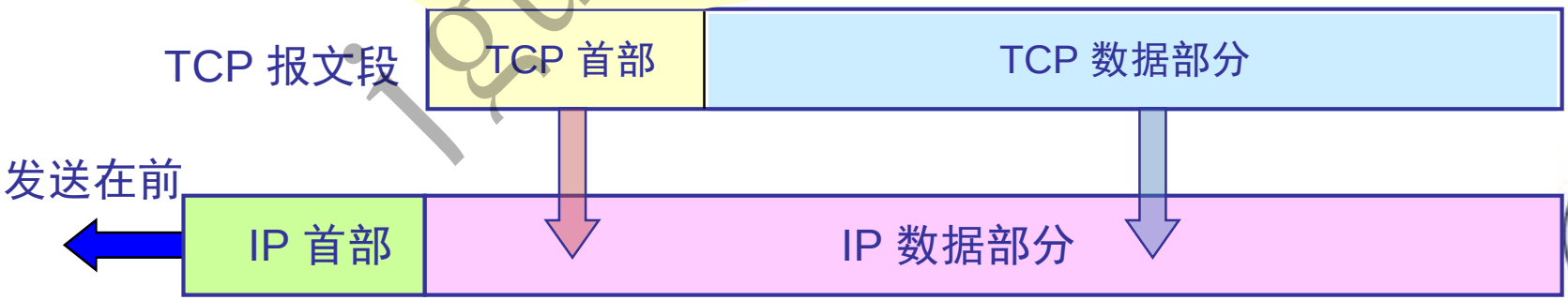
专题5：如何保证端到端的可靠传输

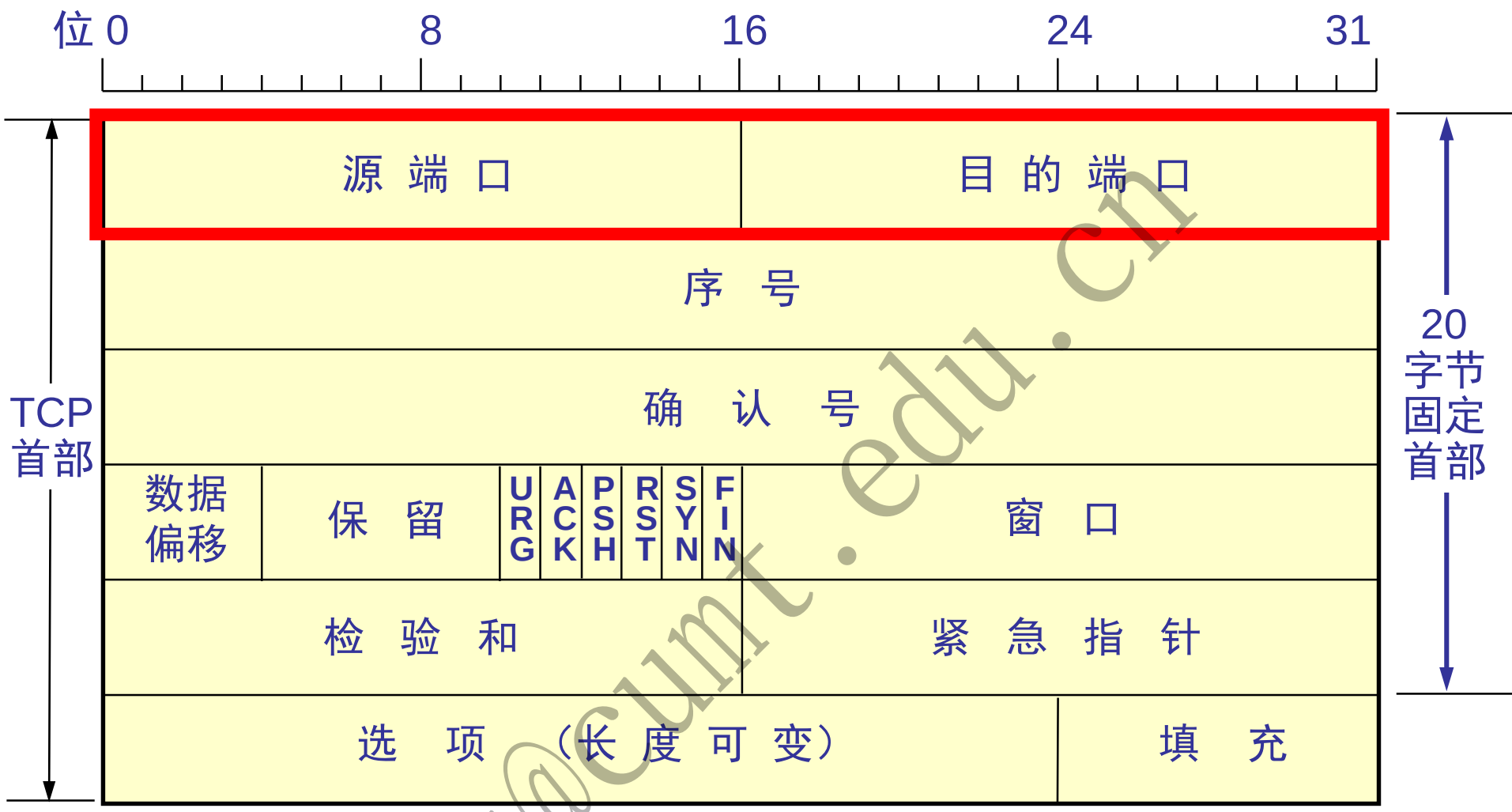


- 应用层(application layer)
- 运输层(transport layer)
- 网络层(network layer)
- 数据链路层(data link layer)
- 物理层(physical layer)

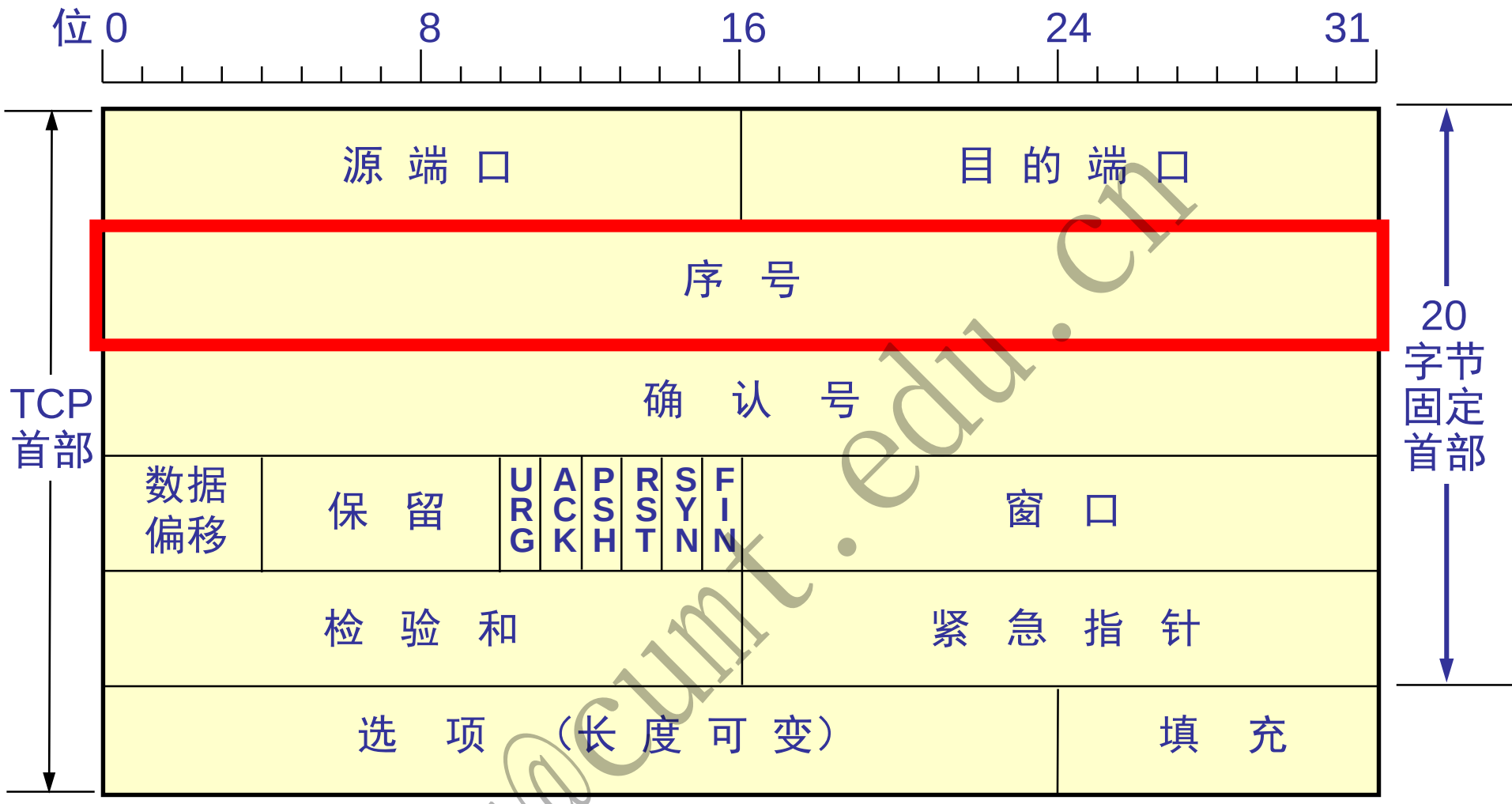
Q15：TCP 报文段如何设计？





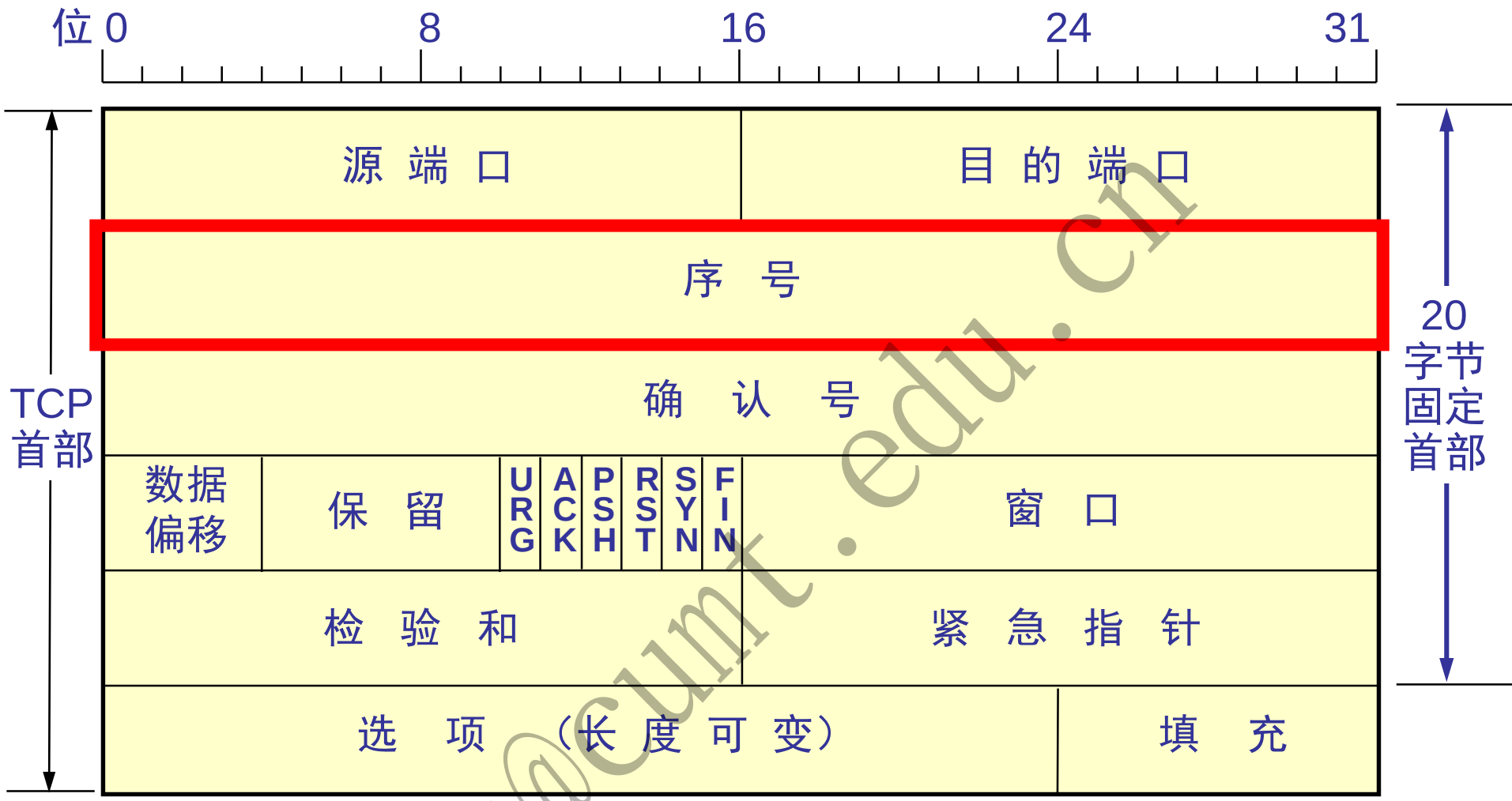


源端口和目的端口字段——各占 2 字节。端口是运输层与应用层的服务接口。运输层的复用和分用功能都要通过端口才能实现。

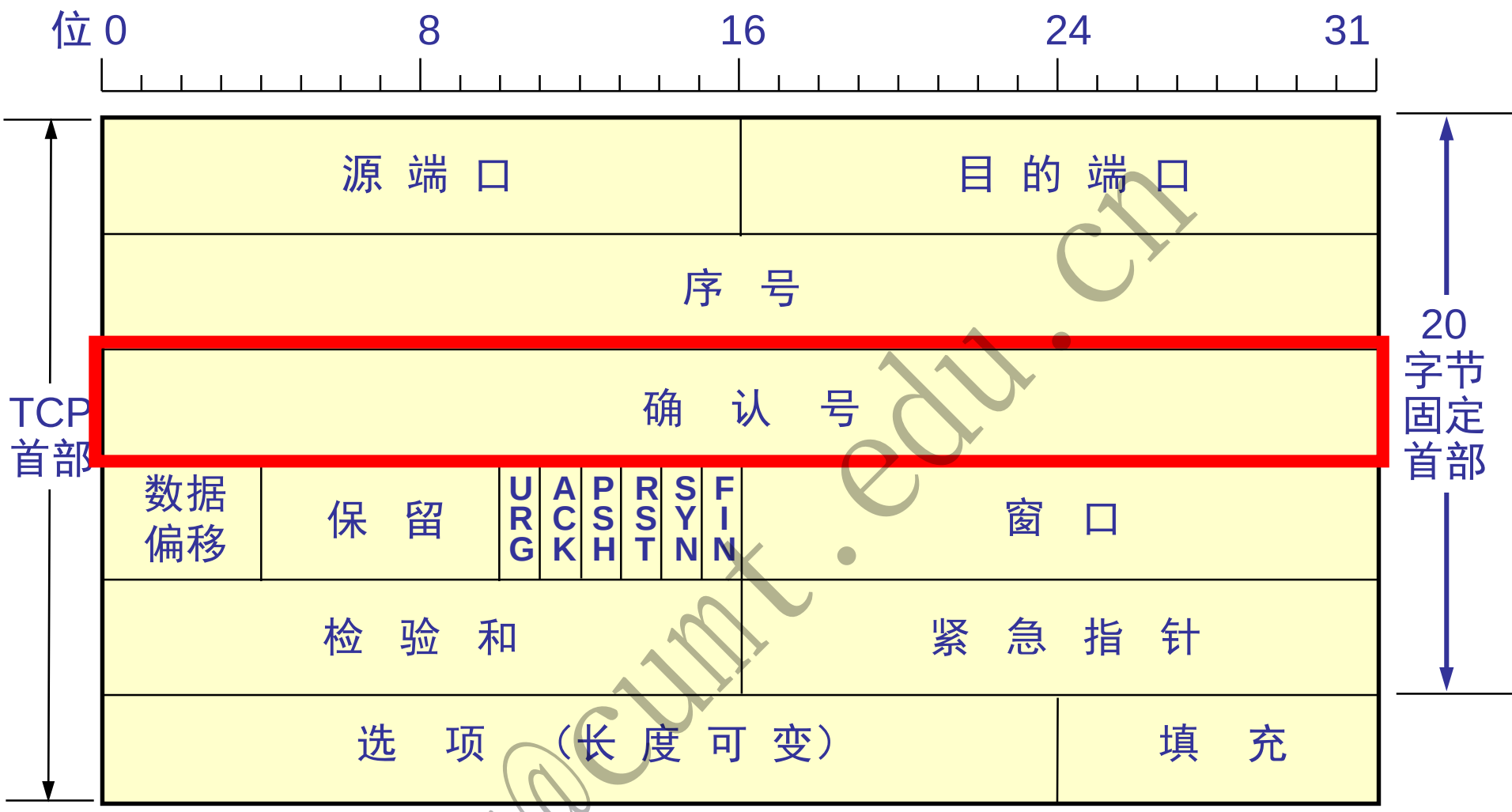


序号字段——占 4 字节。TCP 连接中传送的数据流中的每一个字节都编上一个序号，序号范围是[0, 2^{32} -1]，到达最大值后，下一个序号就又回到0。序号字段的值则指的是本报文段所发送的数据的第一个字节的序号。

- 如果用户要传送的数据的字节长度超过TCP报文段中序号字段可能编出的最大序号，那么还能否用TCP来传送？
  - 答案时肯定的。
  - 因为编号用完后可以再重复使用，但是应设法保证不出现编号的混乱。

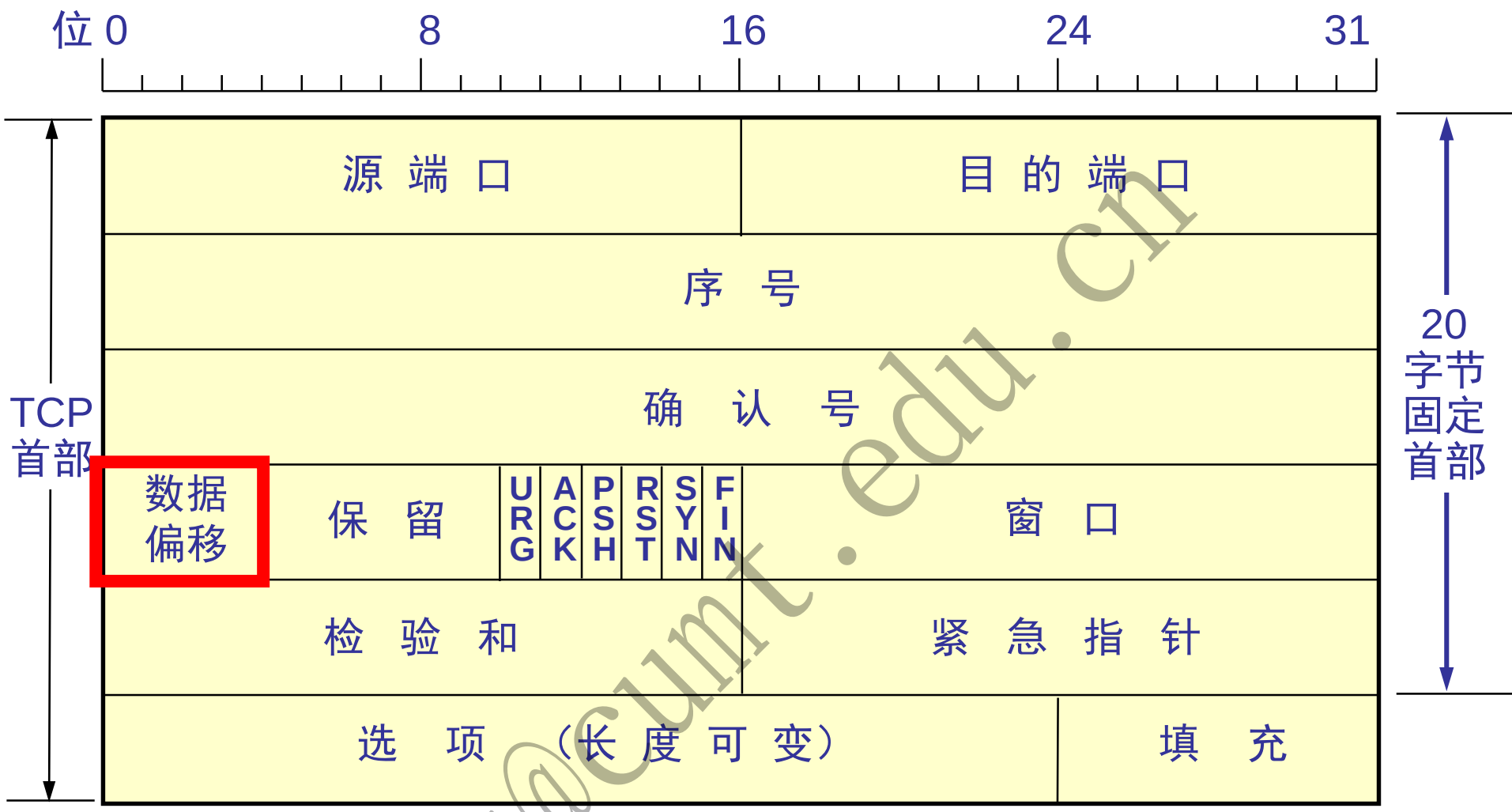


当使用高速网络时，在一次TCP连接的数据传送中序号很可能会被重复使用。例如，当使用1.5Mb/s的速率发送报文段时，序号重复要6小时以上。但若用2.5Gb/s的速率发送报文段，则不到14秒钟序号就会重复。 此时，可以定义时间戳选项配合序号来区分不同的报文段。

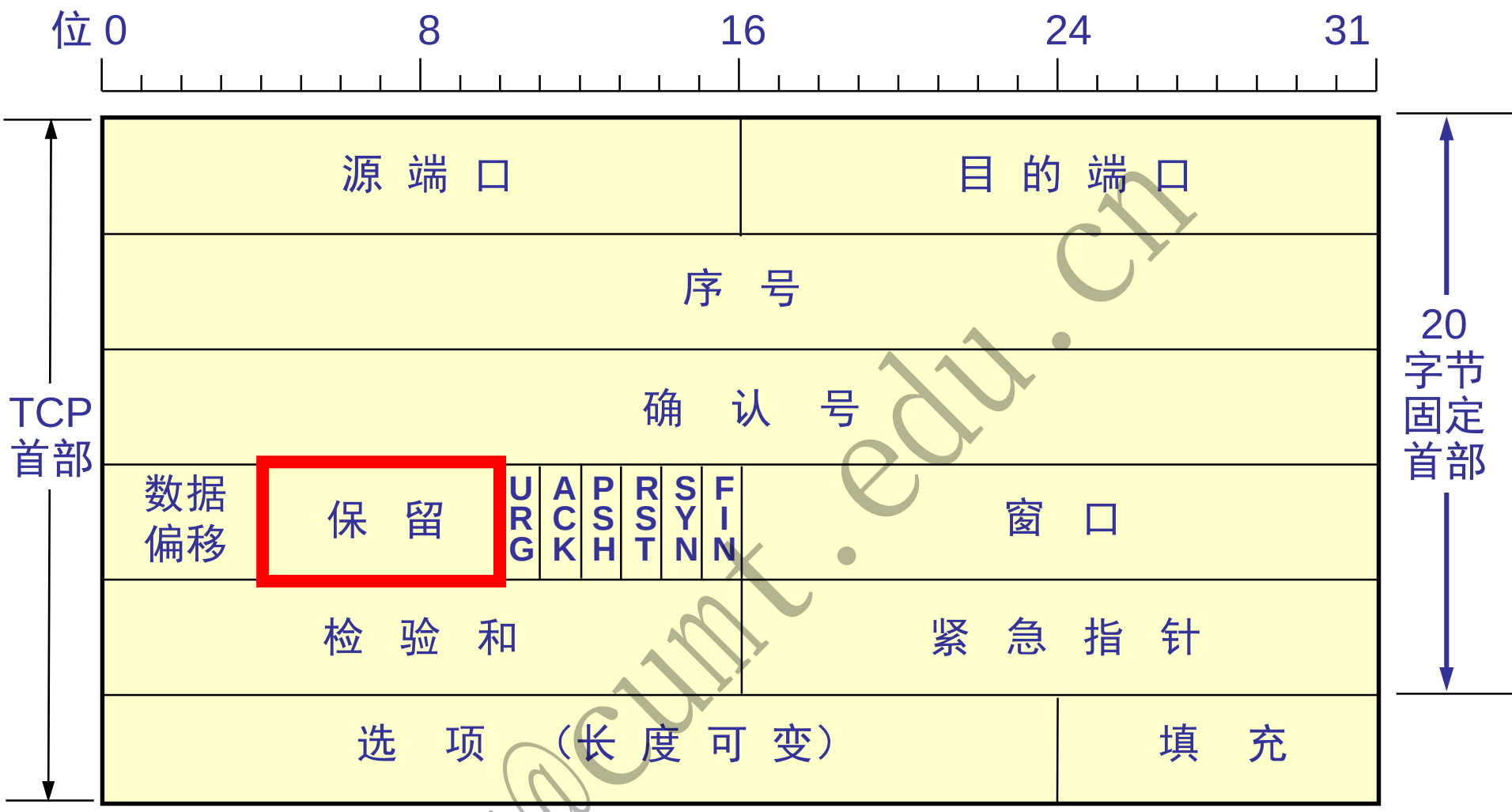


确认号字段——占 4 字节，是期望收到对方的下一个报文段的数据的第一个字节的序号。

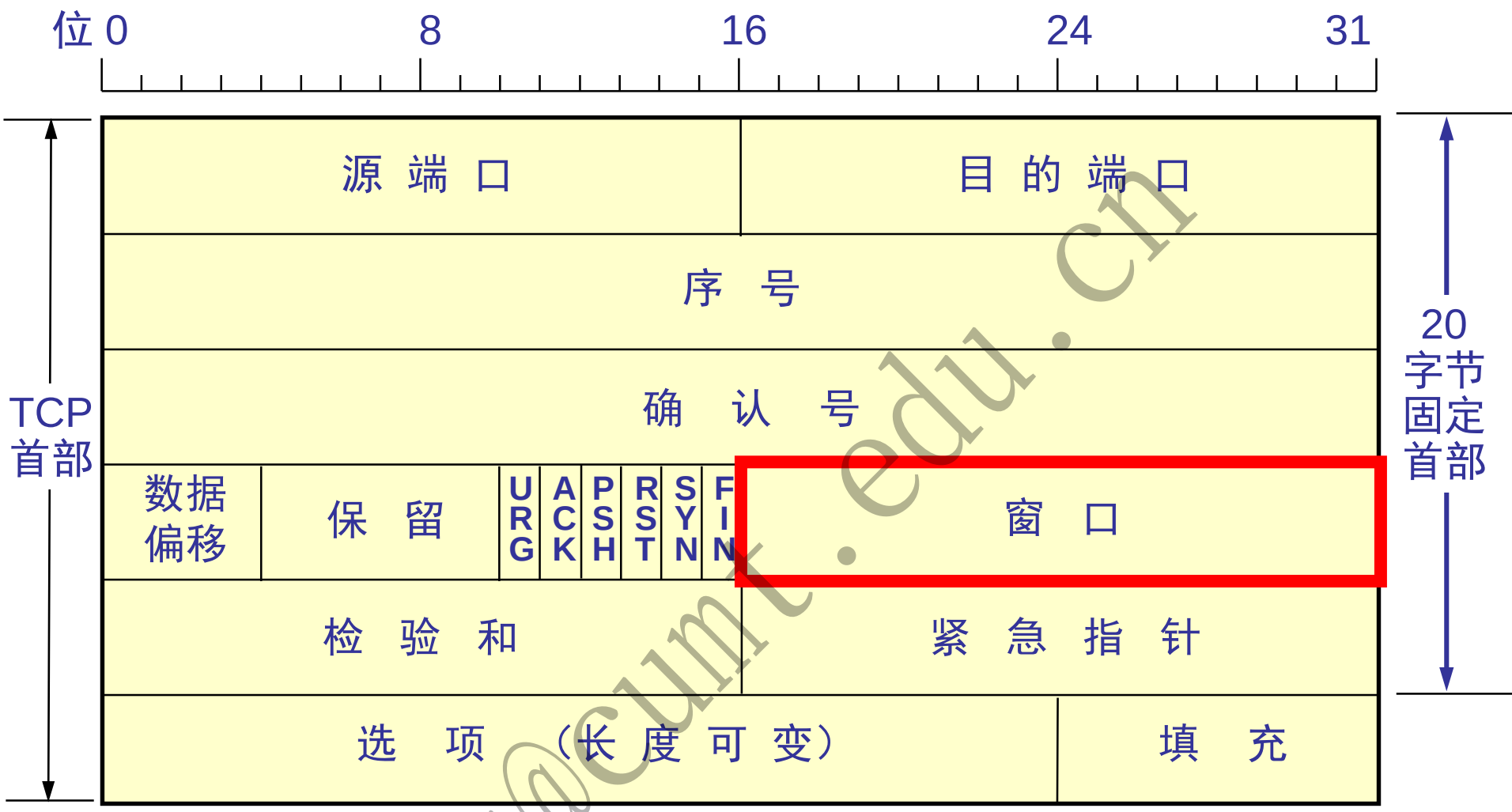
若确认号 = N，则表明：到序号N-1为止的所有数据都已正确收到。



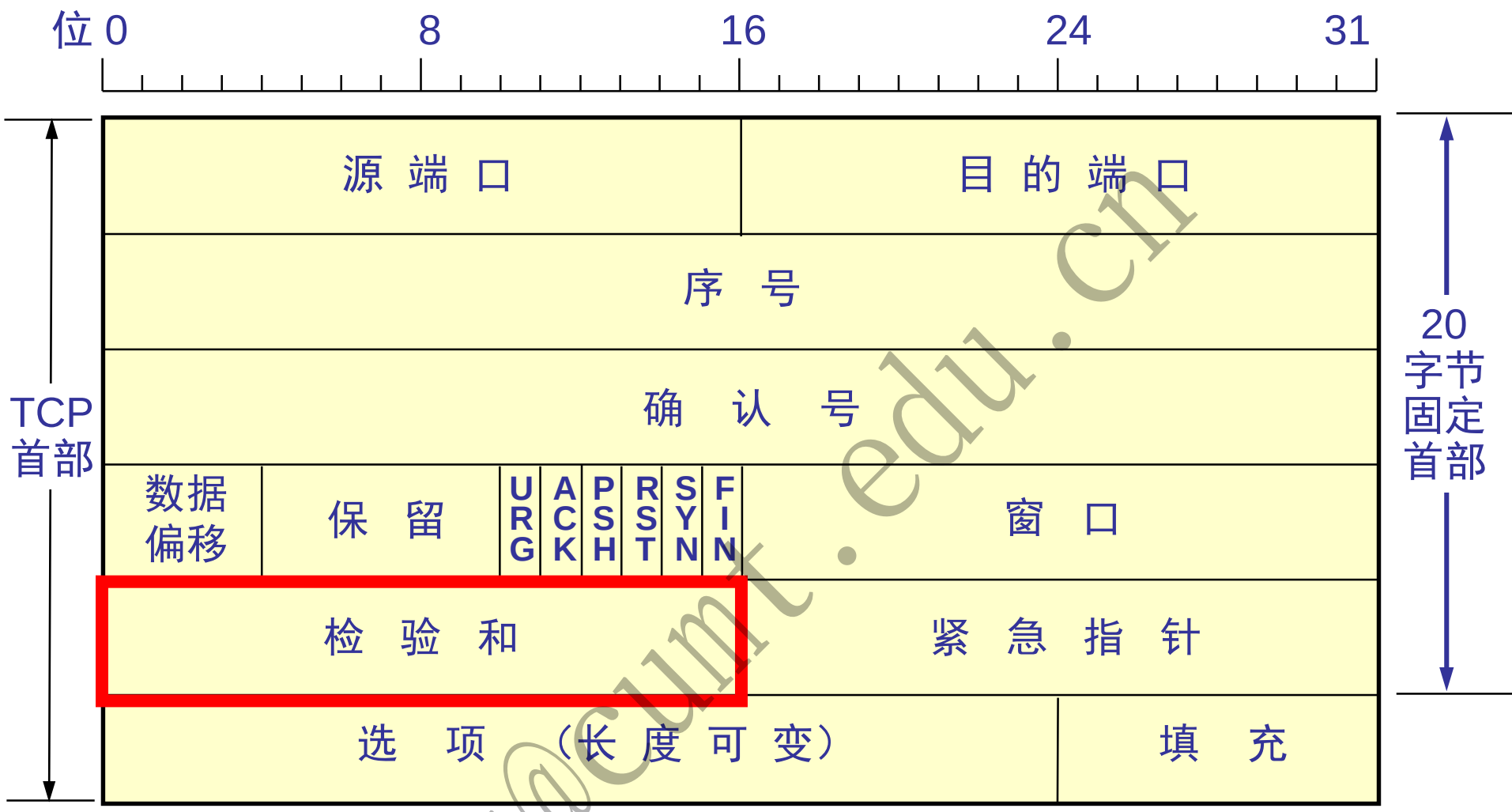
数据偏移（即首部长度）——占 4 位，它指出 TCP 报文段的数据起始处距离 TCP 报文段的起始处有多远。“数据偏移”的单位是 32 位字（ 4 字节），最大值是15*4=60字节，这也是TCP首部的最大长度（即选项长度不能超过40字节）。



保留字段——占 6 位，保留为今后使用，但目前应置为 0。

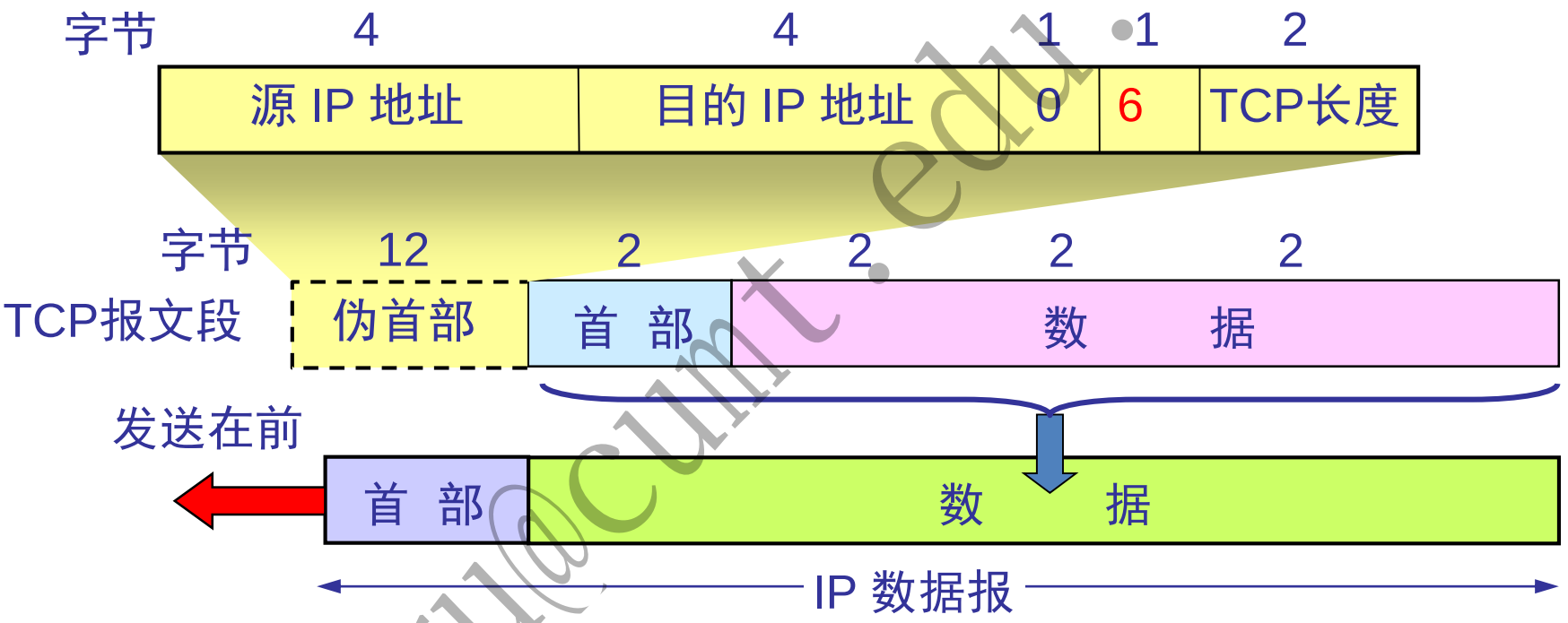


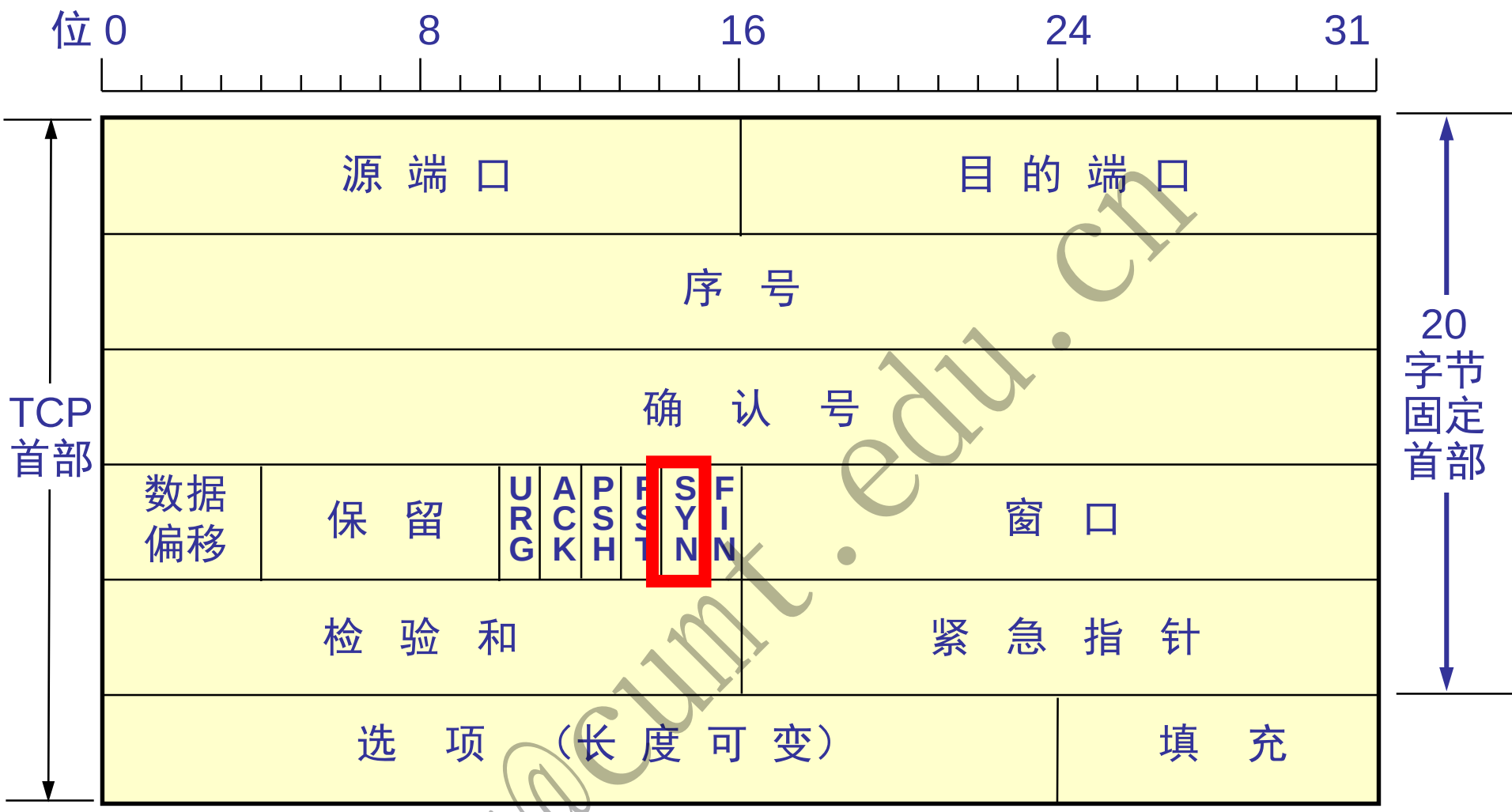
窗口 —— 占 2 字节，窗口大小是[0, 2^{16}-1]字节，作为接收方让发送方设置其发送窗口的依据。窗口字段明确指出了现在允许对方发送的数据量。窗口值经常在动态变化着。如果需要窗口值超过2^{16}-1，可以定义窗口扩大选项来解决。



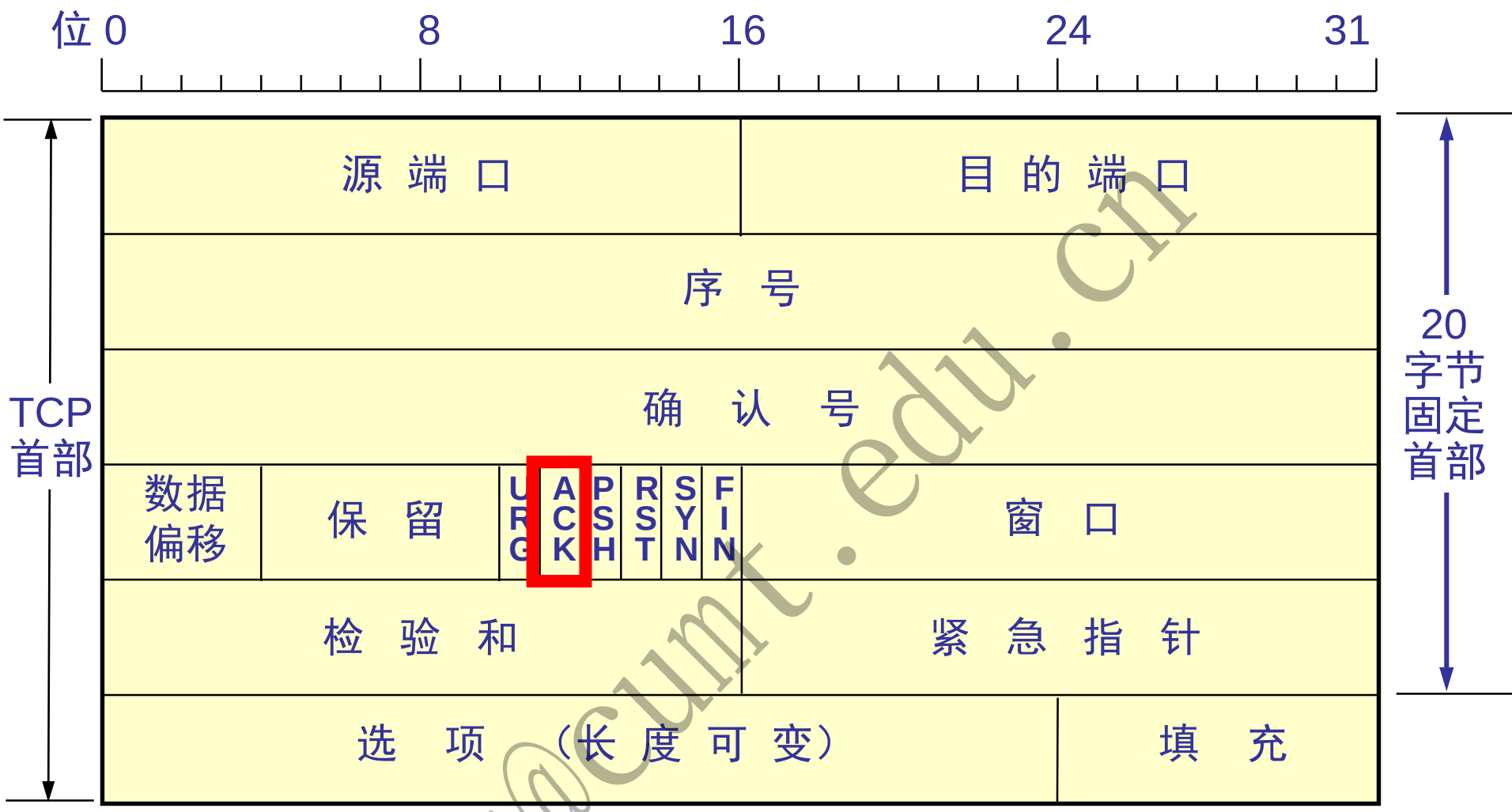
检验和 —— 占 2 字节。检验和字段检验的范围包括首部和数据这两部分。在计算检验和时，要在 TCP 报文段的前面加上 12 字节的伪首部。若使用IPv6，则相应的伪首部也要改变。

TCP的伪首部格式



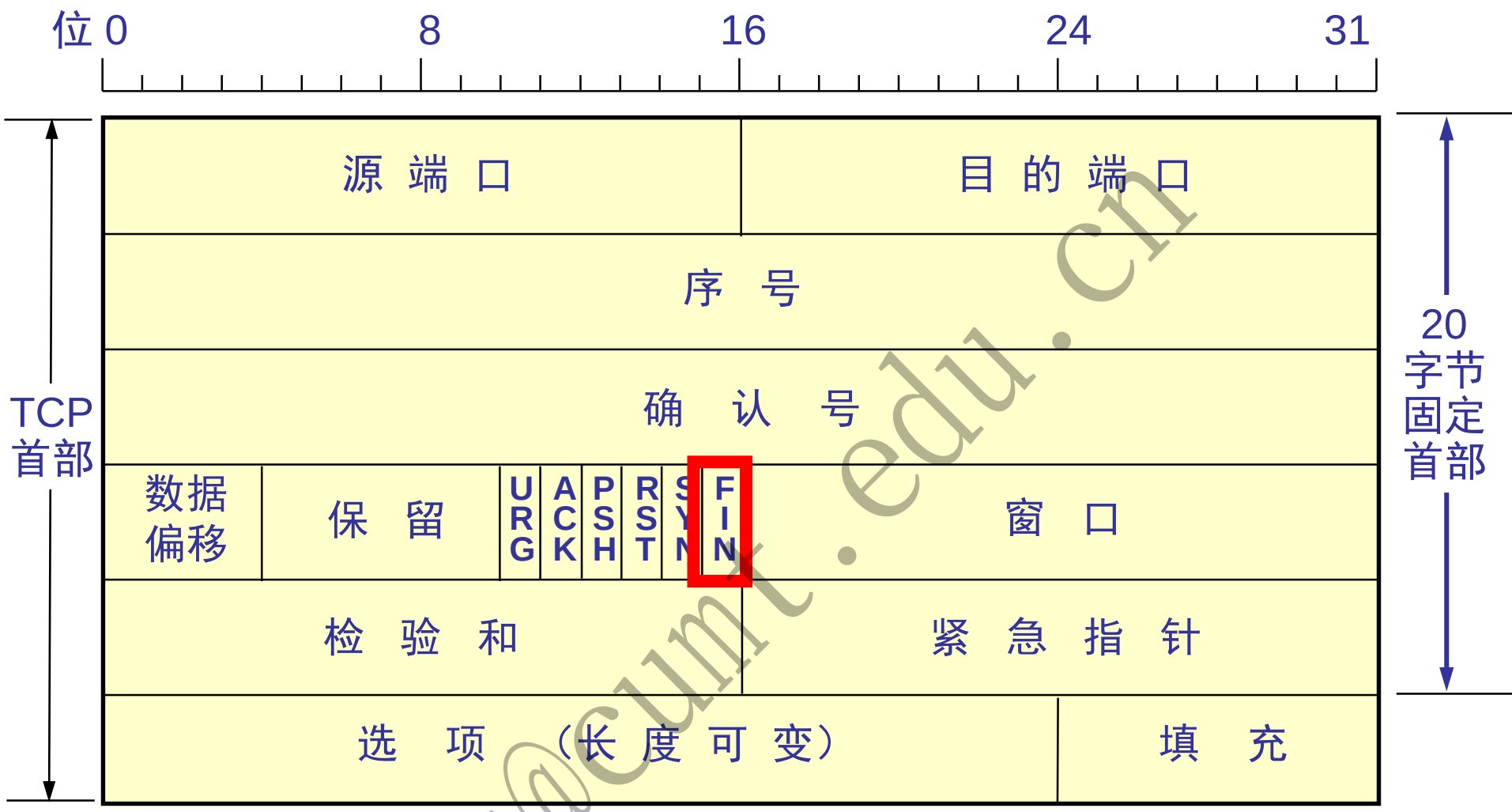


同步 SYN (SYNchronization) —— 在连接建立时用来同步序号。同步 SYN = 1 表示这是一个连接请求或连接接受报文。

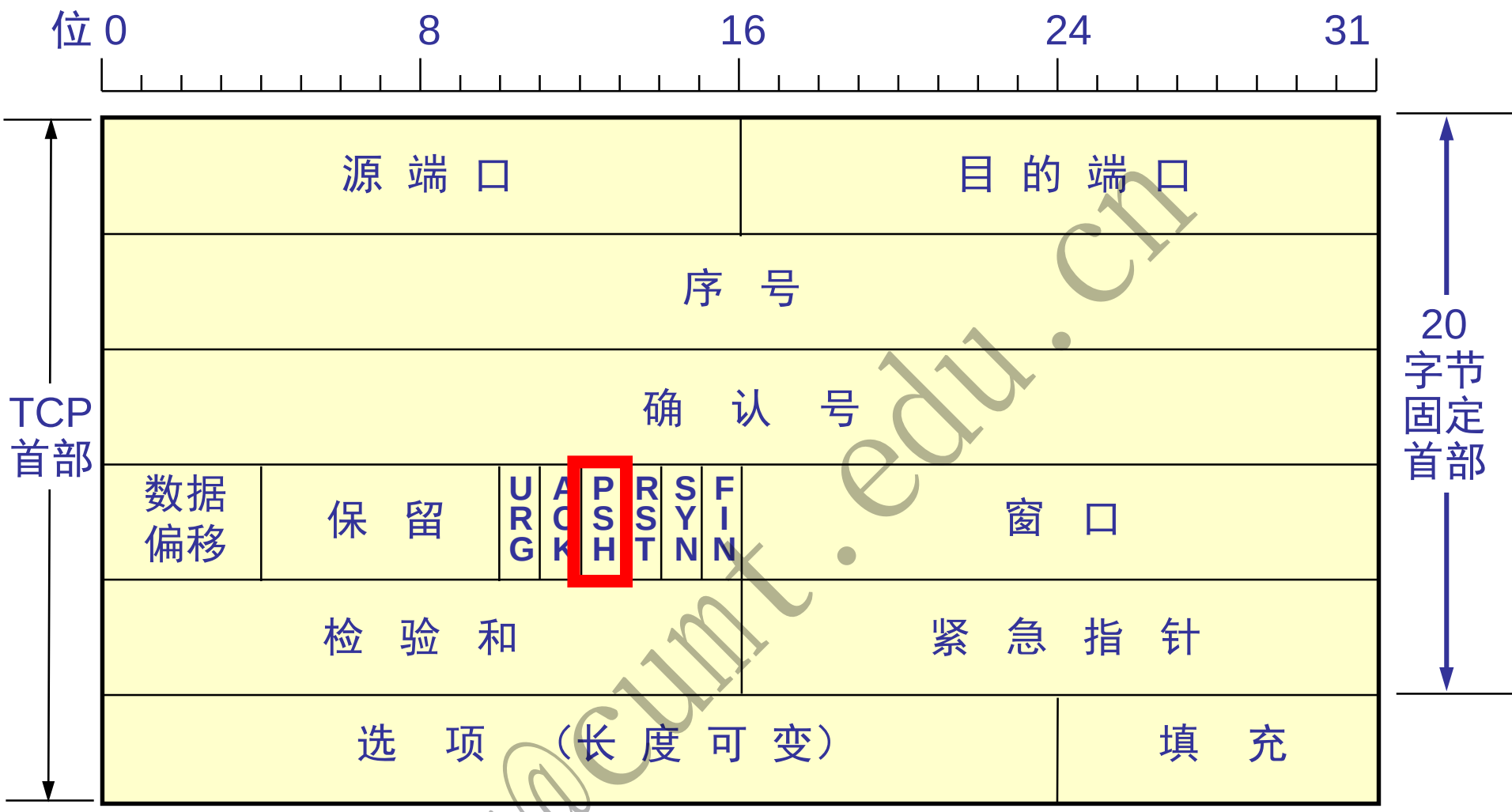


确认 ACK ——只有当 ACK = 1 时确认号字段才有效。当 ACK = 0 时，确认号无效。

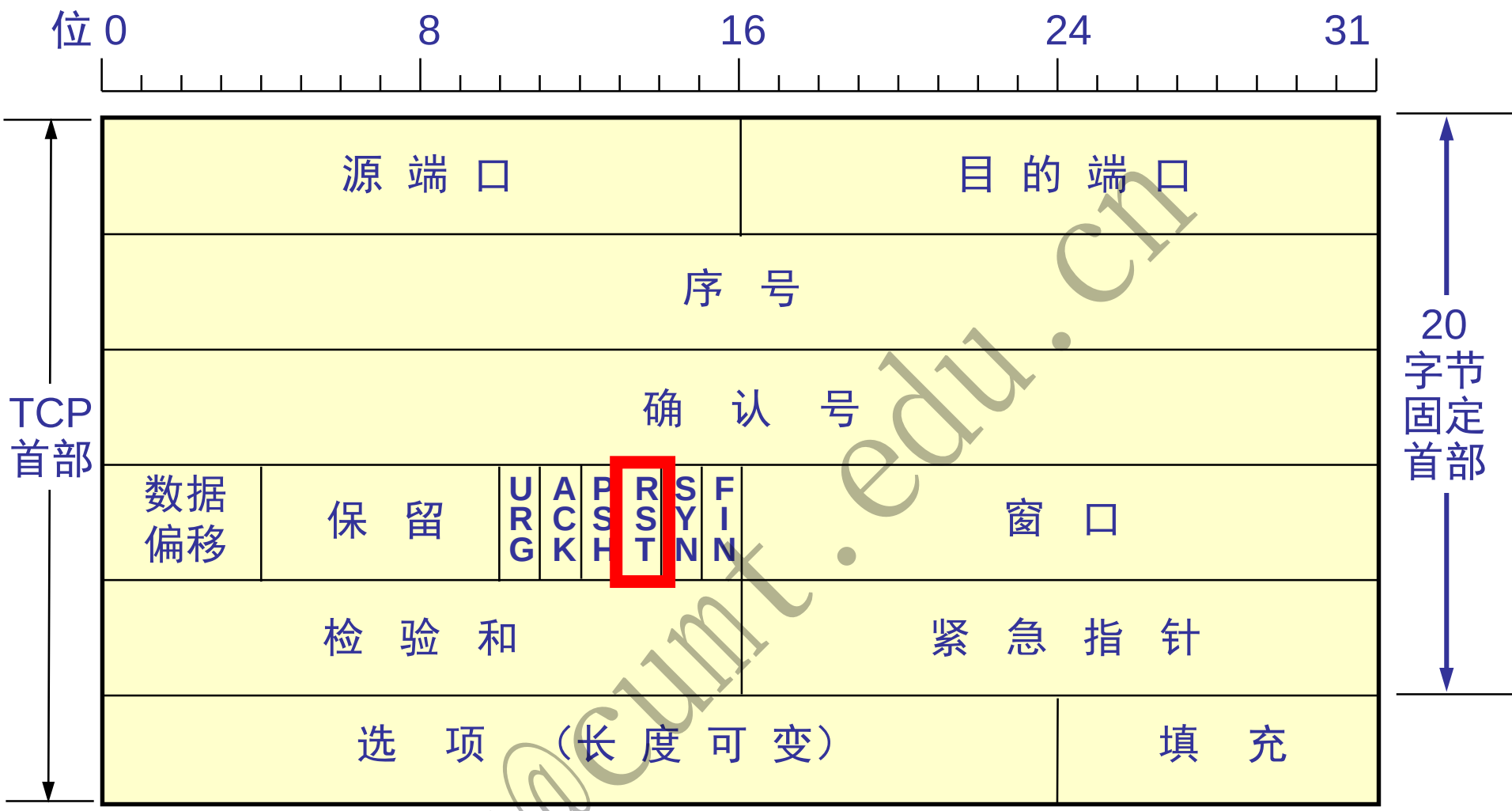
TCP规定，在连接建立后所有传送的报文段都必须把ACK置1。



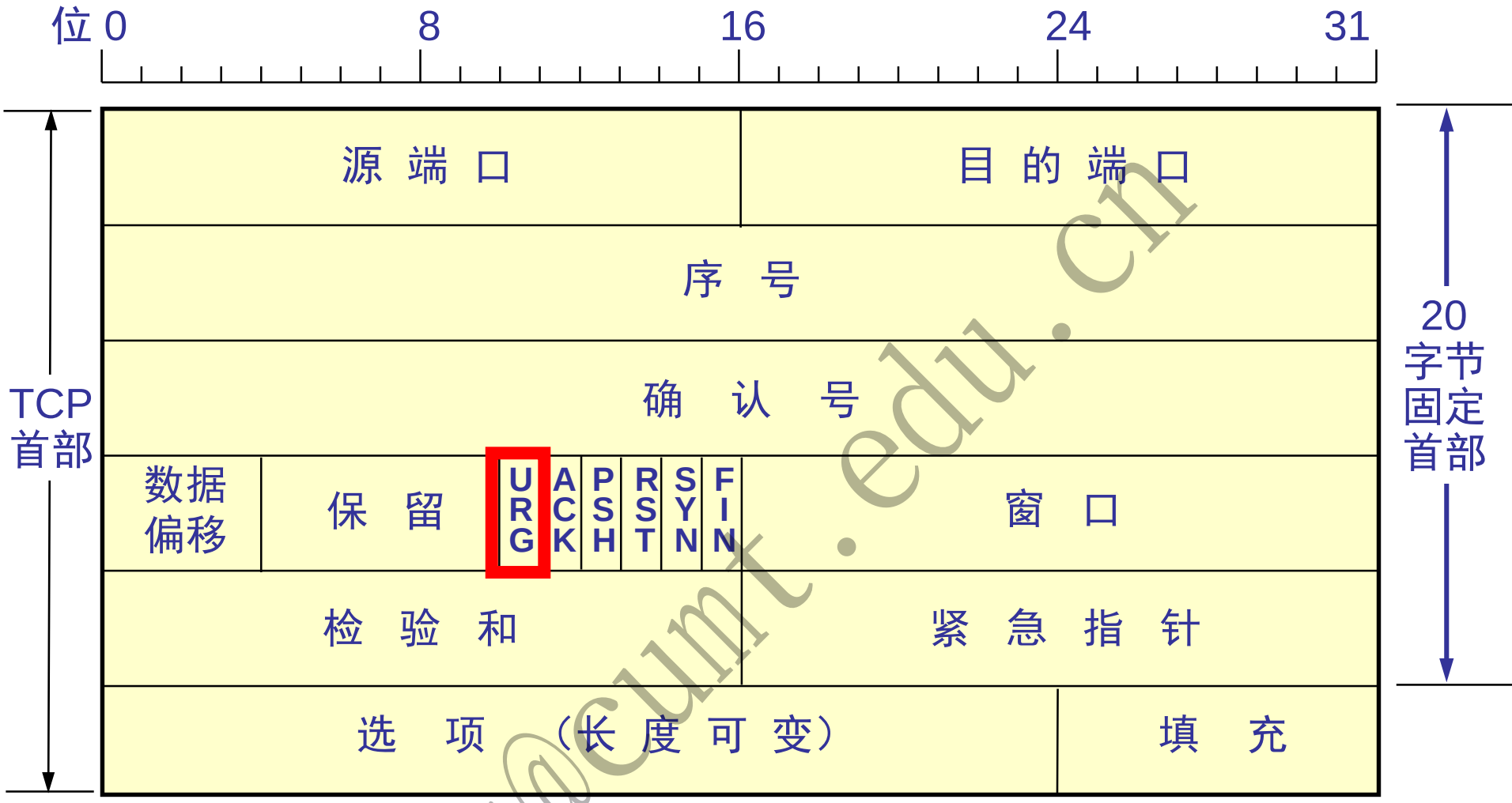
终止 FIN (FINis) —— 用来释放一个连接。FIN = 1 表明此报文段的发送端的数据已发送完毕，并要求释放运输连接。



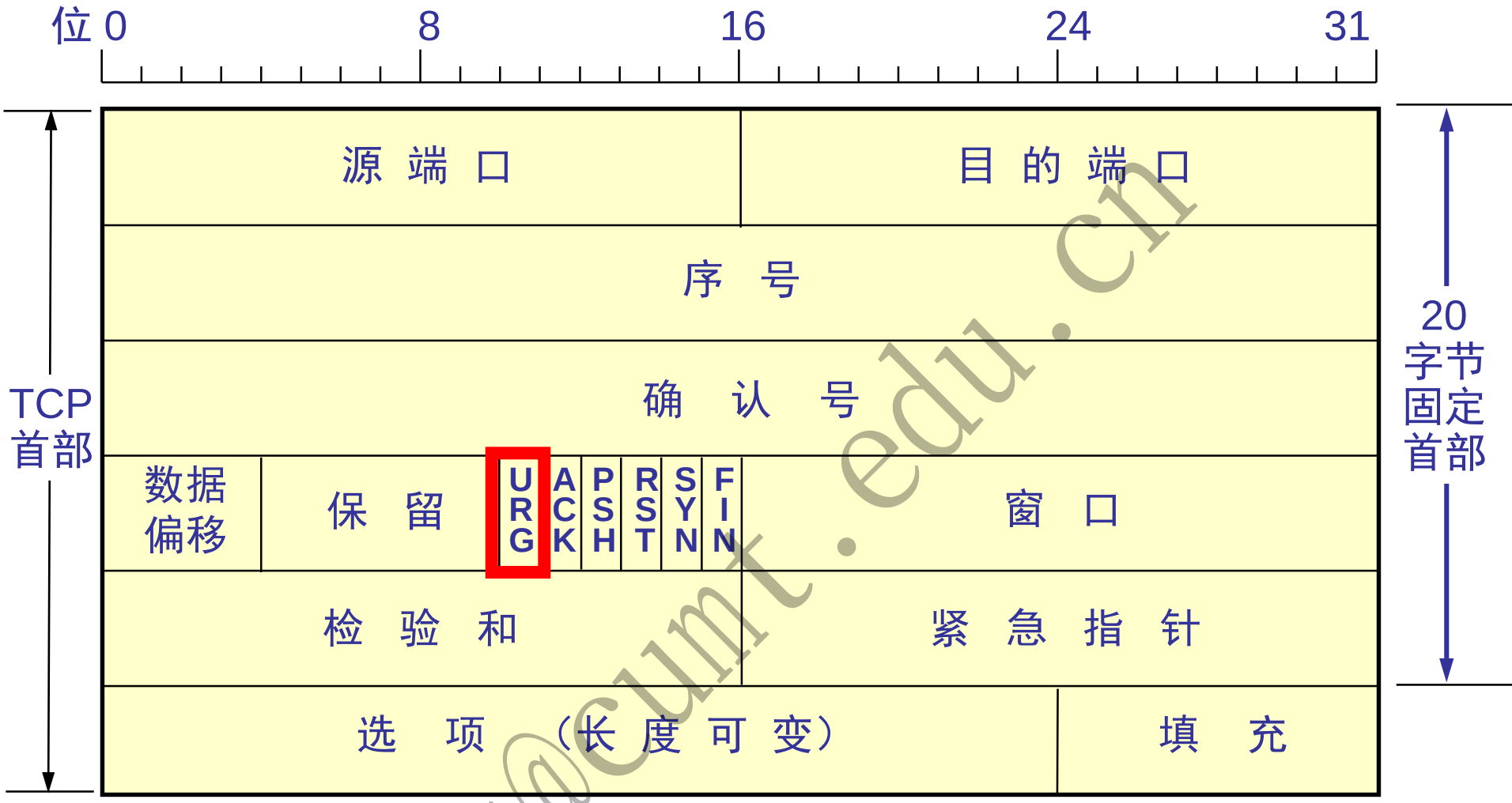
推送 PSH (PuSH) —— 发送方把PSH置1时，就立即创建一个报文段发送出去，而接收 TCP 收到 PSH = 1 的报文段，就尽快地交付接收应用进程，而不再等到整个缓存都填满了后再向上交付。但推送操作很少使用。



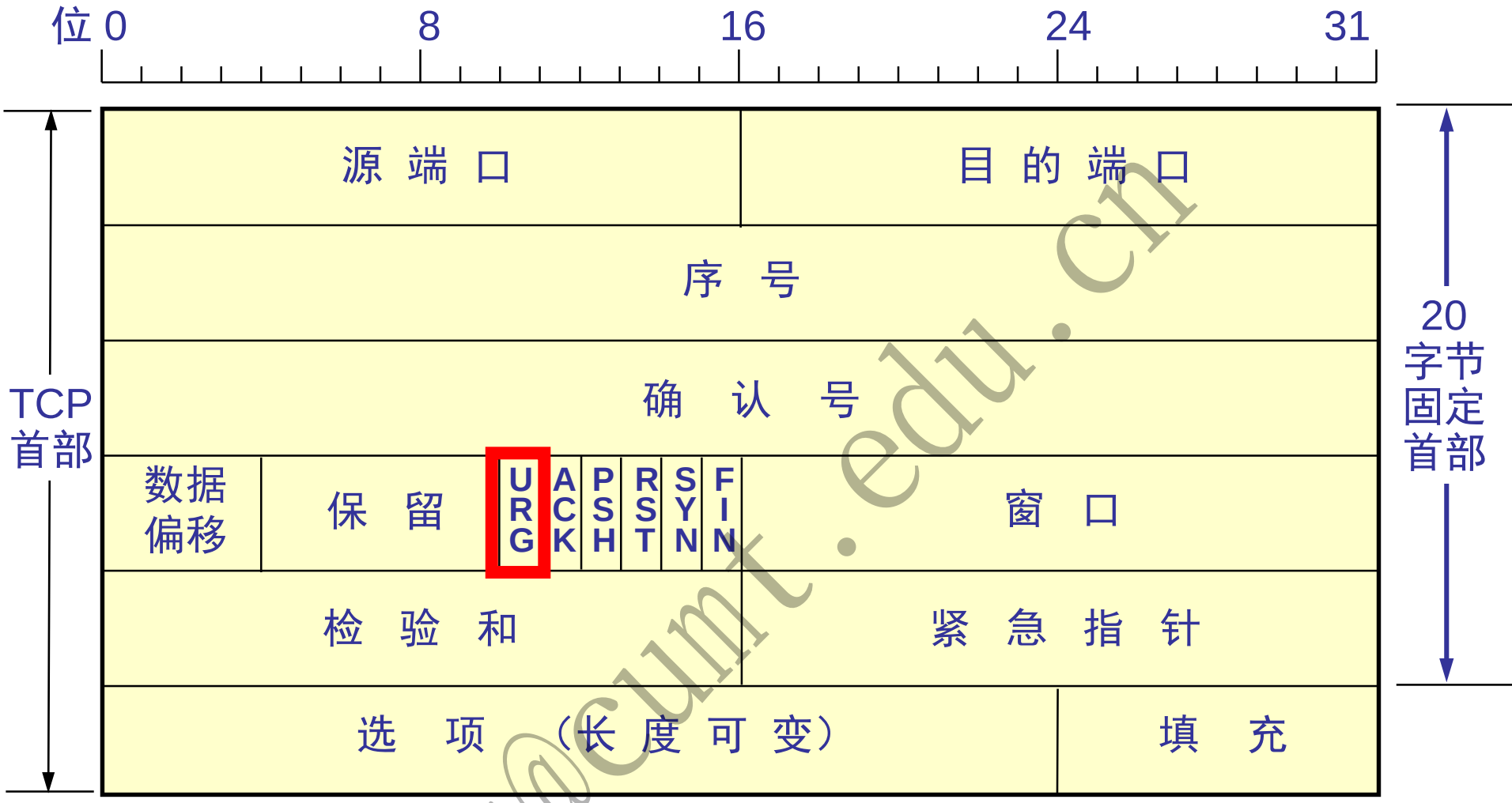
复位 RST (ReSeT) —— 当 RST = 1 时，表明 TCP 连接中出现严重差错（如由于主机崩溃或其他原因），必须释放连接，然后再重新建立运输连接。RST置1还用来拒绝一个非法的报文段或拒绝打开一个连接。



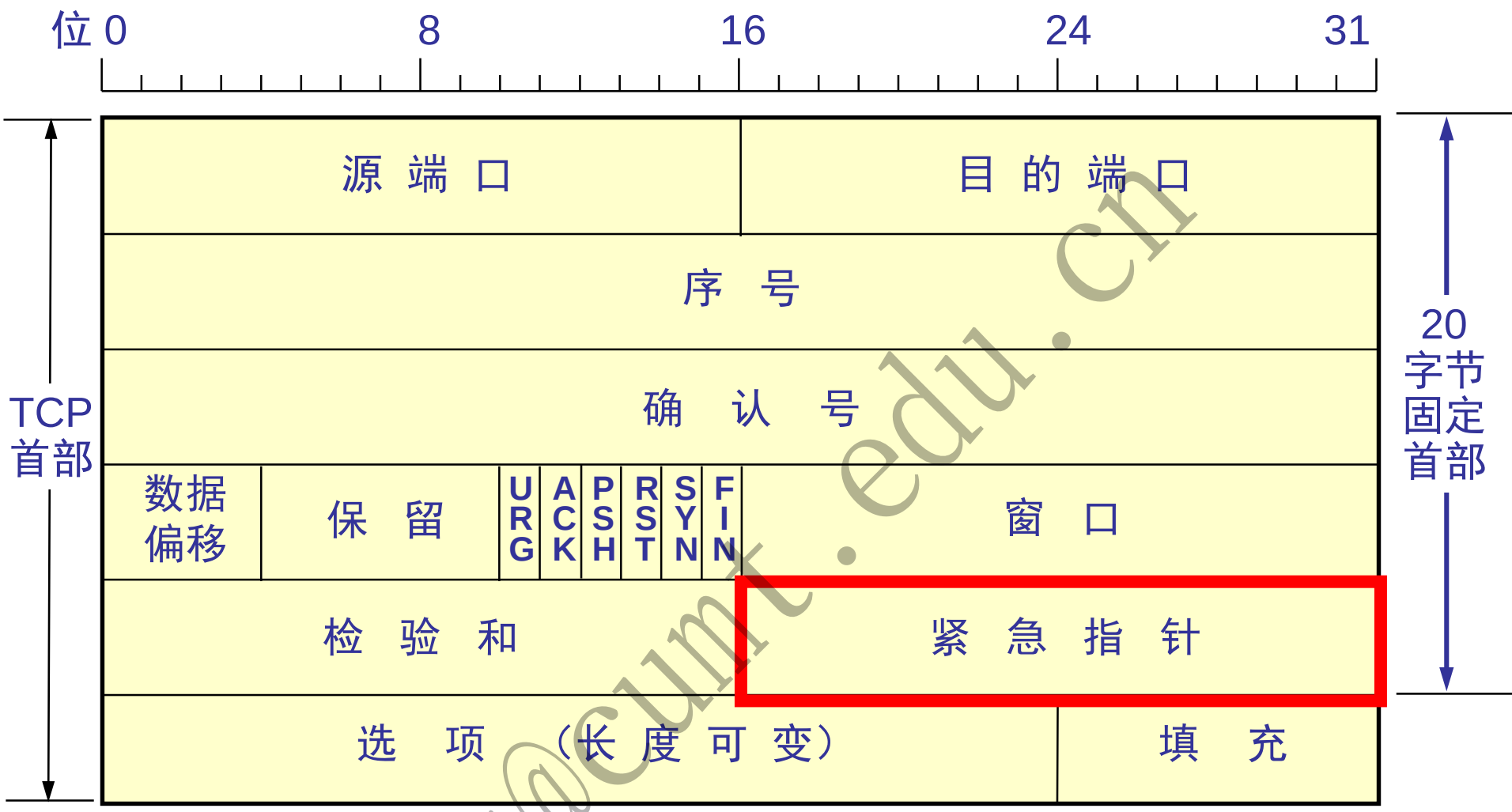
紧急 URG —— 当 URG = 1 时，表明紧急指针字段有效。它告诉系统此报文段中有紧急数据，应尽快传送(相当于高优先级的数据)，而不要按原来的排队顺序来传送。



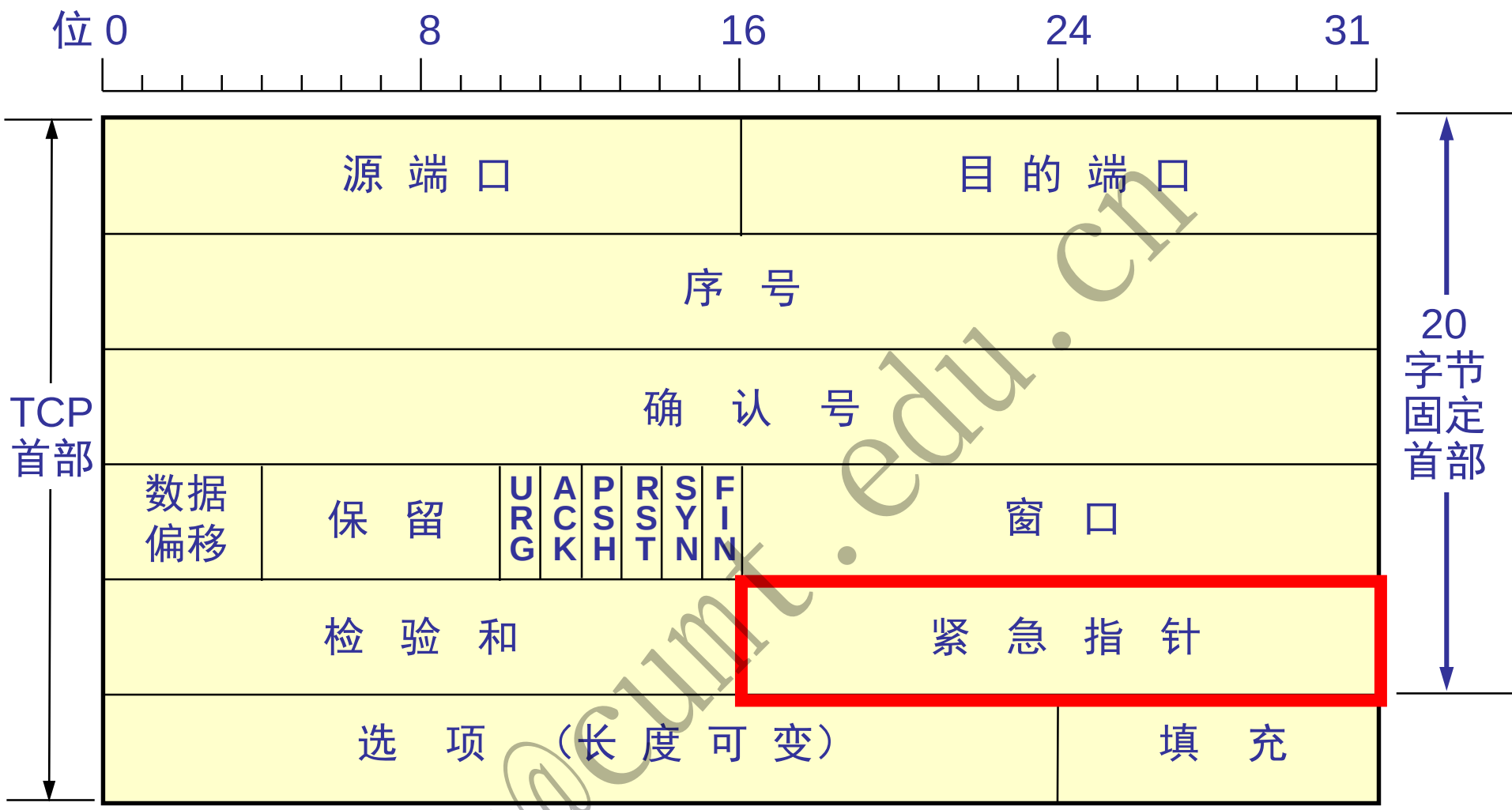
例如，用户想要取消在远程主机上运行的一个程序时，从键盘发出中断命令(Ctrl+C)。如果不使用紧急数据，那么这两个字符将存储在接收TCP的缓存末尾。只有在所有的数据被处理完毕后这两个字符才被交付接收方的应用进程，会误事。



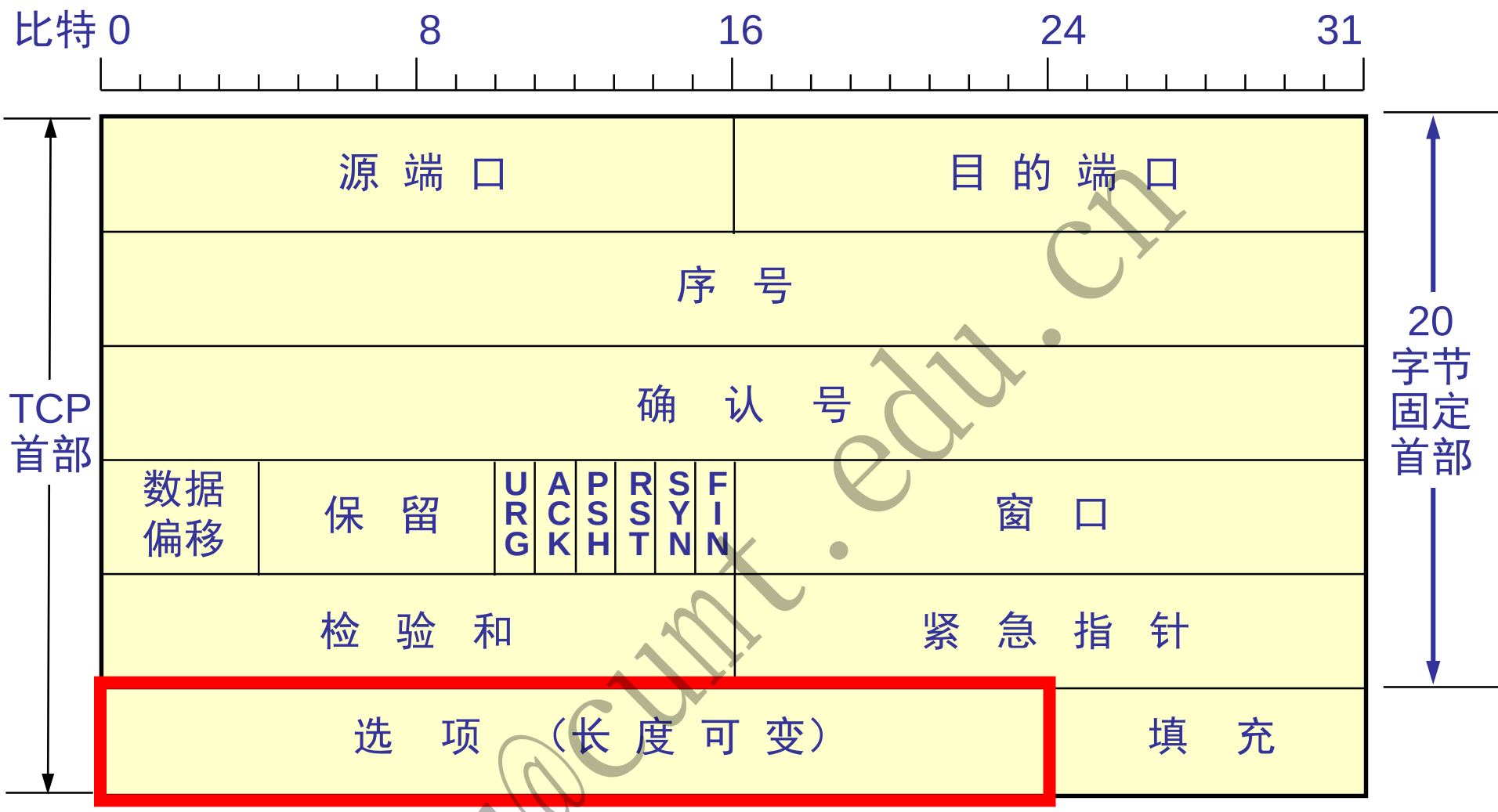
当 URG = 1 时，发送应用进程就告诉发送方的TCP有紧急数据要发送。于是发送方TCP就把紧急数据插入到本报文段数据的最前面，而在紧急数据后面的数据仍是普通数据。 这时要与紧急指针(Urgent Pointer)字段配合使用。



紧急指针字段 —— 占 16 位。紧急数据放在本报文段数据的最前面，而紧急指针指出了紧急数据的末尾在报文段中的位置。这样就可以得出在本报文段中共有多少字节的紧急数据。



在2011年公布的RFC6093中给出了紧急指针字段明确使用方法。当所有紧急数据都处理完时，TCP就告诉应用程序恢复到正常操作。即使窗口为零时也可发送紧急数据。



选项字段——长度可变。

TCP 最初只规定了一种选项，即最大报文段长度 MSS(Maximum Segment Size)——TCP 报文段中的数据字段的最大长度。

MSS的作用

- MSS 告诉对方TCP：“我的缓存所能接收的报文段的数据字段的最大长度是 MSS 个字节，不包括TCP首部长度。”
- MSS 与接收窗口值没有关系。
  - 规定MSS并不是因为要考虑接收方的接收缓存可能放不下TCP报文段中的数据。

为什么要规定 MSS ？

- 若选择较小的 MSS 长度，网络的利用率就降低。
  - 当 TCP 报文段只含有 1 字节的数据时，在 IP 层传输的数据报的开销至少有 40 字节(包括 TCP 报文段的20字节固定首部和 IP 数据报的20字节固定首部)。这样，对网络的利用率就不会超过 1/41。到了数据链路层还要加上一些开销。
- 若 TCP 报文段非常长，也会使开销增大。
  - 在 IP 层传输时就有可能要分解成多个短数据报片。在终点要把收到的各个短数据报片装配成原来的 TCP 报文段。当传输出错时还要进行重传。

MSS值的确定

- 因此，只要在 IP 层传输时不需要再分片，MSS就应尽可能大些。
  - MTU(max transmission unit)实际上是数据链路对IP层包长度要求的最大值，比如以太网802.3帧要求数据部分不得超过1500字节，否则就要求IP层进行分片
  - 由于 IP 数据报所经历的路径是动态变化的，因此在这条路径上确定的不需要分片的 MSS，如果改走另一条路径就可能需要进行分片。
- 但是，最佳的 MSS 是很难确定的。

MSS值的确定

- MSS值的设置方法
  - 在连接建立的过程中，双方都把自己能够支持的MSS写入这一字段，以后就按照这个数值传递数据，两个传送方向可以有不同的MSS值。
  - 默认值是536字节。所有主机都应能接受的报文段长度是536+20=556字节。

窗口扩大选项

- TCP首部中窗口字段长度是16位，因此最大的窗口大小为(64K-1)字节（65535字节）。
  - 虽然这对早期的网络是足够用的，但对于包含卫星信道的网络，传播时延和带宽都很大，要获得高吞吐率需要更大的窗口大小。

窗口扩大选项

- 窗口扩大选项——占 3 字节，其中有一个字节表示移位值 S。
  - 新的窗口值等于TCP 首部中的窗口位数增大到(16 + S)，相当于把窗口值向左移动 S 位后获得实际的窗口大小。
  - 移位值S允许使用的最大值是14，相当于窗口最大值增大到$2^{(16+14)}-1=2^{30}-1$。
  - 窗口扩大选项可以在双方初始建立TCP连接时进行协商。如果连接的某一段实现了窗口扩大，当它不再需要扩大其窗口时，可发送S=0的选项，使窗口大小回到16。

时间戳选项

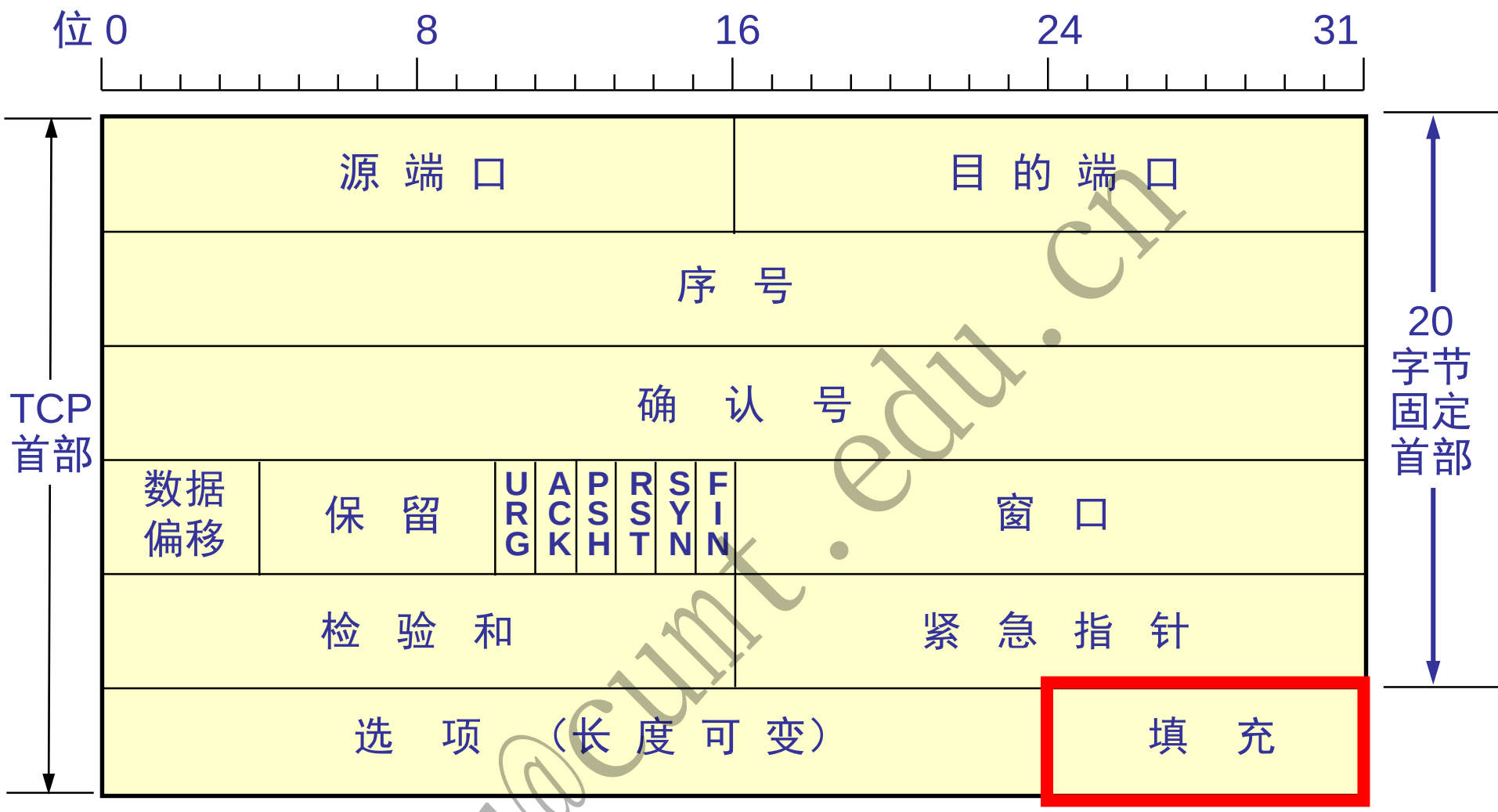
- 时间戳选项——占10 字节，其中最主要的字段时间戳值字段(4 字节)和时间戳回送回答字段(4 字节)。
- 时间戳选项有以下两个功能：
  - 第一，用来计算往返时间RTT。
    - ✓ 发送方在发送报文段时把当前时钟的时间值放入时间戳，接收方在确认该报文段时把时间戳字段值复制到时间戳回送回答字段。因此，发送方在收到确认报文后，可以准确地计算出RTT来。

时间戳选项

– 第二，用于处理TCP序号超过2^{32}的情况，这又称为防止序号绕回PAWS（Protect Against Wrapped Sequence numbers）。
  - ✓TCP报文段的序号只有32位，而每增加2^{32}个序号就会重复使用原来用过的序号。
  - ✓当使用高速网络时，在一次TCP连接的数据传送中序号很可能会被重复使用。
  - ✓为使接收方能够把新的报文段和迟到很久的报文段区分开来，可以在报文段中加上这种时间戳。

选择确认选项

- 选择确认选项——用于SACK机制。
  - 如果使用选择确认，那么原来首部中的“确认号字段”的用法仍然不变。只是以后在 TCP 报文段的首部中都增加了SACK选项，以便报告收到的不连续的字节块的边界。
  - 但是，指明一个边界需要用掉4字节(因为序号有32位，需要使用4个字节表示)，而TCP首部最多40字节，因此SACK选项中最多只能指明4个字节块的边界信息(共8个边界，需要32字节)，另外还需要一个字节指明是SACK选项，另一个字节指明这个选项要占用多少字节。
  - 然而，SACK文档并没有指明发送方应当怎样响应SACK。因此大多数的实现还是重传所有未被确认的数据块。



填充字段 —— 这是为了使整个首部长度是 4 字节的整数倍。

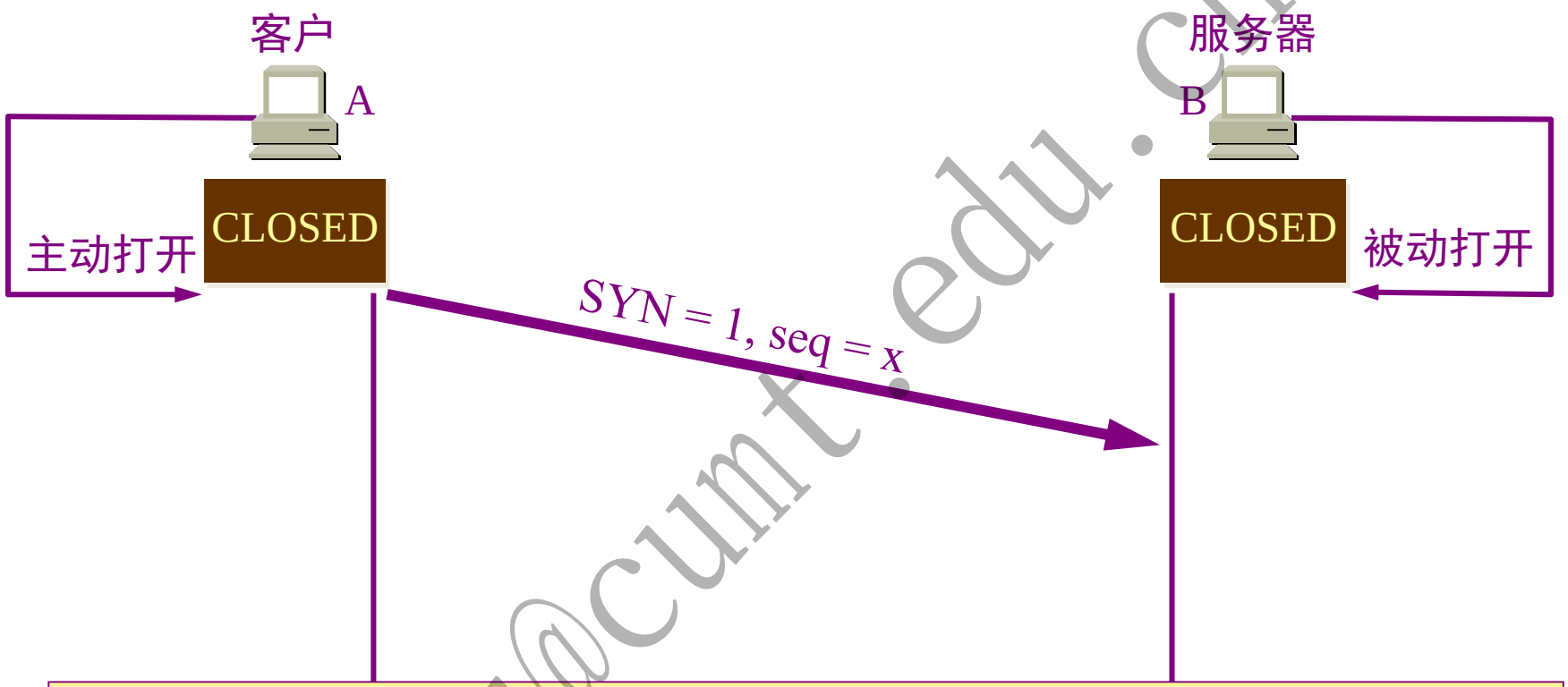
Q16：如何建立TCP连接？

- TCP 是面向连接的协议。
- 运输连接有三个阶段：
  - 连接建立（采用客户服务器方式）
  - 数据传送
  - 连接释放
- 运输连接的管理就是使运输连接的建立和释放都能正常地进行。

TCP 连接建立过程中要解决的三个问题

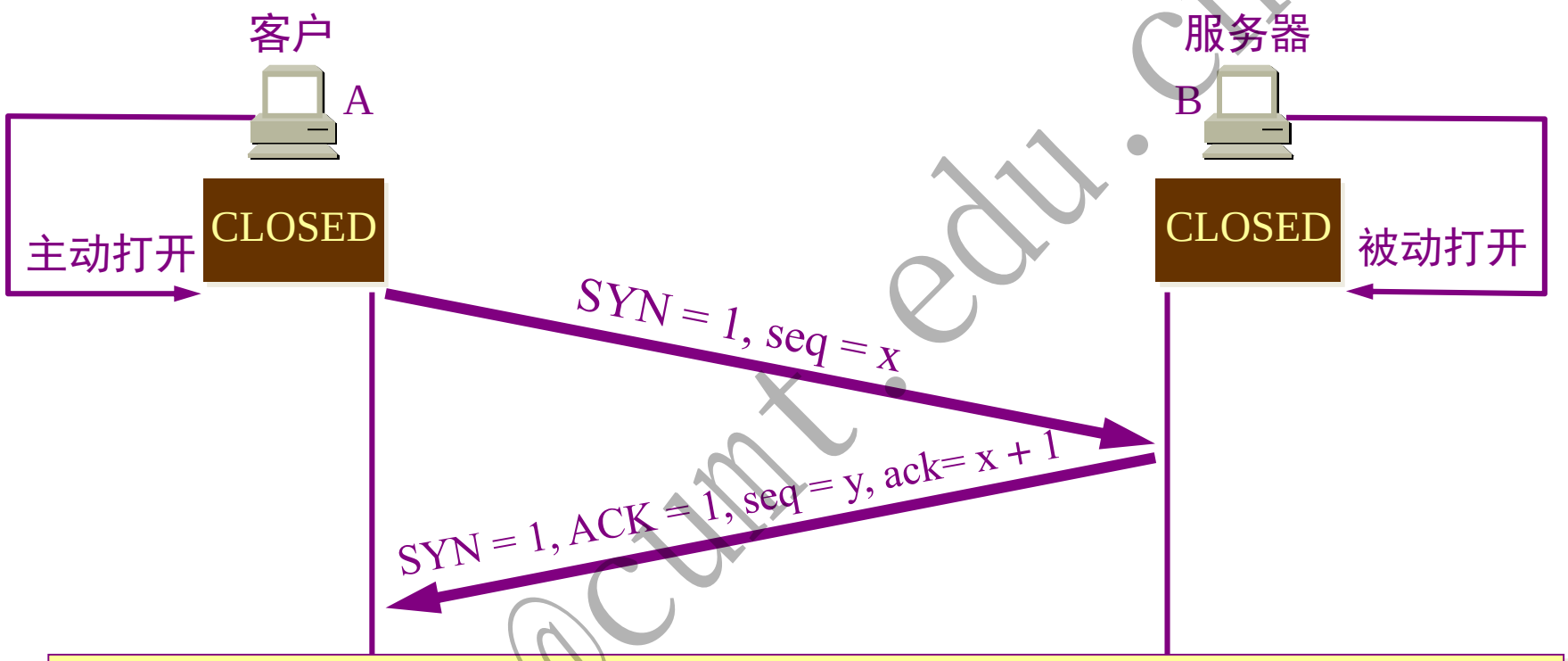
- (1) 要使每一方能够确知对方的存在。
- (2) 要允许双方协商一些参数（如最大窗口值、是否使用窗口扩大选项和时间戳选项以及服务质量等）。
- (3) 能够对运输实体资源（如缓存大小、连接表中的项目等）进行分配。

用三次报文握手建立 TCP 连接



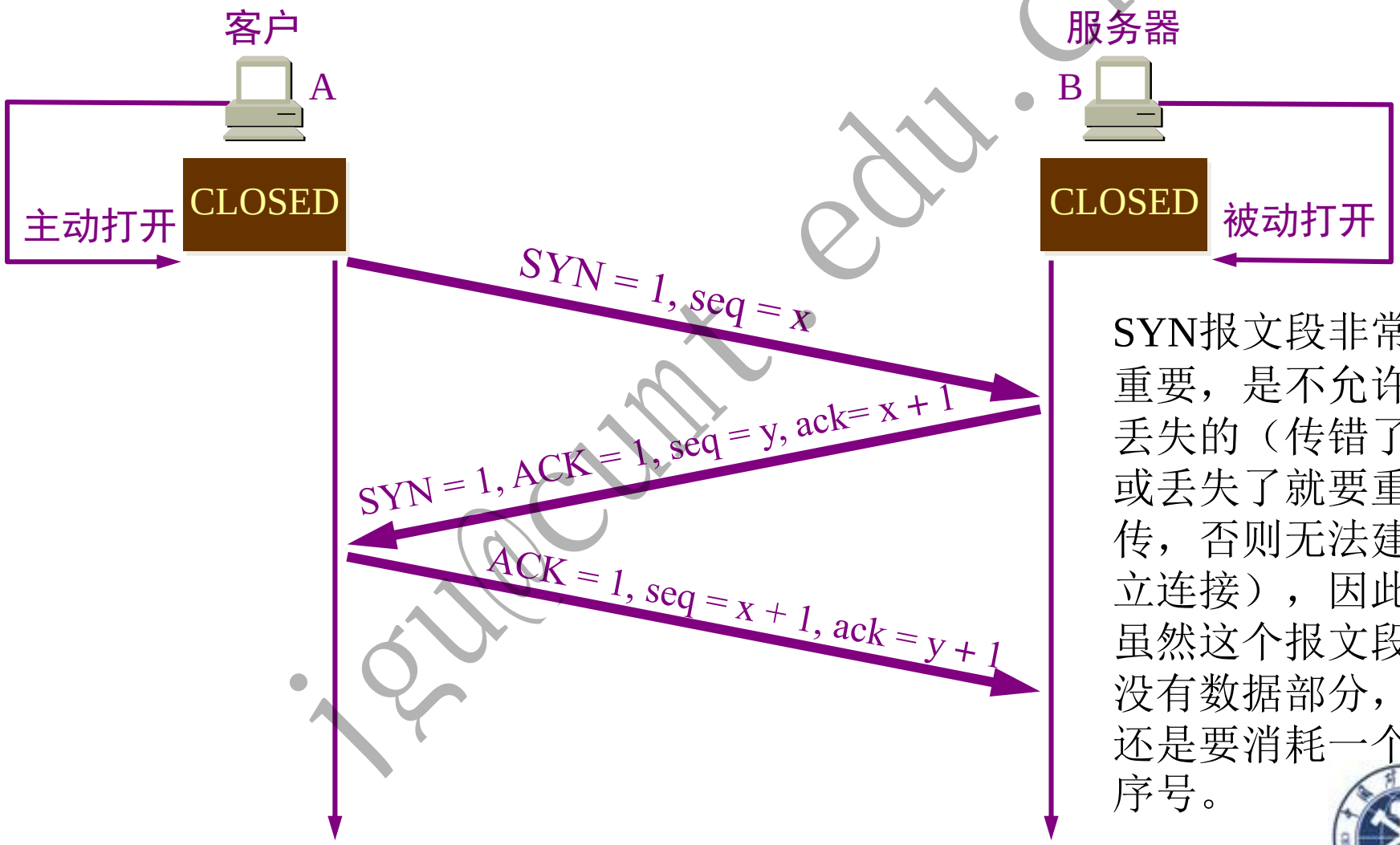
A 的 TCP 向 B 发出连接请求报文段，其首部中的同步位 SYN = 1，并选择序号 seq = x，表明传送数据时的第一个数据字节的序号是 x。

用三次报文握手建立 TCP 连接



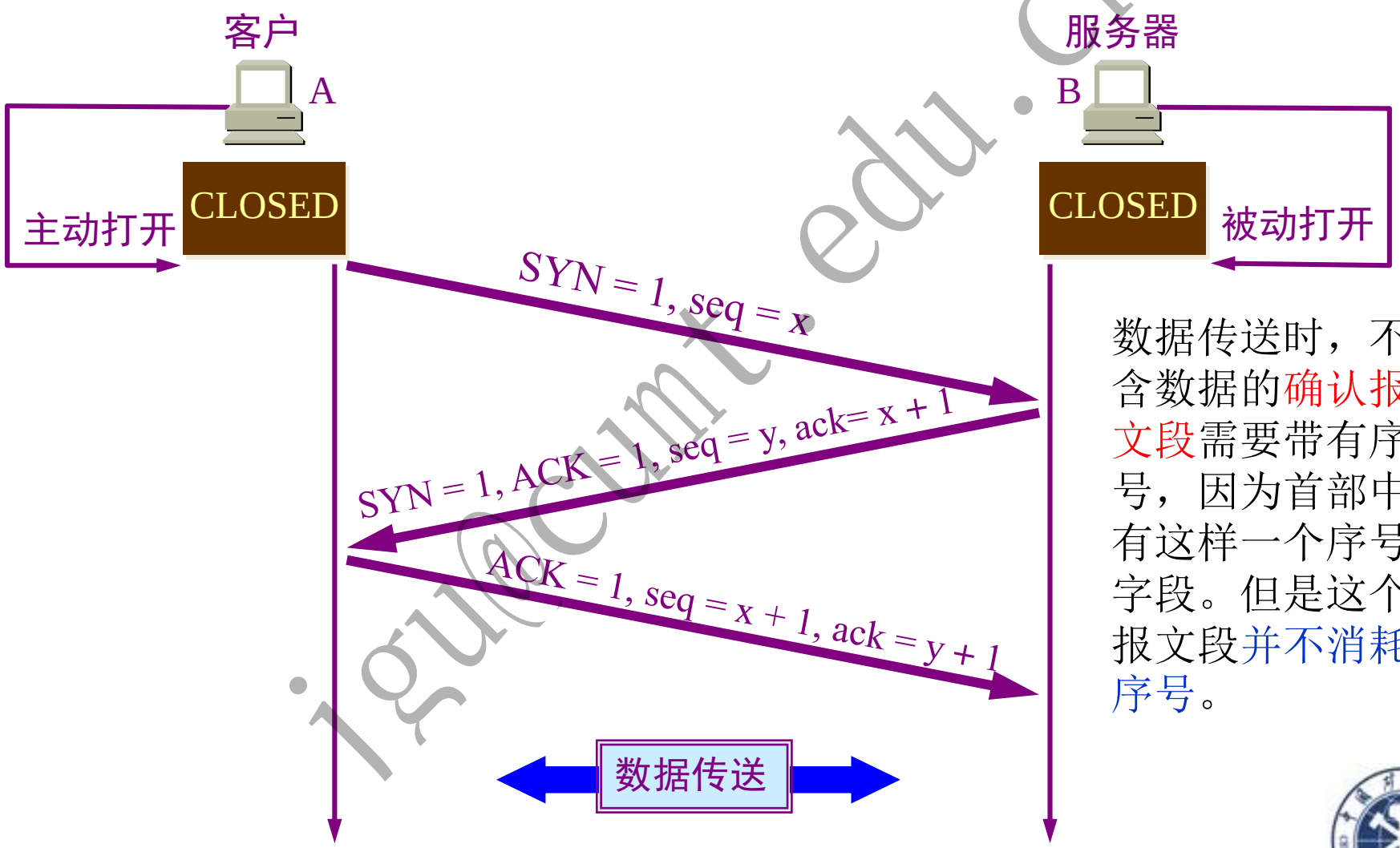
- B 的 TCP 收到连接请求报文段后，如同意，则发回确认。
- B 在确认报文段中应使 SYN = 1，使 ACK = 1，其确认号ack = x + 1，自己选择的序号 seq = y。

- A 收到此报文段后向 B 给出确认，其 ACK = 1，确认号 ack = y + 1。
- A 的 TCP 通知上层应用进程，连接已经建立。



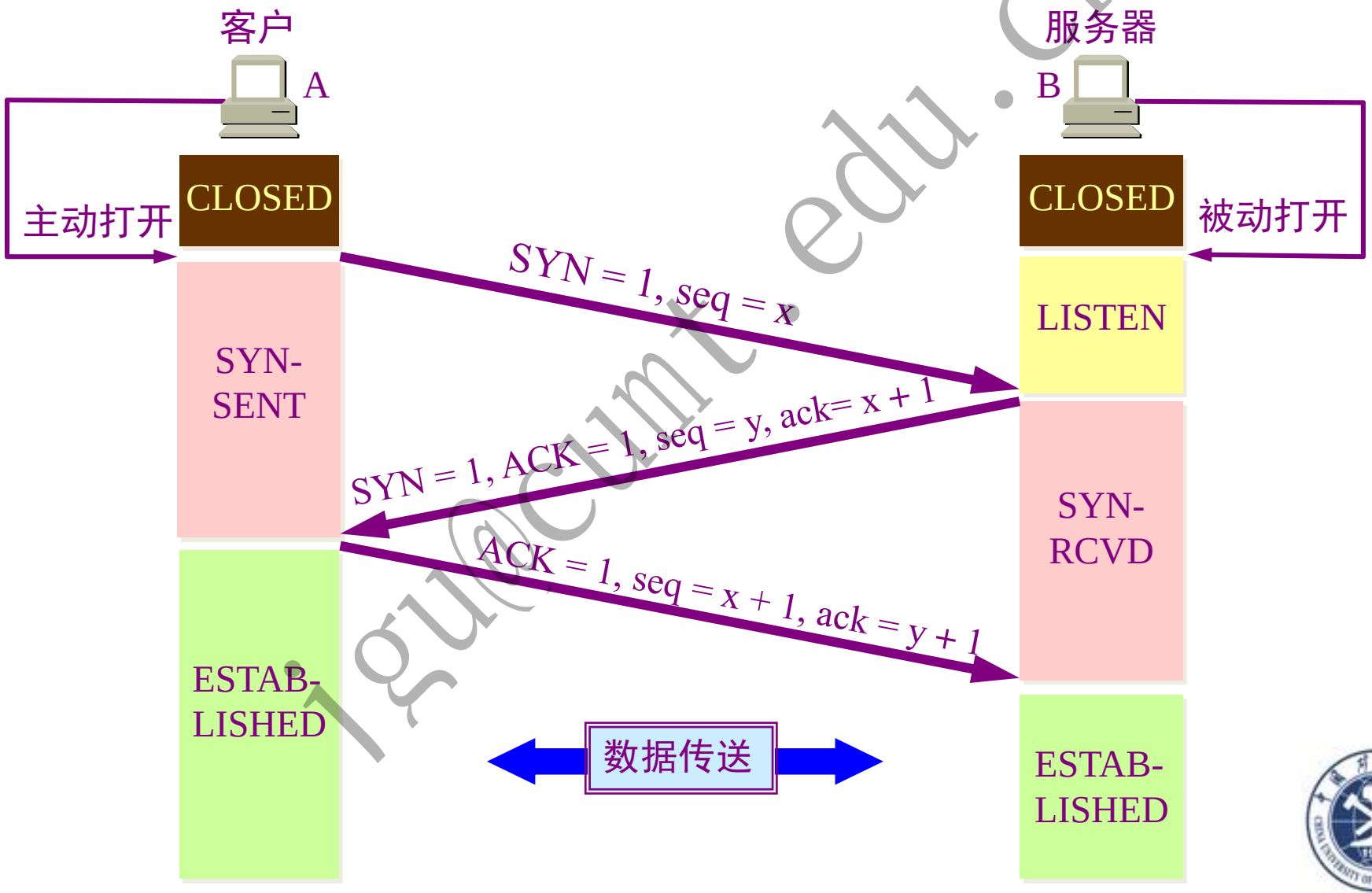
SYN报文段非常重要，是不允许丢失的（传错了或丢失了就要重传，否则无法建立连接），因此虽然这个报文段没有数据部分，还是要消耗一个序号。

- B 的 TCP 收到主机 A 的确认后，也通知其上层应用进程：TCP 连接已经建立。



数据传送时，不含数据的确认报文段需要带有序号，因为首部中有这样一个序号字段。但是这个报文段并不消耗序号。

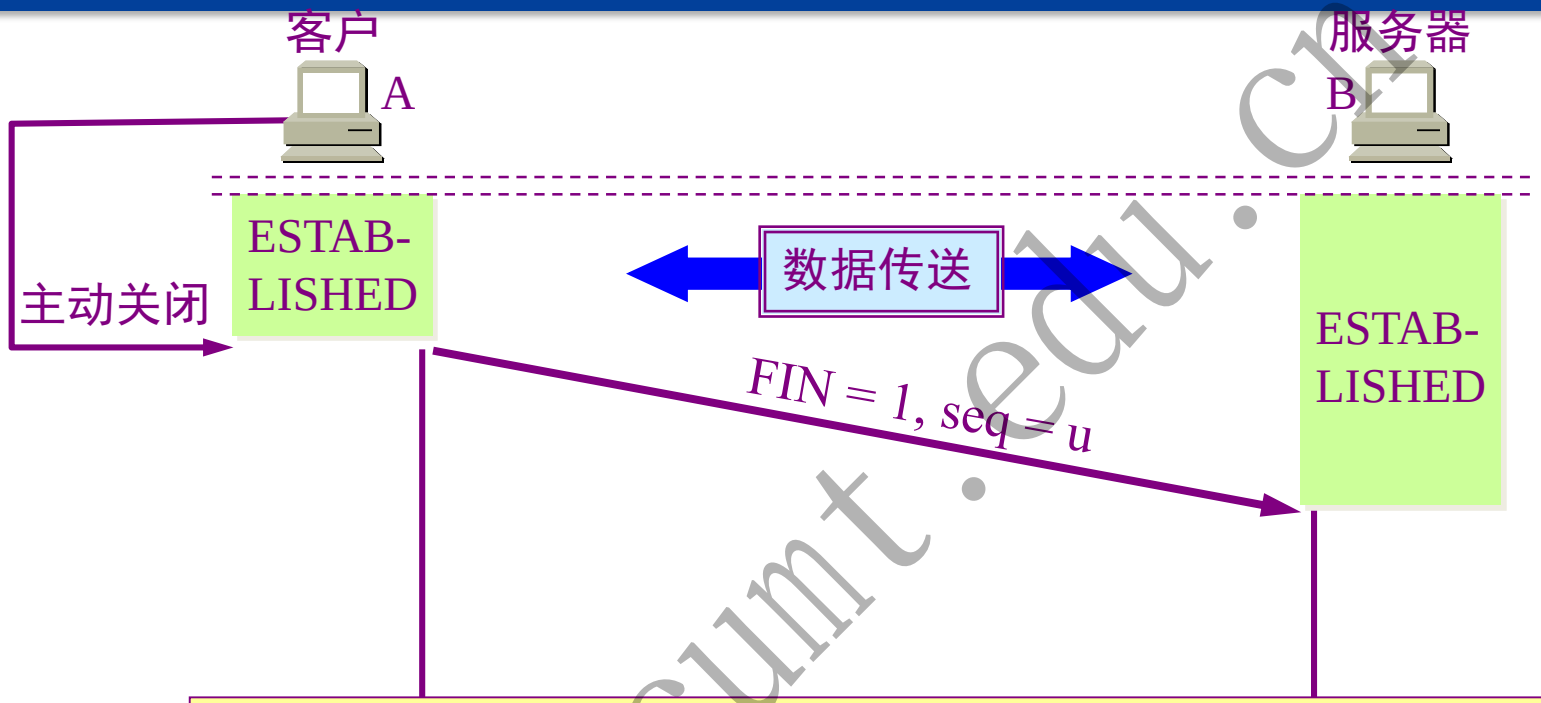
用三次报文握手建立 TCP 连接的各状态



Q17：如何释放TCP连接？

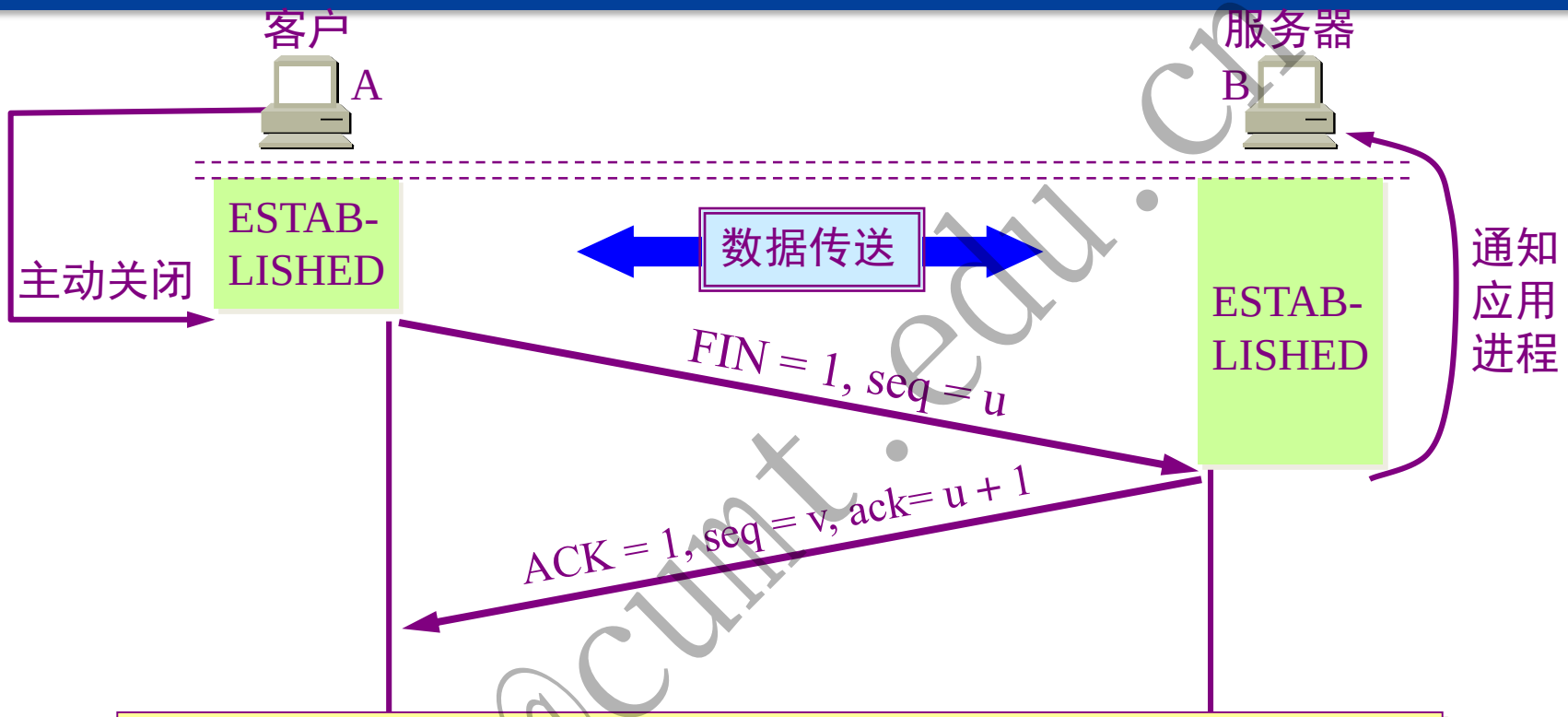
- TCP 连接释放过程比较复杂。
- 数据传输结束后，通信的双方都可释放连接。
- TCP 连接释放过程是四报文握手/挥手。

TCP 的连接释放：采用四报文握手



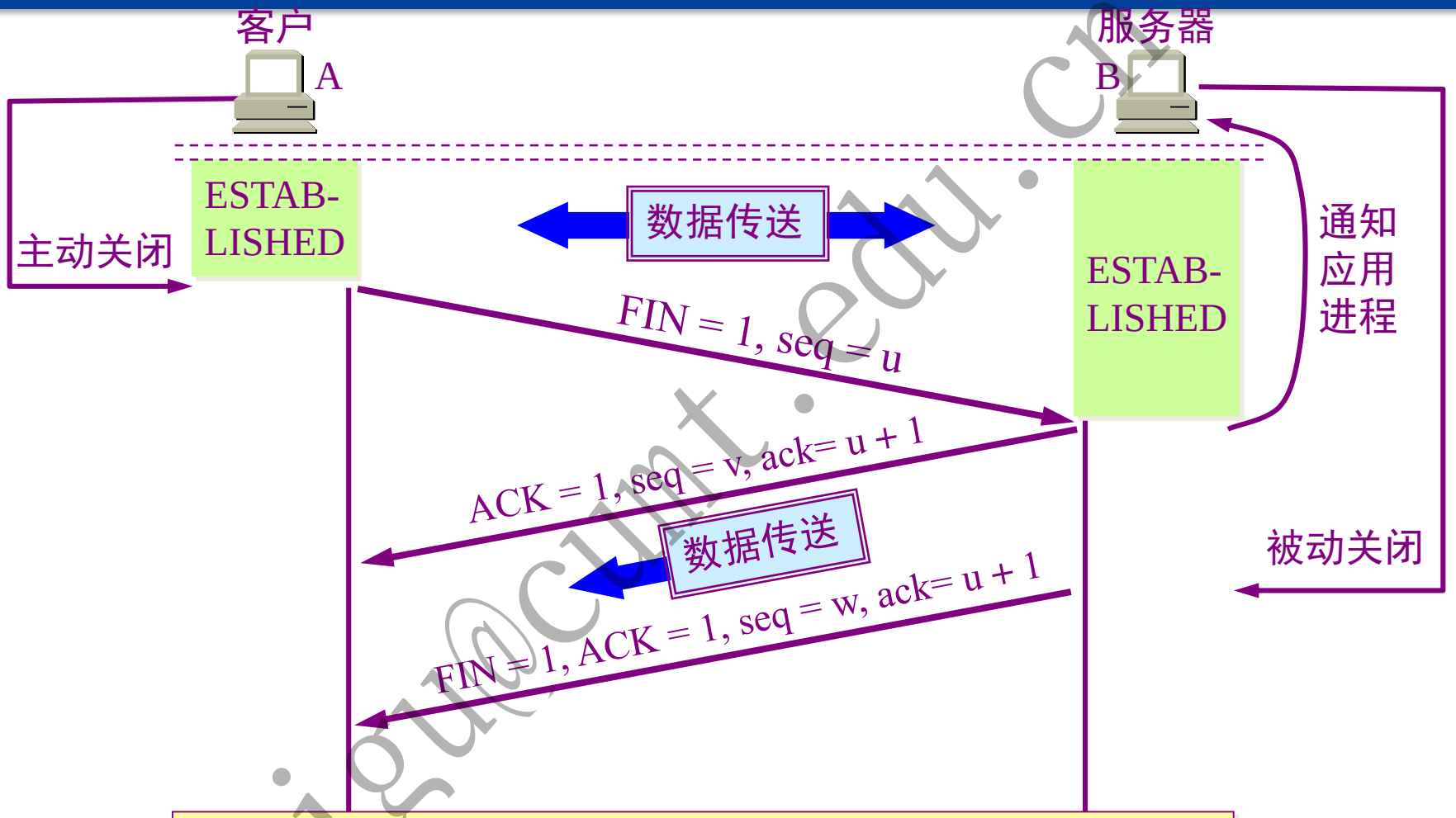
- 数据传输结束后，通信的双方都可释放连接。现在 A 的应用进程先向其 TCP 发出连接释放报文段，并停止再发送数据，主动关闭 TCP 连接。
- A 把连接释放报文段首部的 FIN = 1，其序号 seq = u，等待 B 的确认。

TCP 的连接释放：采用四报文握手



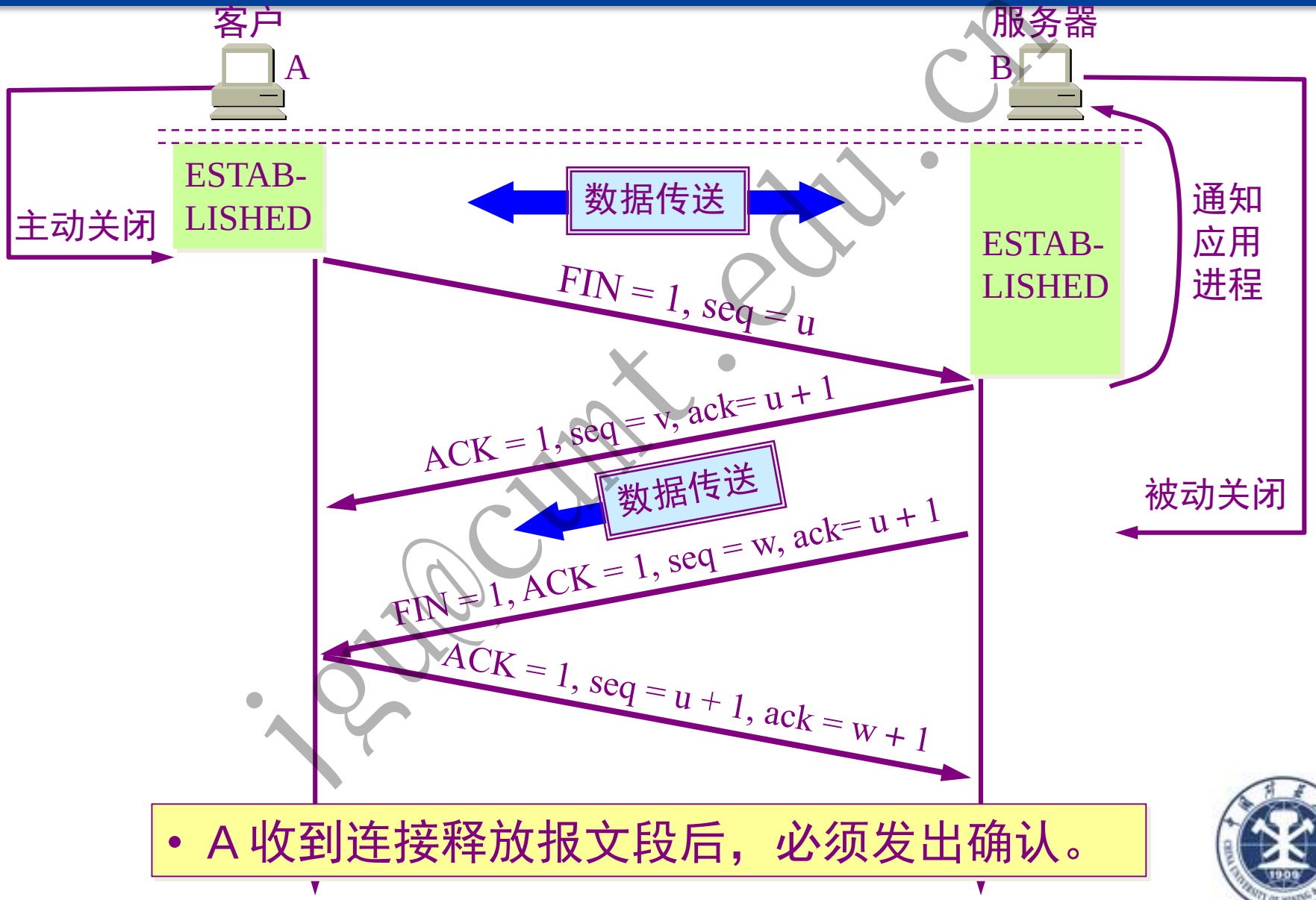
- B 发出确认，确认号 ack = u + 1，而这个报文段自己的序号 seq = v。
- TCP 服务器进程通知高层应用进程。
- 从 A 到 B 这个方向的连接就释放了，TCP 连接处于半关闭状态。B 若发送数据，A 仍要接收。

TCP 的连接释放：采用四报文握手



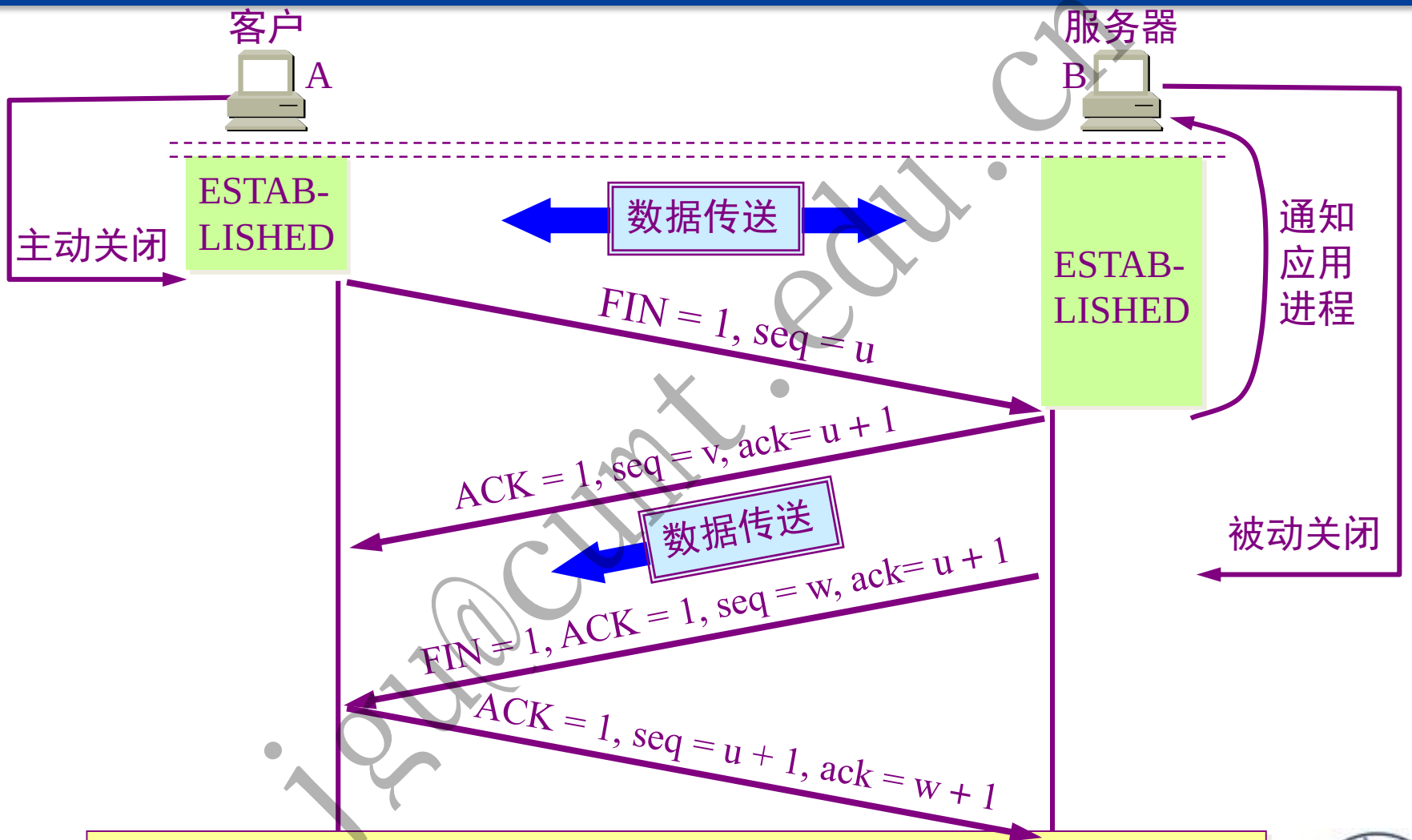
- 若 B 已经没有要向 A 发送的数据，其应用进程就通知 TCP 释放连接。

TCP 的连接释放：采用四报文握手



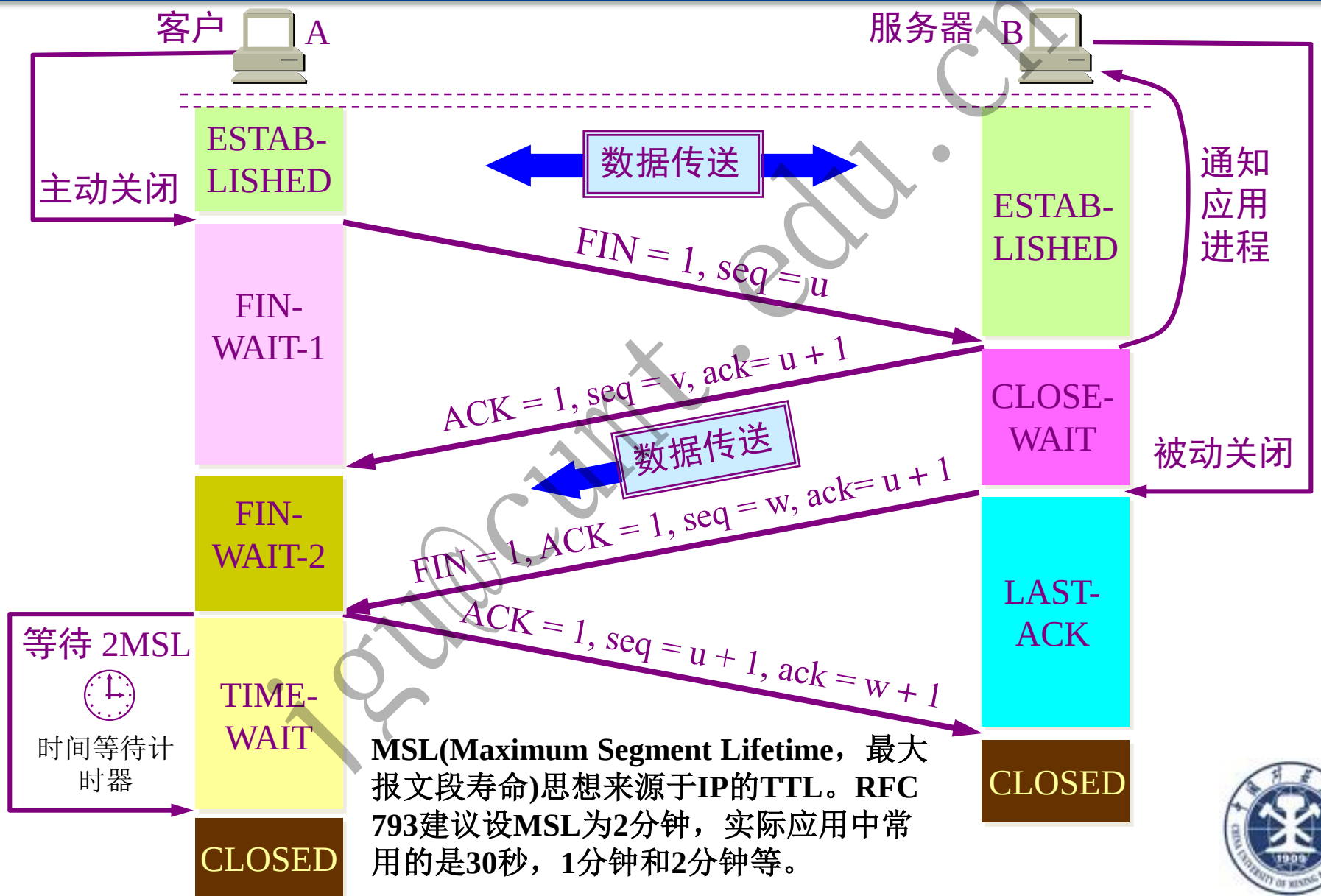
- A 收到连接释放报文段后，必须发出确认。

TCP 的连接释放：采用四报文握手



- 在确认报文段中 ACK = 1，确认号 ack = w + 1，自己的序号 seq = u + 1。

TCP 连接必须经过时间等待计时器(TIME-WAIT timer)设置的时间 2MSL 后，A才进入到CLOSED状态。

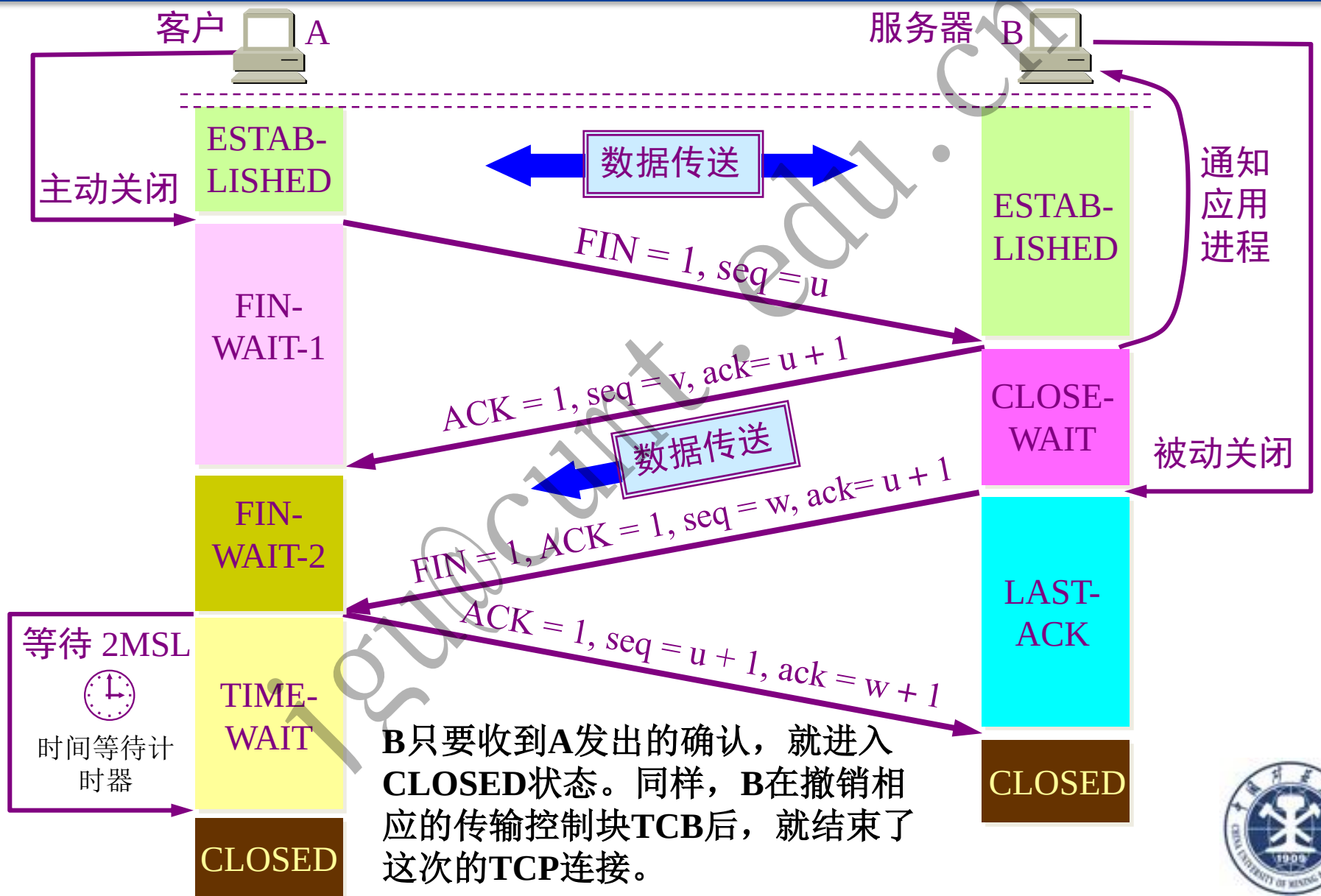


MSL(Maximum Segment Lifetime，最大报文段寿命)思想来源于IP的TTL。RFC 793建议设MSL为2分钟，实际应用中常用的是30秒，1分钟和2分钟等。

A 必须等待 2MSL 的时间

- 第一，为了保证 A 发送的最后一个 ACK 报文段能够到达 B。
  - 这个ACK报文段有可能丢失，使得B超时重传FIN+ACK报文段。此时A就能在2MSL时间内收到这个重传的FIN+ACK报文段，接着A重传一次确认，重启2MSL计时器。最后，A和B都正常进入到CLOSED状态。
  - 如果A在TIME-WAIT状态不等待一段时间，而是在发送完ACK报文段后立即释放连接，B就无法按照正常步骤进入CLOSED状态。
- 第二，防止“已失效的连接请求报文段”出现在本连接中。A 在发送完最后一个 ACK 报文段后，再经过时间 2MSL，就可以使本连接持续的时间内所产生的所有报文段，都从网络中消失。这样就可以使下一个新的连接中不会出现这种旧的连接请求报文段。

TTL与MSL是有关系的但不是简单的相等的关系，MSL要大于等于TTL。



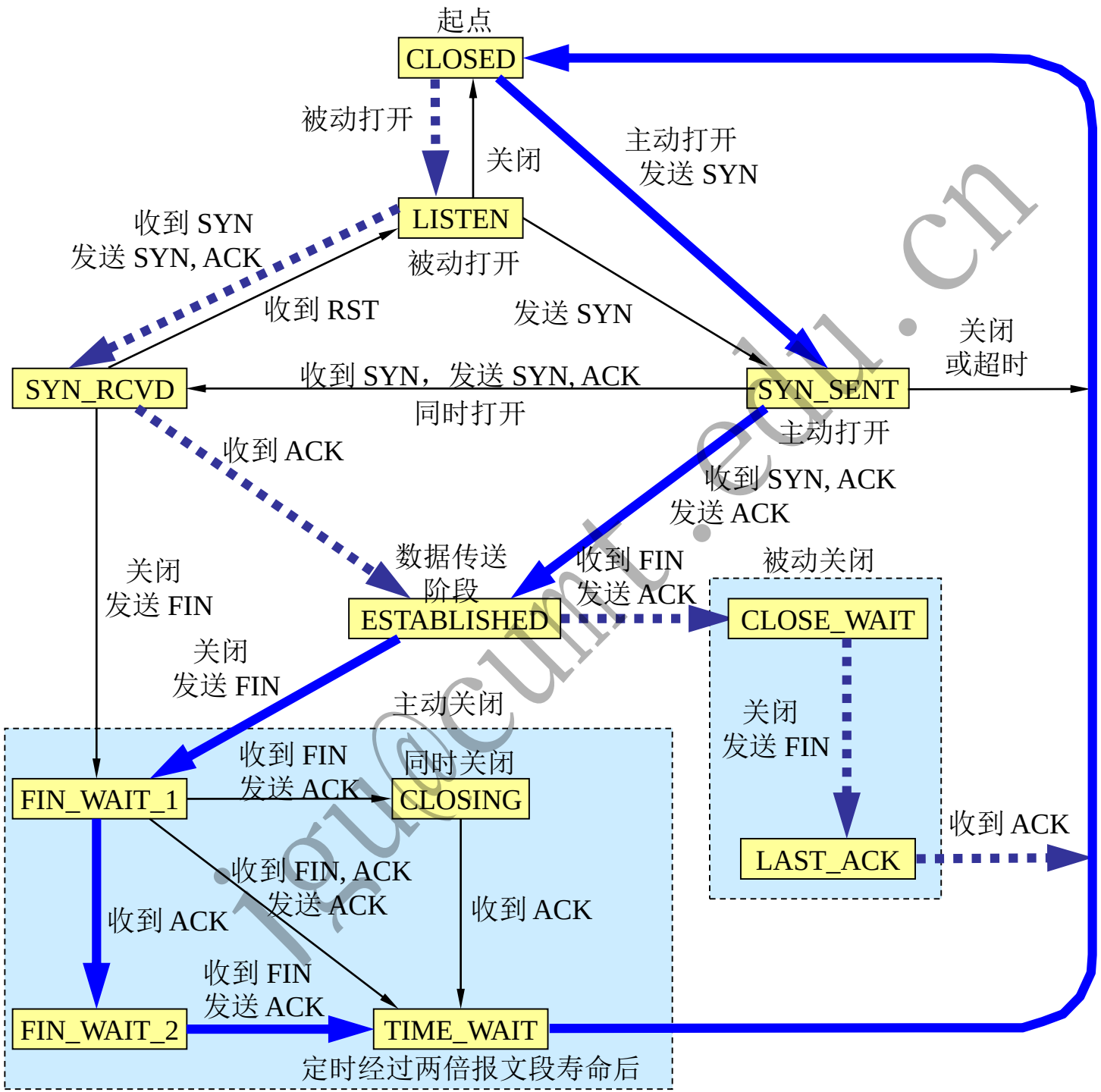
B只要收到A发出的确认，就进入CLOSED状态。同样，B在撤销相应的传输控制块TCB后，就结束了这次的TCP连接。

保活计时器(keepalive timer)

- 应用场景
  - 客户已主动与服务器建立了TCP连接。但后来客户端的主机突然出故障。显然，服务器以后就不能再收到客户发来的数据。因此，应当有措施使服务器不要再白白等待下去。这就是使用保活计时器。
- 使用方法
  - 服务器每收到一次客户的数据，就重新设置保活计时器，时间的设置是2小时。若2小时没有收到客户的数据，服务器就发送一个探测报文段，以后每隔75秒钟发送一次。若一连发送10个探测报文段后仍无客户的响应，服务器就认为客户端出了故障，接着就关闭这个连接。

Q18：TCP的各种状态是如何变迁的?

- TCP 有限状态机的图中每一个方框都是 TCP 可能具有的状态。
  - 每个方框中的大写英文字符串是 TCP 标准所使用的 TCP 连接状态名。状态之间的箭头表示可能发生的状态变迁。
  - 箭头旁边的字，表明引起这种变迁的原因，或表明发生状态变迁后又出现什么动作。
- 图中有三种不同的箭头。
  - 粗实线箭头表示对客户进程的正常变迁。
  - 粗虚线箭头表示对服务器进程的正常变迁。
  - 另一种细线箭头表示异常变迁。



TCP的有限状态机

netstat

netstat命令的功能是显示网络连接、路由表和网络接口信息，可以让用户得知有哪些网络连接正在运行。

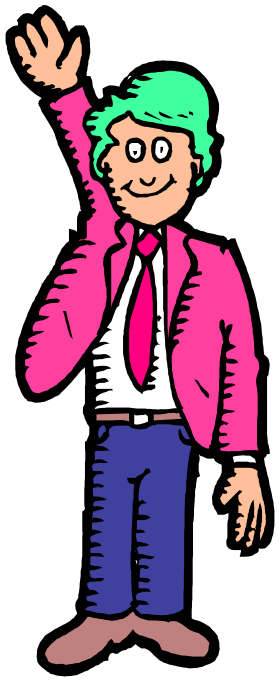
netstat [-a][-e][-n][-o][-p Protocol][-r][-s][Interval]

```
C:\Users\Petri>netstat

活动连接

  协议  本地地址          外部地址        状态
  TCP    127.0.0.1:1027         DESKTOP-BHQACRS:1029   ESTABLISHED
  TCP    127.0.0.1:1028         DESKTOP-BHQACRS:1030   ESTABLISHED
  TCP    127.0.0.1:1029         DESKTOP-BHQACRS:1027   ESTABLISHED
  TCP    127.0.0.1:1030         DESKTOP-BHQACRS:1028   ESTABLISHED
  TCP    127.0.0.1:1031         DESKTOP-BHQACRS:activesync  ESTABLISHED
  TCP    127.0.0.1:1032         DESKTOP-BHQACRS:1033   ESTABLISHED
  TCP    127.0.0.1:1033         DESKTOP-BHQACRS:1032   ESTABLISHED
  TCP    127.0.0.1:1034         DESKTOP-BHQACRS:1031   ESTABLISHED
  TCP    192.168.0.103:1166     180.163.255.156:https  ESTABLISHED
  TCP    192.168.0.103:3927     140.206.78.3:http      ESTABLISHED
  TCP    192.168.0.103:3939     121.51.73.100:https    ESTABLISHED
  TCP    192.168.0.103:3954     121.51.36.198:http     CLOSE_WAIT
  TCP    192.168.0.103:3956     223.167.166.58:http    ESTABLISHED
  TCP    192.168.0.103:4181     182.254.76.104:https   CLOSE_WAIT
  TCP    192.168.0.103:4193     140.206.78.31:http     ESTABLISHED
  TCP    192.168.0.103:4195     101.4.60.77:https      ESTABLISHED
  TCP    192.168.0.103:4204     180.163.255.156:https  ESTABLISHED
  TCP    192.168.0.103:4206     180.163.255.156:https  ESTABLISHED
  TCP    192.168.0.103:4232     180.163.255.156:https  ESTABLISHED
  TCP    192.168.0.103:4298     180.163.255.159:https  ESTABLISHED
  TCP    192.168.0.103:5835     203.208.40.57:http     ESTABLISHED
  TCP    192.168.0.103:13675    47.95.47.253:https     ESTABLISHED
```

Netstat用于显示与IP、TCP、UDP和ICMP协议相关的统计数据，一般用于检验本机各端口的网络连接情况。



THANK YOU!