第 3 章　端到端的传输

物理链路提供了设备之间的连接，而终端之间的传输建立在相应层次的传输协议上。如果通信双方通过一条直接相连的通信链路形成端到端的传输，那么需要采用数据链路层协议来解决数据链路的传输问题；如果通信双方通过多条链路形成端到端的传输，那么需要采用网络层协议来建立路径实现端到端的有效传输；当通信双方处于运输层时，此时的端到端通信需要采用运输层协议来实现可靠传输。

3.1　端到端的传输特性

3.1.1　随机过程的基本概念

随机过程是随机变量概念在时域上的延伸。直观地讲，随机过程是时间参数 t 的函数的集合，在任意观察时刻，随机过程的取值是一个随机变量。或者说，依赖于时间参数 t 的随机变量所构成的总体称为随机过程。随机过程用来描述在一个观察区间内某一实体的随机行为，例如，通信系统中的噪声就是一个典型的随机过程，在某一时间区间内观察到的该随机过程取值的一个时间函数称为随机过程的一个样本函数。通过获取足够多的样本函数，就可以得到随机过程的统计特性。

设 $X(t)$ 是一个随机过程（其样本函数示意图如图 3-1 所示），可以从两个方面来描述 $X(t)$ 的特征：一是在任意时刻 t_1 随机变量 $X(t_1)$ 的统计特征，如一维分布函数、概率密度函数、均值和方差等；二是同一随机过程在不同时刻 t_1 和 t_2 对应的随机变量 $X(t_1)$ 和 $X(t_2)$ 的相关特性，如多维联合分布函数、相关函数、协方差矩阵等。随机过程 $X(t)$ 的一维分布函数定义为

$$F_t(x) = P\{X(t) < x\} \tag{3-1}$$

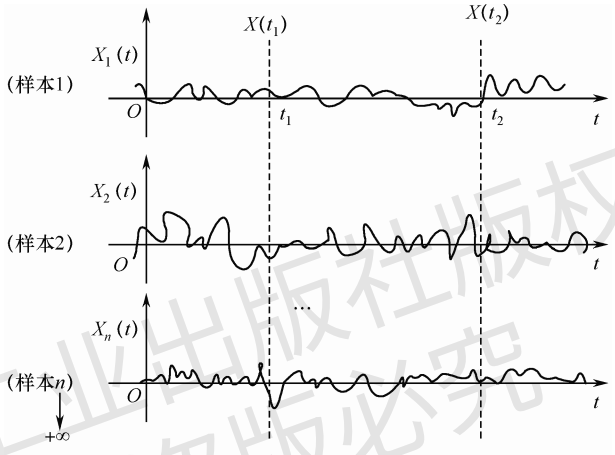


图 3-1　随机过程的样本函数示意图

若 $\int_{-\infty}^{+\infty}|X|\mathrm{d}F_t(x)<+\infty$，则随机过程 $X(t)$ 的均值函数为

$$m_{\mathrm{X}}(t)=E[X(t)]=\int_{-\infty}^{+\infty}x\mathrm{d}F_t(x) \tag{3-2}$$

对任意时刻 t_1 和 t_2，若函数

$$C_{\mathrm{X}}(t_1,t_2)=\mathrm{Cov}[X(t_1),X(t_2)]=E\{[X(t_1)-m_{\mathrm{X}}(t_1)][X(t_2)-m_{\mathrm{X}}(t_2)]\} \tag{3-3}$$

存在，则称 $C_{\mathrm{X}}(t_1,t_2)$ 为 $X(t)$ 的协方差函数。$X(t)$ 的方差函数为

$$D_{\mathrm{X}}(t)=D[X(t)]=E[(X(t)-m_{\mathrm{X}}(t))^2] \tag{3-4}$$

若对任意给定的时间 t_1 和 t_2，$R_{\mathrm{X}}(t_1,t_2)=E[X(t_1)X(t_2)]$ 存在，则称 $R_{\mathrm{X}}(t_1,t_2)$ 为 $X(t)$ 的自相关函数。协方差函数、自相关函数、均值函数有下列关系

$$C_{\mathrm{X}}(t_1,t_2)=R_{\mathrm{X}}(t_1,t_2)-m_{\mathrm{X}}(t_1)m_{\mathrm{X}}(t_2) \tag{3-5}$$

下面介绍几类典型的随机过程。

1．独立随机过程

设有一个随机过程 $X(t)$，若对任意给定的时刻 $t_1,t_2,\cdots,t_n$，随机变量 $X(t_1),X(t_2),\cdots,X(t_n)$ 是相互独立的，也就是其 n 维分布函数可以表示为

$$F_{t_1,t_2,\cdots,t_n}(x_1,x_2,\cdots,x_n)=\prod_{i=1}^{n}F_{t_i}(x_i) \tag{3-6}$$

则称 $X(t)$ 是独立随机过程。该随机过程的特点是任意时刻的状态与其他任何时刻的状态无关。

2．马尔可夫（Markov）过程

设有一个随机过程 $X(t)$，若对于一个任意的时间序列 $t_1<t_2<\cdots<t_n$（$n\geqslant 3$），在给定随机变量 $X(t_1)=x_1,X(t_2)=x_2,\cdots,X(t_{n-1})=x_{n-1}$ 的条件下，$X(t_n)=x_n$ 的分布可以表示为

$$F_{t_n,t_1,t_2,\cdots,t_{n-1}}(x_n\mid x_1,x_2,\cdots,x_{n-1})=F_{t_n,t_{n-1}}(x_n\mid x_{n-1}) \tag{3-7}$$

则称 $X(t)$ 为马尔可夫过程或简称为马氏过程。该过程的基本特点是无后效性，即当该过程在 t_0 时刻的状态为已知条件时，该过程在 $t(t>t_0)$ 时刻所处的状态与该过程在 t_0 时刻之前的状态无关。

3．独立增量过程

设 $X(t_2)-X(t_1)=X(t_1,t_2)$ 是随机过程 $X(t)$ 在时间间隔 (t_1,t_2) 上的增量，若对于时间 t 的任意 n 个值 $0\leqslant t_1<t_2<\cdots<t_n$，增量 $X(t_1,t_2),X(t_2,t_3),\cdots,X(t_{n-1},t_n)$ 是相互独立的，则称 $X(t)$ 为独立增量过程。该过程的特点是：在任意时间间隔上过程状态的改变并不影响未来任意时间间隔上过程状态的改变。可以证明独立增量过程是一种特殊的马尔可夫过程。

4．平稳随机过程

若对于时间 t 的任意 n 个值 $t_1,t_2,\cdots,t_n$ 和任意实数 ε，随机过程 $X(t)$ 的 n 维分布函数满足关系式

$$F_{t_1,t_2,\cdots,t_n}(x_1,x_2,\cdots,x_n)=F_{t_{1+\varepsilon},t_{2+\varepsilon},\cdots,t_{n+\varepsilon}}(x_1,x_2,\cdots,x_n) \tag{3-8}$$

则称 $X(t)$ 为平稳随机过程或简称为平稳过程，该过程的特点是随机过程的统计特性不随时间的平移而变化，因此又称为严（狭义）平稳过程。在实际应用中更关心这样一类过程：其 $E[|X(t)|^2]<+\infty$（称为二阶矩过程）且满足下列条件：（1）均值为常量（与时间 t 无关）；（2）对于任意时刻 s 和 t，其相关函数满足 $R_X(s,t)=R_X(t-s)$，即相关函数仅与时间差 $t-s$ 有关，而与 s、t 的取值无关，称这类过程为宽（广义）平稳过程。实际应用中的随机过程通常是宽平稳过程。

平稳过程一个重要的特征就是是否具有各态历经性。为了说明各态历经性，在时间轴上定义下列两种平均

$$\langle X(t)\rangle=\lim_{T\to+\infty}\int_{-T}^{T}X(t)\mathrm{d}t \tag{3-9}$$

$$\langle X(t)X(t+\tau)\rangle=\lim_{T\to+\infty}\int_{-T}^{T}X(t)X(t+\tau)\mathrm{d}t \tag{3-10}$$

为随机过程 $X(t)$ 的均值和自相关函数。若 $\langle X(t)\rangle=E[X(t)]=m_X$ 依概率 1 成立（对所有样本都成立），则称随机过程 $X(t)$ 的均值具有各态历经性；若 $\langle X(t)X(t+\tau)\rangle\geqslant E[X(t)X(t+\tau)]=R_X(\tau)$ 依概率 1 成立，则称随机过程 $X(t)$ 的自相关函数具有各态历经性；若 $X(t)$ 的均值和自相关函数都具有各态历经性，则称 $X(t)$ 是（宽）各态历经过程，或者说 $X(t)$ 是各态历经的。

3.1.2 Poisson 过程

如果在日常生活中观察顾客进入商店、银行或其他公共服务场所的过程，会发现若把一位顾客的到达视为一个“随机点”，则这是一个源源不断出现随机点的过程。在这一过程中，任意一段时间内到达的顾客数也是随机的。这类描述到达顾客数及其特征的过程通常称为计数过程。一个交换局中电话呼叫到达（人们在拨打电话的行为中拿起电话听筒并拨出对方号码的动作称为一次电话呼叫到达）的过程也具有类似的特征。

设一个随机过程为 $\{A(t),t\geqslant 0\}$，$A(t)$ 的取值为非负整数，若该随机过程满足下列条件，则称该过程为到达率为 λ 的 Poisson（泊松）过程。

（1）$A(t)$ 是一个计数过程，它表示在 $[0,t)$ 区间内到达的用户总数，$A(0)=0$，$A(t)$ 的状态空间为 $\{0,1,2,\cdots\}$。Poisson 过程示意图如图 3-2 所示。对于任意两个时刻 s 和 t，且 $s<t$，$A(t)-A(s)$ 即为 $[s,t)$ 之间到达的用户总数。

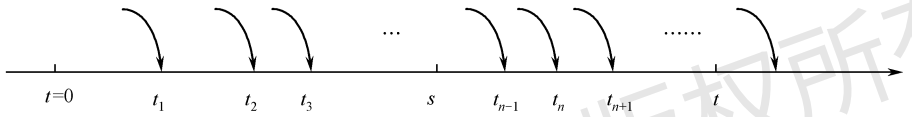


图 3-2 Poisson 过程示意图

（2）$A(t)$ 是一个独立增量过程，即在两个不同时间区间（区间不重叠）内到达的用户数是相互独立的。

（3）在任意一个长度为 τ 的区间内，到达的用户数服从参数为 $\lambda\tau$ 的 Poisson 分布，即

$$P[A(t+\tau)-A(t)=n]=\frac{(\tau\lambda)^n}{n!}\mathrm{e}^{-\tau\lambda},\quad n=0,1,2,\cdots \tag{3-11}$$

其均值和方差均为 $\lambda\tau$。由于在 τ 区间内平均到达的用户数为 $\lambda\tau$，因此 λ 即为单位时间内平均到达的用户数，可将其称为到达率。

Poisson 过程有如下基本特征。

（1）到达间隔 $\tau_n = t_{n+1} - t_n$ 相互独立且服从指数分布，概率密度函数为

$$p(\tau_n) = \lambda \mathrm{e}^{-\lambda\tau_n} \tag{3-12}$$

其分布函数为

$$P(\tau_n < s) = 1 - \mathrm{e}^{-\lambda s},\ s \geqslant 0 \tag{3-13}$$

该特征说明 Poisson 过程的到达间隔服从指数分布。相反，若一个计数过程的到达间隔序列是相互独立同分布且参数为 λ 的指数分布，则该过程是到达率为 λ 的 Poisson 过程。因此，说用户到达过程是到达率为 λ 的 Poisson 过程，与说用户到达间隔相互独立且服从参数为 λ 的指数分布是等价的。

（2）对于一个任意小的区间 $\sigma \geqslant 0$，在一个充分小的时间间隔内没有用户到达的概率为 $1-\lambda\delta$；在一个充分小的时间间隔内，有一个用户到达的概率为 $\lambda\delta$；在一个充分小的时间间隔内，有两个或两个以上用户到达是几乎不可能的。

（3）多个相互独立的 Poisson 过程 A_i 之和 $A = A_1 + A_2 + \cdots + A_k$ 仍是一个 Poisson 过程，其到达率为 $\lambda = \lambda_1 + \lambda_2 + \cdots + \lambda_k$，式中，$\lambda_k$ 是 Poisson 过程 A_k 的到达率。

（4）如果将一个 Poisson 过程的到达以概率 p 和 $1-p$ 独立地分配给两个子过程，那么这两个子过程也是 Poisson 过程。注意，这里是将到达独立地进行分配。如果把到达交替地分配给两个子过程，即两个子过程分别由奇数号到达和偶数号到达组成，那么这两个子过程不是 Poisson 过程。

【例 3-1】　有红、绿、蓝三种颜色的汽车，分别以强度为 λ_{R}、λ_{G}、λ_{B} 的 Poisson 流到达某哨卡，设它们是相互独立的。把汽车合并成单个输出过程（假设汽车长度为 0）。

（1）求两辆汽车之间的时间间隔的概率密度函数。

（2）在 t_0 时刻观察到一辆红色汽车，求下一辆汽车将是：（a）红色的；（b）蓝色的；（c）非红色的概率。

（3）在 t_0 时刻观察到一辆红色汽车，求下三辆汽车是红色的，然后又一辆非红色汽车到达的概率。

解　（1）由于独立的 Poisson 过程之和仍为 Poisson 过程，且其强度为 $\lambda_{\mathrm{C}} = \lambda_{\mathrm{R}} + \lambda_{\mathrm{G}} + \lambda_{\mathrm{B}}$，因此设 Z_{C} 为两辆汽车到达的时间间隔，则其概率密度函数为

$$p_{Z_{\mathrm{C}}}(z) = \begin{cases} \lambda_{\mathrm{C}} \mathrm{e}^{-\lambda_{\mathrm{C}} z}, z \geqslant 0 \\ 0, \qquad z < 0 \end{cases}$$

（2）设 Z_{R}、Z_{G}、Z_{B} 分别为两辆红色、绿色、蓝色汽车到达的时间间隔，Z_{X} 为红色与非红色汽车到达的时间间隔。由于红色汽车的到达与非红色汽车的到达相互独立，因此，Z_{X} 仅与非红色汽车到达间隔有关。非红色汽车是绿色和蓝色两种汽车的复合流，因此 Z_{X} 的概率密度函数为

$$p_{Z_{\mathrm{X}}}(z) = \begin{cases} (\lambda_{\mathrm{B}} + \lambda_{\mathrm{G}}) \mathrm{e}^{-(\lambda_{\mathrm{B}} + \lambda_{\mathrm{G}}) z}, z \geqslant 0 \\ 0, \qquad\qquad\qquad z < 0 \end{cases}$$

由于 Z_{X} 与 Z_{R} 相互独立，因此下一辆是红色汽车的概率为

$$P\{\text{下一辆是红色汽车}\}=P\{Z_{\mathrm{R}}<Z_{\mathrm{X}}\}=\int_0^{+\infty}\lambda_{\mathrm{R}}\mathrm{e}^{-\lambda_{\mathrm{R}}Z_{\mathrm{R}}}\mathrm{d}z_{\mathrm{X}}=\frac{\lambda_{\mathrm{R}}}{\lambda_{\mathrm{R}}+\lambda_{\mathrm{X}}}=\frac{\lambda_{\mathrm{R}}}{\lambda_{\mathrm{R}}+\lambda_{\mathrm{G}}+\lambda_{\mathrm{B}}}$$

令 Z_{Y} 是从 t_0 算起的非蓝色汽车的到达时刻，则同理可得

$$P\{\text{下一辆是蓝色汽车}\}=P\{Z_{\mathrm{B}}<Z_{\mathrm{Y}}\}=\frac{\lambda_{\mathrm{B}}}{\lambda_{\mathrm{R}}+\lambda_{\mathrm{G}}+\lambda_{\mathrm{B}}}$$

$$P\{\text{下一辆是非红色汽车}\}=1-\frac{\lambda_{\mathrm{R}}}{\lambda_{\mathrm{R}}+\lambda_{\mathrm{G}}+\lambda_{\mathrm{B}}}=\frac{\lambda_{\mathrm{G}}+\lambda_{\mathrm{B}}}{\lambda_{\mathrm{R}}+\lambda_{\mathrm{G}}+\lambda_{\mathrm{B}}}$$

（3）下三辆汽车是红色的，然后是一辆非红色汽车的概率为

$$P\left(\frac{\lambda_{\mathrm{R}}}{\lambda_{\mathrm{R}}+\lambda_{\mathrm{G}}+\lambda_{\mathrm{B}}}\right)^3=\frac{\lambda_{\mathrm{G}}+\lambda_{\mathrm{B}}}{\lambda_{\mathrm{R}}+\lambda_{\mathrm{G}}+\lambda_{\mathrm{B}}}$$

3.1.3 马尔可夫链

马尔可夫（Markov）链（简称马氏链）是最简单的马氏过程——时间和状态过程的取值参数都是离散的马氏过程。将可列个发生状态转移（变化）的时刻记为 $t_1,t_2,\cdots,t_n,\cdots$，在 t_n 时刻发生的转移称为第 n 次转移，并且假定在每个时刻 $t_n(n=1,2,\cdots)$，$X_n=X(t_n)$ 所有可能的状态的集合 S 是可数的，即可表示为 $S=\{0,1,2,\cdots\}$。对应于时间序列 $t_1,t_2,\cdots,t_n,\cdots$，马氏链的状态序列为 $i_1,i_2,\cdots,i_n,\cdots$，这时有

$$P\{X_n=i_n\mid X_{n-1}=i_{n-1},\cdots,X_1=i_1\}=P(X_n=i_n\mid X_{n-1}=i_{n-1}) \tag{3-14}$$

式（3-14）表示在给定第 $n-1$ 时刻的状态，即在 $X_{n-1}=i_{n-1}$ 的条件下，第 n 次转移出现 i_n 的概率。若该概率与 n 无关（与转移无关，仅与转移前后的状态有关），则该马氏链为齐次马氏链，否则称为非齐次马氏链，本书仅讨论齐次马氏链。此时式（3-14）可以表示为

$$P_{ij}=P\{X_n=j\mid X_{n-1}=i\} \tag{3-15}$$

式（3-15）称为马氏链的（一步）转移概率。P_{ij} 满足下列条件

$$P_{ij}\geqslant 0,\quad \sum_{j=0}^{+\infty}P_{ij}=1,\quad i=0,1,\cdots \tag{3-16}$$

相应的转移概率矩阵可以表示为

$$\boldsymbol{P}=\begin{bmatrix}P_{00} & P_{01} & P_{02} & \cdots\\ P_{10} & P_{11} & P_{12} & \cdots\\ \vdots & \vdots & \vdots & \ddots\\ P_{i0} & P_{i1} & P_{i2} & \cdots\\ \vdots & \vdots & \vdots & \vdots\end{bmatrix} \tag{3-17}$$

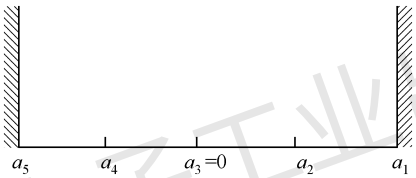


图 3-3 随机游走过程

【例 3-2】 设一盲人（可视为一个随机点）在如图 3-3 所示的线段上游走，其步长为 $t\to+\infty$。假定他只能在 $a_1=2l$、$a_2=l$、$a_3=0$、$a_4=-l$、$a_5=-2l$ 这 5 个点上停留，且只在 $t=1,2,\cdots$ 时刻发生游走。游走的规则是：如果游走前他在 a_2、a_3、a_4 上，那么就分别以 1/2 的概率向

左或向右游走一步；如果游走前他在 $a_1(a_5)$ 上，那么就以概率 1 游走到 $a_2(a_4)$ 上。

以 $X_n = a_i$（$i = 1,2,3,4,5$）表示在时刻 $t = n$ 盲人位于 a_i 处，则容易看出 $X_1, X_2, \cdots$ 是一个马氏链，且他游走的转移概率矩阵为

$$\boldsymbol{P} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

如果用图来表示这一转移过程，那么如图 3-4 所示。图中的圆圈表示马氏过程的状态，图中带箭头的弧线表示状态的转移，弧线上的数字表示一步转移概率。该图称为马氏过程的状态转移图。

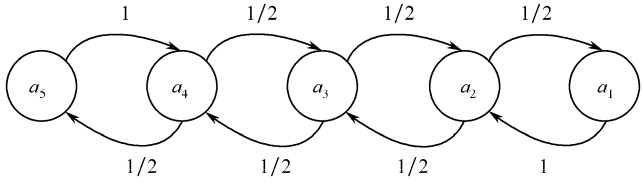


图 3-4　状态转移图

在考察马氏过程时经常会用到 n 步转移概率，即

$$P_{ij}^{n} = P\{X_{n+m} = j \mid X_m = i\} \tag{3-18}$$

它表示当前（第 m 步）的状态为 i，经过 n 步转移后（第 $n+m$ 步）系统的状态为 j 的概率。系统中 $n = m_1 + m_2$ 步状态转移概率可用下式来求解（如图 3-5 所示）

$$P_{ij}^{m_1+m_2} = \sum_{k=0}^{+\infty} P_{ik}^{m_1} P_{kj}^{m_2} \quad n, m \geqslant 0,\ i, j \geqslant 0 \tag{3-19}$$

该公式称为 Chapman-Kolmogorov 等式。

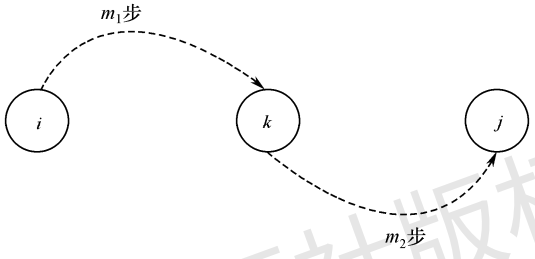


图 3-5　$n=m_1+m_2$ 步状态转移示意图

根据 n 步转移概率可以来定义马氏链状态转移的特性。如果马氏链的两个状态 i 和 j 有下列特性：存在整数 n 和 n'，有 $P_{ij}^{n} > 0$ 及 $P_{ij}^{n'} > 0$（也就是从状态 $i(j)$ 经过 $n(n')$ 步转移到状态 $j(i)$ 的概率大于 0），那么称 i 和 j 是互通的。如果马氏链的所有状态都是互通的，那么该马氏链是不可约的（irreducible）。

如果马氏链的状态 i 有下列特性：存在某个整数 $m \geqslant 1$，使 $P_u^m > 0$，并且存在某个整数 $d > 1$，仅当 m 为 d 的整数倍时有 $P_{ii}^m > 0$，那么状态 i 是有周期性的。如果马氏链中没有一个状态是有周期性的，那么称该马氏链为非周期的。本书仅考虑非周期不可约的马氏链。

对于马氏链，其状态的稳态概率定义为：如果有下式成立

$$P_j = \sum_{i=0}^{+\infty} P_i P_{ij} \quad j = 0,1,\cdots \tag{3-20}$$

那么称概率分布 $\{P_j \mid j \geqslant 0\}$ 是马氏链的稳态概率分布。对于概率分布，若有稳态概率，则反映了系统达到稳态后处于某一状态的可能性（概率）。

稳态概率分布可以表示为

$$P_j = \lim_{n \to +\infty} P\left\{X_n = j \middle| X_0 = i\right\} \quad i = 0,1,\cdots \tag{3-21}$$

即过程从初始状态 $X_0 = i$ 出发，最终转移到状态 $X_n = j$ 的概率。显然，P_j 与初始状态 $X_0 = i$ 无关。稳态概率分布也可以表示为（以概率 1 成立）

$$P_j = \lim_{n \to +\infty} \frac{\text{前}k\text{次转移中访问状态}j\text{的次数}}{k} \tag{3-22}$$

因此 P_j 表示该过程中访问状态 j 的时间比例或频率，且该频率与初始状态无关。

$$P_j \sum_{i=0}^{+\infty} P_{ji} = \sum_{i=0}^{+\infty} P_i P_{ij} \quad j = 0,1,\cdots \tag{3-23}$$

该式称为全局平衡方程（Global Balance Equation），它表示在稳态情况下，从一个状态 j 转移出去的频率［式（3-23）的左边］等于转移进入状态 j 的频率［式（3-23）的右边］。该方程提供了一种典型的求解稳态概率分布的方法。

【例 3-2（续）】 例 3-2 中已知各种状态的转移概率矩阵，设状态 $a_5, \cdots, a_1$ 的稳态概率为 P_5, P_4, P_3, P_2 和 P_1，则根据式（3-20）可得稳态概率的方程为

$$\begin{bmatrix} P_5 & P_4 & P_3 & P_2 & P_1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} P_5 & P_4 & P_3 & P_2 & P_1 \end{bmatrix} \tag{3-24}$$

由于 $\{P_i \mid i = 1,2,\cdots,5\}$ 是稳态概率分布，因此有

$$\sum_{i=1}^{5} P_i = 1 \tag{3-25}$$

求解由式（3-24）和式（3-25）组成的方程组，得稳态概率分布为

$$(P_5, P_4, P_3, P_2, P_1) = \left(\frac{1}{8}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{8}\right) \tag{3-26}$$

式（3-23）是对一个状态而言的，它反映的是从一个状态出发和返回到该状态的转移频率。可将它推广到一组状态，即从一组状态出发和返回该组状态的转移频率。

设 S 是状态空间的一个状态子集，将式（3-23）对 S 中的所有状态（对所有 $j \in S$）相加，得

$$\sum_{j\in S}P_j\sum_{j\notin S}P_{ji}=\sum_{i\notin S}P_i\sum_{j\in S}P_{ij} \tag{3-27}$$

该式表明转出状态子集 S 的频率等于转入状态子集 S 的频率。

3.2　组 帧 技 术

物理层通常仅负责比特传输，而不对比特含义和作用进行区分，数据链路层传输以帧为单位的比特数据块，那么一帧的开始点和结束点如何确定呢？组帧示意图如图 3-6 所示。

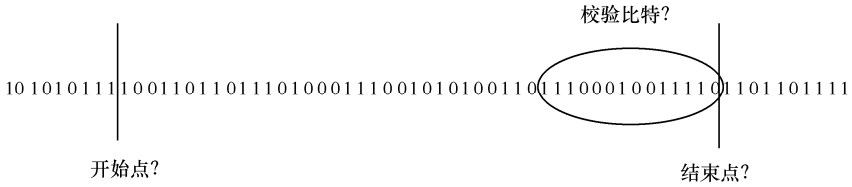


图 3-6　组帧示意图

3.2.1　面向字符的组帧技术

所谓面向字符的组帧，是指数据链路层的帧以字符为基础，Internet 中常用的面向字符的组帧技术的协议有 SLIP 和 PPP。

SLIP 的帧格式如图 3-7 所示，SLIP 帧运载的是高层 IP 数据报，它采用两种特殊字符：END（C0H，这里的 H 表示十六进制）和 ESC（DBH）。C0H 用于表示一帧的开始和结束；将 IP 数据报中的 C0H 转换为 DBH 和 DCH，将 DBH 转换为 DBH 和 DDH。一旦遇到 DBH，就进行字符转换，恢复 IP 报文中原有的 C0H 和 DBH，这样就可以完全以原 IP 数据报向 IP 层提交数据。

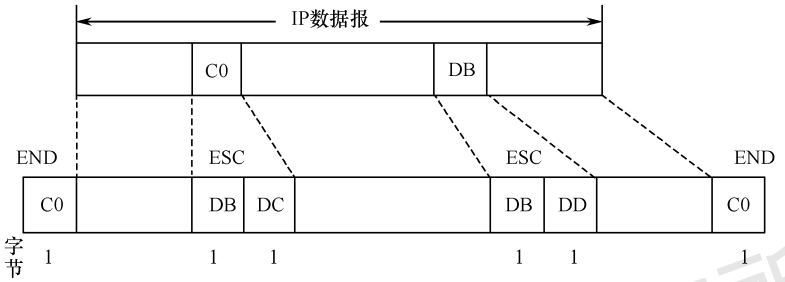


图 3-7　SLIP 的帧格式

PPP 的帧格式如图 3-8 所示，该格式与后面将要讨论的 HDLC 的帧格式相同。PPP 中，7EH 为一帧的开始/结束标志（F）；地址域（A）和控制域（C）取固定值（A=FFH，C=03H）；协议域（2 字节）若取 0021H，则表示该帧的信息是 IP 数据报，若取 C021H，则表示该帧的信息是链路控制数据，若取 8021H，则表示该帧的信息是网络控制数据；帧校验域（FCS）也为 2 字节，用于对信息域进行校验。若信息域中出现 7EH，则转换为 7DH、5EH；若信息域中出现 7DH，则转换为 7DH、5DH；若信息域中出现 03H，则转换为 7DH、23H；当信息流中出现 ASCII 码的控制字符（小于 20H）时，则在该字符前加入一个 7DH，并且改变该字符的编码。

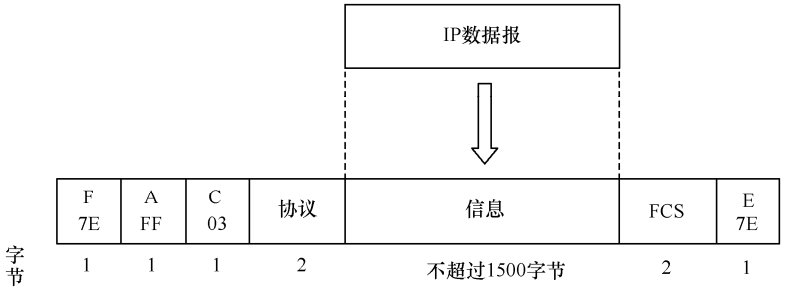


图 3-8 PPP 的帧格式

上述的两种帧格式均支持数据的透明传输，这些帧结构在处理时非常简单，但缺点是效率较低，插入了许多转义字符，且数据长度必须以字节或字符为单位。

3.2.2 面向比特的组帧技术

在面向比特的组帧技术中，通常采用一个特殊的比特串（称为 Flag），如 01^60（1^j 表示连续 j 个“1”）来表示一帧的开始和结束。当信息比特流中出现与 Flag 相同的比特串时，采用的方法是比特插入技术：当发送端信息流中每出现连续的 5 个“1”时，就插入一个“0”，如图 3-9 所示。接收端在收到 5 个“1”以后，若收到的是“0”，则将该“0”删去；若是“1”，则表示一帧结束。

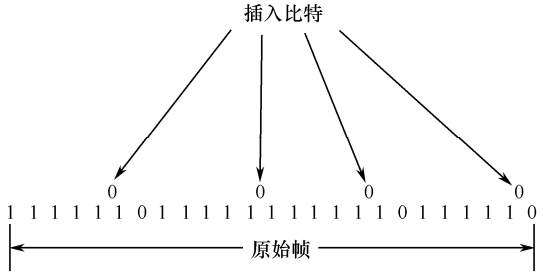


图 3-9 比特插入技术

3.2.3 采用长度计数的组帧技术

组帧技术的关键是正确地表示一帧何时结束，除采用特殊字符和 Flag 外，还可以用帧长度表示一帧的结束位置。如果最大长度为 K_{max}，那么长度域的比特数至少为 Int[lb{K_{max}}]+1，其中，Int[x]表示取 x 的整数部分，lb 表示以 2 为底的对数。长度域的比特数固定，采用长度计数的组帧技术的帧格式如图 3-10 所示。

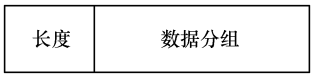


图 3-10 采用长度计数的组帧技术的帧格式

3.3 数据链路层的差错控制

3.3.1 差错检测

数据链路层差错检测的目的是有效地发现一帧数据经过信道传输后是否有错。常用的校

验方法有两类：一类是奇偶校验；另一类是循环冗余校验（CRC）。其基本思路是：发送端按照给定的规则，在 K 个信息比特后增加 L 个校验比特，在接收端对收到的信息比特重新计算 L 个校验比特，并比较接收校验比特和本地重新计算的校验比特。如果结果相同，那么认为传输无误，否则认为传输有误。

1．奇偶校验

奇偶校验码如表 3-1 所示，信息序列长 K=3，校验序列长 L=4。信息比特为 $S_1S_2S_3$，校验比特为 $C_1C_2C_3C_4$。

表 3-1　奇偶校验码

S_1	S_2	S_3	C_1	C_2	C_3	C_4	校验规则（⊕为模 2 加法）
1	0	0	1	1	1	0	$C_1=S_1\oplus S_2$
0	1	0	0	1	1	1	$C_2=S_1\oplus S_2\oplus S_3$
0	0	1	1	1	0	1	$C_3=S_1\oplus S_2$
1	1	0	1	0	0	1	$C_4=S_2\oplus S_3$
1	0	1	0	0	1	1	
1	1	1	0	1	0	0	
0	0	0	0	0	0	0	
0	1	1	1	0	1	0	

例如，设发送的信息比特为{100}，经过奇偶校验码生成的校验比特为{1110}，则发送的信息序列为{1001110}。若经过物理信道传输后，接收的序列为{1011110}，则本地根据收到的信息比特{101}计算出的校验比特应为{0011}。显然，该比特与接收到的校验比特{1110}不同，这表明接收的信息序列有错。

如果 L 取 1，$C=S_1\oplus S_2\oplus S_3\oplus\cdots\oplus S_K$，那么该方法即为最简单的单比特奇偶校验，它使得生成的码字（信息比特+校验比特）所含“1”的个数为偶数。该方法可以发现所有奇数比特错误，但是不能发现任何偶数比特错误。

在实际应用奇偶校验码的过程中，每个码字的 K 个信息比特可以是输入信息比特流中连续的 K 比特，也可以在信息流中每隔一定时间间隔（如 1 字节）取出一比特来构成 K 比特。为了提高检测错误的能力，可将上述两种取法重复使用。

2．CRC 校验

CRC 校验根据输入比特$\{S_{K-1}, S_{K-2},\cdots, S_1, S_0\}$通过下列 CRC 算法产生 L 位的校验比特。

CRC 算法如下。将输入比特表示为下列多项式的系数

$$S(D)=S_{K-1}D^{K-1}+S_{K-2}D^{K-2}+\cdots+S_1D+S_0 \tag{3-28}$$

式中，D 可以视为一个时延因子，D^i对应比特 S_i所处的位置。

设 CRC 生成多项式（用于生成 CRC 校验比特的多项式）为

$$g(D)=D^L+g_{L-1}D^{L-1}+\cdots+g_1D+1 \tag{3-29}$$

则校验比特对应下列多项式的系数

$$C(D)=\text{Remainder}\left[\frac{S(D)\times D^{L}}{g(D)}\right]=C_{L-1}D^{L-1}+\cdots+C_1D+C_0 \tag{3-30}$$

式中，Remainder[]表示取余数。式中的除法与普通多项式的长除法相同，其差别是系数采用二进制，其运算以模 2 为基础，最终形成的发送信息序列为（$S_{K-1}, S_{K-2},\cdots, S_1, S_0, C_{L-1},\cdots, C_1, C_0$）。

生成多项式的选择不是任意的，它必须使生成的校验比特有很强的检错能力。常用的几个 L 阶 CRC 生成多项式如下。

CRC-16（L=16）

$$g(D)=D^{16}+D^{15}+D^{2}+1 \tag{3-31}$$

CR-CCITT（L=16）

$$g(D)=D^{16}+D^{12}+D^{2}+1 \tag{3-32}$$

CRC-32（L=32）

$$\begin{aligned}g(D)=&D^{32}+D^{26}+D^{23}+D^{22}+D^{16}+D^{12}+D^{11}+D^{10}+\\&D^{8}+D^{7}+D^{5}+D^{4}+D^{2}+D+1\end{aligned} \tag{3-33}$$

其中，CRC-16 和 CRC-CCITT 产生的校验比特为 16bit，CRC-32 产生的校验比特为 32bit。

【例 3-3】设输入比特为{10110111}，采用 CRC-16 生成多项式，求其校验比特。

解 输入比特可表示为

$$S(D)=D^{7}+D^{5}+D^{4}+D^{2}+D^{1}+1 \quad (K=8)$$

因为 $g(D)=D^{16}+D^{15}+D^{2}+1$ （$L=16$），所以

$$\begin{aligned}C(D)&=\text{Remainder}\left[\frac{S(D)\times D^{16}}{g(D)}\right]\\&=\text{Remainder}\left[\frac{D^{23}+D^{21}+D^{20}+D^{18}+D^{17}+D^{16}}{D^{16}+D^{15}+D^{2}+1}\right]\\&=D^{9}+D^{8}+D^{7}+D^{5}+D^{4}+D\end{aligned}$$

由此式可得校验比特为{0000001110110010}。最终形成的经过校验后的发送信息序列为{10110111000000111011 0010}。

在接收端，将接收到的序列

$$R(D)=r_{K+L-1}D^{K+L-1}+r_{K+L-2}D^{K+L-2}+\cdots+r_1D+r_0 \tag{3-34}$$

除以 $g(D)$，如果 Remainder[$R(D)/g(D)$]=0，那么认为接收无误。

3.3.2 ARQ 协议

当物理链路上传输的帧到达接收端时，理论上帧到达的顺序与发送的顺序应相同，但实际中可能被时延，可能丢失，也可能出错。接收端发现传输帧有错误时，最简单的处理方法是自动请求发送端重发（ARQ），即接收端收到一帧后，若发现该帧传输错误，则通过反馈信道以某种反馈规则通知发送端重复上述过程，直到接收端收到正确的帧。

停等式 ARQ（Stop-and-Wait ARQ）的基本思想是在开始下一帧传输以前，必须确保当前帧已被正确接收。假定 A、B 之间有一条可双向传输的链路，如图 3-11（a）所示。假定 A 到 B 的传输链路称为正向链路，则 B 到 A 的传输链路称为反向链路。在该链路上 A 要发送数据

帧给 B，具体的传输过程如下。A 发送一个数据帧（DATA）后，若 B 接收正确，则 B 向 A 返回一个肯定应答（ACK）帧，其时间关系如图 3-11（b）所示（图中的横线表示时间轴，A 到 B 之间的连线表示传输方向和相对传播时延）；若 B 接收错误，则 B 向 A 返回一个否定应答（NAK）帧，其时间关系如图 3-11（c）所示。A 必须在收到 B 的 ACK 帧后，才可发送下一帧。若 A 发送一帧（并给定时器设置一个初值）后，在一个规定的时间（定时器溢出的时间）内没有收到对方的 ACK 帧，则重发该帧；若收到了 NAK，则重发该帧，如图 3-11（c）所示。

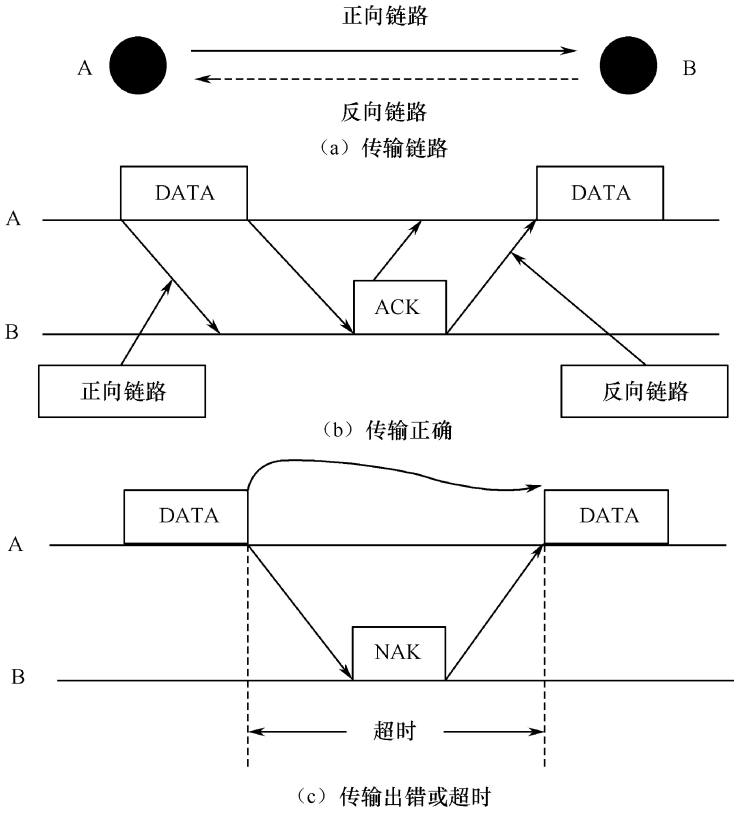


图 3-11　停等式 ARQ 传输示意图

A 与 B 之间的双向链路都可能出错，如何保证该协议能够正确地工作呢？基本方法是在传输的帧中增加发送序号（SN）和接收序号（RN）。

在停等式 ARQ 协议中，假定 A 向 B 发送分组（A→B），节点 A 的发送算法如下。

（1）置 SN =0。

（2）若从高层接收到一个分组，则将 SN 指配给该分组；若没有分组，则等待。

（3）将发送序号为 SN 的分组装入物理帧中，并发送给 B。

（4）若 B 接收的 RN>SN，则将 SN 加 1，返回（2）。若在有限长的规定时间内，没有从 B 接收到 RN>SN 的帧（应答），则返回（3）进行重传。

节点 B 的接收算法如下。

（1）置 RN=0。

（2）无论何时正确地接收一个 SN=RN 的帧，都将该帧中的分组送给高层，并将 RN 加 1。

（3）在接收到该分组后的一个有限长的规定时间内，将 RN 放入一帧的 RN 域中并发送给 A，然后返回（2）。

在上述算法中，“规定时间”通常是采用定时器来确定的。RN 通常是附带在反向数据帧中传给对方的。若 B 没有数据传输给对方，则应单独传输一个包含 RN 的无数据帧给 A。综上可以看出，反向业务流的存在对停等式 ARQ 的机制没有任何影响，它仅对应答的时延有影响，因此在后面的讨论中将忽略反向业务流。

对于上述停等式 ARQ 协议或其他类似的协议，要从两个主要方面对其进行评估：一是算法（协议）的正确性；二是算法（协议）的有效性。算法的正确性是指算法始终能够正常工作，正确接收输入的数据和状态，产生正确的动作与正确的输出结果。算法的有效性可以从三个方面来表述：一是吞吐量（或通过量）；二是链路的利用率；三是分组时延。

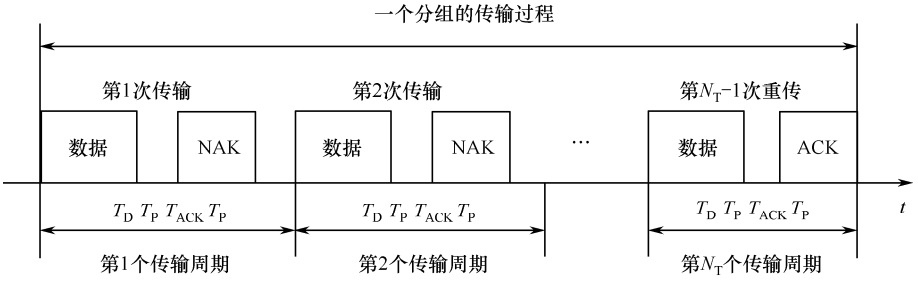
设数据帧是固定帧长，其数据帧长为 T_D（秒），肯定应答和否定应答帧长均为 T_{ACK}（秒），物理链路的传播时延为 T_P（秒），则在忽略算法的处理时延的情况下，一帧的传输周期为 $T_D+T_P+T_{ACK}+T_P$。假定任意一个数据帧平均需要发送 N_T 次（一次初发和 N_T-1 次重发）才能成功，则该帧平均需要 N_T 个传输周期。

传输周期示意图如图 3-12 所示，由图可以看出，物理链路的最大平均利用率（物理链路以最大可能负荷在传输时的平均利用率）为

$$U=\frac{T_D}{N_T\left(T_D+T_P+T_{ACK}+T_P\right)}=\frac{1}{N_T\left(1+2T_P/T_D+T_{ACK}/T_D\right)} \tag{3-35}$$

令 $\alpha=T_P/T_D$，忽略应答帧的传输时间，则有

$$U=\frac{1}{N_T(1+2\alpha)} \tag{3-36}$$



NAK— 否定应答； ACK— 肯定应答； T_P— 传播时延； T_D— 数据帧长； T_{ACK}— 应答帧长

图 3-12 传输周期示意图

假定数据帧的误帧率为 $p=1-q$，应答帧因其长度很短，可以忽略其出错的可能性，即认为应答帧总是可以正确地传输，则一个数据帧需 i 次发送成功的概率为 $p^{i-1}(1-p)$，从而有

$$N_T=\sum_{i=1}^{m}ip^{i-1}(1-p)=\frac{1}{1-p} \tag{3-37}$$

将式（3-37）代入式（3-36），得链路的最大平均利用率为

$$U=\frac{1-p}{1+2\alpha} \tag{3-38}$$

从式（3-38）可以看出，误帧率越高，链路的最大平均利用率就越低，链路的传播时延就越大。即使当 p=0 时，如果 α >0.5（数据帧长小于传播时延的两倍），那么链路的最大平均利用率也小于 50%。

根据图 3-12 可得，停等式 ARQ 的最大平均吞吐量（单位为分组/s）为

$$S=\frac{1}{N_{\mathrm{T}}\left(T_{\mathrm{D}}+T_{\mathrm{P}}+T_{\mathrm{ACK}}+T_{\mathrm{P}}\right)}\approx\frac{1-p}{T_{\mathrm{D}}\left(1+2\alpha\right)} \tag{3-39}$$

停等式 ARQ 的平均分组时延为

$$\begin{aligned}D&=\text{组帧时延}+\left(N_{\mathrm{T}}-1\right)\left(T_{\mathrm{D}}+T_{\mathrm{P}}+T_{\mathrm{ACK}}+T_{\mathrm{P}}\right)+\left(T_{\mathrm{D}}+T_{\mathrm{P}}\right)\\&\approx\text{组帧时延}+N_{\mathrm{T}}T_{\mathrm{D}}+\left(2N_{\mathrm{T}}-1\right)T_{\mathrm{P}}\end{aligned} \tag{3-40}$$

组帧时延是指从高层分组的第一比特到达数据链路层开始，到数据链路层将该分组的所有比特收齐，通过增加控制头（如帧起止标志、发送序号、接收序号等）和校验比特（CRC）形成可传输的数据帧为止。它取决于网络层与数据链路层之间的接口速率和方式，以及数据链路层的处理速度及方式。例如，若网络层与数据链路层运行在相同的微处理器或计算机系统上，采用数据块传递的方式来传递分组，则组帧时延相当小且可以忽略。若网络层与数据链路层采用传输速率为 R（单位为 b/s）的接口交换数据，则组帧时延为 K/R［K 为分组的长度（比特数）］。

【例 3-4】 设有三条物理链路：第一条是卫星链路，其传输速率为 64kb/s，传播时延 T_{P}=270ms；第二条是 5000km 的电话网链路，其传输速率为 9.6kb/s，传播时延 T_{P}=25ms；第三条是 500m 的同轴电缆链路，其传输速率为 10Mb/s，传播时延 T_{P}=2.5μs。试求当 K=1000bit 和 K=10000bit 时停等式 ARQ 的链路的最大平均利用率 U、最大平均吞吐量 S（分组/s）和平均分组时延 D（ms）。

解　根据式（3-38）、式（3-39）和式（3-40）可得不同帧长情况下停等式 ARQ 的性能如表 3-2 所示。从表 3-2 可以看出 α 越小，链路的利用率越高，吞吐量越大，时延越小。由此可以看出：在卫星链路上采用停等式 ARQ 的效率很低。

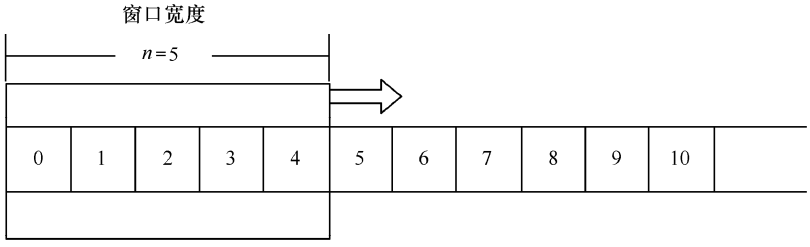
表 3-2　不同帧长情况下停等式 ARQ 的性能

链路类型 / 性能参数	传输速率	T_{P}	p	α		U		S/（分组/s）		D/ms	
				K=1000bit	K=10000bit	K=1000bit	K=10000bit	K=1000bit	K=10000bit	K=1000bit	K=10000bit
卫星链路	64kb/s	270ms	10%	17.3	1.73	0.025	0.202	1.6	1.3	887	1043
电话网链路	9.6kb/s	25ms	5%	0.24	0.024	0.766	0.906	7.4	1.0	187	1174
同轴电缆链路	10Mb/s	2.5μs	1%	0.02	0.002	0.950	0.986	9500	986	0.1	1

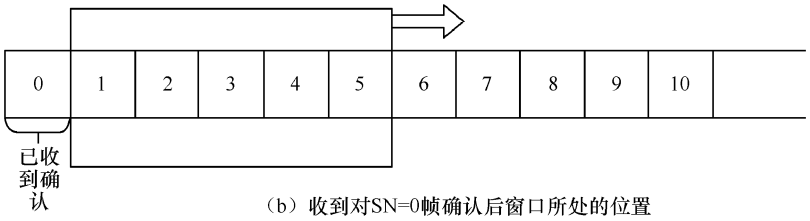
由于停等式 ARQ 的效率很低，人们认为发送端在等待对方应答时应当做更多的事情，因此提出了三种改进方案：返回 n-ARQ（Go Back n ARQ）、选择重发式 ARQ（Selective Repeat ARQ）和并行等待式 ARQ（ARPANET ARQ）。这里讨论返回 n-ARQ，它是一种应用最广泛的 ARQ 协议，已应用在 HDLC、SDLC、ADCCP 和 LAPB 等标准的数据链路控制（DLC）协议中。

返回 n-ARQ（有时也称为连续 ARQ）的基本思路是：发送端在没有收到对方应答的情况下，可以连续发送 n 帧。接收端仅接收正确且顺序连续的帧，其应答中的 RN 表示 RN 以前的所有帧都已正确接收（这里接收端不需要每收到一个正确的帧就发出一个应答，可对接收

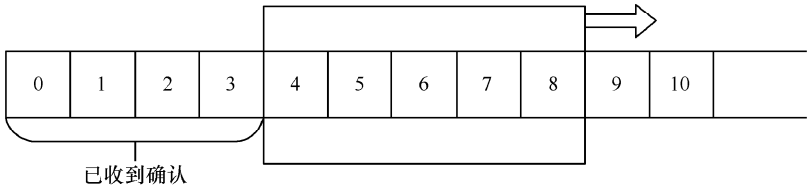
到的正确顺序的最大帧序号进行应答）。这里 n 是一个重要参数，称为（滑动）窗口宽度，如图 3-13 所示。设窗口宽度为 5，则在开始时，发送端可发送 0～4 号共 5 个帧［图 3-13（a）］。当收到对 SN=0 帧的确认后，发送端可发送 1～5 号共 5 个帧［图 3-13（b）］。当收到对 SN=3 帧的确认后，发送端可发送 4～8 号共 5 个帧［图 3-13（c）］。随着应答的不断到达，窗口不断地向前滑动。



（a）开始时窗口所处的位置



（b）收到对SN=0帧确认后窗口所处的位置



（c）收到对SN=3帧确认后窗口所处的位置

图 3-13　滑动窗口示意图

导致返回 n-ARQ 效率下降有三个方面的原因。第一个原因是反向帧过长，这就要求增大 n。增大 n 以后，只要 n 个正向传输帧长之和大于反向帧长的概率大幅提高，就可以有效地减小反向帧过长带来的影响。第二个原因是反向应答出错，这也要求增大 n。增大 n 以后，应答帧被后来的应答帧所补救的概率增大，从而可以降低出错对效率的影响。第三个原因是正向传输出错，如果根据前两个原因的要求增大 n，这将导致正向传输出错后，反向应答到达发送端的时延较长，需上传的帧数大幅增加，反而会导致系统效率下降。解决第三个问题的方法就是提高出错的反馈速度，即返回一个错误帧，立即返回一个短的应答帧，使发送端尽快返回重发。

为了分析简单起见，假定数据帧长是一个固定值，且假定应答帧传输时间很短（可以忽略）。下面来讨论返回 n-ARQ 的效率。

返回 n-ARQ 的效率与链路的传播时延（T_P）、数据帧长（T_D）、窗口宽度（n）等参数紧密相关。从图 3-14 可以看出，当 $nT_D \geqslant d$（$d = T_D + 2T_P$）时，应答帧可以及时返回，在传输

无差错的情形下链路的最大平均利用率为 1，发送端可连续不断地发送帧。当 $nT_D < d$ 时，链路的平均利用率为 nT_D / d，即在 $d = T_D + 2T_P$ 时间内，发送端最多可以发送 n 帧。

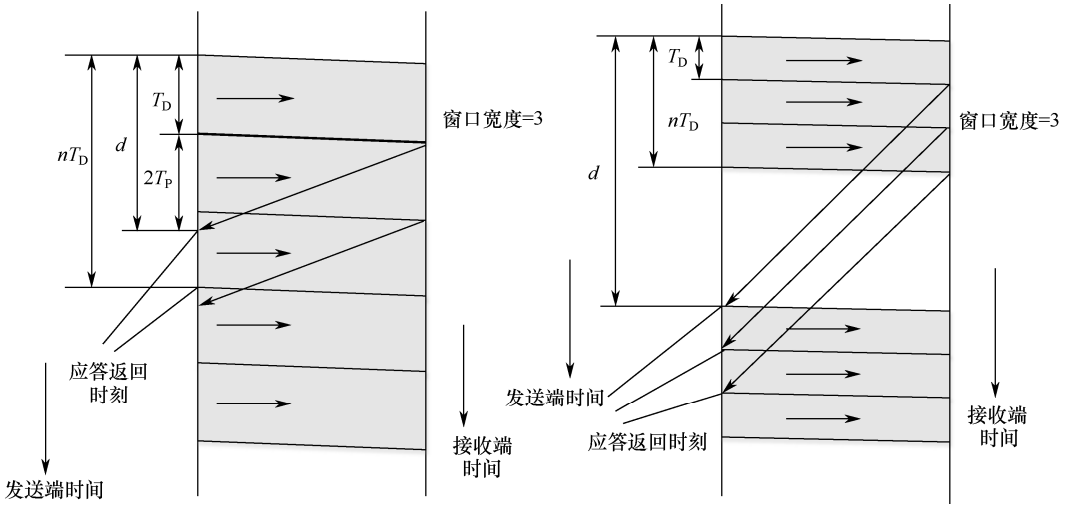


图 3-14　d、T_D、n 的相互关系

设传输过程中的误帧率为 p，每出现一个错帧，发送端就会重复 n 帧。若一帧经过 i 次传输成功（i=1 表示第一次传输，i>1 表示重传），则表明经过了 i−1 次返回后，该帧才传输成功，而每次返回需要传输 n 帧，因而总的所需传输的帧数为 $1+(i-1)n$，其出现的概率 $p^{i-1}(1-p)$。由此可得，成功地传输一帧平均所需传输的帧数 N_f 为

$$\begin{aligned} N_f &= \sum_{i=1}^{n}\left[1+(i-1)n\right]p^{i-1}(1-p) \\ &\approx 1+\frac{np}{1-p} \end{aligned} \tag{3-41}$$

利用该式可得返回 n-ARQ 中链路的最大平均利用率为

$$U = \begin{cases} \dfrac{1}{N_f} & nT_D \geqslant d \\ \dfrac{nT_D}{N_f d} & nT_D < d \end{cases} = \begin{cases} \dfrac{1}{N_f} & n \geqslant 1+2\alpha \\ \dfrac{n}{N_f(1+2\alpha)} & n < 1+2\alpha \end{cases} \tag{3-42}$$

式中，$\alpha = \dfrac{T_P}{T_D}$，将式（3-41）代入式（3-42）得

$$U = \begin{cases} \dfrac{1-p}{1+(n-1)p} & n \geqslant 1+2\alpha \\ \dfrac{n(1-p)}{(1+2\alpha)\left[1+(n-1)p\right]} & n < 1+2\alpha \end{cases} \tag{3-43}$$

当 p=0.01 时，链路的最大平均利用率 U 与 n 的关系如图 3-15 所示。从图中可以看出，U 与 α、n 紧密相关。当 α 相对较大（如在卫星链路中）时，为了达到较高的利用率，应选择较大的 n。从图中可以看出，当 $n=[1+2\alpha]$（$[x]$表示取大于或等于 x 的最小整数）时，利用率最高，也就是最佳的窗口宽度（nT_D）近似等于数据帧长与 2 倍的传播时延之和。

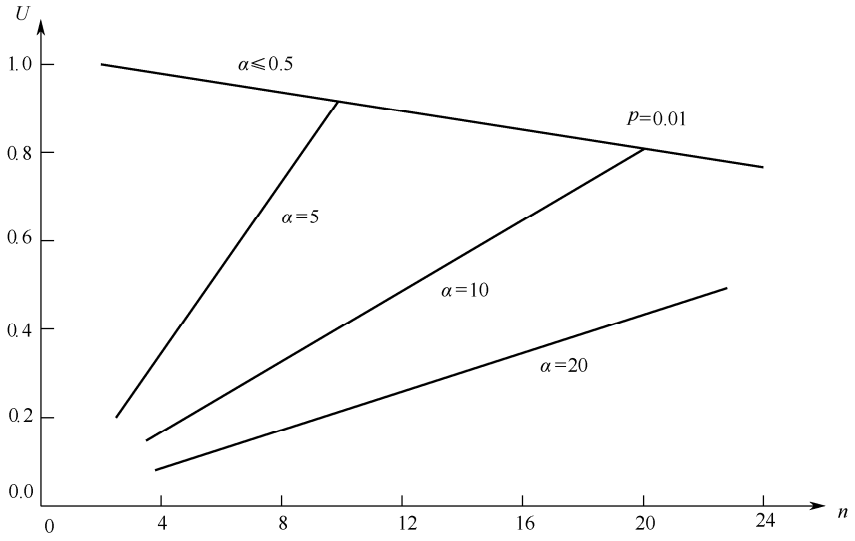


图 3-15 链路的最大平均利用率 U 与 n 的关系

3.3.3 最佳帧长

可以从两个方面来考察最佳帧长：在一条链路上使传输效率最高；在由多条链路构成的传输路径上使传输效率最高。

首先考察一条链路上的最佳帧长。在实际传输过程中，每一帧数据（设帧长为 l_f bit）通常包括 l_d bit 的数据负荷和 l_h bit 的控制信息（l_f=l_d +l_h）。如果帧长较小，控制信息所占的比例较大，那么链路利用率会下降。如果帧长较大，那么在数据传输过程中因信道误码而导致帧传输错误的概率较大，重传的次数将增大，这也会导致链路利用率的下降。因此存在一个最佳帧长，可使链路利用率最高。设链路的误比特率为 p_b，在错误随机的条件（如卫星链路）下，数据帧的差错率或误帧率 p 为

$$p = 1-(1-p_b)^{l_f} \tag{3-44}$$

当 p_b 很小时，式（3-44）可近似为

$$p \approx l_f p_b \ll 1 \tag{3-45}$$

以停等式 ARQ 为例，其链路的利用率为

$$U_e = \frac{l_d(1-l_f p_b)}{l_f(1+2\alpha)} \tag{3-46}$$

式中，$\alpha = \frac{T_P}{T_D} = \frac{T_P}{l_f T_b} = \frac{\alpha_0}{l_f}$，$T_b$ 为比特宽度，$\alpha_0 = \frac{T_P}{T_b}$。将 l_f=l_d+l_h 代入式（3-46）并整理，得

$$U_e = \frac{(1-2l_h p_b)l_d - p_b l_d^2 - p_b l_h^2}{l_d + (l_h + 2\alpha_0)} \tag{3-47}$$

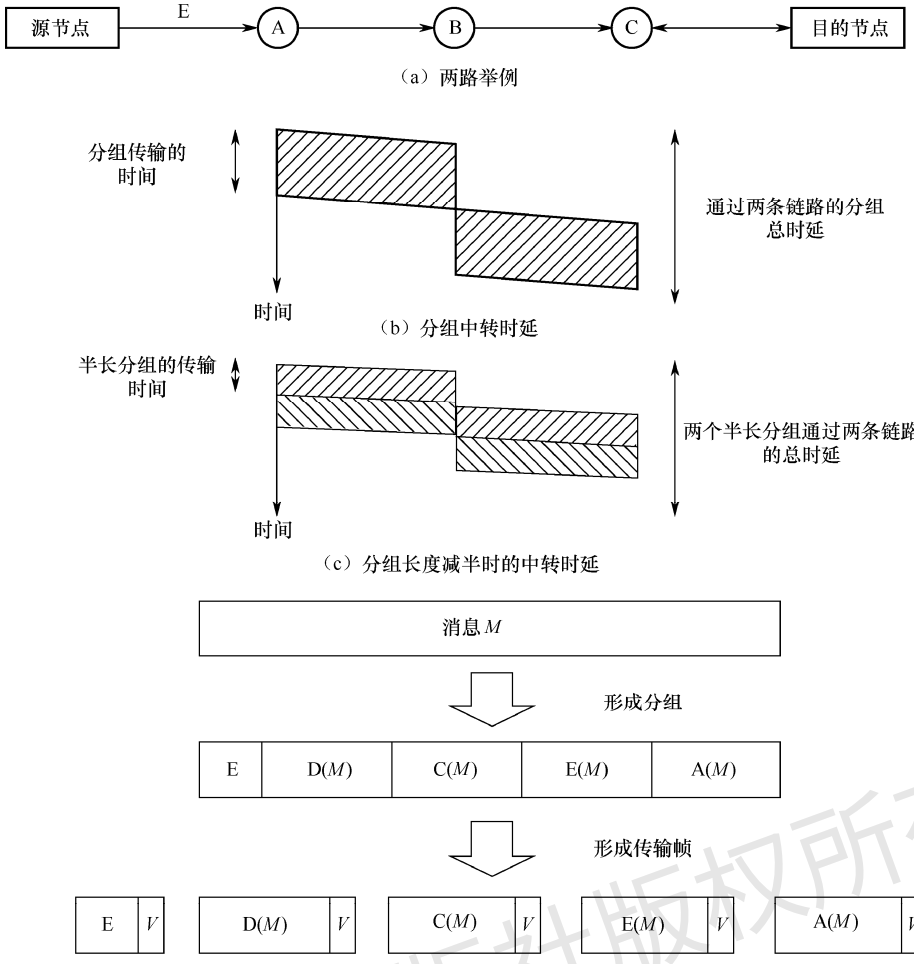
将该式对 l_d 求导并令其为零，可得最佳帧长为

$$l_d = \sqrt{\frac{l_h+2\alpha_0}{p_b}} - (l_h + 2\alpha_0) \tag{3-48}$$

下面讨论当分组经过多次中转才能到达目的节点时，可使网络开销最小和时延最小的最佳帧长。

一条消息分成不同长度的分组，经过中转到达目的节点，如图 3-16（a）所示。从图 3-16（b）可以看出，分组的传输时延为分组长度的 2 倍。在图 3-16（c）中，将分组的长度减少一半，则此时的时延仅为原来的 1.5 倍。因此从缩短时延的角度考虑，分组的长度应尽可能小。

设消息的长度为 M，帧长为 K，通常每一帧都包含固定的开销 V（含头和尾），这样每条消息要分成 $\mathrm{Int}[\frac{M}{K}]^+$ 个分组（ $\mathrm{Int}[x]^+$ 表示大于或等于 x 的最小整数）。在消息的传输过程中，前 $\mathrm{Int}[\frac{M}{K}]$ 个分组均有 K 比特，而最后一个分组的比特数为 1～K。一条消息要传输的总比特数为 $M+\mathrm{Int}[\frac{M}{K}]^+\times V$ 。图 3-16（d）所示为消息的分段封装过程。在图中， $\mathrm{Int}[\frac{M}{K}]^+=4$ 。



（a）两路举例

（b）分组中转时延

（c）分组长度减半时的中转时延

（d）消息的分段封装过程

图 3-16　分组的中转过程

帧长 K 减小，会引起帧数增大，这会导致传输开销和网络处理负荷增大，因此应当增大帧长。当 $M\to+\infty$时，开销所占的比例为 $\frac{V}{V+K}$ 。因此，增大帧长会降低开销。综合考虑时延

和开销两个方面，存在一个最佳帧长。

设每条链路的容量为 C（单位为 b/s），一条消息经过中转传到目的节点的总时间为 T，在忽略各节点的处理和缓存时延的情况下，有

T=消息在最后一条链路上的传输时间+(j–1)条链路引起的时延

$$T=\frac{M+\mathrm{Int}[\frac{M}{K}]^{+}V+(j-1)(K+V)}{C} \tag{3-49}$$

对 T 求平均值，得

$$E(T)=\frac{1}{C}\left\{(K+V)(j-1)+E\{M\}+E\left\{\mathrm{Int}\left[\frac{M}{K}\right]^{+}\right\}V\right\} \tag{3-50}$$

将 $E\left\{\mathrm{Int}\left[\frac{M}{K}\right]^{+}\right\}\approx E\left\{\frac{M}{K}\right\}+\frac{1}{2}$ 代入式（3-50），并使得式（3-50）最小，此时的最佳帧长为

$$K_{\mathrm{opt}}\approx\sqrt{\frac{E\{M\}V}{j-1}} \tag{3-51}$$

在上面的讨论中忽略打包时延，即只关心消息进入系统后到达目的节点的时延。而对于某些流型业务（如语音业务），需要关心从给定的某比特进入网络到该比特离开网络的时延，这时就必须考虑打包时延。

设输入的比特速率为 R，帧长为 K，收发之间各链路的容量分别为 $C_1, C_2, \cdots$（均大于 R），分组的开销为 V 比特，则 1 比特的时延为

$$T=\text{打包时延+各链路的传输时延}=\frac{K}{R}+\sum_i\frac{K+V}{C_i} \tag{3-52}$$

从式（3-52）可以看出，当 C_i 很大时，T 主要由 $\frac{K}{R}$ 决定。例如，对于 64kb/s 的数字语音，通常要求打包时延小于 10ms，因此 K 通常取 500bit 或更小。

以上都是从单个用户的角度来讨论网络处于轻负荷状态下，如何选取最佳帧长的问题的。但是对于网络而言，各用户的最佳长度均不相同，这就会出现当不同长度的分组在一条链路上传输时，很长的分组的传输时延很大，从而阻碍了较短分组的快速传递。这就像在一条单行线上，一辆慢速行驶的货车阻塞了后面小汽车的快速行驶，这种现象称为 Slow Trunk Effect（慢速货车效应）。因此，最佳帧长应由网络来设定，而不应由各终端自行设计。

在以上讨论的系统中，帧长是可变的。另外一种方式就是采用固定帧长，典型的是 ATM 信元，其长度为 53 字节。固定帧长的优点是便于硬件实现，适用于不同类型的业务种类。

3.4 标准的数据链路控制协议及其初始化

前面研究了在数据链路上如何进行有效的帧传输及数据链路层流控，本节将讨论如何在实际的协议中应用这些结果。

3.4.1　标准的数据链路控制协议

目前常用的标准的数据链路控制（DLC）协议有：IBM 提出的 SDLC、ISO 建议的 HDLC、ANSI 规定的 ADCCP 和 CCITT 建议的 LAPB 等。其中，HDLC 与 ADCCP 的功能相同，SDLC 是 HDLC 的一个功能子集，LAPB 也是 HDLC 的一个功能子集。

HDLC（ADCCP）是为多种物理链路设计的，包括多址链路、点对点链路、全双工和半双工链路。它包括三种工作模式：正常响应模式（NRM）、异步响应模式（ARM）和异步平衡模式（ABM）。

正常响应模式（NRM）用于主从式链路，即链路的一端是主站（节点），另一端是从站。主站负责控制和协调双方的通信过程。典型的应用场合是一台计算机与多个外设之间的链路，采用轮询（polling）机制，实现主站与从站之间的通信。

异步响应模式（ARM）也采用主从模式，但对从站没有严格的限制，该方式未被广泛使用，后面将不再讨论。

异步平衡模式（ABM）用于全双工点对点链路，链路两端的节点具有相同的责任进行链路控制，这是应用得最广泛的协议之一。

LAPB 仅使用 ABM 模式，SDLC 使用 NRM 和 ARM 模式。上述 4 种协议及 3 种工作模式都使用如图 3-17 所示的标准 DLC 的帧结构。

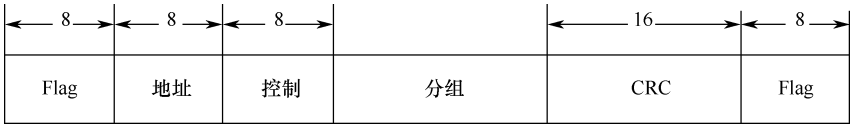


图 3-17　标准 DLC 的帧结构

图 3-17 中采用面向比特的帧结构。Flag =01^{6}0，采用比特插 0 的技术来消除帧中可能出现的与 Flag 相同的比特串。CRC 校验采用 CRC-CCITT，其生成多项式为 $D^{16}+D^{12}+D^2+1$。在具体计算 CRC 校验码时，将地址域和控制域取反，然后与分组数据一起参与 CRC 计算，将求得的余数取反，形成 CRC 比特（或者将 CRC 比特取反后进行传输）。这种实现 CRC 的方法可以避免全 0 的序列满足 CRC 校验的可能性，也可以避免在帧的头或尾增加或删去几个 0 而满足 CRC 校验的可能性。

地址域在常用情况下为 1 字节（8bit），它用于在多用户共享一条链路时区分不同的节点。在不同的工作方式下，地址域的功能可以不同。例如，在 NRM 模式中，地址域总是从站的地址；当用于点对点通信时，地址域没有作用。

控制域用来区分不同的帧类型，其格式如图 3-18 所示。

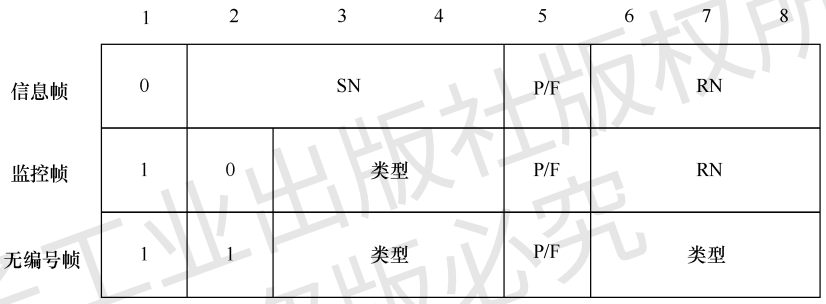


图 3-18　控制域的格式

它有三种格式：信息（I）帧、监控（S）帧和无编号（U）帧。信息帧采用模 8 的返回 n-ARQ 方式进行传输，它对应的控制包括 SN 和 RN。监控帧用于在无数据传输时返回 ARQ 的信息（如 RN）或加速返回 ACK 和 NAK 帧。无编号帧用于链路的建立和终止及附加信息的传输。

控制信息的种类是靠控制域的第 1 比特和第 2 比特来区分的。第 1 比特为“0”表示为信息帧，第 1 比特和第 2 比特为“10”表示为监控帧，第 1 比特和第 2 比特为“11”表示为无编号帧。控制域中的第 5 比特为查询/结束（P/F）比特，在查询时称为 P 比特，在响应时称为 F 比特。在不同的工作模式中，P/F 比特的用法不同。例如，在 NRM 模式中，当主站查询从站时，置 P=1。若从站响应有多个帧，则应将最后的响应帧 F 置 1，其余各响应帧置 F=0。在 ABM 模式中，任意站均可主动发送 S 帧和 I 帧，此时将 P 置 1，对方收到 P=1 的帧后，将 F 置 1 进行应答。

监控帧有 4 种类型：RR 表示接收准备好，RNR 表示接收未准备好，REJ 表示拒绝，SREJ 表示选择拒绝。它们通过控制域中的第 3 比特和第 4 比特来区分。“00”对应 RR，“10”对应 RNR，“01”对应 REJ，“11”对应 SREJ。RR 帧是正常的响应帧，它用于反向信道没有数据帧传输时，携带 RN 对发送端进行应答。RNR 用于在接收端暂不能接收更多帧（缓冲区满）时给对方提示，RNR 也包括 RN。REJ 用于在刚接收的帧出错或帧的序号与期望的序号不符时给发送端应答，发送端将重发序号为 RN 及其后面的分组，同时表明 RN 以前的分组已正确接收，REJ 可以提高 n-ARQ 的效率。SREJ 是一种简单的选择重发方式，它要求发送端重发序号为 RN 的分组。

无编号帧用于链路的建立 、拆除和特殊控制。无编号帧是通过第 3、4、6、7、8 比特来区分不同类型的，目前只定义了 15 种无编号帧。无编号帧用于设置工作模式，如置正常响应模式 SNRM（Set NRM）、置异步平衡模式 SABM（Set ABM）等。例如，节点 A 和 B 之间有一条链路，当 A 发出 SNRM 后用无编号帧 UA（Unnumbered Acknowledgment）予以响应；当对方通信结束后用无编号帧发出拆除 DISC （DISConnect）命令。

下面举例说明数据链路控制（DLC）协议的工作过程。为简化描述，采用下列命令格式：X（Y）Z。其中，X 表示地址；Y 表示信息帧的 SN 和 RN，或者监控帧的类型及参数，或者无编号帧的类型；Z 表示 P/F 比特是否置位。

NRM 的工作过程如图 3-19 所示。图 3-19 中的主要步骤是：A 先与 B 建立链路，再与 C 建立链路，A 与 B、C 传输数据分组的过程及 A 与 B 拆除链路的过程具体如下。

A 首先发出命令 B（SNRM）P 来初始化 A 到 B 的链路，采用 UA 帧进行应答，其对应 A 到 C 的链路。假定 A 到 C 的第一次命令 C（SNRM）P 传输出错，A 经过一定的等待时延后，重新进行初始化。当 A 成功初始化 A 到 B 及 C 的链路后，A 到 B 发送数据，其格式为 B（SN,RN）。A 发出的帧为 B（0,0）、B（1,0）和 B（2,0）P。B（2,0）P 表示 A 已传输结束，询问 B 是否有数据。B 在收到此格式的数据后，开始发送自己的数据，其格式为 B（SN,RN）。B 发出的帧为 B（0,1）和 B（1,1）F。B 置位 F 表示数据传输结束，然后 A 查询 C 并与 C 之间进行数据传输。

从图中可以看到，A→B 及 B→A 之间的数据传输出现了错误。A 在收到帧 B（1,1）F 时，发现 B 希望接收到的帧序号 RN=1，因而 A 将重发 SN=1 及后续（SN=2）帧。当 B 收到 A 重发的 B（2,0）P 后，发现对方未能收到自己的 SN=0 帧，因而重发 SN=0 帧及 SN=1 帧。此时 B 又有新的帧到达，可接着发送该 SN=2 帧，并告知对方传输结束。假定 A 决定结束 A 到 B

之间的链接，则 A 向对方发出 RNR 帧，并同时应答对方 B 已发送的帧。此外，置位 P 还要求 B 对此命令进行应答，此帧的帧格式为 B（RNR,3）P。A 得到对方的确认［收到 B（RR,3）F 帧］后，A 发出拆除链路的命令，命令格式为 B（DISC）P，并要求对方应答，格式为 B（UA）F。当 A 收到对方的拆除应答后，A 与 B 之间的正式链路断开。

ABM 的工作过程如图 3-20 所示。注意在图 3-20 中，地址域总是从站的地址，主站（A）在发出命令和数据时，使用从站（B 或 C）的地址，从站（B 或 C）用自己的地址予以响应或传输数据。

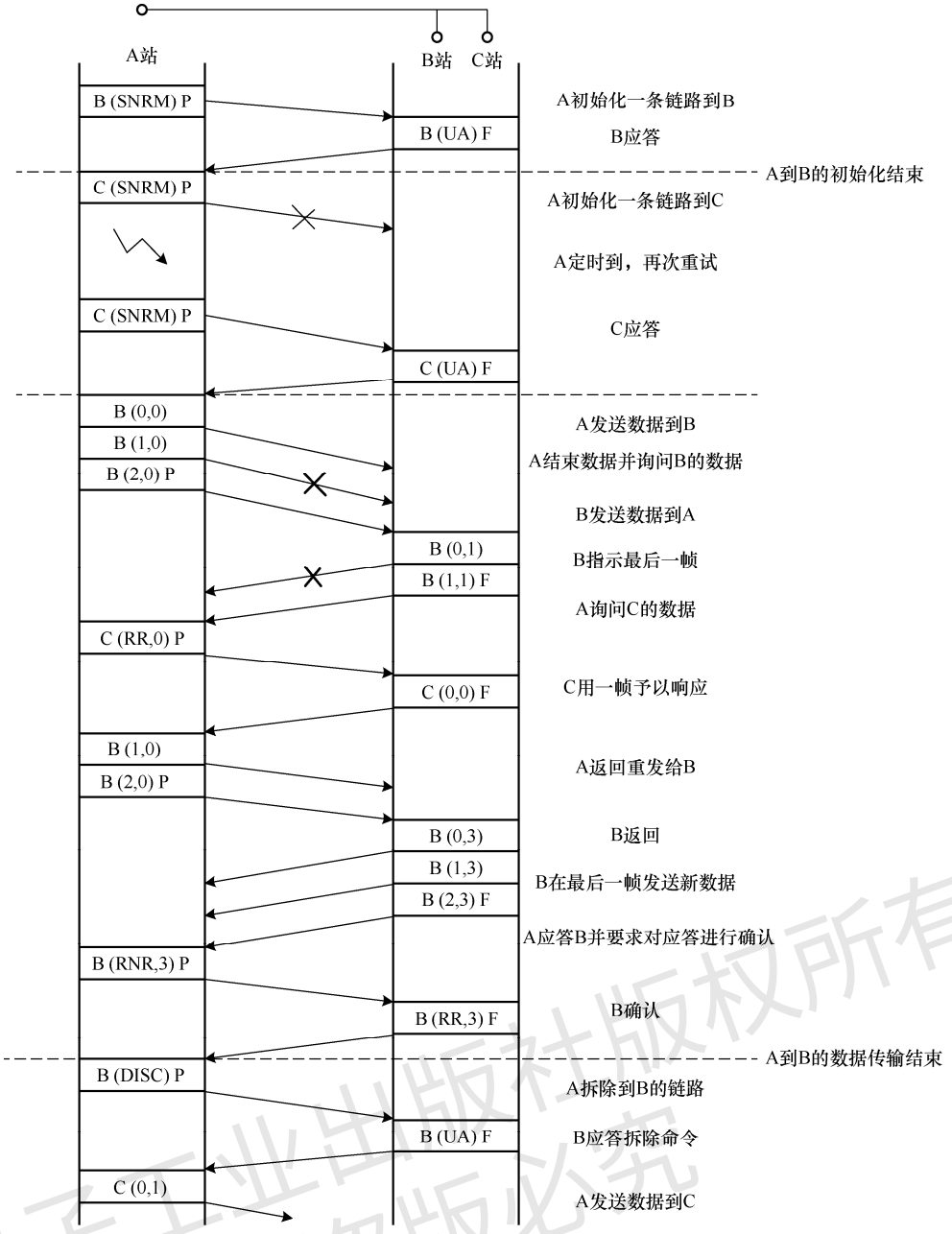


图 3-19　NRM 的工作过程

ABM 的工作包括三个主要的过程：A 与 B 建立链路，A 和 B 同时传输数据，A 拆除链路。在这些过程中，任何一方在发送数据和要求对方应答时都使用对方的地址，在应答时都使用自己的地址。值得注意的是拆除过程，A 首先发送 RNR 监控帧［B（RNR,4）P］，表示不再接收对方的分组，该帧包括 A 已接收到的 B 的分组序号（RN），从而可使 B 明确 A 已接收到的分组。B 通过 RR 帧[B（RR,5）F]予以应答，并同时包括 B 已接收到的 A 的分组（RN），从而可使 A 明确 B 已接收到的分组。这样双方对链路状态有较明确的认识。此时 A 再发出拆除命令，B 收到后予以应答，至此通信过程结束。

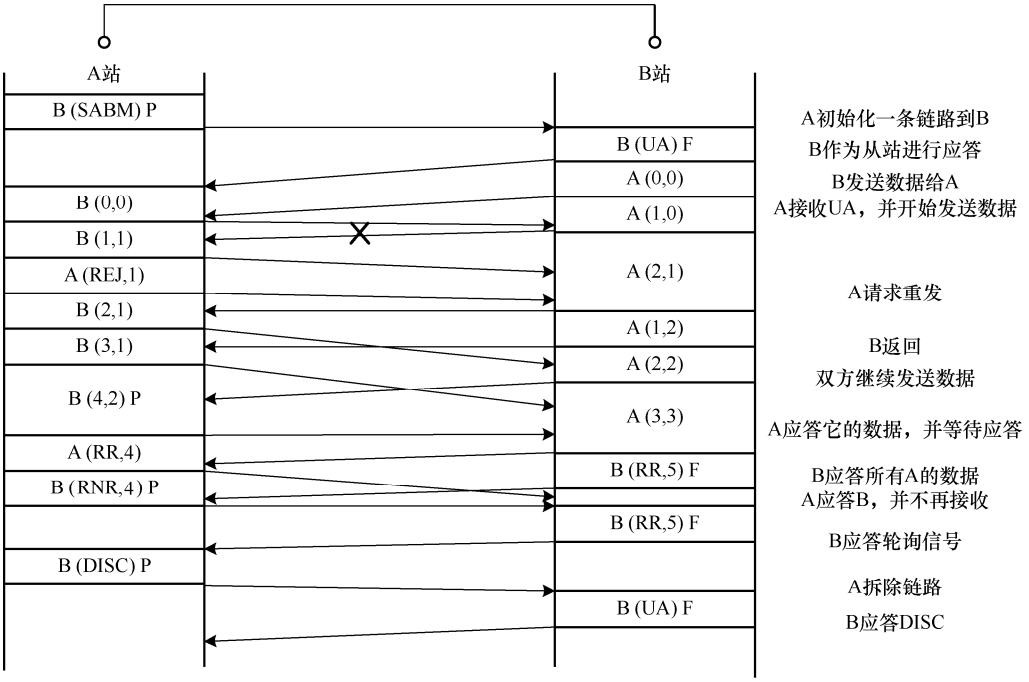


图 3-20　ABM 的工作过程

3.4.2　标准的数据链路控制协议的初始化

通信双方对使用的通信协议进行初始化是通信过程中的基本问题，不仅在数据链路层，而且在网络层、运输层及其他许多协议中都需要初始化。如果通信过程无异常情况（如节点或链路无故障），那么网络的初始化是比较简单的。在前面的 ARQ 协议的讨论过程中，假定通信双方都已正确进行了初始化，即链路上无分组在传输，则双方的 SN 和 RN 均为 0。但是如果有链路故障或节点故障（或因系统掉电后重新启动）存在，那么初始化问题是比较复杂的。

以链路故障为例，当链路出现故障一段时间后，为了保证端到端传输的可靠性，网络层或运输层通常会采取一定的措施，另外选择一条新的链路来传输在旧链路上未传输的分组。当旧链路恢复工作以后，高层会在该链路中建立一条新的链路来传输新的分组流。链路在正常工作时称为 UP 状态，链路在发生故障时称为 DOWN 状态。

数据链路控制（DLC）协议需在每个 UP 状态的开始点进行初始化。在每个 UP 周期的末尾会出现一些分组已进入 DLC 进行传输，但并没有被对方接收和提交给高层的情况。因此，

DLC 的正常工作应当能保证在每个 UP 周期，接收端提交给高层的分组流是对方（发送端）从高层接收到的分组流中的最前面的分组流，如图 3-21 所示。

当链路状态 UP 和 DOWN 交替时，要使 DLC 能正常工作，就必须使双方对当前链路的状态有一致的看法。下面讨论不同工作模式的初始化过程。

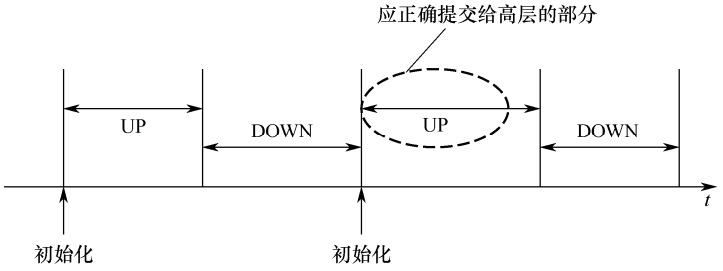


图 3-21　链路的等效工作状态

1．主从模式下的链路初始化

在主从模式（与 NRM 模式相对应）下，简化的链路初始化的过程如图 3-22 所示，采用的是 mod 2 的停等式 ARQ 协议。图 3-22 中，节点 A 为主站，节点 B 为从站。A 决定何时建立链路和拆除链路。A 首先发送初始化命令 INIT（SN =1）进行链路初始化，B 收到 INT 后用 ACKI（RN=0）予以应答。只有当 A 收到 B 的应答后，初始化才能结束。如果 A 在规定的时间内未收到 B 的应答，那么 A 会重发 INIT 命令，直至收到 B 的应答。当 A 决定拆除链路时，A 发送拆除命令 DISC（SN=0），B 收到 A 的命令后，用 ACKD 予以应答。

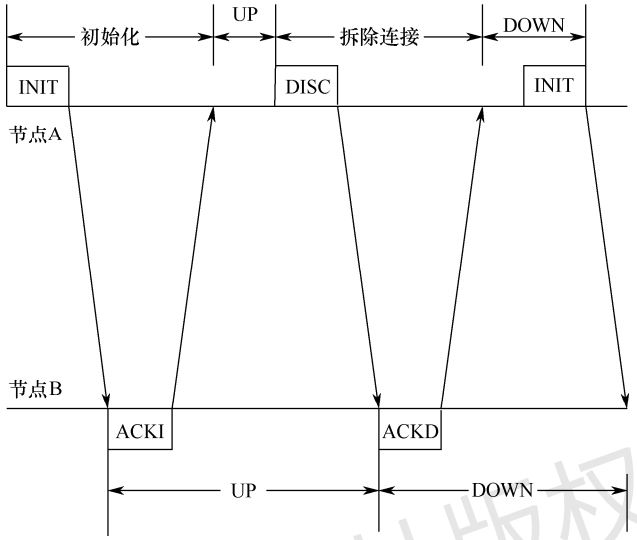


图 3-22　主从模式下简化的链路初始化的过程

从图 3-22 可以看出，对于 A 来说，从 A 发出 INIT 命令到收到 ACKI 为止是初始化阶段。从 A 收到 ACKI 到 A 发送 DISC 命令为止，A 认为链路处于 UP 状态。从 A 发出 DISC 命令到 A 收到 ACKD 为止，是拆除阶段。从 A 收到 ACKD 到 A 再次发送 INIT 命令为止，A 认为链路处于 DOWN 状态。在 DOWN 状态下，A 到 B 将不会传输任何数据。对于 B 来说，从 B 收到 INIT 命令到 B 收到 DISC 命令为止，B 认为链路处于 UP 状态。从 B 收到 DISC 命令

到 B 再次收到 INIT 命令为止，B 认为链路处于 DOWN 状态。当 B 认为链路处于 DOWN 状态时，B 不会发送任何数据。根据停等式 ARQ 可以证明，链路可以正确地进行初始化。

在 HDLC 协议中，SNRM 帧对应于上述的 INIT，DISC 帧对应于上述的 DISC，ACK 帧对应于上述的 ACKI 和 ACKD。

由于在 HDLC 中，对 SNRM 和 DISC 都采用相同的 UA 帧予以应答，因而无法区分是对 SNRM 的应答还是对 DISC 的应答，所以可能导致不正确的操作（如分组丢失）。HDLC 的初始化、数据传输和拆除链路的过程如图 3-23 所示。

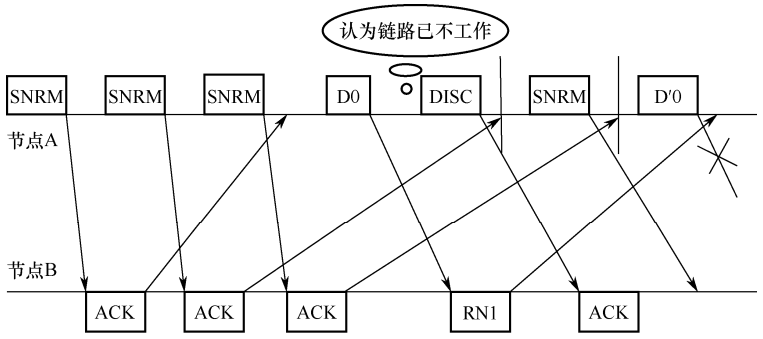


图 3-23　HDLC 的初始化、数据传输和拆除链路的过程

在图 3-23 中，假定传输时延大于发送端等待应答的时延。在图中，A 发送三次 SNRM 才收到 B 的应答。A 在收到 B 的应答后，发送数据序号为 SN=0 的分组 D0。之后若 A 等待一段时间仍未收到 B 的应答，则认为链路已不工作，进而发送 DISC 命令来拆除链路。第二个 ACK（对应于第二次 SNRM 的应答）到达，可认为是对 DISC 的应答，从而使 A 认为链路已拆除。A 时延一定时间后，重新初始化该链路，发出 SNRM 命令。第三个 ACK（对应于第三次 SNRM 的应答）使得 A 认为链路初始化结束，进而发送新的 SN=0 的分组 D0，此时对分组 D0 的应答（RN=1）到达 A，使 A 认为 D′0 已被正确接收。如果 D′0 传输出错，那么将导致 D′0 丢失。该图从概念上说明了 HDLC 应答机制的不完善将有可能导致链路传输出错，因此在进行通信协议设计时，必须对初始化问题进行认真的考虑，否则可能会使所设计的协议不能正常工作。

2．平衡模式下的初始化

在平衡模式中，通信双方是平等的，即当 A 发送数据时，A 是主站，B 是从站；当 B 发送数据时，B 是主站，A 是从站。因而这相当于有两个主从协议在工作。

平衡模式下的初始化过程如图 3-24 所示。在该过程中，应答是嵌入在发送命令之中的。图中符号的含义与图 3-22 中符号的含义相同。链路 UP/DOWN 状态（A→B 及 B→A 的状态）是由 A 和 B 共同确定的。例如，B 在收到对方的 INIT 命令并发出初始化 INIT 命令后，只有在收到对方（A）发来的 ACKI 时，才能认定链路处于 UP 状态。又如，当 A 发出 DISC 后，只有在收到对方的 DISC 命令后，才能认定链路处于 DOWN 状态。而在前面的主从协议中，链路的状态是由主站确定的。

3．在有节点故障时的初始化

节点故障的发生意味着所有与之相连的链路都出现了故障。节点在有故障时，不接收任

何输入，也不产生任何输出，且不执行任何操作。若节点在故障时能记忆其状态，则可以视为数据传输丢失，或高层已将数据改走其他链路，它与链路故障的情况相同。当节点恢复工作时，所有相邻链路都需要进行初始化。若节点在故障时状态丢失，则此时的问题要复杂得多。

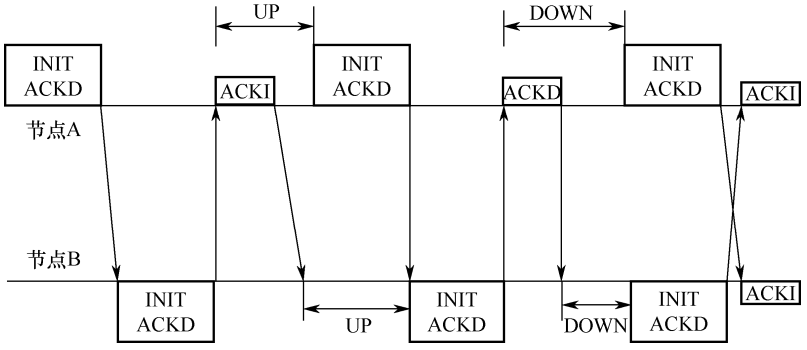


图 3-24　平衡模式下的初始化过程

假定采用主从式初始化协议，节点发生故障时丢失其状态信息，每次节点在从故障状态恢复时，都要进行初始化。假定节点故障的时间和恢复工作的时间与来回传输时延处于同一量级，节点故障及其初始化的过程如图 3-25 所示。在该图中，A 多次出现短时间的故障状态，B 无法判定 A 是否发生故障。A 开始发送 INIT 后出现故障，A 在故障恢复后再次发送 INIT 进行初始化并收到 B 的应答（该应答是对第一次 INIT 的应答），此时 A 认为对方已收到本次 INIT 命令，便开始发送数据分组 D0。A 再次发生故障，A 从故障状态再次恢复后，又发送 INIT 命令。A 在收到 B 的应答（该应答是对第二次 INIT 的应答）后，发送分组 D′0。当 A 收到对 D0 的应答后，会误认为是对 D′0 的应答，如果 D′0 传输出错，那么会导致 D′0 丢失。

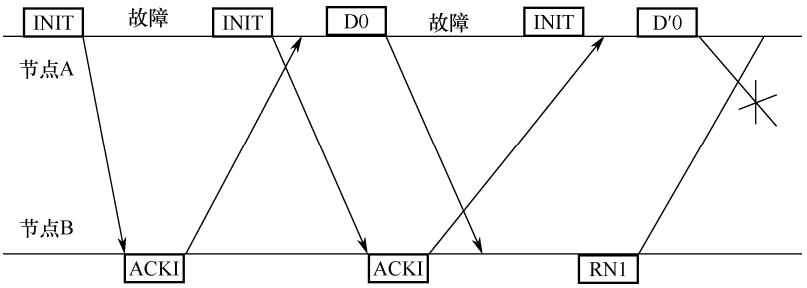


图 3-25　节点故障及其初始化的过程

上面的讨论假定链路的传播时延是没有上限的，因此，无论从什么状态开始及采用何种协议，都有可能发生错误。为了解决节点故障后的初始化问题，可以采用以下几种办法：

（1）采用非易失性的存储器来保存链路的工作状态；

（2）如果链路有一个最大的传播时延，那么可以设计一个足够长的定时器，来避免上述初始化问题；

（3）采用一种随机数的方法来区分正常运行期的不同操作，从而使得不正常操作的发生概率减小。

3.5 网络层的端到端传输

3.4 节研究了两个相邻节点的数据传输的协议，本节将要讨论一个会话过程（session）跨越一个网络中的多条链路，或跨越多个网络的不同传输链路时的分组编号、差错控制、流量控制等问题，以保证任意两个网络节点或两个应用进程之间的可靠的数据传输。

3.5.1 网络层（子网络）的点对点传输协议

1．会话过程和分组的编号

在前面讨论 ARQ 的帧格式时曾指出，为了使数据能正确传输，必须对发送的数据帧和应答帧进行编号，数据帧和应答帧的内容是网络层的一个分组。对于网络的一条链路而言，它通常会被通过该链路的若干会话过程所共享，也就是说，不同的会话过程的分组要共享同一链路。要想将装载在物理帧中的分组送达不同的目的地或区分来自不同源的分组，就必须对不同的会话过程进行标识。标识分组的方法有两种。对于数据报方式，通常应在分组头中包括：源节点和目的节点的地址，以及相同节点中不同会话过程的标识。利用这些信息，可以将任意节点会话过程中的分组送到相应的会话过程。对于虚电路（VC）方式，对不同的会话采用虚电路进行标识，即每个分组含有一个虚电路号。在这两种标识方法中，数据报方式的分组头的开销较大，而虚电路方式的分组头的开销较小，但需有虚电路的建立过程。虚电路方式中不同会话的标识方法如图 3-26 所示。图中，节点 3 与节点 7 之间存在两个 session（A 和 B），session A 由链路（3,5）中的 VC7、链路（5,8）中的 VC4 和链路（8,7）中的 VC11 组成，session B 由链路（3,5）中的 VC13、链路（5,8）中的 VC7 和链路（8,7）中的 VC6 组成。节点 6 与节点 2 之间有一个 session（C），它由链路（6,5）、（5,8）和（8,2）中的 VC3、VC3 和 VC7 组成。

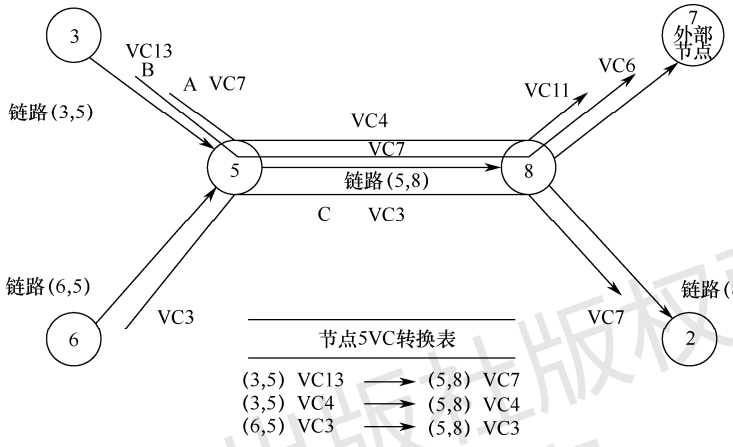


图 3-26 虚电路方式中不同会话的标识方法

不同 session 的分组可以在不同的帧中独立传输，也可以将多个 session 的分组复接在一帧中进行传输。不同 session 的分组复接在一帧中的传输示意图如图 3-27 所示，包括帧头、各 session 头、相应 session 的分组，session 头用于区分不同 session 的分组。

图 3-27　不同 session 的分组复接在一帧中的传输示意图

前面讨论了如何区分不同 session 的分组，那么对于同一 session 发送的分组是否需要进行标识或编号呢？

在数据报方式中，同一 session 的分组可能会经过不同的路径，这样到达目的地的节点顺序就可能与源节点发出分组的节点顺序不同。另外，分组在传输的过程中可能会因链路拥塞、传输错误、节点或链路故障等原因导致回应器分组丢失，因此，必须提供一种方式来使目的节点发现上述问题。解决方法就是对同一 session 发送的分组进行编号。

在虚电路方式中，可能会因下列原因导致分组丢失或传输出错：

（1）虚电路号错误导致不正确的帧通过了 CRC 校验，而把不正确的帧误认为正确的帧；

（2）数据分组中的传输错误未能被 CRC 校验出来；

（3）节点或链路故障可能导致部分分组丢失，如果没有分组编号，那么目的节点就不能发现丢失的分组。

因此，在虚电路方式中，同样需要对同一 session 发送的分组进行编号。分组编号的大小可以采用 L 比特表示，用 mod 2^L 的方式对分组进行循环编号，它可以表示一个 session 的不同分组，也可以表示一个分组传输的第一比特、第一字符或第一字（字由多字节组成）在一个 session 中的编号。

2．网络层的差错控制

网络层的差错控制方式与数据链路层的差错控制方式类似，采用 ARQ 方式，发送端有发送序号 SN，接收端有接收序号 RN。ARQ 的方式包括返回式 ARQ 和选择式 ARQ。

网络层的差错控制与数据链路层的差错控制的主要区别如下。

（1）使用的位置不同。数据链路层的差错控制用于一条物理链路的两端，而网络层的差错控制用于网络中的任意两个节点之间。通常网络中的任意两个节点之间的传输路径由多条链路串联而成。

（2）编号的方式不同。在网络层是对一个 session 的分组（字节或消息）进行统一编号的；而在数据链路层上，是对不同 session 的所有分组（成帧后）进行顺序编号的。

（3）传输顺序的差别。在数据链路层，所有帧都是按顺序传输的；而在网络层，相同源和目的节点的分组可能会经过不同的路径分组传输，从而出现乱序现象。

（4）时延不同。在数据链路层，传输时延（包括传播时延、处理时廷、帧传输的时延）在小范围内变化；而在网络层，传输时延在大范围内变化。

当然，数据链路层和网络层都会出现帧与分组的丢失现象。

由于分组可能会经过不同的路径，从而出现丢失、乱序和任意时延的现象，因此不能保证网络层（和运输层）的差错可以 100%正确恢复。例如，假定分组要用 mod m 的编号方式，则序号为 l mod m 和(l+m) mod m 的分组在接收端是无法区分的。假定这里使用数据报方式，由于分组的时延和丢失现象的存在，有时必须采用端到端的重传，即发送端若在一个给定的时间内没有收到接收端的应答，则发送端重发某一分组（设其序号为 l mod m）。而该重发分组的前一个拷贝可能并没有丢失，可能会因网络的任意时延而潜伏在网络中。发送端继续发

送新的分组直至到达$(l+m) \bmod m$分组，如果潜伏在网络中的序号为$l \bmod m$的分组的一个拷贝先于序号为$(l+m) \bmod m$的分组到达接收节点，那么将会导致不可纠正的错误。尽管这种情况出现的概率很小，但它说明在网络层（或运输层）的差错控制不能确保正确工作。

解决上述问题的办法包括：

（1）网络层最好采用虚电路方式；

（2）分组编号的模值应足够大，可使上述错误出现的概率足够小，使之在可以接受的范围内；

（3）给每个分组规定一个最大的生存时间，在该生存时间内，使分组的序号在使用 mod m 时不可能出现一个循环。

后面将讨论的 TCP 协议采用了后两种方法的组合。前面已经提到，在数据链路层为了检测传输错误，使用了 CRC 校验序列；在网络层为了检测子网中的传输错误和数据链路层中的未检测出的错误，也需要使用某种形式的校验序列。校验序列可以采用 CRC 校验，也可以采用简单的校验方法生成。如将发送信息的 16bit 作为一个字，先求发送序列的每个字的补码再将这些补码相加求和，然后求该和的补码并作为校验序列（该序列的长度为 16bit）。

在分组通过不同的节点时，通常需要对分组头做某些变动（如更改虚电路号），这样每个节点都要重新计算校验序列，这既增大了节点的负荷，又使得校验失去意义。通常的做法是使分组头中的可变部分（如虚电路号）不参加校验，这样每个节点就不需要计算校验和。此外，为了防止分组头出错，导致分组被错误地送到其他节点从而使该节点发生接收错误，可将源节点和目的节点共知的信息（如地址码等）作为信息比特的附加部分，参与校验比特的计算（但这些共知的信息不进行传输），从而防止目的节点出错。

3．网络层的流量控制

流量控制的任务是确保不同接收能力的节点能够在同一网络中工作，即一个快速发送者不能以高于接收者可承受的传输速率来传输数据。流量控制的触发因素是接收端的资源有限，解决方案是控制发送者的传输速率，使其与接收者相匹配。流量控制仅涉及发送者与接收者，例如，假定一台超级计算机的发送速率为 1Gb/s，一台 PC 的接收速率为 100Mb/s，超级计算机通过一个传输容量为 1000Gb/s 的网络向 PC 发送文件。若超级计算机以 1Gb/s 的速率发送，则会在 PC 处产生拥塞，这就需要通过流量控制来降低超级计算机的发送速率，从而适应 PC 的接收速率。

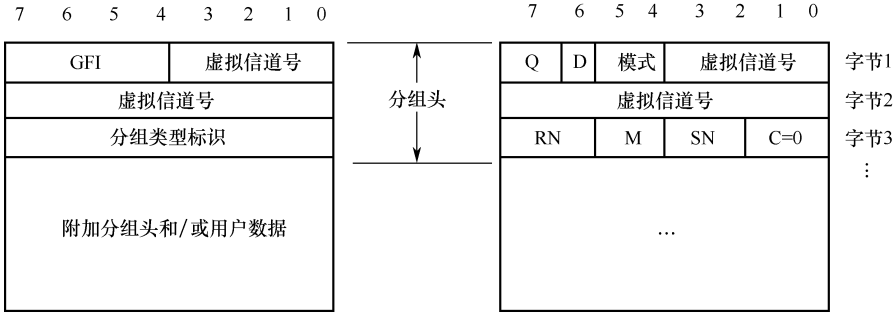
前面讨论的 ARQ 协议（返回 n-ARQ 和选择重发式 ARQ 等）主要用于点对点的差错控制，这里讨论 ARQ 协议如何用于流量控制。ARQ 协议的基本机制是采用滑动窗口（窗口宽度为 n）来限制在一个 session 中发送节点向网络发送的分组数，即第 j 个分组能够被发送给网络的条件是第 $j–n$ 个分组已经被应答。如果网络发生拥塞或传输时延增大，那么应答将被时延，这样信源的发送速率就会降低。因此，利用 ARQ 协议的这种特性，目的节点若想降低接收分组的速率，则可以将含有 RN 的应答分组适当时延再发送。这种控制信源速率的方法称为端到端流量控制。

4．X.25 网络层标准

X.25 网络层标准是由 CCITT（现称为 ITU-T）制定的外部设备（称为数据终端设备 DTE）到网络节点（称为数据通信设备 DCE）之间的标准。其物理层标准称为 X.21，其 DLC 层标

准称为 LAPB，X.25 网络层标准有时称为分组层标准。

X.25 分组有两种：一种是数据分组；另一种是控制分组。其通用格式和数据分组格式如图 3-28（a）和图 3-28（b）所示。其分组头由三字节组成：第一字节的高 4 位是总格式标识（GFI），第一字节的低 4 位和第二字节是虚拟信道号；第三字节是分组类型标识。在 X.25 数据分组格式中，分组头中的第三字节与 DLC 层的格式类似，RN 和 SN 用于表示一个 session 内的分组序号。



（a）通用格式　　（b）数据分组格式

图 3-28　X.25 的分组格式

其他控制比特的含义如下。C 比特用于区分是数据分组还是控制分组，C=0 表示数据分组，C=1 表示控制分组。M 比特表示在一个 session 的传输过程中，当前的分组后面是否还有后续分组，M=1 表示该 session 还有后续分组，M=0 表示这是一个 session 的最后一个分组。虚拟信道号域（12bit）用于表示该分组的虚拟信道号。模式域（2bit）表示 SN 和 RN 采用的模值大小，即表示是 mod 8 还是 mod 128（01 表示 mod 8，10 表示 mod 128）。如果采用 mod 128，那么 RN 和 SN 要扩展为 7bit。D 比特表示应答的类别，D=1 表示端到端应答，即目的节点到源节点的应答，它与 RN 相结合，表示目的节点已正确接收到 RN 以前的所有分组；D=0 表示每条物理链路上的接收节点对发送节点的应答。由于在 DLC 层已有应答，因此采用 RN 的应答功能是多余的，RN 仅用于流量控制。Q 比特是业务类型指标，Q=1 表示是运输层和高层的控制分组；Q=0 表示是数据分组和网络层的控制分组。

X.25 的控制分组包括：呼叫建立分组、流量控制分组、监视分组、证实分组、诊断分组和中断分组。

（1）呼叫建立分组包括 4 类：呼叫请求分组、输入请求分组、接收呼叫分组和呼叫连接分组，这些分组在虚电路的呼叫建立阶段使用。呼叫建立分组应包括源节点和目的节点的地址长度及地址 session 中最大的分组长度、窗口大小、吞吐量协商、逻辑信道号分配、付费等信息。

（2）流量控制分组包括接收准备好（RR）、接收未准备好（RNR）和拒绝接收（REJ）三个控制分组，它类似于 DLC 的监控帧。

（3）监视分组包括重新启动请求/指示分组、清除请求/指示分组和复位请求/指示分组。

（4）证实分组包括重启动证实分组、清除证实分组、复位证实分组和中断证实分组，用于确认前请求的执行情况。

（5）诊断分组用于诊断错误，指示分组被拒绝的原因。

（6）中断分组用于中断网络连接。

X.25 分组交换过程如图 3-29 所示，包括呼叫建立阶段、数据传输阶段和呼叫清除阶段，其数据传输过程类似于 DLC 层。呼叫建立阶段用于确定通信的逻辑链路和通信使用的参数及

初始化双方的工作状态。呼叫清除阶段用于拆除逻辑链路，实际系统的工作情况要比图 3-29 中的复杂。例如，当子网内部因为负载太重而不能建立虚电路，或者目的节点不愿接收呼叫时，网络或目的节点将会向主叫节点发送呼叫清除请求分组，此次呼叫被拒绝。

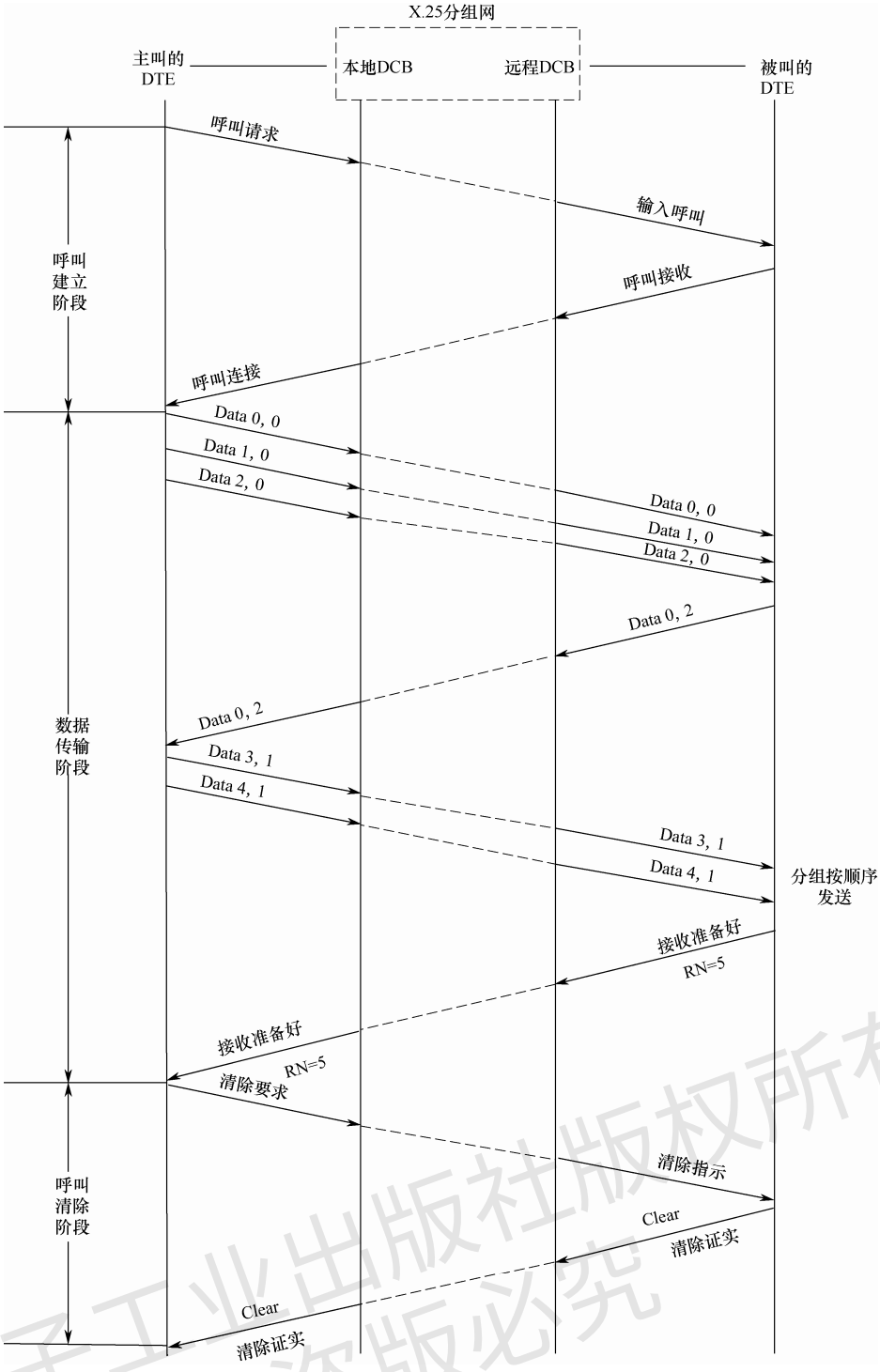


图 3-29 X.25 分组交换过程

3.5.2　网际层（互联层）的传输协议——IP 协议

用户可以使用各种类型的接入网络，要实现任意两个用户之间的通信，就需要将这些网络互联起来。网络互联在一起并进行通信时会遇到许多问题，如不同的寻址方案、不同的最大分组长度、不同的网络接入机制、不同的超时控制、不同的差错恢复方法、不同的状态报告方法、不同的路由选择技术、不同的用户接入控制、不同的服务方式（面向连接服务和无连接服务）、不同的管理与控制方式等。

目前全球最大的、开放的、由众多网络通过路由器采用 TCP/IP 协议族互联而成的是互联网（Internet）。尽管许多网络传输数据报的时延可能是任意的，数据报的传输可能会丢失、重复和乱序，但是互联网使用网际协议 IP（Internet Protocol）尽可能将网络加入，网际采用的是数据报协议传输数据单元，子网内部采用的是数据报方式或虚电路方式。

IP 协议有两个主要版本：IPv4 和 IPv6。IPv4 中的地址长度为 32bit，IPv6 中的地址长度为 128bit。IPv4 的报头是可变长的，而 IPv6 中的报头是固定长度的。IPv4 的地址是分层次的，它由网络号、主机号两部分组成，分为 5 类。不同规模的网络采用不同类型的地址，IPv4 常用的地址类型是 A 类、B 类和 C 类（如图 3-30 所示）：A 类地址用于有大量主机的网络，B 类地址用于中、大规模的网络，C 类地址用于局域网。另外，D 类地址用于广播；E 类地址保留，主要用于实验。

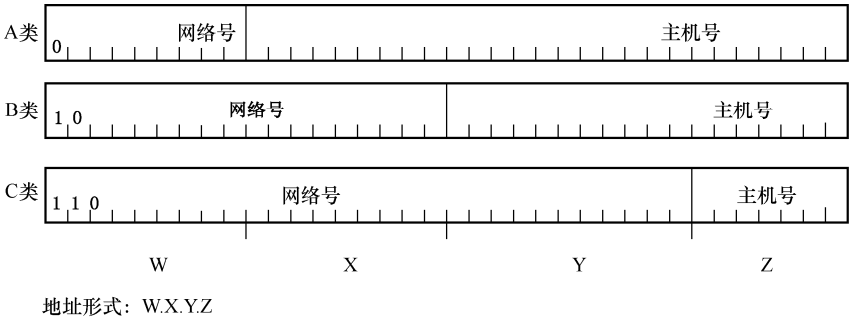


图 3-30　IPv4 常用的地址类型

构造的 IPv4 分组头如图 3-31 所示。图中，版本域表示 IP 的版本（IPv4 或 IPv6）；HL 表示分组头的长度，一般是 4 字节；服务类型表示用户想得到的业务类型，其中优先级占 3bit，D 比特表示要求有更低的时延，T 比特表示要求更大的吞吐量，R 比特表示要求更高的可靠性，C 比特表示要求选择费用更低廉的路由。生存时间（TTL，Time To Live）表示 IP 分组在网络中可生存的时间，IP 分组每经过一个路由器，其 TTL 将被减 1，如果 TTL=0，那么 IP 分组将被丢弃。协议域指示数据报携带的运输层数据使用的是何种协议，如 TCP（6）、UDP（17）、ICMP（1）、GGP（8）、IGP（9）、OSPF（89）等。头校验和用于对头部进行检验，校验的规则是将 IP 头视为以 16bit 字为单位的序列，先将校验和置零，前 5 个 16bit 字相加后，将和的二进制反码写入头校验和字段。

与 IP 协议一起工作的还有三个协议：地址解析协议（Address Resolution Protocol，ARP），逆地址解析协议（Reverse Address Resolution Protocol，RARP）和 Internet 控制报文协议（Internet Control Message Protocol，ICMP）。

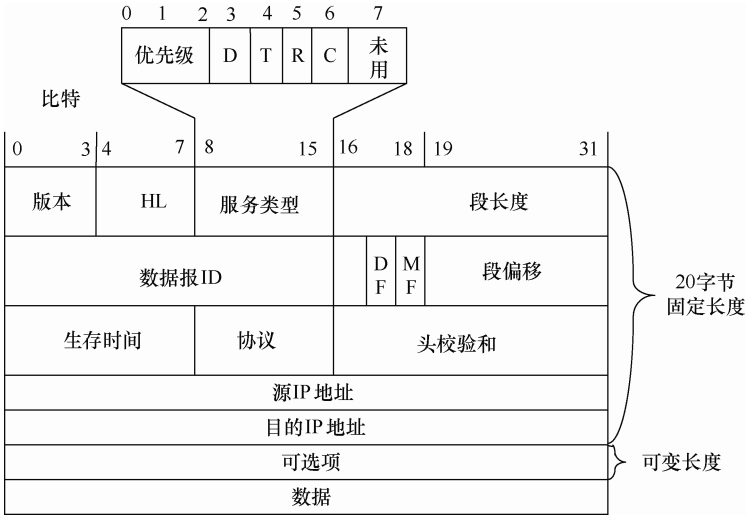


图 3-31 构造的 IPv4 分组头

IPv6 的地址长度为 128 位，采用十六进制数表示，16bit 作为一个基本单元，相互之间采用“：”相连。例如，一个有效的 IPv6 的地址为 4A3F：AE67：F240：56C4：3409：AE52：440F：1403。

3.6 运输层的端到端传输

运输层的功能是将消息分成报文。如果网络层没有合适的差错控制，那么运输层将提供必要的差错恢复。如果网络层没有流量控制，那么运输层将进行流量控制，对 session 进行复接和分接。目前运输层有两个典型的标准协议：TCP 和 OSI TP Class0～TP Class4。这里以 TCP 为例进行介绍。

1．TCP 中的寻址和复接

通常在 TCP 之上有很多用户（或进程），为了区分这些用户（或进程），需要对它们进行编址。TCP 中将 TCP 之上的每个用户（或进程）都称为一个端口（port）（用 16bit 表示）。一些常用的 TCP 端口为：SMTP（25）、FTP（21）、TELNET（23）；一些常用的 UDP 端口为：RPC（111）、SNMP（161）和 TFTP（69）。因此，在 TCP 中，一个完整的地址应当由三部分组成：网络号、主机号、端口号，它们被称为一个套接口（socket）。在 TCP 中，仅将端口号包括在 TCP 报头中，而以参数的形式将网络号和主机号（IP 地址）告知 IP 层，IP 层将 IP 地址放在 IP 头中。具有相同源节点和目的节点的不同端口的消息将复接在 IP 分组中传输，而 IP 本身并不关心 TCP 提交的信息内容。

通过以上的讨论已经看出，寻址问题是通信网络中一个非常重要的问题。不同的层有不同的寻址方式，以 TCP/IP over X.25 网为例，在 TCP 层有 16bit 的端口地址，在 IP 层有 32bit 的 IP 地址，在 X.25 网络层有 12bit 的虚电路号，在数据链路层 LAPB 有 8bit 的地址。每一层都有相应的头，有时这些头中的信息有些多余，但是，对于任意通信网络来说，子网层的头必须能够完成子网内的路由和流量控制，IP 层的头加上子网层的头应能够完成网关（或路由

器）之间的路由，而运输层的头应能够区分复接在一起的不同的 session。

2．TCP 中的差错控制

TCP 中采用的差错控制方式为选择重发式 ARQ。SN 和 RN 的长度为 32bit，这里的 SN 和 RN 不是分组的编号，而是对数据字节进行的编号。SN 表示所传输的分组数据是从当前 session 中的第 SN 字节开始的一段数据，RN 表示希望接收到的分组的第一字节应为当前 session 中的第 RN 字节。如果当前的 SN=m，数据长度为 n，那么下一分组中的 SN=$m+n$。

在 TCP 中，差错恢复主要解决两个问题：一是重发问题；二是连接建立和拆除时的错误问题。重发问题主要是如何确定重发间隔（time-out），这主要是由于 TCP 报文可能会经过不同类型、不同速率的网络，因此传输时延的方差很大。如果重发间隔过小，那么会导致很多报文过早地重发，给网络增加了不应有的负荷。如果重发间隔过大，那么网络的传输效率会降低很多。TCP 采用了一种自适应算法来确定重发间隔，该算法采用式（3-53）来确定报文的平均往返时延 T（往返时延是指报文发送时刻到发送端接收到相应应答时刻之间的时延）

$$T = \alpha \cdot T_{\text{old}} + (1-\alpha) \cdot T_{\text{new}} \tag{3-53}$$

式中，T_{old} 表示旧的往返时延，T_{new} 表示新测得的往返时延，$0 \leqslant \alpha < 1$。α 表示新、旧往返时延对平均往返时延 T 的影响程度，典型的 α 值是 7/8。重发间隔应略大于 T，即

$$\text{重发间隔} = \beta \cdot T \tag{3-54}$$

式中，β 是一个大于 1 的系数。原 TCP 中的推荐值为 β=2，实际上，系数 β 是很难确定的，它必须在传输效率和增加的网络负荷之间取得平衡。该方法的难点在于发送端对第一次发送的分组和重发的分组未进行任何区分，接收端发出的应答也未进行区分。因此，当发送端发出重发分组后，由于网络具有随机时延，因此它将无法确定所收到的应答对应的是哪一次发送的分组（第一次发送还是重发过程的某一分组），这样会导致平均往返时延具有不确定性。为了解决该问题，若分组已进行重发，则不再计算平均往返时延，即仅计算一次发送成功并收到应答的平均往返时延，并以此来计算重发间隔。每重发一次，就将重发间隔增大一次，即

$$\text{重发间隔} = \gamma \times \text{旧的重发间隔} \tag{3-55}$$

式中，γ 的典型值为 2。

在连接建立和拆除过程中，可能会出现如下一些情况：第一次连接拆除后仍有分组到达，这可能导致在第二次连接建立后，会有第一次连接的分组到达，建立连接的分组和拆除连接的分组可能混淆。当网络中一个节点出现故障无法跟踪某一连接时，若该连接仍存在于另一个节点中，则也将导致混淆。TCP 为了解决上述问题，在每次连接开始时都进行初始化，使 SN 和 RN 同步。具体实现的方法如下：源节点和目的节点都有一个 32bit 的时钟计数器，其值每隔 4μs 增大一次（节点间的时钟计数器不同步）。当要建立连接时，发送端（A 节点）以本地的时钟计数器的值作为初始阶段的第一个命令分组的序号（SN），接收端（B 节点）在响应该指令时，对发送端的序号进行响应，即将 B 节点的 RN′置为 SN 并发送给对方（A 节点）。B 采用的初始化 SN′为其本地的时钟计数值，B 在收到对方（A 节点）的应答前不能传输数据。A 节点在接收到 B 节点的 SN′后，将 RN 置为 SN′并发送给对方，TCP 的初始化过程如图 3-32 所示。采用这种随机选择序号的方法，可以避免新、旧连接的序号混淆。32bit 的计

数器循环一圈大约需要 4.6h。

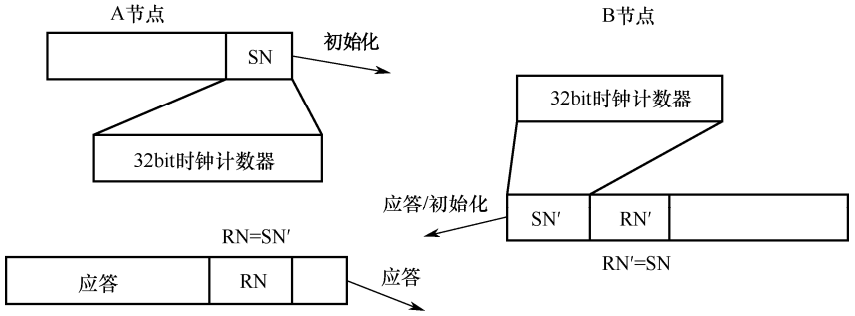


图 3-32　TCP 的初始化过程

根据上面的讨论，可得 TCP 的报文格式如图 3-33 所示。

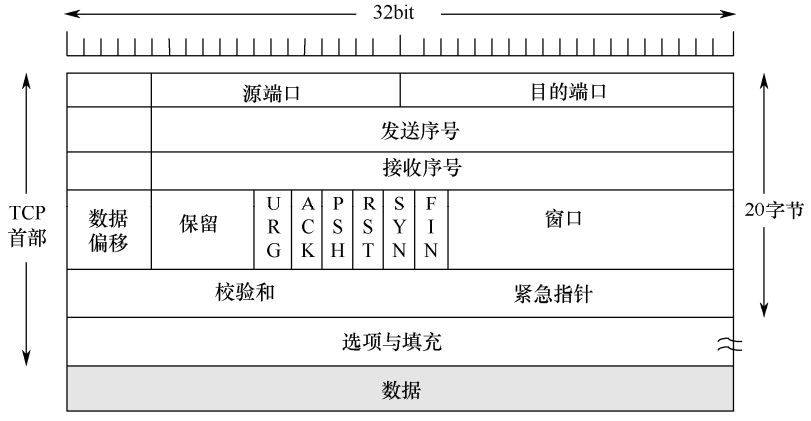


图 3-33　TCP 的报文格式

在图 3-33 中，源端口和目的端口是运输层向高层提供的服务接口，采用 16bit 表示。数据偏移占 4bit，表示在该报文中数据开始点离 TCP 报文段起始点的距离，它实际上是 TCP 报文头的长度。控制域用 6bit 表示，用于对建立和释放连接、应答与报文提交方式等动作进行控制。其中，URG=1 表示此报文应尽快发送，它与紧急指针配合使用，指明紧急数据的长度；ACK=1 表示确认序号字段有意义；PSH=1 表示请求远端 TCP 将本报文段立即传输给其应用层；RST=1 表示要重新建立连接；SYN 和 ACK 组合使用，表示发送建立连接请求和应答，SYN=1 及 ACK=0 表示建立连接的请求报文，SYN=1 及 ACK=1 表示同意建立连接的应答报文；FIN=1 表示要释放一个连接。窗口占 2 字节，表示接收端可接收数据的窗口大小，即告诉对方在未收到应答前可发送的最大数据长度。校验和是对报文头的校验。紧急指针表示在报文段中紧急数据的最后一字节的序号。

3．TCP 的流量控制

流量控制需要考虑两个方面的问题：一是接收者的缓冲区容量大小；二是网络的容量和通过量。假定网络的容量和通过量较大而接收者的缓冲区容量相对较小，这时如果发送端发送的业务量较大，那么会使接收缓冲区溢出而导致报文丢失。如果网络已经发生拥塞、通过量已经很小，那么这时若发送端仍保持较高的发送速率，则会进一步加剧网络的拥塞、减小

通过量，从而导致发送端的报文丢失。

为了解决接收者缓冲区的溢出问题，TCP 采用了窗口允许机制。TCP 报文格式中采用了 16 bit 的窗口域，该窗口称为通知窗口（Advertised Window），用来通知发送端在未收到应答时可以发送的最大字节数，即发送端可以发送序号为 RN 到 RN 加窗口值范围内的数据字节。

在网络传输过程中，有两种原因可能会导致分组丢弃：一是传输出错；二是网络拥塞。由于发送端很难区分这两种情况，因此 Internet 中的 TCP 协议假定分组丢弃都是由网络拥塞引起的。分组的丢弃将导致发送端超时重发，为了控制网络的拥塞，在 TCP 中还引入了第二控制窗口——拥塞窗口。拥塞窗口长度（发送端一次可发送的字节数）是根据网络的拥塞情况动态调整的。发送端真正使用的窗口（称为发送窗口）长度等于通知窗口长度和拥塞窗口长度的最小值，即发送窗口长度=min(通知窗口长度,拥塞窗口长度)。

TCP 采用慢启动（slow-start）、拥塞避免（Congestion Avoidance）和加速递减等技术来进行拥塞控制。发送端在传输一次（发送端将发送窗口中的报文段全部发完，并且收到了对该报文段的所有确认）后，根据拥塞窗口的大小调整一次发送窗口。为了实现控制过程，TCP 引入了一个门限窗口。当拥塞窗口长度大于门限窗口长度时，发送端会降低发送速度，以避免网络拥塞。具体的拥塞窗口的控制算法举例如下。

TCP 的拥塞窗口的控制算法如图 3-34 所示，设初始状态是通知窗口长度为 64KB，门限窗口长度为 64KB，拥塞窗口长度为 64KB。假定在初始时就发生了传输超时，算法将第 0 次传输时的拥塞窗口长度调整为 1KB（最大的分段长度），门限窗口长度减小到传输超时前拥塞窗口长度的一半，即 32KB。第 0 次传输后，每传输一次，拥塞窗口长度增大为原来的 2 倍（按指数增加窗口长度），在拥塞窗口长度等于门限窗口长度（第 5 次传输）后，每传输一次，拥塞窗口长度仅线性增加一个最大的分段长度。若再次发生超时（第 13 次传输），则下次传输时将门限窗口长度减小到拥塞（第 13 次传输）时门限窗口长度的一半，即 20KB，并将拥塞窗口长度调整为 1KB。

图 3-34 中的拥塞窗口长度从 1KB 增大到门限窗口长度的过程为慢启动过程，即每次拥塞后，拥塞窗口长度都降为 1KB，使报文慢慢注入网络。从门限窗口长度开始，拥塞窗口线性增大到最大值的过程称为拥塞避免过程。在前面的慢启动过程中，指数增大变为线性增大是为了避免网络再次出现拥塞。当网络再次出现拥塞时，采用加速递减的方法将门限窗口长度减小到拥塞时发送窗口长度的一半，拥塞窗口长度降为 1KB。

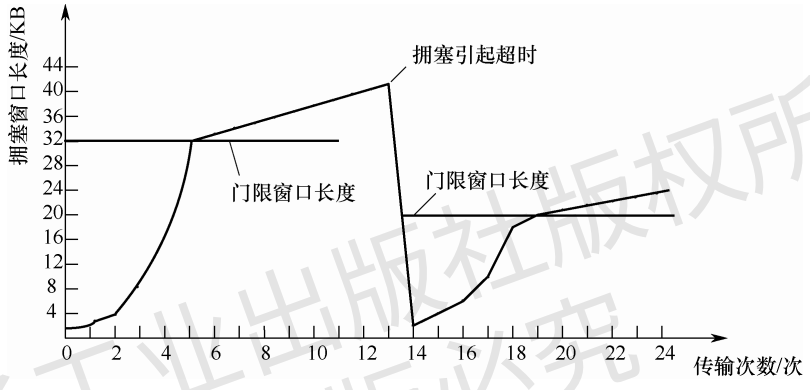


图 3-34　TCP 的拥塞窗口的控制算法

采用上述流量控制方法可使 TCP 的性能得到明显改善。但是当 TCP 用于无线信道时，由于经常会因传输错误而导致发送端超时，因此这时若仍采用上述流量控制方法，则会使得 TCP 传输效率下降，此时需要对 TCP 进行适当的改进。

习 题 3

3.1 设随机过程 $X(t)$定义为 $X(t)=2\cos(2\pi t+Y)$，其中 Y 是离散随机变量，且 $P\{Y=0\}=P\left\{Y=\frac{\pi}{2}\right\}=\frac{1}{2}$。试求该过程在 t=1 时的均值和当 $t_1=0$、$t_2=0$ 时的自相关函数值。

3.2 设随机过程 $X(t)$是一个随机相位信号，即 $X(t)=A\cos(\omega_0 t+\theta)$，式中 A 和 ω_0 为常量，θ 是一个均匀分布的随机变量，其概率密度函数为 $f(\theta)=\frac{1}{2\pi}$（$-\pi<\theta<\pi$）。试求 $X(t)$的均值函数和自相关函数，并讨论其平稳性与各态历经性。

3.3 试求 Poisson 过程的均值函数、方差函数和相关函数。

3.4 设某办公室来访的顾客数 $N(t)$组成 Poisson 流，平均每小时来访的顾客为 3 人，求：（1）上午（8 点到 12 点）没有顾客来访的概率；（2）下午（14 点到 18 点）第一位顾客来访的时间分布。

3.5 设有 3 个黑球和 3 个白球，把这 6 个球任意分给甲、乙两人，并把甲拥有的白球数定义为该过程的状态，则 4 种状态为 0, 1, 2, 3。现每次从甲、乙双方各取一球，然后相互交换。经过 n 次交换后，将过程的状态记为 X_n，该过程是否为马氏链？如是，请计算其一步转移概率矩阵，并画出其状态转移图。

3.6 常用的组帧方式有哪几种？哪种方式的传输开销最小？

3.7 接收机收到了一个采用十六进制数表示的字符串：C0 10 36 87 DB DC DB DC DC DD DB DD CO 7C 8D DC DB DC C0，试根据 SLIP 帧格式来恢复接收的帧。

3.8 针对输入序列 11111111101111010，应用比特插入技术给出相应的输出结果。如果接收到的序列为 01111110100111110111110111110110010101111110100111110O，试移去插入的比特，并指出 Flag 的位置。

3.9 令 $g(D)=D^4+D^2+D+1$，$S(D)=D^3+D+1$，求 $\frac{D^4S(D)}{g(D)}$ 的余数。

3.10 对于一个给定的 L 阶生成多项式 $g(D)$和一个给定的数据比特长度 K，假定输入序列除第 i 位为 1 外，其余全部为 0，即 $S(D)=D^i$（$0\leqslant i\leqslant K-1$），其对应的 CRC 结果为 $C^{(i)}(D)=C_{L-1}^{(i)}D^{L-1}+\cdots+C_1^{(i)}D+C_0^i$，试证明：

（1）对于一个任意的数据多项式 $S(D)$，其 CRC 多项式 $C(D)=\sum_{i=0}^{k-1}S_iC^{(i)}(D)$；

（2）令 $C(D)=C_{L-1}D^{L-1}+\cdots+C_1D+C_0$，则 $C_j=\sum_{i=0}^{k-1}S_iC^{(i)}$，$0\leqslant j<L$，此式说明每个 C_j 都是一个奇偶校验比特，也就是说，CRC 校验码是一种奇偶校验码。

3.11 在停等式 ARQ 中，设重发分组之间的间隔为 T（包括分组传输时间、传播时延、等待应答时间和处理时延等），分组正确接收的概率为 P，试证明最大的可传送的分组到达率 $\lambda=PT$。

3.12 一条双向对称无误码的传输链路的传输速率为 64kb/s，单向传播时延为 15ms。设数据帧长为 3200bit，确认帧长度为 128bit，采用停等式 ARQ 协议，忽略处理时延。问：

（1）在仅有单向数据传输业务的情况下，在 820s 内最多可以传输多少帧？

（2）如果双向都有业务传输，且应答帧的传输只能跟在反向数据帧的尾部（格式为：数据帧 应答帧），那么在 820s 内每个方向最多可以传输多少帧？

（3）假设采用返回 n-ARQ，且 n=3，重新计算（1）和（2）的结果。

3.13　HDLC 是如何保证数据透明传输的？HDLC 有几种工作模式？

3.14　一个通信子网内部采用虚电路方式，沿虚电路共有 n 个节点交换机，在节点交换机中为每个方向都设有一个缓冲区，可存放一个分组。在节点交换机之间用停等式 ARQ 协议，并采用以下措施进行拥塞控制。节点交换机只有在收到分组后才发回确认，但条件是：（1）接收端已成功地收到该分组；（2）有空闲的缓冲区。设发送一个分组需时间 T，传输的差错可忽略不计，用户（DTE）和节点交换机（DCE）之间的数据传输时延也可忽略不计。试问：分组交付给目的用户（DTE）的速率最高是多少？

3.15　A 经过 B 向 C 发送数据的过程有 AB 和 BC 两条链路。B 在收到 A 发来的数据时，可以先向 C 转发再向 A 发确认，也可以反过来。也就是说，B 要做的三件事的顺序是“收数据→转发→发确认”或“收数据→发确认→转发”。现假定 B 在做完第二件事后处理机出现故障，内存中所存的信息全部丢失，但很快又恢复了工作。试证明：只有采用端到端发确认信息的方法（C 向 A 发确认信息），才能保证在任何情况下数据都能从 A 经 B 正确无误地交付给 C。

3.16　两个用户（U1 和 U2）通过主机 H（DTE）同 X.25 网建立了虚电路连接，分层到达网络层的时序图如图 3-35 所示。

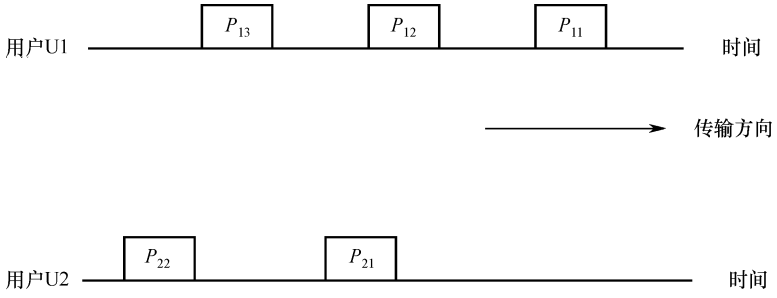


图 3-35　分层到达网络层的时序图

这里 P_{ij} 为从第 i 个用户（i=1, 2）来的第 j 个分组。网络层将虚拟信道号 VC 与发送序号 $P(S)$（网络层的 SN）插入网络层的分组头中。假设 U1 的 VC 为 5，U2 的 VC 为 17，所有分组采用多路复用方式发往数据链路层。数据链路层按顺序将发送序号 $N(S)$（数据链路层的 SN）插入帧头的其他参数中。画出分组在主机 H 与网络间的接口上传送的时序图，按顺序说明每一分组的 $N(S)$、VC、$P(S)$的值。

3.17　一个 TCP 连接使用 256kb/s 链路，其端到端时延为 128ms。经测试，发现吞吐量只有 120kb/s，试问：窗口宽度是多少？

3.18　设 TCP 的拥塞窗口长度为 18KB，当网络发生了超时时，TCP 使用慢启动、加速递减和拥塞避免。设报文段的最大长度为 1KB，试问：拥塞窗口从最小值经过 6 次变化后是多少？

3.19　网络层差错控制与数据链路层差错控制的主要差别是什么？

3.20　ARQ 协议在用于差错控制和流量控制时有何异同？