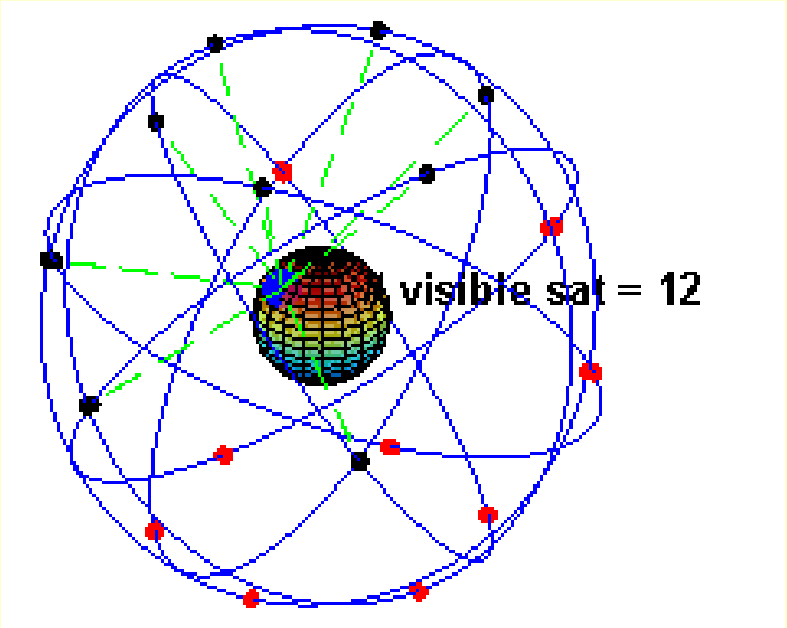
微分模型

北京航空航天大学自动化学院

songxiao@buaa.edu.cn

人造卫星



Animation depicting the orbits of GPS satellites in medium Earth orbit.

§例1 为什么要用三级火箭来发射人造卫星

构造数学模型，以说明为什么不能用一级火箭而必须用多级火箭来发射人造卫星？为什么一般都采用三级火箭系统？

1、为什么不能用一级火箭发射人造卫星？

（1）卫星能在轨道上运动的最低速度

假设：（i） 卫星轨道为过地球中心的某一平面上的圆，卫星在此轨道上作匀速圆周运动。

（ii）地球是固定于空间中的均匀球体，其它星球对卫星的引力忽略不计。

分析：

根据牛顿第三定律，地球对卫星的引力为：$F=\frac{km}{r^2}$

在地面有：$\frac{km}{R^2}=mg$　　得：$k=gR^2$

故引力：$F=mg\left(\frac{R}{r}\right)^2$

R为地球半径，约为6400公里

假设(ii)

卫星所受到的引力也就是它作匀速圆周运动的向心力

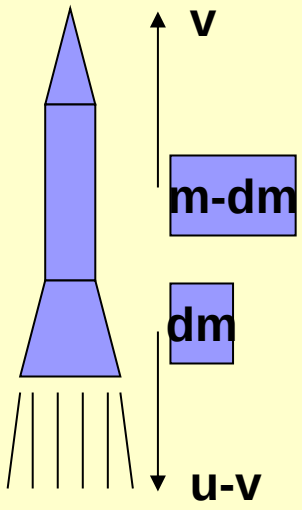
故又有：$F = \frac{m\upsilon^2}{r}$　　从而：$\upsilon = R\sqrt{\frac{g}{r}}$

假设(i)

设g=9.81米/秒^2，得：

卫星离地面高度（公里）	卫星速度（公里/秒）
100	7.86
200	7.80
400	7.69
600	7.58
800	7.47
1000	7.37

考虑t到$t+\Delta t$时刻火箭动量守恒



（2）火箭推进力及速度的分析

假设：空气阻力均不计

分析：记火箭在时刻t的质量和速度分别为$m(t)$和$\upsilon(t)$

有：$m(t+\Delta t)-m(t)=\dfrac{dm}{dt}\Delta t+O(\Delta t^2)$

记火箭喷出的气体相对于火箭的速度为u（常数），

由动量守恒定理：

$$m(t)\upsilon(t)=m(t+\Delta t)\upsilon(t+\Delta t)+\left(-\frac{dm}{dt}\Delta t+O(\Delta t^2)\right)\cdot(\upsilon(t)-u)$$

故：$m\dfrac{d\upsilon}{dt}=-u\dfrac{dm}{dt}$　　由此解得：$\upsilon(t)=\upsilon_0+u\ln\left(\dfrac{m_0}{m(t)}\right)$　　(3.11)

υ_0和m_0一定的情况下，火箭速度$\upsilon(t)$由喷发速度u及质量比决定。

（2）火箭推进力及速度的分析

现将火箭——卫星系统的质量分成三部分：

（i）m_P（有效负载，如卫星）

（ii）m_F（燃料质量）

（iii）m_S（结构质量——如外壳、燃料容器及推进器）。

最终质量为$m_P + m_S$，初始速度为0，

所以末速度：$$\upsilon = u \ln \frac{m_P + m_S + m_F}{m_P + m_S}$$

目前的情况下，只能做到 $m_S \geq \frac{1}{9} m_F$，即 $$\frac{m_S}{m_S + m_F} \geq \frac{1}{10}$$

设 $$\frac{m_S}{m_S + m_F} = \lambda$$ 则燃料燃尽后的末速度为

$$\upsilon = u \ln \frac{m_0}{\lambda m_0 + (1-\lambda) m_p}$$

根据目前的技术条件和燃料性能，u只能达到3千米/秒，即使不带卫星，取 $\lambda = 0.1$，其末速度也不超过7.0公里/秒。 因此目前不可能用单级火箭发射人造卫星。

2、理想过程的实际逼近——多级火箭卫星系统

记火箭级数为n，当第i级火箭的燃料烧尽时，第$i+1$级火箭立即自动点火，并抛弃已经无用的第i级火箭。用m_i表示第i级火箭的质量，m_P表示有效负载。

先作如下假设：

（i）设各级火箭具有相同的λ,即i级火箭中λm_i为结构质量，（1-λ）m_i为燃料质量。

（ii）设燃烧级初始质量与其负载质量之比保持不变，并记比值为k。

考虑二级火箭：

由3.11式，当第一级火箭燃烧完时，其末速度为：

$$\upsilon_1 = u \ln \frac{m_1 + m_2 + m_P}{\lambda m_1 + m_2 + m_P}$$

当第二级火箭燃尽时，末速度为：

$$\upsilon_2 = \upsilon_1 + u \ln \frac{m_2 + m_P}{\lambda m_2 + m_P} = u \ln \left(\frac{m_1 + m_2 + m_P}{\lambda m_1 + m_2 + m_P} \bullet \frac{m_2 + m_P}{\lambda m_2 + m_P} \right)$$

又由假设（ii），$m_2=km_P$，$m_1=k(m_2+m_P)$，代入上式，仍设u=3公里/秒，且为了计算方便，近似取λ=0.1，则可得：

$$\upsilon_2 = 3\ln\left[\frac{\left(\frac{m_1}{m_2+m_P}+1\right)}{\left(\frac{0.1m_1}{m_2+m_P}+1\right)}\bullet\frac{\left(\frac{m_2}{m_P}+1\right)}{\left(\frac{0.1m_2}{m_P}+1\right)}\right] = 3\ln\left(\frac{k+1}{0.1k+1}\right)^2 = 6\ln\left(\frac{k+1}{0.1k+1}\right)$$

要使υ_2=10.5公里/秒，则应使：$\dfrac{k+1}{0.1k+1} = e^{\frac{10.5}{6}} \approx 5.75$

即$k\approx$11.2，而：$\dfrac{m_1+m_2+m_P}{m_P} \approx 149$

类似地，可以推算出三级火箭：

$$\upsilon_3 = u\ln\left(\frac{m_1+m_2+m_3+m_P}{\lambda m_1+m_2+m_3+m_P}\bullet\frac{m_2+m_3+m_P}{\lambda m_2+m_3+m_P}\bullet\frac{m_3+m_P}{\lambda m_3+m_P}\right)$$

在同样假设下：$\upsilon_3 = 3\ln\left(\dfrac{k+1}{0.1k+1}\right)^3 = 9\ln\left(\dfrac{k+1}{0.1k+1}\right)$

3级火箭比二2火箭几乎节省了一半，4级？

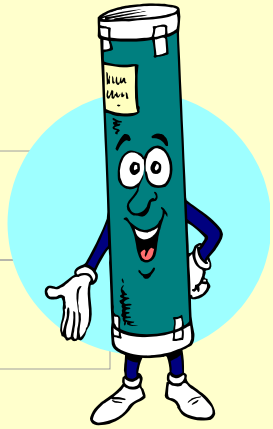
要使υ_3=10.5公里/秒，则$(k+1)/(0.1k+1)\approx 3.21$，$k\approx 3.25$，而$(m_1+m_2+m_3+m_P)/m_P\approx 77$。

考虑N级火箭：

记n级火箭的总质量（包含有效负载m_P）为m_0，在相同的假设下可以计算出相应的m_0/m_P的值，见表3-2

表3-2

n（级数）	1	2	3	4	5	…	∞（理想）
火箭质量/卫星质量	/	149	77	65	60	…	50



- 由于工艺的复杂性及每节火箭都需配备一个推进器，所以使用四级或四级以上火箭是不合算的，三级火箭提供了一个最好的方案。
- 当然若燃料的价钱很便宜而推进器的价钱很贵切且制作工艺非常复杂的话，也可选择二级火箭。

4、火箭结构的优化设计

3中假设(ii)有点理想；现去掉该假设，在各级火箭具有相同 λ 的粗糙假设下，来讨论火箭结构的最优设计。

解条件极值问题：

$$\min\{k_1k_2\cdots k_n\}$$

$$s.t.\ \frac{k_1k_2\cdots k_n}{[\lambda k_1+(1-\lambda)]\cdots[\lambda k_n+(1-\lambda)]}=C$$

或等价地求解无约束极值问题：

$$\min\left\{k_1k_2\cdots k_n-a\left[\frac{k_1k_2\cdots k_n}{[\lambda k_1+(1-\lambda)]\cdots[\lambda k_n+(1-\lambda)]}-C\right]\right\}$$

可以解出最优结构设计应满足：$k_1=k_2=\cdots=k_n$

4、火箭结构的优化设计

3中假设(ii)有点理想；现去掉该假设，在各级火箭具有相同 λ 的粗糙假设下，来讨论火箭结构的最优设计。

记 $W_1=m_1+\ldots+m_n+m_P$

$W_2=m_2+\ldots+m_n+m_P$

……

$W_n=m_n+m_P$

$W_{n+1}=m_P$

应用（3.11）可求得末速度：

$$\upsilon_n = u\ln\left(\frac{W_1}{\lambda m_1+W_2}\cdot\frac{W_2}{\lambda m_2+W_3}\cdots\frac{W_n}{\lambda m_n+W_{n+1}}\right)$$

记 $\dfrac{W_1}{W_2}=k_1,\cdots,\dfrac{W_n}{W_{n+1}}=k_n$

则 $$\upsilon_n = u\ln\left[\frac{\frac{W_1}{W_2}}{\lambda\left(\frac{W_1}{W_2}-1\right)+1}\cdots\cdots\frac{\frac{W_n}{W_{n+1}}}{\lambda\left(\frac{W_n}{W_{n+1}}-1\right)+1}\right]$$

$$= u\ln\frac{k\cdot\cdots\cdots k_n}{[\lambda k_1+(1-\lambda)]\cdots[\lambda k_n+(1-\lambda)]}$$

又 $$\frac{W_1}{W_{n+1}} = \frac{W_1}{W_2}\cdot\frac{W_2}{W_3}\cdots\frac{W_1}{W_{n+1}} = k_1k_2\cdots k_n$$

问题化为，在υ_n一定的条件下，求使$k_1\,k_2\ldots k_n$最小

解条件极值问题：

$$\min\left\{k_1 k_2 \cdots k_n\right\}$$

$$s.t.\frac{k_1 k_2 \cdots k_n}{[\lambda k_1 + (1-\lambda)]\cdots[\lambda k_n + (1-\lambda)]} = C$$

或等价地求解无约束极值问题：

$$\min\left\{k_1 k_2 \cdots k_n - a\left[\frac{k_1 k_2 \cdots k_n}{[\lambda k_1 + (1-\lambda)]\cdots[\lambda k_n + (1-\lambda)]} - C\right]\right\}$$

可以解出最优结构设计应满足：$k_1 = k_2 = \cdots = k_n$

微分方程稳定性模型

北京航空航天大学自动化学院

songxiao@buaa.edu.cn

§例1 军备竞赛

用一个数学模型描述军备竞赛的过程，从定性或定量的角度对竞赛的结果做出解释或预测。

问题分析：用军备这个词表示军事力量的总和。**x(t), y(t)**表示甲乙双方**t** 时刻的军备，包括兵力、装备等。

1、由于互不信任，因此一方军备越大，另一方军备增加越快；

2、由于各方自身经济实力的限制，任一方军备越大，对军备增长的制约作用越大。

3、由于相互敌视或领土争端，每一方都存在增加军备的固有潜力。

§例1 军备竞赛

x(t), y(t)的微分方程：

$$\begin{cases} \dot{x}(t) = -\alpha x + ky + g \\ \dot{y}(t) = -\beta y + lx + h \end{cases} \quad (1)$$

其中的系数均大于或等于0，

- k, l是双方军备刺激程度的度量；
- α，β是己方经济实力制约程度的度量；
- g, h是己方军备竞赛的固有潜力。

§例1 军备竞赛

x(t), y(t)的微分方程：

$$\dot{x}(t) = -\alpha x + ky + g$$

$$\dot{y}(t) = -\beta y + lx + h$$

如果我们关心的是军备竞赛结局由什么因素决定，而不关心竞赛过程，那么只需用微分方程稳定性理论讨论时间充分长以后x(t),y(t)的变化趋势，即上面方程的平衡点的稳定情况。

令方程右端=0，容易算出平衡点为：

$$x_0 = \frac{kh + \beta g}{\alpha\beta - kl}$$

$$y_0 = \frac{lg + \alpha h}{\alpha\beta - kl}$$

§ 例1 微分方程稳定性理论

分析二阶微分方程的稳定性，平衡点是导数为零。
举最简单的特例：线性常系数方程：

$$\dot{x}_1(t) = a_1 x_1 + a_2 x_2$$

$$\dot{x}_2(t) = b_1 x_1 + b_2 x_2$$

系数矩阵A记作：

$$A = \begin{bmatrix} a_1 & a_2 \\ b_1 & b_2 \end{bmatrix}$$

为了研究方程稳定性，假定A的行列式det A !=0

$\dot{x}_1(t) = 0, \dot{x}_2(t) = 0$ 的实根对应平衡点$P_0(x_0,y_0)$,其稳定性由方程的特征方程

$$\det(A - \lambda I) = 0$$

§ 例1 微分方程稳定性理论

或者写成更明晰的方式：

$$\lambda^2 + p\lambda + q = 0$$

$$p = -(a_1 + b_2)$$

$$q = \det A$$

特征根：

$$\lambda_{1,2} = 0.5(-p \pm \sqrt{p^2 - 4q})$$

按照稳定性的定义，当λ_1, λ_2为负数或负实部时，平衡点稳定。
那么，若 **p>0,q>0,**平衡点稳定;
p>0,q<0,平衡点不稳定;
即： **p<0** 或 **q<0,** 平衡点不稳定。

以上是对特例的稳定性结论，对于一般的方程，可以用近似线性分析（**Taylor**展开，取一次项）。

§ 例1 微分方程稳定性理论

方程（1）的系数矩阵为：

$$A=\begin{bmatrix} -\alpha & k \\ l & -\beta \end{bmatrix}$$

于是按照判断平衡点稳定性的方法计算：

$$p=-(a_{11}+a_{22})=\alpha+\beta>0$$

$$q=\det A=\alpha\beta-kl$$

由稳定性准则，当 $\alpha\beta>kl$ 时，平衡点$P_0(x_0,y_0)$,是稳定的。

§例1 军备竞赛

方程（1）的系数均大于或等于0，

- k，l是双方军备刺激程度的度量；
- α，β是己方经济实力制约程度的度量；
- g，h是己方军备竞赛的固有潜力。

$$\begin{cases} \dot{x}(t) = -\alpha x + ky + g \\ \dot{y}(t) = -\beta y + lx + h \end{cases} \quad (1)$$

$$x_0 = \frac{kh + \beta g}{\alpha\beta - kl}$$

$$y_0 = \frac{lg + \alpha h}{\alpha\beta - kl}$$

问题定性分析：

1、当αβ>kl时，平衡点$P_0(x_0,y_0)$,是稳定的，说明当双方的经济约束>双方的刺激程度，平衡点才会趋向稳定；反之军备竞赛将无限进行下去导致战争。

2、如果g=h=0, 则x_0=0，y_0=0是方程（1）的平衡点，并且在 αβ>kl时是稳定的。这说明双方不存在敌视或争端，通过裁军可以达到持久和平。

3、如果g!=h!=0,即使 x_0=0，y_0=0 ，由于x，y导数不为零，也将使双方重整军备。这说明未经和解的裁军是不会持久的。

4. 如果由于某种原因,某方军备大减,比如 x_0=0,那么因为x'=ky+g,也会使该方重整军备.这说明不信任(k!=0)或固有争端(g!=0)的单方面裁军也不会持久.

§例1 微分方程稳定性理论

$$\dot{x}(t) = -\alpha x + ky + g$$
$$\dot{y}(t) = -\beta y + lx + h$$

模型参数的估计，*k, l*：

设x(0)=0,当t较小时，忽略g和-αx的作用，并近似地假定$y=y_1$不变，

由方程(1)得：

$$\dot{x} = ky_1$$

如果当 $t = \tau$ 时$x=y_1$,则：$y_1 = ky_1\tau$

这说明1/k是甲方军备从0赶上乙方军备y_1所需的时间。

例如，德国从1933年开始重整军备，只用了约3年的时间就赶上了它的邻国。假设它增加军备的固有潜力g被制约效应αx所抵消，那么可以认为德国的1/k≈3年，即k≈0.3.

l可以类似地，或者合理地假定它与国家的经济实力成正比，这样若乙国的经济实力是德国的2倍，则可以估计l≈0.6.

§ 例1 微分方程稳定性理论

$$\dot{x}(t) = -\alpha x + ky + g$$
$$\dot{y}(t) = -\beta y + lx + h$$

模型参数的估计，α, β：

设g=0, y=0, 由方程(1)得：

$$x(t) = x(0)e^{-\alpha t}$$

以t=1/α代入，则： $x(\alpha^{-1}) = x(0)e^{-1}$

这说明 t=1/α 是乙方无军备时甲方减少到原来的1/e所需的时间。Richardson认为这大概是一个国家议会的任期，对于任期5年的国家来说，α≈0.2.

§ 例1 微分方程稳定性理论

$$\dot{x}(t) = -\alpha x + ky + g$$

$$\dot{y}(t) = -\beta y + lx + h$$

对模型和参数的粗略检验：考察第一次世界大战前夕，欧洲的两个国家同盟——法俄同盟和德奥匈同盟的军备竞赛情况。

两个同盟的经济实力大致相等，且约为德国的3倍，因为德国的**k≈0.3**，所以这两个同盟的**k=l≈0.9**. 同时假定**α= β ≈0.2**，那么由于**αβ<kl**，它们的 军备竞赛不会趋向稳定。

事实上，当时两个同盟之间既有军备竞赛也有贸易往来，用x_1，y_1表示双方的军事预算，x_2，y_2表示双方的贸易往来，从军事预算中扣除贸易往来昨晚双方的军备，即

$$x = x_1 - x_2, y = y_1 - y_2, k = l, \alpha = \beta$$

$$\frac{d}{dt}(x+y) = (k-\alpha)(x+y) + g + h$$

§ 例1 微分方程稳定性理论

$$\dot{x}(t) = -\alpha x + ky + g$$

$$\dot{y}(t) = -\beta y + lx + h$$

对模型和参数的粗略检验：考察第一次世界大战前夕，欧洲的两个国家同盟——法俄同盟和德奥匈同盟的军备竞赛情况。

两个同盟的经济实力大致相等，且约为德国的3倍，因为德国的**k≈0.3**，所以这两个同盟的**k=l≈0.9**. 同时假定**α= β ≈0.2**，那么由于**αβ<kl**，它们的 军备竞赛不会趋向稳定。

$$\frac{d}{dt}(x_1 + y_1) = (k - \alpha)[(x_1 + y_1) - (x_2 + y_2) + \frac{1}{k - \alpha}\frac{d}{dt}(x_2 + y_2) + \frac{g + h}{k - \alpha}]$$

上式表明x_1+y_1与它的变化率是线性关系。

为了与实际数据比较，书上表1列出了两个同盟 从1909到1913年的军事预算，估值基本相符：

$$\Delta(x_1 + y_1) = 0.73(\overline{x_1 + y_1} - 195)$$

$$k - \alpha = 0.73$$

$$k \approx 0.9, \alpha \approx 0.2$$

由军事预算体现的军备将继续增加。

事实上，军备竞赛终于引发了世界大战。

P与NP问题

- 1 计算复杂度 （时间复杂度）
- 2 P与NP问题
- 3 非确定性图灵机
- 2　P类与NP类语言
- 3　多项式时间验证

计算复杂度（时间复杂度）

- 时间复杂度并不是表示一个程序解决问题需要花多少时间，而是当问题规模扩大后，程序需要的时间长度增长得有多快。
- 对于计算机来说，处理某一个特定算法的效率不易衡量其好坏，而应该看当这个算法的规模变大到数百倍后，程序运行时间是否还是一样，或者也跟着慢了数百倍，或者变慢了数万倍。
- 不管数据有多大，程序处理花的时间始终是那么多的，我们就说这个程序很好，具有O(1)的时间复杂度，也称常数级复杂度；数据规模变得有多大，花的时间也跟着变得有多长，这个程序的时间复杂度就是O(n)，
- **比如找n个数中的最大值；而像冒泡排序、插入排序等，数据扩大2倍，时间变慢4倍的，属于O(n^2)的复杂度。**

计算复杂度

- 还有一些穷举类的算法，所需时间长度成几何阶数上涨，这就是O(a^n)的指数级复杂度，甚至O(n!)的阶乘级复杂度。
- 不会存在O(2*n^2)的复杂度，因为前面的那个“2”是系数，根本不会影响到整个程序的时间增长。同样地，O (n^3 +n^2)的复杂度也就是O(n^3)的复杂度。
- 因此，我们会说，一个O(0.01*n^3)的程序的效率比O(100*n^2)的效率低，尽管在n很小的时候，前者优于后者，但后者时间随数据规模增长得慢，最终O(n^3)的复杂度将远远超过O(n^2)。
- 我们也说，O(n^100)的复杂度小于O(1.01^n)的复杂度。

计算复杂度

- 容易看出，前面的几类复杂度被分为两种级别，其中后者的复杂度无论如何都远远大于前者：一种是O(1), O(log(n)), O(n^a)等，我们把它叫做多项式级的复杂度，因为它的规模n出现在底数的位置；
- 另一种是O(a^n)和O(n!)型复杂度，它是非多项式级的，其复杂度计算机往往不能承受。当我们在解决一个问题时，我们选择的算法通常都需要是多项式级的复杂度，非多项式级的复杂度需要的时间太多，往往会超时，除非是数据规模非常小。

P(Polynomial)类问题

- 一个决策问题D，如果其满足下列条件，则被认为是多项式时间可求解的：
- 1）存在一个算法A，A的输入是D的实例，A总是能正确地输出1或0的答案；
- 2）存在一个多项式函数p，如果D的实例大小为n，则A在不超过p(n)个步骤里终结。

$$f_{EVEN}(n) = \begin{cases} 1, & if\ n\ is\ even \\ 0, & if\ n\ is\ odd \end{cases}$$

算法复杂度？

$m > n$

$\leq n$

$$f_{PRIME}(n) = \begin{cases} 1, & if\ n\ is\ prime \\ 0, & if\ n\ is\ not\ prime \end{cases}$$

NP(Non-deterministically Polynomial)类问题

- 一个决策问题D，如果其满足下列条件，则被认为是非确定性多项式时间可求解的：

In a sense, f_{PRIME} and f_{FACTOR} are also "easy" to compute: we can try division by all possible factors. However, such brute-force factorization of a thousand-digit integer would take millions of years with current technology, so factorization is hardly "easy". (Try factoring 1143816257578888676692357799761466120102182967212423625625618429357069352457338978305971235639587050589890751475992900268795435 41, which has 129 digits.)

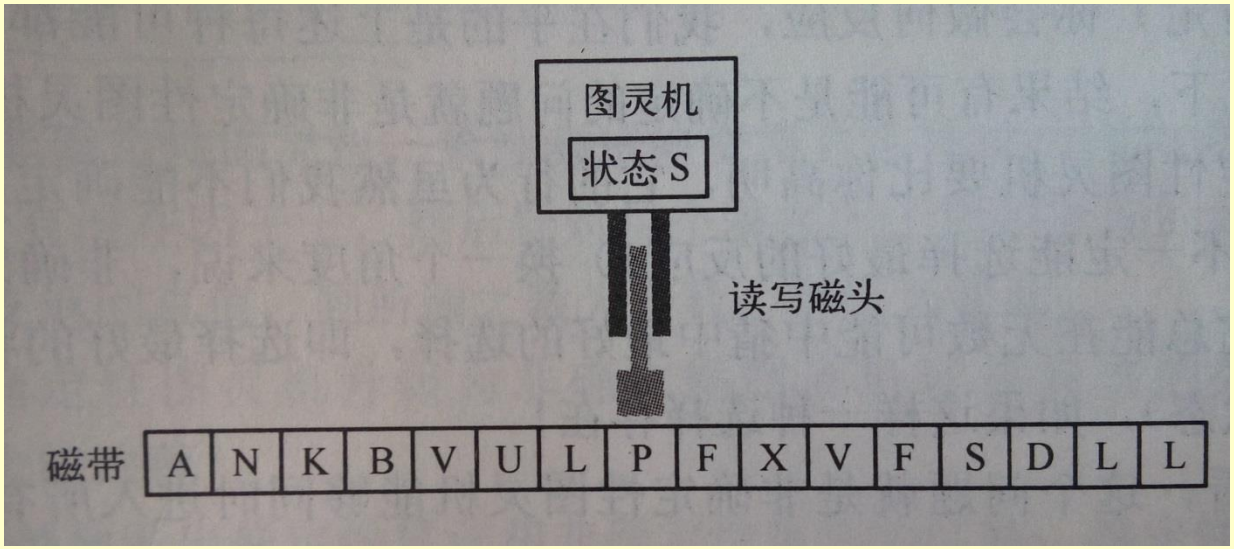
$$f_{FACTOR}(m,n) = \begin{cases} 1, & if\ n\ has\ factor\ such\ that\ 1 < d < m \\ 0, & otherwise \end{cases}$$

P-NP的英文定义

- P is the set of all decision problems for which the instances where the answer is "yes" have efficiently verifiable proofs of the fact that the answer is indeed "yes". More precisely, these proofs have to be verifiable in polynomial time by a **deterministic Turing machine.**
- In an equivalent formal definition, NP is the set of decision problems where the "yes"-instances can be accepted in polynomial time by a non-deterministic Turing machine.
- The equivalence of the two definitions follows from the fact that an algorithm on such a non-deterministic machine consists of two phases, the first of which consists of a guess about the solution, which is generated in a non-deterministic way, while the second consists of a deterministic algorithm that verifies or rejects the guess as a valid solution to the problem.

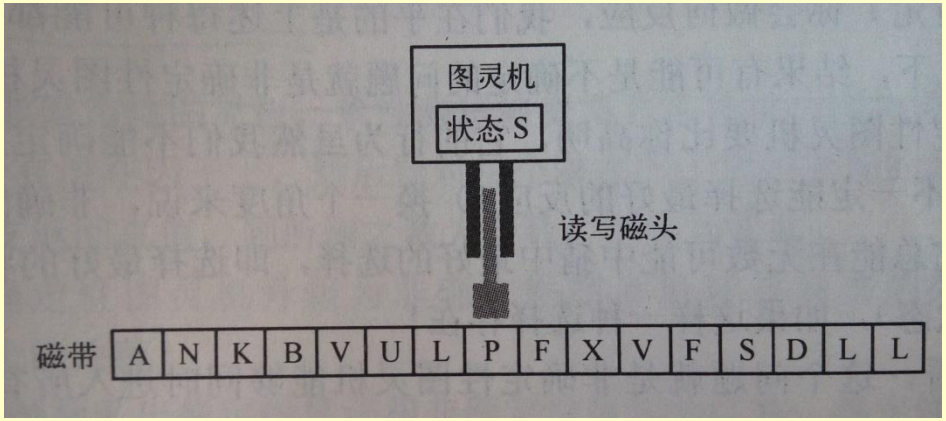
确定性图灵机Deterministic Turing machine， DTM

- 图灵机，自然是拜图灵（英国数学家艾伦-图灵）所赐。他在人们还没有搞清楚什么是图灵机的时候就提出了一个虚无缥缈的图灵机的概念。该虚拟机器（注意，不是今天云计算中的虚拟机）虽然简单，但是功能却极为强大。
- 图灵机简单的说是一个状态机。可以将图灵机抽象成为一个带有很长磁带的机器，机器的磁头在磁带上左右移动。磁头下面的字母为图灵机的输入。



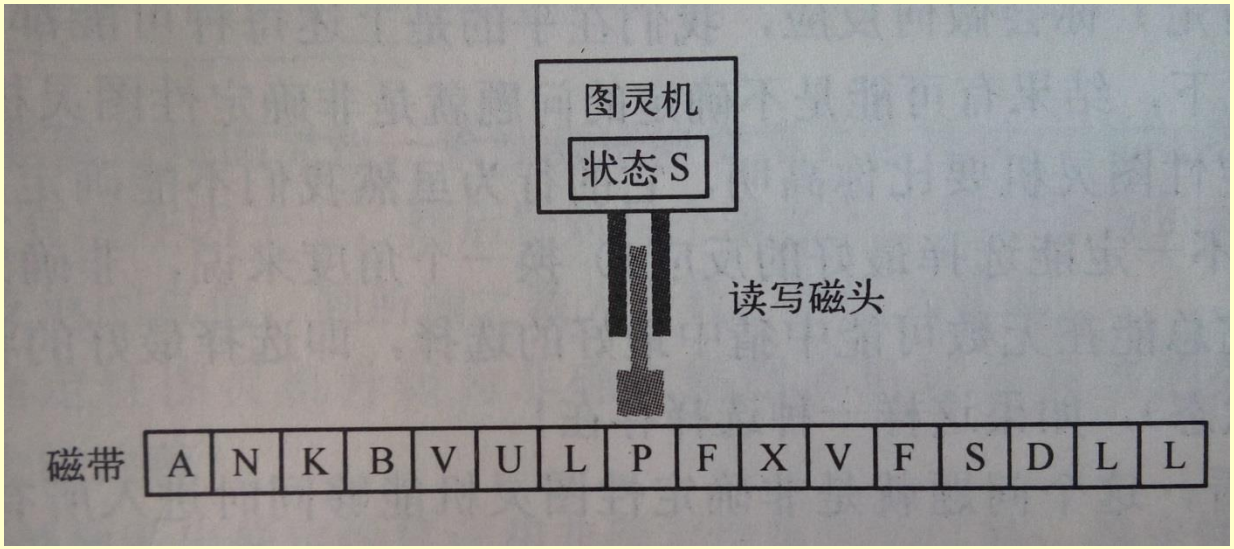
确定性图灵机Deterministic Turing machine， DTM

- 该状态机的状态转换函数具有如下能力：对于给定状态和符号，确定三件事情，即：
  - 输出符号；
  - 磁头移动方向（左或右）；
  - 下一个状态。
- 例如在状态3的时候，如果输入符号为X，则图灵机有可能输出Y，将磁头做一一个位置，进入状态4。换句话说，DTM在给定状态和输入下其行为是唯一确定的。



非确定性图灵机Non-Deterministic Turing machine， NTM

- 非确定性图灵机与DTM的区别是：在给定状态和输入时，其行为将不是唯一确定的。
- 例如在状态3的时候，如果输入符号为X，则NTM有可能输出Y，将磁头做一一个位置，进入状态4；它也有可能输出符号X，将磁头往右移动一位，留在状态3里。
- 换句话说，DTM表述的是因果关系，只能跟踪一条计算路径，NTM表述的是非因果关系，它拥有一个计算树，可以同时跟踪多条计算路径。



A question on computational complexity - P vs NP

- Computing f_{EVEN} and $f_{GREATER}$ is easy. However, it is not obvious that this is so for the following two functions:
- $f_{PRIME}(n) = \begin{cases} 1, if\ n\ is\ prime \\ 0, if\ n\ is\ not\ prime \end{cases}$
- $f_{FACTOR}(m,n) =$
$\begin{cases} 1, \ if\ n\ has\ a\ factor\ d\ such\ that\ 1 < d < m \\ 0, \ otherwise \end{cases}$

8.2 P类与NP类问题

- 8.2.1 非确定性图灵机
- 8.2.2 P类与NP类语言
- 8.2.3 多项式时间验证

8.2.1 非确定性图灵机

在图灵机计算模型中，移动函数δ**是单值的**，即对于$Q\times T^k$中的每一个值，当它属于δ的定义域时，$Q\times(T\times\{L, R, S\})^k$中只有惟一的值与之对应，称这种图灵机为确定性图灵机，简记为**DTM**(Deterministic Turing Machine)。

非确定性图灵机（ **NDTM** ）：一个k带的非确定性图灵机M是一个7元组：$(Q, T, I, \delta, b, q_0, q_f)$。与确定性图灵机不同的是非确定性图灵机允许移动函数**δ具有不确定性**，即对于$Q\times T^k$中的每一个值$(q;x_1, x_2, \ldots, x_k)$，当它属于δ的定义域时，$Q\times(T\times\{L, R, S\})^k$中有惟一的一个**子集**$\delta(q;x_1, x_2, \ldots, x_k)$与之对应。可以在$\delta(q;x_1, x_2, \ldots, x_k)$中随意选定一个值作为它的函数值。

8.2.2 P类与NP类语言

P类和NP类语言的定义：

P={L|L是一个能在**多项式时间内**被一台DTM所接受的语言}

NP={L|L是一个能在**多项式时间**内被一台NDTM所接受的语言}

由于一台确定性图灵机可看作是非确定性图灵机的特例，所以可在多项式时间内被确定性图灵机接受的语言也可在多项式时间内被非确定性图灵机接受。故P $\subseteq$ NP。

8.2.2 P类与NP类语言

NP类语言举例——无向图的团问题。

该问题的输入是一个有n个顶点的无向图G=(V，E)和一个整数k。要求判定图G是否包含一个k顶点的**完全子图(团)**，即判定是否存在V'⊆V，|V'|=k，且对于所有的u，v∈V'，有(u，v)∈E。

若用邻接矩阵表示图G，用二进制串表示整数k，则团问题的一个实例可以用长度为 $n^2+\log k+1$ 的二进位串表示。因此，团问题可表示为语言：

CLIQUE={w#v|w，v∈{0，1}*，以w为邻接矩阵的图G有一个k顶点的团，其中v是k的二进制表示。}

8.2.2 P类与NP类语言

接受该语言CLIQUE的**非确定性算法**：用非确定性选择指令选出包含k个顶点的候选顶点子集V，然后确定性地检查该子集是否是团问题的一个解。算法分为3个阶段：

算法的第一阶段将输入串w#v分解，并计算出n= $\sqrt{|w|}$，以及用v表示的整数k。若输入不具有形式w#v或|w|不是一个平方数就拒绝该输入。显而易见，第一阶段可 $O(n^2)$ 在时间内完成。

在算法的第二阶段中，非确定性地选择V的一个k元子集V'⊆V。

算法的第三阶段是确定性地检查V'的团性质。若V'是一个团则接受输入，否则拒绝输入。这显然可以在 $O(n^4)$ 时间内完成。因此，整个算法的时间复杂性为$O(n^4)$。

非确定性算法在多项式时间内接受语言CLIQUE，故CLIQUE∈NP。

8.2.3 多项式时间验证

多项式时间可验证语言类VP可定义为：

VP={L|L∈Σ*，Σ为一有限字符集，存在一个多项式p和一个多项式时间验证算法A(X，Y)使得对任意X∈Σ*，X∈L当且仅当存在Y∈Σ*，|Y|≤p(|X|)且A(X，Y)=1}。

定理8-5：VP=NP。（证明见书本）

例如(**哈密顿回路问题**)：一个无向图G含有哈密顿回路吗?

无向图G的哈密顿回路是通过G的每个顶点恰好一次的简单回路。可用语言HAM-CYCLE 定义该问题如下：

HAM-CYCLE={G|G含有哈密顿回路}