计算机科学2004Vol. 31№. 1

一种基于算盘的计算模型—算盘机*)

胡海星　宋方敏
(南京大学计算机软件新技术国家重点实验室　南京210093)

摘　要　本文介绍了一种基于算盘的计算模型，定义了“算盘机”和“算盘机可计算”的概念，证明了算盘机的计算能力与递归函数的计算能力相同。

关键词　计算，算盘机，递归函数

A Computational Model Based on the Abacus — Abacus Machine

HU Hai-Xing　SONG Fang-Min
(State Key Lab of Novel Software Technology, Nanjing University, Nanjing 210093)

Abstract　A computational model based on the abacus, Abacus Machine, is introduced in this paper. After the "Abacus Machine" and "Abacus Machine Computable" are defined, it is shown that the computing power of Abacus Machine is identical to that of Recursive Functions.

Keywords　Computing, Abacus Machine, Recursive Function

1. 引言

中国是算盘的故乡，在计算机已被普遍使用的今天，古老的算盘不仅没有被废弃，反而因其灵便、准确等优点，在许多国家方兴未艾。因此，人们往往把算盘的发明与中国古代四大发明相提并论，认为算盘也是中华民族对人类文明的一大贡献。

关于算盘的记载，最早见于东汉末年(献帝建安元年，即公元198年左右)徐岳所著的《数术记遗》一书。该书记录了十四种算法，第十三种即称“珠算”，其文曰：“控带四时，经纬三才”。北周数学家甄鸾对这段文字作了注释：“刻板为三分，其上下二分以停游珠，中间一分，以定其位，位各五珠，上一珠与下四珠色别。其上别色之珠当五，其下四珠各当一。至下四珠所领，故云控带四时；其珠游于三方之中，故云经纬三才也”。这段文字所描述的计算器具，被认为是中国算盘的原型。

西方学者对中国的算盘一直很推崇。早在20世纪60年代，英国学者就提出过基于算盘的计算模型，George S. Boolos等人将该模型称为“算盘机”(Abacus Machine)，并证明了其计算能力与图灵机等价[1]。但在中国目前尚未见到算盘机在任何学术著作中出现。本文介绍了一种算盘机，这是Cohen教授提出的算盘机[2]的变种。我们将形式化地定义“算盘机”和“算盘机可计算”的概念，并证明算盘机的计算能力与递归函数的计算能力相同。

2. 算盘机的定义

为了便于研究算盘的计算能力，我们只考虑具有以下特征的理想算盘：

·理想算盘由无限多的“档”组成，这些档从左到右依次标号为1,2,3,…且不分上下档；

·理想算盘的每一档上可以放置任意多的算珠；

·假设有无限多的备用算珠供理想算盘使用。

事实上，上述理想算盘更接近于《数术记遗》中提到的算盘原型。下面我们根据理想算盘形式化地定义算盘机。

定义1(算盘机的字母表)　算盘机的字母表包括：①A_1，A_2，…，A_n，…，$n\in\mathbb{N}^+$；②S_1，S_2，…，S_n，…，$n\in\mathbb{N}^+$；③(和)$_1$，)$_2$，…，)$_n$，…，$n\in\mathbb{N}^+$，即一个左括号和无穷多个带下标的右括号。

定义2(算盘的格局)　设$\Sigma=\{\xi|\xi=(x_1,x_2,\cdots,)\in\mathbb{N}^\infty$且$\exists\ k\in\mathbb{N}^+,\forall\ i>k, x_i=0\}$，$\xi\in\Sigma$为算盘的一个格局。若$\xi=(x_1,x_2,\cdots)$，表示算盘的第1档上有$x_1$个算珠，第2档上有$x_2$个算珠，……，第$i$档上有$x_i$个算珠。对于$\xi\in\Sigma$，我们用符号$[\xi]_i$表示第$i$档上的算珠数目$x_i$。

定义3(算盘机)　算盘机AM(Abacus Machine)可归纳定义如下：①$A_i, S_i\in AM, i\in\mathbb{N}^+$；②$M_1, M_2\in AM\Rightarrow M=M_1M_2\in AM$；③$M\in AM\Rightarrow(M)_k\in AM, k\in\mathbb{N}^+$；④AM仅限于此。

定义4　设$M\in AM$为一算盘机，$\xi=(x_1,x_2,\cdots)$为算盘的一个格局，我们用$\xi M=\eta$表示算盘机M作用在格局ξ上计算得到格局η。该计算过程可根据M的结构归纳定义如下：

·$(x_1,\cdots,x_{i-1},x_i,x_{i+1},\cdots)A_i=(x_1,\cdots,x_{i-1},x_i+1,x_{i+1},\cdots)$

·$(x_1,\cdots,x_{i-1},x_i,x_{i+1},\cdots)S_i=$

$$\begin{cases}(x_1,\cdots,x_{i-1},x_i-1,x_{i+1},\cdots) & ,x_i>0\\(x_1,\cdots,x_{i-1},0,x_{i+1},\cdots) & ,x_i=0\end{cases}$$

·$\xi M_1M_2=((\xi M_1)M_2)$

·$\xi(M)_k=\xi M^t=\xi(\underbrace{M\cdots M}_{t\uparrow M})$，其中$t=\mu i.([\xi M^i]_k=0$

定义4实际上给出了算盘机的操作语义。根据上述定义，我们知道：

1. A_i表示在第i档上添加1个算珠；

2. S_i表示从第i个档上抹去1个算珠，如果该档上原来没有算珠则不进行任何操作；

*)本文受国家自然科学基金资助。

3. 算盘机按照左结合进行复合运算；

4. $(M)_k$ 表示当第 k 档上的算珠数目不为0时重复地执行操作 M，直到第 k 档上的算珠数目达到0时为止。更具体地，$(M)_k$ 可定义如下：

$$\xi(M)_k=\begin{cases}\xi & ,[\xi]_k=0\\ \xi M^t & ,[\xi]_k\neq 0\text{且}\exists\ t\geqslant 1,[\xi M^t]_k=0,\\ & \text{其中 } t=\mu i.([\xi M^i]_k=0)\\ \text{无定义} & ,\text{否则}\end{cases}$$

定义4中用到了符号 M^t，我们可以严格地定义该符号如下：

定义5 $I=A_1S_1$

$M_i^0=I$

$M^{i+1}=M^iM$

因为 $I=A_1S_1$ 表示在第1档上添加1个算珠然后再将其抹掉，所以 I 是一个恒等算子。

定义6 设 $n\in\mathbb{N}$，$f:\mathbb{N}^n\to\mathbb{N}$，$M\in AM$，M 定义 f 是指 $f(\vec{x})$ 有定义且 $f(\vec{x})=y$ 当且仅当 $(\vec{x},0,0,\cdots)M$ 有定义且 $(\vec{x},0,0,\cdots)M=(y,0,0,\cdots)$。对于函数 $f:\mathbb{N}^n\to\mathbb{N}$，若存在一个算盘机 $M\in AM$ 定义 f，则称函数 f 是算盘机可计算的。

定义7 对于任何 $M\in AM$，归纳定义 $\rho(M)\in\mathbb{N}^+$ 如下：①$\rho(A_i)=i,i\in\mathbb{N}^+$；②$\rho(S_i)=i,i\in\mathbb{N}^+$；③$\rho(M_1M_2)=\max\{\rho(M_1),\rho(M_2)\}$；④$\rho((M)_k)=\max\{k,\rho(M)\},k\in\mathbb{N}^+$。

定理1 设 $\xi,\eta\in\Sigma$，$M\in AM$，若 $\xi M=\eta$，则 $\forall\ i>\rho(M)$，$[\eta]_i=[\xi]_i$。

定理2 设 $\xi_1,\xi_2\in\Sigma$，$M\in AM$，且 $\forall\ i\leqslant\rho(M)$，$[\xi_1]_i=[\xi_2]_i$。若 $\xi_1M=\eta_1$，$\xi_2M=\eta_2$，则

$$[\eta_2]_i=\begin{cases}[\eta_1]_i, & i\leqslant\rho(M)\\ [\xi_2]_i, & i>\rho(M)\end{cases}$$

定理1和定理2很容易通过对 M 的结构做归纳证明。上述定理表明，对于任何 $M\in AM$，在计算过程中只使用到第1至第 $\rho(M)$ 档的数据；换句话说，在执行 M 时，M 对 $\rho(M)$ 档之后数据不起作用。

下面我们给出几个算盘机可计算函数的例子。

例1(清零函数) 设 $Z(x)=0$，则算盘机 $Z=(S_1)_1$ 定义了函数 Z。更一般地，$Z_k=(S_k)_k$ 定义了清空算盘第 k 档的清零函数。

例2(后继函数) 设 $Succ(x)=x+1$，则算盘机 $Succ=A_1$ 定义了函数 $Succ$。

例3(前驱函数) 设 $Pred(x)=\begin{cases}x-1 & ,x>0\\ 0 & ,x=0\end{cases}$，则算盘机 $Pred=S_1$ 定义了函数 $Pred$。

例4(加法) 设 $Add(x_1,x_2)=x_1+x_2$，则算盘机 $Add=(A_1S_2)_2$ 定义了函数 Add。

例5(减法) 设 $Sub(x_1,x_2)=\begin{cases}x_1-x_2 & ,x_1\geqslant x_2\\ 0 & ,x_1<x_2\end{cases}$，则算盘机 $Sub=(S_1S_2)_2$ 定义了函数 Sub。

例6 设算盘机 $DC_{p,q}=(S_pA_q)_p$，对于 $\zeta\in\Sigma$，设 $\xi DC_{p,q}=\eta$，则

$$[\eta]_i=\begin{cases}0 & ,i=p\\ [\xi]_q+[\xi]_p & ,i=q\\ [\xi]_i & ,\text{否则}\end{cases}$$

换句话说，$DC_{p,q}$ 的作用就是将算盘的第 p 档上的算珠全部移动到第 q 档上去。

例7 设 $N(x)=\begin{cases}0,x\neq 0\\ 1,x=0\end{cases}$，则算盘机 $N=A_2(S_2S_1)_1DC_{2,1}$ 定义了函数 N。

例8(投影函数) 设 $U_i^n:\mathbb{N}^n\to\mathbb{N}$，$U_i^n(x_1,\cdots,x_n)=x_i$，则算盘机 $U_i^n=Z_1\cdots Z_{i-1}Z_{i+1}\cdots Z_nDC_{i,1}$ 定义了函数 U_i^n。

例9 设算盘机 $C_{p,q,r}=(S_pA_qA_r)_p(S_rA_p)_r$，对于 $\xi\in\Sigma$，若 $[\xi]_q=[\xi]_r=0$，设 $\xi C_{p,q,r}=\eta$，则

$$[\eta]_i=\begin{cases}[\xi]_p, & i=p\\ [\xi]_p, & i=q\\ [\xi]_i, & \text{否则}\end{cases}$$

换句话说，如果原来算盘的第 q,r 档上的值为0，则 $C_{p,q,r}$ 可以将第 p 档上的值利用第 r 档加到第 q 档上，且保持第 p 档上的值不变。

例10(乘法) 设 $Mul(x_1,x_2)=x_1\times x_2$，则算盘机 $Mul=DC_{1,3}(C_{3,1,4}S_2)_2Z_3$ 定义了函数 Mul。

例11 设算盘机 $E_{p,q,r}=DC_{p,r}DC_{q,p}DC_{r,q}$，对于 $\xi\in\Sigma$，若 $[\xi]_r=0$，设 $\xi E_{p,q,r}=\eta$，则

$$[\eta]_i=\begin{cases}[\xi]_q, & i=p\\ [\xi]_p, & i=q\\ [\xi]_i, & \text{否则}\end{cases}$$

换句话说，如果原来算盘的第 r 档上的值为0，则 $E_{p,q,r}$ 可以利用第 r 档交换第 p 档和第 q 档的值。

例12 设函数 $Sel(x_1,x_2,x_3)=\begin{cases}x_2,x_1=0\\ x_3,x_1\neq 0\end{cases}$，则算盘机 $Sel=(E_{2,3,4})_1Z_1Z_3DC_{2,1}$ 定义了函数 Sel。

3. 算盘机可计算等价于递归可计算

3.1 递归函数

定义8(本原函数) 本原函数包括以下三种函数：①$Z:\mathbb{N}\to\mathbb{N}$ 定义为 $Z(x)=0$；②$Succ:\mathbb{N}\to\mathbb{N}$ 定义为 $Succ(x)=x+1$；③设 $n\in\mathbb{N}$，$k\in\{1,2,\cdots,n\}$，则 k 阶投影函数 $U_i^n:\mathbb{N}^n\to\mathbb{N}$ 定义为 $U_i^n(x_1,\cdots,x_n)=x_k$。

定义9(函数的复合) 设 $m,n\in N$，$f:\mathbb{N}^m\to\mathbb{N}$，$g_1,\cdots,g_m:\mathbb{N}^n\to\mathbb{N}$，则 f 与 $g_1,\cdots,g_m$ 的复合记作 $\mathrm{Comp}[f,g_1,\cdots,g_m]$，表示函数 $h:\mathbb{N}^n\to\mathbb{N}$，且 $h(\vec{x})=f(g_1(\vec{x}),\cdots,g_m(\vec{x}))$。

定义10(原始递归) 设 $m\in\mathbb{N}$，$f:\mathbb{N}^n\to\mathbb{N}$，$g:\mathbb{N}^{n+2}\to\mathbb{N}$，则 f 和 g 的原始递归函数记作表示函数 $\mathrm{Prim}[f,g]$，表示函数 $h:\mathbb{N}^{n+1}\to\mathbb{N}$，且 $h(\vec{x},0)=f(\vec{x})$；$h(\vec{x},y+1)=g(h(\vec{x},y),\vec{x},y)$。

定义11(极小化算子) 设 $m\in\mathbb{N}$，

·设谓词 $R\in\mathbb{N}^{m+1}$，如果 $\forall\ \vec{x}\in\mathbb{N}^m,\exists\ y\in\mathbb{N},(R(\vec{x},y))$，则称 R 是正则的。

·设 $R\in\mathbb{N}^{m+1}$ 是正则谓词，设 $\vec{x}\in\mathbb{N}^m$，用 $\mu y.(R(\vec{x},y))$ 表示集合 $\{y\in N|R(\vec{x},y)\}$ 中的最小元素。

·设 $f:\mathbb{N}^{m+1}\to\mathbb{N}$，如果 $\forall\ \vec{x}\in\mathbb{N}^m,\exists\ y\in\mathbb{N},(f(\vec{x},y)=0)$，则称 f 是正则的。

·设 $f:\mathbb{N}^{m+1}\to\mathbb{N}$ 是正则的，f 的极小化算子记作 $\mathrm{Us}[f]:\mathbb{N}^m\to\mathbb{N}$，定义为 $\mathrm{Us}[f](\vec{x})=\mu y.(f(\vec{x},y)=0)$。

定义12(递归函数) 所有递归可计算函数组成的集合 REC 定义为：①$Z,Succ,U_i^n\in\mathrm{REC}$；②REC 对于 Comp，Prim，Us 算子闭合。

3.2 递归可计算函数是算盘机可计算的

定理3 本原函数是算盘机可计算的。

证明：根据例1，例2，例8 直接可得。 □

在后文我们将用符号 $\prod_{i=1}^{r}M_i$ 表示算盘机 $M=M_1M_2\cdots$

M_r。

定理4 设 $m,n\in\mathbb{N}$，$f:\mathbb{N}^m\to\mathbb{N}$，$g_1,\cdots,g_m:\mathbb{N}^n\to\mathbb{N}$，若 $f,g_1,\cdots,g_m$ 都是算盘机可计算的，则 $\mathrm{Comp}[f,g_1,\cdots,g_m]$ 也是算盘机可计算的。

证明：设算盘机 $F,G_1,\cdots,G_m\in AM$ 分别定义了函数 $f,g_1,\cdots,g_m$，设

$k=\max\{\rho(F),\rho(G_1),\cdots,\rho(G_m)\}+1$

我们按照如下算法计算函数 $\mathrm{Comp}[f,g_1,\cdots,g_m]$：

1. 将第1，…，n 档上的 n 个输入参数分别转移到第 $k+1$，…，$k+n$ 档；

2. 对于 $i=1,2,\cdots,m$ 重复做以下操作：

a)将第 $k+1,\cdots,k+n$ 档上的数据利用第 k 档分别复制到第1，…，n 档；

b)执行 G_i。根据定理2，执行结束后第1档上的值是 $g_i(\vec{x})$；第2，…，$k-1$ 档上的值是0；第 $k,k+1,\cdots$ 档上的值保持不变；

c)将第1档上的计算结果转移到第 $k+n+i$ 档；

3. 将第 $k+n+1,\cdots,k+n+m$ 档上的数据分别转移到第1，…，m 档；

4. 清除第 $k+1,\cdots,k+n$ 档上的数据；

5. 执行 F；

6. 执行完上述操作后，第1档上的值就是 $f(g_1(\vec{x}),\cdots,g_l(\vec{x}))$。

综上所述，设

$$H_1=\prod_{i=1}^{n}DC_{1,k+i}$$

$$H_2=\prod_{i=1}^{m}\left(\prod_{j=1}^{n}C_{k+j,j,k}G_iDC_{1,k+n+i}\right)$$

$$H_3=\prod_{i=1}^{m}DC_{k+n+i,i}$$

$$H_4=\prod_{i=1}^{n}Z_{k+i}$$

$$H=H_1H_2H_3H_4F$$

其中 $k=\max\{\rho(F),\rho(G_1),\cdots,\rho(G_m)\}+1$。则算盘机 H 定义了函数 $\mathrm{Comp}[f,g_1,\cdots,g_m]$。□

定理5 设 $f:\mathbb{N}\to\mathbb{N}$，$g(n)=f^n(0)$ 称为 f 的迭置，若 f 是算盘机可计算的，则 g 也是算盘机可计算的。

证明：设 $F\in AM$ 定义了函数 f，设 $k=\rho(F)+1$，我们按照如下算法计算函数 g：

1. 将第1档上的输入参数 n 转移到第 k 档上，第1档的值变为0；

2. 当第 k 档不为0时重复下列操作：

a)将第1档上的数据作为输入，执行 F。根据定理2，执行结束后第1档上的值是 $f(x)$，这里 x 是上一次循环结束时第1档上的值；第2，…，$k-1$ 档上的值是0；第 $k,k+1,\cdots$ 档上的值保持不变；

b)执行 S_k；

3. 上述循环结束时，第1档上的值就是 $g(n)$。

综上所述，设 $G=DC_{1,k}(FS_k)_k$，其中 $k=\rho(F)+1$，则算盘机 G 定义了函数 g。□

定理6 设 $f:\mathbb{N}\to\mathbb{N}$，设 $g:\mathbb{N}^2\to\mathbb{N}$ 定义为

$g(x,0)=x$

$g(x,n+1)=f(g(x,n))$

即 $g(x,n)=f^n(x)$。f 称为函数 f 的带参复迭。若 f 是算盘机可计算的，则 g 也是算盘机可计算的。

证明：类似定理5，设 $F\in AM$ 定义了函数 f，设 $k=\rho(F)+1$，则算盘机 $G=DC_{2,k}(FS_k)_k$ 定义了函数 g。□

引理7 用本原函数，函数复合，迭置和带参复迭可以生成一切原始递归函数。

该引理的证明参见文[3]。根据定理3，定理4，定理5，定理6 和引理7直接可得如下推论：

推论8 一切原始递归函数都是算盘机可计算的。

定理9 设 $n\in\mathbb{N}$，$f:\mathbb{N}^{n+1}\to\mathbb{N}$ 是正则的，$g:\mathbb{N}^n\to\mathbb{N}$ 且 $g=\mathrm{Us}[f]$，若 f 是算盘机可计算的，则 g 也是算盘机可计算的。

证明：设 $F\in AM$ 定义了函数 f，设 $k=\rho(F)+1$。我们将按照如下算法来计算函数 g：

1. 将第1，…，n 档上的 n 个输入参数分别转移到第 $k+1$，…，$k+n$ 档；

2. 执行 A_1；

3. 当第1档上的值不为0时重复做下列操作：

a)清空第1档的数据；

b)将第 $k+1,\cdots,k+n,k+n+1$ 档上的数据利用第 k 档分别复制到第1，…，$n,n+1$ 档；

c)执行 F。根据定理2，执行结束后第1档上的值是 $f(\vec{x},y)$，其中 $\vec{x}$ 是算盘机的 n 个初始输入，y 是当前第 $k+n+1$ 档的值；第2，…，$k-1$ 档上的值是0，第 $k,k+1,\cdots$ 档上的值保持不变；

d)执行 A_{k+n+1}；

4. 将第 $k+n+1$ 档上的数据转移到第1档上；

5. 执行 S_1；

6. 清空第 $k+1,\cdots,k+n$ 档上的数据；

7. 执行完上述操作后，第一档上的值就是 $g(\vec{x})$。

综上所述，设

$$G_1=\prod_{i=1}^{n}DC_{i,k+i}$$

$$G_2=\prod_{i=1}^{n+1}C_{k+i,i,k}$$

$$G_3=\prod_{i=1}^{n}Z_{k+i}$$

$$G=G_1A_2(Z_1G_2FA_{k+n+1})_1DC_{k+n+1,1}S_1G_3$$

其中 $k=\rho(F)+1$，则算盘机 G 定义了函数 g。□

根据定理3，定理4，推论8，定理9以及定义12，直接可得如下定理：

定理10 一切递归可计算函数都是算盘机可计算的。

3.3 算盘机可计算函数是递归可计算的

定义13 设 $m,n\in\mathbb{N}^+$，$F:\mathbb{N}^n\to\mathbb{N}^m$，若存在递归可计算函数 $f_1,f_2,\cdots,f_m:\mathbb{N}^n\to\mathbb{N}$，使得 $\forall\vec{x}\in\mathbb{N}^n$，$F(\vec{x})$ 有定义且 $F(\vec{x})=(y_1,\cdots,y_m)$ 当且仅当 $f_1(\vec{x}),\cdots,f_m(\vec{x})$ 有定义且 $f_1(\vec{x})=y_1,\cdots,f_m(\vec{x})=y_m$，即

$F(\vec{x})=(f_1(\vec{x}),\cdots,f_m(\vec{x}))$

则称函数 F 是递归可计算的。

定义14 设 $M\in AM$，$k=\rho(M)$，设 $F:\mathbb{N}^k\to\mathbb{N}^k$ 是递归可计算的，且 $F(\vec{x})=(f_1(\vec{x}),\cdots,f_k(\vec{x}))$，称函数 F 递归定义了算盘机 M 是指对于任何格局 $\xi\in\Sigma$，若 ξM 有定义且 $\xi M=\eta$，则满足

$f_i([\xi]_1,[\xi]_2,\cdots,[\xi]_k)=[\eta]_i$，$i=1,2,\cdots,k$

即 $F([\xi]_1,[\xi]_2,\cdots,[\xi]_k)=([\eta]_1,[\eta]_2,\cdots,[\eta]_k)$。若存在这样

的函数 F 递归定义算盘机 M，则称算盘机 M 是递归可定义的。

定理11 设 $n\in\mathbb{N}$，函数 $f:\mathbb{N}^n\to\mathbb{N}$ 是算盘机可计算的，且 $M\in AM$ 定义了 f。若算盘机 M 是递归可定义的，则函数 f 是递归可计算的。

证明：设 $k=\rho(M)$，$F:\mathbb{N}^k\to\mathbb{N}^k=(f_1,\cdots,f_k)$ 设递归定义了 M，其中 $f_1,\cdots,f_k:\mathbb{N}^k\to\mathbb{N}$ 都是递归可计算函数。设 $g,g_1,g_2,\cdots,g_k:\mathbb{N}^n\to\mathbb{N}$ 分别定义为：

$$g_i=\mathrm{Comp}[f_i,U_1^n,U_2^n,\cdots,U_n^n,\underbrace{Z(U_1^n),\cdots,Z(U_1^n)}_{\text{总共}k-n\text{个}}],$$

$$i=1,2,\cdots,k$$

$$g=\mathrm{Comp}[U_1^k,g_1,\cdots,g_k]$$

则 g 是递归可计算的且 $\forall\vec{x}\in\mathbb{N}^n$，$f(\vec{x})$ 有定义 $\Rightarrow g(\vec{x})$ 有定义且 $g(\vec{x})=f(\vec{x})$。所以函数 f 也是递归可计算的。□

定理12 算盘机 $A_k,k\in\mathbb{N}^+$ 是递归可定义的。

证明：根据定义7，$\rho(A_k)=k$，设 $F:\mathbb{N}^k\to\mathbb{N}^k=(f_1,\cdots,f_k)$，其中

$$f_i=\begin{cases}U_i^k & ,i<k\\ \mathrm{Comp}[Succ,U_k^k] & ,i=k\end{cases}$$

则 $f_i,i=1,2,\cdots,k$ 都是递归可计算的，所以函数 F 就递归定义了算盘机 A_k。□

引理13 例3中所定义的前驱函数 $Pred$ 是递归可计算函数。

证明：设 $H=\mathrm{Prim}[Z,U_3^3]$；$Pred=\mathrm{Comp}[H,U_1^1,U_1^1]$，则函数 $Pred$ 是递归可计算的。□

定理14 算盘机 $S_k,k\in\mathbb{N}^+$ 是递归可定义的。

证明：根据定义7，$\rho(S_k)=k$，设 $F:\mathbb{N}^k\to\mathbb{N}^k=(f_1,\cdots,f_k)$，其中

$$f_i=\begin{cases}U_i^k & ,i<k\\ \mathrm{Comp}[Pred,U_k^k] & ,i=k\end{cases}$$

则 $f_i,i=1,2,\cdots,k$ 都是递归可计算的，所以函数 F 就递归定义了算盘机 S_k。□

定义15 设 $p,q,r\in\mathbb{N}^+$，设 $F:\mathbb{N}^r\to\mathbb{N}^q$，$G:\mathbb{N}^p\to\mathbb{N}^r$，$F,G$ 都是递归可计算的且 $F=(f_1,\cdots,f_q)$，$G=(g_1,\cdots,g_r)$。其中 $f_1,\cdots,f_q:\mathbb{N}^r\to\mathbb{N}$，$g_1,\cdots,g_r:\mathbb{N}^p\to\mathbb{N}$ 都是递归可计算函数。设 $R:\mathbb{N}^p\to\mathbb{N}^q=(r_1,\cdots,r_q)$ 且 $r_i:\mathbb{N}^p\to\mathbb{N}=\mathrm{Comp}[f_i,g_1,\cdots,g_r],i=1,2,\cdots,q$ 为递归可计算函数。用符号表示为 $R=\mathrm{Comp}[F,G]$。显然，函数 R 也是递归可计算的，且 $R(\vec{x})=F(G(\vec{x}))$。

定理15 若算盘机 $M_1,M_2\in AM$ 是递归可定义的，则算盘机 $M=M_1M_2$ 也是递归可定义的。

证明：设 $p=\rho(M_1)$，$q=\rho(M_2)$，$n=\rho(M)=\max\{p,q\}$。设 $F=(f_1,\cdots,f_p)$，$G=(g_1,\cdots,g_q)$ 分别递归定义了算盘机 M_1，M_2。令 $F'=(f_1',\cdots,f_n')$，$G'=(g_1',\cdots,g_n')$。其中

$$f_i'=\begin{cases}\mathrm{Comp}[f_i,U_1^n,\cdots,U_p^n] & ,1\leqslant i\leqslant p\\ U_i^n & ,p<i\leqslant n\end{cases}$$

$$g_i'=\begin{cases}\mathrm{Comp}[g_i,U_1^n,\cdots,U_q^n] & ,1\leqslant i\leqslant q\\ U_i^n & ,q<i\leqslant n\end{cases}$$

对于 $\xi=(x_1,\cdots,x_n,0,0,\cdots)\in\Sigma$，设 $\xi M_1=\delta=(y_1,\cdots,y_n,0,0,\cdots)$，因为 F 递归定义了 M_1，所以

$$y_i=\begin{cases}f_i(x_1,\cdots,x_p) & ,1\leqslant i\leqslant p\\ x_i & ,p<i\leqslant n\end{cases}$$

即 $y_i=f'_i(x_1,\cdots,x_n),i=1,2,\cdots,n$

设 $\delta M_2=\eta=(z_1,\cdots,z_n,0,0,\cdots)$，因为 G 递归定义了 M_2，所以

$$z_i=\begin{cases}g_i(y_1,\cdots,y_p) & ,1\leqslant i\leqslant q\\ y_i & ,q<i\leqslant n\end{cases}$$

即 $z_i=g_i'(y_1,\cdots,y_n)=g_i'(F'(x_1,\cdots,x_n)),i=1,2,\cdots,n$

因此，设 $(x_1,\cdots,x_n,0,0,\cdots)M=(z_1,\cdots,z_n,0,0\cdots)$

则 $(z_1,\cdots,z_n)=G'(F'(x_1,\cdots,x_n))$

根据定义14 知函数 $H=\mathrm{Comp}[F',G']$ 递归定义了算盘机 $M=M_1M_2$。□

引理16 设 $m,n\in\mathbb{N}$，$F:\mathbb{N}^n\to\mathbb{N}^m$ 是递归可计算的，设 $g:\mathbb{N}^n\to\mathbb{N}$ 定义为 $g(\vec{x})=U_i^m(F(\vec{x}))$，即 $g=\mathrm{Comp}[U_i^m,F]$，则 g 是递归可计算函数。

证明：根据定义13，设 $F=(f_1,\cdots,f_m)$，则 $g=\mathrm{Comp}[U_i^m,F]=f_i$ 是递归可计算函数。□

引理17 设 $f:\mathbb{N}\to\mathbb{N}$，定义 $h:\mathbb{N}^2\to\mathbb{N}$ 为 $h(x,0)=x$，$h(x,t+1)=f(h(x,t))$，即 $h(x,t)=f^t(x)$，记作 $h=\mathrm{Iter}[f]$。若 f 是递归可计算函数，则 h 也是递归可计算函数。

证明：令 $h(x,0)=U_1^1(x)$，$h(x,t+1)=\mathrm{Comp}[f,U_3^3](h(x,t),x,t)$，则显然 h 是递归可计算函数。□

引理18 设 $n\in\mathbb{N}^+$，$F:\mathbb{N}^n\to\mathbb{N}^n$，定义 $H:\mathbb{N}^{n+1}\to\mathbb{N}^n$ 为 $H(\vec{x},0)=\vec{x}$，$H(\vec{x},t+1)=F(H(\vec{x},t))$，即 $H(\vec{x},t)=F^t(\vec{x})$。若 F 是递归可计算的，则 H 也是递归可计算的。

证明：设 $P:\mathbb{N}^n\to\mathbb{N}$，$P_1,\cdots,P_n:\mathbb{N}\to\mathbb{N}$ 都是递归可计算函数，其中 P 是某个递归可计算的配对函数，且

$$P_i(P(x_1,\cdots,x_n))=x_i,i=1,2,\cdots,n \tag{1}$$

$$\vec{x}=(P_1(P(\vec{x})),P_2(P(\vec{x})),\cdots,P_n(P(\vec{x}))) \tag{2}$$

因为 F 是递归可计算的，设 $F=(f_1,\cdots,f_n)$，其中 $f_1,\cdots,f_n:\mathbb{N}^n\to\mathbb{N}$ 都是递归可计算函数。令

$$g_i:\mathbb{N}\to\mathbb{N}=\mathrm{Comp}[f_i,P_1,\cdots,P_n],i=1,2,\cdots,n$$

$$G:\mathbb{N}\to\mathbb{N}=\mathrm{Comp}[P,g_1,\cdots,g_n]$$

显然 G 是一个递归可计算函数。下面我们来证明

$$\forall t\in\mathbb{N},P(F^t(\vec{x}))=G^t(P(\vec{x})) \tag{3}$$

1. 对于 $t=0$，(3)式显然成立。

2. 假设(3)式对于 $t=k$ 成立，设 $P(F^k(\vec{x}))=G^k(P(\vec{x}))=X$，则对于 $t=k+1$，有：

$$\begin{aligned}F^{k+1}(\vec{x})&=F(F^k(\vec{x}))\\&=F(P_1(P(F^k(\vec{x}))),\cdots,P_n(P(F^k(\vec{x}))))\\&\qquad\text{根据(2)式}\\&=F(P_1(X),\cdots,P_n(X))\\&=(f_1(P_1(X),\cdots,P_n(X)),\cdots,f_n(P_1(X),\cdots,P_n(X)))\\&=(g_1(X),\cdots,g_n(X))\quad\text{根据 }g_i\text{ 的定义}\end{aligned}$$

所以

$$\begin{aligned}P(F^{k+1}(\vec{x}))&=P(g_1(X),\cdots,g_n(X))\\&=G(X)\quad\text{根据 }G\text{ 的定义}\\&=G(G^k(P(\vec{x})))\\&=G^{k+1}(P(\vec{x}))\end{aligned}$$

综上所述，(3)式对于所有 $t\in\mathbb{N}$ 成立。

要证明 $H(\vec{x},t)=F^t(\vec{x})$ 是递归可计算的，只需证明存在递归可计算函数 $h_1,\cdots,h_n:\mathbb{N}^{n+1}\to\mathbb{N}$，使得 $H(\vec{x},t)=F^t(\vec{x})=(h_1(\vec{x},t),\cdots,h_n(\vec{x},t))$。令

$$\begin{aligned}h_i(\vec{x},t)&=P_i(P(F^t(\vec{x})))\quad\text{根据(1)式}\\&=P_i(G^t(P(\vec{x})))\quad\text{根据(3)式}\\&=P_i(\mathrm{Iter}[G](P(\vec{x}),t))\end{aligned}$$

因为 $P,P_i,G,\mathrm{Iter}[G]$ 都是递归可计算函数，所以 h_i 也是递归可计算函数。因此 H 是递归可计算的。□

定理19 若算盘机 M∈AM 是递归可定义的，则算盘机 $(M)_k, k\in\mathbb{N}^+$ 也是递归可定义的。

证明：设 $p=\rho(M), n=\rho((M)_k)=\max\{p,k\}$，设函数 $F=(f_1,\cdots,f_p)$ 递归定义了算盘机 M。令 $G=(g_1,\cdots,g_n)$，其中

$$g_i=\begin{cases}\mathrm{Comp}[f_i,U_1^n,U_2^n,\cdots,U_p^n] & ,1\leqslant i\leqslant p\\ U_i^n & ,p<i\leqslant n\end{cases}$$

显然，G 也是递归可计算的，且对于 $\xi=(x_1,\cdots,x_n,0,0,\cdots)\in\Sigma$，若 $\xi M=\eta$，则 $[\xi]_i=g_i(x_1,\cdots,x_n), i=1,2,\cdots,n$

令 $H:\mathbb{N}^{n+1}\to\mathbb{N}^n$ 定义为 $H(\vec{x},y)=G^y(\vec{x})$，根据引理18，H 也是递归可计算的。令 $t:\mathbb{N}^n\to\mathbb{N}$ 定义为 $t=\mathrm{Us}[\mathrm{Comp}[U_1^n,H]]$，即 $t(\vec{x})=\mu y.(U_1^n(H(\vec{x},y))=0)$，根据引理16，$\mathrm{Comp}[U_1^n,H]$ 是一个递归可计算函数，所以 t 也是递归可计算函数。$t(\vec{x})$ 得到的是使得 $G^y(\vec{x})$ 的第 k 个分量为0的最小的 y。定义 $K:\mathbb{N}^n\to\mathbb{N}=(U_1^n,\cdots,U_n^n,t)$，显然 K 是递归可计算的。设 $W:\mathbb{N}^n\to\mathbb{N}^n=\mathrm{Comp}[H,K]$，则

$$W(\vec{x})=H(K(\vec{x}))=H(\vec{x},t(\vec{x}))$$
$$=G^{t(\vec{x})}(\vec{x})\quad \text{其中 } t(\vec{x})=\mu y.(U_1^n(G^y(\vec{x}))=0)$$

W 也是递归可计算的，根据 $(M)_k$ 的定义知 W 递归定义了 $(M)_k$。□

定理20 所有的算盘机 M∈AM 都是递归可定义的。

证明：根据定理12，定理14，定理15，定理19 及定义3对 AM 的结构做归纳证明即可。

由定理11和定理20直接可得如下推论：

推论21 算盘机可计算函数是递归可计算的。

结束语 以上证明了算盘机的计算能力等同于递归函数的计算能力，而 Turing 在文[4]中证明了图灵机的计算能力等同于递归函数的计算能力，从而算盘机与图灵机的计算能力相同。故算盘机是一种计算模型。

参 考 文 献

1 Boolos G S, Jeffrey R C, Burgess J. Computability and Logic. Cambridge University Press, 2002
2 Cohen D E. Computability and Logic. Ellis Horwood Inc, 1987
3 莫绍揆等，数理逻辑. 高等教育出版社，1985
4 Turing A M. Computability and λ-Definability. The Journal of Symbolic Logic, 1937, 2(4)

（上接第61页）

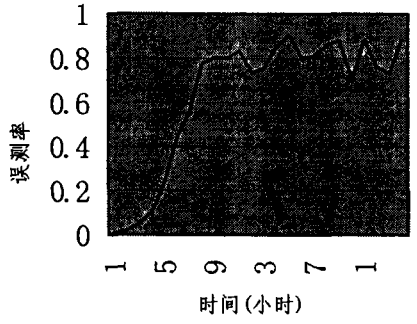

图3 时间—误测率曲线

由图3可看出，随着时间的推移，预测模型的误测率迅速升高，这是因为预测模型对入侵的预测是建立在不完整的特征描述基础上，它只能在短期内比较精确，而从长期来看非常不稳定的。因此，必须定期更新预测模型，以保持其准确性。

第二部分测试在不同的更新周期下模型的准确性，取更新周期为30、60、90分钟等10个点，绘制而成的“更新周期—误测率”曲线如图4所示。

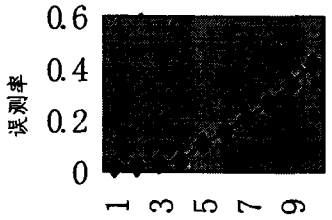

图4 更新周期—误测率

从理论上讲，更新周期越短，预测模型就越准确，实际上整个趋势也是如此，但在一定的更新周期范围内误测率相差无几。考虑到预测模型的频繁更新会给预测器带来较重的负担，因此需要在可容忍的误测率范围内选择尽可能长的更新周期。

4.3 试验2：测试 FPM 的数据处理能力

本试验的方法是启动整个系统（包括预测器），根据预测器的输入量和输出量，计算丢包率衡量 FPM 的吞吐量。实际从监测器出来的流量大约为100M/s，即预测器的输入量为100M/s，输出量为90M/s，即丢包率约为10%，而流向分析器的流量仅为20M/s，分析器的输出量也大约为20M/s，而在缺少 FPM 的情况下，分析器的丢包率高达70%，实际的吞吐量只有约30M/s。这说明

（1）预测器为分析器分担了大部分的数据量，使分析器的丢包率大大降低。

（2）预测器的吞吐量约为90M/s，大大高于分析器的30M/s。

结论 对入侵行为的预测根据时间粒度可以分为长期和短期两种，本文所研究的内容属于后者，其手段是通过建立入侵行为的预测/趋势模型，从而根据当前用户行为的初始特征判断它是否入侵行为，从而能做到预测，同时数据处理速度也大大加快。而一般的滥用检测系统对入侵的判断需要复杂而完整的证据，既不能保证检测速度，也不能保证对入侵的预测。本文介绍了一种预测机制——反馈预测机制，FPM 作为入侵检测的先锋分担其部分工作量，还能从入侵检测系统的成果中挖掘出特殊的知识反馈给预测器，以提高预测的准确性。因此，将 FPM 加入到入侵检测系统中，将能提高入侵检测系统的数据处理能力和实现预测。

在试验中可以看出，随着预测模型更新周期的延长，入侵预测的准确度迅速下降，这说明了 FPM 不能做到长期预测，更新周期的确定应综合考虑到效率和准确度两方面因素。预测模型测度的选择是本方法的另一关键，例如在试验中，如果选源/宿地址作为自变量，则预测模型的准确度太低，因为这些变量的值分布比较分散，规律性不强。

参 考 文 献

1 Snapp S R, Smaba S E. Signature Analysis Model Definition and Formalism. In: Proc. of the Fourth Workshop on Computer Security Incident Handling, Denver, Colorado, Aug. 1992
2 Lee W, Stolfo S J. Data mining approaches for intrusion detection. In: Proc. of the 7th USENIX Security Symposium, 1998, 21(3): 181～199
3 Lee W, et al. Real time data mining-based intrusion detection. In: Proc. Second DARPA Information Survivability Conf. and Exposition, 2001. 85～100
4 Lee W, et al. Toward Cost-Sensitive Modeling for Intrusion Detection and Response. Journal of Computer Security, 2002, 10(1): 318～336
5 Cohen W W. Fast effective rule induction. In: Machine Learing: the 12th Intl. Conf. Lake Taho, CA, Morgan Kaufmann, 1995