Mathematica 语言介绍

“Mathematica 是世界上仅有的科学计算的完全集成环境。1988 年第一次发行时，它已对计算机的许多技术和其它领域中的使用方法产生了深远的影响。

“人们常说 Mathematica 的发行标志着现代科学计算的开始。自 20 世纪 60 年代以来，用于数值、代数、图形和其它任务的单独软件包已经存在。但是 Mathematica 的设想是建一个一劳永逸的系统，以连贯和统一的方式，处理所有科学计算的不同方面。使这成为可能的关建是一种新的符号计算机语言的发明，它首次实现了仅使用很少的基本元素处理在科学计算中涉及的非常广泛的对象。”

——Mathematica 全书（第 4 版）

Mathematica 系统

Mathematica 是一个独立统一的科学计算系统，它有自己的核心语言，有自己的开发环境（Workbench），有自己的文档格式——可计算文档格式（CDF）。Mathematica 的源代码由 C/C++、Java 和 Mathematica 语言组合写成，有几百万行。Mathematica 的界面极为简单，其整个界面只是一菜单条加一个类似于记事本的“Notebook”，几乎所有操作都可以由代码完成，当然也有辅助面板，但那都不是必须的，因为那些面板也是由 Mathematica 语言写成的。Mathematica 的整体架构主要有两部分：内核和前端。在 Mathematica 运行时，这两者分别是不同的进程，他们之间通过 MathLink 通讯。通过 MathLink、J/Link 等，Mathematica 还可以和各种外部程序实现高级连接。

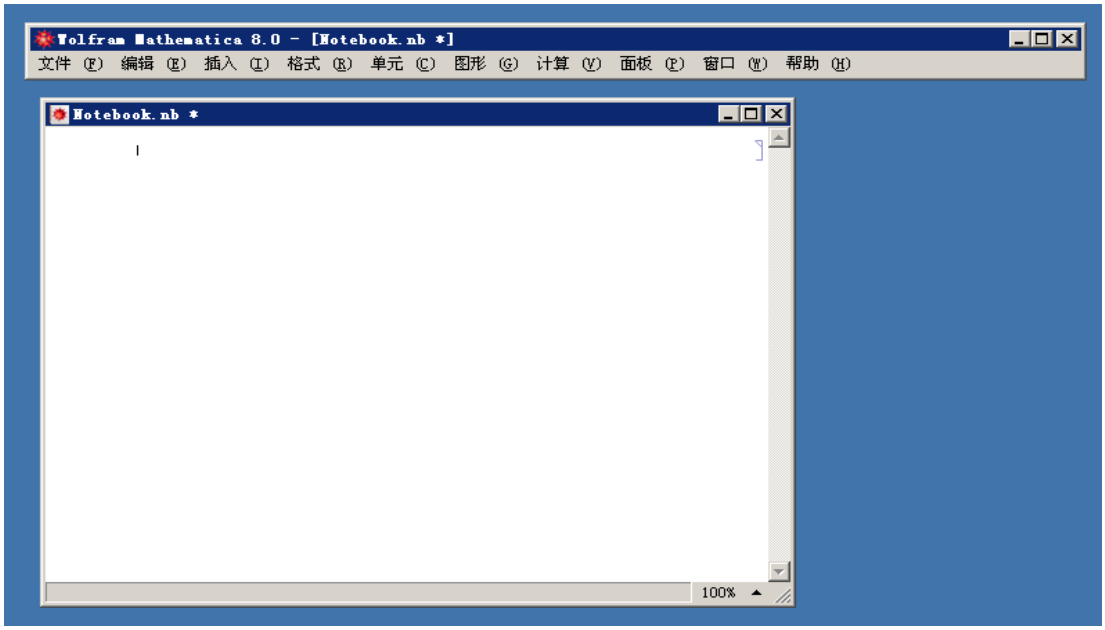


图 1：Mathematica 界面

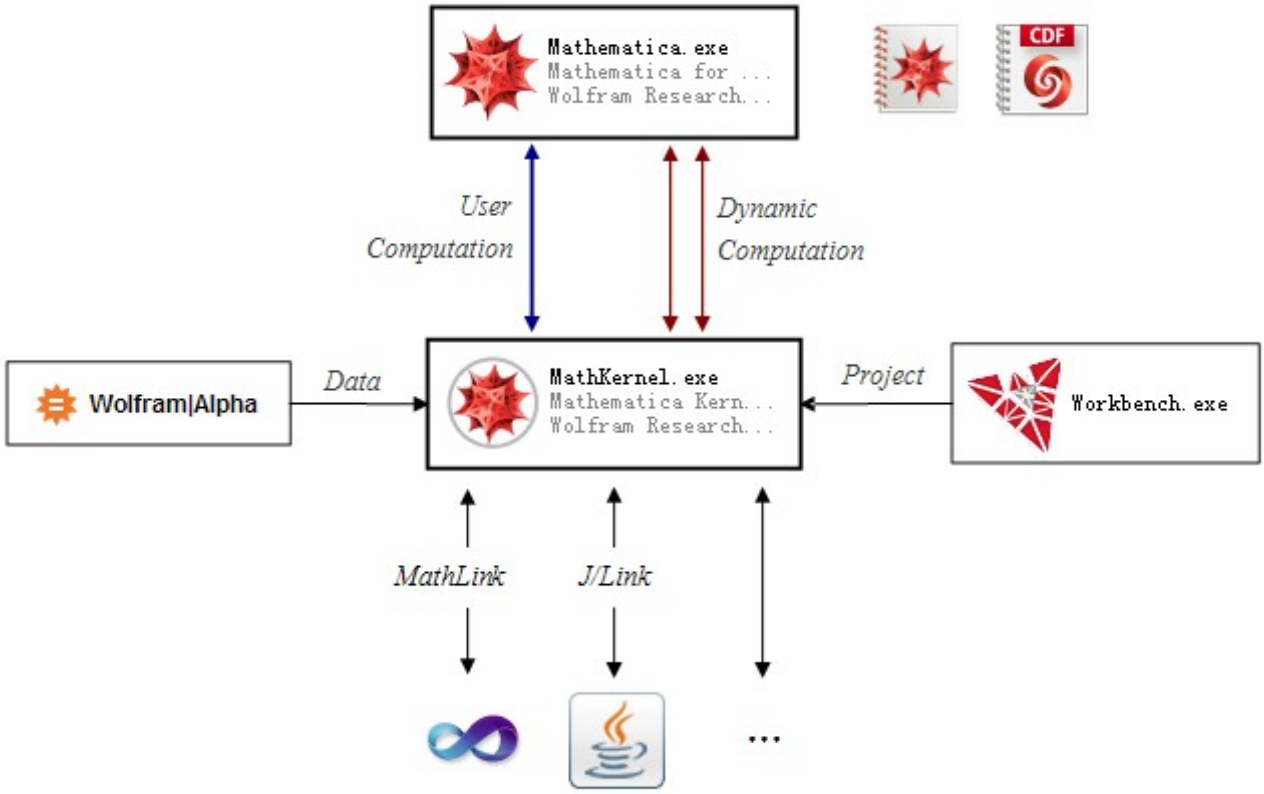


图 2：Mathematica 系统结构

Mathematica 语言特点

Mathematica 的核心其实是其高级的符号式语言，是这种语言把整个 Mathematica 系统紧揍地统一起来。实际上所有的对象——数据、程序、数学公式、图形、声音、窗口乃至整个笔记本文档——都可以用 Mathematica 符号表达式表示。Mathematica 语言支持多种模态的编程，尤其支持函数式编程（Functional Programming），函数式编程往往比过程式编程更高效。

四种括号

(term)	圆括号表示组合
f [x]	方括号表示函数
{ a , b , c }	花括号表示列表
v [[i]]	双方括号表示引索

说明：

（1）Mathematica 与其它语言，及与惯用的数学表示法有一个重要的不同是：函数变量在方括号中，而不是在圆括号里。实际上在 Mathematica 中，圆括号专用于表示项的组合，函数的自变量和对一些项进行组合显然是不同的概念。用相同的表示法表示这两种不同的概念一直被使用于印刷及其它计算机语言中，而 Mathematica 用不同的表示法来区别这两个不同的概念。这样区别有几个优点：若用圆括号，c (1+x) 表示 c [1+x] 还是表示 c* (1+x) 是分不清的，而对函数变量使用方括号就能排除这种二义性；允许不使用*或其它符号表示乘号，所以 Mathematica 能处理像 2 x ，a x ，a (1+x) 这样在数学上是标准的表达式[1]。

这种区别在 Mathematica 中是十分关键的，这是 Mathematica 语言不同于其它计算机语言的最重要的一点。Mathematica 语言比其它的编程语言——比如 Matlab 和 Maple——都灵活，应该也得益于这一点。

（2）列表是 Mathematica 中基本的数据结构，Mathematica 中的大多数数据都是由列表表示，列表中的元素的类型没有限制，可以是相同元素的数组，也可以是各种不同类型元素的任意组合，列表中的元素可以是数字（numbers）、字符串（strings）、符号（symbols）和一般表达式（general expressions），一般表达式可以是另一个列表等。在 Mathematica 内部，列表的不同元素被形式地储存为指针，各元素的实际部分储存在内存的不同部分，形式列表各元素的实际值通过指针能被找到。

一些列子：

{1，2，3，4}　　一个数组或者向量

{{1，2，3}，{4，5，6}，{7，8，9}}　　一个二维数组或者矩阵

{{{2005,1,1,0,0,0},289.451},{{2006,1,1,0,0,0},295.396}，…}

这个嵌套列表可以表示 2005 年、2006 年等某个国家的人口数据，{2005,1,1,0,0,0} 是表示时间的列表。

{1，3.14，a/b，7.8+2.9 I，{1，2，3}，c}　　不同类型数据的组合

对于数字（numbers），Mathematica 中建立了四种基本的数的类型：

Integer	任意长度的精确整数
Rational	integer/integer 的最简形式（分数）
Real	近似实数，具有任意指定的精度
Complex	具有 number+number I 形式的复数

（3）双方括号表示引索，如果 v={3,5,8,2,9}，则 v[[2]] 的值是 5，也可以通过双方括号修改引索所对应的值，若 v[[3]]=25，则 v 由原来的{3,5,8,2,9}改为{3,5,25,2,9}。

对于嵌套列表 w={{1,2,3},{4,5,6}}，w[[2,1]]的值是 4，w[[1,3]] 值是 3，等等。

Mathematica 中有一系列操纵列表的函数，比如：Append[ ]（给列表追加值），Insert[ ]（在列表的指定位置插入值），Sort[ ]（给列表中的元素排序），Length[ ]（求列表的长度） Split[](对列表中元素自动分类），等等。

这四种括号构成 Mathematica 语言的基本骨架。

（4）另外(*comment*)，用于程序的注释，类似于 C 语言中的 /* comment */。

函数式编程

在说函数式编程之前先说一下过程式编程，Mathematica 语言同样支持过程式编程。

（1）赋值：类似于 C 语言中的 =，+=，*=，++，-- 等运算符在 Mathematica 语言中一样使用。

（2）顺序结构：

expr1;expr2;expr3; …

（3）选择结构：

If [condition,expr1,expr2]

Switch[expr,form1,value1,form2,value2,form3,value3, ...]

Which[test1,value1,test2,value2,test3,value3, ...] 等

（4）循环结构：

Do[expr,{i,imin,imax,di}]

While[test,body]

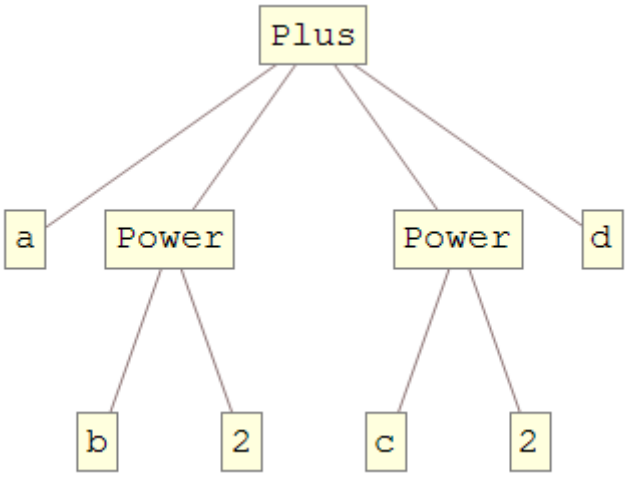
For[start,test,incr,body] 等

（5）控制流：

Return[ ], Break[ ], Continue[ ] 等。

对于懂 C 语言或者 Java 的人来说，这些都是非常熟悉的。而 Mathematica 语言的特点是具有基于规则的、函数式的编程方式。

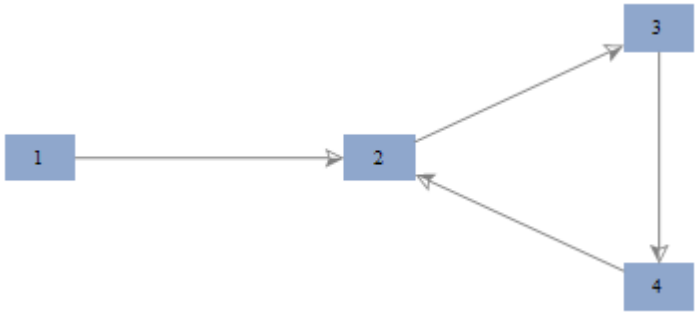
Mathematica 语言是一种树型编程语言，给出一段 Mathematica 程序，从整体上看就是括号套括号的样子，像 LISP。每个表达式的基本骨架就是 f [ ] 。比如：a+b^2+c^2+d，用 Mathematica 表达式表示就是 Plus[a,Power[b,2],Power[c,2],d]，图示如下：



Plus 表示加，Power 表示幂，都是类似于 f [x,y] 的表达示的头部（Head），表达式的头部确定这个表达式是什么。通过表达式的嵌套等，Mathematica 表达式可以表示非常广泛的对象。当然前面所示的只是表示了一个简单的数学式子，再举一些列子：

ArcSin[x]	表示 sin 的反函数
Expand[(x+y)^10]	表示一个代数运算，展开(x+y)^10
s=0; Do[s+=i,{i,1,100}]; s	表示一段程序，求 1 加到 100 的和

List [a,b,List[c,d]]　　表示列表 {a,b,{c,d}}
RGBColor[1,0,0]　　表示颜色红色
Graphics[{Circle[{0,0},1],Point[{0,0}]}]
一个二维图形，包括一个圆心为（0，0）、半径为 1 的圆和一个坐标为（0，0）的点
Sound[{SoundNote["C"],SoundNote["G"]}]　　表示声音
Image[{{0,1,0},{1,0,1},{0,1,0}}]　　表示一个只有 9 个像素的黑白图像
Graph[{1->2, 2->3, 3->4, 4->2}]　　表示一个“图”



在 Mathematica 中所有的表达式都有相同的基本结构，而每个表达式由头部的不同类型表示不同的对象，对于 f [x,y]，f 的含义不仅仅是函数，还可以是命令（如 Expand）、操作符（如 Plus, Power），或者只是一个标签（List, Point, RGBColor），不进行任何计算，只表示一种类型。另外 f [x,y]的形式更多的情况只是 Mathematica 的内部格式或者说标准格式，比如两数相加，在输入时不一定非得输入 Plus[x,y]，只需输入 x+y，这是很自然的 。Mathematica 表达式的输入其实有很大的灵活性：

f [x,y]	f [x,y] 的标准格式
f@x	f [x] 的前缀格式
x//f	f [x] 的后缀格式
x~f~y	f [x,y] 的嵌入格式

Mathematica 表达式具有一种树型结构，又具有层级结构，在 Mathematica 中有许多内部函数可对表达式的结构进行各种处理，而这与处理列表的方式相同，因为列表也是表达式。

了解 Mathematica 表达式是理解 Mathematica 函数式编程的基础。

在 Mathematica 中有一种叫做“纯函数”的概念，即 Function[ ]，举个例子说明：如果 fun=Function[{x,y},x^2+y]，则 fun[2,3]的值是 2^2+3=7，就是纯函数可以作为表达式的头部，类似于数学中的算子。还有一种更为简洁的表示方法：(#1^2+#2)&[2,3] 与 Function[{x,y},x^2+y][2,3] 是等价的，其值也是 7（#1 和#2 中的数字表示参数的顺序）。这样其实是把函数的定义与参数分开，使函数本身也成了一个独立的对象，成为了可被操纵的元素。在数学理论中这被称为“lambda calculus”。也可以把其它的大多数表达式的头部看作是像纯函数这样的算子，把它作为可计算的对象。比如 fun=Composition[Sin,Tan]，表示正弦函数和正切函数的复合，即 fun[x] 的值就是 Sin[Tan[x]] 。Mathematica 的函数式编程方式便由此展开。Mathematica 提供了一系列基本的基于函数的操作，列举部分如下[2]：

1、Map (/@)

Map[f,expr] 或 f/@expr　　将 f 应用到 expr 中第一层的每个元素.

例：

Map[f,{a,b,c}]　　{f[a],f[b],f[c]}

f/@{a,b,c}　　{f[a],f[b],f[c]}
Function[{x},x^2]/@{1,2,3,4}　　{1,4,9,16}
(#^2)&/@{1,2,3,4}　　{1,4,9,16}

2、Apply (@@)

Apply[f,expr] 或 f@@expr　　用 f 替换 expr 的头部.
Apply[f,expr,{1}]或 f@@@expr　用 f 替换 expr 的第一层的头部.

例：

Apply[f,{a,b,c,d}]　　f[a,b,c,d]
f@@{a,b,c,d}　　f[a,b,c,d]
f@@@{{a,b,c},{d,e}}　　{f[a,b,c],f[d,e]}

3、Fold

Fold[f,x,{a,b,c,...}]　　给出 f[f[f[x,a],b],c] ...

4、FoldList

FoldList[f,x,{a,b, ...}]　　给出列表 {x,f[x,a],f[f[x,a],b], ...}

5、Nest

Nest[f,expr,n]　　返回一个将 f 作用于 expr 上 n 次后得到的表达式

例：

Nest[f,x,3]　　f[f[f[x]]]

6、NestList

NestList[f,expr,n]　　将 f 作用于 expr 上 0 到 n 次，给出结果列表

例：

NestList[f,x,4]　　{x,f[x],f[f[x]],f[f[f[x]]],f[f[f[f[x]]]]}

在一个完整的 Mathematica 程序中很容易会看到诸如“/@”、“@@”等符号，这就利用了函数式编程的思想。

Mathematica 语言还有很多重要的不同于其它语言的特点，如模式、变换规则、上下文等，就不一一介绍了，前面主要介绍了 Mathematica 语言几个最为显著的特点。

相关的列一些其它例子：

1、变换规则

x+y/. x->3　　3+y
x+y/.{x->a,y->b}　　a+b

2、模式

f[x_]:=x^2　　函数定义
f[a]+f[b]/.f[x_]->x^2　　a^2+b^2
f[{a,b}]+f[c]/.f[{x_,y_}]->p[x+y]　　f[c]+p[a+b]
{1,x,x^2,x^3}/.x^n_->r[n]　　{1,x,r[2],r[3]}
Cases[{1,1,f[a],2,3,y,f[8],9,f[10]},_Integer]　　{1,1,2,3,9}

3、上下文

用于符号名称的组织，类似于 C++、C#中的命名空间。其基本的思想是任何符号的全名为两部分，形如 context`short，注意在 context 和 short 之间有一个倒引号“`”，这称为称为上下文标记。在 Mathematica 进程中的任何点有一个当前上下文$Context 和上下文路径$ContextPath，上下文的概念主要应用于程序包的开发。

参考资料

[1]Stephen Wolfram,Mathematica 全书(第 4 版),西安交通大学出版社,2002
[2]Salvatore Mangano,Mathematica Cookbook,O'Reilly Media,2010

——Zhengheng Li <zhenghenge@gmail.com> 2012-2-21