

BPA 程序中通用控制器的开发

尹建华 田 杰 韩祯祥

(浙江大学电机系 310027 杭州)

摘 要 介绍了一个 BPA 程序中开发的用于 FACTS 仿真分析的通用用户接口软件包——通用控制器 GCS,说明了它与其它的用户程序接口(UPI)技术相比所具有的特点和优点,介绍了通用控制器仿真语言的描述格式及其丰富的仿真库函数,阐明 GCS 可用于电力系统中各种控制装置的仿真和计算。

关键词 通用控制器 仿真分析 可控串补装置

0 引言

BPA 程序自 80 年代初引入我国以来,在电力系统模拟分析中发挥了重要作用。我国很多电力设计、运行部门都以此软件包作为电力系统仿真分析的重要工具,同时也有许多科研院所在此基础上进行开发,修改和增加了新的模块及功能。但是,由于历史的原因,BPA 程序的局限性也是很明显的,主要是 BPA 程序包中纯 80 列数据卡片的输入数据方法已不能很好地用来描述和模拟在电力系统中迅速发展的电力电子装置及相应控制器。现代电力系统的控制器模型也变得越来越灵活,单靠 BPA 程序包开发人员不断增加新的控制器模型显然不可能满足用户需要。这就需要用户自行设计增加新控制器模块并与原来 BPA 程序进行接口。

1 通用控制器(GCS)仿真语言

由电力科学研究院编制的《电力系统分析综合程序(PSASP)》中所采取的以标准数据文件接口作为用户自定义程序(UPI)和 PSASP 仿真程序的接口方法有严重不足,有时甚至会影响它的实用性。因为用户程序接口模块(UPI)和 PSASP 程序通过接口数据文件交替运行,参与暂态迭代计算过程,在每次交替求解网络方程到微分方程的过程中都要经历两次对标准接口文件的读和写。这种通过数据文件进行用户程序模块和系统程序之间通信的方法因相对要处理很多冗余的信息,浪费了很多计算机时,有时会使通信时间比用户模块的计算时间还要长,因而带有很大的局限性。

作者在 BPA 程序中开发和新增了 FACTS 仿真分析功能模块,在采用可控串补装置(TCSC)模

块的同时,开发了一种 BPA 程序通用控制器 GCS (general controller simulator)模块。它是一种类似 C 语言的仿真语言,可以方便地用来描述任意复杂的控制器模型,取代了多数电力系统软件必须将控制器方块图各种环节中的传递函数分解成相应的基本方块框,然后通过定义的基本数学、逻辑运算和库函数才能模拟整个控制器的烦琐过程,GCS 与 BPA 程序间通过类似 FORTRAN 语言的公用数据的形式来传递数据,解决了电力系统仿真中为实现用户自定义,控制器程序和相应电力系统程序的接口问题,执行速度比采用外部数据文件要快得多。这种 BPA 程序通用控制器有以下几方面的特点。

1.1 通用控制器(GCS)描述格式

BPA 格式的数据输入文件和描述某个控制器的仿真语言的文本文件同时读入到 BPA 程序中,不需要单独加以编译,只要此程序段所描述的控制器符合 GCS 语法即可通过。在 BPA 读入 GCS 仿真语言的过程中,除了对描述控制器的仿真语言程序段进行严格的语法检查之外,在所有数据输入后还针对控制器中所要输入的有关电力系统的参量或状态变量的合法性加以严格检查,使控制器中输入的不存在的电力系统有关信息都能被查出并警告出错。在完成上述语句检查并确认无错误后由 GCS 在计算机内存中动态解释生成一个由动态指针索引的用户自定义的可执行二进制压缩码。

1.2 类 C 仿真语言语句及相应的运算操作符

GCS 中有完备的数据类型定义,数据类型包括简单变量和数组。为模拟控制器中限幅环节、非线性磁滞环节、控制器本身暂态过程及多路选通开关,GCS 提供了下述语句:

赋值语句, $\text{var} = \text{expr}$; 条件语句, $\text{if} \{ \dots \} [\text{else} [\text{if} \{ \dots \}]]$; 循环语句, 包括 $\text{for} ([\dots]; [\dots]; [\dots]) \{ \dots \}$ 语句、 $\text{while} (\text{expr}) \{ \dots \}$ 语句、 $\text{do} \{ \dots \} \text{while}$

(expr) 语句; 开关选择语句, switch(option){case option...[default...]}; 跳转语句, go to; 中断循环 break 语句和 continue 语句。

为模拟控制器对多种输入、输出信号的运算, 取代一般用基本功能框来描述的方法, GCS 提供下述运算操作符:

算术操作符(+, -, *, /, %(求整数之模)); 位操作符(包括 &(与), |(或)和 ^(异或)); 移位操作符(<<(左移操作), >>(右移操作)); 关系和逻辑操作符(包括==(相等比较), !=(不相等), >(大于), <(小于), >=(大于或等于), <=(小于或等于), !(逻辑非), &&(逻辑与), ||(逻辑或)); 复合操作符(包括 +=, -=, *=, /=, %=, >>=, <<=, &=, ^=, |=); 单目运算符(++(增加 1), --(减小 1))。这些运算操作符的优先顺序也按 C 语言约定。

总之, GCS 仿真语言虽是 C 语言的子集, 但 C 语言编程的全部形式几乎都在通用控制器(GCS)仿真语言中实现。

1.3 丰富的仿真库函数

GCS 提供了相应 FORTRAN77 语言所定义的所有三角函数和反三角函数的复数运算作为基本数学库函数, 并提供了丰富的仿真库函数。

1.3.1 基于微分方程求解的连续系统仿真库函数

数值积分方法是连续系统数字仿真中使用的一种基本方法, GCS 提供 Euler 法、Trapezoidal 法、Runge-Kutta 等方法求解系统微分方程组。

图 1 所示的是 SISO 系统结构图, 其中 $G_1(s)$ 是控制器的传递函数, $G_2(s)$ 是受控对象的传递函数, 求解该系统在给定初始条件下对参考输入 r 的动态响应。首先求得闭环系统的传递函数为:

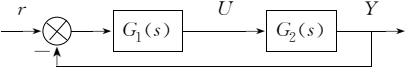


图1 一般控制系统结构图
Fig. 1 General control system block diagram

$$W(s) = \frac{G_1(s) \cdot G_2(s)}{1 + G_1(s) \cdot G_2(s)} \tag{1}$$

式(1)可化为标准的传递函数形式如下:

$$W(s) = \frac{b_1 s^{n-1} + \cdots + b_{n-1} s + b_n}{s^n + a_1 s^{n-1} + \cdots + a_{n-1} s + a_n} \tag{2}$$

利用系统传递函数化为状态方程的方法, 可以将式(1)化为如下的一阶微分方程组:

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{A} \mathbf{x} + \mathbf{B} \mathbf{r} \\ \mathbf{Y} = \mathbf{C} \mathbf{x} \end{cases} \tag{3}$$

由此可用上述各种积分方法求解此微分方程组。

1.3.2 基于指数矩阵的连续系统仿真库函数

在利用微分方程对系统进行仿真分析时, 最终化成相应的状态方程组, 直接利用状态空间分析方法求解系统动态响应, 系统的过渡过程以矩阵指数的形式表示, 关键在于计算指数矩阵 e^{A_h} 及相应的积分 $\int_0^h e^{A\tau} d\tau$ 。GCS 中给出的有关函数对阶跃输入和脉冲输入有很高的计算精度, 同时由于采取了提高程序运行可靠性的措施, 使得在系统的特征值分布很广或相应的计算步长较长时, 也能得到满意的计算精度。由式(3)所表示的状态方程的解可表示如下:

$$\begin{cases} \mathbf{x}(t) = e^{A(t-t_0)} \mathbf{x}(t_0) + \int \mathbf{B} \mathbf{u}(\tau) e^{A(t-\tau)} d\tau \\ \mathbf{y}(t) = \mathbf{C} \mathbf{x}(t) \end{cases} \tag{4}$$

在式(4)中, 设 $\mathbf{x}$ 为 n 维状态向量; $\mathbf{u}$ 为 m 维系统输入量; $\mathbf{y}$ 是 r 维输出量; 系统参数由矩阵 $\mathbf{A}(nn)$, $\mathbf{B}(nm)$, $\mathbf{C}(rn)$ 确定。由式(4)可确定相应的连续系统的离散递推公式为:

$$\begin{cases} \mathbf{x}(k+1) = e^{A_h} \mathbf{x}(k) + \int_0^h \mathbf{B} \mathbf{u}(k) e^{A\tau} d\tau \\ \mathbf{y}(k) = \mathbf{C} \mathbf{x}(k) \end{cases} \tag{5}$$

显然, 实际系统模拟分析表明, 在上述两种方法中, 1.3.2 节方法比 1.3.1 节方法计算精度高, 计算量相对较小, 并且对步距 h 没有特殊的要求。在利用 1.3.1 中方法时, 求解微分方程必须要求有较小的步距, 从而难以解决计算量同精度之间的矛盾; 而在用 1.3.2 节方法对控制器中所表示的系统进行仿真时, 一般在 GCS 仿真语言中定义两个矩阵数组 $\mathbf{F}(n,n)$ 和 $\mathbf{G}(n,r)$ 分别用以存放式(5)中的 e^{A_h} 和 $\int_0^h \mathbf{B} e^{A\tau} d\tau$ 时, 有更快的计算速度。但当系统中有非线性环节时, 用 1.3.1 节方法更有效。

1.3.3 面向环节的连续系统仿真库函数

上述两个方法都是面向整个系统的, 给定的系统模型是整个系统的传递函数或状态方程。系统中有非线性环节时, 或系统由许多环节组成且其中有局部的闭环时, 若要根据各个环节的特性求出整个系统的传递函数和状态方程有时非常困难, 特别在电力系统中, 控制器中一般有各种限幅环节(饱和特性), 死区特性环节和磁滞曲线特性、纯延迟特性等非线性环节, 这样, 只能对一部分局部环节采用线性系统的仿真方法求解。对整个控制器仿真只能用面向环节的方法。

在这里, 把 $\frac{C_i + D_i s}{A_i + B_i s}$ 作为典型的传递函数环节, 改变 A_i, B_i, C_i, D_i 的值可方便地实现控制系统中常见环节: $\frac{K}{s}, K_1 + \frac{K_2}{s}, \frac{K}{T s + 1}, K \frac{T_1 s + 1}{T_2 s + 1}$ 。振荡

环节 $\frac{K}{T^2 s^2 + 2 \zeta T s + 1}$, 可通过在图 1 中令 $G_1(s) = \frac{1}{T s / (2 \zeta) + 1}$, $G_2(s) = \frac{K}{2 \zeta T s}$ 及反馈环节 $\frac{1}{K}$ 来等效。各典型环节的输入输出关系为: $x_i(s) = \frac{C_i + D_i s}{A_i + B_i s} u_i(s)$ 。若令 $z_i(s) = \frac{u_i(s)}{A_i + B_i s}$, 则可得到方程组:

$$\begin{cases} (A_i + B_i s) z_i(s) = u_i(s) \\ x_i(s) = (C_i + D_i s) z_i(s) \end{cases} \quad (6)$$

对 n 个典型环节, 有

$$\begin{cases} \mathbf{A} \mathbf{z} + \mathbf{B} \dot{\mathbf{z}} = \mathbf{u} \\ \mathbf{x} = \mathbf{C} \mathbf{z} + \mathbf{D} \dot{\mathbf{z}} \end{cases} \quad (7)$$

式中

$$\begin{cases} \mathbf{A} = \text{diag}[A_1, A_2, \dots, A_n] \\ \mathbf{B} = \text{diag}[B_1, B_2, \dots, B_n] \\ \mathbf{C} = \text{diag}[C_1, C_2, \dots, C_n] \\ \mathbf{D} = \text{diag}[D_1, D_2, \dots, D_n] \end{cases} \quad (8)$$

若建立输入连接关联矩阵 $\mathbf{R}_{nm} = [R_{ij}]$ 表示控制向量 $\mathbf{u}$ 和各环节输入端的连接关系, 称为输入控制连接矩阵, R_{ij} 表示第 j 个输入 V_j 同第 i 个环节输入端的连接关系。直接相连时, $R_{ij} = 1$; 不相关时, $R_{ij} = 0$ 。 $\mathbf{W}_{nm} = [W_{ij}]$ 表示环节输出向量 $\mathbf{x}$ 与各环节输入端的连接关系, 称为环节连接矩阵, W_{ij} 表示第 j 个环节输出端到第 i 个环节输入端的连接关系。直接相连时, $W_{ij} = 1$; 不相连时, $W_{ij} = 0$ 。 $\mathbf{P}_m = [P_{ij}]$ 表示输出向量 $\mathbf{y}$ 与各环节输出 $\mathbf{x}$ 的连接关系, 称之为输出连接矩阵, P_{ij} 表示第 j 个环节输出 x_j 同第 i 个系统输出 y_i 之间的连接关系。直接连接时, $P_{ij} = 1$; 无关联时, $P_{ij} = 0$ 。则式(7)可化成新的系统状态方程和输出方程:

$$\begin{cases} \dot{\mathbf{z}} = \tilde{\mathbf{A}} \mathbf{z} + \tilde{\mathbf{B}} \mathbf{u} \\ \mathbf{x} = \tilde{\mathbf{C}} \mathbf{z} + \tilde{\mathbf{D}} \mathbf{u} \\ \mathbf{y} = \mathbf{P} \mathbf{x} = \mathbf{P}(\tilde{\mathbf{C}} \mathbf{z} + \tilde{\mathbf{D}} \mathbf{u}) \end{cases} \quad (9)$$

式中

$$\begin{cases} \tilde{\mathbf{A}} = (\mathbf{B} - \mathbf{W} \mathbf{D})^{-1} (\mathbf{W} \mathbf{C} - \mathbf{A}) \\ \tilde{\mathbf{B}} = (\mathbf{B} - \mathbf{W} \mathbf{D})^{-1} \mathbf{R} \\ \tilde{\mathbf{C}} = \mathbf{C} + \mathbf{D} \tilde{\mathbf{A}} \\ \tilde{\mathbf{D}} = \mathbf{D} \tilde{\mathbf{B}} \end{cases} \quad (10)$$

这样, 用户只要通过输入矩阵 $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}$ 的对角元素和输入连接矩阵 $\mathbf{R}, \mathbf{W}, \mathbf{P}$ 就能用来仿真多输入多输出线性控制系统(MIMO)。

1.3.4 FACTS 技术有关动态特性库函数

可控串联补偿装置(TCSC)中有关动态特性和在稳态下基波和谐波阻抗公式及 TCSC 装置所采取的自适应控制方式^[1]下的有关研究成果也可作为

库函数放在其中。这里需着重指出, 在一般情况下, 控制器所经历的时间常数可不予考虑, 但在 TCSC 装置中, 此过渡时间不能忽略。并且, 经历的过渡时间随 TCSC 装置中电容器电压的不同和晶闸管不同触发控制方式而有很大不同。

1.4 GCS 中用户自定义函数和子程序

在通用控制器程序中可由用户定义子例程(procedure)和函数(function)。它同样通过子程序调用和函数调用由形式参数和实际参数实现参数传递, 而参数传递可分为传址和传值两种方式, 从这一点看, GCS 中函数调用相似于 Pascal 语言中有关函数调用规则。

1.5 GCS 中详细的外部变量声明

定义 BPA 程序和通用控制器交换的变量为通用控制器的外部变量(external variable), 这些外部变量几乎包括所有电力系统装置的有关功角、角速度、加速功率、电压、电流、序阻抗、序电流、序电压、有功功率、无功功率和有关切机、切负荷操作及继电器操作信号等。这些外部变量可以是系统中有关发电机, 负荷, 母线, 直流线路及其整流器和逆变器, PSS, SVC 和 TCSC 装置所定义的有关信号。这些外部变量共有 250 个, 随着 GCS 的不断完善和电力系统新装置的出现, 这些变量还会有所增加。

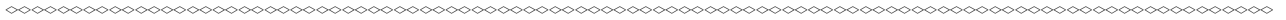
1.6 GCS 生成可快速执行代码

从生成代码的角度来看, 通用控制器 GCS 所生成的可执行解释代码完全由操作系统在动态分配的内存中由动态指针索引, 它不需要用任何外部数据文件与 BPA 程序的基本代码段进行联系, 因此它的执行速度和以源程序编译连接生成有效代码相似。实际应用表明, 由 GCS 生成的控制器代码在参与 BPA 暂态程序交替迭代运行中, 其执行速度比采用外部文件进行数据交换的用户接口技术要快得多。

2 通用控制器(GCS)开发简介

鉴于 BPA 程序的不足, 我们借鉴了当今有代表性的电力系统专业软件, 如电力科学研究院 PSASP 及德国 Siemens 公司的 NETOMAC^[2] 软件包的一些优点, 也参考专业性控制系统模拟软件包 Matlab 中 SimuLink 的特点和计算机软件发展的新趋势, 开发出 BPA 程序中通用控制器仿真语言。近年来, 属于 Microsoft 公司的 Windows 95/NT 操作系统环境下的 Visual C/C++4.2 和 FORTRAN 90 语言编译器已经面世, 期望不久将推出在 Windows 95/NT 环境下运行的带有完整用户接口功能的 GCS。

(下转第 29 页)



(上接第 15 页)

3 结语

为了在 BPA 程序中开发 GCS 仿真语言,作者曾参阅了大量有关计算机编译器开发和计算机形式语言专著^[3~5]。由于学识肤浅及工程量太大,目前 GCS 的某些方面还处于不断测试和完善过程中,仅供有关人员参考。

参 考 文 献

1 田 杰,尹建华,韩祯祥. TCSC 装置的动态特性及控制策略研究. 电力系统自动化. 1996,20(6)

2 韩祯祥,张 琦,徐 政. 一个大型集成化的电力系统仿真计算软件——NETOMAC. 电力系统自动化,1997,21(9)

3 陈火旺,钱家骅,孙永强. 程序设计语言编译原理. 国防工业出版社,1993

4 王广芳,曹兰斌,黄孝慈. 数据结构. 湖南科学技术出版社,1984

5 孙增圻,袁曾任. 控制系统的计算机辅助设计. 清华大学出版社,1988

尹建华,男,1966 年生,博士研究生,主要研究方向为 FACTS 技术仿真分析,包括在 BPA 程序中增加 TCSC 装置仿真分析模块的开发。

田 杰,男,1969 年生,博士,主要研究方向为直流输电和 FACTS 技术。

韩祯祥,男,1930 年生,教授,博士生导师,长期从事电力系统多学科的分析 and 研究。

THE DEVELOPMENT OF GENERAL CONTROLLER SIMULATOR IN BPA PROGRAM

Yin Jianhua, Tian Jie, Han Zhenxiang
(Zhejiang University. 310027. Hangzhou. China)

Abstract As a general user interface software package for the simulation analysis of FACTS. the general controller simulator (GCS) developed in BPA program is introduced in this paper. Compared with the user program interface (UPI) technique. the characteristics and advantages of GCS are illustrated. It can be applied to simulating and analyzing various control devices of electric power system. This paper also proposes the form of describing simulation language of GCS and its abundant simulation library functions.

Keywords general controller simulator (GCS) simulation analysis TCSC