# 新的适于计算机辅助工程的算法集成模型

何允钦 梁国柱

(北京航空航天大学 宇航学院, 北京 100191)

**摘 要**: 为解决计算机辅助工程中"以算法为核心"类软件系统在算法集成方面存在的重要问题,以火箭发动机设计这个较具代表性的特例为切入点,深入分析工程实践中算法的运用特点,并指出现有软件系统的不足. 提出 2 种算法模式,基于这 2 种算法模式建立一种新的算法集成模型——轮循与层次化算法集成模型,其具有松耦合、层次化、灵活等特点,基于该模型可建立一个跨领域、通用、高效的算法集成软件环境.

**关 键 词**: 计算机辅助工程; 算法; 集成; 模型

**中图分类号**: TP 391

**文献标识码**: A **文 章 编 号**: 1001-5965(2009)12-1516-05

## New algorithm integration model applicable for computer aided engineering

He Yunqin Liang Guozhu

(School of Astronautics, Beijing University of Aeronautics and Astronautics, Beijing 100191, China)

**Abstract**: In order to solve the important problem of algorithm integration in computer aided engineering(CAE) domain, which exists in many current engineering software systems taking algorithms as the core, using rocket motor design as a representative case and starting point, the characteristics of algorithm application in engineering practices were deeply analyzed, and flaws in current software system were pointed out. Two algorithm patterns were brought forward, and based on them a new algorithm integration model, named "algorithm integration model based on loop and hierarchy", was set up, which has the characteristics of loose-coupling, hierarchy, and flexibility etc. Based on the model, a cross-domain, general and high efficient algorithm integration software environment could be constructed.

**Key words**: computer aided engineering; algorithms; integration; model

随着计算机技术的飞速发展,工程界对其依赖越来越强,期望值也越来越高,直接导致软件根本复杂性[1]的急剧膨胀. 在计算机辅助工程中各种算法常需按具体任务需求被集成起来以完成计算任务. 以火箭发动机领域为例,美国高级火箭仿真中心开发了 GEN2 软件包,它是流动、燃烧、燃面推移、热防护等大量算法的集成[2];国内高校和研究单位也研制了各自的算法集成系统[3-5]. 但是,这些软件大都针对特定工程任务或局部性能和结构,软件的灵活性、通用性没作为首要考虑. 在工程设计中,常常需要集成多个算法来完成计算任务,但计算任务的复杂多变性所导致的旧算法的演进、新算法的融入、算法之间协作方式的变化等对软件系统的灵活性、通用性提出了较高的要求. 本文深入分析这些问题,并提出一种新的算法集成模型.

## 1 问题分析

在火箭发动机领域,工程设计任务会频繁地使用大量算法,而算法的具体运用又紧密依赖于任务需求与工作场景. 设计人员处理计算任务时的工作流程:根据具体任务需求、限制条件和设计人员历史经验选择并集成任务相关的若干算法,设定计算初始条件,运行计算并分析计算结果,如

**收稿日期**: 2009-05-07

**作者简介**: 何允钦(1980 — ),男,湖南永州人,博士生,heyunqin@gmail.com.

果结果不满足要求,则对相关的算法、参数或算法集成方式等进行调整并重复前述过程直到计算结果符合要求,如图 1 所示. 从图 1 可以看出,工作流程的 4 个重要环节,算法的选择、算法的集成、初始条件的设定以及计算结果的分析,皆以算法为基础并且紧密依赖于具体任务需求、限制条件和设计人员历史经验. 实际工作中,不同的任务需求会导致不同的算法选择、不同的算法集成方式、不同的初始条件设定等,甚至不同的设计人员由于各自历史经验的不同也可能对相同的计算任务采取不同的策略. 而设计人员和任务需求都是易变因素,因此,一个通用实效的火箭发动机算法集成系统必须能够灵活地适应工程任务中多样、易变的工作场景.

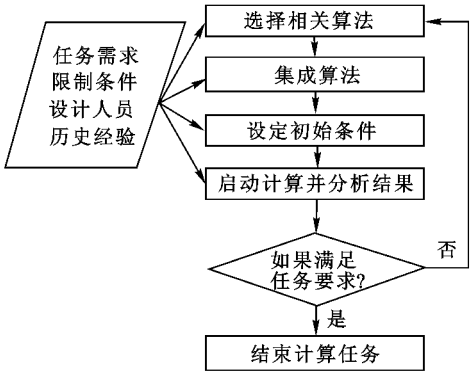



图 1　设计人员处理计算任务工作流程图

目前国内大部分软件系统是根据特定任务需求或设计人员要求来实现的,而且在实现中算法的集成方式、初始条件的设定等易变因素通常采用硬编码方式固化在软件系统中或采用专用的配置方式来设定,这致使每次计算任务调整或变更都需要进行程序的修改、生成和部署等,造成了诸多不便和资源浪费.

## 2　新的算法集成模型

本文提出一种新的算法集成模型——轮循与层次化算法集成模型,相对传统算法集成模型的主控制流程、紧耦合、单一层次、复杂、僵化的特点,其具有各司其职、松耦合、层次化、简明以及灵活的特点,很好地避免现存软件系统在算法集成方面存在的不足.

### 2.1　算法模式

在工程设计中,常常需要开发和使用大量算法,算法之间通过相互协作来进行计算. 通过对火箭发动机领域中算法的输入、输出参数及输入、输出参数与计算流程之间关系的研究,两种基本的算法模式被发现,分别定义为 ICO(Input Calculate Output)算法模式和 OCI(Output Calculate Input)算法模式,它们描述了算法的输入、输出参数同计算流程的两种不同耦合方式.

#### 2.1.1　ICO 算法模式

ICO 算法模式描述的算法输入、输出参数与计算流程耦合的特点:算法首先获取输入,然后计算,最后产生输出,如图 2 所示.

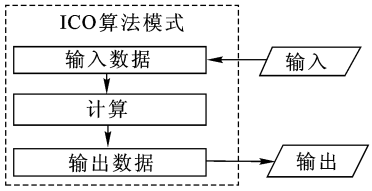



图 2　ICO 算法模式示意图

ICO 算法模式是工程中大部分算法的存在形式. 简单的,如三角函数,面积公式等;复杂的,如火箭发动机领域的热力计算,质量计算等. 它们都要求先获取输入参数,然后通过计算一次性地得到输出参数.

#### 2.1.2　OCI 算法模式

OCI 算法模式描述的算法输入、输出参数与计算流程耦合的特点:算法先产生输出数据并向外界输出,然后计算并等待外界的反馈数据作为输入,最后再对反馈数据进行处理,如图 3 所示.

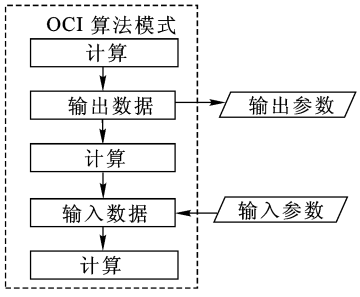



图 3　OCI 算法模式示意图

OCI 算法模式是一种很重要的输入、输出参数与计算流程的耦合方式,本质上它表征了算法间的一种紧密协作方式. 比如,在优化算法中就存在 OCI 算法模式,优化算法依据初始设定,产生设计变量具体值,然后传给用于求目标函数值的诸算法的输入,当目标函数值被求出后被反馈给优化算法,如此各算法通过反复协作来进行优化计算;又如,在复合函数 $f(x,g(y))$ 中,函数 $f$ 为了求值必须先向函数 $g$ 输出参数 $y$ 的具体值,然后当函数 $g$ 计算后将结果反馈给函数 $f$,$f$ 才能求出函数值.

### 2.2　基于算法轮循的计算流程控制模型

对于具体工程计算任务,传统算法集成模型

采用主控制流程方式,这导致算法内部流程与主控制流程紧密耦合,不能灵活地适应工程任务多变性.为此,本文提出"基于算法轮循的计算流程控制模型",采用算法轮循计算和发送参数消息的方法来对计算流程进行控制和解耦,以解决传统算法集成模型面临的问题.下面介绍该模型的思想和原理.

算法本质上是输入、输出数据转换器,在算法进行数据处理时,只要有足够的输入数据,算法就能进行"转换"产生相应的输出.遵循该原则就可以这样来考虑算法的计算流程控制:如果一个算法具有足够的输入时,就驱动它进行计算得到输出,并把输出发送给需要它的其它算法的输入,从而其它算法在得到足够输入后也可继续进行计算,如此反复地驱动具有足够输入数据的算法进行计算,计算流程就自然地形成了.通过该方式,所有具有足够输入数据的算法不断地轮循进行计算,而计算流程控制问题自然而然地就得到了解决.

采用轮循计算方式时,算法有3种状态:就绪、等待、完毕.就绪是指算法具有足够的输入数据了,可以进行计算了;等待是指算法还不具有足够的输入数据,还须等待其它算法的输出数据或者人为的输入设定;完毕状态是指算法已完成了对数据的处理得到了输出.在算法集成计算中,算法的3种状态会不断发生转换,如图4所示.可以看出,不具有足够的输入数据的处于等待状态的算法获得新的输入,并且新的输入使得算法满足进行计算的输入数据要求时,算法就转换成就绪状态,否则算法仍保持等待状态.处于就绪状态的算法计算后它的状态可能转换为完毕状态(如某些函数求值等),也可能是转换成等待状态(如优化算法在计算输出设计变量的具体值后,继续等待目标函数值的返回).处于完毕状态的算法完成了对输入数据的处理,并产生了输出数据,此时如该算法被设定新的输入数据,则其状态马上发生改变,以便对新的输入数据进行处理,如果新设定的输入数据满足算法进行计算的要求,则算法转换成就绪状态,否则算法转换成等待状态,继续等待其余的输入数据.图4未表示出来的是:在计算过程中,算法会产生输出,输出会根据配置好的数据传递路线发送给其它算法作为其输入,导致它们的状态发生改变,本文称之为"发送参数消息".

为了控制各算法进行轮循计算,还需要一个"计算控制器",它反复地驱动处于就绪状态的算法进行计算,并监视所有算法的状态.这样就得到

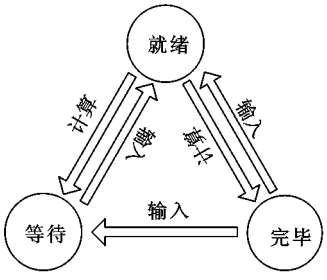



图4 算法状态转换图

了"基于算法轮循的计算流程控制模型",如图5所示.图中展示了计算控制器(大方框)、若干算法(小方框)和算法间的参数消息(带箭头连线).模型的详细运作方式:计算控制器包含了对所有算法引用,可以对算法的状态进行查询和设定,驱动处于就绪状态的算法进行计算;计算控制器包含就绪列表、等待列表和完毕列表3个列表,就绪列表包含处于就绪状态的算法,等待列表包含处于等待状态的算法,完毕列表包含处于完毕状态的算法,计算中当算法的状态发生转换时,会被移动到相应的列表中去;不同算法间存在数据依赖,算法间有参数依赖,当算法进行计算并产生输出时,会发送参数消息给其它算法,相应地算法接收到输入数据时,它的状态可能会发生转换;计算一经开始,计算控制器就会轮循地驱动就绪列表中的各算法进行计算,相应地进行计算的各算法会不断地发送参数消息给其它算法,导致它们的状态发生改变,这些状态发生改变的算法可能从等待状态或者完毕状态转换为就绪状态,从而在下一次轮循中被驱动进行计算,就这样反复持续下去直到就绪列表为空.

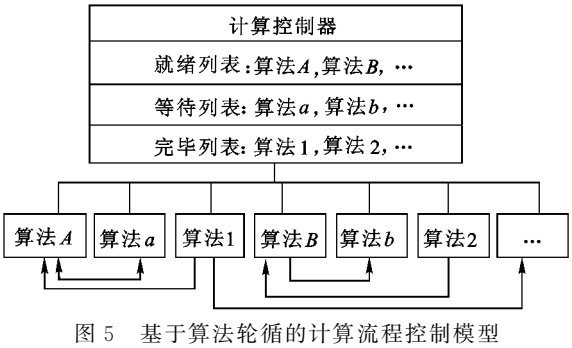



图5 基于算法轮循的计算流程控制模型

从"基于算法轮循的计算流程控制模型"中,可以看到主控制流程消失了,各算法间的独立性大大地增强,而且算法的计算流程也是自然形成的,只需要配置好算法间的数据传递关系,而这对于算法使用者本是很自然的事情.

下面举例来说明该模型.比如,一个优化问题涉及3个算法,算法 $A$(OCI)、算法 $B$(ICO)和算

法 $C$(ICO).算法 $A$ 为优化算法,算法 $B$ 和算法 $C$ 用于求目标函数值,算法 $B$ 的输入参数为 b_in,算法 $B$ 计算产生输出参数 b_out,算法 $C$ 有 2 个输入参数 c_in1,c_in2,算法 $C$ 计算产生输出参数 c_out.在算法集成方案中,b_in 和 c_in1 为设计变量,b_out 的值作为参数消息发送到 c_in2,目标函数值 c_out 作为参数消息发送到算法 $A$ 的输入参数 a_in,算法 $A$ 被设定为在计算中产生输出参数 a_ob,a_oc(设计变量具体值)并作为参数消息分别发送到 b_in,c_in1.为简化,一些需人为设定的输入、输出参数文中没有列出.图 6 描述了计算过程中计算控制器的状态变换.

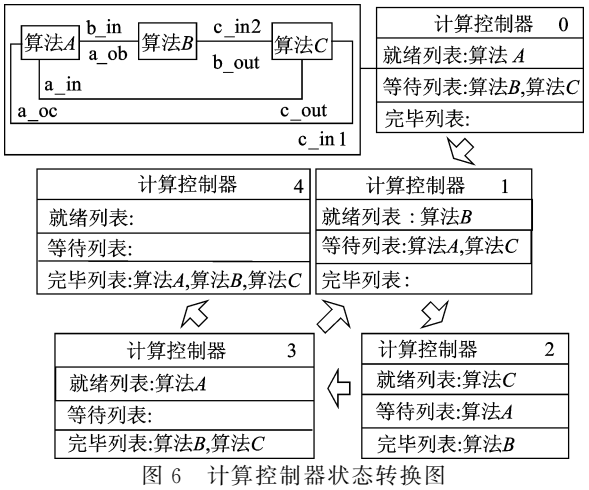


图 6　计算控制器状态转换图

图 6 中各个状态分别编号为 0,1,2,3,4,编号在计算控制器右上角(为避免重复,仅"编号 0"标识了具体算法及参数消息),计算流程描述如下.

0:计算之前,先配置好算法 $A$,$B$,$C$ 间的参数消息,设定各算法的初始输入参数值,此时 b_in,c_in1,c_in2 都为无效值,因此就绪列表中只有算法 $A$,算法 $B$,$C$ 处于等待状态.

0→1:计算控制器驱动就绪列表中的算法 $A$ 进行计算,产生输出参数 a_ob,a_oc,它们作为参数消息分别发送到 b_in 和 c_in1.这时算法 $A$ 被移动到等待列表中去等待目标函数值 a_in 的到来,此时 a_in 为无效值.算法 $B$ 接收到参数消息 a_ob 后转换为就绪状态,算法 $C$ 只获得了 c_in1 的输入,c_in2 仍为无效值,故仍保持等待状态.计算状态由状态 0 转换为状态 1.

1→2:计算控制器继续驱动就绪列表中的算法 $B$ 进行计算,产生输出参数 b_out,并发送参数消息到 c_in2,算法 $B$ 转换为完毕状态,而算法 $C$ 获得了 c_in2 的参数输入后转换为就绪状态,算法 $A$ 的 a_in 仍为无效值,故仍保持不变.计算状态由状态 1 转换为状态 2.

2→3:计算控制器继续驱动就绪列表中的算法 $C$ 进行计算,产生输出参数 c_out,并发送参数消息到 a_in,算法 $C$ 转换为完毕状态.算法 $A$ 获得了 a_in 的输入后转换为就绪状态,算法 $B$ 没有任何参数消息到来,故仍保持不变.计算状态由状态 2 转换为状态 3.

3→1 或 3→4:计算控制器继续驱动就绪列表中的算法 $A$ 进行计算,可能出现 2 种情况.①算法 $A$ 没有满足寻优迭代结束的条件,故继续寻优,此时算法 $A$ 将产生新的设计变量输出参数 a_ob,a_oc,它们作为参数消息分别发送到 b_in,c_in1,然后算法 $A$ 转换为等待状态,等待下一次目标函数值 a_in 的到来.算法 $B$ 收到参数消息并设定新的输入参数 b_in 后,会重置自己的状态,转换为就绪状态,并将 b_out 置为无效值,而 b_out 的无效值也会作为参数消息发送到 c_in2,进而导致 c_in2 变为无效值,进而又导致算法 $C$ 转换为等待状态,等待新的 c_in2 参数的到来.算法 $C$ 进行状态转换时也会将 c_out 设置为无效值,c_out 又会作为参数消息发送到 a_in 导致 a_in 变为无效值,从而使算法 $A$ 也转换成等待状态,这时计算状态由状态 3 转换为状态 1;②算法 $A$ 满足了寻优结束条件,算法 $A$ 转换为完毕状态,此时 3 个算法都处于完毕状态,就绪列表为空,计算到此结束,计算状态由状态 3 转换为状态 4.

### 2.3　算法集成模型的层次化

传统算法集成模型遇到的另一个问题是算法集成的复杂度控制和管理问题.在自然界和人类社会中复杂的系统都是具有层次结构的[6],这样的系统才是有序的,容易理解和演进的.在算法集成中,随着算法的数量规模增大,算法之间的参数传递关系复杂化,算法集成方案也会变得异常复杂.既然复杂系统都可以通过层次化来变得整洁有序而容易理解,也可以运用层次化的思想来对算法集成的复杂性来进行控制.可以这样来进行,将各算法中紧密相关的一些算法归纳在一起成为一个更大的、具有一定现实意义的复合算法.同理复合算法继续可以与其它的算法、复合算法进行归纳操作,这种归纳操作可反复使用直到整个算法集成方案变得整洁、有序和容易理解.

将算法层次化的方法与人类思维从具体到抽象、再由抽象到具体的思维模式相一致的,对算法进行归纳是从具体到抽象,而对归纳后的算法层次结构的理解则是从抽象到具体.层次化后的算法集成方案会变得整洁有序的根本原因是复杂性

被逐层分解和封装起来了，紧密相关的诸算法或者复合算法之间的参数消息发送较为密集，而它们被封装在包容它们的复合算法之内了，对外界来说它们可以整体上作为一个复合算法来理解，而且相对于复合算法内部的参数消息发送来说，复合算法与外界之间的参数消息发送会大为稀疏. 比如，图7为一简单平面拓扑结构的算法集成方案，图中所有算法处于同一层次. 可以看出，整个算法集成方案显得比较复杂、混乱、难以理解，特别是图中的参数消息发送路线，因为难以理解，也难以对这样一个算法集成方案加以维护和演进.

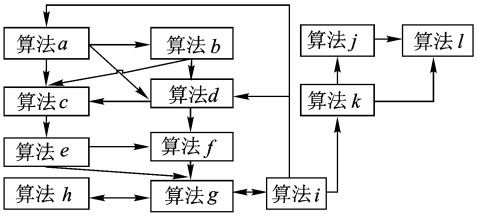



图7 未层次化的算法集成方案

下面分析层次化后的算法集成方案，如图8所示.

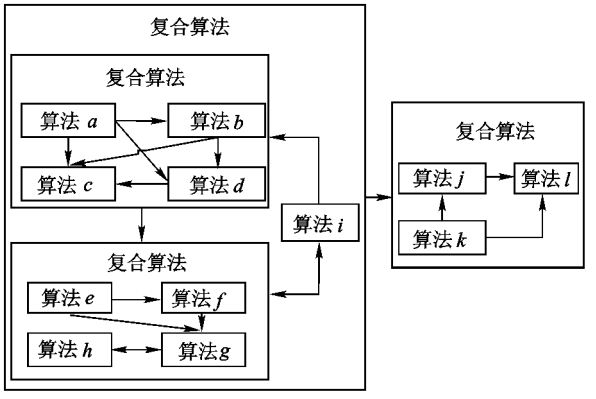



图8 层次化后的算法集成方案

由图8可以看出，各算法层次化后，紧密关联的各算法被归纳成为复合算法，图中算法 $a$、算法 $b$、算法 $c$ 和算法 $d$ 被归纳成复合算法，算法 $e$、算法 $f$、算法 $g$ 和算法 $f$ 也被归纳成复合算法，算法 $j$、算法 $k$ 和算法 $l$ 同样也被归纳成复合算法. 另外前2个复合算法和算法 $i$ 也被归纳成更大的复合算法. 可以看到复合算法内部的各算法或者复合算法间的参数发送较为密集，通过复合算法封装起来后，这种复杂性就被分解并封装起来了，只需在同一个复合算法层次内来理解各算法或复合算法间的参数消息发送. 在实践中，具有较为密集参数依赖的多个算法从整体上来看往往具有明确而独立的物理意义，它使算法集成方案变得容易理解，进而容易维护和演进.

## 3 结 论

通过对火箭发动机领域内算法集成及其规律性的深入研究，本文指出计算机辅助工程领域中传统算法集成模型和技术存在的重要问题；从工程实践中算法的运用特点出发，基于系统论原理提出一种通用、灵活的算法集成模型，即“轮循与层次化算法集成软件模型”，有效地避免了传统算法集成模型和技术的不足，为工程实践中设计、仿真和优化算法的集成提供了一个良好的解决方案. 尽管文中论述的问题是从火箭发动机领域的工程实践中提出来的，但在工程实践中，使用算法来解决设计计算问题基本模式却是不变的，在其它领域（如航空、汽车等）该问题同样会出现，而且文中模型的建立并没掺杂特定领域相关的东西，因而在计算机辅助工程中具有更普遍的应用意义.

### 参考文献（References）

[1] Brooks F P, Phillips F. The mythical man-month: essays on software engineering[M]. New Jersey: Addison-Wesley, 1995

[2] Dick W A, Heath M T. Whole system simulation of solid propellant rockets[R]. AIAA 2002-4345, 2002

[3] 解红雨，张为华，王中伟，等. 基于WEB的固体火箭发动机集成设计平台[J]. 推进技术，2007，28(1)：108－112
Xie Hongyu, Zhang Weihua, Wang Zhongwei, et al. Web based integrated design platform for solid rocket motor[J]. Journal of Propulsion Technology, 2007, 28(1): 108－112(in Chinese)

[4] 梁国柱，张卫华，郭红杰，等. 固体火箭发动机集成方案设计系统SRMCAD[J]. 固体火箭技术，2003，26(3)：18－20
Liang Guozhu, Zhang Weihua, Guo Hongjie, et al. Brief introduction of solid rocket motor integrated conceptual design system SRMCAD[J]. Journal of Solid Rocket Technology, 2003, 26(3): 18－20(in Chinese)

[5] 鲍福廷，蔡体敏，柳有权. 固体火箭发动机集成初步设计CAD系统研究[J]. 西北工业大学学报，2001，19(4)：498－501
Bao Futing, Cai Timin, Liu Youquan. Some progress in development of chinese integrated preliminary CAD design of solid rocket motors[J]. Journal of Northwestern Polytechnical University, 2001, 19(4): 498－501(in Chinese)

[6] Gerald M, Weinbe W D. 系统设计的一般原理[M]. 北京：清华大学出版社，2004
Gerald M, Weinbe W D. General principles of system design [M]. Beijing: Tsinghua University Press, 2004(in Chinese)

（编 辑：张 嵘）