基于 MAS 的 CWE 计算框架关键技术研究

彭小波[1],冯 平[1],Jue Wang[2]
PENG Xiao-bo[1],FENG Ping[1],Jue Wang[2]

1.深圳大学 机电与控制工程学院,广东 深圳 518060
2.英属哥伦比亚大学 机械工程系,加拿大
1.College of Mechatronics and Control Engineering,Shenzhen University,Shenzhen,Guangdong 518060,China
2.Department of Mechanics Engineering,The University of British Columbia,Vancouver,BC V6T1Z4,Canada
E-mail:pengxb@szu.edu.cn

PENG Xiao-bo,FENG Ping,Jue Wang.Research on MAS based CWE extraction framework.Computer Engineering and Applications,2007,43(19):202-205.

Abstract: Cutter/Workpiece Engagement(CWE) extraction is an important problem in process modeling.One approach is to use a B-rep solid modeler to perform the calculations.However,this can have a high computational overhead especially for complicated workpieces.This paper presents a Multi-Agent System(MAS) for B-rep based CWE extraction that allows distributed processing of the modeling steps over the Internet.A task scheduling strategy is presented for decomposing and distributing the task over the framework according to the idleness of the agents.Finally a prototype implementation and an example are given to show the effectiveness and efficiency of the system.

Key words: Cutter/Workpiece Engagement(CWE);virtual machining;Multi-Agent System(MAS)

摘 要:CWE(Cutter/Workpiece Engagement)计算是加工过程仿真与优化研究中的一个重要问题。在 CWE 计算过程中,工件模型不断被更新,其几何越来越复杂,计算开销越来越大。为了提高 CWE 计算的效率,提出了一个基于 MAS(Multi-Agent System)的 CWE 计算方法。该方法将 CWE 计算的任务加以分解,利用 Internet/Intranet 中的各个 Agent 进行分布式计算,对计算的结果进行组装得到最终结果。最后给出了原型系统的实现和一个实际计算的例子。

关键词:CWE;虚拟加工;MAS
文章编号:1002-8331(2007)19-0202-04 **文献标识码**:A **中图分类号**:TP31

1 引言

CWE 计算是加工过程仿真与优化中的一个关键问题。CWE 的本质即加工过程中刀具与工件的交。由该交之几何,即可得到刀具在任一加工位置的进出角,从而可以进行刀具在该位置的切削力预测并优化加工过程[1]。

迄今为止,CWE 计算方面的研究已经有了不少成果。Gupta[2]则对 2½D 的铣加工提出了一个 CWE 计算的解析方法。该方法由单个的交集函数组合得到总的交集函数,并得到了和离散模拟方法相当的实验结果。Imani[3]开发的系统能仿真球头铣刀在加工过程中的几何和受力情况。Yip-Hoi[4]分析归类了 2½D 铣加工过程中的加工特征,并将其运用到 CWE 的计算中,提高了 CWE 计算的效率。

上述研究都基于 B-rep 或者 CSG 模型。其缺陷之一就是随着加工过程的深入,工件模型的几何复杂度越来越高,从而求交计算也越来越复杂,时间花费越来越长。而另一方面,目前许多企业的 CAD/CAM 系统配置都很完善而且很高端,但这些系统经常处于不闲置状态。如果能充分利用这些系统资源,将会大大提高 CWE 计算的效率。为解决这些问题,本文提出了一个基于 MAS 的 CWE 计算方法。近年来 MAS(Multi-Agent System)技术在智能分布式协同系统设计中占据了越来越重要的地位[5]。刘弘[6]为支持协同的概念设计,提出了一种多代理环境中的进化计算方法。许青松[7]在分析了多联盟环境下企业对信息系统需求的基础上,提出了支持动态联盟的多代理企业信息模型。针对 MAS 在设计制造领域的应用,Wang[8]给出了一份全面的调查。

在本文提出的基于 MAS 的 CWE 计算方法中,各 Agent 构成一个计算框架。框架在接收到 CWE 计算请求后,在框架内自动寻找完成 CWE 计算所需要的其它 Agent。框架以最大限度地利用系统资源为原则,在框架内的各个 Agent 之间分配计算任务。另一方面,因为 CWE 的计算不在本地,CWE 计算的发起者不再需要依赖于某种 B-rep 几何引擎,只要发现了可以提供 CWE 计算的 Agent 存在,即可以提交 CWE 计算请求。

基金项目:湖北省数字制造重点实验室基金(No.sz0603);深圳大学科研启动基金(No.200732)。
作者简介:彭小波(1972-),男,讲师,博士,研究方向为 CAD/CAM;冯平(1975-),男,讲师,博士,研究方向为计算机辅助工程与测量;Jue Wang(1968-),男,UBC CAD 实验室 Research Scientist,研究方向为虚拟制造。

本文首先简要描述了本文 CWE 计算所采用的策略,然后给出了针对 CWE 计算的 MAS 结构,确定了 Agent 之间的通讯机制;重点描述了 CWE 计算的任务分配原则,以及 Agent 的状态管理机制。在文章的最后给出了一个 CWE 计算实例。

2　CWE 计算策略与系统架构

如图 1 所示,CWE 计算的输入包括初始工件的 CAD 模型,刀具轨迹,以及刀具的几何描述。通过这些信息,可以得到刀具沿着每条刀具轨迹运动时形成的刀具扫描体(Swept Volume,SV)。SV 与工件求交,即得到每条刀具轨迹所对应的切除体(Removal Volume,RV)。而 RV 与刀具模型的交则构成了 CWE 几何计算的结果。

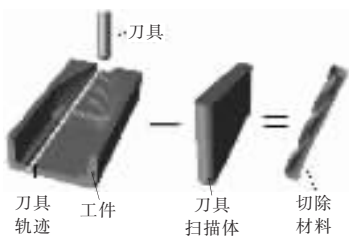


图 1　CWE 计算内容

CWE 计算的策略是尽可能在计算过程中采用并行处理的形式。根据上述 CWE 计算的过程,可将整个计算过程拆分为 SV 计算、RV 计算和 CWE 几何计算(简称 CWE 计算)3 个不同的部分。其中 CWE 计算依赖于 RV 的计算结果,而 RV 的计算依赖于 SV 的计算结果。各条刀具轨迹对应的 SV 计算可以并行计算,和加工时序没有关系;同样,CWE 的计算也可以实现并行;而 RV 计算则因为 RV 之间的相互影响,和加工时序关联,所以并行性较差。尽管如此,本文提出了一个简化的 RV 计算解耦方法,以最大限度地增加 RV 计算的并行性。

本文所提出的计算框架包含如下几种 Agent:每种计算类型对应于一种计算 Agent,即 CWE Agent(CWEA)、RV Agent(RVA)和 SV Agent(SVA);每种计算 Agent 对应一个接口 Agent(IA),用于计算参数的输入和计算结果的可视化显示;数据 Agent(DA)负责计算过程中的数据管理;管理 Agent(MA)负责框架内的 Agent 管理。框架及 Agent 的体系结构将在系统设计与实现一文中详细论述。

在某一次计算过程中,框架内的某一个空闲 Agent 成为主 Agent(MA),其它 Agent 成为从 Agent(SA)。MA 的形成是随机的,任何一个空闲 Agent 都可能成为 MA。一个典型的 CWE 计算工作流程如下:用户向计算框架发送计算请求,框架选中某个 CWEA 作为 MCWEA 主导计算;MCWEA 接受请求并提取计算参数;当 MCWEA 认为需要 RV 计算时,即生成 RV 计算请求,并向某个 MRVA 发送 RV 计算请求;MRVA 完成 RV 计算后,向 MCWEA 返回 RV 计算结果;MCWEA 获取 RV 计算结果,随即在框架内的其它 CWEA,即 SCWEA 之间分配 CWE 计算任务;在各 SCWEA 完成计算任务后,MCWEA 实现 CWE 计算结果的组装,并将最终结果返回给用户。需要指出的是,RV 的计算过程跟 CWE 计算一样,MRVA 可能会发送 SV 请求到 MSVA。

3　消息通讯

框架内 Agent 之间的通讯使用 ACL(Application Content Language)消息。每条 ACL 消息可以承载多项信息内容。每个内容都需要满足一定的语法规则,以便发送和接收方都可以正确理解该内容。

CWE 计算框架内存在如下的 5 类交互类型:请求、查询、应答、通知和问题报告。当 Agent A 想要 Agent B 从事某项工作时,A 向 B 发送一个请求。B 收到请求后,马上根据自己是否可以工作回复一个请求应答给 A。当 B 完成工作以后,会发送一个工作完成的通知到 A。当 A 需要知道 B 的状态时,A 发送状态查询消息到 MA,或者在 B 已知的情况下,直接发送查询消息到 B。一个 Agent 在其生命周期中的任何时候发生了错误,都会发送错误报告给 MA 与相关的 Agent。

除了交互类型,框架内的消息还需要包含完成任务所需要的各种参数,比如计算参数。如图 2 所示,每个消息体包含两部分:消息类型与消息内容。消息内容中包含内容类型与参数。参数部分类似于一个链表,每个参数占据一个节点。当一个 Agent 接收到一个消息以后,它对所接收到的消息体进行逆向解析从而获取消息类型与各参数值。

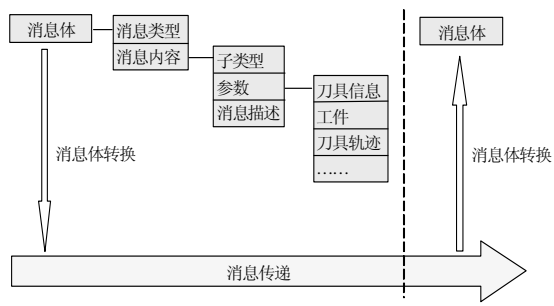


图 2　消息体结构与消息传递

4　任务分配

如前所述,当 MCWEA、MRVA 和 MSVA 接收到任务请求后,需要针对各从 Agent 进行任务分配。以 CWEA 为例,MCWEA 的任务分配流程如下:MCWEA 在收到 RV 计算结果后,即查询 MA 获取空闲 CWEA 列表。随后向该列表内的各 CWEA 发送 QUERY_IF_AVAILABLE 查询消息。CWEA 收到查询消息后,如果它不能参与计算,则回答 RES_NOT_AVAILABLE;否则应答 RES_AVAILABLE。同时,参数带回其空闲因子 k。空闲因子 k 反映某个 Agent 所在主机当前的资源占用情况:$k=1-\sum_{i=1}^{n}R_i/n$。如果总共考虑了 n 项资源情况,则其中 R_i 为第 i 项资源的占用率。假设只考虑 CPU 与内存使用状况,则 $k=(2-R_c-R_m)/2$,这里 R_c 和 R_m 分别是 CPU 占用率和内存占用率。显然,每个 Agent 的 k 值计算方法可以根据自身情况定制,Agent 需要提供典型资源的定制手段。MCWEA 将 CWEA 按照空闲因子的降序排列,并按照式(1)进行任务分配:

$$T_i=T\times\frac{k_i}{\sum_{j=1}^{n}k_j} \tag{1}$$

这里 n 是 CWEA 的个数,T 是 MRVA 返回的参与 CWE 计算的 RV 集合。如果 $T_i<1$,令 $T_i=1$。显然 $T=\sum_{i=1}^{m}T_i\ (m\leq n)$。

根据式(1),T_k 个 RV 被分配到第 k 个 CWEA。当下一组 RV 到达时,MCWEA 重复上述过程直到 CWE 计算完成。

MSVA 的任务分配策略与 MCWEA 相同,不同的是此时式(1)中的 T 是需要计算 SV 的刀具轨迹集合。

RV 计算具有高耦合性，加工时序上位于前列的刀具轨迹所产生的 RV,会对后续的 RV 产生影响。因此,MRVA 的任务分配必须考虑到这种耦合特性。但同时,很多情况下刀具轨迹也有分层和分组的特点,这也为 RV 计算的并行性提供了可能。

RV 的计算依赖于 SV,因此,MRVA 在计算 RV 之前,需要从 MSVA 得到 SV 计算结果。并非所有的 SV 都需要计算,而只需要计算对 RV 的形成有贡献的 SV。为了减少因为位置判断而引入的工作量,本文将加工时序上位于某个 RV 所对应的刀具轨迹之前的所有 SV 全部计算。在接收到 SV 计算结果之后,MRVA 开始 RV 计算。根据 RV 之间的相关性,可以将 RV 计算分为若干组。设 S 为所返回的 n 个 SV 的集合,其可以分为若干个子集合 S_i:

$$\begin{cases} S=\{S_i|i=1\cdots m\} \\ S_i=\{SV_j|j=a\cdots b\} \quad (1\leqslant a\leqslant b\leqslant n) \\ MBB(S_i)\cap MBB(S_j)=0 \quad (i\neq j) \end{cases} \tag{2}$$

其中 MBB 表示最小包容盒。S_i 满足如下条件:

对 $\exists p\in[a,b],\forall q\in[a,b]$,且有:$MBB(SV_p)\cap MBB(SV_q)\neq 0$。

式(2)给出了一个简化的 RV 相关性判断准则,即:如果两组 SV 不相交,则其 RV 不互相影响,那么该两组 RV 可以并行计算。

MRVA 将各 S_i 分配到各可用 RVA 上。分配原则为:S_i 按照其元素个数的降序排列,同样 RVA 按照其空闲因子降序排列,则分配到第 k 个 RVA 上的 SV 子集合个数为:

$$RS_k=\{S_i|i=a\cdot l+k,i\leqslant m,a=0,1,\cdots\} \tag{3}$$

其中 l 为 RVA 的个数。

框架利用滑动窗口来提高 Agent 层次之间的并行性,以提高计算效率。MCWEA 向 MRVA 发送的 RV 计算请求中,有一项参数称为窗口尺寸 WS(Window Size,WS)。MCWEA 利用 WS 来要求 MRVA 每次返回多少 RV 计算结果。比如说,如果 WS 为 5，则 MRVA 总是将所完成的前 5 个 RV 计算结果及时返回给 MCWEA，而不必等到后面所有的 RV 都完成才返回。而 MCWEA 一接收到计算结果,即开始任务分配计算。这样做减少了 MCWEA 等待的时间,提高了计算效率。

以图 3 为例,设 WS 为 3,用户请求计算第 3、4、5、9、10、11 段刀具轨迹对应的 CWE,需要计算相同序号的 RV,而 SV 则需要计算 11 以前的所有序号。在滑动窗口机制下,一旦 MSVA 得到 SV 1、2、3，即向 MRVA 发送计算该部分计算结果,MRVA 即开始计算 RV 3。MSVA 一旦得到 SV 4、5、6 后，立即通知 MRVA,MRVA 即开始计算 RV 4、5。MRVA 在计算完 RV 5 后通知 MCWEA 开始计算 CWE 3。图 3 的虚线部分为不采用滑动窗口时的耗时情况。其中 t_w 与 t_s 分别为两种情况下的计算时间。

5 Agent 的状态管理

计算的过程涉及到 Agent 之间的交互。各 Agent 必须要能够知道自己目前的状态，才能根据交互内容进行相应的动作。每个 Agent 都具有 8 种状态:IDLE、LISTEN、SESS_STARTED、UPQRY_SENT、UPCALC_STARTED、QRY_SENT、CALC_STARTED、SUCCESS。

IDLE 状态:Agent 的初始状态。在该状态下,Agent 等候状态查询和计算请求。

LISTEN 状态:因为 MA 需要预先知道各 SA 的状态以进行任务分配,即 MA 需要先查询其它 Agent 的状态,在任务分配完成后,才能发送任务请求。为了与 MA 的状态同步,处于 IDLE 状态的 Agent 接收到状态请求之后,回应 ACK 消息表示自身可用,同时进入监听状态。其它状态下的查询消息被丢弃或者回应自身不可用消息。处于 LISTEN 状态的 Agent 等待计算请求,在收到计算请求之后需要检验发送方代码,以确保等待对象与计算请求对象的一致性。在确定的时间内收不到计算请求,则超时进入 IDLE 状态。

SESS_STARTED 状态:该状态针对 MA 而言,此时创建一个新的会话过程。处于 LISTEN 状态的 MA 得到计算请求以后,如果需要上级计算(发生 Upstream 事件),比如 RV 计算需要上级的 SV 计算,则查询上级的 MSVA,进入 UPQRY_SENT 状态。直到上级的计算结果返回,则开始本级的 SA 查询,进入 QRY_SENT 状态。在接收到 SA 的回应之后进行任务分配并发送任务请求,转换到 CALC_STARTED 状态。此时,创建一个新的会话过程。本文中,会话过程总是开始于计算请求被接受,终止于计算过程结束。对于 SA 而言,其自身的计算工作结束,会话即结束;而对于 MA 而言,只有所有分发的任务结果被组装完成,会话才结束。

CALC_STARTED 状态:对 SA 而言,在 LISTEN 状态收到计算请求,即进入该状态。而一旦计算完成(FIN 事件发生),即进入 SUCCESS 状态。对于 MA 来说,收到 FIN 消息并不代表计算完成,只有 SESS_FIN 事件发生,才表示整个会话成功结束,Agent 即转入 SUCCESS 状态。

SUCCESS 状态:该状态为一瞬时状态,一旦 Agent 进入该状态,向计算发起方发送成功消息,随即回归到 IDLE。

实际上，每个 Agent 还有一个 ERROR 状态。在计算过程中,任何时候发生错误或者等待超时,则进入错误状态。该状态同样为一瞬时状态,进入该状态后,Agent 向会话发送方发送错误报告,随即转入 IDLE 状态。

在每种状态下,只有规定的消息被处理,而其它的消息则被忽略。图4 给出了 Agent 的有限状态机模型。

6 应用实例

基于上述原理与方法,实现了一个基于 MAS 的 CWE 计算原型系统。该原型系统的计算部分采用 B-rep 模型，基于

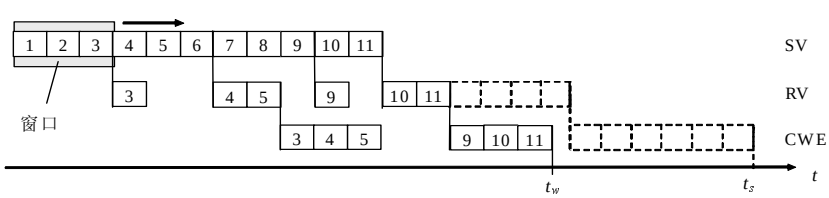


图 3 计算过程中的滑动窗口机制

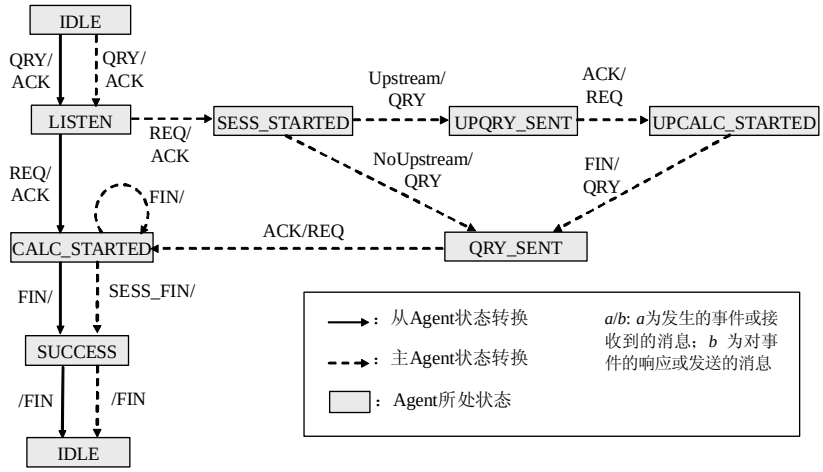
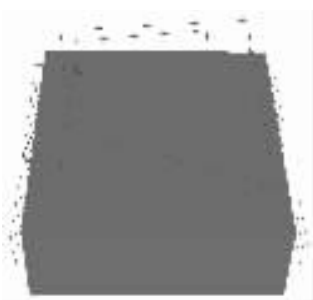


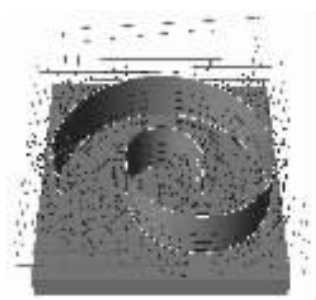
图 4　Agent 的有限状态机模型

VC++ 6.0 与 ACIS v10.0 实现。框架部分基于 JADE 与 Java 实现。任务分配模块通过 JNI 调用计算模块。本章实例的测试环境为 Intranet,计算模块资源环境为 Pentium 1.8 G CPU、256 M 内存。

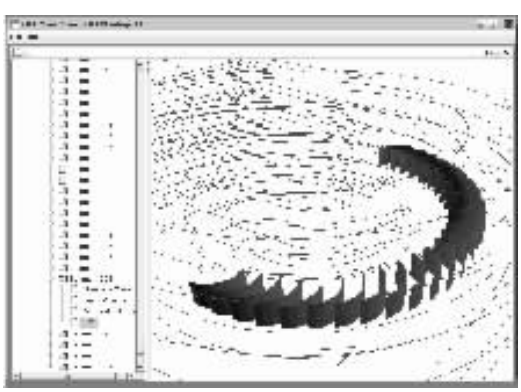
图 5 所示为一个加工实例。图 5(a)为初始工件,图 5(b)为加工完成后的工件以及所加工的 410 条刀具轨迹。为测试框架的计算效率,该实例计算了所有的加工轨迹。本例中共有 5 个 SVA、3 个 RVA、6 个 CWEA,10 个 WS。所有刀具轨迹的 CWE 计算时间花费为 27 s。作为对比,在同样配置的单台 PC 上,采用顺序计算方法计算所有的 CWE 所花费的时间为 68 s。图 5(c)为第 100 条刀具轨迹的 CWE 计算可视化结果。



(a)初始工件



(b)加工后工件及加工刀具轨迹



(c)刀具轨迹的可视化结果

图 5　基于 MAS 的 CWE 计算实例

7　结论和未来的工作

CWE 是虚拟加工和加工过程优化的一个关键问题。随着 B-rep 模型被广泛地应用于 CAD/CAM 领域,目前 CWE 计算方法也主要采用基于 B-rep 模型的方法。这种方法有一个缺点就是,随着计算过程的深入,工件表面几何会变得越来越复杂,使得计算效率越来越低。另一方面,这种方法必须依赖于本地的 B-rep 几何引擎的支持。本文提出了一个基于 MAS 的 CWE 计算框架。该框架接受远程用户的 CWE 计算请求,利用框架内的闲置资源,自动完成 CWE、RV 和 SV 的计算任务,大大提高了计算的效率。另一方面,利用该框架,CWE 计算不再需要依赖本地的 B-rep 几何引擎。

本文的后续工作主要包含如下的几个方面:

(1)继续展开针对基于广域网的计算过程中大数据量传输问题的研究,可能会涉及到模型的简化、分割技术以及数据压缩技术;

(2)进一步改善 RV 计算的并行性能,提高 RV 的计算效率。(收稿日期:2007 年 3 月)

参考文献:

[1] Fussell B K,Jerard R B,Hemmett J G.Modeling of cutting geometry and forces for 5-axis sculptured surface machining[J].Computer Aided Design,2003,35(4):333-346.

[2] Guptaa S K.Geometric algorithms for computing cutter engagement functions in 2.5D milling operations[J].Computer Aided Design,2005,37(14):1469-1480.

[3] Imani B M.An improved process simulation system for ball-end milling of sculptured surfaces[J].International Journal of Machine Tools & Manufacture,1998,38(9):1089-1107.

[4] Huang X,Yip-Hoi D.Cutter engagement feature extraction from solid models for end milling[C]//The Proceedings of the ASME International Mechanical Engineering Congress and Exposition,Anaheim,California,Nov 13-19,2004.

[5] Hao Qi,Shen Wei-ming,Zhang Zhan,et al.A multi-agent framework for collaborative engineering design and optimization[C]//Proc of DETC'04,Salt Lake City,Utah,USA,September 28-October 2,2004.

[6] 刘弘,崔巍.支持概念设计的多代理环境[J].计算机集成制造系统,2006,12(6):935-940.

[7] 许青松,范玉顺,吴澄,等.支持动态联盟的多代理系统[J].控制与决策,2001,16(2):199-202.

[8] Wang L,Shen W,Xie H,et al.Collaborative conceptual design:a state-of-the-art survey[J].Computer-Aided Design,2002,34(13):981-996.