南昌航空大学科技学院

毕业设计（外文翻译）

题目 一种新的工艺计划的方法-代理模型法

系　　别 航空工程系

专业名称 机械设计制造及其自动化

班级学号 068105338

学生姓名 张志刚

指导老师 罗海泉

二０一０ 年 三 月

一种新的工艺计划的方法-代理模型法

F.L.著,S.K.曹公)、保罗张新华吴

部门的制造和工程管理，香港城市大学 AIenue 编织品黄宏斌,香港，中国

1998 年 11 月 30 日收到，1999 年 1 月 25 日接受

摘要：

一个设计巧妙的 CAPP 系统能在 CAD 和 CAM 之间架起相互联系的桥梁 。许多最新设计的系统都是基于专家系统的。但是，由于过于复杂，在实际运用中，上述的许多系统在企业中都不能有效地运作。同时，现代的计算机集成制造系统要求 CAPP 系统在实际运用中具有良好的可扩展性和柔性。因此，仅仅只用一个大型的专家系统来构建工业 CAPP 系统是非常困难的。为了克服上述的缺点，本文提供了一种新的协同代理模型的设计方法。它具有以下优点：良好的独立性、柔性、通用性、模块性和可分离性。依据这框架我们特别设计了这种协同过程计划系统即机械协同 CAPP 来验证上述理论。它的系统模拟、代理结构设计、协作、协同和协同 CAPP 系统的结构研究将在下面介绍。

关键字：CAPP; Cooperative agent; Modelling; CIMS

1. 介绍

过程设计为工厂制造设计好的产品提供了丰富的信息。它分门别类地为每一产品的部分和集体建立了连接，并且明确规定了产品生产的过程、费用以及产品生产的各种限制条件，比如：设计好的产品的几何模型、材料、质量、所使用的机器、工具的可获得性、劳动能力、适用性和相应的作者等等。以前，过程计划通常都是由大量的制造经验丰富的专家来制定。近几十年来，计算机技术的快速发展极大的促进了CAPP的发展。

一般来说，CAPP的实现方法有两种：差异法和生成法。差异法是一种数据检索和编辑的方法。一些标准或成熟的CAPP系统都被使用了成组集成技术并且储存在一个数据库中。当一个新的产品被要求生产时，我们可以从数据库中取出相似的过程计划，通过适当的编辑和改进就可以让它适应新的计划。而生成法则基于知识为基础的方法自动装置能根据部件特性和生产要求自动生成工艺计划的知识库。

差异法的成功实施取决于成组技术、制定者的经验、足够的标准采集或者成熟的工艺计划。这种方法特别适合产品系列比较少而每个产品系中产品比较多的公司。许

多早期的CAPP工具都是根据不同的过程计划方法被分类处理的。典型的例子有CAPP、MIPLAN等等。生成法在最近这几年得到了越来越广泛的关注。它提供了另一种可供选择的计划的一种潜力。典型的例子有APPAS ，EXCAP ，KRONOS，XCUT 6，QTC （Quick turnaround cell），PART，OOPPS（ object-oriented process planning system ），MePlans ， COMPLAN Process Planner CPP 等等。生产法的产生主要源于定向的大公司和研究机构，特别是那些有许多小批量产品的公司的需要。然而，要做出一个真正能满足工业的要求提供结构协调和可兼容的框架、知识表达方法和推理结构合理的生成法工艺计划系统还是困难重重的。

协同代理系统是专门为那些解决特殊并联结构的人设计的。同时，它还可以协助他们解决其他复杂的问题。一个代理协作系统包括许多独立的含有协作引擎的代理。每一个代理都针对一种具体的任务，并且都含有自己的知识库和推理引擎，它们为协作环境下联系其他代理提供了一个合作和交流的桥梁。在不同的机器上，人们需要使用不同的语言和知识来描述。这样，这个系统就提供了一种能适应和解决不同类型问题的集成环境。

2. 过程计划的问题

一个机械过程一般都牵扯到机器工具、操作、固定装置和切割工具等问题，这就要求相关人员有各个不同领域的知识。一般，机械过程都包括以下几个部分:

· 外形识别

· 机器操作选择

· 机器的选择

· 切割工具的选择

· 固定装置的选择和设计

· 后续操作和成本预算

外形识别部分是指由产品设计数据规定制造外形。机器操作选择部分则指根据外形特征和制造环境选择相关的机器操作。在考虑了部件的特性及其机器的处理能力和性质诸如: 工作容量、精确度、动力、固定装置和其它功能之后，执行选定操作的过程就决定了机器设备的选择。固定装置选择部分是根据选择固定装置部分的几何形状、尺寸和制造特性来进行的。切割工具的选择重点关注的是工具的类型、材料、外形和工具的大小。

3. 提议方法

从纯技术的角度看，CAPP依然是一个极其复杂和困难的问题。因此，许多研究人

员把注意力集中于CAPP系统发展的方法和策略方面。然而，许多系统都由独立的专家系统发展而成的。由于CAPP系统的复杂性，这样一种系统结构很少能够解决的问题在制造业界常见的问题。

协同CAPP框架能够减少现行CAPP系统的局限性。实际上，它更强调一个现代的CAPP系统能达到以下有利于实际发展的功能：柔性、模块化、通用性、独立性和可分离性。

·独立性意味着CAPP系统是作为一个独立的系统发展。系统一旦建立，它就能够发展成为一个完整的CAD或CAM系统。每个代理同样能够看成独立和自主的系统。

·柔性是指系统能很容易地把新技术和新方法加到CAPP系统中。

·协同可行性要求不同的机器或者方法能够在一起流畅、顺利地解决问题。

·模块化是指CAPP系统是一个拥有不同子系统的综合系统。

·可分离性能分离CAPP系统来满足用户不同的需要。

协同CAPP构造的设计跟机械工艺计划有关。机械协同CAPP系统，以及它的结构、运行和应用的有关事宜在本文中都有详细的介绍。

3.1. 机械协同CAPP系统的概况

图一给出了机械协同CAPP系统在CAD/CAP/CAM环境下的总体结构。在执行中，一般包括以下六个工艺计划代理（P-agents ）。

·特征识别代理

·机械操作选择代理

·机器选择代理

·切割工具选择代理

·固定装置选择和设计代理

·程序操作和费用预算代理

不同的代理可以在互联网上不同的机器上运行。B代理提供了某个问题的全球状态信息，并且监控操作从属在所有单个P代理中运行的附件。当某个问题发生时，B代理就会激励所有注册过的P代理采取相应的措施。从P代理行动的结果又返回到B代理。在设定的条件下，B代理会保持一致和在特定的约束下的决策区域的一致性和完整性。产品设计资料从计算机辅助设计系统的D代理传给协同CAPP系统来设计系统。协同CAPP系统从流程安排或 shopfloor系统获取产品的生产约束，并根据产品设计资料和约束生成相应的工艺计划。如果没有得到满意的结果，协同CAPP系统就把冲突信息反馈给设计部门或其他相关部门。如此反复，最后，切实可行的替代方案就会传递

给scheduling 或者shopfloor系统来确定时间，从而做出合理的安排。

3.2. 代理的结构

一般的工艺计划代理结构如图二所示。代理由四部分构成：代理控制器、推理引擎、功能适配器和应用库。配置文件夹被用来构成代理。规则和论据构成应用库。基本上,所有P代理的组成部分都含有代理控制器、规则引擎、网络适配器、文件夹适配器、键盘适配器、信息查看适配器和进程适配器。其他适配器如图二所示。在P代理中，处理适配器是十分重要的适配器，因为它能解决工艺计划中提案产生、冲突解决和提案评估过程的制订等问题。为了应用不同产品规则和实物混合体的领域的知识，不同的代理使用不同的适配器来执行。所以，必须根据不同的以实物为导向的描述采用不同的设计。

数据库适配器被习惯于用来存放对B代理和P代理来说有用的数据，如：问题的定义、提案、冲突、评价和解决方案等等。

因为KQML是以代理为基础的程序之间联系时使用最广泛通信的语言,尤其是当它们独立和不同步时,而协同CAPP系统的各个代理作为一种通信使用了以KQML为基础的联系协议。所以，我们就用NetKQML适配器与B代理通信。根据KQML传输的要求，代理之间使用携带离散消息的单向链接相互联系。这些链接有时含有和它们结合在一起的有限消息传输延迟。当一个代理收到一条消息时，它就可以知道消息从何而来。当一个代理要发送消息时，直接发送到输出链接即可。这样，消息就仅仅单一目的的送达它们要去的地方。这种传送信息的方式十分可靠。Windows 95的接口思想也已经被用来执行到NetKQML适配器上。TCP/IP是一个被用于接口通讯传输的协议。

信息查看适配器用来显示监视器的信息。

时间适配器用来计算工艺计划实施过程的时间。

进程适配器用来控制进程事件，诸如提案生成、提案评估和冲突解决。当检测到来自另一个代理的提案时，适配器就为一个评估事件确定时间。当发现一个工艺计划问题时，它就安排一个提案生成事件为这一事件确定时间。这两个事件发生的同时都会赋予相应的优先级。如果一个冲突被要求需要解决方案，那么适配器就安排一个冲突解决事件并赋予最高的优先级。当被要求需要一个协商方案时，适配器会立即停止其他事件，立即生成协商功能器。

当一个问题到达时，P代理会引导‘查看问题’功能器对其检查。然后，它就生成一个问题检查事件，同时，完成对问题的读取。为了回应‘提案生成任务’事件，功能器‘生成提案’相应生成一个提案，并且生成另一个提案‘已经生成的事件’。

这个事件是根据提案而来。

为了回应进程适配器生成的名为‘提案生成’的任务事件，传输过来的事件需要适当的评估。适配器需要检查其是否已经生成一项互动的方案。如果是，它就链接这两个提案，并且查看提案方提案的意图，以此来意图评估这个新提案。在这种情况下，适配器就已经完成了对已提方案的评估。如果提案适配器没有生成互动作用的提案，适配器就在已经安排好但还没有审理的任务中寻找相近的方案。接着它就检查已经开始运行或者正要运行方案。如果有，它就把链接评估和产生分派任务。在这种情况中，它就完成了对正在执行提案的评估。但是，即使这个正在执行的提案可行，我们也没有必要生成一个独立的提案或者跟正在执行提案兼容很好的提案。如果没有相近的提案在运行，评估任务就只能根据代理知识对传输过来的提案进行评估，之后就通知正在提案的P代理。

如果另外的代理传送过来冲突，冲突解决器会采取相应的措施来解决。在适配器对冲突作出反应之后，首先，适配器判断冲突所处的环境,然后它调用相应的策略解决冲突。

3.3. 知识描述

协同CAPP系统的知识都是有关产品和代理方面的。它的每个P代理在协同CAPP系统中都含有三种类型的知识：区域知识、控制知识和冲突解决知识。B代理仅仅含有控制知识。局部描述一般都是关于问题描述的，它被协同CAPP系统中的所有代理所共享。

3.4. 局部描述

机械协同CAPP系统中的实体含有两种数据。一种是有关计划产生约束的，另一种则是关于几何信息的。它们是:

- 约束（生产时间、切削力、机械动力）；
- 实体（Name; Type; MaxSize; Material; InitStatus; HeatTreat; Features．）。Name代表实体的标识符，它必须是唯一的。Type则突出反映它的外形。MaxSize是指实体外壳的最大尺寸。Material则是实体的材料的种类。InitStatus 指能对实体进行初始加工的地方。加热处理指实体加热处理的条件。Features 是所有特征的集合。
- 特征的描述包括:FEATURE (Name; Type; Location; FinalSize; InitSize; Hardness; Tolerances)。

  Name是指一个特征的标识符，它必须是唯一的。Type则突出反映特征的关键词。

Location指初始的摆放位置和摆放方向的矢量。特征尺寸包括最终尺寸(FinalSize)和初始尺寸(InitSize)，都是指特征的三维尺寸。Hardness 指特征的硬度。Tolerances指空间和几何的工差，用不同的关键字来表示和区别它。

3.5. 区域知识

每个P代理的区域知识都是关于自身工艺计划能力的描述，用来生成提案、评价提案和解决冲突。不同的代理可能有不同的区域知识描述版本，诸如数据库和分解算法等等。区域知识可以在制造手册查到，如手册的19到21页。不同的P代理有不同的区域知识内容。

例如,操作选择代理通常用来生成部分特定特征的替代机械操作。对于给定的特征，可能存在不止一种潜在的操作。一些传统的机械加工方法如：锻造、钢模铸造、钻盲孔、镦坯、磨、镗孔、成型、研磨、石磨和钻石磨等等已被归入了区域知识库。它的内容包括操作与机械特征、材料、工差、预操作和累积时间等参变量之间的关系。知识的语义网络结构如图3所示。在知识库中，一个操作过程存储在一个节点里，所有的操作存储在一个表单中。

提案生成器、提案评估器和冲突解决策略都存储在每一个P-agent的解决适配器中。

3.6. 冲突解决知识

冲突解决策略包括两种类型：区域依赖性和区域独立性。区域依赖性策略主要包括如何来解决当一个P代理跟另一个P代理的提案相冲突时的远期建议问题。每一个P代理都有跟其他P代理不同的建议策略，也包括对给出的冲突解决方案的解释。区域独立性策略应用的更加普遍并且在所有的P代理中都是一样的。为了与其他P代理所有解决问题方法思想的一致，它们的设计依据是一些冲突解决方法的基本指导方针。这些方针决定了区域依赖策略的应用和解决策略的更改。下面的这些非区域依赖冲突解决策略一般都用来解决协同CAPP系统中出现的的问题。

· 折衷方案：找到一种快速的在可接受范围内的提案;

· 勉强产生的替代方案：在来自其他非柔性代理约束或者其他代理部分解决方案的基础上产生新的替代方案;

· 类似成功方案参数的设置：根据以往成功方案设定相似的系列参数。

3.7. 控制知识

协同CAPP系统中所有的P代理都能根据相同的控制知识完成一系列任务:

· 生成新的提案;

·评估提案;

·协商处理;

·解决冲突。

以上的任务在执行之前就已经安排好了，每一个任务被设置成了一个事件。P代理首先对

一个事件做出反应，然后才执行相关的任务。它的控制知识则被用来安排任务的时间和对事件做出反应。

作为一个事件，每个任务都被赋予了一定的优先级。根据前面的讨论，我们知道协商是最高等级的任务，冲突解决方法次之，提案评估第三，提案生成是最低的优先。不同的事件安排不是同时进行的。事件响应的原则是“先进，先操作”。

3.8. 规划策略

协同CAPP系统解决的是工艺计划问题。当冲突产生时，本着达成协议、解决问题的原则，它运用一种通用的语言，通过P代理之间的协同合作解决之。因此，系统通过提供一种信息交流、冲突解决结构和协商规则来解决协商问题。

3.8.1. 定义

第二十三本参考书中提到，问题、提案、评估、冲突定义如下：假设问题为 *P*=*{Or, G, C, I}*，其中Or代表初始的问题，G是工艺计划的一系列目的，C是问题的约束条件，I代表诸如部分设计数据等的初始的信息。提案 *Q*=*{ Ow_P, Ac_P, Exp_Q, Cf }*，*Ow_P* 代表提议者，*Ac_P* 是解决既定工艺计划问题的方法， *Exp_Q* 是提案的解释，Cf是支持提案的依据。评估可以表示为 *E*=*{Ow_EÕ , Id_Q, Ac_EÕ, Ra, Re }*，其中 *Ow_EÕ* 是评估人，*Id_Q* 是提案Q本身，*Ac_EÕ*是对提案的一系列评语，Ra是对每个操作的等级评定，Re是对提案肯定或否定的评议。.当Re是否定时，冲突就产生了，它表示的形式为 *Cr* =*{Ow_Cr, Id_P, Ac_Cr, Exp_Cr}*。Ow_Cr代表冲突所有者，Id_P是相关的提案，Ac_Cr是一系列冲突操作，Exp_Cr是对冲突操作的解释。

D代理接受了问题 *P*=*{Or, G, C, I}* ，并提出了最初的解决方案，接着就提交给B代理的问题解决区域。所有注册过的P代理都会收到问题的有关信息。而相关的P代理才会检查问题，接着根据它们的技能、知识和观点开始生成计划提案。当一个P代理生成一个提案 *Q*=*{Ow_P, Ac_P, Exp_Q, Cf }时*，它就被送至B代理的提案区域和其他注册过的P代理。但当某个P代理正在处理其他提案的时候，它是不会被中断来处理这个提案的，仅仅是触发一个评估操作。这个操作首先要决定它是不是马上进行评估。如果不是，它就进入休眠状态，等待下一个提案的到来。如果P代理和这个提案有关，

它就评估提案，并把评估结果*E*={ *Ow_E*Õ, *Id_Q*, *Ac_E*Õ, *Ra*, *Re* }传给B代理的评估区域。如果有冲突，P代理就会检查结果*Cr*={ *Ow_Cr*, *Id_P*, *Ac_Cr*, *Exp_Cr*}来获得最终的评估结果。

在所有的P代理评估完这个新生成的提案后，那些认为这个提案冲突的P代理就会一起解决冲突*Cr*={*Ow_Cr*, *Id_P*, *Ac_Cr*, *Exp_Cr*}。冲突决议的结果是对提议方案的校订或放弃 。如果没有P代理检测到冲突，原来解决区域的部分计划模板就会更新。计划工艺继续进行直到计划模板遇到新的诸如计划目的和约束的要求。

3.8.2. 协调

问题的协调解决有赖于B代理和P代理之间的配合。B代理负责存储和发布每个工艺计划问题的公共信息。为了明显它被分成了四个不同的信息区域：问题、提案、评估和解决方案。问题区域包含了最初的问题描述和工艺计划问题的所有要求。提案区域把部分和完整的提案存储在一些由P代理抽象出来的层中。一个P代理生成的提案有可能被其他的P代理评估。如果提案中有不详细或不完整的工艺，其他P代理会把提案和评估一起提交给B代理的评估区域。评估区域把执行工艺计划时的冲突存起来，并在与冲突有关的P代理之间建立某种链接。由P代理发布的评估结果和冲突解决的推荐方案在评估区域被重新存储。解决方案区域把工艺计划的模板加到没有冲突的工艺当中，这些工艺是由P代理生成的。最终的解决方案再次被存储到B区域的解决方案区域。B代理的结构如图四所示。推理引擎控制了这四个区域之间的信息流。知识库包含了保证问题协调解决的时间和解决方案正确性的事件进程安排知识。

B代理监控了四个区域的数据。一旦P代理传输过来了一项提案，提案区域就会接收它。同时，P代理也会检查相同问题的提案数目是否比固定的更多价值。如果是，B代理就从提案区域的提案清单中选择最佳的提案作为潜在的解决方案，问题的工艺计划自此停止。如果不是，提案将被送往其他的P代理寻求意见。一旦收到P代理的评估，B代理就会检查评估是否有一种冲突的结果。如果确实有任何冲突，提案的P代理将会收到通知。B代理协调这些P代理通过协商来解决冲突。如果没有冲突，B代理就检查是否所有注册过的P代理都同意这个提案。如果是，这个提案就会作为解决问题的方法了。

3.8.3. 协作

正如前面所提到的那样，协同CAPP系统就像是由协同解决问题的代理组成的社会，每个P代理都相对独立并拥有自主知识库的专家系统。P代理独立地解决其特定范围内的问题。因此，它应具有独立社会成员的功能：

· 与其他代理交流时通用的语言;

· 包含协同修订方案时所需的充分目标和历史数据的知识描述;

· 解决问题时及时地提供信息;

· 整合外部解决方案的机制;

· 协商解决问题的机制;

· 外来事件达成内部议事日程的能力。

每个P代理与其他代理交流时使用的都是一种通用语言。提案的产生、评估和解决以及冲突的产生和解决都是按照P代理的内部区域知识来的。

4. 软件的落实

我们选择IBM ABE Toolkit作为协同CAPP系统运行的环境，选择VC++语言作为执行的语言。在Windows 95 或者Windows NT中，系统能够运行。

前面提到，协同CAPP系统的P代理都要用到三种类型的知识：区域知识、控制知识和冲突解决知识。为提高柔性和可分离性，区域知识进一步被分成了三个等级：普通级、商店级和机器级。安装了协同CAPP系统以后，普通级知识在任何状态下都能运行，没有考虑不同公司的需要。当系统被加强了以后，商店级和机器级的知识才能被不同的公司添加和修改。机器级知识仅仅应用于特定的机器。每个代理的知识被用来作为数据库或者文件夹。冲突解决处理器如图5所示。它含有两个功能器“查找冲突”、“解决冲突”，以及一个冲突解决事实文件夹和一个冲突解决规则文件夹。“查找冲突”被习惯于用来标出冲突的位置和环境，“冲突解决”用来解决冲突。冲突解决策略也存放在这个功能器中。

5. 案例学习

案例学习就是举例说明机械协同CAPP系统的各种特性。第一个例子说明基于一个不合理产品设计的反馈。在这个例子中，我们可以看到一个不合理的设计，协同CAPP系统会产生一个输出“未解决的冲突”并把冲突所在的位置和原因传送给D代理。如图六所示，一块横梁的外形尺寸为$50\times40\times30\ mm^3$

***5.1.* 原始数据**

一块外形尺寸为$50\times40\times30\ mm^3$的钢料。

平面1，位置矢量：0:0:0:90:0:0，50:40，50:40，公差：0.04;

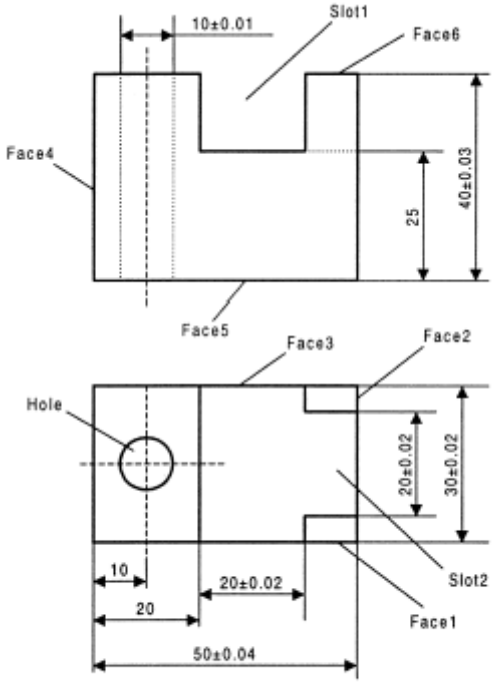

Fig. 6. Test component.

平面2,位置矢量: 0: 50: 0: 0: 90: 0, 40: 30, 40: 30，公差: 0.02;
平面3，位置矢量: y30: 0: 0: y90: 0: 0, 50: 40，50: 40，公差: 0.02;
平面4，位置矢量: 0: 0: 0: 0: y90: 0，40: 30, 40: 30，公差: 0.04;
平面5，位置矢量: 0: 0: 0: 0: 0: 90，50: 30, 50: 30，公差: 0.03;
平面6,位置矢量: 0: 0: 40: 0: 0: y90, 50: 30, 50: 30，公差0.03;
通槽1，开槽位置: 0: 20: 25: 0: 90: 90, 20: 30: 15，0: 0: 0，公差0.02;
通槽2，开槽位置: y5: 50: 25: 90: 0: 90, 20: 15: 10，0: 0: 0，公差0.02;
通孔,通孔位置: y15: 10: 0: 90: 90: 0, 10: 40, 0: 0,公差: 0.01;

5.2. 由初始数据得结果

= = =未解决冲突= = =

操作冲突1；操作代理；建议C操作；槽2: 不可利用的操作;

= = =和冲突= = =

切割机冲突0

= = =冲突是= = =

切割机冲突0，切割机代理；建议10操作;

面2: 拉削操作；面3: 拉削操作；槽1: 拉削操作；槽2: 研磨操作；槽2，拉削操作;

= = =未解决冲突= = =

操作冲突3，操作代理，操作决策1 孔: 不可利用的操作;

= = =和冲突= = =

机械冲突1;

= = =冲突是= = =

机械冲突；机械代理；操作决策1 孔: 排水槽操作;

在这个例子中，仅仅列举了三个P代理的简单应用，同时也说明了P代理之间如何协调解决冲突。B代理接收到了D代理传过得产品设计数据，并生成了“问题 0”。问题 0”被传给Op代理、来生成提案。Op代理生成一个提案“提案 0 Op”并传给B代理。B代理把它传给另两个代理: Current代理和机器代理。Current代理找到“冲突 0

Cutter”和 The“提案 0 Op”之间的冲突。机器代理同样和“提案 0 Op”不匹配，生成一个冲突“冲突 0 机器”。这两个冲突都会传给Op 代理。收到“冲突 0 Cutter”后，Op代理解决不了这个冲突，于是生成“未解决的冲突”使“冲突 1 Op ”对“冲突 0 Op ”做出反应。“冲突 1 Op ”和“冲突 0 Op ”都会反馈给D代理。同时，Op代理也会生成一个解决方案对“冲突 0 机器”做出反应。但是，机器代理不会认同“方案1 Op”，把“冲突1 机器”回复给Op代理。收到这个冲突和原来来自机器代理的冲突后，Op代理解决不了“冲突1 机器”又生成一个新的未解决的冲突“冲突 3 Op ”。“冲突 3 Op ”和“冲突 1机器 ”都会反馈给D代理。问题的解决自此终止。在这个例子中，由于未解决冲突的存在，没有合理方案的生成，仅仅把冲突反馈给了D代理。结果如表1所示。

5.3. 数据的变更

接着，我们看一个成功工艺计划是如何产生的。这个物体的空间尺寸和前面的一样，仅仅是“槽1，槽2”和孔的工差不同。在这个例子中，同样引入三个P代理。

槽1，通槽的位置：0:20:25:0:90:90，20:30:15,0:0:0，公差：0.2;

槽2，通槽的位置：y5:50:25:90:0:90,20:15:10，0:0:0，公差:0.2;

孔，通孔的位置：y15:10:0:90:90:0，10:40，0:0,公差:0.1;

5.4. 数据变更后的结果

= = =平面是= = =

平面19880718150407，局部加工，协同计算机辅助工艺编制，

平面1特征：操作方式FF研磨，平面1操作2，加工机床是高速铣床XH715:15.:1500.:7.5:6000; 切割机：米克朗五坐标轧齿切割机1:3.:0.5:600.:30;

说明：刀具代理从50.:40.开始切割;

平面1特征：操作方式射频研磨，平面1操作2，加工机床是高速铣床XH715:15.:1500.:7.5:6000; 切割机：米克朗五坐标轧齿切割机1:3.:0.5:600.:30;

说明：刀具代理从50.:40.开始切割;

平面4特征：操作方式FF研磨，平面4操作2，加工机床是高速铣床XH715:15.:1500.:7.5:6000; 切割机：米克朗五坐标轧齿切割机1:3.:0.5:600.:30;

说明：刀具代理从40.:30.开始切割;

平面4特征：操作方式射频研磨，平面4操作1，加工机床是高速铣床XH715:15.:1500.:7.5:6000; 切割机：米克朗五坐标轧齿切割机1:3.:0.5:600.:30;

说明：刀具代理从40.:30.开始切割;

平面5特征：操作方式FF研磨，平面5操作2，加工机床是高速铣床XH715:15.:1500.:7.5:6000;切割机:米克朗五坐标轧齿切割机1:3.:0.5:600.:30;说明:刀具代理从50.:30.开始切割;

平面5特征：操作方式射频研磨，平面5操作1，加工机床是高速铣床XH715:15.:1500.:7.5:6000;切割机:米克朗五坐标轧齿切割机1:3.:0.5:600.:30;说明:刀具代理从50.:30.开始切割;

平面6特征：操作方式FF研磨，平面6操作2，加工机床是高速铣床XH715:15.:1500.:7.5:6000;切割机:米克朗五坐标轧齿切割机1:3.:0.5:600.:30;说明:刀具代理从50.:30.开始切割;

平面5特征：操作方式射频研磨，平面6操作1，加工机床是高速铣床XH715:15.:1500.:7.5:6000;切割机:米克朗五坐标轧齿切割机1:3.:0.5:600.:30;说明:刀具代理从50.:30.开始切割;

槽1特征:操作方式射频研磨,槽1操作1,加工机床是高速铣床XH715:15.:1500.:7.5:6000;切割机:米克朗五坐标轧齿切割机2:5.:0.5:550.:30;说明:刀具代理从20.:30.:15开始切割;

槽2特征:操作方式射频研磨,槽2操作1,加工机床是高速铣床XH715:15.:1500.:7.5:6000;切割机:米克朗五坐标轧齿切割机2:5.:0.5:550.:30;说明:刀具代理从20.:15.:10开始切割;

孔特征:操作方式绞刀，孔操作2，加工机床钻床ZA5032:45.:2000.:2.2:9800.，切割机:大型绞床1a3:10.:0.25:100.:,说明:刀具代理从:10.:40.开始切割;

孔特征:操作方式钻头，孔操作1，加工机床钻床ZA5032:45.:2000.:2.2:9800.，切割机:盘旋钻床装置1a10:10.:0.2:500.:,说明:刀具代理从:9.9:40.开始切割;

平面2特征:操作方式FF研磨,平面2操作4,加工机床卧式磨床:M7120A:3000.:3000.:4.225:9999.,切割机:轮式磨床11a1:15.:2.5e-002:6000.:2.e-003,说明:刀具代理从:40.:30.开始切割;

平面2特征:操作方式射频研磨,平面2操作3,加工机床卧式床:M7120A:3000.:3000.:4.225:9999.,切割机:轮式磨床11a1:15.:2.5e-002:6000.:2.e-003,说明:刀具代理从:40.:30.开始切割;

平面2特征:操作方式射频研磨，平面2操作2，加工机床高速铣床:XH715:15.:1500.7.5:600.,4.225:9999.,切割机:米克朗五坐标轧齿切割机1:3.:0.5:600.:30;说明:刀具代理从:40.:30.开始切割;

平面2特征: 操作方式RE研磨，平面2操作1，加工机床高速铣床:XH715:15.:1500.:7.5:600.,4.225:9999.,切割机: 米克朗五坐标轧齿切割机1:3.:0.5:600.:30; 说明:刀具代理从:40.:30.开始切割;

平面3特征: 操作方式FF研磨，平面3操作4，加工机床卧式磨床:M7120A:3000.:3000.:4.225:9999.,切割机: 轮式磨床11a1:15.:2.5e-002:6000.:2.e-003,说明: 刀具代理从:50.:40. 开始切割;

平面3特征: 操作方式射频研磨，平面3操作3，加工机床卧式床:M7120A:3000.:3000.:4.225:9999.,切割机: 轮式磨床11a1:15.:2.5e-002:6000.:2.e-003,说明: 刀具代理从:50.:40. 开始切割;

平面3特征: 操作方式FE研磨，平面3操作2，加工机床高速铣床:XH715:15.:1500. :7.5:600.,4.225:9999.,切割机: 米克朗五坐标轧齿切割机1:3.:0.5:600.:30; 说明:刀具代理从:50.:40.开始切割;

平面3特征: 操作方式RE研磨，平面3操作1，加工机床高速铣床:XH715:15.:1500. :7.5:600.,4.225:9999.,切割机: 米克朗五坐标轧齿切割机1:3.:0.5:600.:30; 说明:刀具代理从:50.:40.开始切割;

B代理收到来自D代理的产品设计数据，生成问题0。Op代理、Cutter代理和机器代理收到问题0后生成提案。Op代理生成“提案0Op”，反馈给B代理。B代理转发给其他两个P代理: Cutter代理和机器代理。机器代理同意“提案0Op”。而The Cutter 代理与“提案0Op”有冲突“冲突0 Cutter ”。冲突被反馈给 Op代理。收到冲突后，Op代理生成“解决方案0Op”来应对“冲突0 Cutter ”。“解决方案0Op”被发给Cutter代理。这次，Cutter代理同意了这个方案。当Op代理收到“解决方案0Op”后，它就根据解决方案的原则生成“提案1Op”。“提案1Op”被送到其他的P代理去评估。在机器代理和Cutter代理把自己的观点加到这个方案时，所有的三个P代理都同意了新的方案“方案2机器”。 B代理检察“方案2机器”并确认这三个P代理都同意这个方案。B代理就生成解决“问题0”的方案了。解决方案19980718150407就发给了D代理。同时，B代理也会通知所有的P代理这个问题已经告一段落了。表2 显示的是解决方案提案的评估。

这表明，工艺计划的协调是在P代理之间完成的。一旦问题传到协同CAPP系统，B代理就会马上把它转发给所有注册过的P代理。当一个P代理生成提案时，又会马上传给B代理。B代理检查完提案后又通知其他P代理。相关的P代理对提案评估完后，会给出一个评估结论。这个结论要么同意或冲突，要么同意或者附上一个新提案。提议者

试着解决这个冲突。当前的问题或者解决或者解决不了。解决不了的就反馈给D代理。如果冲突解决了并且方法合适，一个新的符合要求的提案就产生了。如果不合适，这两个P代理就会通过谈判来解决。如果其他P代理都同意某个提案，B代理就模仿这个提案来提出解决方法。一旦方案好了，B代理就通知P代理终止工艺计划。所有新安排的任务也会终止。问题提案都是通过协商产生的。没有哪个P代理能做好一个完整的解决方法。每个P代理只是完成一部分。协商期间，仅仅冲突的所有者和提案的所有者参与冲突的解决和评估。这样就可以降低解决问题的难度。通过以上两个例子，我们可以看出机械协同CAPP系统能够成功地解决工艺计划的问题。.系统也就达到了自主性、柔性、通用性、模块性和可分离性的要求。

6.结论

本文引入了协同CAPP的代理模式。这个新的模式充分地利用了各个代理，并试图达到自主性、柔性、通用性、模块性和可分离性的要求。通过对所提供模式的使用，一个实验性质的机械协同CAPP系统已经开发出来了。这个协同CAPP系统跟已有的CAPP系统有显著的差异。它利用了协同的方式使各个代理和自身的专家系统很好的结合起来。系统中的每个代理都处理和自己领域相关的工艺计划。这与其他系统自始至终只用一个专家系统是有天壤之别的。因此，这个系统具有柔性和可升级性。在科技进步日益加速和频繁的今天，这种特性对工艺计划的制定和修改是大有裨益的。

本文使用了一种典型的机械组件测试了机械协同CAPP系统的性能。结果表明这个系统能有效地处理事件，效果很好。它能根据产品的设计信息和已有的材料生成合理的工艺计划。系统满足了预定的设计要求。实际应用中，协同CAPP系统具有以下特点:

- 在不影响系统正常运行的情况下，它的P代理能在任何时间下方便的加入和删除;
  在不不影响其他进程的情况下，它能对个别代理进行升级和更新；这样，系统就满足了模块化和柔性
  特征的要求。
- 每个P代理仅生成在其自己知识范围内的方案。因此，复杂的问题就可以通过模块分解成很多简单的子问题。
- 问题的最终解决方案是各个P代理方案的综合体。
- 根据不同的需要，系统可以单独发展某个P代理或者某个协同知识的分析计划。这个特性简化了CAPP系统的实行。

运用先进的代理技术，本文模拟了工艺计划代理。根据这个模型，各个代理协同工作，而不是分成独立的部分。冲突解决策略的加入就是为了适应这种系统的。另外，

工艺计划知识库由各个独立发展的知识库组成。这样，各个推理引擎所需的搜索空间大大减少。基于这样的协同CAPP系统，最优化的工艺计划很容易就完成。试验表明，完整的产品设计信息能很容易的加入协同CAPP系统的框架 。它能修复部分不合理的设计。这个协同CAPP系统为CAPP的发展提供了一种新方法。它提供了一种新的开放性的框架结构，很好的适应了分布式CAPP系统的发展潮流。这种改进应是未来研究的重点。一般来说，这种试验性的系统仅包含三个P代理。其他P代理如特征识别等可能在以后加入这种系统。

参考文献

[1]. Alting, H. Zhang, Computer aided process planning: the state-of-the-art survey, International Journal of Production Research 4（1989）553－585.

[2]T.C. Chang, Expert Process Planning for Manufacturing, Addison-Wesley Publishing, 1990.

[3]T.C. Chang, R.A. Wysk, An Introduction to Automated Process Planning Systems, Prentice-Hall, Englewood Cliffs, NJ, 1985.

[4]B.J. Davies, I.L. Darbyshire, The use of expert systems in process-planning, Annals of the CIRP 33（1984）303－306.

[5]A. Sergio, R. Maurer, Automatic Process Planning for Turned Parts, CAPE’88, Edinburgh, UK, 1998, pp. 293－300.

[6]K.E. Hummel, S.L. Brooks, Using hierarchically structured problem solving knowledge in a rule-based process planning system, in: M.D. Oliff Ed. , Expert Systems and Intelligent Manufacturing, 1998, pp. 120－137.

[7]F.J.A.M., VanHouten, et al., PART: A Feature Based Computer Aided Process Planning System, in: Proceedings 21st CIRP Seminar on Manufacturing System, 1, 1990, pp. 25－39.

[8]P. Gu, D.H. Norrie, Intelligent Manufacturing Planning, Chapman & Hall, London, 1995.

[9]SCRA Team report, Final Report for the RAMP Site Proveout of STEP Filesets Project, http://ramp.scra.org/pnep.html#papers, 1996.

[10]J. Kempenaers, J. Pinte, J. Detand, J.-P. Kruth, A collaborative process planning and scheduling system, Advances in Engineering Software 25（1996）3－8.

[11]R.C. Mark, et al., PACT: An Experiment in Integrating Concurrent Engineering Systems, IEEE Computer, January 1993, pp. 28－37.

[12]K.j. Werkman, Using negotiation in DAI to support concurrent engineering, in: P. Gu, A. Kusiak Eds. , Concurrent Engineering: Methodology and Applications, 1993, pp. 175－
204.

[13]G.Q. Huang, An agent-based framework for cooperating expert systems in concurrent engineering, Engineering Application of Artificial Intelligence 6 1994 685－693.

[14]A. Molina et al., A review of computer-aided simultaneous engineering systems, Research in Engineering Design 7（1995）38－63.

[15]P.S.Y. Wu, F.L. Zhao, A framework of cooperating expert system to support CAPP, in: J.-G.C. Jacob, A. Mital（Eds.）, 1st Annual International Conference on Industrial Engineering Applications and Practice I, Houston, USA, International Journal of Industrial Engineering, TX, December 1996, pp. 47－52.

[16]F.L. Zhao, P.S.Y. Wu, A cooperative framework for process planning, International Journal of Computer Integrated Manufacturing 17 1998 .

[17]DARPA Knowledge Sharing Initiative External Interfaces Working Group, Specification

of the KQML, Agent-Communication Language, 1993.
[18]M.F. Arnett, Inside TCPrIP, New Riders Publishing, 1994.
[19]P.E., DeGarmo, T.J. Black, R.A., Kohser, Materials and Processes in Manufacturing, Macmillan, London, 1984. ( ) F.L. Zhao et al.rComputers in Industry 41 2000 83–97 97
[20]United States Cutting Tool Institute, Metal Cutting Tool Handbook, Industrial Press, 1989.
[21]R. Zhao, Metal Cutting and Machining Engineer—Handbook (in Chinese), Shanghai Scientific Technology Publishing Press, 1990
[22]M.R. Adler et al., Conflict resolution strategies for nonhierarchical distributed agents, Distributed Artificial Intelligence 2 1989 139–161.
[23]F. Polat, S. Shekhar, H.A. Guvenir, Distributed conflict resolution among cooperating expert systems, Expert Systems 10 4 1993 .
[24]IBM Intelligent Agent Center, IBM Agent Building Environment DeveloperToolkit, Level 5, 1997.

Dr.SK.Tso 博士从香港大学获得理学士学位，并且在美国的伯明翰大学获得理学硕士和博士学位。他在香港城市大学拿到机电一体化和自动化的教授职位以后在香港大学服务了很长一段时间，他现在是智能设计、自动化操作和制造业中心的主任；他是电机工程师协会的特殊会员，并且是电气电子工程师协会的高级会员。

F.L.Zhao 在浙江大学获得理学士和理学硕士学位，他现在是香港大学的在职博士。

Dr.Paul.S.Y.Wu 博士在国立成功大学获得理学士学位、在纽约大学获得硕士学位和在美国的辛辛那提大学获得博士学位都是关于机械工程方面的，他是 ASME 和 SPIE 的会员。

A cooperative agent modelling approach for process planning

F.L. Zhao, S.K. Tso), Paul S.Y. Wu
Department of Manufacturing Engineering and Engineering Management, City Uni ersity of Hong Kong, Tat Chee A enue, Hong Kong,China

Received 30 November 1998; accepted 25 January 1999

Abstract:

A well designed computer-aided process planning（CAPP）system bridges the gap between CAD and CAM. A number of systems have recently been developed relying on a stand-alone expert system. However, because of over-complexity, many such systems cannot be effectively applied to industrial enterprises in practice. Moreover, the modern computer integrated manufacturing system(CIM)requires the CAPP system to be extendible and flexible for practical industrial applications. It is hardly possible to develop the extensive industrial CAPP system by using only one large expert system. To overcome these weaknesses, a new cooperative agent model is presented for process planning in this paper that satisfies five major requirements: Autonomy, Flexibility, Interoperability, Modularity and Scalability. In accordance with this framework proposed, a machining cooperative process planning system(Machining CoCAPP)is specifically developed for demonstration purpose. The system modelling, agent structure design, cooperation and coordination mechanism, and case study of the Machining CoCAPP are presented.

Keywords: CAPP; Cooperative agent; Modelling; CIM

1. Introduction

Process planning provides information to the shopfloor on how to produce the designed products. It addresses each part of the product separately and collectively. It defines the process, cost and production lead time under the constraints such as the designed geometry, material, quantity, machine and tooling availability, labour capacity and suitability, Corresponding author. etc. In the past, process plans were often generated by human process planners who had plenty of manufacturing domain knowledge and worthy experience.In the recent decades, computer technologies have stimulated the advance toward computer- aided process planning(CAPP).

Generally, there are two CAPP approaches: variant and generative. The variant approach is a data retrieval and editing method. Some standard or mature process plans are collected based on the group technology and stored in a database. When a new part is required to be

produced, a similar process plan is retrieved from the database and edited to adjust it to suit the new part. The generative approach is a knowledge-based method which automatically generates a process plan according to the part's features and manufacturing requirements.

The success of the variant approach depends on the group technology, planner's experience and a sufficient collection of standard or mature process plans. This method is especially suitable for companies with few product families and a large number of parts per family. Most earlier CAPP tools can be categorized under the variant process planning approachw1x. Typical examples are CAPP [2], MIPLAN[3], etc. The generative process planning approach has attracted more attention in recent years. It offers a potential of producing an optimal plan.Typical examples are APPAS [2] EXCAP [4], KRONOS [5], XCUT [6], QTC(Quick turnaround cell)[2], PART [7], OOPPS (objectoriented process planning system)[8], MePlans[9], COMPLAN Process Planner (CPP)[10], etc. Generative process planning systems are mostly oriented towards the needs of large companies and research organizations, especially those which have a number of products in small lot sizes. However, there is still difficulty in developing a truly generative process planning system which can meet industrial needs and provide an appropriate and compatible generic framework, knowledge representation method, and inference mechanism.

Cooperative agent systems attempt to distribute activities to multiple specialized problem solvers and to coordinate them to solve complex problems [11–14]. A cooperative agent system consists of many individual agents with cooperation engines. Each agent which has its own knowledge base and inference engine is responsible for a specific task. It provides a cooperation interface to communicate with other agents in the cooperative envir -onment. A different language and different knowledge representation may be employed by each agent which may well be located in a different machine. Such a system organiza-tion provides an integration environment of heterogeneous and scalable architecture suitable to solving different problems.

2. Process planning problem

A machining process generally involves many machine tools, operations, fixtures, and cutting tools. Its planning requires knowledge from diversified fields. Generally, a machining process planning includes the following parts:

¯feature recognition;

- machining operation selection;
- machine selection;
- cutting tool selection;
- fixture selection and design;
- sequencing operation and cost estimate.

The feature recognition part identities manufacturing features from the product design data. The machining operation selection part selects the relevant machining operation according to the feature characteristics and the manufacturing environment. The required machine equipment is selected for implementing the selected operations after considering the nature of the parts and the machine processing capabilities such as the working volume, accuracy, power, fixturing, and other functions. The fixture selection part chooses the fixtures according to the part geometric shapes and dimensions as well as manufacturing features. The main concerns of the cutting tool selection include the tool types, materials, shapes, and tool dimensions.

3. The proposed method

From the technological viewpoint, CAPP is still a very complex and difficult problem. Many research efforts have focused on CAPP system development, using different methodologies and strategies. However, most systems are developed by using standalone expert systems. Due to the complexity of CAPP, such a system structure is hardly able to solve the problems normally found in the manufacturing industry.

A cooperative CAPP framework is proposed to reduce the limitation of currently available CAPP systems. In particular, it highlights the requirements that a modern CAPP system should meet in order to facilitate practical development: flexibility, modularity, interoperability, autonomy, and scalability.

- Autonomy means that the CAPP system is developed as independent system. Once developed, it can readily be integrated into the CAD/CAM system. Each agent is also treated as an independent and autonomous system.
- Flexibility permits new technologies and new methods to be easily added into the CAPP system.
- Interoperability permits multiple heterogeneous machines or approaches to work smoothly together in solving problems.

¯Modularity enables the CAPP to function as an integration of multiple subsystems.

¯ Scalability offers the ability to scale the CAPP system architecture according to the user's transaction requirements.

The design of the CoCAPP framework is discussed with relation to a machining process planning(Machining CoCAPP). Its structure, implementation, and its application case study are detailed in this paper.

3.1. Overview of Machining CoCAPP

The overall structure of the Machining CoCAPP system in the integration environment of CAD/CAPP/CAM is shown in Fig. 1. The following six process planning agents (P-agents)are in general included in the implementation.

¯feature recognition agent;

¯machining operation selection agent;

¯machine selection agent;

¯cutting tool selection agent;

¯fixture selection and design agent;

¯sequencing operation and cost estimate agent.

Each agent may run in a different computer connected to the internet. The B-agent supplies the global state information of the problem and monitors the operational dependencies among all the individual P-agents. Once there is a problem, the B-agent will notify all registered P-agents for actions. The results from the P-agents will be posted on the B-agent. The B-agent maintains the consistency and integrity of the decision space within the given constraints.The product design data are sent to the CoCAPP system through the D-agent from the computer-aided design system. The CoCAPP system gets the production constraints from the scheduling/ shopfloor system. The CoCAPP system generates the process plans according to the product design data and production constraints. If no accept-able results can be obtained, the CoCAPP system will feed the conflict information back to the design department or other relevant departments. The feasible process plan alternatives will eventually be transmitted to the scheduling/shopfloor system for scheduling.

3.2. Agent infrastructure

The general structure of process planning agents(P-agent) is shown in Fig. 2. The agent is composed of four parts: agent controller, inference engine, functional adapters, and

application libraries. The configuration file is used to construct the agent. The rules and facts form the application libraries. The agent controller, rule-based engine, internet adapter, file adapter, keyboard adapter, information view adapter, and schedule adapter are the commonly used components for all the P-agents. Other adapters are also shown in Fig2.

The solver adapter is a very important adapter in the P-agents because it is used to accomplish proposal generation, conflict resolution and proposal evaluation of the process planning. In order to utilize the knowledge of each domain in the combination of production rules and objects, the adapter is differently implemented for different agents. It must be specifically designed to deal with the knowledge of object-oriented description.

The database adapter is used to store data useful to the B-agent and P-agents, such as problem definitions proposals, conflicts, evaluations, solutions, etc.

Because KQML [17] is the most commonly used language for communication among agent-based programs, particularly when they are autonomous and asynchronous, the Co-CAPP system has chosen a KQML-based communication protocol as a communication language used by each agent. The NetKQML adapter is used to communicate with the B-agent According to the requirements of KQML transport, agents are connected by unidirectional communication links that carry discrete messages. These links may have a finite message transport delay associated with them. When an agent receives a message, it knows from which incoming link the message has arrived; when an agent sends a message, it directs it to the intended outgoing link. Messages to a single destination arrive in the order they were sent; message delivery is reliable. The socket concept of Windows 95 is used to implement the NetKQML adapter. The TCP/IP is a protocol for socket communication transport w[18].

The information view adapter is used to display information on the monitor.

The time adapter is used to count time in the course of process planning.

The schedule adapter is responsible for scheduling events such as proposal generation, proposal evaluation, and conflict resolution. When an incoming proposal from another agent is detected, the adapter schedules an evaluation event. When a process planning problem is detected, it then schedules a proposal generation event. Both kinds of events will be assigned a priority at the same time of event generation. If a conflict resolution is

required, then the adapter schedules a conflict resolution event and assigns it the highest priority. Whenever a negotiation is required, the adapter immediately suspends other events and fires a negotiation effector.

When a problem arrives, the P-agent first invokes the effector ‘WatchProblem’ to examine the problem.Afterwards, it creates a ‘problem examined’event. At the same time, it generates the facts of the problem. Responding to the ‘proposal generationtask’ event, the effector ‘GenerateProposal’ generates a proposal and creates a proposal ‘Generated-Event’. This event results in the facts of the proposal.

Responding to the task event created by the schedule adapter, when the task name is ‘proposal generation’, the incoming proposal is evaluated. The adapter checks if the adapter has already generated an interacting proposal. If yes, it links the two proposals and notifies the proposal originators of its intent to evaluate the new proposal. In this case, it evaluates the already generated proposal. If the proposal adapter has not yet generated an interacting proposal, the adapter searches the scheduled pending tasks for related generation tasks. It checks to see if it has already started working on a proposal or is planning to start. If yes, it links the evaluation and generation tasks. In this case, it evaluates the proposal with the triggering proposal generation. If the triggering proposal is acceptable, it may not be necessary to generate a separate proposal or a proposal can be generated which is tailored to integrate smoothly with the triggering proposal. If there are no current plans to work on a related proposal, the evaluation task only evaluates the incoming proposal according to agent knowledge. The P-agent which originated the triggering proposal is then notified.

If a conflict is notified from another agent, the conflict resolution responds to this event to resolve the conflict. After the adapter responds to the conflict event, it first judges the conflict situation. Then it invokes the relevant strategy to resolve the conflict.

3.3. Knowledge representation

The CoCAPP system’s knowledge is about product representation and agent knowledge. Each P-agent in the CoCAPP system has three types of knowledge: domain knowledge, control knowledge, and conflict resolution knowledge. The B-agent has only control knowledge. The part representation is about the problem description. It is commonly shared by all agents within the CoCAPP system.

3.4. Part representation

A part in the Machining CoCAPP system consists of two kinds of data. One kind is about the constraints to the generating plan. Another kind is about its geometric information. They are described as follows:

¯CONSTRAINT (production time, cutting force, machining power);

¯PART (Name; Type; MaxSize; Material; InitSta-tus; HeatTreat; Features).

The Name is the identifier of a part. It must be unique. The Type is the keyword of the part to show its outlook shape. The MaxSize gives the maximum envelope size of the part. The Material is the material kind of the part. The InitStatus is the raw status of workpiece with which the part can be fabricated.The HeatTreat is about the part heat-treatment condition.The Features are a collection of all features. A feature is represented as follows:

¯FEATURE (Name; Type; Location; FinalSize;InitSize; Hardness; Tolerances).

The Name is the identifier of a feature. It must be unique. The Type is the keyword of the feature. The feature Location is about the original position and directional vector of the feature. The feature size,including final size(FinalSize).and initial size(Init-Size), is the dimensional value of the feature. The Hardness is about the feature's hardness. The Tolerances are about dimensional and geometric tolerances. A keyword is assigned to tolerance to distinguish the tolerance content.

3.5. Domain knowledge

The domain knowledge of each P-agent is about the descriptions of its process planning capabilities, and used to generate proposals, evaluate proposals, and resolve conflicts. Different agents may have different formats for domain knowledge representations such as databases, analytical algorithms, etc. The domain knowledge can be extracted from manufacturing handbooks such as[19–21]. Each P-agent has a different domain knowledge content.

For example, the operation selection agent is used to generate machining operation alternatives for defined features of parts. For each given feature, there may exist more than one possible operation. Some traditional machining methods such as forging, die casting, drilling, turning, milling, boring, shaping, grinding, lapping, honing, and diamond boring, etc.have been built into the domain knowledge base. Its content includes the relationship between operations with parameters such as machinable feature, workpiece material,

tolerances, preparatory operation, time calculation equation. The knowledge is represented as facts in a semantic net as shown in Fig. 3. In the knowledge base, an operation fact is stored in one node. All facts are stored in a list.

The proposal generation, proposal evaluation, and conflict resolution strategies are embedded in the solver adapter of each P-agent.

3.6. Conflict resolution knowledge

The conflict resolution strategies include two categories: domain-dependent and domain-independent.The domain-dependent strategies mainly involve how to suggest further measures when one P-agent conflicts with the other P-agent's proposals. Each P-agent has its own suggestion strategies different from the other P-agent's. It also includes the explanation to the conflict resolution proposed. The domain-independent strategies are more common and can be the same for all the P-agents. They are designed as a set of conflict resolution facts with some basic guide-lines(conflict resolution rules)for deciding which domain-dependent strategy to apply and altering the resolution strategy in order to improve its understanding of the overall problem according to the other P-agent's action of conflict resolutions. The following strategies are used to resolve conflicts in the CoCAPP system as domain-independent conflict resolution strategies [22]

¯Compromise: finding an immediate proposal that is within an acceptable range;

¯GenerateConstrainedAlternatives: generating new alternatives based on constraints that are received from an inflexible agent or based on some other agent's partial solution;

¯CasebasedParameterSetRetrieval: finding a previous solution that succeeded in resolving a conflict involving a similar set of parameters.

3.7. Control knowledge

All the P-agents in the CoCAPP system must be able to perform a set of tasks with the same control knowledge:

¯generate new proposals;

¯evaluate proposals;

¯negotiate;

¯resolve conflicts.

The above tasks are scheduled before they are performed. Each task is scheduled as an event. The P-agents first respond to an event, then perform the relevant tasks. The control

knowledge of the P-agents is used to schedule tasks and to respond to events.

Each task is assigned a priority as an event. As previously discussed, the negotiation is the highest priority task. The conflict resolution shares the second highest priority. The proposal evaluation has the third highest priority. The proposal generation has the lowest priority. The scheduled tasks are performed asynchronously. The event is responded to according to the principle of 'first come, first served'.

3.8. Planning strategy

The CoCAPP system solves the process planning problem, using coordination and cooperation between the P-agents through a commonly shared language with the expectation about how to reach agreements when conflicts occur. Therefore, the system is designed to support cooperative problem solving by providing a communication and conflict resolution structure and the protocol required for cooperative interaction.

3.8.1. Definition

According to Ref. [23], the problem, proposal, evaluation, and conflict are defined as follows: Suppose that a problem is represented as P={Or,G,C, I}, where Or denotes the originator of the problem, G is the set of goals of process planning problem, C is the constraints for this problem, and I contains the initial information such as part design data. A proposal is in the form of Q={Ow_P, Ac_P, Exp_Q,Cf}, where Ow_P denotes the owner of the proposal, Ac_P is a set of actions to solve the given process planning problem, Exp_Q is the explanation to the proposal, and Cf is the confidence factor for the proposal. An evaluation result can be represented by Ev = {Ow_Ev ,Id_Q, Ac_Ev,Ra, Re}, where Ow_Ev denotes the owner of the evaluation, Id_Q is the identity of the proposal Q, Ac_Ev is the set of evaluation actions for the proposal, Ra is the set of ratings for each action, and Re is the overall result for the proposal which is either disagreement or agreement. When Re says disagreement, a conflict appears, expressed in the form Cr ={Ow_Cr, Id_P, Ac_Cr, Exp_Cr}, where Ow_Cr denotes the owner of the conflict, Id_P is the identity of the related proposal, Ac_Cr is the set of conflict actions, and Exp_Cr is the explanation for the conflict actions.

The problem solving is initiated by the D-agent accepting a problem and putting the problem definition P={Or,G,C, I}into the problem area of the B-agent. All registered P-agents are notified of the problem information. The interested P-agents will examine the

problem definition and start producing planning proposals related to their expertise, knowledge and viewpoints. When a P-agent generates a proposal Q={Ow_P, Ac_P, Exp_Q,Cf}, it is put into the proposal area of the B-agent. The proposal is posted to other registered P-agents. A P-agent is not interrupted if it is already working on another proposal, but immediately triggers an evaluation thread. This thread first determines whether it is going to criticize the proposal. If not, it will go to sleep and wait for another proposal to be asserted. If the P-agent is interested in the proposal, it then evaluates the proposal and posts the evaluation result Ev={Ow_Ev, Id_Q, Ac_Ev,Ra, Re}into the evaluation area of the B-agent. If there is a conflict, the result Cr={Ow_Cr, Id_P, Ac_Cr, Exp_Cr}will be examined in order to obtain the final evaluation result.

After all the interested P-agents finish evaluating the newly asserted proposal, those P-agents which identify the proposal under consideration as conflict work together to resolve the conflict Cr={Ow_Cr, Id_P, Ac_Cr, Exp_Cr}. The result of the conflict resolution is either a revision or an abandonment of the proposal scheme. When none of the interested P-agents detect any conflict related to the proposal, the partial planning template residing in the solution area is updated. The planning process continues until the planning template meets the requirements, including planning goals and constraints.

3.8.2. Coordination

The cooperative problem solving is carried out among the B-agent and P-agents. The B-agent is responsible for storing and announcing the public information involved in each process planning problem. It is partitioned into four areas for four distinct groups of information: problem, proposal, evaluation and solution. The problem area contains the initial problem definition and the overall requirements of the process planning problem. The proposal area stores partial and complete proposals at several layers of abstraction issued by the P-agents. A proposal from a P-agent might be evaluated by other P-agents.If there is any inaccurate or incomplete process in the proposal, other P-agents can put their critiques related with the proposal to the evaluation area of the B-agent. The evaluation area stores the conflicts that occur during the process planning, and provides a means of communication among the P-agents who are involved in the conflict. The evaluation results and conflict resolution recommendations issued by the P-agents are also recorded in the evaluation area. The solution area includes the evolving process planning template to

which non-conflicting process commitments produced by the P-agents are added. The final solution is recorded into the solution area of the B-agent. The organization of the B-agent is shown in Fig. 4. The inference engine provides the control of information flow among the four areas. The knowledge base contains the event scheduling knowledge for cooperative problem solving and the justifications of the solutions to the problems.

The B-agent monitors the data of the four areas. Once a proposal is received from a P-agent, it will be placed in the proposal area. At the same time, the P-agent will check if the number of proposals for the same problem is greater than a fixed value. If yes, the B-agent will choose the best proposal from the proposal list in the proposal area as a possible solution. The process planning for the problem will be terminated. If not, the proposal will be posted to other registered P-agents for their review. Once an evaluation is received from a P-agent, the B-agent will check if the evaluation has a conflict result. Ifthere is any conflict, the owner of the proposal will be notified. The B-agent will coordinate these P-agents having conflict to settle by negotiation. If there is no conflict, the B-agent will check if all the registered P-agents have agreed with the proposal. If yes, the proposal will be evolved into a solution to the problem.

3.8.3. Cooperation

As mentioned previously, the CoCAPP system environment is organized as a community of cooperative problem-solving agents, where each P-agent is a relatively independent and autonomous knowledge-based expert system. The P-agent solves problems in its limited domain independently. Therefore, it should have the capabilities to act as a member of a community. These capabilities include:

- a shared communication language with other agents;
- internal knowledge representations which capture sufficient goal and history information to allow for solution revision to be carried out cooperatively;
- provisions for sharing information in a timely manner for problem-solving;
- mechanisms for incorporating externally produced partial solutions;
- mechanisms for negotiation to settle conflicts;
- the ability to coordinate an internal agenda with external events.

Each P-agent communicates with other agents by using a common shared language. The proposal generation and evaluation, solution, and conflict generation and resolution are

produced according to the internal domain knowledge of the P-agent.

4. Software implementation

The IBM ABE Toolkit [24] is chosen as the development environment of the CoCAPP system. The Visual C++ language is chosen as the implementation language. The system can run in the platform of Windows 95 or Windows NT.

As mentioned, each P-agent in the CoCAPP system makes use of three types of knowledge: domain knowledge, control knowledge, and conflict resolution knowledge. In order to enhance the flexibility and scalability, the domain knowledge is further classified into universal-level, shop-level, and machine-level knowledge. The universal-level knowledge is applicable to any status without considering individual companies and is established when the CoCAPP system is in development and is often fixed after the system has been constructed. The shop-level and machine-level knowledge can be added and modified by individual companies when the system is scaled. The machine-level knowledge is only applicable to a specific machine. The knowledge of each agent is implemented as a database or file.

The conflict resolution handler is shown in Fig. 5. It consists of two effectors: 'WatchConflict' and 'ResolveConflict', as well as one conflict resolution facts file and one conflict resolution rules file. The 'WatchConflict' is used to map out the conflict problem space and conflict situation. The 'Resolve-Conflict' is used to resolve conflicts. The conflict resolution strategies are implemented in this effector.

5. Case study

The case study is used to illustrate the characteristics of the Machining CoCAPP system. The first example demonstrates the feedback due to an unreasonable product design. In this example, as an unreasonable part design is provided, the CoCAPP system generates an 'unresolved conflict' output and reports to the D-agent the locations and causes of the conflict. The part is shown in Fig. 6, a bar with an envelope size of $50 \times 40 \times 30$ mm^3.

5.1. Initial Part Design Data

timeless:7200;

PartOne;

Rectangular;

50:40:30;

MildSteel;

Bar;

;

Face1,FlatFace, 0:0:0:90:0:0, 50:40, 50:40, DimTol:0.04;

Face2,FlatFace, 0:50:0:0:90:0, 40:30, 40:30, DimTol:0.02;

Face3,FlatFace, y30:0:0:y90:0:0, 50:40, 50:40, DimTol:0.02;

Face4,FlatFace, 0:0:0:0:y90:0, 40:30, 40:30,DimTol:0.04;

Face5,FlatFace, 0:0:0:0:0:90, 50:30, 50:30, DimTol:0.03;

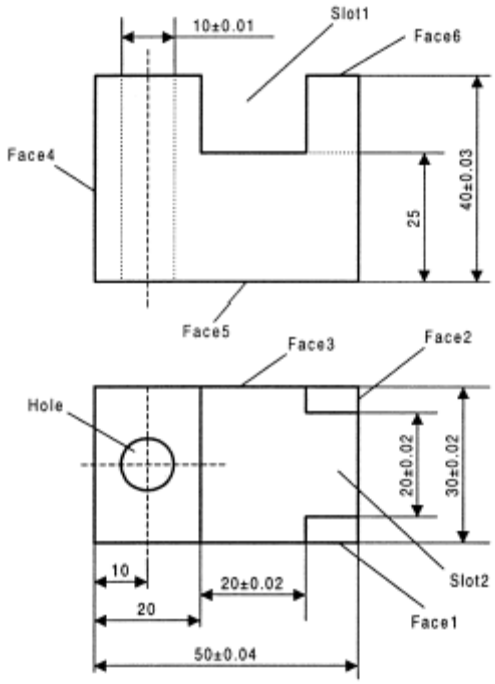

Fig. 6. Test component.

Face6,FlatFace, 0:0:40:0:0:y90, 50:30, 50:30,DimTol:0.03;

Slot1,ThroughSqSlot, 0:20:25:0:90:90, 20:30:15,0:0:0, DimTol:0.02;

Slot2,ThroughSqSlot, y5:50:25:90:0:90,20:15:10, 0:0:0, DimTol:0.02;

Hole, ThroughHole,y15:10:0:90:90:0, 10:40, 0:0,DimTol:0.01;

5.2. Planning result based on initial design data

= = =UNRESOLVED CONFLICT= = =

Conflict1Op;

Op-agent;

ProposalCOp;

Slot2:operation::,notavailable;

Slot2:operation::,notavailable

= = =AND CONFLICT= = =

Conflict0Cutter

= = =CONFLICT IS= = =

Conflict0Cutter;

Cutter-agent;

Proposal10Op;

Face2:operation:Broaching:,notsupportforop;

Face3:operation:Broaching:,notsupportforop;

Slot1:operation:Broaching:,notsupportforop;

Slot2:operation:FFGrinding:,notsupportforop;

Slot2:operation:Broaching:,notsupportforop

= = =UNRESOLVED CONFLICT= = =

Conflict3Op;

Op-agent;

Resolution1Op;

Hole:operation::,notavailable

= = =AND CONFLICT= = =

Conflict1Machine

= = =CONFLICT IS= = =

Conflict1Machine;

Machine-agent;

Resolution1Op;

Hole:operation:RHoning:,notsupportforop

In this example, only three P-agents are involved for purpose of simpler illustration, and cooperative process planning revealing unresolved conflicts among the P-agents is demonstrated. The B-agent receives the product design data from the D-agent and forms a problem ‘Problem0’. The ‘Problem0’ is sent to the Op-agent, Cutter-agent, and Machine-agent for them to generate proposals. The Op-agent generates a proposal ‘Proposal0Op’ and sends this proposal to the B-agent. The B-agent posts this proposal to the other two P-agents: Cutter-agent and Machine-agent. The Cutter-agent finds a conflict ‘Conflict0Cutter’ with ‘Proposal0Op’. The Machine-agent also disagrees with ‘Proposal0Op’ and finds a conflict ‘Conflict0Machine’. Both conflicts are posted to the Op-agent. After reviewing ‘Conflict0Cutter’, the Op-agent finds that it cannot resolve this conflict, and generates an ‘unresolved conflict’ reply ‘Conflict1Op’ to ‘Conflict0Cutter’.Both conflicts ‘Conflict0Cutter’ and ‘Conflict0Op’ are fed back to the D-agent. At the same time, the Op-agent generates a resolution ‘Resolution1Op’ in reply to

‘Conflict0Machine’. But the Machine-agent cannot agree with the resolution ‘Resolution1Op’, and a conflict reply ‘Conflict1Machine’ is posted to the Op-agent. After reviewing this conflict and former conflict from the Machine-agent, the Op-agent cannot resolve the conflict ‘Conflict1Machine’ and generates an unresolved conflict ‘Conflict3Op’. Both conflict ‘conflict3Op’ and ‘Conflict1Machine’ are fed back to the D-agent. The problem solving activity is terminated. In this example, due to the unresolved conflict existing, no plan is obtained. Only the unresolved conflicts are fed back to the D-agent. The results are summarized in Table 1.

5.3. Modified data

A different design is next used to demonstrate a successful planning process output. The part design has the same dimensional size as the previous example, but has different dimensional tolerance for the feature ‘Slot1,Slot2’ and Hole. Again three P-agents are involved in this example

Slot1,ThroughSqSlot, 0:20:25:0:90:90, 20:30:15,0:0:0, DimTol:0.2;

Slot2,ThroughSqSlot, y5:50:25:90:0:90,20:15:10, 0:0:0, DimTol:0.2;

Hole, ThroughHole,y15:10:0:90:90:0, 10:40, 0:0,DimTol:0.1;

5.4. Planning result based on modified data

= = =PLAN IS= = =

Plan19980718150407;

Machining;

CoCAPP;

PartOne;

feature:Face1,operation:FFMilling:Face1Op2:::,machine:VMillingMachine:XH715:15.:1500.:7.5:6000.,cutter:MFEndMill:MillingCutter5a1:3.:0.5:600.:30.,explanation:Cutter-agent:cuttingto:50.:40.;

feature:Face1,operation:RFMilling:Face1Op1:::,machine:VMillingMachine:XH715:15.:1500.:7.5:6000.,cutter:MFEndMill:MillingCutter5a1:3.:0.5:600.:30.,explanation:Cutter-agent:cuttingto:50.:40.;

feature:Face4,operation:FFMilling:Face4Op2:::,machine:VMillingMachine:XH715:15.:1500.:7.5:6000.,cutter:MFEndMill:MillingCutter5a1:3.:0.5:600.:30.,explanation:Cutter-agent:cuttingto:40.:30.;

feature:Face4,operation:RFMilling:Face4Op1:::,machine:VMillingMachine:XH715:15.:1500.:7.5:6000.,cutter:MFEndMill:MillingCutter5a1:3.:0.5:600.:30.,explanation:Cutter-agent:cuttingto:40.:30.;
feature:Face5,operation:FFMilling:Face5Op2:::,machine:VMillingMachine:XH715:15.:1500.:7.5:6000.,cutter:MFEndMill:MillingCutter5a1:3.:0.5:600.:30.,explanation:Cutter-agent:cuttingto:50.:30.;
feature:Face5,operation:RFMilling:Face5Op1:::,machine:VMillingMachine:XH715:15.:1500.:7.5:6000.,cutter:MFEndMill:MillingCutter5a1:3.:0.5:600.:30.,explanation:Cutter-agent:cuttingto:50.:30.;
feature:Face6,operation:FFMilling:Face6Op2:::,machine:VMillingMachine:XH715:15.:1500.:7.5:6000.,cutter:MFEndMill:MillingCutter5a1:3.:0.5:600.:30.,explanation:Cutter-agent:cuttingto:50.:30.;
feature:Face6,operation:RFMilling:Face6Op1:::,machine:VMillingMachine:XH715:15.:1500.:7.5:6000.,cutter:MFEndMill:MillingCutter5a1:3.:0.5:600.:30.,explanation:Cutter-agent:cuttingto:50.:30.;
feature:Slot1,operation:RFMilling:Slot1Op1:::,machine:VMillingMachine:XH715:15.:1500.:7.5:6000.,cutter:MFEndMill:MillingCutter5a2:5.:0.5:550.:30.,explanation:Cutter-agent:cuttingto:20.:30.:15.;
feature:Slot2,operation:RFMilling:Slot2Op1:::,machine:VMillingMachine:XH715:15.:1500.:7.5:6000.,cutter:MFEndMill:MillingCutter5a2:5.:0.5:550.:30.,explanation:Cutter-agent:cuttingto:20.:15.:10.;
feature:Hole,operation:RReaming:HoleOp2:::,machine:DrillPress:ZA5032:45.:2000.:2.2:9800.,cutter:ChunkingReamer:Reamer1a3:10.:0.25:100.:,explanation:Cutter-agent:cuttingto:10.:40.;
feature:Hole,operation:Drilling:HoleOp1:::,machine:DrillPress:ZA5032:45.:2000.:2.2:9800.,_ .cutter:TwistDrillTS :Drill1a10:10.:0.2:500.:,explanation:Cutter-agent:cuttingto:9.9:40.;
feature:Face2,operation:FFGrinding:Face2Op4:::,machine:FlatGrinder:M7120A:3000.:3000.:4.225:9999.,cutter:FGrindingWheel:GrindingWheel11a1:15.:2.5e-002:6000.:2.e-003,explanation:Cutteragent:cuttingto:40.:30.;
feature:Face2,operation:RFGrinding:Face2Op3:::,machine:FlatGrinder:M7120A :3000.:30

00.:4.225:9999.,cutter:FGrindingWheel:GrindingWheel1a1:15.:2.5e-002:6000.:2.e-003,explanation:Cutteragent:cuttingto:40.:30.;
feature:Face2,operation:FEMilling:Face2Op2:::,machine:VMillingMachine:XH715:15.:1500.:7.5:600.,cutter:MFEndMill:MillingCutter5a1:3.:0.5:600.:30.,explanation:Cutter-agent:cuttingto:40.:30.;
feature:Face2,operation:REMilling:Face2Op1:::,machine:VMillingMachine:XH715:15.:1500.:7.5:6000.,cutter:MFEndMill:MillingCutter5a1:3.:0.5:600.:30.,explanation:Cutter-agent:cuttingto:40.:30.;
feature:Face3,operation:FFGrinding:Face3Op4:::,machine:FlatGrinder:M7120A :3000.:3000.:4.225:9999.,cutter:FGrindingWheel:GrindingWheel1a1:15.:2.5e-002:6000.:2.e-003,explanation:Cutteragent:cuttingto:50.:40.;
feature:Face3,operation:RFGrinding:Face3Op3:::,machine:FlatGrinder:M7120A :3000.:3000.:4.225:9999.,cutter:FGrindingWheel:GrindingWheel1a1:15.:2.5e-002:6000.:2.e-003,explanation:Cutteragent:cuttingto:50.:40.;
feature:Face3,operation:FEMilling:Face3Op2:::,machine:VMillingMachine:XH715:15.:1500.:7.5:6000.,cutter:MFEndMill:MillingCutter5a1:3.:0.5:600.:30.,explanation:Cutter-agent:cuttingto:50.:40.;
feature:Face3,operation:REMilling:Face3Op1:::,machine:VMillingMachine:XH715:15.:500.:7.5:6000.,cutter:MFEndMill:MillingCutter5a1:3.:0.5:600.:30.,explanation:Cutter-agent:cuttingto:50.:40.

The B-agent receives the product design data from the D-agent and forms a problem Problem0. The Problem0 is sent to the Op-agent, Cutter-agent, and Machine-agent for them to generate proposals. The Op-agent generates a proposal Proposal0Op and sends this proposal to the B-agent. The B-agent posts this proposal to the other two P-agents: Cutter-agent and Machine-agent. The Machine-agent agrees with Proposal0Op. The Cutter-agent finds a conflict Conflict0Cutter with Proposal0Op. The Op-agent is informed of the conflict Conflict0Cutter. After reviewing Conflict0Cutter, a resolution Resolution0Op is generated in reply to Conflict0Cutter. Resolution0Op is posted to the Cutter-agent. This time the Cutter-agent agrees with the resolution. After the Op-agent receives resolution agreement to Resolution0Op, it starts forming a new proposal Proposal1Op based on the resolution results. Proposal1Op is posted to the other P-agents

for their evaluation. After the Machine- agent and Cutter-agent add their views to the problem solving, finally, all three P-agents agree with the proposal Proposal2Machine. The B-agent examines Proposal2Machine and finds that all the P-agents have agreed with Proposal2Machine. The B-agent starts composing a solution to Problem0. A solution Plan19980718150407 is sent to the D-agent. The B-agent also sends the message Complete to all the P-agents to inform them of the end of the problem solving stage. The evolution from a proposal to the solution is shown in Table 2.

It is shown that the cooperative process planning is carried out among the P-agents. Once a problem definition arrives in the CoCAPP system, the B-agent immediately posts it to all the registered P-agents. When a P-agent generates a proposal to the problem, it will immediately post it to the B-agent. The B-agent informs the other P-agents of the proposal after it examines the proposal. An interested P-agent will evaluate the proposal and give an evaluation result: either agreement, conflict, or agreement and with an added new proposal. The proposal owner will try to resolve the conflict. The presented problem is either resolved, or unresolved. The unresolved conflict will be fed back to the D-agent. If the conflict is resolved and the method of resolution is agreed, a new proposal will be proposed based on the resolution result. If the resolution method is still not agreed by the conflict owner, both P-agents involved will trigger a negotiation program to handle this conflict. If a proposal is agreed by all the P-agents, the B-agent will form a solution based on this proposal. Once a solution is generated, the B-agent will inform the P-agents to end the process planning. Any new scheduling task will be terminated. The solution to a problem is generated in a cooperative way. No one P-agent can generate a full solution. Each P-agent can only contribute a partial solution. During negotiation, only the conflict owner and proposal owner involved are invited to carry out the conflict resolution or evaluation. This reduces the difficulty of problem solving. From the two examples given, it is seen that the Machining CoCAPP system can successfully deal with the process planning problem. The designed system is able to meet the proposed five requirements: Autonomy, Flexibility, Interoperability, Modularity, and Scalability.

6. Conclusion

The cooperative agent model for CAPP was introduced in this paper. The model makes use of intelligent agents and tries to satisfy five major requirements simultaneously:

Autonomy, Flexibility, Interoperability, Modularity, and Scalability. An experimental Machining CoCAPP system has been developed by using the proposed model. The developed CoCAPP system is different from other CAPP systems available; it utilizes cooperative and coordination mechanisms built into distributed agents with their own expert systems. Each agent in this system deals with a relatively independent domain of process planning. This is in sharp contrast to other CAPP systems utilizing a single standalone expert system to perform the entire process planning. The system is hence flexible and upgradable. This feature is especially useful as the change of process planning methods or revision due to technology advances is increasingly more common and frequent.

In the paper, a typical mechanical component is considered to test the performance of the Machining CoCAPP system. The experimental results show that the system can effectively deal with the process planning problems. It can generate process plans according to the product design data and available manufacturing resources. The system has met the proposed design requirements. In particular, the Co-CAPP system has the following characteristics.

¯Its P-agents can be added and deleted at any time without affecting system operation, and can be individually updated without affecting the others, thus reflecting thmodularity and flexibility features embodied in the system.

¯ Each P-agent only generates a partial solution related to its knowledge; therefore, a complex problem can be decomposed into many simpler sub-problems on a modular basis.

¯ The whole solution to a problem is obtained by integrating each P-agent’s proposal together with other proposals.

¯Each P-agent can be an individually developed expert system or an analytical program with cooperation knowledge included according to demand. This autonomous feature greatly simplifies the implementation of the CAPP system.

This paper presents the method to model the process planning agent[P-agent]by using intelligent agent technology. According to this model, each agent is interoperable and not confined to any machine platform. The model for conflict resolution strategy is developed to suit the CoCAPP system. In addition, the process-planning knowledge base is divided into multiple knowledge bases which are independently established. This has greatly

reduced the search space of each inference engine. Based on the proposed CoCAPP system, the optimization of process plans would be feasible and easier to obtain. The experimental results have shown that the Co-CAPP framework can be easily integrated into the concurrent engineering environment to implement integrated product design; it can deal with unreasonable part designs. The proposed CoCAPP framework opens up a new approach to the CAPP development. It provides an open framework. It is very suitable for distributed CAPP system development. Further investigations should focus on the improvement and extension of the system. Currently, the experimental

Machining CoCAPP system only includes three P-agents. Other P-agents such as feature recognition, etc. may be added to the system in the work of future development.

References

[1]. Alting, H. Zhang, Computer aided process planning: the state-of-the-art survey, International Journal of Production Research 4（1989）553－585.

[2]T.C. Chang, Expert Process Planning for Manufacturing, Addison-Wesley Publishing, 1990.

[3]T.C. Chang, R.A. Wysk, An Introduction to Automated Process Planning Systems, Prentice-Hall, Englewood Cliffs, NJ, 1985.

[4]B.J. Davies, I.L. Darbyshire, The use of expert systems in process-planning, Annals of the CIRP 33（1984）303－306.

[5]A. Sergio, R. Maurer, Automatic Process Planning for Turned Parts, CAPE’88, Edinburgh, UK, 1998, pp. 293－300.

[6]K.E. Hummel, S.L. Brooks, Using hierarchically structured problem solving knowledge in a rule-based process planning system, in: M.D. Oliff Ed. , Expert Systems and Intelligent Manufacturing, 1998, pp. 120－137.

[7]F.J.A.M., VanHouten, et al., PART: A Feature Based Computer Aided Process Planning System, in: Proceedings 21st CIRP Seminar on Manufacturing System, 1, 1990, pp. 25－39.

[8]P. Gu, D.H. Norrie, Intelligent Manufacturing Planning, Chapman & Hall, London, 1995.

[9]SCRA Team report, Final Report for the RAMP Site Proveout of STEP Filesets Project, http://ramp.scra.org/pnep.html#papers, 1996.

[10]J. Kempenaers, J. Pinte, J. Detand, J.-P. Kruth, A collaborative process planning and scheduling system, Advances in Engineering Software 25（1996）3－8.

[11]R.C. Mark, et al., PACT: An Experiment in Integrating Concurrent Engineering Systems, IEEE Computer, January 1993, pp. 28－37.

[12]K.j. Werkman, Using negotiation in DAI to support concurrent engineering, in: P. Gu, A. Kusiak Eds. , Concurrent Engineering: Methodology and Applications, 1993, pp. 175－
204.

[13]G.Q. Huang, An agent-based framework for cooperating expert systems in concurrent engineering, Engineering Application of Artificial Intelligence 6 1994 685－693.

[14]A. Molina et al., A review of computer-aided simultaneous engineering systems, Research in Engineering Design 7（1995）38－63.

[15]P.S.Y. Wu, F.L. Zhao, A framework of cooperating expert system to support CAPP, in: J.-G.C. Jacob, A. Mital（Eds.）, 1st Annual International Conference on Industrial Engineering Applications and Practice I, Houston, USA, International Journal of Industrial Engineering, TX, December 1996, pp. 47－52.

[16]F.L. Zhao, P.S.Y. Wu, A cooperative framework for process planning, International Journal of Computer Integrated Manufacturing 17 1998 .

[17]DARPA Knowledge Sharing Initiative External Interfaces Working Group, Specification

of the KQML, Agent-Communication Language, 1993.
[18]M.F. Arnett, Inside TCPrIP, New Riders Publishing, 1994.
[19]P.E., DeGarmo, T.J. Black, R.A., Kohser, Materials and Processes in Manufacturing, Macmillan, London, 1984. ( ) F.L. Zhao et al.rComputers in Industry 41 2000 83–97 97
[20]United States Cutting Tool Institute, Metal Cutting Tool Handbook, Industrial Press, 1989.
[21]R. Zhao, Metal Cutting and Machining Engineer—Handbook (in Chinese), Shanghai Scientific Technology Publishing Press, 1990
[22]M.R. Adler et al., Conflict resolution strategies for nonhierarchical distributed agents, Distributed Artificial Intelligence 2 1989 139–161.
[23]F. Polat, S. Shekhar, H.A. Guvenir, Distributed conflict resolution among cooperating expert systems, Expert Systems 10 4 1993 .
[24]IBM Intelligent Agent Center, IBM Agent Building Environment DeveloperToolkit, Level 5, 1997.

Dr. S.K. Tso, obtained his BSc (Eng) degree from the University of Hong Kong, and his MSc and PhD degrees from the University of Birmingham, UK. He has taken up the Professorship of Mechatronics and Automation in the City University of Hong Kong after a long service at the University of Hong Kong. He is currently the Director of the Centre for Intelligent Design, Automation and Manufacturing. He is a Fellow of IEE and Senior Member of IEEE.
F.L. Zhao, obtained his BSc and MSc degrees from Zhejiang University, P.R. China. He is currently a PhD candidate at the City University of Hong Kong.
Dr. Paul S.Y. Wu, received the BS degree from Cheng-Kung University, the MS degree from New York University, and PhD degree from the University of Cincinnati, all in mechanical engineering. He is a member of ASME and SPIE.