DCSP 和 DCOP 求解研究进展*)

贺利坚[1] 张 伟[1] 石纯一[2]

(烟台大学计算机学院智能信息处理实验室 烟台 264005)[1] (清华大学计算机科学与技术系 北京 100084)[2]

摘 要 分布式约束满足问题(DCSP)和分布式约束最优问题(DCOP)的研究是分布式人工智能领域的基础性工作。本文首先介绍了 DCSP 和 DCOP 的形式化描述及对实际应用问题的建模方法。在 DCSP 和 DCOP 的求解中,通常对问题要进行限制和要求,同时要满足分布性、异步性、局部性、完备性的原则。异步回溯(ABT)、异步弱承诺搜索(AWC)和分布式逃逸(DB)算法是求解 DCSP 的有代表性的算法;DCSP 算法对 DCOP 求解产生了影响,但由 DCSP 一般化到 DCOP 的算法,仅适用于解决部分特定的问题,DCOP 的最优、异步算法有异步分布式约束最优算法(Adopt)和最优异步部分交叉算法(OptAPO)。本文讨论了上述算法的性能。相关的研究工作在多局部变量的处理、超约束 DCSP、算法性能度量、通信的保密等方面进行了扩充,在对问题本身的研究、建模方法学、算法、与其他方法的结合以及拓展应用领域等方面仍有许多问题需要进一步研究。

关键词 分布式约束满足问题,分布式约束最优问题,多 Agent 系统

A Research on Solving DCSP and DCOP

HE Li-Jian[1] ZHANG Wei[1] SHI Chun-Yi[2]

(Intelligent Information Processing Laboratory in School of Compute, Yantai University, Yantai 264005)

(Department of Compater S'cience and Techllology, Tsinghau University, Beijing 100871)

Abstract The research of Distributed Constraint Satisfaction Problem(DCSP) and Distributed Constraint Optimization Problem (DCOP) have already became one of the most important basic research topic in distributed artificial intelligence domain. Formal description of DCSP and DCOP, as well as the modeling method to practical problem are introduced at the beginning. In order to solve DCSP and DCOP, there are some limitation and restriction in problem itself, and problem solving process needs to satisfy the principle of distributing, asynchronism, locality and completeness. Asynchronous Backtracking (ABT), Asynchronous Weak-Commitment search(AWC) and Distributed Breakout(DB) are representative algorithms for solving DCSP, which have some affect to solving DCOP. Due to the DCOP algorithm generalized by DCSP can only used to some special problem, there need asynchronous distributed constraint optimization (Adopt) and Optimal Asynchronous Partial Overlay(OptAPO) to provide optimal and asynchronous solution of DCOP. The performances of the above algorithms are discussed also. The relative work expanded in multi-local variability, over constraint DCSP, matrix of algorithms performance, privacy in communication, and etc. As the future work, progress should be made in DCSP and DCOP themselves, modeling methodology, algorithm, combination with other method and extending the application area.

Keywords DCSP, DCOP, Multi-agent system

1 引言

人工智能领域中的许多组合问题可以用约束满足问题(CSP,Constraint Satisfaction Problem)进行建模。分布式人工智能中的应用对 CSP 提出了新的要求,同时利用多 Agent 合作对 CSP 进行求解也成为一种新的思路。在分布式环境下的 CSP 称为分布式约束满足问题(DCSP, Distributed Constraint Satisfaction Problem)。分布式约束最优问题(DCOP, Distributed Constraint Optimization Problem)在 DCSP 基础上,进一步提出了最优的要求。DCSP 和 DCOP 适合解决多 Agent 合作的问题以及在分布式环境下的任务分配和资源分配问题。因此,DCSP 和 DCOP 的研究成为分布式人工智能领域的基础性工作,成为许多合作求解问题的重要的和有用的抽象[1]。从上个世纪 90 年代以来,Makoto Yokoo 和 Milind Tambe 等学者在 DCSP 和 DCOP 的求解和应用方面做了大量工作[2~6]。针对求解算法、算法性能度量[7]、处理多局部变量及超约束满足问题[4]、信息保密[8]等方面的研究也方兴未艾。不过,对 DCSP 和 DCOP 求解的研究尚有待进一步深入,在对问题本身、建模方法、求解算法及与其他方法的结合等方面仍面临许多挑战。

本文对近年来 DCSP 和 DCOP 求解研究的进展情况进行了综述。第 2 节是关于 DCSP 和 DCOP 的形式化描述和对典型问题建模方法的介绍;第 3 节概述了 DCSP 和 DCOP 求解算法;第 4 节讨论了在求解中还需要考虑的几个问题;最后提出进一步的研究方向。

2 问题的定义及应用建模

CSP 的目的在于求解多个变量的一致性赋值。有 n 个变量 $x_1, x_2, \cdots, x_n, D_1, D_2, \cdots, D_n$ 是各个变量的取值域。变量间的约束由谓词 $p_k(x_{k_1}, \cdots, x_{k_j})$: $D_{k_1} \times \cdots \times D_{k_j} \rightarrow \{$true,

*)本文得到国家自然科学基金项目(编号:60496323)的资助。贺利坚 讲师,硕士,主要研究方向:分布式人工智能;张 伟 教授,博士,主要研究方向为分布式人工智能;石纯一 教授,博士生导师,主要研究方向:人工智能应用基础。

false}定义。求解 CSP 就是要发现一组或全部对所有变量的赋值，使所有的约束都得到满足。

在分布式环境中，假设 CSP 中的变量分布在多个 Agent 中，求解包含在多 Agent 系统中的 CSP 可视为在多个 Agent 中获得对多个变量的一致性赋值，这样的问题称为 DCSP。

形式上，有 Agent 集合 $A=\{A_1,A_2,\cdots,A_n\}$ 和变量集合 $V=\{x_1,x_2,\cdots,x_m\}$，域集合 $D=\{D_1,D_2,\cdots,D_m\}$，D_i 是变量 x_i 可能取值的有限集。每个 Agent 拥有一个或者多个变量，每个变量只属于一个 Agent。变量间的约束仍由谓词 $p_k(x_{k_1},\cdots,x_{k_j}):D_{k_1}\times\cdots\times D_{k_j}\rightarrow\{\text{true, false}\}$ 表示，约束分布在 Agent 内部或多个 Agent 之间。当分布在各个 Agent 中的所有变量的赋值满足所有约束时，该赋值即为 DCSP 的一个解。

在 DCOP 中，将变量间是否存在约束扩展到多变量间的定量关系上。Agent 集合、变量集、域集合及变量和 Agent 之间的所属关系的定义同 DCSP。变量间的约束由成本(Cost)函数 $f_i:D_{i_1}\times\cdots\times D_{i_j}\rightarrow N$ 定义，N 是自然数集。约束也分布在 Agent 内部或多个 Agent 之间。DCOP 的求解目标是获得使总成本最小的对所有变量的赋值。即对由目标函数 F 定义每一种可能赋值对应的成本函数之和：

$$F(Assign)=\sum f_i(x_{i_1},\cdots,x_{i_j}),x_{i_k}\in D_{i_k}$$

求出使总成本最小的赋值 $Assign^*$：

$$Assign^*=\arg\min F(Assign)$$

DCSP 和 DCOP 适用于多 Agent 合作求解问题，包括协调、冲突解决、维护合作 Agent 的共同信念等。目前成功的应用除传感器网络(Sensor Network)[9,10]外，还有灾难救援仿真(Disaster rescue simulation)[3]、会议日程安排(Meeting schedule)[8]、分布式通信网络(Distributed communication network)[4]、信任维护系统[11]等。

例如，传感器网络由如图 1(a)所示的传感器阵列构成。跟踪一个移动目标(Target)需要三个传感器，一个传感器仅可跟踪一个移动目标，并且仅允许相邻的传感器进行通信，每个移动目标仅在传感器附近才可被感知。明显地，在图中的 4 个移动目标中，仅有两个能被跟踪。在多个移动目标的运动过程中，由传感器阵列完成对它们的跟踪。

Ramón Béjar[9]用一个 Agent 代表一个移动目标，每个 Agent 有三个不同的变量，每个变量的值为可以分配给该移动目标的传感器。变量的取值范围是可供分配的传感器的集合。一个 Agent 内 3 个变量间的约束是 3 个传感器彼此不同且两两相邻，不同 Agent 中的变量间的约束是任一传感器至多能被一个 Agent 选择。这样，将传感器网络问题映射成用 DCSP 建模的资源分配问题。

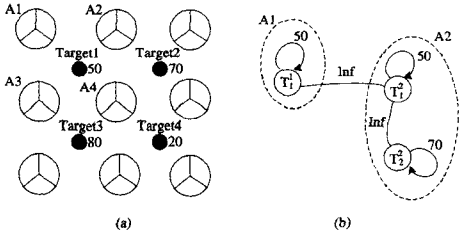

图 1　传感器网络及用 DCOP 建模

考虑各个移动目标的重要性，P. J. Modi[6]等人将传感器网络问题作为 DCOP 进行求解。必须通过有效的协调，使跟踪到的移动目标的权重之和最高。对每个目标 j 和每个可能跟踪目标的 Agent i，创建变量 T_j^i。变量 T_j^i 的取值域为能跟踪目标 j 的 3 个 Agent 的组合，如 T_1^i 的域是 $\{A_1A_2A_3, A_1A_2A_4, A_1A_3A_4, A_2A_3A_4, Ignore\}$，$Ignore$ 表示不给目标 j 分配 Agent。Agent i 和 k 的两个变量 T_j^i 和 T_j^k 的等值约束 $f(T_j^i, T_j^k)$ 为 i 和 k 同意跟踪同一个目标；对 Agent i，变量 T_j^i 和 T_k^i 的互斥约束 $f(T_j^i, T_k^i)$ 表示 i 不能参与跟踪两个目标。违反上述两种约束均对应一个很大的值 Inf。$f(T_j^i)$ 表示 Agent i 忽略目标 j 要付出的代价，如 $f(T_2^2)=70$。图 1(b)给出了 A_1 和 A_2 中的变量的取值。

Paul Scerri 等[10]用 DCOP 形式化了任务分配问题。T_a 为所有可能任务的集合，$|T_a|=K$，在任一时刻只能处理 N 个任务，$N<K$。实际正在被处理的任务集合为 T_p($|T_p|=N$)。对 T_a 中的每个任务，DCOP 变量的取值范围为 $\{A, N, I\}$，分别代表已分配(*Allocated*)、未处理(*NotPresent*)和忽略(*Ignored*)。如果一个任务被分配了 I 值，由于忽略任务而付出的代价由成本函数 $w:T_a\rightarrow N\cup\infty$ 量化，当资源有限时，DCOP 需要 Agent 为变量赋值，让资源分配给那些最重要的任务，而忽略成本小的任务。

由于传感器网络能够充当测试分布式任务分配和资源分配问题求解的基准问题，文[10]还提供了传感器网络的实例生成器 SensorDCSP，从而可以按照不同的指标生成测试实例。

3　DCSP 和 DCOP 求解算法

3.1　算法的限制及要求

由于问题本身的特点，DCSP 和 DCOP 中的 Agent 需要在共同的协作机制下进行决策，而不是像在通常的 MAS 中，Agent 可以自治地决定是否合作。除此之外，现有的 DCSP 和 DCOP 求解算法对问题本身还做了一些限制和要求。

一般要求每个 Agent 只包含一个变量(在下文的叙述中，变量和 Agent 将视为同义词)，Agent 知道所有和自己的变量相关的约束；约束是一元或二元的；通过 Agent 之间的通信完成求解，Agent 知道通信对方的网络标识，以保证通信能够正常进行；Agent 传递消息的延时是随机但有限的。这些限制不会对适用的问题造成大的影响，可以通过技术手段使求解环境满足要求[6]，例如多元约束可以转换为多个二元约束进行处理。

针对分布式计算中实际问题，对求解算法的一般要求有：(1)分布性：仅使用局部通信完成求解，不设固定的中心 Agent 对求解过程进行集中控制；(2)异步性：Agent 异步执行和通信，不存在空闲等待其他 Agent 的时间；(3)局部性：每个 Agent 仅知道整个问题的部分信息；(4)完备性：对存在最优解的 DCOP，要确保能找到最优解，保证求解质量。

3.2　DCSP 求解算法

DCSP 求解的最原始算法是过滤算法(Filtering Algorithm)[2]。每个 Agent 向其邻居发送其值域，在得到反馈后，不断地从域中除去不能满足约束的值，最后形成问题的解。在基于超消解的相容算法(Hyper-Resolution-Based Consistency Algorithm)[4]中，提出将约束表示为变量值禁止的组合，记为 *nogood*。如 *nogood* $\{x_1=a, x_2=b\}$ 代表 x_1 和 x_2 分别取值 a 和 b 时不能满足约束。每个 Agent 交换已求得的 *nogood*，并使用超消解律得到更“大”的 *nogood*，直到求得问题的解。任何 *nogood* 的超集将不可能为问题的解，所以当某个 *nogood* 是空集时，问题无解。异步回溯(ABT, Asyn-

chronous Backtracking)[2]、异步弱承诺搜索(AWC, Asynchronous Weak-Commitment search)[4]和分布式逃逸(DB, Distributed Breakout)[4]算法都是求解 DCSP 的有代表性的算法。

ABT 算法是回溯算法的异步版本。每个变量都具有预先定义的优先级,优先级一般由变量名的字母序确定。在求解过程中,高优先级变量将其当前赋值通过 *ok*? 消息发送给低优先级变量,低优先级变量检查其赋值与 *agent_view* 结构(用来保存其他变量的赋值)中的值是否冲突。如果变量的当前赋值与高优先级的变量的赋值不一致,则改变当前赋值。如果在变量的取值域中不存在与高优先级变量一致的值,则生成一个 *nogood*,并通知高优先级变量,要求高优先级变量改变赋值。ABT 无需考虑所有约束,仅在必要时生成当前情境下的 *nogood*。每个变量在异步、并发的执行过程中,仅从自己的视角维护对其他变量的当前赋值,可能会包含过时的信息。

AWC 算法克服了 ABT 中变量优先级静态不变而导致的问题——如果高优先级变量赋值不当,需要低优先级变量的大量搜索后才能予以纠正。AWC 引入了一种最小冲突启发来降低不当赋值的出现,并且通过动态改变变量的优先级,无须多余搜索就能纠正不当赋值。最小冲突启发是指选择值域中那些与其他 Agent 的赋值产生最小冲突的值作为自己的赋值。Agent 间传送的消息除 *nogood* 外,还需要加上变量的优先级。如果 Agent 当前赋值不满足与高优先级变量的约束,将用最小冲突启发改变赋值。如果 Agent 不能找到与高优先级变量一致的赋值,该 Agent 的优先级将变为相邻 Agent 中最大优先级再加 1,并继续等待消息。

DB 算法针对求解过程可能进入局部最小(local-minima)状态而采取了另外的求解思路。局部最小是指一些约束没有被满足而出现了冲突,但这些冲突的数目不能通过单独改变任何一个变量的值来减少。DB 算法是一种跳出局部最小状态的方法,算法为每个约束定义权重,所有冲突约束的权重之和作为求解的评价值。邻居 Agent 之间交换赋值,仅有能最大程度地改进评价值的 Agent 才获得修改赋值的权利。如果两个 Agent 不相邻,则可以同时改变赋值。DB 算法未检测整个 Agent 是否限入局部最小,而是考察准局部最小(quasi-local-minima),即某一 Agent 的赋值使部分约束产生冲突时,该 Agent 和所有邻居的可能提高值均为 0。当陷入准局部最小时,算法增加当前状态中产生冲突的约束的权值,从而跳出准局部最小状态,开始新的搜索。

ABT 和 AWC 算法具有完备性,DB 算法的完备性不能得到保证[4]。可以用求解问题需要的计算周期数量(cycle)考察算法的性能,一个计算周期包括读取所有消息、执行局部计算和发送消息。AWC 算法要明显优于 ABT,因为前者不需要穷尽查找就能修正不当的赋值。设 Agent 数目分别为 $n=90$, 120 和 150 时对 AWC 和 DB 算法进行比较,约束较稀疏时(约束数 $m=n\times 2$),AWC 优于 DB 算法。而当问题在临界跳变点时($m=n\times 2.7$),DB 算法要优于 AWC。

3.3 DCOP 求解算法

DCSP 算法对 DCOP 求解产生了影响,但由 DCSP 一般化到 DCOP 的算法(如异步回溯算法 ABT),仅适用于解决部分特定的问题。DCOP 需要有超出 DCSP 解算法的技术,向全局最优进行扩展。在分布式环境中,异步搜索算法(Synchronous Search)模拟集中的搜索算法,所有成本在赋值完成前需要在单个的 Agent 中累加。Greedy Repair 算法由一个 Agent 收集全局成本信息处理分布变量,但这些算法均不能同时满足分布性、异步性和完备性[6]。

异步分布式约束最优算法 Adopt (Asynchronous distributed constraint optimization)[6]是第一个同时满足分布性、异步性和完备性的 DCOP 求解的算法。为了执行分布式回溯搜索,Adopt 算法在深度优先的搜索树中为 Agent 划分优先等级,形成上下级关系,并用 *VALUE*、*COST* 和 *THRESHOLD* 三种消息(如图 2 所示)进行通信。*VALUE* 消息包含上级 Agent 的当前取值,向下级 Agent 传送,下级 Agent 也仅接受其直接上级 Agent 传送的 *VALUE* 消息。下级 Agent 在接到 *VALUE* 消息后,根据本地成本及其下级通过 *COST* 消息反馈的成本,计算出对全局最优解成本下界的估计,并用 *COST* 消息向上级 Agent 反馈。COST 消息中包含了表示上级和本地赋值的当前上下文 *CurrentContext* (类似 ABT 中的 *agent_view*)以及对当前最优解下界的最高估计 *ub* 和最低估计 *lb*。Adopt 算法提供了内建的终止检测机制,当 *ub* 和 *lb* 相等时,保证每个 Agent 在获得最优解后独立地自动停止执行。上级 Agent 根据收到 *COST* 消息后再次进行计算,如果需要试探新值,将再次向其下级发送 *VALUE* 消息。Agent 采用最好优先(Best-First)策略,在计算当前取值时总是选择最好的值,随着交互的进行,下界越来越精确,直到 *ub* 和 *lb* 相等。在求解过程中,由于上级 Agent 值的改变,Agent 可能会放弃当前的部分解,而这个部分解有可能在后续的求解中会被重新用到,Adopt 算法能够通过 *THRESHOLD* 消息传送回溯阈值,从而有效地重新构造以前考虑过的部分解,使在最坏情况下的空间复杂度限制在多项式级。另外,有界误差(Bounded-error)近似技术使在不需要最优解的时候,能够得到解质量与计算时间之间的有效折衷。

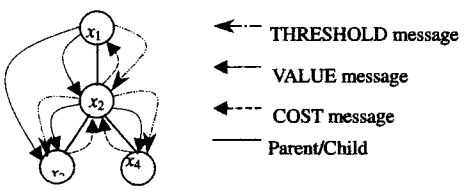

图 2 Adopt 算法中三种消息及其传递

最优异步部分交叉算法 OptAPO (Optimal Asynchronous Partial Overlay)[5]利用了称为调停者(mediator)的 Agent。调停者在算法执行中动态产生,可以将部分变量及其约束集中起来。OptAPO 算法在调停者的干预下直接地通信约束,以局部地集中问题。OptAPO 允许 Agent 扩展和交叉它们在求解过程中用于本地决策的上下文。当 Agent 充当一个调停者时,它计算整个问题中的部分解,在调停过程中向其他 Agent 推荐值的改变。调停者使用集中式的基于分枝限界算法的最优化例程去发现局部问题的最优解。一个调停者 Agent 从其他 Agent 接收到约束信息后,会将该 Agent 加入到称为 goodlist 的数据结构中,使用 Agent 的 goodlist 的大小去度量问题的集中化程度。OptAPO 使用了一种新颖的成本调整技术驱动对约束的通信,使算法提供了快速、分布、异步的问题求解的同时,无需大量的通信。OptAPO 能够同时发挥集中技术的速度优势和分布式求解的能力优势。

Adopt 算法和 OptAPO 算法是目前 DCOP 求解中最有代表性的两种方法,文[11]利用称为 Cycle-Based Runtime

(CBR)的度量方法,综合考虑计算周期间内的计算时间和周期间等待时间,对 Adopt 算法和 OptAPO 算法的性能做了较全面的比较。实验表明,实施部分集中的 OptAPO 需要的周期数要少于 Adopt,但 OptAPO 在每个周期中花费更多的计算时间,OptAPO 更适宜于通信能力差而计算能力强的环境。在通信能力相对计算能力更强的环境中,由于能更平均地分配 DCOP 的求解,Adopt 比 OptAPO 更出色。

对 DCOP 求解算法的改进主要集中在通过预处理加速算法[15]、解决不可靠通信下的适应性[16]、减少消息量[17]等方面。

4 问题的扩充

4.1 多局部变量的处理

DCSP 和 DCOP 中大多数的应用对应多局部变量,即一个 Agent 中有多个变量的情形。而在前面的算法中,一般假设一个 Agent 只有一个变量。处理多局部变量的基本思路有两种[4]:(1)每个 Agent 首先求出本地的所有可能解,并将给定的问题重新形式化为一个 Agent 一个变量的问题,新问题中变量的取值域是所有本地解的集合。当本地问题很复杂时,由于需要求出很多可能无用的本地解,而使算法整体效率低下。Agent_ordering AWC 方法[13]可以仅在必要时搜索本地解,而不是提前找到所有本地解。(2)为每个 Agent 创建多个虚拟 Agent,每个虚拟 Agent 对应一个变量。这样,一个 Agent 需要执行多个虚拟 Agent 的并发求解行为。这种方法的缺点是,不区分 Agent 的内部通信和更为昂贵的 Agent 之间的通信,从而造成通信资源的浪费,也影响了求解效率。Variable_ordering AWC[14]方法要求 Agent 只有在求出了一个相容的本地解时,才向其他 Agent 发送消息,充分利用 Agent 内的通信,减少 Agent 之间的交互。

4.2 超约束 DCSP

超约束 DCSP(over-constraint DCSP)[4]也称为分布式部分约束满足问题(Distributed Partial CSP),是指约束太强,以致于不存在满足所有约束的解的 DCSP。在实际应用中,需要放松原有过强的约束,同时希望放松后的问题尽可能接近原问题。全局距离函数(global distance function)用于度量放松约束的程度。从每个 Agent 的角度,设 P_i 为未放松约束的 DCSP,d_i 为移去约束的问题与原问题的距离,全局距离 $D=\max_{i\in A} d_i$。

采用 Distributed maximal CSPs[18]标准求解时,要求每个 Agent 试图搜索能够最小化冲突约束的最大数目的解。设 PS_i 是通过从 P_i 中删除约束获得的所有可能的 DCSP,P_i 和 PS_i 的距离 d_i 为删除的约束的个数。每一个 Agent 尽可能满足更多的自身约束,并使多个 Agent 之间获得合理的折衷,使违反约束的总数最小。

采用 Distributed hierarchical CSPs[19]标准时,每个约束都有一个权重,要求搜索能够最小化冲突约束的最大约束权值之和的解。PS_i 由$\{p_i^0, p_i^1, p_i^2\cdots\}$组成,每个 p_i^q 是从 P_i 中移去权值小于或等于 α 的所有约束而获得的 DCSP,P_i 和 p_i^q 的距离 d_i 定义为 α。在异步回溯算法求解时,如果发现问题是超约束的,需要放弃重要性低于阈值的约束。阈值可以由产生空 nogood 的约束的权重计算而来。接下来再用相同的方法进行求解,必要时再次放松某些低权值的约束,直到求出某个放松后的 DCSP 的解。

4.3 算法性能度量

算法性能的度量不仅直接用于评价算法性能,而且对算法的研究有指导作用。DCSP 和 DCOP 算法的性能主要由求解所需的运行周期数和每一周期所需的约束检查的时间进行度量。在算法执行的每一周期中,除了进行约束检查外,还需要进行必要的计算。对限制计算能力的 Agent 而言,本地计算的负担是不能忽略的。另外,消息的大小及数量也对算法的效率造成较大的影响。

文[12]中提出的 CBR 度量模型综合考虑了在一个周期间内的计算时间和周期间的通信等待时间。文[7]提供了完整的度量运行时间的模型。在算法执行的第 k 个周期中执行时间为

$$R(k)=\max_{i\in A_g}(L_i^k+C_i^k)$$

其中,L_i^k 和 C_i^k 分别为第 k 个周期中需要的本地计算时间和用于传输消息的通信时间。计算 L_i^k 和 C_i^k 时需要考虑接受到的消息数目、每个消息的大小、处理消息的计算时间、需要进行检查的约束数目、约束检查需要有计算时间、发出消息的数目、发出消息的大小、处理不同大小发出消息的计算时间和传输不同大小的消息的通信时间等因素。需要 K 个周期完成的算法执行的总时间为

$$Totalrn\text{-}time=\sum_{k=1}^{K}(R(k))$$

该模型细化到了各个消息的大小和数量,既考虑了消息处理的时间,又考虑了通信的时间。模型能够包容以往的各种性能度量方法,同时又有一定的灵活性,当忽略某些参数时,可以针对应用的特点和关键的性能指标,找出进一步改进算法的启示。

4.4 通信的保密

DCSP 和 DCOP 的求解过程中需要 Agent 之不断地交换有关信息,但在电子商务等应用中,有些私有资源不必由其他成员共享,或者有些信息涉及隐私与安全而不能在 Agent 间共享。文[8]结合信息安全技术研究了 DCSP 的求解。假定每个变量属于一个 Agent,所有变量的域相同(变量域是公共知识)。问题中的信息的保密性表现为:属于 Agent i 的变量的单目约束及取值仅由 i 知道,Agent i 和 j 的变量 x_i 和 x_j 之间的约束,由 i 和 j 掌握,任何 Agent 不知道其他 Agent 的赋值。算法采用公钥加密模式对约束进行编码,针对密文,由计算服务器(即搜索控制器)和译码者(即值选择器)分布地完成求解。在求解过程中,计算服务器也不能知道任何变量的赋值。算法的完备性是有保证的,但在计算效率和降低通信成本方面仍有待进一步改进。

5 进一步研究的方向

对问题本身的研究。约束满足与约束最优问题属于 NP 难解问题。在分布式环境下,涉及多个求解实体,各 Agent 的计算能力与通信能力等方面存在诸多限制。需要进一步研究具体 DCSP、DCOP 问题的难度,包括内、外部约束对问题难度的影响。除多局部变量、部分约束满足、信息保密等方面的问题外,多目标优化(Multi-criteria optimization)也应予以考虑。多目标优化将突破针对单一的全局目标进行优化的限制,有更广阔的应用空间。

建模方法学。利用 DCSP 和 DCOP 对实际问题建模的工作仍处于经验阶段,缺乏统一的方法学指导。需要对适用的问题给出更高抽象层次的描述,在归纳总结现有建模方法的基础上,通过形式化方法,达到自动建模的目的。

算法的进一步的研究。需要对现有 DCSP 和 DCOP 算法

进行合理分类，考虑对求解环境的要求，明确各种算法适合的问题类。求解中多 Agent 之间的协调主要通过规定的通信机制进行，多个 Agent 之间的合作形式呆板、机械，引入 Agent 的社会性特征以及更灵活、有效的求解组织结构也能为 DCSP 和 DCOP 的求解带来益处。现有算法对求解环境的动态性和实时要求方面的研究仍不够深入，在多局部变量、信息保密等方面专门针对 DCOP 的工作还未见到。

与其他方法的结合。在大规模的复杂领域中，需要组合各种技术，彼此扬长避短。可以和 DCSP 与 DCOP 结合的、用于构建多 Agent 系统的方法包括 BDI 方法、对策论和拍卖方法、DisPOMDP 方法等，其中，DCOP 发挥利用局部交互获得局部或全局最优的优势。在这方面 M. Tambe 等人的工作[3]已经做了很好的示范，也还有待深入。

应用领域的拓展。传统 CSP 的应用已经拓展到了语义 Web 等领域[20,21]。在 Internet 背景下，DCSP 和 DCOP 应用于知识工程领域也将是一件很自然的选择，这也对 DCSP 和 DCOP 的求解提出更高的要求。

参 考 文 献

1 Wooldridge M, Dunne P E. On the computational complexity of coalitional resource games. Artificial Intelligence, 2006, 170: 835～871

2 Yokoo M, Ishida T. Search Algorithms for Agents. In: Weiss G ed. Multiagent Systems, Springer, 1999

3 Tambe M, et al. Conflicts in Teamwork: Hybrids to the Rescue. In: Fourth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'05), Utrecht, Netherlands, 2005. 3～10

4 Yokoo M, Hirayama K. Algorithms for distributed constraint satisfaction: A review. Autonomous Agents and Multi-Agent Systems, 2000, 3(2): 185～207

5 Mailler R, Lesser V. Solving Distributed Constraint Optimization Problems Using Cooperative Mediation. In: Third International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'04), New York, USA, 2004. 438～445

6 Modi P, Shen W, Tambe M, et al. Adopt: Asynchronous distributed constraint optimization with quality guarantees. Artificial Intelligence Journal, 2005, 161: 149～180

7 Jung H, Tambe M. On Communication in Solving Distributed Constraint Satisfaction Problems. In: Proceedings of the 4th International Central and Eastern European Conference on Multi-Agent Systems (CEEMAS), 2005

8 Yokoo M, Suzuki K, Hirayama K. Secure distributed constraint satisfaction: reaching agreement without revealing: [private information]. Artificial Intelligent, 2005, 161: 229～245

9 Béjar R, Domshlak C, Fernández C, et al. Sensor networks and distributed CSP: communication, computation and complexity. Artificial Intelligence, 2005, 161: 117～147

10 Scerri P, Modi P J, Shen Wei-Min, et al. Applying Constraint Reasoning to Real-world Distributed Task Allocation. In: Proceedings of Autonomous Agents and Multi-Agent Systems Workshop on Distributed Constraint Reasoning, 2002

11 Zhang W, Wang G, Xing Z, et al. Distributed stochastic search and distributed breakout: Properties, comparison and applications to constraint optimization problems in sensor networks. Artificial Intelligence, 2005, 161: 55～87

12 Davin J, Modi P J. Impact of problem centralization in distributed constraint optimization algorithms. In: Fourth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'05), Utrecht, Netherlands, 2005. 1057～1063

13 Armstrong A, Durfee E. Dynamic Prioritization of Complex Agents in Distributed Constraint Satisfaction Problems. In: Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI 1997), 1997. 620～625

14 Yokoo M, Hirayama K. Distributed constraint satisfaction algorithm for complex local problems. In: Proceedings of the Third International Conference on Multi-Agent Systems (ICMAS-98), Paris, France, 1998. 372～379

15 Ali S, Koenig S, Tambe M. Preprocessing Techniques for Accelerating the DCOP Algorithm ADOPT. In: Fourth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'05), Utrecht, Netherlands, 2005

16 Modi P J, Ali S, Shen Wei-Min, et al. Distributed constraint reasoning under unreliable communication. In: Proceedings of Distributed Constraint Reasoning Workshop at International Joint Conference on Autonomous Agents and Multiagent Systems, 2003

17 Petcu A, Faltings B. A scalable method for multiagent constraint optimization. In: Proceedings of the 19th International Joint Conference on Artificial Intelligence, IJCAI-05, Edinburgh, Scotland, Aug. 2005

18 Hirayama K, Yokoo M. Distributed partial constraint satisfaction problem. In: Smolka G, ed. Proc. of the 3rd Int'l Conf on Principles and Practice of Constraint Programming (CP'97). Berlin: Springer-Verlag, 1997. 222～236

19 Yokoo M. Constraint relaxation in distributed constraint satisfaction problem. In: IEEE, ed. Proc. of the 5th Int'l Conf on Tools with Artificial Intelligence. Los Alamitos: IEEE Computer Society Press, 1993. 56～63

20 Preece A, Chalmers S, McKenzie C, et al. Handling Soft Constraints in the Semantic Web Architecture. WWW2006, Edinburgh, UK, 2006. 22～26

21 van Otterloo S, van der Hoek W, Wooldridge M. Knowledge Condition Games. In: Proceedings of the Sixth Workshop on Game Theoretic and Decision Theoretic Agents (GTDT-04), New York, NY, 2004

(上接第 131 页)

构。概念格理论和粗糙集理论之间的关系已经引起了许多研究者的关注。本文讨论了一种将形式背景转化为集值信息系统的方法，证明了形式背景中的对象粒协调集与由该形式背景导出的集值信息系统的协调集是等价的，说明了粗糙集与概念格之间存在着一定的内在联系；并且给出了形式背景中三种不同类型的对象粒属性特征及每一种对象粒属性的判定定理。

参 考 文 献

1 Pawlak Z. Rough Sets: Theoretical Aspects of Reasoning About Data. Boston: Kluwer Academic Publishers, 1991

2 Ganter B, Wille R. Formal Concept Analysis. Mathematical Foundations[M]. Berlin, Germany: Springer-Verlag, 1999

3 Kent R E. Rough concept analysis: a synthesis of rough sets and formal concept analysis. Fundamenta Informaticae[J], 1996, 27: 169～181

4 Yao Y Y, Chen Y H. Rough set approximations in formal concept analysis. In: Dick S, Kurgan L, Pedrycz W, et al. eds. Proceedings of 2004 Annual Meeting of the North American Fuzzy Information Proceeding Society (NAFIPS 2004)[C], June 2004. 73～78

5 Hu K, Lu S Y, et al. Concept approximation in concept lattice. In: Knowledge Discovery and Data Mining. Proceedings of the 5th Pacific-Asia Conference, LNCS 2035, 2001. 167～173

6 王志海，胡可云，刘宗田，等. 概念格上的粗糙集合运算与函数依赖生成. 清华大学学报(自然科学版)，1998, 38(S2): 1～4

7 Yao Y Y. Concept lattices in rough set theory. In: Dick S, Kurgan L, Pedrycz W, et al. eds. Proceedings of 2004 Annual Meeting of the North American Fuzzy Information Proceeding Society (NAFIPS 2004)[C], June 2004. 796～801

8 Wu Wei-Zhi. Knowledge Reduction in Formal Contexts. In: Proceedings of 2006 Xi'an Symposium on Rough Set Theory, 2006

9 张文修，梁怡，吴伟志. 信息系统与知识发现. 北京：科学出版社，2003