

构建一种多 agent 并行计算模型

郝水侠¹, 李凡长²

(1. 徐州师范大学 数学系, 江苏 徐州 221116;

2. 苏州大学 计算机科学与技术学院, 江苏 苏州 215006)

摘要:文中旨在找出一种新的构建多 agent 并行计算模型的方法。通过对组合数学中二分图理论的研究,发现可以利用这个数学思想建立一种新的多 agent 并行计算模型。文中在对多 agent 计算本质分析的基础上,基于二分图理论,建立了一种新多 agent 并行模型,给出了基于 BDI 模型的多 agent 并行计算模型的流程算法,最后还给出了这种模型的优越性是能最大可能地减少 agent 之间的冲突。通过上述工作,可为多 agent 并行求解问题研究提供理论依据。一方面弥补了现有理论的不足,另一方面也为人们进一步研究多 agent 的并行计算提供了一种解决方案。

关键词:多 agent 系统;并行计算模型;二分图

中图分类号:TP18

文献标识码:A

文章编号:1673-629X(2006)05-0071-03

Designing a Kind of Multi-agent Parallel Computing Model

HAO Shui-xia¹, LI Fan-zhang²

(1. Department of Mathematics, Xuzhou Normal University, Xuzhou 221116, China;

2. College of Computer Science and Technology, Soochow University, Suzhou 215006, China)

Abstract: Aims at searching for a new method of building multi-agent parallel computing model. By analyzing binary graph theory of combination math, find can use the math method to build a new multi-agent parallel computing model. So based on analyzing the multi-agent computing and studying binary graph theory, the paper built a novel multi-agent parallel computing model and gave an algorithm of BDI, pointed at the advantage of the model which is mostly reducing collision of multi-agent. Through the work above, propose the basic theory of solving parallel problem for multi-agent parallel computing model. One aspect makes up for the absence of theory. Another aspect provides a method of solving problem.

Key words: multi-agent system; parallel computing model; binary graph

目前关于构建多 agent 并行理论基础的文献主要有:文献[1]在基于 Internet 的环境下,把 agent 技术引入到并行计算来,提出了基于 Internet 的并行计算模型,提供了一种使用大量的分散的计算资源来进行大规模的、复杂的计算的方法。文献[2]提出了一种基于多 agent 的分布式计算模型,研究了模型中的各 agent 的功能及它们之间的协调关系,但该模型缺乏通用性。文献[3]用移动代理实现并行计算,由于移动代理的自治性、移动性、可扩展性可进一步提高计算性能。文献[1]、[3]都是充分利用空闲资源,但对系统的如何运作及 agent 之间的关系并没有涉及到,只是从 agent 的概念和角度出发。针对这些问题,文中借助二分图的思想,将多个 agent 组织起来,真正达到多个 agent 并行。另外以 BDI 为基础建立 B, D, I 之间的

关系,构建了多 agent 并行计算模型并讨论了这种模型的性质。

1 二分图的基本概念

1.1 二分图

如果图中的结点集合 $V = X \cup Y$, $X \cap Y = \emptyset$, 且图中的每条边均为 $[x, y]$, 其中 $x \in X, y \in Y$, 则该图称为二分图^[3], 记作 $\langle X, \Delta, Y \rangle$, 其中, Δ 表示边集合。令 $X = \{1, 2, \dots, n\}$, 在二分图 $\langle X, \Delta, Y \rangle$ 中定义 $A_i = \{y \mid y \in Y, [i, y] \in \Delta\} (1 \leq i \leq n)$, 则可将二分图与集族联系起来。

1.2 二分图匹配

在二分图 $\langle X, \Delta, Y \rangle$ 的某个边集合 $\{[x_1, y_1], [x_2, y_2], \dots, [x_i, y_i]\}$ 中, 如果都是两两互不相同的则称它是该二分图的一个匹配。

1.3 覆盖

在二分图 $\langle X, \Delta, Y \rangle$ 中, 如果结点集合 $S \subseteq X \cup Y$, 使得 Δ 中的每条边至少有一个结点在其中, 则称 S 是 Δ 的一个覆盖。

收稿日期:2005-08-21

基金项目:江苏省自然科学基金(BK2002040);江苏省教育厅自然科学基金(02KJB520001)

作者简介:郝水侠(1973-),女,陕西大荔人,讲师,硕士,主要研究方向为人工智能、多 Agent 系统等。

1.4 相关事实和定理

令 $X = \{1, 2, \dots, n\}$, 二分图 $\langle X, \Delta, Y \rangle$ 关联的集族为 $\{A_1, A_2, \dots, A_n\}$, 则有以下事实:

(1) 若 $\{[i_1, y_{i_1}], [i_2, y_{i_2}], \dots, [i_k, y_{i_k}]\}$ 是二分图的一个匹配, 则 $y_{i_1}, y_{i_2}, \dots, y_{i_k}$ 是集族 $A_{i_1}, A_{i_2}, \dots, A_{i_k}$ 的一个相异代表系;

(2) $\{A_1, A_2, \dots, A_n\}$ 有相异代表系, 当且仅当二分图有一个 n 边的匹配(称为完全匹配);

(3) $\{A_1, A_2, \dots, A_n\}$ 有相异代表系的最大子集数等于二分图的最大匹配边数。

定理: 设 $\langle X, \Delta, Y \rangle$ 是一个二分图, 它的匹配的最大边数等于覆盖 Δ 的最小结点数。

证明略。

2 多 agent 并行计算模型

2.1 并行计算模型公约

多 agent 并行计算模型^[4]: agent 的模型可用一个五元组表示, 即 $Ag_m = \{B, D, I, P, A\}$, 其中 B 表示 agent 的信念, 是 agent 解决问题的依据; D 表示 agent 的愿望, 是解决问题的动力; I 表示 agent 的意图, 按照某种规划设定解决问题的行为或行动; P 表示问题; A 表示 agent 的 BDI 对应求解问题的知识库。它们都由一系列状态组成, $B = \{b_i\}$; $D = \{d_i\}$; $I = \{i_i\}$, $i = 1, 2, \dots, n$; $P = \{P_j\}$, $j = 1, 2, \dots, M$, $P_j = (n_1, n_2, \dots, n_k)$; A 可能对应 B, D, I 中的一个要素, 也可能对应 B, D, I 二个要素或三个要素, 一般情况对应三个要素形成的知识库。

2.2 多 agent 并行计算模型描述

用 agent 进行问题求解时, 常常会遇到这样的问题: 一个 agent 需要解决多个问题, 对于不同类的问题, 就对应不同的知识库, 对于解决同一个问题, 需要的知识库是相同的。但一个任务是可以由多个 agent 解决的, 这时候往往就出现调用知识库的冲突, 为了避免这种问题的发生, 就利用多 agent 并行计算模型来进行问题求解, 在一个时间片内, 每一个 agent 只能对一个问题进行求解。

多 agent 并行计算模型指多个 agent 并行地执行计算或执行动作。在以往的多 agent 并行计算模型的研究上, 往往是从网络的角度考虑充分利用网络资源, 但对于如何更好地将多 agent 有机地组织起来, 真正达到内部的并行的文献并不多见。文中借助二分图的思想, 将若干个 agent 有机地组织起来并行求解问题。为了进一步说明多 agent 是如何并行求解问题的, 这里进一步给出构造多 agent 并行计算模型的概念。

在 agent 的研究中, 其中以 BDI 模型的研究最为典型, 那就以 BDI 模型为 agent 计算模型的研究基础, 约定: $Agents(ag_1, ag_2, \dots, ag_i)$ 表示一组 agent, $KL(kl_1, kl_2, \dots, kl_i)$ 表示 agent 的知识库。

2.2.1 多 agent 关系图

多 agent 关系图表示为: $\langle Agents, \Delta, KL \rangle$, 其中, Δ

表示 $Ag_i \in Agents$ 和 $kl_i \in KL$ 之间的关联。

多 agent 之间, 除了 agent 与知识库之间的关联以外, 还存在多种关联。应充分挖掘它们之间的这种关联问题, 以便更好地实现问题求解目的。下面就以 agent 的 BDI 进行研究, 从多方面考虑它们之间的关联问题。

约定: 每个 agent 用 $ag_i(B_i, D_i, I_i, P_i, A_i)$ 表示, 其中的每一个 B_i, D_i, I_i, P_i, A_i 都是以一维数组的形式表示。对于某个 ag_i 来说, $B \times D \rightarrow I$ 表示: 每个 agent 都根据自己的信念和愿望形成自己的意图, 也就是形成自己的行动规划。在这个过程中, ag_i 要调用知识库中的知识。在多个 agent 中, 单一的 agent 在进行问题求解时, 往往需要了解别的 agent 的信念、愿望或者意图, 以便达成统一的规划, 这可以使多 agent 尽量减少冲突, 共同完成问题。为了减少多个 agent 之间的冲突, 减少资源的浪费, 在进行问题求解时, 应该找到更多的共同点, 这样就可以借助 agent 之间的关系, 利用数学工具描述 agent 并找出其最大共性。

多个 agent 的信念组用 (B_1, B_2, \dots, B_n) 表示, 这样就组成了一个矩阵, 可以用矩阵求其共同的信念, 表示为:

$$D' = (B_1, B_2, \dots, B_n) = \begin{pmatrix} b_{11} & \dots & b_{1n} \\ \vdots & & \vdots \\ b_{m'1} & \dots & b_{m'(\cdot)n} \end{pmatrix}$$

有了这种表达形式, 就可以借助矩阵的运算规则进行处理, 由此有:

(1) 以 b_{11} 为基准, 将每一列与 b_{11} 相同的调换到第一行。

(2) 以 b_{21} 为基准, 将每一列与 b_{21} 相同的调换到第二行。

(3) 依次类推, 直到 $b_{m'1}$ 。

若找到最大相同的信念, 则形成 B' , B' 表示共同的信念。这就为多个 agent 进去共同求解问题提供了共同的信念。信念达到了最大化的一致, 但对于愿望, 仍需要达到最大的并行, 不然即使信念相同, 不同的愿望也会形成不同的意图。

同理, 对于 agent 的愿望, 也可以用矩阵表示为:

$$D' = (D_1, D_2, \dots, D_n) = \begin{pmatrix} d_{11} & \dots & d_{1n} \\ \vdots & & \vdots \\ d_{m'1} & \dots & d_{m'(\cdot)n} \end{pmatrix}$$

上述愿望矩阵也用求相同的信念的办法求出愿望 D' 最大矩阵 D' , 对于愿望, 多个 agent 之间存在许多不同的愿望, 这里只求出共同的愿望, 并舍弃其它不同的愿望。在形成多 agent 小组时, agent 也可以先通过感知等手段将多个愿望进行协作, 并形成新的愿望子集, 然后再用矩阵的方式给出共同的愿望。

在形成最大的信念矩阵和共同愿望后, 利用 $B' \times D' \rightarrow I'$ 这个公式, 形成新的意图, 每一个愿望形成一个新的意图, 然后将意图送给 agent 小组的每个成员, 再对问题进行求解。

$$B^c \times D^c = (b_1^c \ b_2^c \ \cdots \ b_m^c) \times (d_1^c \ d_2^c \ \cdots \ d_n^c)$$

$$= \begin{pmatrix} (b_1^c \ b_2^c \ \cdots \ b_m^c) \times d_1^c \\ (b_1^c \ b_2^c \ \cdots \ b_m^c) \times d_2^c \\ \cdots \\ (b_1^c \ b_2^c \ \cdots \ b_m^c) \times d_n^c \end{pmatrix} = \begin{pmatrix} i_1^c \\ i_2^c \\ \vdots \\ i_n^c \end{pmatrix}$$

2.2.2 多 agent 关系图中的关联

令 $Agents = \{1, 2, \dots, n\}$, 在多 agent 关系图 $\langle Agents, \Delta, KL \rangle$ 中定义:

$A_i = \{kl | kl \in KL, [i, kl] \in \Delta\} (1 \leq i \leq n)$, 则可将 agent 与知识库关联起来。

2.2.3 多 agent 并行计算模型及其并行策略

从构建多 agent 模型的过程可以看出, 我们构建的多 agent 关系图具有二分图的性质, 因此可以利用二分图的性质找到多 agent 的并行模型。

在多 agent 关系图 $\{[ag_i, kl_i], [ag_i, kl_i], \dots, [ag_i, kl_i], kl_i\}$ 中, 如果 ag_i, ag_i, \dots, ag_i 及 kl_i, kl_i, \dots, kl_i 都是两两互不相同的, 则称多 agent 的一个并行计算模型, 记为 $A(A_1, A_2, \dots, A_k)$

并行策略及其分类: 从 agent 的思维状态看, 每个 agent 都有自己的信念、愿望和意图, 这是 agent 解决问题的关键。信念是 agent 解决问题的依据; 愿望是 agent 解决问题的动力, 只有 agent 有某种愿望时, 才能驱使 agent 形成某种意图; 意图是 agent 按照某种规划设定的解决问题的行为或行动。按照 agent 的主要思维可以分为信念并行, 愿望并行, 意图并行。解决每种问题都得找到与之相对应的数据库, 特别是在组成多 agent 的协作是为了解决 agent 之间的冲突, 应该寻求最大化的并行。

例如: 4 个 agent (ag_1, ag_2, ag_3, ag_4) 去完成 5 个任务 (T_1, T_2, T_3, T_4, T_5)。每完成一个任务, 对应一个 kl。如何能保证 agent 最大的并行化, 就可以寻求 agent 的最大并行模型。假设 ag_1 能完成 T_1, T_3, T_4, T_5 ; ag_2 能完成任务 T_1, T_2, T_4 ; ag_3 能完成 T_2, T_4 ; ag_4 能完成 T_1, T_2, T_3, T_4 。这样就可以利用下面的求多 agent 并行计算模型的最大匹配来解决这个问题。

2.3 利用匈牙利算法求多 agent 的并行计算模型

算法^[5]: 设 $\langle Agents, \Delta, KL \rangle$ 是一个多 agent 关系图, 其中, $Agents = \{ag_1, ag_2, \dots, ag_m\}$, $Y = \{kl_1, kl_2, \dots, kl_m\}$ 。M 是该关系图的一个匹配。

首先用 (*) 标记 agents 中所有与 M 中的边不相关联的结点, 反复执行下面两步, 直到不能进一步标记为止:

第一步, 对于所有的最新标记的结点 ag_i , 经由不在 M 中的边, 用 (ag_i) 标记该边关联的所有 KL 中的结点。如果该结点已被标记, 则不再重新标记。

第二步, 对于所有最新标记的结点 kl_i , 经由在 M 中的边, 用 (kl_i) 标记该关联的所有 X 中的结点。如果该结点已被标记, 则不再重新标记。

所谓不能进一步标记, 有如下两种情况:

(1) 在第二步中, 存在最新标记的结点不与 M 中的任何边相关联, 这时叫做第二步结束。此时, 从该结点开始, 通过标记回溯到标记为 (*) 的 agents 中结点, 得到一条关于匹配 M 的交错链 r 。用 r 中不在 M 中的边代替 r 中在 M 中的边, 再加上不在 r 中的 M 中的边, 构成新的匹配 M' , 显然, M' 比 M 的边数多 1。用 M' 重新执行算法。

(2) 在第一步中, 所有新标记的结点或者不关联非 M 的边, 或者经由不在 M 中的边所关联的 KL 中的结点均已被标记, 这时叫做第一步结束。此时, 匹配 M 是多 agent 关系图具有最大边数的匹配。

利用这个算法, 上例的问题就可以解决了, 此算法能保证分配到任务的 agent 都并行地执行任务, 它们对知识库的调用也不会发生冲突。但是存在的缺陷是不能保证完全匹配。

2.4 多 agent 并行计算模型的相关性质

(1) 若 $\{[ag_i, kl_i], [ag_i, kl_i], \dots, [ag_i, kl_i]\}$ 是多 agent 的一个并行计算模型, kl_i, kl_i, \dots, kl_i 是执行 agents 的一个相异代表系。

(2) $\{A_1, A_2, \dots, A_n\}$ 有相异代表系, 当且仅当多 agent 关系图有一个 n 边的匹配 (称为完全匹配); 如果存在一个完全匹配的话, 则多个 agent 就是完全并行了。

(3) $\{A_1, A_2, \dots, A_n\}$ 有相异代表系的最大子集数等于多 agent 模型的最大并行数。

(4) 利用二分图这种求最大的匹配的方法也可以应用到多 agent 冲突问题的求解上, 当多个 agent 在多个问题上存在冲突时, 可以抽象出模型的二分图, 然后在利用求二分图最大匹配的方法, 解决多 agent 之间的冲突。

3 小结

借助二分图一些基本概念构建了多 agent 并行计算模型; 分析了多 agent 模型之间信念、愿望和意图的关系, 给出了求解某一个问题的共同信念、相同的愿望的方法, 再利用 $B \times D \rightarrow I$ 的公式, 求出共同的意图, 然后再将共同的意图给每个 agent 执行; 对多 agent 并行计算模型进行分析, 并提出了并行计算模型的一些相关公约, 给出了求并行计算模型的算法。

参考文献:

- [1] 王治国, 王利, 刘广钟. 基于 Internet 的 agent 并行计算[J]. 广西师范大学学报(自然科学版), 2003, 21(1): 156-159.
- [2] 韩泉叶, 张锋. 一种基于多 agent 的分布式计算模型研究[J]. 兰州铁道学院学报, 2003, 22(1): 98-100.
- [3] 张建, 陆鑫达. 用移动代理实现并行计算[J]. 计算机工程, 2002, 28(7): 30-31.
- [4] 郝水侠, 李凡长. 多 agent 的并行思维算法[J]. 计算机工程与应用, 2004, 40(10): 42-45.
- [5] Brualdi R A. 组合数学(英文版第 3 版)[M]. 北京: 机械工业出版社, 2002.