学校代码 10530　　学　　号 201111031652
分 类 号 TP18　　密　　级 公开

湘潭大学

硕 士 学 位 论 文

不确定多 agent 规划中的观察信息约简

学 位 申 请 人 伍　选
指 导 教 师 文中华 教授
学 院 名 称 信息工程学院
学 科 专 业 计算机科学与技术
研 究 方 向 智能规划

二〇一四年 五 月 五 日

Nondeterministic Observation Reduction in Multi-Agent Domain

Candidate Wu Xuan

Supervisor Prof. Wen Zhonghua

College College of Information Engineering

Program Computer Science and Technology

Specialization Automated Planning

Degree Master of Engineering

University Xiangtan University

Date May ,5th 2014

湘潭大学

学位论文原创性声明

本人郑重声明：所呈交的论文是本人在导师的指导下独立进行研究所取得的研究成果。除了文中特别加以标注引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写的成果作品。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律后果由本人承担。

作者签名：　　日期：　　年　　月　　日

学位论文版权使用授权书

本学位论文作者完全了解学校有关保留、使用学位论文的规定，同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权湘潭大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

涉密论文按学校规定处理。

作者签名：　　日期：　　年　　月　　日

导师签名：　　日期：　　年　　月　　日

摘　要

观察信息约简是智能规划的前沿领域。目前国内外对观察信息约简的研究主要是集中于针对单个 agent 的强规划解，还有一些问题有待于进行观察信息约简方面的研究，尤其在多 agent 规划领域中，求解 agent 协同规划解需要考虑每个 agent 之间的关系，使它们在竞争和冲突的情况下，实现自己或集体的目标。如何在多 agent 协同规划中进行观察信息约简，提高多 agent 协同工作的效率并降低执行中的代价，是一个值得研究的问题。

本文针对不确定多 agent 规划系统的问题，讨论了多 agent 规划领域中的观察信息约简问题。文章吸纳了不确定规划中观察信息约简的基本思想，将它推广到多 agent 领域。本文首次提出了多 agent 规划领域的最小观测集问题。文章设计了一种可区分性矩阵，来表示观察变量与需要区分的状态对的对应关系。基于可区分性矩阵，本文建立了一种算法，通过分枝限界法搜索可区分性矩阵不同的化简路径来求解多 agent 规划的最小观测集问题。经过算法示例和理论证明，表明算法能够正确找到解，且找到的解是符合最优要求的。算法的时间复杂度不高于 $O(2^m km)$不低于 $\Omega(km^2)$,其中 k 表示需区分状态对的数目，m 表示观察变量的数目。

本文在最小观察集问题的基础上扩展地提出并定义了多 agent 规划领域的最优观察集合问题。并提出了带权值的可区分性矩阵来反映观察变量与需要区分的状态对之间的关系。并针对该问题，设计了一种遗传算法与贪心法相结合的混合算法求解问题。通过实验和理论证明，算法能有效地解决该问题。

关键词：观察信息约简，不确定规划，智能规划，多 agent 规划，分层算法；

ABSTRACT

Observe the information reduction is a frontier field of intelligent planning. The current research on observation information reduction abroad are mainly focused on planning for the single agent solution, and some problems need to be studied to observe the information reduction, especially in the area of multi agent planning, solving the agent collaborative planning solution needs to consider the relationship between each agent, so that they are in competition and conflict situations next, the achievement of their own or collective goal. How collaborative planning were observed information reduction in agent, improve the efficiency of cooperation of multi agent and reduce the execution cost. Is a problem worthy of study.

Aiming at the uncertain multi agent planning system, discusses the problems in the field of multi observation information reduction in agent programming. This article absorbs the basic thoughts of uncertain information reduction plan, it can be extended to the field of multi agent. This paper first presents the minimum observed multi agent planning problem. This paper presents a distinguishable matrix, to represent the observed variables and the need to distinguish between the state of the corresponding relationship. Distinguish matrix based, this paper establishes a set of algorithms, through the branch and bound method to search the minimum observed distinguishable matrix Jane path different to solve multi agent planning. As proved by the algorithm application and theory, which indicates that the algorithm can find the solution, and find the solution is consistent with the optimal requirements. The time complexity of the algorithm is not higher than O (2^mkm) of not less than (km^2), where k denotes the number of the state should be distinguished, M represents the number of observed variables.

Based on the minimum set problem on observation of extended forward and a detailed definition of the optimal multi agent planning field observation set problem. And put forward the weighted distinguishable matrix to reflect the observed variables and the need to distinguish between the state of the relationship between. And to solve this problem, a hybrid algorithm for solving the problem and design a genetic algorithm with the greedy method. Through experiment and theory analysis, the algorithm can effectively solve the problem.

Keywords: Observation information reduction , nondeterministic planning,
intelligent planning, multi-agent planning , state layered algorith

目 录

第 1 章 引言

1.1 课题的研究意义和研究现状

智能规划是人工智能的重要研究方向，由于其研究领域具有的理论价值和实用价值[1-5]，一直受到研究者们的重视。其应用领域从娱乐生活中电子游戏的 AI 设计，工业生产的机器人控制和物流优化，一直到尖端的航天航空。可以说只要智能规划的研究取得很小的进展，对于我们的生活就会产生巨大的影响。近年来，关于智能规划的研究已取得长足进步。其研究重点已从单个 agent 领域扩展到多 agent 领域，从确定环境确定动作下的经典规划转变到了未知环境下的不确定规划。

多 agent 规划（MAP，Multi-Agent Plan）。也是智能规划当下的研究热点[6-10]。agent 是指问题领域中的智能实体。由于存在的实际环境的不同，其特征和能力也会不同。但一般认为，Agent 具有自主性，主动性，交互性，反应性，智能性。实际问题中的 agent 常常不止一个。这些活跃在同一个规划领域中的几个，几十个，甚至更多的 agent 共享资源，活动和目标。怎样使它们在或者协同，或者竞争，或者漠不关心的状态下一起行动。这样的问题就是多 agent 规划（MAP，Multi-Agent Plan）问题。它出现在多机器人环境、分布于互联网的协同软件、物流、制造业、军事领域以及游戏等应用领域。2002 年，AAAI 就设立了 MAP 的 workshop; 2005 年和 2008 年, 国际规划与调度会议(International Conference on Automated Planning and Scheduling, ICAPS) 设立了 MAP 的 workshop;2010 年,ICAPS 和 MAS 研究领域的顶级会议(International Conference on Autonomous Agents and Multi-agent Systems, AAMAS) 在多伦多举办,且有多个 joint section 讨论 MAP。作为分布式智能和多 Agent 系统（MAS，Multi-Agent System）的交叉领域。与传统的单 agent 规划领域相比较，无疑，它有更高的资源利用率和问题求解速度。但是同时，它需要关注 agent 之间的关系。怎样使 agent 协同工作，怎样避免或最大减少 agent 之间的冲突。怎样合理分配资源而防止死锁的发生，等等都是多 agent 规划领域所要涉及的问题。多 agent 规划问题的主要困难在于 agent 之间的合作[7]，在寻找其规划解的过程中，个体 agent 往往只考虑自身的执行情况，缺乏全局观，且系统环境也在动态变化中。目前，对于解决多 agent 规划问题的方法一般有：以 A*,WHCA*,LRA*为代表的启发式搜索[8-10]，类似蚁群算法[11]，遗传算法的智能算法[12]，与博弈论结合的算法[13]，引入一定社会规则的算法[14]，基于协商机制的合作求解算法[15]，等等。总之，如何让 agent 在互不

干扰的情况下，甚至能够有一定的合作关系，共同完成一个任务，解决这一难题是非常有意义的。

在现实世界中，不论是客观事物还是主观认识都存在不确定性。其中随机性和模糊性是不确定性的两种重要形式。为了更好地反映现实问题，智能规划中有专门的分支来研究在未知环境下的不确定动作效果的规划，这就是不确定规划(non-deterministic planning, NDP)。近年来，IJCAI(International Joint Conferences on Artificial Intelligence)及 AAAI(Association for the Advancement of Artificial Intelligence Conference on Artificial Intelligence)都设立了专门的 Session 来讨论不确定规划。在不确定领域中，观察信息约简是其研究的前沿领域。观察信息约简，在 2007IJCAI 上首次提出[19]，从 07 年至今，已有不少研究者提出了关于观察信息约简的新问题和新方法[20-25]。2007IJCAI 上第一次提出了观察信息约简的概念，后来有研究者在这方面进行了研究。其中文献 20 和文献 21 提出了基于动作对的观察信息约简方法和基于上下文的观察信息约简方法。文献 22 提出了构造最小观察变量集和最优观察变量集的通用方法，并从 3 个方面确立了单个 agent 的观测约简的三个基本问题，一是如何找最小观测集合,二是如何在观测代价不均等时找最优观测集合,三是如何找到容错的最优观测集合。文献 22 基本上是基于图论中的覆盖理论和整数规划理论来解决问题，但是这些方法一般要已知规划解后才能进行信息约简，而且算法时间复杂度是指数级的；文献 23 提出了基于一致性规划的构造最小观察变量集的方法，假设问题的规划域是部分可观察或完全不可观察，假设所有的状态变量都不是观察变量，在此基础上逐步增加必要的观察变量，从而最终得到一个必要的观察变量集合；在添加必要的观察变量过程中；该方法不要求得到所有变量的相关信息；从而有很好的通用性；根据是否存在单个观察变童能够区分城中任意两个状态的问题就能求得规划解；文献 23 的贡献在于它可以在规划解未知的情况下进行观察信息约简，但是它所求得的解是一个尽可能小的观察变量集合，,并不能保证该集合最小。文献 24 针对强循环规划的观察信息展开研究，提出了一个对强循环规划观察信息进行约简的算法 ORSCP 算法，该算法能够找出在完全可观察条件下强循环规划的最小观察变量集合，通过使用最小观察变量集合，可以进一步提高执行强循环规划解的效率。文献 25 提出将多 agent 规划算法与观察信息约简相结合该文设计了一种用于不确定规划领域中多 agent 求解协同规划解的 ORMAP 算法。该算法首先根据基于模型检测的不定规划中的状态分层思想，将问题领域的所有状态进行分层；以此来减少不同的 agent 的冲突，再利用以最小代价优先的回溯法搜索协同规划解，同时在解的搜索过程中选择最小的观察信息集，使求出的协同规划解是众多符合条件的协同规划解中所需要的观察信息最小或接近最少，这样就达到了信息约简的目的。

1.2 本文主要工作和创新点

本文主要研究了多 agent 规划的信息约简问题。定义了多 agent 规划领域的最小观察集合，提出了在多 agent 规划解中求最小观察集合问题，并设计了基于可区分矩阵的算法，来求解问题。并证明算法正确和有效。定义了多 agent 规划领域的最优观察集合，提出了在多 agent 规划解中求最优观察集合问题，并设计了带权值的可分区矩阵，在带权值的可区分矩阵上，建立了基于遗传算法的算法来求解多 agent 规划领域的最优观察集合问题，并证明该算法正确和有效。本文还讨论了在要求观察冗余的情况下，如何求最优观察集合的问题。

在现有的求解多 agent 规划解算法中，一般是基于完备的环境信息和确定的动作效果。而对于不完备的环境信息和不确定的动作效果下的多 agent 规划问题，却研究不多。本文的创新之处在于是在多 agent 规划领域中进行观察信息约简的初步探索，将多 agent 规划和不确定规划这两个智能规划的前沿领域结合起来。不但提出了如何在多 agent 规划领域进行信息约简这个问题，同时设计了针对该问题的算法。在算法中，首次提出了用可区分矩阵来反映可区分状态对和观察变量的关系。以上都是具有首创性的工作。

1.3 本文的组织结构

第一章概要介绍了智能规划的研究背景、研究现状和发展状况。

第二章介绍了不确定规划、观察信息约简以及多 agent 规划领域的基本知识和相关术语。

第三章讨论了在已知多 agent 规划领域规划解的情况下，如何进行观察信息约简的情况。其中定义了多 agent 规划领域的最小观察集问题并提出了求解算法，定义了多 agent 规划领域的最优观察集问题并提出了求解算法，且均给出了算法的证明。

第四章讨论了在要未知多 agent 规划领域规划解的情况下，如何进行观察信息约简的情况。并提出了 CLMAP 算法来解决该问题。

第五章对全文的研究工作进行了总结，并展望了未来的相关工作。

第 2 章 相关知识

本章主要介绍后文中会用到的不确定规划、多 agent 规划以及观察信息约简的基本概念。

2.1 不确定规划

在客观世界中，不确定规划研究的意义在于：解决实际应用中带有不确定性动作的规划问题。不确定规划与经典规划相比较，规划问题没有了确定性、完全可观察性以及可达性目标这些条件的限定，使得不确定规划问题更加复杂的同时也更加贴近于现实。由于现实社会中的信息具有非常高的复杂性和动态性，所以非常完整的信息非常难获得，很多领域的研究问题如：决策学、系统科学、信息科学、管理科学、工业工程、计算机科学等都存在许多人为或者客观的不确定因素。

不确定规划研究领域中有关几个限制要求的解释如下：

1. 确定性：确定性的概念是指对客观世界的一个简单描述，它的前提是假设客观世界的发展是完全按着一条可推测的线路进行的，即认为世界的所有动态变化是能够全部确定的，而且是固定在某个发展轨迹上的[26]。对于建模系统来说，执行系统中的动作可以预知其效果，即对每个可执行的动作，其产生的结果是完全确定的的，都是从一个状态具体转移到另一个唯一确定的状态。
2. 不确定性：相对于确定性来说，不确定性更贴近于现实，现实中发生的事件大多是不能确定会产生什么样的结果的，所谓理想的系统模型在现实世界中是不可能存在的。对于一个包含不确定性的系统模型而言，不确定动作具有可执行性，但执行该动作后，其产生的结果是不可预知的，即执行不确定动作所达到的状态有可能是一个也有可能是多个。
3. 部分可观察性：在许多应用中，系统的状态在运行时并非是完全可观察的，而是部分可观察的，也就是说系统所提供的观察信息并不能区分所有的状态。控制器不能掌握整个状态转移系统的信息，而只能观察到其中的部分信息，各个观察值对应的不是单个状态而是一个状态集合。无论是在理论还是实践中，基于部分可观察性的规划问题都被证明是一类困难的问题。
4. 完全可观察性：在实际应用中，存在系统可以完全提供观察信息的情况。对于完全可观察的规划问题而言，可以通过观察获得整个不确定规划系

统的当下状态的一切信息，规划器获得的一个观察值只对应系统的一个状态。

5. 可达性目标：在经典规划中，由于没有不确定动作，理解可达性目标非常容易：执行目标能够到达一个最终目标状态，且在这个规划过程中存在一个唯一的状态序列对应执行过程。而不确定规划中，目标不再是单个状态，而是一个状态集合。由于动作的不确定性，一个规划的执行可能存在一个或多个的状态序列。

随着不确定性问题研究的兴起及其向各个领域的渗透，不确定性研究和智能规划研究相互融合产生了不确定规划。不确定规划能够更方便地对现实中不确定情况进行分析处理，比确定规划具有更强的问题处理能力。由于对不确定性的引入，不确定规划也要比经典规划复杂许多，需要处理更大的状态空间和更复杂的动作路径。不确定规划的主要任务是对真实世界中具有不确定性的各种状态、动作、变换等进行处理分析，抽象成对应的规划问题，并制定出相应的规划序列来解决问题。与经典规划相比较，不确定规划放宽了确定性、目标可达性等限定条件。这使得不确定规划模型更加符合真实世界的情况，但同时也使得不确定规划问题更加复杂。在客观世界中，许多领域的研究中都存在不同程度的人为或客观的不确定性，特别是决策管理学科、系统学科、信息学科等复杂度高或人为参与度较高的领域。不确定规划的实际应用已经非常广泛，也是由于其在实际应用领域中的作用和意义，推动着不确定规划的研究继续下去。

模型检测是一种自动验证技术，是在 1981 年由 Clark和Emerson 等人提出的[38-39]。模型检测利用状态迁移表示动作行为，用模态时序逻辑公式描述系统性质[40-43]，其核心思想是将一个过程或者系统抽象成为一个有穷状态模型。模型检测早期主要用于硬件设计的验证，由于其在硬件检测中的成功运用，以及计算机计算性能的提升，模型检测的运用范围已经扩大到各种软件的验证上，例如：程序查错，通信协议检测等，并且在其它领域也有推广运用。模型检测方法也可运用到智能规划的研究。基于模型检测规划方法的优势在于它能够处理现实生活中许多存在不确定情况的问题，已经产生了 MBP[44]、MIPS[45]、TALplanner[46]及 GAMER[47]等功能强大的规划器。

模型检测规划将规划领域抽象为一个不确定状态转移系统。该系统中，在相同状态下执行相同动作可能得到不同的结果，需要等到动作执行完成，才能确定转移到了哪个状态。

在使用模型检测求不确定规划解时，根据从初始状态执行规划解最终到达目标状态的可靠性，将规划解分为强规划解、强循环规划和弱规划三类[29]。弱规划最为简单但实用性不大，强规划实用性高但相对复杂，强循环规划是其中

最复杂的一类规划问题，仍有待进一步研究。

定义 2.1（不确定规划领域） 不确定规划领域是一个不确定的状态转移系统$\sum = (S, A, \gamma)$。其中，S、A 分别为有限状态集合和有限动作集合，$\gamma : S \times A \to 2^S$ 则表示状态转移函数。由于本文讨论的是不确定规划，在提及规划领域时，均指不确定规划领域。

定义 2.2(规划问题) 规划问题 P 是一个三元组$(\sum, S_0, S_g)$，其中$\sum = (S, A, \gamma)$为对应的规划领域，$S_0 \subseteq S$ 是初始状态集，$S_g \subseteq S$ 是目标状态集。

定义 2.3（规划解）设 $P = (\sum, S_0, S_g)$是一个规划问题，规划解 $\pi = \{(s_0, a_0), (s_1, a_1), \ldots(s_i, a_i), \ldots\}$是一个状态动作序偶集合。其中 $s_i \in S$、$a_i \in A$，每个状态 s 只出现一次，也即每个状态 s 下只会执行一个动作 a；根据规划解能够选择当前状态下执行的动作从初始状态开始最终转移到达目标状态。

定义 2.4（执行结构） 设 π 是规划领域$\sum = (S, A, \gamma)$中的一个状态动作序偶集合。对于$\sum$中的规划问题 $P = (\sum, S_0, S_g)$，π 的执行结构为 $K = (N, T)$，其中，$N \subseteq S$、$T \subseteq S \times S$ 是按照下列要求生产的最小集合[41]：

(1) 若 $s \in S_0$，则 $s \in N$。

(2) 若 $s \in N$ 且 $\exists (s, a) \in \pi, s' \in \gamma(s, a)$，则 $s' \in N$ 且$(s, s') \in T$。

也就是说，从任意个初始状态开始执行 π，直至到达目标状态或者状态转移停止时结束，在这个过程中所有可能经过的状态转移路径组成执行解结构 K。K 的终止状态集记为 $S_{terminal}(K)$。

执行结构 K 可表示为有向图，其中 N、T 分别为，从 S_0 开始执行规划解可能到达的所有状态的集合及所有可能的状态转移的集合。

定义2.5（执行路径）设$K = (N, T)$是规划解π的执行结构，有序状态序列$L = \{s_0, s_1, s_2, \ldots, s_i, \ldots s_t\}$称为一条执行路径，其中$s_i \in N$；$s_t \in S_{terminal}(K)$；对任意$0 \leqslant i < t, s_i \notin S_{terminal}(K), (s_i, s_{i+1}) \in T$。

在模型检测规划方法中，根据规划解的执行结构中所包含规划路径的不同类型，即规划解执行的可靠性不同，可以将规划解划分为：弱规划解、强规划解和强循环规划解三种类型。

定义 2.6（弱规划解） 设 $P = (\sum, S_0, S_g)$是一个规划问题，π 是$\sum$中的状态动作序偶集合，由 π 导出的执行解结构 K 中存在执行路径 $l \in L(\pi)$是从初始状态开始至目标状态结束，则称 π 为规划问题的弱规划解。

执行弱规划解有可能到达目标状态，而不能保证到达，π是 P 的弱规划解当且仅当 S_0 中的任意状态 s，都可能通过π转移到 S_g 中的某个状态 s′ 。

定义 2.7（强循环规划解） 设 $P = (\sum, S_0, S_g)$是一个规划问题，π 是$\sum$中的状态动作序偶集合，由 π 导出的执行解结构 K 中的任意执行路径 $l \in L(\pi)$都是

从初始状态开始至目标状态结束，则称 π 为规划问题的强循环规划解。

执行强循环规划解可以保证系统最终到达目标状态，但对到达目标状态所需要的状态转移步数没有保证，极端情况下甚至需要无限步。π是 P 的强循环规划解当且仅当π的执行结构中所有的终止状态都属于 S_g。

定义 2.8（强规划解） 设 $P = (\sum, S_0, S_g)$是一个规划问题，π 是$\sum$中的状态动作序偶集合，由 π 导出的执行解结构 K 中的任意执行路径 $l \in L(\pi)$都是从初始状态开始至目标状态结束，且所有路径中都不存在环路，则称 π 为规划问题的强规划解。

执行强规划解可保证通过有限步的状态转移到达目标状态，π是 P 的强规划解当且仅当任意 $l \in L(\pi)$长度有限，且终止状态都属于 S_g。

由定义可知，强规划解一定是强循环规划解，强循环规划解一定是弱规划解。

2.2 多 agent 规划

多 agent 系统（Multi-Agent System ,MAS）是那些包括多个信息或者兴趣发散的自治体的系统。Agent 之间逻辑上独立，但通过共享知识，任务或者协同工作。多 Agent 规划（Multi-Agent Planning ,MAP）就是处理了包含多个 agent 的规划问题。多 Agent 规划的形式化描述是基于现有的形式化语言 PDDL/STRIPS 上的扩展如 MA-STRIPS。为了便于理解本文后面的内容，我们使用下面的定义来描述多 Agent 规划。

定义 2.5（不确定多 agent 规划领域）规划领域是一个不确定的状态转移系统，即执行动作后的状态是不确定的，可能是一个状态，也可能是一个状态集合。$\sum = \{S, n, A_{i=1\ldots n}, R\}$，其中：

- S 是有限状态集；
- n 是 agent 的数目；
- A_i 代表 agent_i 的有限动作集；
- $R \subseteq S' \times A \times S''$是一个不确定的转移关系，其中 $A = A_1 \times \ldots \times A_n$，$S' \subseteq S$ 为当前所有 agent 的状态集合，$S'' \subseteq S$ 为所有 agent 执行相应动作后的状态集合。

定义 2.6（不确定多 agent 规划问题）规划问题 $P = \{\sum, I_{i=1\ldots n}, G_{i=1\ldots n}\}$，$\sum$ 是一个规划领域，$I_i \subseteq S$ 是 agent_i 的可能初始状态集合，$G_i \subseteq S$ 是 agent_i 的目标状态集合。

定义 2.7 （不确定多 agent 规划解）对于一个规划问题 P 的规划解$\pi = \{\pi_1, \pi_2, \ldots, \pi_n\}$，其中$\pi_i$ 是 agent_i 的规划解，规划解π保证任意一个 agent 在执行动作

序列的过程中不会出现冲突的情况，并且能使每个 agent 都能到达自己的目标状态（所有 agent 是在同一时间执行动作，没有先后顺序）。

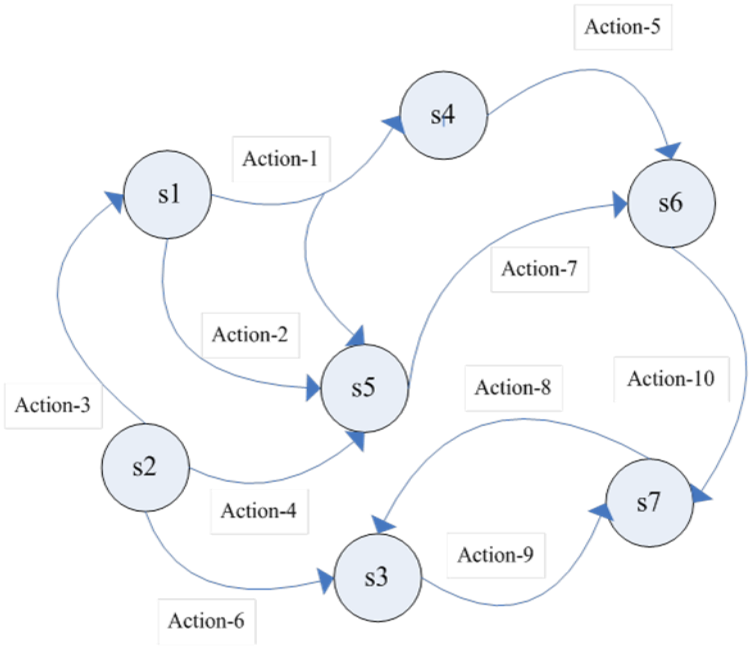

图 2.3　一个简单的不确定多 agent 规划领域

图 2.3 是一个简单不确定多 agent 领域$\sum$，假设有一个规划问题 $P = \{\sum, I_{1,2}, G_{1,2}\}$，其中 $I_1 = \{s_1\}$和 $I_2 = \{s_2\}$，$G_1 = \{s_6\}$和 $G_2 = \{s_7\}$。那么$\pi = \{\pi_1, \pi_2\}$是这个规划问题的一个协同规划解，其中$\pi_1 = \{(s_1, a_2), (s_4, a_5), (s_5, a_7)\}$，$\pi_2 = \{(s_2, a_6), (s_3, a_9)\}$

2.3 观察信息约简

2007IJCAI 上第一次提出了观察信息约简的概念[41]，后来有研究者在这方面进行了研究。其中有研究者提出了构造最小观察变量集和最优观察变量集的通用方法[42]；基于一致性规划的构造最小观察变量集的方法[43]；基于动作对的观察信息约简方法[44]和基于上下文的观察信息约简方法[45]等等。在不确定规划领域中，由于存在不确定动作，所以控制器在执行一个不确定动作后不知道 agent 当前所在状态，从而就不知道当前应该执行的动作，使得控制器不能继续执行，导致执行失败。通过观察信息就可以判断控制器在执行一个不确定动作后 agent 的所在状态，然而在执行过程中，并不需要所有观察信息。去除不必要的观察信息，达到减少开销的目的。

下面我们先理解一下定义：

定义 2.8（观察函数）设 S 是一个状态转换系统的状态集，V 是一个有限的观察变量集合。对观察变量 $v \in V$ 的赋值是函数$\chi: S \times V \to \{T, \perp\}$。

在完全可观察的情况下，任何不同的状态之间都是可分辨的，即$(\exists v \in V)(\chi(s, v) \neq \chi(s', v))$。而在完全不可观察的情况下，任何状态之间都是不可分辨的，即$(\forall v$

$\in V)(\chi(s, v) = \chi(s', v))$。

定义 2.9（需要区分的状态对集合）在执行规划解的过程中，控制器需要根据观察变量的值通过区分必要的状态对以确定当前 agent 所在状态，这些状态对组成的集合就是需要区分的状态对集合 *fsp*。

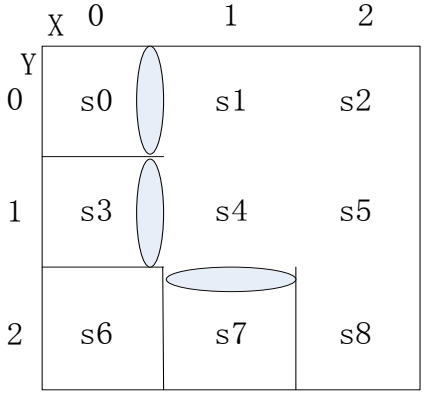

图 2.4　不确定的机器人移动领域

如图 2.4 所示假设有一方针 π_F= {(s0, GoEast), (s3, GoEast), (s1, GoSouth), (s4, GoSouth),(s7,GoEast)}，其中动作(s0, GoEast) = {s1, s4}和(s3, GoEast) = {s1, s4, s7}是两个不确定动作。现在一共有 10 个观察变量，分别为 WallS，WallN，WallE，WallW，X0，X1，X2，Y0，Y1，Y2。在完全可观察条件下，机器人可以通过观察变量获得对应的信息。例如，执行动作(s3, GoEast)会到达一个状态集合，控制器不知道 agent 的确切状态。在这个状态集合中，可以利用 WallN 将集合分为两部分{s1}和{s4, s7}，再利用 WallS 可以将{s4, s7}区分。这样用两个观察变量就能确定当前 agent 的确切状态，使得控制器能继续执行方针。在这个过程中并不需要所有的观察变量,实际上这个方针　F 只需要 2 个观察变量就能正确执行，这样可以节约 8 个观察变量获得观察信息的代价，从而减少不必要的开销。在该问题领域中，规划方针 π_F 需要区分的状态对 fsp={(s1,s4},(s1,s7),(s4,s7)}.而可观测的观察变量集 V={WallN，WallS，WallE，WallW，X0，X1，X2，Y0，Y1，Y2}.

定义 2.10（最小观察集合）能够用最少的观察变量保证规划解的执行的观察变量集合，就称为最小观察集合 V_{obs}。

直观地说，假设 *fsp* 是 *n* 个状态对的集合，*V* 是 m 个观察变量的集合。存在 V_{obs}（保证规划解的正确执行）$\subseteq V$，若对于任意一个 *V*'（保证规划解的正确执行）$\subseteq V$，都有$|V_{obs}| < |V'|$，则 V_{obs} 是一个最小观察变量集合。

定义 2.11 (代价) 设 fsp={state-pair$_0$,state-pair$_1$,...,state-pair$_{n-1}$}是状态对集合，V={v_0,v_1,...,v_{m-1}}是观测集合，对 $v_i \in V(0 \le i \le m)$,设其代价为 c_i,c_i 是一个非负实数，则得到一个与观测对应的代价集合 Cv={c_0, c_1,...,c_{m-1}}。

对任意观察集 $\Lambda \subseteq V$, 其代价（cost of observation set,简称 COST）是

$$C_{\Lambda} = \sum_{v_i \in \Lambda} C_i$$

定义 2.12（最优观察集合） fsp={state-pair$_0$,state-pair$_1$,...,state-pair$_{n-1}$}是状态对集合，V={v_0,v_1,...,v_{m-1}}是观测集合，Cv={c_0, c_1,...,c_{m-1} }是代价集合。对 fsp 有观测集$\Lambda \subseteq V$，若对任意 fsp 的观测集$\Lambda' \subseteq V$有$C_{\Lambda} \leq C_{\Lambda'}$,则称 Λ 是最优观测集。

2.4 本章小结

本章介绍了多 agent 规划，以及观察信息约简的相关背景知识，多 agent 规划领域的符号表示，多 agent 规划的解的符号表示；以及信息约简的基本概念，观察函数的定义，需要区分的状态对的集合的定义，最小观察集合的定义，代价的定义，最优观察集合的定义等。期间采用示例对相关内容进行了说明。

第 3 章 多 agent 规划的最小观察集

对于不确定规划领域的信息约简问题，已有不少学者对此进行了研究。但是还未延伸到多 agent 规划领域。首先来讨论一下当我们已经得到一个多 agent 规划领域的规划解，如何在执行过程中进行观察信息约简。观察信息约简的目标是省去多余的观察变量，而判断是否达成这一目标的条件是，是否获取了该问题领域的最小观察变量集和最优观察变量集。最小观察变量集针对的是没有权值的观察花费的规划问题，而最优观察变量针对的是有权值的观察花费的规划问题。我们参考单 agent 的最小观察集问题的定义[34]来扩展描述多 agent 规划的最小观察集问题。

3.1 问题描述

定义 3.1（多 agent 的观察函数）N 是参与规划的 agent 的集合，S 是状态转换系统的状态集合，V 是一个有限观察变量的集合，有观察函数χ：$N\times S\times V\rightarrow\{T,\perp\}$。

例如 $\chi(i,s_j,v_k)=\mathrm{T}$ 其中$i\in N,s_j\in S,v_k\in V.$一般来讲，不同观测者如果处于相同的状态和观测相同的观察变量，其观测结果也一样。即：

$$\chi(i,s,v)=\chi(j,s,v)\quad i,j\in N \qquad *$$

那么观察函数的第一个参数也可省略，注意下文如果未提出特殊说明，式（*）一般作为共识条件。

定义 3.2（多 agent 规划的最小观察变量集合）对于多 Agent 规划领域$\Sigma=\{S, n, A_{i=1\ldots n}, R\}$的规划解$\pi=\{\pi_1,\pi_2,\ldots,\}$，存在 $V_{obs}\subseteq V$，保证规划解的正确执行，并且对于任意一个 $V'\subseteq V$ 保证规划解的正确执行，都有$|V_{obs}|<|V'|$，则 V_{obs} 是一个最小观察变量集合。

由定义可知最小观察变量集合 V_{obs} 的性质

定理 3.1 对于任意一个多 Agent 规划领域的规划解π的最小观察变量集合 V_{obs} 总有

$V_{obs}=V_{obs}^{1}\cup V_{obs}^{2}\cup\ldots\cup V_{obs}^{i}\cup\ldots\cup V_{obs}^{n}$ 其中 V_{obs}^{i} 是第 i 个 agent 的规划解π_i 的最小观察变量集合。

由于 V_{obs} 保证整个规划解 π 的正确执行，所以 V_{obs} 也必然保证某一个 agent 的规划解。

Π_i 的正确执行,同样 V_{obs}^{i} 也保证序号为 i 的 agent 的规划解。因此 $V_{obs}^{i}\subseteq V_{obs}$。

根据定理 3.1，我们可以设计一种算法，已知一个多 agent 规划的规划解来

求它的最小观察变量集。

3.2 可区分性矩阵

由于在问题领域的方针中，一部分观察变量可以区分部分状态对。我们根据这些区分关系定义一个矩阵。这个矩阵可以表示观察变量与状态对的对应关系。而且围绕可区分性矩阵，我们可以建立一些计算和算法，来表达规划的动态过程。

定义 3.3（观察变量集与状态对的可区分性矩阵）

$$D_{sp-v}[i][j]=\begin{cases}0 & \chi(s_i^1,v_j)=\chi(s_i^2,v_j)\\ 1 & \chi(s_i^1,v_j)\neq\chi(s_i^2,v_j)\end{cases}$$

$$其中(s_i^1,s_i^2)\in fsp, v_j\in V, i=1..n, j=1...m$$

由定义可知图 2.4 中 π_F 的可区分性矩阵为：

$$\begin{array}{c} \\ (s_1,s_4)\\(s_1,s_7)\\(s_4,s_7)\end{array}\begin{array}{c}\begin{array}{cccccccccc}W_N & W_S & W_E & W_W & X_0 & X_1 & X_2 & Y_0 & Y_1 & Y_2\end{array}\\ \begin{pmatrix}1&0&0&0&0&0&0&1&1&0\\1&1&1&1&0&0&0&1&0&1\\0&1&1&1&0&0&0&0&1&1\end{pmatrix}\end{array}$$

3.3 基于可区分矩阵求解多 agent 规划的最小观察集合的问题

如图 3.1 所示，算法由 FindTotalStatePair,ConstructDivisiveMatrix,Reduction 这三个主要过程组成。FindTotalStatePair 是根据多 agent 领域的规划解求得全过程需要区分的状态对集合 ftsp，而 ConstructDivisiveMatrix 则是根据求出的 ftsp 和观察变量集合得出区分性矩阵 DM,最后的 Reduction 过程则是根据 DM 得到多 agent 规划的最小观察变量集合 V_{obs}。

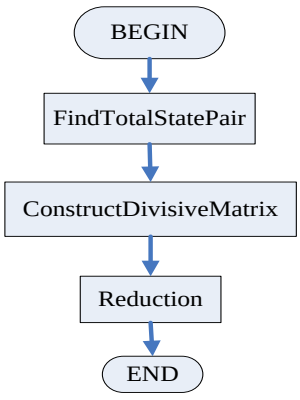

图3.1　求多agent规划的最小观测集问题算法整体流程

FindTotalStatePair 过程(算法 3.1)主要是依据 $ftsp=fsp_1\cup fsp_2\cup....Ufsp_n$ 其中

fsp_i 表示第 i 个 agent 规划解 π_i 的区分状态对，遍历全部 agent 的规划解后,就可以得到全过程的需要区分的状态对 ftsp。算法 3.1 中步骤 03：选取规划解中的一个非确定动作 a;步骤 05：找到动作 a 的执行后的状态集。从中任取得两个状态，组成状态对。步骤 06：如果得到的状态对还没有加入状态对集合 ftsp 中，就加入 ftsp 中。

Procedure1 FindTotalStatePair

Input:multi-agent domain plan problem $P(\sum(n,A,S,R),I_{1,2\ldots n,}G_{1,2\ldots n,})$,

a state-action table for multi-agent domain $\pi_f(\pi_1,\pi_{2,}..\pi_n)$,

Output:total state pair set ftsp

Begin

01 ftsp=∅;

02 for i=1 to n

03 if action a such that $(s,a)\in\pi_i$ and $|\gamma(s,a)|>1$;

04 then

05 select any $s_j,s_k\in\{s'|s'=\gamma(s,a)\}$ and j<k;

06 add (s_j,s_k) to ftsp;

07 endfor;

End

算法3.1 全过程需区分的状态对搜索

Procedure2 ConstructDivisiveMatrix

Input:total state pair set ftsp$\{sp_1,sp_2,..sp_k\}$, observation setV$\{v_1,v_2,\ldots,v_m\}$,

observation function χ.

Output:divisive matrix DM(dm)

Begin

01 for i=1 to k

02 for j=1 to m

03 if $\chi(s_i^1,v_j)\neq\chi(s_i^2,v_j)$ and $(s_i^1,s_i^2)\in$ ftsp

04 then dm[i][j]=1;

05 else dm[i][j]=0;

06 endfor

07 endfor

End

算法3.2 构造区分矩阵

ConstructDivisiveMatrix过程(算法3.2)主要是构造可区分性矩阵DM,dm[i][j]表示观察变量v_j是否可以区分状态对sp_j.构造这种数据结构主要为下一个过程做准备。

Procedure3 Reduction

Input:divisive matrix DM (dm),observation set V{v_1,v_2,...v_m}.

Output:minimal observation set V_{obs}

Begin:

```
01 for i=1 to k
02     for    j=1 to m
03           tm[i][j]=dm[i][j];
04     endfor
05 endfor
06    for i=1 to k
07            tr[i]=tm[i][1]+tm[i][2]+…+tm[i][m];
08     endfor
09     for j=1 to m
10            tc[j]=tm[1][j]+tm[2][j]+…+tm[k][j];
11     endfor
```

12 V_{obs}=∅;

13 V_t=V;

14 V_{obs}=dpSearch(TM,tr,tc,V_{obs},V_t);

15 return V_{obs};

End

算法3.3 观察信息约简

Reduction过程(算法3.3)是约简算法的主要步骤，根据文献[9]，我们可以知道求单个agent的最小观测集的过程等同于图论中求集合的最小覆盖集过程，同是NP难度问题。那么多agent的最小观测集问题同样是NP问题。可以使用贪心算法求近似解，也可以使用穷举法求最优解。此外，我们建立了可区分性矩阵，用矩阵计算，来表示搜索过程。其中行计数数组元素tr[i]表示第i个状态对能被多少个观察变量区分，列计数数组元素tc[j]表示第j个观察变量能区分状态对的个数。

Procedure4 dpSearch

Input:temporal matrix TM I,observation set V{v_1,v_2,...v_m}

minimal observation set V_{obs}, total row count array tc,

total column count array tr .

Output:minimal observation set V_{obs}

Begin:

01 if I , j such that tm[i][j]=0 then return V_{obs};

02 if x , y, i such that tm[i][x]≥tm[i][y]

03 then return dpSearch(15bandon(TM,tr,tc,V_{obs},V,y));

04 if r ,c such that tr[r]=1 and tm[r][c]=1

05 then return dpSearch(pickV(TM,tr,tc,V_{obs},V,c));

06 select $V_j \in V$ such that tc[j] is 15bandon in (tc[1],tc[2],..,tc[m]);

07 V_{obs}'=dpSearch(15bandon(TM,tr,tc,V_{obs},V,j));

08 V_{obs}"=dpSearch(pickV(TM,tr,tc,V_{obs},V,j));

09 if |V_{obs}'|≤|V_{obs}"| then return V_{obs}';

10 else return V_{obs}";

End

算法3.4 回溯搜索

dpSearch过程(算法3.4)是回溯搜索最优算法，根据需要也可以调整为其它的近似算法。步骤06-07：如果一个观察变量可以区分的任何一个状态对，另一个观察变量也可以区分，那么最小观测集一定不包含这个变量；步骤08-09：如果一个状态对只能被一个观察变量区分，那么最小观测集一定包含这个变量；步骤10：选择可以区分状态对最多的观察变量进行判断；步骤11-14：进行回溯搜索，那种搜索路径产生的V_{obs}最小，就采用哪种搜索路径。

Procedure5 abondanV

Input:temporal matrix TM I,observation set V{v_1,v_2,...v_m}

minimal observation set V_{obs}, total row count array tc,

total column count array tr ,index c of observation v_c

Output:temporal matrix TM I,observation set V{v_1,v_2,...v_m}

minimal observation set V_{obs}, total row count array tc,

total column count array tr.

Begin:

4 for i=1 to k

02 if tm[i][c]=1

03 then

04 tr[i]=tr[i]-1;

05 tm[i][c]=0;

06 endfor

```
07    tc[c]=0;
08    V=V-{v_c};
09    return tm,tr,tc,V_obs,V
End
```

算法3.5 约减观察变量过程

```
Procedure6 pickV
Input:temporal matrix TM I,observation set V{v_1,v_2,...v_m}
            minimal observation set V_obs, total row count array tc,
            total column count array tr ,index c of observation v_c
Output:temporal matrix TM I,observation set V{v_1,v_2,...v_m}
              minimal observation set V_obs, total row count array tc,
              total column count array tr.
Begin:
01          for i=1 to k
02          if tm[i][c]=1
03          then
04                for j=1 to m
05                   if    tm[i][j]=1;
06                   then
07                         tm[i][j]=0;
08                         tc[j]=tc[j]-1;
09                 endfor
10                 tr[i]=0;
11    endfor
12    tc[c]=0;
13    V_obs=V_obs ∪ {v_c}
14    V=V-{v_c};
15    return tm,tr,tc,Vobs,V.
End
```

算法3.6 选定观察变量过程

abandon(算法3.5)和pickV(算法3.6)分别表示约减观察变量和选定观察变量的可区分性矩阵的操作。在abandonV中，步骤01-06：要约减观察变量V_c，先将矩阵c列全部置0，再更新行计数数组和列计数数组。在pickV中，步骤01-12：要

选定观察变量V_c,先将表示V_c的矩阵c列全部置0，然后将观察变量V_c可以区分的状态对所对应的矩阵行也全部置0，最后更新行计数数组和列计数数组。

3.4 算法分析

3.4.1 算法的正确性

因为该算法的终结条件只有一个，矩阵元素 tm[i][j]全部为零。分两种情况，一种情况是区分状态对集合 ftsp 为空，一种情况是区分状态对集合 ftsp 不为空。由算法知，矩阵元素 tm[i][j]为 0 表示可以区分的状态对已经没有了，而区分状态对集合为空，则表示需要区分的状态对已经没有。在第一种情况下，因为需要区分的状态对都已经被标记区分，该算法返回的解一定是可以区分所有状态的观测集。第二种情况，表示在这种选择观察变量组合的情况下，无法区分所有的状态对。依据算法，这时应该将最小观测集 V_{obs} 定义为 ERROR。而且$|V_{obs}|$赋值为 M.其中 M 表示一个比|V|大的整数。当算法在 dpSearch 过程 11-14 步，比较观测集大小决定选择时，算法不会选择大小为 M 的观测集，只会选择较小的观测集。这样就保证不能区分所有状态对的观察变量的组合不被选择。如果任意观察变量组合大小都为 M，则算法返回 ERROR，表示问题无解。

又因为算法在 dpSearch 过程有两个步骤 06-07 和 08-09 可以通过判断最小观测集一定包含哪些变量和一定不包含哪些观察变量来化简观测集，而且在选择观测集时，总选择较小的观测集。因此算法得到的解必定是可以区分所有状态对同时包含观察变量最少的集合。

综上所述，如果问题有解，该算法输出的必定是最小观测集。因此算法是正确的。

3.4.2 算法的时间复杂度

FindTotalStatePair,ConstructDivisiveMatrix,Reduction 的时间复杂性分别设为 T1,T2,T3。在一个规划问题 $P = \{\sum\{S, n, A_{i=1\ldots n}, R\}, I_{i=1\ldots n}, G_{i=1\ldots n}\}$中，agent 个数为 n,有限状态集 S 且|S|=s,观察变量集 V 且|V|=m

因为 FindTotalStatePair 过程是找需要区分的状态对的过程，必须遍历所有的 agent 的规划解。由于有 n 个 agent,所以遍历 n 次，每次遍历一个 agent 的规划解。要搜索 n 次，所以 $T_1=O(ns^2)$。ConstructDivisiveMatrix 是构造区分矩阵过程。设有 k 对状态队。$T_2=O(km)$。

Reduction 中的 dpSearch 过程是算法的主要部分。先考虑最坏的情况，假如

该过程的化简语句是无效的，即无法马上判断哪些观察变量是一定包含或者不包含在最小观测集内，算法必须遍历如图 3.2 树的每一个分支，而且在该图的除根结点的每一个结点都进行 abandonV 或者 pickV 操作。又由于 abandonV 或者 pickV 过程都是矩阵操作。则 $T_{abandonV}=T_{pickV}=Ckm$,其中 C 表示常数。又整个树有 2m-1 个结点，则 $T_3\leq(2^{m-1})(Ckm)$.即 $T3\leq O(2^m km)$。

然后考虑最好的情况，即化简语句每次都起到了作用的，算法只进行 m 次，且每次是 pickV 操作，那么 $T_3\geq\Omega$（km^2）。

那么, Reduction 过程的时间复杂度 T_3 不高于 $O(2^m km)$且不低于 Ω（km^2），其中 k 是状态对数目，m 是观察变量数目。

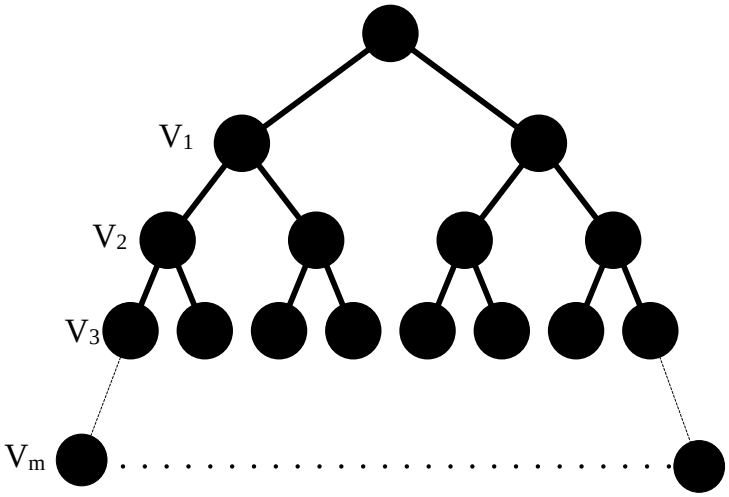

图 3.2 dpSearch 过程完全搜索树

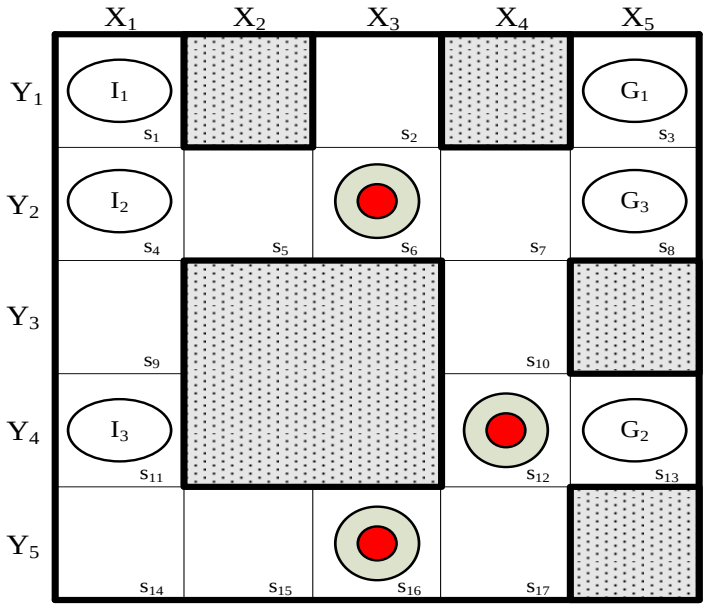

图 3.3　多机器人路径规划的二维网格图

3.5 算法举例

现在我们举例来说明算法，如图 3.3 三个机器人的二维路径规划问题，其中机器人 $agent_1$ 的初始位置 $I_1=\{s_1\}$,目标位置 $G_1=\{s_4\}$;机器人 $agent_2$ 的初始位置 $I_2=\{s_4\}$,目标位置 $G_2=\{s_{13}\}$; 机器人 $agent_3$ 的初始位置 $I_3=\{s_{11}\}$,目标位置 $G_3=\{s_{13}\}$。可以采取的动作 action={go-east,go-west,go-north,go-south}。观察变量集合 V={X_1,X_2,X_3,X_4,X_5,Y_1,Y_2,Y_3,Y_4,Y_5,WallE,WallW,WallN,WallS}。阴影部分表示不可进入区域。S_6,s_{12},s_{16} 是特殊区域，特殊之处在于，如果机器人从临近格子向该格移动，则机器人不会落在该格上，而是不确定地移动到该格的临近格子上，比如 go-east(s_5)=\{s_2,s_5,s_7\}。除了 s_6,s_{12},s_{16} 这三格外，其它网格上机器人的移动都是确定的，比如 go-east(s_4)=\{s_5\}，go-south(s_7)=\{s_{10}\}。

现在如果该问题有一个规划解 π（π_1，π_2，π_3），我们根据规划解 π，来求它的最小观测集 V_{obs}。其 π_1={(s_1,go-south),(s_4,go-east),(s_5,go-east),(s_2,go-west),(s_7,go-east),(s_8,go-north)};π_2={(s_4,go-east),(s_5,go-east),(s_2,go-south),(s_7,go-south),(s_{10},go-south),(s_{17},go-north)};π_3={(s_{14},go-east),(s_{15},go-east),(s_{17},go-north),(s_{13},go-west),(s_{10},go-north),(s_7-go-east)}。其中(s_5,go-east)，(s_2,go-west)，(s_{10},go-south)，(s_{13},go-west)，(s_{15},go-east)，(s_{17},go-north)是不确定性动作。

经过 FindTotalStatePair 过程，可以得到全局区分状态对集合 ftsp={(s_2,s_5),(s_2,s_7),(s_5,s_7),(s_{10},s_{13}),(s_{10},s_{17}),(s_{14},s_{17}),(s_{15},s_{17})}。

经过 ConstructDivisiveMatrix 过程，可以得到一个区分性矩阵 DM。

$$
\begin{array}{c}
 & X_1 & X_2 & X_3 & X_4 & X_5 & Y_1 & Y_2 & Y_3 & Y_4 & Y_5 & W_E & W_W & W_N & W_S \\
(s_2,s_5) & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\
(s_2,s_7) & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
(s_5,s_7) & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
(s_{10},s_{13}) & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\
(s_{10},s_{17}) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
(s_{13},s_{17}) & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\
(s_{16},s_{17}) & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0
\end{array}
$$

在 Reduction 过程中，经历一个如图 3.4 的搜索过程。最后可以得到该规划的最小观测集 V_{obs}={X_4,WallS},或者 V_{obs}={X_4,WallW}。

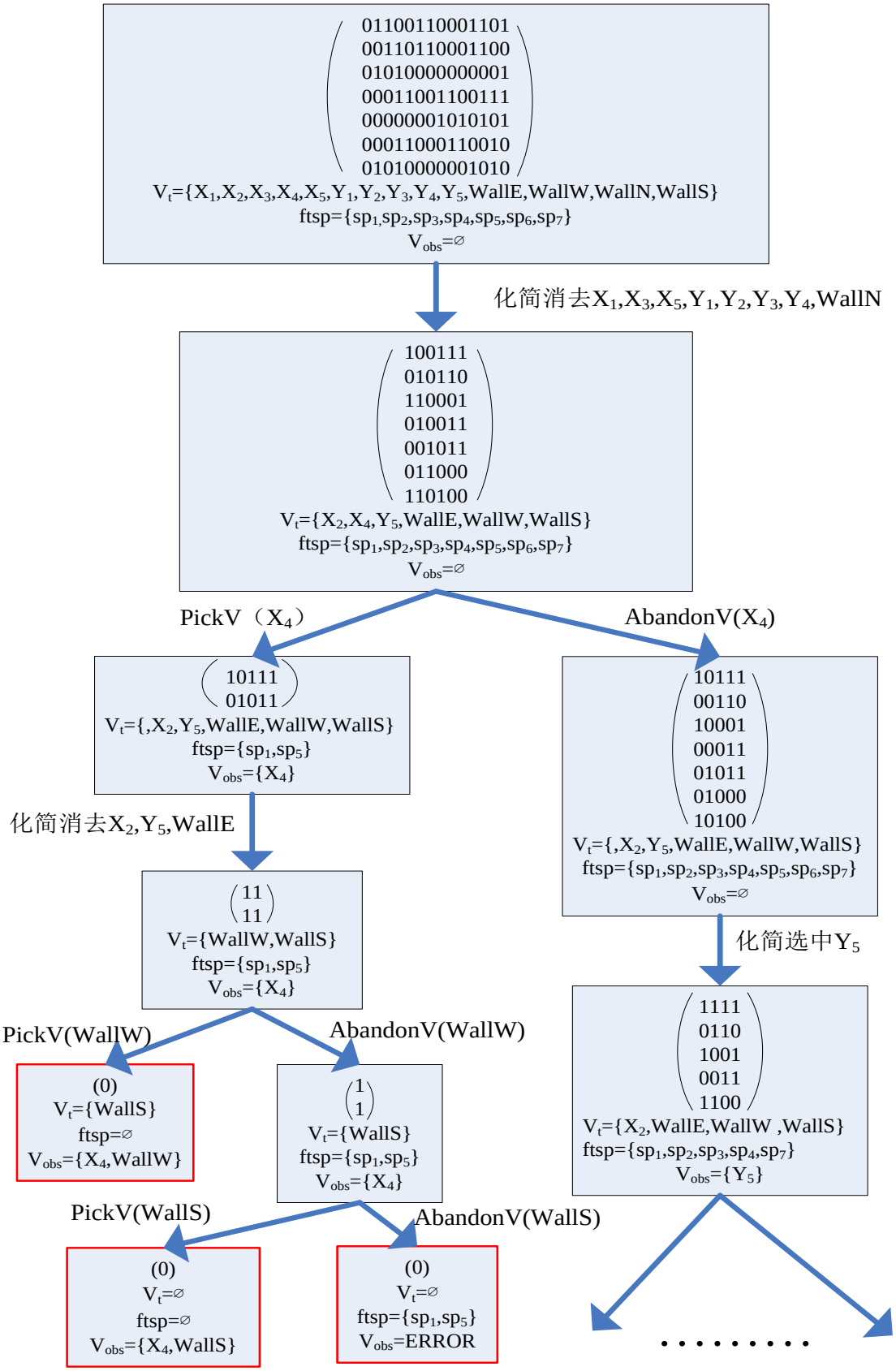

图3.4 观察信息约简的部分搜索图

第 4 章 多 agent 规划的最优观察集

在现实环境中，进行观测的观察变量并不是一样。尤其在多 agent 领域，单个 agent 由于规划行动的时间，环境等条件的不一样。必然会使观察代价不同。对于此类观测花费不一样的观察信息约简问题，我们称之为多 agent 规划领域的最优观察集问题。

4.1 问题描述

定义 4.1(代价矩阵)V={v_0,v_1,…..,v_{m-1}}是观察函数的集合，代价矩阵 CD={c_{ij}}。每个 A_i 对应 v_j 的花费为 c_{ij}，其中 $v_j \in V$，c_{ij} 是非负实数。

假设 fsp_i={state-pair$_{i0}$,state-pair$_{i1}$,state-pair$_{i2}$,…,state-pair$_{ik}$}是 A_i 的需要区分的状态集合对。对于 fsp 有观察集={$\Lambda_0, \Lambda_1, \Lambda_2, \ldots, \Lambda_{n-1}$},其中 Λ_i 表示对应 fsp_i 的可区分观察集。

那么对于 A_i 观察代价

$$C_i = \sum_{j=1}^{k} C_{ij} \bullet Ob_{ij} \; if v_j \in \Lambda_i \; then Ob_{ij} = 1 else Ob_{ij} = 0$$

整个规划系统的观察代价为：$C_\Lambda = \sum_{\Lambda_i \in \Lambda} C_i$

定义 4.2（多 agent 规划的最优观察集）在多 agent 规划领域中$\sum\{S, n, A_{i=1\ldots n}, R\}, I_{i=1\ldots n}, G_{i=1\ldots n}$ 中，对于 fsp 有观察集 $\Lambda=\{\Lambda_0, \Lambda_1, \Lambda_2, \ldots, \Lambda_{n-1}\}$，其中 Λ_i 表示对应 fsp_i 的可区分观察集。当对 fsp 的任意观察集 $\Lambda^* \subseteq V$,都有 $C_\Lambda \square C_{\Lambda^*}$,则观察集 Λ 是最优观察集。

由以上定义可知，最优观察集问题实际上是最小观察集问题的扩展。而后者是前者的一种特殊形式。如果代价矩阵里的花费值都一样，那么求最优观察集问题就转变成求最小观察集问题。

4.2 带权值的可区分矩阵

我们仍然用矩阵来表示观察变量与需要区分的状态对的对应关系。但是由于观测的花费值不同，沿用上一章的 0-1 矩阵就不足够承载观察变量的代价值这一信息。那么我们用一个无穷大和正实数组成的矩阵来表示对应关系。

定义 4.3（带权值的可区分矩阵）

$$VD_{sp-v}[ik][j]=\begin{cases}0 & \chi(s_{ik}^1,v_j)=\chi(s_{ik}^2,v_j)\\ c_{ij} & \chi(s_{ik}^1,v_j)\neq\chi(s_{ik}^2,v_j)\end{cases}$$

其中$(s_{ik}^1,s_{ik}^2)\in fsp_i, v_j\in V, i=1..n, j=1...m, k=i_1...i_k, c_{ij}\in DC$

如果观察变量可以区分状态对。则矩阵元素为观察变量区分该状态对所花的代价；如果观察变量无法区分状态对，则意味观察变量没有区分该状态对，那么所花的代价为 0。计算过程中，无穷大可以用一个大的实数表达来进行计算。如图 2-4 所示的多 agent 规划系统。有三个 agent，它们的协同规划解 $\pi_F=\{\pi_1, \pi_2, \pi_3\}$其中 π_1={(s0, GoEast), (s4, South), (s1, GoSouth), (s5, GoNorth),(s7,GoEast)}，π_2={(s3, GoEast), (s4, South), (s1, GoSouth), (s5, GoNorth),(s7,GoEast)}，π_2={(s7, GoEast), (s4, South), (s1, GoSouth), (s5, GoNorth) } 那么它的 fsp={$spair_{11}$,$spari_{21}$,$spair_{31}$,
$spair_{32}$}假设它的观察代价矩阵 DC 为：

$$\begin{array}{c} \\ agent_1\\ agent_2\\ agent_3\end{array}\begin{array}{c}\begin{array}{cccccccccc}W_N & W_S & W_E & W_W & X_0 & X_1 & X_2 & Y_0 & Y_1 & Y_2\end{array}\\ \begin{pmatrix}c_{00} & c_{01} & c_{02} & c_{03} & c_{04} & c_{05} & c_{06} & c_{07} & c_{08} & c_{09}\\ c_{10} & c_{11} & c_{12} & c_{13} & c_{14} & c_{15} & c_{16} & c_{17} & c_{18} & c_{19}\\ c_{20} & c_{21} & c_{22} & c_{23} & c_{24} & c_{25} & c_{26} & c_{27} & c_{28} & c_{29}\end{pmatrix}\end{array}$$

那么它的带权值的可区分矩阵 VD 为：

$$\begin{array}{c} \\ spair_{11}(s1,s4)\\ spair_{21}(s1,s4)\\ spair_{31}(s1,s4)\\ spair_{32}(s1,s7)\\ spair_{33}(s4,s7)\end{array}\begin{array}{c}\begin{array}{cccccccccc}W_N & W_S & W_E & W_W & X_0 & X_1 & X_2 & Y_0 & Y_1 & Y_2\end{array}\\ \begin{pmatrix}C_{00} & 0 & 0 & 0 & 0 & 0 & 0 & C_{07} & C_{08} & 0\\ C_{10} & 0 & 0 & 0 & 0 & 0 & 0 & C_{17} & C_{18} & 0\\ C_{20} & 0 & 0 & 0 & 0 & 0 & 0 & C_{27} & C_{28} & 0\\ C_{30} & C_{31} & C_{32} & C_{33} & 0 & 0 & 0 & C_{37} & 0 & C_{39}\\ 0 & C_{31} & C_{32} & C_{33} & 0 & 0 & 0 & 0 & C_{38} & C_{39}\end{pmatrix}\end{array}$$

此时问题转变成求 VD 的子矩阵 sub，该子矩阵必须满足以下要求：

- 该子矩阵行数 R_{sub}□VD 的行数 R_{VD}；
- 该子矩阵列数 C_{sub}=VD 的列数 C_{VD}；
- 该子矩阵每行的元素必须至少有一个元素不是 0；
- 该子距阵中所有的元素之和必须是所有满足前面三个要求的子矩阵中最小的。

显然这是一个 NP 难度的组合优化问题。如果使用穷举法所搜索最优解，一旦观察变量数量过多，算法时间就是指数级增加。如果使用贪心法搜索近似最优

解，就会陷入局部最优解。我们知道遗传算法来求解优化问题，是比较合适的，但是也存在过早收敛于局部最优解的问题。所以，本文采用遗传算法结合贪心算法的启发式算法来求解该问题。

4.3 遗传算法

4.3.1 遗传算法简介

遗传算法是基于模拟进化的智能算法[48]。其中的假设一般被描述为二进制的位串，位串的含义基于具体的应用。算法从若干的初始群体和集合开始，搜索合适的假设。当前的群体的成员产生下一代群体的方式是通过模拟生物的进化，例如随机变异和交叉。每次变异后，根据已有的适应值来评估当前的假设是否是适宜的。然后，通过一定的随机算法挑选出适应值最高的假设，作为能够产生下一代的父代。遗传算法多应用与优化问题和机器学习当中。

遗传算法是受生物进化启发，最初与 1975 年被美国 Michgan 大学的 Holland 教授提出，它并非是从一般到特殊或者从单一到复杂地搜索假设。而是通过变异和重组，已有的优秀假设来产生子代的建设。从每一步，将当前的群体的假设称作更新，是使用目前适应度最高的假设的后代来替代群体的某个部分。这个过程形成了假设的生成并测试柱状搜索，其中若干个最佳假设的变体最有可能在下一步被考虑。遗传算法的发展和普及主要是因为：进化被认为是在生物系统中，最成功的自适应方法之一，而且具有很好的健壮性；遗传算法的群体空间中，每个个体由多个部分组成，每一部分对总的适应度的影响难以建模；遗传算法易于并行化，且可降低由于使用超级计算机所带来的高昂代价。

4.3.2 遗传算法通用步骤

遗传算法的算法实质是构建假设空间，搜索假设空间并确定最优的假设。在遗传算法中，“最优假设”被定义为适应度最佳的假设，而适应度测试被预先定义好的数学定量，例如，如果算法解决的是背包问题，那么适应度就应该是背包里物品的总价值。

尽管遗传算法的不同实现在小地方会有所不同，但它们都有相同或相似的结构：算法迭代更新一个假设池，这个假设池称为群体。在每代的迭代中，根据适应度函数评价群体中的所有成员，一部分保持原样进入下一代群体，另外一部分通过遗传方法产生下一代成员。

算法 4-1 描述了遗传算法的原型。算法需要输入的包括：用来排序的候选假

设的适应度函数；定义算法终止时适应度的阀值；要维持的物体的大小；决定如何产生后继群体的参数，即每一代群体个数中被淘汰的比例和变异率。在这个算法的每一次迭代中，基于当代的群体产生下一代假设。从父代的群体中选择一定数量的假设包含在下一代中。由概率算法选择假设，其中假设选择 h_i 的概率是通过下面的式子计算的：

$$\Pr(h_i) = \frac{Fitness(h_i)}{\sum_{j=1}^{QuaPop} Fitness(h_j)}$$

$$P(h_i) = \sum_{j=1}^{i} \Pr(h_j) = \frac{\sum_{j=1}^{i} Fitness(h_j)}{\sum_{j=1}^{QuaPop} Fitness(h_j)}$$

GA(Fitness, Fitness_threshold, p, r, m)

Fitness:适应度评分函数，为给定假设赋予一个评价得分

Fitness_threshold:指定终止判据的阀值

p:群体中包含的群体数目

r:每一步中通过交叉取代群体成员的比例

m:变异率

- 初始化群体：P←随机产生的p个假设
- 评估:对于P中的每一个h，计算Fitness(h)
- 当[MAXFitness(h)]<Fitness_threshold,做：

产生新的一代Ps:

1.选择：用概率方法选择P的（1-r）p个成员加入Ps。从P中选择假设h_i的概率$\Pr(h_i)$

2.交叉：根据给出的$\Pr(h_i)$,从P中按概率选择r•p/2对假设。对于每对假设<h_1 h_2>,应用交叉算子产生两个后代。把所有的后代加入Ps

3.变异:使用均匀的概率从Ps中选择m%的成员。对于选出的每个成员，在它的表示中随机选择一个位取反

4.更新：P←Ps

5.评估：对于P中的每个h计算Fitness(h)

- 从 P 中返回适应度最高的假设

算法4.1 遗传算法原型

4.3.3 遗传算法的基本要素

下面是遗传算法的一些基本要素：

表示假设：编码是遗传算法第一个要解决的问题。先了解一下编码的定义。编码是把一个问题的可行解从其解空间转换到遗传算法所能处理的搜索空间的转换方法。而由遗传算法解空间向问题空间的转换称为解码。编码必须遵守三个规范：（1）完备性：问题空间的所有解在遗传算法空间中的值都有对应；（2）健全性：遗传算法控件中的值在问题空间中都有解对应；（3）非冗余性：两个空间的值满足一一对应关系。De Jong 提出了较为客观明确的编码评估准则，包括两个规则：（1）有意义积木块编码规则：所定编码应易于生成与所求问题相关的短距和低阶的积木块。（2）最小字符集编码规则：所定编码应采用最小字符集以使问题得到自然的表示或描述。常用的编码方式有：二进制编码，格雷码编码，实数编码，和排列编码等等。

遗传算子：遗传算法由算子操作来产生后代，算子对当前选定的群体成员进行重组和变异最常见的算子有：复制，交叉和变异。复制操作：某些群体成员由于适应度高，直接进入下一代。这是根据精英保留策略：即当前种群中适应度最高的个体不参与交叉运算和变异运算，而是用它来替换掉本代群体中经过交叉、变异等遗传操作后所产生的适应度最低的个体。交叉操作：从两个父串中通过选定位产生两个新的子串。每个字串的第i位是从它的某个父串的第i位继承而来。而交叉掩码决定到底继承哪个父串。常见的交叉方法有：单点交叉，两点交叉，均匀交叉。变异操作：子代直接由一个父串决定，父串上直接取一位，然后取反，生成子串。

适应度函数：适应度函数也叫评价函数，用于评价个体的优劣程度，适应度越大，个体越好，反之适应度越小，则个体越差；根据适应度的大小对个体进行选择，以保证适应性能好的个体有更多的机会繁殖后代，使优良特性得以遗传。适应度函数对遗传算法的好坏起决定性作用。一般来讲，适应度函数值应该是非负实数。

假设选择：从群体中挑选适应度高的个体，淘汰适应度低的个体的操作叫选择。选择的目的是把优化的个体直接遗传到下一代或通过配对交叉产生新的个体再遗传到下一代。选择操作是建立在群体中个体的适应度评估基础上的，目前常用的选择算子有以下几种：轮盘赌选择、随机遍历抽样法、局部选择法等等。

4.4 基于遗传算法求解多 agent 规划的最优观察集问题

4.4.1 算法基本思想

我们知道当观察变量数量很小时，求多 agent 规划的最优观察集问题可以使用回溯法搜索最优解，当观察变量数目很大，由于回溯法的算法时间复杂度可以

达到 $0(2^n)$，这时我们就不得不考虑使用近似算法。尤其是面对最优集问题，观察变量的取舍将更加复杂。所以，本文提出了一种基于遗传算法与贪心法相结合的混合算法来求解多 agent 规划的最优观察集问题。使用遗传算法作为算法基本的框架，然后使用观察变量的平均观测花费来引导局部搜索，也就是平均观测花费值小的优先的贪心法思想。对于遗传算法中生成的群体成员，有些个体代表的假设是不可行解，我们使用不可行解修复来改变这个成员解，有些个体虽然是可行解但明显观测过剩，我们使用可行解修正来处理该成员，以此来引导搜索。

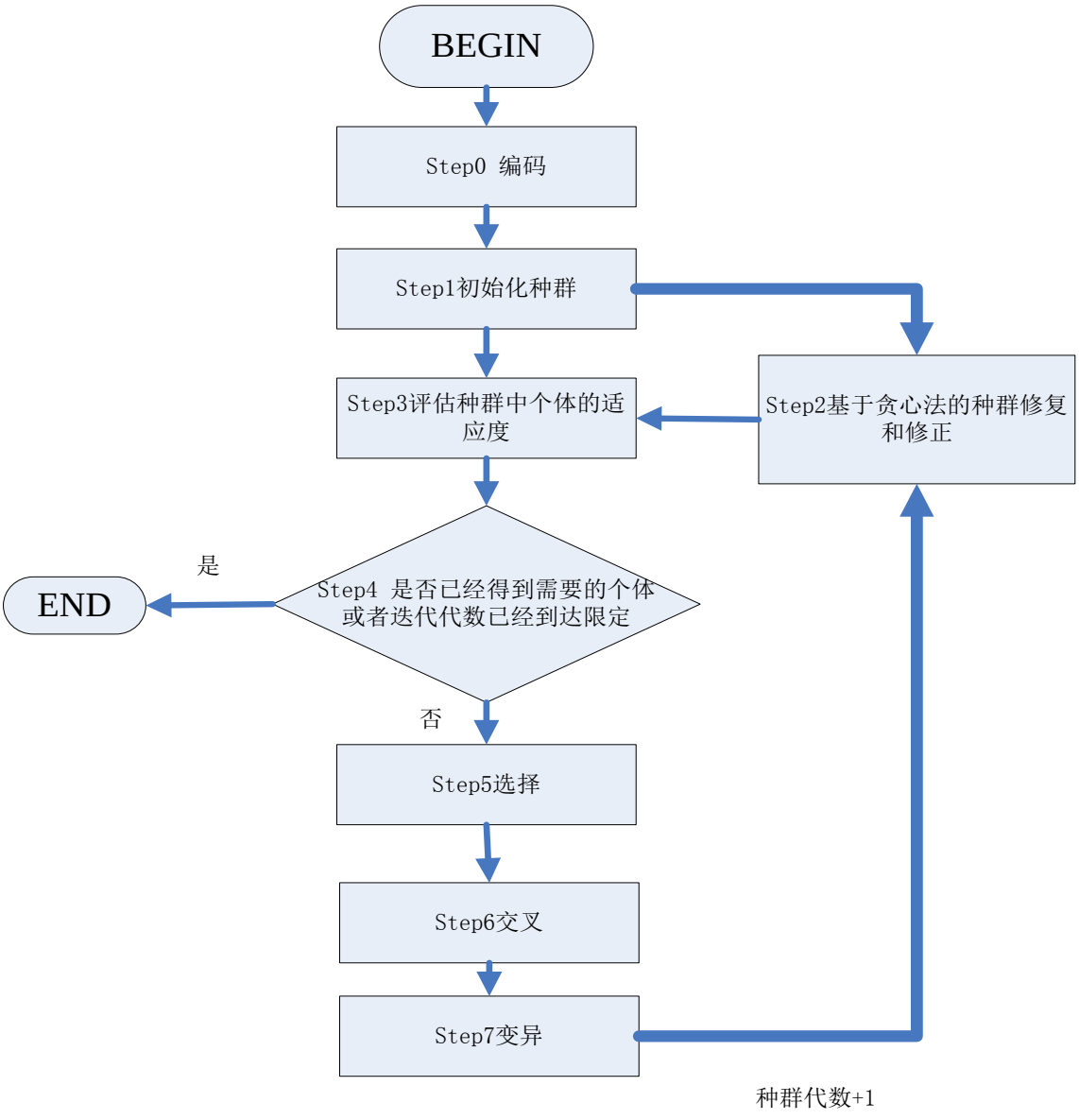

图4.1 算法基本流程图

4.4.2 算法步骤说明

如图 4-1 是混合算法的基本流程。算法开始时输入的参数为可区分矩阵 VD。

VD=$(vd)_{n*m}$ 其中矩阵行数 n 是全局需要区分的状态对数目，矩阵列数 m 是观察变量的数目，矩阵元素表示该观察变量区分该状态对需要的消耗。

Step0:算法采用二进制编码，设多 agent 规划领域的观察变量数目为 m,则种群成员为 $h_i(X_0, X_1, X_2, \ldots, X_{m-1})$。其中 X_i 是一个二进制位码，表示第 i 个观察变量是否选入观测集合中。如果 $X_i=0$ 表示该观察变量未被选入观测集合，反之，$X_i=1$ 表示该观察变量选入了观测集合。在图 3-3 所示的规划领域中如果一个观测集合为{X1,Y2,WallE},则该成员的编码为：10000010001000。

Step1: 设种群数量为 QuaPop，然后随机产生 QuaPop 个种群个体$(P_1,P_2,\ldots,P_{QuaPop})$。由于随机产生的种群个体可能会有不可行解，所以算法通过 Step2 进行不可行解搜索和过度观察搜索，以此来保证每个成员都是可行解且不是过度观察。

Step2:如图 3-6，首先定义衡量观察变量的优先的指标观察变量的平均观测花费 AOC(Average Observed Cost)。

$$AOC_i = \frac{\sum_{j=0}^{n-1} vd[j][i]}{count\_if(vd[0,\ldots,n-1][j] > 0)}$$

其中 count_if(vd[0…n-1][j]>0)表示观察变量 v_i 可以区分需要区分的状态对数目，用总花费 $\sum_{j=0}^{n-1} vd[j][i]$ 除以状态对数目就可以得到平均观测花费。显然平均观测值小的观察变量更符合目标，是优秀基因，进化需要朝向这个方向。

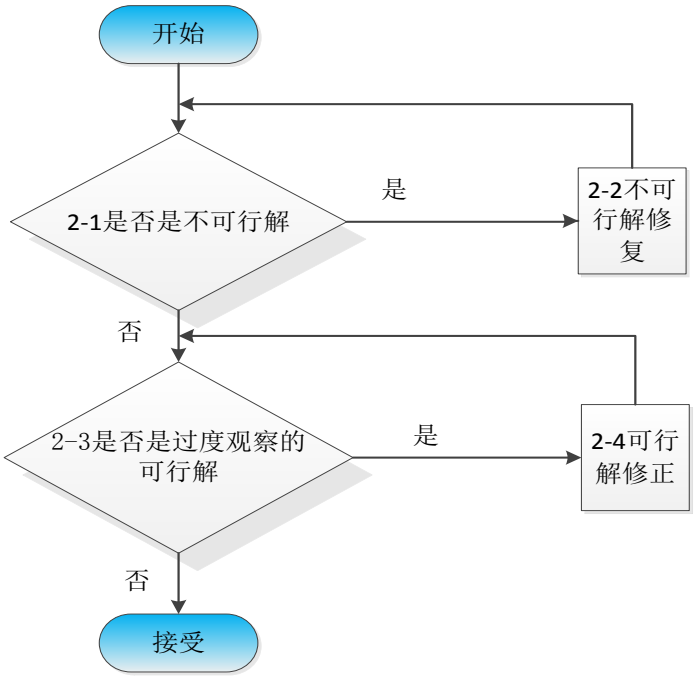

图4.2　step2基本流程图

2-1 由最优观察变量集合定义可知，如果该种群成员是可行解的话，那么他所涵盖的观察变量必须可以区分所有的状态对。可以描述为：

$$\forall i \in \{0,1,2,\ldots,n-1\} such that \sum_{j\in\{j|X_j==1\}} vd[i][j] != 0$$

假如不满足这个条件则该个体是不可行解，必须被修复。

2-2 不可行解修复，对于不可行解，必定不满足上面的条件，所以必定存在一个 i， i∈{0,1,2,…,n-1}而且满足 vd[i][j=0,1,2,…,m-1]的和为 0。找到这个 i，然后查找 j∈{0,1,2,…,n-1}而且满足 vd[i][j]>0。得到所有满足条件的 j。假设所有满足这个条件的 j 的集合为 A。对于所有 i∈A，计算所有 AOC_i。然后将 A 集合里的元素按从 AOC_i 小到大排列。令 min 等于 A 集合的排第一的元素，最后将 P 中的 X_{min} 设为 1。处理完毕，再次检查该成员是否是可行解，如果仍然不是，就再进行一次不可行解修复。如果是，就进入下一步判断。

2-3 判断是否是存在过度观察，过度观察的意思是：如果存在两个观察变量 V_i，V_j。如果 V_i 涵盖的可区分状态对，V_j 也涵盖，而且 $AOC_i<=AOC_j$，那么算法认为 V_i 必定不需要。

2-4 可行解修正，将 X_i 设置为 0。处理完毕后，再次检查是否还存在过度观察，如果没有就接受该成员，反之，则需要再次进行可行解修正。

Step3:计算个体的适宜度。由于目标函数观测总花费是求最小值，所以不宜将目标函数直接作为适应度。将目标函数观测总花费稍作变化，用可区分矩阵 VD 减去观测总花费的值来作为适宜度函数。

$$Fitness(h_p) = \sum_{i=0}^{n-1}\sum_{j=0}^{m-1} vd[i][j] - \sum_{i=0}^{n-1}\sum_{j\in\{0,1,2,\ldots,m-1\}\cap\{j|X_j=1\}} vd[i][j]$$

Step4:算法终止条件，一般是算法终止条件有两个，一个条件是最好的个体的适宜度已经超过了预期获得的适宜度，即超过了阀值；另一个是，当前的进化代数超出了已定的最大进化代数。两者满足其一，就终止算法。

即 Max_Fitness>= Fitness_threshold； gen>=Maxgen。

Step5:选择算子采用轮盘赌选择，同时结合精英保留策略。计算每个个体的选择概率函数 $Pr(h_p)$，然后计算累积概率函数 $P(h_p)$。其中：

$$\Pr(h_i) = \frac{Fitness(h_i)}{\sum_{j=1}^{QuaPop} Fitness(h_j)}$$

$$P(h_i) = \sum_{j=1}^{i} \Pr(h_j) = \frac{\sum_{j=1}^{i} Fitness(h_j)}{\sum_{j=1}^{QuaPop} Fitness(h_j)}$$

然后随机产生一个[0,1]的实数 rand，如果 rand∈($P(h_{i-1}),P(h_i)$)，则表示成员 h_i 被选中。同时记录适应度最大的个体和记录适应度最小的个体，用最优个体取代最劣个体，以此保证最优个体进入下一代和最劣个体无法进入下一代。

Step6：采用均匀交叉，设交叉率为 Pc，摇一个随机数 rand_c∈[0,1],如果 rand_c<Pc，则继续摇数，否则，随机选择两个父串，再随机产生与位串等长的交叉掩码，然后根据父串和交叉掩码产生 两个新的子串。将两个子串修复处理和修正处理后，替代父串。一共摇数 QuaPop/2 次。

Step7：采用均匀变异，遍历每个个体，设变异率为 Pm，摇一个随机数 rand_c∈[0,1],如果 rand_c<Pc，则该个体不变异，否则，随机生成与位串等长的变异掩码，然后根据变异掩码产生新的子串，将子串进行修复处理和修正处理后，替代父串。

4.5 算法分析

4.5.1 遗传算法的收敛性分析

因为算法对不可行解进行了修复工作，所以保证算法结束时，总是会有可行解。我们讨论一下算法的收敛性，遗传算法要保证全局收敛，首先要实现任意初始种群经过有限次进化，总会获得最优解，其次算法必须由确保优者保留来防止最优解丢失。影响算法的收敛性的主要因素包括种群规模，选择操作，交叉概率和变异概率。通常，种群规模太小不足以提供充分的采样点，会导致收敛性差；而种群规模太大，虽然会增加优化机会，阻止早熟收敛，但会增加计算量，造成收敛时间过长，收敛速度缓慢。选择操作时评价高的种群个体比较评价低的种群个体有更大的概率生存，从而提高算法的全局收敛性，如果爱算法中采用精英留存策略，即父代中的优秀个体保留，不参加交叉和变异，直接进入下一代。最终可是遗传算法以概率 1 收敛于全局最优解。交叉操作作用于两个父代个体，产生新的两个子代个体，实际上是在解空间中进行有效搜索。交叉概率过大时，种群中个体更新过快，容易造成高适应度的个体被很快破坏掉；概率过小时，交叉操作进行很少，从而会使搜索停顿不前，造成算法不会收敛。变异操作是对种群模式的扰动，趋向于增肌种群的多样性。然而突变概率太小则很难产生新模式，变异概率太大则会使遗传算法变成随机搜索算法。

4.5.2 遗传算法的时间复杂度分析

遗传算法运算量较大，而且不确定性大。尤其是终止算法的阀值不好确定，

特别对于规模大的组合优化问题，不知道什么时候的解才是最优解。但是，我们可以从种群规模和个体的染色体长度，和种群规模这几个方面来讨论算法时间复杂度。设种群规模为 QuaPop,染色体长度即观察变量数量为 m，进化世代为 gen。初始化阶段：设赋值操作为 t。则 T0=t*QuaPop*m；修复和修正阶段：设访问操作位为 v.则 T1=QuanPop*(m(m-1)/2*v+mlogm)；计算适应度阶段：设计算适应度操作消耗时间 c,则 T2=c*m*QuanPop；轮盘法选择：T3=x*QuaPop；交叉组合：T4=y*m*QuaPop；突变组合：T5=z*m*QuaPop；总时间复杂度为：T=T0+（T1+T2+T3+T4+T5）*gen=O(QuaPop*m*m*gen) 。

4.6 实验与分析

根据伪代码设计实验求解不确定多 agent 规划领域中的最优观察集合，随机生成 m*m 的可区分矩阵，m=50,60,70,80,100,200,400 的矩阵各 10 个。然后分别使用混合遗传算法和贪心算法计算这些矩阵的最优观察集合。然后将不同 m 值的平均参数作比较。随后又将遗传算法不同的参数带入进行比较。

本实验的实验环境为双核 T2300 1.66GHz CPUs，3.00GHz 处理器，2GB 内存，Microsoft Windows XP Professinal 操作系统，Microsoft Visual Studio 2008，算法用 C++语言编程实现。

表 4.1　两种算法的运行时间对比

观察变量数目 m	混合遗传算法时间/ms	混合遗传算法观察信息约简比	贪心算法时间/ms	贪心算法观察信息约简比
50	20.087376	32/50	10.141646	31/50
60	28.502377	40/60	29.247112	39/60
70	41.390125	46/70	67.505488	52/70
80	53.481258	54/80	123.88776	64/80
100	82.627679	61/100	235.33693	75/100
200	321.08314	133/200	1457.9864	158/200
400	1206.3112	257/400	8964.7193	273/400

实验结果如表 4-1，可以分析得到混合遗传算法和贪心法的算法时间都随着观察变量增加而增加，但是贪心法增加的算法时间的复杂程度超过混合遗传算法。同时混合遗传算法观察信息约简的程度远超过贪心算法，显然比贪心算法更接近最优解。

表 4.2　遗传算法与进化世代对比

	1000 代		5000 代		10000 代	
	平均值	最优值	平均值	最优值	平均值	最优值
倒置	108.74	105.79	105.30	102.48	104.22	102.48
交换	158.16	148.06	143.14	133.00	137.79	124.51
插入	247.36	233.15	163.99	141.29	140.88	130.04

表 4.3　遗传算法与变异率

参数: 种群数量:50　进化代数: 3000　交叉算法:OX　选择算法:轮盘赌　变异算法: 倒置							
变异率	0.01	0.1	0.3	0.4	0.5	0.6	0.8
平均值	193.9	109.8	106.08	105.38	106.4	114.08	191.09
最优值	181	105.8	104.97	102.48	104.14	111.34	188.8

表 4.4　遗传算法与交叉率

参数: 种群数量:50　进化代数: 3000　交叉算法:OX　选择算法:轮盘赌　变异算法: 倒置							
交叉率	0.01	0.1	0.3	0.4	0.5	0.6	0.8
平均值	90.28	67.93	66.98	66.89	66.52	67.79	101.38
最优值	83.34	66.48	64.82	64.82	64.82	66.48	97.77

实验结果如图表 4.2 表 4.3 表 4.4 表明当变异率在 0.3--0.5 之间,无论是最优值还是平均值都比较理想.一般遗传算法的变异率在 0.001--0.1 之间,然而实验结果表明0.001-0.1的变异率对于本问题显然不合适.如果变异率太小,收敛速度实在太慢,如果将变异率提高到 0.3--0.5 之间,时间上大大缩短,同时平均值和最优值也比较理想。

4.7 本章小结

本章提出并详细定义了多 agent 规划领域的最小观察集合问题，围绕该矩阵设计了在已知多 agent 规划解的情况下，如何求它的最小观察集合。并分析了该算法的优势。初步探索了信息约减在多 agent 规划的运用。本章在最小观察集的基础上扩展地提出并详细定义了多 agent 规划领域的最优观察集合问题并提出了可区分性矩阵来反应观察变量与需要区分的状态对之间的关系。并提出了带权值的可区分性矩阵来反应观察变量与需要区分的状态对之间的关系。转换了问题。

设计了一种遗传算法与贪心法相结合的混合算法求解问题。并分析了该算法的优势。本章内容是在信息约减在多 agent 规划上的运用近一步的研究。

第 5 章 总结和展望

本文主要探究的是在多 agent 规划领域中如何进行信息约简。通过综合考虑多 agent 规划领域和不定规划的观察信息约减相关内容，本文首先提出了多 agent 规划解的最小观测集问题。针对该问题，本文定义了可区分性矩阵，而且基于可区分性矩阵设计了解决该问题的算法。具体做法是，设计的可区分性矩阵代表状态对与观察函数的对应区分关系，然后通过一系列矩阵化简变换，在得到一个最简的观察变量集合后，回溯搜索，选择含观察变量个数最少的观测集合。在此基础上，本文又扩展性地提出了多 agent 规划解的最优观察集问题，同样也设计带权值的可区分矩阵来表示多 agent 规划解的观察函数集合。本文用结合遗传算法和贪婪法的混合算法来求解这个问题。。

本文的进一步工作主要有：

1. 本文是在完全可观察条件下进行观察信息约减，那么如何在部分可观察条件下进行观察信息约减，可以做进一步的研究；

2. 将观测信息花费作为一个考虑因素，与运动花费，通信花费综合起来，考虑多 agent 规划的最优规划解。

参考文献

[1] D. Weld. Recent advances in AI planning[J]. AI Magazine, 1999, 20(2), pp: 93 ~ 123.

[2] Qiang Yang. Intelligent planning: A decomposition and abstraction based approach[J].Springer, New York, 1997.

[3] Avrim Blum and Merrick Furst. Fast planning through graph analysis[J]. Artificial Intelligence, 1997, pp: 281 ~ 300.

[4] J. Hoffmann. Extending FF to numerical state valiables[C]. In Proceedings of the 15th European Conference on Artificial Intelligence(ECAI-02), Lyon, France, 2002, pp: 571 ~575.

[5] J. Hoffmann. The metric-FF planning system: Translating "Ingoring Delete Lists" to numerical state variables[J]. Journal of Artificial Intelligence Research, 2003, 20, pp: 291 ~341.

[6] H. Kautz, D. Mcallester, and B. Selman. Encoding plans in propositional logic[C]. In Proceedings of the 5th International Conference of Principles of knowledge Representation and Reasoning(KR-96), Boston, MA, 1996, pp: 374 ~ 385.

[7] B. Pell, D. E. Benard, A. S. Chien, E. Gat, N. Muscettola, P. P. Nayak, M. D. Wagner, and B.C. Williams. An autonomous spacecraft agent prototype[C]. In Proceedings of the First International Conference on Autonomous Agents, Marina del Rey, CA, 1997, pp: 253 ~ 261.

[8] D. McAllester and D. Rosenblitt. Systematic nonlinear planning[C]. In Proceedings of the 9th National Conference on Artificial Intelligence(AAAI-91), Anaheim, California, USA, 1991. volume 2, pp: 634 ~ 639.

[9] S. Soderland and S. D. Weld, Evaluating Nonlinear Planning[R], Technical Report 91-02-03, Department of Computer Science and Engineering, University of Washington, 1991.

[10] H. Kautz and B. Sclman. Pushing the envelope: Planning, propositional logic, and stochastic search[C]. In Proceedings of the Thirteenth National Conference On Artificial Intelligence(AAAI-96), Portland, Oregon, 1996, pp: 1194 ~ 1201.

[11] H. Kautz and B. Selman. The role of domain-specific knowledge in the planning as satisfiability framework[C]. In Proceedings of the 4th International Conference on Artificial Intelligence Planning and Scheduling Systems(AIPS-98), Pittsburgh, PA, 1998, pp: 181 ~189.

[12] H. Kautz and B. Selman. BLACKBOX: A New Approach to the Application of Theorem Proving to Problem Solving, Working notes of the Workshop on Planning as Combinatorial Search, AIPS-98, Pittsburg, PA, 1998, pp: 58 ~ 60.

[13] 凌应标. 基于 SAT 的规划理论与算法研究[D]. 中山大学博士毕业论文, 2005.

[14] 吴康恒. 领域模型在智能规划中的学习及应用[D]. 中山大学博士毕业论文, 2006.

[15] 英国诺丁汉大学 ASAP 研究组. Automated Scheduling Optimisation and Planning[EB/OL].http://www.asap.cs.nott.ac.uk/.

[16] 美国亚利桑那州立人学 Yochan 研究组[EB/OL]. http://rakaposhi.eas.asu.edu/yochan.html/.

[17] 陈建林. 强规划解、弱规划解的研究[D]. 湘潭大学硕士毕业论文, 2011.

[18] S Chien, G Rabideau, R Knight, et al. ASPEN-Automating Space Mission Operations using Automated Planning and Scheduling[C]// International Conference on Space Operations (SpaceOps 2000). Toulouse, France, 2000, (7), pp: 200～210.

[19] D. Weld. Recent advances in AI planning. AI Magazine, 1999, 20(2), pp: 93～123.

[20] Wei Huang, Hong Peng. Observation Reduction for State-action Tables[C]. In Proceedings of the International Conference on Computational Intelligence and Security, Beijing, 2009: 10 ~ 14.

[21] Wei Huang, ZhongHua Wen, YunFei Jiang and Hong Peng.Structured Plans and Observation Reduction for Plans with Context[C]. In Proceeding of 21th International Joint Conference on Artificial Intelligence(IJCAI 09), Pasadena, 2009:1721~1727

[22] 饶东宁, 蒋志华, 姜云飞, 朱慧泉. 对不确定规划中观察约简的进一步研究[J].软件学报, 2009,20(5): 1254 ~ 1268.

[23] 周俊萍, 殷明浩, 谷文祥, 孙吉贵. 部分可观察强规划中约减观察变量的研究[J].软件学报, 2009,20(2): 290 ~ 304.

[24] 常青, 文中华, 胡雨隆, 陈建林.强循环规划的观察信息约减[J].计算机工程与应用, 2012,48(2): 148-150.

[25] P. Bertoli, A. Cimatti, M. Pistore, M. Roveri, and P. Traverso. (2001). MBP: A Model Based Planner. In Proceedings of the 17th International Conference on Artificial Intelligence (IJCAI-01) workshop on Planning under Uncertainty and Incomplete Information, Seattle, USA, 2001, pp: 93～97.

[26] Stefan Edelkamp. Symbolic Pattern Databases in Heuristic Search Planning, In Proceedings of the Sixth International Conference on Artificial Intelligence Planning Systems (AIPS-02), Toulouse, France, 2002, pp: 274～283.

[27] Stefan Edelkamp and Malte Helmert. The model checking integrated planning system(MIPS). AI Magazine, 2001, 22(3), pp: 67～71.

[28] P. Doherty, J. Gustafsson, L. Karlsson and J. Kvarnstrom. Temporal Action Logics: Language Specification and Tutorial. Electronic Transactions on Artificial Intelligence, 1998, 2(3), pp: 273～306.

[29] L. Karlsson, J. Gustafsson, and P. Doherly. Delayed Effects of Actions. In Proceedings of the 13th European Conference on Artificial Intelligence(ECAI-98), Brighton, UK, 1998, pp:

542～546.

[30] Jonas Kvarnstrom and Patrick Doherty. TALplanner: A temporal logic based forward chaining planner. Annals of Mathematics and Artificial Intelligence, 2000, 30, pp: 119～169.

[31] F. Bacchus and F. Kabanza. Using Temporal Logic to Control Search in a Forward Chaining Planner. New Direction in Planning. M. Ghallab and A. Milani(Eds), IOS Press, 1996, pp: 141～153.

[32] F Bacchus and F. Kabanza. Planning for Temporally Extended Goals. Annals of Mathematics and Artificial Intelligence, 1998, vol. 22, pp: 5～27.

[33] F. Bacchus and F. Kabanza. Using temporal logics to express search control knowledge for planning. Artificial Intelligence. 2000, 116, pp: 123～191.

[34] M. Ghallab, D. Nau, P. Traverso. Automated Planning Theory and Practice[M]. California: Morgan Kaufmann Publishers, 2004.

[35] 文中华, 黄巍, 刘任任, 姜云飞. 模型检测规划中的状态之间的可达关系研究, 计算机学报, Vol. 35, No. 8, 2012, pp: 1634～1643.

[36] P. Bertoli, A. Cimatti, M. Roveri, and P. traverse. Strong planning under partial observability[J]. Artificial Intelligence, 2006, 170, pp: 337～384.

[37] P. Bertoli, A. Cimatti, M. Roveri, and P. Traverso. Planning in nondeterministic domains under partial observability via symbolic model checking[C]. In Proceedings of the 17th International Joint Conference on Artificial Intelligence(IJCAI-01). California, Morgan Kaufmann Publishers, 2001, pp: 473～478.

[38] 陈建林, 文中华, 朱江, 常青. 正向搜索方法求强规划解[J]. 计算机工程与应用. Vol. 47, No. 6, 2011, pp: 52-54.

[39] Huang W, Wen ZH, Jiang YF, Peng H. Structured Plans and Observation Reduction for Plans with Contexts[C]//Proc.of IJCAI'09. AAAI Press, 2009: 1721 – 1727.

[40] Huang W, Wen ZH, Jiang YF, Wu LH. Observation reduction for strong plans[C]//Proc. of IJCAI'07. Hyderabad, India: AAAI Press, 2007: 1930 − 1935.

[41] F. Pecora, A. Cesta. Planning and scheduing ingredients for a multi-agent sysytem[C]. In Proceedings of the 4th International Conference on MultiAgent Systems(ICMAS-2002). Rome, 2002.

[42] J. Dix, H. Munoz-Avila, D. S. Nua and L. Zhang. IMPACTing SHOP: Putting and AI planner into a multi-agent environment[J]. Annals of Mathematics and AI. Vol. 37, No. 4, 2003, pp: 381-407.

[43] E. Duffee. Scaling up agent coordination strategies[J]. IEEE Computer. Vol. 34,No. 7, 2001, pp: 39-46.

[44] T. Standley. Finding Optimal Solutions to the Multi-Agent Pathfinding Problem Using Heuristic Search[C]. In Proceedings of the 2010 AAAI, 2010, pp: 173 ~ 178.

[45] R. Jansen, N. Sturtevant. A new approach to cooperative pathfinding[C]. In Proceedings of the 2008 AAMAS, 2008, pp: 1401 ~ 1404.

[46] Y. Shoham, M. Tennenholtz. On social laws for artificial agent societies: off-line design[J]. Artificial Intelligence, 1995, Vol. 73, No. 4, pp: 231-252

[47] R. B. Larbi, S. Konieczny and P. Marquis. Extending Classical Planning to the Multi-agent Case: A Game-theoretic Approach[C]. In Proceedings of the ECSQARU-07, 2007, pp: 731 ~ 742.

[48] Ko-Hsin. C. Wang, A. Botea. Fast and Memory-Efficient Multi-Agent Pathfinding[C]. In Proceedings of the ICAPS-08, 2008, pp: 380 ~ 387.

[49] W. Huang, D. Zhang, Y. Zhang et al. Bargain over Joint Plans[C]. In Proceedings of the PRICAI-10, 2010, pp: 608 ~ 613.

[50] M.Ghallab, D.Nau, P.TRAVERSO. Automated planning theory and practice [M] . San Francisco : Morgan Kaufmann Publishers,2003.

[51] 饶东宁,蒋志华,姜云飞.多 agent 规划综述[J]. 计算机应用研究,2011,28(3):801-804

[52] Trevor Standley. Finding Optimal Solutions to the Multi-Agent Pathfinding Problem Using Heuristic Search[C]. In Proceedings of the 2010 AAAI, Atlanta, 2010: 173 ~ 178.

[53] Silver D. Cooperative pathfinding[C].The 1st Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE'05). 2005: 23-28.

[54] Renee Jansen, Nathan Sturtevant. A new approach to cooperative pathfinding[C]. In Proceedings of the 2008 AAMAS, Estoril, 2008: 1401 ~ 1404.

[55] Ramzi Ben Larbi, S ́ebastien Konieczny, and Pierre Marquis. Extending Classical Planning to the Multi-agent Case: A Game-theoretic Approach[C]. In Proceedings of the ECSQARU-07, Hammamet, 2007: 731 ~ 742.

[56] Yoav Shoham, Moshe Tennenholtz.On social laws for artificial agent societies:off-line design[J].Journal of Artificial Intelligence 2003(73): 231-252

[57] Wei Huang, Zhonghua Wen, Yunfei Jiang, and Lihua Wu. Observation reduction for strong plans[C]. In Proceedings of the 20th International Joint Conference on Artificial Intelligence(IJCAI–07), Hyderabad, 2007, : 1930 ~ 1935.

[58] 文中华, 黄巍, 刘任任, 姜云飞. 模型检测规划中的状态分层方法[J]. 软件学报, 2009, 20(4):858 – 869

[59] M. Ghallab, D. Nau, P. Traverso. Automated Planning: Therory and Practice[M]. 姜云飞, 杨强, 凌应标等. 北京: 清华大学出版社, 2006, pp: 339-343.

[60] Qiang Yang. Intelligent planning: A decomposition and abstraction based approach. Springer, New York, 1997.

[61] 陈建林，文中华，朱江，常青．正向搜索方法求强规划解[J]．计算机工程与应用. Vol. 47, No. 6, 2011, pp: 52-54.

[62] 林惠民，张文辉．模型检测：理论、方法与应用[J]，电子学报，Vol. 30, No. 12A, 2002, pp: 1907～1912.

致　谢

三年研究生生活很快就过去了，我要感谢很多人，有了他们的帮助，才有了我的成长。首先感谢我的导师——文中华教授，文老师是一位严格要求的好导师，在学习上。文老师悉心指导教会我们怎么去拥有独立寻找问题、分析问题、解决问题的能力。对于学生的科研活动，文老师给予极大的支持和鼓励。而生活上。文老师又像慈父一样关怀着我们。文老师知识渊博，治学严谨，工作负责，却又不慕名利，对于学生来说，他就像高山上的风，令人仰止轻柔拂面；春夜里的雨，谆谆善诱润物无声。三年跟随文老师的时间，让我学到了许多做人和做学问道理，为我确立了终身学习的目标，将以后的长时间里引导我的生活。借此机会，我谨向文老师致以深深地谢意。

读研期三年期间，信息工程学院各位领导和老师在学习上给予我很大帮助与指导，我要感谢高协平教授、刘任任教授、石跃祥教授、段斌教授、黎自强教授、欧阳健权教授、李枚毅教授、戴永教授、郭云飞老师、文获和老师、陈姝老师、杨奇为老师等人，有了他们的关心和帮助，才有我学习上的进步。

感谢我实验室的老师和同学，他们是王求真老师，朱江老师，陈建林师兄，常青师兄，胡雨隆师兄，黄丽芳师姐，吴正成师兄，汪泉师弟，王进宗师弟，唐杰师弟，龙凤师妹，伍小辉师弟，李洋师弟，劳佳琪师弟，陈秋茹师妹，给予了我很多的帮助，我们一起学习，一起讨论课题研究项目，一起生活的这三年必定是令人深刻的三年。

感谢我的家人，他们在生活上给予我无微不至的关怀，始终支持并激励我不断向前。

最后，感谢所有评审老师利用您宝贵的时间来评审我的论文。

附录 A（攻读硕士学位期间发表的论文）

1. 伍选，文中华，汪泉，等. 多 agent 的观察信息约简[J]. 计算机科学(已录用)
2. 汪泉，文中华，伍选，等. 分层法求强循环规划解[J]. 计算机科学，2013, 40(11): 291-294.
3. 汪泉，文中华，伍选. 求强规划解的快速状态分层算法[J]. 计算机工程, 2014, 40(2): 35-38.

附录 B（攻读硕士学位期间参与的科研项目）

1. 不确定规划中的观察信息约简方法及其应用研究，国家自然科学基金项目(项目编号: 61272295)
2. 基于模型检测的不确定规划的状态可达性及其应用研究，国家自然科学基金(项目编号: 61070232)