中国科学技术大学

硕士学位论文

基于马尔可夫决策过程理论的Agent决策问题研究

姓名：石轲

申请学位级别：硕士

专业：计算机应用技术

指导教师：陈小平

20100501

摘 要

人工智能被认为其主要目标是构造可以决策出智能行为的 Agents，即这些 Agents 能够在多方面再现人类可以做出的智能行为。马尔可夫决策过程（MDP）可以用来描述和处理大规模不确定性环境下的 Agent 决策问题。

RoboCup 机器人世界杯是国际上一项为促进分布式人工智能、智能机器人技术及其相关领域的研究与发展而举行的大型比赛和学术活动，RoboCup 仿真 2D 比赛是 RoboCup 所有项目中以 Agent 决策为重点的一个分支。

本文以马尔可夫决策过程的相关理论为基础，以 RoboCup 仿真 2D 比赛为实验平台，对 Agent 决策相关问题进行了研究。本文的主要工作可以概括为以下三个方面：

- 本文重构并实现了一个完整的 RoboCup 仿真 2D 球队决策系统 WE2009。该系统以部分可观察随机博弈（POSG）的模型为理论基础，包括信息处理、高层决策和行为执行三个模块。特别是高层决策模块，采用基于独立行为生成器的结构设计，不仅可以充分利用 Agent 的决策时间，而且可以提高团队合作的效率。
- 本文提出了一类特殊的马尔可夫决策过程，即行动驱动的马尔可夫决策过程（ADMDP）。本文分析了 ADMDP 的理论模型，提出了 ADMDP 的相关求解方法。该方法采取离线值迭代与在线搜索相结合，在本文中用来求解 RoboCup 仿真 2D 比赛中的不离身带球问题，使 Agent 的带球性能有了较大的提高。
- 本文提出了一类特殊的马尔可夫博弈，即基于阵型的零和马尔可夫博弈（FZSMG）。本文分析了 FZSMG 的理论模型，并以此为基础来描述 RoboCup 仿真 2D 比赛中的 Anti-Mark 问题。针对 Anti-Mark 问题，本文提出了一个基于阵型变换的启发式求解方法，使球队在与盯人防守的对手比赛时取得了较好的效果。

本文的所有工作都是基于 WE2009 实现的，WE2009 在完成后参加了 2009RoboCup 机器人世界杯和 2009 中国机器人大赛两次重要比赛，并且全部获得冠军。

关键词： 人工智能　Agent 决策　多 Agent 系统　马尔可夫决策过程　马尔可夫博弈　RoboCup　仿真 2D

ABSTRACT

As most people thought, the goal of Artificial Intelligence is to construct Agents which can make intelligent behaviors, and it also means that these agents will recreate intelligent human behaviors in all respects. Markov Decision Process (MDP) could be used to describe and process Agent decision problems in large size and probabilistic environments.

RoboCup is an international competition and scientific activity to prompt decentralized Artificial Intelligence, intelligent robotics and related fields. The 2D competition of soccer simulation league is a branch of RoboCup which is emphasis on Agent decision problems.

In this dissertation, we have done research on Agent decision problems based on the theory of MDP and the test bed of RoboCup 2D soccer simulation. The three main contributions of this dissertation are as below:

- We design and realize a complete 2D soccer simulation team system which is called WE2009. WE2009 is based on the theory of Partially Observable Stochastic Games (POSG) and consists of three modules: message parser, high level decision and low level actions. The high level decision module which adopts a structure based on independent behavior generator, can not only make use of the decision time sufficiently, but also increase the efficiency of teamwork.
- We propose a special kind of MDP, which is called Action-Driven Markov Decision Process (ADMDP). We analyze the theory model of ADMDP and propose the algorithm for solving ADMDP. This algorithm based on offline value iteration and online search is used for the proximal dribble problem in 2D soccer simulation. The empirical result shows that it is much better than the old algorithm of our team in Agent's dribble performance.
- We propose a special kind Markov Game, which is called Formation-based Zero-Sum Markov Game (FZSMG). We analyze the theory model of FZSMG which is used to describe the Anti-Mark problem in 2D soccer simulation. We propose a new heuristic method based on formation change to solve the Anti-Mark problem, which gets a better performance in the competition with the opponents depending on mark defense system.

All above works are realized in WE2009 2D soccer simulation team. This team has participated RoboCup 2009 and RoboCup China Open 2009 and won two champions!

Key Words: Artificial Intelligence, Agent Decision, Multi-Agent System, Markov Decision Process, Markov Game, RoboCup, 2D Soccer Simulation

图表目录

中国科学技术大学学位论文原创性声明

本人声明所呈交的学位论文，是本人在导师指导下进行研究工作所取得的成果。除已特别加以标注和致谢的地方外，论文中不包含任何他人已经发表或撰写过的研究成果。与我一同工作的同志对本研究所做的贡献均已在论文中作了明确的说明。

作者签名：石轲　　签字日期：2010.5.24

中国科学技术大学学位论文授权使用声明

作为申请学位的条件之一，学位论文著作权拥有者授权中国科学技术大学拥有学位论文的部分使用权，即：学校有权按有关规定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，可以将学位论文编入《中国学位论文全文数据库》等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、汇编学位论文。本人提交的电子文档的内容和纸质论文的内容相一致。

保密的学位论文在解密后也遵守此规定。

☑公开　□保密（___年）

作者签名：石轲　　导师签名：陈小平

签字日期：2010.5.24　　签字日期：2010-5-24

第 1 章 绪论

随着计算机与人工智能技术的发展，Agent 与多 Agent 系统的研究成为当今计算机科学研究的一个热点。人工智能被认为其主要目标是构造可以决策出智能行为的Agents，即这些Agents能够在多方面再现人类可以做出的智能行为。那么，Agent 决策便成为众多研究者关注的重点，也是本文工作开展的主要动机所在。

马尔可夫决策过程可以用来描述和处理大规模不确定性环境下的 Agent 决策问题。本文以马尔可夫决策过程的相关理论为基础，以 RoboCup 仿真 2D 比赛为实验平台，对 Agent 决策相关问题进行了研究。本章主要是对本文工作的研究背景、相关理论和实验平台做一个整体的介绍，以让读者对本文有一个大致的了解。

本章主要包括以下内容：

- 1.1 介绍本文的研究背景，即 Agent 决策在人工智能技术中的意义；
- 1.2 介绍基本马尔可夫决策过程的理论模型；
- 1.3 介绍从基本马尔可夫决策过程扩展出来的实际应用中需要考虑的其他 Agent 决策的相关模型；
- 1.4 介绍本文的实验平台，包括 RoboCup 仿真 2D 比赛和 WrightEagle 仿真 2D 机器人足球队；
- 1.5 概括本文的主要工作和创新之处。

1.1 人工智能与Agent决策

人工智能是计算机学科的一个分支，“人工智能”一词最初是在 1956 年 Dartmouth 学会上提出的。从 20 世纪 50 年代到现在，人工智能获得了巨大的发展，已经引起不同学科和不同领域的越来越多的研究人员的重视，成为一门涉及知识广泛的交叉学科。人工智能研究的主要目标是使机器能够完成一些通常需要人类智能才能完成的复杂工作。不同的时代，不同的人对“复杂工作”的理解是不同的。复杂工作的定义随着时代的发展和技术的进步而变化，人工智能这门学科的具体目标也自然随着时代的变化而发展。它一方面不断获得新的进展，一方面又转向更有意义、更加困难的目标[Stuart, 2003]。

目前能够用来研究人工智能的主要物质手段以及能够实现人工智能技术的机器就是计算机，人工智能的发展是和计算机学科的发展联系在一起的。因此，

可以进一步认为人工智能是研究如何让计算机去完成以往需要人的智力才能胜任的工作，也就是研究如何应用计算机的软硬件来模拟人类某些智能行为的基本理论、方法和技术[Zixing, 2003]。

随着计算机与人工智能技术的发展，Agent 与多 Agent 系统的研究成为当今计算机学科研究的一个热点。Agent 通常指的是一个高度开放的智能系统，如果说智能系统强调了实用，甚至是固定领域内的实用性的话，人工智能最初赋予 Agent 的含义则可以理解为一个更加通用的智能系统。

正如人工智能本身并不存在一个严格的定义一样，Agent 的概念也有诸多不同的版本：Shoham 认为 Agent 是具有包括信念、能力、选择和承诺等精神状态的一个实体[Shoham, 1993]；Lane 认为 Agent 是一个具有控制问题求解机理的计算单元，它可以指一个机器人、一个专家系统、一个过程、一个模块或一个求解单元[Lane, 1994]；Stan 则认为：具有自主性的 Agent 是能够不断感知环境并作用于环境，以完成其计划的一类系统[Stan, 1996]；Wooldridge 给出的定义是：Agent 是封装在一些环境中的计算机系统，为了达到设计好的目标，它能够执行灵活自主的行为[Wooldridge, 1997]。

基于 Agent 的各种定义，人工智能出现了一种新的定义：人工智能被认为其主要目标是构造可以决策出智能行为的 Agents，即这些 Agents 能够在多方面再现人类可以做出的智能行为。这一观点已被人工智能领域的大部分研究者所认同[Wooldridge, 1994]。1994 年，英国著名杂志《New Scientist》曾做出如下的预言："基于 Agent 的计算（Agent-Based Computing）将是软件开发领域下一个重大突破" [Ellen, 1994]。因此，在大规模不确定性环境下，如何描述 Agent，如何让 Agent 进行决策便成为众多研究者关注的重点。

1.2 马尔可夫决策过程

Agent 进行决策总是与一个过程相联系的，Agent 要在过程中做出合适的选择，将过程的发展引入对自身有利的方向，必须要了解描述过程发展变化的知识。相对于穷举所有的变化，如果某些知识不止一次可被用来推断过程发展，就成为规律[Changjie, 2008a]。

马尔可夫过程（Markov Processes）就是具有一类普遍共性的过程。马尔可夫过程的原始模型是马尔可夫链，由俄罗斯数学家 Markov 于 1907 年提出。该过程具有如下特性：某阶段的状态一旦确定，则此后过程的演变不再受此前各状态的影响。也就是说，当前的状态是此前历史的一个完整总结，此前的历史只能通过当前的状态去影响过程未来的改变。在现实世界中，有很多过程都是

马尔可夫过程，如液体中微粒所做的布朗运动、传染病受感染的人数、车站的候车人数等。我们以花丛中一只蜜蜂的采蜜来形象化说明：蜜蜂按照它自己的想法从一朵花跳到另一朵花上，因为蜜蜂是没有记忆的，它处在当前的位置时，下一步跳向哪一朵花和它之前路径无关。如果用 $X_0, X_1, ..., X_n$ 分别表示蜜蜂的初始花朵号码以及第1次至第 n 次的花朵号码，则 $\{X_0, X_1, ..., X_n\}$ 就是马尔可夫过程[Daniel, 2000]。

马尔可夫决策过程（Markov Decision Processes，简称为 MDP）与马尔可夫过程的本质区别就是多了 Agent 即决策者的介入。在人工智能领域中，经典的决策方法一般是基于确定性的环境，如盲目搜索、启发式搜索等，这类方法在现实应用中有很大的局限性。面对现实中的决策问题，Agent 对环境的认知以及自己行动的结果往往带有不确定性，马尔可夫决策过程的模型则可以处理类似的问题[Changjie, 2008a]。20 世纪 50 年代，Bellman 研究动态规划时和 Shapley 研究随机对策时已经出现了马尔可夫决策过程的基本思想。Howard 和 Blackwell 等人的研究工作奠定了马尔可夫决策过程的理论基础[Bellman, 1957]。

1.2.1　基本模型

马尔可夫决策过程是描述Agent与环境之间相互作用的一种模型，如图 1.1 所示。Agent接受环境的状态作为输入，并产生动作作为输出，而这些动作会影响环境的状态。在马尔可夫决策过程的理论框架中，重要的一点是Agent具有完全的感知能力。Agent的行动会对环境产生不确定的影响，但Agent对环境的感知是确定的[Eugene, 2002]。

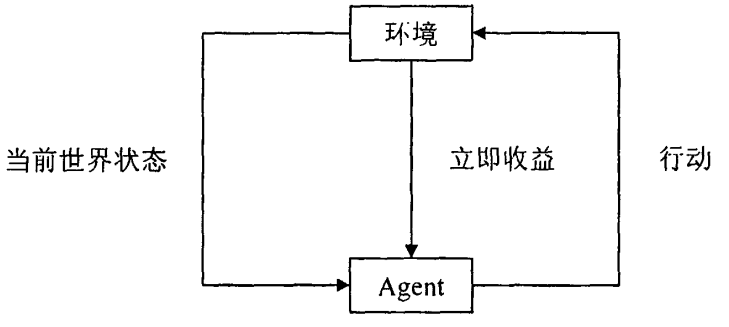

图 1.1　马尔可夫决策过程的基本模型

基本马尔可夫决策过程的模型是一个四元组 $<S, A, T, R>$ [Puterman, 1994]：

- S：表示可能的世界状态的有限集合；

- A：表示可能的行动的有限集合；
- T：是状态转移函数，用$T(s'|s,a)$表示在状态s执行行动a到达状态s'的概率，即$T(s'|s,a)=\Pr\{s_{t+1}=s'|s_t=s,a_t=a\}$；
- R：是立即收益函数，用$R(s,a)$表示 Agent 在状态s执行行动a可以获得的立即收益。

1.2.2 世界状态

世界状态是在某一时间点对该世界或该系统的描述。不同的应用中，人们对状态的具体定义是不一样的，但一般来说，定义的状态必须包括所有当前世界中可以让 Agent 做出决策的信息。最一般化的表示状态的方式是平铺式的表示，即对所有可能的世界状态进行标号，用$s_1,s_2,s_3,\ldots$类似这样的方式表示。这种情况下，标号状态的数目也就代表了状态空间的大小。更加自然的方式是因子化的表示方法，即将每一个状态都看成由多个因子组成的多元组[Changjie, 2008a]。本文的工作也都是基于因子化的状态表示方法。

我们用概率的方法来处理 Agent 对自己所处的当前状态认知的不确定性。随机变量s_t从状态集合S中取值，其并非由未来时刻的状态所决定，而是由过去的状态影响。

图 1.2表示一个离散的、随机的动态系统，图中的每个节点表示在某一时刻的某一状态。连接两个节点的弧，表示前一状态对后一状态有直接的概率影响，随机变量s_t，$\Pr(s_t|s_0,s_1,\ldots,s_{t-1})$为一条件概率。

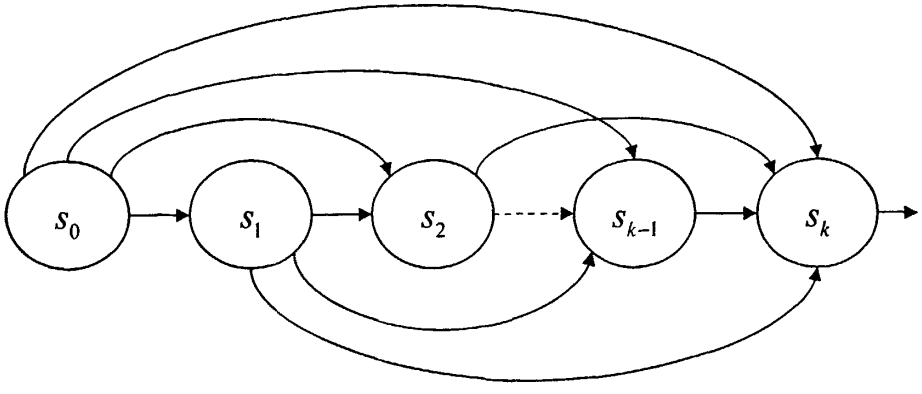

图 1.2 一般随机过程

图 1.3表示一个马尔可夫过程，每一个状态只依赖于它的前一个状态，而

与之前的其他状态都没有关系，即 $\Pr(s_t \mid s_0, s_1, \ldots, s_{t-1}) = \Pr(s_t \mid s_{t-1})$。

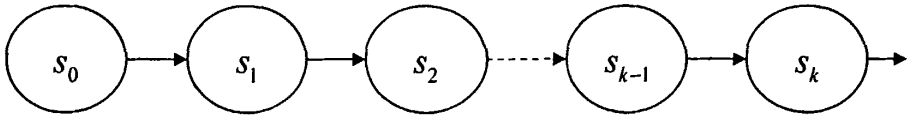

图 1.3　马尔可夫过程

1.2.3　行动

Agent 的行动会改变当前的世界状态，马尔可夫决策过程模型的一个重要部分就是 Agent 用于做决策的行动集合。当某一行动被执行，该状态将会发生改变，随机的转换为另一状态，不过与所执行的行动有关。在每一个时刻 t，都会对应一个状态 s_t 以及一个行动集合 A_{s_t}，在该时刻执行行动 a 后，后继状态的概率分布为 $\Pr(s_{t+1} \mid s_t = s, a_t = a)$。

在马尔可夫决策过程中，任何行动可以在任何状态下执行，但是这个行动可能对某一状态没有影响甚至有负面影响，这种情况就需要在行动转移矩阵中表现出来，这也是和经典的人工智能规划的不同之处。在经典的人工智能规划中，我们需要按照先决条件来决定哪些行动在该种状态下有意义。这样的优点是因为有先决条件许多行动不需要考虑，因此计算时所需考虑的后继状态空间也比较小。在马尔可夫决策过程中，我们同样需要借鉴经典人工智能规划的这一特点，在决策时尽量缩小每一个特定状态下的行动空间。

一般情况下，我们讨论的都是时齐马尔可夫决策过程，即所有行动的执行时间是相同的，状态转移的时间间隔一致，这种行动也可以被称为系统的原子动作。在该系统内，行动已经对应最小的时间划分，原子动作不可再分割。比如在 RoboCup 仿真 2D 比赛中，时间是按照周期不断推进的，每一个周期 Agent 可以转身，可以踢球，也可以奔跑，这些构成了该多 Agent 系统下 Agent 的原子动作集合。

1.2.4　状态转移函数

状态转移函数描述了系统的动态特性，确定环境和随机环境下的行动具有以下的区别：

- 确定环境下的行动：$T: S \times A \to S$，在某个状态 s 执行行动 a 可以得到

一个确定的状态；

- 随机环境下的行动：$T:S\times A\to \Pr(S)$，在某个状态s执行行动a，得到的是一个状态的概率分布，记作$T(s'|s,a)$。

图 1. 4显示了给定某个行动后，状态间的转移情况。在一些简单问题中，状态转移函数可以用表的形式来记录。

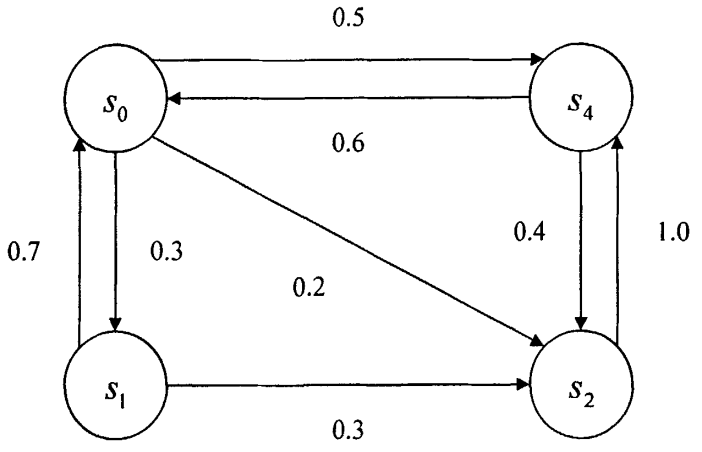

图 1.4 状态间的转移

1.2.5 收益函数

我们希望 Agent 能够按照某个标准来选择动作以使长期收益达到最大化。比如有限阶段的最优准则，要求最大化有限阶段的期望总收益最大，也就是 $\max E[\sum_{t=0}^{k-1} R_t]$，其中 R_t 为 Agent 在第 t 步得到的收益。这种模型往往需要知道 k 的值到底是多少，如果需要得到更理想的结果，就需要考虑整个生存周期的无限阶段。在此，我们考虑的是 Agent 在整个过程中的总收益，为了更贴近实际情况，我们引入了一个折扣因子γ，其中$0<\gamma<1$。这样 Agent 选择动作所得到的总收益就是 $\max E[\sum_{t=0}^{\infty} \gamma^t R_t]$，折扣因子保证了总收益的收敛性[Changjie, 2008a]。

马尔可夫决策过程的解被称为策略，是从状态集合到行动集合的一个映射，即$\pi:S\to A$。按照策略解决问题的过程是：Agent 首先需要知道当前所处的状

态s，然后执行策略对应的行动$\pi(s)$，并进入下一状态，重复此过程直到问题结束。在马尔可夫决策过程的一些材料中对策略有如下区分：如果行动的选取只与当前的状态有关，而与时间无关，则称为平稳策略；如果对于同样的状态，在过程的不同时刻可能会对应不同的行动，则称为非平稳策略，非平稳策略是经时间索引后的一系列状态到行动的映射。

1.2.6 值函数与策略求解

对于任何一个策略，我们都可以采用执行这个策略所能获得的长期期望收益来评价其优劣，定义值函数$V^{\pi}(s)$为 Agent 在状态s采取策略π时的期望收益（s_t表示t时刻的状态）：

$$V^{\pi}(s)=E[\sum_{t=0}^{\infty}R(s_t,\pi(s_t))] \tag{1.1}$$

用递归的方式来表示如下：

$$V^{\pi}(s)=R(s,\pi(s))+\gamma\sum_{s'\in S}T^{\pi(s)}(s,s')V^{\pi}(s') \tag{1.2}$$

对每个策略π，其对应的值函数V^{π}是一系列线性方程的唯一公共解（每个状态s对应一个方程）[Geffner, 1998]。

通过上面的定义，我们可以从某一种策略计算其对应的值函数，下面我们来介绍如何从值函数来计算其对应的策略。我们引入一个在求解过程中经常用到的中间变量，定义行动值函数$Q^{\pi}(s,a)$为 Agent 在状态s采取行动a，其它状态采取策略π时的期望收益：

$$Q^{\pi}(s,a)=R(s,a)+\gamma\sum_{s'\in S}T^{a}(s,s')V^{\pi}(s') \tag{1.3}$$

当策略没有显示记录，只有值函数V时，行动值函数记为Q，策略π可以通过下式计算得到：

$$\pi(s)=\arg\max_{a\in A}Q(s,a) \tag{1.4}$$

其实这里采用的是一步前瞻的贪婪搜索，因此我们也称这样获得的策略为贪婪策略，即：

$$\pi(s)=\arg\max_{a\in A}\{R(s,a)+\gamma\sum_{s'\in S}T^{a}(s,s')V(s')\} \tag{1.5}$$

同时，值函数可以按照下式进行更新：

$$V^{\pi}(s)=\max_{a\in A}\{R(s,\pi(s))+\gamma\sum_{s'\in S}T^{a}(s,s')V(s')\} \tag{1.6}$$

我们将最优策略记为π^{*}，其对应的最优值函数记为V^{*}。如果一个策略π满足对状态s，有$V^{*}(s)-V^{\pi}(s)\le\varepsilon$时，我们称$\pi$为状态$s$处的$\varepsilon$最优策略；如果策

略π对所有状态均满足上述条件，则称π为该问题的ε最优策略。

1.3 其他决策模型介绍

基于上面介绍的基本马尔可夫决策过程的模型，结合实际应用，我们来介绍几个扩展的决策模型，分别是：部分可观察马尔可夫决策过程（Partially Observable Markov Decision Processes，简称为POMDP），分布式马尔可夫决策过程（Decentralized-MDP，简称为Dec-MDP），分布式部分可观察马尔可夫决策过程（Decentralized-POMDP，简称为Dec-POMDP），部分可观察随机博弈（Partially Observable Stochastic Games，简称为POSG）[Changjie, 2008a]。这几个决策模型的关系如图 1.5所示。

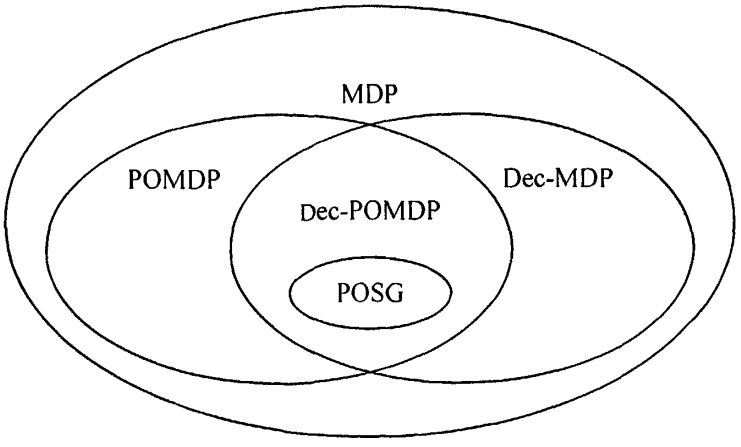

图 1.5 决策模型的关系图

POMDP是相对于MDP的完全可观察而言的，但在MDP中引入部分可观察并不是一个简单的添加过程。在MDP的求解过程中给出了每个状态上的值或者策略，使用MDP求解的时候，需要在任意时刻知道系统的状态，并且是完全可观察的状态。部分可观察使我们对当前状态的感知变得不确定，这使得我们在选择行动时困难增大。POMDP的模型中同样包含有一个状态集合，一个行动集合，状态转移函数和立即收益函数，唯一不同的是在POMDP的基本模型中引入了观察集合。Agent通过观察得到的一些信息来判断自己当前所处的状态。由于观察可能是概率的，因此我们需要描述观察模型，即该观察模型可以告诉我们模型中每个状态下得到每个观察的概率。POMDP对于表达只能在非确定环境下

的Agent决策问题是非常有效的模型。在每一时刻，Agent接受与环境状态相关的一些随机观测结果，基于这些信息，Agent执行一系列动作而随机的改变世界状态。图 1.6是POMDP的基本模型。

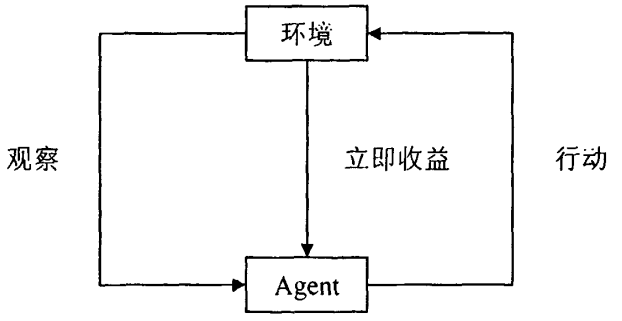

图 1.6 部分可观察马尔可夫决策过程的基本模型

Dec-MDP 是用来处理多 Agent 的决策问题。MDP 模型以及 POMDP 模型中都认为决策的 Agent 只有一个，并把其他一切因素都归于客观环境。这些因素中一部分是确定性的知识；另一部分则已经归入统计概率的不确定性，即认为在当前条件下，从处理问题的实际情况出发，不再适合进行探究，只做概率推理。当一个系统中存在多个 Agent 来共同完成一项任务时，MDP 模型或 POMDP 模型是否还能够适用的关键是对于一个特定的 Agent 来说，其他 Agent 的策略是否已知。如果认为其他 Agent 的策略已知，无论是确定性的策略还是用概率表示的不确定性的策略，其他 Agent 都可以归为环境的一部分，仍然可以使用 MDP 模型或者 POMDP 模型来处理；反之，其他 Agent 在当前状态会如何决策也是需要考虑的，Agent 在生成自身策略的同时，也要生成其他 Agent 的策略，这是客观过程本身的模型决定的。Dec-MDP 就是用来处理这类多 Agent 的合作决策问题的。如果环境是部分可观察的，相应的则可以用 Dec-POMDP 来处理[Changjie, 2008a]。

POSG 是 Dec-POMDP 进一步扩展的决策模型，用来处理系统中既有合作又有对抗的多 Agent 问题。在现实应用中，多 Agent 之间在共同合作完成一个任务的同时可能会遇到其他 Agent 的对抗，即我们通常所说的博弈。对于合作的问题，各个 Agent 之间具有相同的收益评价，即他们有着共同的目标；对于博弈的问题，双方 Agent 之间的收益出现差别，甚至可能是完全对立的。本文工作的实验平台 RoboCup 仿真 2D 比赛环境就是一个典型的既存在合作又存在对抗的多 Agent 系统，将在下一节具体介绍。

1.4 实验平台

1.4.1 RoboCup机器人世界杯

RoboCup 机器人世界杯是国际上一项为促进分布式人工智能、智能机器人技术及其相关领域的研究与发展而举行的大型比赛和学术活动。它通过提供一个标准的比赛平台来检验各种智能机器人技术。RoboCup 的最终目标是：在 2050 年组建一支机器人足球队，去打败人类足球世界杯的冠军。回顾人类的发展历史：第一次工业革命，蒸汽机的发明和使用使人类进入“蒸汽时代”；第二次工业革命，电力的发明和广泛应用使人类由“蒸汽时代”跨入“电气时代”；第三次工业革命，原子能、航天技术、电子信息、生物工程和合成材料等领域取得的突出成就使人类进入信息社会，知识经济时代到来。展望未来：人工智能技术的发展将使人类逐渐从脑力劳动中解放出来，而机器人则是所有这些技术的一个综合应用。作为 RoboCup 的最终目标，也作为一项极其艰巨的挑战，人类能否再次完成这一新的里程碑式的任务，将具有深远的意义[Ke, 2007]。

机器人足球赛的最初想法由加拿大不列颠哥伦比亚大学的 Alan Mackworth 教授于 1992 年正式提出。日本学者立即对这一想法进行了系统的调研和可行性分析。1993 年，Minoru Asada、Hiroaki Kitano 和 Yasuo Kuniyoshi 等著名学者创办了 RoboCup 机器人世界杯，RoboCup 是 Robot World Cup 的简称。与此同时，一些研究人员开始将机器人足球作为研究课题。日本政府的电子技术实验室的 Itsuki Noda 教授以机器人足球为背景展开多 Agent 系统的研究，美国卡耐基梅隆大学的 Veloso 教授等也开展了同类工作。1997 年，在国际最权威的人工智能系列学术大会“第 15 届国际人工智能联合大会”（The 15th International Joint Conference on Artificial Intelligence）上，机器人足球被正式列为人工智能的一项挑战。至此，机器人足球成为人工智能和机器人学相关领域新的标准问题[Kitano, 1997]。

开展机器人足球的研究是人工智能从基础理论走向实际应用的一个战略性步骤，将机器人足球作为未来人工智能和机器人领域的标准问题是十分恰当的，主要是由于机器人足球具有以下特点：

- 典型性，RoboCup 机器人足球队的研究和开发涉及当前人工智能研究的大多数热点，因而构成一个典型问题；
- 可行性，多数的多 Agent 系统背景十分复杂，以致研究人员在目前的条件下难以把握，无法兼顾具体细节分析与基本问题探索，而在机器人中则较易兼顾二者，易于深入；

- 客观性，机器人足球比赛提供了一种实验平台和评价各种理论与技术的客观方法，便于研究者的“观察”和相互交流；
- 综合性，在以往的研究中，各种技术通常被分别开发和考察，综合集成工作一般由面向最终用户的应用部门来完成，这种方式不利于相关技术在更高层次上的衔接和在更深层次上的创新，而机器人足球是一个深层次的“综合平台”。

1.4.2 RoboCup仿真2D机器人足球比赛

RoboCup仿真2D机器人足球比赛是RoboCup所有比赛中参加人数最多的子项目，仅需要几台计算机就可以开展相关的研究工作，全部活动由计算机模拟完成。由于避免了现实物理环境和当前机器人制造技术的限制，RoboCup仿真2D比赛主要把研究重点放在球队高层决策的研究上，包括动态不确定环境中的多Agent合作、实时推理与决策、机器学习和策略获取等当前人工智能的热点问题。比赛由组委会提供了一个标准的比赛软件平台Server，平台设计充分体现了控制、通讯、传感和人体机能等方面的实际限制，使仿真球队程序易于转化为硬件球队的控制软件，我们将在第2章详细介绍Server的结构特点。

参加RoboCup仿真2D比赛的球队在决策时遇到的难点可以归纳为以下几点[Mao, 2003]：

- 问题复杂，在仿真2D比赛中，如果对于场上22名球员的位置和速度、球的位置和速度等特征完全描述，状态空间极其巨大，参赛球队必须考虑如何合理的描述状态并求解决策问题；
- 信息不完全，仿真2D比赛中的球员并不能完全了解场上的所有信息，平台限制了球员获取信息的途径，每一个球员都必须依赖自身获得的有限信息进行决策，参赛球队必须考虑如何在部分可观察的情况下进行合理的决策；
- 决策的实时性，仿真2D比赛的平台提供了一个实时的环境，环境可能发生不可预期的改变，将使得原有的决策不再适用，在这种情况下，Agent必须能够根据场上的情况变化，及时做出反应，保证决策的实时高效；
- 通讯带宽有限且不可靠，仿真2D比赛的平台对多Agent之间的通讯给与了一定的限制，在有限带宽且不可靠的通讯上，参赛球队必须考虑如何保证合作的顺利；
- 多Agent的合作与对抗，仿真2D比赛存在多个独立决策的Agent，同一个球队的Agent之间是合作的关系，不同球队的Agent之间是对抗

的关系，如何使球员间协商、规划以实现合作完成任务并在对抗中取得最大的收益，这是一个重要的研究课题。

1.4.3 WrightEagle仿真 2D机器人足球队

WrightEagle机器人团队于 1998 年建立，隶属于中国科学技术大学计算机科学与技术学院多智能体系统实验室[1]。WrightEagle仿真 2D机器人足球队[2]是WrightEagle机器人团队的第一个分支，从 1999 年开始参加每年的RoboCup机器人世界杯比赛。在近 5 年的世界杯比赛中，WrightEagle仿真 2D球队是世界上唯一一支始终保持世界杯仿真 2D比赛前两名的球队，分别获得两次世界冠军（RoboCup2006 和RoboCup2009）和三次世界亚军（RoboCup2005、RoboCup2007和RoboCup2008）[Aijun, 2010]。

WrightEagle 仿真 2D 球队的目标是在备战世界杯比赛的过程中研究多 Agent 系统中 Agent 如何更好的进行决策，以及其他具有挑战性的相关问题。我们始终坚持自己首创和长期倡导的攻势足球特色，以决策论为理论基础，在 Agent 决策领域开展相关研究。本文的所有工作都是在 WrightEagle 仿真 2D 球队上进行实验的，第 2 章将详细介绍球队的具体结构。

1.5 本文的主要工作及章节安排

本文以马尔可夫决策过程的相关理论为基础，以 RoboCup 仿真 2D 比赛为实验平台，对 Agent 决策相关问题进行了研究。本文的工作在 WrightEagle 仿真 2D 球队中进行了实验，并取得了较好的实验结果。本文一共分为 5 章，包括三方面的主要工作，分别在第 2 章、第 3 章和第 4 章介绍。

第 1 章介绍本文的研究背景和马尔可夫决策过程等 Agent 决策模型，并介绍本文的实验平台，包括 RoboCup 仿真 2D 机器人足球比赛和 WrightEagle 仿真 2D 机器人足球队。

第 2 章介绍本文重构并实现的一个完整的 RoboCup 仿真 2D 球队决策系统 WE2009。该系统以部分可观察随机博弈的模型为理论基础，包括信息处理、高层决策和行为执行三个模块。特别是高层决策模块，采用基于独立行为生成器的结构设计，不仅可以充分利用 Agent 的决策时间，而且可以提高团队合作的效率。

第 3 章介绍本文提出的一类特殊的马尔可夫决策过程，即行动驱动的马尔

[1] http://ai.ustc.edu.cn/
[2] http://wrighteagle.org/2d/

可夫决策过程（ADMDP）。本文分析了 ADMDP 的理论模型，提出了 ADMDP 的相关求解方法。本文所提方法采取离线值迭代和在线搜索相结合，用来求解 RoboCup 仿真 2D 比赛中的不离身带球问题。实验结果表明，本文提出的方法使 Agent 的带球性能有了较大的提高。

第 4 章介绍本文提出的一类特殊的马尔可夫博弈，即基于阵型的零和马尔可夫博弈（FZSMG）。本文分析了 FZSMG 的理论模型，并以此为基础来描述 RoboCup 仿真 2D 比赛中的 Anti-Mark 问题。针对 Anti-Mark 问题，本文提出了基于阵型变换的启发式求解方法，使球队在与盯人防守的对手比赛时取得了较好的效果。

第 5 章对本文的所有工作进行总结，并展望未来的工作。

第 2 章　WE2009 仿真 2D球队决策系统

WrightEagle仿真 2D球队经过多年不同届成员的开发与维护，以及仿真 2D 平台Server的不断发展，到 2008 年时继续在原有的球队上开发并参加比赛已经比较困难。因此，本文的一个基础性工作就是在WE2008 的基础上，重构并实现了一个完整的仿真 2D球队决策系统WE2009[1]。WE2009 以部分可观察随机博弈的模型为理论基础，包括信息处理、高层决策和行为执行三个模块。特别是高层决策模块，采用基于独立行为生成器的结构设计，不仅可以充分利用Agent 的决策时间，而且可以提高团队合作的效率。本文的另外两个主要工作都是在 WE2009 上实现的，因此本章是后面两章内容的基础。

本章主要包括以下内容：

- 2.1 介绍 RoboCup 仿真 2D 平台 Server 的结构特点，这是 WE2009 架构时在需求方面所要考虑的主要因素；
- 2.2 介绍 POSG 的理论模型以及 WE2009 如何抽象建模为一个 POSG 的 Agent 决策系统；
- 2.3 对 WE2009 的主要模块进行分析，并重点介绍高层决策模块采用的基于独立行为生成器的结构设计；
- 2.4 进行本章小结。

2.1　RoboCup仿真 2D平台

RoboCup 仿真 2D 平台是一套能够让由不同语言编写的自主球员程序进行仿真足球比赛的系统。比赛的执行采用的是服务器/客户端（Server/Client）模式：Server 端程序提供了一个虚拟场地并且模拟包括球和球员在内的所有物体移动；每个 Client 端程序相当于一个球员大脑，控制场上该球员的移动。Server 端和 Client 端之间都是通过 UDP/IP 协议进行信息交互的，也就是说，开发者可以使用任何支持 UDP/IP 协议的程序设计语言来设计球队程序。通过 UDP/IP 协议，Client 端程序可以发送指令去控制相应的场上球员，而 Server 端则按照规则给每个客户端发送它所能获得的信息。每个 Client 端程序只允许控制一名球员，所以比赛双方的球队必须同时运行与比赛球员数目相等的 Client 端程序，包括 11 个普通 Agent 程序和 1 个 Coach 程序。Client 端程序之间的通讯必须通

[1] WE2009 仿真 2D 球队决策系统是本文作者作为 WrightEagle 仿真 2D 球队队长期间主持开发的，团队主要开发成员还包括柏爱俊和台运方等。

过 Server 端来进行转发，任何不经 Server 端转发而是 Client 端直接通讯的行为都是违反规则的，这一点也充分体现了多 Agent 系统中分布式控制的特点。当一场比赛开始时，双方 12 个独立的 Client 端程序连接到 Server 端进行比赛，每个队的目标就是将球踢进对方的球门同时阻止球进入自己的球门。

2.1.1　Server端

Server端程序由RoboCup官方发布并持续更新，每个参赛球队都可以免费从互联网获得[1]。Server端程序主要由球场仿真模块、裁判模块和消息板模块三部分组成，如图 2.1所示。

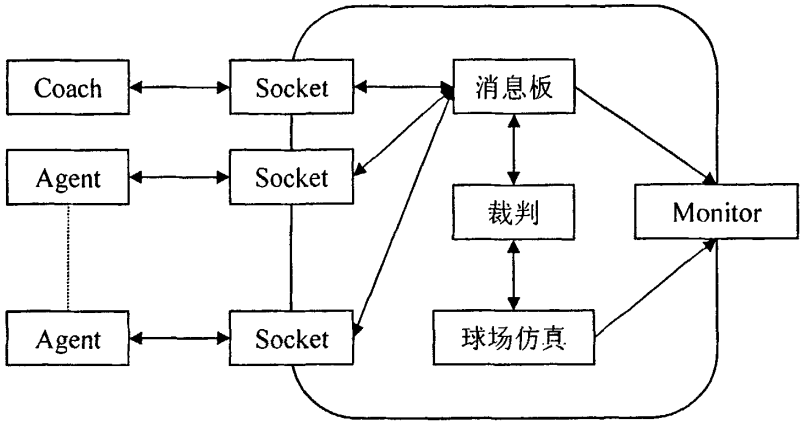

图 2.1　Server 结构图

球场仿真模块负责计算球场上对象的运动，检测他们之间是否碰撞等。球场上的对象包括场上两支球队的 22 名球员、球、球门、标记和标志线等，Coach 不出现在球场上。球员和球都具有大小、位置、速度和加速度等属性，球员还有身体方向和体力等属性。在每个周期末，该模块将会根据动力学定律更新所有对象的属性，如果对象之间发生重叠，则按照规则进行碰撞处理。

裁判模块根据与人类足球相似的比赛规则来控制比赛的进程。仿真 2D 比赛环境具有动态、实时、不确定的特点，比赛不可能按照事先的设计按部就班的进行，因此需要一个带有“智能”的裁判。裁判用来检测比赛中的一些简单情形，比如进球、界外球、越位和犯规等。但是，由于仿真 2D 比赛不存在第三维高度坐标，因此还需要人为裁判来检测一些影响公平竞赛的行为，比如所

[1] http://sourceforge.net/projects/sserver/files/

有球员堵住球门等。

消息板模块负责 Server 端和 Client 端之间的通讯。Server 采用离散化的模式运行，即所有程序运行都以仿真周期为单位，1 个周期为 100 毫秒。在每个周期开始时，Server 根据当前场上各个球员的状态，向球员发送特定格式的信息，包括自身感知信息、视觉信息和听觉信息，由于每个球员的场上状态不同，他们收到的信息也就不同，这也体现了仿真 2D 平台部分可观察的特点；在每个周期结束前，Server 统一执行所有球员发来的动作指令，并更新场上所有球员的状态，然后进入下一周期。对于一些互斥的动作，比如转身、奔跑和踢球等，每个球员在每个周期只能发送其中的一个动作指令，如果多于一个，Server 将只会执行第一个。同时，如果球员在一个周期内没有发送动作指令，它将失去该周期的行动机会，这在一个实时对抗的环境中是非常不利的。

仿真 2D比赛的所有情况都可以通过Monitor程序显示在电脑屏幕上，它直接与Server进行通讯，可以供球队开发者更方便的观看正在进行的比赛以及回放比赛场景，如图 2.2所示。

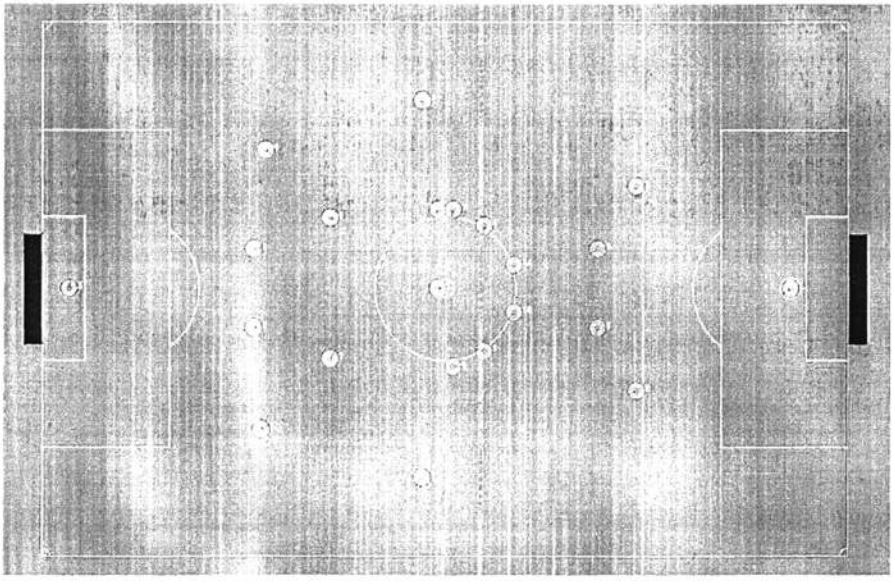

图 2.2　RoboCup 仿真 2D 比赛场景

2.1.2　Client端

Client 端程序通过 UDP 接口连接到 Server 端程序，通过这个接口，Client 端程序可以发送动作指令去控制场上的球员如何行动并接受 Server 发来的各种信息。简单的说，一个 Client 端程序就相当于球员的大脑，从 Server 端获取信

息，并发送动作指令到 Server 端。

每个球员程序都是独立的进程，通过各自不同的端口与 Server 端程序连接。当 Client 端程序在开场之前与 Server 端程序建立好连接后，所有通讯信息都通过这个端口传输。前面已经提到过，Server 端是一种以离散周期为时间单位工作的实时系统，Client 端程序必须在每个仿真周期结束前及时做出决策并将动作指令发送给 Server 端，否则将错过执行动作的机会。这就要求 Agent 决策必须具有较高的实时性。

为了尽可能的模拟现实环境，仿真 2D 比赛平台给场上的球员加了很多限制，Client 端程序必须在这些限制下进行决策。比如：每个球员都有一定的可视范围，每次所能获得的视觉信息只是自己可视范围内的对象，而且视觉信息中也加入了随机误差；每个球员都有自己的体力值，在连续跑动一段距离之后，球员将因体力过低而无法继续跑动，而且每个球员都有一定的体力总量，如何更好地分配体力也是 Agent 决策时需要考虑的；为了反映出实际比赛中球及球员运动的不确定性，Server 引入了噪声的干扰，使比赛更趋于真实。

2.2　理论模型

前面已经提到过，RoboCup 仿真 2D 平台是一个动态、实时和不确定的多 Agent 系统，因此，Client 端程序在考虑 Agent 决策时不能仅仅认为系统中只存在一个 Agent，而需要用多 Agent 的决策模型来进行建模。WE2009 是在 POSG 的理论模型的基础上进行设计和开发的，在具体分析 WE2009 各模块之前，了解和熟悉 POSG 的理论模型也就非常重要。

2.2.1 POSG的基本模型

20 世纪 50 年代早期，来自美国加利福尼亚大学的 Lloyd 教授提出了随机博弈的概念。在博弈论中，随机博弈是一类包含一个或多个 Agent 进行的具有状态转移概率的动态博弈过程。在每一个阶段的开始，博弈处在某个特定状态下，Agent 选择自身的策略并获得相应的由当前状态和策略决定的收益，然后博弈按照概率的分布和参与者的策略随机转移到下一个阶段。在新的状态阶段，Agent 重复上一次的策略选择过程，然后博弈继续进行[Lloyd, 1953] [Nicolas, 2002]。

POSG，也就是部分可观察随机博弈，是在随机博弈的基础上加入了部分可观察的特性，也就是 Agent 对当前的状态不是完全可知的，而是存在一定的不确定性。在 POSG 问题的每一步，所有 Agent 都会获得新的观察，并据此进行

决策从而得到一定的收益，目标是使期望收益达到最大值。至于 Agent 之间到底是合作的关系还是对抗的关系，取决于收益函数是否相同，这也是 POSG 与 Dec-POMDP 的区别所在。一个 POSG 问题可能是一个无限阶段的过程或者有限阶段的过程：对于无限阶段的问题，Agent 的策略与时间是没有关系的，也就是平稳策略；对于有限阶段的问题，Agent 的策略必须考虑当前时刻在整个阶段中所处的位置，也就是非平稳策略。RoboCup 仿真 2D 比赛因为每场比赛有时间的限制，因此可以看成一个有限阶段的 POSG 问题。来自美国卡耐基梅隆大学的 Colin 博士通过实验证明了在 RoboCup 比赛中不同时刻采取不同的策略可以整体上获得更高的胜率[Colin, 2007]。比赛刚开始的时候，Agent 可以采取正常的策略，而且与时间没有关系。但是当比赛快要结束的时候，如果比分领先，Agent 可能会采取比较保守的策略，以期望能够在比赛结束前保持住胜果；如果比分落后，Agent 可能会采取更加重视进攻的策略，以期望能够在比赛结束前扳平比分甚至反超获胜。

下面，我们来介绍 POSG 的数学描述，一个 POSG 问题是一个七元组 $<\tau,S,\{b^0\},\{A_i\},\{O_i\},P,\{R_i\}>$[Eric, 2004]：

- τ：是 Agent 索引的有限集合，表示为 $1,...,n$；
- S：是世界状态的有限集合；
- $\{b^0\}$：$b^0 \in \Delta(S)$ 表示初始时刻时的状态分布；
- $\{A_i\}$：A_i 是第 i 个 Agent 所有可用行动的有限集合，$\vec{A}=\times_{i\in\tau} A_i$ 是所有 Agent 联合行动的集合，例如 $\vec{a}=<a_1,...,a_n>$ 表示一个联合行动；
- $\{O_i\}$：O_i 是第 i 个 Agent 观察的有限集合，$\vec{O}=\times_{i\in\tau} O_i$ 是所有 Agent 联合观察的集合，例如 $\vec{o}=<o_1,...,o_n>$ 表示一个联合观察；
- P：是马尔可夫状态转移和观察概率的集合，$P(s',\vec{o}|s,\vec{a})$ 表示所有 Agent 在状态 s 执行联合行动 $\vec{a}$ 后，到达状态 s' 并得到联合观察 $\vec{o}$ 的概率；
- $\{R_i\}$：R_i 表示第 i 个 Agent 的立即收益函数。

由于博弈双方的收益函数是不同的，也就说明他们的目标是不同的，整个博弈的过程就是双方 Agent 互相追求各自收益最大化的过程。

2.2.2 WE2009 的POSG建模

RoboCup 仿真 2D 比赛可以看成是一个典型的部分可观察随机博弈问题，比赛双方有着正好相反的目标，分别希望攻入对手较多的球而自己被进较少的球。另外，Server 加入的噪声引出了观察和行动的不确定性，Server 对 Agent 视觉的限制引出了 Agent 部分可观察的特性。因此，WE2009 可以用 POSG 来

进行建模。另外，我们结合 RoboCup 仿真 2D 平台的特点在其中加入了行为生成器的模块。WE2009 被抽象为一个九元组 $<\tau,S,\{b^0\}\{A_i\},\{O_i\},P,\{R_i\},K,\{B_k\}>$[Ke, 2009]：

- τ：表示场上双方所有 22 名球员的索引，由于 Coach 并不出现在场上，在这里不再考虑，$\tau=\{1,...,22\}$；
- S：表示场上世界状态的集合，由于仿真 2D 比赛状态空间巨大且都是连续量，我们采取因子化的状态表示方法，状态因子包括球和球员的位置、速度以及球员特有的一些属性；
- $\{b^0\}$：初始时刻的状态对应于开场比赛前的情况，一般来说，开场前 Agent 对于自己队友的情况是已知的，而对于对手都是未知的；
- $\{A_i\}$：表示 Server 规定的原子动作的集合，除了双方各自的守门员多出一个扑球的原子动作外，所有 Agent 的原子动作集合都是一样的，但并不是在任何时候都可以执行所有的原子动作，比如球不在自己的可踢范围时就不能踢球；
- $\{O_i\}$：表示 Server 通过 UDP 协议传给所有 Agent 的各种信息，包括自身感知信息、视觉信息和听觉信息，Server 不仅在信息中加入了噪声，而且不同 Agent 由于各自状态不同，得到的信息也不同；
- P：对应于状态转移函数，输入是状态和行动，输出是可能的后继状态以及达到这些后继状态对应的概率；
- $\{R_i\}$：R_i 对应于第 i 个 Agent 的立即收益函数，由于每个 Agent 所在球队不同，所处的球队角色也不同，因此不同的 Agent 有着不同的立即收益函数；
- K：表示行为生成器的索引集合，进攻行为包括传球、带球和射门等，防守行为包括盯人和封堵等；
- $\{B_k\}$：B_k 对应于第 k 个行为生成器，输入是当前状态和指定的 Agent 索引，输出是对应该行为的有效集合，以及各个行为执行后的后继状态和概率。

2.3　系统结构分析

通过上一节，我们已经了解了 WE2009 的抽象模型，这是 WE2009 的系统结构设计的理论基础。在这一节，我们首先介绍 WE2009 的系统决策流程，然后分析 WE2009 系统结构中的三个主要模块。

2.3.1　系统决策流程

在 RoboCup 仿真 2D 比赛过程中，Server 程序维持着整个赛场的客观世界状态，而对应于每一个 Agent，他们通过观察维持着各自的主观世界状态。周期开始时，Agent 接收 Server 发来的已经加入噪声的世界状态信息，并通过底层更新算法转化为自己的主观世界状态，在完成决策后，Agent 将动作指令发送给 Server 并进入下一周期，比赛按照这个过程不断地进行。

图 2.3为WE2009 的系统决策流程图，其中加入了行为生成器这一部分，是 WE2009 比POSG基本模型多出来的重要部分。可以看出，每一个周期的Agent 决策都经历了三个重要的过程，分别是信息处理、高层决策和行动执行，这三个过程也对应了WE2009 系统结构中的三个主要模块。

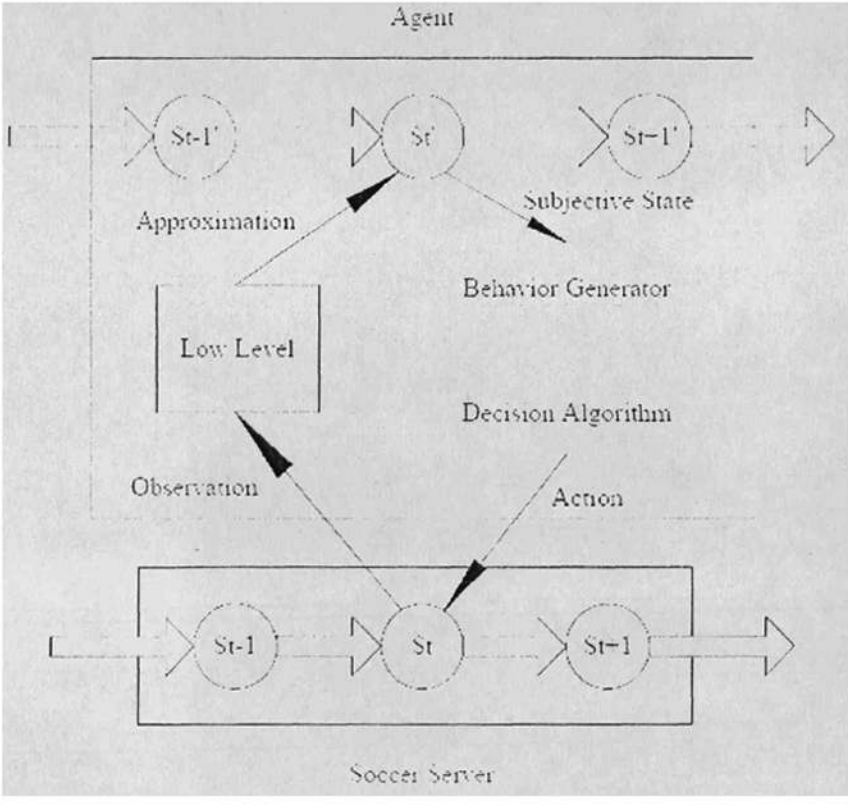

图 2.3　系统决策流程图

2.3.2　信息处理模块

信息处理模块负责将Agent从Server接收到的所有Socket字符串信息转化为Agent决策时可以直接使用的状态信息。前面已经说过，当一个新的周期开始时，Server会根据每个Agent的状态不同，向Agent发送三种信息：分别是自身感知信息、视觉信息和听觉信息。自身感知信息相当于人类的感觉器官，包括Agent的体力、速度、手臂方向以及是否碰撞等；视觉信息相当于人类的眼睛，包括在Agent视角范围内所有对象相对于Agent的距离和方向，当然，视觉信息中包含了Server加入的随机噪声；听觉信息相当于人类的耳朵，是由场上的其他Agent或者Coach通过Server要传达的字符串信息。信息处理模块就是负责把这三种信息的字符串通过一定的计算，转换为当前场上的世界状态。图 2.4为信息处理模块的流程图，图中的每一个方框对应了一个子模块。

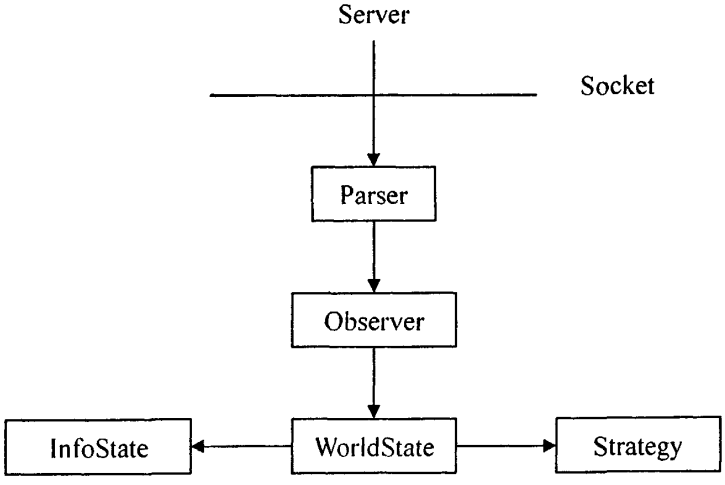

图 2.4　信息处理模块流程图

Parser 直接接收 Server 发来的 Socket 字符串，并把这些字符串信息储存在 Observer 相应的数据结构中。Observer 从字面意思理解就是观察，它本身也对应了 POSG 模型中的O。但是，Agent 的观察中大部分都是相对信息，比如球相对于自己的位置和速度、球场上标志相对于自己的位置等，甚至在 Observer 中都不包括 Agent 自身的位置，因此 Observer 中的信息无法在 Agent 决策时直接使用，我们必须让这些观察信息转化为状态信息，WorldState 的功能就是完成这部分的转化。WorldState 子模块更新结束后，在其数据结构中就储存了当前场上的世界状态。其实从理论上讲，由观察得到的是一个状态的概率分布，

但为了提高 Agent 决策的效率，我们在大多数不需要精确求解的决策时只考虑当前概率最大的状态，只有在必要的时候才考虑所有可能状态的概率分布。由于 RoboCup 仿真 2D 平台的复杂性和特殊性，Agent 仅仅靠 WorldState 中的信息是远远不够的，InfoState 和 Strategy 是以 WorldState 为基础更深层次的“状态信息”。

InfoState 被称为信息状态，它是根据 WorldState 中的信息通过一定的计算得到的，也是完全客观的信息。InfoState 主要包括下面 2 个部分：

- PositionInfo：位置信息，包括各个球员之间的相对距离和相对角度，双方球员的攻防位置关系和双方的越位线等，是对场上对象“静态”的分析；
- InterceptInfo：拦截信息，对场上的拦截情况进行分析，并计算所有球员的拦截优先级列表，是对场上对象“动态”的分析。

Strategy 被称为策略，它与 InfoState 的区别在于它是完全主观的信息，也就是说 Strategy 里保存的是 WE2009 特有的一些信息，其他球队并不一定有或者虽然有但是可能和 WE2009 计算的信息结果却不同。比如，在进攻时 Strategy 会计算当前的形势下带球 Agent 的目标前进方向以及最多可以前进的距离，在防守时 Strategy 会计算如何给不同的 Agent 分配防守任务等，类似的这些信息对于高层决策模块是非常有用的。

总之，经过信息处理模块，WorldState、InfoState 和 Strategy 的信息就会得到更新，Agent 决策时就可以根据需要分别调用这三个子模块的相关接口。

2.3.3　高层决策模块

高层决策模块是 WE2009 决策系统的核心，也是任何一个仿真 2D 球队研究的重点，因为 Agent 决策后是否能够起到好的效果主要由这一模块来决定。该模块采用了独立行为生成器的结构设计，只要输入状态和 Agent 索引，行为生成器就能决策出相应的行为集合。这样的设计可以带来两大特点：一是高层决策模块可以设计成 Anytime 的决策框架，以便充分利用每周期 Agent 的决策时间，保证决策出的行为是给定当前时间下的最好行为；二是 Agent 决策时反算队友或者反算对手非常方便，可以提高多 Agent 之间团队合作的效率。我们首先介绍行为生成器，然后分别介绍这两个特点。

WE2009 中的行为生成器主要分为进攻和防守两类，每个行为生成器都与人类足球比赛的某一种行为相关。

带有进攻特点的行为生成器主要包括：

- 射门 Shoot；

- 传球 Pass；
- 带球 Dribble；
- 铲球 Tackle；
- 截球 Intercept；
- 跑位 Position。

带有防守特点的行为生成器主要包括：

- 盯人 Mark；
- 封堵 Block；
- 按阵型跑位 Formation。

行为生成器的输入是当前状态和指定的 Agent 索引，当前状态包括信息模块更新后的三类信息，Agent 索引一般情况下都是指自身，但在某些行为下需要反算队友或对手时就变为相应的 Agent。行为生成器的输出是对应该行为的有效集合，以及各个行为执行后的后继状态和概率，比如 Agent 进行射门决策时，Shoot 行为生成器可能会输出很多个射门行为，每个行为的射门点都不同，如果靠近门柱的射门点则被对方守门员扑到球的概率较小但偏出门柱的概率较大，如果靠近球门中心的射门点则偏出门柱的概率较小但被对方守门员扑到球的概率较大。

行为生成器在产生了行为的有效集合后按照下式对所有行为进行收益评价：

$$Eva = Succ\_Poss \times Succ\_Eff + Fail\_Poss \times Fail\_Eff \tag{2.1}$$

我们将每一个行为的后继状态归为两类，成功或者失败，$Succ\_Poss$ 和 $Fail\_Poss$ 分别对应了该行为的成功概率和失败概率。$Succ\_Eff$ 和 $Fail\_Eff$ 分别是该行为成功时和失败时对整个球队的一个收益，一般情况下，成功收益为正值，失败收益为负值。最终，集合内的所有行为都会得到一个总收益 Eva，Agent 选择 Eva 最大的行为作为当前状态下此行为生成器产生的最佳行为。

上面已经介绍了单个行为生成器是如何完成对应行为的决策的，下面，我们将分别介绍 Anytime 的决策框架和基于反算的团队合作方法。

Anytime 算法不同于传统算法，它扩充了传统算法的“输入—处理—输出”的计算过程，提供了运行中动态输出结果的功能。Anytime 算法把每一组输入和特定的计算时间映射为一组输出结果，随着计算时间的不同，每一组输入有相应的多组输出结果。这一特征使得 Anytime 算法可以在任何时候被中断，返回特定质量的解，因此被称为 Anytime 算法[Nikos, 2004] [Joshua, 2005]。

WE2009 的高层决策模块就是一个Anytime的决策框架，如图 2. 5所示。椭圆节点表示状态节点，对应了场上的一个实际状态或决策过程中产生的虚拟状

态；矩形节点表示不同的行为生成器。Agent从State0 状态开始决策，分别经过 Shoot、Pass、Dribble等行为生成器的计算后，会产生若干的后继状态，对应于图中的State1.1、State1.2 等节点，然后Agent会分别从这些状态节点继续进行决策，产生第二步的行为，像这样不断地进行下去。

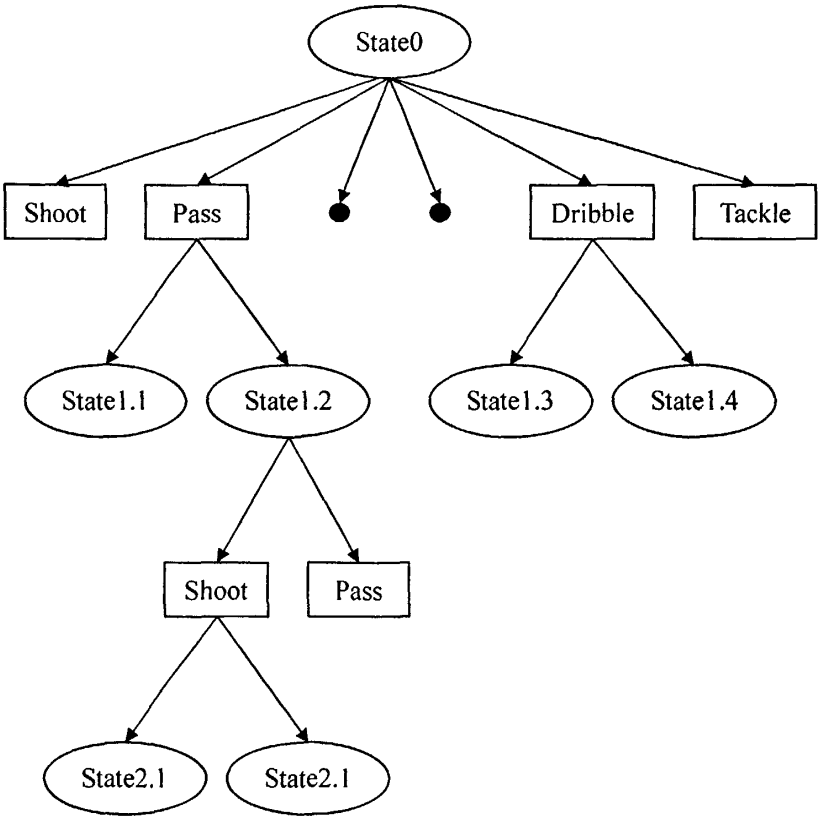

图 2.5　Anytime 决策框架

如果决策时间允许，Agent 通过多步行为生成器的搜索，可以找到直到射门成功的一条最优行为链。然而我们前面曾经提到过，RoboCup 仿真 2D 平台是实时决策的，每个周期的决策时间最多只有 100 毫秒，因此要求周期结束前必须产生一个最优行为，并转化为 Server 可以接受的动作指令发给 Server，又因为对于不同的状态和不同的 Agent，行为生成器所花的时间都是不同的，只有采取这样 Anytime 的计算方法，才能保证最大限度的利用有限的决策时间。也就是说，在有些周期，Agent 可能只进行一步决策时间就到了，这时就必须停止下一步的搜索；在另外一些周期，Agent 一步决策之后还有很多剩余时间，

这时可以继续向下进行搜索，直到周期快要结束的时候停止。另外，由于决策时间与计算机硬件的关系非常大，而我们的球队参加比赛可能会遇到性能或高或低不同的硬件环境，Anytime 的决策框架可以保证在任何的硬件环境下，Agent 决策都可以达到尽可能好的效果。

在类似 RoboCup 仿真 2D 这样一个多 Agent 系统中，反算成为 Agent 决策时保证一定的决策效果所采取的重要手段。反算包括反算队友和反算对手两种：前者是为了多个 Agent 之间更好的合作，比如正在执行 Position 的 Agent 通过反算带球队友的传球决策，就能够主动跑向更有利的接球点，结合通讯后，可以很大程度上提高配合的成功率；后者是为了在博弈过程中能够更好的揣测对方可能的决策，比如正在执行 Block 的 Agent 通过反算带球对手的带球决策，可以知道对手的最佳带球方向，进而提前做好封堵的准备，提高了封堵成功的概率。由于每个行为生成器已经非常独立和模块化，因此从工程角度来讲，反算也非常简单，只需要新建某个行为生成器的实例，并传入相应的状态和 Agent 索引，通过该行为生成器的计算，就可以得到产生的多个行为，以及各个行为的后继状态和收益。

图 2. 6是仿真 2D比赛中截取的一个反算在跑位Agent决策中的应用场景，我们通过分析该场景来介绍反算在团队合作中的应用。黄色圆圈表示我方的进攻球员，蓝色圆圈表示对手的防守球员，球员旁边的数字表示其号码。在这个周期中，10 号球员正在向前场带球，4 号球员正在执行跑位。图中红色的圆点就是 4 号球员通过反算 10 号球员的传球行为得到的所有可能的传球点，其中标注“PA”的点是所有点中收益最高的。也就是说，4 号球员认为 10 号球员传向“PA”点的可能性最大，因此 4 号球员将此点作为自己本周期的最佳跑位点，并通过通讯的方式告知 10 号球员。同样的，该周期 10 号也进行了传球的决策，而且和 4 号球员的反算调用的是同一个行为生成器，虽然 4 号球员和 10 号球员的世界状态并不是完全一样，但由于两名球员相距很近，因此误差非常小，一般情况下 10 号球员决策出的传球点也在PA附近。当然，这属于比较完美的情况，有时虽然两名球员在当前周期的决策不同，接下来的周期还可以结合通讯去弥补使其一致。通过反算，球员之间配合的成功率会有很大程度上的提高，WE2009 在进攻时经常打出匪夷所思的身后球也基本都是依靠反算来进行配合的。

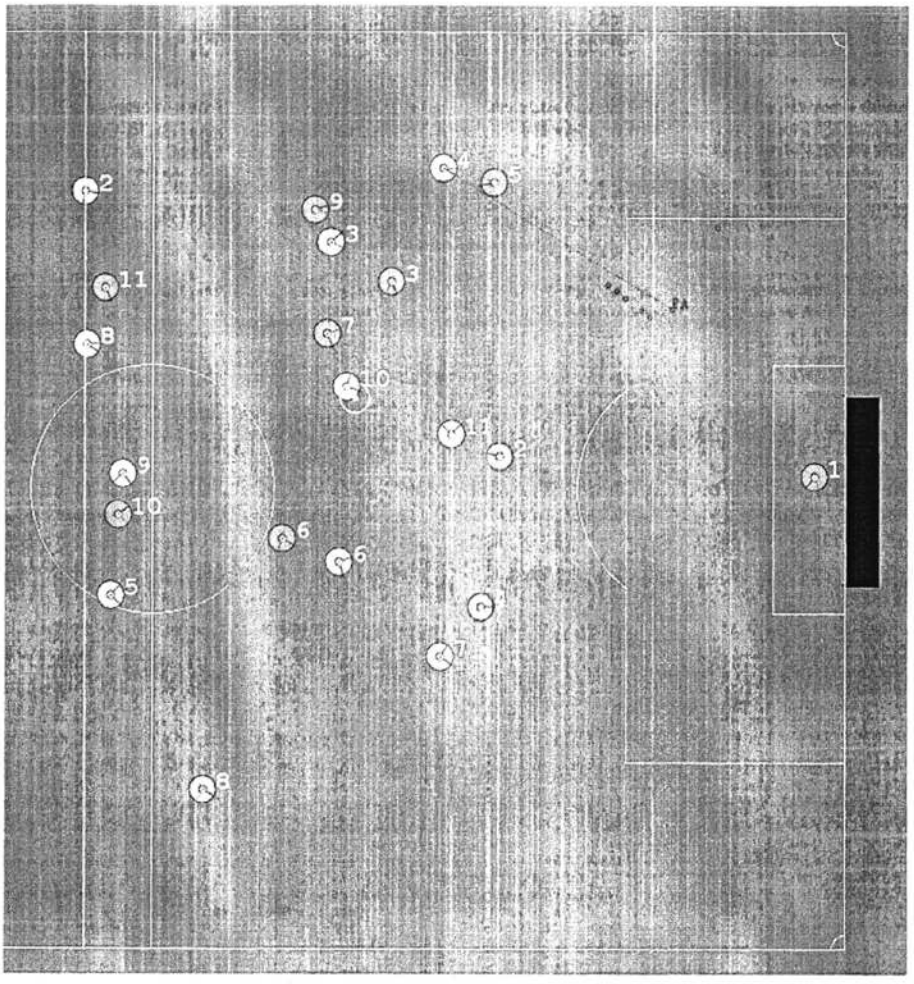

图2.6 反算的应用场景

2.3.4 行为执行模块

行为执行模块负责将高层决策模块产生的最优行为转化为Server可以接受的原子动作并发给Server。比如当前Agent决策出要将球传向某一点，但是传球这个行为是无法直接被Server接受的，必须转化成一周期或多周期的kick指令。Server提供的原子动作包括：踢球kick，转身turn，奔跑dash，铲球tackle和说话say等。我们提供了若干子模块来分别将不同的行为转化为这些原子动作，比如：

- Kicker：输入指定的Agent索引与踢球的目标速度，进行一周期或多周期的kick规划；
- Dasher：输入指定的Agent索引与目标点，进行多周期的dash和turn

的规划：

- Comunicator：输入指定的通讯内容，负责对内容进行编码并作为 say 的参数。

Agent将原子动作通过Socket发给Server后，就结束了当前周期的决策，继续等待Server发来新的信息，准备下一周期的到来。图 2.7是行为执行模块的流程图。

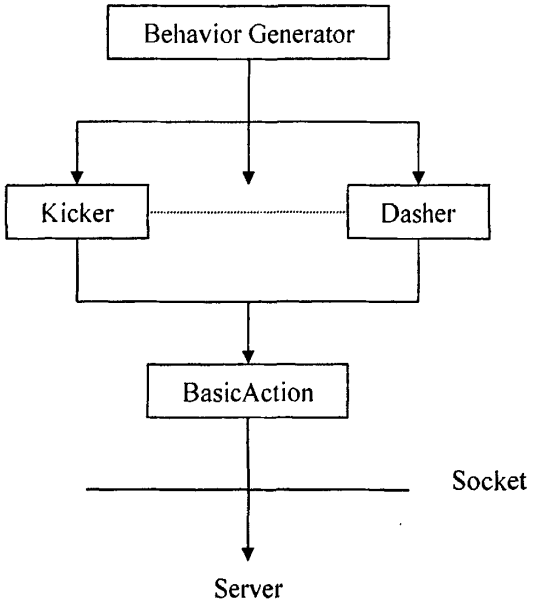

图 2.7　行为执行模块流程图

2.4　小结

本章介绍了本文的第一个主要工作，即本文重构并实现的仿真 2D 球队决策系统 WE2009。首先，本章介绍了 RoboCup 仿真 2D 平台的特点；其次，本章介绍了 POSG 的基本模型，并以此为基础对 WE2009 进行建模；然后，本章介绍了 WE2009 的系统决策流程，以及信息处理、高层决策和行为执行三个功能模块。特别是高层决策模块，采用基于独立行为生成器的结构设计，不仅可以充分利用 Agent 的决策时间，而且可以提高团队合作的效率。本文的另外两个主要工作都是在 WE2009 上实现的，因此本章是后面两章的基础。

第 3 章　行动驱动的马尔可夫决策过程

通过第 1 章的介绍，我们已经对马尔可夫决策过程有了一个初步的了解，它适用于行动结果具有不确定性的规划问题[Craig, 1999]，并且正在被应用于求解 Agent 决策的各个领域。来自美国哈佛大学的 Parkes, David C.等人用 MDP 模型来描述经济学当中的机制设计问题，通过选取最优策略保证了所有 Agent 的利益最大化[Parkers, 2003]；来自南京大学的高阳等人采用“非零和 MDP”作为多 Agent 系统的学习框架，并保证了其提出的元对策 Q 算法收敛在“非零和 MDP”的最优解[Yang, 2000]。本章在分析相关工作的基础上，针对一类特殊的 Agent 决策问题，提出了行动驱动的马尔可夫决策过程（Action-Driven MDP，简称为 ADMDP）的概念[Ke, 2010]。本章分析了 ADMDP 的理论模型，提出了 ADMDP 相关问题的求解方法，并在 RoboCup 仿真 2D 比赛的不离身带球问题中对算法进行了实验。

本章主要包括以下内容：

- 3.1 提出了两个实际存在的 Agent 决策问题，并分析该类问题的特殊性；
- 3.2 提出 ADMDP 的概念，并分析 ADMDP 的理论模型；
- 3.3 介绍 ADMDP 的精确求解算法和启发式求解算法；
- 3.4 介绍 ADMDP 在 RoboCup 仿真 2D 球队不离身带球问题中的应用；
- 3.5 介绍不离身带球问题的实验结果并进行分析；
- 3.6 进行本章小结。

3.1　问题的提出

问题一：对于一个家庭服务机器人，如果给他安排的任务是在有限时间内打扫一间很大的屋子。在整个打扫过程中，机器人必须在某一时间到指定地点进行充电，否则将会因为没电而停止打扫。在这个问题中，机器人有两类行动“打扫”和“充电”。机器人“打扫”是具有正收益的，而“充电”虽然具有零收益，但不在适当的时候到指定地点充电又无法继续“打扫”。总目标是希望机器人在指定的时间内能够打扫尽可能多的地方。

问题二：在 RoboCup 仿真 2D 比赛中，球员经常需要向身体前方进行球不离身的快速带球。在整个带球过程中，球员必须在某些周期进行踢球，否则球将会脱离自己的身体而不能继续带球。同问题一类似，球员有两类行动“奔跑”和“踢球”。球员“奔跑”将获得正收益，而“踢球”虽然具有零收益，但若不

在适当的时候踢球，球将脱离球员的可踢范围造成带球失误。总目标是在特定的周期下球员能够带更长的距离。

上面两个问题具有很大的相似性：首先，Agent 在决策时只会考虑两类行动，其中一类具有正收益，另外一类具有零收益；其次，这类问题中都存在一个或多个吸收状态，而 Agent 一旦进入吸收状态就不能再继续进行决策，这是不希望遇到的。Agent 之所以不能始终执行具有正收益的行动也正是因为要避免进入吸收状态。

3.2　理论模型

在第 1 章我们已经介绍过，基本 MDP 模型是一个四元组：$<S,A,T,R>$，目标是能够找到使得 Agent 长期收益最大的策略 $\pi:S\to A$。对于无限阶段的决策问题，长期收益的目标函数中需要加上折扣因子以保证收敛[Leslie, 1996]；对于有限阶段的决策问题，折扣因子不再需要，长期收益就是各个阶段立即收益的总和。本章主要考虑有限阶段的 MDP 问题。

为了描述上一节介绍的一类特殊问题，本文在基本 MDP 模型的基础上提出了行动驱动的马尔可夫决策过程（ADMDP）的概念，并将 ADMDP 定义为一个五元组 $<M,b,p,f,h>$：

- M 为一个标准 MDP；
- b 表示一个可能的吸收状态的集合，一旦决策过程进入该集合中的某一状态，则决策过程将永远不可能跳出该状态，即该集合中所有状态转移到其他状态的概率为 0；
- p 是吸收状态函数，用 $p(s,a)$ 表示在状态 s 执行行动 a 到达吸收状态的概率（$p(s,a)\in[0,1]$），即 $p(s,a)=\Pr\{s_{t+1}\in b \mid s_t=s,a_t=a\}$；
- f 是行动驱动函数，与 Agent 当前所处的状态 s 无关，用 $f(a)$ 表示 Agent 执行行动 a 可以获得的立即收益（$f(a)\geq 0$）；
- h 是一个正整数，表示 MDP 的有限决策阶段。

对于一个 ADMDP 问题，每一步决策后：首先，Agent 会根据执行行动的不同而获得一个正收益或零收益；另外，如果 Agent 执行过该行动后有可能到达吸收状态，则还会获得一个负收益。因此，将 ADMDP 的立即收益函定义为：

$$R(s,a)=f(a)-p(s,a) \tag{3.1}$$

如果 $f(a)>0$，则称 a 为“主动行动”，将所有“主动行动”的集合定义为 A_a；如果 $f(a)=0$，则称 a 为“被动行动”，将所有“被动行动”的集合定义为 A_p。直观上讲，A_a 表示可以带来正收益的行动集合，A_p 表示虽然不能带来正收益但却

可以避免进入吸收状态的行动集合。因此，ADMDP 的特殊性在立即收益函数 R 的定义中得到了很好的体现。ADMDP 要考虑的是多步决策问题，通过计算所有策略的长期收益，最终我们可以得到对应该问题的最优策略。

3.3 求解算法

3.3.1 MDP的经典求解算法

MDP 的经典求解算法包括后向迭代和前向搜索两大类：前者我们介绍策略迭代和值迭代，后者我们介绍基于与或图的搜索与实时动态规划算法[Changjie, 2008b]。

在策略迭代中，策略是显式表示的，可以计算得到相应的 V^{π}，然后通过下面的公式不断改进策略：

$$Q^{\pi}(s,a)=R(s,a)+\gamma\sum_{s'\in S}T^{a}(s,s')V^{\pi}(s') \tag{3.2}$$

$$\pi'(s)=\arg\max_{a\in A}Q(s,a) \tag{3.3}$$

由于可能的策略数目是有限的，而策略迭代的过程总是在改进当前的策略，算法在经过有限步的迭代之后总会收敛于最优策略。

在值迭代中，策略没有显式表示，整个过程按照动态规划的 Bellman 公式不断迭代更新来改进值函数，Bellman 公式如下[Martin, 2005]：

$$V_{n+1}(s)=\max_{a\in A}\{R(s,a)+\gamma\sum_{s'\in S}T(s'|s,a)V_{n}(s')\} \tag{3.4}$$

其中，$V_0(s)=R(s,a)$ 且 $\gamma\in(0,1]$ 为折扣因子。对于无限阶段的 MDP 问题（$\gamma<1$），当 $k\to\infty$ 时 V_k 会收敛于最优值函数 V^*；对于有限阶段的 MDP 问题（$\gamma=1$），可以利用 V_k 来得到 k 阶段的最优策略。

从上面可以看出，不管是策略迭代还是值迭代，他们考虑的都是所有状态，最终求得的也是所有状态的最优策略。然而在某些具体问题中，我们需要求解的仅仅是从固定状态开始的策略，虽然用上面两种方法同样可以求解，但他们都没有利用初始状态的相关知识，没有去把计算集中在由初始状态可能达到的状态集上。下面，我们简单介绍基于与或图搜索的 MDP 求解算法。

与或图是在经典人工智能中经常提到的一个概念，它用来对问题进行规约，能够方便地用一个类似图的结构来表示把问题规约为后继问题的替换集合。比如，一个问题A可以单独由问题B解决，也可以分解为两个子问题C和D来分别解决，还可以分解为三个子问题E、F和H来分别解决，如图 3.1所示。图中的X、

Y和Z三个节点可以理解为具有问题描述的作用，被称为或节点；其他节点都是其父节点规约成的子问题集合中的元素，被称为与节点。通过与或图，把某个单一问题规约算符具体应用于某个问题描述，依次产生出一个中间或节点及其与节点后裔。

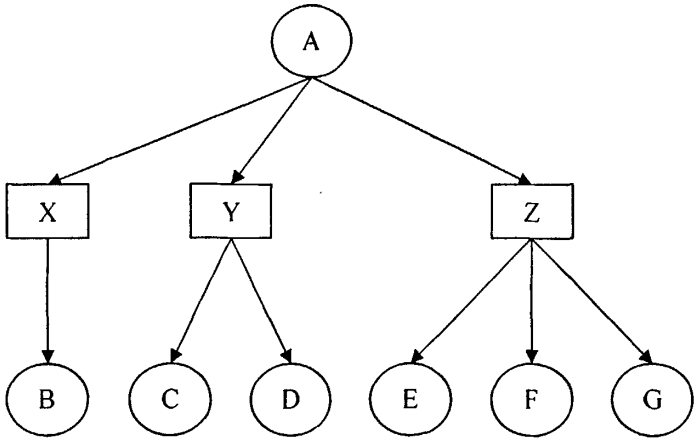

图 3.1　一个与或图

其实，图 3.1也可以用来描述一个存在不确定性的Agent决策问题。根节点A表示Agent决策的初始状态，X、Y和Z是或节点，表示Agent在初始状态可以选择的行动，其余的节点表示在执行父节点对应的行动后Agent可能到达的状态，比如Agent在初始状态执行Y行动后，会到达C或者D这两个状态之一，所有状态节点都对应了一个概率。与或图给出了更好的描述MDP求解过程的一种基本数据结构，其本质是利用了状态可达性的原理。Barto提出了与基于与或图搜索类似的方法，被称为实时动态规划算法，其目的也是避免计算所有的状态[Barto, 1995]。

3.3.2　ADMDP的精确求解算法

以值迭代为基础，本文采用离线值迭代与在线搜索相结合的方法来精确求解 ADMDP 问题。

首先，离线VI_alg算法（如图 3.2所示）采用值迭代来求解有限阶段的最优值函数V_h，整个过程按Bellman公式不断地对值函数进行更新，直到达到预定的视野长度h，迭代完成后将最优值函数保存成表。

对所有s，$V_0(s) := 0; t := 0;$
loop
　$t := t+1;$
　loop 对所有 s
　　loop 对所有 a
　　　if $p(s,a) < 1$
　　　　$Q_t(s,a) := R(s,a) + \sum_{s'} T(s' \mid s,a) V_{t-1}(s')$
　　　end if
　　end loop
　　$V_t(s) := \max_a Q_t(s,a)$
　end loop
until $t > h$

图 3.2　VI_alg 算法

其次，由于Agent在线决策时只需要求解初始状态的策略，为了提高决策效率，在线ADMDP_alg算法（如图 3.3所示）采用一步贪婪策略，对于每一个行动，其立即收益直接按照模型中的定义计算，而后继状态则通过离线计算的最优值函数进行评价，从而做出当前周期的决策。

对当前状态 s
loop 对所有 a
　if $p(s,a) < 1$
　　$Q(s,a) := R(s,a) + \sum_{s'} T(s' \mid s,a) V_h(s')$
　end if
end loop
$\pi^*(s) := \arg\max_a Q(s,a)$

图 3.3　ADMDP_alg 算法

ADMDP_alg 算法最终可以得到 ADMDP 的最优策略。但是，在 ADMDP_alg 算法中，对行动集合的所有行动 a 进行了遍历，这其实并没有考虑 ADMDP 的

特殊性。在具体的 ADMDP 问题中，A_a 和 A_p 两个行动集合中的所有行动都会存在某种与该领域相关的“优先级”。一方面，Agent 会在 A_a 集合中选择 $f(a)$ 较大的行动去获得尽量大的正收益；另一方面，Agent 也会在 A_p 集合中选择 $p(s,a)$ 较小的行动去获得尽量小的负收益。基于这种考虑，本文在下一节将提出一个启发式求解算法。

3.3.3　ADMDP的启发式求解算法

图 3.4描述的启发式求解算法被称为ADMDP_alg*。

对当前状态 s，$i := 1$，
$A_a\_depth := [1, sizeof(A_a\_list)]$
$A_p\_depth := [1, sizeof(A_p\_list)]$
loop
　if　$i \le A_a\_depth$
　　$a := A_a\_list(i)$
　　if　$p(s,a) < 1$
　　$Q(s,a) := R(s,a) + \sum_{s'} T(s' \mid s,a) V_h(s')$
　　end　*if*
　end　*if*
　if　$i \le A_p\_depth$
　　$a := A_p\_list(i)$
　　if　$p(s,a) < 1$
　　$Q(s,a) := R(s,a) + \sum_{s'} T(s' \mid s,a) V_h(s')$
　　end　*if*
　end　*if*
until　$i > A_a\_depth$ 而且 $i > A_p\_depth$
$\pi^*(s) := \arg\max_a Q(s,a)$

图 3.4　ADMDP_alg*算法

ADMDP_alg*算法仍然基于 VI_alg 算法离线求出的最优值函数表。但是，在线执行该算法前需要首先离线计算启发式信息。最基本的方法是对

ADMDP_alg 算法进行大量的实验，统计所有“主动行动”和“被动行动”趋于稳定的使用概率。这样，每一个行动 a 都有一个实值与其对应，表示 Agent 在线决策时选择该行动的概率。分别对两个行动集合按照使用概率递减的顺序进行排序，可以得到两个有序的行动列表 A_a_list 和 A_p_list。

在线执行 ADMDP_alg*算法时，需要提前确定两个量 A_a_depth 和 A_p_depth，分别表示每次决策时两个行动集合的搜索深度。这样，在对行动集合进行遍历时只需要考虑 A_a_list 的前 A_a_depth 个行动，以及 A_p_list 的前 A_p_depth 个行动即可。搜索深度可以通过多次实验来最终确定。ADMDP_alg*算法其实是 ADMDP_alg 算法的特例，如果这两个搜索深度分别等于 A_a 行动集合和 A_p 行动集合的大小，则 ADMDP_alg*算法就变成了 ADMDP_alg 算法。

在一般情况下，ADMDP_alg*算法求解的是 ADMDP 问题的近似最优策略，但启发式信息的加入可以使算法的执行时间大大减少。当然，启发式信息不仅可以通过上面提到的基于统计的方法计算，还可以考虑实际问题的背景通过经验来设定等。由于 ADMDP 一般用来描述实时的 Agent 决策问题，也就是对其在线算法的性能要求比较高，而离线算法是用来为在线算法服务的。因此，ADMDP_alg*算法采用的“用离线时间换取在线时间”的思想在某些具体问题中可以得到广泛的应用。

3.4　不离身带球问题的求解

随着 RoboCup 仿真 2D 比赛的不断发展，许多研究者在提高 Agent 的基本技能上做了大量工作。清华大学将机器学习的方法用在了球员的脚法上，提高了球员的踢球技能[Shi, 2001]；巴西的 Luiz 等人在 Q 学习算法中加入了启发式策略，并在守门员的扑球决策中起到了效果[Luiz, 2007]；德国的 Thomas 等人采用基于神经网络的方法提高了球员的封堵技能[Thomas, 2007]。目前的大量工作主要是基于强化学习的方法，强化学习是一种非监督学习的方法，它通过从环境中得到奖惩来自主的发现能够得到最大奖励的策略[Cohen, 2005]。这些工作起到了良好的效果，推进了强化学习算法的研究。但是，强化学习方法的最大问题是经常会出现学习缓慢，甚至难于收敛的现象从而无法找到最优策略。

上面提到的大部分强化学习方法在学习过程中是假设环境完全未知的，而在仿真 2D 的某些具体问题中，由于 Server 相关模型的存在，状态及状态的变化具有一定的规律。因此，Agent 不需要在环境中去盲目的学习，而可以更有效率的直接使用 MDP 算法去求解策略。在这一节，我们会具体介绍如何用 ADMDP 来求解 RoboCup 仿真 2D 球队中的不离身带球问题。

3.4.1 问题分析

在RoboCup仿真 2D球队中，带球是Agent的重要技能之一，带球又分为趟球、不离身带球等。不离身带球是指保证每周期球均在Agent可踢范围的前提下尽可能以最快的速度带球。不离身的目的是Agent每周期都有机会将球传给机会更好的队友，快速的目的是尽量不给对手进行充足的布防时间。图 3. 5可以形象的表示Agent的不离身带球问题。其中，黄色大圆和黄色小圆分别为Agent的可踢范围和身体范围，白色小圆为球。初始状态时，以Agent的中心位置为坐标原点，以Agent的身体方向为x轴的正方向，顺时针旋转 90 度为y轴的正方向建立平面直角坐标系。

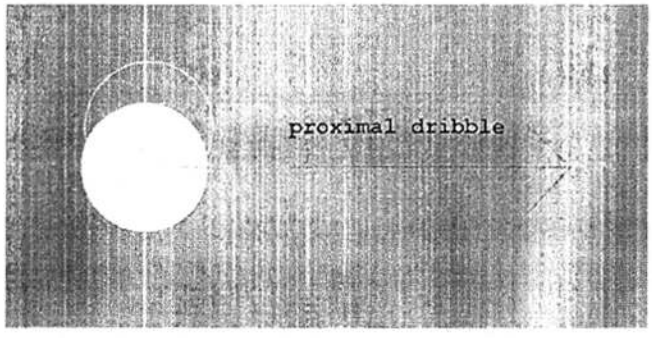

图 3.5 不离身带球示意图

WrightEagle 仿真 2D 球队曾基于搜索的方法实现了对该问题的求解并取得了一定的效果[Feng, 2007]。但是，由于老算法基于手工编码，算法中与带球行动的不确定性相关的参数需要经过大量的调试来最终确定。而且，一旦 Server 的相关参数发生变化，这些参数需要重新进行调试以保证不离身带球的效果。如果参数设置的比较冒险，则带球速度较快，但带球成功率较小；反之带球成功率提高，但带球速度会降低。为了后面便于比较实验结果，我们将老算法通过设置不同的参数分成了两个版本，倾向于带球速度的被称为 SEARCH_alg1，倾向于带球成功率的被称为 SEARCH_alg2。

3.4.2 模型建立

由于 Agent 在其中心位置 3 米范围内的所有信息每周期都可以感知到，因此不离身带球问题可以近似认为是完全可观察的，本文基于 ADMDP 的理论模型来对该问题进行建模。

- S：包括 Agent 的位置向量、速度向量和身体角度以及球的位置向量和速度向量。由于这些量均为连续量，对其进行离散，采用因子化的状态表示方法。
- A：两类原子动作：A_a 为 dash(power)，表示 Agent 向身体正前方加速，参数 power 表示加速力量，直接进行离散；A_p 为 kick(power, angle)，表示将球踢到某一位置，参数 power 和 angle 分别表示踢球力量和踢球角度，为了减小行动空间，这里并不直接对两个参数进行离散，而是将 Agent 的可踢范围进行离散，表示将球踢到某一位置。
- T：Server 会在两个方面加随机误差，造成了带球问题的不确定性。一是对 kick 行动，Server 会在 kick 对球产生的加速度上加随机误差；二是对运动的 Agent 和球，每个周期 Server 会在 Agent 和球的速度向量上加随机误差。由于采用因子化的状态表示方法，按照 Server 相关的误差模型可以计算出每个状态变量可能的后继值，进而可以得到总的后继状态集合。
- R：Agent 带球一个周期所获得的立即收益，按照 ADMDP 模型的定义，包含 $f(a)$ 和 $p(s,a)$ 两部分。
- b：在该问题中，规定吸收状态是球不在 Agent 的可踢范围内这样一类状态，即通常所说的“带球失误”。在这种情况下 Agent 不能再继续带球，而必须去截到球。在线比赛如果出现这种情况，对于整个球队将是非常不利的。
- p：与 T 类似，按照 Server 的相关模型可以计算下一周期进入吸收状态的概率。
- f：如果 a 是 dash 行动，则 $f(a)=dash\_power/100.0$，表示向前方加速越快，其收益越大；如果 a 是 kick 行动，由于对 Agent 的加速不会产生积极的影响，因此 $f(a)=0$。
- h：由于在线比赛时存在对手的原因，Agent 并不知道将要执行不离身带球的确切周期数。根据不离身带球本身的特点，h 太大或太小都将失去实际意义，因此设定 $h=5$。

在因子化的表示世界状态时，由于直接对球速离散会增加不必要的状态空间规模，而利用 Server 的相关模型，球速可以通过连续两个周期球相对位置计算得到：

$$\begin{aligned} & b\_vel_t \\ &= (b\_pos_t - b\_pos_{t-1}) \times b\_decay \\ &= \left[(a\_pos_t + b\_rel\_pos_t) - (a\_pos_{t-1} + b\_rel\_pos_{t-1})\right] \times b\_decay \\ &\approx \left[a\_vel\_x_t / a\_decay + (b\_rel\_pos_t - b\_rel\_pos_{t-1})\right] \times b\_decay \end{aligned} \tag{3.5}$$

在式(3.5)中，以 b 和 a 开始的状态变量分别对应 $ball$ 和 $agent$，b_decay 和 a_decay 都是常量。因此，我们的状态空间只需要一个三元组就可以表示：$<a\_vel\_x_t, b\_rel\_pos_{t-1}, b\_rel\_pos_t>$，三个状态变量分别表示 t 周期 agent 的速度在 x 方向的分量，球在 $t-1$ 周期和 t 周期的相对位置。

综上，该问题的目标是求解最优策略 π，使 Agent 在线比赛时可以快速的进行不离身带球，而且尽量减少“带球失误”。

3.5　实验结果及分析

基于上一节对不离身带球问题的建模，本文在 WE2009 中实现了 VI_alg、ADMDP_alg 和 ADMDP_alg* 算法，并用 ADMDP_alg 算法对 Agent 的行动使用概率进行了统计并排序。

图 3.6中，X坐标表示dash行动使用概率的降序索引（行动空间有 100 个dash行动，对应 100 个power参数值），Y坐标表示相应的使用概率。根据统计结果，ADMDP_alg*算法中的搜索深度 A_a_depth 将设为 65。

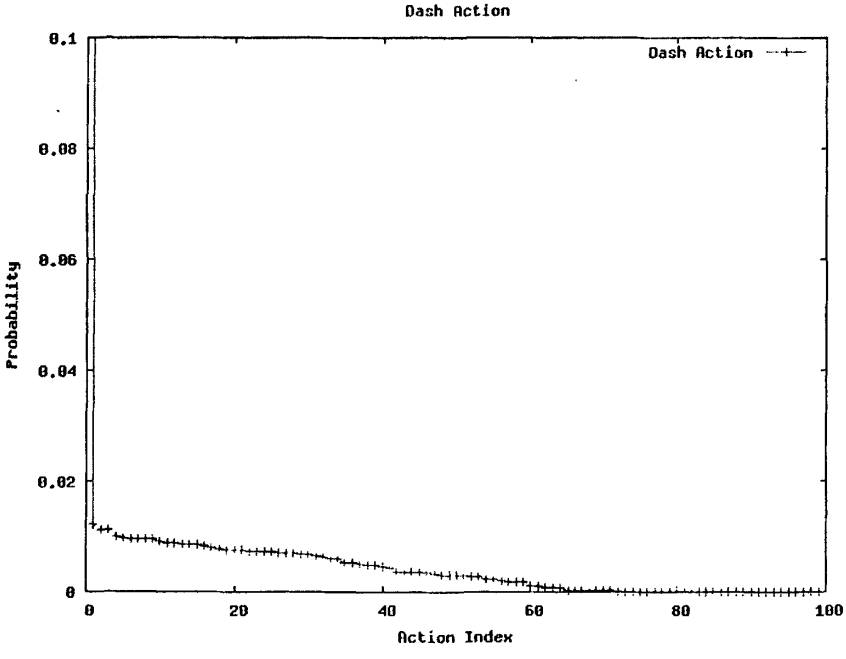

图 3.6　dash 行动的使用概率

图 3.7中，X坐标表示kick行动使用概率的降序索引（行动空间有 273 个kick

行动，对应Agent可踢范围内的 273 个位置），Y坐标表示相应的使用概率。根据统计结果，ADMDP_alg*算法中的搜索深度 A_P_depth 将设为 105。

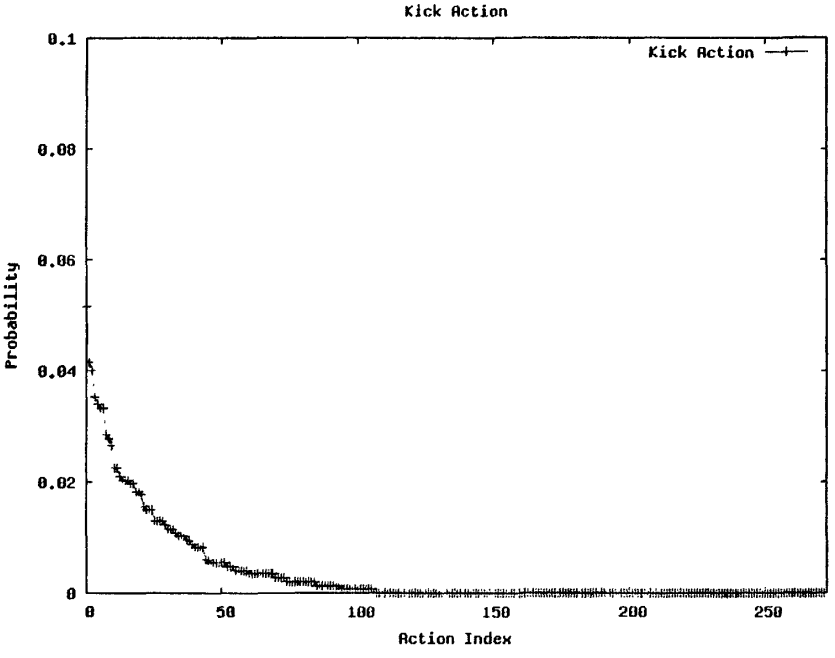

图 3.7　kick 行动的使用概率

本文对 SEARCH_alg1、SEARCH_alg2、ADMDP_alg 和 ADMDP_alg* 这四个算法分别进行了测试。测试的硬件环境是：AMD Athlon(tm) 64 Processor 3500+，2GB Memory，软件环境是：Ubuntu 9.10，rcssserver-14.0.1（RoboCup 官方于 2009 年 11 月 13 日发布）。为了排除测试的不确定性带来的影响，对每个算法分别测试了 200 轮共 10000 周期（1 周期=100 毫秒）。每一轮开始时由 Server 任意设定初始状态，然后 Agent 每周期在线决策，开始不离身带球。Server 分别记录第 1 到 40 周期时 Agent 已经前进的距离，如果在某一周期 Agent 已经进入吸收状态，则认为 Agent 在该轮测试余下的周期中带球失败。

图 3. 8和图 3. 9给出了测试结束后统计出的不离身带球问题的两个重要性能指标，X坐标均表示周期数，Y坐标分别表示当前周期对应的带球成功率和带球距离。

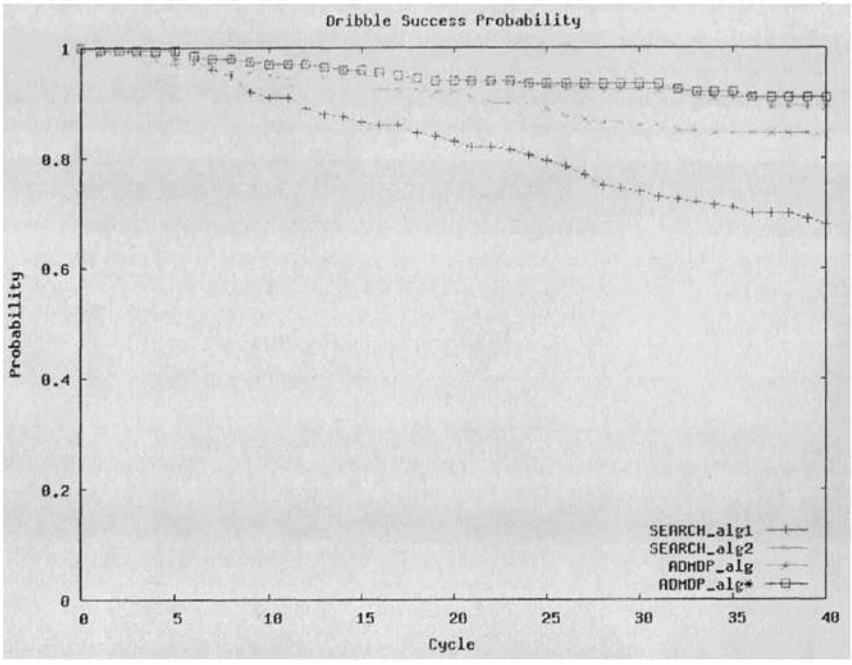

图 3.8　不离身带球的成功率

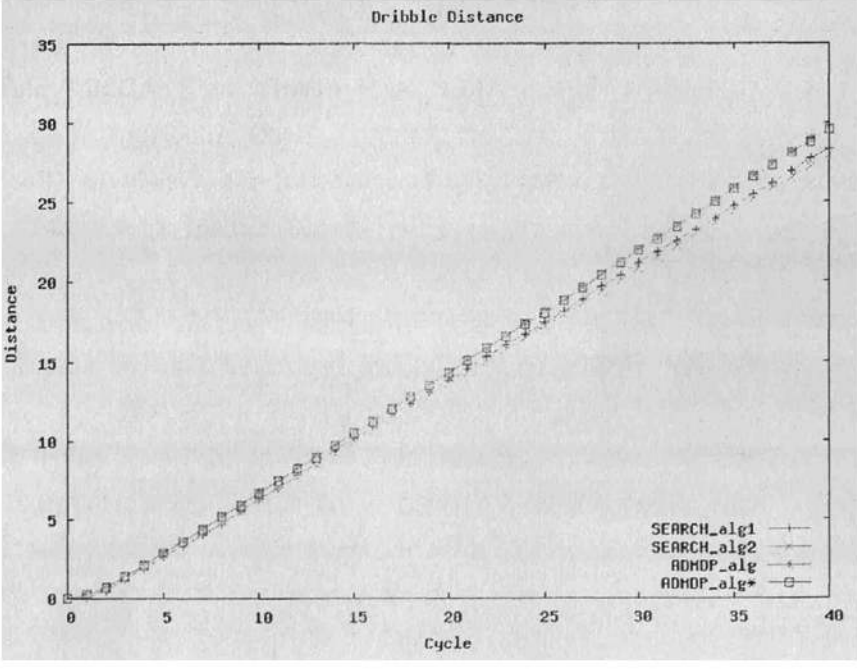

图 3.9　不离身带球的距离

表 3. 1给出了测试中各个算法的执行时间。

表 3.1　算法执行时间

	SEARCH_alg1	SEARCH_alg2	ADMDP_alg	ADMDP_alg*
执行次数	8011	8920	9525	9399
最长时间 (μs)	669	599	823	442
最短时间 (μs)	158	151	297	94
平均时间 (μs)	193.206	185.392	343.403	141.245

从测试结果可以看到：对于带球成功率，ADMDP_alg算法和ADMDP_alg* 算法明显好于两个老算法；对于带球速度，由于Agent的dash速度理论最大值是1，加上必须有kick行动的存在，因此图 3. 9中ADMDP_alg算法和ADMDP_alg* 算法在带球距离上比两个老算法的增加幅度也已经非常可观；对于算法执行时间，ADMDP_alg比两个老算法要多出很多，而考虑了启发式信息的ADMDP_alg* 算法则相对较少。而且，ADMDP_alg* 算法与ADMDP_alg算法相比，在带球成功率和带球速度上基本没有受到影响。

整体来说，ADMDP_alg* 算法的性能是最优的，我们对三个评价标准分别来分析：

- 带球成功率提高，主要是将该问题用 ADMDP 模型来建模，按照 Server 的模型考虑了行动的不确定性，而且收益函数中考虑了进入吸收状态的概率；
- 带球速度提高，主要是采用了离线值迭代和在线搜索相结合的求解方法，很好的解决了手工编码不能考虑到所有可能状态的问题；
- 算法执行时间减少，主要是考虑了启发式信息，结合不离身带球问题的相关特点，用近似最优解进行在线决策。

从实验结果可以看到，新算法在考虑了启发式信息之后，算法执行时间才有所减少。如果不考虑启发式信息，则执行时间明显多于老算法，其本质原因在于庞大的状态空间。像 RoboCup 仿真 2D 的比赛环境一样，在许多其他现实问题中，所面临的都是连续状态空间，对其因子化的状态表示是最基本的方法。该方法的缺点是容易带来维度灾难[Li, 2005]，如何更好的表示连续状态空间是今后的主要工作之一。另外，ADMDP 模型具有其特殊性，在本文实验的问题中取得了较好的效果，后面的工作包括继续对其进行理论分析，期待该模型能够应用于更多的 Agent 决策问题中。

3.6　小结

本章介绍了本文的第二个主要工作。首先，本章提出了现实生活中存在的两个具体的 Agent 决策问题并进行了分析；其次，本章针对前面的一类特殊问题，提出了行动驱动的马尔可夫决策过程的概念并分析了其理论模型；然后，本章在介绍了 MDP 的经典求解算法后，提出了求解 ADMDP 的精确求解算法和启发式求解算法；最后，本章将所提出的方法用来求解 RoboCup 仿真 2D 比赛中的不离身带球问题，实验结果表明，本文提出的方法使 Agent 的带球性能有了较大的提高。该方法已经用于本文第 2 章介绍的 WE2009 仿真 2D 球队决策系统。

第 4 章　基于阵型的零和马尔可夫博弈

在第 2 章，本文从分布式部分可观察马尔可夫决策过程（Dec-POMDP）的角度简单介绍了部分可观察随机博弈（POSG）的基本模型，并用该模型对 WE2009 仿真 2D球队决策系统进行建模。本章将从博弈论的角度更加具体的讨论随机博弈，即马尔可夫博弈[1]。本章提出了一类特殊的马尔可夫博弈，即基于阵型的零和马尔可夫博弈（Formation-Based Markov Zero-Sum Game，简称为 FMZSG）。本章分析了FMZSG的理论模型，并以此为基础来描述RoboCup仿真2D比赛中的Anti-Mark问题。针对Anti-Mark问题，本章提出了一个基于阵型变换的启发式求解方法，使球队在与盯人防守的对手比赛时取得了较好的效果。

本章主要包括以下内容：

- 4.1 介绍博弈论的相关背景；
- 4.2 分析 FMZSG 的理论模型，并以此为基础来描述 RoboCup 仿真 2D 比赛中的 Anti-Mark 问题；
- 4.3 介绍一个基于阵型变换的启发式求解方法，用来求解 Anti-Mark 问题；
- 4.4 介绍 Anti-Mark 问题的实验结果并进行分析；
- 4.5 进行本章小结。

4.1　博弈论相关背景

博弈论(Game Theory)又称为对策论、游戏理论、竞赛理论等。博弈论考虑游戏中个体的预测行为和实际行为，并研究它们的优化策略。具有竞争或对抗性质的行为称为博弈行为。在这类行为中，参加斗争或竞争的各方各自具有不同的目标或利益。为了达到各自的目标和利益，各方必须考虑对手各种可能的行动方案，并力图选取对自己最为有利或最为合理的方案[Drew, 1991]。

博弈论思想古已有之，我国古代的《孙子兵法》不仅是一部军事著作，而且算是最早的一部博弈论专著。从 20 世纪初开始，越来越多的人开始关注并研究博弈论。1944 年，Von Neumann 和 Oskar Morgenstec 发表了对博弈论建立具有里程碑意义的标志著作《Game Theory and Economic Behavior》。20 世纪 50 年代，Nash、Selten 和 Harsanyi 丰富并发展了博弈论，使博弈论最终成熟并投

[1] 马尔可夫博弈是随机博弈更直观的叫法[Owen, 1982]，为了与习惯叫法相统一，本章采用马尔可夫博弈的叫法。

入使用。近年来，博弈论作为分析和解决合作与冲突的工具，在经济学、管理学和信息科学等领域受到了人们更多的关注，也得到了更多的应用[Weiwei, 2009]。

4.1.1　博弈与零和博弈

关于博弈论的定义，到目前也没有一个统一的结论。一般情况下，博弈包含以下要素[Osborne, 1994]：

- Agent：在一场竞赛或博弈中，每一个有决策权的参与者称为一个 Agent，只有两个 Agent 的博弈称为“两人博弈”，而多于两个 Agent 的博弈称为“多人博弈”；
- 策略：一局博弈中，每个局中人都要选择实际可行的完整的行动方案，即方案不是某阶段的行动方案，而是指导整个过程的一个方案， Agent 的一个可行的自始至终全局筹划的行动方案，称为这个 Agent 的一个策略；
- 收益：一局博弈结束时的结果称为收益，收益可以为正也可以为负，每个 Agent 在一局博弈的收益，不仅与该 Agent 自身所选择的策略有关，而且与其他所有 Agent 选取的一组策略有关，收益函数是所有 Agent 各选定一个策略后形成的策略组合的函数；
- 次序：博弈各方的决策有先后之分，而且一个博弈方要做不止一次的决策选择，就出现了次序问题，其他要素相同次序不同，博弈就不同。
- 均衡：均衡是平衡的意思，在经济学中，均衡意味着相关量处于稳定值，比如在供求关系中，某一商品市场如果在某一价格下，想以此价格买此商品的人均能够买到，想卖的人均能卖出，此时就可以说该商品的供求达到了均衡。

根据分类的角度不同，博弈可以有多种分类方式。其中，从所有 Agent 的收益函数的代数和是否为零，可以分为零和博弈与非零和博弈。零和博弈属于非合作博弈，指参与博弈的各方，在严格竞争下，一方的收益必然意味着另一方的损失，博弈各方的收益和损失相加总和永远为零，双方不存在合作的可能。RoboCup 仿真 2D 比赛就是一个典型的零和博弈问题，因为任何一方的进球必然意味着另外一方的失球，比赛结束时，一方的胜利也意味着另一方的失败。本章主要以零和博弈为基础展开研究。

4.1.2　零和马尔可夫博弈

马尔可夫博弈可以看成是经典博弈论和马尔可夫决策过程相结合的统一体，顾名思义，马尔可夫博弈是博弈论在马尔可夫环境下的一个扩展[Van, 1981]。而零和马尔可夫博弈（Zero-Sum Markov Game）是指在马尔可夫博弈的基础上，博弈双方是严格的竞争关系，即一方获得正收益必然意味着另外一方获得等量的负收益。许多实际问题都可以看成是零和马尔可夫博弈，比如军事上的追捕逃犯问题。而且，许多用策略空间描述的经典零和博弈问题如果用零和马尔可夫博弈来描述，将会得到更有效的解决[Michail, 2002]。最为著名的例子就是 Lilttman 提出的方格足球问题，下面我们来简单介绍下这个问题。

方格足球问题如图 4.1所示。场地由一个4×5 的方格矩阵组成，A和B是两个Agent，圆圈表示足球。在任何时刻，两个Agent各占有一个方格，并且可以选择 5 个行动，分别是：上、下、左、右和继续保持在原地。当两个Agent在某一时刻都选择行动时，两个行动会按照随机的次序执行。当持球的Agent带着球进到对方球门时（A向左边进攻，B向右边进攻），该Agent会得 1 分，并且A和B会被安排在图中的位置，球随机的放在A或者B的位置。如果持球的Agent执行自己的行动后将要与对手重合，则该行动不会执行，并将球权让给对手以作为惩罚。直观上讲，防守Agent最好的策略就是停在对手Agent将要到达的位置，然而对手如何选择行动它并不知道，这就存在着一个不断博弈的过程。

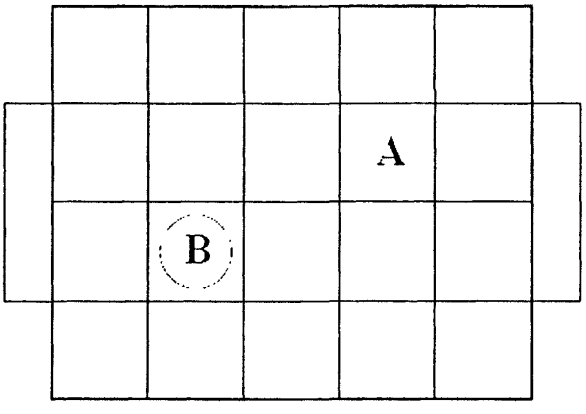

图 4.1　方格足球问题示意图

通过上面的例子，我们对零和马尔可夫博弈有了一个初步的了解，下面，我们来介绍它的理论模型。经典的零和马尔可夫博弈被定义为一个六元组

$<S, A, O, P, R, \gamma>$ [Littman, 1994]：

- S：表示有限的博弈状态集合；
- A, O：分别对应博弈双方各自的行动集合；
- P：是一个马尔可夫状态转移函数，$P(s,a,o,s')$表示在状态s博弈双方分别执行行动a和o后到达状态s'的概率；
- R：是一个收益函数，$R(s,a,o)$表示在状态s博弈双方分别执行行动a和o后己方得到得立即收益，因为是零和博弈，对手得到的立即收益为$-R(s,a,o)$；
- γ：$\gamma \in (0,1]$是一个折扣因子，用来保证长期收益的收敛。

在零和马尔可夫博弈中，策略π是从状态集合S到行动集合A的概率分布的映射，即$\pi: S \to \Omega(A)$，这一点与普通的 MDP 不同。因为有博弈对手的存在，零和马尔可夫博弈的最优策略可能是随机的，$\pi(s)$表示在状态s时行动的概率分布，$\pi(s,a)$表示在状态s时执行行动a的概率。

4.2　理论模型及应用实例

在这一节，我们将重点介绍本文以零和马尔可夫博弈为基础提出的一类特殊的博弈问题，即基于阵型的零和马尔可夫博弈，并介绍其在 RoboCup 仿真 2D 比赛中的具体应用实例。

4.2.1　角色与阵型

来自美国德州大学奥斯汀分校的 Peter 曾提出了在多 Agent 系统的研究中普遍存在的一类领域，即 PTS（Periodic Team Synchronization）领域[Peter, 1999]。PTS 领域有下面几个特点：多个 Agent 合作去完成一个共同的目标；Agent 之间通讯是受限制的；Agent 决策是在动态实时的环境中。现实中有很多 PTS 领域的问题，RoboCup 仿真 2D 就是一个典型的例子。角色（Role）和阵型(Formation)是 PTS 领域中存在的两个重要概念。

角色规定了一个 Agent 可以做的所有行动，不同角色的 Agent 可以完成的任务也有所不同。角色可以严格规定一个 Agent 的所有行为，也可以是比较灵活的，留给 Agent 一定程度上的选择权。举一个例子，假设一个 Agent 的任务是看钟表并吹口哨。角色r可以严格规定 Agent 必须每一个小时都吹口哨；同时，角色r也可以灵活的赋给 Agent，比如规定 Agent 以高于 25%低于 75%的概率在每一个小时吹口哨，这种情况下角色r就包含了一定的参数，参数可以指明 Agent 具有多大的灵活性。下一节我们在求解 Anti-Mark 问题时也是基于

Agent 的灵活角色的。

阵型是包含一个团队所有 Agent 角色的集合，指明了所有 Agent 的角色。同时，阵型中还可以包括子阵型，子阵型是角色集合的子集，指明一部分 Agent 的角色。对于具有 n 个 Agent 的团队 $A=\{a_1,a_2,...,a_n\}$ 来说，团队的阵型可以形式化定义为：$F=\{R,\{U_1,U_2,...,U_n\}\}$。$R$ 是 Agent 所有可能角色的集合 $R=\{r_1,r_2,...,r_n\}\}$，并满足 $i\neq j\Rightarrow r_i\neq r_j$。虽然 Agent 的个数与角色的个数相等，但有时可以将同一角色赋予不同的 Agent。U_i 是 Agent 一部分角色的集合 $U_i=\{r_{i1},r_{i2},...,r_{ik}\}\}$，并满足 $r_{ia}\in R$ 且 $a\neq b\Rightarrow r_{ia}\neq r_{ib}$。一个团队的阵型不同，Agent 决策时所选择的策略就会不同。

4.2.2　理论模型

本文将角色和阵型的概念引入了零和马尔可夫博弈中，将两者结合后提出了一类特殊的博弈，称为基于阵型的零和马尔可夫博弈（FZSMG），并将其抽象为一个多元组 $<R,T,O,F_T,F_O,A,O,P,R'>$：

- R：$r_1\sim r_m$，Agent 所有可能角色的集合，在具体领域可能有不同的含义，比如在 RoboCup 仿真 2D 比赛中，Agent 的角色可以根据球员的位置划分为前锋，中场和后卫等，在军用机器人中，Agent 的角色可以根据所要执行任务的不同划分为扫雷机器人和救援机器人等；
- T,O：$t_1\sim t_n$ 和 $o_1\sim o_n$ 分别对应所有队友 Agent 和所有对手 Agent 的集合，因为是一个博弈问题，所以必然存在队友和对手，队友和对手内部存在合作的关系，队友和对手之间只存在竞争的关系；
- F_T,F_O：表示队友和对手的阵型，$f_{T1}\sim f_{Tn}$ 和 $f_{O1}\sim f_{On}$ 分别对应队友 Agent 和对手 Agent 的角色，f_{Ti} 和 f_{Oj} 分别表示第 i 个队友 Agent 和第 j 个对手 Agent 的角色；
- A,O：分别对应队友 Agent 和对手 Agent 的行动集合，由于问题的特殊性，这里的行动是指改变每个 Agent 的角色，对于团队来说就是改变团队的阵型；
- P：是一个马尔可夫状态转移函数，$P(f_T,f_O,a,o,f_T',f_O')$ 表示博弈双方在各自的阵型 f_T 和 f_O 的情况下分别执行行动 a 和 o 后，阵型分别变为 f_T' 和 f_O' 的概率；
- R'：是一个收益函数，$R'(f_T,f_O,a,o)$ 表示博弈双方在各自的阵型 f_T 和 f_O 的情况下分别执行行动 a 和 o 后队友 Agent 获得的整体立即收益，这里的收益是指整个团队，因为阵型是与整个团队有关的，单个 Agent 的收益我们不再考虑，同时，对手得到的立即收益为 $-R'(f_T,f_O,a,o)$。

下面，我们来分析该模型与前面介绍的基本零和马尔可夫决策过程模型的联系与区别。首先，该模型引入了角色与阵型，因此模型中加入了 Agent 角色的集合，并且用博弈双方的阵型作为当前的博弈状态，这是该模型的最主要特点；其次，该模型是从多人博弈的角度出发的，因此模型中加入了队友 Agent 和对手 Agent 的集合；然后，该模型中的立即收益指的是团队的整体收益，而非单个 Agent 的收益，因为阵型是相对于整个团队而言的，单个 Agent 角色的改变并不会给这个 Agent 带来收益，只有处于某一特定的阵型中才会给团队带来收益，目前在理论上还无法给出立即收益函数的通用解析表达式，但在具体问题中收益函数都会有具体体现；最后，该模型中并没有加入折扣因子 γ，因为博弈双方采取换阵型一般在某一阶段只进行一次，可以把团队的立即收益看成该阶段行动的总收益，整个博弈过程就是多次重复的多 Agent 单步决策。

下面，我们来介绍在 FMZSG 问题中“均衡”的概念。经典的博弈均衡是指博弈双方实现了各自对博弈结果的满意，所有 Agent 都不想改变自己的策略这样一种相对静止的状态。在 FMZSG 问题中，“均衡”也是指博弈双方都不改变阵型的一种状态，但有时通过行动去打破这种“均衡”往往能够给一方带来正收益，这也是下一节在求解 Anti-Mark 问题时采用的主要思想。因此，FMZSG 问题的主要目标就是决策如何通过行动，改变团队的阵型 f_T，即所有队友或部分队友的角色，从而提高团队收益。比如在 RoboCup 仿真 2D 比赛中，在必要的时刻可以通过改变场上球员的角色分配，去打乱对手的防守体系，从而获得更多的进攻机会；在军用机器人执行战争任务的时候，在某些情况下可能需要机器人改变角色，去完成不是自己原来角色的任务，从而在整体上提高战胜敌人的可能性。

4.2.3　Anti-Mark问题中的应用

在第 1 章我们曾经提到过，RoboCup 仿真 2D 比赛的特殊性在于其世界模型是一个二维的环境，不存在真实足球比赛中的高空球。因此，越来越多的球队开始重视盯人防守，通过切断对手的传球路线来破坏对手的进攻，其中最著名的是来自德国 Osnabrueck 大学的 Brainstormers 和来自中国厦门大学的 AmoyNQ。Brainstormers 通过强化学习的方法来得到 Agent 更富侵略性的防守行为[Riedmiller, 2009]，曾在 2007 年和 2008 年连续两次获得机器人世界杯的冠军；AmoyNQ 通过模糊评价和模糊推理实现异构球员的分配[Jianhuai, 2008]，以此作为盯人防守的基础，曾获得 2008 机器人世界杯的第 4 名以及 2008 中国机器人大赛的冠军。在这两支球队参加的所有比赛中，他们的平均失球数非常少，通过盯人防守建立了非常稳固的防守体系。

WrightEagle 一直以来把进攻作为自己的重要特点，并在多智能体合作方面取得了一定的成果。顾名思义，Anti-Mark 问题所要解决的就是如何在比赛中能够打破盯人防守球队的防守体系，从而获得更多的进攻机会，去赢得比赛。下面，我们将用基于阵型的零和马尔可夫博弈的理论模型来对 Anti-Mark 问题进行建模，以介绍该模型的具体应用：

- R：对应一个球队阵型中的若干角色，锋线上分为前锋、中场和后卫等，位置上分为左、中和右等。
- T,O：比赛双方各有 12 个 Agent，其中 11 个 Agent 在场上比赛，另外一个 Agent 被称为在线教练，在线教练与人类足球的教练类似，可以一定程度上指导其他 Agent 去完成比赛，但并不能完全决定其他 Agent 的行动，而且在线教练和其他 Agent 之间的通讯也是受 Server 限制的；
- F_T,F_O：代表比赛双方所采用的阵型，这里的阵型不仅仅包括人类足球中的 4-3-3 或 4-4-2 等球员站位方法，如果每个球员站位方法中球员之间的位置不同，也属于不同的阵型；
- A,O：在 RoboCup 仿真 2D 球队中，改变阵型有两种方式：一种是依靠教练，即教练决策出新的阵型后，通过通讯告知所有 Agent 执行转换角色的行动；一种是 Agent 之间通过协商的方式进行角色转换，从而改变阵型；
- P：对于上面介绍的两种改变阵型的方式，他们都带有不确定性，首先，通讯的受限可能会导致通讯失败，以及 Agent 体力的限制导致 Agent 无法到达新阵型的位置，从而不能让阵型变为与预期完全一致，其次，由于对手同时也在执行行动，可能会出现比赛双方的行动冲突导致不能尽快完成行动，比如多名球员发生碰撞等；
- R'：一场比赛刚开始时，比赛双方都会有一个初始的阵型，这时双方都会有各自的收益。如果我方阵型分配的较为合理，则收益为正，同时导致对手的收益为负。我们的目标是通过改变自己的阵型 F_T，去提高团队的收益，从而获得比赛的主动权。

Anti-Mark 问题是针对盯人防守的球队而言的，在与盯人防守的对手比赛时，如果我们改变自己的阵型，对手由于人盯人而无法识别我们的策略，导致其 f_O 随着我们 f_T 的改变而改变，造成其防守体系暂时被破坏，从而给我方的进攻带来更多机会。本文根据此思路，针对 Anti-Mark 问题，提出了基于阵型变换的启发式求解方法，将在下一节具体介绍。

4.3　Anti-Mark问题的求解

上面已经提到，RoboCup 仿真 2D 比赛平台提供了在线教练这样一个特殊的 Agent，为了简化问题模型和提高团队行动的成功率，在求解 Anti-Mark 问题时，本文采取通过在线教练作为整体决策者来指导 Agent 在特定时期采取阵型变换的方式。本文提出的求解 Anti-Mark 问题的方法包括两个部分，分别是多角色异构分配策略和角色变换策略，将分为两个小节详细阐述。

4.3.1　多角色异构分配策略

在人类足球比赛中，教练经常会通过换人让原来若干球员的角色发生变化，以形成新的进攻组合阵型，令对手难以防范。RoboCup 仿真 2D 比赛的历史上很少出现这种策略，其主要原因是 RoboCup 仿真 2D 平台规定了异构球员，异构球员的选取与角色有关，一旦在比赛期间进行角色转换就意味着放弃了开场前认为的最佳异构分配，对自己不利。为了解决这个问题，并为后面的角色变换策略打下基础，本文提出了多角色异构分配策略。

在 RoboCup 仿真 2D 比赛中，为了更加逼近人类足球的特点，Server 在开场时会随机产生 18 种异构类型，并告知所有 Agent，比赛双方的在线教练在收到异构类型后必须从这些类型中选取合适的类型分配给所有 11 个 Agent 并告知 Server，而且每一种异构类型只能使用一次。WrightEagle 仿真 2D 球队在之前采用了情景采样法进行异构球员的分配，并取得了一定的效果[Changjie, 2008a]。其基本思想是：首先给所有待分配的角色进行优先级排序，表明某一个角色的 Agent 对于整个球队的重要性；然后对所有角色依次进行情景采样，模拟该角色在正常比赛时最可能出现的情景，再将采样结果最好的而且还未被使用过的异构类型分配给该角色的 Agent。

我们以上面介绍的情景采样法为基础，提出了多角色异构分配策略：

1. 设定 3 个参数 N_{FMD}，N_{FM} 和 N_{DM}，分别表示待分配的 10 个 Agent 中（由于守门员的角色比较特殊，在阵型变换时不会考虑，因此异构分配时也不考虑守门员），有 N_{FMD} 个 Agent 的角色可能是前锋（Forward）、中场(Midfielder)或后卫（Defender），有 N_{FM} 个 Agent 的角色可能是前锋或中场，有 N_{DM} 个 Agent 的角色可能是中场或后卫（满足 $N_{FMD}+N_{FM}+N_{MD}=10$）。
2. 对未使用的所有异构类型分别进行三条锋线的场景测试，将三条锋线测试各自的得分相加便是最后的总得分，选取得分最高的前 N_{FMD} 个类型，保存在集合 T_{FMD} 中，然后用类似的方法分别得到集合 T_{FM} 和 T_{MD}。

3. 此时，集合T_{FMD}、T_{FM}和T_{MD}中一共保存了 10 种异构类型，现在需要将它们分配给各个 Agent，由于每个 Agent 都有一个初始的角色，给 Agent 分配异构也就转化成给每个角色分配异构，我们只需要按照基本的“情景采样法”来分配即可。

在上面第 3 步过程中，我们对于每一个优先级，需要进行二次场景测试，跟基本方法的不同之处是，这里的测试候选并不是所有的异构类型，而是前面选出的 10 种异构类型中可以作为该条锋线的异构类型。举一个例子，比如说现在要选一个左前锋，我们只需要对T_{FMD}和T_{FM}两个集合中的异构类型进行左前锋的场景测试即可，选出以后标记该类型已经被用过。优先级的设定同之前的方法类似：对于锋线间的优先级，由于我们是重视进攻的球队，因此先考虑前锋，再考虑后卫，最后是中场；对于锋线之间的优先级，我们主要考虑左右角色的平衡性，不出现某一边太弱的局面。

按照上面的步骤进行之后，我们就得到了一个初步的角色分配方案。同时可以看出，在选出的 10 种异构类型中，可以做前锋的有$N_{FMD}+N_{FM}$个，可以做后卫的有$N_{FMD}+N_{MD}$个，可以做中场的有 10 个。这为后面的阵型变换策略打下了基础。

4.3.2　阵型变换策略

在这一节，我们将主要介绍比赛期间在线教练所采用的阵型变换策略来应对盯人防守的球队。该策略就是前面分析的 FZSMG 模型的策略，即通过改变我们团队的阵型来提高团队的收益。阵型变换策略的基本思想是：由在线教练进行整体决策，决策出场上新的阵型方案后，通过通讯告知所有的 Agent 执行行动。

面对盯人防守的球队时，对手的防守球员对我方的进攻球员采取贴身防守，导致进攻球员很难摆脱对手，因此很难轻松的接到队友的传球，阵型变换策略可以一定程度上打乱对手的盯人策略，在对手意识到之前抓住机会破门得分。然而，由于对手在过了一定时间之后会识别出我们的新阵型，从而改变其f_O，导致我们的收益下降，对手的收益提高。因此，我们必须继续做出相应的改变，为了能够再次打乱对手的防守。在整场比赛中可能多次采取该策略，因此是多次重复的多 Agent 单步决策。可以看出，在线教练的决策内容包括两个部分：一个是何时执行阵型变换；一个是如何执行阵型变换。

对于第一部分，我们暂时只考虑在我方的一些特殊模式执行阵型变换。因为阵型变换是一个比较花时间的过程，而我方特殊模式时有 200 周期的可控时间。比赛开始之后，我们所有的 Agent 按照在线教练初始分配的角色开始比赛，

在线教练的世界模型中会实时的统计对方进攻时的阵型。比如，对方的哪几名球员处于前锋位置，哪几名球员处于中场和后卫等。等遇到我方特殊模式时，在线教练会判断该特殊模式开始后，对方的阵型是否还与其进攻的阵型保持一致或者基本一致。如果是，说明对手已经意识到我们的新阵型，并调整了他们的盯人策略，这时就执行阵型变换；如果不是，说明说明对手还没有调整他们的盯人策略，因此在这个特殊模式放弃执行。

下面我们来介绍第二部分，即如何执行阵型变换，我们提出了一个启发式的求解方法。我们的目的是打乱对手的盯人防守，在对手重新布防之前进行射门得分。从博弈的角度讲，就是我们在博弈的过程中取得先机，对手将因为行动的负收益而导致局势上的劣势。除去守门员之外，我们的目的是让每一个 Agent 的角色都发生改变，这样可以尽可能的给对手的防守造成混乱。实际上，如果把 10 个角色看成书架上的 10 个位置，把 10 个 Agent 看成 10 本书，这就变成了概率论中著名的“错排问题”。错排公式为（$[x]$为取整函数）：

$$D_n = [n!/e + 0.5] \tag{4.1}$$

我们需要从所有错排方案中选择一个作为我们本次博弈决策的目标阵型。由于我们是针对盯人防守的球队，在这里主要考虑两方面的启发式信息：一是所有 Agent 角色切换的范围尽可能大，一是行动的时间尽可能短。这两点可以保证我们在博弈中获得较大的立即收益。

我们给每一个角色指定了一个坐标，用以说明该角色在整个阵型中的位置信息，图 4.2示意了最常见的 4-3-3 阵型的角色坐标。

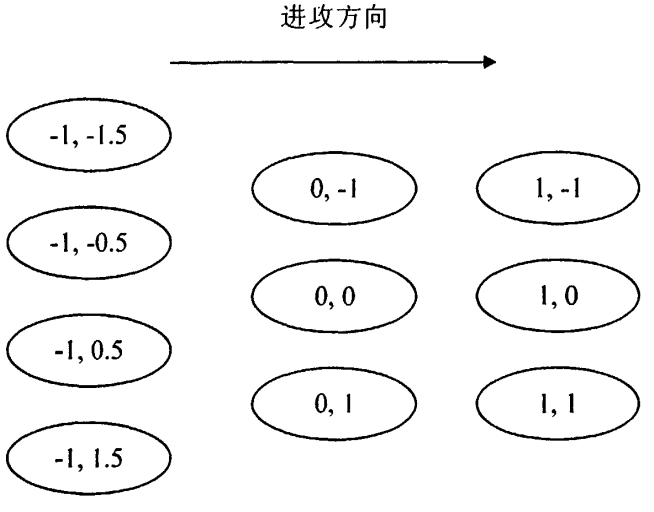

图 4.2　4-3-3 阵型的角色坐标

对于角色r，我们用X_r表示其水平方向坐标，用Y_r表示其竖直方向坐标。比如对于4-3-3阵型中的中前锋，$X_r=1$，$Y_r=0$。定义初始的角色分配方案为：$R_0=(r_{00},r_{01},...,r_{0n})$，$r_{0i}$表示第$i$个Agent的初始角色。定义所有错排方案的集合为：$R=\{R_1,R_2,...R_n\}$，$R_i=(r_{i0},r_{i1},...r_{in})$，$r_{ij}$表示第$i$种错排方案中第$j$个Agent对应的角色。

对于每一个可能的错排方案R_i，我们定义它的距离得分为：

$$S_{R_i}=\sum_{r_{ij}\in R_i}(fabs(X_{r_{ij}}-X_{r_{0j}})+fabs(Y_{r_{ij}}-Y_{r_{0j}})) \tag{4.2}$$

S_{R_i}越大，表明其所对应的错排方案相对于初始的角色分配变化越大，即越容易打乱对手的防守。

另外，我们必须考虑到所有Agent完成阵型变换的时间，因为越快的完成，将对我们较快的组织进攻更有利。在线教练的决策体系中，由于Server发给在线教练的世界状态是无噪声的，因此在线教练始终维护着每周期各个Agent的体力等相关参数。这样，对于一个可能的错排方案R_i，我们可以计算第j个Agent到达目标位置所花的时间T_{ij}（在保持一定体力的情况下），取最长时间作为我们的时间花费：

$$T_{R_i}=\max_j(T_{ij}) \tag{4.3}$$

在计算出任何一个错排方案R_i的S_{R_i}和T_{R_i}后，最优错排方案R^*可以通过下式获得：

$$R^*=\arg\max_{R_i}\{\alpha S_{R_i}+\beta\frac{1}{T_{R_i}}\} \tag{4.4}$$

式(4.4)中的α和β是两个表示权重的参数，需要根据不同的对手通过调试来最终确定，αS_{R_i}和$\beta\frac{1}{T_{R_i}}$可以看成是采取阵型变换策略所得到的立即收益的两个部分。错排方案确定后，新的阵型也就确定了，最后由在线教练统一发给所有Agent。Server规定特殊模式时在线教练的语言可以被所有Agent立即听到，因此所有Agent会立即跑向自己新角色的本位点，待所有人都就位以后发球的队友开球，进入正常比赛模式。对手的防守往往会在这时出现较大的漏洞，从而给我们制造了较多的进攻机会。从博弈的角度来讲，我们通过这一个特殊模式的行动，获得了正的立即收益，在接下来的比赛过程中将会处于优势。

4.4　实验结果及分析

本章所介绍的求解 Anti-Mark 问题的最初想法是在我们备战 2009RoboCup 机器人世界杯的过程中逐渐形成的，世界杯前我们在 WE2009 仿真 2D 球队决策系统中实现了该方法。在世界杯 WE2009 参加的所有比赛中，我们一共 2 次与盯人防守的球队相遇：第 1 次是在第 2 轮小组赛上与 AmoyNQ 相遇，从战略上考虑，为了不提前暴露实力，我们并没有采取阵型变换的策略，结果以 0 比 1 的比分输给 AmoyNQ；第 2 次是在八强赛的第一场与 Brainstormers 相遇，在这场比赛中，我们采取了阵型变换的策略，起到了非常好的效果，以 4 比 0 的比分获胜，Brainstormers 在本届世界杯之前的比赛中仅仅丢过 1 球。因为该场比赛的失利，卫冕冠军 Brainstormer 最终只名列第四名，这是该队自 2000 年以来首次被排斥在世界三强之外。

在世界杯结束后，为了能够进一步测试本文所提方法的效果，我们进行了更充分的实验。实验的硬件环境为：AMD Phenom(tm) II X4 955 Processor，4GB Memory，软件环境是：Ubuntu 9.10，rcssserver-13.2.2（RoboCup 官方于 2009 年 6 月 19 日发布）。本文的实验中包括 WE2009 的如下三个版本：

- WE2009：没有针对 Anti-Mark 问题做任何特殊的策略；
- WE2009*：针对 Anti-Mark 问题，在 WE2009 的基础上采用了本文所提出的基于阵型变换的启发式方法；
- WE2009^{s}：针对 Anti-Mark 问题，在 WE2009 的基础上采用了随机变换阵型的方法，即不考虑本文所提方法中使用的两方面启发式信息。

本文将 WE2009 的三个版本分别与 2009RoboCup 机器人世界杯中成绩最好的两支盯人防守的球队进行比赛（Brainstormers 与 AmoyNQ 分别获得世界杯的第四名和第五名）。为了排除单场比赛的随机性，WE2009 的三个版本与两支对手球队分别各打了 100 场比赛。

表 4.1 为与Brainstormers比赛的测试结果。

表 4.1　与 Brainstormers 比赛的测试结果

Team	Points	Scored	Conceded	Games	Win	Fail	Draw
WE2009	221	215	71	100	67	13	20
WE2009*	274	400	65	100	88	2	10
WE2009^{s}	253	226	44	100	80	7	13

表 4.2为与AmoyNQ比赛的测试结果。

表 4.2　与 AmoyNQ 比赛的测试结果

Team	Points	Scored	Conceded	Games	Win	Fail	Draw
WE2009	212	225	97	100	66	20	14
WE2009*	295	435	61	100	98	1	1
WE2009^{5}	231	272	70	100	72	13	15

从测试结果可以看出，对于测试中的世界上盯人防守实力最强的两支对手球队 Brainstormers 和 AmoyNQ，在采用了本文所提出的基于阵型变换的启发式方法之后，我们球队的胜率分别由 67%和 66%提高到 88%和 98%，进球数分别由 215 和 225 提高到 400 和 435，可见本文提出的方法在求解 Anti-Mark 问题时效果比较明显，进球数的明显增多表明效果提高的主要原因是由于球队进攻能力的提高。另外，在采用了随机变换阵型的方法之后，我们球队的比赛效果也有一定程度的提高，从而说明了两个主要问题：首先，阵型变换在求解 Anti-Mark 问题时的确是一个较为可行的方法，即使采用随机策略，也能使我们在博弈均衡的状态时采取行动后获得一个正收益；其次，随机变换阵型的比赛效果与本文提出的启发式方法的比赛效果还有一定差距，说明本文在求解策略时所利用的两方面启发式信息起到了一定的效果，虽然还无法在理论上证明该问题的最优策略，但给未来的继续研究打下了基础。

本章所提出的方法是对 FZSMG 问题的一个初步尝试，未来的工作包括：从理论上继续对 FZSMG 进行分析，包括收益函数的完善和策略求解的通用算法等，因为阵型变换的思想给博弈中的多智能体合作提供了一种新思路，所以希望能将这种思想应用到更多其它领域；本文目前提出的方法相对来说比较单一，计划在后面对其继续完善，包括角色改变的同时改变阵型，非特殊模式时尝试小范围的角色改变，以及和 Agent 跑位相结合等；另外，当对手已经知道我们会采取该方法时，我们如何应对对手的改变也是值得考虑的重要问题，从宏观上讲这也是一个不断博弈的过程。

4.5　小结

本章介绍了本文的第三个主要工作。首先，本章介绍了博弈论的相关背景知识，包括零和博弈与零和马尔可夫博弈的概念；其次，本章分析了基于阵型

的零和马尔可夫博弈的理论模型，并以此为基础对 RoboCup 仿真 2D 比赛中的 Anti-Mark 问题进行了描述；然后，本章介绍了一个基于阵型变换的启发式求解方法，用来求解 Anti-Mark 问题，实验结果表明，本文所提出的方法使球队在与盯人防守的对手比赛时取得了较好的效果。

第 5 章 总结与展望

5.1 总结

人工智能被认为其主要目标是构造可以决策出智能行为的 Agents，即这些 Agents 能够在多方面再现人类可以做出的智能行为。马尔可夫决策过程可以用来描述和处理大规模不确定性环境下的 Agent 决策问题。本文以马尔可夫决策过程的相关理论为基础，以 RoboCup 仿真 2D 比赛为实验平台，对 Agent 决策相关问题进行了研究。

首先，本文重构并实现了一个完整的 RoboCup 仿真 2D 球队决策系统 WE2009。该系统以部分可观察随机博弈（POSG）的模型为理论基础，包括信息处理、高层决策和行为执行三个模块。特别是高层决策模块，采用基于独立行为生成器的结构设计，不仅可以充分利用 Agent 的决策时间，而且可以提高团队合作的效率。

其次，本文针对一类特殊的 Agent 决策问题，提出了行动驱动的马尔可夫决策过程（ADMDP）的概念。本文分析了 ADMDP 的理论模型，提出了 ADMDP 相关问题的求解方法。该方法采取离线值迭代与在线搜索相结合，在本文中用来求解 RoboCup 仿真 2D 比赛中的不离身带球问题，使 Agent 的带球性能有了较大的提高。

最后，本文从博弈论的角度提出了一类特殊的马尔可夫博弈，即基于阵型的马尔可夫博弈（FZSMG）。本文分析了 FZSMG 的理论模型，并以此为基础来描述 RoboCup 仿真 2D 比赛中的 Anti-Mark 问题。针对 Anti-Mark 问题，本文提出了一个基于阵型变换的启发式求解方法，使球队在与盯人防守的对手比赛时取得了较好的效果。

本文的所有工作都是基于 WE2009 实现的，WE2009 在完成后已经代表中国科学技术大学参加了 2009RoboCup 机器人世界杯（奥地利格拉茨）和 2009 中国机器人大赛（中国大连）两次重要比赛，并且全部获得冠军。

5.2 展望

尽管本文在上述三个方面进行了深入的研究并取得了一定的成果，但是，基于马尔可夫决策过程的 Agent 决策问题仍然有很多挑战性的工作。以本文的工作为基础，作者认为未来的工作可以包括以下几个方面：

- WE2009 具有非常好的结构特性，未来需要对其更加完善，使其能够应用于更多求解 Agent 决策问题的相关领域；
- 求解连续状态空间的 MDP 一直是难于解决的问题，未来需要对如何更好地表示连续状态空间进行更多的研究，从而提高求解该类问题的算法效率；
- 行动驱动的马尔可夫决策过程可以很好的描述一类特殊的 Agent 决策问题，未来需要继续对其进行理论分析，并希望该模型能够应用于更多的实际 Agent 决策问题中；
- 阵型变换的思想给博弈中的多智能体合作提供了一种新思路，未来需要从理论上继续对 FZSMG 进行分析，包括收益函数的完善和策略求解的通用算法等，希望能将这种思想应用到更多其它领域；
- 目前求解 Anti-Mark 问题的方法相对来说还比较单一，未来需要对其继续完善，使其能够随着仿真 2D 的发展而对整个球队发挥更重要的作用。

目前，我们正在备战 2010RoboCup 机器人世界杯，上面提到的一些后续工作也正在进行，希望 WrightEagle 能够在今年的世界杯比赛中蝉联冠军！

参考文献

[Aijun, 2010] Aijun Bai, Jing Wang, Guanghui Lu, et al. WrightEagle 2D Soccer Simulation Team Description 2010. RoboCup International Symposium 2010, June, 2010.

[Barto, 1995] Barto A G., Bradtke S J, Singh S P. 1995. Learning to act using real-time dynamic programming. Artif. Intell., 72, 81-138.

[Bellman, 1957] R. Bellman. A Markovian Decision Process. Journal of Mathematics and Mechanics 6, 1957.

[Changjie, 2008a] Changjie Fan. Research on Planning Based on Markov Decision Theory. A dissertation for doctor's degree of University of Science and Technology of China, 2008.

[Changjie, 2008b] Changjie Fan and Xiaoping Chen. Optimal Action Criterion and Algorithm Improvement of Real-Time Dynamic Programming. Journal of Software, 2008, 19(11): 2869-2878.

[Cohen, 2005] Cohen S, Maimon O. Reinforcement-Learning: An Overview from a Data Mining Perspective. The Data Mining and Knowledge Discovery Handbook 2005: 469-486.

[Colin, 2007] Colin McMillen and Manuela Veloso. Thresholded Rewards: Acting Optimally in Timed, Zero-Sum Games. In Proceedings of AAAI'07, Outstanding Paper Award, Vancouver, Canada, July 2007.

[Craig, 1999] Craig Boutilier, Thomas Dean and Steve Hanks. Decision-theoretic planning: structural assumptions and computational leverage. The Journal of Artificial Intelligence Research, 1999, 11: 1-94.

[Daniel, 2000] Daniel W. Stroock. An Introduction to Markov Processes (Graduate Texts in Mathematics). Springer Berlin Heidelberg New York, 2000.

[Drew, 1991] Drew Fudenberg, Jean Tirole. Game Theory. MIT Press, August 1991.

[Ellen, 1994] Ellen Germain. Software's special agents: Tired of sifting through electronic mail, searching databases and scanning networks for interesting news? An intelligent agent could be what you need. New Scientist, 1994, 142(1920): 19-20.

[Eric, 2004] Eric A. Hansen, Daniel S. Bernstein and Shlomo Ziberstein. Dynamic Programming for Partially Observable Stochastic Games. 19th National Conference on Artificial Intelligence (AAAI-04).

[Eugene, 2002] Eugene A. Feinberg and Adam Shwartz. Handbook of Markov Decision Processes (Methods and Applications). Kluwer 2002.

[Feng, 2007] Feng Wu and Xiaoping Chen. Solving Large-Scale and Sparse-Reward

DEC-POMDPs with Correlation-MDPs. Proceedings of RoboCup International Symposium 2007. Atlanta, America, July 2007.

[Geffner, 1998] Geffner, H. Modelling Intelligent Behaviour: The Markov Decision Process Approach. In Proceedings of the 6th Ibero-American Conference on Ai: Progress in Artificial intelligence (October 05 - 09, 1998).

[Jianhuai, 2008] Jianhuai Cai, Xiao Lin, Wuyi Yu, et al. Application of Fuzzy Evaluation and Inference in RoboCup. Knowledge Acquisition and Modeling Workshop, 2008, pages 593-596, 21-22 Dec. 2008.

[Joshua, 2005] Joshua Grass and Shlomo Zilberstein. Programming with Anytime Algorithms. In Proceedings of the IJCAI-95 Workshop on Anytime Algorithms and Deliberation Scheduling, 2005.

[Ke, 2007] Ke Shi. Research and Application of Anytime Algorithm in RoboCup 2D Soccer Simulation Team. A dissertation for bachelor's degree of Hefei University of Technology, 2007.

[Ke, 2009] Ke Shi, Aijun Bai, Yunfang Tai, et al. WrightEagle2009 2D Soccer Simulation Team Description Paper. RoboCup International Symposium 2009, July, 2009.

[Ke, 2010] Ke Shi and Xiaoping Chen. Action-Driven Markov Decision Process and the Application in RoboCup. Journal of Chinese Computer Systems, in press, 2010.

[Kitano, 1997] Kitano, H., Asada, M., Kuniyoshi, Y., Noda, I., abd H. Matsubara, E.O.: RoboCup: A Challenge Problem for AI. AI Magazine 18 (1997) 73–85.

[Lane, 1994] Lane D M, Mcfadzean A G. 1994. Distributed problem solving and real-time mechanisms in robot architectures. Engineering Applications of Artificial Intelligence Journal. 7(2): 105-117.

[Leslie, 1996] Leslie Pack Kaelbling, Michael L. Littman and Andrew W. Moore. Reinforcement learning: A survey. Journal of Artificial Intelligence Research, 1996, 4: 237-285.

[Li, 2005] Li Li-hong and Michael L. Littman. Lazy Approximation for Solving Continuous Finite horizon MDPs. In AAAI-05, pages 1175-1180, Pittsburgh, PA, 2005.

[Littman, 1994] Littman, M. L. Markov games as a framework for multiagent reinforcement learning. Proceedings of the 11th International Conference on Machine Learning (ML-94). Morgan Kaufmann.

[Lloyd, 1953] Lloyd Shapley, Stochastic games, Proc. Nat. Acad. Sciences, 39:1095-1100, 1953.

[Luiz, 2007] Luiz A. Celiberto Jr., Carlos H. C. Ribeiro, Anna Helena Reali Costa, etc. Heuristic Reinforcement Learning applied to RoboCup Simulation Agents. Proceedings of RoboCup International Symposium 2007, Atlanta, July 2007.

[Mao, 2003] Mao Chen, Klaus Dorer, Ehsan Foroughi, et al. Robocup Soccer Server manual 7.07, February 11, 2003. RoboCup Federation.

[Martin, 2005] Martin L. Puterman. Markov Decision Processes: Discrete Stochastic Dynamic Programming. Wiley-Interscience; 1 edition (March 3, 2005).

[Michail, 2002] Michail G. Lagoudakis and Ronald Parr. Value Function Approximation in Zero-Sum Markov Games. In Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence, 2002.

[Nicolas, 2002] Nicolas Vieille. Stochastic games: Recent results. In Handbook of Game Theory, 1833–1850. Elsevier Science, 2002.

[Nikos, 2004] Nikos Vlassis, R. Elhorst, J. R. Kok. Anytime Algorithms for Multiagent Decision Making Using Coordination Graphs. In Proc. Intl. Conf. On Systems, Man and Cybernetics, 2004.

[Osborne, 1994] Osborne, M. and A. Rubinstein. A Course in Game Theory, Cambridge and London: The MIT Press, 1994.

[Owen, 1982] Owen, Guillermo 1982. Game Theory: Second edition. Academic Press, Orlando, Florida.

[Parkers, 2003] Parkes, David C. and Singh, Satinder. An MDP-based approach to Online Mechanism Design. In Proc. 17th Annual Conf. on Neural Information Processing Systems, 2003.

[Peter, 1999] Peter Stone and Maria Manuela Veloso, "Task Decomposition, Dynamic Role Assignment, and Low-Bandwidth Communication for Real-Time Strategic Teamwork," Artificial Intelligence, 1999.

[Puterman, 1994] Puterman M. Markov decision processes. John Wiley and Sons, New York, 1994.

[Riedmiller, 2009] Riedmiller, M., Gabel, T., Hafner, R., and Lange, S. 2009. Reinforcement learning for robot soccer. Auton. Robots 27, 1 (Jul. 2009), 55-73.

[Shi, 2001] Shi Li. Research of Learning Problems in Multi-Agent System Based on Fuzzy Neural Network. A dissertation for doctor's degree of Tsinghua University, 2001.

[Shoham, 1993] Shoham Y. 1993. Agent-oriented programming. Artificial Intelligence. 60(1): 51-92.

[Stan, 1996] Stan Franklin, Arthur C. Graesser. Is it an Agent, or Just a Program? A Taxonomy for Autonomous Agents. ATAL 1996: 21-35.

[Stuart, 2003] Stuart Russell and Peter Norvig. Artificial Intelligence: A Modern Approach (Second Edition). Pearson Education, Inc., 2003.

[Thomas, 2007] Thomas Gabel, Martin Riedmiller, and Florian Trost. A Case Study on Improving Defense Behavior in Soccer Simulation 2D: The NeuroHassle Approach. Proceedings of RoboCup International Symposium 2008, July, July 2007.

[Van, 1981] Van Der Wal, J. 1981. Stochastic dynamic programming. In Mathematical Centre Tracts 139. Morgan Kaufmann, Amsterdam.

[Weiwei, 2009] Weiwei Zhang. Research and Design about P2P Incentive Mechanism Based on Game Theory. A dissertation for master's degree of Northwest University, 2009.

[Wooldridge, 1994] Wooldridge, M. J. and Jennings, N. R. Agent Theories, Architectures and Languages: A Survey. In: ECAI94 Workshop on Agent Theories Architectures and Languages, Amsterdam, The Netherlands. pp. 1-32.

[Wooldridge, 1997] Wooldridge M. and Fisher M.. Agent-based software engineering. IEEE Proceedings Software Engineering. 1997, 144(1): 26-37.

[Yang, 2000] Yang Gao, Zhihua Zhou, Jiazhou He, et al. Research on Markov Game-based Multi-Agent Reinforcement Learning Model and Algorithms. Journal of Computer Research and Development, 2000, 37(3): 257-263.

[Zixing, 2003] Zixing Cai and Guangyou Xu. Artificial Intelligence: Principles and Applications (Third Edition). Tsinghua University Press, 2003.

致谢

值此论文完成之际，我也即将结束三年的硕士研究生学习和生活。在此，向所有曾给予我帮助和关心的老师、同学、朋友和家人表示衷心的感谢！没有你们的支持，我就不会取得现在的成绩！

感谢我的导师陈小平老师！我在合肥工业大学上本科四年级的时候，陈老师就给了我进入蓝鹰机器人团队学习和锻炼的机会，并在我完成本科毕业论文的过程中给与了悉心的指导。在我正式来到中国科学技术大学读研后，陈老师给了我担任蓝鹰仿真 2D 机器人足球队队长的机会，并在学习和研究中给予了无微不至的关心。在三年来备战机器人比赛的过程中，以及平常的科研工作中，自己的每一个进步都离不开陈老师的鼓励、指导和关心。在我们因为不重视细节和经验不足而屈居世界亚军后，陈老师认真的帮我们分析原因，给我们以鼓励；在我们努力拼搏夺回阔别 2 年的世界冠军后，陈老师仍然严格要求我们，给我们提出了更高的要求，并继续给我们以深入的指导；在我开始理论研究工作准备毕业论文的过程中，陈老师在选题、研究方法、论文写作等各个方面给我以关心，让我更深的理解了研究与工程的不同之处，以及理论研究的重大意义所在。总之，陈老师严谨的学术态度，忘我的工作精神和平易近人的作风令我钦佩不已，这些潜移默化的熏陶也必将使我未来的工作和生活收益。

感谢计算机科学与技术学院的老师们！他们是许胤龙老师、黄刘生老师、岳丽华老师、王煦法老师、陈恩红老师、熊焰老师、邵晨曦老师、金培权老师、刘贵全老师、唐珂老师、于天顺老师、卢贤若老师、张荣老师和钱海老师等。他们认真的授课态度和博学的知识让我了解了计算机学科其他领域的研究成果和最新进展，开阔了我的研究思路；他们在行政工作中对学生的关心和负责也使我能够将更多的精力投入到学习和研究中。

感谢蓝鹰仿真 2D 机器人足球队的前队友和现队友！他们是范长杰、吴锋、王继良、蔡景南、章宗长、刘腾飞、柏爱俊、台运方、王文奎、王宇航、祝元宠、卢光辉、张吴翀和王静等。蓝鹰 2D 的发展是所有新老队员共同努力的结果，和他们的每一次备战过程都会让我终身难忘。另外，范长杰师兄和吴锋师兄在研究工作中对我的指导给我的论文写作起到了很大的帮助。

感谢多智能体系统实验室的所有老师和同学！他们是杨斌老师、宋志伟老师、姜节汇、徐凯、刘飞、李欢、刘津甦、吉建民、杨方凯、范正洁、薛峰和靳国强等。实验室活泼的研究氛围是大家共同创造的，和他们在实验室的学习和生活中充满了无尽的快乐。

感谢我的妻子！在三年的异地恋过程中，我们虽然不在同一个地方，但她在工作上给我以支持，在生活上给我以关心，所有这些都成为我不断奋斗的动力。

最后，谨以此文先给我最亲爱的父母！从我记事起，父母的爱就给我留下了深刻的印象。在我的成长过程中，他们在各个方面对我严格要求，让我从一个不懂事的孩子成长为一个能够不断适应社会的青年。在我离开家乡在合肥求学的 7 年时间中，他们的每一个电话都能让我感到无尽的关怀，他们的每一次教导都能使我更加明确前进的方向。父母是我完成学业最坚强的后盾，向他们表示出衷的感谢，并祝愿他们永远身体健康！

石轲

2010 年 4 月写于中国科学技术大学多智能体系统实验室

攻读学位期间发表的学术论文与取得的其他研究成果

学术论文：

[1] 石轲，陈小平. 行动驱动的马尔可夫决策过程及在 RoboCup 中的应用. 小型微型计算机系统. （已录用）

[2] **Ke Shi**, Aijun Bai, Yunfang Tai, et al. WrightEagle2009 2D Soccer Simulation Team Description Paper. RoboCup 2009: Robot World Cup, July, 2009.

[3] **Ke Shi**, Tengfei Liu, Aijun Bai, et al. WrightEagle2008 2D Soccer Simulation Team Description Paper. RoboCup 2008: Robot World Cup, July, 2008.

其他研究成果：

[1] 2009RoboCup 机器人世界杯仿真 2D 比赛冠军，奥地利格拉茨，2009 年 7 月。

[2] 2008RoboCup 机器人世界杯仿真 2D 比赛亚军，中国苏州，2008 年 7 月。

[3] 2008 中国机器人大赛仿真 2D 比赛亚军，中国中山，2008 年 12 月。

[4] 2007 中国机器人大赛仿真 2D 比赛冠军，中国济南，2007 年 10 月。