第4卷 第3期 2022年6月
Vol.4 No.3 Jun.2022

专刊：先进智能计算平台及应用（下）
Special Issue: Advanced Intelligent Computing Platform and Application

ISSN 2096-742X
CN 10-1649/TP

文献CSTR:
32002.14.jfdc.
CN10-1649/
TP.2022.03.005

文献DOI：
10.11871/jfdc.issn.
2096-742X.2022.
03.005

文献PID：
21.86101.2/jfdc.
2096-742X.2022.
03.005

页码：66-77

获取全文

基于蒙特卡洛树搜索的通用博弈系统的构建与优化研究

梁思立[1]，姜桂飞[1*]，陈泰劼[2]，邓益超[3]，战瑀璠[4]，张玉志[1]

1. 南开大学，软件学院，天津 300350
2. 香港大学，工程学院，香港特别行政区 999077
3. 新加坡国立大学，计算机学院，新加坡 117417
4. 南开大学，金融学院，天津 300381

摘　要：【背景】作为人工智能的主要研究领域，通用博弈策略（General Game Playing，简称GGP）旨在构建具有通用智能的博弈系统。这些系统能够基于给定的博弈规则在没有人为干涉的情况下成功地进行多个甚至是全新构造的博弈。【目的】与专门的博弈系统不同，通用博弈系统所使用的策略生成算法并不针对特定博弈，而是能够根据给定的博弈规则自动生成博弈策略的具有通用性的算法。GGP发展至今已成为检测人工智能水平，特别是通用智能发展的重要研究领域。如何构建高效的通用博弈系统是GGP研究的主要问题。【文献范围】通用博弈策略的生成算法是构建通用博弈系统的关键技术。目前所使用的主流算法是蒙特卡洛树搜索算法及其变种。这类算法在工作过程中并不依赖特定的博弈信息，因而被广泛地应用于GGP领域。然而，由博弈规则推导出来的关于博弈的专门信息，往往对建立针对这一博弈的有效决策算法具有重要的作用。【方法】为此，本文通过在蒙特卡洛树搜索算法上增加记忆结构来存储在线博弈过程中的实时信息，用记忆结构中博弈状态的相似状态来估计该状态的好坏，以提高状态评估的准确性。【结果】本文基于这一方法构建了通用博弈系统并对其性能进行了全面地评估。实验结果表明，与原始的蒙特卡洛方法相比，本文所构建的通用博弈系统在决策水平和效率上都有显著提升，特别在双人信息对称的零和回合制博弈中胜率保持在55%以上，且其性能随着博弈规模的增大而显著提升，在Connect 5、Breakthrough等大规模的游戏上有着绝对优势，即达到100%胜率。【结论】这表明本文所提出的方法通过利用博弈的专门信息能够有效地提升蒙特卡洛树搜索算法的性能。

关键词： 通用博弈策略；蒙特卡洛树搜索；算法博弈论；多智能体系统

Optimizing Monte Carlo Tree Search for General Game Playing

LIANG Sili[1], JIANG Guifei[1*], CHEN Taijie[2], DENG Yichao[3], ZHAN Yufan[4], ZHANG Yuzhi[1]

1. College of Software, Nankai University, Tianjin 300350, China
2. Department of Computer Science, University of Hong Kong, Hong Kong 999077, China

基金项目：国家重点研发计划（2021YFB0300104）；国家自然科学基金（61806102）
*通信作者：姜桂飞（E-mail: g.jiang@nankai.edu.cn）

3. School of Computing, National University of Singapore, Singapore 117417, Singapore
4. School of Finance, Nankai University, Tianjin 300381, China

Abstract: **[Background]** General Game Playing (GGP) is concerned with creating intelligent agents that understand the rules of previously unknown games and learn to play these games well without human intervention. **[Objective]** Unlike specialized systems, a general game player cannot rely on algorithms designed in advance for specific games. Such a system rather requires a form of general intelligence that enables it to autonomously generate strategies based on the given game rules. With the decade of development, GGP provides an important testbed for AI, especially artificial general intelligence. The main problem of GGP is how to build an efficient general game player. Strategy generation is the core technique for building a general game player. **[Scope of the literature]** The main algorithms used in previous successful general game players are the Monte Carlo Tree Search (MCTS) algorithm and its variants. **[Methods]** To improve MCTS during the online real-time search, this paper incorporates it with a memory structure, where each entry contains information about a particular state. This memory is used to generate an approximate value estimation by combining the estimations of similar states. **[Results]** Based on this method, we implement and evaluate a general game player. The experimental results show that it can outperform the original Monte Carlo player in a variety of games. Especially, in two-person zero-sum and turn-based games with symmetric information, the built general game player achieves a winning rate of more than 55%, and its performance improves significantly with the increase of the game size, even 100% winning rate in large-scale games such as Connect 5 and Breakthrough. **[Conclusions]** These results have confirmed the feasibility of the proposed method to use game-dependent information for improving the performance of MCTS.

Keywords: general game playing; Monte Carlo tree search; algorithmic game theory; multi-agent systems

引　言

人工智能在近二十年取得了一系列的显著成果。譬如，1997 年 IBM 的超级计算机“深蓝”（Deep Blue）战胜国际象棋冠军 Kasparov；2011 年 IBM 的超级电脑“沃森”（Watson）击败了综艺节目“Jeopardy!”的常胜冠军；2017 年 Google DeepMind 开发的“阿尔法围棋”（AlphaGo）程序战胜国际围棋冠军柯洁。这些成果标志着人工智能在许多特定的领域，如象棋、围棋等，取得了巨大的成功。但是，正如计算机科学家 Feng-hsiung Hsu 所指出的，这些成果对通用人工智能（Artificial General Intelligence）的意义是有限的，因为这些智能系统都是高度专一化、面向单一任务的，并且它们所展现出来的智能很大程度上依赖于程序员对算法的设计 [1]。正如“深蓝”和“沃森”不能下围棋，“阿尔法围棋”不能下象棋或参加综艺节目“Jeopardy!”，甚至达不到最基本的水平，它们的智能并不具有通用性。

为了构建具有通用性的智能系统，斯坦福大学的计算机科学家 Michael Genesereth 等研究者在 2005 年提出了名为通用博弈策略（General Game Playing, GGP）的研究项目，旨在设计能够成功地以人类水准进行任意已知或未知博弈的人工智能系统 [2]。也就是说，它要求智能系统不仅能够进行简单的井字棋游戏（Tic-Tac-Toe），还能够进行围棋、国际象棋等复杂游戏。这类智能系统被称为通用博弈系统（General Game Player）。自 2005 年起，美国人工智能协会通过在国际人工智能顶级会议 AAAI 或 IJCAI 上举办 GGP 竞赛来推动这一项目的研究。到目前为止，这一竞赛已成功举办了 12 届。经过十多年的发展与研究，GGP 竞赛已成为检测人工智能水平，特别是通用智能发展的重要阵地 [3]。

目前通用博弈系统所使用的主流算法是蒙特卡

洛树搜索算法（Monte Carlo Tree Search, MCTS）及其变种[4]。MCTS 算法以其通用性广泛地被用于 GGP 竞赛，然而与针对特定博弈所设计的算法相比，其性能仍有很大的提升空间。一般来说，由博弈规则所推导出的关于这一博弈的专门信息，往往对设计该博弈的有效决策算法具有重要作用，而原始的 MCTS 算法在工作过程并没有结合博弈的专门信息进行决策[5-6]。同时，在 GGP 平台运行中，通用博弈系统的准备时间、行动时间都是十分有限的。由此，如何在实时的在线搜索过程中获取并运用博弈的专门信息来提升 MCTS 算法的搜索效率成为这一领域的研究热点[7]。

另一方面，随着“阿尔法围棋”的巨大成功，强化学习在博弈领域中的作用日益重要[8-9]。近期，一些研究工作提出了结合强化学习技术来提升 MCTS 算法的性能。譬如，深度神经网络可以用于学习博弈的专门信息，并基于此对博弈状态价值进行近似估值，然后再结合 MCTS 算法来提升搜索空间的效率[9]。特别地，Xiao 等研究者利用强化学习中的泛化思想，即相似状态共享信息，提出了记忆增强的蒙特卡洛算法（Memory-Augmented Monte Carlo Tree Search）[10]。该算法在 MCTS 算法的基础上，维护一个记忆结构，用记忆结构中该状态的相似状态来估计该状态的价值，用以提高状态价值估计的准确性。论文的研究结果表明记忆增强的蒙特卡洛算法在理论和实践上都改善了 MCTS 算法的性能[10]。本文尝试将这一方法扩展应用于 GGP 领域，以构建高效的通用博弈系统。实验结果表明，与基于原始的蒙特卡洛方法所构建的通用博弈系统相比，本文所构建的通用博弈系统无论在决策水平还是效率上都有了显著的提升。

本文的主要结构组织如下：第 1 部分主要讨论了与本文密切相关的研究工作；第 2 部分概述了通用博弈策略 GGP 项目，包括 GGP 平台，通用游戏描述语言；第 3 部分重点阐述了蒙特卡洛树搜索算法的优化方案，通用博弈系统的构建和实现；第 4 部分基于 GGP 平台，通过对比实验全面评估了所构建的通用博弈系统的性能水平；第 5 部分对本文的主要工作进行了总结并指出了进一步的研究方向。

1 相关工作

关于通用博弈系统的研究可以追溯到 Pitrat 和 Pell 的工作，他们设计了原则上能够进行任意给定的象棋类棋盘游戏的智能系统[11-12]。然而，这些早期系统局限于特殊的二人棋盘游戏，只能够解决相对狭窄的一类游戏。对通用博弈系统更广泛的研究兴趣是由国际人工智能顶级会议 AAAI 或 IJCAI 上举行的年度 GGP 竞赛所激发的。经过十多年的发展，关于通用博弈系统的研究已成为人工智能领域一个重要的研究主题[13]。

博弈策略的生成算法是构建通用博弈系统的关键技术。在 GGP 发展的初期，以 MiniMax 算法及其变种为主的基于评估的搜索算法一度占据主导地位，如 GGP 竞赛冠军 Cluneplayer 和 Fluxplayer 采用的就是基于评估的搜索算法[14-15]。自从 2007 年冠军 Cadiaplayer 首次采用基于模拟的搜索算法开始，基于模拟的搜索算法逐渐成为构建通用博弈系统的主流算法[16]。其中，MCTS 算法正是基于模拟的搜索算法的典型代表。这一算法使用非平衡扩展的方法，在扩展阶段快速地将大部分未来发展可能较差的分支排除，使得计算机能将更多的算力分配在对未来发展可能更加优秀的分支的探索上。因此，MCTS 算法及其变种是目前通用博弈系统使用的主流算法[7]。

为了改善 MCTS 算法的性能，构建更高效的通用博弈系统，近期一些研究工作尝试将强化学习技术运用到 GGP 领域并取得了初步的结果。譬如，研究者 Benacloch-Ayuso 提出了框架 RL-GGP 用以测试不同的强化算法在 GGP 领域的运用，但其并没有给出关于这些算法的性能评估结果[17]。也有研究工作将时间差分学习（Temporal-Difference Learning）用于改善 MCTS 算法[18]。特别地，有研究者将强

化学习的 Q-learning 与 MCTS 算法相结合应用到 GGP 领域，但实验结果表明，结合后的算法不仅存在收敛速度慢的问题，且决策效果并没有比 MCTS 算法更好 [19]。而 Goldwaser 和 Thielscher 研究者将 AlphaZero 所使用的通用深度强化学习算法扩展应用到 GGP 领域，与 MCTS 算法的变种—上限置信区间算法（Upper Confidence Bound Apply to Tree, UCT）进行对比，实验表明，同深度强化学习算法相结合的 MCTS 算法的决策效果得到了显著的改善 [20]。本论文受论文 [10] 提出方法的启发，进一步推进了这一方向的研究工作。

2 GGP 概述

本部分将对 GGP 的基本框架进行整体介绍。

2.1 GGP 平台

GGP 平台为研究者提供在线竞技评测服务，同时收集大量的博弈实战信息供研究者使用，为研究者设计和改进通用博弈系统提供数据支持。经过十多年的发展，GGP 平台已成为衡量博弈系统智能程度的重要阵地。

具体地，GGP 平台通过博弈主机（Game Manager）向研究者提供数百种博弈的规则描述、比赛记录、图形化展示以及博弈状态数据，并通过网络与玩家通信，接收他们的博弈决策；通用博弈系统在本地运行，通过 GGP 平台所提供的应用程序编程接口与博弈主机相互通信。

在线对弈时，博弈主机是玩家对弈的裁判，其工作流程如图 1 所示。博弈主机首先向 GGP 玩家同时发送博弈开始的信息，包括博弈规则、玩家在博弈中扮演的角色、博弈的准备时间以及行动时间。收到博弈开始的信息，各个玩家有一段时间来理解博弈规则，即准备时间，通常是 30 秒到 120 秒。准备时间结束后，博弈正式进入第一回合。每个回合，博弈主机会告知玩家是否要采取行动。玩家有思考决策时间，即行动时间，一般在 60 秒内。超过行动时间，博弈主机会随机给玩家指派一个行动。在下一个回合时，博弈主机会将上一回合各个玩家采取的行动发送给各个玩家，玩家可以据此更新博弈状态并结合博弈规则推断出当前状态的合法行动。同时博弈主机在接收到所有玩家的行动后会计算出下一回合的博弈状态，并判定这一状态是否为博弈结束状态。如果是结束状态，则告知各玩家博弈结束并给出博弈结果；如果不是，

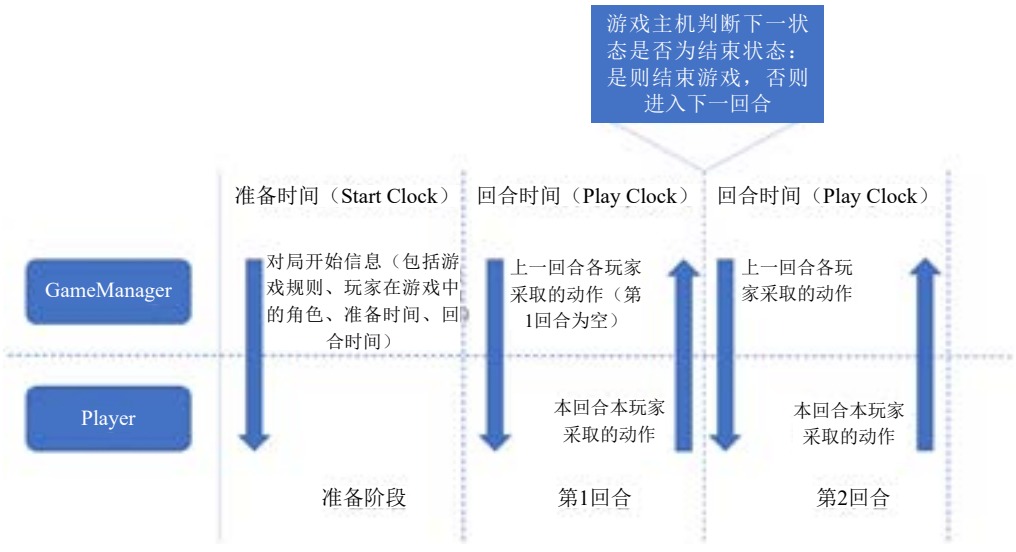

图 1 博弈主机工作流程 [7]

Fig.1 The workflow diagram of the Game Manager[7]

则进入下一个博弈回合。其中，博弈主机与各玩家的信息交互遵循的是 TCP/IP 协议 [7]。

2.2 通用博弈描述语言

GGP 领域的官方语言即通用博弈描述语言（General Game Description Language, GDL）是由斯坦福大学 GGP 项目团队在 2005 年提出的 [21]。

GDL 是一种声明式的逻辑程序语言，使用命题集合来表示状态。对于一个博弈，GDL 定义一个状态命题集合P，用来描述博弈状态的基本信息；在博弈状态集合S与P的幂集2^P之间建立一个单射$f: S \to 2^P$，用来表示博弈状态上的事实信息。对任意一个状态s和任意一个命题p，若有$p \in f(s)$，则p在s上为真，反之则为假。同时，GDL 定义了动作集合A，且任意玩家r的合法动作集$L_{s,r}$都是A的子集。在此基础上，GDL 定义了描述博弈要素的集合，如表 1 所示。

表 1 描述博弈要素的命题集合 [7]

Table 1 The proposition set for representing elements of games[7]

命题	命题为真时的含义
$(role\ r)$	r是一个玩家
$(init\ p)$	状态命题p是初始状态时为真
$(true\ p)$	状态命题p在当前状态下为真
$(legal\ r\ a)$	玩家r在当前状态下a是合法动作
$(does\ r\ a)$	玩家r采取了动作a
$(next\ p)$	状态命题p在下一状态下为真
$terminal$	当前状态时结束状态
$(goal\ r\ n)$	玩家r在当前状态下的得分是n

GDL 语言虽然简单但足够描述任意有穷的完全信息的博弈规则 [21]。值得一提的是，GDL 最初是为了描述完全信息博弈而设计的，最近被扩展成 GDL-II、GDL-III 分别用于描述不同程度的非完全信息博弈，如扑克牌、Bughouse 国际象棋等 [22-23]。

3 通用博弈系统

本部分将首先介绍 GGP 领域的主流算法 MCTS 算法及其变种 UCT 算法，然后阐述本文所使用的方法，最后探讨如何基于这一方法来构建和实现通用博弈系统。

3.1 蒙特卡洛树搜索算法

MCTS 算法是一种在给定的决策空间中寻找最优决策的方法，通过在决策空间中抽取随机样本，并根据这些样本构建搜索树。对于一些可以被表示为序贯决策树的人工智能领域，比如博弈和规划问题，MCTS 算法已经产生了深远的影响。

MCTS 算法的核心思想是通过迭代来构建搜索树，直到达到某个预定义的预算上限（这个上限通常可以是时间限制、内存限制或迭代次数限制），此时搜索停止，并返回根节点的最佳行动。对于 GGP，搜索树中的每一个节点都代表着博弈状态空间中的一个状态。

在 MCTS 算法的每次迭代中，主要包含选择、扩展、模拟、反向传播四个步骤。在具体阐述每一步骤之前，先对以下符号进行说明：

· 集合S是博弈状态的集合。

· 对任意状态$s \in S$，

$\mathrm{A}(s)$表示状态s下的合法行动的集合。

$\mathrm{N}(s)$表示状态s被访问的次数。

$\widehat{\mathrm{V}}(s)$表示状态s的状态价值，取值在$[0,1]$，在原始的 MCTS 算法中，$\widehat{\mathrm{V}}(s) = \frac{\sum_{i=1}^{t} R_{i,s}}{\mathrm{N}(s)}$，表示$t$次迭代后状态$s$的奖励值的均值，其中$R_{i,s}$表示第$i$次迭代状态$s$的奖励值。

$\delta(s,a)$表示状态 s 选择行动a所到达的状态。

下面对 MCTS 算法所包含的四个步骤进行逐一说明。

（1）选择

从根节点开始，递归地应用选择策略来对搜索树进行遍历，直到达到一个可扩展节点。如果一个节点代表着一个非结束状态，并且它有未被访问过（扩展过）的子节点，则称它是可扩展的。原始的 MCTS 算法的选择策略为：

$$\mathbf{argmax}_{a\in \mathrm{A}(s)}\widehat{\mathrm{V}}\left(\delta(s,a)\right) \quad (1)$$

即选择当前状态下行动价值最大的行动。

（2）扩展

当搜索到达一个可扩展节点后，算法将对其一未被加入搜索树的子节点加入搜索树中，并对该子节点执行模拟和反向传播，完成后进入下次迭代。

（3）模拟

在新节点运行模拟策略直至博弈结束状态，并返回奖励值。模拟策略通常包含两种：一种是均匀随机策略，通过均匀随机选择动作进行对弈直到到达博弈结束状态；另一种则是启发式策略，引入特定博弈的专门信息，极大地降低模拟深度、减少分支，加快收敛的速度，但也会相应的降低搜索空间，有可能会陷入局部最优的困境。

（4）反向传播

将模拟得到的奖励值回传到搜索路径上每一个节点。定义路径$L=\{s_0,s_1,...,s_T\}$，其中，s_0表示起始状态，s_T表示执行模拟的状态，模拟得出的奖励值为R，那么对于每一个$s_t\in L$，有如下更新策略：

$$\mathrm{N}(s_t)\leftarrow \mathrm{N}(s_t)+1 \quad (2)$$

$$\widehat{\mathrm{V}}(s_t)\leftarrow \widehat{\mathrm{V}}(s_t)+\frac{R-\widehat{\mathrm{V}}(s_t)}{\mathrm{N}(s_t)} \quad (3)$$

即相应地更新状态s_t的访问次数与状态价值。

在选择阶段，原始的 MCTS 算法只选择当前行动价值最高的行动，但事实上，当前行动价值低的节点可能在多次迭代后，发现它具有更好的表现，因此为了避免陷入局部最优，需要更好地平衡探索和利用的关系。

多臂老虎机问题（Multi-Armed Bandits）正是一个探究如何平衡探索和利用的研究方向。其中，Auer 等研究者提出的 UCB1 方法（Upper Confidence Bound-1）在该问题上取得了很不错的效果[25]。同时，Kocsis 等研究者还将 UCB1 方法与 MCTS 算法相结合，提出 UCT 算法，将多臂老虎机的平衡探索和利用的思想应用到序贯决策问题[26]。

具体地，不同于 MCTS 算法，UCT 算法在选择阶段不再选取当前行动价值最大的行动，而是选择最大置信上界的行动，即：

$$\mathbf{argmax}_{a\in \mathrm{A}(s)}\left(\widehat{\mathrm{V}}\left(\delta(s,a)\right)+C\sqrt{\frac{\ln \mathrm{N}(s)}{\mathrm{N}\left(\delta(s,a)\right)}}\right) \quad (4)$$

其中，C为控制探索的参数，一般情况下取$\sqrt{2}$。

3.2 蒙特卡洛树搜索算法的优化

在巨大的博弈状态空间和相对有限的搜索时间的情况下，MCTS 算法估计状态价值的方差比较大，无法保证估计值的准确性，而估计值的不准确将误导搜索树的构建，严重降低程序的性能。为了解决上述问题，有研究者提出了记忆增强的蒙特卡洛树搜索算法（M-MCTS），利用相似状态间的共同信息，在在线搜索中提高状态价值估计的准确性[10]。M-MCTS 算法的结构如图 2 所示。

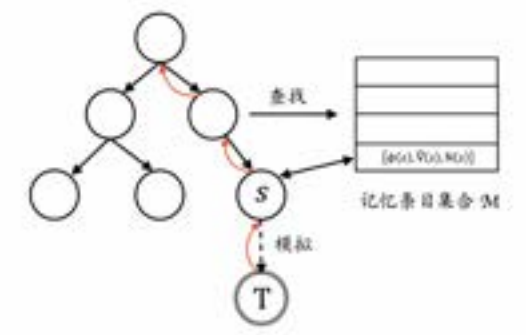

图 2 M–MCTS 算法原理示意图[10]

Fig.2 An illustration of M-MCTS[10]

其中，

特征函数$\phi: S\rightarrow R^D$，将博弈状态S单射到D维的向量。

特征向量距离函数$d(s,x)=-\cos\left(\phi(s),\phi(x)\right)$，用来描述特征向量间的距离。

$\mathrm{N}_{\mathrm{M}}(s)$表示状态$s$在记忆条目集合 $\mathcal{M}$ 中记忆被访问的次数。

$\widehat{\mathrm{V}}_{\mathrm{M}}(s)$表示状态$s$在记忆条目集合 $\mathcal{M}$ 中记忆状态价值。

M-MCTS 算法维护了一个最大条目数为M的记忆条目集合$\mathcal{M}$，$\mathcal{M}$中的每个条目都对应一个搜索树中特定的状态s，储存该状态的三元组信息$\{\phi(s), \widehat{\mathrm{V}}(s), \mathrm{N}(s)\}$，分别为该状态的特征向量、状态价值、被访问次数。$\mathcal{M}$中定义了以下三个操作：

（1）更新（Update）：当搜索树的节点中的信息发生更新时，需要同时更新$\mathcal{M}$中的对应项。

（2）加入（Add）：当搜索树进行扩展时，需要将新节点也对应加入到$\mathcal{M}$中；若$\mathcal{M}$满了，则使用最近最少访问的策略进行替换。

（3）查找（Query）：通过查找操作，找到该状态最相近的K个状态记为$\{s_1, ..., s_K\}$，将这些状态的状态价值$\widehat{\mathrm{V}}$的加权平均作为该状态的记忆状态价值$\widehat{\mathrm{V}}_{\mathrm{M}}$，即：

$$\widehat{\mathrm{V}}_{\mathrm{M}}(s) = \sum_{i=1}^{K} w_i(s)\,\widehat{\mathrm{V}}(s_i) \tag{5}$$

特别地，在 M-MCTS 算法中，使用的是熵正则策略优化（Entropy Regularized Policy Optimization）的方法来确定权重w_i。

M-MCTS 算法的核心思想是用记忆状态值$\widehat{\mathrm{V}}_{\mathrm{M}}$来改善原先的状态价值$\widehat{\mathrm{V}}$，故在选择阶段使用如下策略，以引导生成更有效的搜索树：

$$\widehat{\mathrm{V}}'_{\mathrm{M}}(s) = (1-\lambda_s)\widehat{\mathrm{V}}\big(\delta(s,a)\big) + \lambda_s \widehat{\mathrm{V}}_{\mathrm{M}}\big(\delta(s,a)\big) \tag{6}$$

$$\mathbf{argmax}_{a\in \mathrm{A}(s)}\left(\widehat{\mathrm{V}}'_{\mathrm{M}}\big(\delta(s,a)\big) + C\sqrt{\frac{\ln \mathrm{N}(s)}{\mathrm{N}\big(\delta(s,a)\big)}}\right) \tag{7}$$

3.3 通用博弈系统的建立

本节介绍如何将 M-MCTS 算法的思想用于通用博弈系统的构建。

为此，本文对 M-MCTS 算法进行了如下修改，用以构建通用博弈系统：

（1）将 UCB1 方法加入到 M-MCTS 算法中，同时完全使用记忆状态值$\widehat{\mathrm{V}}_{\mathrm{M}}$而不是状态价值$\widehat{\mathrm{V}}$来引导搜索树的构建，故选择策略为：

$$\mathbf{argmax}_{a\in \mathrm{A}(s)}\left(\widehat{\mathrm{V}}_{\mathrm{M}}\big(\delta(s,a)\big) + C\sqrt{\frac{\ln \mathrm{N}(s)}{\mathrm{N}\big(\delta(s,a)\big)}}\right) \tag{8}$$

（2）在进行反向传播更新时，不再使用原先的增量均值的更新，而是使用以下更新策略：

$$\widehat{\mathrm{V}}(s_t) \leftarrow \widehat{\mathrm{V}}(s_t) + \sigma\left(R - \widehat{\mathrm{V}}(s_t)\right) \tag{9}$$

其中，σ为更新比例，取值在$[0,1]$，接近 1 表示更考虑当前奖励，接近 0 表示更考虑历史奖励。

需要说明的是，在具体实现时，本文在查询操作时仅使用其自身的状态价值$\widehat{\mathrm{V}}$。

基于上述方法，本文利用 GGP 平台提供的基础代码来具体实现通用博弈系统。基础代码涵盖了博弈状态机的构建、维护和状态转换等功能，同时提供了设置博弈和展示对战双方博弈状况的可视化界面。所构建的系统将在准备时间内进行对博弈的理解和一些预处理，接受该博弈的 GDL 语言描述，理解博弈规则并生成该博弈的状态机，并在剩余的时间内运行通用博弈系统进行决策。

在基础代码中，主要有两个核心类：一个是状态机类（StateMachine），负责各种博弈状态机的构建和维护，提供博弈结束状态的奖励值、给定角色在当前状态可以选择的合法动作以及执行给定动作后转换到的后继状态；另一个是博弈玩家类（Gamer），提供了获取博弈当前状态、获取博弈当前拥有控制权的角色、获取当前博弈状态机等功能。

具体地，我们需要实现本系统的核心函数——动作选择函数，在该函数内我们将从状态机获取博弈当前状态作为根节点，构建决策树，并在每步时限内运行我们修改的 M-MCTS 算法，递归地进行搜索、扩展、模拟、更新并提取记忆、反向传播，最终选择平均奖励值最高的动作返回。

4 实验结果和分析

本部分以 GGP 平台上提供的原始的蒙特卡洛 MC 方法的通用博弈系统作为基准，通过对比试验全面评估所构建通用博弈系统的性能水平。

4.1 实验平台及环境设置

本文的实验是在 Ubuntu 系统上现实的，实验运行的服务器设备是 Linux 内核，Intel(R) 的 CPU E5-2650，128GB 内存，采用 GGP 平台进行对比实验。

为了全面评估所构建的通用博弈系统的智能水平，本文从以往 GGP 竞赛中选取六组不同类型的博弈游戏进行测试 [27]。具体博弈特征如表 2 所示。

在实验过程中，博弈准备时间统一设置为 30s，通过对行动时间、先后手以及更新比例σ三个参数的设置来全面的评估本文所构建通用博弈系统在六组博弈上的性能表现。具体地，为了保证模拟次数，玩家采取行动的时间分别设置为 15s、30s、45s、60s；由于先后手对博弈，特别是双人零和完全信息博弈的胜率具有影响，实验中每组博弈对弈双方在进行 50 局后互换角色；为了考察最优的记忆更新比例，σ分别设置为 0.5、0.8。

表 2 实验测试游戏

Table 2 Games in the experiments

游戏名称	玩家数目	合作与否	信息	行动顺序
Tic-Tac-Toe (Large) (5×5棋盘)	2	零和	信息对称	回合制
Connect 4 (6×7棋盘)	2	零和	信息对称	回合制
Connect 5 (8×8棋盘)	2	零和	信息对称	回合制
Breakthrough (6×6棋盘)	2	零和	信息对称	回合制
Babel	3	合作	信息对称	同时
Pacman3p (6×6棋盘)	3	合作/零和	非信息对称	回合制/同时混合

4.2 实验结果及分析

Tic-Tac-Toe (Large) 游戏是井字棋的变种，两个玩家 X,O 依次在 5×5 的空格内填写自己的标记，任意一方在横、竖或斜方向五个标记形成一条直线，则获胜。如图 3 所示，相比于基准的通用博弈系统，我们所构建的通用博弈系统性能相对较高，获胜率保持在 55% 以上。同时，随着行动时间的增长，所构建的通用博弈系统的获胜率逐步下降，而更新比例σ的变化对其胜率影响并不明显。

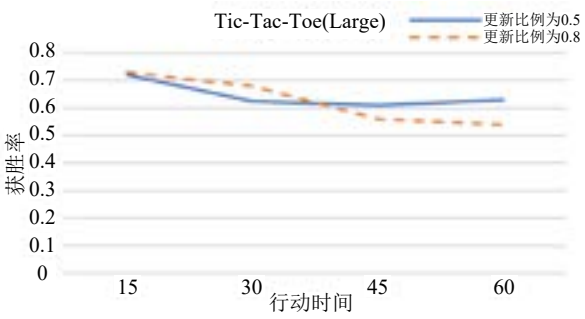

图 3 所构建的通用博弈系统与基准系统在 Tic–Tac–Toe (Large) 游戏上对弈的获胜率相对于行动时间的变化。

Fig.3 The winning rate of our player vs. baseline in Tic-Tac-Toe (Large).

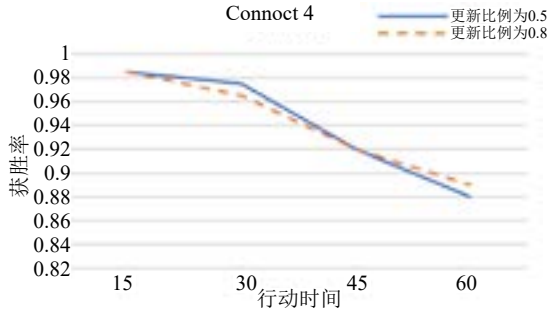

图 4 所构建的通用博弈系统与基准系统在 Connect 4 游戏上对弈的获胜率相对于行动时间的变化。

Fig.4 The winning rate of our player vs. baseline in Connect 4.

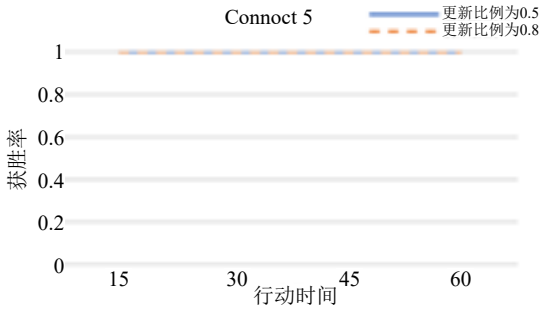

图 5 所构建的通用博弈系统与基准系统在 Connect 5 游戏上对弈的获胜率相对于行动时间的变化。

Fig.5 The winning rate of our player vs. baseline in Connect 5.

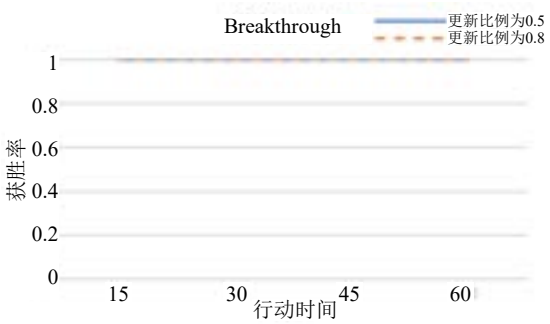

图 6 所构建的通用博弈系统与基准系统在 Breakthrough 游戏上对弈的获胜率相对于行动时间的变化。

Fig.6 The winning rate of our player vs. baseline in Breakthrough.

如图 4 所示，所构建的通用博弈系统在规模增大的游戏 Connect 4 上的优势更加显著，其获胜率最高达 99%，最低在 88% 以上。类似于 Tic-Tac-Toe (Large) 游戏，随着行动时间的增长，所构建的通用博弈系统的获胜率逐步下降，但更新比例σ的变化对其影响并不明显。在游戏规模进一步增大的 Connect 5、Breakthrough 上，所构建的通用博弈系统的获胜率高达 100%，且不受行动时间的增长和记忆更新率的影响。

可见，在双人完全信息零和博弈中，所构建的通用博弈系统的性能会随着游戏规模的增大而显著提升，甚至在 Connect 5、Breakthrough 等规模庞大的游戏上对基于 MC 方法的通用博弈系统有着绝对的优势，即 100% 的胜率，如图 5、图 6 所示。而在 Tic-Tac-Toe (Large)、Connect 4 这些规模相对较小的游戏上，由于搜索空间相对小，基于 MC 方法的通用博弈系统通过随机模拟能在一些场合下达到最优结果，且随着行动时间的增长，其有更大的概率能够搜索到最优的状态；另一方面，对于所构建的通用博弈系统，由于其效率已经较高，能在相对短时间内到达最佳状态，无论是行动时间是否增长其都已达到最优决策，故其性能不会因为时间增长而发生变化。因此，在 Tic-Tac-Toe (Large)、Connect 4 游戏中，所构建的通用博弈系统尽管优势明显但没有达到 100% 胜率，且随着行动时间的增长，其胜率会随之下降。

本文以 Pacman 3P 吃豆人游戏为例评估了所构建的通用博弈系统在多人非对称信息游戏中的性能表现。Pacman 3P 游戏一共三个玩家，其中一个玩家控制 Pacman，另外两个玩家每人分别控制一个 Ghost 联合抓捕 Pacman，控制 Pacman 的玩家要躲避 Ghost，吃掉所有的 Pellet 才能获胜。为了对实验结果有更直观的观察，本文进行了两组实验：首先由基于 MC 方法的通用博弈系统控制 Pacman，所构建的通用博弈系统控制两个 Ghost；然后互换角色。如图 7 所示，所构建的通用博弈系统相比基于 MC 方法的通用博弈系统在 Pacman 和 Ghost 的得分比上具有明显优势，如在更新比例σ为 0.8 的情况下，所构建的通用博弈系统控制的 Pacman 每局游戏平均得分在 0.5 左右，而基于 MC 方法的通用博弈系统控制的 Pacman 得分比则不超过 0.3。由于玩家之间的信息并不对称，游戏的复杂度高，行动时间的增长以及更新比例σ的变化，对所构建的通用博弈系统的性能影响并不明显。

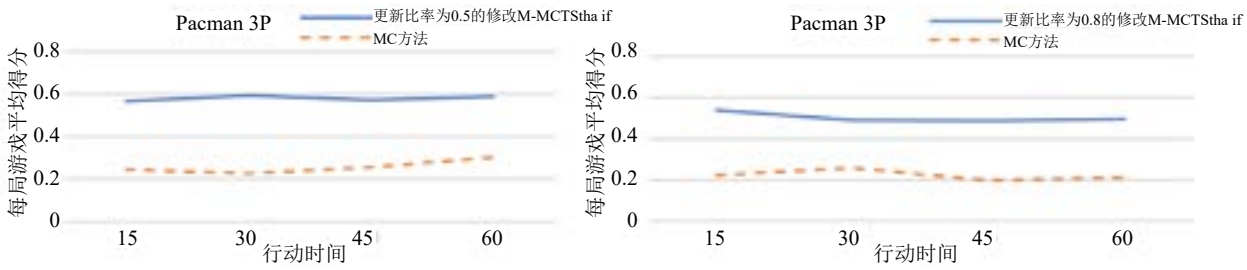

图 7 所构建的通用博弈系统与基准系统在 Pacman 3P 游戏上对弈的平均得分相对于行动时间的变化（左：更新比例 σ =0.5；右：更新比例σ =0.8）。

Fig.7 The average score of our player vs. baseline in Pacman 3P (Left: σ = 0.5 ;Right: σ = 0.8.).

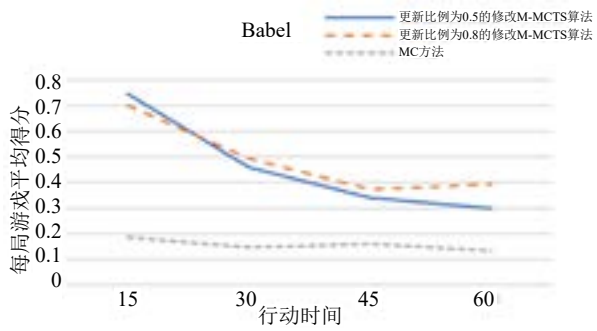

图 8 所构建的通用博弈系统与基准系统在 Babel 游戏上对弈的平均得分相对于行动时间的变化。

Fig.8 The average score of our player vs. baseline in Babel.

最后，以 Babel 游戏为例评估了所构建的通用博弈系统在合作游戏上的性能表现。Babel 是三人合作建高塔的游戏。为了直观地显示对比效果，本文考察了三个玩家都是所构建的通用博弈系统与三个玩家都是基于 MC 方法的通用博弈系统的两种极端情况，具体结果如图 8 所示。从得分上可以看出，整体上都是所构建的通用博弈系统的性能要显著地优于都是基于 MC 方法的通用博弈系统的性能。而随着行动时间的增长，基于所构建的通用博弈系统的性能会逐步下降。其主要原因可能在于作为三人合作游戏 Babel 玩家行动组合规模大，游戏搜索宽度过大，随着时间的增长，所构建的通用博弈系统模拟次数的增加带来记忆数目的增长，甚至达到记忆上限，这使得最新的记忆项目将替代最近最少查询的项目，而这些被替换的项目很可能会导致新近扩展状态的丢失，从而大大减弱了记忆增强的效果。当更新比例σ为 0.5 时，记忆增强算法的性能下降比 0.8 时相对明显，一定程度上支持了这一观点。不过整体而言，同前面游戏的实验结果类似，记忆更新率的变化对记忆增强算法性能的影响并不明显。

5 总结与展望

本文在蒙特卡洛树搜索算法上增加记忆结构以有效地利用博弈的专门信息，提升蒙特卡洛树搜索的性能；基于此构建了通用博弈系统，并依托于 GGP 平台全面地评估了所构建系统的性能。实验结果表明，与 GGP 平台上的 MC 通用博弈系统相比，本文所实现的通用博弈系统无论在智能水平还是决策效率上都有了显著提升。如何进一步提升通用博弈系统在复杂博弈特别是非完美信息博弈上的性能；如何将经典的对抗算法与先进的机器学习，特别是深度学习技术相结合以推动 GGP 领域的发展，通用人工智能研究的深入，无疑具有重要的理论价值和广阔的应用前景。

利益冲突声明

所有作者声明不存在利益冲突关系。

参考文献

[1] Hsu F H. Behind Deep Blue: Building the Computer that Defeated the World Chess Champion[M]. Princeton University Press, 2002: x.

[2] Genesereth M, Love N, Pell B. General game playing: Overview of the AAAI competition[J]. AI magazine, 2005, 26(2): 62-62.

[3] Thielscher M. General Game Playing in AI Research and Education [C]// the German Annual Conference on Artificial Intelligence (KI), Springer, Berlin, Heidelberg, 2011, 7006: 26-37.

[4] Browne C B, Powley E, Whitehouse D, et al. A Survey of Monte Carlo Tree Search Methods[J]. IEEE Transactions on Computational Intelligence and AI in Games, 2012, 4(1): 1-43.

[5] Świechowski M, Park H, Mańdziuk J, et al. Recent Advances in General Game Playing[J/OL]. The Scientific World Journal, 2015:1-22. https://pubmed.ncbi.nlm.nih.gov/26380375/.

[6] Świechowski M, Mańdziuk J. A Hybrid Approach to Parallelization of Monte Carlo Tree Search in General Game Playing[M]// Challenging Problems and Solutions

in Intelligent Systems, Springer, Cham, 2016: 199-215.

[7] 张海峰, 刘当一, 李文新. 通用对弈游戏:一个探索机器游戏智能的领域[J]. 软件学报, 2016, 27(11): 2814-2827.

[8] Mnih V, Kavukcuoglu K, Silver D, et al. Human-level Control through Deep Reinforcement Learning[J]. nature, 2015, 518(7540): 529-533.

[9] Silver D, Huang A, Maddison C J, et al. Mastering the Game of Go with Deep Neural Networks and Tree Search [J]. nature, 2016, 529(7587): 484-489.

[10] Xiao C, Mei J, Müller M. Memory-Augmented Monte Carlo Tree Search[C]// the 32nd AAAI Conference on Artificial Intelligence, 2018: 1455-1461.

[11] Pitrat J. A General Game Playing Program[M]// Findler N and Meltzer B: Artificial Intelligence and Heuristic Programming, Edinburgh University Press, 1971:125-155.

[12] Pell B. Strategy Generation and Evaluation for Meta-Game Playing[D]. University of Cambridge, 1993.

[13] Thielscher M. Systems with General Intelligence: A New Perspective (Invited Talk)[Z]// the 24th AAAI Conference on Artificial Intelligence, 2010.

[14] Clune J. Heuristic Evaluation Functions for General Game Playing[C]// the 22nd AAAI Conference on Artificial Intelligence, 2007: 1134-1139.

[15] Schiffel S, Thielscher M. Fluxplayer: A Successful General Game Player[C]// the 22nd AAAI Conference on Artificial Intelligence, 2007: 1191-1196.

[16] Finnsson H, Björnsson Y. Simulation-Based Approach to General Game Playing[C]// the 23rd AAAI Conference on Artificial Intelligence, 2008: 259-264.

[17] Benacloch-Ayuso J L. RL-GGP[CP/OL]. http://users.dsic.upv.es/~flip/RLGGP/. 2012.

[18] Finnsson H, Björnsson Y. Learning Simulation Control in General Game-Playing Agents[C]// the 24th AAAI Conference on Artificial Intelligence, 2010: 954-959.

[19] Wang H, Emmerich M, Plaat A. Monte Carlo Q-learning for General Game Playing[J/OL]. arXiv preprint arXiv: 1802.05944, 2018.

[20] Goldwaser A, Thielscher M. Deep Reinforcement Lea-rning for General Game Playing[C]// the 34th AAAI Conference on Artificial Intelligence, 2020, 34(02): 1701-1708.

[21] Love N, Hinrichs T, Haley D, et al. General Game Playing: Game Description Language Specification[R]. Stanford Logic Group Computer Science Department Stanford University. 2008.

[22] Thielscher M. A General Game Description Language for Incomplete Information Games[C]// the 24th AAAI Conference on Artificial Intelligence, 2010: 994-999.

[23] Thielscher M. GDL-III: A Description Language for Epistemic General Game Playing[C]// the 26th International Joint Conference on Artificial Intelligence, 2017: 1276-1282.

[24] Méhat J, Cazenave T. A Parallel General Game Player[J]. KI-künstliche Intelligenz, 2011, 25(1): 43-47.

[25] Auer P, Cesa-Bianchi N, Fischer P. Finite-Time Analysis of the Multi-armed Bandit Problem[J]. Machine learning, 2002, 47(2): 235-256.

[26] Kocsis L, Szepesvári C. Bandit-Based Monte-Carlo Planning[C]//European conference on machine learning, Springer, Berlin, Heidelberg, 2006, 4212: 282-293.

[27] Genesereth M, Björnsson Y. The International General Game Playing Competition[J]. AI Magazine, 2013, 34(2): 107-111.

收稿日期：2022 年 2 月 7 日

梁思立，南开大学，软件学院，硕士研究生，主要研究方向为通用博弈策略。

本文主要负责核心算法设计和实验实现。

LIANG Sili is currently a postgraduate

at the College of Software, Nankai University, Tianjin, China. His research focuses on general game playing.

In this paper, his contributions are the design of the core part of the algorithm and the implementation of the experiments.

E-mail: 2120210551@mail.nankai.edu.cn

姜桂飞，南开大学，软件学院，博士，讲师，主要研究方向为通用博弈策略、算法博弈论、多智能体系统。发表SCI/EI论文15篇以上。

本文主要负责文章整体框架设计，论文写作与修改。

JIANG Guifei, Ph.D., is currently an assistant professor at the College of Software, Nankai University, Tianjin, China. Her research interests include general game playing, algorithmic game theory, multi-agent systems. She has published more than fifteen high quality research papers.

In this paper, her contributions are the design of the overall framework of the paper, and the manuscript writing and revision.

E-mail: g.jiang@nankai.edu.cn

陈泰劼，香港大学，工程学院，研究助理，主要研究方向为深度学习和强化学习。

本文主要参与实验实现。

CHEN Taijie is currently a research assistant at the Department of Computer Science, University of Hong Kong, Hong Kong, China. His research interests include deep learning and reinforcement learning.

In this paper, his contributions are the design and implementation of the experiments.

E-mail: ctj21@connect.hku.hk

邓益超，新加坡国立大学，计算机学院，硕士研究生，主要研究方向为人工智能。

本文主要参与实验实现。

DENG Yichao is currently a postgraduate at the School of Computing, National University of Singapore. His research focuses on artificial intelligence.

In this paper, his contributions are the design and implementation of the experiments.

E-mail: e0792453@u.nus.edu

战瑀璠，南开大学，金融学院，硕士研究生，主要研究方向为金融科技。

本文主要负责模型的建立和写作。

ZHAN Yufan is currently a postgraduate at the School of Finance, Nankai University, Tianjin, China. Her research focuses on fintech.

In this paper, her contributions are the design and implementation of the general game player.

E-mail: 2120202478@mail.nankai.edu.cn

张玉志，南开大学，软件学院，院长，博士，讲席教授，主要研究方向为人工智能。

本文主要参与论文整体框架指导。

ZHANG Yuzhi, Ph.D., is currently a distinguished professor and the dean of the College of Software, Nankai University, Tianjin, China. His research focuses on artificial intelligence.

In this paper, his contributions are the guidance of the overall organization of the paper.

E-mail: zyz@nankaiedu.cn

梁思立,姜桂飞,陈泰劼,邓益超,战瑀璠,张玉志. 基于蒙特卡洛树搜索的通用博弈系统的构建与优化研究[J]. 数据与计算发展前沿, 2022, 4(3):66-77. CSTR: 32002.14.jfdc.CN10-1649/TP.2022.03.005.DOI:10.11871/jfdc.issn.2096-742X.2022.03.005.PID:21.86101.2/jfdc.2096-742X.2022.03.005.

LIANG Sili, JIANG Guifei, CHEN Taijie, DENG Yichao, ZHAN Yufan, ZHANG Yuzhi. Optimizing Monte Carlo Tree Search for General Game Playing [J]. *Frontiers of Data & Computing*, 2022, 4(3):66-77. CSTR: 32002.14.jfdc.CN10-1649/TP.2022.03.005.DOI:10.11871/jfdc.issn.2096-742X.2022.03.005.PID:21.86101.2/jfdc.2096-742X.2022.03.005.