双人博弈问题中的蒙特卡洛树搜索算法的改进

季 辉 丁泽军
(中国科学技术大学物理系 合肥 230026)

摘 要 蒙特卡洛树搜索(MCTS)是一种针对决策类博弈游戏,运用蒙特卡洛模拟方法进行评估博弈策略的启发式搜索算法。但是,在面对计算机围棋这种复杂的决策过程时,简单的蒙特卡洛树搜索过程往往由于计算量大,收敛速度非常慢。由于双人博弈游戏中的蒙特卡洛树搜索不能收敛于双人博弈的最佳决策策略,因此提出蒙特卡洛树搜索结合极大极小值算法的改进算法,使得搜索结果不会因为蒙特卡洛方法的随机性而失真。为了进一步提高复杂双人博弈游戏中搜索算法的计算效率,还结合了几种常见的剪枝策略。实验结果说明,所提算法显著改进了蒙特卡洛树搜索的准确性和效率。

关键词 蒙特卡洛树搜索,剪枝策略,双人博弈问题
中图法分类号 TP3 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2018.01.023

Improvement of Monte Carlo Tree Search Algorithm in Two-person Game Problem

JI Hui DING Ze-jun
(Department of Physics, University of Science and Technology of China, Hefei 230026, China)

Abstract Monte Carlo tree search (MCTS) is a heuristic search algorithm for some kinds of decision processes, most notably those employed in game play. In the decision process of a very complex game, like a computer Go game, the basic Monte Carlo tree search method converges very slowly due to large computation cost. This article indicated that MCTS cannot converge to the best strategy of the two-person complete information games. Therefore, this article proposed a new search algorithm which combines MCTS with the min-max algorithm in order to avoid the failure due to the randomness of Monte Carlo method. For further improving computation efficiency of MTCS in complex two-person games, this article also considered to employ some progressive pruning strategies. An experimental test shows that the new algorithm significantly improves the accuracy and efficiency of MCTS.

Keywords Monte Carlo tree search, Progressive pruning, Two-person games

1 引言

随着计算机技术的突飞猛进,人工智能在双人博弈的游戏上有着长足的进步。在双人博弈游戏中,经典的算法就是极大极小值算法(Min-Max),以及对其改进后的 alpha-beta 剪枝算法,两者在很多决策游戏中都有着不错的表现[1-3]。但上述算法比较依赖于与当前棋盘状态相关的评估函数。当评估的节点不是最终节点时,评估函数往往来自于人为编写的专家系统,而专家系统的优劣直接决定了评估函数的准确度。不准确的评估函数会舍去一些合理的决策分支,使得博弈结果不理想[4]。

近十年,通过对博弈游戏的不断研究,人们在蒙特卡洛模拟评估的基础上发展出了蒙特卡洛树搜索(MTCS)算法。不同于对决策树进行简单的蒙特卡洛模拟,蒙特卡洛树搜索是一种对蒙特卡洛模拟结果进行不断反馈调整的启发式搜索算法。

双人博弈游戏是指由两名玩家参与的博弈。参与的两人具有竞争关系,根据博弈规则,每人将选择自己最优的博弈行为以取得博弈的胜利。零和博弈指所有游戏参与者的总收益之和为零,即对于一个双人博弈游戏只会有一个胜利者。完全信息博弈是指所有的参与者在任何时刻对于博弈游戏的局面能获取的信息是完全相同的。我们可以通过构建一个博弈树来展示一个博弈游戏,树的节点表示博弈游戏的局面,分支则表示一种行棋策略。

原始的蒙特卡洛树搜索算法主要由两个问题组成[5-6]:第一个是起源于多臂匪徒问题(multi-armed bandit problem)的选择策略,即上限信心界策略[7](Upper Confidence Bound strategy, UCB),运用到搜索树中形成了上限信心界应用树算法[8],它决定了如何从模拟的或好或坏的众多结果中"学会"提高下一次的决策;第二个是模拟策略问题,其本质是蒙特卡洛模拟评估[1],简单来说就是通过随机扔骰子的方式来探索

到稿日期:2017-05-08 返修日期:2017-09-21
季 辉(1990-),男,硕士,主要研究方向为计算物理,E-mail:jhmochen@mail.ustc.edu.cn;丁泽军 教授,博士生导师,主要研究方向为计算物理、表面分析、电子显微学等,E-mail:zjding@ustc.edu.cn(通信作者)。

博弈树，并用多次模拟后的胜率来评估每个节点的优劣，但是难点在于面对巨大的样本空间时，如何抽样才能使样本更有效且更准确。

本文主要针对这两个策略提出相应的改进方法，并对完全随机的蒙特卡洛模拟运用在双人博弈问题中可能出现的一系列问题进行讨论；此外，用极大极小值算法的思想对蒙特卡洛模拟的过程进行约束，使得蒙特卡洛模拟的结果更加符合双人博弈游戏的结果。我们会随机产生一个博弈树，并在这个博弈树中进行探索，从而使实验更加具有普遍性。

本文第 2 节介绍了原始算法中存在的诸多问题，并进行了简单论述；第 3 节针对第 2 节说明的问题提出了改进方法；第 4 节设计了相关实验，并由实验证明了提出的改进算法是有效的；最后总结全文。

2　原始算法的缺陷

2.1　收敛速度

纯粹的随机掷点可以保证良好的随机性，但是随机性很强会影响到做出正确评估的收敛速度，尤其是对于类似于围棋的决策游戏，状态空间非常庞大，本身能探索到的空间只是一个小的部分，太强的随机性反而会使很多资源浪费到不必要考虑的状态空间中。因此，很多著名的围棋 AI 系统（如 fuego，Mango，AlphaGo）在运用蒙特卡洛树搜索时都会进行剪枝，从而使计算资源更多地分布到合理的对弈策略中。

2.2　非最优解

使用蒙特卡洛模拟评估的方法对双人博弈的局面进行评估，而分析得到的结果并不能收敛到双人博弈游戏的最优策略。本节从理论上分析其不合理性，并在下一节进行进一步的实验验证。

2.2.1　双人博弈游戏中的最优解

在任意一个双人博弈游戏中，无论它多么复杂，一定会存在一个先手或者后手的必胜策略（若考虑平局的情况，则一定存在一个先手或者后手的不败策略）。证明如下：

对于任意一个博弈游戏，都可以通过构建一个博弈树来表示所有决策。设有 A，B 两位玩家，其在博弈树中的决策如图 1 所示（若有两种决策则有两个节点，若有多种决策则有多个节点）。

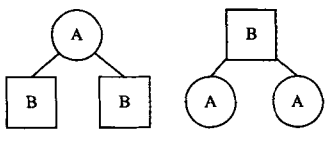

图 1　双人博弈树中的两种子树

Fig. 1　Two subtrees in the game tree

图 1 给出了博弈树中的两种子树（不一定是二叉树，也可以有多个节点），圆圈表示 A 的决策行动，方块表示 B 的决策行动。

假设 A，B 两位玩家都绝对聪明，可以分析出完整的决策树。此时从结果开始逆向推理胜负状态。在此，将选手 A 胜利记为 1，将其失败记为 −1。以 A 为例，设 A 走最后一步，有如下 3 种情况（见图 2）：情况 1 为 A 的所有决策分支都是 1；情况 2 为 A 的决策分支至少有一个 1；情况 3 为 A 的所有决策分支都是 −1。

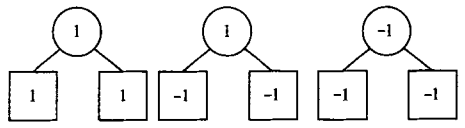

图 2　A 走棋的 3 种可能性

Fig. 2　Three possibilities for player A to play game

容易得知，由于此时最后一步由 A 来选择，则情况 1 和情况 2 为 A 必胜，情况 3 为 A 必败。用相同的方法就可以把结果向父节点传递，1 表示 A 的必胜态，−1 表示 A 的必败态。同理，对于 B 也有 3 种情况，且完全相反于 A。这样，由结果出发，一定可以计算出整棵决策树的根节点的结果，而且是唯一的。若根节点为 1，则一定是 A 有必胜策略；若根节点为 −1，则一定是 B 有必胜策略。

若存在平局的或者以分数来划分结果的游戏，则玩家 A 选择子节点中的最大值传递给父节点；玩家 B 则选择子节点中的最小值传递给父节点。这样，一定可以找到对于 A 或者 B 的最优值解。实际上，整个算法就是极大极小值算法。

如此，便证明了在任意一个双人博弈游戏中，无论其多么复杂，一定会存在一个先手或者后手的必胜策略（若考虑平局的情况，则一定存在一个先手或者后手的不败策略）；并且寻找到这个最优解的方法是极大极小值算法（Min-Max 算法）。

2.2.2　双人博弈游戏中模拟评估法不能收敛到最优解

本节给出一个极端例子进行反证：假设对于玩家 A 的博弈树只有一层（即只要 A 做出选择就能得到胜负结果），这一层有 100 个节点，其中一个节点为胜利，其他所有节点都是失败。此时如果用蒙特卡洛模拟评估法会得出玩家 A 在该局面下的胜率为 1%，而实际情况是 A 的胜率为 100%，因为 A 一定会选择胜利的决策。由此，便通过反例及理论证明了双人博弈游戏中蒙特卡洛模拟评估法不能收敛到最优解。

3　改进算法

本节对蒙特卡洛树搜索算法中的两个部分分别进行改进和讨论。

3.1　剪枝加速策略

首先介绍几种简单的剪枝策略。

（1）绝对剪枝策略[6,9-10]

绝对剪枝可以表示为：对于一个多节点的多臂匪徒问题，若有一个节点的访问次数达到了预计访问总次数的一半，则不再继续进行模拟测评。

由于 UCT 算法的原理是上限索引值越大的节点会被更多地访问，因此若一个节点的访问次数已经超过了总访问次数的一半，则不会再访问更多的节点。这证明该访问最多的点一定是上限索引值最大的，此时可以提前结束模拟过程，即可以节约运算资源。

（2）渐进展开式的剪枝策略[10]

这种剪枝策略依旧是针对 UCB 策略的剪枝方法，主要思路是，K 个节点的多臂匪徒问题并不会一开始就同时访问所

有节点，而是按照某一种方法对 K 个节点进行排序。设 t_i 表示前 i 个节点的访问总次数，则前 N 个节点的访问次数满足如下递推公式：

$$t_{i+1}=t_i+[C\times d^i]$$

其中，$C(C>0)$和 $d(d>1)$的实数都是可调节参数，中括号为取整运算。

这种算法可使排序靠前的节点获得更多的计算资源，从而更有可能将运算资源分配到可能比较合理的节点上。

3.2 评估算法的改进

3.1 节已经详细分析了双人博弈游戏的蒙特卡洛树搜索的模拟过程，即蒙特卡洛模拟评估的方法不能收敛到最优解。其根本原因在于，博弈游戏中玩家会根据局势选择对自己最有利的策略，而完全随机模拟的分布由于随机性太强，模拟结果的分布不能收敛到最优解。之前的蒙特卡洛模拟评估的过程会根据搜索的结果更新每个节点的平均收益，根据 3.1 节的分析容易知道，对于一个博弈问题，应该更新的是每个节点的最优可能收益。

本文进行如下改进：对于一个双人博弈问题，应该类比极大极小值算法把整个决策树按照先后手顺序分为 MAX 层和 MIN 层。MAX 层应该选择最大可能收益，MIN 层应该选择最小可能收益。

具体的改进后的过程如下：某一方在博弈过程中达到了某个局面，该局面有 K 个合法的策略。设这 K 个策略为搜索树中的 k 个节点，第 i 个节点有参数 v_i（该节点的最优收益）和 $i\in[1,K]$（第 i 个节点）。从根节点开始算起，设根节点为 MAX 层，则其子节点为 MIN 层，以此类推下一层为 MAX 层。测评中，随机选择一个节点进行模拟，直到游戏结束，得到一个结果 R，并从子节点开始向父节点传递结果值。若父节点是第一次访问，则直接将结果传递给父节点，若父节点已经存在一个 v_x 值，则与其子节点的值进行比较。当 v_x 在 MAX 层时，若 $v_x<R$，则令 $v_x=R$；若 $v_x>R$，则 v_x 不变。相反，当 v_x 在 MIN 层时，若 $v_x>R$，则令 $v_x=R$；若 $v_x<R$，则 v_x 不变。用上述方法不断地向上传递到根节点，该值就作为一次模拟得到的数值。这种方法不再平均最后的收益情况，而是反馈了双人博弈的过程中如何取得最大收益的一种算法。这样的方法更加符合双人博弈游戏的要求。

4 实验及分析

4.1 剪枝策略的改进实验

本节将对两种剪枝策略（相对剪枝策略和展开式剪枝策略）的有效性进行实验验证。

实验中将用到一个开源的围棋图形用户界面程序 gogui (Graphical user interface to Go programs)。这是一个功能强大的围棋界面程序，用户可以自由设置人机、人人、程序相互对弈模式，并且可以任意设置棋盘大小，任意接入不同的 AI 引擎；同时该程序还支持设置对弈局数并且记录对弈时间和对弈过程。

选择 Fuego 程序的 1.1-2 版本作为改进版的围棋引擎，Fuego 是基于蒙特卡洛树搜索算法的开源围棋对弈程序；对照程序选择了运行效率比较高的 Gnugo 程序 3.8 版本。

对于已有的开源围棋程序 Fuego（采用了蒙特卡洛树搜索算法的程序），按照上述方式进行改进并与另外一种围棋 AI 程序 Gnugo 进行对弈（见表 1）。可以看出，剪枝策略的确可以提高运算效率。

表 1 不同剪枝策略下围棋程序 Fuego 与 Gnugo 的对弈结果

Table 1 Palying results of Fuego and Gnugo Go programs in different pruning strategies

对 Fuego 改进情况	平均用时/min	胜利局数	胜率/%
原策略	17.2	192	49
绝对剪枝策略	14.1	211	54
展开剪枝策略	16.8	228	59
绝对剪枝策略＋展开剪枝策略	15.7	230	61

4.2 模拟评估的改进实验

实验中先随机生成一棵树作为某次双人游戏的博弈树。为了简化运算，假设这棵博弈树为一棵完全二叉树。设 A 和 B 为两个博弈选手，A 先手，B 后手，并且对树的叶子节点进行随机赋值（若 A 胜利则为记为 1 分，B 胜利则记为－1 分）。首先通过极大极小值算法，可以得到一个标准结果；然后分别使用蒙特卡洛模拟评估的方法以及改进后的评估法搜索决策树，并与之前的标准结果进行对比。

（1）对原始模拟评估法进行测评

此实验中，生成了 400 棵不同的 20 层的博弈树。每棵博弈树将进行 100000 次模拟评估。对于不同深度，将其与标准结果进行了比对。图 3 中横坐标为决策树的深度，纵坐标为 400 次模拟中符合极大极小值计算结果的百分比。可以看出，随着搜索深度的增加，原来的蒙特卡洛模拟评估与极大极小值算法的计算结果的差别越来越大，在 10 层时符合度依旧保持在 98%左右，而在 20 层时只有 25%左右的搜索结果符合。随着层数的增加，符合率呈指数递减，说明随着博弈树深度的不断增加，原始的模拟评估法的结果与最优解的差距越来越大。

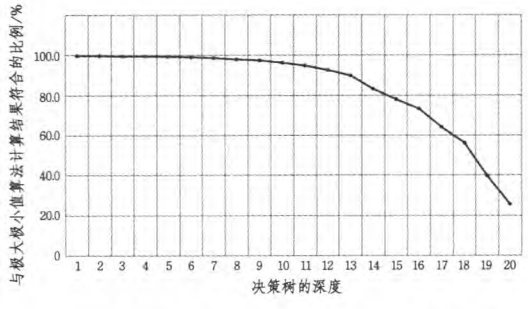

图 3 原蒙特卡洛模拟评估法在不同层数下与极大极小值算法的结果对比

Fig. 3 Comparison of original algorithm and Min-Max algorithm under different search statistics

（2）对改进后的模拟评估法进行测评

对上述生成的 400 棵不同的决策树，使用改进后的蒙特卡洛模拟进行搜索，以对比在不同的搜索次数（1000～10000）下第 20 层的结果与标准结果的符合比例，结果如图 4 所示。

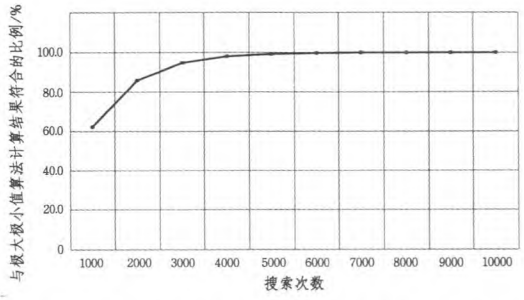

图 4　改进后的模拟评估法与极大极小值算法的计算结果在不同搜索次数下的对比图

Fig. 4　Comparison of the improved algorithm and Min-Max algorithm under different search statistics

由图 4 可知，随着搜索次数的增加，改进后的蒙特卡洛模拟评估法的结果会越来越接近极大极小值计算的结果；而对于原始的蒙特卡洛模拟评估法，搜索次数高达 100000，其结果的符合度也只有 25%。以上说明，改进后的评估法会使计算结果更易收敛于最优解。

(3)对更大的博弈树进行评估

这里生成 400 棵 25 层的博弈树，使用改进后的方法进行搜索，并对比不同搜索次数下改进后的评估算法与标准结果的对比符合度(搜索次数为 10000～160000)，结果如图 5 所示。

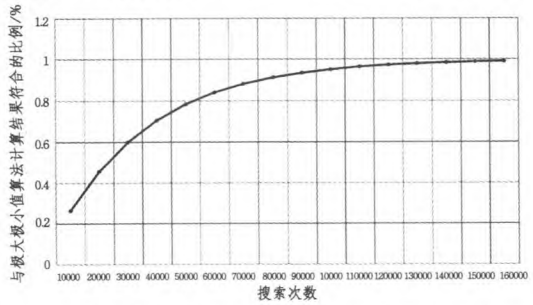

图 5　改进后的 MC 评估法与极大极小值算法计算结果的对比图

Fig. 5　Comparison of the improved algorithm and Min-Max algorithm under different search statistics

实验中将决策树由 20 层增加到 25 层后发现，虽然改进后的蒙特卡洛评估法依旧可以收敛到极大极小值算法的计算结果中，但是随着博弈树层数的增加，收敛所需要的计算次数也呈指数上升，虽然只增加了 5 层，但搜索次数高达 100000 时符合度才达到 96%左右。由此可知，改进后的蒙特卡洛评估法还不能直接运用到围棋这种博弈树巨大的双人博弈问题中。

4.3　实验分析

通过上述实验可以得知：1)合理的剪枝算法可以在一定程度上提高蒙特卡洛树搜索算法的效率；2)蒙特卡洛树搜索算法的模拟过程中，采用随机结果的平均值来评估双人博弈游戏中的局面并不能选择出最优的行棋策略；3)改进后的蒙特卡洛评估法在双人博弈问题中能够更好地收敛到最优的博弈策略。如果将改进后的蒙特卡洛评估法更新到蒙特卡洛树搜索算法中，会使蒙特卡洛树搜索的结果更接近于最优的博弈策略。

在目前的研究中发现，如果直接将改进后的蒙特卡洛评估法与 UCT 算法结合，搜索效率并不高，从而使得改进后的算法无法直接运用到状态空间巨大的决策树搜索中。如果要将其推广到一般的双人博弈问题中，还需要进行进一步的改进。

结束语　本文介绍了蒙特卡洛树搜索算法在双人博弈游戏中的运用，说明了该算法在博弈游戏中存在的缺陷，在面对策略性游戏时，通过蒙特卡洛模拟评估得到的平均胜率或平均得分的高低与决策的优劣之间并没有直接的联系。本文针对蒙特卡洛树搜索算法中的问题提出了改进的算法，使得蒙特卡洛模拟所得到的结果更能反映出博弈游戏中决策的优劣。由于类似于围棋的双人博弈游戏的状态空间比较大，直接使用蒙特卡洛树搜索算法时效率较低，在改进的算法上融合几种剪枝算法会使得计算效率得到进一步的提升。

参 考 文 献

[1] CAMPBELL M, HOANE A J, HSU F. Deep blue[J]. Artificial Intelligence, 2002, 134(1/2): 57-83.

[2] SCHAEFFER J. Game Over: Black to Play and Draw in Checkers[J]. ICGA Journal, 2007, 30(4): 187-197.

[3] SCHAEFFER J, BURCH N, BJÖRNSSON Y, et al. Checkers is solved[J]. Science, 2007, 317(5844): 1518-1522.

[4] BOUZY B, CAZENAVE T. Computer Go: an AI oriented survey [J]. Artificial Intelligence, 2001, 132(1): 39-103.

[5] COULOM R. The Monte-Carlo Revolution in Go[C]// The Japanese-French Frontiers of Science Symposium (JFFoS 2008). Roscoff, France, 2009.

[6] CHASLOT J B, et al. Progressive strategies for Monte-Carlo tree search[J]. New Mathematics and Natural Computation, 2008, 4(3): 343-357.

[7] CHANG H S, FU M C, HU J, et al. An adaptive sampling algorithm for solving Markov decision processes[J]. Operations Research, 2005, 53(1): 126-139.

[8] KOCSIS L, SZEPESVÁRI C. Bandit based Monte-Carlo planning[C]// European Conference on Machine Learning. Springer Berlin Heidelberg, 2006: 282-293.

[9] HUANG J. Application and Enhancement of the UCT Algorithm in Computer Go[D]. Beijing: Beijing University of Posts and Telecommunications, 2011. (in Chinese)
黄晶. 计算机围棋博弈中 UCT 算法的应用及改进[D]. 北京：北京邮电大学，2011.

[10] YU Y B. Computer Go Game Research Based on Monte Carlo Tree Search[D]. Dalian: Dalian Maritime University, 2015. (in Chinese)
于永波. 基于蒙特卡洛树搜索的计算机围棋博弈研究[D]. 大连：大连海事大学，2015.

[11] ABRAMSON B, KORF R E. A Model of Two-Player Evaluation Functions[C]// AAAI. 1987: 90-94.

[12] EGELLY S, WWANG Y. Exploration exploitation in Go: UCT for Monte-Carlo Go[C]// NIPS: Neural Information Processing Systems Conference On-line trading of Exploration and Exploitation Workshop. 2006.

[13] 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.