基于 Pierre Dellacherie 算法的俄罗斯方块 AI 实现

1710229 李玉奇 [1], 1710641 李宗显 [2], 1711109 刘雨濠 [3],1710182 张美涵 [4]

(2020 年 12 月 19 日)

摘要：人工智能和机器学习一直在寻求俄罗斯方块的自动算法。本文基于 Pierre Dellacherie 算法, 将人类玩俄罗斯方块的策略参数化，通过使目标函数局部最优化确定下落的位置，实现了一个俄罗斯方块的 AI。并讨论了此算法的改进方向。

关键词：Pierre Dellacherie 算法; 目标函数

1 引言

俄罗斯方块是一款家喻户晓的游戏。在 1997 年 Heidi Burgiel 用极大极小值算法证明了当出现的七种方块数完全随机时，俄罗斯方块必定会结束。所以人们开始研究使俄罗斯方块得分高的算法。下面我们介绍一种局部最优的算法 Pierre Dellacherie 算法。的核心思想是对于每一个可以放置方块的位置，我们给它一个评价函数。评价函数中对有利于后续的参数赋予正权重，不利于后续的参数赋予负的权重。使该函数值最大的位置就是当前最优位置。我们用基于此算法用 python 制作了一个 AI，开源代码请见 https://github.com/LIyvqi/AI_Teries/tree/master。

2 Pierre Dellacherie 算法

Pierre Dellacherie 算法需要将人类策略参数化。我们将分别介绍人类策略、参数化过程以及目标函数的制定。

2.1 局面好坏的评价因素

评价局面好坏的因素多种多样，我们将人类视角的评判因素归纳成以下五条：

1. 板块放置后消除的行数，显然消除的行数越多越好。
2. 板块放置后，该板块最高点高度也是重要因素。最高点越低，后续可能放置的板块越多。所以相同条件下，有限放置在最高点低的位置。
3. 板块放置后，与该板块接触的小方块的数量是一个重要因素。与之接触的越多越紧密，产生“空洞”的可能性越小，越有利于后续局面。
4. 每一行未被填充区域的的分布情况。显然它们分布的越零散，后续的局面越难处理。
5. 游戏过程中形成的空洞“空洞”极难处理。我们将空洞数作为一个指标，越少越好。

2.2 评价因素的参数化

Pierre Dellacherie 算法将上述因素抽象成以下六个因素：

1. **landingHeight**：板块放置后，板块重心位置距离底部的距离。
2. **erodedPieceCellsMetric**: 我们充分考虑消去的行数与该板块对消去行数的贡献。定义此参数为消除的行数与当前板块中被消除的

方格数的乘积。如图 1 所示，消除的行数为 2，该板块被消除的方格数为 3，所以该参数为 $2 \times 3 = 6$。

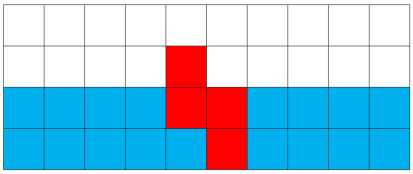

图 1

3. **boardRowTransitions**: 对于每一行，从无方格到有方格，从有方格到无方格我们都成为一个“变换”，该参数是每行“变换”数目相加。该参数刻画了未填充方格的离散程度。如图 2 所示，每一行黑线对应一次“变换”。所以第一行的变换次数是 6，第二行的变换次数是 4，所以此局面对应的参数为 10。

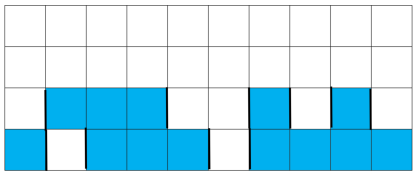

图 2

4. **boardColTransitions**: 类似于行的定义，这是每一列的变换次数之和。

5. **boardBurieHoles**: 各列中“空洞”数目总和。如图 3 所示，“空洞”的数目为 7。

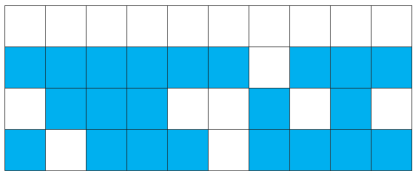

图 3

6. **boardWells**: 各列中“井”的连加和。“井”的定义是，两边 (包括边界）都有方块填充的空列。它刻画了板块落到这里与其它小方格接触的紧密程度。可以想到“井”越深，我们方块落到此位置与其它方块接触越“松”，处理掉它需要多次操作。如图 4 所示，黑线所标即为两个“井”，所以对于此图，此参数为 $(1+2)+(1+2+3)=9$。

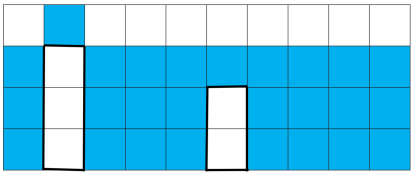

图 4

2.3 目标函数的选取

我们发现以上只有 erodedPieceCellsMetric 对局面有利，我们将此参数赋予正的权重，其余参数赋予负的权重。由于“空洞”极难处理，我们将对应于“空洞”的参数 boardBurieHoles 权重调高，我们写出初始的评价函数：

$$\begin{aligned} value = &- landingHeight + erodedPieceCellsMetric \\ &- boardRowTransitions - boardColTransitions \\ &- (4 \times boardBurieHoles) - boardWells. \end{aligned}$$

后来人们经过大量游戏实例的分析，将评价准则修改为：

$$\begin{aligned} value = &- 45 \times landingHeight - 34 \times boardWells \\ &+ 34 \times erodedPieceCellsMetric \\ &- 93 \times boardColTransitions \\ &- 79 \times boardBurieHoles \\ &- 32 \times boardRowTransitions. \end{aligned}$$

其中 value 最大的值为最优位置。但是可能会出现两个局面同分的情况，此时我们定义一个优先度函数，相同分数下操作（旋转、移动）越少，也即 prior 越小，为更优的位置。我们可以定义优先度函数：

$$\begin{aligned} priority = &100 * times\ of\ horizontal\ operation \\ &+ times\ of\ selection\ operation. \end{aligned}$$

当 value 相同时，优先选择 priority 小的位置下落方块。

3 算法改进

此算法使一个局部最优化算法，但是最高可以达到消去 3500 万行的成绩。但是我们依然可以对它进一步改进，一种是基于算法局部最优的改进，我们不再只考虑当前落下的方块，将 value 和 priority 函数视为当前方块和下一个方块的函数，然后优化两个函数确定下落位置。如果下一个方块未知，我们认为六种方块出现可能性相同，对 value 和 priority 函数对第二个方块做平均在进行优化。另一种优化方法是基于机器学习的思想。比如将最终得分作为训练目标，优化参数的系数。且 Pierre Dellacherie 算法是基于人类视角的参数化，人类的视角不一定是最优的视角 (比如 AlphaGo 在自我对抗时很多棋路超越人类的视觉)，所以我们可以将最终得分作为目标，利用强化学习的方法让 AI 在玩游戏的过程中积累数据，最后自己生成评价局面的参数。

致谢

十分感谢张老师一学期的悉心教授，让我们对机器学习的脉络有了宏观把控，也对机器学习中的数学知识有了了解。此外，也十分感谢助教一学期的悉心指导，耐心答疑。最后十分感谢小组同学精诚合作，不畏困难，一起完美完成了此次机器学习大作业。

参考文献

[1] Bertsekas, Dimitri P and Tsitsiklis, John N. Neuro- Dynamic Programming. Athena Scientific, 1996.

[2] Ontanón, Santiago, Synnaeve, Gabriel, Uriarte, Alberto, Richoux, Florian, Churchill, David, and Preuss, Mike. A survey of real-time strategy game ai research and competition in starcraft. IEEE Transactions on Computational Intelligence and AI in games, 5(4):293–311, 2013.

[3] Scherrer, Bruno, Ghavamzadeh, Mohammad, Gabillon, Victor, Lesner, Boris, and Geist, Matthieu. Approxi- mate Modified Policy Iteration and its Application to the Game of Tetris. The Journal of Machine Learning Re- search, 16:47, 2015.

[4] Fahey, Colin. Tetris ai, June 2003.

[5] Burgiel, Heidi. How to Lose at Tetris. The Mathematical Gazette, 81(491):194 – 200, 1997.