
PACMAN 游戏实验报告

王一栋 (1512201113、646842131@qq.com)

(南京大学 计算机科学与技术系, 南京 210093)

摘要: 理解并阐述程序中使用的强化学习算法和过程, 并且修改程序提高学习性能。

关键词: 人工智能; 强化学习; Q-Learning

中图法分类号: TP301 **文献标识码:** A

1 引言

本次作业目的是使用强化学习来自主玩 Mr. PACMAN 游戏, 理解并阐述程序中使用的强化学习算法和过程, 并且修改程序提高学习性能。

2 实验内容

2.1 阐述强化学习的方法和过程并回答问题

本次实验使用的强化学习方法是带 epsilon-greedy 的 Q-Learning 方法, 具体过程如下:

在每次调用 act 函数时, 通过 learnPolicy 函数进行强化学习, 在 learnPolicy 中, 程序通过 simulate 函数进行 10 次 mc 模拟采样, 每次模拟采样都是一个 episode, 在每次采样中采用 epsilon-greedy 策略进行模拟采样, 即有一定概率不利用最佳 Q 值决策, 而是进行探索。采样用 reptime 决策树算法学习收集到的状态的特征并构建决策树模型, 这样在遇到未知状态时也能估计当前状态的 Q 值。每次在真正决策时选择当前状态下能获得最大 Q 值的那个动作。

在 simulate 中, 程序根据如下公式进行迭代计算 (两次 heuristic 的 evaluatestate 就是在计算 R 值):

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha [R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)]$$

其中 alpha 在此程序中默认为 1, 即学习速率为 1。

(1)策略模型用什么表示? 该表示有何缺点? 有何改进方法?

策略模型是带 epsilon-greedy 的 Q-Learning 方法, 该表示的缺点是学习速率 alpha 设为 1, 可能找不到最优策略, epsilon 的值固定不变, 后期随机动作的概率应该降低 (即 epsilon 应该随着程序的运行而变小), 且可能会出现对 Q 值过度估计的问题, 解决方案是可以动态设置 alpha, 动态设置 epsilon, 使用 double Q-Learning。

(2)Agent.java 代码中 SIMULATION_DEPTH, m_gamma, m_maxPoolSize 三个变量分别有何作用?

SIMULATION_DEPTH 是每个 episode 的长度, 即模拟采样时向前模拟步数的长度;

m_gamma 是折扣因子, 为了平衡现在和未来的 reward, 使 reward 收敛, 折扣因子越大表示对未来的回报越感兴趣, 折扣因子为 0 时程序是短视的;

m_maxPoolSize 是存储状态的最大个数, 即估算 Q 值的决策树模型中最多的状态数, 因为 pacman 的状态数量太多无法一一存储, 所以需要用决策树存储少量状态来估算其余未知状态的 Q 值。

(3)QPolicy.java 代码中, `getAction` 和 `getActionNoExplore` 两个函数有何不同? 分别用在何处?

`getAction` 是带 `epsilon-greedy` 的 `q-learning` 方法, 即有一定概率不选取最大 `Q` 值来决定下一步而是随机走一步, 这么做是为了避免程序在一块很小的区域内打转, 用在探索阶段, 探索尽可能多的状态, 得到更精确的状态与 `Q` 值的映射, 即使 `Q` 值更收敛于真实值。

`getActionNoExplore` 是不带 `epsilon-greedy` 的 `q-learning` 方法, 根据当前状态的可能动作的 `Q` 值来选取动作, 用在利用阶段, 即真正决策的阶段。

2.2 尝试修改特征提取方法

我尝试提取了 `avatar` 与钻石和敌人的距离, 敌人的状态, 目前分数来当做特征, 钻石具有重要战略意义 (可以反杀敌人获取大量的分数), 不同状态的敌人意义也不同 (正常状态敌人危险, 在自己吃了钻石之后敌人变成了提款机) 具体提取方式如下:

```
//敌人状态
if ( obs.getNPCPositions() != null )
    for (ArrayList<Observation> l : obs.getNPCPositions()) allobj.add(1);
for (Observation o : allobj) {
    feature[i]=o.itype;
    i++;
}
```

```
//敌人距离
for (Observation o:allGhost) {
    feature[i] =o.position.dist(avatarPos);
    i++;
}
```

```
//钻石距离
LinkedList<Observation> allobj2= new LinkedList<>();
if ( obs.getResourcesPositions() != null )
    for (ArrayList<Observation> l : obs.getResourcesPositions()) allobj2.add(1);
for (Observation o : allobj2) {
    Vector2d p = o.position;
    feature[i]=p.dist(avatarPos);
    i++;
}
```

修改特征后与原特征之间的对比 (每个关卡玩了 10 次, 取得分平均值进行对比):



2.3 尝试修改强化学习参数

观察原程序，我们可以发现折扣因子为 0.99，这太注重未来的收益，我把折扣因子改为了 0.9（当下的收益也很重要）。单次模拟步数并不做调整，20 步已经足够长，但抽样次数只有 10 次，太少了，我把它改成了 50 次，因为只有 10 次的话可能 10 次抽样全是走向了不好的结局，并没有使 Q 值收敛。m_maxPoolSize 改为 2000，增加未知状态计算 Q 值的准确度。

修改强化学习特征后对比：



3 结束语

本次实验通过阅读并理解 PACMAN 程序，加深了对强化学习的理解。

致谢 在此,我们向对本文的工作给予支持和建议的同学和老师,特别是南京大学计算机科学与技术系的俞杨教授表示感谢。

References:

- [1] Stuart J. Russell, Peter Norvig. Artificial Intelligence: A Modern Approach (3rd edition), Pearson, 2011.