

# Table of Contents

- [1. 22秋程序设计PJ:小黄和它的罐子 ——简单的遗传算法实例](#)
  - [1.1. 项目背景](#)
  - [1.2. 项目要求](#)
    - [1.2.1. PJ1](#)
    - [1.2.2. PJ2](#)
  - [1.3. 评分标准](#)
    - [1.3.1. PJ1](#)
    - [1.3.2. PJ2](#)
  - [1.4. 注意事项](#)

## 1. 22秋程序设计PJ:小黄和它的罐子 ——简单的遗传算法实例

### 1.1. 项目背景

很久很久以前,有一群小黄快乐地生活在10\*10的有围墙的世界里.

每个格子里会有50%的概率出现小黄喜欢的罐子,小黄的任务是捡起尽量多的罐.它们只能看到自己所在的格子和自己上下左右的四个格子(一共五个格子,每个格子可能出现的情况包括:罐子、空格子、墙).

### 1.2. 项目要求

#### 1.2.1. PJ1

##### 1. 随机初始化的棋盘

为每一只小黄准备10\*10的有围墙的棋盘,棋盘上每一个格子都有50%的概率有罐子.小黄的初始位置为左上角的格子,记为(1,1).游戏开始时你需要生成一张随机棋盘,棋盘满足以下条件

1. 棋盘边缘的四个边被围墙覆盖
2. 围墙以内每个格子有50%的概率出现罐子或空格子

##### 2. 动作

小黄可以做的事包括:

1. 随机移动,即随机出现以下行为中的任何一种
2. 向上移动,如果小黄上方的格子为墙,则小黄位置不变,视为一次撞墙
3. 向下移动,如果小黄下方的格子为墙,则小黄位置不变,视为一次撞墙
4. 向左移动,如果小黄左边的格子为墙,则小黄位置不变,视为一次撞墙
5. 向右移动,如果小黄右边的格子为墙,则小黄位置不变,视为一次撞墙
6. 不移动
7. 捡罐子<sup>1</sup>,如果小黄此时的位置上有罐子,则罐子被捡起,即此格子变为空

##### 3. 得分

这些行为,对应了不同的得分

- 撞墙-5分
- 捡到罐子+10分
- 做了捡罐子的动作但是没有捡到罐子-2分

在PJ1中,你需要实现从标准输入读入命令,让小黄完成动作,并使得分发生相应变化.

#### 1.2.2. PJ2

在PJ1中,你上帝视角(能看见所有格子)指导小黄捡罐子.在PJ2中,我们要为小黄找到最佳策略,使得他们可以根据有限的已知信息作出好的决定.每一个小黄的策略表长度为243,对应小黄可能看到的35种情况.其值为0~6的整数,对应小黄可以做的7种动作.也就是说,有了这个策略表,小黄可以根据自己看到的情况做出唯一选择.

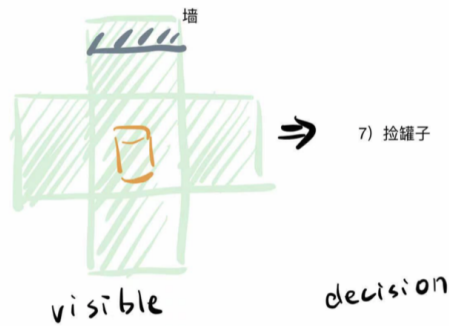


Figure 1: 小黄需要根据自己能看到的五个格子的情况进行反映,面对一种特定的五个格子的组合,小黄只能做出一种固定的决策

## 1. 读入策略

在PJ2中,你需要实现从 .txt <sup>2</sup>读入策略,让小黄根据策略完成动作,并使得分发生相应变化,完成这一步你应该能为小黄指定一个策略表并看见小黄按照你给定的策略自动在棋盘上找罐子.

## 2. 用遗传算法<sup>3</sup>找到最优策略

我们随机生成第一批小黄(200只)的策略表.我们也为每一只小黄准备了100个棋盘,每一只小黄在它自己的100个棋盘上各走200步,记下它在每个棋盘上的分数的平均值作为它的最终分数.

由于我们的策略表是随机生成的,小黄的得分并不理想.

为了优化小黄的决策,我们循环执行以下步骤1000代:

1. 对小黄按照得分进行排序, rank 为排名,  $i$  为  $201 - \text{rank}$ .生成  $1 \sim 20100$  之间的随机数 rand,根据 rand 的值抽取小黄,这样能保证得分高的小黄更容易被抽到

(rank=201-i)

i	rand
1	1
2	2,3
3	4,5,6
...	...
200	19900~20100

Figure 2: 生成随机数 rand

2. 按照1.中方法抽取2个小黄,“交配”得到两个子代.交配,即将亲本的决策表每60一切割(最后一段大于60),并交替拼接得到子代

1. 子代1的决策表: 亲本1片段+亲本2片段+亲本1片段+...

2. 子代2的决策表: 亲本2片段+亲本1片段+亲本2片段+...

id	0	1	...	60	...	120	...	180	...	240	241	242
$201-i$	1	0		6		5		2		3	1	2

Figure 3: 交配

3. 在生成子代决策表的时候,子代决策表的每一个决策都有一定概率发生突变.(可以尝试变异率的倒数 MU 的值为200,400,800 时的不同效果)

```
Tips:  
变异: 利用随机数。设定变异率的倒数MU, 在复制  
亲本的每一个策略 (数字0~6) 时  
if random_int(0,MU) != 0:  
    list_decision_child[i]=list_decision_parent[i]  
else:  
    list_decision_child[i]=random_int(0,7)
```

Figure 4: 变异

4. 利用2.,3.描述的交配繁殖方法,抽取95对亲本(亲本可以重复),完成190个子代的生成.另外10个子代直接复制前十名的亲本的决策表
5. 每一只子代小黄在它自己的100个棋盘上各走200步,记下它在每个棋盘上的分数的平均值作为它的最终分数.回到1.

### 3. 迭代结果

我们将每一代200个小黄的平均得分输出,如下图所示,可以看出,到最后小黄的得分已经非常高了

```
g_988_ ave_335.325012  
g_989_ ave_338.470001  
g_990_ ave_334.355011  
g_991_ ave_334.890015  
g_992_ ave_332.515015  
g_993_ ave_332.894989  
g_994_ ave_329.605011  
g_995_ ave_332.429993  
g_996_ ave_333.390015  
g_997_ ave_332.244995  
g_998_ ave_331.714996  
g_999_ ave_330.450012
```

Figure 5: 迭代结果

我们也可以从更加直观的统计散点图上看小黄的进步(选做)

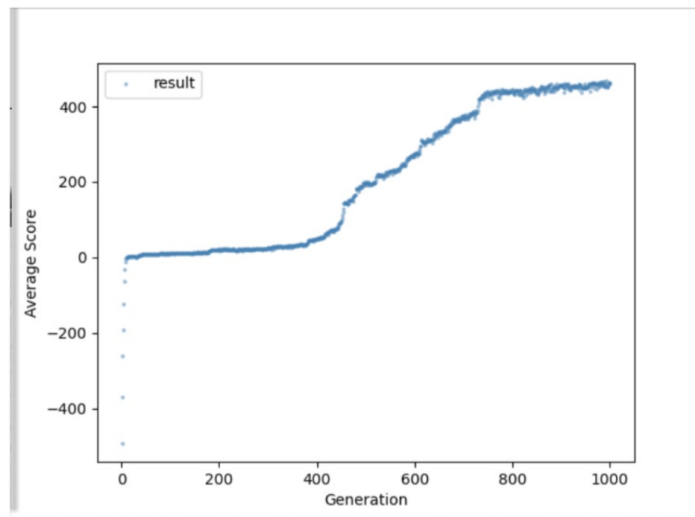


Figure 6: 散点图

#### 4. 最终结果的可视化

1. 将最后一代小黄的最高分决策写入txt中,数字之间用制表符隔开
2. 将此小黄的决策表作为我们采纳的最终决策表
3. 利用此决策表在一个棋盘上走200步
4. 输出初始棋盘:

```
score:0
# # # # # # # # # # #
# ! @ @ @ @ @ @ @ @ @ #
# @ @ @ @ @ @ @ @ @ @ #
# @ @ @ @ @ @ @ @ @ @ #
# @ @ @ @ @ @ @ @ @ @ #
# @ @ @ @ @ @ @ @ @ @ #
# @ @ @ @ @ @ @ @ @ @ #
# @ @ @ @ @ @ @ @ @ @ #
# @ @ @ @ @ @ @ @ @ @ #
# @ @ @ @ @ @ @ @ @ @ #
# # # # # # # # # # #
```

Figure 7: 初始棋盘,其中 # 为墙壁, @ 为罐子, score 表示分数, ! 为小黄当前位置

5. 按照最终决策表执行,并且每执行一步输出一张棋盘.例如:执行了一次向右走和捡罐子之后的输出:

```
score:10
# # # # # # # # # # #
# ! @ @ @ @ @ @ @ @ @ #
# @ @ @ @ @ @ @ @ @ @ #
# @ @ @ @ @ @ @ @ @ @ #
# @ @ @ @ @ @ @ @ @ @ #
# @ @ @ @ @ @ @ @ @ @ #
# @ @ @ @ @ @ @ @ @ @ #
# @ @ @ @ @ @ @ @ @ @ #
# @ @ @ @ @ @ @ @ @ @ #
# @ @ @ @ @ @ @ @ @ @ #
# # # # # # # # # # #
```

Figure 8: 执行了一次向右走和捡罐子之后的输出

#### 5. Bonus

1. 尝试其他算法(例如人工神经网络<sup>4</sup>)
2. GUI 图形界面实现<sup>5</sup>
3. 绘制散点图,每一代200个小黄的平均得分为一个点

### 1.3. 评分标准

#### 1.3.1. PJ1

需要提交<sup>6</sup>

- 源代码和可执行文件
- 项目说明文档

实现一个键盘控制的捡罐子游戏,正确计算用户操作后的得分变化

得分点	得分	评分标准
随机生成棋盘	10	需要随机生成符合要求的棋盘,即每个坐标出现的物体随机
正确打印棋盘	15	将棋盘按照 #为墙壁,@为罐子,score表示分数,!为小黄当前位置 的标志打印到标准输出
正确处理用户输入	25	针对用户输入的7种 <sup>7</sup> 可能,正确改变棋盘状态
正确计算分数	25	根据打分规则 <sup>8</sup> ,正确改变当前得分
良好且一致的代码风格	10	可参考 Google Style Guides <sup>9</sup> 和其他流行规范文档 <sup>10</sup>
合理的模块化(和复用)	10	按照不同功能,将PJ实现分成不同的文件
清晰的说明文档	10	实现思路,实验结果,遇到的困难和解决过程,对PJ的改进建议(可选),参考资料(可选),其他(可选)

#### 1.3.2. PJ2

需要提交<sup>11</sup>

- 源代码和可执行文件(有两个可执行文件.一个用于训练,运行后则开始训练,训练过程中输出每一代小黄的得分,最终创建并得到一个包含策略表的 .txt 文件;另一个用于测试策略,运行后读入策略表,打印地图,开始自动找罐子)
- 项目说明文档
- 你得到的最优策略表
- 你的训练迭代结果(每一代的得分)
- Bonus<sup>12</sup>: 其他算法实现的源代码和可执行文件
- Bonus<sup>12</sup>: 图形化界面的可执行文件
- Bonus<sup>12</sup>: 散点图

实现文件读取策略表,通过遗传算法训练得到最优策略,输出迭代结果,将策略可视化.

得分点	得分	评分标准
读取策略表文件	15	从 .txt 文件读取策略表,根据策略表自动选择动作
你得到的最优策略表	15	根据你的策略,小黄得分在400以上计5分
遗传算法实现	20	实现遗传算法,通过迭代训练找到最优策略
训练迭代结果	20	打印每一轮迭代的训练结果
良好且一致的代码风格	10	可参考 Google Style Guides <sup>9</sup> 和其他流行规范文档 <sup>10</sup>
规范的项目结构	10	根据需要提交的文件组织提交的文件夹的结构 <sup>11</sup>
清晰的说明文档	10	实现思路,实验结果,遇到的困难和解决过程,对PJ的改进建议(可选),参考资料(可选),其他(可选)
Bouns1	10	尝试其他算法(遗传算法必做/不可替代)
Bouns2	10	GUI图形化界面
Bouns3	5	绘制散点图,将训练过程的每代得分绘制成散点图,体现策略的进步

### 1.4. 注意事项

1. 合理安排时间,尽早动手,完成每个功能点任务,不犯拖延症
2. 推荐使用 IDE 进行编写,调试,维护项目结构
3. 如果有任何疑问,及时和TA们联系

4. 注意自己的代码结构和风格(选择一种风格后需要保持一致),PJ2在PJ1基础上进行编写
5. 鼓励同学互相讨论,和TA讨论,找寻解决问题的办法,但严禁抄袭,一旦发现抄袭者和被抄袭者都作0分处理

## Footnotes:

<sup>1</sup> 走到罐子所在的格子不会使罐子自动被捡起,需要发出指令

<sup>2</sup> 文件格式为一行243个字符,每个字符可能是[1, 7]

<sup>3</sup> [遗传算法 - 维基百科,自由的百科全书](#)

<sup>4</sup> [人工神经网络 - 维基百科,自由的百科全书](#)

<sup>5</sup> 推荐一个为C语言提供图形化接口的库:[Simple DirectMedia Layer - Wikipedia](#),这是一个简单教程[SDL library in C/C++ with examples - GeeksforGeeks](#),项目主页[Simple DirectMedia Layer - Homepage](#)

<sup>6</sup> PJ1推荐提交文件夹按如下结构组织,压缩成 .zip 格式后提交

```
PJ1Handin_19302010001张三
├── PJ1.out
├── README.pdf
└── src #游戏源代码, src文件夹下具体的文件名可以不同
    ├── map.h
    ├── map.c
    ├── ...
    ├── main.h
    └── main.c
```

<sup>7</sup> 合法输入有

1. 随机移动,即随机出现2~行为中的任何一种;
2. 向上移动,如果小黄上方的格子为墙,则小黄位置不变,视为一次撞墙;
3. 向下移动,如果小黄下方的格子为墙,则小黄位置不变,视为一次撞墙;
4. 向左移动,如果小黄左边的格子为墙,则小黄位置不变,视为一次撞墙;
5. 向右移动,如果小黄右边的格子为墙,则小黄位置不变,视为一次撞墙;
6. 不移动;
7. 捡罐子,如果小黄此时的位置上有罐子,则罐子被捡起,即此格子变为空;

<sup>8</sup> 撞墙-5分,捡到罐子+10分,做了捡罐子的动作但是没有捡到罐子-2分

<sup>9</sup> [Google Style Guides | styleguide](#)

<sup>10</sup> [Pintos Projects: Coding Standards](#), [GNU Coding Standards - GNU Project - Free Software Foundation...](#)

<sup>11</sup> PJ2推荐提交文件夹按如下结构组织,压缩成 .zip 格式后提交

```
PJ2Handin_19302010001张三
├── README.pdf
├── auto.out #读入策略表后开始自动寻找的可执行文件
├── bonus
│   ├── Bonus.out #其他获得最优策略表的算法的训练程序(对应train.out)
│   ├── GUI.out #auto.out 的GUI版本
│   ├── scatter.png #散点图
│   └── src #其他算法的源代码, src文件夹下具体的文件名可以不同
│       ├── main.c
│       ├── main.h
│       ├── ...
│       ├── map.c
│       └── map.h
├── iteration.txt #训练过程的输出
├── src #遗传算法的源代码, src文件夹下具体的文件名可以不同
│   ├── main.c
│   ├── main.h
│   ├── ...
│   ├── map.c
│   └── map.h
├── strategy.txt #你最终的最优策略
└── train.out #遗传算法的训练程序

3 directories, 16 files
```

