

# 博弈论及其应用

## 期中大作业: 纳什均衡求解器

### 1 实验目的

- 掌握求解纳什均衡的相关算法
- 锻炼数学基础能力及编程求解问题的能力

### 2 实验内容

本次实验要求使用 Python 语言在给定代码框架下编程求解纳什均衡 (Nash Equilibrium, NE), 包括纯策略 NE 与混合策略 NE, 并提交相应源码、输出文件以及实验报告.

#### 2.1 实验环境

请确保可以在以下离线环境中运行你的代码, 如在使用保留数据评测时代码运行失败, 将认为相应测试样例全部未通过, 并扣除相应分数. 请注意: 除 Python 标准库外, 仅允许额外引入 `numpy` 和 `scipy` 两个包.

- Python 3.7.9
- `numpy` 1.19.4
- `scipy` 1.6.1

#### 2.2 框架文件

- 框架代码在 `main.py` 文件中, 要求完成 `nash(in_path, out_path)` 函数, 读取解析 `in_path` 对应的文件, 计算并将 NE 保存到 `out_path` 中, 不要求返回任何值.
- 部分测试数据存储在 `input` 文件夹下, `output` 文件夹保存输出结果.
- `examples` 文件夹下为一些输入输出样例.

## 2.3 输入格式

- 每个博弈的数据分别存储在以 .nfg 为后缀的文件中.
- 文件格式请参考 Gambit format 中 “The strategic game (.nfg) file format, payoff version” 一节中的说明, 请特别注意**收益序列的顺序**.

## 2.4 输出格式

- 每个博弈的 NE 分别保存到以输入文件名命名且后缀为 .ne 的中, 如输入文件为 1.nfg, 则输出文件名为 1.ne
- 若博弈有两个玩家, 则应输出**所有混合策略NE** (包括纯策略NE, 当混合策略 NE 有无穷多个时, 可表示为若干凸集的并集, 应输出每个凸集的极点). 若博弈有至少三个玩家, 则只要求输出**所有纯策略 NE**.
- 输出文件每一行对应一个 NE, 用 Python 中 list 对应的格式表示, list 中元素依次表示从第一个玩家到最后一个玩家采取每个动作的概率. 如有 2 个玩家, 玩家 1 有 3 种动作  $a_1, a_2, a_3$ , 玩家 2 有 2 种动作  $b_1, b_2$ , 那么 list 中应有 3+2 个元素, 为  $[p_1(a_1), p_1(a_2), p_1(a_3), p_2(b_1), p_2(b_2)]$ , 其中  $p_i(a)$  表示玩家  $i$  采取动作  $a$  的概率 (可使用小数或分数表示, 见 examples 文件夹下的样例).
- 当有多个 NE 时, 输出顺序不做要求.

## 2.5 样例解析

以 examples/example\_0.nfg 为例进行说明:

```
NFG 1 R "some
lines
of
comments" { "Player 1" "Player 2" } { 3 2 }

1 1 0 2 0 2 1 1 0 3 2 0
```

- 输入文件开头的 “NFG 1 R” 为统一标识, 无需处理.
- 接下来双引号内的部分为注释, 无需处理.
- 接下来的花括号内是以空格分隔的玩家名, 只需关注玩家数量, 名称无需处理. 有2个玩家.
- 接下来的花括号内是以空格分隔的玩家动作数. 玩家1有3个动作, 玩家2有2个动作.
- 接下来是策略收益, “1 1” 是玩家1采取行动1且玩家2采取行动1的双方收益, “0 2” 是玩家1采取行动2且玩家2采取行动1的双方收益, 等等. 对应的收益矩阵如下 (玩家1为row player, 玩家2为column player):

		Player 2	
		1 1	1 1
Player 1	0 2	0 2	0 3
	0 2	0 2	2 0

- 该博弈的纯策略 NE 为 $((1,0,0),(1,0))$ , 列方程可解得一个混合策略 NE 为 $((1,0,0),(0.5,0.5))$ , 进一步可得所有的 NE 构成一个凸集  $\{((1,0,0),(0.5 + \alpha, 0.5 - \alpha)) \mid 0 \leq \alpha \leq 0.5\}$ , 且  $((1,0,0),(1,0))$  和  $((1,0,0),(0.5,0.5))$  是该集合的极点. 所以输出文件 out\_0.ne 内容为

```
1,0,0,1,0
1,0,0,1/2,1/2
```

或者

```
1,0,0,1,0
1,0,0,0.5,0.5
```

## 2.6 友情提示

- 可参考Algorithmic Game Theory中第三章的相关内容.
- 测试时将使用额外测试数据, 并随机选择部分公开数据进行复现对比, 请不要提交如直接输出结果的这类违规程序.
- 在评价 NE 时会考虑小数舍入的问题, 不需要担心因小数点后若干位的误差而错判.
- 评分时将按照找到的 NE 数量给分(暂定), 视同学们的完成情况, 在一定运行时间内找到的 NE 正确且达到一定数量就可得到大部分分数. 完成较优秀者有额外附加分数 (可溢出到其他作业得分).
- 凡发现两份代码重复部分超过30%, 两份均记0分.**

## 3 提交方式

- 提交内容包括: main.py 以及其他可能的 Python 代码文件; report.pdf: 实验报告, 简要说明使用的算法、实现过程并做简单分析即可, 切忌长篇大论; output文件夹, 其中应包含input文件夹下所有博弈对应的输出文件.
- 将以上内容直接压缩为“学号\_姓名.zip”, 使用sftp提交, 若要更新提交内容, 请命名为“学号\_姓名\_v2.zip”, “学号\_姓名\_v3.zip”... 以此类推. 主机地址: www.lamda.nju.edu.cn 作业文件夹地址: /D:/personal\_web/qint/GTA\_fall21 账号: gta2021 密码: course01234!@#\$
- 提交截止时间为 2021.05.31 23:55, 不接受补交.**