

离散优化建模：作业四

皇家狩猎

1 问题描述

汉献帝希望由刘备组织一场狩猎来庆祝自己的生日。为了令狩猎举办成功，刘备需要谨慎地为朝廷要员匹配相应的马，令他们能够愉快地享受这次狩猎。朝廷要员的数量有可能比马的数量还要多，所以有些人可能不能骑马。在这场狩猎中也有以下一些不成文的规定需要被遵守：

- 皇上需要比其他人都更享受这次狩猎。
- 除非马的数量不够，否则所有人都应该骑马。
- 如果一个要员比另外一个要员阶级更高，那么就应该符合以下任意一种情况：(a) 等级较高的要员骑的马应该比等级较低的要员骑的马更骏伟；(b) 等级较低的要员没有骑马；(c) 两个人都没有骑马。
- 如果一匹马比另外一匹马更快，那么就应该符合以下任意一种情况：(a) 骑快马的人的骑马技术不应该比骑慢马的人的更差；(b) 没有人骑快马；(c) 两匹马都没有人骑。

这次的目标是要使所有参加这次狩猎的人的愉悦度最大化。实际上这些约束通常很难同时被满足。所以最后的约束是可以被违反的，不过每次违反都会对目标数值有 100 的降低惩罚。

2 数据格式说明

皇家狩猎的输入是 `data/royalhunt_p.dzn` 的数据文件，其中 p 是问题的序号。其中 n 代表朝廷要员的数目（第一个朝廷要员就是皇上）。 $rank$ 是一个数组，代表不同的朝廷要员的等级（数值越高代表等级越高）。 $ability$ 是一个数组，代表不同的朝廷要员的骑术水平（数值越高代表水平越高）。 m 代表的是马的数量。 $beauty$ 是一个数组，代表不同的马的骏伟程度（数量越高代表越骏伟）， $speed$ 是一个数组代表不同的马的速度。 $enjoy$ 是一个二维数组，代表不同的宫廷要员骑不同的马的愉悦程度。如果其中一项为负数，则表示对应的马匹不能被分派给这个人。

数据变量与决策变量的声明如下：

```
int: n; % number of court members
set of int: COURT = 1..n;
int: emperor = 1;
array[COURT] of int: rank;
```

```

array[COURT] of int: ability;

int: m; % number of horses
set of int: HORSE;
array[HORSE] of int: beauty;
array[HORSE] of int: speed;

array[COURT,HORSE] of int: enjoy;

```

数据文件的样本如下:

```

n = 6;
rank = [8,5,5,4,2,2];
ability = [0,1,0,1,0,1];
m = 5;
beauty = [1,5,3,8,8];
speed = [6,3,5,4,4];

enjoy = [| 3,4,5,7,3
        | 1,6,3,4,8
        | 2,3,4,3,3
        | 9,6,2,3,4
        | 4,-1,3,3,3
        | 4,4,4,4,4 |];

```

其中里面是 6 个宫廷要员和 5 匹马的情况。你的模型输出应该给出每个骑手（即宫廷要员）对应分配到马的序号（或者是 0 代表他们没有分配到马），还要输出最终的目标数值。比如它的输出可以是（这不一定是最优解）:

```

horse = [4,2,5,3,1,0];
obj = -178;

```

其中这个解两次违反了对于马的约束：马匹 1 比马匹 2 更快，不过 5 号骑手比马匹 2 的骑手 2 号的骑术水平更低。同样地，马匹 1 比马匹 2 更快，不过 1 号马的骑手比 3 号马的骑手骑术水平更低。所以总的愉悦度是 $7 + 6 + 3 + 2 + 4 - 200 = -178$ 。

3 指引

你可以编辑 `royalhunt.mzn` 模型文件来解决上述优化问题。你实现的模型 `royalhunt.mzn` 可以用提供的文件进行测试。在 MINIZINC IDE 中，你可以点击 *Run* 按钮在本地测试和运行。或者在命令行中输入

```
mzn-gecode ./royalhunt.mzn ./data/<inputFileName>
```

进行本地测试和运行。在两种情况下，你的模型都是用 MINIZINC 进行编译然后用 GECODE 求解器求解。

参考资料 你可以在 `data` 文件夹下找到讲义中的几个问题实现。

提交作业 这次的作业包含有 7 个答案提交部分和 1 个模型提交部分。对于答案提交部分，我们将会提交求解器求解你的模型所得到的最好 / 最后的答案，然后检查它的正确性和得分。对于模型提交部分，我们将会提交你的模型文件 (`.mzn`) 然后用一些隐藏的数据文件来做进一步检查。

在 MINIZINC IDE，点击 *coursera* 图标可以用于提交作业。若采用命令行方式，`submit.py` 可以用于提交作业。无论采用那种方法，你都需要根据本指引中的要求完成作业各部分的 MiniZinc 模型。你可以多次提交，最终作业分数是你的最高的一次。¹作业的打分过程可能需要几分钟，请耐心等待。你可以在课程网站上的编程作业 版块查看你的作业提交状况。

4 软件要求

为了完成作业，你需要安装 MINIZINC 2.1.x 和 GECODE 5.0.x 求解器。这些软件都会包含在 MINIZINC IDE 2.1.2 (<http://www.minizinc.org>) 的集成版本中。如果你需要通过命令行提交作业，你需要安装 Python 3.5.x。

¹答案提交部分并没有次数限制。但是，模型提交部分只能提交有限次。