离散优化建模：作业三
美人计：舞蹈

1 问题描述

王允密谋帮助打败董卓。他想用计令董卓和他的大将吕布同时爱上他的养女貂蝉。貂蝉有沉鱼落雁之容。他计划邀请吕布到府上，让貂蝉为其表演舞蹈，使吕布为之倾心。这个舞蹈的陷阱是美人计的其中一部分。

貂蝉为此需要准备极具诱惑力的舞蹈。在舞蹈的每一部分她需要选择她手部的动作，脚步和脸上的表情。而这一系列的动作有不同的约束。

由于吕布很容易会感到厌倦，她需要选择舞蹈的长度使舞蹈最吸引人。通常来说，舞蹈越长，它必须设计得吸引人。同一个动作使用的次数也是有上限的，否则吕布会渐渐觉得十分无聊。

她的脚步有如下的选择：旋转, 跳跃, 华尔兹, 曲膝, 跳跃 的预备 和直立 。她必须在跳跃 前先预备 。紧接旋转 之后她只能做曲膝, 预备 或者 直立 。紧接跳跃 之后她只能做旋转, 华尔兹, 或者直立 。她最多可以连续做三次华尔兹 。她不可以在曲膝 后马上做预备 动作。最后，在任意一个的华尔兹 和随后的曲膝 动作之间必须有一个直立 动作。

她手部的动作可以是：招手 ，伸 手，举 手，手环抱 身体，和放下 。她只可以在伸 或者举 手之后招手 。她只可以在招手, 环抱, 或者放下 之后环抱 。除了放下 之外，她不可以连续两次做相同的动作。

脸部的动作上她可以是：微笑 ，眨眼 ，抛媚眼 ，思索 ，粲然一笑 或者平静 。除了平静 之外，她不可以连续做两次相同的表情。她不可以紧接在微笑 或者抛媚眼 之后思索 。

她在每一步的脚部和手部动作的组合决定了她舞蹈的得分。她在每一步的手部动作和脸部表情决定了她舞蹈的吸引程度。

有一些脚步和手部动作的组合是不可能的。同样地，有一些手部动作和脸部表情是令人非常不快的，所以她需要尽量避免。

在舞蹈结束后她的动作需要一直保持在基础的状态，也就是 直立 ，放下 ，和平静 。

她的目的是令到她的舞蹈的吸引力最大化。这个吸引力的数值是由她舞蹈的得分，加上舞蹈的吸引程度，然后减去吕布的无聊程度和舞蹈长度之积所得到的。

2 数据格式说明

美人计问题的输入是 `data/dancetrap_p.dzn` 文件，其中 p 是问题的序号。其中，$maxlen$ 是舞蹈的最大长度，$boredom$ 是吕布的无聊程度。$dance\_value$ 是一个二维数组，其中的数值代表手部，腿部动作的组合所对应的舞蹈的得分（负数数值代表其对应的组合是不允许的）。同样地，

$entice\_value$ 是一个二维数组，其中的数值代表手部动作和脸部表情的组合所对应的舞蹈的吸引程度（负数数值代表其对应的组合是不被允许的）。$maxlegs$ 是一个一维数组，代表每个脚步最多可以使用的次数。$maxarms$，$maxface$ 与此类似，分别代表着手部动作和脸部表情最多可以使用的次数。

输入的数据保证直立 和放下 组合对应的舞蹈的出色程度是 0，也保证放下 和平静 组合对应的舞蹈的吸引程度是 0。同时，输入也保证每一个基础状态（直立, 放下, 和 平静 ）使用的最多次数至少是 $maxlen$，所以在整个舞蹈中维持基础状态也是可以的。

所以数据与决策变量的声明如下：

```
enum LEGS = {spin, leap, waltz, curtsey, prep, stand};
enum ARMS = {beckon, out, up, wrapped, neutral};
enum FACE = {smile, wink, batt, think, glow, blank};

int: maxlen;
set of int: STEP = 1..maxlen;
array[LEGS] of int: maxlegs;
array[ARMS] of int: maxarms;
array[FACE] of int: maxface;
constraint assert(maxlegs[stand] >= maxlen, "maxlegs[stand] smaller than maxlen");
constraint assert(maxarms[neutral] >= maxlen, "maxarms[neutral] smaller than maxlen");
constraint assert(maxface[blank] >= maxlen, "maxface[blank] smaller than maxlen");
array[LEGS,ARMS] of int: dance_value;
array[ARMS,FACE] of int: entice_value;
constraint assert(dance_value[stand,neutral] = 0, "incorrect dance_value array");
constraint assert(entice_value[neutral,blank] = 0, "incorrect entice_value array");
int: boredom; % how bored each step makes the viewer

var STEP: len;
array[STEP] of var LEGS: legs;
array[STEP] of var ARMS: arms;
array[STEP] of var FACE: face;
```

数据文件的例子如下：

```
maxlen = 10;
boredom = 8;
dance_value = [| -1,  5,  3,  2,  2
```

```
                |  2,  4,  2, -1,  1
                |  4,  4,  0, -1,  2
                |  5,  1, -1,  2,  2
                |  0,  0,  0,  0,  0
                |  2,  1,  1,  2,  0 |];
entice_value = [|  4,  6,  5, -1,  3, -1
                 |  3,  2,  3,  3,  1,  1
                 |  2,  4,  5,  4, -1,  0
                 |  5,  3,  2,  5,  6,  0
                 |  4,  2,  1,  4, -1,  0 |];
maxlegs = [3, 3, 6, 2, 4, 20];
maxarms = [3, 3, 3, 3, 20];
maxface = [5, 2, 2, 7, 3, 20];
```

上面的数据说明，舞蹈的最大长度是 10，无聊的程度是 8，而且貂蝉可以用**华尔兹** 最多 6 次，用**曲膝** 最多 2 次。

你的模型应该输出所有的决策变量和对应的目标函数数值。比如，你的模型可能有以下输出（不一定是最优解）：

```
len = 9;
legs = [stand, stand, spin, curtsey, spin, curtsey, waltz, waltz, spin, stand];
arms = [wrapped, up, out, beckon, out, beckon, out, beckon, wrapped, neutral];
face = [glow, batt, smile, batt, smile, wink, smile, wink, glow, blank];
obj = 4;
```

注意到舞蹈的长度只有 9，而在第 10 步的时候貂蝉是处在基础动作的状态下的。你可以检查整个舞蹈满足所有的约束，而目标函数数值是 4，这是由舞蹈中每一步的得分 [2, 1, 5, 5, 5, 5, 4, 4, 2, 0] 和每一步的吸引程度 [6, 5, 3, 5, 3, 6, 3, 6, 6, 0] 之后，减去无聊程度 8×9 所得到的。

3 指引

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

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

进行本地测试和运行。两种情况下，你的模型都是用 MiniZinc 进行编译同时用 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例（的数据文件）。

提交作业 这次的作业包含有 5 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，请耐心等候。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。