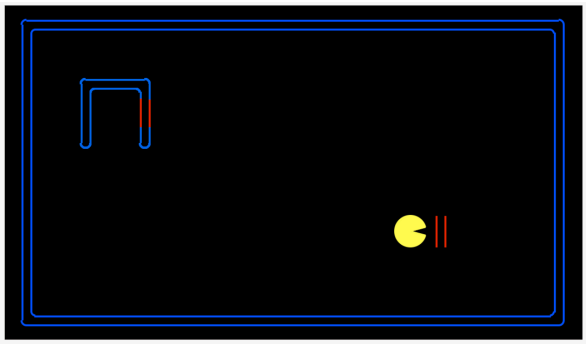
13 Inference in Bayes Net I



Much truth is unseen.
How will Pacman become sure?
Bayes Net Inference

本节实验目标：熟悉贝叶斯网络的一种构造方法，为赋能吃豆人完成一个游戏做准备。

报告说明（重要）：本部分“贝叶斯网络”包含两周内容，本节课为第一周内容，请留意该文档目录中标有“报告应实现任务[分值]”的部分，请在实现后及时保存文件和运行结果截图，在下一周课程报告中一并提交。

Introduction

在本堂课中，你将学习用python程序表示贝叶斯网络的一种方法，实现贝叶斯网络中的逐点相乘算法。

你可以通过以下形式的命令为特定测试运行自动评分器：

```
python autograder.py -t test_cases/q1/1-small-board
```

你将要编辑的文件	
factorOperations.py	Operations on Factors (join, eliminate, normalize).
inference.py	Inference algorithms (enumeration, variable elimination, likelihood weighting).
bayesAgents.py	Pacman agents that reason under uncertainty.
你应该阅读但不要编辑的文件：	
bayesNet.py	The BayesNet and Factor classes.
你可以忽略的文件	
graphicsDisplay.py	Graphics for Pacman
graphicsUtils.py	Support for Pacman graphics
textDisplay.py	ASCII graphics for Pacman
ghostAgents.py	Agents to control ghosts
keyboardAgents.py	Keyboard interfaces to control Pacman

你将要编辑的文件	
layout.py	Code for reading layout files and storing their contents
autograder.py	Project autograder
testParser.py	Parses autograder test and solution files
testClasses.py	General autograding test classes
test_cases/	Directory containing the test cases for each question
bayesNets2TestClasses.py	Project's specific autograding test classes

Treasure-Hunting Pacman 游戏说明

吃豆人进入了一个神秘的世界。 最初，整个地图是不可见的。 当他探索地图时，他会了解有关相邻格子的信息。 地图上有两个房子：一个可能大部分墙壁是红色的鬼屋和一个可能大部分墙壁是蓝色的食物屋。 Pacman 的**目标是在避开鬼屋的同时进入食物屋。**

Pacman 将根据他的观察来推断哪个房子是鬼屋哪个是食物屋，并在冒险进屋或收集更多证据之间的权衡作出决定。 为此，你将使用贝叶斯网络实现概率推理。

请运行以下代码自己试玩几轮，结合该例子，理解下面Question 1用到的变量：

```
python hunters.py -p KeyboardAgent -r
```

Bayes Nets and Factors

首先，我们把这个问题用贝叶斯网络建模。查看 `bayesNet.py` 了解你将使用的类 - `BayesNet` 和 `Factor` 。 你还可以运行此文件 `python bayesNet.py` 以查看下面这个示例的 `BayesNet` 和相关的 `Factors` 。

你应该**查看** `bayesNet.py` **中的** `printStarterBayesNet` **函数来对比观察** `python bayesNet.py` 的输出 - 有一些有用的注释可以让你在本堂课中显得更轻松。

在 `printStarterBayesNet` 这个函数中创建的贝叶斯网络将如下图所示：

$(Raining -- > Traffic < -- Ballgame)$

一些术语如下：

- `Bayes Net`：这是一个有向无环图和一组条件概率表的概率模型，上面的交通贝叶斯网络就是一个例子。
- `Factor`：这存储了一个概率表，尽管表中条目的总和不一定是 1。一个因子的一般形式是 $P(X_1, ..., X_m, y_1, ..., y_n | Z_1, ..., Z_p, w_1, ..., w_q)$。 **其中小写字母形式的变量已经获得赋值**。 对于 X_i和 Z_j 变量的每个可能的赋值，这个因子存储一个数值。 Z_j, w_k是这个因子的条件变量，X_i, y_l是非条件变量。
- 条件概率表 Conditional Probability Table (CPT): 这是一个满足以下两个性质的因子。
  1. 对于条件变量的每个赋值，其条目的总和必须为 1
  2. 非条件变量只有一个（回忆一下，贝叶斯网络中的CPT的形式，每个变量结点有一个CPT）

例如：交通贝叶斯网络存储以下条件概率表: $P(Raining), P(Ballgame), P(Traffic | Ballgame, Raining)$.

项目中可能需要用到的变量

在 `BayesAgents.py` **的顶部，我们已经定义了一些该项目可能要用到的变量和其取值，有需要可以直接使用。**

```
X_POS_VAR = "xPos"
FOOD_LEFT_VAL = "foodLeft"
GHOST_LEFT_VAL = "ghostLeft"
X_POS_VALS = [FOOD_LEFT_VAL, GHOST_LEFT_VAL]

Y_POS_VAR = "yPos"
BOTH_TOP_VAL = "bothTop"
```

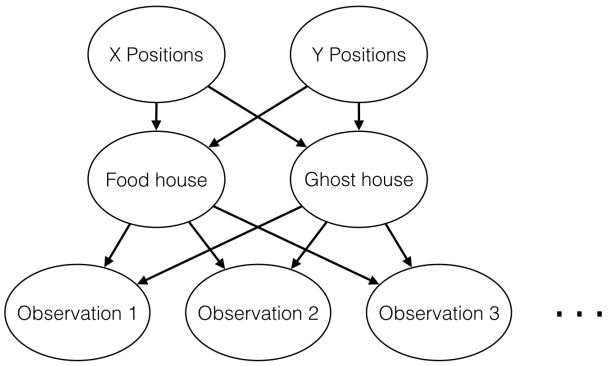
```
BOTH_BOTTOM_VAL = "bothBottom"
LEFT_TOP_VAL = "leftTop"
LEFT_BOTTOM_VAL = "leftBottom"
Y_POS_VALS = [BOTH_TOP_VAL, BOTH_BOTTOM_VAL, LEFT_TOP_VAL, LEFT_BOTTOM_VAL]

FOOD_HOUSE_VAR = "foodHouse"
GHOST_HOUSE_VAR = "ghostHouse"
HOUSE_VARS = [FOOD_HOUSE_VAR, GHOST_HOUSE_VAR]

TOP_LEFT_VAL = "topLeft"
TOP_RIGHT_VAL = "topRight"
BOTTOM_LEFT_VAL = "bottomLeft"
BOTTOM_RIGHT_VAL = "bottomRight"
HOUSE_VALS = [TOP_LEFT_VAL, TOP_RIGHT_VAL, BOTTOM_LEFT_VAL, BOTTOM_RIGHT_VAL]

OBS_VAR_TEMPLATE = "obs(%d,%d)"

BLUE_OBS_VAL = "blue"
RED_OBS_VAL = "red"
NO_OBS_VAL = "none"
OBS_VALS = [BLUE_OBS_VAL, RED_OBS_VAL, NO_OBS_VAL]

ENTER_LEFT = 0
ENTER_RIGHT = 1
EXPLORE = 2
```

Question 1: Bayes Net Structure 报告应实现任务[1分]

- 说明：该小题截图只需截 python autograder.py -q q1 的运行结果，获得评分器给的3分中，这道题才算得它占的1分。

在 `BayesAgents.py` 中实现 `constructBayesNet` 函数。 它构造了一个空贝叶斯网络，其结构如下所述。（我们将在Question 2中指定其实际因子(条件概率值)。）

Treasure hunting 世界根据以下贝叶斯网生成：



如果这看起来很复杂，但请不要担心！我们会一步一步来。如 `constructBayesNet` 的代码中所述，我们通过列出所有变量、它们的值以及它们之间的边来构建**空结构**。该图显示了变量和边，但是它们的值呢?

- X Positions 决定哪个房子在棋盘的哪一边。它的取值要么是 `foodLeft`(食物屋在左边)，要么是 `ghostLeft`(幽灵屋在左边)。
- Y Positions 决定了房屋的垂直方向朝向。它同时表示两栋房屋的垂直位置，其取值范围为以下四个：`bothTop`、`bothBottom`、`leftTop` 和 `leftBottom`。`leftTop` 的意思是：棋盘左边的房子开口朝上，**同时棋盘右边的房子开口朝下。**

- 食物屋（Food house）和鬼屋（Ghost house）指定了两间屋子的实际位置。它们受“X Positions”和“Y Positions”影响。
- Observations 是吃豆人在棋盘上走动时所做的观察。请注意，这些节点有很多，每个棋盘位置都有一个节点。如果给定位置没有房子墙壁，则相应的观察结果为无；否则它要么是红色要么是蓝色，颜色的精确分布取决于房子的种类。

要测试和调试你的代码，请运行

```
python autograder.py -q q1
```

Question 2: Bayes Net Probabilities [不算分，但你得实现了才能做Question 3 ^v^]

在 `bayesAgents.py` 中实现 `fillYCPT` 函数。 它采用你在上一个问题中构建的贝叶斯网络，并指定控制 Y 位置变量的因子。（我们已经为你填写了 X position、house和observation factors。）

有关如何构造因子的示例，请查看 `fillXCPT` 中 X positions的因子的实现。

Y positions 由值 `BOTH_TOP_VAL, BOTH_BOTTOM_VAL, LEFT_TOP_VAL, LEFT_BOTTOM_VAL` 给出。 这些变量及其相关概率 `PROB_BOTH_TOP, PROB_BOTH_BOTTOM, PROB_ONLY_LEFT_TOP, PROB_ONLY_LEFT_BOTTOM` 由文件 `bayesAgents.py` 顶部的常量提供。

如果你有兴趣，可以查看房屋位置的计算。 你需要记住的是，每栋房屋都可以位于以下四个位置之一：top-left, top-right, bottom-left, bottom-right。

提示： Y position 因子中只有四个条目，因此你可以手动指定每个条目。

要测试和调试你的代码，请运行

```
python autograder.py -q q2
```

Question 3: Join Factors 报告应实现任务[1分]

- **说明：该小题截图只需截 python autograder.py -q q3 的运行结果，获得评分器给的5分中，这道题才算得它占的1分。**

在 `factorOperations.py` 中实现 `joinFactors` 函数。 它接受一个 `Factor` 列表并返回一个新的 `Factor`，其概率条目是输入 `Factors` 的相应行的乘积。

`joinFactors` 可以用作乘法规则，例如，如果我们有一个$P(X|Y)$ 形式的因子和另一个 $P(Y)$ 形式的因子，那么合并这些因子将产生$P(X,Y)$。 因此， `joinFactors` 允许我们合并条件变量（在本例中为 Y）的概率。 但是，你不应该假设只能在概率表上调用 `joinFactors` ——我们还可以在行总和不为 1 的 `Factors` 上调用 `joinFactors` 。

要测试和调试你的代码，请运行

```
python autograder.py -q q3
```

在调试期间运行特定测试可能会很有用，以便只看到一组因子打印出来。 例如，要仅运行第一个测试，请运行:

```
python autograder.py -t test_cases/q3/1-product-rule
```

Hints and Observations:

- 你的 `joinFactors` 应该返回一个新的 `Factor`.
- 运用 `joinFactors` 的一些例子:
  - joinFactors$(P(X|Y),P(Y)) = P(X,Y)$
  - joinFactors$(P(V,W|X,Y,Z),P(X,Y|Z)) = P(V,W,X,Y|Z)$
  - joinFactors$(P(X|Y,Z),P(Y)) = P(X,Y|Z)$
  - joinFactors$(P(V|W),P(X|Y),P(Z)) = P(V,X,Z|W,Y)$
- 对于一般的 `joinFactors` 操作，返回的因子中哪些变量是非条件变量的？ 哪些变量是条件变量？这给你定义新的 `Factor` 时的非条件和有条件变量时有何启发？
- 因子存储一个 `variableDomainsDict`，它将每个变量映射到它可以被赋的值列表（它的域）。 一个 `Factor` 从实例化它的 `BayesNet` 中获取它的 `variableDomainsDict` 。 因此，它包含了 `BayesNet` 的所有变量，*不*仅包含 `Factor` 使用的非条件变量

和条件变量。 对于这个问题，你**可以假设所有输入的`Factor`都来自同一个`BayesNet`，所以它们的`variableDomainsDicts`都是一样的。**

- Useful functions:（具体可以查看bayesNet.py里的Factor类）

```
Factor.getAllPossibleAssignmentDicts
Factor.getProbability
Factor.setProbability
Factor.unconditionedVariables
Factor.conditionedVariables
Factor.variableDomainsDict
```

本节课实现内容保存说明 - 为提交第六次上机报告做准备

- **第14周上机课结束后，一并提交本部分”贝叶斯网络“的报告，该报告包含本节课的Q1,Q3的代码和运行截图，请按Q1,Q3大标题下的说明进行截图，保存。**
- **本节课实现的代码文件也请保存，你将需要把其中实现的代码加到第14周相应的代码文件中跟第14周的任务实现一并进行提交。**