知识表示：用产生式系统表示“动物识别系统”

# 1实验目的

1. 了解产生式表示法；
2. 了解产生式系统的组成与原理；
3. 了解如何编程构造产生式系统，并利用产生式系统求解问题；
4. 思考产生式系统的优缺点。

# 2实验要求

本次试验后，要求学生能：

1. 用python编写程序实现产生式系统，并能进行动物识别；
2. 了解产生式系统的组成与原理，理解产生式系统中推理机的工作流程。

# 3实验原理

## 3.1产生式

产生式常用于表示事实、规则以及它们不确定性度量，适合表示事实性知识和规则性知识。

产生式与蕴含式相似，实际上蕴含式是特殊的一种产生式。产生式除了表示逻辑蕴含外，还可以包含操作、规则、变换、算子、函数等，例如“如果炉温超过上限，则立即关闭风门”是产生式，却不是蕴含式。

另外，蕴含式只能表示确定的知识，其真值要么为真要么为假，产生式却可以表示不确定性知识。对于逻辑蕴含式，其前提条件的匹配必须是精确的，但产生式表示系统中，条件的匹配可以精确，也可以只按照某种算法模糊地匹配。

（1）确定性规则知识的产生式基本形式为：

**IF** P **THEN** Q

或者

P→Q

例如：R1：**if** 动物有毛发 **then** 动物是哺乳动物

（2）不确定性规则知识的产生式基本形式为：

**IF** P **THEN** Q （置信度）

或者

P→Q （置信度）

（3）确定性事实性知识的产生式一般用三元组表示：

（对象，属性，值）

或者

（关系，对象1，对象2）

（4）不确定性事实性知识的产生式一般用四元组表示：

（对象，属性，值，置信度）

或者

（关系，对象1，对象2，置信度）

## 3.2产生式系统

把一组产生式放在一起配合使用，一个产生式的结论可以作为另一个产生式的已知事实，从而求得问题的解，这样的系统称为产生式系统（production system），由规则库、综合数据库、推理机三部分组成。

（1）规则库

用于描述相应领域内知识的产生式集合称为规则库。

（2）综合数据库

综合数据库又称为事实库、上下文、黑板等。它是一种用于存放问题求解过程中各种当前信息的数据结构，如问题的初始状态、原始证据、推理中得到的中间结论以及最终结论。

（3）推理机

推理机由一组程序组成，除了推理算法，还控制整个产生式系统的运行，实现对问题的求解，主要功能为：1.推理；2.冲突消解；3.执行规则；4.检查推理终止条件。

# 4实验步骤

## 4.1实验介绍

本实验以一个动物识别系统为例，介绍产生式系统求解问题的过程。这个动物识别系统是识别老虎、金钱豹、斑马、长颈鹿、企鹅、鸵鸟、信天翁等七种动物的产生式系统。

用户输入待识别动物的若干个特征，实现产生式系统的自动推理识别。

## 4.2构造产生式系统

首先，根据动物识别的专家知识，建立如下**规则库**。

R1：if 动物有毛发 then 动物是哺乳动物

R2：if 动物有奶 then 动物是哺乳动物

R3：if 动物有羽毛 then 动物是鸟

R4：if 动物会飞 and 会生蛋 then 动物是鸟

R5：if 动物吃肉 then 动物是食肉动物

R6：if 动物有犀利牙齿 and 有爪 and 眼向前方 then 动物是食肉动物

R7：if 动物是哺乳动物and有蹄then动物是有蹄类动物

R8：if 动物是哺乳动物and反刍then动物是有蹄类动物

R9：if 动物是哺乳动物and是食肉动物and有黄褐色 and 有暗斑点 then 动物是豹

R10：if 动物是哺乳动物 and是食肉动物and有黄褐色 and有黑色条纹 then 动物是虎

R11：if动物是有蹄类动物 and 有长脖子and有长腿and有暗斑点 then 动物是长颈鹿

R12：if 动物是有蹄类动物 and有黑色条纹 then 动物是斑马

R13：if 动物是鸟and不会飞 and有长脖子and有长腿 and有黑白二色 then 动物是鸵鸟

R14：if 动物是鸟 and不会飞 and会游泳 and有黑白二色 then 动物是企鹅

R15：if 动物是鸟 and善飞 then 动物是信天翁

虽然只是识别7种动物，但是设计了15条规则。其基本思想是先利用R1~R8进行比较粗略的分类，如“哺乳动物”、“鸟”等，然后根据R9~R15判断具体的动物。

这样做的好处有两点：一是当已知事实不完全时，虽然不能得到最终结论，但是可以得到粗略的分类结果；二是当需要增加对其他动物的识别时，只需加入该种动物个性方面的知识，如R9~R15，R1~R8可以直接利用。

之后，建立一个集合作为**综合数据库**，用于存放问题求解过程中产生的各种信息。

最后，编程构造**推理机**，根据规则库进行推理，由给出的动物特征识别动物。

## 4.3编程实现

### 4.3.1实现规则库。

为各个规则（rule1~rule15）建立对应的集合（r1~r15），将各个规则前提中的特征加入对应集合中，最后将所有规则对应的集合汇总成规则库。

# 特征

Features = [" ", "有毛发", "有奶", "有羽毛", "会飞", "会下蛋", "吃肉", "有犬齿", "有爪", "眼盯前方", "有蹄",

"反刍", "黄褐色", "暗斑点", "黑色条纹", "长脖子", "长腿", "不会飞", "有黑白二色", "会游泳", "善飞",

"哺乳动物", "鸟", "食肉动物", "蹄类动物"]

# 动物种类

Animals = [" ", "老虎", "金钱豹", "斑马", "长颈鹿", "企鹅", "鸵鸟", "信天翁"]

# 规则

r1 = {"有毛发"} # 哺乳动物

r2 = {"有奶"} # 哺乳动物

r3 = {"有羽毛"} # 鸟

r4 = {"会飞", "会下蛋"} # 鸟

r5 = {"吃肉"} # 食肉动物

r6 = {"有犬齿", "有爪", "眼盯前方"} # 食肉动物

r7 = {"哺乳动物", "有蹄"} # 蹄类动物

r8 = {"哺乳动物", "反刍"} # 蹄类动物

r9 = {"哺乳动物", "食肉动物", "黄褐色", "暗斑点"} # 金钱豹

r10 = {"哺乳动物", "食肉动物", "黄褐色", "黑色条纹"} # 老虎

r11 = {"蹄类动物", "长脖子", "长腿", "暗斑点"} # 长颈鹿

r12 = {"蹄类动物", "黑色条纹"} # 斑马

r13 = {"鸟", "长脖子", "长腿", "不会飞", "有黑白二色"} # 鸵鸟

r14 = {"鸟", "会游泳", "不会飞", "有黑白二色"} # 企鹅

r15 = {"鸟", "善飞"} # 信天翁

# 规则库

rules = [r1, r2, r3, r4, r5, r6, r7, r8, r9, r10, r11, r12, r13, r14, r15]

### 4.3.2建立综合数据库

综合数据库用于保存初始信息、原始证据、推理中得到的中间结论，以及最后的识别结论。输出提示信息，让用户按照要求输入待识动物的特征信息，保存在综合数据库known中。

# 综合数据库

known = []

# 提示信息

print("欢迎使用动物识别系统！")

print("本识别系统可识别的动物种类：")

for info in Animals:

print(info, end=' ')

print("可选择提供的已知信息：")

for info in Features:

print(info, end=' ')

print("")

# 用户输入已知特征

temp = input("请输入已知事实：（用空格将各个特征隔开）")

# 已知特征加入综合数据库

known = set(temp.split(" "))

### 4.3.3实现推理机

定义函数rule\_result(bumber)，作用是找出第number条规则对应的推理结果。根据规则序号number，选择对应的规则进行推理，返回推理出的新特征或推理出的动物种类。

def rule\_result(number):

if number == 1 or number == 2:

print("由规则%d推出该动物是哺乳动物" % number)

return "哺乳动物"

if number == 3 or number == 4:

print("由规则%d推出该动物是鸟" % number)

return "鸟"

if number == 5 or number == 6:

print("由规则%d推出该动物是肉食动物" % number)

return "肉食动物"

if number == 7 or number == 8:

print("由规则%d推出该动物是蹄类动物" % number)

return "蹄类动物"

if number == 9:

print("由规则%d推出该动物是金钱豹" % number)

return "金钱豹"

if number == 10:

print("由规则%d推出该动物是老虎" % number)

return "老虎"

if number == 11:

print("由规则%d推出该动物是长颈鹿" % number)

return "长颈鹿"

if number == 12:

print("由规则%d推出该动物是斑马" % number)

return "斑马"

if number == 13:

print("由规则%d推出该动物是鸵鸟" % number)

return "鸵鸟"

if number == 14:

print("由规则%d推出该动物是企鹅" % number)

return "企鹅"

if number == 15:

print("由规则%d推出该动物是信天翁" % number)

return "信天翁"

如果规则库中表示第i条规则的集合Ri**包含于**当前综合数据库known（即，Ri与known的交集等于Ri），说明第i条规则的前提条件可以被满足，可以用第i条规则进行推理，调用rule\_result(i)得出推理结果。将第i条规则的推理结果加入到综合数据库，也就是集合known中，并将第i条规则标记为“已匹配”，之后不再重复使用它进行推理。

推理机循环重复上述操作，不断比较各个规则的集合与当前known集合。直到没有新的规则匹配成功（信息不足，只能输出大致的分类）；或者Rule9~Rule15比配成功（说明已经成功推理出动物的具体类别）。最后输出推理结果

# 循环搜索是否有匹配的规则并进行推理

flag = 1

j = 0 # 循环中规则序号计数

new\_fact = ""

while flag:

flag = 0

for k in range(0, 14):

if set(rules[k]) & known == set(rules[k]):

flag = 1 # 本次找到了一条符合的规则，执行完本规则后需再次搜索是否有其他规则

rules[k] = "Matched" # 从规则库中删除已匹配过的规则

j = k + 1 # 列表由0开始计数，规则由1开始计数

new\_fact = rule\_result(j)

known.add(new\_fact)

# 终止条件:如果使用了规则R9~R15说明已经得到了具体的动物种类

if j >= 9:

break

# 打印出推理结果

if new\_fact == "":

print("信息严重不足或错误，无法进行任何推理！")

elif j < 9:

print("信息不足，无法准确识别，只能得出动物是" + new\_fact)

else:

print("最终识别出来的动物是" + new\_fact)

# 5实验结果

运行程序后，运行结果如下：

（1）信息充足时的推理结果。

由事实：暗斑点、长脖子、长腿、有奶、有蹄，可以推出是长颈鹿。

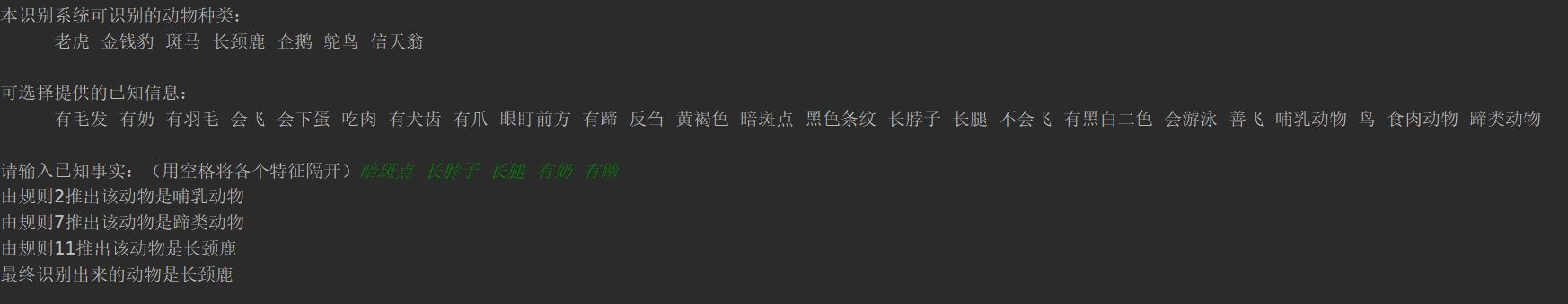


图1 信息充足

（2）信息不足时，只能推出大致的分类。

由事实：有奶、有蹄、暗斑点，只能推出是蹄类动物

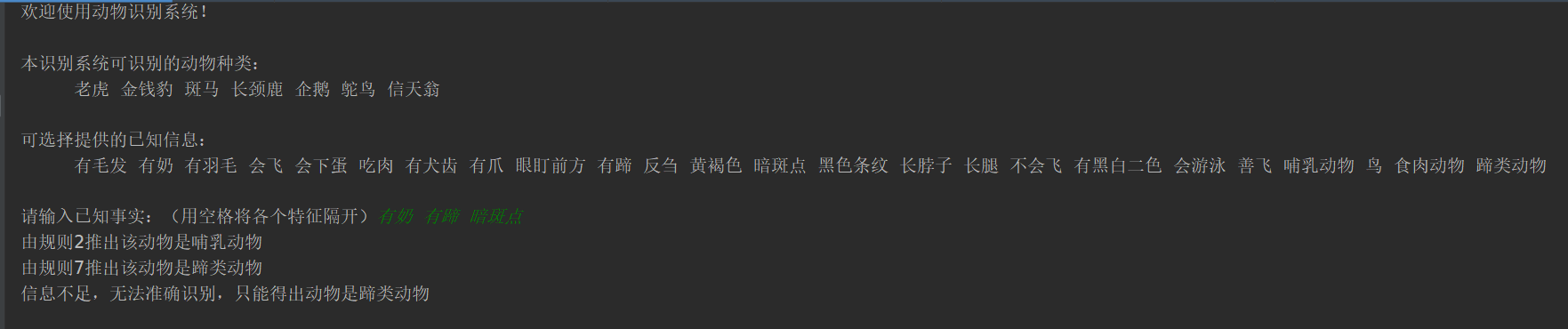


图2 信息不足

（3）信息严重不足，或输入错误时，无法推理。

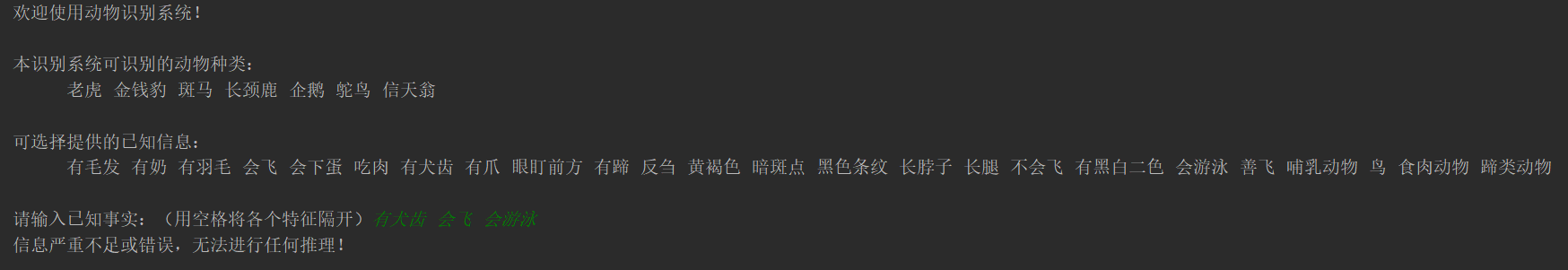


图3 无法推理