基于分解法模板的逻辑问题求解

# 基于图的建模

## 分解法模型表示

# 基于ASP的求解

## 图到ASP实现的转换

## 基于分解法的ASP模板

|  |
| --- |
| knowledge.  common\_sense.  answer.  2{options； rule}2 :- answer.  MINO{option(1..On)}MAXO :- options.  MINS{situation(1..Sn)}MAXS :- rule.  #show option/1. |

## 代码转换为约束满足或父节点在头部的情况

# 案例的应用

## 应用结果

## 分解法与约束满足法编码及求解的对比

# 总结与不足

# 附录

## 火腿还是猪排

### 问题

阿德里安、布福德和卡特三人去餐馆吃饭，他们每人要的不是火腿就是猪排。

（1） 如果阿德里安要的是火腿，那么布福德要的就是猪排。

（2） 阿德里安或卡特要的是火腿，但是不会两人都要火腿。

（3） 布福德和卡特不会两人都要猪排。

谁昨天要的是火腿，今天要的是猪排？

### 分解法解答

person(a).

person(b).

person(c).

food(ham).

food(pork).

day(yes).

day(tod).

1{eat(P, F, D):food(F)}1 :- person(P), day(D).

answer.

2{goal; rule}2 :- answer.

1{answer(P) : person(P)}3 :- goal.

3{eat(P, ham, yes); eat(P, pork, tod); person(P)}3 :- answer(P).

6{p(1, yes); p(2, yes);p(3, yes); p(1, tod);p(2, tod);p(3, tod)}6 :- rule.

1{p(11,D);p(12,D)}1 :-p(1,D).

2{eat(a,ham,D);eat(b,pork,D)}2 :-p(11,D).

not eat(a,ham,D) :-p(12,D).

1{eat(a,ham,D);eat(c,ham,D)}1 :-p(2,D).

0{eat(b,pork,D);eat(c,pork,D)}1 :-p(3,D).

#show answer/1.

## 谁要参加活动

### 问题

某医院刘佳、郑毅、郭斌、丁晓、吴芳、施文6位医生拟报名参加“一心向党，健康为民”进社区义诊活动，已知下列情况为真：

（1）要么刘佳参加，要么郑毅参加；

（2）只有吴芳参加，刘佳才参加；

（3）如果郭斌和吴芳都参加，那么施文也会参加；

（4）或者丁晓不参加，或者郭斌参加；

（5）施文、丁晓至少有1人参加。

现施文确定无法参加，那么6位医生中最后参加义诊活动的是：

A.刘佳、郭斌、丁晓

B.郑毅、郭斌、丁晓

C.郑毅、丁晓、吴芳

D.刘佳、丁晓、吴芳

### 分解法解答

person(l;z;g;d;w;s).

-join(X) :- not join(X), person(X).

answer.

2{options; rule}2 :- answer.

2{pre; ops}2 :- options.

0{join(s)}0 :- pre.

1{option(a;b;c;d)}1 :- ops.

3{join(l;g;d)}3 :- option(a).

3{join(z;g;d)}3 :- option(b).

3{join(z;d;w)}3 :- option(c).

3{join(l;d;w)}3 :- option(d).

5{p(1;2;3;4;5)}5 :- rule.

1{join(l;z)}1 :- p(1).

1{p(21;22)}1 :- p(2).

2{join(w;l)}2 :- p(21).

not join(l) :- p(22).

1{p(31;32)}1 :- p(3).

3{join(g;w;s)}3 :- p(31).

0{join(g;w)}1 :- p(32).

1{not join(d); join(g)}1 :- p(4).

1{join(s;d)}1 :- p(5).

% #show join/1.

% #show -join/1.

#show option/1.

## 身份匹配

### 问题

甲、乙、丙、丁4人，一人是教师，一人是医生，一人是作家，一人是律师。现已知:

①甲的年龄比教师大；

②乙和律师的籍贯不同；

③丙与作家的籍贯相同；

④作家的年龄比乙小；

⑤甲与律师来自相同的城市；

⑥教师的籍贯与乙相同。

x1,x2,x3,x4 -年龄

y1,y2,y3,y4 -籍贯

根据以上的信息，以下说法不正确的是:

A.作家的年龄比教师大

B.医生与律师的籍贯相同

C.医生的年龄比作家大

D.律师与教师的籍贯不同

### 分解法解答

person(a;b;c;d).

identity(teacher;doctor;writer;lawyer).

:- identity\_match(a,teacher).

:- identity\_match(b,lawyer).

:- identity\_match(c,writer).

:- identity\_match(b,writer).

:- identity\_match(a,lawyer).

:- identity\_match(b,teacher).

same\_hometown(P1, P2) :- person(P1), person(P2), -different\_hometown(P1, P2), P1 != P2.

different\_hometown(P1, P2) :- person(P1), person(P2), -same\_hometown(P1, P2), P1 != P2.

different\_hometown(P1, P2) :- person(P1), person(P2), different\_hometown(P2, P1), P1 != P2.

-same\_hometown(P1, P2) :- person(P1), person(P2), different\_hometown(P1, P2), P1 != P2.

-different\_hometown(P1, P2) :- person(P1), person(P2), same\_hometown(P1, P2), P1 != P2.

same\_hometown(P1, P2) :- person(P1), person(P2), person(P3), same\_hometown(P1, P3), same\_hometown(P2, P3), P1 != P2.

same\_hometown(P1, P2) :- person(P1), person(P2), same\_hometown(P2, P1), P1 != P2.

unknown\_hometown(P1, P2) :- person(P1), person(P2), not same\_hometown(P1, P2), not different\_hometown(P1, P2), P1 != P2.

elder(P1, P2) :- person(P1), person(P2), -smaller(P1, P2), P1 != P2.

smaller(P1, P2) :- person(P1), person(P2), -elder(P1, P2), P1 != P2.

-elder(P1, P2) :- person(P1), person(P2), smaller(P1, P2), P1 != P2.

-smaller(P1, P2) :- person(P1), person(P2), elder(P1, P2), P1 != P2.

elder(P1, P2) :- person(P1), person(P2), smaller(P2, P1), P1 != P2.

smaller(P1, P2) :- person(P1), person(P2), elder(P2, P1), P1 != P2.

elder(P1, P2) :- person(P1), person(P2), person(P3), elder(P1, P3), elder(P2, P3), P1 != P2.

smaller(P1, P2) :- person(P1), person(P2), person(P3), smaller(P1, P3), smaller(P2, P3), P1 != P2.

unknown\_age(P1, P2) :- person(P1), person(P2), not elder(P1, P2), not smaller(P1, P2), P1 != P2.

1{identity\_match(P, I) : identity(I)}1 :- person(P).

:- identity\_match(P1, I), identity\_match(P2, I), P1 != P2.

answer.

2{options; rule}2 :- answer.

% failed to code in composition; it needs further edition

% 0{option(1);option(2);option(3);option(4)}4 :- options.

answer(a) :- elder(P1, P2), identity\_match(P1, writer), identity\_match(P2, teacher).

answer(b) :- same\_hometown(P1, P2), identity\_match(P1, doctor), identity\_match(P2, lawyer).

answer(c) :- elder(P1, P2), identity\_match(P1, doctor), identity\_match(P2, writer).

answer(d) :- different\_hometown(P1, P2), identity\_match(P1, lawyer), identity\_match(P2, teacher).

5{situation(1..5)}5 :- rule.

1{elder(a, P) : identity\_match(P, teacher)}1 :- situation(1).

1{different\_hometown(b, P) : identity\_match(P, lawyer)}1 :- situation(2).

1{same\_hometown(c, P) : identity\_match(P, writer)}1 :- situation(3).

1{smaller(P, b) : identity\_match(P, writer)}1 :- situation(4).

1{same\_hometown(b, P) : identity\_match(P, teacher)}1 :- situation(5).

#show option/1.

### 约束满足法解答

% rule

elder(a, P) :- person(P), identity\_match(P, teacher).

different\_hometown(b, P) :- person(P), identity\_match(P, lawyer).

same\_hometown(c, P) :- person(P), identity\_match(P, writer).

smaller(P, b) :- person(P), identity\_match(P, writer).

same\_hometown(b, P) :- person(P), identity\_match(P, teacher).

% options

answer(a) :- elder(P1, P2), identity\_match(P1, writer), identity\_match(P2, teacher).

answer(b) :- same\_hometown(P1, P2), identity\_match(P1, doctor), identity\_match(P2, lawyer).

answer(c) :- elder(P1, P2), identity\_match(P1, doctor), identity\_match(P2, writer).

answer(d) :- different\_hometown(P1, P2), identity\_match(P1, lawyer), identity\_match(P2, teacher).

### 难点

- 如何对 options 进行正确分解？更倾向于asp的语法问题。

- 对规则的分解是否合理？

- 使用约束满足法，基本是所见即所得，会快捷地运行出正确结果；分解法却要求考虑更加周到

## 谁射中了鹿

### 问题

古代一位国王和他的张、王、李、赵、钱五位将军，一同出外打猎，各人的箭上都刻有自己的姓氏。打猎中，一只鹿中箭倒下，但不知是何人所射。

张说：“或者是我射中的，或者是李将军射中的。”

王说：“不是钱将军射中的。”

李说：“如果不是赵将军射中的，那么一定是王将军射中的。”

赵说：“既不是我射中的，也不是王将军射中的。”

钱说：“既不是李将军射中的，也不是张将军射中的。”

国王让人把射中鹿的箭拿来，看了看，说：“你们五位将军的猜测，只有两个人的话是真的。”

请根据国王的话，判定以下哪项是真的？

（A） 张将军射中此鹿。

（B） 王将军射中此鹿。

（C） 李将军射中此鹿。

（D） 赵将军射中此鹿。

（E） 钱将军射中此鹿。

### 分解法解答

person(z).

person(w).

person(l).

person(zhao).

person(q).

answer.

2{options; rule}2 :- answer.

1{shoot(X):person(X)}1 :- options.

-p(X) :- not p(X), person(X).

2{p(z;w;l;zhao;q)}2 :- rule.

1{shoot(z); shoot(l)}1 :- p(z).

not shoot(q) :- p(w).

1{shoot(zhao); shoot(w)}1 :- p(l).

0{shoot(zhao); shoot(w)}0 :- p(zhao).

0{shoot(l); shoot(z)}0 :- p(q).

0{shoot(z); shoot(l)}0 :- -p(z).

shoot(q) :- -p(w).

0{shoot(zhao); shoot(w)}0 :- -p(l).

1{shoot(zhao); shoot(w)}1 :- -p(zhao).

1{shoot(l); shoot(z)}1 :- -p(q).

#show p/1.

#show shoot/1.

### 约束满足法解答

p(z) :- shoot(z), not shoot(l).

p(z) :- not shoot(z), shoot(l).

p(w) :- not shoot(q).

p(l) :- shoot(zhao).

p(l) :- not shoot(zhao), shoot(w).

p(zhao) :- not shoot(zhao), not shoot(w).

p(q) :- not shoot(l), not shoot(z).

### 难点

- 题目中"只有两个人的话是真的"暗含的是两个命题为真，其他命题为假。若只是分解为2/5成立，则只能保证其中2个命题为真，无法保证其他命题为假，会出现如下后果。

### “命题错误”定义缺失时

Answer: 1

p(z) p(w) shoot(z)

Answer: 2

p(z) p(zhao) shoot(z)

Answer: 3

p(z) p(w) shoot(l)

Answer: 4

p(z) p(zhao) shoot(l)

Answer: 5

p(w) p(q) shoot(w)

Answer: 6

p(w) p(l) shoot(w)

Answer: 7

p(w) p(q) shoot(zhao)

Answer: 8

p(w) p(l) shoot(zhao)

Answer: 9

p(l) p(q) shoot(w)

Answer: 10

p(l) p(q) shoot(zhao)

Answer: 11

p(zhao) p(q) shoot(q)

Answer: 12

p(w) p(zhao) shoot(z)

Answer: 13

p(w) p(zhao) shoot(l)