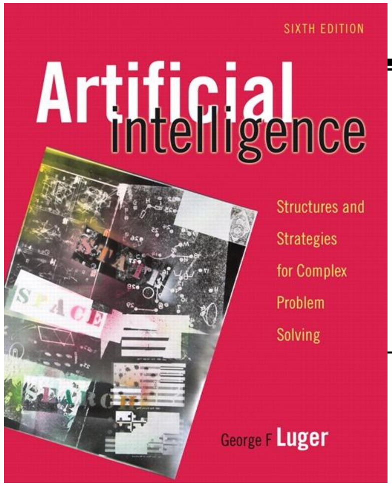
14

Automated Reasoning



George F Luger

ARTIFICIAL INTELLIGENCE *6th edition*

Structures and Strategies for Complex Problem Solving

14.2 Resolution（归结）Theorem Proving

14.2.1 Introduction

- Resolution is a technique for proving theorems in the propositional or predicate calculus that has been a part of AI problem-solving research.

- Resolution is a sound inference rule that, when used to produce a refutation（反证）, is also complete.

Resolution refutation proofs involve the following steps:

1. Put the premises or axioms into *clause form* (13.2.2).
2. Add the negation of what is to be proved, in clause form, to the set of axioms.
3. *Resolve* these clauses together, producing new clauses that logically follow from them (13.2.3).
4. Produce a contradiction by generating the empty clause.
5. The substitutions used to produce the empty clause are those under which the opposite of the negated goal is true (13.2.4).

•**Resolution refutation proofs require that the axioms and the negation of the goal be placed in a normal form called *clause form* .**

- **–A literal is an atomic expression or the negation of an atomic expression.**

- **–Clause form represents the logical database as a set of disjunctions of literals.**

•**The most common form of resolution called *binary resolution* .**

A simple example

- **Given :**
  - **"all dogs are animals"**
  - **"all animals will die"**
  - **"Fido is a dog"**
- **Prove that "Fido will die"**
- **Representation :**
  - **All dogs are animals:**

    $$\forall(X)(dog(X) \rightarrow animals(X))$$
  - **All animals will die:**

    $$\forall(Y)(animals(Y) \rightarrow die(Y))$$
  - **Fido is a dog:**

    $$dog(fido)$$

Traditional Theorem Proving

- **All dogs are animals:**

  $$\forall(X)(dog(X) \rightarrow animals(X))$$

- **Fido is a dog: dog(fido)**
- **Modus ponens and { fido / X } gives :**

  animals (fido)

- **All animals will die:**

  $$\forall(Y)(animals(Y) \rightarrow die(Y))$$

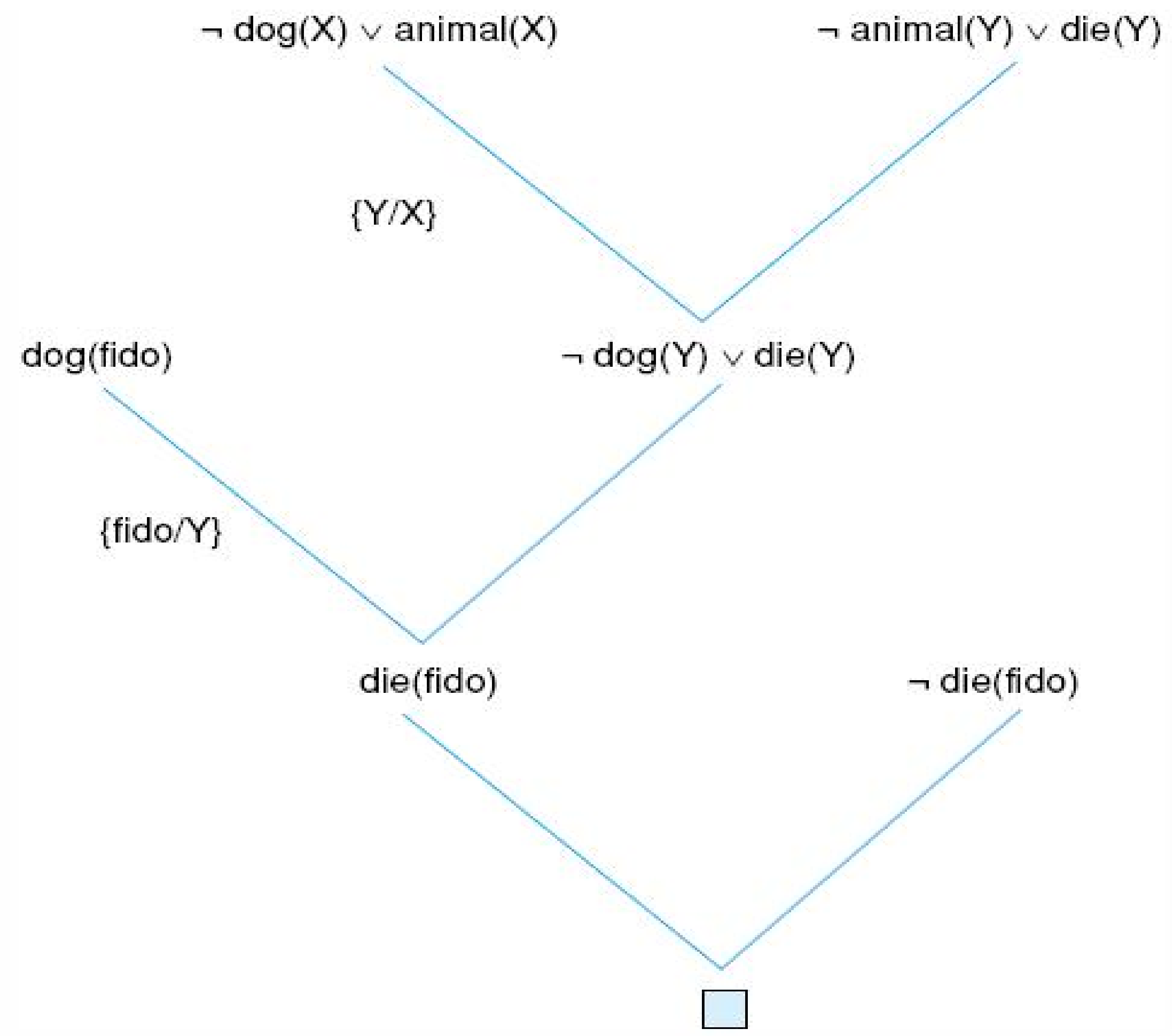
- **Modus ponens and { fido / Y } gives :**

  die(fido).

Resolution Theorem Proving

- **Convert the predicates to clause form:**
  - **predicate form**
    1. $\forall(X)(dog(X) \rightarrow animal(X))$
    2. $dog(fido)$
    3. $\forall(Y)(animals(Y) \rightarrow die(Y))$
    4. $\neg\ die(fido)$
  - **clause form**
    1. $\neg dog(X) \lor animal(X)$
    2. $dog(fido)$
    3. $\neg animal(Y) \lor die(Y)$
    4. $\neg\ die(fido)$

Fig 14.3 Resolution proof for the “dead dog” problem.



14.2.2 Producing the Clause Form for Resolution Refutation

- **The resolution proof procedures all statements in the database describing a situation to be converted a standard form called *clause form*.**
- **The form the database takes is referred to as a *conjunction of disjuncts*.**
- **It is a conjunction because all the clauses that make up the database are assumed to be true at the same time.**

- **An algorithm for reducing any predicate expressions to clause form.**
- **We demonstrate the process of the algorithm through an example:**
- **Suppose X, Y, Z are variables and I is a constant**

$$(\forall X)\ ([\ a(X) \wedge b(X)\ ] \rightarrow$$
$$[\ c(X, I) \wedge (\exists Y)\ ((\exists Z)[\ c(Y, Z)\ ] \rightarrow d(X, Y)\ )\ ]\ ) \vee$$
$$(\forall X)\ (\ e(X)\ )$$

1. **Eliminate the →**

$$(\forall X)\ (\neg [\ a(X) \wedge b(X)\ ] \vee$$
$$[\ c(X, I) \wedge (\exists Y)\ (\neg (\exists Z)[\ c(Y, Z)\ ] \vee d(X, Y)\ )\ ]\ ) \vee$$
$$(\forall X)\ (\ e(X)\ )$$

1. **Eliminate the →**

$(\forall X)\,(\neg\,[\,a(X) \wedge b(X)\,] \vee$

$\quad [\,c(X, I) \wedge (\exists Y)\,(\neg\,(\exists Z)[\,c(Y, Z)\,] \vee d(X, Y)\,)\,]\,) \vee$

$(\forall X)\,(\,e(X)\,)$

2. **Reduce the scope of ¬ to atoms**

$(\forall X)\,(\,[\,\neg\,a(X) \vee \neg\,b(X)\,] \vee$

$\quad [\,c(X, I) \wedge (\exists Y)\,(\,(\forall Z)[\,\neg\,c(Y, Z)\,] \vee d(X, Y)\,)\,]\,) \vee$

$(\forall X)\,(\,e(X)\,)$

2. Reduce the scope of ¬ to atoms

$(\forall X)\ (\ [\neg a(X) \lor \neg b(X)\ ] \lor$

$[\ c(X, I) \land (\exists Y)\ (\ (\forall Z)[\neg c(Y, Z)\ ] \lor d(X, Y)\ )\ ]\ ) \lor$

$(\forall X)\ (\ e(X)\ )$

3. Variable standardization : Renaming variables so that each quantifier has unique variable name.

$(\forall X)\ (\ [\neg a(X) \lor \neg b(X)\ ] \lor$

$[\ c(X, I) \land (\exists Y)\ (\ (\forall Z)[\neg c(Y, Z)\ ] \lor d(X, Y)\ )\ ]\ ) \lor$

$(\forall W)\ (\ e(W)\ )$

3. **Variable standardization : Renaming variables so that each quantifier has unique variable name.**

$(\forall X) ([\neg a(X) \vee \neg b(X)] \vee$

$[c(X, I) \wedge (\exists Y) ((\forall Z)[\neg c(Y, Z)] \vee d(X, Y))]) \vee$

$(\forall W) (e(W))$

4. **Move all quantifiers to the left without changing their order. (** ***prenex normal form)***

$(\forall X) (\exists Y) (\forall Z) (\forall W) ([\neg a(X) \vee \neg b(X)] \vee$

$[c(X, I) \wedge ([\neg c(Y, Z)] \vee d(X, Y))] \vee$

$e(W))$

4. **Move all quantifiers to the left without changing their order.**

$(\forall X)\ (\exists Y)\ (\forall Z)\ (\forall W)\ (\ [\neg a(X) \lor \neg b(X)\ ] \lor$

$[\ c(X, I) \land (\ [\neg c(Y, Z)\ ] \lor d(X, Y)\ )\ ] \lor$

$e(W)\ )$

5. **Eliminate all existential quantifiers by skolemization.**

$(\forall X)\ (\forall Z)\ (\forall W)\ (\ [\neg a(X) \lor \neg b(X)\ ] \lor$

$[\ c(X, I) \land (\ [\neg c(\ f(X), Z)\ ] \lor d(X, f(X))\ )\ ] \lor$

$e(W)\ )$

5. **Eliminate all existential quantifiers by skolemization.**

$$(\forall X)(\forall Z)(\forall W)\,(\,[\,\neg\, a(X) \vee \neg\, b(X)\,] \vee$$

$$[\, c(X, I) \wedge (\,[\,\neg\, c(\, f(X), Z)\,] \vee d(X, f(X))\,)\,] \vee$$

$$e(W)\,)$$

6. **Drop all universal quantification.**

$$[\,\neg\, a(X) \vee \neg\, b(X)\,] \vee$$

$$[\, c(X, I) \wedge (\,[\,\neg\, c(\, f(X), Z)\,] \vee d(X, f(X))\,)\,] \vee$$

$$e(W)$$

6. **Drop all universal quantification.**

$[\neg a(X) \vee \neg b(X)] \vee$

$[c(X, I) \wedge ([\neg c(f(X), Z)] \vee d(X, f(X)))] \vee$

$e(W)$

7. **Convert the expression to the conjunct of disjuncts form.**

$[[\neg a(X) \vee \neg b(X) \vee c(X, I) \vee e(W)] \wedge$

$[\neg a(X) \vee \neg b(X) \vee \neg c(f(X), Z) \vee d(X, f(X)) \vee e(W)]]$

7. Convert the expression to the conjunct of disjuncts form.

$[\, [\neg a(X) \vee \neg b(X) \vee c(X, I) \vee e(W)] \wedge$

$[\neg a(X) \vee \neg b(X) \vee \neg c(f(X), Z) \vee d(X, f(X)) \vee e(W)]\,]$

8. Call each conjunct a separate clause. (drop $\wedge$)

① $[\neg a(X) \vee \neg b(X) \vee c(X, I) \vee e(W)]$

② $[\neg a(X) \vee \neg b(X) \vee \neg c(f(X), Z) \vee d(X, f(X)) \vee e(W)]$

8. Call each conjunct a separate clause. (drop $\wedge$)

① $[\neg a(X) \vee \neg b(X) \vee c(X, I) \vee e(W)]$

② $[\neg a(X) \vee \neg b(X) \vee \neg c(f(X), Z) \vee d(X, f(X)) \vee e(W)]$

9. Standardize the variables apart again : give the variables in each clauses different names.

① $[\neg a(X) \vee \neg b(X) \vee c(X, I) \vee e(W)]$

② $[\neg a(U) \vee \neg b(U) \vee \neg c(f(U), Z) \vee d(U, f(U)) \vee e(V)]$

This nine-step process is used to change any set of predicate calculus expressions to clause form. The completeness property of resolution is not lost.

例：将谓词公式

$G = (\forall x)((\forall y)P(x,y) \rightarrow \sim (\forall y)(Q(x,y) \rightarrow R(x,y)))$

化为子句集。

解:(1)取消“→”连接词。

$(\forall x)(\sim(\forall y)P(x,y) \vee \sim (\forall y)(\sim Q(x,y) \vee R(x,y)))$

(2)将“～”的辖域减少到最多只作用于一个谓词。

$(\forall x)((\exists y)\sim P(x,y) \vee (\exists y)(Q(x,y) \wedge \sim R(x,y)))$

(3)变量换名。

$(\forall x)((\exists y)\sim P(x,y) \vee (\exists z)(Q(x,z) \wedge \sim R(x,z)))$

(4)消去存在量词。

$(\forall x)(\sim P(x,f(x)) \vee (Q(x,g(x)) \wedge \sim R(x,g(x))))$

(5) 全称量词左移。

$(\forall x)(\sim P(x,f(x))\vee(Q(x,g(x))\wedge\sim R(x,g(x))))$

(6) 将母式化为合取范式。

$(\forall x)((\sim P(x,f(x))\vee(Q(x,g(x)))\wedge(\sim P(x,f(x))\vee\sim R(x,g(x))))$

(7) 消去全称量词。

$((\sim P(x,f(x))\vee(Q(x,g(x)))\wedge(\sim P(x,f(x))\vee\sim R(x,g(x))))$

(8) 消去合取词，得G的子句集S。

$S=\{\sim P(x,f(x))\vee(Q(x,g(x)),\ \sim P(x,f(x))\vee\sim R(x,g(x))\}$

例：求下列公式G的子句集。

$$(\forall x)(\exists y)(\exists z)((\sim P(x,y)\vee R(x,y,z))\wedge(Q(x,z)\vee R(x,y,z)))$$

解：首先求得G的Skolem标准型如下：

$$(\forall x)((\sim P(x,f(x))\vee R(x,f(x),g(x)))$$
$$\wedge(Q(x,g(x))\vee R(x,f(x),g(x))))$$

则G的子句集为：

$$S=\{\sim P(x,f(x))\vee R(x,f(x),g(x)),$$
$$Q(x,g(x))\vee R(x,f(x),g(x))\}$$

14.2.3 The Binary Resolution Proof Procedure

- **An example from the propositional clauses**

To prove "a" from the following axioms :

①$a \leftarrow b \wedge c$

②b

③$c \leftarrow d \wedge e$

④$e \vee f$

⑤$d \wedge \neg f$

- **Reduced to clause form :**

① $a \lor \neg b \lor \neg c$

② b

③ $c \lor \neg d \lor \neg e$

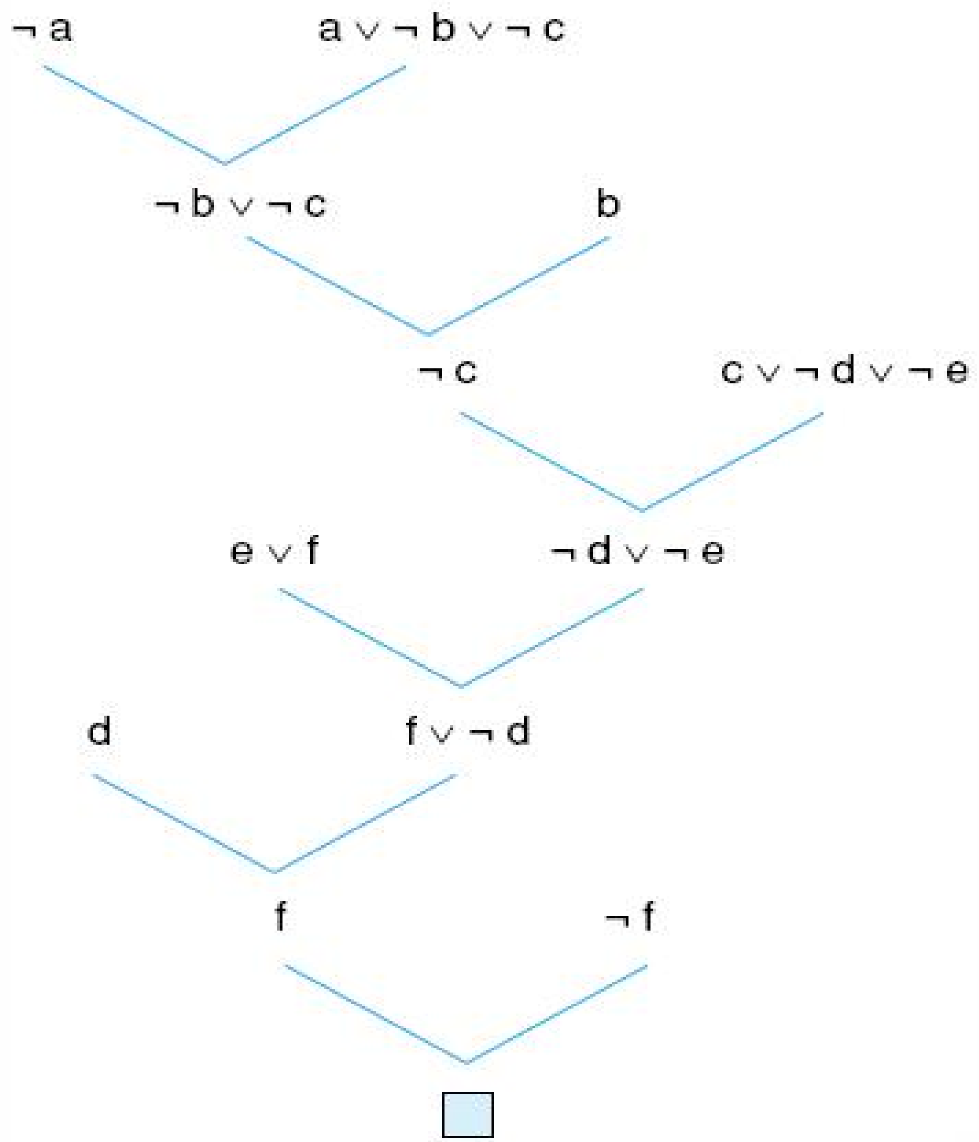
④ $e \lor f$

⑤ d

⑥ $\neg f$

- **Fig 14.4 shows one resolution proof.**

Fig 14.4 One resolution proof for an example from the propositional calculus.



归结原理

定义:若P是原子谓词，则称P与¬ P 为互补文字。

1、命题逻辑中的归结原理

定义:设C_1与C_2是子句集中的任意两个子句，如果C_1中的文字L_1与C_2中的L_2互补，那么从C_1和C_2中分别消去L_1和L_2，并将而个子句中余下的部分析取，构成一个新子句C_{12}，则称这一过程为归结，称C_{12}为C_1和C_2的归结式，称C_1和C_2为C_{12}的亲本子句。

定理:归结式C_{12}是其亲本子句C_1与C_2的逻辑结论。

定理证明：

设$C_1 = L \lor C_1'$，$C_2 = \neg L \lor C_2'$

通过归结可得到：$C_{12} = C_1' \lor C_2'$

C_1和C_2是C_{12}的亲本子句。

因为：$C_1' \lor L \Leftrightarrow \neg C_1' \rightarrow L$

$\neg L \lor C_2' \Leftrightarrow L \rightarrow C_2'$

所以：$C_1 \land C_2 = (\neg C_1' \rightarrow L) \land (L \rightarrow C_2')$

根据假言三段论得到：

$$(\neg C_1' \rightarrow L) \land (L \rightarrow C_2') \Rightarrow \neg C_1' \rightarrow C_2'$$

因为：$\neg C_1' \rightarrow C_2' \Leftrightarrow C_1' \lor C_2' = C_{12}$

所以：$C_1 \land C_2 \Rightarrow C_{12}$

可知 C_{12}是其亲本子句C_1和C_2的逻辑结论。

推论1

设C_1与C_2是子句集S中的两个子句， C_{12}是它们的归结式，若用C_{12}代替C_1和C_2得到新子句集S_1 ，则由S_1的不可满足性可推出原子句集S的不可满足性，即

S_1的不可满足性$\Rightarrow$S的不可满足性

推论2

设C_1与C_2是子句集S中的两个子句， C_{12}是它们的归结式，若把C_{12}加入S中得到新子句集S_2 ，则S与S_2在不可满足的意义上是等价的，即

S_2的不可满足性$\Leftrightarrow$ S的不可满足性

2. 谓词逻辑中的归结原理

- 设C_1与C_2两个没有相同变元的子句， L_1和L_2分别是C_1和C_2中的文字，若 σ是L_1和$\neg L_2$的最一般合一，则称

$$C_{12}=(C_1\sigma-\{L_1\sigma\})\cup(C_2\sigma-\{L_2\sigma\})$$

为C_1和C_2的二元归结式， L_1和L_2称为归结式的文字。

例 设$C_1 = P(a) \vee \neg Q(x) \vee R(x)$

$C_2 = \neg P(y) \vee Q(b)$

1) 若选$L_1 = P(a)$，$L_2 = \neg P(y)$，则$\sigma = \{a/y\}$是L_1与$\neg L_2$的最一般合一，根据定义，可得：

$$C_{12} = (C_1\sigma - \{L_1\sigma\}) \cup (C_2\sigma - \{L_2\sigma\})$$

$$= (\{P(a), \neg Q(x), R(x)\} - \{P(a)\}) \cup$$

$$(\{\neg P(a), Q(b)\} - \{\neg P(a)\})$$

$$= (\{\neg Q(x), R(x)\}) \cup (\{Q(b)\})$$

$$= \{\neg Q(x), R(x), Q(b)\}$$

$$= \neg Q(x) \vee R(x) \vee Q(b)$$

2) 若选$L_1 = \neg Q(x)$，$L_2 = Q(b)$，$\sigma = \{b/x\}$，可得：

$$C_{12} = (\{P(a), \neg Q(b), R(b)\} - \{\neg Q(b)\}) \cup$$

$$(\{\neg P(y), Q(b)\} - \{Q(b)\})$$

$$= (\{P(a), R(b)\}) \cup (\{\neg P(y)\})$$

$$= \{P(a), R(b), \neg P(y)\}$$

$$= P(a) \vee R(b) \vee \neg P(y)$$

例：设$C_1 = P(x) \vee Q(a)$， $C_2 = \neg P(b) \vee R(x)$

由于C_1与C_2有相同的变元，不符合定义的要求。为了进行归结，需修改C_2中的变元的名字，

令$C_2 = \neg P(b) \vee R(y)$。此时，对$C_1$和$C_2$有

$$L_1 = P(x), L_2 = \neg P(b)$$

L_1与$\neg L_2$的最一般合一$\sigma = \{b/x\}$，

则 $C_{12} = (\{P(a), Q(a)\} - \{P(b)\}) \cup$

$$(\{\neg P(b), R(y)\} - \{\neg P(b)\})$$

$$= \{Q(a), R(y)\}$$

$$= Q(a) \vee R(y)$$

例 设有如下两个子句

$$C_1 = P(x) \vee P(f(a)) \vee Q(x),$$

$$C_2 = \neg P(y) \vee R(b)$$

在C_1中有可合一的文字$P(x)$与$P(f(a))$，若用它们的最一般合一$\theta=\{f(a)/x\}$进行代换，得到

$$C_1\theta = P(f(a)) \vee Q(f(a))$$

此时可对$C_1\theta$和C_2进行归结，从而得到C_1与C_2的二元归结式。

对$C_1\theta$和C_2分别选$L_1=P(f(a))$, $L_2=\neg P(y)$。L_1和$\neg L_2$的最一般合一是 $=\{f(a)/y\}$，则

$$C_{12} = R(b) \vee Q(f(a))$$

定义

子句C_1和C_2归结式是下列的二元归结式之一：

1） C_1和C_2的二元归结式。

2） C_1和C_2的因子$C_2\sigma_2$的二元归结式。

3） C_1的因子$C_1\sigma_1$与 C_2的二元归结式。

4） C_1的因子$C_1\sigma_1$与 C_2的因子$C_2\sigma_2$二元归结式。

归结反演

定义：应用归结原理证明结论为真的过程称为归结反演。

设F为已知前提的公式集，Q为目标公式（结论），用归结反演证明Q为真的步骤是：

1）否定Q，得到¬ Q。

2）把¬ Q并入到公式集F中，得到{F，¬ Q}。

3）把公式集{F， ¬ Q }化为子句集S。

4）应用归结原理对子句集S中的子句进行归结，并把每次归结得到的归结式都并入S中。如此反复进行，若出现了空子句，则停止归结，此时就证明了Q为真。

例 已知

F: $(\forall x)((\exists y)(A(x,y)\wedge B(y))\rightarrow(\exists y)(C(y)\wedge D(x,y)))$

G: $\neg(\exists x)C(x)\rightarrow(\forall x)(\forall y)(A(x,y)\rightarrow\neg B(y))$

求证：G是F的逻辑结论。

证:首先把F和¬G化为子句集：

F: (1) $\neg A(x,y)\vee\neg B(y)\vee C(f(x))$

(2) $\neg A(x,y)\vee\neg B(y)\vee D(x,f(x))$

¬G: (3) $\neg C(z)$

(4) $A(a,b)$

(5) $B(b)$

下面进行归结：

(6) $\neg A(x,y) \vee \neg B(y)$　由(1)与(3)归结，$\{f(x)/z\}$

(7) $\neg B(b)$　由(4)与(6)归结，$\{a/x, b/y\}$

(8) NIL(空子句)　由(5)与(7)归结

所以G是F的逻辑结论。

上述归结过程可用图1所示的归结树表示

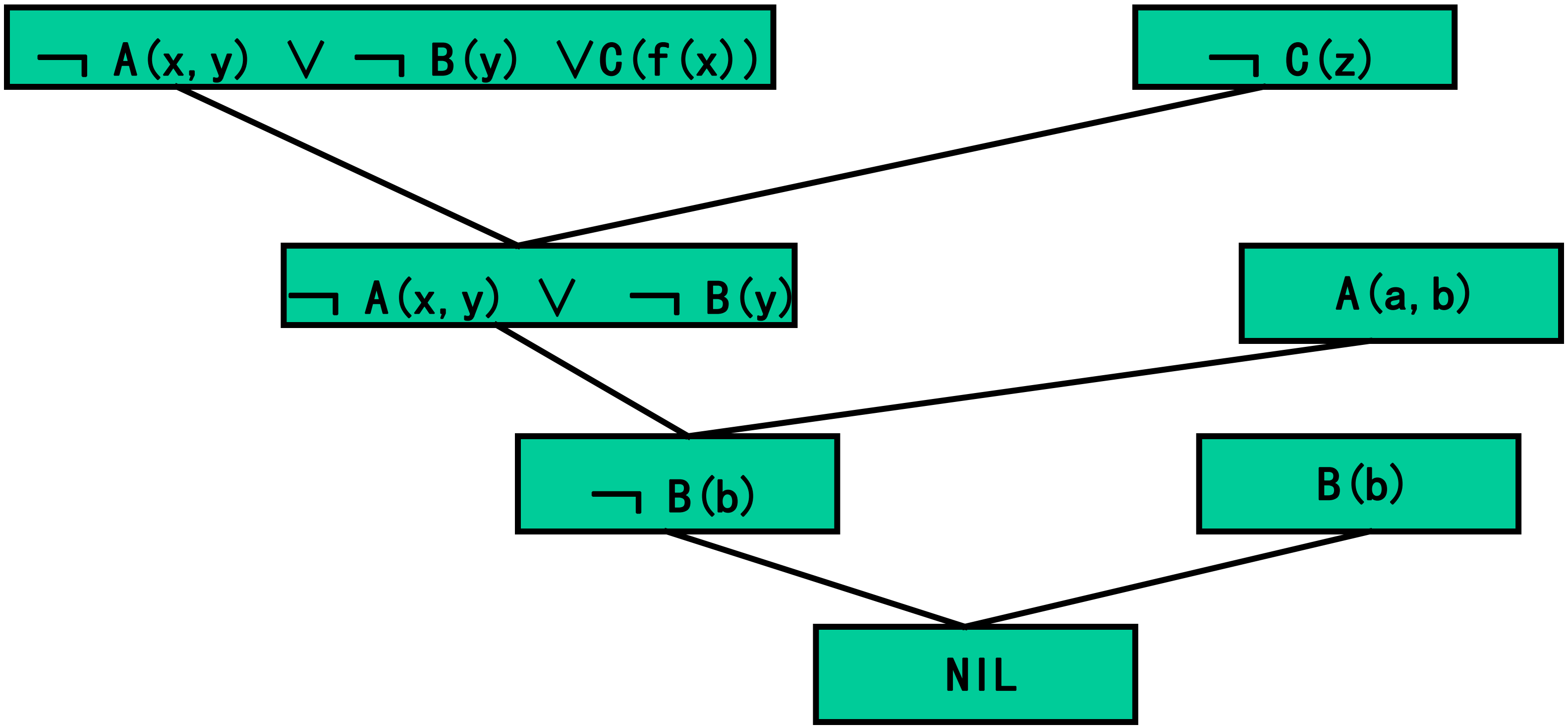


图1　归结树

例 如下公式：

F_1: $(\forall x)(N(x) \rightarrow (GZ(x) \wedge I(x))$

自然数都是大于零的整数。

F_2: $(\forall x)(I(x) \rightarrow E(x) \vee O(x))$

所有整数不是偶数就是奇数。

F_3: $(\forall x)(E(x) \rightarrow I(s(x))$

偶数除以2是整数。

求证：所有自然数不是奇数就是其一半为整数的数。

证：首先把求证的问题用谓词公式表示出来：

$G:(\forall x)(N(x)\rightarrow(O(x)\vee I(s(x))))$

将F_1，F_2，F_3及$\neg G$化成子句集：

(1) $\neg N(x) \vee GZ(x)$

(2) $\neg N(u) \vee I(u)$

(3) $\neg I(y) \vee E(y) \vee O(y)$

(4) $\neg E(z) \vee I(s(z))$

(5) $N(t)$

(6) $\neg O(t)$

(7) $\neg I(s(t))$

对上述子句进行归结：

(8) $\neg I(y) \vee E(y)$　　(3) 与(6)归结，{y/t}

(9) $\neg E(z)$　　(4) 与(7)归结，{z/t}

(10) $\neg I(y)$　　(8) 与(9)归结，{y/z}

(11) $\neg N(u)$　　(2) 与(10)归结，{u/y}

(12) NIL　　(5) 与(11)归结，{u/t}

所以所有自然数不是基数就是其一半为整数的数。

上述归结过程可用图所示的归结树表示

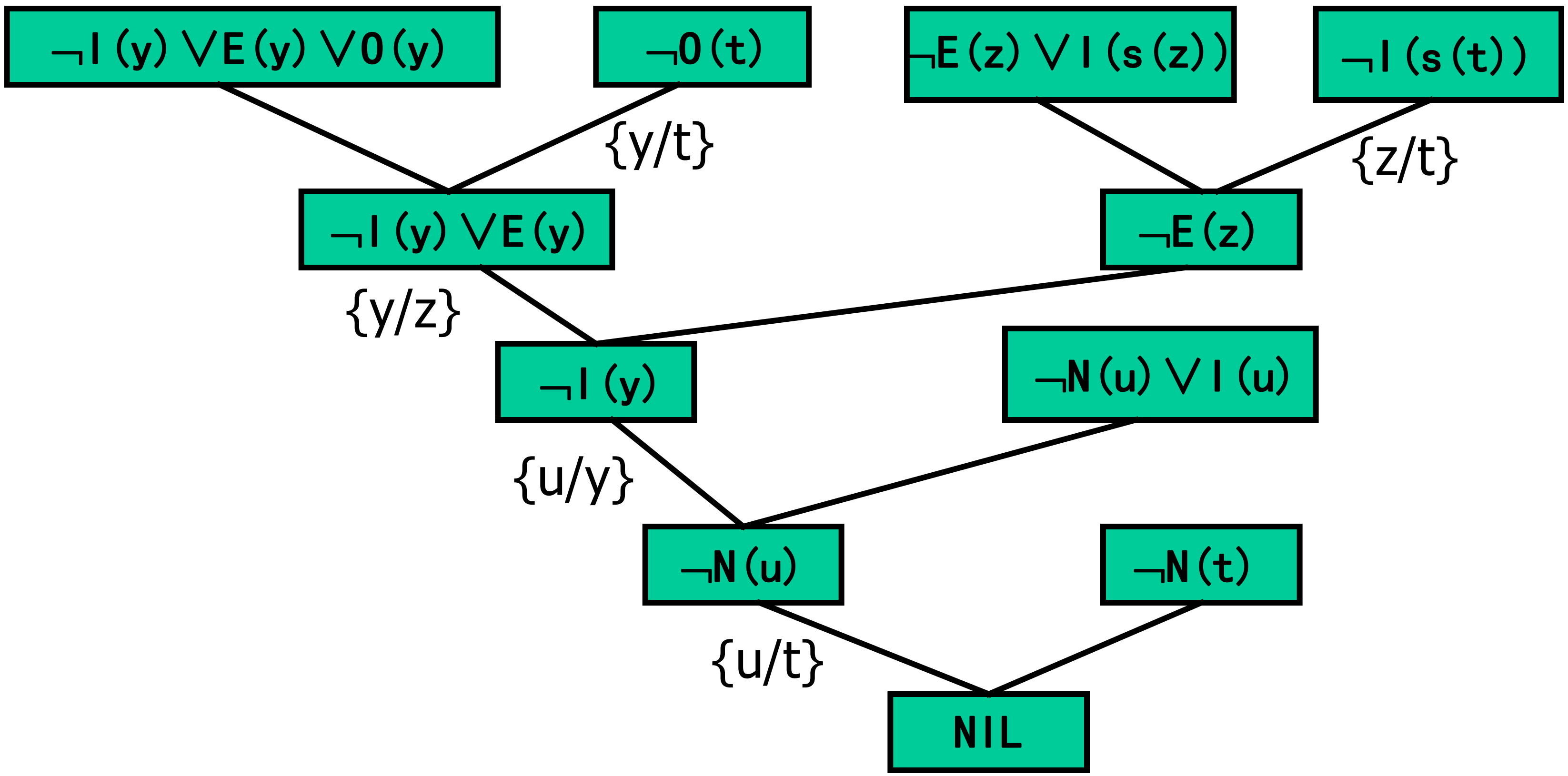


图2 归结树

归结举例

Sam、Clyde和Oscar是大象。关于它们，我们知道以下事实：

1）Sam是粉色的；

2）Clyde是灰色的且喜欢Oscar；

3）Oscar是粉色或者是灰色（但不是两种颜色）且喜欢Sam。

用归结法证明：一头灰色大象喜欢一头粉色大象。

即证明：$\exists x \exists y[Gray(x) \wedge Pink(y) \wedge Likes(x,y)]$

谓词： Gray(x)，Pink(x)，Like（x,y）

事实： 1）Pinky(Sam)

2）$Gray(Clyde) \wedge Like(Clyde,Oscar)$

3）$(Gray(Oscar) \vee Pink(Oscar)) \wedge Likes(Oscar,Sam)$

子句集：1）Pink(Sam)

2）Gray(Clyde)

3）Like(Clyde,Oscar)

4）Gray(Oscar)∨Pink(Oscar)

5）Likes(Oscar,Sam)

6） ¬ Gray(x) ∨ ¬ Pink(y) ∨ ¬ Likes(x,y)

归结： 7） ¬ Gray(Oscar) ∨ ¬ Pink （Sam ） 5，6得

8） ¬ Gray(Clyde) ∨ ¬ Pink(Oscar) 3，6得

9） Pink(Oscar) ∨ ¬ Pink （Sam ） 4，7得

10） ¬ Pink(Oscar) 8，2得

11）¬ Pink （Sam ） 9，10得

12）NIL 1，11得

•The algorithm for resolution on the predicate calculus is very much like that on the propositional calculus except that :

- **–A literal and its negation in parent clauses produce a resolvent only if they unify under the most general unifier σ, σ is then applied to the resolvent before adding it to the clause set.**

- **–The unification substitutions used to find the contradiction offer variable bindings under which the original query is true.**

- **An example of a resolution refutation for the predicate claulus.**

- **Consider the following story of the “happy student” :**

  1. **Anyone passing his history exams and winning the lottery(中奖) is happy.**

  2. **Anyone who studies or is lucky can pass all his exams.**

  3. **John did not study but he is lucky.**

  4. **Anyone who is lucky wins the lottery.**

  5. **Is John happy?**

Anyone passing his history exams and winning the lottery is happy.

$\forall$ **X (pass (X,history) $\wedge$ win (X,lottery) $\rightarrow$ happy (X))**

Anyone who studies or is lucky can pass all his exams.

$\forall$ **X** $\forall$ **Y (study (X) $\vee$ lucky (X) $\rightarrow$ pass (X,Y))**

John did not study but he is lucky.

$\neg$ **study (john) $\wedge$ lucky (john)**

Anyone who is lucky wins the lottery.

$\forall$ **X (lucky (X) $\rightarrow$ win (X,lottery))**

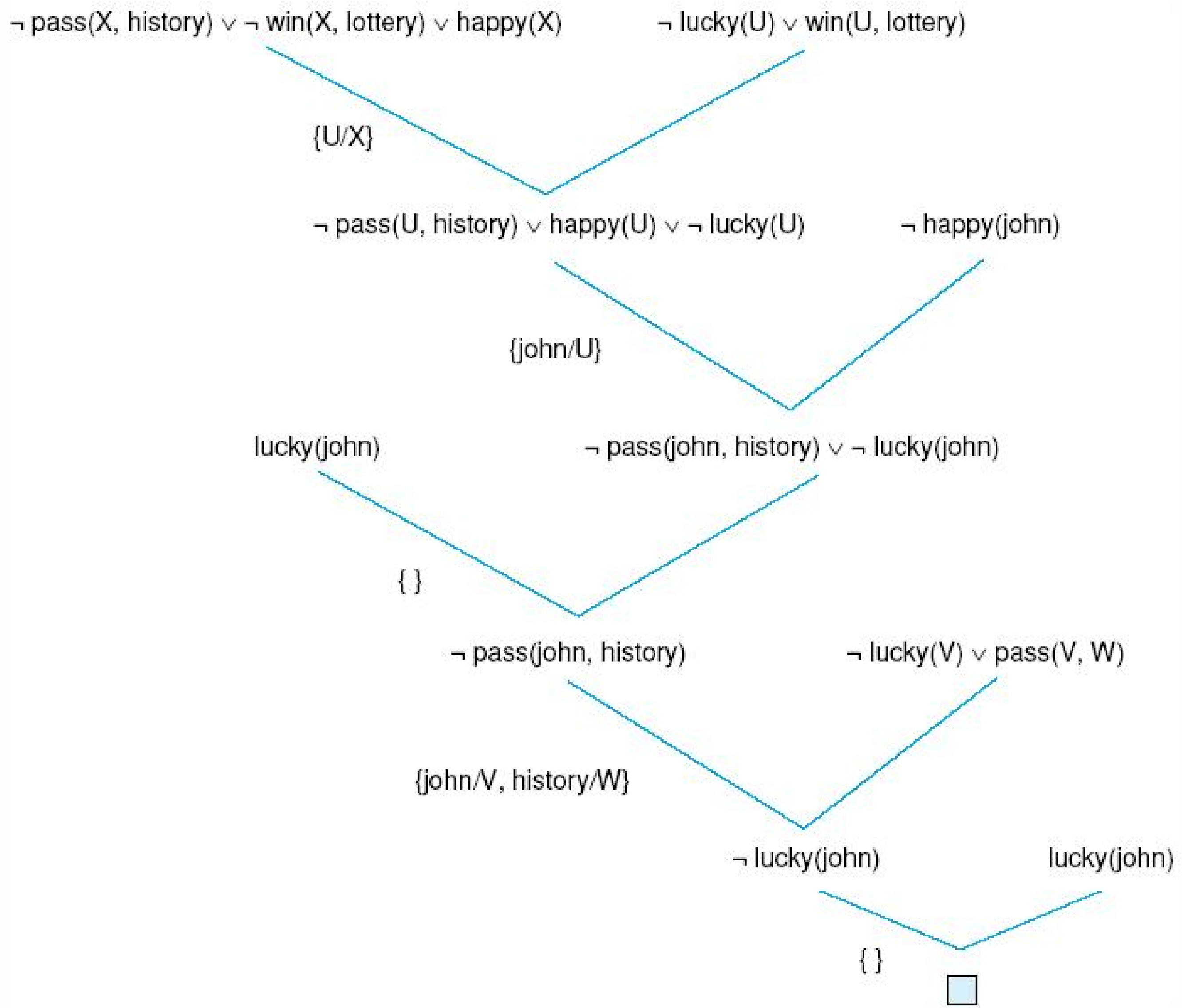
These four predicate statements are now changed to clause form (Section 12.2.2):

1. $\neg$ **pass (X, history) $\vee$ $\neg$ win (X, lottery) $\vee$ happy (X)**
2. $\neg$ **study (Y) $\vee$ pass (Y, Z)**
3. $\neg$ **lucky (W) $\vee$ pass (W, V)**
4. $\neg$ **study (john)**
5. **lucky (john)**
6. $\neg$ **lucky (U) $\vee$ win (U, lottery)**

Into these clauses is entered, in clause form, the negation of the conclusion:

7. $\neg$ **happy (john)**

Fig 14.5 One refutation for the “happy student” problem.



All people who are not poor and are smart are happy. Those people who read are not stupid. John can read and is wealthy. Happy people have exciting lives. Can anyone be found with an exciting life?

We assume $\forall$**X (smart (X)** $\equiv \neg$ **stupid (X)) and** $\forall$**Y (wealthy (Y)** $\equiv \neg$ **poor (Y)), and get**:

$\forall$**X (**$\neg$ **poor (X)** $\wedge$ **smart (X)** $\rightarrow$ **happy (X))**

$\forall$**Y (read (Y)** $\rightarrow$ **smart (Y))**

read (john) $\wedge \neg$ **poor (john)**

$\forall$**Z (happy (Z)** $\rightarrow$ **exciting (Z))**

The negation of the conclusion is:

$\neg \exists$ **W (exciting (W))**

These predicate calculus expressions for the "exciting life" problem are transformed into the following clauses:

poor (X) $\vee \neg$ **smart (X)** $\vee$ **happy (X)**

$\neg$ **read (Y)** $\vee$ **smart (Y)**

read (john)

$\neg$ **poor (john)**

$\neg$ **happy (Z)** $\vee$ **exciting (Z)**

$\neg$ **exciting (W)**

Fig 14.6 Resolution proof for the “exciting life” problem.

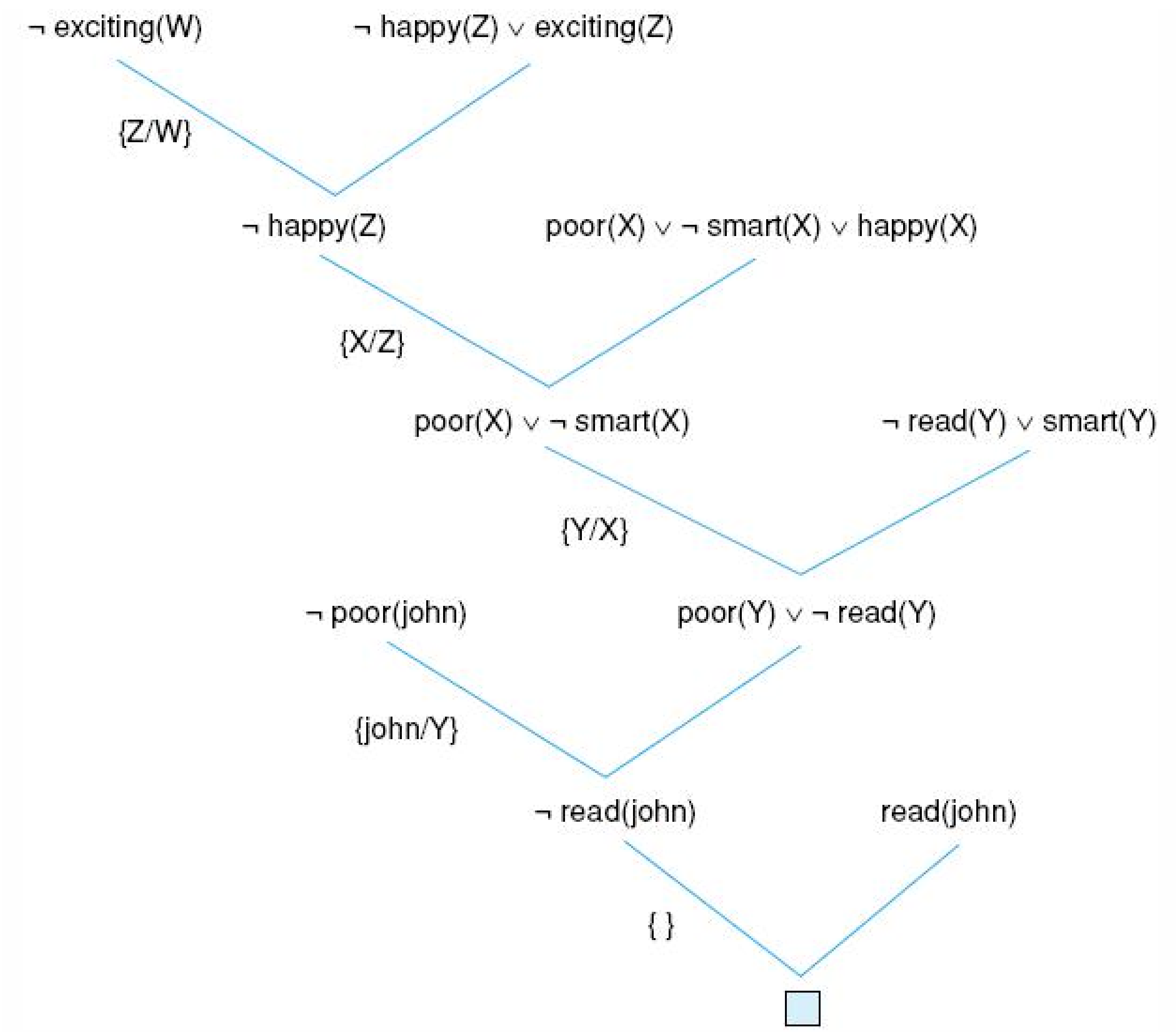
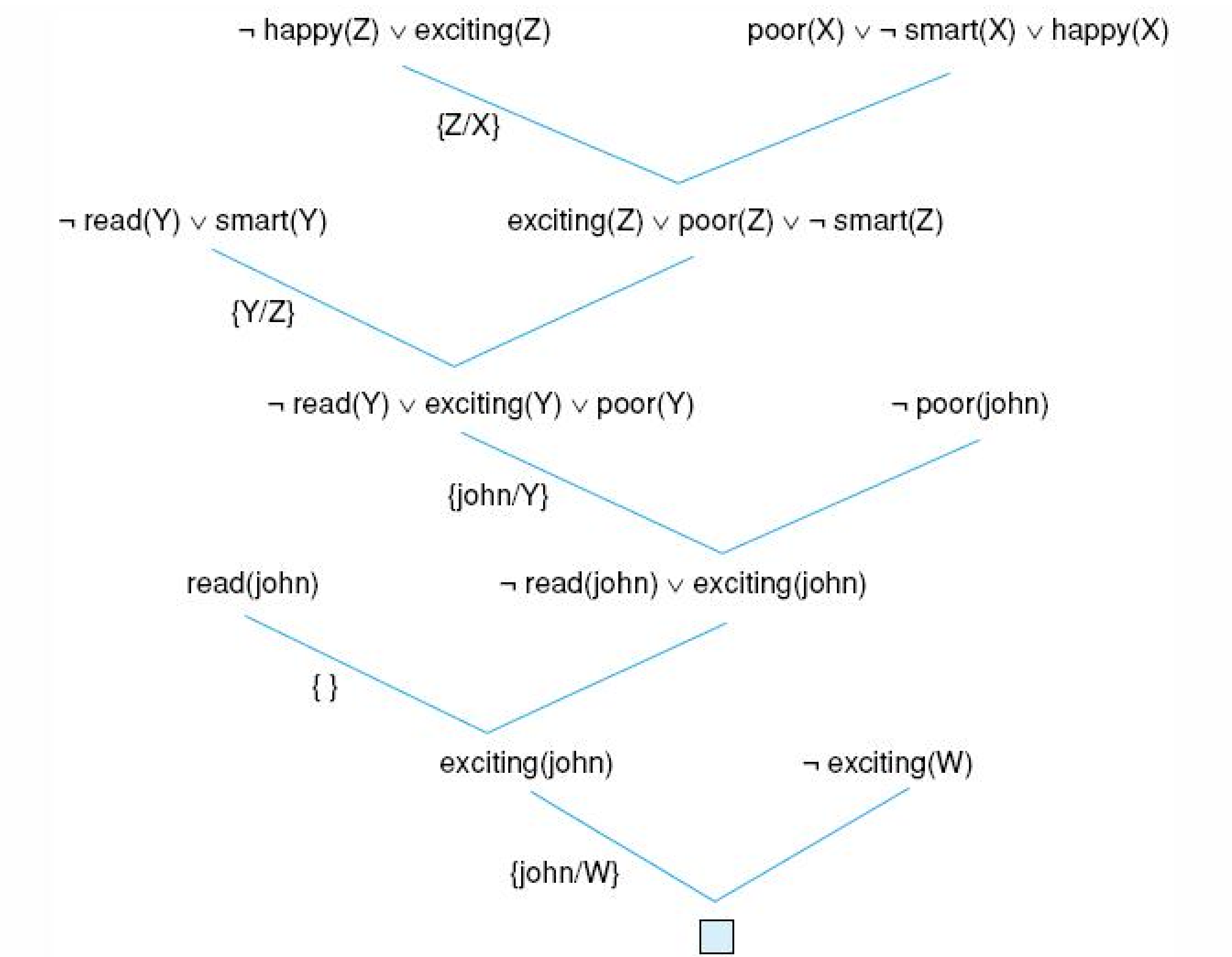


Fig 14.7 another resolution refutation for the example of Fig 14.6.



14.2.4 Strategies and Simplification Techniques for Resolution

- **Possible combinations of clauses may cause exponential complexity.**
- **Search heuristics are very important in resolution proof procedures, as they are in all weak method problem solving.**

14.2.5 Answer Extraction from Resolution Refutations

- Bookkeeping（记账）method :
- The original conclusion is the bookkeeper（会计员）of all unifications that are made as part of the refutation.
- The original conclusion is transformed by the unifications that are made on solution path.
- May need extra pointers
- Control mechanism such as backtracking may be necessary.

Fig 14.10 Unification substitutions of Fig 14.6 applied to the original query.

$$
\begin{tikzcd}
\text{exciting(W)} \arrow[d, swap, "\{Z/W\}"] \\
\text{exciting(Z)} \arrow[d, swap, "\{X/Z\}"] \\
\text{exciting(X)} \arrow[d, swap, "\{Y/X\}"] \\
\text{exciting(Y)} \arrow[d, swap, "\{john/Y\}"] \\
\text{exciting(john)} \arrow[d, swap, "\{\ \}"] \\
\text{exciting(john)}
\end{tikzcd}
$$

- **Second example :**
  - **Fido the dog goes wherever John, his master, goes.**
  - **John is at the library. Where is Fido?**
- **The predicates :**

  (at (john, X) → at (fido, X))

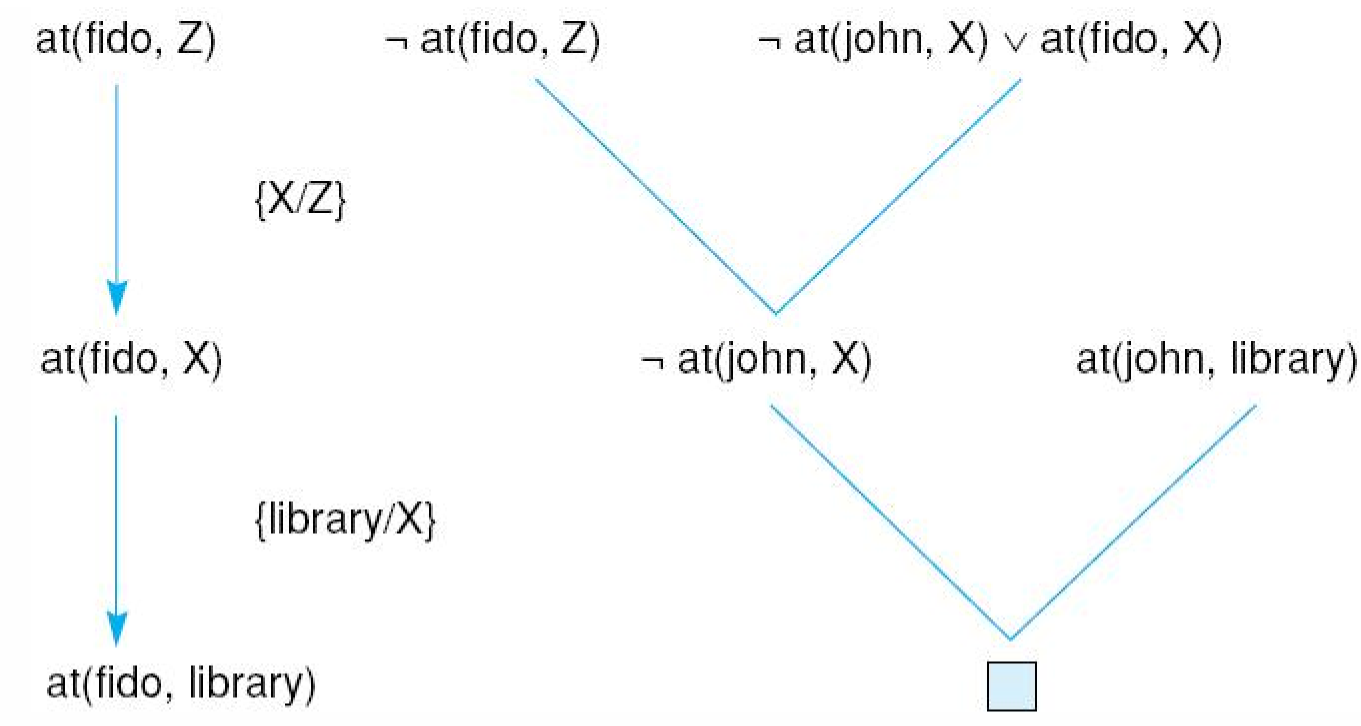
  at (john, library)
- **The clause:**

  ¬ at (john, Y) ∨ at (fido, Y)

  At (john, library)
- **The conclusion negated:**

  ¬ at (fido, Z), (Fido is nowhere !)

Fig 14.11 Answer extraction process on the “finding fido” problem.



- **The final example:**

  Everyone has a parent. The parent of a parent is grandparent. Given the person John, prove that John has a grandparent.

- **The following sentences represent the facts and relationships in the situation above:**

  $(\forall X)(\exists Y)p(X, Y)$

  $(\forall X)(\forall Y)(\forall Z)p(X, Y) \wedge p(X, Z) \rightarrow gp(X, Z)$

- **The goal is to find a W such that gp(john, W) or** $(\exists W)(gp(john, W))$.
- **The negation of the goal is** $\neg(\exists W)(gp(john, W))$
- **Or :** $\neg\, gp(john, W)$

- **The clause form for the predicates of this problem is:**

  ①$p(X, pa(X))$

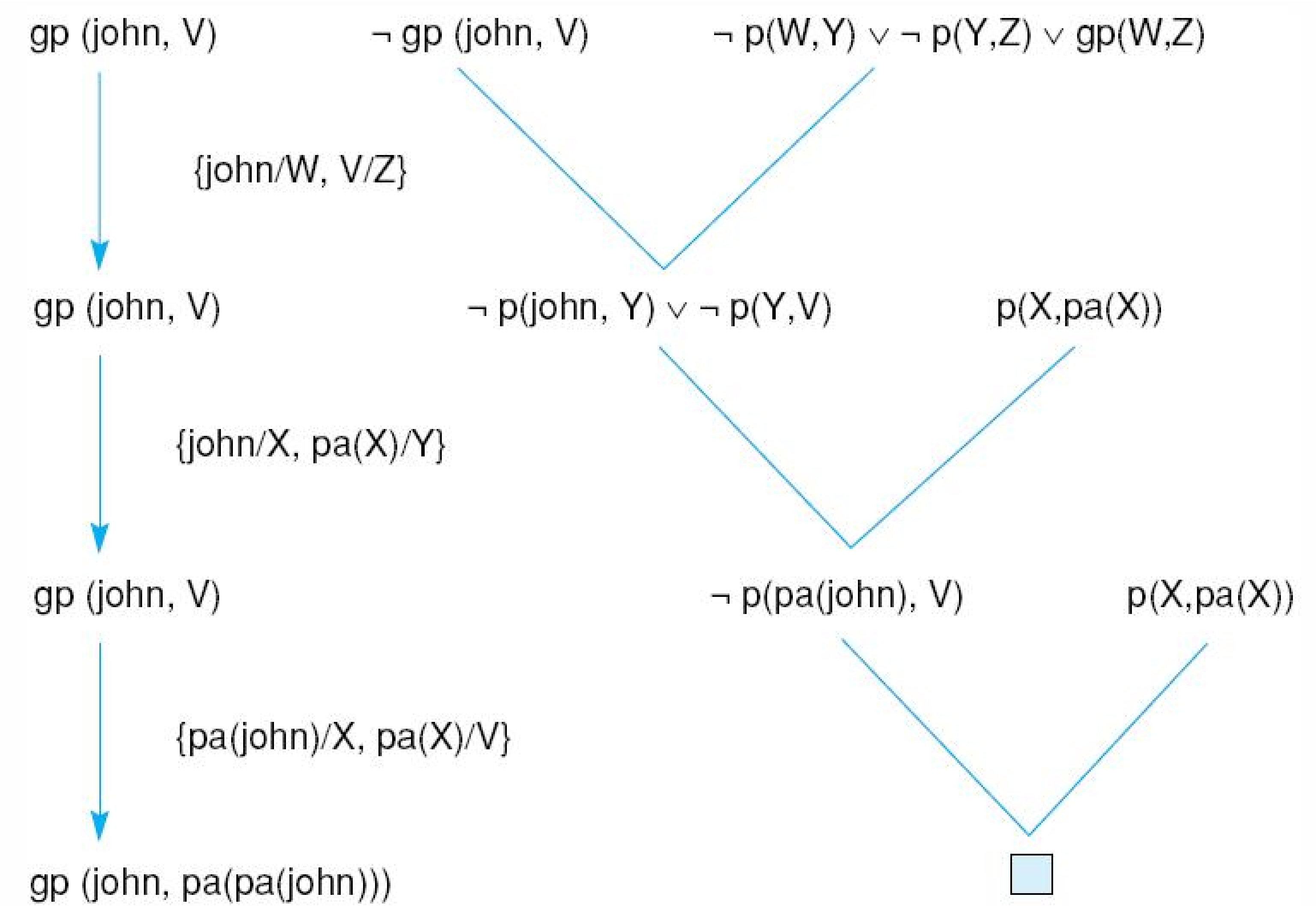
  ②$\neg p(W,Y) \lor \neg p(Y, Z) \lor gp(W, Z)$

  ③$\neg gp(john, V)$

- **The unification substitutions result is:**

  $$gp(john, pa(pa(john)))$$

Fig 14.12 Skolemization as part of the answer extraction process.



基于归结反演的问题求解步骤

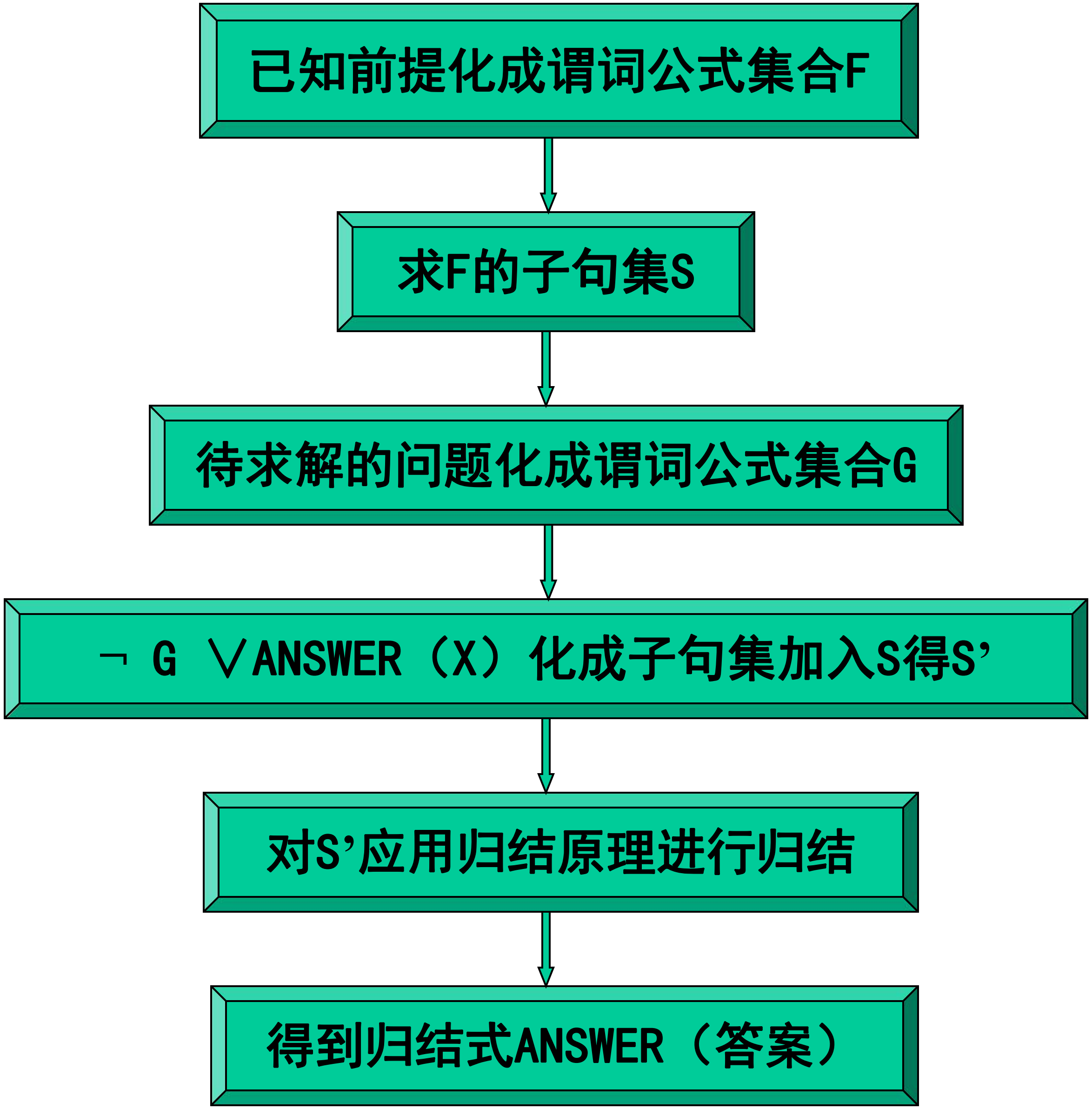
1）把已知前提用谓词公式表示出来，并且化为相应的子句集S。

2）把待求解的问题也用谓词公式表示出来，然后把它的否定式与谓词ANSWER构成一个析取式，ANSWER是一个为了求解问题而专设的谓词，其变元必须与问题公式的变元完全一致。

3）把此析取式化为子句集，并且把该子句集并入到子句集S中，得到子句集S’。

4）对S’应用归结原理进行归结。

5）若得到归结式ANSWER，则答案就在ANSWER中


已知前提化成谓词公式集合F
求F的子句集S
待求解的问题化成谓词公式集合G
¬ G ∨ANSWER（X）化成子句集加入S得S’
对S’应用归结原理进行归结
得到归结式ANSWER（答案）

已知 F_1 ：王(Wang)先生是小李(Li)的老师。

F_2 ：小李与小张(Zhang)是同班同学。

F_3 ：如果x与y是同学，则x的老师也是y的老师。

求：小张的老师是谁？

解: 首先定义谓词：

$T(x, y)$: x是y的老师。

$C(x, y)$: x与y是同班同学。

把已知前提及待求解的问题表示成谓词公式：

F_1 : $T(Wang, Li)$

F_2 : $C(Li, Zhang)$

F_3 : $(\forall x)(\forall y)(\forall z)(C(x, y) \wedge T(z, x) \rightarrow T(z, y))$

G: $\neg(\exists x) T(x, Zhang) \vee ANSWER(x)$

把上述公式化为子句集：

(1) $T(Wang, Li)$

(2) $C(Li, Zhang)$

(3) $\neg C(x, y) \vee \neg T(z, x) \vee T(z, y)$

(4) $\neg T(u, Zhang) \vee ANSWER(u)$

应用归结原理进行归结：

(5) $\neg C(Li, y) \vee T(Wang, y)$ (1)与(3)归结

(6) $\neg C(Li, Zhang) \vee ANXWER(Wang)$ (4)与(5)归结

(7) $ANXWER(Wang)$ (2)与(6)归结

由ANSWER(Wang)得出小张的老师是王老师。

上述归结过程可用图3所示的归结树表示。

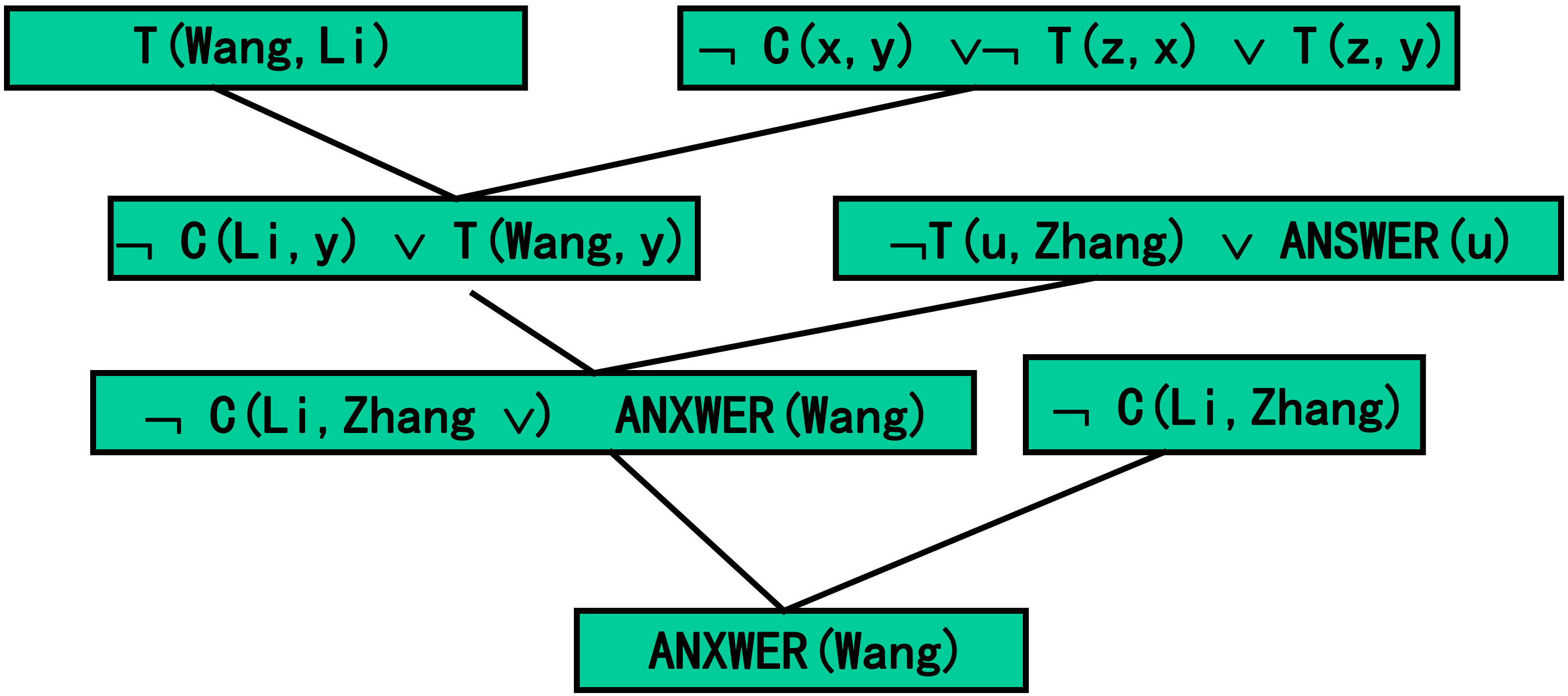

T(Wang, Li)
¬ C(x, y) ∨¬ T(z, x) ∨ T(z, y)
¬ C(Li, y) ∨ T(Wang, y)
¬T(u, Zhang) ∨ ANSWER(u)
¬ C(Li, Zhang ∨) ANXWER(Wang)
¬ C(Li, Zhang)
ANXWER(Wang)

图3 归结树

例:设A，B，C三人中有人从不说真话，也有人从不说假话，某人向这三人分别提出同一个问题：谁是说谎者？A答："B和C都是说谎者"；B答："A和C都是说谎者"；C答："A和B中至少有一个是说谎者"。

求谁是说谎者?

解: 设谓词T(x)：x说真话。

如果A说的是真话，则有 $T(A) \rightarrow \neg T(B) \wedge \neg T(C)$

如果A说的是假话，则有$\neg T(A) \rightarrow T(B) \vee T(C)$

对B和C说的话作相同的处理，可得：

$T(B) \rightarrow \neg T(A) \wedge \neg T(C)$

$\neg T(B) \rightarrow T(A) \vee T(C)$

$T(C) \rightarrow \neg T(A) \wedge \neg T(B)$

$\neg T(C) \rightarrow T(A) \vee T(B)$

把上面的这些公式化成子句集，得到S：

(1) $\neg T(A) \vee \neg T(B)$

(2) $\neg T(A) \vee \neg T(C)$

(3) $T(A) \vee T(B) \vee T(C)$

(4) $\neg T(B) \vee \neg T(C)$

(5) $\neg T(C) \vee \neg T(A) \vee \neg T(B)$

(6) $T(C) \vee T(B)$

(7) $T(C) \vee T(B)$

- 首先求谁说的是真话。

把目标公式$(\exists x)T(x)$的否定式$\neg(\exists x)T(x)$与ANSWER(x)组成的析取式化为子句得到：

(8) $\neg T(x) \lor ANSWER(x)$

- 应用归结原理对S’进行归结：

(9) $\neg T(A) \lor T(C)$　　(1)和(7)归结

(10) $T(C)$　　(6)和(9)归结

(11) ANSWER(C)　　(8)和(10)归结

所以C是老实人，即C从不说假话。

- 此外，推不出ANSWER（B）和ANSWER（A）。

- 证明A和B不是老实人。

设A不是老实人，则有$\neg T(A)$，把它否定并入S中，得到子句集S_2，即S_2比S多如下一个子句：

(8) $\neg(\neg T(A))$即$T(A)$

应用归结原理对S2进行归结：

(9) $\neg T(A) \vee T(C)$　　(1) 和(7)归结

(10) $\neg T(A)$　　(2) 和(9)归结

(11)NIL　　(8) 和(10)归结

所以A不是老实人。

- 同理，可证明B也不是老实人。

Put the following predicate calculus statements in clause form:

$$\forall X \forall Y father(X,Y) \vee mother(X,Y) \rightarrow parent(X,Y) \wedge$$

$$\forall X \forall Y \forall Z parent(X,Y) \wedge parent(X,Z) \rightarrow sibling(X,Y)$$

Solution:

$$xa \neg father(X,Y) \wedge \neg mother(X,Y) \vee parent(X,Y)$$

$$xb \neg parent(U,V) \wedge \neg parent(U,W) \vee sibling(V,W)$$

Prove “Socrates is mortal” from the statements “All men are mortal”, “all philosophers are men” and “Socrates is a philosopher” using resolution refutation. Following the steps in Section 14.2.1 on page 583, illustrate how to change the premises to predicates, and how to convert the predicates to clauses. Finally, produce a graph similar to Figure 14.3.

Solution:

Predicates:

$$\forall X(man(X) \rightarrow mortal(X))$$

$$\forall Y(philosopher(Y) \rightarrow man(Y))$$

$$philosopher(Socrates)$$

$$\neg mortal(Socrates)$$

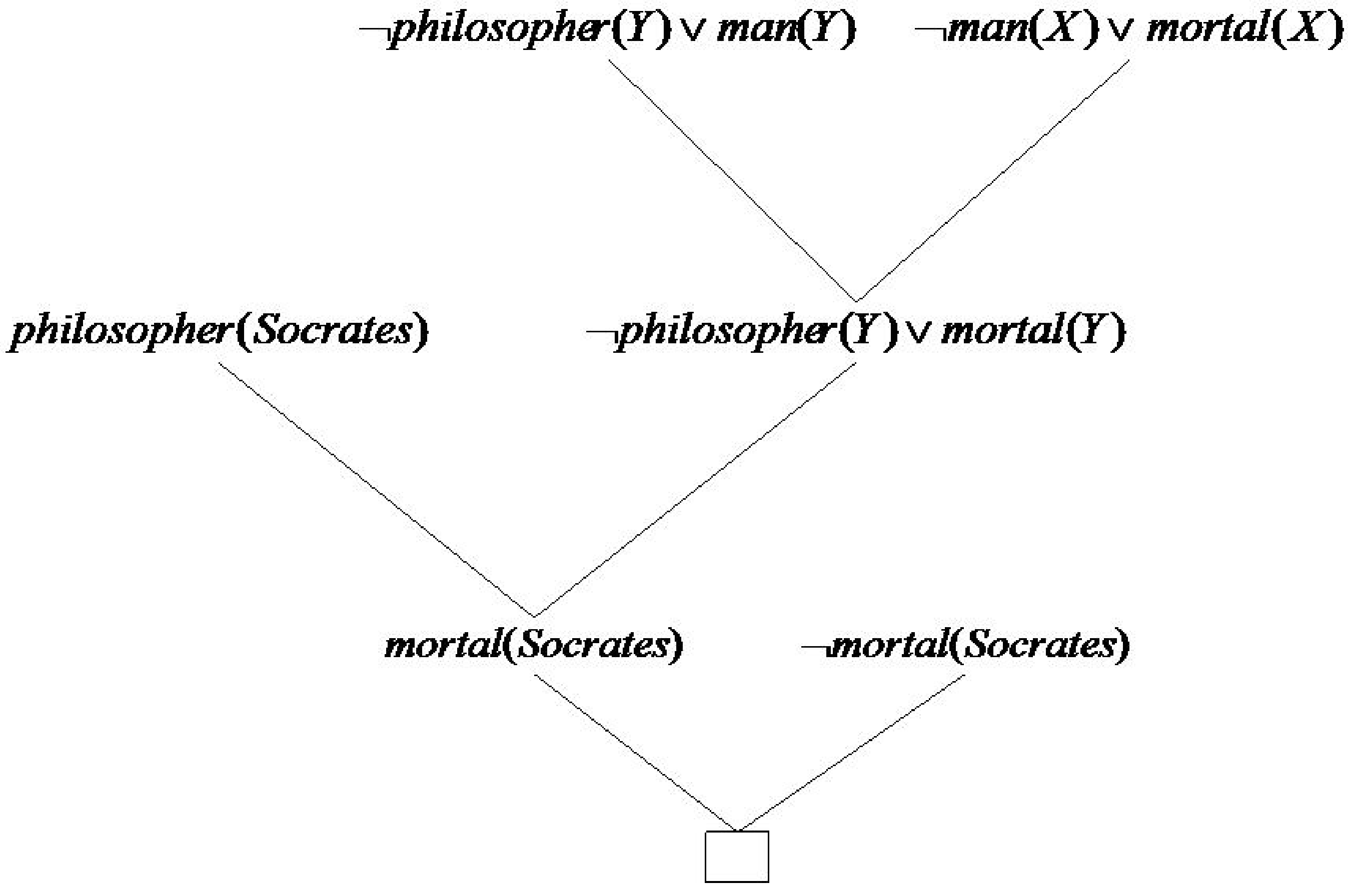
Clause form:

$$\neg man(X) \vee mortal(X)$$

$$\neg philosopher(Y) \vee man(Y)$$

$$philosopher(Socrates)$$

$$\neg mortal(Socrates)$$


¬philosopher(Y) ∨ man(Y)
¬man(X) ∨ mortal(X)
philosopher(Socrates)
¬philosopher(Y) ∨ mortal(Y)
mortal(Socrates)
¬mortal(Socrates)

Exercises

3.

13.