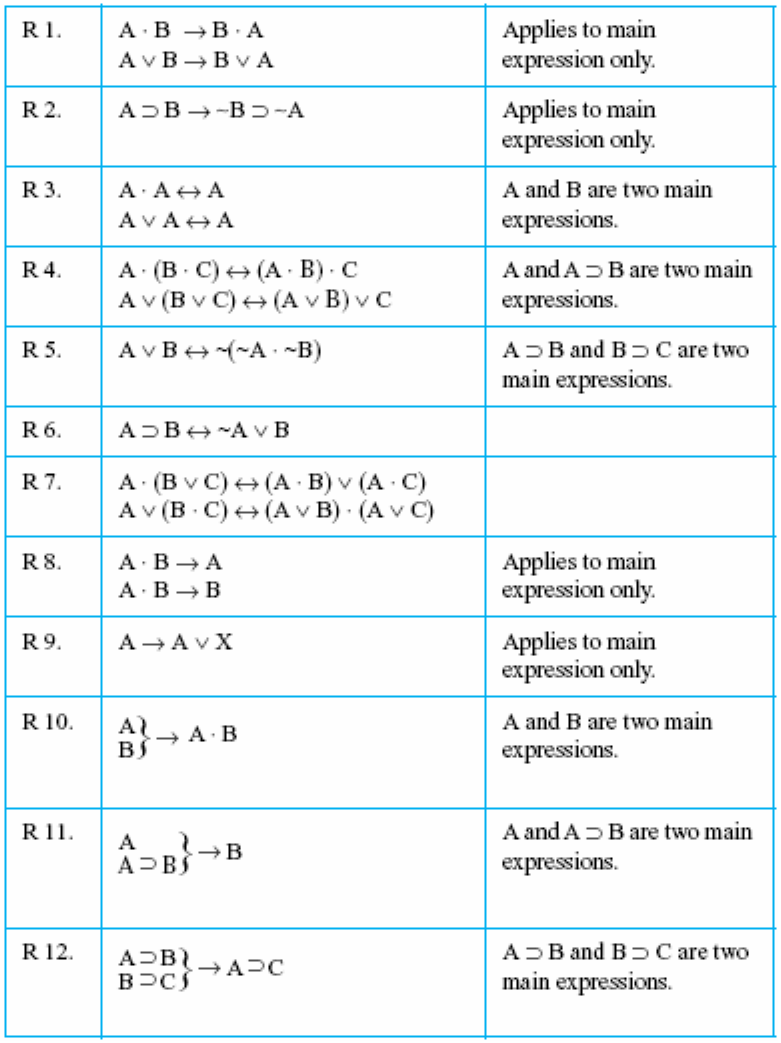
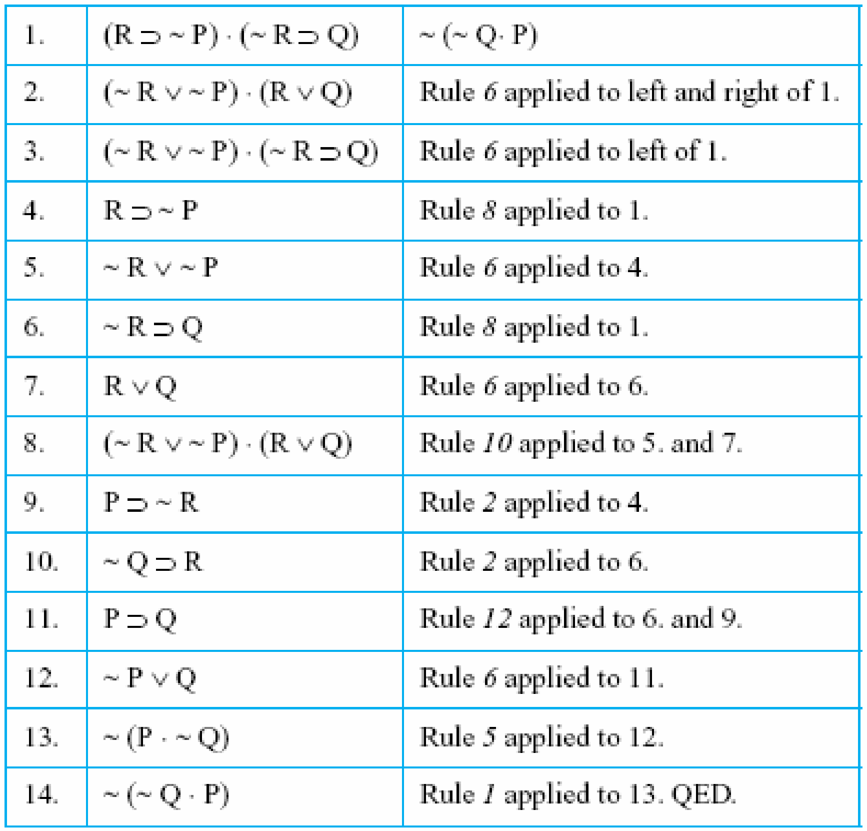
Fig 13.1a Transformation rules for logic problems, from Newell and Simon (1961).



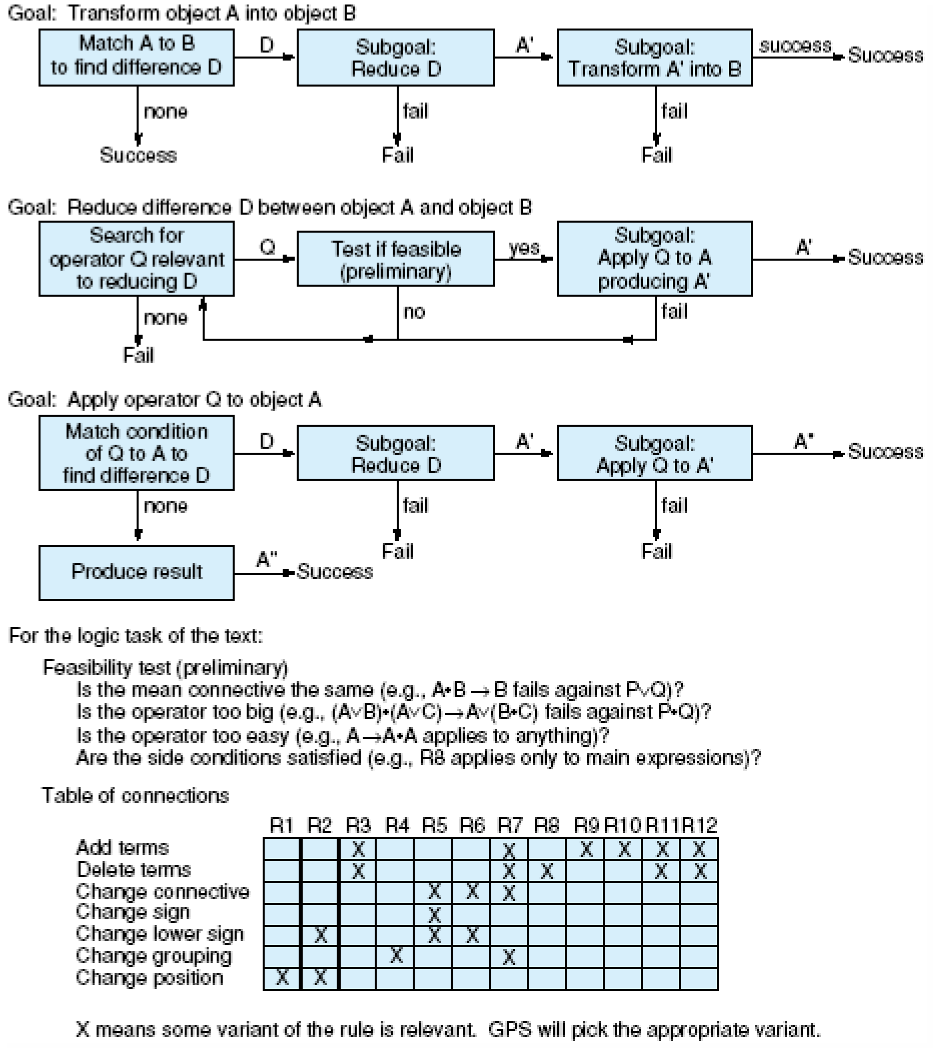
R 1.	$A \cdot B \rightarrow B \cdot A$ $A \vee B \rightarrow B \vee A$	Applies to main expression only.
R 2.	$A \supset B \rightarrow \sim B \supset \sim A$	Applies to main expression only.
R 3.	$A \cdot A \leftrightarrow A$ $A \vee A \leftrightarrow A$	A and B are two main expressions.
R 4.	$A \cdot (B \cdot C) \leftrightarrow (A \cdot B) \cdot C$ $A \vee (B \vee C) \leftrightarrow (A \vee B) \vee C$	A and $A \supset B$ are two main expressions.
R 5.	$A \vee B \leftrightarrow \sim(\sim A \cdot \sim B)$	$A \supset B$ and $B \supset C$ are two main expressions.
R 6.	$A \supset B \leftrightarrow \sim A \vee B$	
R 7.	$A \cdot (B \vee C) \leftrightarrow (A \cdot B) \vee (A \cdot C)$ $A \vee (B \cdot C) \leftrightarrow (A \vee B) \cdot (A \vee C)$	
R 8.	$A \cdot B \rightarrow A$ $A \cdot B \rightarrow B$	Applies to main expression only.
R 9.	$A \rightarrow A \vee X$	Applies to main expression only.
R 10.	$\left.\begin{array}{l}A\\B\end{array}\right\}\rightarrow A \cdot B$	A and B are two main expressions.
R 11.	$\left.\begin{array}{l}A\\A \supset B\end{array}\right\}\rightarrow B$	A and $A \supset B$ are two main expressions.
R 12.	$\left.\begin{array}{l}A \supset B\\B \supset C\end{array}\right\}\rightarrow A \supset C$	$A \supset B$ and $B \supset C$ are two main expressions.

Fig 13.1b A proof of a theorem in propositional calculus, from Newell and Simon (1961).



1.	$(R \supset \sim P) \cdot (\sim R \supset Q)$	$\sim(\sim Q \cdot P)$
2.	$(\sim R \vee \sim P) \cdot (R \vee Q)$	Rule *6* applied to left and right of 1.
3.	$(\sim R \vee \sim P) \cdot (\sim R \supset Q)$	Rule *6* applied to left of 1.
4.	$R \supset \sim P$	Rule *8* applied to 1.
5.	$\sim R \vee \sim P$	Rule *6* applied to 4.
6.	$\sim R \supset Q$	Rule *8* applied to 1.
7.	$R \vee Q$	Rule *6* applied to 6.
8.	$(\sim R \vee \sim P) \cdot (R \vee Q)$	Rule *10* applied to 5. and 7.
9.	$P \supset \sim R$	Rule *2* applied to 4.
10.	$\sim Q \supset R$	Rule *2* applied to 6.
11.	$P \supset Q$	Rule *12* applied to 6. and 9.
12.	$\sim P \vee Q$	Rule *6* applied to 11.
13.	$\sim(P \cdot \sim Q)$	Rule *5* applied to 12.
14.	$\sim(\sim Q \cdot P)$	Rule *1* applied to 13. QED.

Fig 13.2 Flow chart and difference reduction table for the General Problem Solver, from Newell and Simon (1963b).


Goal: Transform object A into object B
Match A to B to find difference D
D
Subgoal: Reduce D
A'
Subgoal: Transform A' into B
success
Success
none
Success
fail
Fail
fail
Fail
Goal: Reduce difference D between object A and object B
Search for operator Q relevant to reducing D
Q
Test if feasible (preliminary)
yes
Subgoal: Apply Q to A producing A'
A'
Success
none
Fail
no
fail
Goal: Apply operator Q to object A
Match condition of Q to A to find difference D
D
Subgoal: Reduce D
A'
Subgoal: Apply Q to A'
A"
Success
none
fail
Fail
fail
Fail
Produce result
A"
Success

For the logic task of the text:

Feasibility test (preliminary)

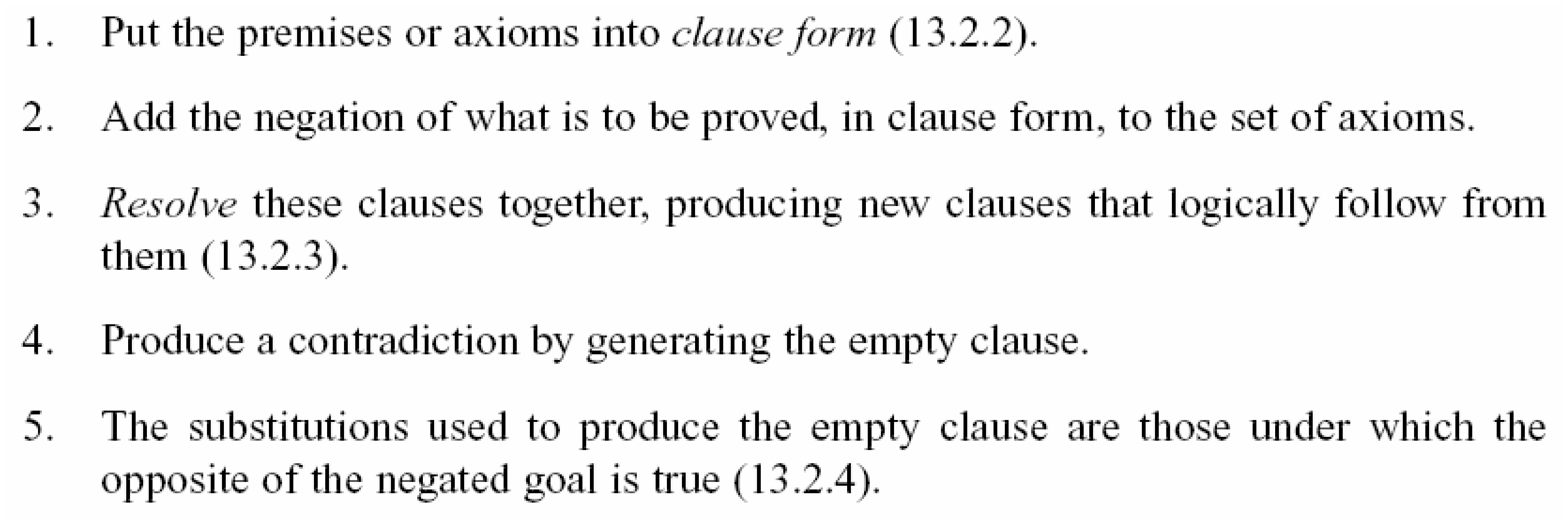
Is the mean connective the same (e.g., $A \cdot B \to B$ fails against $P \vee Q$)?
Is the operator too big (e.g., $(A \vee B) \cdot (A \vee C) \to A \vee (B \cdot C)$ fails against $P \cdot Q$)?
Is the operator too easy (e.g., $A \to A \cdot A$ applies to anything)?
Are the side conditions satisfied (e.g., R8 applies only to main expressions)?

Table of connections

	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12
Add terms			X				X		X	X	X	X
Delete terms			X				X	X			X	X
Change connective					X	X	X					
Change sign					X							
Change lower sign		X			X	X						
Change grouping				X			X					
Change position	X	X										

X means some variant of the rule is relevant. GPS will pick the appropriate variant.

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).

Clausal Form

- Before resolution can be applied, the WFF must be in a clausal form. The basic idea of clausal form is to express WFFs in a standard form that uses only the V, and possibly ¬.
- This conversion is necessary because resolution is an operation on pairs of disjuncts which produces new disjuncts, which simplifies the WFF.
- The full clausal form can express any predicate logic formula.

Algorithm to convert to clausal form (1)

1. Eliminate conditionals →, using the equivalence

   $P \rightarrow Q = \neg P \lor Q$

2. Eliminate negations or reduce the scope of negation to one atom.

   e.g., $\neg \neg P = P$

   $\neg(P \land Q) = \neg P \lor \neg Q$

   $\neg (\forall X)\ P(X) = (\exists X)\ \neg P(X)$

3. Standardize variables within a WFF so that the bound or dummy variables of each quantifier have unique names.

   e.g.,

Algorithm to convert to clausal form (2)

4. Eliminate existential quantifiers, by using Skolem functions, named after the Norwegian logician Thoralf Skolem.

   e.g., $(\exists X)\ L(X)$ is replaced by $L(a)$

   $(\exists X)\ (\exists Y)\ L(X, Y)$ is replaced by

   $(\exists X)\ L(X, f(X))$

5. Convert the WFF to prenex form which is a sequence of quantifiers followed by a matrix.
6. Convert the matrix to conjunctive normal form, which is a conjunctive of clauses. Each clause is a disjunction.
7. Drop the universal quantifiers.
8. Eliminate the conjunctive signs by writing the WFF as a set of clauses
9. Rename variables in clauses, if necessary, so that the same variable name is only used in one clause.

Fig 13.3 Resolution proof for the “dead dog” problem.

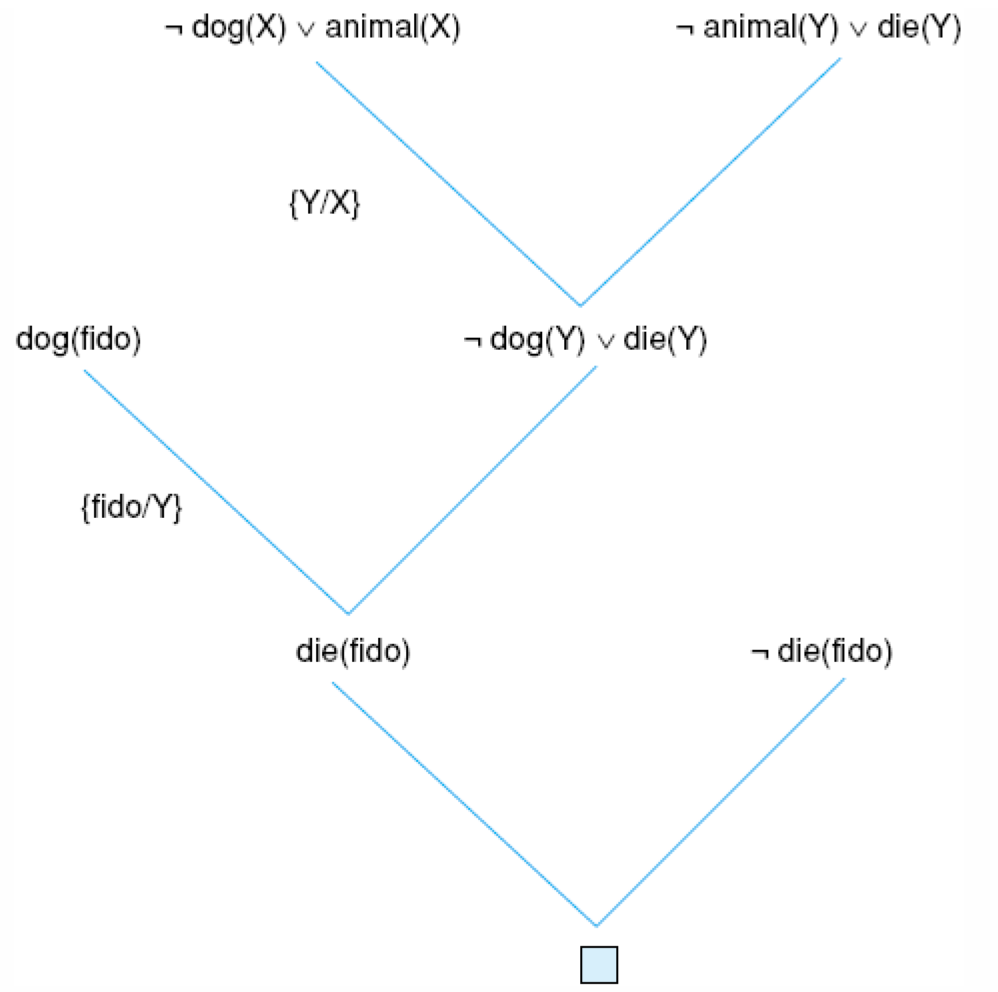
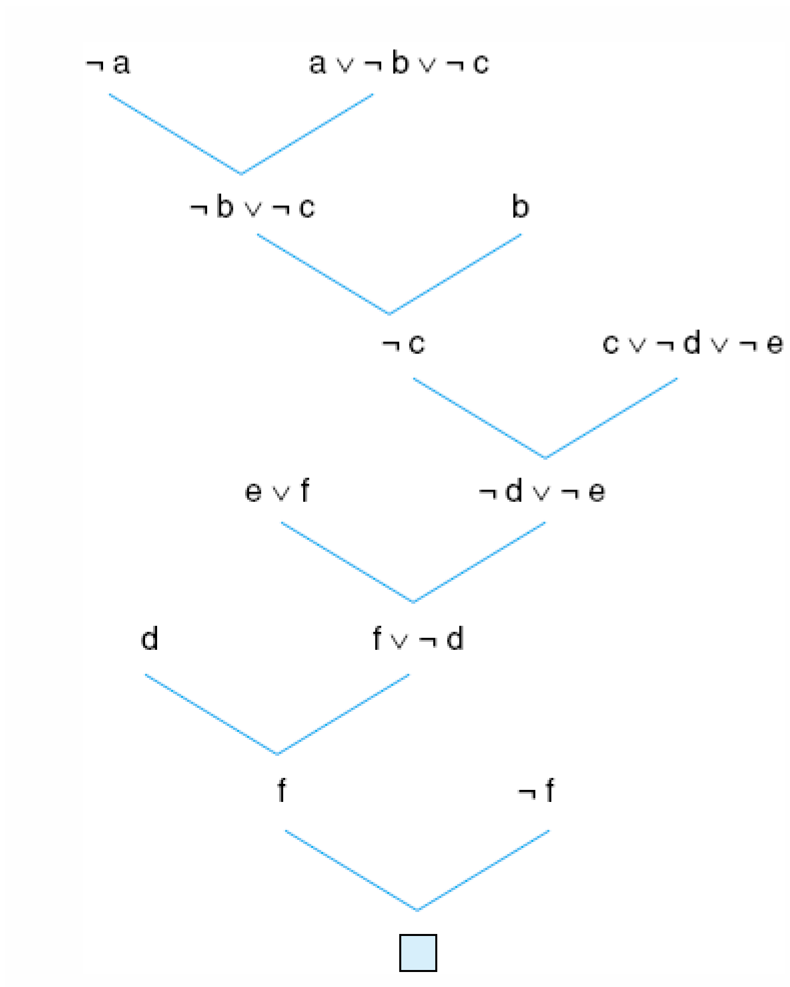


Fig 13.4 One resolution proof for an example from the propositional calculus.



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)**

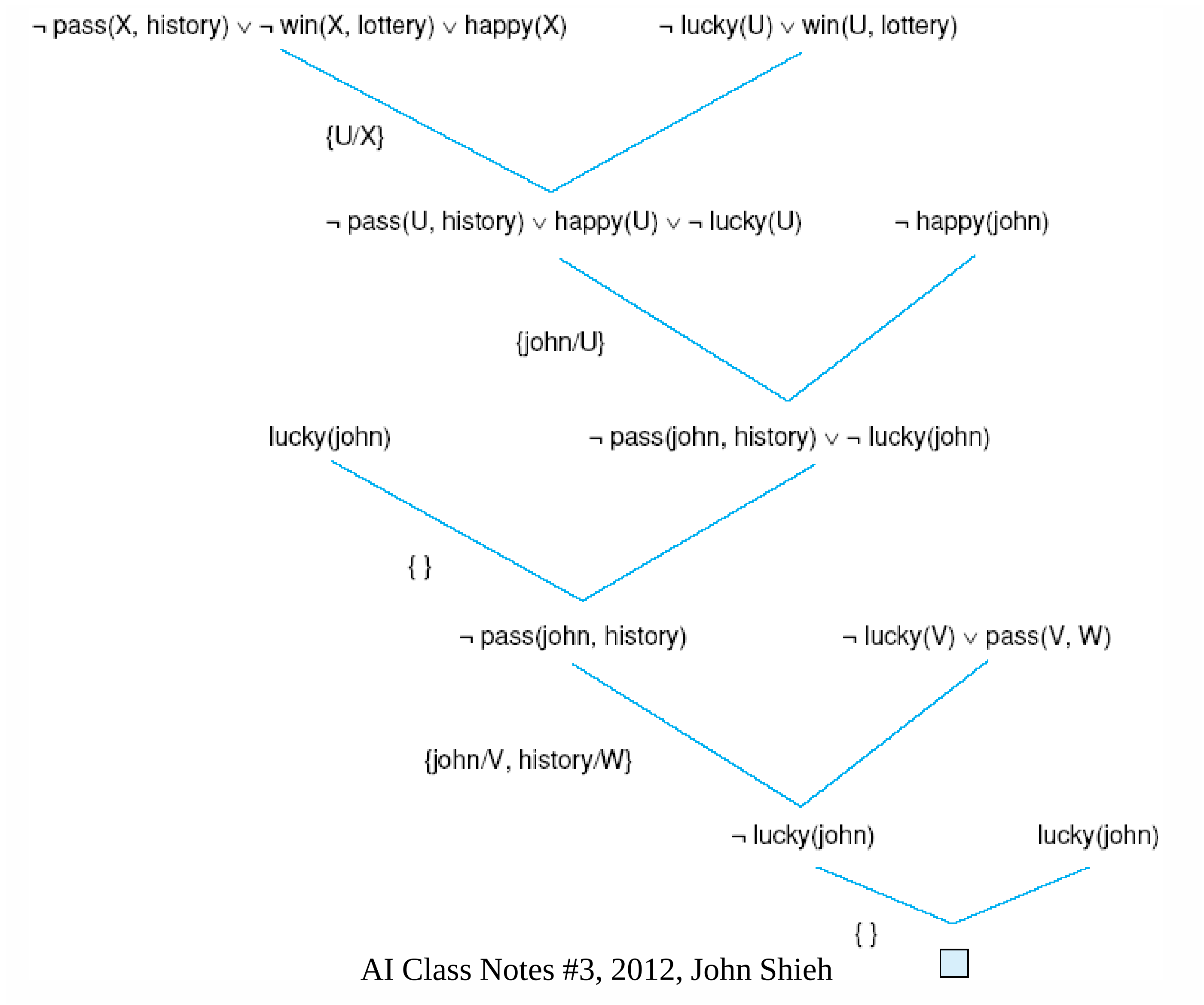
5. **lucky (john)**

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 13.5 One refutation for the “happy student” problem.


¬ pass(X, history) ∨ ¬ win(X, lottery) ∨ happy(X)
¬ lucky(U) ∨ win(U, lottery)
{U/X}
¬ pass(U, history) ∨ happy(U) ∨ ¬ lucky(U)
¬ happy(john)
{john/U}
lucky(john)
¬ pass(john, history) ∨ ¬ lucky(john)
{ }
¬ pass(john, history)
¬ lucky(V) ∨ pass(V, W)
{john/V, history/W}
¬ lucky(john)
lucky(john)
{ }

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 13.6 Resolution proof for the “exciting life” problem.

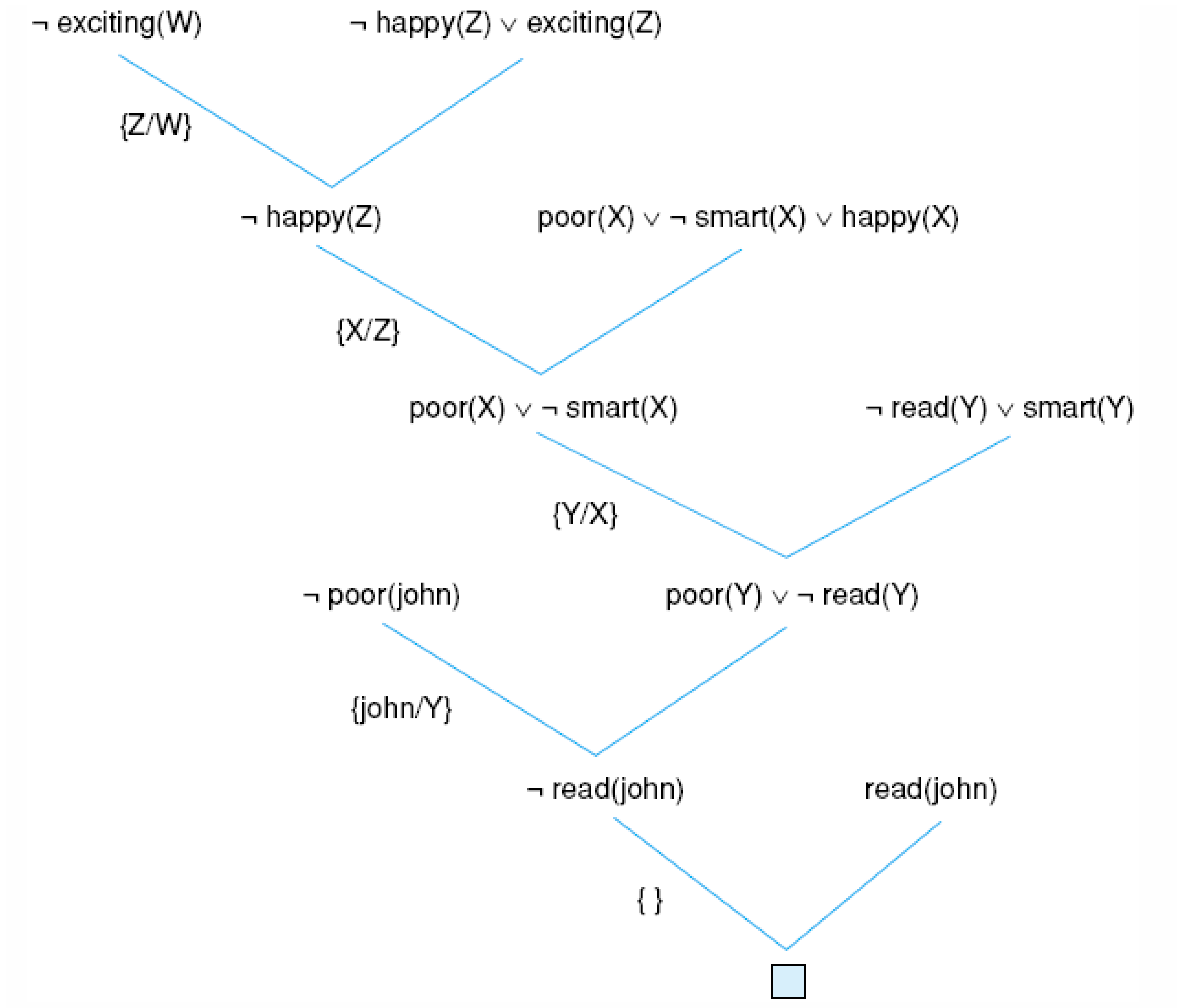


Fig 13.8 Complete state space for the “exciting life” problem generated by breadth-first search (to two levels).

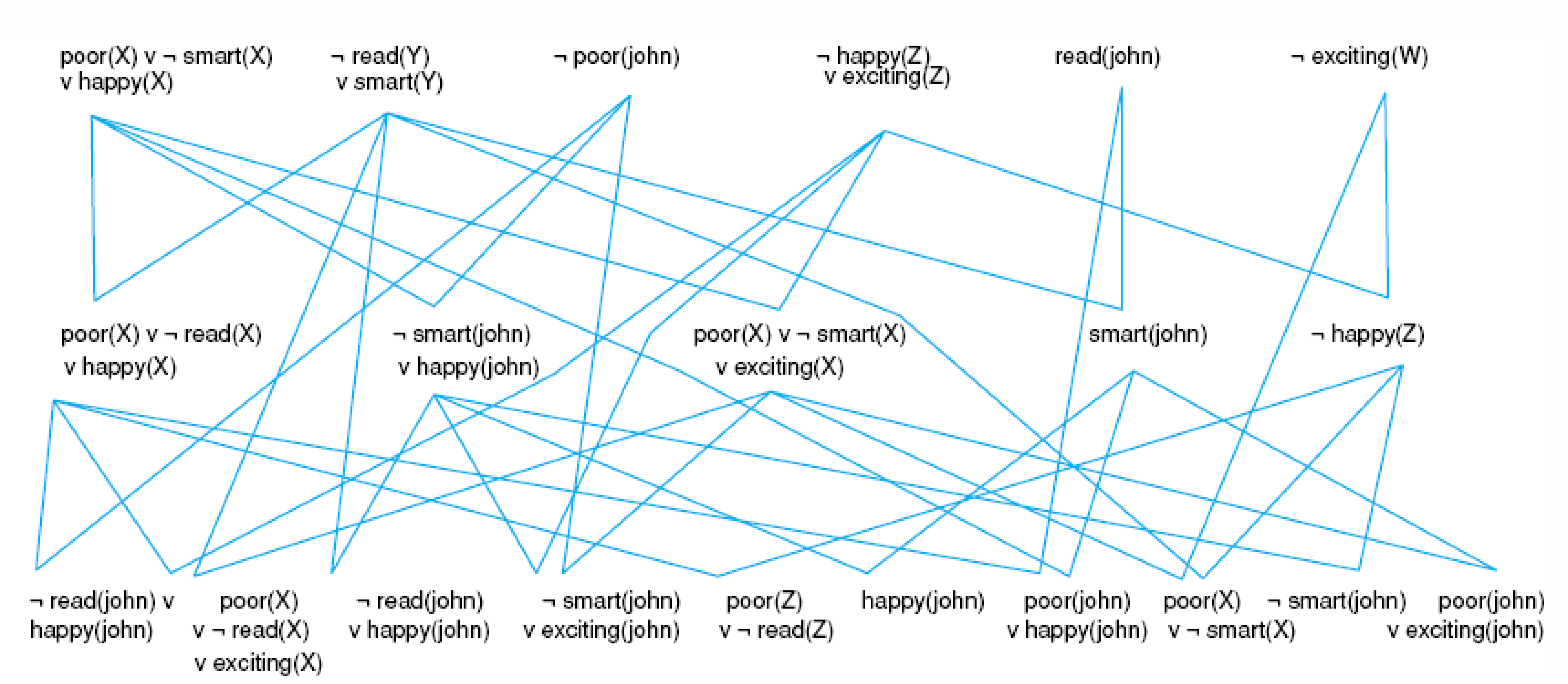

poor(X) v ¬ smart(X) v happy(X)
¬ read(Y) v smart(Y)
¬ poor(john)
¬ happy(Z) v exciting(Z)
read(john)
¬ exciting(W)
poor(X) v ¬ read(X) v happy(X)
¬ smart(john) v happy(john)
poor(X) v ¬ smart(X) v exciting(X)
smart(john)
¬ happy(Z)
¬ read(john) v happy(john)
poor(X) v ¬ read(X) v exciting(X)
¬ read(john) v happy(john)
¬ smart(john) v exciting(john)
poor(Z) v ¬ read(Z)
happy(john)
poor(john) v happy(john)
poor(X) v ¬ smart(X)
¬ smart(john)
poor(john) v exciting(john)

Fig 13.7 another resolution refutation for the example of Fig 13.6.

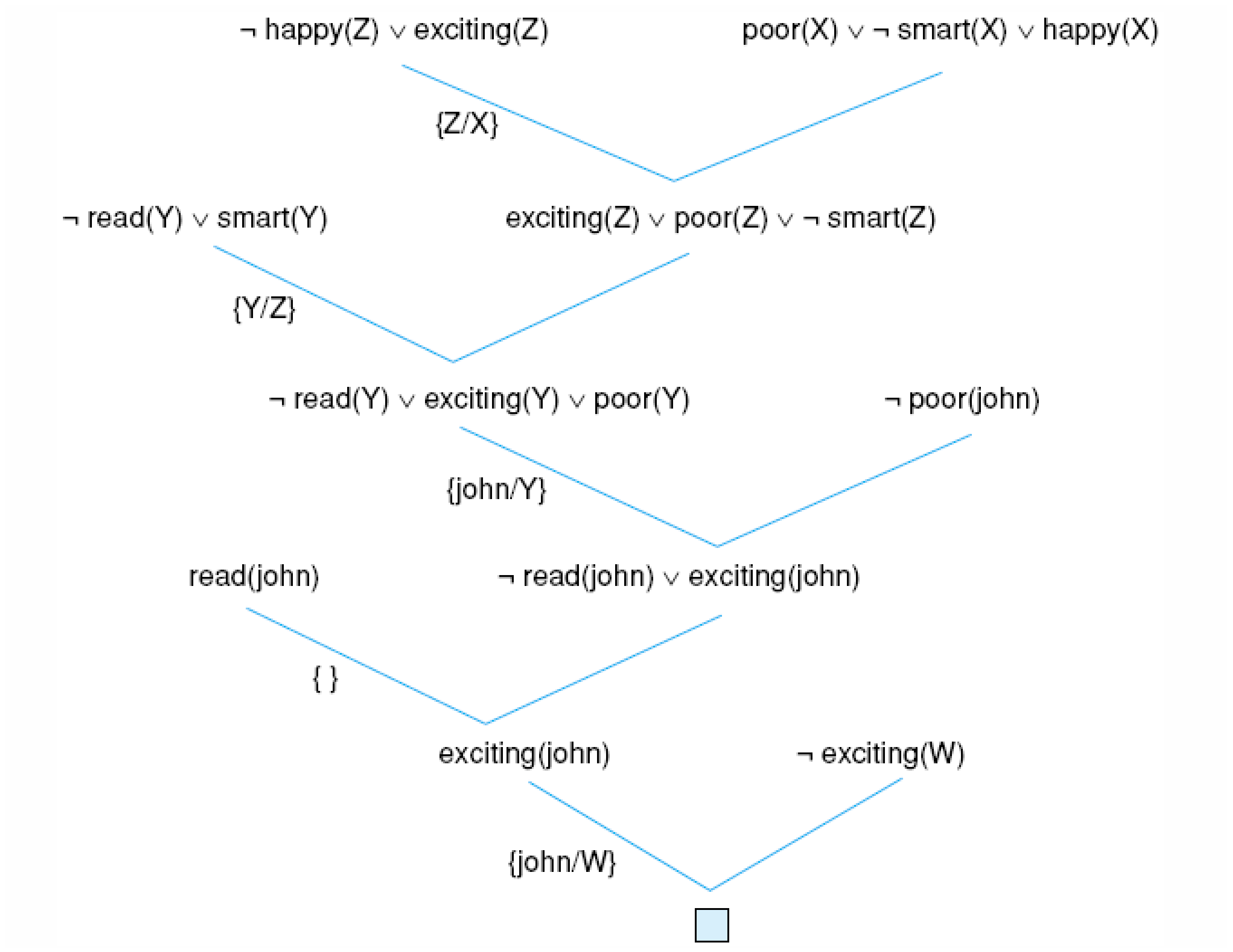

¬ happy(Z) ∨ exciting(Z)
poor(X) ∨ ¬ smart(X) ∨ happy(X)
{Z/X}
¬ read(Y) ∨ smart(Y)
exciting(Z) ∨ poor(Z) ∨ ¬ smart(Z)
{Y/Z}
¬ read(Y) ∨ exciting(Y) ∨ poor(Y)
¬ poor(john)
{john/Y}
read(john)
¬ read(john) ∨ exciting(john)
{ }
exciting(john)
¬ exciting(W)
{john/W}

Fig 13.9 Using the unit preference strategy on the “exciting life” problem.

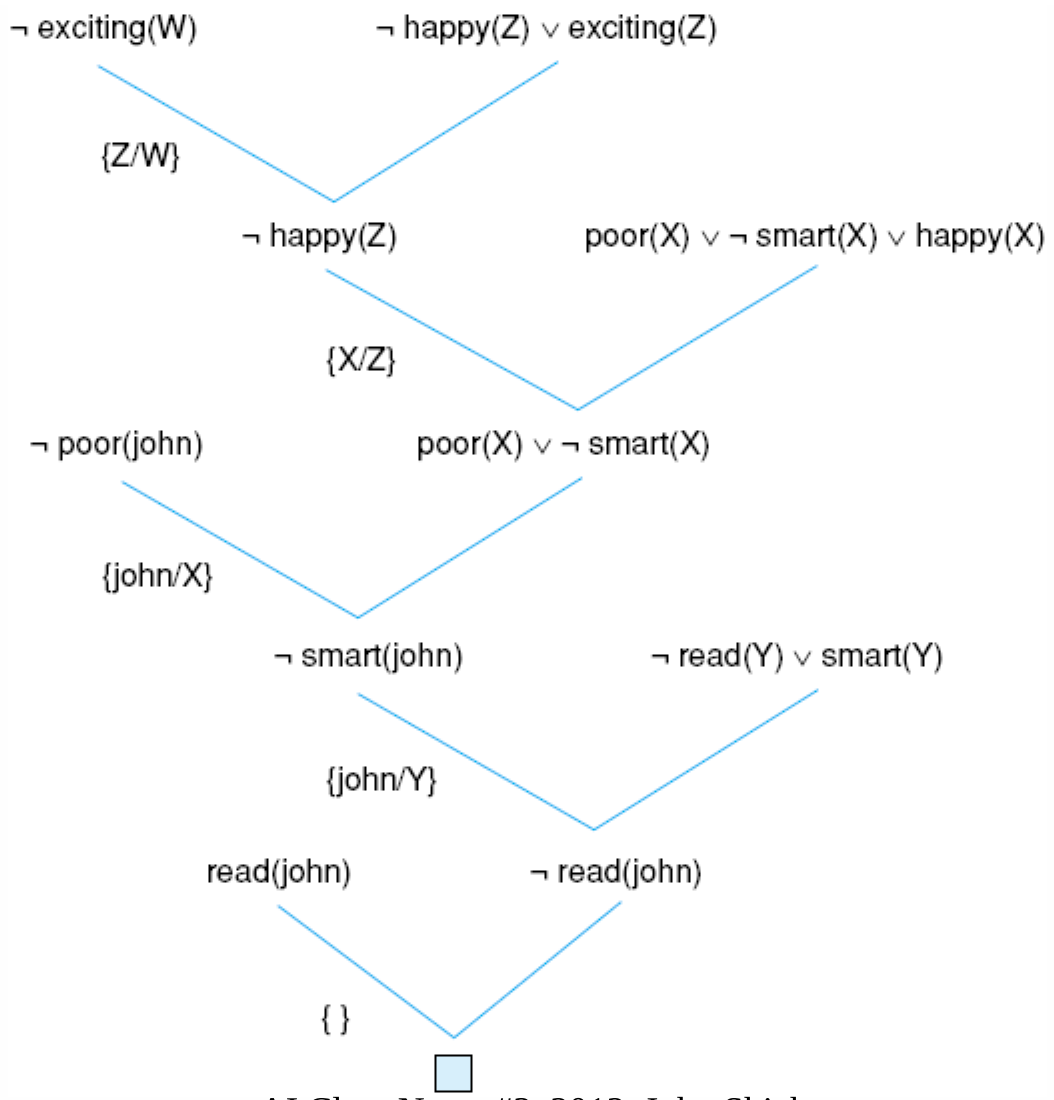


Fig 13.10 Unification substitutions of Fig 13.6 applied to the original query.

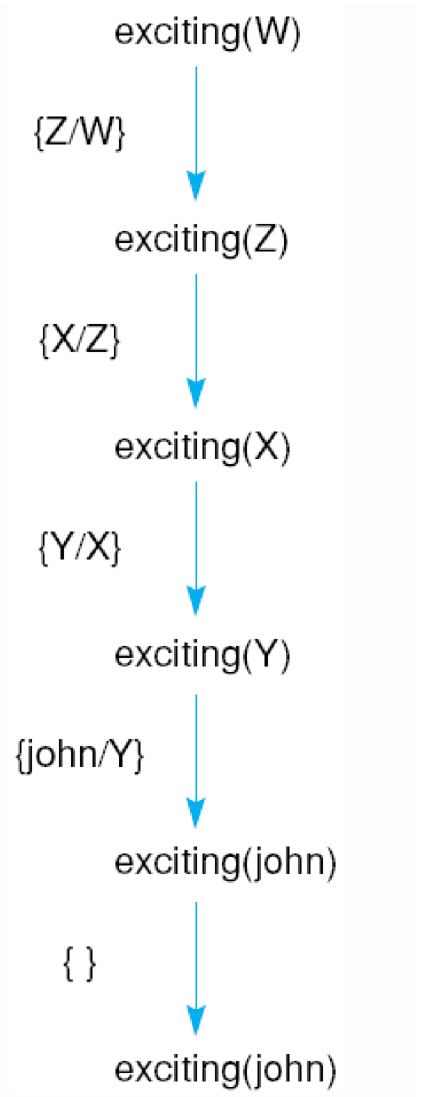


Fig 13.11 Answer extraction process on the “finding fido” problem.

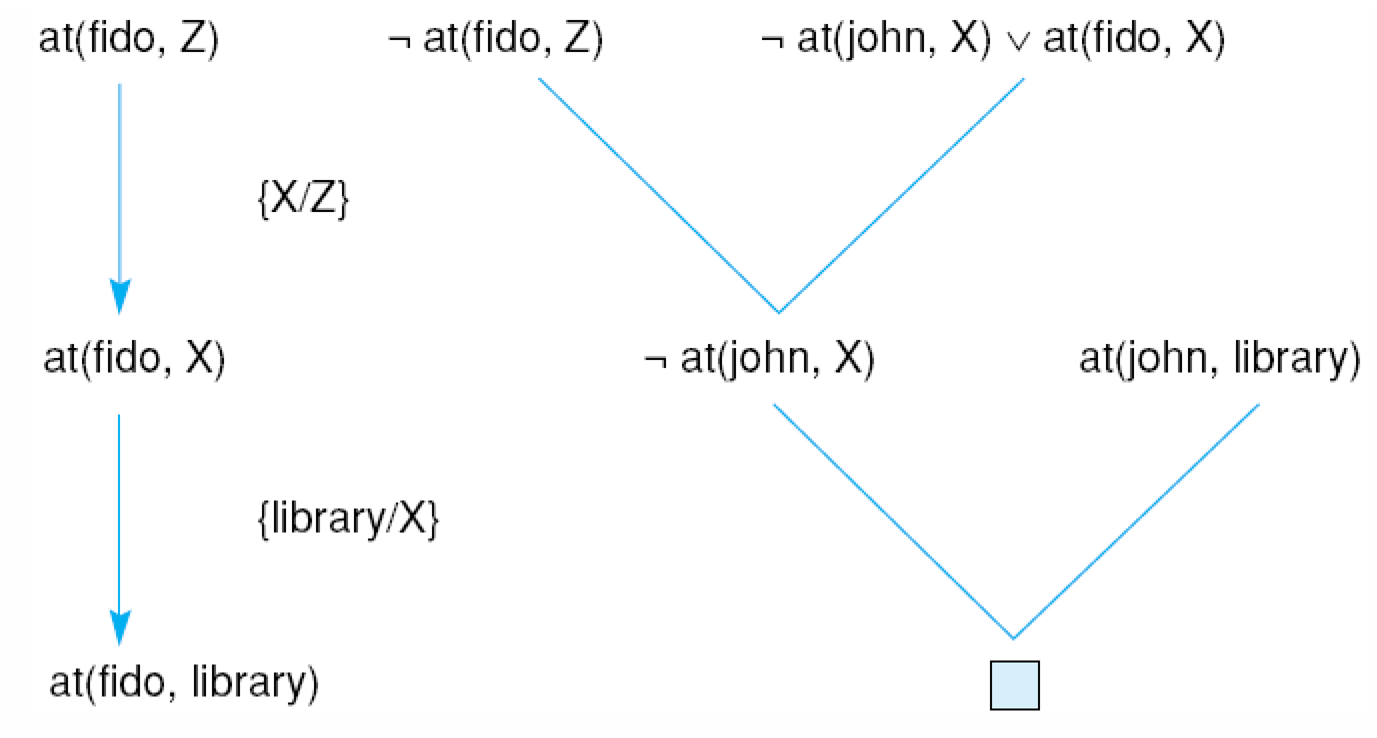
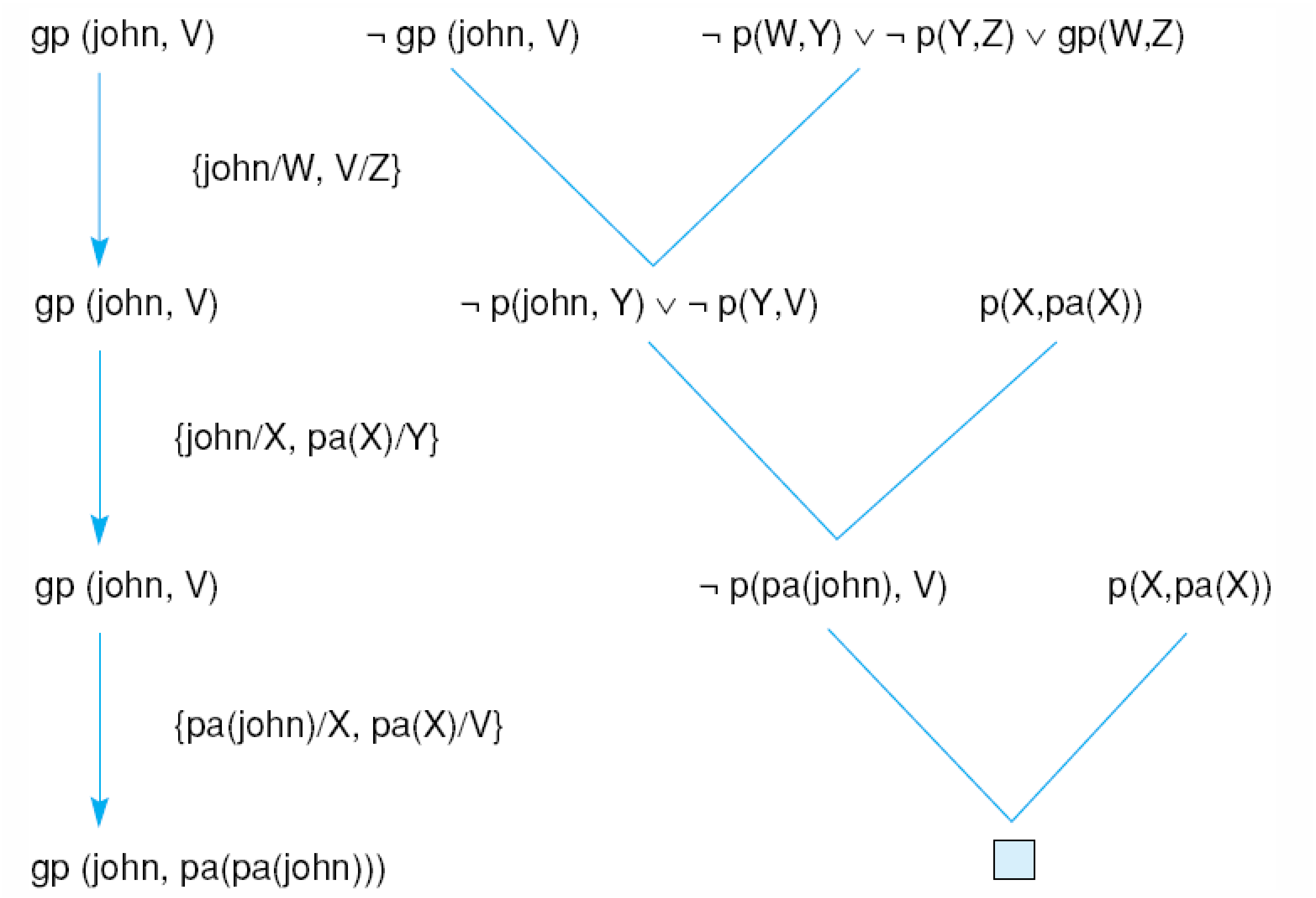


Fig 13.12 Skolemization as part of the answer extraction process.


gp (john, V)
¬ gp (john, V)
¬ p(W,Y) ∨ ¬ p(Y,Z) ∨ gp(W,Z)
{john/W, V/Z}
gp (john, V)
¬ p(john, Y) ∨ ¬ p(Y,V)
p(X,pa(X))
{john/X, pa(X)/Y}
gp (john, V)
¬ p(pa(john), V)
p(X,pa(X))
{pa(john)/X, pa(X)/V}
gp (john, pa(pa(john)))