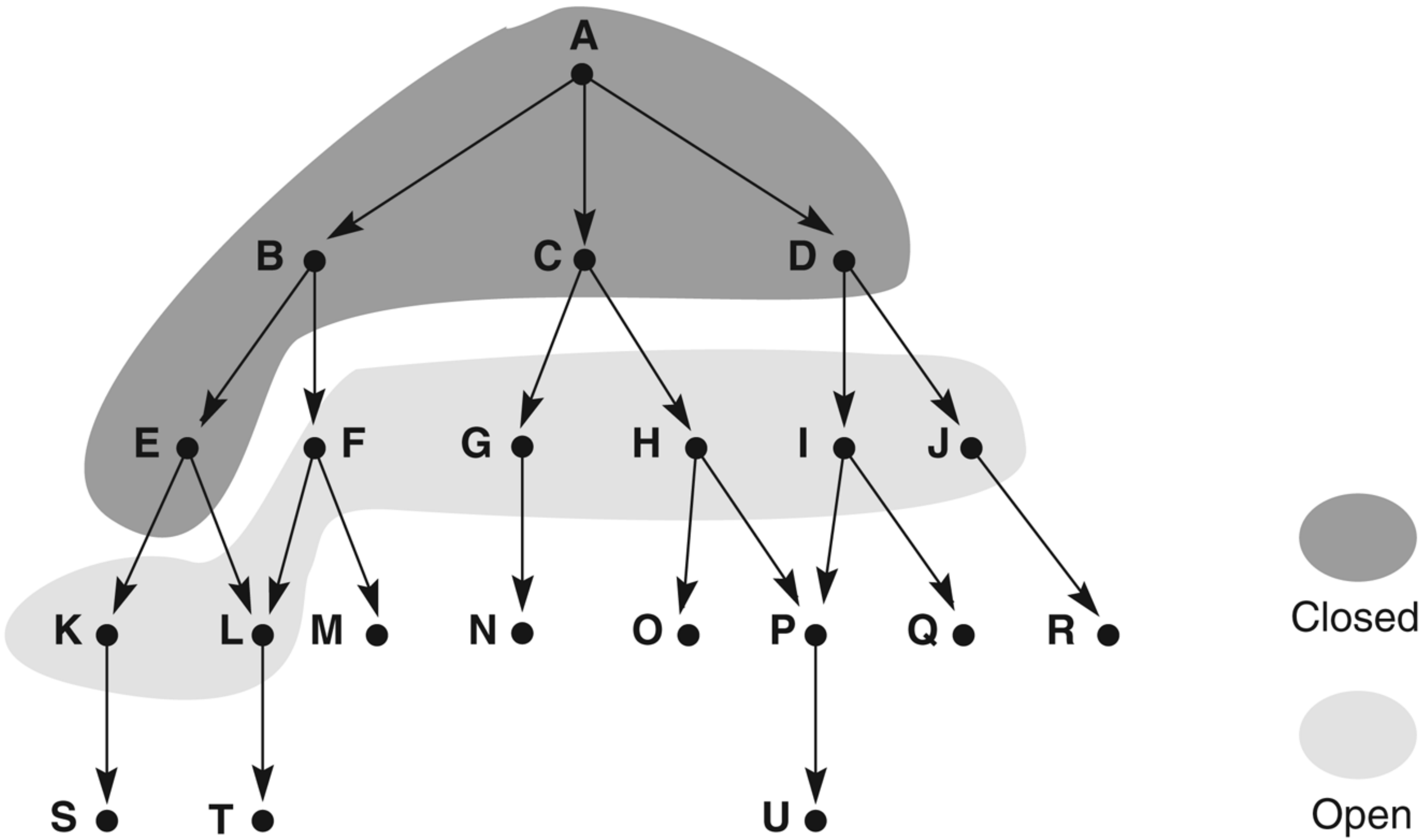
3 STRUCTURES AND STRATEGIES FOR STATE SPACE SEARCH

Figure 3.14: Graph of Figure 3.13 at iteration 6 of breadth-first search. States on open and closed are highlighted.



Function depth_first_search algorithm

```
begin
   open := [Start];                                                    % initialize
   closed := [ ];
   while open ≠ [ ] do                                                 % states remain
      begin
         remove leftmost state from open, call it X;
         if X is a goal then return SUCCESS                            % goal found
            else begin
               generate children of X;
               put X on closed;
               discard children of X if already on open or closed;     % loop check
               put remaining children on left end of open              % stack
            end
      end;
   return FAIL                                                         % no states left
end.
```

A trace of depth_first_search on the graph of Figure 3.13

1. **open = [A]; closed = [ ]**
2. **open = [B,C,D]; closed = [A]**
3. **open = [E,F,C,D]; closed = [B,A]**
4. **open = [K,L,F,C,D]; closed = [E,B,A]**
5. **open = [S,L,F,C,D]; closed = [K,E,B,A]**
6. **open = [L,F,C,D]; closed = [S,K,E,B,A]**
7. **open = [T,F,C,D]; closed = [L,S,K,E,B,A]**
8. **open = [F,C,D]; closed = [T,L,S,K,E,B,A]**
9. **open = [M,C,D],** as L is already on **closed**; **closed = [F,T,L,S,K,E,B,A]**
10. **open = [C,D]; closed = [M,F,T,L,S,K,E,B,A]**
11. **open = [G,H,D]; closed = [C,M,F,T,L,S,K,E,B,A]**

Figure 3.15: Breadth-first search of the 8-puzzle, showing order in which states were removed from open.

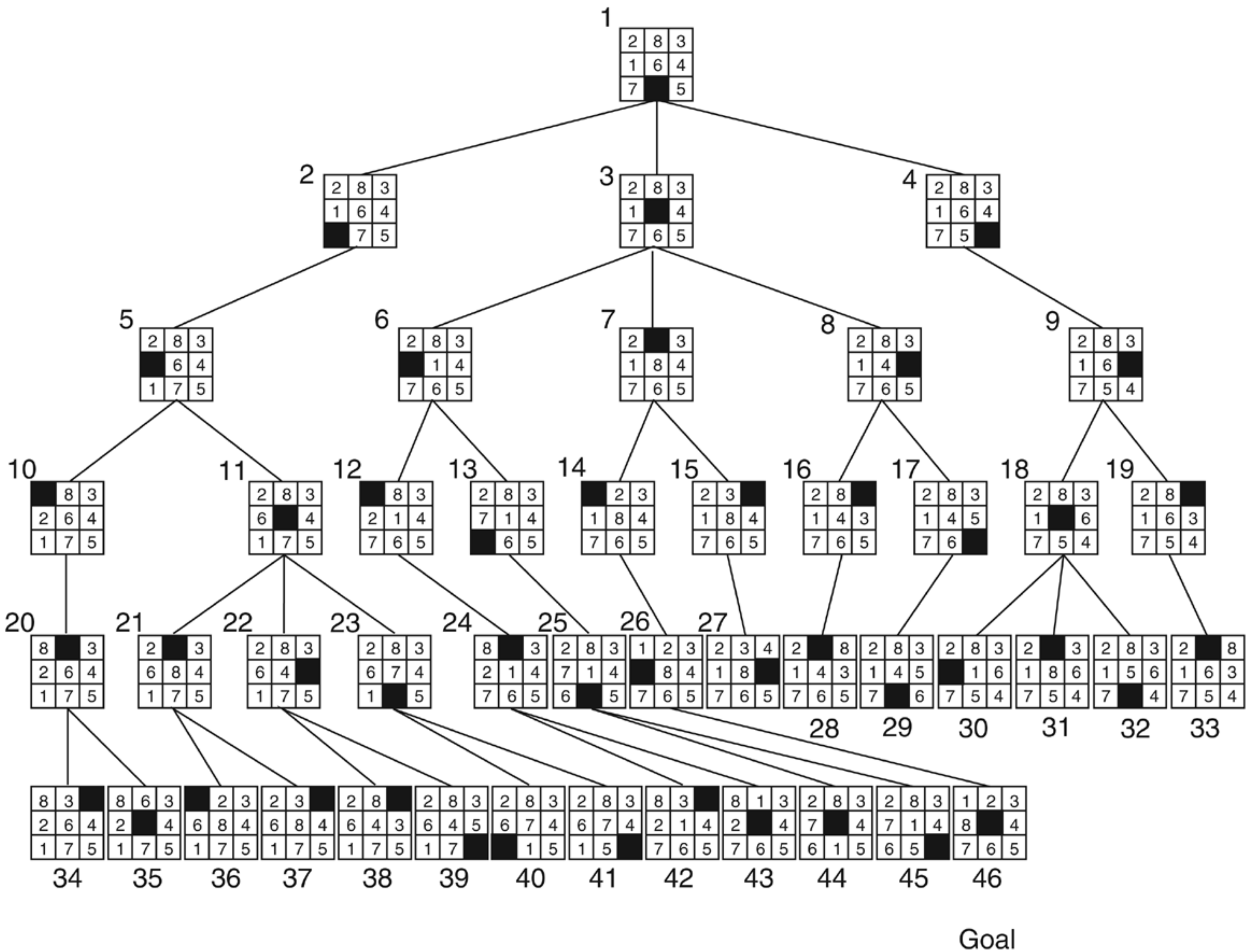


Figure 3.16: Graph of Figure 3.13 at iteration 6 of depth-first search. States on open and closed are highlighted.

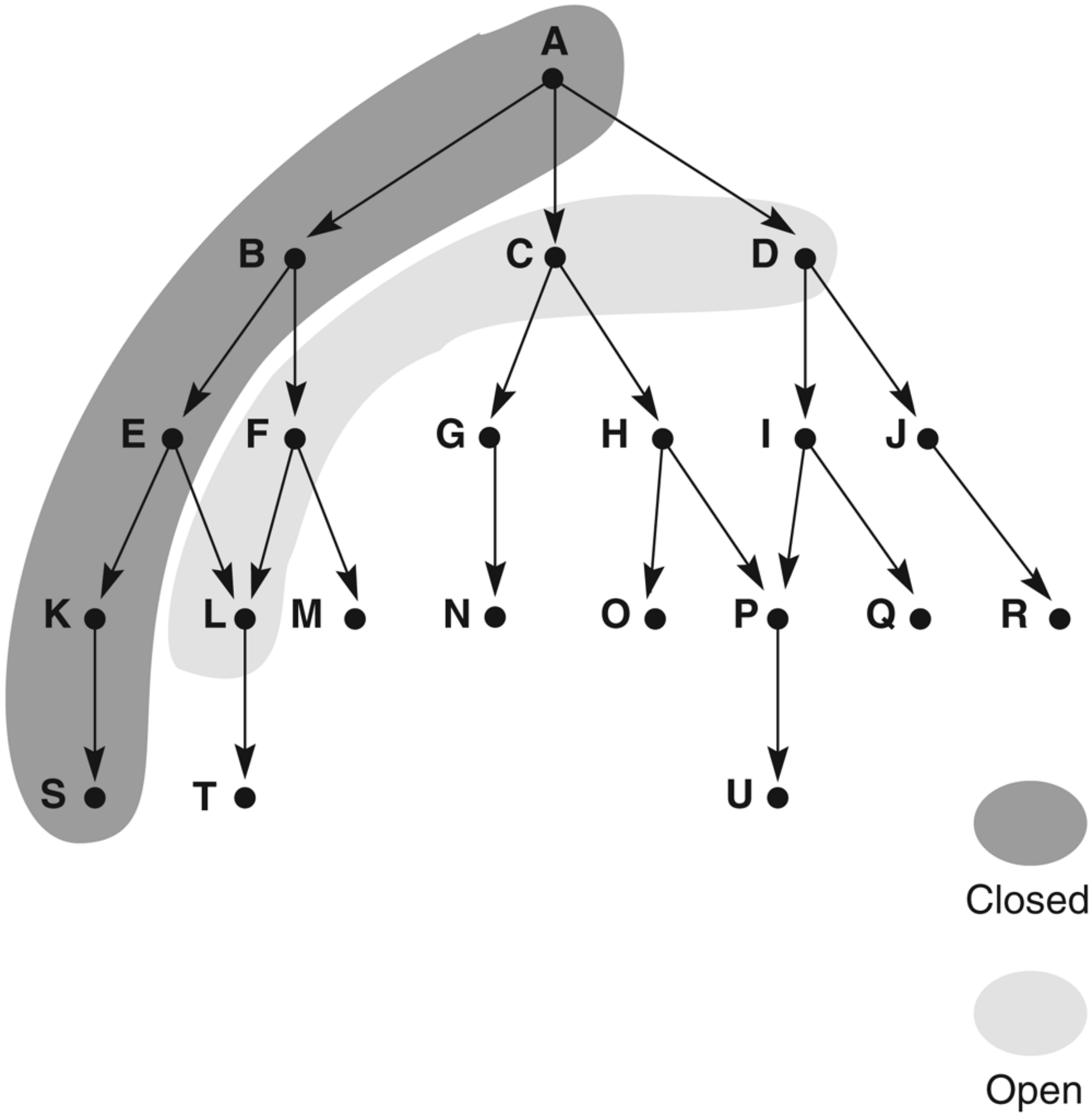


Figure 3.17: Depth-first search of the 8-puzzle with a depth bound of 5.

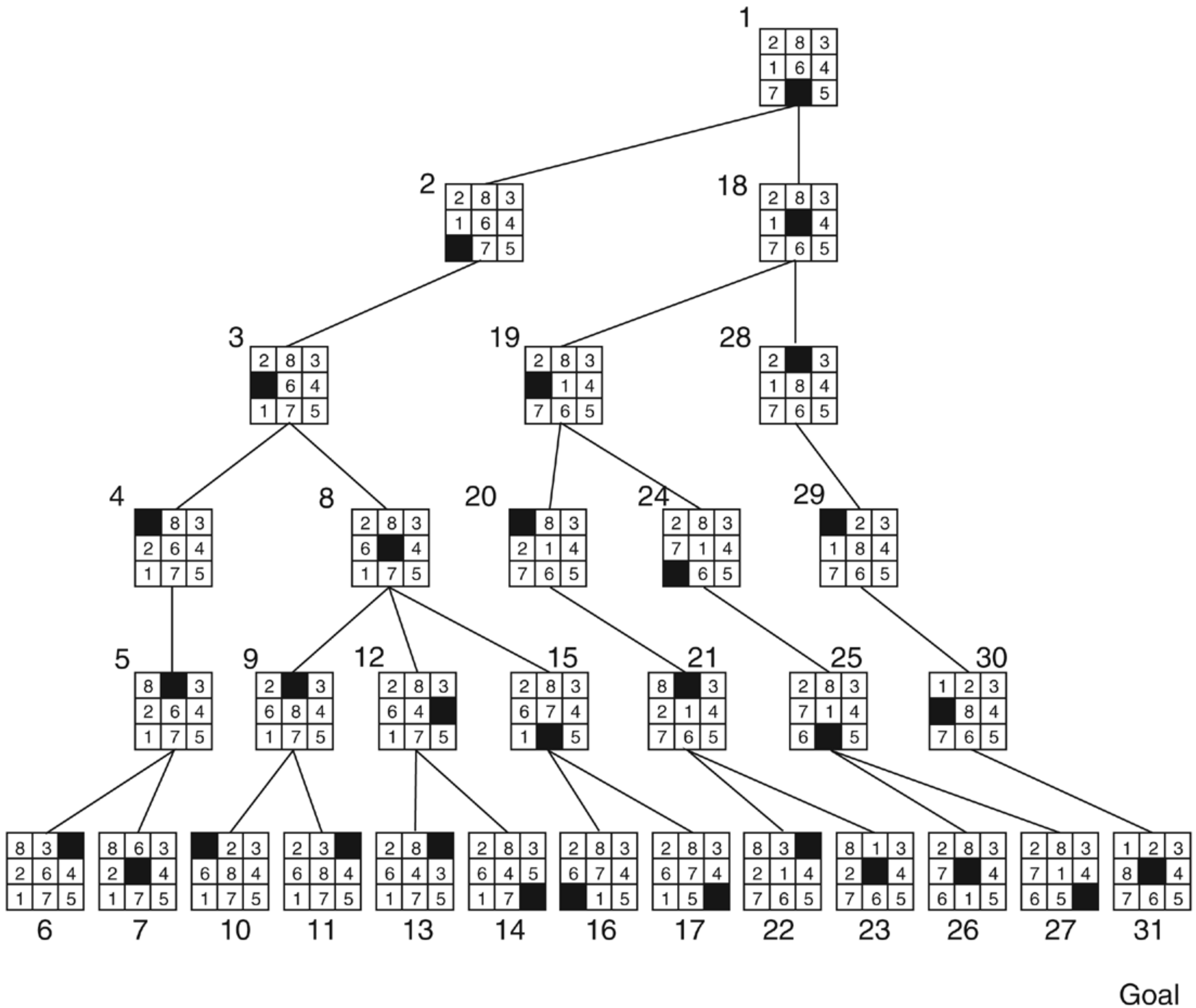


Figure 3.18: State space graph of a set of implications in the propositional calculus.

$q \to p$
$r \to p$
$v \to q$
$s \to r$
$t \to r$
$s \to u$
s
t

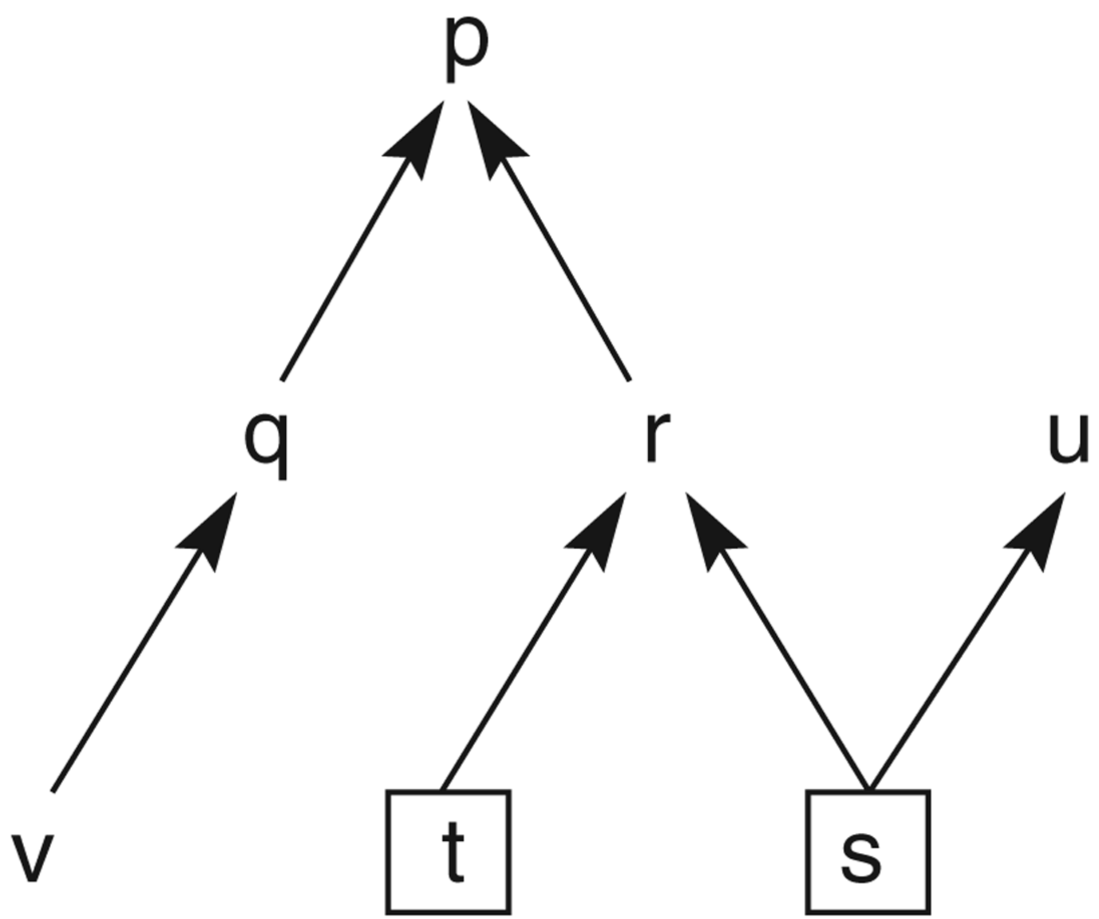
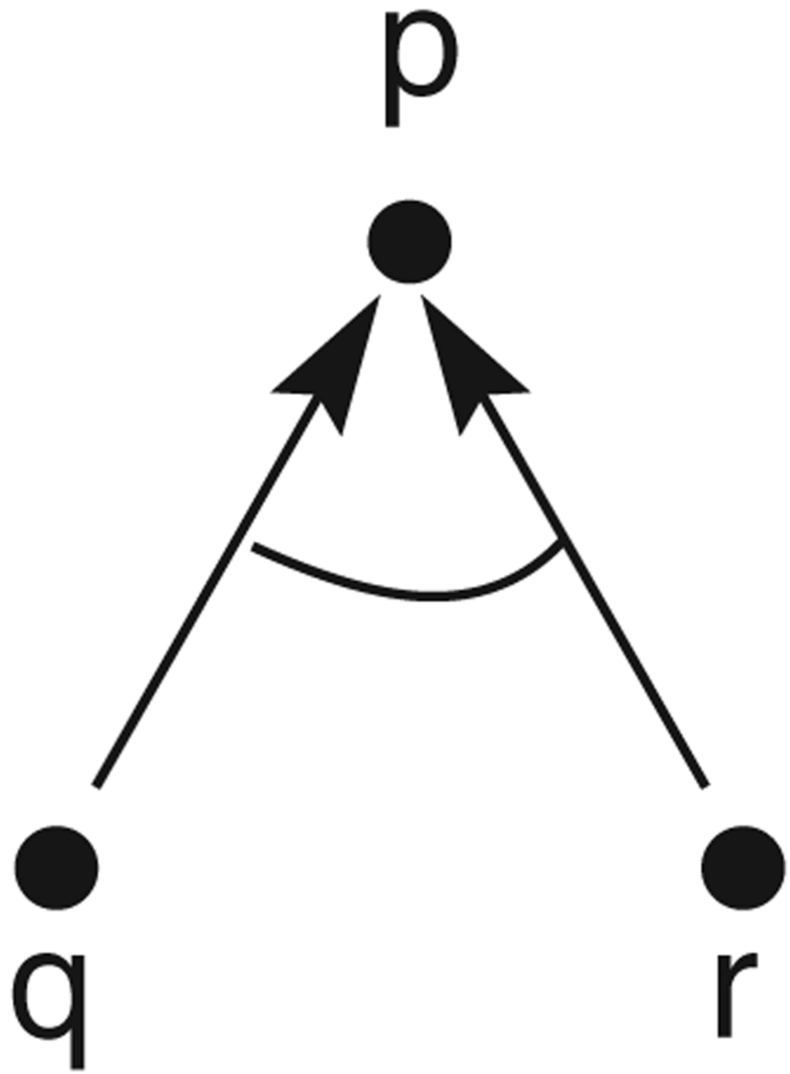


Figure 3.19: And/or graph of the expression q □ r ∅ p.



DEFINITION

HYPERGRAPH

A hypergraph consists of:

N, a set of nodes.

H, a set of hyperarcs defined by ordered pairs in which the first element of the pair is a single node from **N** and the second element is a subset of **N**.

An ordinary graph is a special case of hypergraph in which all the sets of descendant nodes have a cardinality of 1.

Figure 3.20: And/or graph of the expression q Δ r ∅ p.

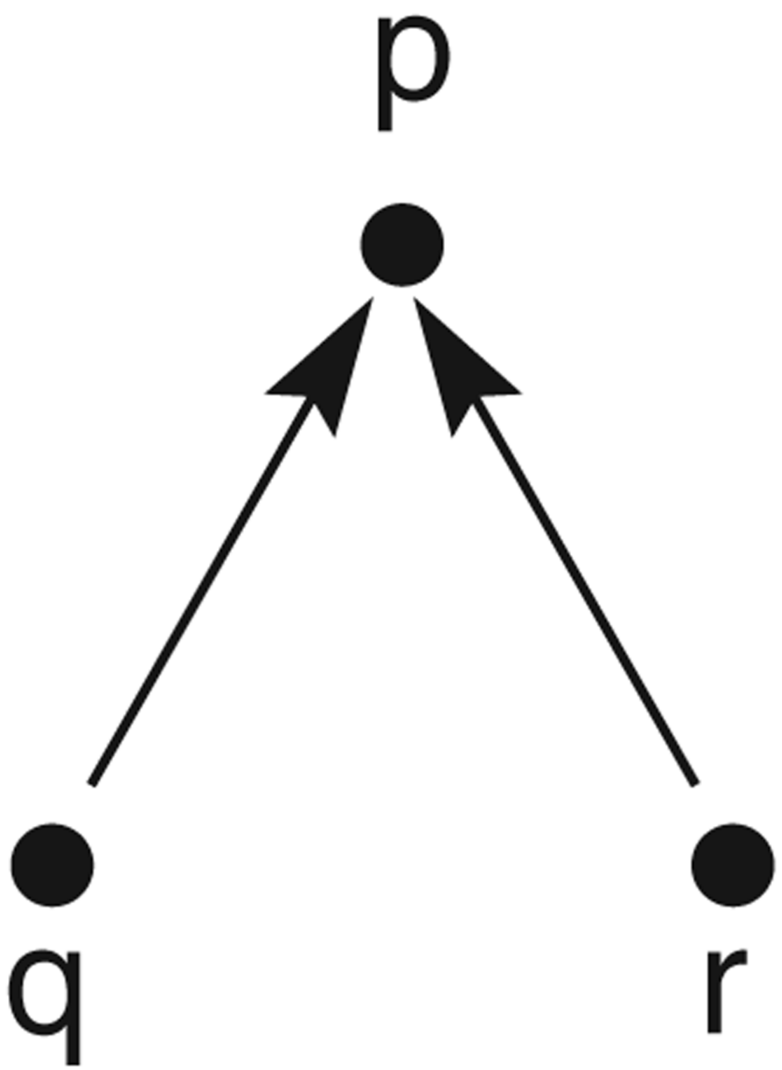


Figure 3.21: And/or graph of a set of propositional calculus expressions.

a

b

c

$a \wedge b \rightarrow d$

$a \wedge c \rightarrow e$

$b \wedge d \rightarrow f$

$f \rightarrow g$

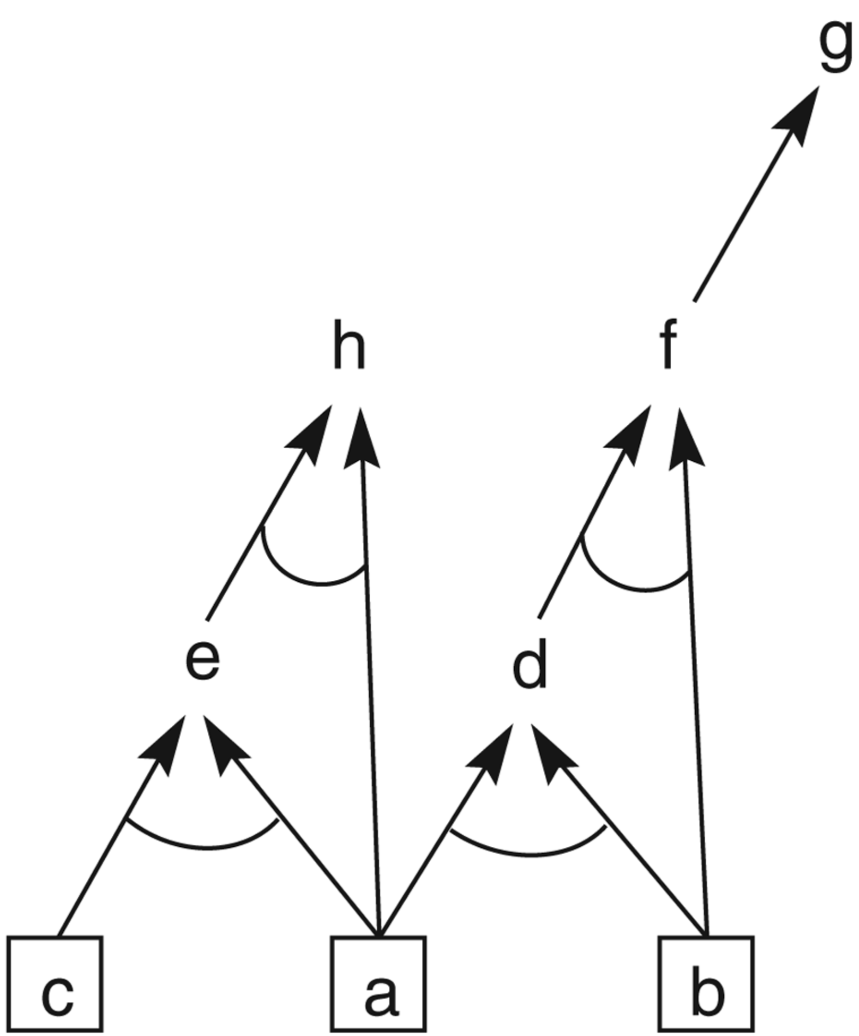
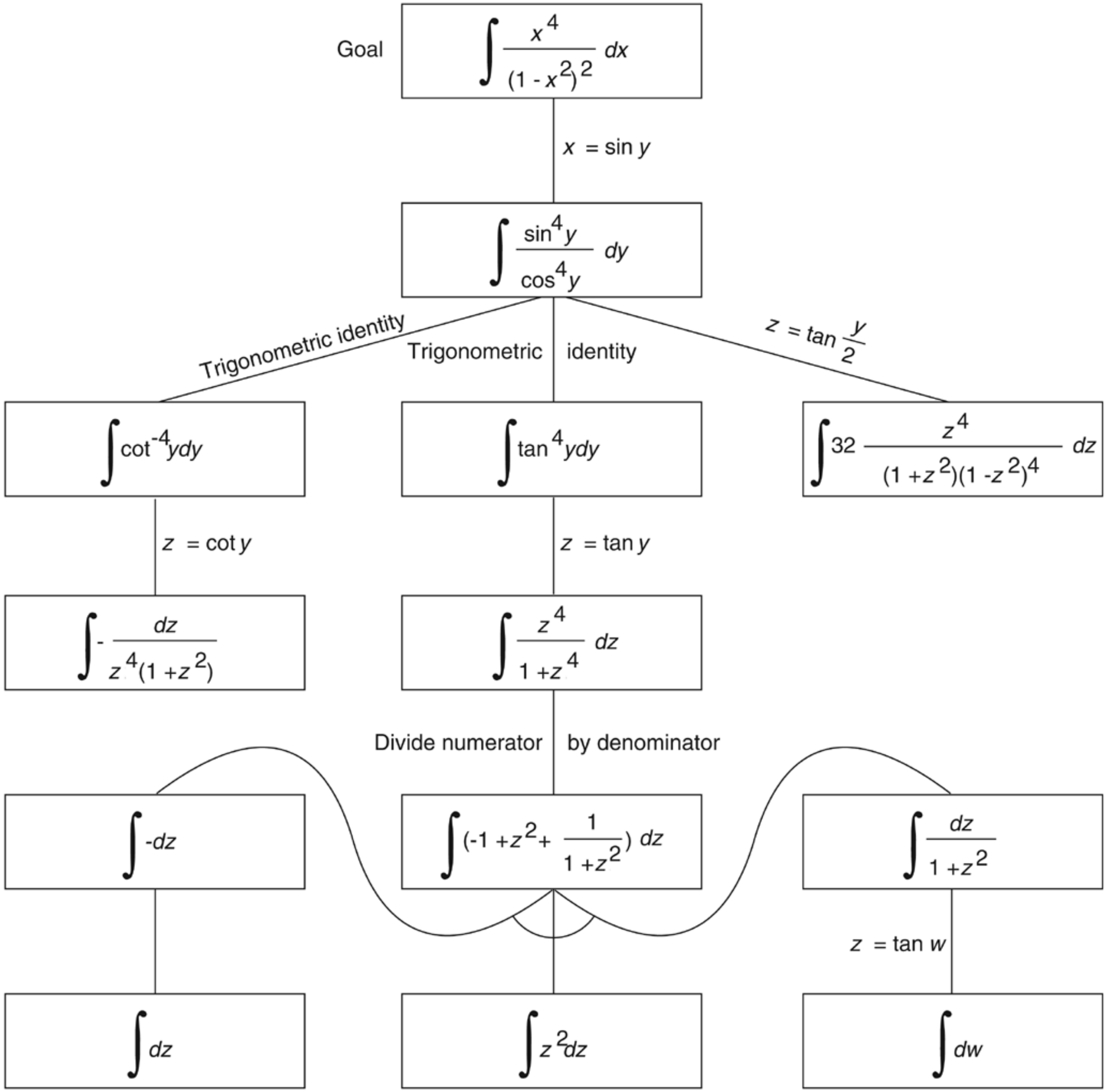


Figure 3.22: And/or graph of part of the state space for integrating a function, from Nilsson (1971).



The facts and rules of this example are given as English sentences followed by their predicate calculus equivalents:

1. Fred is a collie.
   collie(fred).

2. Sam is Fred's master.
   master(fred,sam).

3. The day is Saturday.
   day(saturday).

4. It is cold on Saturday.
   $\neg$ **(warm(saturday)).**

5. Fred is trained.
   trained(fred).

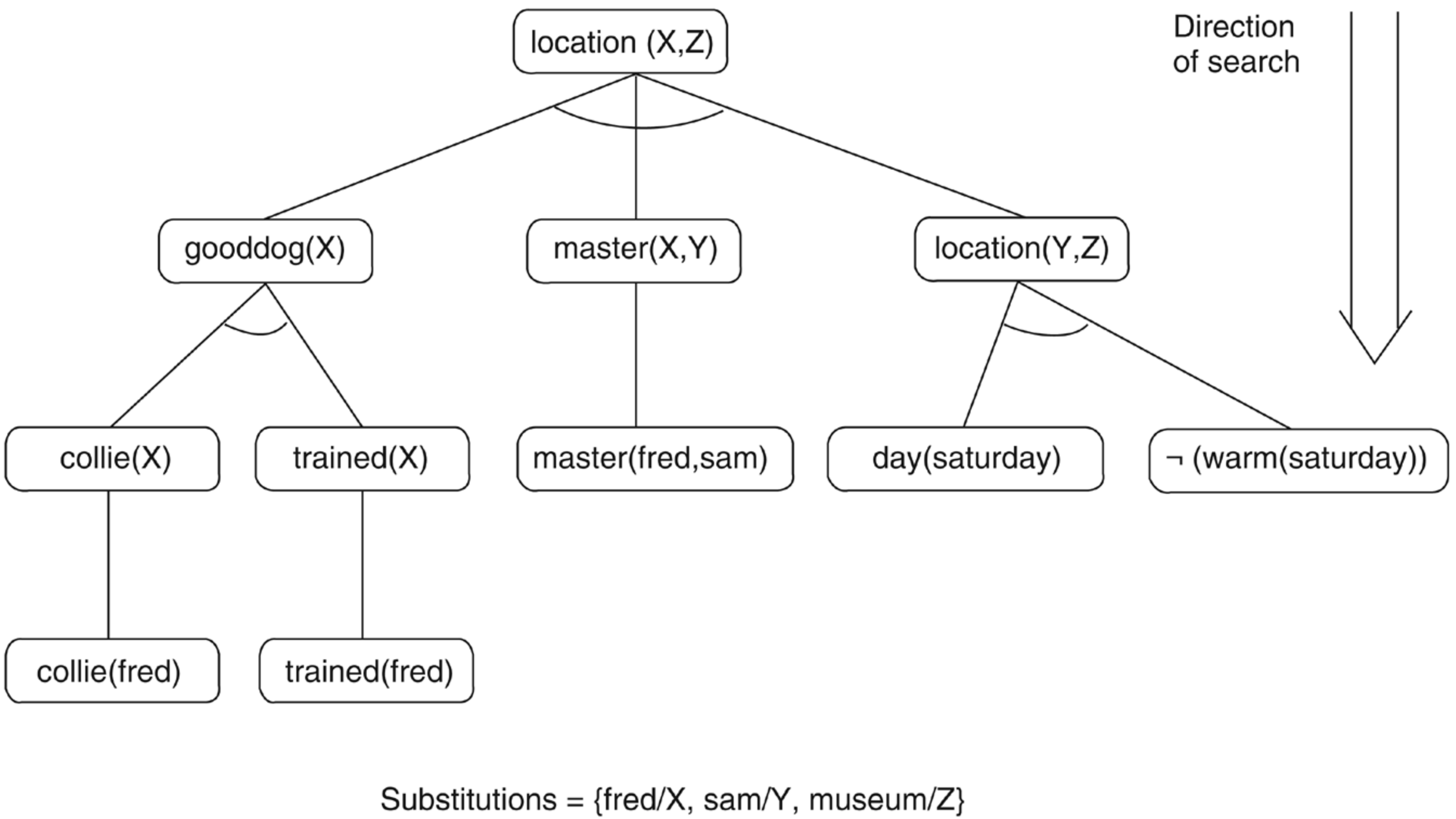
6. Spaniels are good dogs and so are trained collies.
   $\forall$ **X[spaniel(X)** $\vee$ **(collie(X)** $\wedge$ **trained(X))** $\rightarrow$ **gooddog(X)]**

7. If a dog is a good dog and has a master then he will be with his master.
   $\forall$ **(X,Y,Z) [gooddog(X)** $\wedge$ **master(X,Y)** $\wedge$ **location(Y,Z)** $\rightarrow$ **location(X,Z)]**

8. If it is Saturday and warm, then Sam is at the park.
   (**day(saturday)** $\wedge$ **warm(saturday))** $\rightarrow$ **location(sam,park).**

9. If it is Saturday and not warm, then Sam is at the museum.
   (**day(saturday)** $\wedge$ $\neg$ **(warm(saturday)))** $\rightarrow$ **location(sam,museum).**

Figure 3.23: The solution subgraph showing that fred is at the museum.



Five rules for a simple subset of English grammar are:

1. **A sentence is a noun phrase followed by a verb phrase.**
   sentence ↔ np vp
2. **A noun phrase is a noun.**
   np ↔ n
3. **A noun phrase is an article followed by a noun.**
   np ↔ art n
4. **A verb phrase is a verb.**
   vp ↔ v
5. **A verb phrase is a verb followed by a noun phrase.**
   vp ↔ v np
6. **art ↔ a**
7. **art ↔ the**
   (“a” and “the” are articles)
8. **n ↔ man**
9. **n ↔ dog**
   (“man” and “dog” are nouns)
10. **v ↔ likes**
11. **v ↔ bites**
    (“likes” and “bites” are verbs)

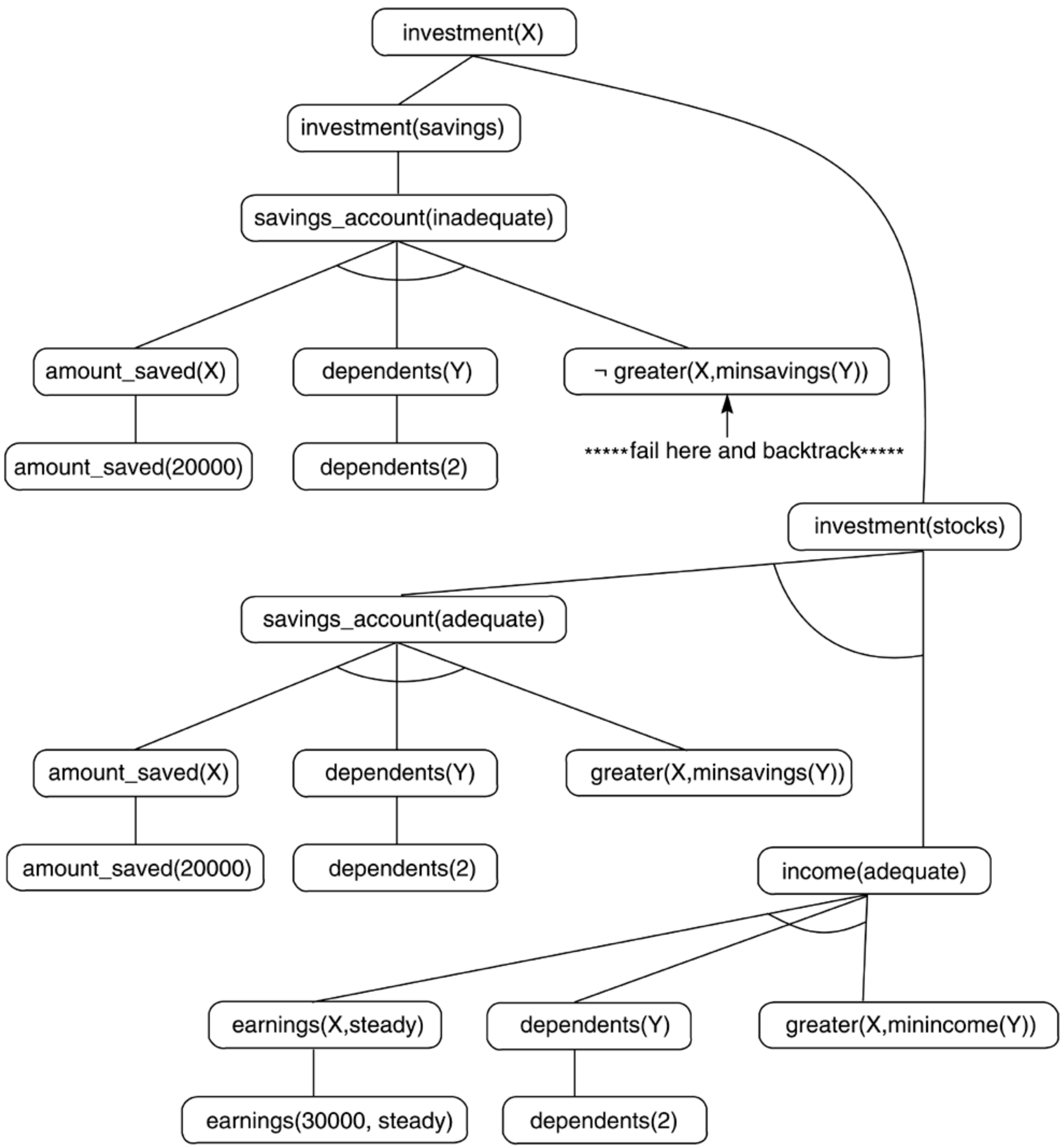
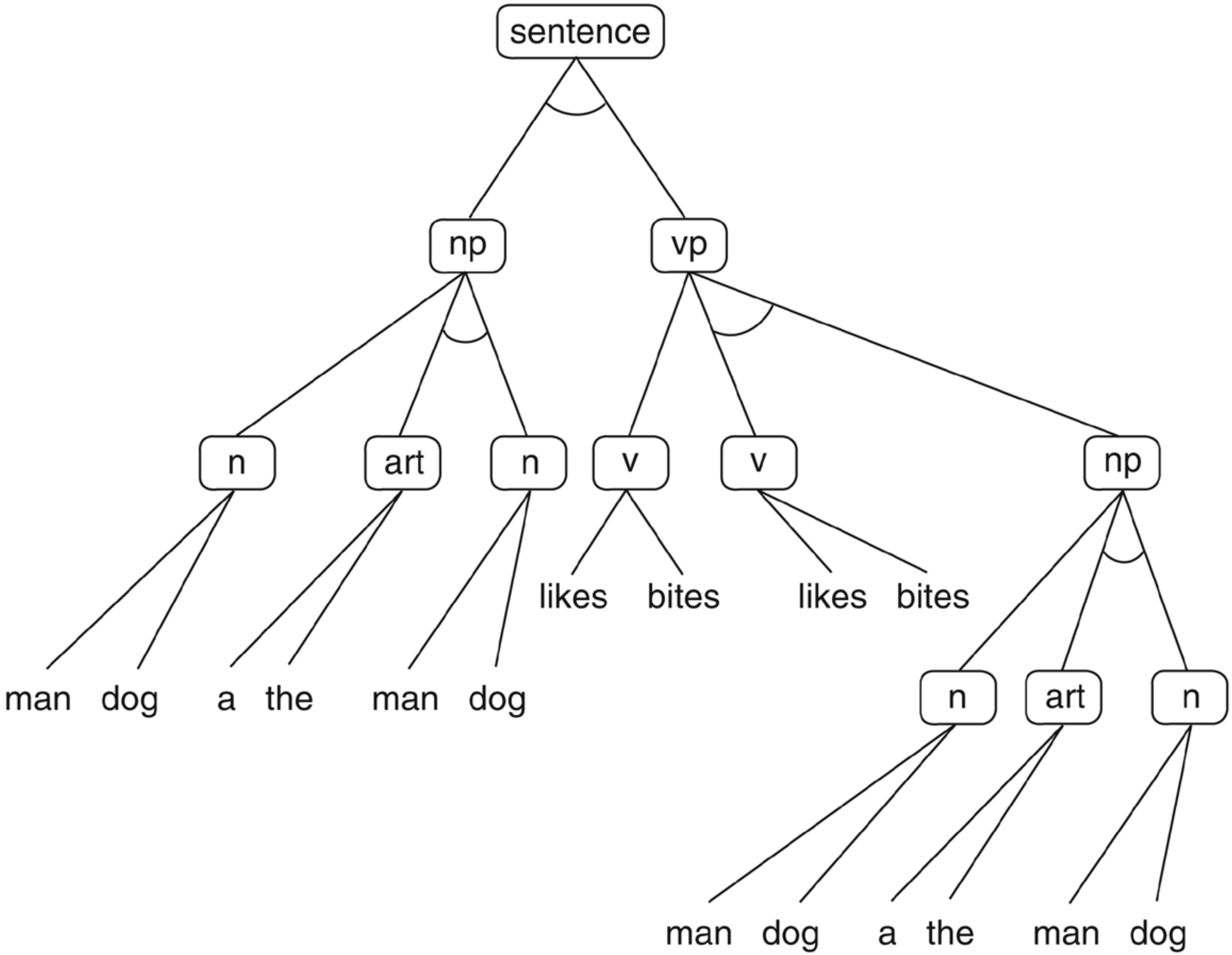


Figure 3.24: And/or graph searched by the financial advisor.

Figure 3.25: And/or graph for the grammar of Example 3.3.6. Some of the nodes (np, art, etc.) have been written more than once to simplify drawing the graph.



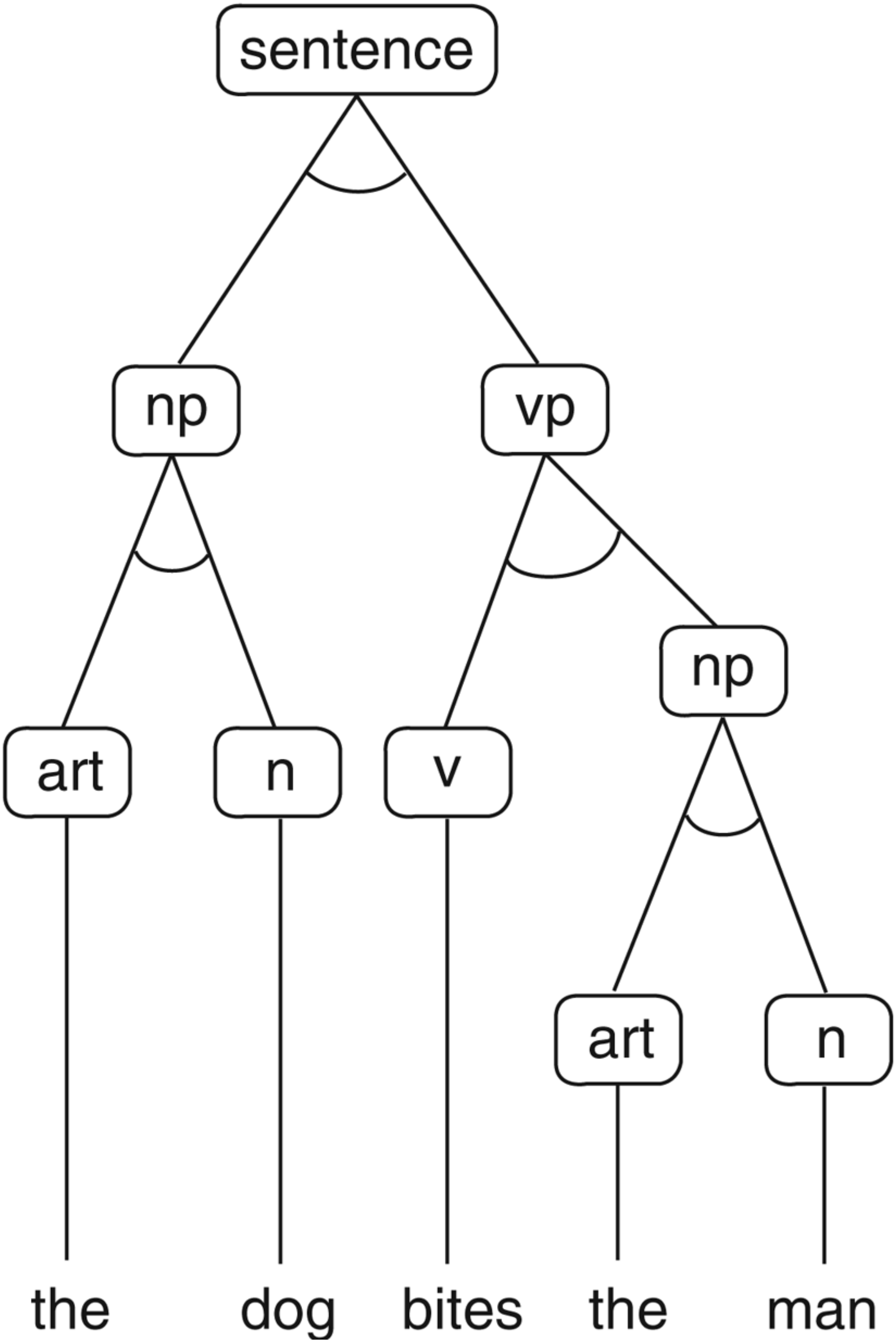


Figure 3.26: Parse tree for the sentence “The dog bites the man.” Note that this is a subtree of the graph of Figure 3.25.

Figure 3.27: A graph to be searched.