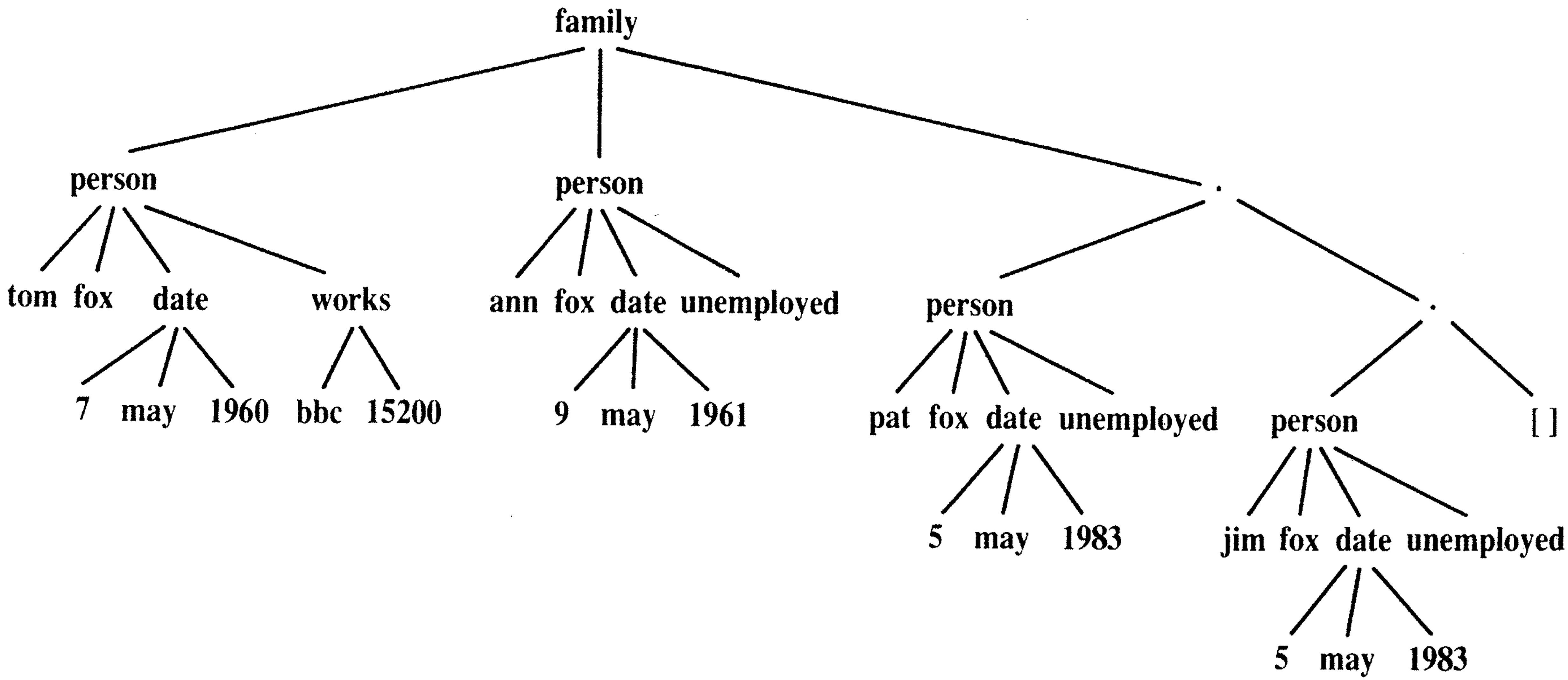

Figure 4.1 Structuring information about the family.

```
family(
  person( tom, fox, date(7,may,1960), works(bbc,15200) ),
  person( ann, fox, date(9,may,1961), unemployed),
  [ person( pat, fox, date(5,may,1983), unemployed),
    person( jim, fox, date(5,may,1983), unemployed) ] ).
```

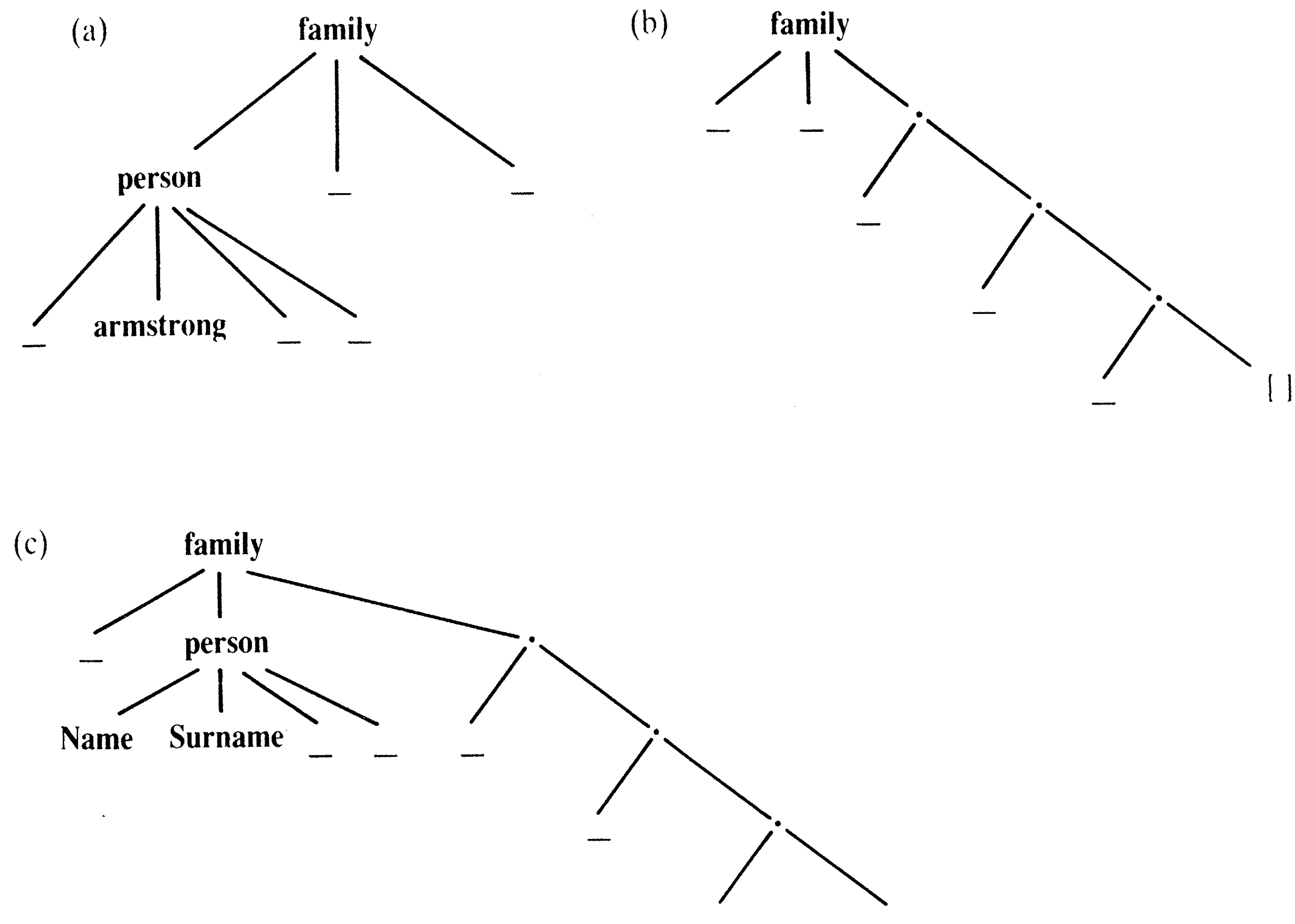

Figure 4.2 Specifying objects by their structural properties: (a) any Armstrong family; (b) any family with exactly three children; (c) any family with at least three children. Structure (c) makes provision for retrieving the wife's name through the instantiation of the variables **Name** and **Surname**.

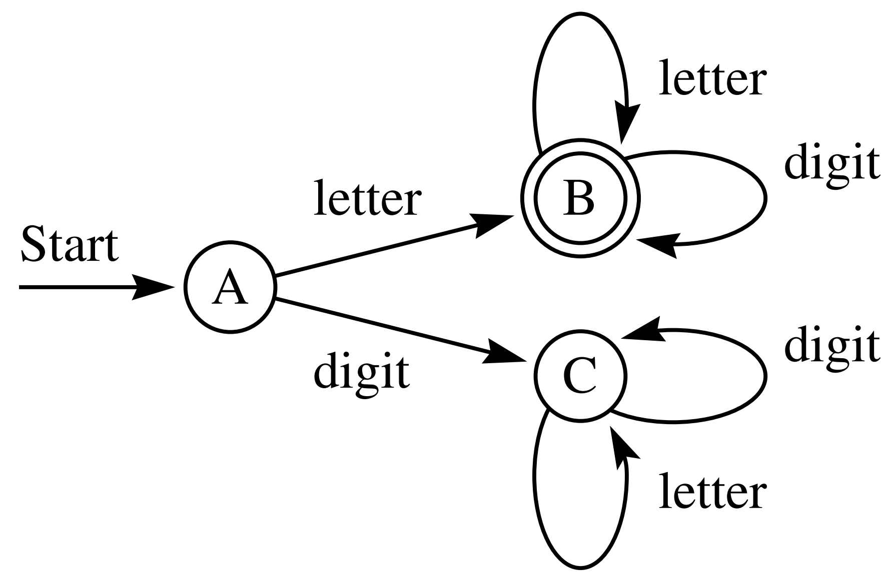

Figure 7.10

A finite state machine (FSM) to parse an identifier.

Example 7.4 To parse the string `cab3`, you would make the following transitions:

Current state: A	Input: `cab3`	Scan `c` and go to B.
Current state: B	Input: `ab3`	Scan `a` and go to B.
Current state: B	Input: `b3`	Scan `b` and go to B.
Current state: B	Input: `3`	Scan `3` and go to B.
Current state: B	Input:	Check for final state.

Because there is no more input and the last state is B, a final state, `cab3` is a valid identifier.

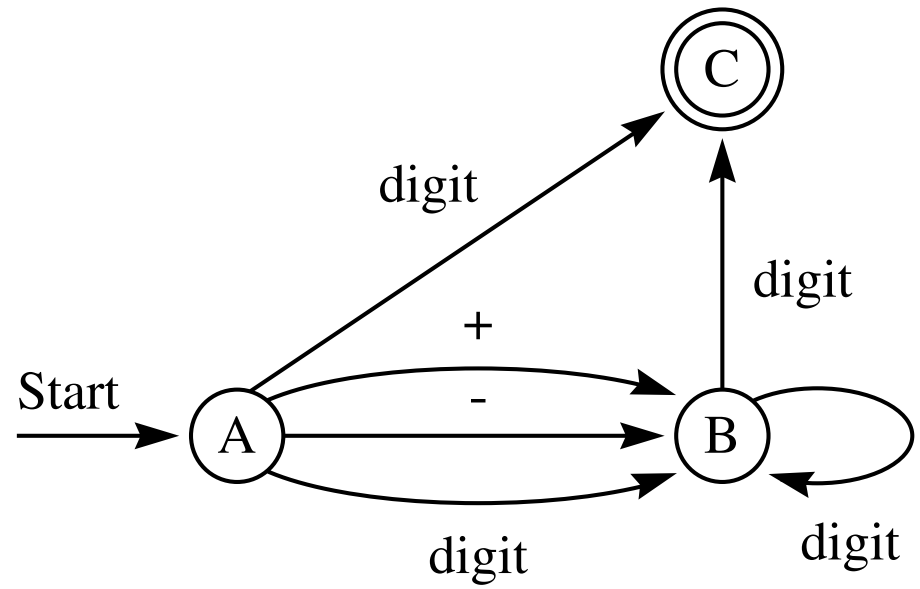

Figure 7.14

A nondeterministic FSM to parse a signed integer.

Example 7.5 You must make the following decisions to parse `+203` with this nondeterministic FSM:

Current state: A	Input: `+203`	Scan + and go to B.
Current state: B	Input: `203`	Scan `2` and go to B.
Current state: B	Input: `03`	Scan `0` and go to B.
Current state: B	Input: `3`	Scan `3` and go to C.
Current state: C	Input:	Check for final state.

Because there is no more input and you are in the final state C, you have proven that the input string `+203` is a valid signed integer.

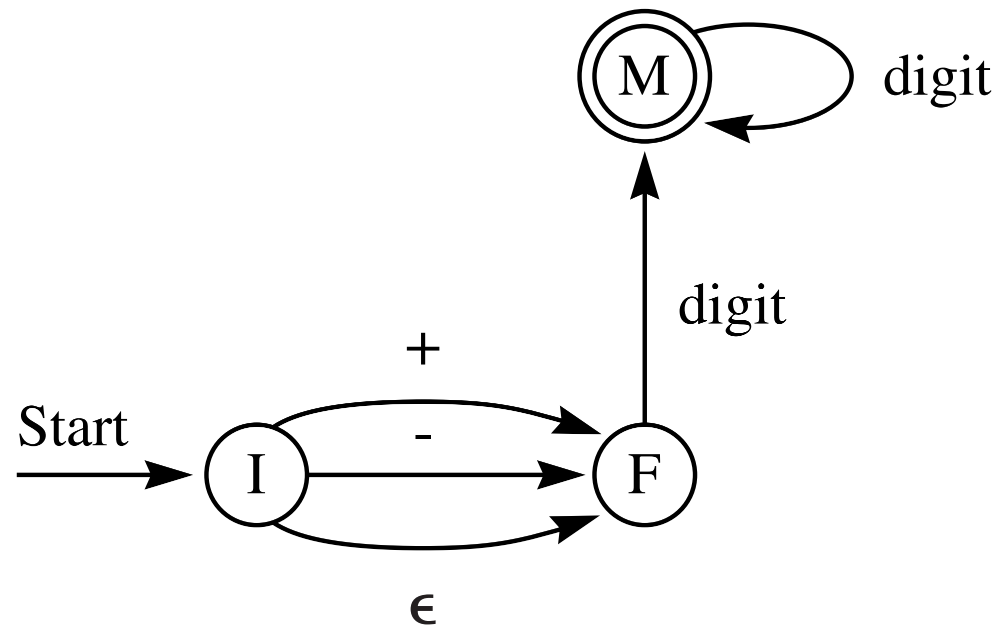

Figure **7.17**

An FSM with an empty transition to parse a signed integer.

Example 7.6 To parse 32 requires the following decisions:

Current state: I	Input: 32	Scan ϵ and go to F.
Current state: F	Input: 32	Scan 3 and go to M.
Current state: M	Input: 2	Scan 2 and go to M.
Current state: M	Input:	Check for final state.

The transition from I to F on ϵ does not consume an input character. When you are in state I, you can do one of three things: (a) scan + and go to F, (b) scan − and go to F, or (c) scan nothing (that is, the empty string) and go to F.

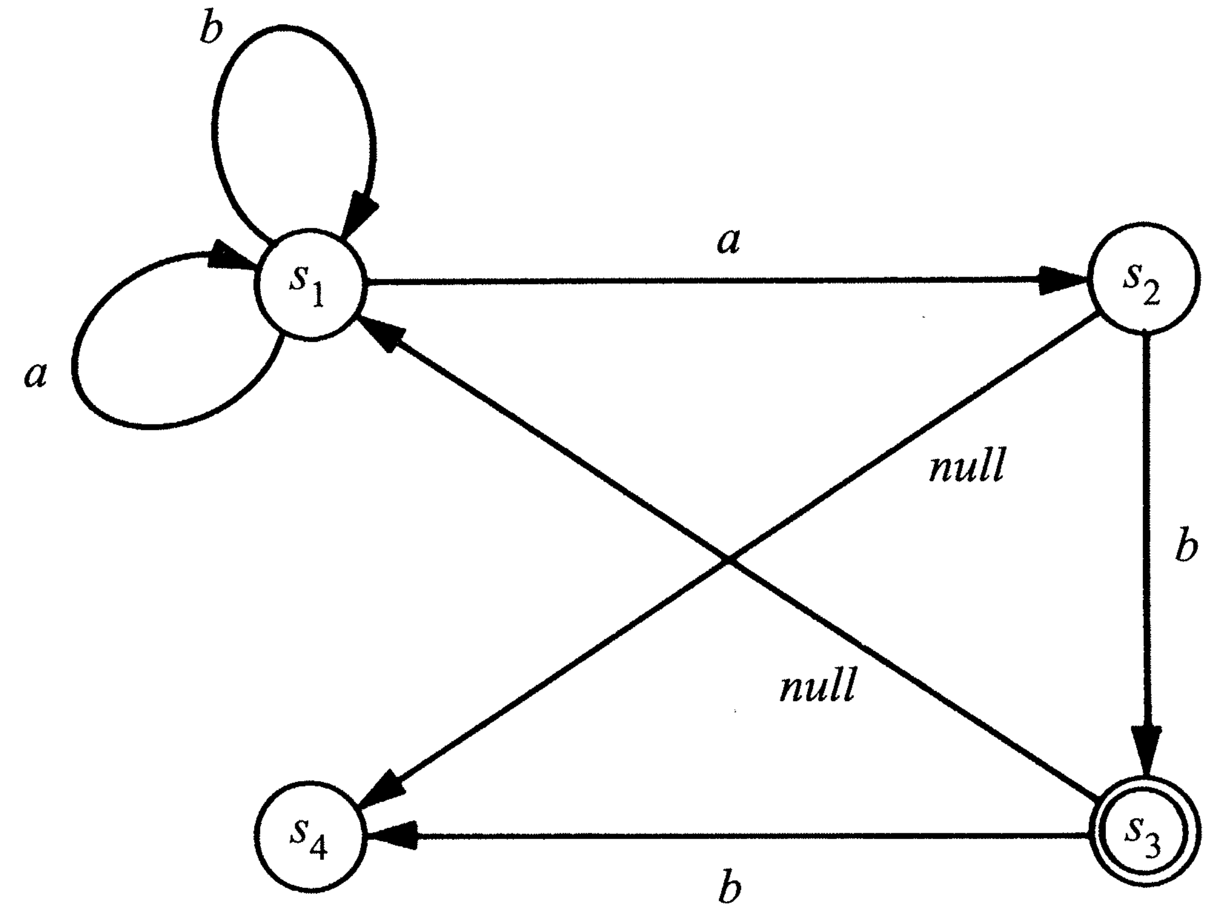

Figure 4.3 An example of a non-deterministic finite automaton.

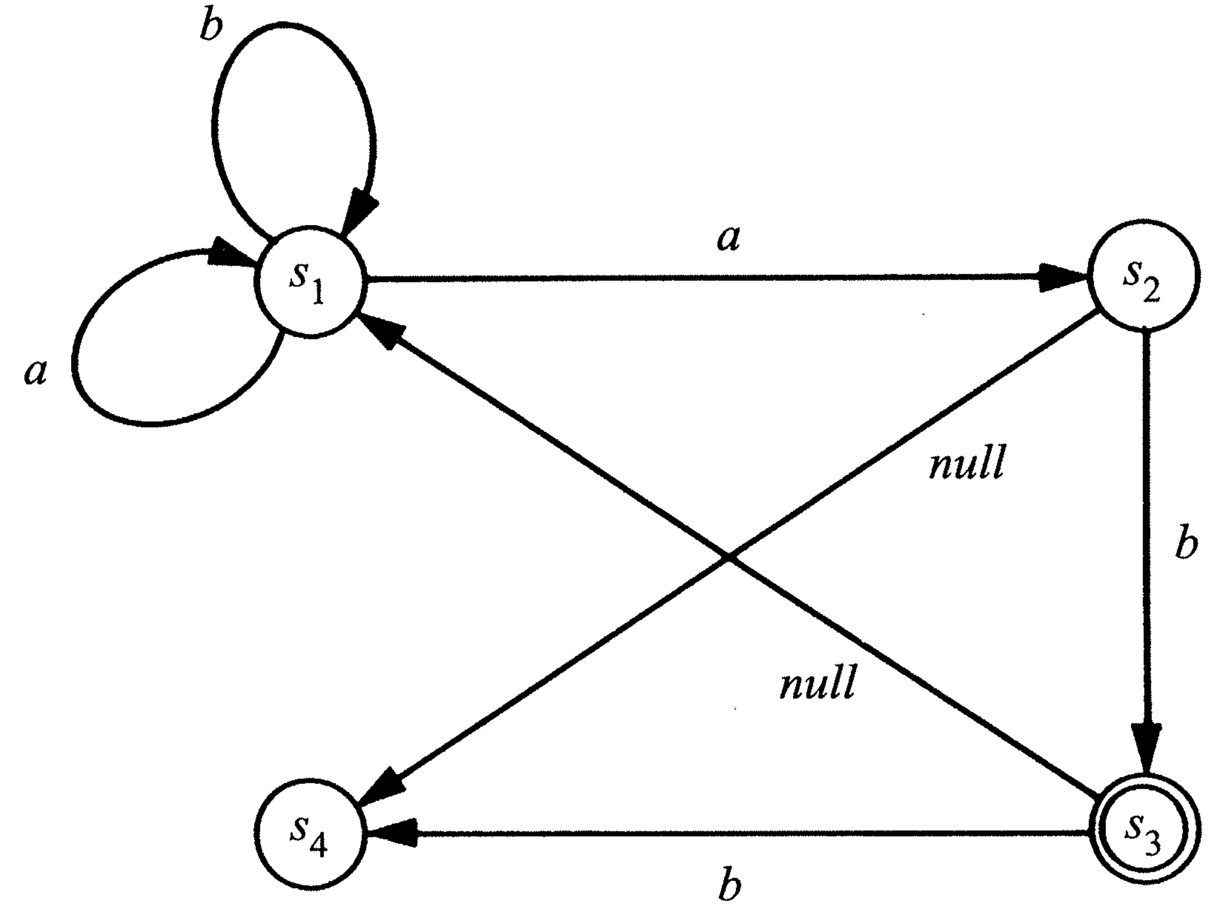

Figure 4.3 An example of a non-deterministic finite automaton.

```
final( s3).

trans( s1, a, s1).
trans( s1, a, s2).
trans( s1, b, s1).
trans( s2, b, s3).
trans( s3, b, s4).

silent( s2, s4).
silent( s3, s1).
```

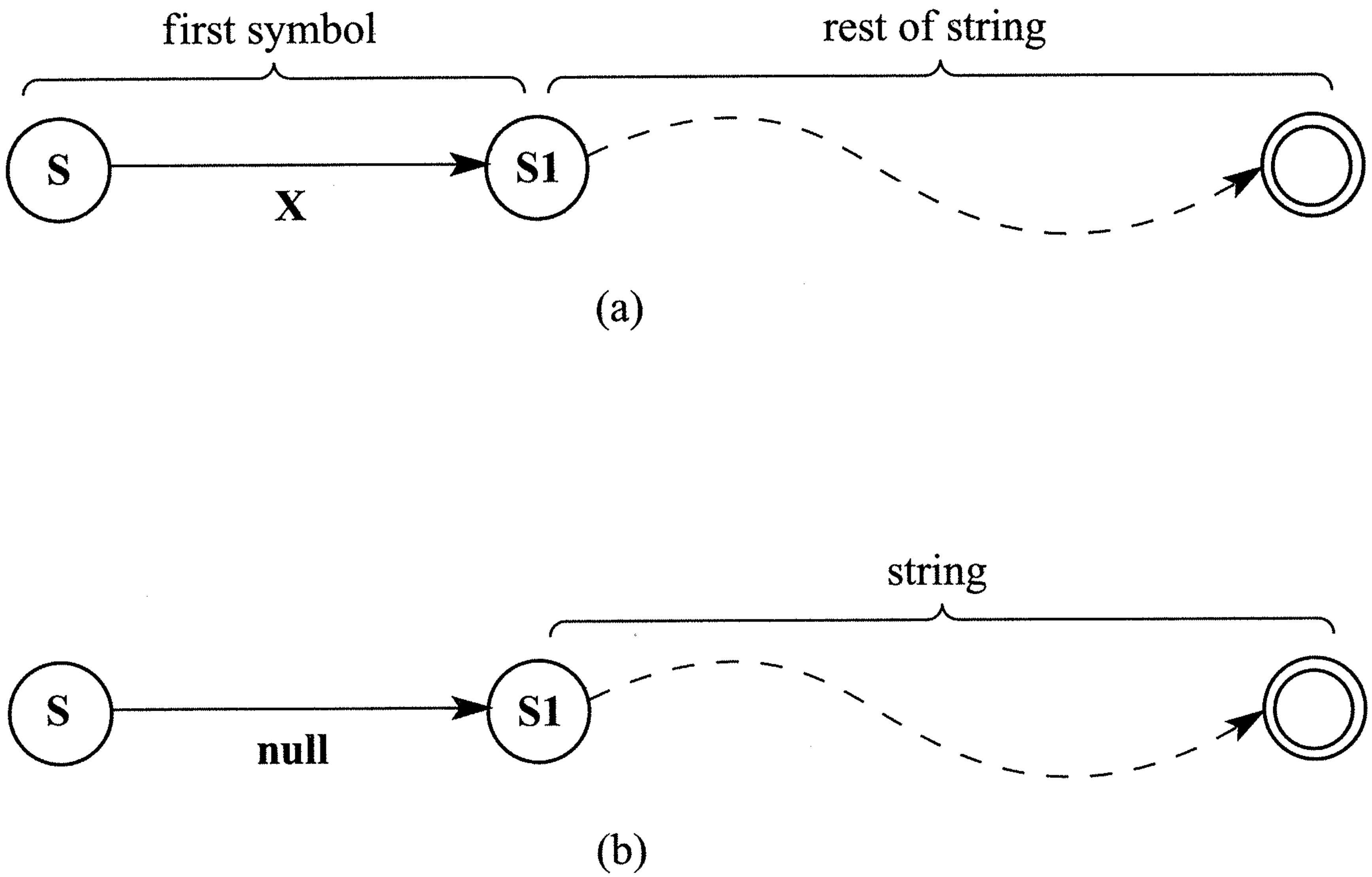

Figure 4.4 Accepting a string: (a) by reading its first symbol X; (b) by making a silent move.