```
(defmacro with-parses (node sent &body body)
  (with-gensyms (pos regs)
    `(progn
       (setq *sent* ,sent)
       (setq *paths* nil)
       (=bind (parse ,pos ,regs) (,node 0 '(nil))
         (if (= ,pos (length *sent*))
             (progn ,@body (fail))
             (fail))))))
```

Figure 23.5: Toplevel macro.

Before going on to look at a more representative ATN, let's look at a parsing generated from the tiny ATN defined earlier. The ATN compiler (Figure 23.3) generates code which calls types to determine the grammatical roles of a word, so first we have to give it some definition:

```
(defun types (w)
  (cdr (assoc w '((spot noun) (runs verb)))))
```

Now we just call with-parses with the name of the start node as the first argument:

```
> (with-parses s '(spot runs)
    (format t "Parsing: ~A~%" parse))
Parsing: (SENTENCE (SUBJECT SPOT) (VERB RUNS))
@
```

23.5 A Sample ATN

Now that the whole ATN compiler has been described, we can go on to try out some parses using a sample network. In order to make an ATN parser handle a richer variety of sentences, you make the ATNs themselves more complicated, not the ATN compiler. The compiler presented here is a toy mainly in the sense that it's slow, not in the sense of having limited power.

The power (as distinct from speed) of a parser is in the grammar, and here limited space really will force us to use a toy version. Figures 23.8 through 23.11 define the ATN (or set of ATNs) represented in Figure 23.6. This network is just big enough to yield several parsings for the classic parser fodder "Time flies like an arrow."

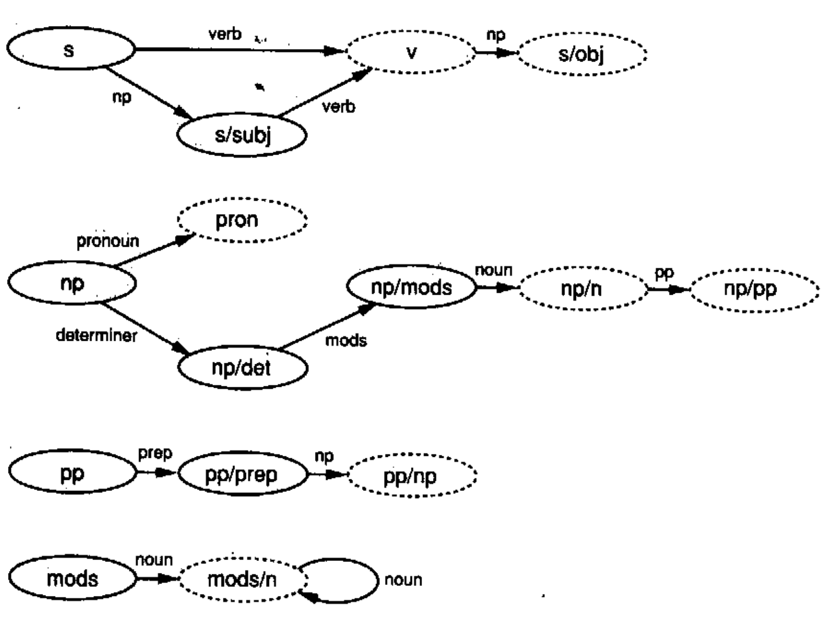

Figure 23.6: Graph of a larger ATN.

```
(defun types (word)
  (case word
    ((do does did) '(aux v))
    ((time times) '(n v))
    ((fly flies) '(n v))
    ((like) '(v prep))
    ((liked likes) '(v))
    ((a an the) '(det))
    ((arrow arrows) '(n))
    ((i you he she him her it) '(pron))))
```

Figure 23.7: Nominal dictionary.

We need a slightly larger dictionary to parse more complex input. The function types (Figure 23.7) provides a dictionary of the most primitive sort. It defines a 22-word vocabulary, and associates each word with a list of one or more simple grammatical roles.