Syntactic Predicates: Selective Backtracking

1. Definition
2. Difference between semantic / syntactic predicates
3. Examples

1. Definition

- Syntactic predicates are grammar fragments that describe a syntactic context that must be satisfied before application of an associated production is authorized

- Syntactic predicates are a form of selective backtracking that allow the recognition of constructs beyond the capabilities of conventional parsing.

- Form: `rule :` (α)? β | γ ;

- Shorthand: `rule :` (α)? | γ ;

2. Difference between sem / syn preds

- Semantic predicates alter the parse with semantic information such as symbol table info.

- Syntactic predicates use purely syntactic information; i.e., the next n symbols of lookahead.

- Use a semantic predicate to disambiguate things that are identical syntactically, but differ according to context; e.g., function call versus array reference in FORTRAN (`VAL(42,I)`).

- Use a syntactic predicate for language constructs that do not look the same, but cannot be resolved with the normal $LL(k)$ parsing mechanism.

- Exponentially slow in worst-case: use sparingly.

- Can be used to avoid ANTLR analysis delays; shuts off full $LL(k)$ lookahead computation.

3. Examples

Using shorthand:

```
stat : ( list "=" list )?
     | list
     ;
list : ID ( "," ID )* ;
```

Or, more efficiently:

```
stat : ( list "=" )? list "=" list
     | list
     ;
list : ID ( "," ID )* ;
```

C Example

```
decl    :   ( type declarator block )?  // function
        |   ( type declarator ";" )?    // var decl
        |   "struct" ...                // struct def
        ;
type    :   built_in_type
        |   "struct" ...
        ;
block   :   "\" ... "\"
        ;
```

Ellis and Stroustrup on C++:

"There is an ambiguity in the grammar involving *expression-statements* and *declarations*... The general cases cannot be resolved without backtracking... In particular, the lookahead needed to disambiguate this case is not limited."

`T(*a)->m=7;` // *expression-statement; type cast to T*

`T(*a)(int);` // *pointer to function declaration*

Ellis and Stroustrup's Solution:

"In a parser with backtracking the disambiguating rule can be stated very simply:

1. If it looks like a *declaration*, it is; otherwise
2. if it looks like an *expression*, it is; otherwise
3. it is a syntax error."

ANTLR solution using syntactic predicates:

```
stat:   (declaration)? declaration
    |   expression
    ;
```