Grammar Description Format[1]

William Prahl, February 13, 2014

Abstract

We describe the syntax and formal semantics of Grammar Description Format, a scheme for describing the grammars of human languages for use in a natural language processing (e.g. parsing) or generation (e.g. summarization) system.

1 Uses in Sentences

This language description scheme treats links between words as the basic unit of grammar. A very simple example:

```
FOO as foo expect bar .
BAR as bar .
```

By convention, uses have identifiers in capital letters. Each use is defined by how it modifies the listener's expectations. In the example, the use of a Foo causes the listener to expect a `bar`, and a Bar can be used to satisfy that expectation. The "`:`" operator provides shorthand for the "as" syntax, for example: "`(BAR as foobar)`" is equivalent to "`BAR:foobar`.

Multiple such declarations can be given for any one use, and all will be combined in the final definition. For example:

```
FOO as foo expect bar .
BAR as bar ;
    as bar expect baz .
BAZ as baz .
```

In this example, Foo Bar and Foo Bar Baz are both acceptable patterns of use.

2 Agreement

Consider the (not uncommon) case where a language forces the agreement of its adjectives with its nouns in gender and number. This could be expressed with a large number of particular use cases, such as in the following description:

```
JJ  as jj_m_sg expect NN:nn_m_sg ;
    as jj_f_sg expect NN:nn_f_sg ;
    as jj_m_pl expect NN:nn_m_pl ;
    as jj_f_pl expect NN:nn_f_pl .

NN  as nn_m_sg ;
    as nn_m_pl ;
    as nn_f_sg ;
    as nn_f_pl .
```

However, enumerating each individual case of agreement is a painful, error-prone process that does not make explicit the larger patterns of agreement. Therefore Grammar Description Format provides the concept of a syntactic category, indicated as NN.M.SG or JJ.F.PL. The above description can be rewritten:

```
JJ expect nn .
NN as nn .

JJ  agree nn on M , F ;
    agree nn on SG, PL .
```

Syntactic categories can be used explicitly as part of use-identifiers. They may also be left unpsecified, in which case the statement is understood to apply to the use in all of its categories.

The only constraint on syntactic categories is that any two syntactic categories in the same "`agree . . . on`" clause must contrast; that is, in the above example, an NN.M.F must be impossible.

```
JJ  as attr expect attributed .

JJ  agree attributed on M, F ;
    agree attributed on SG, PL ;
    agree attributed on NOM, OBL .

NN.NOM  as subj expect pred ;
        as subj, attributed expect pred ;
NN.OBL  as obj ;
        as attributed, obj ;

VB as pred expect obj .
JJ.NOM as pred .

NN agree pred on SG, PL .
NN agree JJ:pred on M, F .
```

In this example, simple subject-verb-object and subject-adjective sentences can be formed, where a noun (either a subject or an object) can be modified by an optional attributive adjective. Subjects must be in the nominative case, and objects in the oblique.

An attributive adjective must have the same gender, number, and case as the noun that it modifies. A verb or predicate adjective must have the same number as its subject noun, and a predicate adjective must also have the same gender, and be in the nominative case.

3 Patterns

As of yet, our example languages have had no words. Grammar Description Format provides patterns as a way to let words take on uses. Patterns—though they can be regular or irregular—are the only way that a word can be given a use. Returning to the example language of section 1, here are some example patterns:

```
foo_pat regular ?s where
    FOO is ?s .
barbaz_pat regular ?s-n where
    BAR is ?s-n &
    BAZ is ?s-a .
irr_pat closed ?s where
    BAR, BAZ is ?s .
```

We have defined three different patterns, one for Foo, and two covering both Bar and Baz. Both `foo_pat` and `barbaz_pat` are regular patterns, that is, they are productive, while *irr_pat* applies only to a closed class of words.

In a pattern definition, a `?variable` captures any non-null sequence of segments, and can be reused in the inner patterns. The '-' operator is used to concatenate variables and sequences of literal segments.

```
fa, hey, la, ton follow foo_pat .
ban, gin, ton follow barbaz_pat .
fizz follow irr_pat .
```

From these uses, we can create some example sentences and non-sentences in this language:

1. *fa ban*
2. **ban fa*
3. **fa fa*
4. *fa fizz fizz*
5. *fa ban gia*

6. **fa gia ban*

7. *ton ton toa*

The uses of any particular dictionary word are defined by its patterns. One can expand the example lexicon into all of its derived uses with the following table:

Word	`?s`	`foo`	`bar`	`baz`
ban	ba		*ban*	*baa*
fa	fa	*fa*		
fizz	fizz		*fizz*	*fizz*
gin	gi		*gin*	*gia*
hey	hey	*hey*		
la	la	*la*		
ton	to	*ton*	*ton*	*toa*

Patterns may also make use of syntactic categories. In fact, since patterns *cannot* reference the different roles that a particular use can have, this is the preferred way to provide conjugation and declension. As an example:

```
1st_declension_masculine regular ?s where
    NN.M.SG.NOM is ?s &
    NN.M.PL.NOM is ?s-as &
    NN.M.SG.OBL is ?s-u &
    NN.M.SG.OBL is ?s-us .
```

4 Segment Alternation

Although it is theoretically possible to capture all the possible patterns of segmental alternation and gradation using a large number of patterns involving literal segment patterns, this would not capture the systematic nature of many of these phenomena. Therefore the Grammar Description Format provides a way to define segment alternations.

Suppose we want to define a scheme where unvoiced stops are sometimes weakened to fricatives. We could set up an alternation like:

```
{T} alternates 2 grades
    p   f ;
    t   s ;
    k   x .
```

Then we can use this alternation to define patterns. For the following example, assume that the use NN has the syntactic categories NOM and OBL.

```
nn_pat regular ?s where
    NN.NOM is ?s &
    NN.OBL is ?s-e .
nn_alt regular ?s.T#1 where
    NN.NOM is ?s-$1#1 &
    NN.OBL is ?s-$1#2-e .

cas, gad follow nn_pat .
fat, pop follow nn_alt .
```

The # operator specifies which grade of an alternating literal is desired. The $ operator takes a number n, and refers to the row of the alternating segment matched by the nth alternating segment in the pattern. Its meaning for alternations is similar to the meaning of the ? operator for variables.

The example lexicon will therefore be found in the following declined forms.:

Word	Nn.Nom	Nn.Obl
cas	*cas*	*case*
fat	*fat*	*fase*
gad	*gad*	*gade*
pop	*pop*	*pofe*

The above patterns are correct, but a more elegant solution is possible. We can expand the alternation to cover even the terminal consonants that don't have gradation—they just need to be the same in both grades—so that a single pattern covers the lexicon. In the following example, we have included only the consonants that we see in the terminal position, for brevity.

```
{T} alternates 2 grades
    p   f ;
    t   s ;
    k   x ;
    s   s ;
    d   d .

nn_pat regular ?s-T#1 where
    NN_NOM as ?s-$1#1 ;
    NN_OBL as ?s-$1#2-e .

cas, fat, gad, pop follow nn_pat .
```